Genomic Foundationless Models: Pretraining Does Not Promise Performance

Kirill Vishniakov [1] Karthik Viswanathan [1] Aleksandr Medvedev [1] Praveenkumar Kanithi [1]
Marco AF Pimentel [1] Ronnie Rajan [1] Shadab Khan [1]

Abstract

The success of Large Language Models has inspired the development of Genomic Foundation Models (GFMs) through similar pretraining techniques. However, the relationship between pretraining performance and effectiveness in downstream genomic tasks remains unclear. Additionally, the high computational cost of pretraining raises questions about its cost-efficiency. To assess the usefulness of pretraining in genomics, we evaluated seven different GFMs across 52 diverse genomic tasks, comparing them to their counterparts with randomly initialized weights. Surprisingly, we found that randomly initialized models can match or even surpass the performance of pretrained GFMs in finetuning and feature extraction tasks. We also discovered that pretrained GFMs fail to capture clinically relevant genetic mutations, which are crucial for understanding genetic disorders and phenotypic traits. Our results indicate that most of the current pretrained GFMs lack a "foundational" understanding of genomics and provide minimal utility, even for basic tasks such as sequence classification. These findings collectively highlight the need for critically rethinking the pretraining approaches for genomics. Our code is available at github.com/m42-health/gfm-random-eval.

1. Introduction

Recent advances in language modeling have led to the application of similar unsupervised pretraining approaches in genomics. This facilitated the emergence of Genomic Foundation Models (GFMs) (Consens et al., 2025) which learn representations from genomic sequences. This line of work has attracted considerable attention due to the potential of GFMs to revolutionize our understanding of genomics (Benegas et al., 2025b).

GFMs typically use a two-step training approach akin to Large Language Models: unsupervised pretraining on a large dataset, followed by a supervised training. The pretraining phase usually involves either next token prediction (Brown et al., 2020) or masked language modeling (Devlin et al., 2019). The promise of unsupervised pretraining is to extract knowledge from vast genomic datasets (Consortium et al., 2015) and compress it into the model's parameters, with the aim of producing a generalist model applicable to a diverse set of tasks.

While some studies have explored scaling laws for GFMs (Nguyen et al., 2023; 2024), the relationship between pretraining and downstream performance remains unclear, with no single GFM consistently proving to be the best (Marin et al., 2024). Combined with large model sizes (Dalla-Torre et al., 2024), long input sequences (Nguyen et al., 2023; 2024; Brixi et al., 2025) and massive datasets, the pretraining step demands substantial computational resources.

The natural question arises: *how effective is unsupervised pretraining in the genomics domain?* To answer this, we conduct extensive experiments with seven recent GFMs across finetuning, feature extraction and genomic variation analysis as summarized in Figure 1.

First, our experiment results reveal that in standard finetuning tasks (Figure 1A), such as Nucleotide Transformer Benchmark (Dalla-Torre et al., 2024), GUE (Zhou et al., 2024), and Genomics Benchmark (Grešová et al., 2023), randomly initialized models trained from scratch in a supervised manner perform either better then or on par with finetuned pretrained GFMs, suggesting that current pretraining approaches may not provide a significant advantage over random weight initialization.

This surprising trend continues to be observed in feature extraction task (Figure 1B), where embeddings from frozen models are used to train a simple classifier. Here, one would expect the benefits of pretraining to be most pronounced, as a randomly initialized models receive no tuning whatsoever, and, hence, there weights remain completely random. Instead, we find the opposite. Not only do randomly initialized models perform competitively, but simple architecture modifications, such as changing the tokenizer and increas-

[1] M42, Abu Dhabi, UAE. Correspondence to: Shadab Khan <skhan.shadab@gmail.com>.

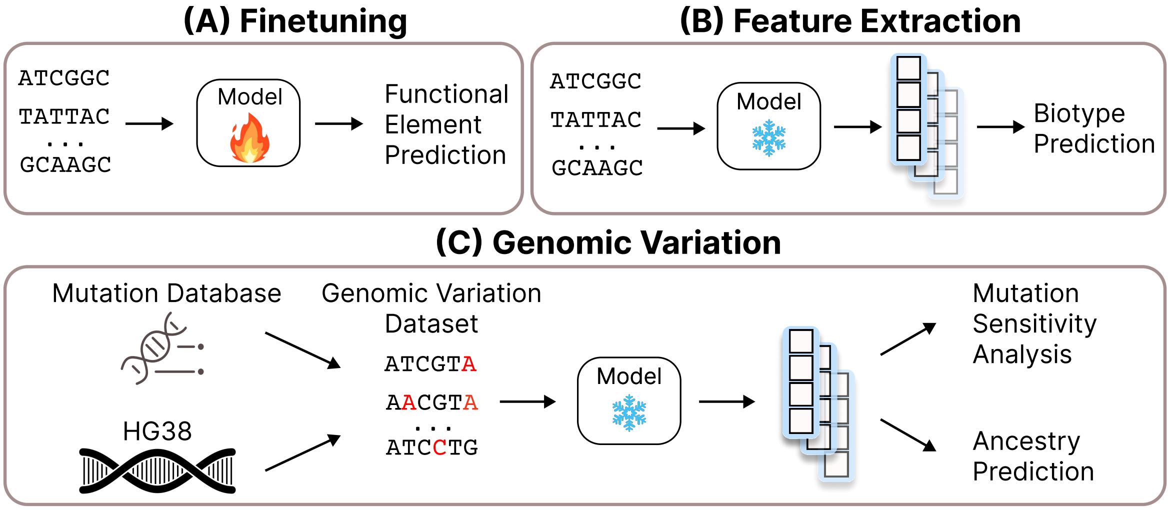

Figure 1. **Overview of the experiments.** (A) **Finetuning:** we finetune models on different functional element classification tasks. (B) **Feature Extraction:** For biotype classification, we extract embeddings from frozen models and train a simple classifier to predict gene types using these embeddings. (C) **Genomic Variation:** We evaluate models' ability to capture genetic variations through two tasks: (1) Mutation sensitivity analysis measures how well models distinguish between original and mutated sequences by computing embedding similarities, and (2) Ancestry prediction uses model embeddings with XGBoost to classify population groups based on genomic variants. Both tasks use sequences constructed by combining HG38 reference genome with mutation data.

Model	#Params	Architecture	Tokenizer	Vocab Size	Seq Len (tokens)	#Tokens	Data
HyenaDNA	450K	Decoder	Char	12	1024	2.6B	HRG
NT 500M	500M	Encoder	k-mer	4107	1000	300B	1000G
NTv2 50M	50M	Encoder	k-mer	4107	2048	300B	Multispecies
GENA-LM	110M	Encoder	BPE	32000	512	1T	HRG+1000G
DNABERTv2	117M	Encoder	BPE	4096	128	262B	Multispecies
Caduceus	8M	Encoder	Char	12	131K	35B	HRG
Mistral	580M	Decoder	Char	12	4096	150B	1000G

Table 1. **Description of models evaluated in this study.** The analyzed models differ in architecture, pretraining objective, tokenizer, model size, and pretraining dataset. We analyze the pretrained models and their randomly initialized counterparts. *#Tokens* refers to the number of tokens seen by the model during the pretraining. *Data* refers to the pretraining dataset source.

ing the embedding dimension, allow an entirely untrained HyenaDNA counterpart to significantly outperform every pretrained GFM on this benchmark.

Finally, we assessed the models on one of the most practically important applications for genomics: detecting subtle genomic variations (Figure 1C). This scenario require models to be highly sensitive to single nucleotide changes within long sequences. We found that most pretrained GFMs fail in these tasks. For instance, even when up to half of the nucleotides in a DNA sequence are changed, some GFMs still produce embeddings with over 0.99 cosine similarity to the original sequence. As a result, GFMs are currently unsuitable for applications that rely extensively on mutation data, including variant pathogenicity prediction, eQTL (Zhou & Troyanskaya, 2015), sQTL (Garrido-Martín et al., 2021), and phenotype prediction.

Our results challenge current unsupervised pretraining methods used in genomics, suggesting that simply adapting NLP techniques is insufficient for developing true genomic understanding. Rather than continuing to invest substantial computational resources in existing pretraining methods, we advocate for critically rethinking the fundamental building blocks of genomic foundation models. This includes developing biologically-informed tokenization strategies and establishing new robust benchmarks that comprehensively test for the understanding of genomic mechanisms.

2. Models

We selected six recently published GFMs for evaluation and also trained our own version of the Mistral (Jiang et al., 2023) model on 50 samples from the 1000 Genomes dataset (Consortium et al., 2015). The models in our analysis exhibit significant diversity in their architectures, pretraining objectives, tokenizers, model sizes, and pretraining datasets. Our model selection includes both encoder and decoder architectures, transformer-based and state-space models, with model sizes ranging from 450K to 580M parameters. Interestingly, our Mistral outperforms all other previous GFMs on many tasks. We attribute the success of Mistral to an advanced architecture recipe which includes

RoPE embeddings, big embedding dimension and character tokenizer. Model configurations are summarized in Table 1, and model descriptions are provided in Section A.2 of the Appendix.

We excluded the EVO model (Nguyen et al., 2024) from our analysis as it was trained on bacterial genomes and performed poorly in our preliminary tests on the Nucleotide Transformer Benchmark.

Random weight initialization of models throughout the paper follows the procedure from the Transformers library (Wolf et al., 2020) for each particular model. This usually involves initializing linear layers with values drawn from $\mathcal{N}(0, 0.02)$, and LayerNorm layers are initialized with $\gamma = 1$. Full random initialization details for each model are provided in Section A.3 in Appendix.

