Understanding the natural language of DNA using encoder–decoder foundation models with byte level precision

Kaixi Xu
Ziyang Mao

Abstract

Summary:

- This study proposes ENBED (Ensemble Nucleotide Byte-level Encoder-Decoder), a Transformer-based foundation model for genomic sequence analysis.
- ENBED uses byte-level tokenization and a sequence-to-sequence encoder–decoder architecture to analyze DNA sequences with single-nucleotide resolution.
- It applies masked language modeling for pretraining and a subquadratic attention mechanism for scalable long-range modeling.

Key Contributions of ENBED

Core Applications:

1. Identifies enhancers, promoters, and splice sites
2. Detects sequencing noise (insertions, deletions, mismatches)
3. Annotates biological functions of sequences
4. Predicts viral mutations (e.g., Influenza)

Advantages:

Outperforms encoder-only / decoder-only models

Strong generalization across species and tasks

The Role of Foundation Models in Bioinformatics

Foundation models have revolutionized natural language understanding and are now entering bioinformatics.

Their generalizability and adaptability allow them to perform well on unlabeled biological data.

Early applications include:

- Protein structure prediction
- Mutation effect analysis

Recent models expand to DNA and RNA:

- Identifying regulatory elements
- Analyzing chromatin and evolution
- Predicting mutations from genomic data

Prior Work: Encoder-only vs Decoder-only

Transformer-based models for DNA fall into two categories:

- Encoder-only models (e.g., BERT-style)

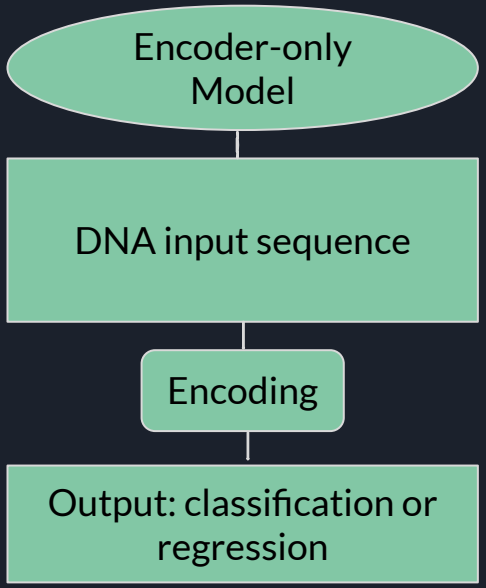
  → Good for classification, regression tasks

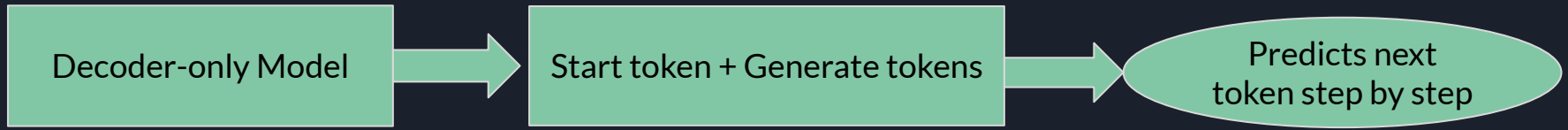
  → Lack generative ability

- Decoder-only models (e.g., GPT-style)

