Anatomy of a Typical Genomic Foundation Model

Ikram Ullah, Staff Scientist

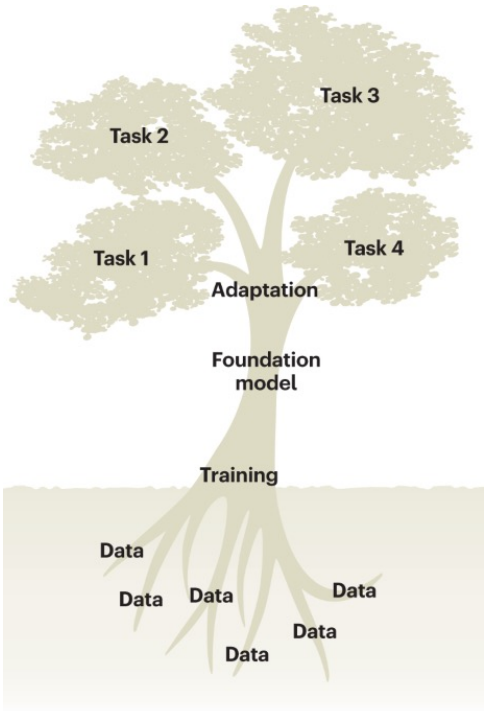


Image taken from – Tang, Lin. "Large models for genomics." *Nature Methods* 20.12 (2023): 1868-1868.

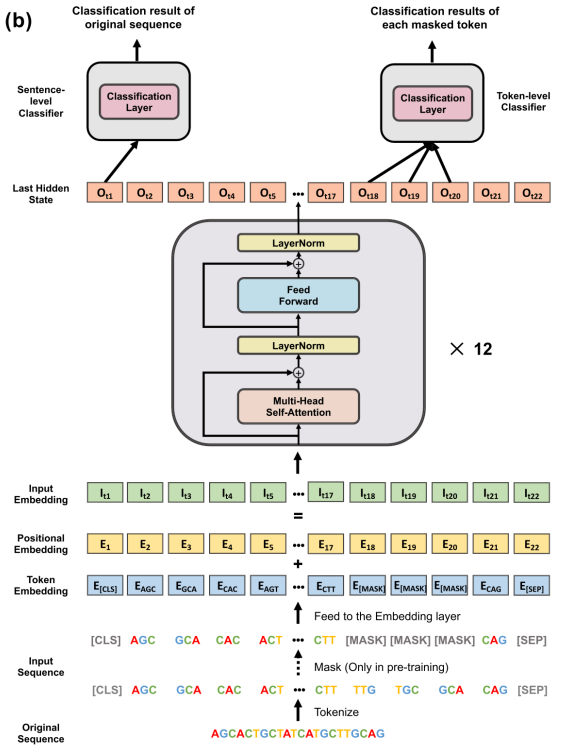
Agenda

- Overview
- Sequence Tokenization
- Embedding Layer
- Neural Network Model
- Output Layer
- Common Training Strategies
- Practical: Running a Genomic Foundation Mode

Cui, Haotian, et al. "scGPT: toward building a foundation model for single-cell multi-omics using generative AI." *Nature Methods* 21.8 (2024): 1470-1480.

Anatomy of a typical Genomic Foundation Model

DNA Sequence (Input)	The raw genomic sequence data	Foundation for all analysis
Tokenization	Breaking down raw DNA sequence into discrete units called tokens	Single nucleotide, kmer, BPE
Embedding Layer	Converting tokens into a dense vector representation	Capturing semantic meaning and context of token within sequence
Neural Network Model	Core of the DNA Foundation Model, usually based on Transformer architecture (encoder, decoder, or both)	Uses mechanisms like self-attention to learn complex relationships and dependencies between tokens
Output Layer (Task-Specific)	Language Modeling Head: For predicting masked tokens during pretraining or generating new DNA sequences	Regression/Classification Head: For prediction tasks



Ji, Yanrong, et al. "DNABERT: pre-trained Bidirectional Encoder Representations from Transformers model for DNA-language in genome." Bioinformatics 37.15 (2021): 2112-2120.