3. Experiments and Results

3.1. Finetuning

To verify the usefulness of pretraining, we finetuned both pretrained and randomly initialized versions of the models on **Nucleotide Transformer Benchmark** (Dalla-Torre et al., 2024), **Genome Understanding Evaluation (GUE)** (Zhou et al., 2024), and **Genomic Benchmarks** (Grešová et al., 2023) with exactly the same set of hyperparameters. This set of benchmarks together constitutes **52** genomic classification tasks. In total, we conducted nearly **10,000** finetuning experiments, this considers: seven models, both pretrained and random, evaluated across different tasks, folds, and learning rates.

To ensure robustness, we performed a broad hyperparameter search over learning rate, weight decay, batch size, warm-up steps, LoRA (Hu et al., 2022) vs full finetuning, and others. Because performance was most sensitive to the learning rate, the final run consisted of a sweep over six learning-rate values, and we report the best result obtained. We also found that full finetuning consistently outperformed LoRA (Table 7), suggesting that it provides the best opportunity for the model to reach the full score.

Moreover, each task was run on 3 different folds, and the results were averaged. We used a validation holdout set for model selection and reported test scores for the epoch that corresponded to the highest score on validation set. Full hyperparameter details for these experiments are provided in Section A.5 in the Appendix. We display our results for these finetuning experiments in Figure 2.

For each task, we first find the highest score among all randomly initialized models; for example, if the scores obtained from randomly initialized models are 0.3, 0.4, and 0.5, we consider 0.5 as the best random score. We then plot the difference between each pretrained model's performance and the best random score. Green bars show where pretrained models outperform the best random model, while red bars show where they underperform. Ideally, if the pretraining is useful, we expect to see a predominance of tall green bars.

NT Benchmark results for histone and enhancer tasks are displayed in the top part of Figure 2. For the GUE Benchmark, we aggregate results by task categories. For example, in the Epigenetic Marks category, we average the scores across all histone modification tasks. Similarly, we compute average scores for other categories: Promoter Detection, TF Prediction Human and Mouse, Core Promoter Detection, and Splice Site Detection (which contains a single task). These aggregated results are presented in the middle part of Figure 2. For Genomic Benchmarks we display the performance for six different tasks in the bottom part of Figure 2.

The results in Figure 2 demonstrate that big pretrained models often perform worse than small randomly initialized models. This is visible by the big proportion of the red bars indicating that the best random model performance is higher than of the pretrained models. Notably, the randomly initialized Caduceus, despite having only 8M parameters, emerges as the best random model in six out of twelve tasks on NT Benchmark, four out of six tasks on GUE, and in two out of six tasks on Genomic Benchmarks. In general, randomly initialized Caduceus significantly outperforms larger pretrained models, including NT 500M, NTv2 50M, GENA-LM with 110M parameters, and DNABERTv2 with 117M parameters, and often even its own pretrained version. On NT Benchmark tasks of H3K9ac, H3K4me1, and H3K36me3 the randomly initialized Caduceus outperforms NTv2 50M, HyenaDNA, and GENA-LM, it is also better than NT 500M by about 0.1 MCC, **while being 60 times smaller (8M vs 500M)**.

Another good randomly initialized model is DNABERTv2. On challenging histone tasks on NT Benchmark, the randomly initialized DNABERTv2 with 117M parameters outperformed the pretrained NT 500M by about 0.35 MCC on H3K4me3 and by more than 0.2 MCC on H3K4me2 and H3K14ac. This difference in MCC is quite significant. In general, for most datasets, the best randomly initialized model outperformed, on average, three or four pretrained models and consistently achieved performance comparable to the best pretrained model.

The results on GUE in the middle part of Figure 2 demonstrate an even more pronounced advantage of randomly initialized models compared to the NT Benchmark. In TF Prediction (Mouse), randomly initialized Caduceus shows remarkable performance, outperforming all pretrained models. Similar trend is observed on Core Promoter Detection Group where randomly initialized Mistral outperforms all pretrained models including its own pretrained version.

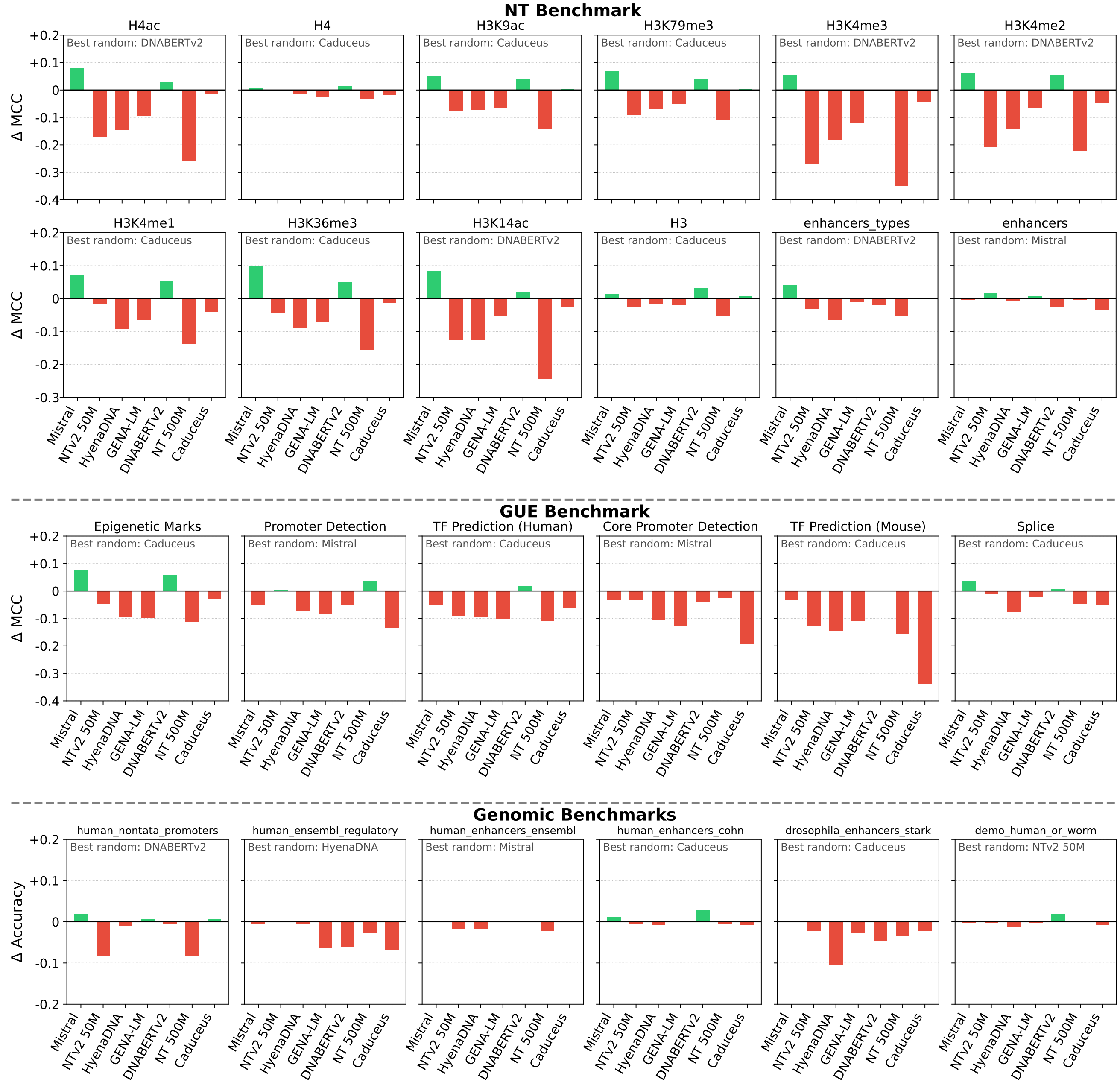

Figure 2. **Difference of performance between pretrained and the best random model on NT Benchmark.** For each task, we finetuned each model, starting from both pretrained and randomly initialized weights. Green bars indicate the advantage of pretrained models, and red bars indicate the advantage of the best random model. The best random model consistently outperforms several pretrained ones on each task, highlighting the inefficiency of current pretraining approaches in genomics. In most cases, the best random model is Caduceus which has only 8M parameters, yet it has better performance than much bigger pretrained models such as NT 500M, GENA-LM, DNABERTv2, NTv2 50M, and Mistral.

In general on GUE benchmark the best randomly initialized model outperforms five to seven pretrained models. In Genomic Benchmarks similar trend of competitiveness of randomly initialized models can be observed.

To showcase the individual performance of randomly initialized models, we present their results on NT Benchmark alongside pretrained models in Figure 3. For instance, the "Enhancers" subgroup includes all enhancer-related tasks, while the "Histone" subgroup covers all histone tasks, and so on. In addition, we also show this plot for Splice Sites and Promoter on NT Benchmark in Figure 6, and also for GUE and Genomic Benchmarks in Figure 7 and Figure 8 in the Appendix. We also provide results for all models on NT Benchmark in Table 18 in Appendix.

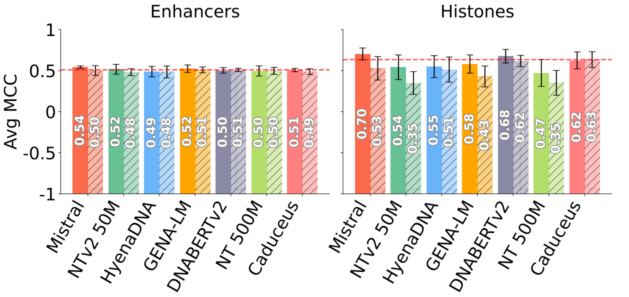

Figure 3. **NT Benchmark performance per subgroup.** Pretrained models are shown with clear bars, and randomly initialized with dashed. For enhancer subgroup all random models show competitive performance with pretrained. For histones random Caduceus outperforms five pretrained model including its own pretrained version. Red dashed line indicates MCC score of the best randomly initialized model.