  → Can do sequence generation

  → But limited in capturing source–target relationships

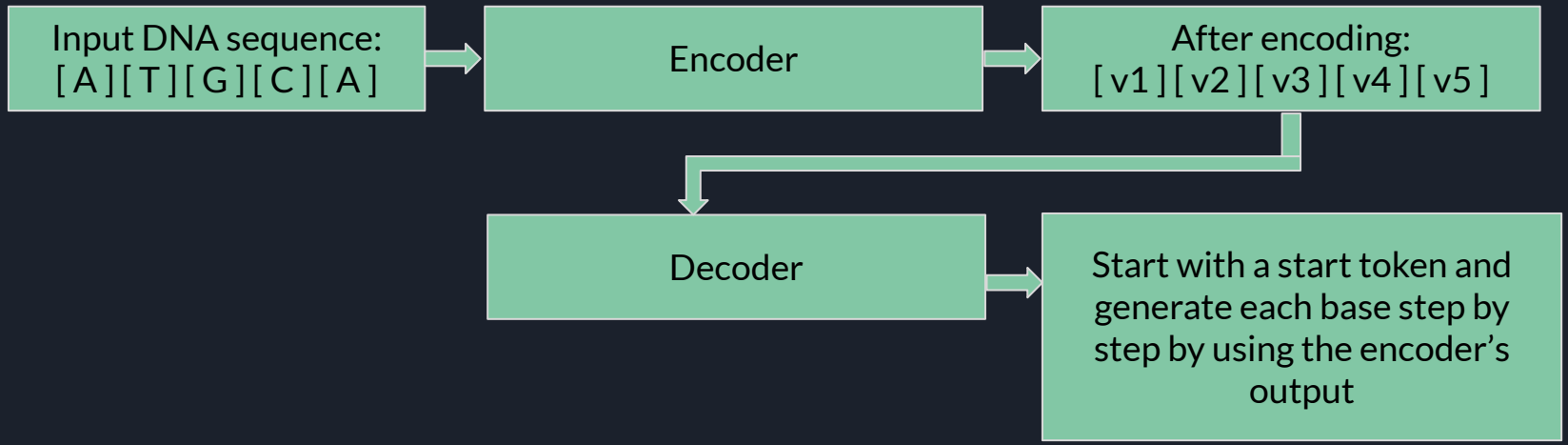




Why ENBED Uses Encoder–Decoder Architecture

Benefits of encoder–decoder structure:

- Handles input and output sequences asymmetrically (more biologically realistic)
- Uses cross-attention to link encoded input with generated output
- Performs better on sequence-to-sequence tasks like mutation prediction



Limitations of Traditional Tokenization Methods for DNA

- DNA sequences are made of 4 symbols (A, C, T, G).
- Tokenizers are needed to convert sequences into Transformer-readable tokens.
- Traditional NLP tokenization (e.g., SentencePiece, BPE) assumes linguistic structure, which DNA lacks.
- DNA has no words, spaces, or punctuation → token boundaries are unclear.
- Methods like k-mer, BPE, SentencePiece:
  - Create variable-length tokens
  - Are sensitive to small changes: one base change → completely different token
- ENBED adopts byte-level tokenization:
  - Each base is a single token (A, T, C, G)
  - Sequence becomes longer, but more robust to mutations

ENBED: A Foundation Model for DNA Sequences

- ENBED = Ensemble Nucleotide Byte-level Encoder–Decoder
- Uses a Transformer with encoder–decoder architecture
- Applies byte-level tokenization (1 token per nucleotide)
- Enables sequence-to-sequence modeling for DNA

Pretrained on:

Human & maize telomere-to-telomere genomes

Key Design Choices in ENBED

- Byte-level tokenization:
  - Avoids ambiguity in defining token boundaries in DNA
  - Handles single nucleotide variants (SNVs) robustly
  - Downside: Longer token sequences → higher compute cost

- Subquadratic attention:
  - Reduces compute cost of long DNA sequence modeling
  - Maintains performance by combining sliding window + global attention

ENBED Evaluation: Benchmarks and Noise Detection

Performance on Genomic Benchmarks :

- Outperforms State-of-the-Art on 21/25 tasks from NT & GB datasets
- Tasks include: enhancer/promoter/splice site/histone mark prediction
- Datasets span human, mouse, yeast, fly, worm DNA

Sequencing Noise Identification :

- ENBED distinguishes noisy vs accurate sequences
- Uses data from telomere-to-telomere assemblies
- Achieves 97.6% accuracy on synthetic noisy data

ENBED Evaluation: Biological Function & Mutations

Biological Function Annotation

Predicts gene function classes from genomic sequences

After tuning achieves F_1 score = 74.1 on common functional classes

Mutation Prediction (Seq2Seq) :

ENBED is also applied to simulate and predict genetic mutations.

It uses an encoder–decoder architecture, enabling the model to generate mutated sequences step by step, similar to language translation.

The experiments use Influenza virus data from the NCBI Influenza Virus Resource, where parent–child pairs of mutated sequences are constructed based on a phylogenetic tree.

ENBED demonstrates strong performance in modeling genomic mutations.

2 Method

1. model architecture
2. Input sequence Tokenization
3. Attention layer
4. Application of foundation model
5. Benchmarks and tasks

2.1 Model architecture

- ENBED consists of encoder and decoder blocks.
- Both encoder and decoder blocks contain **attention layers** and **feed-forward neural networks**
- Attention layers: process a sequence by replacing each element with a **weighted sum of** input embeddings
- Feed-forward neural network: normalize and pass-through the input sequence, dropout applied

The model is encoder-heavy

- encoder-to-decoder ratio to 2:1, 1% performance increase in Maked Language Modeling accuracy for ByT5(another architecture which is 3:1 encoder-to-decoder ratio)

- The reason why **encoder-heavy**: tokens are better encoded, because a larger share of parameters to these blocks(encoder and decoder).

