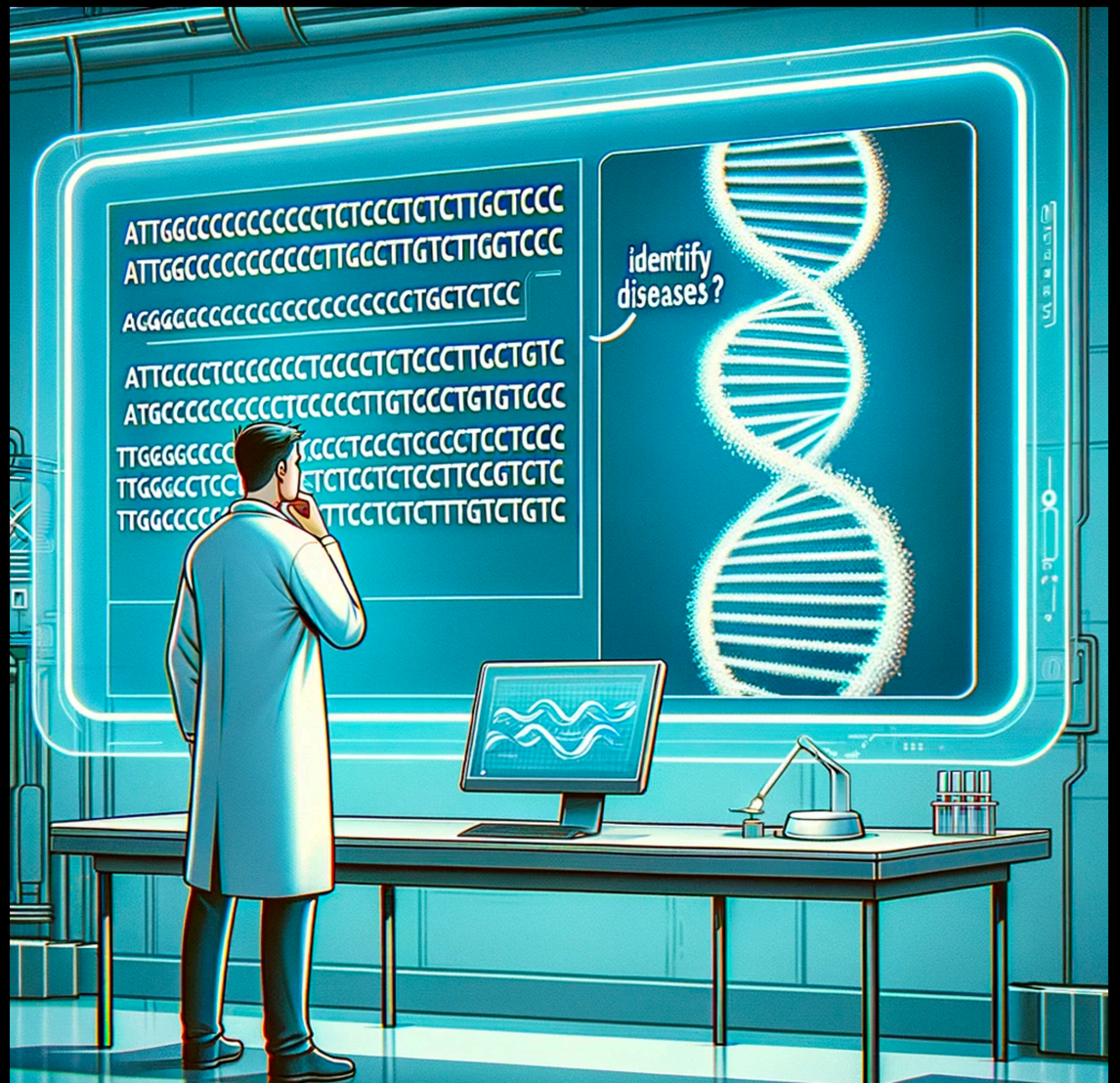
HyenaDNA: Long-Range Genomic Sequence Modeling at Single Nucleotide Resolution

Aadit Kapoor

What?



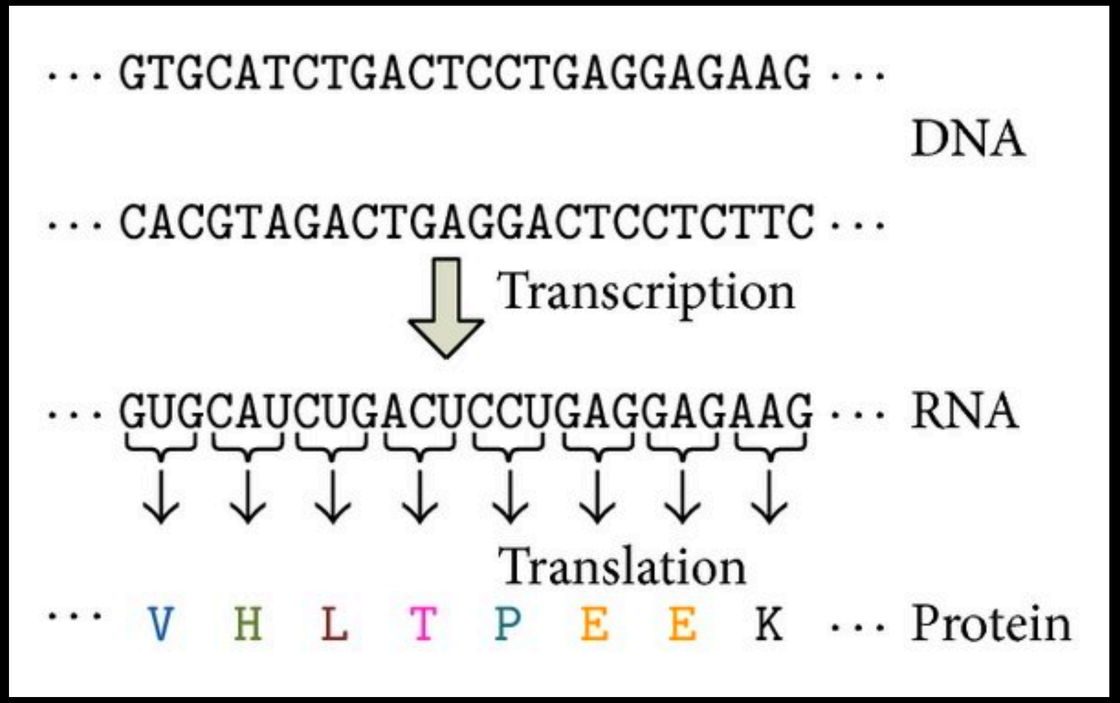
Can we use LLMs to understand/decode DNA Sequences?

