Pangenome-Informed Language Models for Synthetic Genome Sequence Generation

Pengzhi Huang
Cornell University
ph453@cornell.edu

François Charton
FAIR, Meta
charton@meta.com

Jan-Niklas M. Schmelzle
UTHSC
Cornell University
schmelzle@uthsc.edu

Shelby S. Darnell
UTHSC
sdarnell@uthsc.edu

Pjotr Prins
UTHSC
pjotr@prins.net

Erik Garrison
UTHSC
egarrison@uthsc.edu

G. Edward Suh
NVIDIA, Cornell University
gs272@cornell.edu

Abstract

Language Models (LM) have been extensively utilized for learning DNA sequence patterns and generating synthetic sequences. In this paper, we present a novel approach for the generation of synthetic DNA data using pangenomes in combination with LM. We introduce three innovative pangenome-based tokenization schemes, including two that can decouple from private data, while enhance long DNA sequence generation. Our experimental results demonstrate the superiority of pangenome-based tokenization over classical methods in generating high-utility synthetic DNA sequences, highlighting a promising direction for the public sharing of genomic datasets.

1 Introduction

Public availability of genome datasets, such as the Human Genome Project (HGP) (Lander et al., 2001), the 1000 Genomes Project (Consortium et al., 2012), The Cancer Genome Atlas (TCGA) (Weinstein et al., 2013), GenBank (Benson et al., 2012), the International HapMap Project (Gibbs et al., 2003), the Human Pangenome Project (Liao et al., 2023), and the Telomere-to-Telomere project (Nurk et al., 2022), has been instrumental in advancing genomics research. However, large-scale genome sequencing remains costly and resource intensive due to the sophisticated equipment and computational resources required (Wetterstrand, 2021; Van Dijk et al., 2018). Additionally, the release of real genomic data raises significant privacy concerns, as re-identification risks persist despite anonymization efforts (Sweeney et al., 2013; Wjst, 2010; Ohm, 2009).

Synthetic data generation offers a scalable and relatively private alternative, enabling researchers to perform analyzes without exposing sensitive information (Yelmen et al., 2021). Specific tasks such as De Novo genome assembly (Tran et al., 2017, 2019; Yang et al., 2019) and genotype imputation (Browning and Browning, 2016) inherently involve the generation of unknown sequences, making them also suitable applications for synthetic data. A good generative model can significantly improve their accuracy and efficiency by predicting missing or incomplete segments.

Deep learning models are widely used in different tasks, even in processing genome sequences and related data (Yun et al., 2020; Kolesnikov et al., 2021; Kim and Kim, 2018; Elbashir et al., 2019). While generative adversarial networks (GANs) have been explored for synthetic genome generation, their output is limited to short sequences (Bae et al., 2019; Gupta and Zou, 2018). LMs have shown their capability to generate synthetic natural languages that are almost indistinguishable from real data. The generated language text can be used to train other models (Kumar et al., 2020; Yoo et al., 2021; Hartvigsen et al., 2022), including those in the medical domain (Peng et al., 2023b; Guevara et al., 2024). Proven to be extraordinarily good at processing human language, LMs can also interpret and generate broader text, such as code for programming tasks (Chen et al., 2021), thereby pushing the boundaries of their application beyond strictly spoken language-based domains.

While LMs present a promising alternative for generating long synthetic DNA sequences, effective tokenization of DNA sequences is crucial for leveraging LMs. Traditional methods, such as single nucleotide tokenization and k-mer tokenization, segment sequences into individual nucleotides or substrings of length k (Lanchantin et al., 2017; Bae et al., 2019; Yelmen et al., 2021; Peng et al., 2023b; Alipanahi et al., 2015; An et al., 2022; Fishman et al., 2023). Classical approaches like k-mer tokenization (GKMT) are particularly sensitive to small mutations such as insertions or deletions: a single-base shift can disrupt all downstream tokens,

severely affecting model stability and learning. Additionally, its divergence from natural language processing (NLP) segmentation approaches limits the model's ability to capture DNA sequence patterns. Byte Pair Encoding (BPE) is used in recent work (Zhou et al., 2023), but still requires segmenting long DNA sequences into shorter chunks due to computational and memory constraints during tokenizer training. This study explores how NLP and pangenome-inspired tokenization can enhance LMs' ability to learn DNA sequence structures.

To build a practical genome sequence generation model that protects the privacy of the dataset, we propose LM-based synthetic data generation using two novel pangenome graph (see §2.2)-based tokenization schemes: Pangenome-based Node Tokenization (PNT) and Pangenome-based k-mer Tokenization (PKMT). PNT leverages nodes in the graph as tokens, while PKMT segments sequences using graph nodes before generating k-mers, enabling future applications of privacy techniques such as differential privacy (DP).

This work presents the first comparative analysis of classical and pangenome-based tokenization schemes for LMs, specifically GPT-2 and Llama, in learning DNA sequence patterns and generating long synthetic sequences. Our findings reveal that the pangenome graph structure embeds significant information that enhances neural networks' comprehension of DNA sequences. Representing DNA sequence segmentation through node-based tokenization improves the understanding of sequence structures and model performance. Additionally, including positional information from node IDs further boosts the training and predictive performance of DNA LMs. Our results show that pangenome-based tokenization schemes reduce training time and improve scalability compared to traditional methods, addressing the computational challenges of training LMs. Our contributions are as follows:

1. We introduce two pangenome graph-based tokenization schemes, PNT and PKMT, which provide more contextual information, enhancing LMs' ability to learn DNA sequence patterns and structures.

2. We propose a variant of pangenome graph segmentation that decouples from any private training data, enabling potential privacy-preserving training.

3. We demonstrate through experiments that our tokenization schemes outperform classical methods in training efficiency, predictive accuracy, and generation quality for LMs.

Following the introduction, the paper is structured as follows: Section 2 covers background on synthetic genome generation, Section 3 details tokenization schemes, Section 4 outlines evaluation metrics, Section 5 presents experiments, Section 6 discusses related work, and Section 7 concludes with limitations, implications, and future directions.

2 Background

2.1 Language Models

Large language models are advanced artificial intelligence systems designed to understand and generate language text based on the data on which they have been trained. These models, such as Mistral (Jiang et al., 2023), Anthropic's Claude (Anthropic, 2023), OpenAI's GPT series (Radford et al., 2019; OpenAI, 2023), Google's T5 (Raffel et al., 2020), Lamda (Thoppilan et al., 2022) and Gemini (Team et al., 2023), Meta's OPT (Zhang et al., 2022), BLOOM (Le Scao et al., 2023) and LLama (Touvron et al., 2023a,b), etc., take advantage of vast amounts of textual information to learn patterns, nuances, and complexities of language. LMs can perform a variety of language-related tasks, including answering questions, translating languages, and even participating in casual conversations. Their ability to process and generate coherent and contextually appropriate responses makes them invaluable tools across multiple fields, from customer service and education to creative writing and technical support.

In this paper, we focus on text generation tasks using LMs. The process involves three key steps:

Tokenization: The raw input text is converted into tokens based on different tokenization approaches.

Training: The model is trained from scratch on specific datasets.

Generation: In generative models such as GPT, the trained model predicts the next tokens given an initial prompt.

2.2 Pangenome Graph

The pangenome graph (Eizenga et al., 2020) represents genetic diversity within a species by integrating multiple genome sequences into a single comprehensive graph. In a pangenome graph,

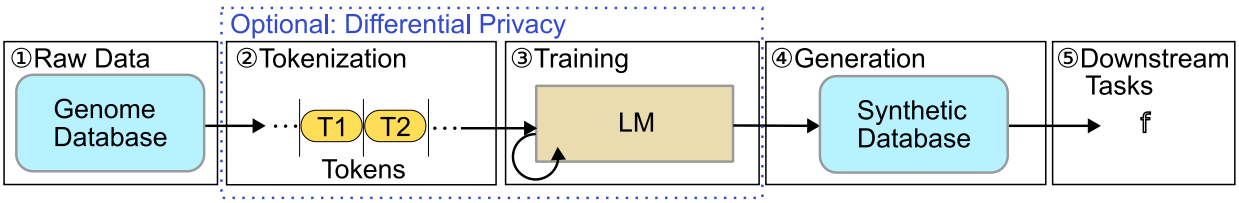


Figure 1: The whole pipeline of synthetic data generation and utilization.

nodes represent sequences of nucleotides, edges connect these sequences, showing the possible paths through the graph, and the paths through the pangenome graph represent the genomes of individuals, as demonstrated in Figure 3. The nodes in the pangenome graph represent the genetic sequences that are shared between the groups, while the edges represent the genetic variations. Tasks like genome-wide association (GWA) focus on the genotype matrix of the graph rather the exact DNA sequences. In this sense, it is the graph structure rather than the actual nucleotides that carries information.

2.3 Synthetic Genome Sequence Generation using LMs

In this work, our aim is to generate a synthetic genome sequence using LMs. In this section, we describe the complete pipeline for synthetic genome sequence generation using LMs, detailing each step from the original data processing to the downstream tasks, as shown in Figure 1.