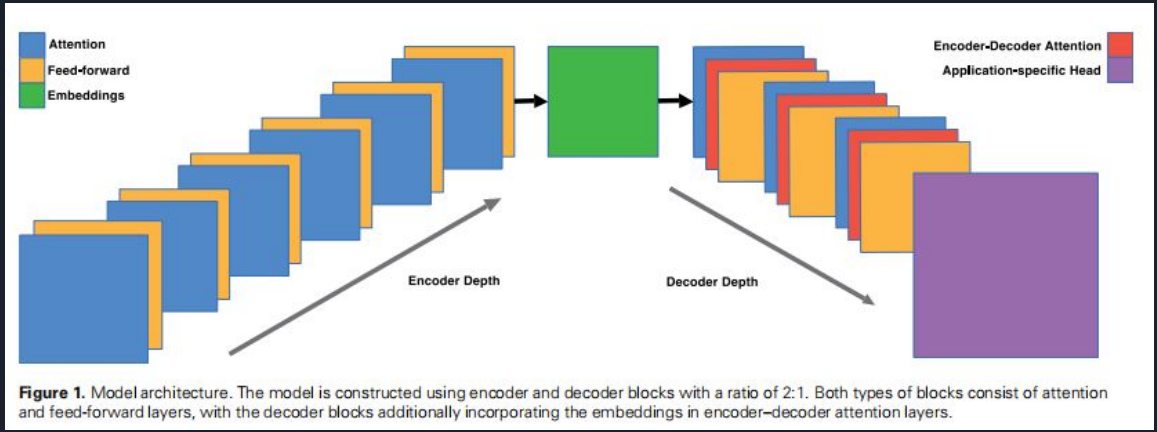


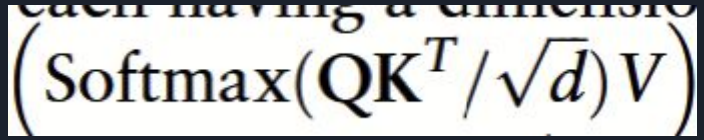
Figure 1. Model architecture. The model is constructed using encoder and decoder blocks with a ratio of 2:1. Both types of blocks consist of attention and feed-forward layers, with the decoder blocks additionally incorporating the embeddings in encoder–decoder attention layers.

2.2 Input sequence tokenization

- Sequence are tokenized, by breaking down the input into token. Each token consists of s ingle nucleotide
- Even though Alphabet only contains A,T,C,G, authors still keep whole set of ASCII chars, because it can aid future tasks like sequence-to-sequence transformation
- This tokenization method require more operations ,since it increases tokenized sequence length for DNA sequence
- Therefore we need to reduce memory by reducing the complexity of attention layers

2.3 Attention layer

- It can be seen as a soft-lookup of a query Q in a dictionary(Key value pairs K V).
- The attention score: the similarity between Q and K, each has dimension d.



However, there is a challenge regarding attention !!!

Challenge about Attention

- It is hard to increase the sequence length L, since the complexity is O(L^2), L<512 is a limit
- In order to reduce the complexity, author modified the architecture by replacing the dense attention with two subquadratic variants of attention:
- (i) sliding-window attention
- (ii) global attention.

Types of Attention

Sliding-window attention

- Tokens within a sliding window of radius r are used to calculate the attention scores, bringing the complexity down to $O(L \times r)$.

Global attention

- Divide the input sequence into k blocks and calculate a global token by summing the embedding for every token in the block
- Scores are computed for every input token by letting it add to the neighboring token, the complexity is $O(L \times (r+k))$

2.4 Applications of foundation models using transfer learning

- Our first step is to pretrain it on high-quality reference sequences, this step uses Masked Language Modeling

- MLM objective:
  Reconstruct tokens that have been deleted and replaced with a MASK token
  Identify the correct element that belong in the masked segment

- Optimizer: **AdamW**

- **Learning rate:** 1e-5.

- **Loss function:** Cross-entropy.

- **Activation function:** Softmax.

Fine-tuning for downstream tasks

- The model is fine-tuned by adding a final layer into a task-specific configuration.
- This final layer is called ‘head’ and is attached to final layer
- Layers are gradually unfrozen in reverse order during fine-tuning.
- Unfrozen: Transformer to integrate with the attached head while remaining frozen the initial layer
- There are two types of heads:
  Classification head
  Language modeling head

Classification head

- A fully connected (dense) layer is added to the output of the base model, called classification head
- A softmax activation is applied to the output layer, to produce class probabilities.
- Commonly used for sequence-level classification tasks.