- HyenaDNA -> a genomic foundation model
  - DNA Sequences (“language”) encodes a lot of information about a living organism. Mapping DNA instructions to genotypes and phenotypes is a very important and useful task.
  - What is a Foundation Model?
    - A large language model (LLM) trained/pre-trained on a huge number of data modalities (text or image or multimodal) (Example would GPT 4).
      - Why use a FM?
        - Lots of unstructured genomic data available (generator model are a perfect candidate)
        - The “embeddings” can be fine tuned for a number of genomic tasks
  - HyenaDNA proposes the use of DNA sequences as input and a special operator/ operation known as Hyena [Inputs can span unto 100k in length and more
    - Focus on single nucleotide

Gene Loci	Chromosomes	Proteins	Forms of PD and Age Onset	Reference
PARK1	4q21	α-Synuclein (SNCA)	Autosomal dominant, early onset	[5,6,33,69]
PARK2	6q25-27	Parkin	Autosomal recessive, early onset	[5,6,33,69]
PARK3	Unknown	2p13	Autosomal dominant,	[6,33,69]
PARK4	4q21	SNCA	Autosomal dominant, early onset	[6,33,66,69
PARK5	4p14	UCH-L1	Autosomal dominant, idiopathic	[5,6,33,69]
PARK 6	p35–p36	PINK1	Autosomal recessive, early onset	[5,6,33,69]
PARK7	1p36	DJ-1	Autosomal recessive, early onset	[5,6,33,69]
PARK 8	12q12	LRRK2	Autosomal dominant, idiopathic	[5,6,33,66,6
PARK 9	1p36	ATP13A2	Kufor-Rakeb Syndrome, early onset	[6,69]

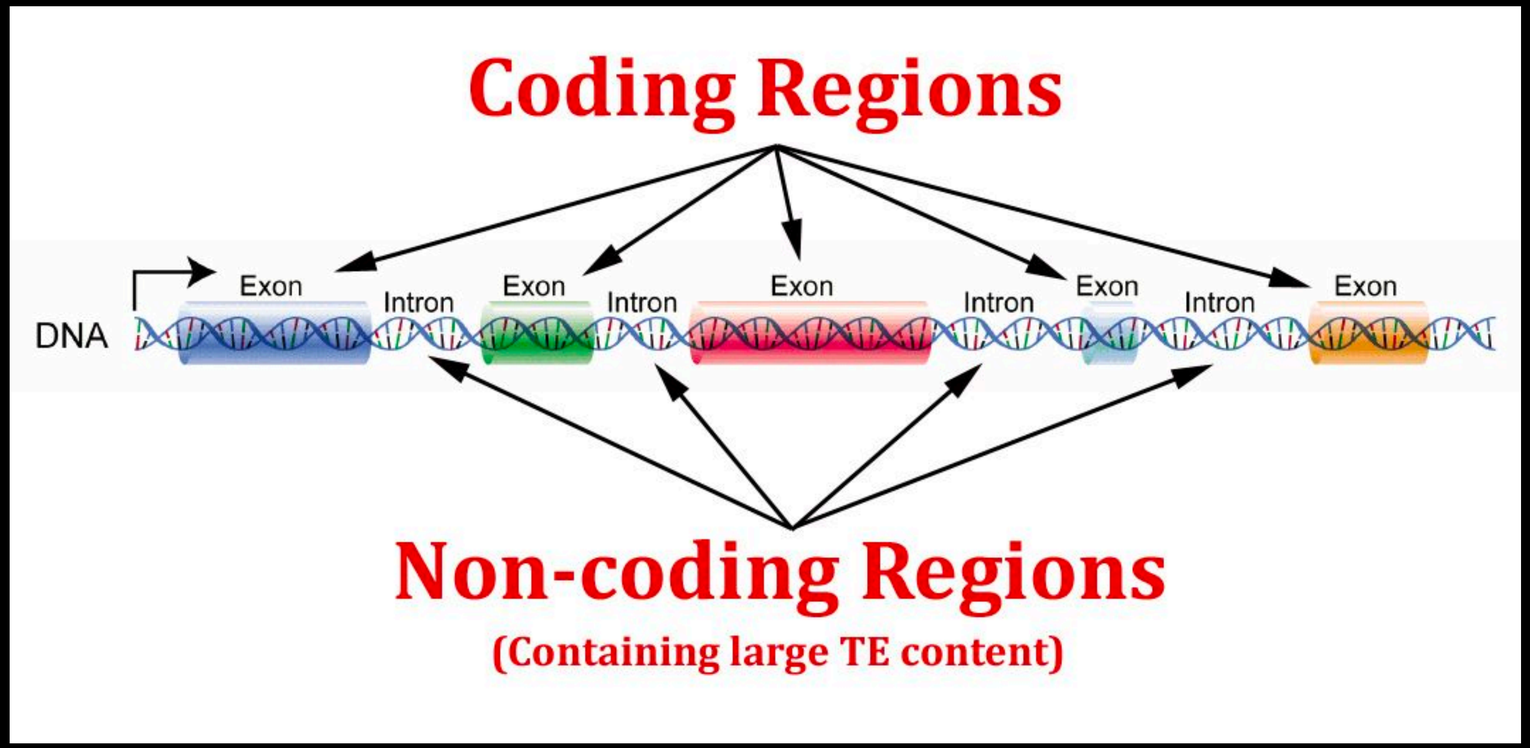
Let’s start with a basic DNA sequences Primer for Computer Scientists