The results presented in Figure 3 highlight that randomly initialized models can perform remarkably well across all subgroups of the NT Benchmark. In the ”Enhancers” subgroup, all randomly initialized models perform comparably to their pretrained counterparts. In histone tasks, the best random models, DNABERTv2 and Caduceus, reach average MCC scores of 0.62 and 0.63, outperforming pretrained NTv2 50M, HyenaDNA, GENA-LM, and NT 500M. In case of randomly initialized Caduceus it also outperforms its own pretrained version.

The results across all three benchmarks demonstrate that while not all randomly initialized models consistently outperform pretrained ones, we identified several randomly initialized models like Caduceus, DNABERTv2, and HyenaDNA that can match or exceed pretrained performance across a wide range of tasks.

Unlike foundation models in other domains, such as computer vision (Radford et al., 2021) and NLP (Brown et al., 2020), where pretraining typically leads to significant improvements in downstream task performance, the current pretraining strategies in genomics are barely able to outperform randomly initialized models. Moreover, even in cases where pretrained models maintain an advantage, the gains from pretraining are surprisingly small - typically within 2–3%. Together, these modest gains may not justify the large amounts of compute required for pretraining in genomics (Dalla-Torre et al., 2024), especially for these commonly used fine-tuning tasks.

> ***Finding 1:*** Randomly initialized models can match or outperform pretrained models across diverse finetuning tasks tested here, indicating that current genomic pretraining strategies do not consistently yield substantial improvements on these widely-used benchmarks.

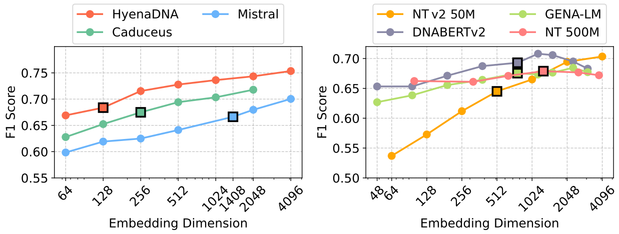

Figure 4. **Embedding dimension experiments.** We change embed_dim in randomly initialized models when using together with char tokenizer. The default embed_dim values are shown with square marker. Increasing embed_dim improves the performance.

3.2. Feature Extraction

The biotype classification task assesses the quality of features extracted from both pretrained and randomly initialized models. Unlike in the NT Benchmark, where models undergo finetuning, this task does not involve updating model weights. In other words, *embeddings for randomly initialized models were extracted without any finetuning and were entirely based on their initial random weights*.

Using sequences and biotype labels from the Gencode repository (Harrow et al., 2012), we extracted features from models with frozen weights and applied max pooling along the token dimension. These pooled features were then used to train an XGBoost classifier to predict among nine biotype labels. Detailed information about the dataset is presented in Section A.6 of the Appendix.

We observed that the choice of tokenizer significantly impacts the performance of randomly initialized encoder-only models. In particular, switching these models from their default k-mer or BPE tokenizers with large vocabularies (Table 1) to a character tokenizer that has only four tokens substantially improved their performance (third row of Table 2, right part). For example, for NTv2 50M, the performance increased from 0.48 to 0.64. Character-level tokenization is standard for decoder models, hence the identical results in the second and third rows of Table 2 for decoder-only models. This improvement likely stems from random models' difficulty handling large vocabulary sizes. In contrast, the char tokenizer's reduced search space (four tokens) enables better predictions from random models.

Initially, the random HyenaDNA model achieved the highest F1 score (0.69) among random models despite its small 128-dimension embeddings, prompting us to investigate how embedding size affects performance. We tested various embedding dimensions across all models, keeping other parameters constant and ensuring divisibility by the number of attention heads. Complete embedding configurations are detailed in Table 10 in the Appendix. Figure 4 presents detailed plots for the embedding dimension experiments. It reveals a clear trend of improved performance as the embedding dimension increases for all five models examined.

Tokenizer	Pretrain	Decoder-only Mistral	 HyenaDNA	 Caduceus	Encoder-only NTv2 50M	 GENA-LM	 DNABERTv2	 NT 500M
DEFAULT	✓	0.730	0.638	0.423	0.679	0.704	0.654	0.662
DEFAULT	✗	0.667	0.690	0.674	0.482	0.574	0.651	0.603
CHAR	✗	0.666	0.690	0.674	0.642	0.668	0.696	0.669
+LARGER EMBED DIM	✗	0.700	0.753	0.717	0.703	0.684	0.708	0.678
PRETRAINED − RANDOM		3.0%	-11.5%	-29.4%	-2.4%	2.0%	-5.4%	-1.6%

Table 2. **Biotype classification results.** Embeddings extracted from pretrained and randomly initialized models were used to train an XGBoost classifier. Switching to character tokenizer (3rd row) and increasing the embedding dimension (4th row) significantly improved performance, allowing most randomly initialized models to surpass their pretrained counterparts. The bottom row shows difference in performance between pretrained model and optimized randomly initialized model. Negative values indicate the advantage of the random models. F1 score is reported.

HyenaDNA shows consistent improvements, reaching an F1 score of nearly 0.75 at 4096 dimensions. NTv2 50M exhibited a more dramatic improvement, with its F1 score rising from 0.53 to 0.71. Additionally, we performed the same set of experiments on 10 Histone modification tasks from GUE benchmark. As shown in Table 12, random HyenaDNA with embedding dimension of 2048 is best on 9 out of 10 tasks, outperforming every pretrained model.

As shown in the fourth row of Table 2, increasing the embedding dim and using a char tokenizer allowed randomly init. models to outperform pretrained in 5 out of 7 cases.

> ***Finding 2:*** Embeddings from pretrained models do not show a clear advantage over those from models with random weights. Additionally, random models optimized with simple changes, like swapping the tokenizer and increasing the embedding dimension, outperform pretrained models.

3.3. Genomic Variation

This section transitions from functional element classification to genomic variation tasks, analyzing mutations like single nucleotide polymorphisms (SNPs), insertions, and deletions between individuals. While functional elements remain largely consistent across populations, genomic variations are individual-specific and can significantly impact phenotype and disease risk. These tasks present a unique challenge for GFMs, which must detect and interpret subtle sequence differences, often down to single nucleotide changes, to understand human genetic diversity and its health implications.

3.3.1. Ancestry Prediction

Ancestry prediction is a multilabel classification task that predicts an individual's ancestry using a small portion of their genome. We constructed an ancestry dataset from 1000G data (Consortium et al., 2015), using HG38 and applying mutations from each 1000G sample to obtain 32K-base consensus sequences. These sequences differ by 0.5% of positions, with an average of 33 variants (SNPs, insertions, and deletions). Embeddings generated from these sequences were used as features for XGBoost classification.

Model	Pretrained init.	Random init.
Mistral	0.74	0.67
NTv2 50M	0.68	0.68
HyenaDNA	0.68	0.69
GENA-LM	0.74	0.71
DNABERT	0.74	0.74
NT 500M	0.69	0.69
Caduceus	0.75	0.76

Table 3. **Ancestry classification results.** All the experiments are done with models initialized from pretrained checkpoints and random initialization. F1 score is averaged over eleven regions. Randomly initialized Caduceus is achieving the highest score outperforming all pretrained models.

When generating the dataset, we selected eleven different regions of the genome, treating each as a separate fold, and evaluated our models on each region independently, reporting average metrics (Table 3). A detailed description of the benchmark is available in Section A.7 in the Appendix.

Results in Table 3 show that randomly initialized models generally match pretrained models' performance. Only Mistral and GENA-LM showed marginal improvement with pretraining (F1 difference: 0.07 and 0.03). Caduceus achieved the highest F1 score (0.76) in both random and pretrained versions. The NT 500M model, despite being trained on 1000G variants, showed no advantage over its random initialization. This performance pattern could stem from two factors: the high masking probability (15%) in masked language modeling, which exceeds the natural mutation rate (0.5%), and the k-mer tokenization (6 nucleotides) that poorly captures single nucleotide variations. These choices might have caused the model's ability to learn meaningful genetic variant representations.