DNA as a Language – Tokenization

Breaking down the continuous DNA sequence into discrete units called tokens

k-mer Tokenization

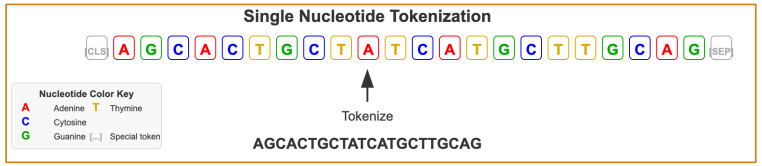
- Dividing the sequence into overlapping or non-overlapping subsequences of length k, Biologically interpretable but fixed length
- Used in DNABERT, Nucleotide Transformer

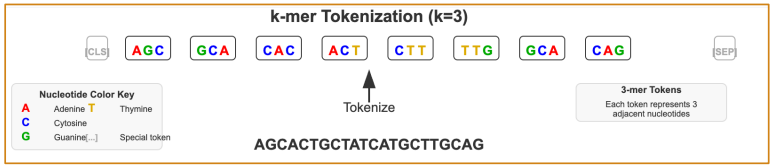
Single Nucleotide Tokenization

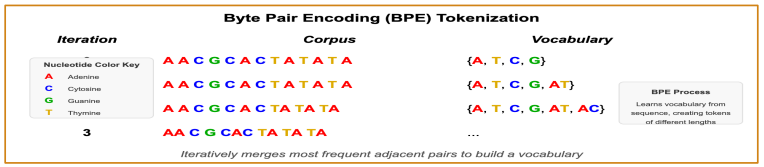
- Treating each nucleotide (A, C, G, T) as an individual token, long context but high computational cost
- Used in HyenaDNA

Byte Pair Encoding (BPE)

- Iterative algorithm that searches for nucleotides combinations and builds the vocabulary at the same time. Capture longer motifs but large memory requirements
- Used in DNABERT-2

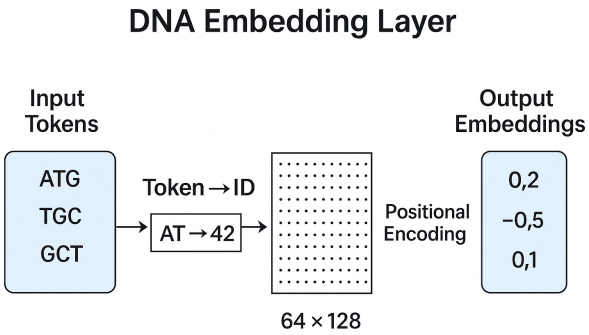






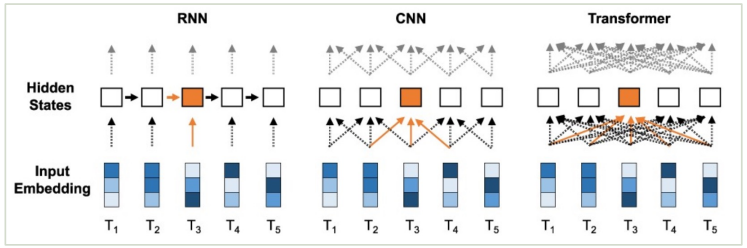
Embedding Layer

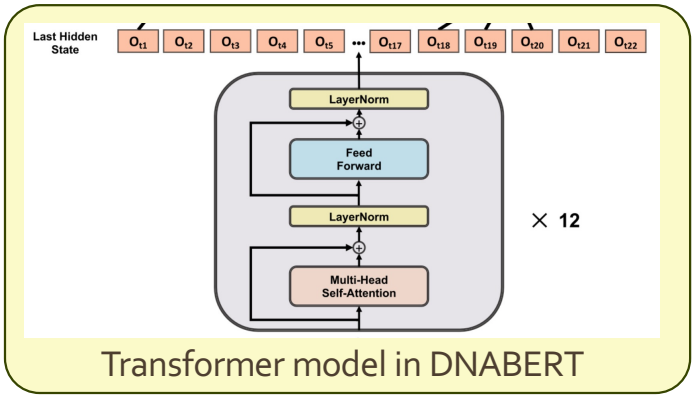
- Input
  - DNA sequence split into overlapping 6-letter substrings
  - Vocabulary size $4^3 = 64$ unique tokens
  - Example: ATGTGCGCT → [ATG, TGC, GCT]
- Embedding Process
  - Token → ID: Each 3-mer mapped to an integer ID, e.g., ATG → 42
  - Embedding Lookup: Each ID looks up its corresponding 128-dimensional dense vector from the 64×128 embedding matrix (different than one-hot encoding)
  - Positional Encoding: Adds order info to embeddings (e.g., sinusoidal or learned)
- Output Embeddings
  - If the input was 32 k-mer, token is now a 64-dim vector, e.g., [0.2, -0.5, …, 0.1]
  - After training, semantically or biologically similar k-mers cluster in vector space
  - (e.g., ATG ≈ ATT)