- Over the past few years, the cost sequencing has reduced drastically from $15 million to just $1000. This leads to plethora of new sequencing data
- You can think of a DNA sequence as a string of letters (“A”, “T”, “G”, “C”). These are called nucleotide and can thought as a binary system. DNA Sequencing technologies are used to determine the exact order of these letters.
- A group of 3 letters is called codons (ATG etc). A specific segment of a sequence is called a gene and this is used in the transcription process. Each molecular machine (Think of this as a function(x) where x is the segment of gene) is translated into proteins that powers our body.
    - DNA Sequencing is used to:
        - Identifying changes (or a regulatory region contains differences or mutations)
        - Identifying diseases/phenotypes



Coding and Non-Coding DNA for Computer Scientists

- For a DNA sequence, the coding DNA refers to the DNA that gets transcribed into proteins while Non-Coding DNA does not transcribe into proteins and has different functions such as regulation etc.
- The frequency of coding and non coding is different for each organism (Humans have 1-2% coding DNA while bacterial have 88% as coding DNA)
- Think of Coding DNA as the code that gets compiled (that is actually used) and Non Coding DNA as the code that act as comments or unused code or configuration files etc.

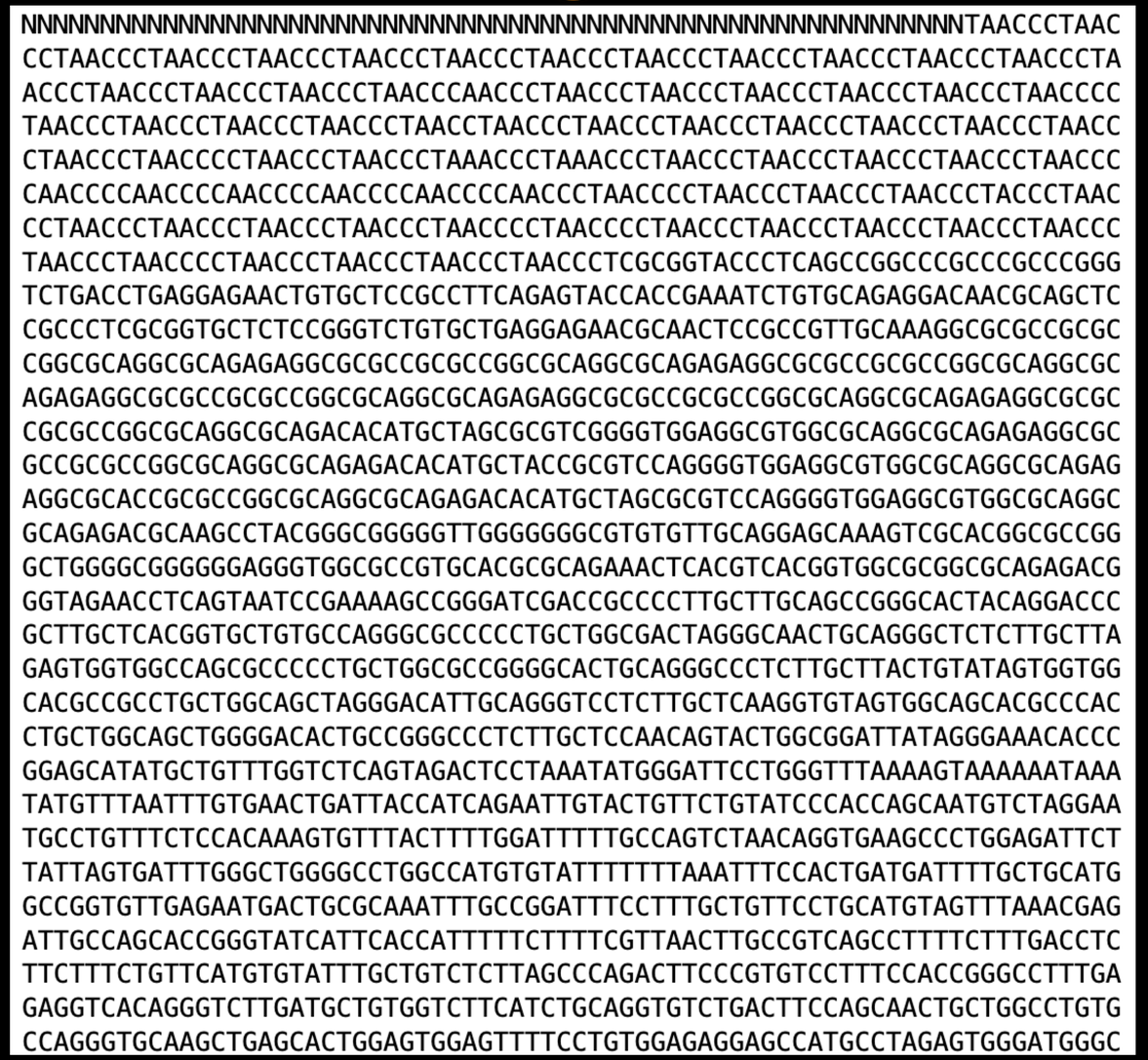


Let’s see how on how to train an LLM from scratch?