3.3.2. Mutation Sensitivity Analysis

We further investigated models' limited ancestry prediction performance and assess their capability in capturing subtle genomic variations, by conducting experiments to explicitly measure model sensitivity to SNP-level mutations. These

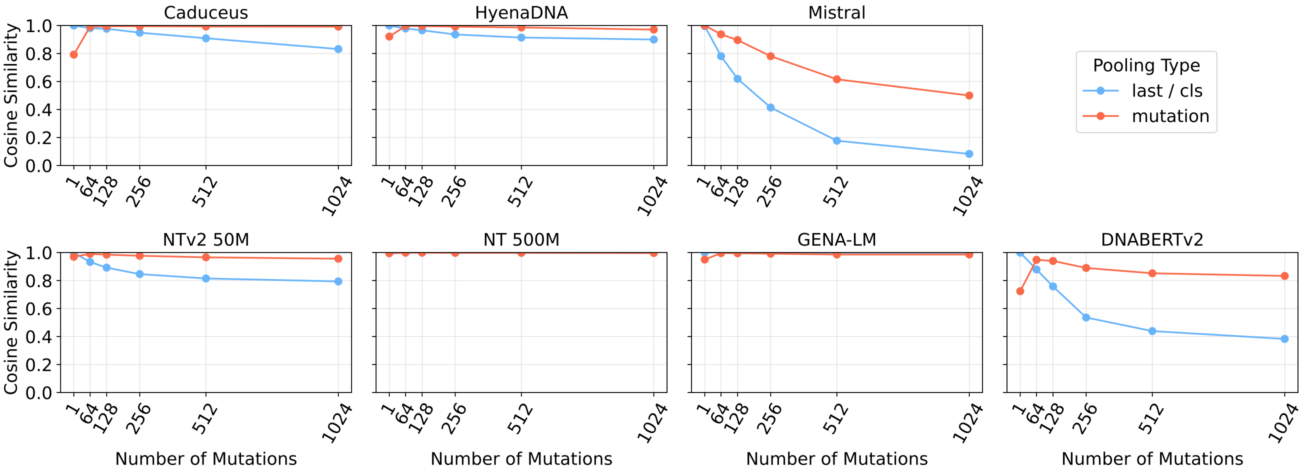

Figure 5. **Mutation sensitivity.** Although cosine similarity decreases with more mutations, the values remain high, indicating the models are mostly insensitive to the mutations. For blue markers, depending on the model type (encoder / decoder), we use last / cls tokens.

experiments aimed to evaluate the models' ability to detect differences between reference sequences and sequences with inserted SNPs. We focused exclusively on SNPs to eliminate sequence length as a confounding factor (by keeping a fixed sequence length) and assess direct attribution of embedding differences to the SNPs.

We measure the cosine similarity between the embeddings of the reference DNA sequence and the embeddings of the same sequence with SNPs introduced. Lower similarity scores indicate better model sensitivity in detecting biologically significant changes, while high scores suggest the model fails to distinguish important genetic variations.

For experimental robustness, we conducted this experiment by sampling twenty-five 1024-length sequences from chromosomes 7, 11, 12, 17, and 19 in HG38, with five sequences per chromosome. The 1024-length was chosen to avoid chunking effects, ensuring it fits within all models' context windows. For each sequence, we created variants by introducing mutations at increasing frequencies (1, 64, 128, 256, 512, 1024) at random positions. Embeddings for both the reference and mutated sequences are then generated using different methods: last / cls tokens (based on the decoder or encoder), and pooling the tokens only at mutation sites (denoted as "mutation" in Figure 5). These two approaches were tried to cover different ways of interpreting the embeddings generate by the model during downstream evaluations.

Figure 5 illustrates the cosine similarity between reference and altered sequences across different pooling types. Despite the models using different tokenizers, the results are generally poor. For most models, both last / cls and mutation pooling produced high cosine similarity values even for a single mutation regime, typically 0.9 or higher. As the number of mutations increases the cosine similarity also tends to increase, presumably due to averaging effect.

Among the models tested, the Mistral-based DNA model showed the lowest cosine similarity for last pooling and relatively low cosine similarity for mutation pooling. In contrast, GENA-LM and NT 500M produced high cosine similarity scores close to 0.999 for both pooling types. These results indicate that most models are not significantly affected by mutations, thereby highlighting their limited ability to detect subtle sequence alterations, irrespective of their tokenization strategies.

3.3.3. ClinVar Experiments and Log-Likelihood Analysis

To further investigate the sensitivity of genomic models to sequence alterations, we conducted additional experiments using ClinVar data (Landrum et al., 2014), which includes genetic variations among individuals. These experiments aim to verify our previous findings in a more realistic setting, utilizing real-world genetic variations from ClinVar. We chose to analyze the TP53, BRCA2 and CFTR genes and obtained their gene sequences from the NCBI database (Sayers et al., 2022).

First, we filtered the variants to include only exonic mutations. This ensures a focus on mutations that affect protein-coding regions, which are of greatest interest in clinical genetics. Next, we categorized the variants into two groups based on clinical significance: benign and pathogenic. The benign group included variants labeled as 'Benign', 'Likely benign', or 'Benign/Likely benign', while the pathogenic group comprised variants classified as 'Pathogenic', 'Likely pathogenic', or 'Pathogenic/Likely pathogenic'. This grouping enables us to compare the model's sensitivity to mutations with different clinical impacts.

		NT 500M	NTv2 50M	DNABERTv2	HyenaDNA	Mistral	GENA-LM	Caduceus
TP53	Benign	0.985	0.991	0.995	0.999	0.976	1.000	0.985
	Pathogenic	0.983	0.993	0.996	0.999	0.988	1.000	0.990
BRCA2	Benign	0.999	0.984	0.964	0.996	0.907	0.996	0.996
	Pathogenic	1.000	0.984	0.955	0.999	0.981	1.000	0.973
CFTR	Benign	1.000	0.998	0.998	1.000	0.999	1.000	0.999
	Pathogenic	1.000	0.999	0.998	1.000	0.996	1.000	0.999

Table 4. **Gene-specific Variant Detection Performance**. Average performance across different models for TP53, BRCA2, and CFTR genes, showing benign and pathogenic variant detection capabilities. Lower values indicate better performance in distinguishing variants.

After preprocessing the data, we take five chunks of 1024 base pairs for each gene independently that have both benign and pathogenic mutations. For each chunk, we created three versions: a reference sequence without mutations, a sequence with only pathogenic mutations, and a sequence with only benign mutations. The distribution of mutations is shown in Table 15 in Appendix.

This variation in mutation density allows us to observe the model's sensitivity across different levels of sequence alteration. For each chunk, we applied max pooling to the model outputs and computed the cosine similarity between the reference sequence and both the benign and pathogenic versions, repeating this process for each model. Finally, we averaged cosine similarity over five selected chunks. The results presented in Table 4 showed consistently high similarity scores across all models and mutation types, regardless of the number of mutations in each chunk, indicating the consistent failure of models to reflect genomic variance in their embeddings.

Log-Likelihood Ratio Analysis. We further probed the sensitivity of genomic models, by implementing an additional evaluation based on log-likelihood ratios (Benegas et al., 2025a). Here, for each pathogenic variant in BRCA2 and CFTR, we computed the log probabilities assigned by pretrained models directly at mutation sites, calculating the ratio between mutated and reference nucleotides. The log-likelihood ratio approach, being site-specific and more sensitive to single-nucleotide differences, is particularly suited for evaluating intrinsic model sensitivity without additional finetuning. We found that this different evaluation approach leads us to similar conclusion. Model performances remained near random chance (AUROC scores between 0.345–0.536; as shown in Table 16), which further validates our earlier findings regarding limited sensitivity of existing pretrained GFMs to clinically significant mutations.

> ***Finding 3:*** Current pretrained human-centric GFMs show limited effectiveness on variant-based tasks, likely because their embeddings are not sufficiently sensitive to single-nucleotide mutations.

4. Discussion

GFMs pretrained on extensive genomic datasets using substantial computational resources have generated significant enthusiasm due to their promise to serve as foundational tools capable of capturing genomic complexity. However, our comprehensive evaluation presents a notable contradiction: despite expectations, pretrained GFMs show limited or no advantage over randomly initialized models across diverse genomic tasks. This result highlights important limitations in current genomic pretraining strategies, calling for a critical reassessment of their effectiveness.

These findings carry meaningful implications, especially considering the significant computational investments involved. Pretraining typically demands weeks of processing on large-scale GPU infrastructure (Nguyen et al., 2024), with corresponding financial and environmental costs. Our results suggest that such extensive pretraining might primarily function as a form of weight initialization, offering improvements that are non-existent or marginal at best over much simpler random initializations.

Furthermore, our analyses identified a critical challenge shared by current GFMs – their limited sensitivity to genomic variations at the single-nucleotide level. Tasks like ancestry prediction, mutation sensitivity analysis, and clinical variant interpretation heavily depend on precisely detecting these subtle genomic differences. Yet, models consistently produced embeddings with very high similarity between reference and mutated sequences, even for clinically relevant variants. This limited sensitivity was particularly evident in realistic ClinVar variant analyses, where models performed near chance-level discrimination (AUROC 0.5). We further validated this limited sensitivity through additional mutation-site-specific analyses, confirming poor model discrimination even at critical mutation locations, using a log-likelihood ratio evaluation. This shortcoming restricts the applicability of current GFMs for many clinical and biomedical scenarios requiring sensitivity to such mutations.

Our analysis further identified certain technical aspects inherent to many existing genomic pretraining approaches that may contribute to these observed limitations. Specifically, high masking probabilities (15%) used in masked language modeling considerably exceed the natural genomic variation rates. Moreover, the frequent use of k-mer or subword tokenization methods can obscure single-nucleotide variant information, which partly explains many of the findings in our study where models utilizing single-nucleotide tokenization fared comparatively better. These insights point to areas for further research in future genomic modeling efforts.

Additionally, although we utilized full-parameter finetuning in our study, which is superior than parameter-efficient finetuning approaches, we briefly explored different finetuning approaches as well to rule this out as a possible explanation. As expected, our comparative experiments clearly showed that full finetuning consistently outperformed parameter-efficient approaches such as LoRA adapters (Table 7). When considered jointly with our choice to run extensive hyperparameter search for each fine-tuning task, our results reflect optimal conditions for finetuning models, further supporting the robustness of our conclusions.