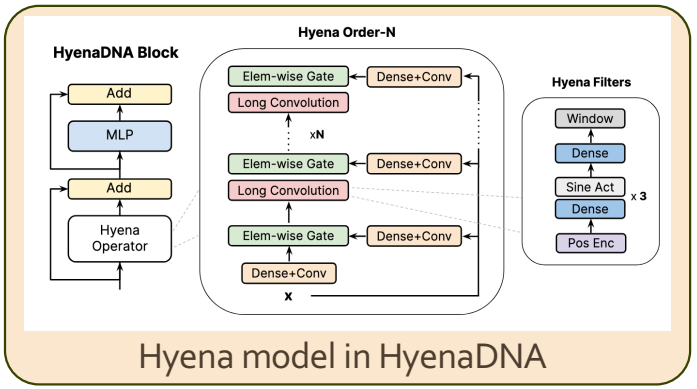
Model Architecture Layer

- The core of the DNA Foundation Model
- Often based on the Transformer architecture
  - Self-attention for learning complex relationships and dependencies between tokens
  - Optimizations: Sliding window (Longformer) CUDA-based (FlashAttention)
- Newer models uses architectures like CNN, Hyena and Mamba
  - Necessitated by local pattern learning and long context handling





Transformer model in DNABERT



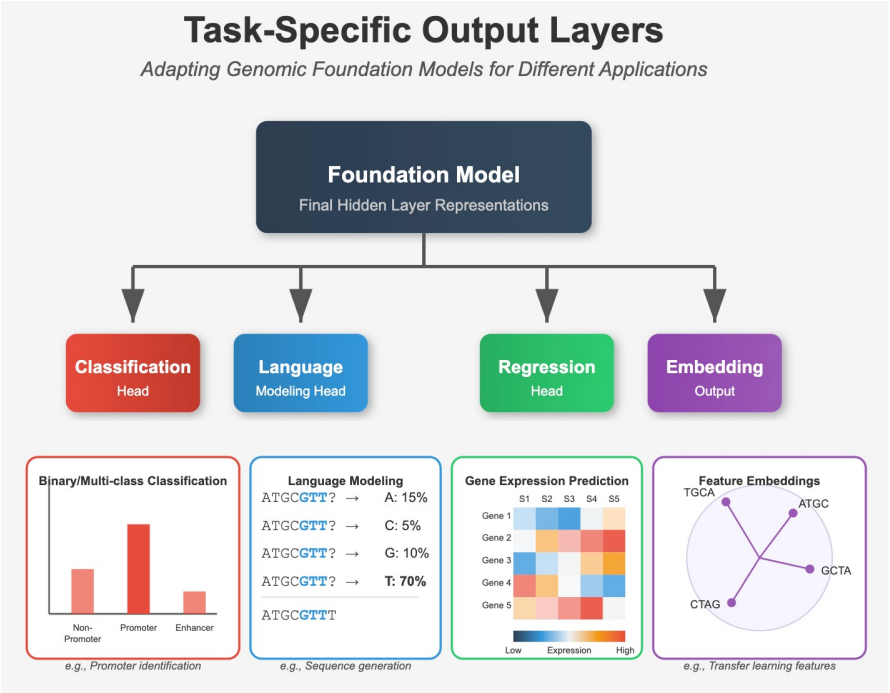
Hyena model in HyenaDNA

Beltagy, Iz, Matthew E. Peters, and Arman Cohan. "Longformer: The long-document transformer." arXiv preprint arXiv:2004.05150 (2020).

Dao, Tri, et al. "Flashattention: Fast and memory-efficient exact attention with io-awareness." *Advances in neural information processing systems* 35 (2022): 16344-16359.