- In the most simplest sense, for DNA sequences (and any other LLM) a basic recipe is comprised of the following steps:
  - Step 1: Gather pertaining data (should be diverse and can be of any modality)
  - Step 2: Clean and preprocess the data
  - Step 3: Tokenize the data (converting words into tokens or numeric representations)
  - Step 4: If using an LLM like GPT, use a transformer architecture
  - Step 5: Pre-train to predict the next token/word
  - Step 6 (Optional): Supervised fine tuning or RLHF to get desired answers

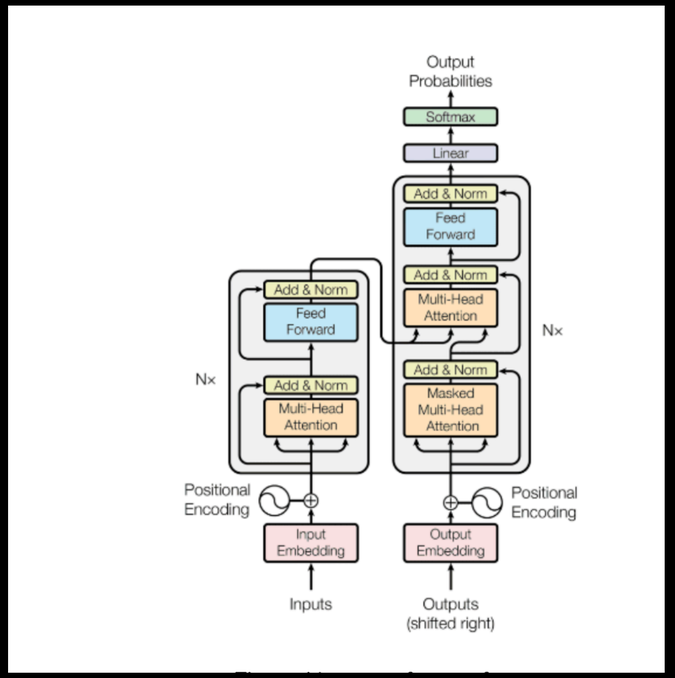
But can’t we do the same for DNA sequences(ATGC)?

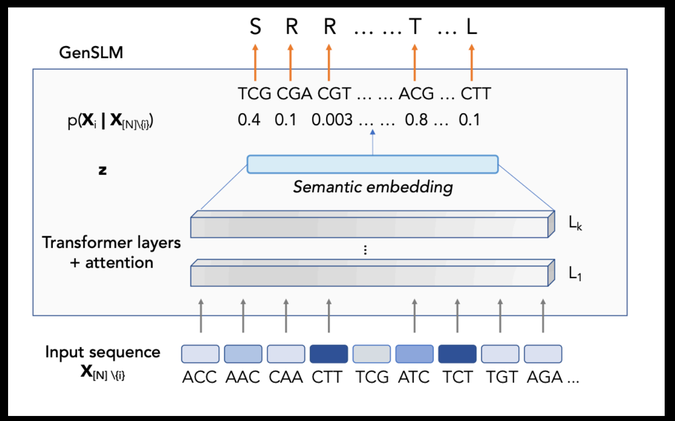
No ☹️



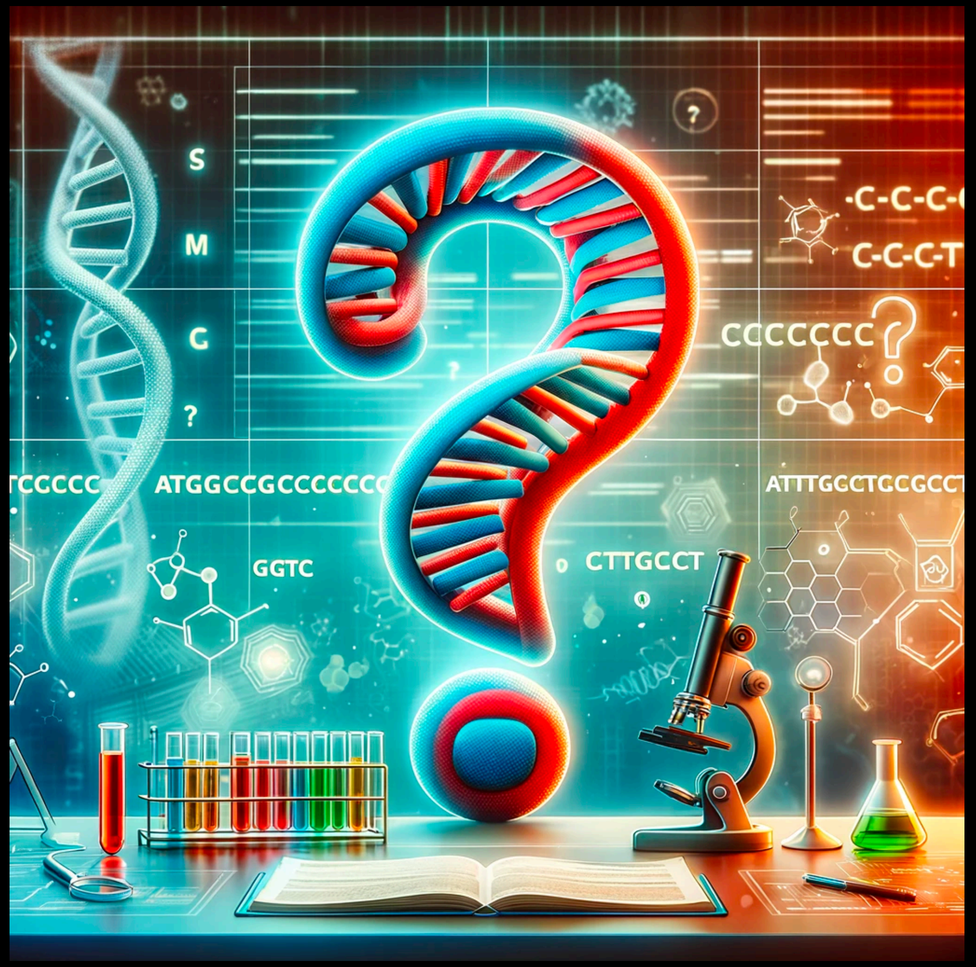
What has already been done?