Our results notably contrast with a subset of findings from earlier literature (e.g., (Dalla-Torre et al., 2024), Supp. Table 7 in their paper), where pretrained models were shown to outperform random initialization. This apparent discrepancy arises primarily from methodological differences. Specifically, our study conducted rigorous hyperparameter optimization across multiple folds (3-fold cross-validation per task) and extensive learning rate sweeps (six values per task), ensuring each model was evaluated under its optimal conditions. In contrast, (Dalla-Torre et al., 2024) et al. utilized fixed hyperparameters, which were derived from natural language processing literature, and were not validated to be broadly applicable for the entirely new domain of genomic data. As a result, our approach provides a fairer and more rigorous assessment of the pretraining utility.

With these observations, it is still important to recognize that genomic pretraining might still hold considerable value in specialized contexts, particularly for generative tasks, as suggested by recent work (Nguyen et al., 2024; Brixi et al., 2025). Moreover, specialized architectures which incorporate biological biases, have shown to achieve good performance on various variant tasks (Benegas et al., 2025a; Zhai et al., 2024). Such studies generally support our observations that the approach of pretraining often utilized in building GFMs from human DNA sequences should be reconsidered. A defining feature of foundation models like GPT (Brown et al., 2020), BERT (Devlin et al., 2019) and CLIP (Radford et al., 2021) is their proven generalizability across diverse tasks. Achieving a similar level of broad applicability and robustness remains challenging in genomics.

Finally, we emphasize that our study evaluated 52 diverse tasks spanning structural, functional, and regulatory genomics, which have been broadly used in the genomics machine learning community. We utilized statistically robust experimentation, and with the breadth and depth of the evaluation conducted, our findings are robust. However, there does remain open an opportunity to develop more rigorous benchmarks for future progress in building GFMs. Crucially, such benchmarks should not only evaluate model performance in controlled settings but also translate into meaningful improvements for real-world clinical and biomedical applications. We hope our findings encourage the community to rethink development of the GFMs.

5. Conclusion

Our comprehensive evaluation of GFMs highlights significant limitations in current pretraining approaches. Contrary to expectations, we found that the pretrained GFMs demonstrated limited or no advantage over randomly initialized counterparts across a wide array of genomic tasks. We also found that existing GFMs exhibit insufficient sensitivity to variants, which limits their utility in tasks requiring variant interpretation.

We identify areas for methodological refinement, including optimizing masking approach, employing character-level tokenization, and designing specialized architectures better attuned to biological sequence complexity. Additionally, our extensive hyperparameter optimization helped ensure the robustness of these conclusions.

While genomic pretraining still holds promise, particularly in specialized generative contexts, realizing the vision of broadly applicable, clinically relevant GFMs will require a fundamental reassessment of current practices. We hope our findings encourage the genomic modeling community to develop more biologically informed approaches, rigorous benchmarks, and targeted strategies, ultimately bridging the gap between computational advancements and tangible biomedical impacts.

Data & Code Availability

The data used in our work available in the following places:

- 1000 Genomes Project VCF files can be accessed at International Genome Sample Resource (IGSR) (Fairley et al., 2020).
- GRCh38 reference genome assembly can be downloaded from (National Center for Biotechnology Information (NCBI)).
- Gencode (Frankish et al., 2019) gene annotation used for biotype labelling.
- NT Benchmark datasets are introduced in (Dalla-Torre et al., 2024).
- Genome Understanding Evaluation (GUE) multi-species benchmark is introduced in (Zhou et al., 2024).

- Genomic Benchmarks is introduced in (Grešová et al., 2023).
- ClinVar variant records for TP53, BRCA2 and CFTR and corresponding gene sequences are downloaded from NCBI (National Center for Biotechnology Information (NCBI)).

Our code is publicly available at github.com/m42-health/gfm-random-eval.

References

Benegas, G., Albors, C., Aw, A. J., Ye, C., and Song, Y. S. A dna language model based on multispecies alignment predicts the effects of genome-wide variants. *Nature Biotechnology*, pp. 1–6, 2025a.

Benegas, G., Ye, C., Albors, C., Li, J. C., and Song, Y. S. Genomic language models: Opportunities and challenges. *Trends in Genetics*, 2025b.

Brixi, G., Durrant, M. G., Ku, J., Poli, M., Brockman, G., Chang, D., Gonzalez, G. A., King, S. H., Li, D. B., Merchant, A. T., et al. Genome modeling and design across all domains of life with evo 2. *bioRxiv*, pp. 2025–02, 2025.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *NeurIPS*, 2020.

Consens, M. E., Dufault, C., Wainberg, M., Forster, D., Karimzadeh, M., Goodarzi, H., Theis, F. J., Moses, A., and Wang, B. Transformers and genome language models. *Nature Machine Intelligence*, 2025.

Consortium, . G. P. et al. A global reference for human genetic variation. *Nature*, 2015.

Dalla-Torre, H., Gonzalez, L., Mendoza-Revilla, J., Carranza, N. L., Grzywaczewski, A. H., Oteri, F., Dallago, C., Trop, E., de Almeida, B. P., Sirelkhatim, H., et al. The nucleotide transformer: Building and evaluating robust foundation models for human genomics. *Nature Methods*, 2024.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *NAACL*, 2019.

Fairley, S., Lowy-Gallego, E., Perry, E., and Flicek, P. The international genome sample resource (igsr) collection of open human genomic variation resources. *Nucleic acids research*, 2020.

Feng, H., Wu, L., Zhao, B., Huff, C., Zhang, J., Wu, J., Lin, L., Wei, P., and Wu, C. Benchmarking dna foundation models for genomic sequence classification. *bioRxiv*, 2024. doi: 10.1101/2024.08.16.608288. URL https://www.biorxiv.org/content/early/2024/08/18/2024.08.16.608288.

Fishman, V., Kuratov, Y., Petrov, M., Shmelev, A., Shepelin, D., Chekanov, N., Kardymon, O., and Burtsev, M. Gena-lm: A family of open-source foundational models for long dna sequences. *Nucleic Acids Research,*, 2025.

Frankish, A., Diekhans, M., Ferreira, A.-M., Johnson, R., Jungreis, I., Loveland, J., Mudge, J. M., Sisu, C., Wright, J., Armstrong, J., et al. Gencode reference annotation for the human and mouse genomes. *Nucleic acids research*, 2019.

Garrido-Martín, D., Borsari, B., Calvo, M., Reverter, F., and Guigó, R. Identification and analysis of splicing quantitative trait loci across multiple tissues in the human genome. *Nature communications*, 2021.

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

Harrow, J., Frankish, A., Gonzalez, J. M., Tapanari, E., Diekhans, M., Kokocinski, F., Aken, B. L., Barrell, D., Zadissa, A., Searle, S., et al. Gencode: the reference human genome annotation for the encode project. *Genome research*, 2012.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W., et al. Lora: Low-rank adaptation of large language models. *ICLR*, 2022.

Ji, Y., Zhou, Z., Liu, H., and Davuluri, R. V. DNABERT: pre-trained Bidirectional Encoder Representations from Transformers model for DNA-language in genome. *Bioinformatics*, 2021.

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., Casas, D. d. l., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

Landrum, M. J., Lee, J. M., Riley, G. R., Jang, W., Rubinstein, W. S., Church, D. M., and Maglott, D. R. Clinvar: public archive of relationships among sequence variation and human phenotype. *Nucleic acids research*, 2014.

Li, S., Wang, Z., Liu, Z., Wu, D., Tan, C., Zheng, J., Huang, Y., and Li, S. Z. Vqdna: Unleashing the power of vector quantization for multi-species genomic sequence modeling. *ICML*, 2024.

Lindsey, L. M., Pershing, N. L., Habib, A., Stephens, W. Z., Blaschke, A. J., and Sundar, H. A comparison of tokenization impact in attention based and state space genomic language models. *bioRxiv*, 2024.

Marin, F. I., Teufel, F., Horrender, M., Madsen, D., Pultz, D., Winther, O., and Boomsma, W. Bend: Benchmarking dna language models on biologically meaningful tasks. *ICLR*, 2024.

National Center for Biotechnology Information (NCBI). Ncbi [internet]. URL https://www.ncbi.nlm.nih.gov/.

Nguyen, E., Poli, M., Faizi, M., Thomas, A. W., Wornow, M., Birch-Sykes, C., Massaroli, S., Patel, A., Rabideau, C. M., Bengio, Y., Ermon, S., Re, C., and Baccus, S. HyenaDNA: Long-range genomic sequence modeling at single nucleotide resolution, 2023.

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

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

Sanabria, M., Hirsch, J., Joubert, P. M., and Poetsch, A. R. Dna language model grover learns sequence context in the human genome. *Nature Machine Intelligence*, 2024.

Sayers, E. W., Bolton, E. E., Brister, J. R., Canese, K., Chan, J., Comeau, D. C., Connor, R., Funk, K., Kelly, C., Kim, S., et al. Database resources of the national center for biotechnology information. *Nucleic acids research*, 2022.

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

Tang, Z., Somia, N., Yu, Y., and Koo, P. K. Evaluating the representational power of pre-trained dna language models for regulatory genomics. *bioRxiv*, 2024.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, 2020.

Zhai, J., Gokaslan, A., Schiff, Y., Berthel, A., Liu, Z.-Y., Lai, W.-Y., Miller, Z. R., Scheben, A., Stitzer, M. C., Romay, M. C., et al. Cross-species modeling of plant genomes at single nucleotide resolution using a pre-trained dna language model. *bioRxiv*, 2024.

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

Zhang, Y.-z., Bai, Z., and Imoto, S. Investigation of the bert model on nucleotide sequences with non-standard pre-training and evaluation of different k-mer embeddings. *Bioinformatics*, 2023.

Zhou, J. and Troyanskaya, O. G. Predicting effects of noncoding variants with deep learning–based sequence model. *Nature methods*, 2015.

