Deep Learning Integration for Genome Annotation

Anonymous
EECS 182
University of California, Berkeley
Berkeley, CA
April 22, 2025

Abstract

Accurate genome annotation is essential for understanding gene regulation and predicting protein-coding regions in complex eukaryotic genomes, where the identification of splice sites remains a major challenge due to variable sequence motifs and the prevalence of alternative splicing. This project explores the application of deep learning, specifically a Mamba-based sequence model, to classify DNA sequence windows from grass genomes as donor splice sites, acceptor splice sites, or non-splice regions. Initial results demonstrate that the Mamba-based model achieves competitive accuracy in splice site detection, suggesting that deep learning approaches can complement or surpass traditional homology-based and statistical annotation methods, thereby advancing the state of genome annotation for novel and complex eukaryotic species.

Codebase (Github Link): `https://github.com/KishC/cs182_genomics_group_project/tree/main`

1 Introduction

Genome annotation is the foundational process of identifying genes and describing the functional elements within genome sequences, particularly in identifying the coding and noncoding regions within the sequence to predict functional proteins. In eukaryotes (non-bacterial life), this process is especially challenging due to the complexity of gene structures, including the presence of introns and exons, alternative splicing, and the non-coding regions that make up the majority of the genome sequence. Accurate annotation allows for protein products to be predicted, a crucial result required for biological research and applications. This project investigates whether deep learning methods can improve the identification of protein-coding regions and splice sites in genomes, starting with grass species, in comparison to traditional homology-based and statistical approaches.

1.1 Background and Motivation

The central dogma of molecular biology describes how DNA is transcribed into RNA and then translated into protein. In eukaryotic genomes, the initial RNA transcript undergoes splicing, during which the introns, or non-coding sequences of genes, are removed, and exons, or the coding sequences, are spliced together before translation. This boundary between exons and introns is referred to as a splice site, and these regions are crucial for determining the correct protein sequence.

Accurate annotation is crucial in uncovering how genes are regulated and expressed in various conditions, providing insight on gene regulation. Further, identifying coding regions allows for protein sequences to be correctly predicted, facilitating the study of protein functionality and interactions. However, this annotation process often presents significant challenges. Splice sites typically show great variability and are often difficult to be correctly identified (Zerbino et al., 2020). Additionally, exons can be arbitrarily small while introns can potentially be very long sequences, making it highly difficult to detect exons with long introns complicating boundary recognition. Genes can also undergo alternative splicing, where exons can be exhibited in varied combinations, complexifying the process of identifying boundaries and structure of the sequence.

Tools such as BLAST and DIAMOND such as BLAST and DIAMOND are used to compare such proteins pairwise to produce scores for relatedness of protein sequences using edit distance-like algorithms. These tools are used to compare protein sequences against existing databases of proteins using pairwise comparison. Newer tools like Miniprot (2023) extend this by aligning proteins to genomes while accounting for splicing and frameshifts, enabling annotation in non-model species. Miniprot can take the proteins from different species, and map them to their corresponding locations in the novel genome. Miniprot's efficiency in cross-species alignment makes it a promising candidate for integration into automated pipelines, but it will fail to align if the proteins are too different from the one truly found in the novel species. These homology-based methods alone fail to identify novel proteins that lack database matches, or do not align well to different species. Due to this limitation, the exploration of deep learning approaches that can learn complex sequence patterns directly from genomic data is motivated, potentially facilitating a method to allow genomics researchers to efficiently classify exons, introns, and splice sites and reveal valuable insights into gene regulation, protein functionality, and alternative splicing events.

1.2 Research Question

Can deep learning models trained on annotated grass genomes, accurately identify splice sites and protein-coding regions, including in species not represented in the training data? Can Mamba architecture exhibit better performance than existing deep learning methods and can these models outperform or complement traditional homology-based annotation methods?

1.3 Literature Review

Homology-based annotation: BLAST, DIAMOND, and Miniprot are widely used for protein-to-genome mapping, but their reliance on existing databases limits their ability to discover novel genes.