① **Raw Data** (§5.1). The process begins with the acquisition of genomic data, which provides the genetic information needed for LM training.

② **Tokenization** (§3). Genomic sequences are converted into smaller units suitable for training using certain tokenization schemes.

③ **LM Training**. Tokenized sequences are used to train a GPT-style model using a next-token prediction approach, allowing the LM to learn patterns from the data without supervision.

④ **Generation** (§5.1). The trained LM generates synthetic genomic sequences by predicting subsequent tokens based on learned patterns.

⑤ **Downstream Tasks** (§4). Genomic tasks to which synthetic sequences can be applied.

We compare our schemes with the classical schemes by comprehensive experiments in §5.2.

3 Tokenization of a genome sequence

In this section, we first describe the widely used tokenization schemes and then introduce our tokenization schemes based on the pangenome graph. A glossary is provided in Table 4 in §A.1.

3.1 Classical tokenizations

3.1.1 Genome-based Single Nucleotide Tokenization (GSNT)

Genome-based Single Nucleotide Tokenization (GSNT) is a straightforward method to tokenize genome sequences, previously applied in (Nguyen et al., 2024b; Schiff et al., 2024). In this scheme, each nucleotide (A, C, G, T) is treated as an individual token. For instance, the genome sequence "ACGTA" would be tokenized as "A", "C", "G", "T", and "A".

3.1.2 Genome-based k-mer Tokenization (GKMT)

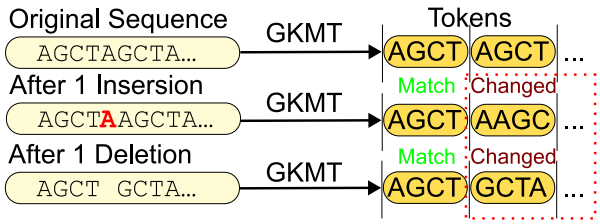


Figure 2: Insertion or Deletion of a sigle nucleotide change all following GKMT (stride equal to $k = 4$) tokens.

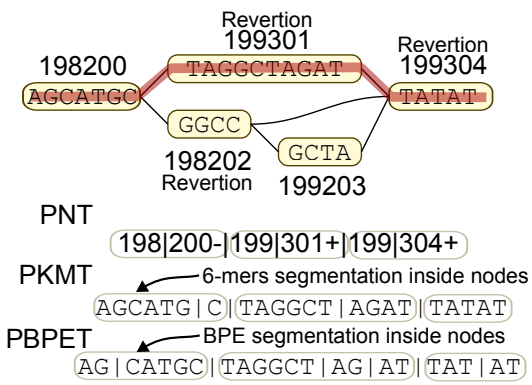


Figure 3: The pangenome graph based tokenizations output different segmented sequences of the red path. The above graph shows a slice of a pangenome graph with nodes marking the variations, edges marking possible paths, and the numbers marking the IDs.

An alternative is Genome-based k-mer Tokenization (GKMT), where k-mers, substrings of length k, used as tokens. For example, all 3-mers in the sequence "ACGTAG" are "ACG", "CGT", "GTA", and "TAG". Depending on the stride, the k-mers may overlap or not overlap (with a stride equal to k). We focus on the non-overlapping alternative. Compared to GSNT, GKMT provides a longer effective context length, but is also highly sensitive to sequence mutations or errors: a single nucleotide insertion or deletion can change all subsequent tokens as shown in Figure 2.

3.1.3 Genome-based BPE Tokenization (GBPET)

Genome-based Byte Pair Encoding Tokenization (GBPET), which is also used in recent studies (Zhou et al., 2023), applies the BPE algorithm (Sennrich et al., 2016) to genome sequences. BPE begins with single nucleotide tokens and iteratively merges the most frequent pairs of adjacent tokens to create a vocabulary of longer subword-like tokens. However, BPE training requires too large computational resources if very long DNA sequences are given as inputs. Manual segmentation is needed in GBPET, which could cause the same issue as GKMT.

3.2 Pangenome graph based tokenization

To address the limitations of standard schemes, we propose three novel tokenization methods based on the pangenome graph, illustrated in Figure 3.

3.2.1 Pangenome-based Node Tokenization (PNT)

The first scheme, Pangenome-based Node Tokenization (PNT), tokenizes DNA sequences based on the nodes in the pangenome graph. In this method, each node is treated as a token, where a node contains both the DNA sequence it represents and its position on the graph. Multiple nodes may correspond to the same DNA sequence but differ due to their positions in the graph. Consequently, the node ID vocabulary can be much larger (e.g., around 400K) compared to standard language vocabularies (e.g., 50K), presenting challenges for model training. To reduce the vocabulary size, we split the node IDs into two parts (first and second half) and include an additional indicator for sequence reversion (e.g., node 198202 in Figure 3 with reversion representing "GGCC" would be tokenized as '198' and '202+', and the unreverted node 198202 should be "CCGG").

A limitation of PNT is that it does not accommodate new sequences in the existing pangenome graph. Introducing new sequences requires rebuilding the entire graph, generating new IDs, and potentially altering the representation of previously established sequences learned by the model.

3.2.2 Pangenome-based k-mer Tokenization (PKMT)

The second scheme, Pangenome-based k-mer Tokenization (PKMT) segments DNA by splitting each node's sequence in the pangenome graph into non-overlapping k-mers. Unlike PNT, it uses nucleotide sequences rather than node IDs. We set $k = 6$ without padding; for example, the node sequence "TAGGCTAGAT" yields "TAGGCT" and "AGAT" in Figure 3. PKMT is more robust to insertions or deletions than GKMT, as the graph preserves alignment and isolates variations to affected nodes. However, it lacks the graph's positional encoding found in PNT, which may limit its ability to capture structural patterns in DNA.

3.2.3 Pangenome-based BPE Tokenization (PBPET)

The third scheme, Pangenome-based BPE Tokenization (PBPET), applies the Byte Pair Encoding algorithm to the sequences of nodes in the pangenome graph. Instead of segmenting node sequences into fixed-length k-mers as in PKMT, PBPET learns a vocabulary of frequently occurring sub-sequences across the nodes. In Figure 3, sub-sequences like "AG" or "AT" are identified. The learned vocabulary is then used to tokenize sequences, still with a first-step segmentation already done between nodes. This approach retains the graph-informed alignment of sequences, similar to PKMT, but benefits from the adaptive vocabulary of BPE.

4 Evaluating synthetic DNA generation quality

A main challenge of proving the utility of our schemes is how to evaluate the quality of the synthetic genome sequence generation. In our study, we use the prediction accuracy of the model to measure the quality of the generative model. Furthermore, we compare the similarity between synthetic and real genome sequences through sequence alignment.

4.1 Model prediction accuracy

Next token prediction accuracy: measures how often the model correctly predicts the next token given the correct previous tokens, making it the primary metric for models like GPT. However, this does not fully reflect sequence accuracy when tokenization is not single nucleotide-based. Predicting "AAAAAC" or "GCTGCT" for the true k -mer token "AAAAAA" count both as simply incorrect.

Character-level prediction accuracy: measures the percentage of nucleotides predicted correctly for each token, providing a more granular assessment of prediction quality. For example, predicting "AAAAAC" for the true token "AAAAAA" yields an accuracy of 0.83, while predicting "GCTGCT" results in an accuracy of 0.

4.2 Sequence alignment scores

The measurement of similarity between two genome sequences is done using sequence alignment, which is an essential process in many bioinformatic and computational biology tasks. Sequence alignment involves arranging the sequences of DNA, RNA, or even proteins, usually to identify regions of similarity. In our case, we use wfmash (Guarracino et al., 2021) where the wavefront algorithm (Marco-Sola et al., 2021) is primarily used for pairwise alignment between real and generated DNA sequences. Visualized results (introduced and shown in §5) and multiple scores can be used to evaluate the quality of the alignment.

An example of alignment between a reference sequence and a query sequence is shown in Figure 4.

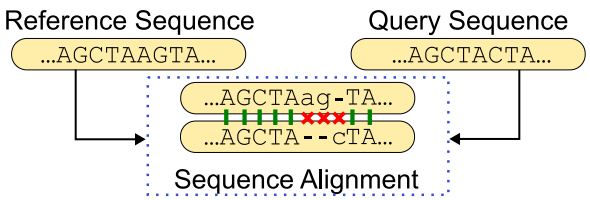


Figure 4: An alignment between two sequence. Capitalized nucleotide and green links indicate matches; lowercase nucleotide and red crosses indicate no match; the dashes in the sequences represent the gaps during matching.

An alignment score of 0 indicates no similarity, while a score of 1 represents a perfect match. Alignment scores can be defined and computed in two primary ways:

- BLAST identity (BI): $7/10 = 0.7$. Defined as the number of matching bases in relation to the number of alignment columns.
- Gap-Compressed Identity (GI): $7/9 = 0.78$. Counting the consecutive gaps in the query as one difference.

