POLITECNICO DI TORINO

Master's Degree in Data Science and Engineering

Master's Degree Thesis

Optimizing Genome Representations for Cancer Type Classification

Supervisors

Prof. Alfredo BENSO

Prof. Joana P. GONÇALVES

Dr. Colm SEALE

Candidate

Flavio SPURI

July 2024

Abstract

In recent years Large Language Models (LLMs) have been successfully adapted to the field of Genomics, as shown by state-of-the-art models like DNABERT, DNABERT-2, and Nucleotide Transformer. Despite this, their application in the challenging field of Cancer Genomics remains unexplored.

This thesis examines whether cancer genome analysis can benefit from large, pre-trained Transformer-based models, specifically focusing on the newly introduced HyenaDNA architecture. In HyenaDNA, traditional Attention Layers are replaced by so-called Hyena Filters, which consist of recursions of an element-wise multiplicative gating and a long convolution, allowing for the processing of longer sequences while maintaining single-base resolution, and achieving a subquadratic computational cost, aligning well with the specific needs of Cancer Genomics.

This study begins by assessing HyenaDNA's capabilities to represent genomic data. Using UMAP, we visualized embeddings computed by a HyenaDNA model pre-trained on the Human Reference Genome, showing that the model effectively learns to distinguish between different genomic regions.

Following this, we fine-tuned pre-trained HyenaDNA models on the novel task of classifying cancer types directly from the mutated sequences. Given the lack of existing datasets tailored to our specific application, we built a custom dataset combining data from the DepMap dataset and the Human Reference Genome. Comparing performances obtained in various experimental settings, we conclude that this approach shows potential to be effectively employed in Cancer Genomics.

Acknowledgements

I would like to thank everyone who supported me during the writing of this thesis and throughout this degree.

Firstly, I owe a special thanks to Professor Benso for enabling me to pursue this thesis abroad.

I extend my heartfelt gratitude to Professor Joana for welcoming me and providing invaluable guidance and support. Her insights and expertise have been instrumental in shaping this work. I am also immensely grateful to Colm, who followed my progress closely and offered continuous assistance and encouragement.

I would like to thank all the members of Joana's Lab at TU Delft for welcoming me warmly and creating a collaborative and stimulating environment during my stay.

My appreciation also goes to all the professors in my course for their teaching and support throughout my academic journey. Each of you has contributed to my growth and understanding in the field.

I would also like to acknowledge my colleagues and friends from Politecnico di Torino. Special thanks to Matteo B., Aurora, Fabio, Matteo D., Giacomo, and Leo for being exceptional partners in various projects. I am particularly grateful to Luca A., who has almost always been with me from the first year, working on numerous projects together, and to Alessio and Luca Z., who not only collaborated with me on projects and were my study companions but also supported me during the long writing of this thesis.

I am deeply thankful to my parents, Luigina and Emilio, for their unwavering support, even as I moved to different cities, and for allowing me to devote myself to my studies without any worries. I thank them, along with the rest of my family, for their constant love and belief in me.

Grazie ai miei nonni, Enzo e Franca, per il vostro amore incondizionato e per essere una continua fonte di ispirazione per me.

A very special thank you goes to my brother, who has always supported and helped me, even in the writing of this thesis. I have always looked up to him, and

I thank him for always being there for me in the difficult and confusing times.

I extend my gratitude to all of my friends from Rome, and Anna, my roommate throughout these years. Your friendship and support have been invaluable, and your company has given me a lot of encouragement.

My heartfelt thanks go to the friends I made here in the Netherlands for making my stay enjoyable and memorable.

Lastly, I want to express my deepest gratitude to my girlfriend, Jieyi, for her love, patience, and unwavering support throughout this journey. From almost the very beginning of this long thesis work, she has been by my side, supporting me when things got hard and cheering me up. We have made many wonderful memories together, and I look forward to creating many more. 谢谢你，我爱你。

.

Table of Contents

List of Tables

List of Figures

Acronyms

ML
Machine Learning

AI
Artificial Intelligence

DL
Deep Learning

NLP
Natural Language Processing

FFNN
Feed-Forward Neural Network

LLM
Large Language Model

MLM
Masked Language Modeling

NSP
Next Sentence Prediction

HRG
Human Reference Genome

bp
Base pairs

BPE

Byte Pair Encoding

Chapter 1

Introduction

1.1 Background

Decoding the intricacies of DNA has been a primary focus of genomic research for decades. Despite significant advancements, much remains to be uncovered about the intricate workings of our genetic code. Recently, the ability to effectively represent genomic sequences has provided powerful tools for analyzing and interpreting complex biological data, revolutionizing genomics and offering new insights into genetic information.

Central to these advancements are foundation models, which are Machine Learning (ML) models trained on broad sets of general-purpose data. These models have profoundly transformed Machine Learning (ML) and Artificial Intelligence (AI), impacting and revolutionizing numerous sectors.

Foundation models are generally based on the Transformer architecture, introduced in 2017 with the seminal paper "Attention Is All You Need". The training of these models follows a two-stage process:

- Pre-training - During pre-training, models are trained on large, general, unsupervised datasets to build a broad, intrinsic representation of the application field.

- Fine-tuning - During Fine-tuning, models are trained on smaller, supervised datasets.

The great success of Transformers across various applications is indeed due to their capability to create deep, contextualized representations of sequential data in an unsupervised manner, providing a robust foundation for efficient downstream task modeling.

One of the fields that has most benefitted from the application of Transformer-based foundation models is Natural Language Processing (NLP). When applied in the context of NLP, these models have enabled the development of the family of models known as Large Language Models (LLMs), which have been successfully applied to a wide range of tasks, including language translation, sentiment analysis, text summarization, and question answering. The ability of LLMs to capture the nuances and complexities of human language has set new benchmarks in the field of NLP.

Another field that greatly benefited from the use of foundation models is genomics, as genomic data is inherently sequential, particularly when working with raw genetic sequences.

The parallels between natural and genomic languages have been extensively studied. Genomic sequences can be likened to sentences, with nucleotide bases acting as words. These sequences are read in order, and follow grammatical rules similarly to how human languages do. Still, it is important to note some differences between the two fields. Genomic sequences are typically longer and lack the semi-structured organization found in texts, such as sentences and paragraphs. Additionally, the genetic "vocabulary" is limited to just four elements: the nucleotide bases A, C, G, and T.

Leveraging these similarities, NLP architectures have been successfully adapted for use in genomics. Notable examples include DNABERT and DNABERT-2, which are adaptations of the widely used BERT model. These models have achieved state-of-the-art results in numerous genomic tasks, such as predicting promoter regions, identifying binding sites, and recognizing splice sites. By learning deep representations of genetic data, these models can capture intricate patterns and relationships within DNA sequences, accelerating discoveries and enhancing our understanding of genomics.

1.2 Motivation

The adaptation of foundation models, often Large Language Models (LLMs), for genomics, underscores their versatility and power. These models not only handle the complexities of genomic data but also push the boundaries of what is possible in genomic research. The ability to analyze vast amounts of genetic data with high accuracy and efficiency opens new avenues for scientific discovery and medical advancements.

Despite the success of these technologies in general genomics, their application in cancer genomics remains largely unexplored.

This specific domain presents many unique challenges: mutations appear sparsely in the genome, thus necessitating long contexts to analyze long-range dependencies

between neighboring mutations and their surroundings. Moreover, mutations act at the nucleotide level, requiring high resolutions to accurately capture them.

However, we believe that the foundation models' ability to build an intrinsic understanding of long-range dependencies and patterns can be valuable in the analysis of cancer genomes.

Despite the domain-specific challenges, we aim to make a first step in applying Transformer architectures directly to cancer genome sequences. To the best of our knowledge, this represents the first attempt to apply large pre-trained Transformer-based models directly to raw mutated genomic sequences.

1.3 Objectives

The main research question behind this thesis is whether applying foundation models directly to mutated sequences, rather than just considering mutational signatures (i.e., the specific list of mutations), can be a promising approach.

To address and overcome the domain-specific challenges mentioned before, this work will employ the novel HyenaDNA architecture, which was recently introduced in November 2023.

In HyenaDNA, the traditional attention layers are replaced with Hyena operators, which consist of recursions of an element-wise multiplicative gating and a long convolution, which offers two main advantages:

- The ability to handle very long contexts while maintaining single-base resolution, allowing the model to consider larger context windows while focusing on the single nucleotide bases, which is crucial to analyze mutations.
- The reduction in computational complexity from $\mathcal{O}(L^2)$, inherent in the attention mechanism, to $\mathcal{O}(LlogL)$, making it more feasible to train on larger contexts even with limited resources.

The objectives and research questions for this Thesis can be schematized as follows:

1. To analyze how representation learning has benefited the field of genomics, visualizing the embedding created for genetic sequences by a model pre-trained on genomic data.
2. To create a specialized Dataset for a cancer-type classification task on genomic sequences, necessary as no previous attempt has been made in this domain.
3. To fine-tune various models on our novel task, aiming to determine whether the intrinsic representations learned by the models can also benefit cancer-specific tasks.

1.4 Thesis Structure

The structure of the remaining of the thesis is the following:

- Chapter 2: **Literature Review** – This chapter provides a comprehensive review of the relevant literature. It begins with an overview of the Transformer model and its most popular applications in NLP, specifically focusing on those that have later been adapted to genomics. The chapter then covers the core concepts from genomics required in this thesis. Following this, it discusses the application of Transformer-based foundation models in genomics. Finally, it offers a detailed description of the HyenaDNA model.
- Chapter 3: **Methodology** – This chapter outlines the creation of the ad-hoc dataset required for the novel task introduced in this thesis, along with the original datasets from which the data was sourced. It then describes the methodological details of the experiments, explaining how HyenaDNA and the other employed techniques were utilized and implemented.
- Chapter 4: **Results and Discussion** – This chapter presents the results of the experiments and discusses their implications. It then describes the limitations of this study and outlines potential directions for future research based on the findings.

Chapter 2

Literature Review

This chapter provides a comprehensive review of the relevant literature essential for understanding the context and framework of this thesis.

It begins with an overview of the Transformer model, highlighting its most popular applications in NLP and specifically focusing on those implementations that have been successfully applied to genomics.

Following this, the chapter covers the core concepts from genomics that are needed for this research.

It then describes the application of Transformer-based foundation models within the field of genomics, illustrating their impact and utility, as well as their common limitations.

Finally, the chapter presents a detailed description of the HyenaDNA model, the primary model of interest in this study.

2.1 Transformers

Foundation Models are a class of ML models trained on broad sets of general-purpose data, capable of performing a wide range of tasks across different domains. Within the context of NLP, when trained on massive text datasets, they are known as Large Langauge Models (LLMs).

At the core of these foundation Models is the Transformer architecture, which has revolutionized the field of machine learning and is the backbone of many state-of-the-art models. While a comprehensive description of the Transformer architecture is beyond the scope of this thesis, an overview of its fundamental mechanisms will be provided to aid in understanding the subsequent discussion.

2.1.1 The Attention Mechanism

Attention is a technique designed to give a model the ability to focus on important parts of the input sequences. At a high level, this means that the model can learn to dynamically attend to related parts.

It was introduced in 2014 by Bahdanau et. Al [1] to improve neural machine translation, and has since then been widely adopted in many applications.

Formally, the attention mechanism in a sequence-to-sequence task is computed through the following steps:

1. Given an input sequence, some function a is used to compute the alignment scores $e_{i,j}$ between the inputs around position j and the output at position i.

   $$e_{i,j} = a(s_{i-1}, h_j)$$

   where a is generally a Feed Forward Neural Network.

   Intuitively, the alignment scores $e_{i,j}$ represent how strongly the inputs around position j and the output at position i are related.

2. The attention scores $\alpha_{i,j}$ are then computed by applying the softmax function to the alignment scores $e_{i,j}$

   $$\alpha_{i,j} = \text{softmax}(e_{i,j}) = \frac{\exp(e_{i,j})}{\sum_k \exp(e_{i,k})}$$

3. Finally, the redefined context vector c_i is computed as the linear combination of the input hidden states weighted by the attention scores

   $$c_i = \sum_j \alpha_{i,j} h_j$$

The *self-attention* mechanism is a variation of attention in which the goal is to relate a sequence to itself: instead of relating salient portions of the input to the output, self-attention focuses on related portions of the input to create a better representation of it. It can be written as an operation between matrices following these steps:

1. Encode the positional information into the input sequence, obtaining a positional embedding.

2. Compute three matrices, *query* Q, *key* K, and *Value* V, by means of three separate Feed Forward Neural Networks (FFNN), all taking the (same) positional embeddings as input.

3. Compute the alignment scores e as the scaled dot product between Q and K

$$e = \frac{QK^T}{\text{scaling}}$$

4. Compute the attention scores α as the softmax of the alignment scores

$$\alpha = \text{softmax}(\frac{QK^T}{\text{scaling}})$$

5. Obtain the final attention value A by multiplying V with the attention scores

$$A(Q, K, V) = \alpha V = \text{softmax}(\frac{QK^T}{\text{scaling}})V$$

From this matrix representation, it is almost immediate that the computational complexity of self-attention is $\mathcal{O}(L^2)$, where L is the length of the input. Intuitively, this quadratic complexity arises as the mechanism aligns the input sequence with itself, resulting in $L \times L$ operations.

Transformers Overview

The Transformer architecture was introduced in 2017 by *Vaswani et al.* [2] in the seminal paper "Attention is All You Need". In it, a novel architecture was presented, based on an encoder-decoder and leveraging the self-attention mechanism.

The self-attention mechanism described so far is usually contained in a single module called *attention-head*, which is represented in Figure 2.1. In the presented Transformer architecture, stacks of multiple attention heads were employed, allowing the model to attend to multiple portions and granularities in parallel. The overall mechanism is thus called *multi-head self-attention*.

The overall Transformer architecture, visualized in Figure 2.2, can be summarized as follows:

- The encoder consists of a stack of identical layers, each composed of two main sub-layers: a multi-head self-attention mechanism and a position-wise FFNN. Each sub-layer has a residual connection followed by layer normalization. The self-attention allows the encoder to attend to different parts of the input sequence while the FFNN is used to process the attended information.