Statistical models: Hidden Markov Models (HMMs), as implemented in tools like GeneMark and AUGUSTUS, are effective for prokaryotic genomes but struggle with the complexity of eukaryotic gene structures.

Deep learning approaches: Recent studies have shown that models such as bidirectional LSTM-RNNs and transformers can capture long-range dependencies in DNA sequences, improving splice site and exon/intron boundary prediction. For example, the use of Bayesian optimization and ensemble methods has yielded promising results in protein annotation tasks.

2 Methodology

2.1 Modeling Task

Through deep learning methods, the aim is to classify DNA sequence windows, extracted from annotated genomes, into one of three classes: donor splice site, acceptor splice site, or neither. The input data consists of fixed-length DNA sequences (L=512), centered on candidate splice sites or sampled negative examples, with each nucleotide encoded and embedded for model processing. The model's objective is to accurately identify whether a given sequence window contains a donor site, an acceptor site, or does not correspond to a splice site at all, enabling improved genome annotation and splice site detection in complex eukaryotic genomes.

2.2 Key Technologies

2.2.1 Mamba Sequence Model Integration:

This implementation takes advantage of Mamba architecture and optimization, a sequencing modeling approach that combines the efficiency of RNNs with transformers. Mamba uses selective state space models (SSMs) that can dynamically adjust their parameters based on input content, allowing in this context for genomic sequences to be processed while maintaining linear time complexity (https://arxiv.org/abs/2312.00752).

2.3 Model Architecture and Training Process

The proposed model, MambaSequenceClassifier, is designed to classify DNA sequences into three classes (acceptor site, donor site, or neither) using a hybrid architecture that combines learnable embeddings, a state-space model (SSM), attention pooling, and a classification head. This design balances computational efficiency with the ability to capture both local and global sequence patterns

critical for splice site detection. Below, we detail each component and its role in the model's architecture:

TODO: Diagram of Architecture

2.3.1 Model Architecture Overview

The architecture is structured as follows:

Nucleotide Embedding Layer: Converts raw DNA sequences into dense vectors.

Positional Embedding Layer: Encodes positional information for each nucleotide.

Mamba State-Space Layers: Process the sequence with efficient, inductive-bias-driven sequence modeling. (Mamba)

Attention Pooling: Aggregates contextual features from the sequence.

Classification Head: Maps pooled features to class probabilities.

2.3.2 Component-by-Component Explanation

The following depicts the major components of the pipeline:

(a) Nucleotide Embeddings

DNA sequences are sequences of nucleotides (A, C, G, T) with ambiguous bases (N) and padding (P). A learnable embedding layer maps each nucleotide to a d-dimensional vector (d=128 in our implementation). This allows the model to learn semantic relationships between nucleotides (e.g., A and T often appear together in complementary strands).

(b) Positional Embeddings

Positional information is critical for splice site detection, as specific motifs (e.g., GT at donor sites) depend on their location relative to the splice junction. Learned positional embeddings are added to nucleotide embeddings, enabling the model to distinguish between identical nucleotides at different positions.

(c) Mamba State-Space Layers

The core of the model uses Mamba layers, a lightweight alternative to transformers for sequential data. Each Mamba layer processes the sequence with a state-space model (SSM), which models temporal dependencies with linear complexity in sequence length (O(L)) compared to transformers' quadratic complexity (O(L$\hat{2}$)). Key features:

Efficiency: Mamba layers process sequences in linear time, critical for long genomic sequences (L=512 in our case).

Inductive Bias: The SSM structure imposes constraints on how information propagates, reducing the risk of overfitting.

Stacked Layers: We use 4 layers to progressively refine contextual representations.