DNA sequences, including those in the MHC region, naturally exhibit high homology even across individuals, due to fundamental biological constraints. The alignment scores themselves can be considered sufficient as a representation of the utility of the synthetic sequences by measuring how close they are to the real data, preserving the properties needed. Alignment metrics align directly with the practical goals of genomic applications compared to divergence measures (Pillutla et al., 2021). Previous academic discussions (Frith, 2020; Durbin et al., 1998) have shown that alignment score effectively shows sequence similarity, and scores can indicate the potential usefulness of the compared data in downstream genomic tasks (5 in §2.3). A typical workflow involves projecting reads or mapping new data onto the reference genome, and then calling variants such as single nucleotide polymorphisms (SNPs) and insertions/deletions (indels). A higher score of a generated sequence against the real sequence suggests that the synthetic data can reliably substitute the real data, as further discussed in Appendix D.

5 Experiments

5.1 Datasets and LM choice

In our experiments, we used the human major histocompatibility complex (MHC) region of chromosome 6 as our dataset, which is cut out of the PGGB graph of HPRC year 1 assemblies (Liao et al., 2023). A total of 126 samples with 447 million nucleotides are in the dataset, with 80% of the samples used as the training set and 20% as the test set. During hyperparameter tuning, the "reference genome" was temporarily used as a validation set before being added back to the training set for final training. We tested the performance of the openly available GPT-2 (Radford et al., 2019) and Llama (Touvron et al., 2023a) model architectures with 90M parameters, which support a prompt length of 1024/2048 tokens, using the Hugging Face 4.24.0 library of transformers (Wolf, 2019). GPT-2 and Llama are chosen due to their well-established performance and robustness as a classical publicly available language model, and the relatively small 90M total parameter size is selected to balance performance and computational

overhead. We split the long genome into 10k base pairs sequences in GBPET training and set the vocabulary size to 4096 for both BPE methods, as in DNABERT2 (Zhou et al., 2023).

Table 1: Training time (hours) of each tokenization scheme on 90M models for 90 epochs.

Model	GSNT	GKMT	PKMT	GBPET	PBPET	PNT
GPT-2	56	11	15	17	24	7
LLaMA	20	5	6	9	12	3

5.2 Experiment results

We trained the GPT-2 and Llama models on the dataset using four tokenization schemes: GSNT, GKMT, PNT and PKMT. Training was carried out for 90 epochs (§B.1 shows results with more epochs) with a batch size of 16/8 and 1024/2048-token sequences for GPT-2/Llama. The dataset comprises 124 DNA samples totaling 447 million nucleotides. Training times are shown in Table 1, obtained on 8 NVIDIA A5500 GPUs. Figure 5 displays token and character-level prediction accuracies. PNT not included in the character-level accuracy figures due to the vague definition on predictions and targets with too varied lengths.

Training times and model performance differ significantly across tokenization schemes as shown in Table 1 and Figure 5. The final accuracies are shown in Table 2. PNT demonstrated the fastest training time, while GSNT is generally the slowest due to its larger token set. BPE based method are slower than k-mer based but faster than GSNT. PNT reaches the best peak accuracy the fastest, while GKMT has the worst performance. GSNT initially trains much faster than PKMT for token prediction, but converges to a similar final accuracy. We will see how they perform differently in the alignment. Despite having almost the same token tables, we can clearly tell PKMT's pangenome graph-aided segmentation helps the model to outperform the on trained by GSNT. The training of PBPET tokenizer takes around 20 seconds, while the training of GBPET tokenizer takes about 10 minutes, largely due to the larger sequence chunks, and they both have moderate training time.

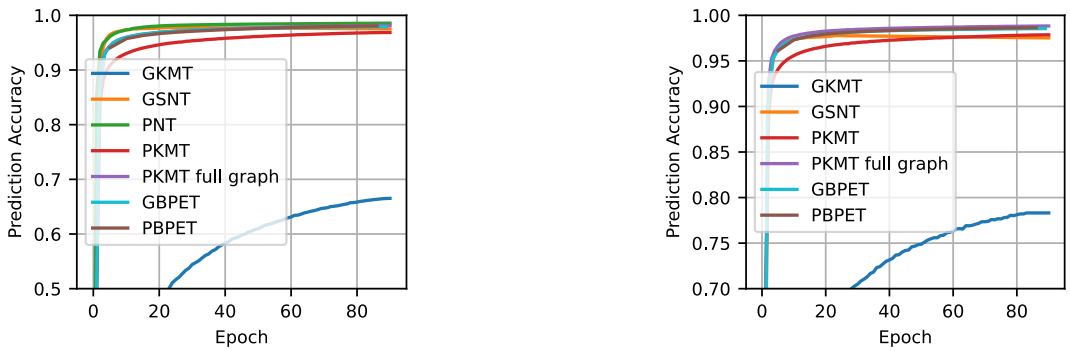
We present the alignment results for the GPT-2 generated sequences of the tokenization schemes in Figure 6 (GKMT barely generates sequences that align at all), aligned against the reference sequence of the dataset. The X-axis represents reference sequence positions, and the Y-axis shows different generated sequences aligned to the reference. Each dot or line marks a generated sequence position aligned with the reference genome. After 90 epochs, only PNT generates sequences closely aligned with the reference over long contexts for GPT-2. Some sequences show no alignment, likely due to random sampling for diversity and learned misalignments from the training data. Llama, achieving comparable token prediction accuracy, performs very similar to GPT-2 results. However, it is generally with less dense dots and dashes, indicating fewer matches, as shown in Appendix B. Llama is also capable of generating long sequences using PNT. However, the alignments tend to terminate prematurely. Even with longer prompts, Llama appears to struggle more in regions with higher mutation rates (observe the denser dots along the alignment lines), causing the generation to deviate more easily from the intended sequence. Llama cannot generate long sequences even with PKMT or PBPET.

Table 2: Final accuracy of each tokenization scheme on 90M models trained for 90 epochs.

Model	GSNT	GKMT	PKMT	GBPET	PBPET	PNT
Token Prediction Accuracy						
GPT-2	97.1%	65.9%	96.9%	97.9%	98.0%	98.6%
LLaMA	98.7%	81.8%	97.7%	98.5%	98.6%	98.8%
Character-Level Accuracy						
GPT-2	97.1%	78.3%	97.9%	98.6%	99.0%	–
LLaMA	98.7%	85.3%	98.6%	99.0%	99.3%	–

To quantify generation quality, we show the alignment scores of the generated sequences against the entire data set (the best match of a query against the entire dataset) in Table 3, with the results for real data as a comparison. In addition to GI / BI scores, we show the alignment percentage, indicating the proportion of well-aligned sequences. The segment length refers to the size of the minimizer window during alignment. PNT achieves the highest alignment scores across all segment lengths, while GSNT performs the worst.

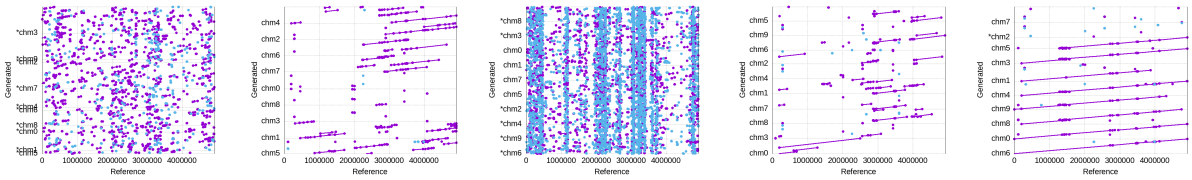
PNT demonstrates superior token-level prediction accuracy, while GKMT achieves the highest character-level accuracy in GPT-2 and closely rivals PNT in Llama. Traditional methods underperform, with GKMT achieving less than 70% accuracy and GSNT training significantly slower. The accuracy gap is more pronounced in alignment scores (Table 3), where PNT consistently excels with GI and BI scores of around 0.99 in segment lengths of 1k to 200k, closely mirroring the performance of real data. Although PKMT produces fewer high-quality sequences than GSNT that align



(a) Token prediction accuracy

(b) Character level prediction accuracy

Figure 5: Model prediction accuracies of all tokenization schemes during GPT-2 training.



(a) GSNT (b) PKMT (c) GBPET (d) PBPET (e) PNT

Figure 6: Alignment of a batch of GPT-2 generated sequences against the reference sequence. The X-axis represents the reference sequence; the Y-axis shows generated sequences. Longer lines indicate consistent alignment, and denser dots represent frequent short matches. Alignment results from Llama are presented in Appendix B.

Table 3: Alignment percentages and weighted GI/BI scores of the 20 generated sequences per scheme for different segment lengths, aligned against the original dataset. Real data metrics are computed using 80% of samples as references and 20% as queries.