- Models such as BERT (Bidirectional Encoder Representation Transformers) and other models (attention based) built on protein sequences/DNA sequences have already found success in a niche category of genomic tasks.
    - DNABert: Utilizes BERT for modeling kmers (combinations of single nucleotide)
    - Nucleotide Transformer: Utilizes the transformer model for modeling nucleotides (Context Length is in the range of 512 - 4098 i.e quadratic scaling of attention (n^2))
    - GenSLM: Not trained on the human genome. Limited by the quadric scaling of attention (Utilizes kmers and not single nucleotide)
    - GenaLM: Is significantly better than other LMs but does can handle 36K context





Why?

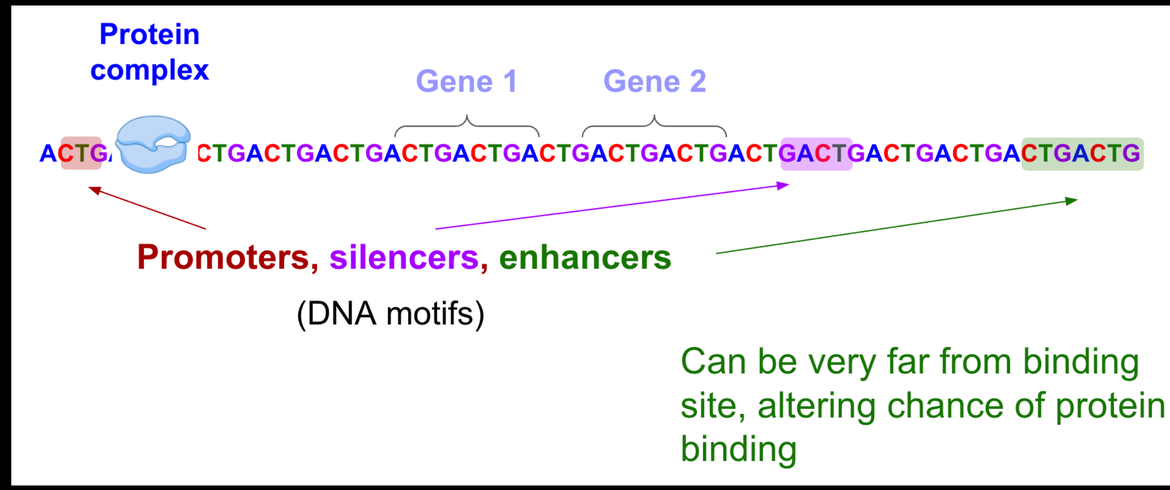


Benefits of modeling long range DNA Sequences and Why not just feed the sequences in a GPT based model and train it on scale?

- Why not just use an attention based transformer?
  - **Problem with the context length:** The problem with this approach is that it scales upto 512 - 4096 token (context). DNA sequences are much larger (up till 100K).
    - Attention based methods are quadratric. Is there any way to make it quadratic?
  - **Problem with the tokenization approach**: Existing approaches take into account kmers (fixed set of sequences) and not a single nucleotide (single DNA sequence). This results in a lot of information loss (such as mutation changes etc)
    - If we can make the process sub-quadratic, can we use a different tokenization approach that utilizes single nucleotide or letter resolution?