- The decoder also consists of a stack of identical layers, with each consisting of three main sub-layers: apart from the two sub-layers already present in an encoder layer, there also is a multi-head attention mechanism attending to the encoder's output. Like in the encoder, each sub-layer in the Decoder

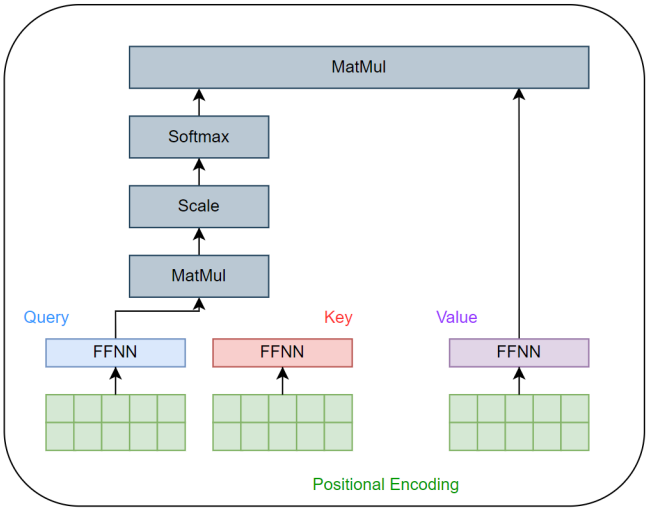


Figure 2.1: Scheme of an Attention Head, illustrating the steps involved in generating query, key, and value vectors and computing attention scores.

has a residual connection followed by layer normalization. Note that the first self-attention sub-layer in the decoder is masked to prevent attending to future positions, ensuring that predictions can depend only on the information before that position.

BERT

BERT, or Bidirectional encoder Representations from Transformers, is a Large Language Model (LLM) based on the Transformer architecture, specifically using its encoder. Introduced in 2018 by *Devlin et al.* [3], its primary innovation is the employment of bidirectional training instead of autoregression, allowing the model to consider the context from both directions (left-to-right and right-to-left).

BERT's architecture comes in different configurations, which differ by the number of Transformer layers, hidden units, and self-attention heads. For any configuration, the BERT model is capable of handling sequences of up to 512 tokens. Roughly, a token can be equated to a word, although more specifically, BERT employs the WordPiece tokenizer, where words are split into smaller parts in order to handle rare and out-of-vocabulary words.

The training of BERT follows the pre-training and fine-tuning paradigm.

BERT's pre-training is done with two unsupervised tasks, designed to enable the model to learn deep bidirectional representations:

- Masked Language Modeling (MLM) - in MLM, a percentage of the input

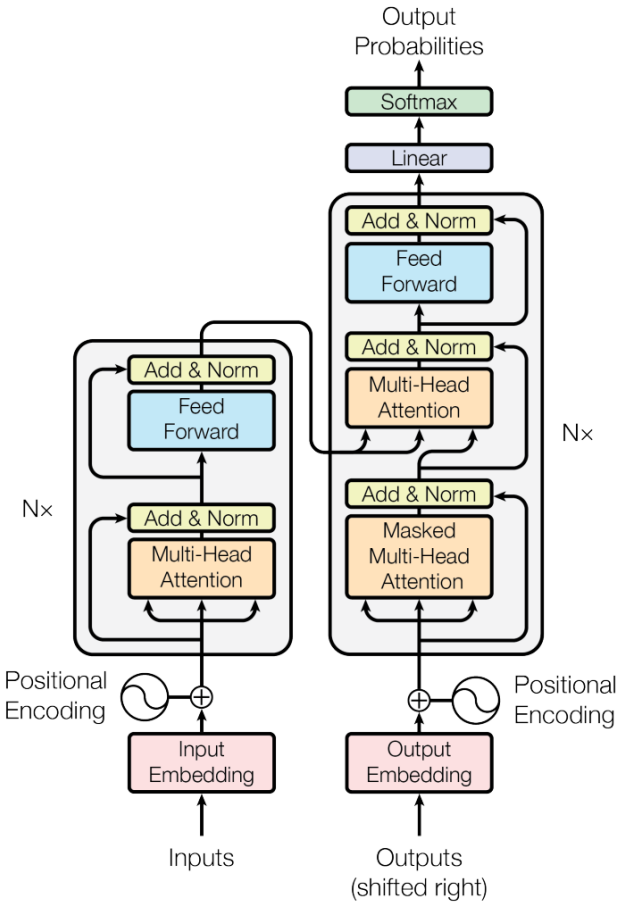


Figure 2.2: Diagram of a Transformer Architecture - *source: [2].*

tokens are masked at random, and the model is trained to predict these masked tokens based on their context. More specifically, 15% of the input token are selected at random; of them, 80% are replaced with the [MASK] token, 10% are left unchanged, and the remaining 10% are replaced by a randomly selected token in the vocabulary. This task is what allows BERT to consider the context of words from both directions.

- Next Sentence Prediction (NSP) - in NSP, the model is trained to predict whether a given pair of sentences follows each other in the original text. This task helps BERT understand the relationship between sentences, providing a broader understanding of the complete text.

After pre-training, BERT can be fine-tuned on specific downstream tasks by adding additional layers to the pre-trained backbone. The type of layer added

in this phase is generally task-specific. Since it was first presented, BERT has been fine-tuned successfully for a wide range of NLP tasks, including Question Answering, Sentiment Analysis, Named Entity Recognition, Text Classification, and Language Translation.

The impact of BERT's introduction was not limited to the NLP community, where at its release it achieved state-of-the-art results and generated numerous follow-up models that aim to improve upon it, such as RoBERTa [4], ALBERT [5], and DistilBERT [6].

2.2 Introduction to Genomics

Genomics is the branch of molecular biology dedicated to the comprehensive study of genomes, which are the complete sets of DNA within an organism's single cell. Understanding the genome is crucial for deciphering the biological information that dictates how organisms grow, develop, and function.

The field of genomics has evolved rapidly with advancements in sequencing technologies, which have made it possible to determine the precise order of nucleotides within DNA molecules quickly and accurately. These technological breakthroughs have expanded our understanding of genetic variation and enabled large-scale genomic studies, paving the way for significant scientific and medical discoveries.

Additionally, the integration of ML approaches has further propelled the field, offering powerful tools for analyzing and interpreting vast amounts of genomic data, uncovering patterns, and making predictions that were previously unattainable.

2.2.1 DNA and Genomic Sequences

DNA is the self-replicating, hereditary material found in most organisms. It's composed of nucleotides, which are organic molecules consisting of three components: a nitrogenous base (or nucleobase), a pentose sugar, and a phosphate group. The four nucleobases in DNA are adenine (A), cytosine (C), guanine (G), and thymine (T). DNA molecules form a double-stranded helix, with the nitrogenous bases of one strand pairing with the complementary bases of the opposite strand through hydrogen bonds, following the base pairing rules: A pairs with T, and C pairs with G. The two strands run in opposite directions, making the DNA molecule antiparallel.

A genome is the complete set of DNA within an organism's single cell, encompassing all of its genetic material, including both genes (coding regions) and non-coding sequences of the DNA.

Genomic sequences are the precise order of nucleotides within a DNA molecule. Sequencing these nucleotides is crucial for understanding genetic information.

Advances in sequencing technologies have revolutionized genomics, allowing for the rapid and accurate determination of DNA sequences. Their length is usually expressed in base pairs (bp).

DNA molecules are packaged into thread-like structures called chromosomes. Each chromosome contains a single, continuous DNA molecule coiled tightly around histone proteins. In humans, each cell typically contains 23 pairs of chromosomes, for a total of 46. These include 22 pairs of autosomes and one pair of sex chromosomes (XX for females and XY for males).

2.2.2 Cancer Genomics

Cancer genomics is the sub-field of genomics that studies cancer-associated genes. Unlike normal cells, cancer cells show uncontrolled growth and division due to mutations in their genetic material.

In cancer cells, different types of somatic mutations can appear:

- Nucleotides Polymorphisms - These are changes in a group of contiguous nucleotide bases, which can result in a different amino acid being incorporated into a protein, potentially altering its function.
- Insertions and Deletions (Indels) - These are additions or losses of small DNA segments, which can disrupt the reading frame of genes and lead to nonfunctional proteins.
- Copy Number Variations (CNVs) - These are changes in the number of copies of a particular gene or genomic region. CNVs can result in gene amplification or deletion, affecting gene dosage and cellular functions.
- Structural Variations - These are larger alterations in the genome, such as translocations, inversions, and large deletions or duplications. These changes can disrupt gene function and regulatory mechanisms.

2.3 Transformers in Genomics

The ability to efficiently represent genomic data, enabled by foundation models, has significantly advanced the field of genomics. These models provide deep contextualized numerical embeddings of genome sequences that capture complex relationships and dependencies, facilitating a deeper understanding and more sophisticated analysis of genetic information.

The similarities between the genomic and the human languages, including aspects going from alphabets and lexicons to grammar and phonetics, have been analyzed

and confirmed by linguistic studies long before the advent of ML, with research such as *Brengel and Buss*, 1984 [7] and *Searls*, 1992 [8]. From a computational point of view, and particularly in the context of ML, numerous similarities can also be observed in the nature of the data between NLP and genomics, especially when working directly on genomic sequences:

- Sequential Data - Both NLP and genomics involve processing sequential data. In NLP, the sequence is generally a series of words, while in genomics, it is a series of nucleotide bases.

- Order Matters - The order of elements is crucial in both fields. In NLP, the meaning of a sentence depends on the order of words. Similarly, in genomics, the biological function of a DNA sequence depends on the order of nucleotides.

However, it is also important to acknowledge the differences between the two fields.

- In NLP, the vocabulary can be very large, encompassing thousands of words, each with multiple meanings depending on context. Instead, in genomics, the vocabulary is generally limited to four primary characters (A, C, T, G) representing the nucleotide bases and an additional character, N, representing an unknown base.

- In NLP, texts have a clear semi-structure, as they can be further divided into sub-structures like sentences, periods, and paragraphs, allowing a natural way to split a text into smaller pieces. Instead, in genomics, DNA sequences lack such structures: while genes and chromosomes provide some level of organization, the sequences contained in them are still much longer than those that can be processed by most NLP tools. Table 2.1 provides a comparison of lengths in words or base pairs (bp) for various text and genomic structures.

Despite the differences, several methods originally designed for NLP have been successfully adapted to genomics. In particular, we will now give an overview of the most successful Transformer-based foundation models that have been employed in genomics

2.3.1 DNABERT

DNABERT is a pre-trained bidirectional encoder presented in 2021 by *Ji et al.* [9], with the aim of capturing a global and transferrable understanding of genomic sequences based on bidirectional nucleotide contexts, successfully adapting BERT. Most methods until this point adopted CNN-based architectures, and others

Table 2.1: Size Comparison of Text Structures and Genomic Elements, evidencing the different lengths of the sequences in the two fields.

Structure	**Length (Words/bp)**	**Field**
Sentence	10-20 words	NLP
Paragraph	100-200 words	NLP
Short Text (e.g., article)	500-1000 words	NLP
Book	50K-120K words	NLP
Chromosome	50M-250M bp	Genomics
Entire DNA (Human Genome)	$\approx$ 3.2B bp	Genomics

attempted the use of RNN-based models, such as LSTMs and GRUs, which weren't usable to handle large contexts.

The first step in adapting BERT to genomics was considering an appropriate tokenizer. Instead of regarding each base as a single token, DNABERT employs the *k-mer tokenization*, widely used in various genomic models. This approach leverages k-mer representations, where overlapping k-mers (groups of k contiguous nucleotide bases) are considered as tokens. Since different values of k lead to different tokenizations, the model has been pre-trained for different values of $k = 3,4,5,6$, resulting in different models denoted as DNABERT(k).

For any DNABERT(k), the vocabulary consists of all the possible permutations of a k-mer, as well as the usual tokens [CLS] for classification, [PAD] for padding, [UNK] for unknown, [SEP] for separation, and [MASK] for masked, totaling $4^k + 5$ possible tokens in the vocabulary

DNABERT's architecture mirrors that of BERT, specifically matching the dimensions of BERT-base thus maintaining a context length of 512 input tokens.

For pre-training, DNABERT retains only the MLM task from the original BERT, omitting the Next Sentence Prediction NSP task. Given the context length of 512 input tokens, the model is pre-trained on genomic sequences of length between $10 + k$ and $510 + k$ bp, extracted from the human genome using two approaches: direct non-overlap splitting and random sampling.

The extracet sequences are then tokenized into of 10 to 510 k-mers, with the addition of the special tokens [CLS] at its beginning, which represents the sequence as a whole, and [SEP] at its end, which identifies the end of a sequence. They are then padded to reach the fixed size of 512. The masking strategy employed is the same as BERT.

The pre-trained DNABERT(k) can be fine-tuned to various sequence- and token-specific prediction downstream tasks. Overall, the best-performing model was DNABER(6).

As a strategy to handle sequences longer than 512 tokens, they are split into subsequences. Then, the representation for each subsequence is concatenated, obtaining the representation of the long sequence. This architecture is named DNABERT-XL.

The DNABERT architecture, as depicted in the original paper, is illustrated in Figure 2.3.

2.3.2 Nucleotide Transformers

The Nucleotide Transformer is a foundation language model presented in 2023 by [10], designed to analyze genomic sequences. It employs the full Transformer architecture and uses an MLM task analogous to the one used for BERT for its pre-training.

The model was developed and pre-trained in four different sizes, ranging from 500M to 2.5B parameters, a significant increase from the 110M parameters of BERT-base. All configurations handle a context length of 1000 tokens.

For its tokenization, the Nucleotide Transformer uses non-overlapping 6-mer tokenization. This means that the input sequences are split, starting from the first nucleotide base, into non-overlapping groups of six. Distinct groups represent distinct tokens, thus resulting in a vocabulary size of $4^6 = 4096$ possible tokens plus the usual special tokens.

Given that the input dimension is 1000 tokens, and considering that the [CLS] special token is always appended at the beginning of a sequence, Nucleotide Transformer can handle input sequences of up to 5994 bp, however involving a certain level of compression.

2.3.3 DNABERT-2

DNABERT-2 was presented in 2023 by *Zhou et al.* [11] as the successor of DNABERT. It aims to improve upon the previous successful pre-trained foundation models, DNABERT and Nucletude Transformer.

DNABERT-2 uses the same architecture as BERT, and consequently DNABERT, while incorporating several advancements from recent deep learning and Transformer research, such as ALiBi [12], Flash Attention [13], and LoRa [14].