Segment	1k			20k			50k			200k		
GPT-2	**Align %**	**GI**	**BI**	**Align %**	**GI**	**BI**	**Align %**	**GI**	**BI**	**Align %**	**GI**	**BI**
GSNT	81.66	0.8712	0.9955	21.55	0.8834	0.9893	1.42	0.8323	0.9849	0.00	0.0000	0.0000
PKMT	52.96	0.9443	0.9856	50.34	0.9036	0.9932	47.87	0.8977	0.9936	8.82	0.8656	0.9919
GBPET	71.81	0.9873	0.9981	53.19	0.9105	0.9931	36.93	0.9041	0.9921	0.00	0.0000	0.0000
PBPET	44.75	0.9081	0.9914	42.03	0.9044	0.9943	42.29	0.9007	0.9935	9.39	0.9029	0.9955
PNT	89.34	0.9977	0.9999	31.01	0.9961	0.9990	33.27	0.9920	0.9985	36.96	0.9873	0.9982
LLaMA	**Align %**	**GI**	**BI**	**Align %**	**GI**	**BI**	**Align %**	**GI**	**BI**	**Align %**	**GI**	**BI**
GSNT	7.17	0.7927	0.9906	0.00	0.0000	0.0000	0.00	0.0000	0.0000	0.00	0.0000	0.0000
PKMT	34.45	0.9666	0.9960	21.19	0.8323	0.9907	6.85	0.8232	0.9876	0.00	0.0000	0.0000
GBPET	12.90	0.9543	0.9870	0.00	0.0000	0.0000	0.00	0.0000	0.0000	0.00	0.0000	0.0000
PBPET	33.00	0.9817	0.9969	6.41	0.8533	0.9861	5.86	0.8356	0.9878	0.00	0.0000	0.0000
PNT	28.80	0.9796	0.9958	5.97	0.9970	0.9984	8.49	0.9958	0.9987	15.02	0.9938	0.9977
Real data	99.97	0.9994	0.9999	69.23	0.9996	0.9999	61.37	0.9991	0.9997	50.67	0.9981	0.9993

with the reference, it achieves slightly higher alignment scores than GSNT in more settings and has a chance for relatively good generation for large segments. The newer non-pengenome-based method, GBPET, performs better in alignment score specifically under smaller segment length, but still lacks stable long-sequence generation compared with pangenome powered PBPET. PNT-generated sequences hold greater potential for applications resembling real data, while others may require further refinement or model optimization. Llama overall shows the same trend, but lags behind GPT-2 in sequence generation, despite higher prediction accuracy and longer prompt length, likely due to greater performance degradation in very limited parameter numbers for continuous predictions. Llama specifically underperforms in non-pangenome based tokenization methods. Overall, PKMT performs better than GSNT (and GKMT), and PBPET performs better than GBPET, directly indicating the usefulness of involving pangenome graph structure in tokenization. One limitation we observe is that pangenome-based models occasionally generate almost entire no match. Classical methods, although they generate fragmented pieces, do not completely miss. For PNT specifically, adding a small 20 token

prompt will completely fix this issue.

Discussion. To our knowledge, this work is the first to compare the effectiveness of pangenome-based tokenization schemes to classical tokenization schemes when utilizing the LMs to learn the pattern of DNA sequences; and also the one of the first to demonstrate the efficacy of LMs in generating very long DNA.

Our findings reveal that the pangenome graph structure embeds significant and meaningful information, improving neural networks' understanding of DNA sequences. Our experiments demonstrate how this information can be effectively exploited. The significant gap between GKMT and PKMT emphasizes the effectiveness of leveraging graph structure in tokenization. Despite having similar token tables, the graph-aided segmentation of PKMT provides more stable and learnable structural information, resulting in better model training speed and overall generation quality. Our results underscore the trade-offs between computational cost and model performance, with pangenome graph-based tokenization schemes showing higher accuracy across tasks. Previous work (Liao et al., 2023) demonstrates how improved matching is the key point of the pangenome, which "aligns" with our use of the pangenome graph here.

6 Related work

In this section, we introduce two common genome tasks wwith machine learning application. Table 7 in Appendix C summarizes this section.

6.1 Classification Tasks

Classification tasks are common in genomics, including (more details in Appendix D):

Variant Calling: ML models identify genetic variants such as SNPs and indels in genomes, linking them to diseases or traits. DeepVariant (Poplin et al., 2018), a CNN-based variant caller, outperforms traditional methods, influencing many others (Yun et al., 2020; Kolesnikov et al., 2021). Clairvoyante (Luo et al., 2019) excels in single-molecule sequencing (SMS), while Clair (Luo et al., 2020) offers faster RNN-based inference with fewer parameters, without sacrificing accuracy.

Gene Expression Analysis: ML models analyze gene expression data to reveal gene-disease relationships. Classical methods like KNN (Kim and Kim, 2018), linear/logistic regression (Han et al., 2019), and SVMs (Wan et al., 2019) are used to predict driver genes or cancer risk. CNNs (Lyu and Haque, 2018; Elbashir et al., 2019) are also applied for cancer classification with RNA-seq data.

Beyond these, CNNs model protein binding (Alipanahi et al., 2015), cell type identification (Yao et al., 2019), and non-coding variants (Zhou and Troyanskaya, 2015). RNNs predict non-coding DNA functions (Quang and Xie, 2016) and RNA-protein binding preferences (Shen et al., 2020). Transformer models like DNA-BERT (Ji et al., 2021; Zhou et al., 2023; Dalla-Torre et al., 2023, 2025) provide strong contextual embeddings for molecular phenotype prediction but face context size limitations due to quadratic scaling. Recent models like Hyena (Nguyen et al., 2024b) and MambaDNA (Schiff et al., 2024) address these limitations with sub-quadratic scaling for longer contexts. More recent applications of DNA LM like MoDNA (An et al., 2022) for promoter prediction, and GENA (Fishman et al., 2023) for multiple tasks, both use traditional GKMT. Some papers like GPN-MSA (Benegas et al., 2024) for genome-wide variant effect prediction uses GSNT. DNABERT-2 (Zhou et al., 2023) and following work (Karollus et al., 2024) for evolutionary conservation and functional annotation prediction use BPE.

A recent paper (Zhang et al., 2024) presents a similar tokenization approach using pangenome graphs. Although both works independently develop this idea, ours differs by incorporating PNT and PBPET, and focusing on long-sequence generation. In contrast, their work handles shorter sequences (max 5000bp) with node-aided k-mer tokenization and focuses on classification tasks.

6.2 Generation Tasks

Synthetic Data Generation: Synthetic data mimics real data for privacy concerns. GANs have been used for synthetic medical data (Bae et al., 2019) and DNA sequences coding for proteins (Gupta and Zou, 2018), though limited by fixed output sizes and requiring DP for stronger guarantees. Some work (Avdeyev et al., 2023) utilizes transformers but with limited generation length, and a more recent large model (Nguyen et al., 2024a) shows generation of submillions in length with a certain level of genomic organization.

De Novo Genome Assembly: This involves reconstructing a genome from short DNA fragments without a reference. Deep learning has been applied to de novo peptide sequencing (Tran et al., 2017, 2019; Yang et al., 2019).

7 Limitations

While our study focused on smaller models to establish a proof-of-concept for our tokenization scheme, we acknowledge that larger models may improve results but raise practical concerns around efficiency and resource use. Furthermore, emerging architectures designed for long-context processing (e.g., (Gu et al., 2021; Nguyen et al., 2024b,a; Gu and Dao, 2023; Peng et al., 2023a)) could potentially further enhance the performance of all tokenization schemes. These models, by enabling longer effective context windows, could improve both the understanding of long-range dependencies in DNA and the consistency of sequence generation. Although we believe that pangenome-based tokenization retains advantages in effective segmentation, such models may help close the performance gap for other tokenization methods. We agree that this is a valuable direction and suggest that future work explores scaling to larger models and incorporating long-context architectures to more fully assess their potential impact.

References

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

Weizhi An, Yuzhi Guo, Yatao Bian, Hehuan Ma, Jinyu Yang, Chunyuan Li, and Junzhou Huang. 2022. Modna: motif-oriented pre-training for dna language model. In *Proceedings of the 13th ACM international conference on bioinformatics, computational biology and health informatics*, pages 1–5.

Anthropic. 2023. Claude 2. *Anthropic Blog*. Accessed: 2024-09-03.

Pavel Avdeyev, Chenlai Shi, Yuhao Tan, Kseniia Dudnyk, and Jian Zhou. 2023. Dirichlet diffusion score model for biological sequence generation. In *International Conference on Machine Learning*, pages 1276–1301. PMLR.

Ho Bae, Dahuin Jung, Hyun-Soo Choi, and Sungroh Yoon. 2019. Anomigan: Generative adversarial networks for anonymizing private medical data. In *Pacific Symposium on Biocomputing 2020*, pages 563–574. World Scientific.

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

Dennis A Benson, Mark Cavanaugh, Karen Clark, Ilene Karsch-Mizrachi, David J Lipman, James Ostell, and Eric W Sayers. 2012. Genbank. *Nucleic acids research*, 41(D1):D36–D42.

Brian L Browning and Sharon R Browning. 2016. Genotype imputation with millions of reference samples. *The American Journal of Human Genetics*, 98(1):116–126.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, and 1 others. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

1000 Genomes Project Consortium and 1 others. 2012. An integrated map of genetic variation from 1,092 human genomes. *Nature*, 491(7422):56.