Zhou, Z., Ji, Y., Li, W., Dutta, P., Davuluri, R. V., and Liu, H. DNABERT-2: Efficient foundation model and benchmark for multi-species genomes. *ICLR*, 2024.

A. Appendix

A.1. Related Works

Genomic Foundation Models. Encoder-only approaches have proven effective in sequence prediction tasks, using k-mer tokenization (Ji et al., 2021; Dalla-Torre et al., 2024), Byte Pair Encoding (Zhou et al., 2024; Sanabria et al., 2024), and learnable vector quantization codebooks (Li et al., 2024) to enhance efficiency and manage longer sequences. Certain encoder architectures have been enhanced with recurrent memory mechanisms (Fishman et al., 2025) to capture long-range dependencies more effectively, while others utilize whole-genome alignments (Benegas et al., 2025a) to incorporate evolutionary context. More recent work has explored pan-genome graph representations (Zhang et al., 2024) to better capture genetic variation diversity.

Meanwhile, decoder-only architectures have shown potential by integrating structured state-space models (Nguyen et al., 2023; Schiff et al., 2024), achieving competitive performance with minimal parameters and supporting long context lengths. Hybrid architectures (Nguyen et al., 2024; Brixi et al., 2025), incorporating both attention and state-space blocks, have emerged, demonstrating great generative capabilities spanning from molecular to genome scales. Our work introduces a GFM based on Mistral architecture (Jiang et al., 2023) and performs performance analysis of the most recent GFMs.

Genomic Foundation Models Analysis. It was shown that k-mer embeddings pretrained on random DNA sequences can reach similar performance to those of trained on the real-world biological data (Zhang et al., 2023). Another study found that character tokenization outperforms other methods in state-space models (Lindsey et al., 2024). Evaluation of GFMs across the BEND benchmark reveals that they capture limited information on long-range features (Marin et al., 2024). It was also shown that mean pooling improves performance of GFMs for genomic sequence classifications and closes the performance gap between them (Feng et al., 2024). Pretrained DNA models were benchmarked (Tang et al., 2024) showing they do not offer great advantage over conventional machine learning methods. In contrast to this study, our analysis includes finetuning and variant-based tasks, more models and also shows that randomly initialized models can be better as feature extractors.

A.2. Models

We use the following GFMs in our analysis:

- **HyenaDNA** (Nguyen et al., 2023): Decoder-only state-space model with 450K parameters. Uses character tokenizer and was pretrained on the Human Reference Genome with a 1024 base pair sequence length.
- **Caduceus** (Schiff et al., 2024): Decoder-only model with 8M parameters. Trained on sequences of 131k base pairs on HRG. Combines a bidirectionally equivariant decoder with character tokenizer.
- **Mistral** (our version): Decoder-only transformer model with 580M parameters. Uses character tokenization and was trained on the 1000 Genomes dataset (Consortium et al., 2015). Details for pretraining are provided in Section A.4 in Appendix.
- **Nucleotide Transformer** (Dalla-Torre et al., 2024): Encoder-only model presented in two versions: a 500M parameter model trained on the 1000 Genomes Project data and its v2 with 50M parameter model trained on multispecies data. Both use k-mer tokenization.
- **GENA-LM** (Fishman et al., 2025): Encoder-only model with 110M parameters. Employs BPE tokenizer and was pretrained on the HRG with 1000G augmentations.
- **DNABERTv2** (Zhou et al., 2024): Encoder-only model with 117M parameters. Uses BPE tokenization and was trained on multispecies data.

A.3. Random Weight Initialization

We initialized the model weights following a procedure using standard Hugging Face Transformers library (Wolf et al., 2020) initialization methods:

- **For Linear Layers:** Weights were initialized from a normal distribution $\mathcal{N}(0, 0.02)$, biases were initialized to zero.
- **For LayerNorm:** The scaling factor (gamma) was initialized to 1. The bias term (beta) was initialized to 0.
- **For Embedding Layers:** Embeddings were initialized from the same normal distribution $\mathcal{N}(0, 0.02)$.

For Caduceus and HyenaDNA we performed prenorm residual rescaling, which is the default weight initialization procedure for these models. Biases for linear layers were initialized as zeros.

A.4. Mistral Pretraining

We pretrain a Mistral model on 50 random individual samples from the Genome1000 project. Table 5 provides the Mistral configuration details and Table 6 provides the Mistral training configuration. Specifically, reverse complement of sequences formed with Genome1000 VCFs is used with a probability of 0.5. All the chromosomes (chr1 - chrX) are used for sequence formation from 50 individuals. Individuals are sampled in a stratified way, 10 from each superpopulation. We filtered out sequences where number of unknown nucleotides was more than half of sequence length. Total number of tokens is 150B.

A.5. Finetuning Experiments

We use the following datasets for finetuning, more details about them can be found the corresponding original papers:

- **NT Benchmark** (Dalla-Torre et al., 2024) consists of the following group of tasks histones, enhancers, promoters and splice sites.
- **Genomic Benchmarks** (Grešová et al., 2023) contains several datasets focused on regulatory element classification tasks across three organisms: human, mouse, and roundworm.
- **Genome Understanding Evaluation (GUE)** (Zhou et al., 2024) is a comprehensive multi-species benchmark containing 28 datasets across 7 genomic analysis tasks including promoter detection, transcription factor prediction, splice site detection, etc. with sequence lengths ranging from 70 to 1000 base pairs.

We finetune random and pretrained initializations of the chosen model using the configuration provided in Table 8. In our preliminary experiments, we found that max pooling performed better than cls / last pooling for randomly initialized models while maintaining performance for pretrained, so we used max pooling consistently across all experiments.

Additionally, we perform an ablation for full finetuning vs LoRA (Hu et al., 2022) finetuning and present the result in Table 7. In all the cases, full finetuning outperforms LoRA, suggesting that our full finetuning method gives the best chance for the models (both pretrained and random) to achieve their best scores.

config	value
num_hidden_layers	16
num_attention_heads	16
hidden_size	1408
vocab_size	12
intermediate_size	7168

Table 5. **Mistral model architecture.**

config	value
tokenizer	character
sequence_len	4096
num_epochs	1
initial_lr	7.2e-4
final_lr	4.2e-5
optimizer_momentum	$\beta_1, \beta_2 = 0.9, 0.95$
lr schedule	cosine with warmup
batch_size	64
num_genomes	50

Table 6. **Mistral training configuration.**

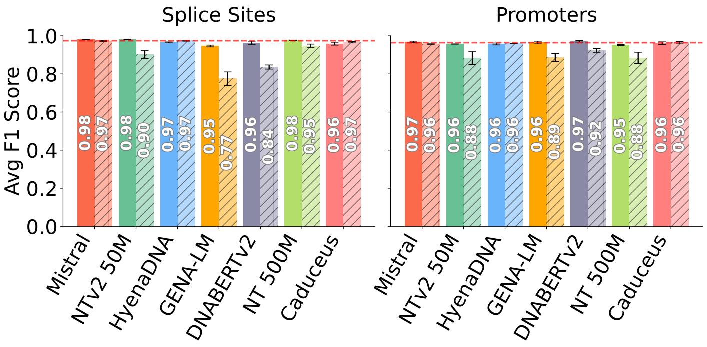

Figure 6. **NT Benchmark performance per subgroup for splice sites and promoter tasks.** Randomly initialized models are competitive with pretrained.