The main contribution of DNABERT-2 is the analysis and improvement of the tokenization method. As discussed before, previous works largely relied on k-mer

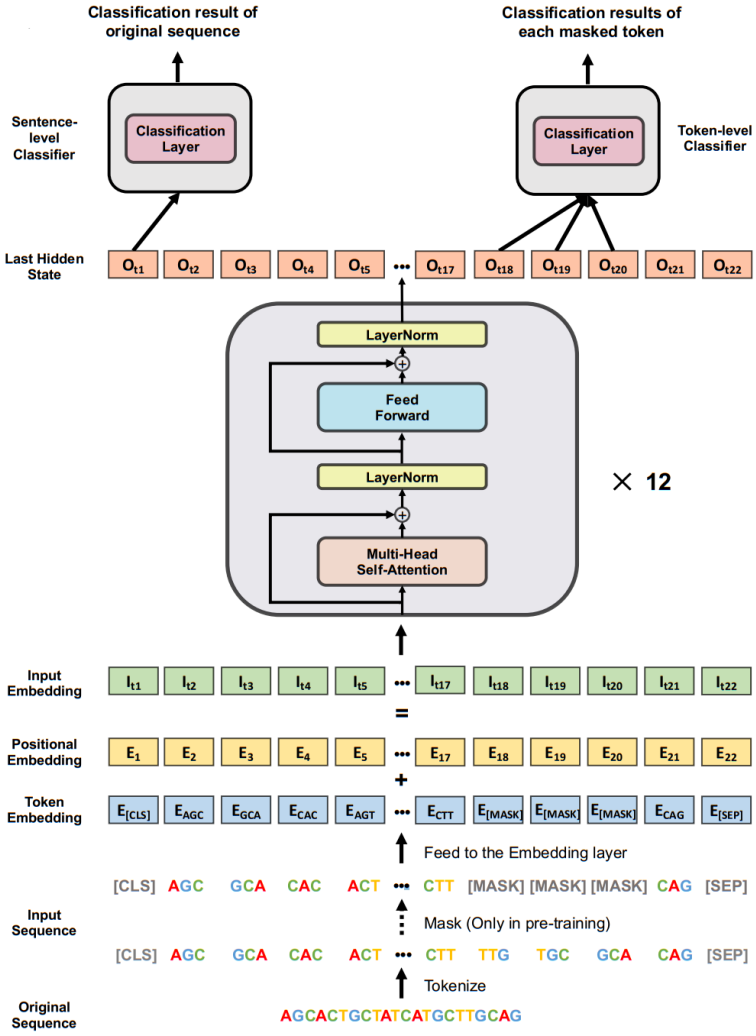


Figure 2.3: Architecture of the DNABERT(3) model, illustrating the employed overlapping k-mer tokenization and how it's followed by the standard BERT architecture - *source: [9].*

tokenization, whether overlapping, as in DNABERT, or non-overlapping, as in Nucleotide Transformer.

However, this approach presents several limitations, which can be visualized in Figure 2.4:

- Overlapping k-mer tokenization can lead to potential information leakage and poor computational efficiency. Additionally, it compresses the data without gaining the ability to handle sequences longer in terms of nucleotide bases.

- Although with Non-overlapping k-mer tokenization for the same amount of compression, it is possible to consider sequences longer in terms of nucleotide bases, this method suffers from poor sample efficiency. In fact, the change of even a single base can lead to drastically different tokenized sequences, which means that the model needs to align very different representations of otherwise nearly identical sequences.

- In-between methods, where k-mers are slid with a stride $1 < t < k$, still suffer from these limitations.

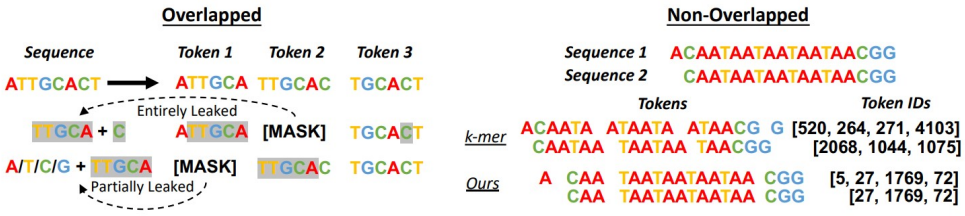


Figure 2.4: Drawbacks of the k-mer Tokenizations; in the overlapped version, it is possible to observe the information leakage, while in the non-overlapped version, the poor sample efficiency is visualized - *source: [11].*

To overcome the limitations caused by the use of k-mer tokenization, DNABERT-2 adapts the SentenPiece tokenization framework based on Byte Pair Encoding (BPE), already largely employed in large language models like GPT-2.

BPE is a technique originally developed as an algorithm for text compression. Starting from the single characters, BPE recursively computes the most frequent pairs of tokens and adds them to its vocabulary until a target size is reached. Eventually, this process creates a vocabulary of tokens of varying lengths, with the compression level dependent on the number of recursive steps performed.

Specifically, DNABERT-2 considers vocabularies with target sizes ranging from 2^8 to 2^{15} tokens, with the optimal found at a vocabulary size of $2^{12} = 4096$ tokens, signifying strong compression of the input data.

2.4 HyenaDNA

HyenaDNA is a novel genomic foundation model presented in November 2023 by *Nguyen et al.* [15].

As the computational cost of attention grows quadratically with the input sequence length, previous Transformer-based foundation genomic models could only

use a context of 512 to 1K tokens, limiting their capability to represent long-range interactions.

HyenaDNA aims to replace attention with a novel, subquadratic operator that maintains its desired properties:

- It must be data-controlled, similarly to how the attention matrix A is determined by the input sequence x, $A = A(x)$.
- It must allow for unrestricted context, i.e., the ability to access all the input tokens without locality constraints or memory loss.

The proposed solution is the Hyena operator, presented in 2023 by *Poli et al.* [16]. This operator is defined by the recurrence of two subquadratic primitives:

- An element-wise multiplicative gating.
- A *long convolution*.

The number N of recurrences of these two steps defines the order of the Hyena operator.

Once properly defined, this new operator can simply be used as a drop-in block to replace the attention layers in a decoder-only Transformer. A more detailed description of this operator will be provided later.

The HyenaDNA model can then be described as a decoder-only Transformer with the attention layer replaced by a Hyena operator. As illustrated on the left of Figure 2.5, a HyenaDNA block thus consists of a Hyena operator followed by a multi-Layer perceptron, using skip connection in both sub-layers

HyenaDNA is a foundation model pre-trained on the task of Next Token Prediction. It comes in different versions, named n-HyenaDNA, where n denotes the context length, ranging from 1024 to 1M tokens. Moreover, it uses a simple character-level tokenizer, thus maintaining single-nucleotide resolution.

Notably, the smallest version, with 1024 input tokens, is already comparable to or larger than other commonly used foundation genomic models. Moreover, given the use of implicit convolutions, as later detailed, this increased context length is obtained with a relatively small number of parameters. Specific details can be seen in Table 2.2,

Hyena Operator

We will now provide a further description of the Hyena operator, which is also illustrated in the center of Figure 2.5. Given an input sequence x of length L

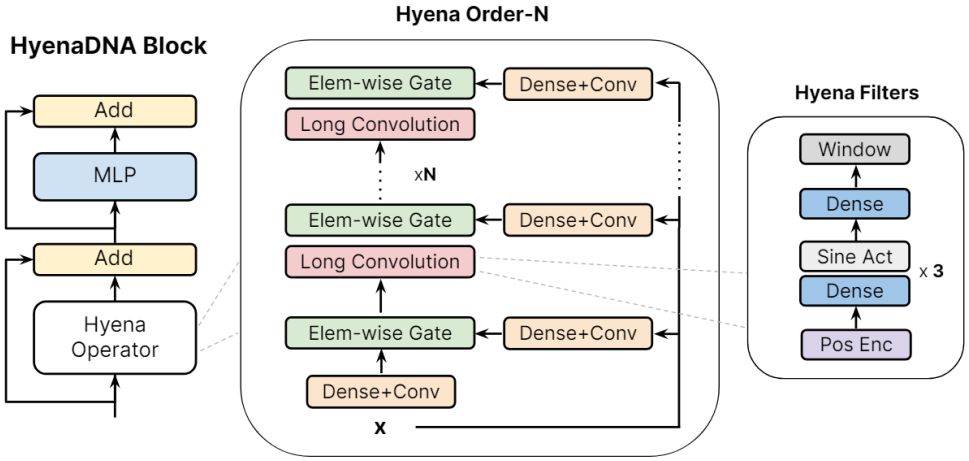


Figure 2.5: Diagram of a HyenaDNA Block. On the left, the overall structure of a HyenaDNA block. In the middle, a detailed view of a Hyena operator of order N. On the right, a detailed view of an Hyena filter - *source: [15]*.

Table 2.2: Comparison of Different Genomic Models; HyenaDNA stands out for the length of the inputs, while retaining a relatively small number of parameters.

Model Name	Input Tokens	Input bp	Number of Parameters
DNABERT(3)	512	3,072	110M
DNABERT-2	512	/	110M
Nucleotide Transformer (500M)	1,000	5,994	500M
Nucleotide Transformer (2.5B)	1,000	5,994	2.5B
1K-HyenaDNA	1,026	1,024	≈436K
16K-HyenaDNA	16,386	16,384	≈1.7M
32K-HyenaDNA	32,770	32,768	≈3.3M
160K-HyenaDNA	160,002	160,000	≈6.6M
450K-HyenaDNA	450,002	450,000	≈6.6M
1M-HyenaDNA	1,000,002	1,000,000	≈6.6M

1. The input is linearly projected using FFNNs into $N+1$ representations, named $v, x_1, ..., x_N$, similarly to how in attention an input is projected into the Q, K, and V matrices. This linear projection is performed by adding a very short convolutional layer, with a kernel length of $k = 2$, on top of the FFNNs.

2. For any subsequent recurrence i, a long convolution is applied to the output

z_i of the previous step. A long convolution is defined as a convolution with a kernel set to be equal to the sequence length, $k = L$.

3. Following each convolution, an element-wise multiplication between the current projection x_i and the output of the convolution is performed in an operation named element-wise multiplicative gating.

The overall algorithm can be represented as a linear input-output function $y = H(x)v$ for some Hyena matrix H, similarly to how attention can be represented as $y = A(x)V$.

To understand where the improvement in computational cost, let's write the Hyena operator as a recurrent equation

$$\begin{cases} z_1 = v \\ z_{n+1} = x_n(h_n * z_n), \;\; n = 1, ..., N \\ y = z_{N+1} \end{cases} \tag{2.1}$$

this can be rewritten as the input-output map

$$y = (x_N \cdot (h_N * (x_{N-1} \cdot (h_{N-1} * (...))))) \tag{2.2}$$

Recalling that, under suitable conditions, the Convolution Theorem states that the Fourier transform of a convolution between two functions is equivalent to the pointwise product of their Fourier transforms:

$$\mathcal{F}(a * b) = \mathcal{F}(a) \cdot \mathcal{F}(b) \tag{2.3}$$

we can rewrite Equation 2.2 as the successive applications of convolutions in the time and the Fourier domain, as

$$h * z = \mathcal{F}^{-1}(\mathcal{F}(h) \cdot \mathcal{F}(z)) \tag{2.4}$$

Having expressed the Hyena operator as a chain of element-wise products between Fourier and inverse Fourier transform, by leveraging the Fast Fourier Transform algorithm, we can conclude that the overall cost of a Hyena recursion is $\mathcal{O}(LlogL)$, and thus a Hyena operator of order N will have a cost of $\mathcal{O}(NLlogL)$.

Note that, since the long convolutions are not performed directly in the data but parametrized in a different space, they are defined *implicit*.

Finally, let's quickly give some further details about the Hyena filters used in the implicit long convolutions. As illustrated on the right of Figure 2.5, the convolution filter h is obtained by mapping the input index positions with a sub-module called Hyena filter, which

0. Given in input the positional encoding
1. Use a set of shallow FFNNs with *sine* activation functions
2. Finally, compute a smoothing operation named *window*

To understand the concept of implicit convolutions intuitively, consider how a line can be defined by just two parameters: its slope m and y-intercept q. Instead of needing to know all the pairs of (x, y) values that make up the line, knowing m and q is sufficient to compute any point on the line. Similarly, implicit convolutions use a set of parameters to represent complex operations over the entire sequence, allowing efficient computation without explicitly processing every individual element.

Chapter 3

Methodology

This section outlines the methodology used in this thesis to investigate the application of Transformer-based foundation genomic models, particularly HyenaDNA, in cancer genomics.

We begin by detailing the datasets employed, including the Human Reference Genome (HRG) and the DepMap dataset. Then, we describe the process of creating an ad hoc dataset tailored to our classification task. After that, we explain the steps for fine-tuning HyenaDNA to analyze the cancer genome.

3.1 Datasets

3.1.1 Human Reference Genome

The Human Reference Genome (HRG) is a comprehensive digital database representative of the complete set of DNA sequences in the human genome. It was obtained as a result of the Human Genome Project [17], which began in 1990 and was completed in 2003, and since then, it has been regularly updated.

The HRG is publicly accessible through various genomic databases, such as the National Center for Biotechnology Information (NCBI). Its latest version, GRCh38, also known as HG38, was released in 2017 and is regularly updated.

In genomics research, the HRG serves as a standardized reference for comparing individual genomes, helping to identify genetic variations and mutations that may contribute to diseases or other traits.

For this thesis, the GRCh38 version is considered, which includes the sequenced genomes for the 22 autosomes, the X and Y sex chromosomes, and the mitochondrial chromosome M.

In the GRCh38, three types of genomic sequences are found:

- NC, or Nucleotide Reference Sequences - These refer to curated reference sequences. For example, the sequence *NC_000001.11* refers to the reference sequence for chromosome 1. These sequences form the primary assembly of the genome.
- NW, or Nucleotide Working Sequences - These represent working draft sequences that are not yet part of the finished reference assembly. They may include unlocalized or unplaced scaffolds.
- NT, or Nucleotide Transcript Sequences - These include alternate loci, patch sequences, and other non-primary assembly sequences. They provide additional context for regions of the genome that are highly variable or not completely resolved in the primary assembly.

The sequences utilize various symbols to represent the nucleotide bases:

- A, C, G, T - When in uppercase, these symbols simply represent the standard nucleotide bases.
- a, c, g, t - When the standard nucleotide bases are in lowercase, it indicates a soft masking was applied to signal repetitive regions.
- N - The symbol N signifies that a specific nucleotide base could not be determined during sequencing. These undetermined bases often appear at the edges of chromosomes.
- R, M, S, Y, K - This set of symbols represents uncertainty between two specific nucleotide bases.