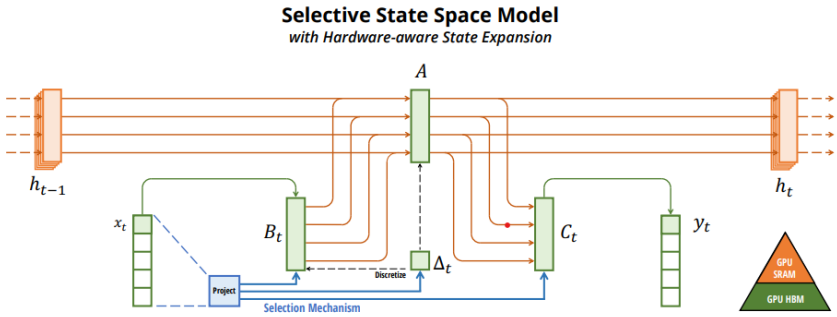

Figure 1: Mamba Architecture Diagram from https://arxiv.org/abs/2312.0075

(d) Attention Pooling

After processing the sequence with Mamba layers, a learnable attention mechanism pools the sequence into a fixed-size vector. The attention scores highlight regions of the sequence most relevant to classification (e.g., conserved splice site motifs). This replaces global pooling (e.g., mean/max) with a learned weighting scheme, improving focus on discriminative features.

(e) Classification Head

The pooled features are passed through a linear layer to produce logits for the three classes. Label smoothing (α=0.1) is applied during training to reduce overconfidence in predictions.

2.3.3 Training Process

The model is trained using the following pipeline:

(a) Data Preparation

Grass genomes with chromosome-scale annotations were sourced from NCBI, using .fna and .gff files. Splice sites (donor/acceptor) were extracted based on annotated gene structures, each represented by a motif-centered sequence window capturing upstream and downstream genomic context. To address dataset imbalance, negative examples were sampled uniformly from genomic positions matching the positive motifs and strands but located away from annotated splice junctions. This approach ensured balanced classes across motif-strand combinations for unbiased model training. The dataset is split into training, validation, and test sets with stratification to preserve class balance. Our primary modeling data set species was: Zea mays (TaxID: 4577), Assembly: GCF_902167145.1 (B73-REFERENCE-NAM-5.0).

Motifs refer to short patterns in genomic sequences that have well-defined biological functions (Kellis et al.). In reference to the project's objective, splicing signals are an example of motifs in RNA sequences.

Rows meeting the following conditions are filtered out prior to training:

Truncated sequences (I.e. Splice sites or negative samples from near the boundary of the genome)

Motif length < 2

Motifs/sequences with letter "N" or other data errors

The DNASequenceDataset class pads sequences to a fixed length (L=512) and maps nucleotides to indices.

(b) Training Loop

Loss Function: Cross-entropy loss with label smoothing to mitigate overfitting.

Optimizer: AdamW with weight decay (0.01) and learning rate scheduling (reduced by 10x if validation loss plateaus).

Regularization: Gradient clipping (L2 norm =1.0) and dropout (p=0.1) to prevent overfitting and improve generalizability.

Early Stopping: Training halts if validation loss does not improve for 10 epochs.

(c) Validation and Testing

Validation metrics (accuracy, loss) are tracked to select the best model checkpoint. Final evaluation on the held-out test set provides unbiased performance estimates.

2.3.4 Architectural Rationale and Literature Alignment

The choice of Mamba layers aligns with recent advances in efficient sequence modeling (Albert et al., 2023), where SSMs outperform transformers in speed and memory usage while maintaining competitive accuracy. The attention pooling mechanism draws inspiration from transformers (Vaswani et al., 2017) but is simplified to avoid quadratic complexity. This approach is ideal to balance biological interpretability (via positional and nucleotide embeddings) with computational practicality for genomic data.

2.3.5 Limitations and Future Directions

While the Mamba architecture efficiently processes long sequences, its reliance on inductive bias may limit adaptability to datasets with atypical sequence patterns. Future work could explore hybrid architectures that combine SSMs with transformer-like attention for greater flexibility. Additionally, our dataset is not set up to take full advantage of Mamba architecture's efficiency for long sequences.