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, and 1 others. 2023. The nucleotide transformer: Building and evaluating robust foundation models for human genomics. *bioRxiv*, pages 2023–01.

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, and 1 others. 2025. Nucleotide transformer: building and evaluating robust foundation models for human genomics. *Nature Methods*, 22(2):287–297.

Richard Durbin, Sean Eddy, Anders Stærmose Krogh, and Graeme Mitchison. 1998. Biological sequence analysis: Probabilistic models of proteins and nucleic acids.

Jordan M Eizenga, Adam M Novak, Jonas A Sibbesen, Simon Heumos, Ali Ghaffaari, Glenn Hickey, Xian Chang, Josiah D Seaman, Robin Rounthwaite, Jana Ebler, and 1 others. 2020. Pangenome graphs. *Annual review of genomics and human genetics*, 21:139–162.

Murtada K Elbashir, Mohamed Ezz, Mohanad Mohammed, and Said S Saloum. 2019. Lightweight convolutional neural network for breast cancer classification using rna-seq gene expression data. *IEEE Access*, 7:185338–185348.

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

Martin C Frith. 2020. How sequence alignment scores correspond to probability models. *Bioinformatics*, 36(2):408–415.

Richard A Gibbs, John W Belmont, Paul Hardenbol, Thomas D Willis, Fuli L Yu, HM Yang, Lan-Yang Ch'ang, Wei Huang, Bin Liu, Yan Shen, and 1 others. 2003. The international hapmap project.

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

Albert Gu, Karan Goel, and Christopher Ré. 2021. Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*.

Andrea Guarracino, Njagi Mwaniki, Santiago Marco-Sola, and Erik Garrison. 2021. wfmash: whole-chromosome pairwise alignment using the hierarchical wavefront algorithm.

Marco Guevara, Shan Chen, Spencer Thomas, Tafadzwa L Chaunzwa, Idalid Franco, Benjamin H Kann, Shalini Moningi, Jack M Qian, Madeleine Goldstein, Susan Harper, and 1 others. 2024. Large language models to identify social determinants of health in electronic health records. *npj Digital Medicine*, 7(1):6.

Anvita Gupta and James Zou. 2018. Feedback gan (fbgan) for dna: A novel feedback-loop architecture for optimizing protein functions. *arXiv preprint arXiv:1804.01694*.

Yi Han, Juze Yang, Xinyi Qian, Wei-Chung Cheng, Shu-Hsuan Liu, Xing Hua, Liyuan Zhou, Yaning Yang, Qingbiao Wu, Pengyuan Liu, and 1 others. 2019. Driverml: a machine learning algorithm for identifying driver genes in cancer sequencing studies. *Nucleic acids research*, 47(8):e45–e45.

Thomas Hartvigsen, Saadia Gabriel, Hamid Palangi, Maarten Sap, Dipankar Ray, and Ece Kamar. 2022. Toxigen: A large-scale machine-generated dataset for adversarial and implicit hate speech detection. *arXiv preprint arXiv:2203.09509*.

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

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, and 1 others. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Alexander Karollus, Johannes Hingerl, Dennis Gankin, Martin Grosshauser, Kristian Klemon, and Julien Gagneur. 2024. Species-aware dna language models capture regulatory elements and their evolution. *Genome Biology*, 25(1):83.

Byung-Ju Kim and Sung-Hou Kim. 2018. Prediction of inherited genomic susceptibility to 20 common cancer types by a supervised machine-learning method. *Proceedings of the National Academy of Sciences*, 115(6):1322–1327.

Alexey Kolesnikov, Sidharth Goel, Maria Nattestad, Taedong Yun, Gunjan Baid, Howard Yang, Cory Y McLean, Pi-Chuan Chang, and Andrew Carroll. 2021. Deeptrio: variant calling in families using deep learning. *bioRxiv*, pages 2021–04.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. *arXiv preprint arXiv:2003.02245*.

Jack Lanchantin, Ritambhara Singh, Beilun Wang, and Yanjun Qi. 2017. Deep motif dashboard: visualizing and understanding genomic sequences using deep neural networks. In *Pacific symposium on biocomputing 2017*, pages 254–265. World Scientific.

Eric S Lander, Lauren M Linton, Bruce Birren, Chad Nusbaum, Michael C Zody, Jennifer Baldwin, Keri Devon, Ken Dewar, Michael Doyle, William Fitzhugh, and 1 others. 2001. Initial sequencing and analysis of the human genome. *Nature*, 409(6822):860–921.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, and 1 others. 2023. Bloom: A 176b-parameter open-access multilingual language model.

Heng Li. 2018. Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics*, 34(18):3094–3100.

Wen-Wei Liao, Mobin Asri, Jana Ebler, Daniel Doerr, Marina Haukness, Glenn Hickey, Shuangjia Lu, Julian K Lucas, Jean Monlong, Haley J Abel, and 1 others. 2023. A draft human pangenome reference. *Nature*, 617(7960):312–324.

Ruibang Luo, Fritz J Sedlazeck, Tak-Wah Lam, and Michael C Schatz. 2019. A multi-task convolutional deep neural network for variant calling in single molecule sequencing. *Nature communications*, 10(1):998.

Ruibang Luo, Chak-Lim Wong, Yat-Sing Wong, Chi-Ian Tang, Chi-Man Liu, Chi-Ming Leung, and Tak-Wah Lam. 2020. Exploring the limit of using a deep neural network on pileup data for germline variant calling. *Nature Machine Intelligence*, 2(4):220–227.

Boyu Lyu and Anamul Haque. 2018. Deep learning based tumor type classification using gene expression data. In *Proceedings of the 2018 ACM international conference on bioinformatics, computational biology, and health informatics*, pages 89–96.

Santiago Marco-Sola, Juan Carlos Moure, Miquel Moreto, and Antonio Espinosa. 2021. Fast gap-affine pairwise alignment using the wavefront algorithm. *Bioinformatics*, 37(4):456–463.

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, and 1 others. 2024a. Sequence modeling and design

from molecular to genome scale with evo. *Science*, 386(6723):eado9336.

Eric Nguyen, Michael Poli, Marjan Faizi, Armin Thomas, Michael Wornow, Callum Birch-Sykes, Stefano Massaroli, Aman Patel, Clayton Rabideau, Yoshua Bengio, and 1 others. 2024b. Hyenadna: Long-range genomic sequence modeling at single nucleotide resolution. *Advances in neural information processing systems*, 36.

Sergey Nurk, Sergey Koren, Arang Rhie, Mikko Rautiainen, Andrey V Bzikadze, Alla Mikheenko, Mitchell R Vollger, Nicolas Altemose, Lev Uralsky, Ariel Gershman, and 1 others. 2022. The complete sequence of a human genome. *Science*, 376(6588):44–53.

Paul Ohm. 2009. Broken promises of privacy: Responding to the surprising failure of anonymization. *UCLA l. Rev.*, 57:1701.

OpenAI. 2023. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Donovan H Parks, Michael Imelfort, Connor T Skennerton, Philip Hugenholtz, and Gene W Tyson. 2015. Checkm: assessing the quality of microbial genomes recovered from isolates, single cells, and metagenomes. *Genome research*, 25(7):1043–1055.

Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Matteo Grella, and 1 others. 2023a. Rwkv: Reinventing rnns for the transformer era. *arXiv preprint arXiv:2305.13048*.

Cheng Peng, Xi Yang, Aokun Chen, Kaleb E Smith, Nima PourNejatian, Anthony B Costa, Cheryl Martin, Mona G Flores, Ying Zhang, Tanja Magoc, and 1 others. 2023b. A study of generative large language model for medical research and healthcare. *arXiv preprint arXiv:2305.13523*.

Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. 2021. Mauve: Measuring the gap between neural text and human text using divergence frontiers. *Advances in Neural Information Processing Systems*, 34:4816–4828.

Ryan Poplin, Pi-Chuan Chang, David Alexander, Scott Schwartz, Thomas Colthurst, Alexander Ku, Dan Newburger, Jojo Dijamco, Nam Nguyen, Pegah T Afshar, and 1 others. 2018. A universal snp and small-indel variant caller using deep neural networks. *Nature biotechnology*, 36(10):983–987.

Daniel Quang and Xiaohui Xie. 2016. Danq: a hybrid convolutional and recurrent deep neural network for quantifying the function of dna sequences. *Nucleic acids research*, 44(11):e107–e107.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

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

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

Zhen Shen, Qinhu Zhang, Kyungsook Han, and De-Shuang Huang. 2020. A deep learning model for rna-protein binding preference prediction based on hierarchical lstm and attention network. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 19(2):753–762.

Latanya Sweeney, Akua Abu, and Julia Winn. 2013. Identifying participants in the personal genome project by name (a re-identification experiment). *arXiv preprint arXiv:1304.7605*.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, and 1 others. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, and 1 others. 2022. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, and 1 others. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Ngoc Hieu Tran, Rui Qiao, Lei Xin, Xin Chen, Chuyi Liu, Xianglilan Zhang, Baozhen Shan, Ali Ghodsi, and Ming Li. 2019. Deep learning enables de novo peptide sequencing from data-independent-acquisition mass spectrometry. *Nature methods*, 16(1):63–66.