3.1.2 DepMap Dataset

The DepMap Portal offers a public collection of open-source data obtained from cancer cell lines, which are permanently established cell cultures that proliferate indefinitely under appropriate conditions. This dataset is curated as part of the Cancer Dependency Map project, an initiative aimed at creating a detailed map of genetic dependencies across a wide range of cancer types.

It can be used to obtain various information about cell lines, which is used for different research objectives. For our study, we specifically focus on two types of information: the mutations occurring in cancer cell lines, which we use to mutate the genomic sequences, and the cancer types. We will refer to this specific collection of data as the DepMap dataset.

In the DepMap dataset, somatic mutations are represented using the following five fields:

- Type - The type of the somatic mutation.
- Chrom - The chromosome in which the mutation has occurred.
- Pos - The position (relative to the chromosome) where the mutation occurred.
- Ref - The list of the nucleotide bases that are found in the HRG at the specified position.
- Alt - The list of nucleotide bases resulting from the mutation, replacing the original bases described in Ref.

The types of mutations considered are the nucleotide polymorphisms and the indels, and more specifically:

- SNP (Single Nucleotide Polymorphism) - in which the Ref nucleotide base at position Pos of chromosome Chrom is replaced with Alt.
- DNP (Double Nucleotide Polymorphism) - in which the Ref nucleotide bases at positions Pos and Pos+1 of chromosome Chrom are replaced with Alt.
- TNP (Triple Nucleotide Polymorphism) - in which the Ref nucleotide bases at positions Pos, Pos+1, and Pos+2 of chromosome Chrom are replaced with Alt.
- INS (Insertion) - in which the nucleotide base in position Pos of chromosome Chrom is substituted with the list of bases in Alt. This means that there was an insertion of bases between positions Pos and Pos+1.
- DEL (Deletion) - in which the Ref nucleotide bases starting at position Pos of chromosome Chrom are substituted with the first base of such string. This means that there was a deletion of bases starting from position Pos+1

Overall, the DepMap dataset contains an extensive collection of approximately 1.4M mutations from 1738 different cancer cell lines.

3.1.3 Custom Dataset

Here, we will first explain the process for selecting and cleaning the data from both the HRG and the DepMap dataset. Then, we will provide a high-level description of the mechanism developed to generate mutated sequences based on the combined information from the DepMap dataset and the HRG.

HRG

The data from the Human Reference Genome (HRG) is straightforward and well-curated, requiring minimal data cleaning. However, it contains additional information that is not needed for our purposes.

Firstly, the text is normalized to be all capitalized, as the soft masking for the repetitive region is not useful in our application.

The symbol N is kept and later used as the unknown token [UNK], similar to its use in NLP applications. The remaining symbols representing uncertainty are ignored, as they don't appear in the primary assembly, from which the sequences will be extracted.

Regarding the chromosomes, only the autosomes are considered, discarding the X, Y, and M chromosomes. The X and Y chromosomes are excluded due to their unique patterns of inheritance and higher variability, which could introduce additional complexity. The M (mitochondrial) chromosome is excluded because it is significantly smaller and has different biological characteristics compared to autosomes.

After these steps of data selection and cleaning, we are left with a dataset composed solely of the autosomes, normalized and filtered for relevant symbols. The details of the counts of the different considered symbols can be seen in Table 3.1.

Nucleotide	**Count**
A	∼867M
C	∼599M
G	∼601M
T	∼870M
N	∼151M
Total	**∼3.1B**

Table 3.1: Counts of the Occurrences of Each Nucleotide Base in the pre-processed HRG data.

DepMap Dataset

The 1731 cell lines present in the DepMap dataset are characterized by 31 possible types. The distribution of the organ is illustrated in Figure 3.1, and the total amount of mutations is $\sim 1.4M$.

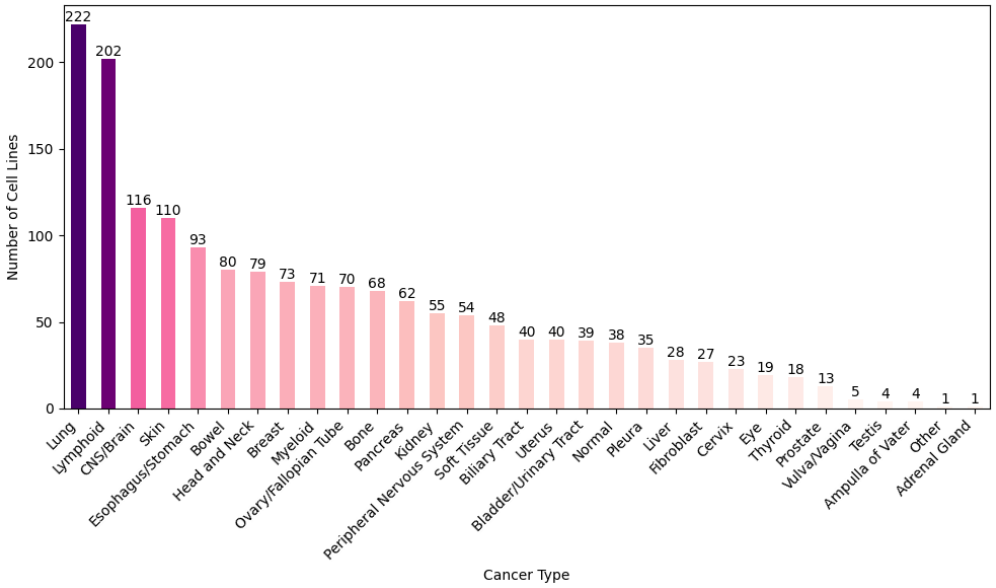


Figure 3.1: Cancer Types Distribution for the 1738 cancer cell lines contained in the DepMap dataset.

This study focused on lung and lymphoid cancer types, as these had the most data available. Additionally, given the anticipated complexity of the problem due to the specific challenges of cancer genomics, concentrating on these two cancer types allowed us to simplify the task to a binary classification.

In total, there are 222 lung cancer cell lines and 202 lymphoid cancer cell lines. The number of mutations coming from the cell lines of these types is approximately 188K and 178K, respectively. These cell lines were divided into a training set of 70 from each type and a test set of 20 from each type, with the remaining cell lines used for the embeddings that we will later describe in the first set of experiments.

For the selected cancer types, the number of mutations appearing across different cell lines varies greatly, as can be seen Table 3.2 and further visualized in Figures 3.2 for Lung, 3.3 for Lymphoid, and 3.4 for the aggregated count.

It's easy to visualize how the distribution of the number of mutations across the cell lines is skewed towards a larger number of mutations: although some cell lines only contain a small number of mutations, most of the distribution's weight is concentrated in the higher mutation counts.

Given this observation, and in order to leverage as much information as possible from the mutated sequences, we decided to only consider the cell lines with a number of mutations greater than the aggregated median. This allows us to encounter an increased number of mutations per sequence, giving the model the possibility to

Type	Min	Max	Avg	Median
Lung	214	3214	∼809	676,5
Lymphoid	152	9479	∼ 843	485,5
Aggregated	**152**	**9479**	∼ 825	596,5

Table 3.2: Distributions of the Mutations across the cell lines of the individual selected cancer types and their aggregated total.

analyze the relationship between them.

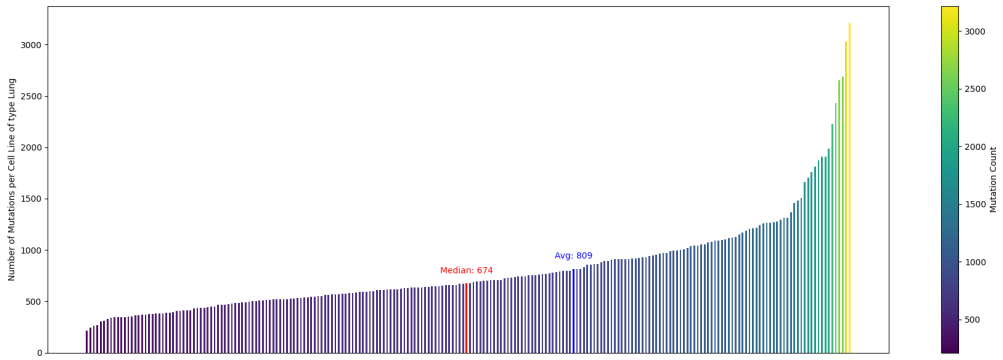


Figure 3.2: Distribution of the Number of Mutations across the cell lines of type lung.

Creating the Custom Dataset

To perform a cancer-type classification task directly on the mutated genome sequences, we need a mechanism designed to combine information from the DepMap dataset and the HRG. Specifically, we want to enable the model to analyze the relationship between mutations and their surroundings, including any neighboring mutations.

The mechanism designed for this process involves the following high-level steps:

0. Given a desired length L, selected a mutation from a cell line.

1. Compute a window of length L around the selected mutation, applying a random offset.

2. Retrieve all mutations from the same cell line that fall within the window.

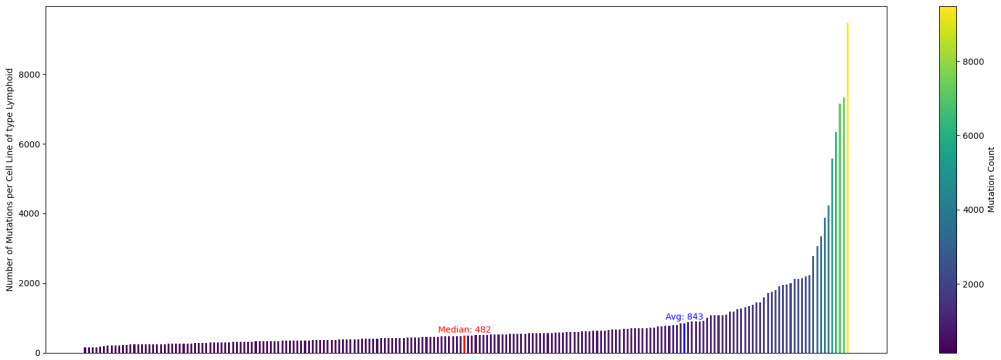


Figure 3.3: Distribution of the Number of Mutations across the cell lines of type lymphoid.

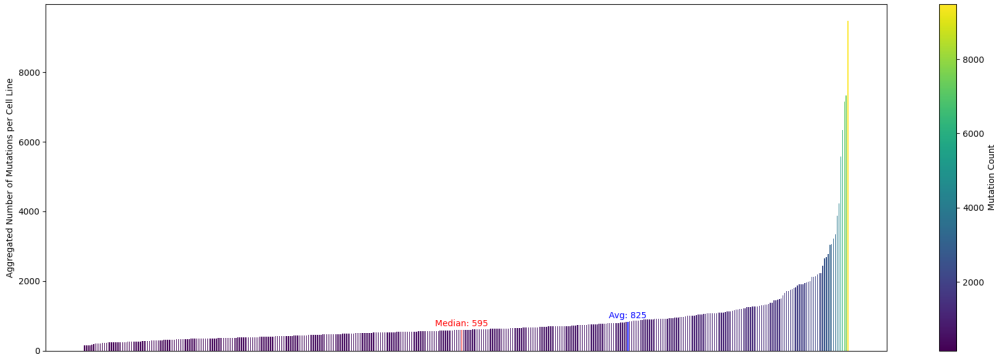


Figure 3.4: Aggregated Distribution of the Number of Mutations across the cell lines of types lung and lymphoid.

3. Extract the original genomic sequence corresponding to the computed window from the HRG and modify it according to the retrieved mutations.

4. If a deletion is found within the window, adjust the window size to ensure the output sequence is of the desired length, repeating the process until no new deletions are found.

The pseudocode to mutate the sequences is provided in Algorithm 1. This algorithm outlines the high-level steps required to create the mutated genomic sequences necessary for our analysis.

Algorithm 1 Sequence Mutating Function

Require:
s = selected sequence length
idx = index of the considered mutation
$MutationList$ = list of the selected mutations for the considered cell line

$m \leftarrow MutationList[idx]$
$type, chrom, pos, ref, alt \leftarrow m$
$c \leftarrow$ RetrieveChromosmeSequence($Chrom$)
$offset \leftarrow$ ComputeRandomOffset(s)
$AdjustmentFlag \leftarrow True$
$AdujstL \leftarrow 0$
while $AdjustmentFlag$ **do**
    $AdjustmentNeeded \leftarrow False$
    $MutationsBin \leftarrow$ FindNeighboringMutations($MutationList, idx, s, adjustL$)
    if Del was found **then**
        $AdjustL+ = len(Del)$
        $AdjustmentFlag \leftarrow True$
    end if
end while
$output \leftarrow$ MutateSeq($MutationsBin, c, s$)

3.2 Experiments

3.2.1 Visualization of Learned Embeddings

The first experimental step of this study was to determine whether HyenaDNA had learned an effective representation of the genomic data through its pre-training. Recalling that Transformer-based architectures learn to map the input data to a high-dimensional vector representation during pre-training, we first visualized the latent space learned by HyenaDNA.

For this purpose, we selected the largest HyenaDNA model feasible for our setup. Since we were only making inferences, computing the embeddings from the already pre-trained model without performing any additional training, we were able to employ the 450K-HyenaDNA, which can be used for inference with a single GPU of 40GB.

In the 450K configuration, the HyenaDNA model processes inputs of up to 450,002 tokens, which correspond, excluding the two special tokens at the extremities, to up to 450,000 bp. It then projects the input sequences into 256-dimensional embedding vectors.

To visualize these embeddings, we applied UMAP to reduce them to two dimensions. UMAP, introduced by McInnes et al. in 2018 [18], is a powerful stochastic dimensionality reduction technique effective for visualizing high-dimensional data. It aims to preserve both the local and global structures of data when projecting it into a lower-dimensional space.

UMAP is widely used in fields such as genomics and NLP to uncover underlying patterns, cluster data, and improve interpretability, as it helps in identifying relationships and structures that might not be immediately apparent in the high-dimensional space.

It is important to note that UMAP is a stochastic method, meaning its results can vary with different runs because it incorporates randomness in its algorithm. This variability can highlight different structures within the data that may not be as prominent otherwise, making traditional metrics for evaluating clusters not directly applicable.

The UMAP algorithm is characterized by two key parameters that significantly influence its behavior and the resulting visualization:

- n_neighbors - This parameter controls the local neighborhood size used in manifold approximation. Intuitively, it determines how UMAP balances between preserving the local and the global structure in the data. A smaller value makes the model focus more on capturing the local structure, making the algorithm sensitive to fine details, while a larger value tends to preserve the global structure, which can help capture broader patterns but potentially

smooth local variations.