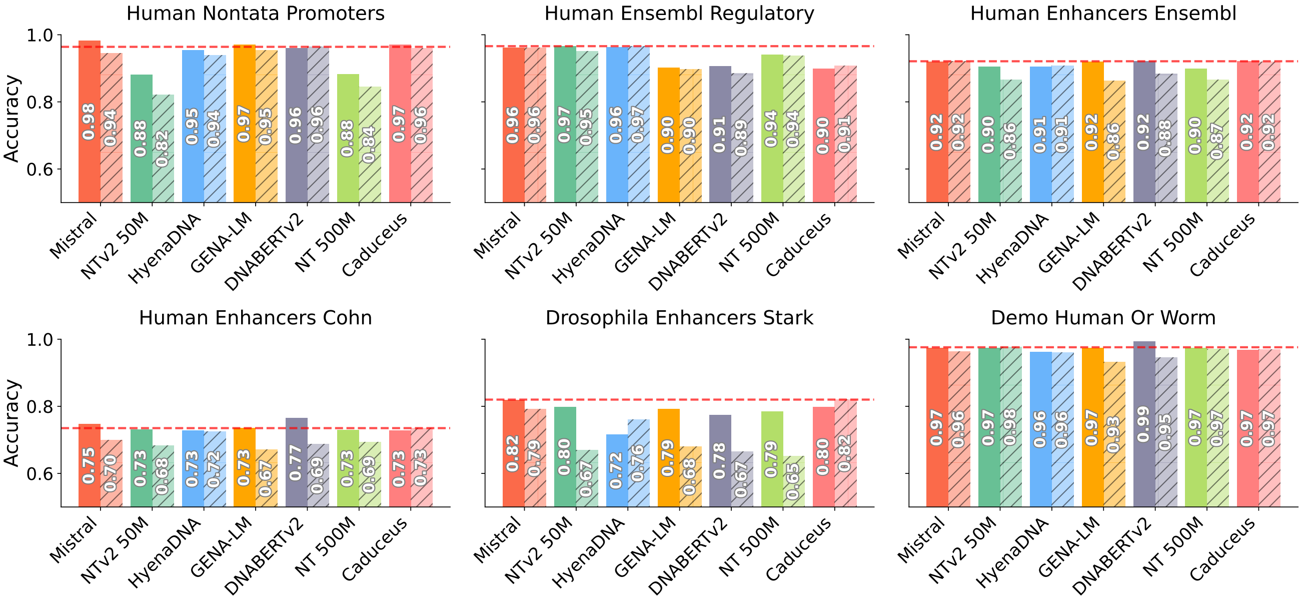

Figure 8. **Genomic Benchmarks performance for each model.**

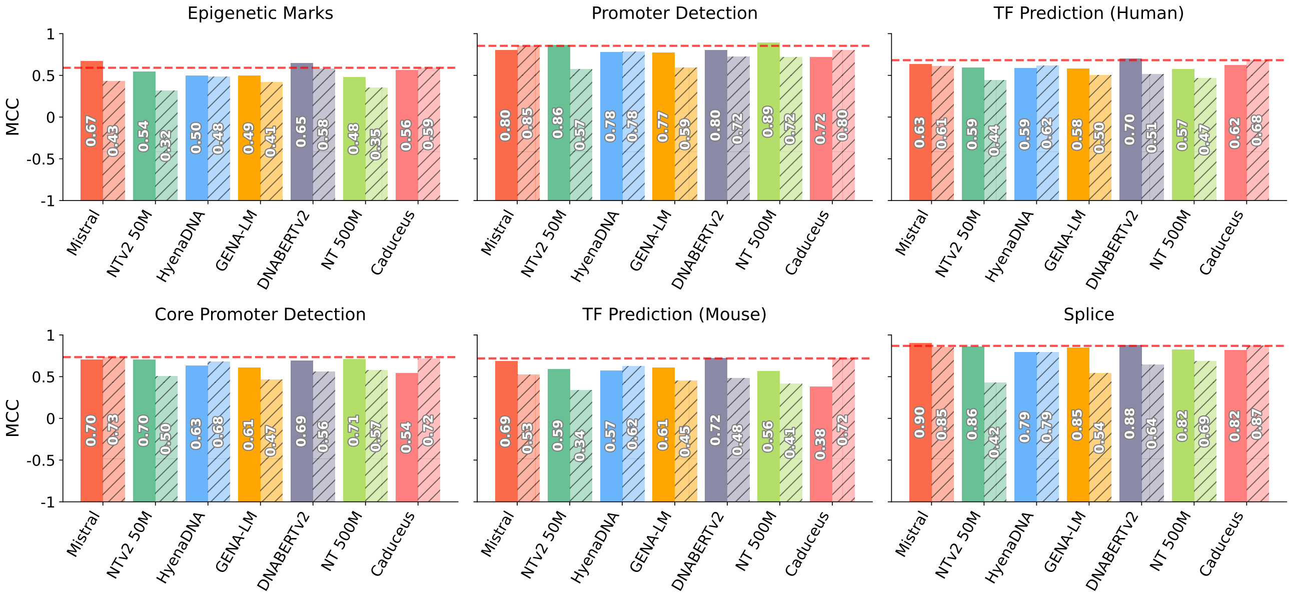

Figure 7. **GUE performance for each model.** The best randomly initialized model is consistently competitive with pretrained models.

Model	Method	enhancers	H3K4me1	promoter_all	splice_sites_all
GENA-LM	LoRA	0.449	0.260	0.908	0.501
	Full	**0.560**	**0.466**	**0.966**	**0.935**
Mistral	LoRA	0.470	0.267	0.906	0.525
	Full	**0.550**	**0.557**	**0.965**	**0.980**
NTv2 50M	LoRA	0.405	0.267	0.852	0.458
	Full	**0.551**	**0.524**	**0.957**	**0.979**

Table 7. **LoRA *vs.* full finetuning** on four representative NT tasks. The metric is MCC for enhancers and histone, and F1 for promoters and splice sites. Across all cases full finetuning performs better than LoRA.

config	value
optimizer	AdamW
learning_rate	1e-5, 3e-5, 5e-5, 8e-5, 1e-4, 3e-4
weight_decay	0
optimizer_momentum	$\beta_1, \beta_2 = 0.9, 0.999$
batch_size	32
lr schedule	cosine
epochs	20 / 100

Table 8. **Hyperparameters for finetuning experiments.** For GUE we finetune for 20 epochs for NT Benchmark and Genomic Benchmarks we use 100 epochs.

A.6. Biotype Classification

Biotype task is a sequence classification task into nine different labels. Our dataset consists of a total of 19605 sequences. The detailed statistics for sequences belonging to each gene type is provided in Table 9. For the supervised training step, we perform a train-test split of 80% : 20% using stratification by class label. We use XGBoost with the hyperparameters provided in Table Table 11. All metrics are reported on the test set.

In addition, we also perform similar feature extraction experiments on the subset of GUE benchmark displayed in Table 12. Randomly initialized HyenaDNA with large embedding size outperforms all pretrained models.

Gene Type	Count	Avg Length	Max Length	Min Length
TEC	1056	1613.26	18662	87
lncRNA	3000	32359.59	957949	87
miRNA	1879	81.89	180	41
misc RNA	2212	206.49	464	57
processed pseudogene	3000	798.02	12016	28
protein coding	3000	69971.51	2059620	159
snRNA	1901	110.46	328	50
snoRNA	943	118.86	791	55
unprocessed pseudogene	2614	5025.27	233909	28

Table 9. **Statistics of biotype genes.**

Model	Embedding Dimensions
HyenaDNA	64, 128, 256, 512, 1024, 2048, 4096
Caduceus	64, 128, 256, 512, 1024, 2048
NTv2 50M	64, 128, 256, 512, 1024, 2048, 4096
DNABERTv2	48, 96, 192, 384, 768, 1152, 1536, 2304, 3072
GENA-LM	48, 96, 192, 384, 768, 1152, 1536, 2304, 3072
NT 500M	100, 320, 640, 1280, 2560, 3840
Mistral	64, 128, 256, 512, 1408, 2048, 4096

Table 10. **Embedding dimensions for biotype experiments.**

config	value
objective	multi:softmax
num_classes	9
max_depth	3
learning_rate	0.1
n_estimators	1000
eval_metric	mlogloss
tree_method	hist

Table 11. **Biotype XGBoost configuration.**

		Pretrained					
Task	**HyenaDNA Random ED 2048**	**Mistral**	**HyenaDNA**	**NTv2 50M**	**GENA-LM**	**DNABERTv2**	**NT 500M**
H3	**0.650**	0.626	0.510	0.502	0.546	0.566	0.557
H3K14ac	0.275	0.227	0.190	0.272	0.208	**0.338**	0.220
H3K36me3	**0.408**	0.267	0.252	0.330	0.321	0.397	0.308
H3K4me1	**0.320**	0.224	0.211	0.275	0.244	0.295	0.267
H3K4me2	**0.265**	0.243	0.186	0.176	0.218	0.185	0.245
H3K4me3	**0.207**	0.126	0.105	0.147	0.113	0.189	0.121
H3K79me3	**0.522**	0.428	0.367	0.463	0.437	0.520	0.406
H3K9ac	**0.429**	0.373	0.288	0.273	0.318	0.343	0.343
H4	**0.671**	0.649	0.491	0.575	0.577	0.658	0.612
H4AC	**0.282**	0.227	0.202	0.227	0.200	0.259	0.225
Average	**0.403**	0.339	0.280	0.324	0.318	0.375	0.330

Table 12. **Feature Extraction on Histone Tasks from GUE.** Embeddings extracted from pretrained and randomly initialized models were used to train an XGBoost classifier. Randomly initialized HyenaDNA with $embed\_dim$ 2048 outperforms every pretrained model on every task except H314ac. MCC on test set is reported.

A.7. Ancestry Benchmark

Each task is the sequence classification task with five labels, South Asian, European, African, American, East Asian. Each label is a superpopulation from 1000 Genomes dataset. We selected eleven different regions on chromosome with the length of 32K nucleotides, where each region corresponds to a different variant. The start indices with respect to the human reference genome used for sequence construction is provided in Table 13. Each task has 3202 samples.

Training involves two stages: embedding generation from the model of interest and supervised training on the embeddings with XGBoost. During the embedding generation step, sequence embeddings are constructed similarly to the biotype classification task. For the supervised training step, we split the dataset into train, validation and test set with sizes 72%, 8%, and 20% respectively. We use XGBoost with hyperparameters mentioned in Table 14. All metrics are reported on the test set and averaged over eleven tasks across each chromosome.

chromosome	**start position**
chr1	119478211
chr3	2015011
chr5	85769129
chr7	74672986
chr9	75197358
chr11	62543311
chr13	52182164
chr15	45995594
chr17	36628720
chr19	24308808
chr21	18354991

Table 13. **Sample chromosome positions.**

config	**value**
objective	multi:softmax
num_class	5
max_depth	3
learning_rate	0.1
n_estimators	1000
colsample_bytree	0.5
eval_metric	mlogloss
tree_method	hist
early_stopping rounds	100

Table 14. **Ancestry XGBoost configuration.**

A.8. ClinVar Experiments

Each chunk used for ClinVar experiments consists of benign and pathogenic mutations. Three types of sequences are formed: reference sequence, sequence with benign mutations, and sequence with pathogenic mutations. The distribution of mutations in these chunks for all three genes is presented in Table 15.

Chunk Index	TP53		BRCA2		CFTR	
	Benign	Pathogenic	Benign	Pathogenic	Benign	Pathogenic
1	122	27	138	46	32	18
2	60	61	268	74	19	11
3	51	50	187	57	32	30
4	76	42	35	18	9	13
5	38	10	37	6	7	11

Table 15. **Mutation Data Distribution by Gene and Chunk.** Distribution of benign and pathogenic mutations across different chunks for TP53, BRCA2, and CFTR genes.