Ngoc Hieu Tran, Xianglilan Zhang, Lei Xin, Baozhen Shan, and Ming Li. 2017. De novo peptide sequencing by deep learning. *Proceedings of the National Academy of Sciences*, 114(31):8247–8252.

Erwin L Van Dijk, Yan Jaszczyszyn, Delphine Naquin, and Claude Thermes. 2018. The third revolution in sequencing technology. *Trends in Genetics*, 34(9):666–681.

Nathan Wan, David Weinberg, Tzu-Yu Liu, Katherine Niehaus, Eric A Ariazi, Daniel Delubac, Ajay Kannan, Brandon White, Mitch Bailey, Marvin Bertin, and 1 others. 2019. Machine learning enables detection of early-stage colorectal cancer by whole-genome sequencing of plasma cell-free dna. *BMC cancer*, 19:1–10.

John N Weinstein, Eric A Collisson, Gordon B Mills, Kenna R Shaw, Brad A Ozenberger, Kyle Ellrott, Ilya Shmulevich, Chris Sander, and Joshua M Stuart. 2013. The cancer genome atlas pan-cancer analysis project. *Nature genetics*, 45(10):1113–1120.

K. A. Wetterstrand. 2021. Dna sequencing costs: Data from the nhgri genome sequencing program (gsp). https://www.genome.gov/sequencingcostsdata. National Human Genome Research Institute.

Matthias Wjst. 2010. Caught you: threats to confidentiality due to the public release of large-scale genetic data sets. *BMC medical ethics*, 11:1–4.

T Wolf. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Hao Yang, Hao Chi, Wen-Feng Zeng, Wen-Jing Zhou, and Si-Min He. 2019. pnovo 3: precise de novo peptide sequencing using a learning-to-rank framework. *Bioinformatics*, 35(14):i183–i190.

Kai Yao, Nash D Rochman, and Sean X Sun. 2019. Cell type classification and unsupervised morphological phenotyping from low-resolution images using deep learning. *Scientific reports*, 9(1):13467.

Burak Yelmen, Aurélien Decelle, Linda Ongaro, Davide Marnetto, Corentin Tallec, Francesco Montinaro, Cyril Furtlehner, Luca Pagani, and Flora Jay. 2021. Creating artificial human genomes using generative neural networks. *PLoS genetics*, 17(2):e1009303.

Kang Min Yoo, Dongju Park, Jaewook Kang, Sang-Woo Lee, and Woomyeong Park. 2021. Gpt3mix: Leveraging large-scale language models for text augmentation. *arXiv preprint arXiv:2104.08826*.

Taedong Yun, Helen Li, Pi-Chuan Chang, Michael F Lin, Andrew Carroll, and Cory Y McLean. 2020. Accurate, scalable cohort variant calls using deepvariant and glnexus. *Bioinformatics*, 36(24):5582–5589.

Haoyang Zeng, Matthew D Edwards, Ge Liu, and David K Gifford. 2016. Convolutional neural network architectures for predicting dna–protein binding. *Bioinformatics*, 32(12):i121–i127.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, and 1 others. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Xiang Zhang, Mingjie Yang, Xunhang Yin, Yining Qian, and Fei Sun. 2024. Deepgene: An efficient foundation model for genomics based on pan-genome graph transformer. *bioRxiv*, pages 2024–04.

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

Zhihan Zhou, Yanrong Ji, Weijian Li, Pratik Dutta, Ramana Davuluri, and Han Liu. 2023. Dnabert-2: Efficient foundation model and benchmark for multi-species genome. *arXiv preprint arXiv:2306.15006*.

A More on tokenization schemes

A.1 Glossary of Frequent Acronyms

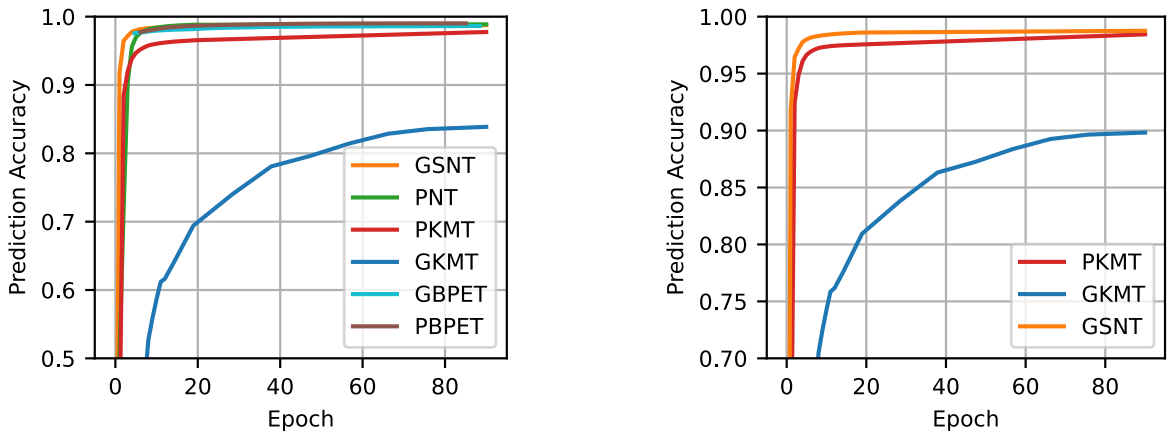
Table 4: Glossary of Frequent Acronyms

Acronym	Explanation
GSNT	Genome-based Single Nucleotide Tokenization
GKMT	Genome-based k-mer Tokenization
GBPET	Genome-based BPE Tokenization
PNT	Pangenome-based Node Tokenization
PKMT	Pangenome-based k-mer Tokenization
PBPET	Pangenome-based BPE Tokenization

A.2 Public graph-based PKMT tokenization

The proposed PKMT schemes aim to provide more context and help models learn DNA sequence patterns more effectively. However, they can also risk leaking sensitive information about individual samples. For additional techniques like Differentially Private Stochastic Gradient Descent (DP-SGD) to be implemented during training, tokenization should be *independent of the private dataset* or protected by appropriate mechanisms for the whole scheme to be DP compatiable.

Making PNT DP-friendly is challenging, as the ID-to-sequence mapping can expose private data. Although the static vocabulary of PKMT avoids this issue during token mapping, the use of the pangenome graph, where segmentation depends on every sequence in the dataset, still breaks the guarantee. To mitigate this, we propose building a "public" pangenome graph from publicly accessible



(a) Token prediction accuracy of the model across different training epochs

(b) Character-level prediction accuracy of the model across different training epochs

Figure 7: Model prediction accuracy of the four tokenization schemes during LLaMA training. PNT is excluded from the character-level accuracy plot due to the ambiguity in defining accuracy when predicted and target sequences differ in length.

Table 5: Alignment percentages and weighted GI/BI scores for segment lengths 5k and 100k.

Segment	5k			100k		
Model	**Align %**	**GI**	**BI**	**Align %**	**GI**	**BI**
GPT-2						
GSNT	59.49	0.8919	0.9910	0.00	0.0000	0.0000
PKMT	53.76	0.9015	0.9960	38.43	0.8928	0.9939
GBPET	63.20	0.9082	0.9961	14.62	0.9035	0.9884
PBPET	46.22	0.9019	0.9966	38.04	0.8927	0.9920
PNT	73.27	0.9970	0.9997	33.17	0.9945	0.9988
LLaMA						
GSNT	0.41	0.7414	0.9784	0.00	0.0000	0.0000
PKMT	30.82	0.8362	0.9917	0.00	0.0000	0.0000
GBPET	0.10	0.3070	0.9479	0.00	0.0000	0.0000
PBPET	24.75	0.8609	0.9916	0.00	0.0000	0.0000
PNT	25.79	0.9977	0.9997	10.96	0.9964	0.9990
Real data	97.97	0.9994	0.9999	60.44	0.9989	0.9997

datasets. This graph can then be used to tokenize the private dataset. Subsequences identified in the public graph are tokenized as corresponding nodes, while unrecognized subsequences are tokenized as standard k-mers, preserving privacy. The pseudocode for this segmentation is provided in §A.2, and is used in the experiments for PKMT. As the key idea behind PKMT is to be more extendable than PNT to new unknown (test) data, we use this realization in our experiments with the graph only built on the training set.

In our experiment, we split an existing graph as a public graph and the private sequences. We build the public pangenome graph as shown in Protocol 1 and then complete the PKMT as shown in Protocol 2.