- min_dist - This parameter controls the minimum distance between points in the low-dimensional representation. Intuitively, it affects how tight the clusters are in the embedding space. Smaller values sacrifice inter-cluster for intra-cluster distances, which can help distinguish dense clusters, while larger values tend to favorite the intra-cluster distances, which can help in maintaining the overall structure but might make it harder to see dense clusters.

For our experiment, we selected 3,960 genomic sequences from the first three chromosomes of the human genome and computed the 256-dimensional embeddings using the pre-trained 450K-HyenaDNA model.

We then applied UMAP dimensionality reduction for all the 36 possible combinations of the values of the parameters shown in Table 3.3. Using different combinations allowed us to better determine if the emerging structure was given by an actual local and global structure in the data or whether it was only a sub-product of the algorithm.

n_neighbors	**min_dist**
5	0.0
10	0.1
20	0.25
50	0.5
100	0.8
200	0.99

Table 3.3: Considered Values for the UMAP Parameters used to reduce the dimensionality of the embeddings computed by the pre-trained 450K-HyenaDNA.

The dimensionality-reduced vectors were then visualized using different labels. This involved plotting the vectors color-coded by their respective labels using standard libraries, specifically `matplotlib`.

First, we considered as labeling the chromosomes from which the sequences were taken. This approach aimed to determine whether broad positional information was captured during pre-training.

Next, we refined this approach by using finer positional information, dividing the chromosomes into equal-width bins. Given that the first three chromosomes have an average length of approximately 229 million nucleotide bases, about 509 non-overlapping sequences of length 450K can be extracted from them. By focusing

on more local positional information, we could better determine the positional understanding built by the model.

Finally, as a control test, we considered mutated sequences extracted with the mechanism in Algorithm 1, using the cancer type as the label. Clearly, we did not anticipate any significant clustering or structure, given that the model could not yet build any understanding regarding the mutations.

3.3 HyenaDNA Fine-Tuning

The second set of experiments was designed to evaluate the effectiveness of fine-tuning HyenaDNA for application in cancer genomics. Specifically, we considered the novel task of classifying cancer types directly from the mutated sequences.

To the best of our knowledge, this is the first attempt at the aforementioned task. A similar approach is seen in the Mutation Attention (MuAt) model, presented in 2023 by *Sanjaya et al.* [19]. However, unlike our approach, MuAt does not directly reconstruct cancer types from mutated sequences. Instead, it embeds the lists of mutations and uses the (binned) positions of mutations to provide extra positional information to the model.

As already discussed in Section 1, our approach aims to enable the model to attend to the entire mutated sequences. We hypothesize that, by doing so, the model will be able to identify relationships between mutations and their surrounding reference genome, potentially leading to a more accurate classification of cancer types.

Both the 32K-HyenaDNA and 1K-HyenaDNA models were trained under different settings. The 32K model consists of approximately 3.3 million parameters and can process 32,770 input tokens, corresponding to sequences of up to 32,768 nucleotide bases. The 1K version, on the other hand, comprises approximately 436,000 parameters and can process 1,026 tokens, or up to 1,024 nucleotide bases. Both models map the input sequences into 256-dimensional vectors and require at least 16GB of GPU memory for fine-tuning.

We note that the 32K-HyenaDNA already largely surpasses the context that was previously available in the literature. The BERT-based models could only process inputs of up to 512 nucleotide bases, while the Nucleotide Transformer considered input lengths of 6000 nucleotide bases compressed in 1000 tokens.

We deem that a much larger context than that previously available in the literature is needed for applications in cancer genomics due to the challenges discussed in Section 1.

Due to the computational intensity of training these models, we did not perform

extensive hyperparameter tuning. Instead, we primarily relied on the hyperparameters set in the original work.

Data Preprocessing

To prepare for fine-tuning, the selected cell lines were split into training and test sets. Specifically, from each considered cancer type, approximately 42,000 mutations were selected for training, and around 10,000 mutations for testing.

In addition to balancing between the two classes, we ensured that the number of mutations from each cell line was balanced as well, to avoid inducing unwanted biases in the model. Given the variation in mutation counts, we randomly selected a number of mutations from each cell line equal to the minimum number of mutations across all considered ones. This ensures that only a set number of mutations, the largest possible while keeping the dataset balance, was used from each cell line.

Once the mutations were selected, we employed the mechanism described in Algorithm 1 to mutate the reference sequences extracted from the pre-processed HRG dataset, with the sequence length set to be as large as possible for the considered model.

The resulting mutated genomic sequences were then tokenized using a simple character tokenizer. This tokenizer breaks down the sequences into individual characters, assigning a distinct token to each nucleotide base. The character 'N' was replaced with the [UNK] token.

Additionally, we appended the usual special tokens at the start and end of each sequence. This helps the model recognize the boundaries of input sequences, facilitating a better understanding and processing of the data.

Metrics

During training, we used binary cross-entropy loss, which is well-suited for classification tasks involving balanced classes. We recall that the binary cross entropy loss function is

$$L = -\frac{1}{N}\sum_{j=1}^{N}\left(t_j log(p_j) + (1-t_j) log(1-p_j)\right)$$

where N is the number of data points, t_j the truth value, and p_j the softmax probability for the j-th data point.

To evaluate performance, acuracy was chosen for consistency with the original paper and as a straightforward, easily interpretable measure of the proportion of correctly classified sequences. This choice is particularly appropriate given that

the dataset was perfectly balanced between the two classes, ensuring that accuracy provides a reliable indicator of model performance.

Additionally, gradient clipping was applied with a value of 1.0 to help preventing gradients from exploding, ensuring stable and efficient training. This technique helps maintain the stability of the learning process, especially when dealing with large models and complex datasets.

Optimization and Scheduling

The optimization of the model parameters was performed using the AdamW optimizer, with the specific hyperparameters visible in Table 3.4.

AdamW is a widely favored choice for applications involving large-scale neural networks due to its ability to handle sparse gradients on noisy problems, as it combines the benefits of both adaptive learning rate methods and weight decay regularization. The adaptive learning rate mechanism adjusts the step size for each parameter dynamically, which helps in converging faster and more effectively. The weight decay component helps in regularizing the model, preventing overfitting by penalizing large weights.

Parameter	**Value**
lr	0.0006
wd	0.1
(β_1, β_2)	(0.9, 0.999)

Table 3.4: Considered Values for the Parameters of the AdamW Optimizer used for fine-tuning 1K-HyenaDNA and 32K-HyenaDNA.

Additionally, the learning rate schedule was managed using a cosine scheduler with a linear warmup. The parameters for this scheduler are detailed in Table 3.5.

This scheduler initially increases the learning rate linearly during the warm-up phase, then decays it following a cosine schedule. This approach aids in achieving better convergence and stable training. The graph followed by the scheduler is illustrated in Figure 3.5.

Further Implementation Details

Although HyenaDNA reduces computational cost significantly by using a sub-quadratic operator, as discussed in Section 2, training these models is still computationally intensive. This intensity is especially pronounced when dealing with larger input sizes.

Parameter	Value
t_in_epochs	False
t_initial	25000
lr_min	5.9999999999999995e-05
warmup_lr_init	1.0e-06
warmup_t	250.0

Table 3.5: Considered Values for the Parameters of the Cosine Warmup Scheduler used for fine-tuning 1K-HyenaDNA and 32K-HyenaDNA.

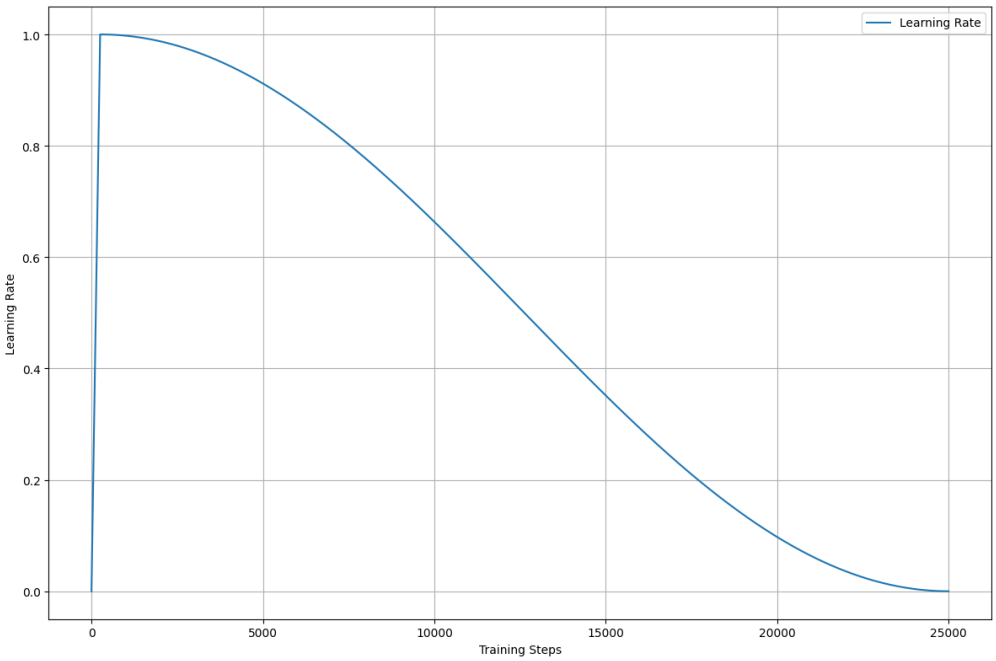


Figure 3.5: Plot of the Cosine Scheduler with Linear Warmup.

Given the size of the sequences, we were limited to using a global batch size of 2 for the 32K-HyenaDNA model. Despite HyenaDNA's efficiency improvements, handling such large sequences remains demanding on hardware resources.

For the smaller 1K-HyenaDNA model, we could employ a larger global batch size of 8.

All models were fine-tuned to completion for a set number of epochs ranging from 20 to 200.

Experiments Design

In the fine-tuning experiments, our objective was to determine whether the representations learned during pre-training by foundation models were beneficial for downstream tasks directly applied to mutated sequences, testing the feasibility of applying the foundation model in cancer genomics.

Thus, the experiments were designed to assess the efficacy of HyenaDNA models for this task by:

- Assessing whether HyenaDNA can be successfully fine-tuned for our classification task, meaning it can be employed directly on mutated genomic sequences.
- Identifying the specific factors that contribute to the success of these models in cancer genomics.
- Determining whether the efficacy of these models can be attributed to their deep architecture or if the learned representations are essential for their success.

As discussed in Section 1, a major challenge when working with cancer genomic data is the sparsity of the mutation. We thus hypothesize that models must leverage a wide context to capture long-range dependencies between neighboring mutations and their surroundings effectively.

Our primary experiment aimed to determine whether increasing the context length is necessary for effectively working with mutated genomic sequences.

To this end, we fine-tuned and compared two versions of the HyenaDNA model: the 1K-HyenaDNA, with a context length of 1026 tokens, and the 32K-HyenaDNA, with a context length of 32768 tokens. Further details of these configurations are provided in Section2.4. Both models were trained following the settings described previously.

Accuracy and loss statistics were tracked using `TensorBoard`, a suite of visualization tools designed to understand, debug, and optimize models within the popular ML library `TensorFlow`.

We hypothesize that the 32K model, with its significantly longer context, should outperform the 1K model due to its enhanced ability to capture extensive contextual information. A longer context not only allows for a more comprehensive understanding of the genomic data during pre-training but also, as shown in Figure 3.6, enables the model to consider sequences with, on average, more mutations per sequence, thereby better capturing the relationships between mutations.

Notably, most foundation models in genomics typically support context lengths ranging from 512 to 1K tokens, as already detailed in Table 2.2.

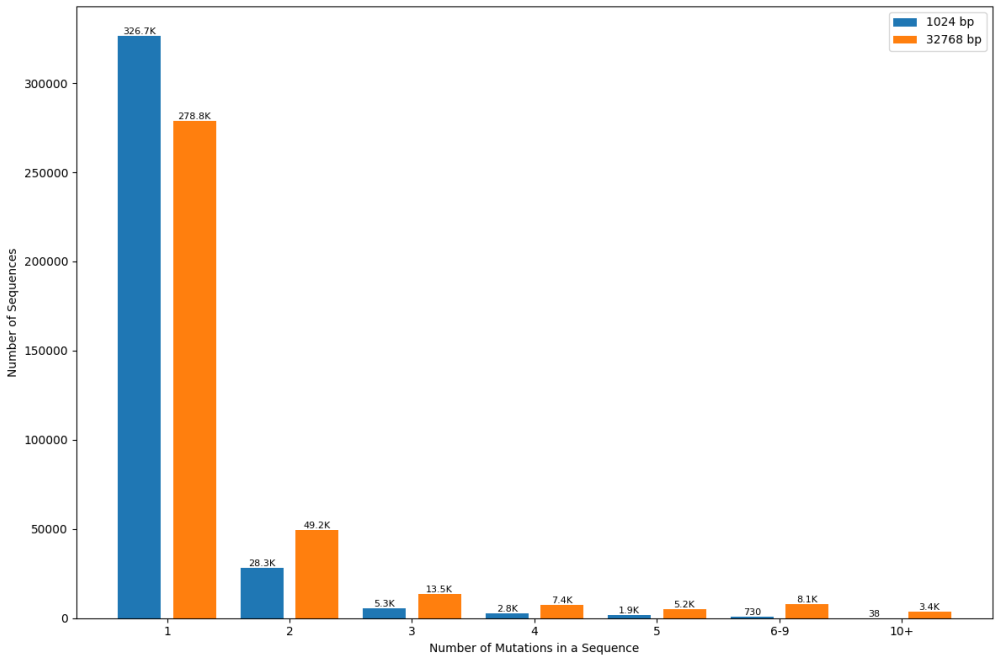


Figure 3.6: Counts of the Number of Mutations Encountered in a Sequence for context lengths of 1024 and 32768 bp.

To further investigate the importance of learned representations in cancer genomics, we explored the impact of different initialization strategies for the 32K model. We evaluated the model's performance under two scenarios: randomly initializing the model's weights and freezing the backbone while training only the classification head.

In the first scenario, we randomly initialized the weights of the 32K model instead of loading pre-trained ones. This approach aims to assess whether the model can learn effectively from scratch without leveraging pre-trained knowledge. While similar approaches have been effective for general genomic tasks, such as the HyenaDNA from-scratch experiment, given the complexity of cancer genomic data, we anticipated that this strategy might struggle to perform well.