DNA sequences as a combinatoric problem

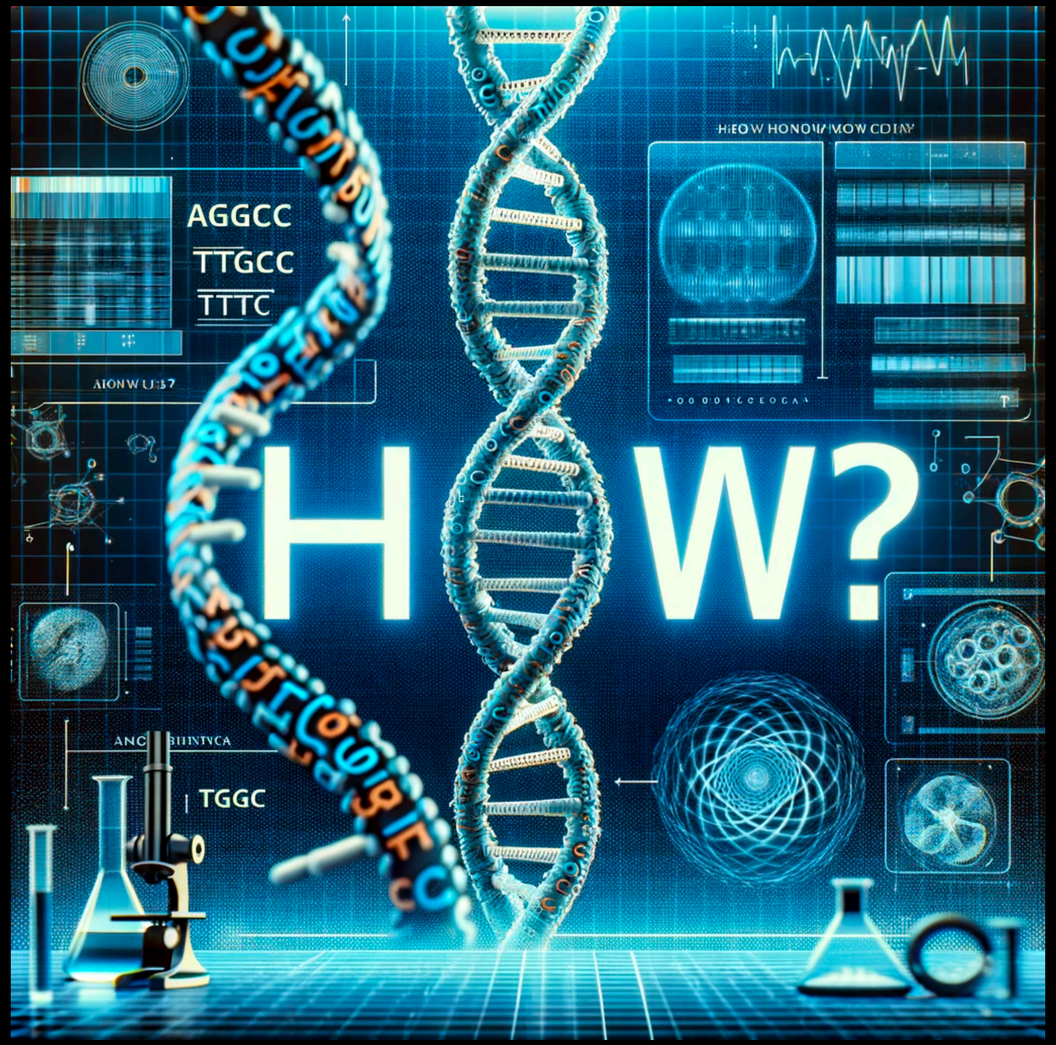
- In the most simplest sense, let's think of DNA as a combinatoric problem.
  - For a DNA sequence, each cell copies the DNA (heart, brain etc). Each cell contains the same DNA sequence/instructions. The reason a heart or a brain acts as it should is because of "gene regulation". Now for a function to be performed, a specific set of genes are turned on "expressed". For a liver cell to remove toxic substance, the genes that perform this function are turned on and similarly for other cells.
  - Think of this in a similar fashion to how a program works. Gene regulation is similar to a program's if-else or a loop determining where and how the algorithm runs.
    - Let's think of a DNA sequence as a class (Object Oriented Programming) and and the instantiation process of theses classes can be thought of as gene regulation. Now in a similar fashion, just as classes contain a lot of methods or properties that might not be used but are essential for the instantiation process, the non coding DNA affects the gene regulation.
  - How is this a combinatoric problem?
    - For gene expression to begin, a protein complex called RNA polymerase must bind to the DNA at specific sites. The probability of RNA polymerase binding is influenced by DNA motifs, which are specific sequence patterns found in the non-coding regions
    - Why not just build a classifier?
      - No data (Don't know what may cause gene regulation)
      - Manual work i.e time consuming and human genome is huge
      - Not computationally feasible



Solution to Long range nucleotide language modeling as a solution to CP

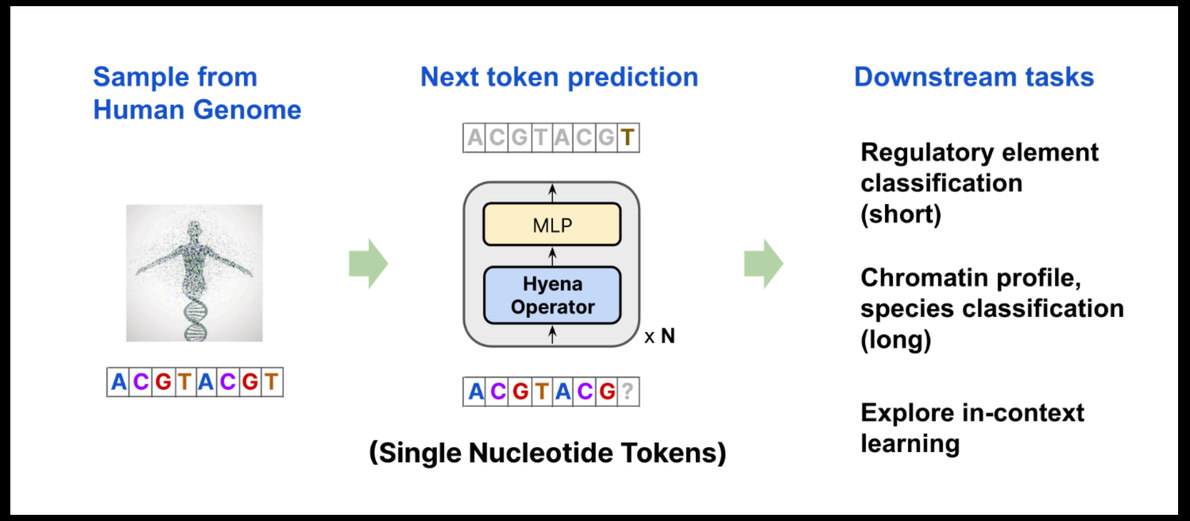
- The DNA motifs and what it binds to (Long or short range interactions can be thought of as a combinatoric process). Thus modeling nucleotide resolution is extremely important.
  - Existing methods:
    - Does not take into account single nucleotide
    - Attention based methods scale quadratically so it’s impossible to consider a single letter as input or do next word prediction with DNA sequences.
    - **Solution: A drop in replace for attention blocks known as Hyena (Hyena Operator)**