B More experiment details and results

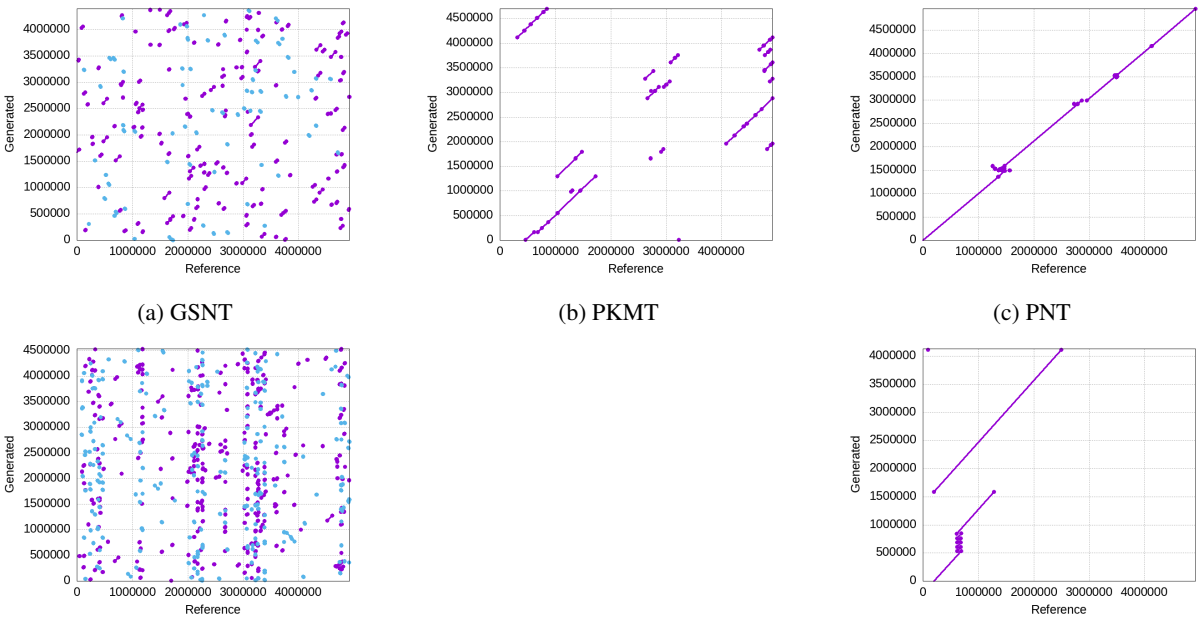
We run the experiments in a cluster of 8 NVIDIA A5500 GPUs. The GPT-2 model uses the gelu_new activation function, consists of 12 transformer layers, each with 12 attention heads, and an embedding dimension of 768 with maximum prompt being 1024. The LLaMA model uses the SiLU activation function and consists of 6 transformer layers, each with 8 attention heads, and an embedding dimension of 768. It has an intermediate size of 4096, and supports sequences up to a maximum of 2048 positions. We used a grid search for the best hyperparameters. We use 3e-4 (except 5e-4 for GSNT - GPT-2 and 1e-4 for 1e-4 for GSNT Llama) leaning rate, batch size 8/16 for GPT-2/Llama training; and topk=10, topp=0.92, topk_decend_min=5 for generation, which is also determined by grid search. To avoid the risk of memorization, we ensure the enough randomness in the generation, and that the alignment scores are not perfect (up to 1) and include real-to-real alignment baselines as a reference. We show additional alignment scores in Table 5 and the Llama accuracy in Figure 7. A

Algorithm 1 $G_{pub} \leftarrow \Pi_{PubGraph}(G, Pub)$: Define Public Pangenome Graph Nodes

Input: A pangenome graph G, list of indexes Pub with public sequences. We use $G[i][j]$ to represent the node j of the sequence i in G and $\mathsf{Seq}(G[i][j])$ to represent the actual sequence.
Output: The way nodes are merged in the public pangenome graph recorded in M_{pub}.
Initialization:
Initialize M_{pub} as an empty dictionary to store the public pangenome graph nodes.
for each sequence i in Pub **do**
  for each node j in $G[i]$ **do**
    if $G[i][j]$ has fixed previous/next nodes in G **then**
      Combine $G[i][j]$ with the fixed previous/next nodes as a single node.
      Record the combined node in M_{pub}.
    else
      Record $G[i][j]$ as an independent node in M_{pub}.
    end if
  end for
end for
Return: M_{pub} as the public pangenome graph nodes.

Algorithm 2 $\mathsf{Segmented} \leftarrow \Pi_{PKMT}(G, Pub, Priv)$: Perform PKMT Based on Public Sequences Only

Input: A pangenome graph G, list of indexes Pub with public sequences and $Priv$ with private sequences. We use $G[i][j]$ to represent the node j of the sequence i in G. We use $\mathsf{Seq}(G[i][j])$ to represent the actual sequence.
Output: Segmented DNA sequences recorded in $\mathsf{Segmented}$.
$G_{pub} = \Pi_{PubGraph}(G, Pub, Priv)$
Initialize $\mathsf{Segmented} = \{\}$
for each sequence i in $\{Pub, Priv\}$ **do**
  Initialize $\mathsf{Chain} = [\,]$
  Initialize $\mathsf{UndefinedChain} = [\,]$
  Initialize $\mathsf{Segmented}[i] = [\,]$
  for each node j in $G[i]$ **do**
    Add $\mathsf{Seq}(G[i][j])$ to Chain
    if current node chain ends according to M_{pub} **then**
      Append $\mathsf{UndefinedChain}$ to $\mathsf{Segmented}[i]$ as a segment of the sequence $G[i]$
      Append Chain to $\mathsf{Segmented}[i]$ as a segment of the sequence $G[i]$
      Clear $\mathsf{UndefinedChain}$
      Clear Chain
    else if current node pattern is not recorded in M_{pub} **then**
      Append Chain to $\mathsf{UndefinedChain}$
      Clear Chain
    end if
  end for
  Cut each segment in $\mathsf{Segmented}[i]$ into non-overlapping 6-mers
end for
Return: $\mathsf{Segmented}$



(a) GSNT (b) PKMT (c) PNT

(d) GBPET (e) PBPET

Figure 8: Alignment of a single generated sequence against the reference. Longer lines represent continuous alignment regions, while scattered dots show shorter matching fragments.

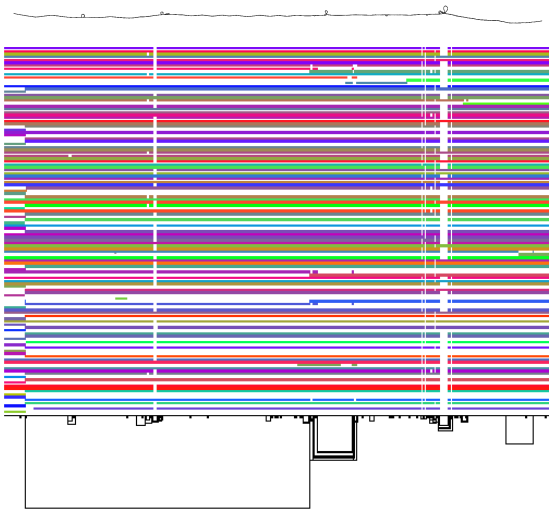


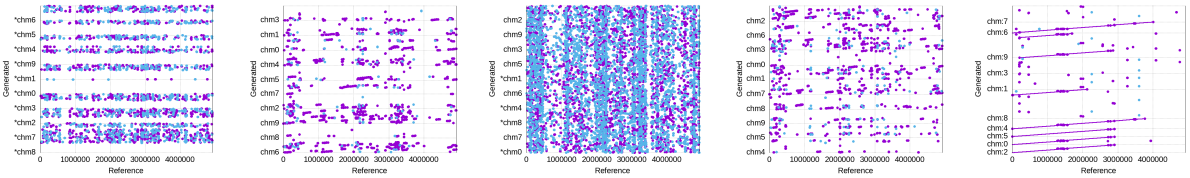
Figure 9: The pangenome graph of the human major histocompatibility complex (MHC) region of chromosome 6 of the PGGB graph of HPRC year 1 assemblies, with 2D graph visualization (above) and matrix view (below). The circled in the 2D graph and the gaps in the matrix view indicate mutations.

clearer single query view of alignment is shown in Figure 8 for a single generated sequence, and the alignment figures for Llama are in Figure 10. Figure 9 shows a simple illustration of a small pangenome graph of the MHC data we use.

B.1 Effects of extensive training

During our experiment, we found that PNT, GBPET and PBPET did not benefit from more training epochs but GSNT and PKMT had the potential for further improvement. We trained the better-performing GPT-2 model on half of the training dataset for an extra 200 epochs, keeping other parameters the same to further investigate the best possible performance these two tokenization schemes can provide. The token prediction accuracy increased by about 0.4% for PKMT and 0.3% for GSNT, which is marginal, but we observed significant improvements in generation quality for both methods in alignment score. While prediction accuracy gains may appear small, they have a compounding effect during generation, where errors accumulate across long sequences. Accuracy reflects only top-1 correctness for the next token, whereas generation samples probabilistically from the top candidates, making it more sensitive to distributional improvements. The results are shown in Figure 11 and Table 6. Both methods achieved slightly higher alignment scores and aligned length, especially with larger segments. Both tokenization schemes still underperformed compared to PNT, even after extensive training. Figure 11 additionally clearly shows that GKMT generates relatively longer sequences with more longer lines.