In the second scenario, we loaded the pre-trained weights and froze the decoder backbone of the 32K model. This implies that only the classification head, which is comprised of a simple, fully connected linear layer, could be trained. This approach was designed to test whether the fixed representations learned during pre-training were sufficient for the classification task. We expected this limited model to struggle due to its inability to adapt the embeddings or incorporate any new knowledge about the mutations.

In summary, our fine-tuning experiments aimed to compare the performance of the 1K and 32K models, test the impact of different initialization strategies, and understand the role of context length and pre-trained representations for the application of foundation models in cancer genomics.

We expected that a longer context length would provide significant advantages and that pre-trained weights would be crucial for achieving optimal performance in this complex task.

A recap of the experiments can be seen in Table 3.6.

Experiment	**Input Length**	**Seq. Length**	**Weight Initialization**	**Batch Size**
1K-main	1026	1024	Pre-trained	8
32K-main	32770	32768	Pre-trained	2
32K-random	32770	32768	Random	2
32K-frozen	32770	32768	Frozen Backbone	2

Table 3.6: Summary of the HyenaDNA Fine-Tuning Experiments.

Chapter 4

Results and Discussion

In this chapter, we will delve into the results obtained from the experiments outlined in Chapter 3.

We will begin by discussing the visualization of the embeddings generated by the pre-trained HyenaDNA model, using UMAP to project these high-dimensional representations into two dimensions.

This will be followed by an analysis of the results from the fine-tuning experiments. We will explore how these results demonstrate the potential and promise of our approach for advancements in the field of cancer genomics.

After presenting and analyzing these findings, we will discuss the limitations encountered during this study and propose potential future directions for further research and improvement.

4.1 Visualization of Learned Embeddings

As discussed in the Methodology, this experiment aimed to evaluate the effectiveness of representation learning in genomics. Specifically, we visualized the embeddings generated by the pre-trained 450k-HyenaDNA model using UMAP, labeling them based on different granularities of positional data. Additionally, we labeled these embeddings against cancer-type mutations, noting that the model was only pre-trained on generic genetic data.

4.1.1 Positional Information

Chromosome-Level Analysis

By analyzing the visualization obtained with the methodology described in the previous Chapter, it was evident that the model had effectively captured an intrinsic representation of the positional information within the genomic sequences.

The results of our first experiment, which aimed to visualize the dimensionality-reduced embeddings labeled based on the chromosome from which the original sequence was extracted, can be seen in Figure 4.1.

Of the possible 36 combinations of the two UMAP parameters described in 3, we only report a selected choice of 6 in the Figure to avoid redundancy. However, note that similar results can be observed for any choice of parameters, indicating the robustness of the representations learned by the model.

From the visualization, it is evident that the sequences coming from the same chromosome do not form a single cluster. Instead, the sequences from each chromosome form multiple distinct, smaller clusters.

However, this is an expected outcome since the sequence length is much shorter than the chromosome length, and no Next Sentence Prediction (NSP) task was involved in the HyenaDNA pre-training.

When learning representation using an unsupervised task during pre-training, perfect separation is not anticipated as the model is not specifically trained for such purposes. The important observation is that, although trained on unsupervised data and not given any information about the chromosome of origin, the model has still effectively learned to distinguish between different chromosomes. This implies that during subsequent fine-tuning, the model can leverage its pre-trained information. It needs only to incorporate the information about mutations into the representations, slightly skewing the embeddings in the latent space to accommodate this new information.

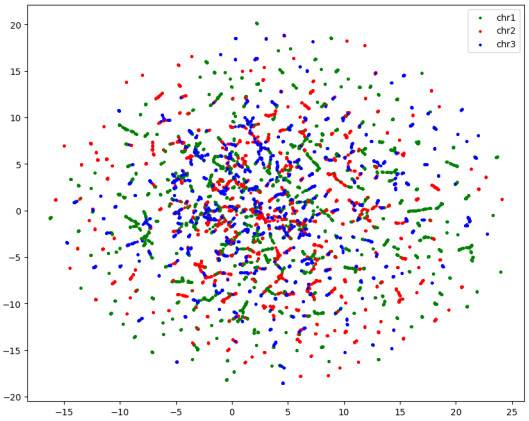
Thus, this first experiment on macro-level positional information already demonstrates quite well that the model has an intrinsic ability to differentiate genomic sequences by their chromosome of origin, highlighting the effectiveness of the HyenaDNA model in capturing relevant positional data.

Binned Positional Analysis

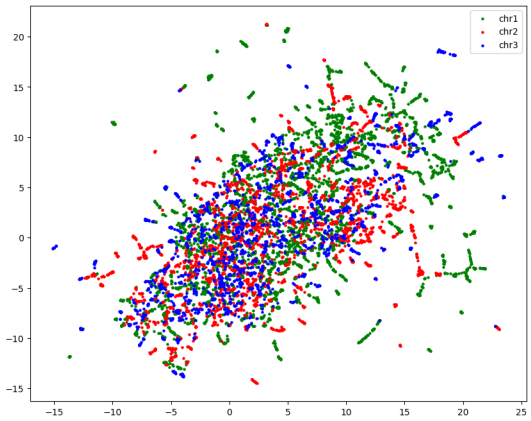
This differentiation becomes even more pronounced when visualizing the embeddings against binned positions within chromosomes, thus considering more localized positional information. By dividing each chromosome into smaller, equal-width bins, we can observe how more refined positional distinctions are captured even better by the model.

As shown in Figure 4.2, the visualization of four specific bins demonstrates even clearer cluster formations. Each bin represents a localized region of the chromosome, and the sequences within each bin form distinct, although still not perfectly separated, clusters, further indicating that the model effectively captures local positional dependencies.

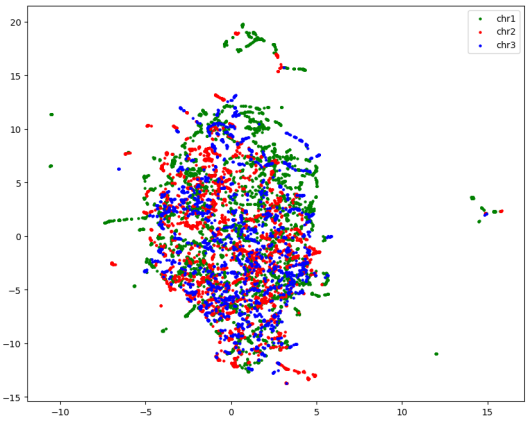
Figure 4.1: Visualization of the Dimensionality-Reduced Embeddings computed by the pre-trained 450k-HyenaDNA. The embeddings are colored by the chromosomes of origin, specifically focusing on the first three autosomes.



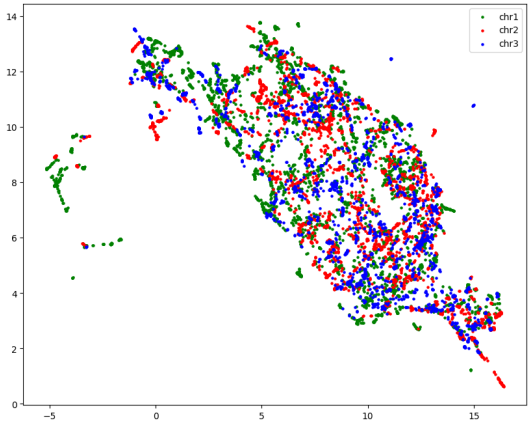
$(n\_neighbors, min\_dist) = (5,\ 0.25)$



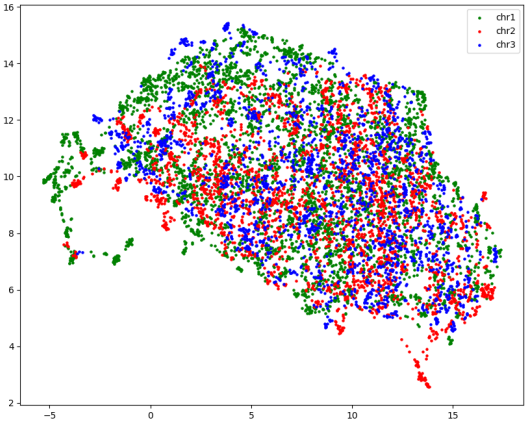
$(n\_neighbors, min\_dist) = (10,\ 0.5)$



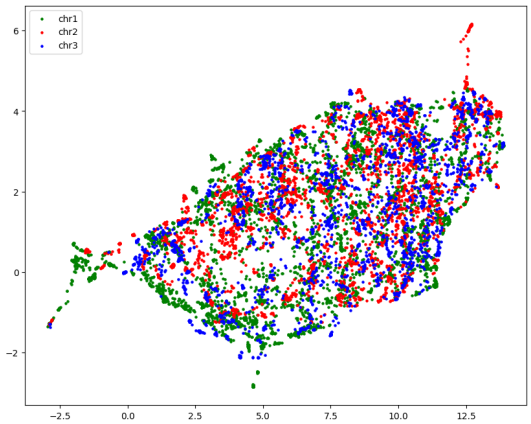
$(n\_neighbors, min\_dist) = (20,\ 0.25)$



$(n\_neighbors, min\_dist) = (50,\ 0.1)$



$(n\_neighbors, min\_dist) = (100,\ 0.5)$



$(n\_neighbors, min\_dist) = (200,\ 0.1)$

This outcome was expected, as the model can more easily capture detailed positional information with the binning dimension growing closer to the input length.

Furthermore, by plotting the remaining sequence embeddings from the same chromosome with a uniform color as a background, as shown in Figure 4.3, the distinct clusters of the selected bins become even more pronounced when compared to the global structured of the remaining sequences.

Mutational Clustering Analysis

We also colored the embeddings based on the mutations, taking mutated sequences from our two considered cancer types. As shown in Figure 4.4, the visualization demonstrates the lack of distinct clusters when embeddings are colored based on mutations, regardless of the parameters used for UMAP.

This outcome is expected, as no information regarding the mutations could have been learned during pre-training.

Conclusions

These observations further prove how the model has built an effective intrinsic representation of positional information. By simply mapping the sequences to its latent space, the model can discern sequences based on the region of the chromosome they were extracted from, without ever being explicitly given this information.

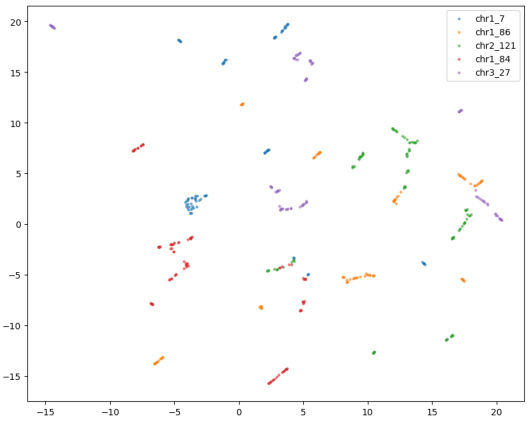
The clear clustering at both the chromosome and bin levels underscores the robustness of the HyenaDNA model in capturing various levels of positional data within genomic sequences.

4.2 HyenaDNA Fine-Tuning

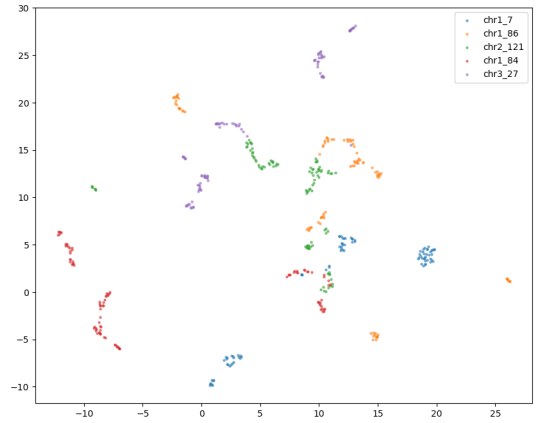
From the visualization of the embeddings, we observed that the HyenaDNA model effectively captured an intrinsic representation of positional information during pre-training. This was evident from the clear clustering of sequences based on their chromosome of origin and the more refined positional distinctions observed within binned chromosome regions.

Now, we shift our focus to the fine-tuning experiments. These experiments aim to evaluate the model's performance in a specific task: cancer-type classification from mutated genomic sequences. By fine-tuning the pre-trained HyenaDNA model, we investigate the impact of context length and representation learning on model performance, comparing the results of different fine-tuning strategies to understand their effectiveness in this challenging task.

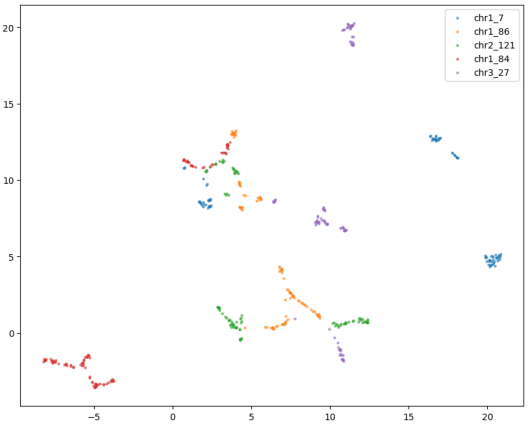
Figure 4.2: Visualization of the Dimensionality-Reduced Embeddings computed by the pre-trained 450k-HyenaDNA. The embeddings are colored by the binned position of origin, specifically focusing on the top 5 most common positions.