A.9. Log-Likelihood Ratio Analysis

We follow the approach for log-likelihood ratio analysis presented in (Benegas et al., 2025a). For a single-nucleotide variant defined by a reference base REF and an alternative base ALT at position i, we compute $\log \frac{P(\mathrm{ALT})}{P(\mathrm{REF})}$. After that, using the labeled data from ClinVar for pathogenic and benign variants, we compute the AUROC scores from the log-likelihood ratio values. As shown in Table 16, the AUROC scores range from 0.345 to 0.511, which suggests that models cannot reliably separate benign from pathogenic variants.

Gene	NT 500M	NTv2 50M	HyenaDNA	Mistral	Caduceus	GENA-LM	DNABERTv2
BRCA2	0.511	0.478	0.439	0.495	0.505	0.408	0.535
CFTR	0.442	0.365	0.454	0.345	0.442	0.421	0.536

Table 16. AUROC values calculated from log-likelihood ratio scores on ClinVar variants.

A.10. Model checkpoints

Checkpoints for all the pretrained models were obtained from Hugging Face. Table 17 provides detailed checkpoint IDs which can be loaded using the transformers library.

Model	Checkpoint
NTv2 50M	InstaDeepAI/nucleotide-transformer-v2-50m-multi-species
NT 500M	InstaDeepAI/nucleotide-transformer-500m-1000g
Caduceus	kuleshov-group/caduceus-ps_seqlen-131k_d_model-256_n_layer-16
HyenaDNA	LongSafari/hyenadna-tiny-1k-seqlen-hf
DNABERTv2	zhihan1996/DNABERT-2-117M
GENA-LM	AIRI-Institute/gena-lm-bert-base-t2t

Table 17. **Checkpoints used for pretrained models.**

Dataset	Metric	Pretrained							Random						
		Mistral	NTv2 50M	HyenaDNA	GENA-LM	DNABERTv2	NT 500M	Caduceus	Mistral	NTv2 50M	HyenaDNA	GENA-LM	DNABERTv2	NT 500M	Caduceus
H4ac	MCC	0.702 ± 0.034	0.450 ± 0.001	0.476 ± 0.015	0.526 ± 0.006	0.652 ± 0.009	0.362 ± 0.003	0.610 ± 0.004	0.395 ± 0.032	0.254 ± 0.015	0.427 ± 0.008	0.382 ± 0.002	0.622 ± 0.006	0.245 ± 0.009	0.606 ± 0.007
H4	MCC	0.802 ± 0.011	0.792 ± 0.000	0.783 ± 0.005	0.773 ± 0.013	0.809 ± 0.004	0.762 ± 0.001	0.778 ± 0.005	0.760 ± 0.012	0.593 ± 0.008	0.787 ± 0.003	0.619 ± 0.012	0.669 ± 0.006	0.629 ± 0.012	0.795 ± 0.007
H3K9ac	MCC	0.664 ± 0.004	0.540 ± 0.009	0.541 ± 0.004	0.551 ± 0.008	0.654 ± 0.004	0.473 ± 0.005	0.618 ± 0.005	0.470 ± 0.005	0.345 ± 0.007	0.532 ± 0.012	0.435 ± 0.018	0.593 ± 0.002	0.366 ± 0.016	0.615 ± 0.020
H3K79me3	MCC	0.753 ± 0.005	0.596 ± 0.005	0.616 ± 0.005	0.633 ± 0.006	0.725 ± 0.006	0.574 ± 0.005	0.688 ± 0.020	0.666 ± 0.008	0.435 ± 0.010	0.576 ± 0.007	0.505 ± 0.001	0.670 ± 0.002	0.445 ± 0.015	0.685 ± 0.005
H3K4me3	MCC	0.662 ± 0.016	0.340 ± 0.017	0.427 ± 0.004	0.487 ± 0.018	0.607 ± 0.002	0.259 ± 0.016	0.565 ± 0.008	0.444 ± 0.018	0.168 ± 0.004	0.354 ± 0.009	0.223 ± 0.129	0.606 ± 0.014	0.162 ± 0.003	0.555 ± 0.002
H3K4me2	MCC	0.574 ± 0.024	0.303 ± 0.004	0.367 ± 0.012	0.445 ± 0.018	0.564 ± 0.005	0.289 ± 0.012	0.463 ± 0.007	0.270 ± 0.001	0.207 ± 0.009	0.302 ± 0.006	0.316 ± 0.010	0.511 ± 0.000	0.220 ± 0.004	0.493 ± 0.002
H3K4me1	MCC	0.603 ± 0.007	0.518 ± 0.002	0.441 ± 0.003	0.468 ± 0.005	0.585 ± 0.002	0.397 ± 0.013	0.493 ± 0.015	0.442 ± 0.027	0.260 ± 0.009	0.421 ± 0.004	0.334 ± 0.006	0.499 ± 0.007	0.254 ± 0.008	0.534 ± 0.003
H3K36me3	MCC	0.725 ± 0.007	0.581 ± 0.003	0.538 ± 0.008	0.555 ± 0.011	0.676 ± 0.011	0.469 ± 0.008	0.613 ± 0.022	0.573 ± 0.014	0.371 ± 0.018	0.495 ± 0.004	0.416 ± 0.008	0.611 ± 0.016	0.335 ± 0.011	0.625 ± 0.013
H3K14ac	MCC	0.724 ± 0.011	0.516 ± 0.013	0.516 ± 0.007	0.587 ± 0.009	0.659 ± 0.006	0.396 ± 0.024	0.615 ± 0.008	0.578 ± 0.013	0.281 ± 0.009	0.462 ± 0.007	0.403 ± 0.016	0.641 ± 0.003	0.259 ± 0.005	0.637 ± 0.011
H3	MCC	0.809 ± 0.003	0.769 ± 0.006	0.778 ± 0.007	0.776 ± 0.005	0.826 ± 0.012	0.741 ± 0.013	0.802 ± 0.012	0.679 ± 0.012	0.573 ± 0.008	0.770 ± 0.005	0.656 ± 0.001	0.742 ± 0.009	0.597 ± 0.008	0.795 ± 0.002
enhancers_types	MCC	0.528 ± 0.006	0.456 ± 0.008	0.424 ± 0.028	0.477 ± 0.012	0.469 ± 0.002	0.433 ± 0.023	0.487 ± 0.018	0.441 ± 0.028	0.438 ± 0.025	0.412 ± 0.029	0.474 ± 0.013	0.488 ± 0.019	0.454 ± 0.008	0.451 ± 0.011
enhancers	MCC	0.557 ± 0.003	0.576 ± 0.010	0.552 ± 0.009	0.569 ± 0.010	0.536 ± 0.009	0.557 ± 0.027	0.527 ± 0.012	0.561 ± 0.004	0.523 ± 0.016	0.556 ± 0.022	0.545 ± 0.015	0.529 ± 0.002	0.540 ± 0.011	0.523 ± 0.024
splice_sites_all	F1 Score	0.980 ± 0.000	0.980 ± 0.000	0.962 ± 0.006	0.941 ± 0.002	0.950 ± 0.001	0.977 ± 0.002	0.959 ± 0.003	0.977 ± 0.000	0.908 ± 0.015	0.975 ± 0.001	0.726 ± 0.003	0.835 ± 0.005	0.951 ± 0.003	0.967 ± 0.003
splice_sites_donors	F1 Score	0.979 ± 0.001	0.981 ± 0.002	0.966 ± 0.003	0.945 ± 0.003	0.964 ± 0.002	0.977 ± 0.001	0.949 ± 0.009	0.971 ± 0.001	0.925 ± 0.006	0.971 ± 0.001	0.811 ± 0.008	0.823 ± 0.003	0.956 ± 0.003	0.962 ± 0.004
splice_sites_acceptors	F1 Score	0.981 ± 0.001	0.983 ± 0.001	0.968 ± 0.000	0.953 ± 0.002	0.973 ± 0.002	0.975 ± 0.002	0.968 ± 0.003	0.972 ± 0.002	0.874 ± 0.017	0.976 ± 0.003	0.787 ± 0.000	0.850 ± 0.014	0.934 ± 0.000	0.970 ± 0.001
promoter_tata	F1 Score	0.962 ± 0.007	0.956 ± 0.003	0.950 ± 0.002	0.955 ± 0.003	0.965 ± 0.004	0.947 ± 0.000	0.951 ± 0.007	0.958 ± 0.004	0.836 ± 0.009	0.960 ± 0.004	0.856 ± 0.012	0.909 ± 0.003	0.843 ± 0.002	0.955 ± 0.007
promoter_no_tata	F1 Score	0.970 ± 0.003	0.958 ± 0.002	0.961 ± 0.000	0.970 ± 0.002	0.975 ± 0.000	0.954 ± 0.001	0.967 ± 0.002	0.957 ± 0.001	0.906 ± 0.002	0.958 ± 0.001	0.899 ± 0.003	0.929 ± 0.001	0.903 ± 0.002	0.969 ± 0.001
promoter_all	F1 Score	0.971 ± 0.002	0.957 ± 0.002	0.960 ± 0.001	0.969 ± 0.001	0.973 ± 0.001	0.953 ± 0.000	0.965 ± 0.000	0.955 ± 0.001	0.907 ± 0.003	0.958 ± 0.001	0.903 ± 0.003	0.932 ± 0.001	0.908 ± 0.003	0.968 ± 0.000

Table 18. **Pretrained and randomly initialized models performance on NT Benchmark.**