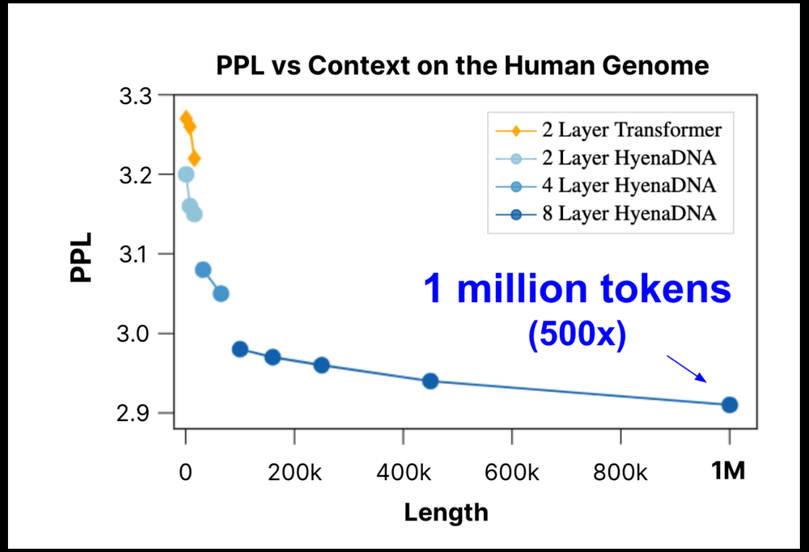
How?



Step-by-step process to train HyenaDNA

- Step 1: Gather a single human reference genome
- Step 2: Create a tokenizer capable of handling single nucleotide resolution
    - A vocab of 4 letters and special tokens (start, end etc) (A, G, C, T, N)
- Step 3: Swap all attention blocks with the Hyena operator
    - Instead of computing A(x) directly, use a for loop and compute projections (convolution filter) using a for loop. This results in sub-quadratic complexity hence allowing a larger context (NLogN).
        - Loss Function: Cross Entropy
        - Techniques: Early Stopping, Gradient Checkpointing
            - Layers: 2 - 8 layers
            - 126 - 256 dimension (embedding size)
            - Sequence length ranging from 1024 - 1Million
            - Optimizer: AdamW
            - LR: 1.5 - 6e-4
            - Batch Size: 64 - 256
            - Total Params: 400k - 6.6M
- Step 4: Pre-train to predict next word resolution
- Note: The model was trained on short sequences first and then gradually increase sequence length (lower the Perplexity, the better the LM)





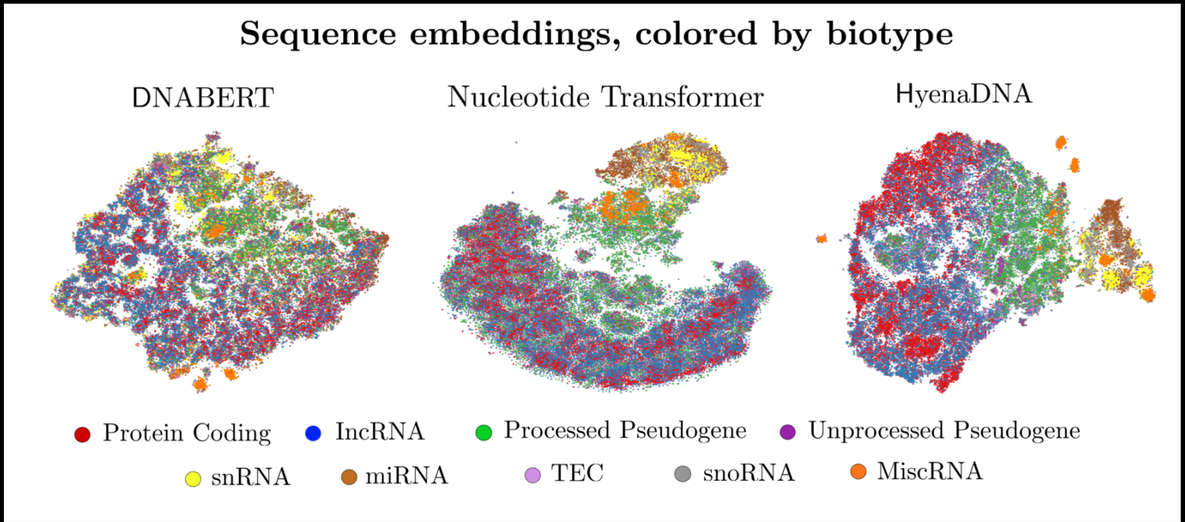
Possible limitations of the method proposed

- Pretraining has been done on a single human reference genome so that as useful
  - Increasing species diversity can allow the model to express emergent capabilities
  - Adding protein and chemical sequences can also enhance the model
    - Possible solution: Build a diverse genetic dataset utilizing high throughput sequencing data
- Scaling law has not been explored (The effect of increasing the model size)
- Due to the unavailability of a lot more genomic benchmarks, we cannot generalize the capability, additionally the lack of understanding of the DNA sequences

How do we evaluate this?

Evaluation

- Embeddings can be an effective way to evaluate if the model is actually discriminating the features accurately.
- HyenaDNA outperforms most genomic prediction tasks and and is **160x** faster than the baseline (Transformers)



Dataset	CNN	Transformer	HyenaDNA
Mouse Enhancers	69.0	80.1	84.3 (+15.3)
Coding vs Intergenomic	87.6	88.8	87.6 (+3.5)
Human vs Worm	93.0	95.6	96.5 (+3.5)
Human Enhancers Cohn	69.5	70.5	73.8 (+4.3)
Human Enhancers Ensembl	68.9	83.5	89.2 (+20.3)
Human Regulatory	93.3	91.5	93.8 (+0.5)
Human Nontata Promoters	84.6	87.7	96.6 (+12)
Human OCR Ensembl	68.0	73.0	80.9 (+12.9)

How to apply this?