Output Layer (Task-Specific)

- The final layer of the model, which is adapted based on the specific task the GFM is designed for.
  - **Language Modeling Head**: For predicting masked tokens during pretraining or generating new DNA sequences
  - **Classification Head**: For predicting categories (e.g., promoter vs. non-promoter)
  - **Regression Head**: For predicting continuous values (e.g., gene expression levels)
  - **Embedding Output**: The learned representations can be used as features for other downstream analyses.



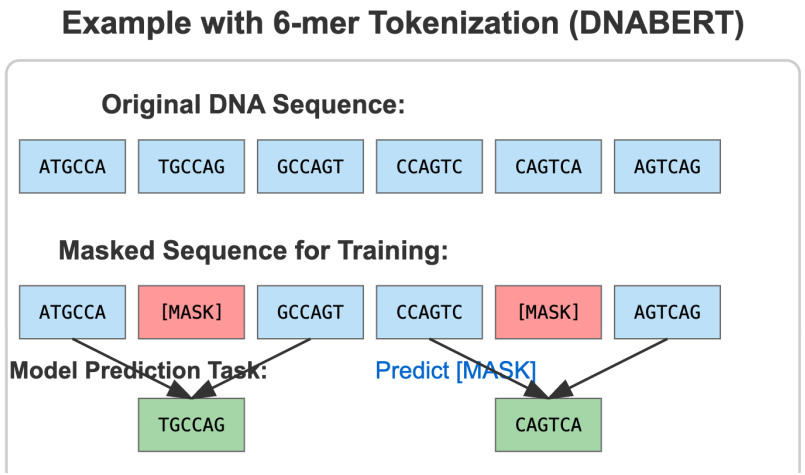
Training Strategies for Genomic Foundation Models

- Genomic sequences lack explicit grammar — models must learn regulatory syntax and structure from raw sequences
- Choice of training objective determines:
  - What the model learns, e.g., sequence motifs (MLM), positional dependencies (AR)
  - How well it generalizes to downstream tasks (e.g., enhancer detection, variant effect prediction)

Strategy	Description	Model Type	Example Use Case
Masked Language Modeling (MLM)	Randomly mask parts of the input and train the model to predict them using full context	Transformer Encoder (BERT-style) DNABERT	Promoter prediction
Autoregressive Modeling (AR)	Predict the next token in the sequence based only on previous tokens	Transformer Decoder (GPT-style) HyenaDNA	Genome sequence generation

Masked Language Modeling: Capturing Local Regulatory Syntax

- How MLM Works
  - Randomly mask ~15% of tokens in DNA sequences
  - Train model to predict the masked elements using bidirectional context
  - Used primarily in transformer encoder architectures (BERT-style)
- Key Advantages
  - Bidirectional context, efficient transfer learning
- Applications
  - Promoter prediction, variant effect prediction

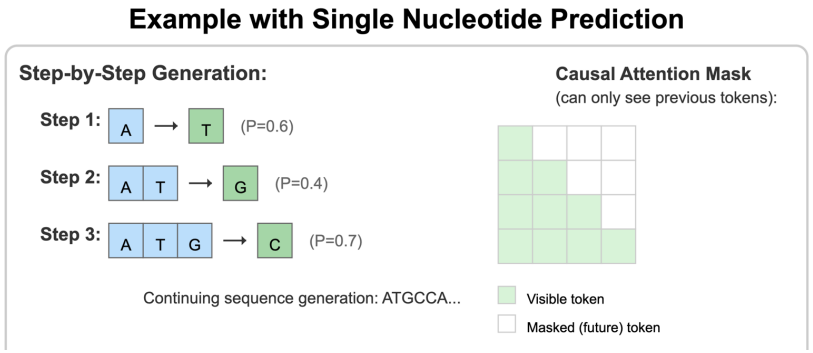


Predict missing kmer based on surrounding context

```
from transformers import (
    BertTokenizer,
    BertForMaskedLM,
DataCollatorForLanguageModeling,
    BertConfig,
    Trainer,
    TrainingArguments
)
```

Autoregressive Modeling: Sequence Generation and Contextual Dependencies

- How Autoregressive Pre-training Works
  - Predict each token based only on previous tokens in the sequence
  - Train model step-by-step to predict the next element in sequence
  - Used primarily in transformer decoder architectures (GPT-style)
- Key Advantages
  - Sequence generation, conditional synthesis
- Applications
  - Synthetic biology, enhancer design, genome generation




```
from transformers import
AutoModelForCausalLM,
AutoTokenizer
```

Questions & Comments