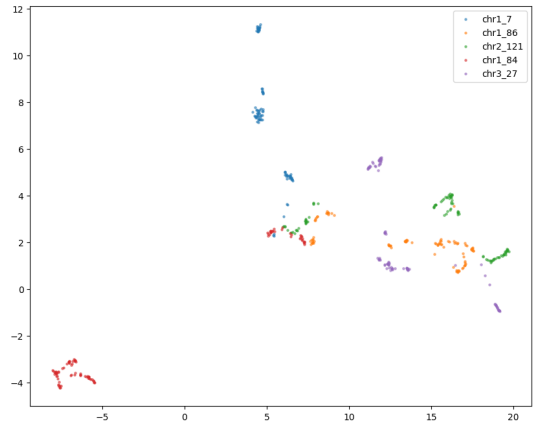
$(n\_neighbors, min\_dist) = (5,\ 0.25)$



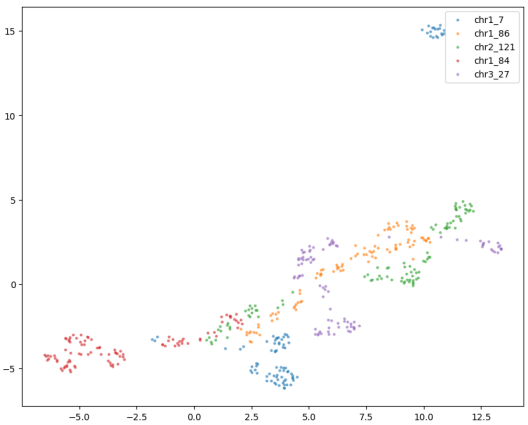
$(n\_neighbors, min\_dist) = (10,\ 0.5)$



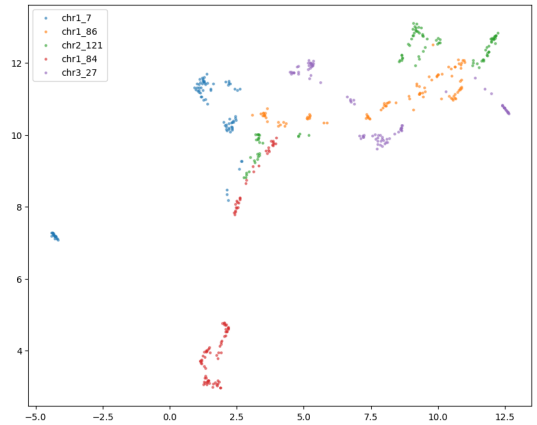
$(n\_neighbors, min\_dist) = (20,\ 0.25)$



$(n\_neighbors, min\_dist) = (50,\ 0.1)$

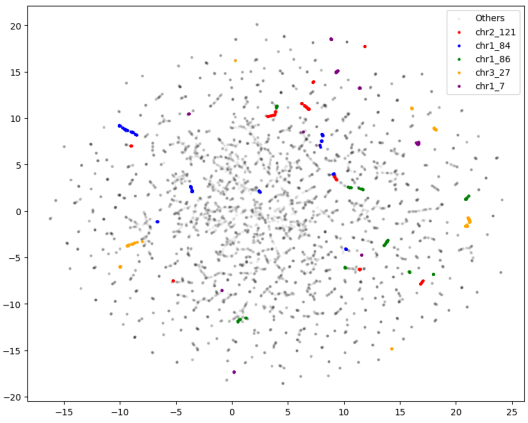


$(n\_neighbors, min\_dist) = (100,\ 0.5)$

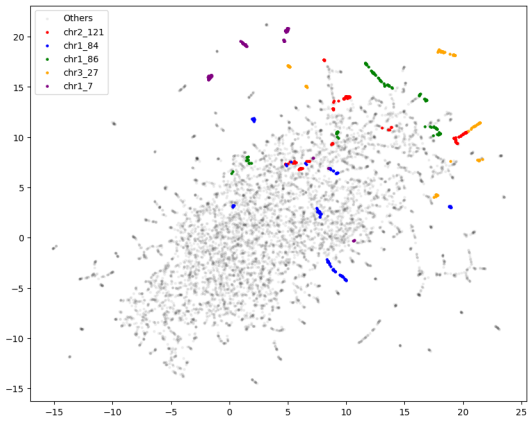


$(n\_neighbors, min\_dist) = (200,\ 0.1)$

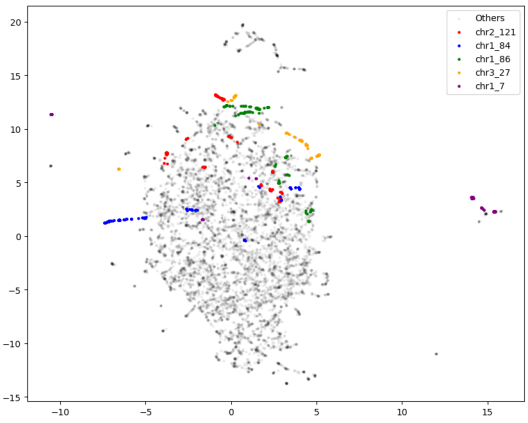
Figure 4.3: Visualization of the Dimensionality-Reduced Embeddings computed by the pre-trained 450k-HyenaDNA. The embeddings are colored by the binned position of origin, specifically focusing on the top 5 most common positions. The remaining sequences are visualized in black to enhance the clarity of the visualization.



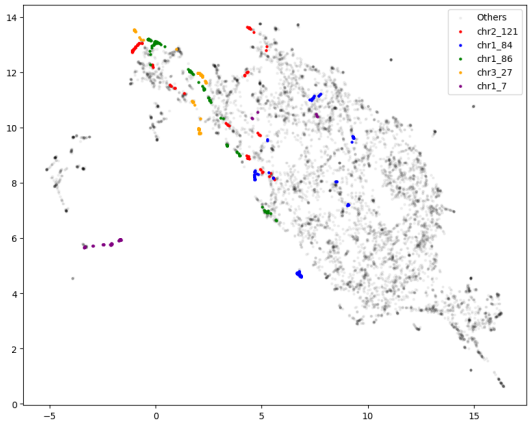
$(n\_neighbors, min\_dist) = (5,\ 0.25)$



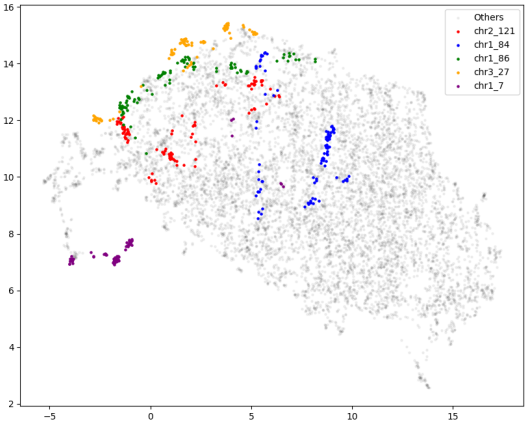
$(n\_neighbors, min\_dist) = (10,\ 0.5)$



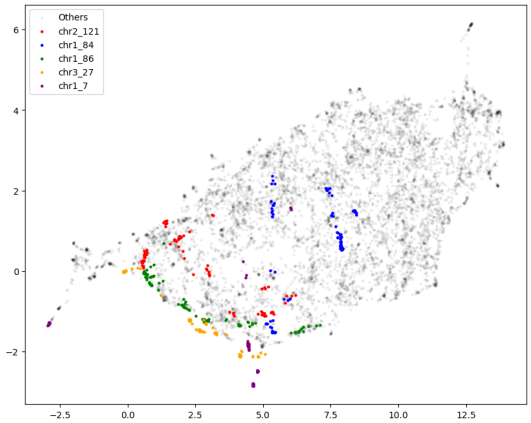
$(n\_neighbors, min\_dist) = (20,\ 0.25)$



$(n\_neighbors, min\_dist) = (50,\ 0.1)$

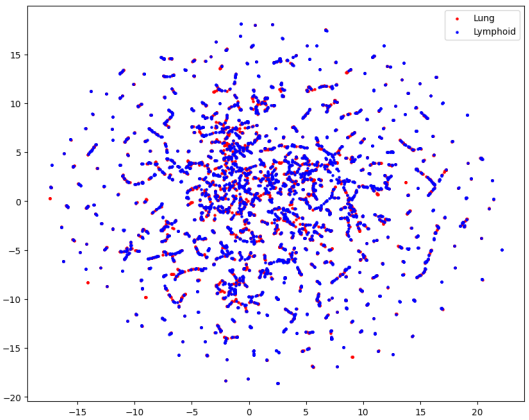


$(n\_neighbors, min\_dist) = (100,\ 0.5)$

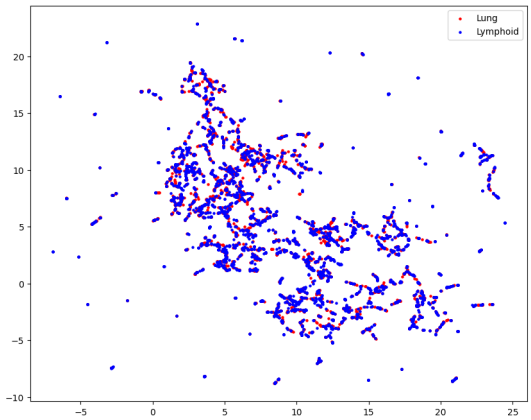


$(n\_neighbors, min\_dist) = (200,\ 0.1)$

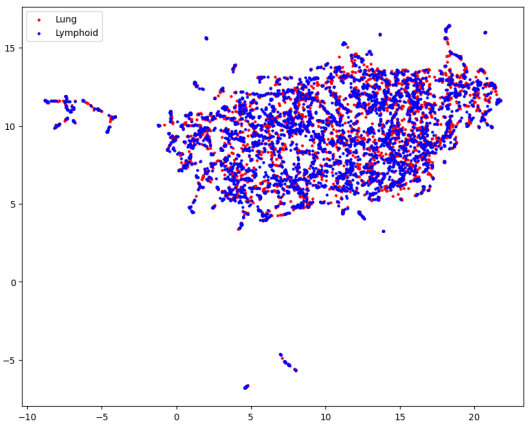
Figure 4.4: Visualization of the Dimensionality-Reduced Embeddings computed by the pre-trained 450k-HyenaDNA. The embeddings are derived from sequences mutated by cancer mutations, selected from lung and lymphoid cancer cell lines.



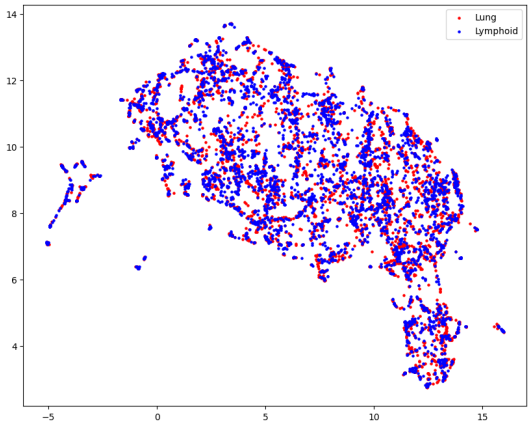
$(n\_neighbors, min\_dist) = (5,\ 0.25)$



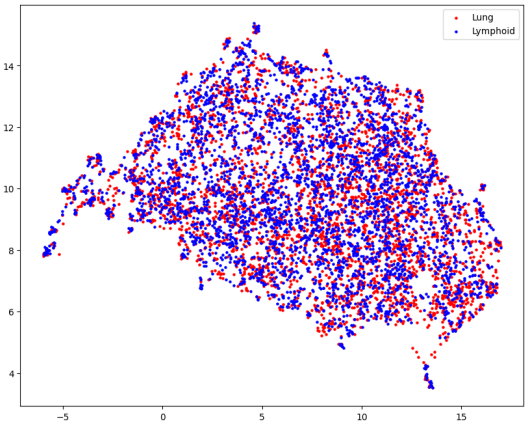
$(n\_neighbors, min\_dist) = (10,\ 0.5)$



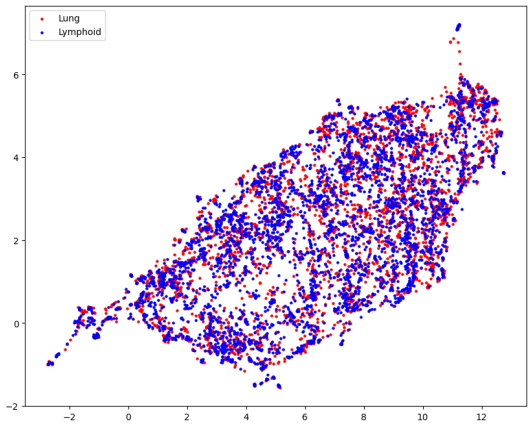
$(n\_neighbors, min\_dist) = (20,\ 0.25)$



$(n\_neighbors, min\_dist) = (50,\ 0.1)$



$(n\_neighbors, min\_dist) = (100,\ 0.5)$



$(n\_neighbors, min\_dist) = (200,\ 0.1)$

4.2.1 Enlarging the Sequence Context

As can be seen from the top accuracies reported in Table 4.1, the increase in the context length resulted in a clear improvement in model performance, demonstrating the advantage of a longer context.

The 1K model, which already considers an input length equal to or larger than other foundation genomic models in the literature, proved to be capable of learning the task effectively.

Notably, the 32K model showed a clear furhter increase in performance, achieving an improvement of approximately 72% over the 1K model.

The loss function curves for the 32K models, directly tracked using `TensorBoard` and depicted in Figures 4.5 and 4.6, respectively, indicate that the training was overall stable and the improved performance was not caused by random fluctuations.

Although not perfectly converged, the curves are approaching a plateau zone.

Several factors could have contributed to the increased performance of the model with a larger context.

The ability to process longer sequences allows HyenaDNA to capture more extensive dependencies and interactions within the genomic data, which is crucial given the sparsity of them mutations across the sequences.

Additionally, a larger model inherently has more computational capacity, often enabling it to handle complex tasks more efficiently if enough data is considered.

However, these factors alone are not enough to explain the improvement. In many genomic tasks, a limited context length of 512 tokens has been sufficient to achieve state-of-the-art performances without the need for broader sequences, as can be seen with works like DNABERT, DNABERT-2, and Nucleotide Transformer. This suggests that for applications in cancer genomics, the ability to process larger sequence lengths is fundamental, highlighting the unique challenges and requirements of this specific field.

4.2.2 Evaluating the Need for Representations

As discussed in Section 3, to further test our hypothesis on the importance of representations in cancer genomics, we performed two additional experiments of fine-tuning HyenaDNA under different configurations.

Specifically, in the first experiment, we fine-tuned the model with a randomly initialized backbone, while in the second, we trained the model on our classification task with the decoder backbone frozen.

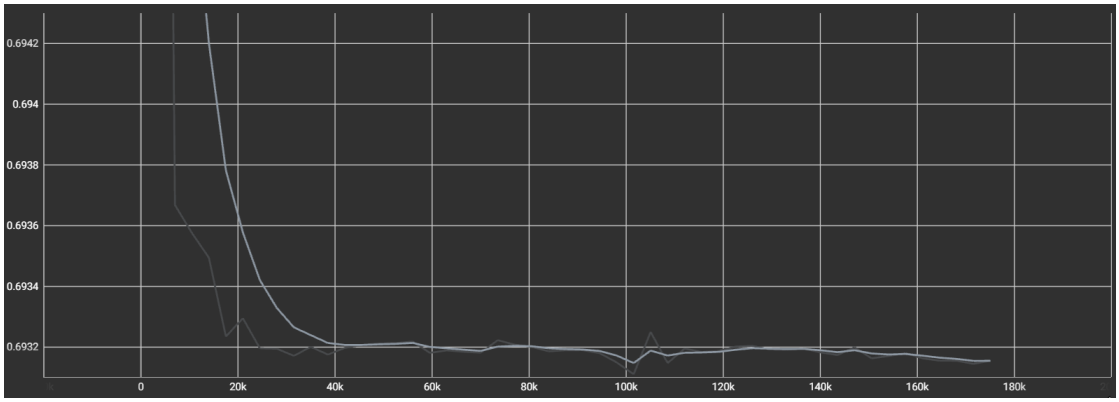


Figure 4.5: Plot of the Train Loss for the 32K-HyenaDNA experiment, obtained using `TensorBoard`.