Language modeling head

- Consists of a single feed forward neural network layer, followed by a softmax activation function.
- Takes hidden representations from previous layers as input, and outputs a probability distribution over the vocabulary.
- The softmax function converts raw scores into probabilities, representing the likelihood of each token at a specific position.

2.5 Application domains

Use GB(Genomic benchmarks) and NT(Nucleotide transformer) Benchmarks:Evaluate performance on fundamental sequence classification tasks.

Noise Identification Task:Measures ENBED’s ability to distinguish genuine sequences from artifacts.

Biological Function Annotation Task:Tests ENBED’s ability to associate sequence patterns

Mutation Generation Task: Focuses on predicting viral mutations through sequence-to-sequence modeling.

Benchmarks

2.5.1 Genomic benchmarks

The dataset consists of sequences from four organisms:Human, Mouse, Roundworm, Fruit fly

2.5.2 Nucleotide transformer benchmarks

The NT benchmarks consist of four datasets, Epigenetic Marks in Yeast Genome, Enhancer Dataset, Promoter Sequences, Splice Site Dataset

Noise identification

Synthetic dataset generated using 512-nucleotide segments randomly selected from TeloBase (A comprehensive database of telomere motif diversity)

Noise in shallow sequencing data is well-approximated by a negative binomial distribution.

Balanced dataset created with both positive (clean) and negative (noisy) samples.

Model learns to differentiate genuine genomic sequences from noisy artifacts.

Biological function annotation

Gene annotation is treated as a classification task.

Input: DNA sequence fragment (≤ 512 base pairs).

Output: Probability distribution over annotation types.

Total examples:

- Training set: 9,216 examples
- Validation set: 1,024 examples

Mutation generation

- Model Used: Sequence-to-sequence Transformer with a language modeling head.
- Input: DNA sequence
- Generation Method:
  - Beam search : Generates five candidate sequences autoregressively
  -
  - Ranking Candidates: Uses noise identification pipeline.
  - Final Selection: Picks the sequence least likely to be identified as noise.

3.Result

3.1 ENBED outperforms state-of-the-art models on GB datasets

The classification head is attached to the final encoder block's embedding output for fine-tuning.

NT Benchmark Evaluation:

- Compared against **NT (v2) (encoder-only model)**
- Compared against **HyenaDNA (decoder-only model)**

GB Dataset Evaluation:

- Compared against **HyenaDNA**
- Compared against **CNN model(baseline model)**

3.1 ENBED outperforms state-of-the-art models on GB datasets

ENBED outperforms state-of-the-art models, achieving:

- **Superior results in 15/17 NT benchmarks**
- **Higher accuracy in 6/8 GB datasets**

Table 1. Nucleotide transformer (NT) benchmarks.

NT benchmark	Enformer	DNABERT-2	NT (2.5B)	HyenaDNA (1 kb)	ENBED (GRCh38)	ENBED
H3	0.719	0.785	0.791	0.779	0.723	**0.802**
H3K14ac	0.288	0.516	0.537	0.612	0.537	**0.636**
H3K36me3	0.344	0.591	0.616	0.613	0.611	**0.624**
H3K4me1	0.291	0.511	0.544	0.512	0.498	**0.591**
H3K4me2	0.211	0.336	0.322	0.455	0.433	**0.501**
H3K4me3	0.158	0.352	0.408	0.549	0.580	**0.587**
H3K79me3	0.496	0.613	0.621	0.672	0.648	**0.756**
H3K9ac	0.420	0.542	0.550	0.581	0.427	**0.590**
H4	0.732	0.796	0.807	0.763	0.750	**0.823**
H4ac	0.273	0.463	0.489	0.564	0.548	**0.605**
Promotor (all)	0.909	0.943	0.950	0.920	0.906	**0.961**
Promotor (non-TATA)	0.909	0.944	0.952	0.921	0.892	**0.959**
Promotor (TATA)	0.920	0.910	0.919	0.882	0.883	**0.944**
Splice acceptor	0.829	0.950	**0.973**	0.915	0.754	0.943
Splice donor	0.814	0.926	**0.974**	0.898	0.835	0.911
Enhancer	0.451	0.516	0.548	0.517	0.577	**0.585**
Enhancer Types	0.309	0.423	0.450	0.386	0.459	**0.482**