The higher utility of the extensively trained model indicates that substantial investment in com-

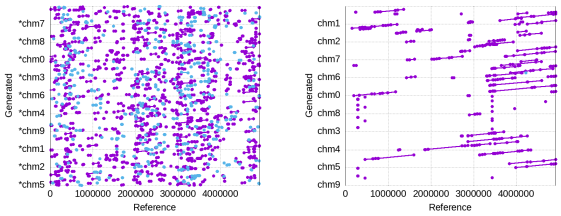


(a) GSNT (b) PKMT (c) GBPET (d) PBPET (e) PNT

Figure 10: Alignment of a batch of LLaMA-generated sequences against the reference. The X-axis is the reference, and the Y-axis shows the generated sequences. Longer lines indicate consistent alignment, and denser dots indicate frequent short matches.

Table 6: Alignment percentages and weighted GI/BI scores of the 20 generated sequences each scheme for different segment lengths of the generated sequences with extensively trained GPT-2 model, against the test set as reference.

Segment	**1k**			**5k**			**20k**		
	Align %	**GI**	**BI**	**Align %**	**GI**	**BI**	**Align %**	**GI**	**BI**
GSNT	90.67	0.8818	0.9972	75.52	0.8922	0.9926	42.17	0.8916	0.9920
PKMT	81.42	0.9842	0.9978	81.87	0.9027	0.9969	79.74	0.9044	0.9956
Segment	**50k**			**100k**			**200k**		
	Align %	**GI**	**BI**	**Align %**	**GI**	**BI**	**Align %**	**GI**	**BI**
GSNT	9.76	0.8801	0.9916	0.00	0.0000	0.0000	0.00	0.0000	0.0000
PKMT	72.52	0.9011	0.9940	65.51	0.8936	0.9943	16.19	0.8883	0.9935



(a) GSNT (b) PKMT

Figure 11: Alignment of a batch of generated sequences (after extensive GPT-2 training) against training sequences. The X-axis is the reference; the Y-axis contains generated sequences. Longer lines indicate consistent alignments, while denser dots reflect frequent short matches.

putational power has its potential.

C Summarizing related work

Here we provide a table to summarize our discussion in §6, with a detailed list of the related work of ML/DL doing genomic tasks.

D Alignment scores and downstream tasks

Alignment-based evaluations provide a more direct assessment of how well synthetic data supports real-world genomic applications. For example, datasets like those from the Human Pangenome Project depend heavily on alignment-based metrics to assess data quality and interpret genetic variation. Read alignment to a reference genome followed by variant calling is a widely adopted pipeline, and here alignment consistency and accuracy are critical. In this context, alignment scores are not only practical but also well-recognized within the genomics community as meaningful indicators of quality.

In this section, we introduce two essential tasks to show how alignment scores can determine the utility of sequences, and how synthetic sequences can play a role.

D.1 Variant calling

Read alignment and variant calling are foundational tasks in bioinformatics pipelines, especially in genome resequencing studies. In this process, DNA reads generated by sequencing technologies are aligned to a reference genome to reconstruct the original genetic material and identify variants (e.g., calling the inserting and deletion in the bottom two sequences when compared with the top reference in Figure 2). Determining an accurate alignment is critical because downstream variant calling algorithms rely on these mappings to compare the sample DNA against the reference. Numerous tools have been developed to perform this task efficiently and accurately, including Minimap2 (Li, 2018) and wfmash (Guarracino et al., 2021). Most work in §6.1 measure the alignment in their experiment.

A high alignment score indicates a strong match between the sequenced read and a region in the reference genome, minimizing mismatches, gaps, or ambiguous placements. This is essential to identify true variants confidently, ruling out sequencing errors or misalignments. An incorrect alignment may map a query DNA sequence to the wrong location in the reference genome, leading to wrong variant

calls. An example is given in Figure 12. Synthetic sequences can serve as references in variant calls or generate potential variant combinations that are not observed in natural samples.

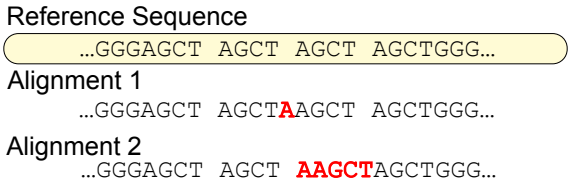


Figure 12: Two possible alignment of a sequence to a reference sequence. Alignment 1 calls for one insertion while Alignment 2 calls for 4 deletion then 5 insertion. Alignment 1 will have higher alignment scores with more matched nucleotides, and is considered a better alignment. Therefore the variant calling based on Alignment 1 is considered better than Alignment 2.

D.2 De novo assembly

De novo assembly reconstructs a genome from short sequencing reads without relying on a reference genome. This process stitches overlapping reads into contiguous sequences (contigs) or scaffolds, aiming to rebuild the original genome as accurately as possible. Since there is no reference during assembly, evaluation is typically performed by aligning the assembled contigs back to a trusted reference genome, or comparing them to known markers or conserved genes.

A high alignment score here indicates that the assembler has likely reconstructed a biologically accurate sequence. This suggests high contiguity, low error rates, and minimal misassemblies. Low alignment scores often signal fragmented or misassembled regions. Synthetic sequences can act as trussted reference, improving the assembly.

Many utility metrics used in existing genome modeling studies are fundamentally rooted in sequence alignment. For example, in recent work such as (Nguyen et al., 2024a), tools like CheckM (Parks et al., 2015) are used to report quality metrics, including gene density and stop codon frequencies. These tools rely on foundational components like profile Hidden Markov Models (pHMMs) that are directly constructed from multiple sequence alignments, with alignment quality and consistency playing a central role in shaping their parameters and performance. In this context, a high alignment score indicates strong homology or functional similarity between the generated sequence and known sequences, providing evidence of biological plausibility.

Table 7: DL models used in genome tasks.

Job Type	Paper	Task	Architecture	Input
Classification	(Poplin et al., 2018; Yun et al., 2020; Kolesnikov et al., 2021)	Variant Calling	CNN	hundreds of base pairs
	(Luo et al., 2019)	Variant Calling	CNN	hundreds of base pairs
	(Lyu and Haque, 2018; Elbashir et al., 2019)	Cancer Prediction	CNN	RNA-seq gene expression data
	(Alipanahi et al., 2015)	Protein Binding	CNN	10-100 nucleotides & binding specificities
	(Zeng et al., 2016)	Protein Binding	CNN	10-100 base pairs & binding specificities
	(Yao et al., 2019)	Cell Type Identification	CNN	cell images
	(Zhou and Troyanskaya, 2015)	Non-coding DNA function prediction	CNN	1k base pairs
	(Luo et al., 2020)	Variant Calling	RNN	binary alignment map (BAM)
	(Shen et al., 2020)	RNA-protein binding preference	LSTM	embedded k-mers
	(Quang and Xie, 2016)	Non-coding DNA function prediction	CNN/BLSTM	one hot encoded nucleotides
	(Kim and Kim, 2018)	Cancer Prediction	KNN	SNP genotype syntaxes (8-mers)
	(Han et al., 2019)	Cancer Prediction	Rao score	Mutation Annotation Format (MAF)
	(Wan et al., 2019)	Cancer Prediction	SVM	Human EDTA plasma samples
	(Ji et al., 2021; Zhou et al., 2023)	Molecular Phenotype Prediction	Transformer	tokenized k-mers
	(Dalla-Torre et al., 2023)	Molecular Phenotype Prediction	Transformer	tokenized k-mers
	(Nguyen et al., 2024b)	5-way Species Classification	Transformer	single nucleotide tokens
	(Schiff et al., 2024)	Genome Tasks	Mamba	single nucleotide tokens
	(Luo et al., 2019)	Variant Calling	CNN	Hundreds of base pairs
	(An et al., 2022)	Promoter Prediction	Transformer	6-mers of up to 512bp
	(Karollus et al., 2024)	Evolutionary Conservation / Functional Annotations	Transformer	6-mers for 128bp sequences
	(Fishman et al., 2023)	Multiple Tasks	Transformer	BPE tokens, up to 36000bp sequence
	(Benegas et al., 2024)	Genome-wide Variant Effect Prediction	Transformer	GSNT for 128bp sequences
	(Dalla-Torre et al., 2025)	Multiple Prediction Tasks	Transformer	Thousands of k-mer tokens
Generation	(Tran et al., 2017)	De novo peptide sequencing	LSTM/CNN	tandem mass spectrometry (MS/MS) Spectrum
	(Tran et al., 2019)	De novo peptide sequencing	LSTM/CNN	data-independent acquisition (DIA) mass spectrometry data
	(Yang et al., 2019)	De novo peptide sequencing	learning-to-rank	tandem mass spectrometry data
	(Bae et al., 2019)	Synthetic Medical Data	GAN	medical data
	(Gupta and Zou, 2018)	Synthetic DNA Sequences	GAN	DNA sequences
	(Avdeyev et al., 2023)	Synthetic DNA Sequences	Transformer	Up to 1024 base-pairs
	(Nguyen et al., 2024a)	Synthetic DNA Sequences	Transformer	Up to 131072 base-pairs