Hybrid N-gram and Neural Language Model for Genomic Sequences

Texas A&M – CSCE 489: NLP

Rodrigo Orozco, Jesse Zheng

October 14, 2025

1 Introduction and Motivation

In a recent work, Liu et al. [2] demonstrated that hybrid models combining traditional n-gram statistics with neural language models can improve performance on language modeling tasks. Specifically, interpolating unbounded n-gram models with large language models yields better perplexity and downstream task performance than either approach alone. This suggests that n-gram statistics capture complementary information that neural models benefit from.

We hypothesize that this hybrid approach can also be effective for modeling genomic sequences. To test this, we propose combining n-gram statistics similar to Liu et al., with the Nucleotide Transformer [1] and evaluate the perplexity and effectiveness in promoter classification.

This hybrid approach may be particularly effective for genomic sequences because:

- DNA contains highly repetitive regions where n-gram statistics are informative.
- Known regulatory motifs (TATA box, CAAT box, etc.) are short, fixed patterns that n-grams naturally model.
- The combination may provide better interpretability by revealing when n-gram patterns dominate vs. when long-range context is needed.

2 Dataset

2.1 Datasets Used

We will use one primary genome dataset for building an n-gram index and evaluating perplexity for the Nucleotide Transformer and the hybrid model, and a second dataset to add task specific labels for an extrinsic evaluation of the two.

GRCh38 human reference genome, the most current assembly of the human genome, will be used as our primary dataset. It includes a genome size of 3.1 Gb and 24 total chromosomes. The genome will be partitioned by chromosome to create distinct training, validation, and test sets:

- Training set: Chromosomes 2-22
- Validation set: Chromosome 1
- Test set: Chromosomes X and Y

This single genome serves multiple purposes building the n-gram suffix array index from chr2-22, evaluating language modeling performance (perplexity) on held-out chromosomes (chr1, chrX, chrY), and providing the source sequences for promoter classification task.

For evaluating whether our hybrid model helps with real biological tasks, we need labels for promoters and nonpromoters. We will use Eukaryotic Promoter Database (EPDnew), a curated collection of approximately 30,000 experimentally validated human promoters. This provides us genomic coordinates (chromosome, position) marking experimentally confirmed promoter locations. Additionally, the coordinates are in GRCh38 reference system so they align with our language model training data.

2.2 Labeled Dataset

EPDnew will act as an annotation layer over GRCh38 telling us where promoters are located. We will extract the actual DNA sequences from GRCh38 at those coordinates with a window size of 500 base pairs. This window captures core and proximal promoter elements.

Postive Examples:

- extract ± 250 base pairs around each EPDnew transcription start site from GRCh38
- these sequences contain characteristic promoter features: regulatory motifs (TATA box, CAAT box), elevated GC content, and nucleosome positioning signals

Negative Examples:

- Sample 500 base pairs sequences from intergenic regions (between genes, no regulatory function)
- GC-content matching: For each promoter, select a non-promoter with similar GC percentage (within 5 percent)
- This prevents models from learning simple composition biases and ensures they must learn promoter-specific patterns (motif organization, positioning)

3 Proposed Approach

Our approach follows Liu et al. adapted for genomic sequences, combining statistical n-gram patterns with neural context understanding.

3.1 N-gram Index Construction

We will build an unbounded n-gram index on the training chromosomes using the infini-gram codebase [2]. This process involves:

- extracting and tokenizing training sequences using Nuclotide Transformer's tokenization scheme
- constructing a suffix array that stores all n-gram frequencies
- applying backoff and smoothing to handle unseen patterns

Fallback plan: If infini-gram integration proves incompatible with DNA tokenization, we will use KenLM to build a fixed-order n-gram model 5-gram with smoothing.

3.2 Neural Language Model

We will use the pre-trained **Nucleotide Transformer (NT-500M-human-ref)** from InstaDeep [1] as our neural component. The model is a 500M parameter autoregressive transformer trained on human genome sequences. We selected NT because its autoregressive architecture naturally aligns with n-gram interpolation, NT will be used frozen without any fine-tuning. Note that a fine-tuned NT on promoter vs nonpromoters classification could be explored as an extension, if time permits.

3.3 Hybrid Interpolation

We combine n-gram and neural probabilities via linear interpolation:

$$P_{\text{hybrid}}(y \mid x) = \lambda \cdot P_{\text{ngram}}(y \mid x) + (1-\lambda) \cdot P_{\text{NT}}(y \mid x)$$

where $\lambda \in [0,1]$ is the interpolation weight.

We will tune λ on the validation set via grid search over $0.0, 0.1, 0.2, \ldots, 1.0$, selecting the value that minimizes perplexity. This fixed λ will be used for all test sequences.

4 Evaluation Plan

4.1 Intrinsic Evaluation: Language Modeling Metrics

Perplexity will be used on the validation set to tune the hyperparameter in the hybrid model. Additionally, perplexity will be used to evaluate performance of the n-gram model, Nucleotide Transformer, and hybrid model on the held-out chromosomes, the test set. Perplexity measures model uncertainty, with lower values indicating better prediction of genomic sequences

4.2 Extrinsic Evaluation: Promoter Classification

To evaluate the language models on classifying promoter vs. non-promoter sequences we will use the model's negative log-likelihood (NLL) as a classification score under the assumption that promoters, having characteristic regulatory patterns, should be more "expected" by a good language model, lower NLL. Then the following methodology will be used:

- For each test sequence compute NLL under each model
- Invert NLL to create classification scores: lower NLL $\rightarrow$ higher score $\rightarrow$ predict promoter
- Use final scores to rank sequences

To determine the threshold for the classification score we will use the validation set and maximize it's F1 score. We can then determine accuracy, precision, recall, and F1 score on the test set using the same threshold to evaluate classification performance.

References

[1] Hugo Dalla-Torre, Liam Gonzalez, Javier Mendoza-Revilla, Nicolas Lopez Carranza, Adam Henryk Grzywaczewski, Francesco Oteri, Christian Dallago, Evan Trop, Hassan Sirelkhatim, Guillaume Richard, et al. Nucleotide transformer: building and evaluating robust foundation models for human genomics. *Nature Methods*, 21:2325–2334, 2024.

[2] Jiacheng Liu, Sewon Min, Luke Zettlemoyer, Yejin Choi, and Hannaneh Hajishirzi. Infini-gram: Scaling unbounded n-gram language models to a trillion tokens. *arXiv preprint arXiv:2401.17377*, 2024.