2.4 Approach

2.4.1 Initial Model Approach

In the first approach to this problem statement, the proposed model fused the contextual embeddings provided from DNABert with the categorical nucleotide

embeddings. These aspects were concatenated then passed through a normalization and linear transformation followed by a GELU activation (in place of RELU, which was encountering model collapse) and dropout for regularization. The output is then processed through a stack of Mamba layers, with clamping in between to prevent numerical instability. After global average pooling, the resulting vector is passed through a classification head consisting of another normalization, dropout, and a final linear layer to produce logits (or the raw, unnormalized output values) for the three target classes. The model is trained using cross-entropy loss. In short, the data is honed in using BERT embeddings to superimpose feature candidates over the base pair information, then pass this onto Mamba layers.

2.4.2 DNABert Integration

DNABert adapts the deep learning BERT (Bidirectional Encoder Representations from Transformers) architecture to the analysis of DNA sequences by treating them as being composed of k-mer "words" (Ji et al., 2021). This encoder representation considers both upstream and downstream nucleotide contexts, taking advantage of its bidirectional capabilities to capture both local and long-range dependencies critical for tasks including the predictions of promoters, splice sites, and transcription factor binding sites. The initial approach for this product involved integrating DNABert to extract high-level sequence features, which are then integrated alongside additional nucleotide-level information. However, including these embeddings resulted in less than satisfactory results when computing results on the specific data.

2.4.3 Ensemble Modeling (BERT and Mamba)

An alternative approach that was tested was an ensemble approach combining BERT embeddings and classification with a Mamba-based classifier that uses its own separate embeddings. The BERT-based classifier projects DNA sequences into BERT's embedding space, then has a classification layer. The Mamba portion has a nucleotide embedding (non-BERT) along, positional embedding, attention pooling, and a final classification layer. The ensemble model then uses a learnable weight parameter to combine the predictions of both models. Interestingly, the learned weight parameter is 0.5253, which suggests that the model learns to evenly leverage both predictions rather than rely more heavily on one. This model showed slightly worse performance than the pure Mamba model, though could be refined with further tuning and experimentation.

2.4.4 LoRA Integration

Low-Rank Adaptation (LoRA) is a parameter-efficient fine-tuning technique designed to adapt large pre-trained modules, such as a transformer, to new tasks, allowing for original model weights to be frozen while introducing small, trainable low-rank matrices into selected model layers. LoRA maintains the performance

of full fine-tuning, providing a scalable and flexible way to adapt large models while preserving its original knowledge. This technique was integrated alongside DNABert for initial testing, yet still posed challenges and low validation.

2.5 Compute Power

Model training and computing was done on remote servers providing the necessary compute for working with large datasets.

3 Preliminary Results and Findings

3.1 Initial Testing Metrics

Initially, DNABert2 was directly used for sequence classification. This involved using BPE (byte pair encoding) tokenization on raw DNA sequences, standard fine-tuning of pre-trained BERT model, simple classification head on top of BERT outputs, and a single training phase

Results from pure DNABert2 classifier for baseline analysis:

DNABert2 with Classification Head Metrics [note: no special architecture added] [note: DNABert2 does not use k-mer tokenization]

Table 1: Model Evaluation Results

Settings			**Test Metrics**	
Seq Len	**Sample**	**Epochs**	**Accuracy**	**F1-scores [0,1,2]**
256	50000	10	0.84	0.81, 0.82, 0.87
256	50000	5	0.81	0.76, 0.79, 0.85
128	50000	5	0.82	0.79, 0.80, 0.86
512	50000	10 [stopped at 8]	0.84	0.81, 0.81, 0.86

After a fix on the data, now using splice_sites_full_centered_balanced_V3.csv, the metrics were as follows:

Table 2: Model Evaluation Results

Settings			**Test Metrics**	
Seq Len	**Sample**	**Epochs**	**Accuracy**	**F1-scores [0,1,2]**
256	50000	10	0.83	0.80, 0.82, 0.86

Following this, an experimental approach was used involving a combination of CNN and DNABert through a hierarchical architecture. Components include: Multiple parallel CNN layers with different kernel sizes (3,5,7) to capture motifs of different lengths, global max pooling to capture most important features, BERT layer for contextual understanding, and enhanced classification head with

dropout. A two-phase training process was also used, with Phase 1 involving training the CNN + classifier with frozen BERT, and Phase 2 involving fine-tuning the entire model.