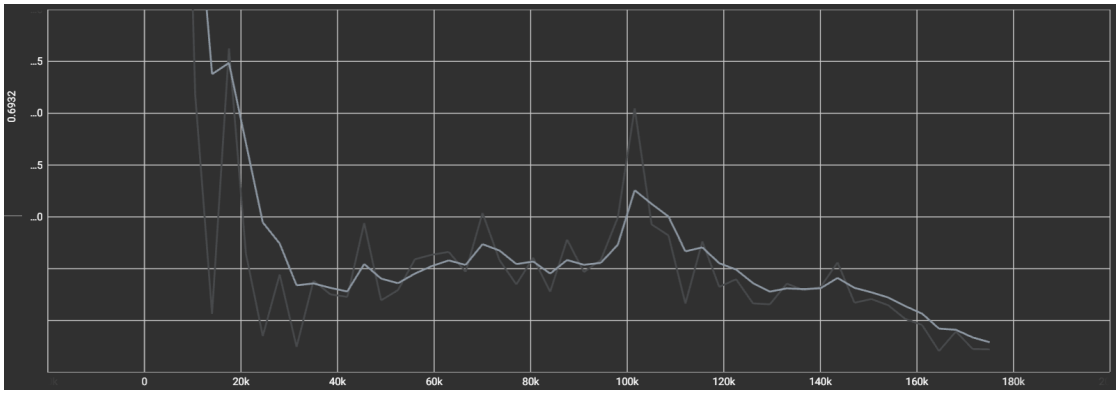


Figure 4.6: Plot of the Validation Loss for the 32K-HyenaDNA experiment, obtained using `TensorBoard`.

Random Initialization

The 32K-random model failed to improve its performance beyond the random-choice baseline. Without pre-trained weights, the model had to learn to efficiently represent the mutated sequences based solely on the sparse information about the mutations. Moreover, the learning of the representation had to be driven exclusively by the classification task, rather than benefiting from pre-training tasks specifically designed to optimize representation learning.

This proved too challenging, as the model was unable to effectively capture the intricate dependencies between the sparse mutations and the genomic environment necessary for accurate cancer-type classification without the support of a previously learned efficient representation of the data, thus indicating that training the full

architecture from scratch is too complex in our context.

Notably, the same approach applied to the Genomic Benchmark dataset [20], a curated collection of various sequence classification datasets in the field of genomics commonly used for benchmarking foundation models, achieves state-of-the-art results.

Frozen Backbone

Conversely, the 32K-frozen model, where only the classification head was trainable, faced different challenges. With the backbone being frozen, the information about the mutations could only be captured by the classification head, while the backbone provided only positional information, which was not descriptive enough for accurate cancer-type classification. Intuitively, this means that the classification head had to learn to predict the cancer types based only on the positional information of the input sequences.

Given that the head is a simple, fully connected linear layer, it was likely unable to learn to distinguish the embedded input sequences supplied by the pre-trained backbone based solely on the sparse mutations.

As a result, the performance quickly degraded over time, eventually causing the loss to explode. These limitations made the task too complex for the model.

Conclusions

These observations underscore the necessity for an adaptable and trainable backbone to handle the intricacies of genomic data.

The randomly initialized model lacked pre-trained weights, meaning it did not have sufficient prior information to effectively represent the mutated sequences, resulting in poor performance. The frozen backbone attempted to classify based only on positional information, which was insufficient.

	1K-HyenaDNA	**32K-HyenaDNA**		
		Pre-trained	**Random**	**Frozen**
Top Accuracy	0.5268	0.5462	0.5	/

Table 4.1: Top Accuracies Obtained by the Fine-Tuning Experiments

4.3 Discussion

4.3.1 Conclusions

In this study, we conducted a series of experiments to explore the capacity of current foundation model architectures, specifically HyenaDNA, to handle cancer-type-specific genomic data. Our experiments focused on the importance of deep, contextualized representations in cancer genomics.

- We utilized UMAP to visualize the embeddings generated by the pre-trained 450K-HyenaDNA model, color-coded based on different granularities of positional information. The distinct clustering observed indicated that the model successfully learned during its unsupervised pre-training to map the genomic sequences to a latent space that efficiently captures positional information.
- We evaluated the impact of context length on model performance by comparing models with different input lengths, ranging from 1K to 32K tokens. The results demonstrated that increasing the context length clearly improved model performance, as anticipated due to the nature of the data. A larger input length means that not only the model can access to a broader context, but also that a larger number of mutations is found on average in the input sequences.
- We experimented with a randomly initialized model, which struggled to improve beyond the random-choice baseline as it had to learn to represent the mutated sequences solely during fine-tuning. This highlighted the complexity of training the full architecture from scratch in our context and the importance of leveraging effective representations of the data.
- We also experimented with a model where only the classification head was trainable while the pre-trained backbone remained frozen. This meant that the backbone could only provide positional information, and the classification head was insufficient to capture the detailed mutation information needed, leading to degraded performance.

These findings collectively underscore the importance of representations in cancer genomics. They highlight the need for an adaptable and trainable backbone to handle the complexities of cancer genomic data. The experiments demonstrated that both pre-trained positional knowledge and the ability to incorporate mutational information during fine-tuning are critical for approaching accurate cancer-type classification.

4.3.2 Limitations of the Study

The limitations encountered in this study will now be addressed and discussed..

First, we note that, to the best of our knowledge, this is the first attempt made to apply a foundation model directly to mutated genomic sequences. This means that there is a lack of comparable studies to validate our results. Consequently, it's not possible to assess our model's performance relative to existing methods. Moreover, it's hard to establish an upper bound for its effectiveness.

Second, while HyenaDNA is highly suitable for our case, its recency means that the library is still quite immature. Thus, a significant amount of effort was required to ensure the model's operational stability and integrate it effectively into our research framework. This included addressing bugs, optimizing performance, and customizing certain functionalities to meet our specific research needs.

Finally, although HyenaDNA is generally lighter to train compared to other models, it remains very resource-intensive. Training larger versions of the model or incorporating more data could be challenging without adequate computational resources.

Overall, these limitations highlight the pioneering nature of our work and underscore the need for continued efforts in this area. Addressing these challenges will be crucial for advancing the application of foundation models in cancer genomics and unlocking their full potential for scientific discovery and medical advancements.

4.3.3 Future Directions

While this work represents a promising initial step, there are several possibilities for future research that could further enhance the performance and applicability of this approach.

Currently, an even larger version of the model, the 160K-HyenaDNA, is being trained. This expansion should further increase performance by allowing the model to attend to more mutations and incorporate a broader genomic context, building on the improvements observed when increasing the context length from 1K to 32K tokens.

As discussed previously, a larger model would not only be better equipped to address the task and provide a more comprehensive genomic context, but it would also include more mutations within each sequence, providing the model with richer data to improve its accuracy.

With sufficient resources, the largest HyenaDNA models available, such as the 450K- and 1M- models, could also be employed.

An immediate improvement could be achieved through a more thorough hyperparameter tuning process. Given the high cost of training these models, we largely relied on the settings from the original work. However, a comprehensive hyperparameter search would enable us to maximize the performance of the considered

configurations.

Optimizing hyperparameters such as learning rates and regularization techniques, as well as employing more advanced stopping criteria for the training, would allow us to obtain the most out of the current configurations.

With sufficient resources, further or even exclusive pre-training on cancer-specific genomic data could enable the model to incorporate mutations into its intrinsic representation more efficiently than relying solely on the fine-tuning phase.

This specialized pre-training would allow the model to develop a deeper understanding of the unique characteristics of cancer genomic data, leading to improved performance in cancer genomics.

However, pre-training a foundation model requires significantly more computational resources and data compared to fine-tuning, which can prove a substantial barrier.

Finally, to further pre-train on mutated sequences, one could develop a specialized pre-training task adapted for cancer genomic data to allow the model to focus more on the mutations rather than the already well-represented genomic context.

In summary, these future directions aim to build on the initial successes of this study by expanding model capacity, optimizing training processes, and tailoring the approach specifically to the complexities of cancer genomic data. These efforts hold the potential to significantly advance the field and improve the accuracy and robustness of the application of foundation models in the field of cancer genomics.

Bibliography

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. *Neural Machine Translation by Jointly Learning to Align and Translate*. 2016. arXiv: 1409.0473 [cs.CL]. URL: https://arxiv.org/abs/1409.0473 (cit. on p. 6).

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL]. URL: https://arxiv.org/abs/1706.03762 (cit. on pp. 7, 9).

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL]. URL: https://arxiv.org/abs/1810.04805 (cit. on p. 8).

[4] Yinhan Liu et al. *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. 2019. arXiv: 1907.11692 [cs.CL]. URL: https://arxiv.org/abs/1907.11692 (cit. on p. 10).

[5] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. *ALBERT: A Lite BERT for Self-supervised Learning of Language Representations*. 2020. arXiv: 1909.11942 [cs.CL]. URL: https://arxiv.org/abs/1909.11942 (cit. on p. 10).

[6] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. *DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter*. 2020. arXiv: 1910.01108 [cs.CL]. URL: https://arxiv.org/abs/1910.01108 (cit. on p. 10).

[7] V. Brendel and H. G. Busse. «Genome structure described by formal languages». In: *Nucleic Acids Research* 12.5 (Mar. 1984), pp. 2561–2568. DOI: 10.1093/nar/12.5.2561 (cit. on p. 12).

[8] David B. Searls. «The Linguistics of DNA». In: *American Scientist* 80.6 (1992), pp. 579–591. URL: http://www.jstor.org/stable/29774782 (cit. on p. 12).

[9] Yanrong Ji, Zhihan Zhou, Han Liu, and Ramana V Davuluri. «DNABERT: pre-trained Bidirectional Encoder Representations from Transformers model for DNA-language in genome». In: *Bioinformatics* 37.15 (Feb. 2021), pp. 2112–2120. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btab083. eprint: https://academic.oup.com/bioinformatics/article-pdf/37/15/2112/57195892/btab083.pdf. URL: https://doi.org/10.1093/bioinformatics/btab083 (cit. on pp. 12, 15).

[10] Hugo Dalla-Torre et al. «The Nucleotide Transformer: Building and Evaluating Robust Foundation Models for Human Genomics». In: *bioRxiv* (2023). DOI: 10.1101/2023.01.11.523679. eprint: https://www.biorxiv.org/content/early/2023/01/15/2023.01.11.523679.full.pdf. URL: https://www.biorxiv.org/content/early/2023/01/15/2023.01.11.523679 (cit. on p. 14).

[11] Zhihan Zhou, Yanrong Ji, Weijian Li, Pratik Dutta, Ramana Davuluri, and Han Liu. *DNABERT-2: Efficient Foundation Model and Benchmark For Multi-Species Genome*. 2024. arXiv: 2306.15006 [q-bio.GN]. URL: https://arxiv.org/abs/2306.15006 (cit. on pp. 14, 16).

[12] Ofir Press, Noah A. Smith, and Mike Lewis. *Train Short, Test Long: Attention with Linear Biases Enables Input Length Extrapolation*. 2022. arXiv: 2108.12409 [cs.CL]. URL: https://arxiv.org/abs/2108.12409 (cit. on p. 14).

[13] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. *FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness*. 2022. arXiv: 2205.14135 [cs.LG]. URL: https://arxiv.org/abs/2205.14135 (cit. on p. 14).

[14] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. *LoRA: Low-Rank Adaptation of Large Language Models*. 2021. arXiv: 2106.09685 [cs.CL]. URL: https://arxiv.org/abs/2106.09685 (cit. on p. 14).

[15] Eric Nguyen et al. *HyenaDNA: Long-Range Genomic Sequence Modeling at Single Nucleotide Resolution*. 2023. arXiv: 2306.15794 [cs.LG]. URL: https://arxiv.org/abs/2306.15794 (cit. on pp. 16, 18).

[16] Michael Poli, Stefano Massaroli, Eric Nguyen, Daniel Y. Fu, Tri Dao, Stephen Baccus, Yoshua Bengio, Stefano Ermon, and Christopher Ré. *Hyena Hierarchy: Towards Larger Convolutional Language Models*. 2023. arXiv: 2302.10866 [cs.LG]. URL: https://arxiv.org/abs/2302.10866 (cit. on p. 17).

[17] Francis S. Collins and Leslie Fink. «The Human Genome Project». In: *Alcohol Health and Research World* 19.3 (1995). PubMed-not-MEDLINE, pp. 190–195. ISSN: 0090-838X (cit. on p. 21).

[18] Leland McInnes, John Healy, and James Melville. *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction*. 2020. arXiv: 1802.03426 [stat.ML]. URL: https://arxiv.org/abs/1802.03426 (cit. on p. 29).

[19] Prima Sanjaya, Sebastian M. Waszak, Oliver Stegle, Jan O. Korbel, and Esa Pitkänen. «Mutation-Attention (MuAt): deep representation learning of somatic mutations for tumour typing and subtyping». In: *bioRxiv* (2022). DOI: 10.1101/2022.03.15.483816. eprint: https://www.biorxiv.org/content/early/2022/03/17/2022.03.15.483816.full.pdf. URL: https://www.biorxiv.org/content/early/2022/03/17/2022.03.15.483816 (cit. on p. 31).

[20] Katarina Gresova, Vlastimil Martinek, David Cechak, Petr Simecek, and Panagiotis Alexiou. «Genomic Benchmarks: A Collection of Datasets for Genomic Sequence Classification». In: *bioRxiv* (2022). DOI: 10.1101/2022.06.08.495248. eprint: https://www.biorxiv.org/content/early/2022/06/10/2022.06.08.495248.full.pdf. URL: https://www.biorxiv.org/content/early/2022/06/10/2022.06.08.495248 (cit. on p. 47).