Application: Prediction of chromatin profiles and epigenetic markers from DNA sequences

- Epigenetics factors change the DNA by external factors. Basically genes are silenced by environmental factors in a human genome (Diseases include Cancer etc)
- The base model “HyenaDNA”, fine tuned with 919 features was used to predict the changes.
  - By adding an extra layer, using a pretrained HyenaDNA model, the model is fine-tuned to predict the epigenetic and chromatin changes (Classification task).
- **Note**: This is an interesting problem because it occurs at a single nucleotide level.

DNA Sequencing Artifacts Classification

- Given the sequence (200 characters long), can we fine tune a HyenaDNA network to classify the labels?
  - Why do this?
    - Pre-trained networks contain a lot of granular features about the pertained texts
      - As seen by existing GPT networks trained on English Language
      - Utilizing traditional fine tuning methods
        - Variable sequence length
        - More flexibility (Can do binary classification/ multi-class all with the same model)

	labels	CHROM	POS	REF	ALT	sequence
2755949	1	chr18	55788098	A	G	GTTGTGGCATAGAATACGTGCTTAAGAAAAGTTCTTTTAGTATGAA...
2215798	1	chr12	95101483	G	T	GCTGTTCATTACACCTGTCTGGTACTGATCTGGAAATCATGAACAT...
1362553	1	chr7	28914667	A	G	GAGAATTTTAAAGATTCCTCTTACCTCCCCACTGCAAGATCCTGTT...
1880082	1	chr10	71683900	G	T	TTAAAACTGACCTGGGCTGGGCGCGGTAATCCCAGCACCCTGGGAG...
2379984	1	chr13	109536210	C	T	GCTGCCTCGCTTAGAGAGGGGAGGGCTGCAGACGGGGCAACTTACA...
...	...	...	...	...	...	...

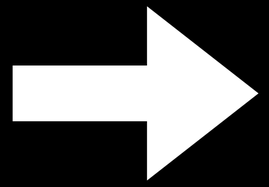
```
Train Epoch: 0 [0/2467785 (0%)] Loss: 0.954370
Train Epoch: 0 [2560/2467785 (0%)]      Loss: 0.117834
Train Epoch: 0 [5120/2467785 (0%)]      Loss: 0.152483
Train Epoch: 0 [7680/2467785 (0%)]      Loss: 0.139049
Train Epoch: 0 [10240/2467785 (0%)]     Loss: 0.068685
Train Epoch: 0 [12800/2467785 (1%)]     Loss: 0.081760
Train Epoch: 0 [15360/2467785 (1%)]     Loss: 0.154833
Train Epoch: 0 [17920/2467785 (1%)]     Loss: 0.122528
Train Epoch: 0 [20480/2467785 (1%)]     Loss: 0.115277
Train Epoch: 0 [23040/2467785 (1%)]     Loss: 0.106830
Train Epoch: 0 [25600/2467785 (1%)]     Loss: 0.151920
Train Epoch: 0 [28160/2467785 (1%)]     Loss: 0.098997
Train Epoch: 0 [30720/2467785 (1%)]     Loss: 0.096300
```

- Current problems
  - Loss not decreasing
    - Need to utilize a new model with more params
      - Change objective function or tokenizer
    - Increase sequence length
  - Use PEFT to lower memory consumption

Thanks! QA?

Appendix

$$y = A(x)x$$



Attention based mechanism

Algorithm 1 Projection

Require: Input sequence $u \in \mathbb{R}^{L \times D}$

1. In parallel across L: $\hat{z} = \mathsf{Linear}(u)$, $\mathsf{Linear} : \mathbb{R}^{D} \to \mathbb{R}^{(N+1)D}$
2. In parallel across D: $z = \mathsf{DepthwiseConv1d}(h, \hat{z})$, h is a short convolution filter
3. Reshape and split z into $x^1, \dots, x^N, v$. Dimensions of one element are $x^n \in \mathbb{R}^{D \times L}$

Return $x^1, \dots, x^N, v$, x^n

Algorithm 2 Hyena Filter

Require: Sequence length L, positional embedding dimension D_e

1. $t = \mathsf{PositionalEncoding}(L)$, $t \in \mathbb{R}^{L \times D_e}$
2. In parallel across N, L: $\hat{h} = \mathsf{FFN}(t)$, $\mathsf{FFN} : \mathbb{R}^{D_e} \to \mathbb{R}^{ND}$, $\hat{h} \in \mathbb{R}^{L \times ND}$
3. Reshape to $\hat{h} \in \mathbb{R}^{N \times D \times L}$
4. $h = \hat{h} \cdot \mathsf{Window}(t)$, $h \in \mathbb{R}^{N \times D \times L}$
5. Split h into $h^1, \dots, h^N$

Return $h^1, \dots, h^N$

Algorithm 3 Forward pass of Hyena

Require: Input sequence $u \in \mathbb{R}^{L \times D}$, order N, model width D, sequence length L, positional embedding dimension D_e

1. $x^1, \dots, x^N, v = \mathsf{Projection}(u)$
2. $h^1, \dots, h^N = \mathsf{HyenaFilter}(L, D_e)$

for $n = 1, \dots, N$ **do**

3. In parallel across D: $v_t \leftarrow x_t^n \cdot \mathsf{FFTConv}(h^n, v)_t$

end for

Return $y = v$

```
x, v = input_projections(u)
for o in range(hyena_orders):
    h = hyena_filter(L)  # long conv filter parameterized via an MLP
    v = x[o] * fftconv(h, v)  # elem-wise mult & fftconv
)
```

Hyena operating with parallel linear projections and convolutions resulting in a significant boost in latency