Table 3: Model Evaluation Results

Settings			**Test Metrics**	
Seq Len	**Sample**	**Epochs**	**Accuracy**	**F1-scores [0,1,2]**
256	50000	10	0.50	0, 0, 0.66

This model appears to be predicting class 2 for every data point - degenerating into a useless classifier. The same metrics are seen for both phase 1 and phase 2, which indicates that the fine tuning is overshadowed by the CNN and classifier trained in Phase 1.

3.2 Final Testing Metrics

The best approach involved using Mamba directly along with a learned embedding directly from nucleotides (no DNABert2 embedding) and a positional embedding. This results in the following final metrics. The loss/accuracy curves as well as a confusion matrix are shown in Figures 2 and 3.

Table 4: Model Evaluation Results – Basic Metrics

Settings			**Test Metrics**	
Seq Len	**Sample**	**Epochs**	**Accuracy**	**F1-scores [0,1, 2]**
512	200000	20*	0.9447	0.9525, 0.9532, TODO

Table 5: Model Evaluation Results – Class-wise Precision and Recall

Class	**Precision**	**Recall**
0	0.9463	0.9589
1	0.9659	0.9408
2	0.9227	0.9343

*Early stopped at epoch 9

4 Appendix

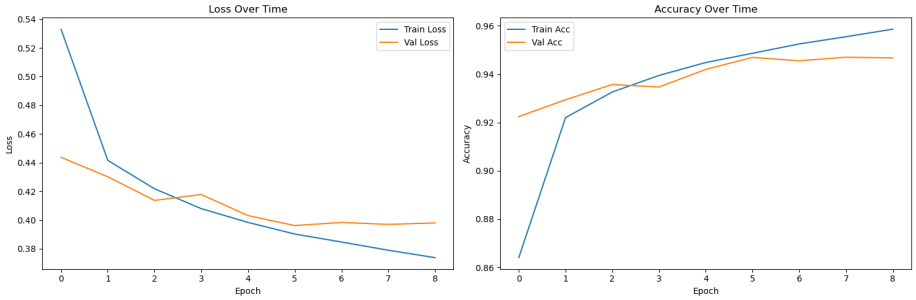

Figure 2: Loss and accuracy curves over epochs for Mamba model with learned embeddings on a sample of 200,000 observations.