Genomic benchmark	CNN	DNABERT	GPT	HyenaDNA (Nguyen *et al.* 2023)	ENBED (GRCh38)	ENBED
Mouse Enhancers	69.0	66.9	80.1	85.1	81.1	**90.3**
Human Enhancers (Cohn)	69.5	74.0	70.5	**74.2**	70.8	71.2
Human Enhancers (Ensembl)	68.9	85.7	83.5	89.2	90.2	**92.2**
Coding versus Intergenomic	87.6	92.5	88.8	91.3	90.7	**93.0**
Human versus Worm	93.0	96.5	95.6	96.6	94.4	**97.3**
Human Regulatory Elements	93.3	88.1	91.5	**93.8**	85.6	90.2
Human Promoter (Non-TATA)	84.6	85.6	87.7	96.6	90.4	**97.2**
Human OCR (Ensembl)	68.0	75.1	73.0	80.9	76.2	**81.9**

ENBED identifies noise in genomic sequences

Sequence-level classification of erroneous genomic sequences using a synthetic dataset.

ENBED achieves 97.1% F1 score, outperforming existing models:

- DNABERT : 84.9%
- NT : 91.8%

Table 3. Erroneous sequence identification.

Model	Reference	F_1 score
DNABERT	(Ji *et al.* 2021)	84.9
Nucleotide Transformer	(Dalla-Torre *et al.* 2023)	91.8
ENBED	This article	97.6

ENBED identifies biological function annotations

Classification of biological function annotations in the Human reference assembly

ENBED achieves an F1 score of 74.1, outperforming:

- DNABERT : 63.2
- NT : 67.5
- HyenaDNA : 72.8

Table 4. Biological function identification.

Model	Reference	F_1 score
DNABERT	(Ji *et al.* 2021)	63.2
Nucleotide Transformer	(Dalla-Torre *et al.* 2023)	67.5
HyenaDNA	(Nguyen *et al.* 2023)	72.8
ENBED	This article	**74.1**

ENBED generates mutations using sequence-to sequence transformation

Evaluating mutation generation accuracy using an encoder–decoder Transformer.

Mutation uses 2 Evaluation Metrics

- Top-1 and Top-5 Accuracy (%):
    - Compared with real-world mutations from the Influenza Virus Resource.
    - Exact matches only counted as correct.
- Levenshtein Distance:

  Measures edit distance from real-world mutated sequences.

  Mean distance: 2.3 edits per 500 bp (~99.5% similarity).

Model	Top-1 accuracy	Top-5 accuracy	Mean LD	Median LD
Transformer (BPE tokenization)	32.0	56.1	30.6	24
ENBED (decoder-only)	53.1	72.1	6.1	4
ENBED	76.9	95.4	2.3	1

Encoder-decoder VS Decoder only

Also trained a decoder-only version of ENBED to compare sequence-to-sequence task performance.

Encoder–decoder architecture outperforms decoder-only due to better contextual feature extraction.

Higher accuracy (99.5%) due to fine-grained tokenization and better sequence representation.

Model	Top-1 accuracy	Top-5 accuracy	Mean LD	Median LD
Transformer (BPE tokenization)	32.0	56.1	30.6	24
ENBED (decoder-only)	53.1	72.1	6.1	4
ENBED	**76.9**	**95.4**	**2.3**	**1**

Ablation Study – Encoder vs Decoder Architectures

Compared three architectures:

- Decoder-only: 24 decoder layers, next-token prediction
- 1:1 Encoder–Decoder: 12 encoder + 12 decoder layers
- ENBED: 24 encoder + 12 decoder layers (2:1 ratio)

All models have ~800M parameters

Key Findings :

- Adding encoder blocks improves understanding of input
- Cross-attention allows decoder to leverage encoder embeddings
- ENBED's 2:1 structure yields significant improvement in pretraining accuracy
- Decoder-only models are limited to unidirectional generation → Not ideal for sequence-to-sequence mutation modeling

Discussion – Pretraining Data and Generalization

High-quality pretraining data contributes to ENBED's success

Uses complete assemblies (e.g., T2T-CHM13, GRCh38)

More accurate and diverse data → better generalization

- Model trained on T2T-CHM13 outperformed GRCh38 across the board
- Suggests that completeness and accuracy of reference genomes matter

Discussion – Key Insights

Discussion – Key Insights

- ENBED achieves **top performance** in 21/25 tasks (NT & GB benchmarks)
- Byte-level tokenization enhances mutation robustness (**97.6% accuracy**)
- Encoder–decoder excels at **sequence generation**, ideal for mutation modeling
- Success also tied to **data quality** and **architecture design**