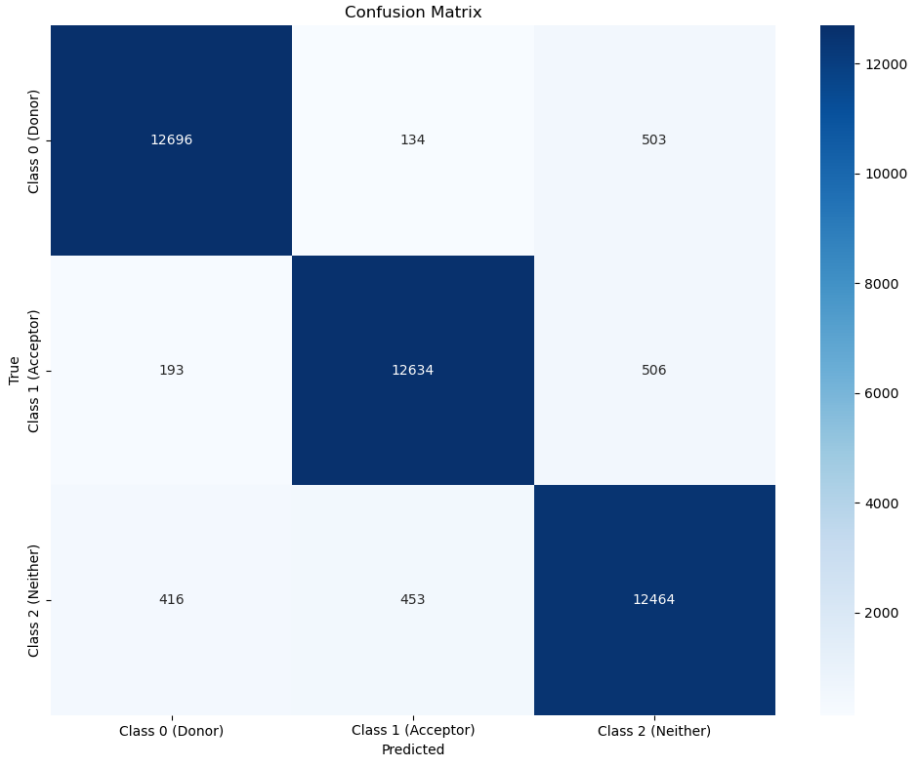

Figure 3: Confusion matrix results for Mamba model with learned embeddings

4.1 Generative AI Use

AI tools, including ChatGPT, Kimi AI, and Perplexity, were used in order to refine technical explanations of the relevant technologies and our code base breakdown. We further refined our paper through prompting existing LLMs to act as peer reviewers and give specific feedback in the perspective of someone who is unaware of the specific domain explored through this study.

4.2 Self Review

What is the main goal of the project?

The main goal of the project is to improve genome annotation in complex eukaryotic species by developing deep learning models, particularly using Mamba architecture, to accurately identify splice sites and protein-coding regions in grass genomes. This addresses limitations in traditional homology-based methods like BLAST and Miniprot, which struggle with novel genes and cross-species alignment.

What are the main claims? The main claim is that the Mamba-based model expresses high performance in splice site classification with linear time complexity, enabling efficient analysis of long genomic sequences.

What are the experiments?

Experiments include baseline comparisons against DNABert2, architectural alterations to test embeddings and specific tool integrations - including attempts with CNN and BERT, DNABert, and balanced data.

What is the evaluation protocol?

The evaluation protocol uses accuracy, per-class F1-scores, and precision/recall metrics on stratified splits of 200K Zea mays sequences, with regularization techniques like gradient clipping and dropout.

What is the data?

The data used in this project consists of chromosome-scale annotated grass genomes, specifically from Zea mays sourced from NCBI using .fna (genome sequence) and .gff (gene annotation) files. Splice sites (donor and acceptor) were extracted based on annotated gene structures, and each data point is a 512-nucleotide window centered on a splice site or a matched negative sample, with truncated sequences filtered out to ensure data quality and class balance

What is the task?

Through deep learning methods, the aim is to classify DNA sequence windows, extracted from annotated genomes, into one of three classes: donor splice site, acceptor splice site, or neither. The input data consists of fixed-length DNA sequences (L=512), centered on candidate splice sites or sampled negative examples, with each nucleotide encoded and embedded for model processing. The model's objective is to accurately identify whether a given sequence window contains a donor site, an acceptor site, or does not correspond to a splice site at all, enabling improved genome annotation and splice site detection in complex eukaryotic genomes.

How do the experiments support the goal/claims of the paper?

The experiments validate claims of the paper by demonstrating that the Mamba-based model achieves 94.47% test accuracy in splice site classification, significantly outperforming DNABert2 (83-84%) and resolving class imbalance with balanced precision/recall across donor, acceptor, and non-splice classes.

Are any of the limitations discussed in the paper?

THe paper mentions limitations involved when using DNABert and other related tools and integrations. Further, the paper mentions untested potential for combining Mamba with transformer attention layers.

What are the strengths of the paper?

This paper introduces a novel hybrid Mamba-based model for splice site detection, achieving a 94.47% accuracy. It provides multiple methods by which this task was approached, alongside thorough explanations on the tools involved, enhancing reproducibility and technical transparency.

What are the weaknesses of the paper?

A weakness includes a lack of direct comparisons to homology-based tools like Miniprot or BLAST and limited information on how the model functions across species.

Provide a suggestion for improving the paper.

The paper can further expand to include cross-species testing and direct benchmarks against traditional tools like Miniprot. Further, going into more detail about hyperparameters and data cleaning processes could help make results more reproducible.

What is the relevant related work?

Traditional homology-based methods (BLAST, Miniprot) and deep learning frameworks (DNABert, CNNs, DRANetSplicer) are key tools involved in related work. Mamba's application builds on state-space models for efficiency, while retrieval-augmented generation (RAG) in recent work enhances interpretability.

Is the paper reproducible?

Yes, the attempts and integrations mentioned are all included in the attached codebase.

Can you rerun the experiments?

Yes, experiments can be easily rerun by using the code provided.

Can you reproduce the results in the paper?

Results are reproducible through the code provided in the Github link, although more details could be provided on how the data was processed from the raw file into the file used for testing, as well as how hyperparameters were chosen.

Are all the plots in the paper clearly interpretable with well-defined and explained axes, with the methodology clearly explained in the paper text?

Yes, the plots are directly produced by code provided in the codebase, are well labeled, and directly support the content.

Is the English in the paper correct and clear?

Yes. Could provide more detail.

Do you have any feedback on any TODOs that the authors have left at this stage?

TODOs include adding in the final F1-scores in Table 4 and adding in the architecture diagram which will be further developed as more methods are tested and explored throughout the rest of the project timeline.

References

[1] https://pmc.ncbi.nlm.nih.gov/articles/PMC7116059/#:~:text=One%20of%20the%20difficulties%20for,connect%20previously%20independent%20gene%20loci. Zerbino, D. R., Frankish, A., Flicek, P., & Harrow, J. (2020). Progress, challenges, and surprises in annotating the human genome. Nucleic Acids Research, 48(8), 151–166. https://pmc.ncbi.nlm.nih.gov/articles/PMC7116059/

[2] https://bio.libretexts.org/Bookshelves/Computational_Biology/Book%3A_Computational_Biology_-_Genomes_Networks_and_Evolution_(Kellis_et_al.)/17%3A_Regulatory_Motifs_Gibbs_Sampling_and_EM/17.02%3A_Introduction_to_regulatory_motifs_and_gene_regulation#:~:. Kellis, M., Wold, B., Snyder, M. P., Bernstein, B. E., Kundaje, A., Marinov, G. K., Ward, L. D., Birney, E., Crawford, G. E., & Dekker, J. (n.d.). Introduction to regulatory motifs and gene regulation. In Computational Biology: Genomes, Networks, and Evolution. LibreTexts.

[3] https://arxiv.org/abs/2106.09685. Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. arXiv preprint arXiv:2106.09685 (2021).

[4] https://arxiv.org/abs/2312.00752. Albert Gu, Tri Dao. Mamba: Linear-Time Sequence Modeling with Selective State Spaces. arXiv preprint arXiv:2312.00752 (2023).

[5] https://arxiv.org/abs/1706.03762. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin. Attention Is All You Need. arXiv preprint arXiv:1706.03762 (2017).

[6] https://pubmed.ncbi.nlm.nih.gov/33538820/. Yanrong Ji, Zhihan Zhou, Han Liu, Ramana V. Davuluri. DNABERT: Pre-trained Bidirectional Encoder Representations from Transformers Model for DNA-Language in Genome.

Bioinformatics, 37(15):2112–2120 (2021). DOI: 10.1093/bioinformatics/btab083.

[7] https://www.mdpi.com/2073-4425/15/4/404#B47-genes-15-00404. Xueyan Liu, Hongyan Zhang, Ying Zeng, Xinghui Zhu, Lei Zhu, Jiahui Fu. DRANetSplicer: A Splice Site Prediction Model Based on Deep Residual Attention Networks. *Genes*, 15(4):404 (2024). DOI: 10.3390/genes15040404.