Training Transformers For Genetic Sequence Tasks

Anonymous Author(s)
Affiliation
Address
email

Abstract

The development of Next Generation Sequencing (NGS) technology has significantly improved the speed and accuracy of genome sequencing, allowing for more comprehensive exploration of the non-coding regions of DNA. In this paper, we introduce a new benchmark for evaluating the ability of neural models to understand the functional semantics of genomic sequences. Besides proposing the benchmark, we investigated the attention model that has gained popularity in the NLP community, despite the continued prevalence of CNN as the primary model structure in the bioinformatics community. Our research found that significant improvements can be made on these benchmark tasks by making some simple optimizations to the existing attention structures in NLP. Our experiments also demonstrate the necessity of a multitask benchmark.

1 Introduction

NGS technology enables sequencing of non-coding regions across the entire genome[1], offering insights into the conservation and diversity of non-coding sequences in different tissues and species. Consequently, the human reference genome has been extensively analyzed and annotated, although some challenges persist. First, numerous biological targets' expressions in different tissues or cell lines remain unmeasured, necessitating substantial laboratory resources and scientific expertise to fill these gaps[2]. Second, although the human reference genome comprises the most prevalent cases, humans have several variants and genetic differences that may or may not have functional significance, which are challenging to annotate solely through sequencing and are often absent from the reference genome[3]. Furthermore, besides the human genome, biologically significant organisms with less studied genomes also require thorough investigation as annotated sequence information is still lacking[4].

Considering these challenges, we must employ the new technologies in natural language processing to extract relevant information from numerous unannotated DNA sequences. We believe this is a matter of interest for machine learning technology; hence, we proposed a benchmark to assess DNA sequence representation. This benchmark comprises six supervised genomics-related tasks evaluating the functional semantic understanding of learned DNA sequence embeddings in several aspects. These tasks are significant in bioinformatics and broadly applicable to different DNA sequence research areas[5, 6, 7, 8].

Besides proposing the benchmark, we investigated the attention model that has gained popularity in the NLP community, despite the continued prevalence of CNN as the primary model structure in the bioinformatics community[9]. Our research found that significant improvements can be made on these benchmark tasks by making some simple optimizations to the existing attention structure

Submitted to NeurIPS 2021 AI for Science Workshop.

models[10, 11]. By comparing the three baseline models we trained, we believe that the attention structure can become the mainstream paradigm in bioinformatics research.

2 Background

The central dogma is one of the basic principles of modern molecular biology[12], which describes the process of transcribing DNA into RNA and then translating RNA into protein. However, with the in-depth study of the genome, people found that the non-coding regions of DNA also have important biological functions[13]. Non-coding regions of DNA are generally defined as regions of DNA that are not involved in protein coding and usually occupy most of the entire genome. These regions have multiple functions, including regulating gene expression, maintaining genome stability, and providing materials for gene evolution[14].

The regulation of gene expression is one of the most important functions of non-coding regions, which can be divided into two categories: promoters and enhancers. The promoter is usually located in the upstream region of the gene and is the starting point for RNA polymerase binding[15]. There are usually some conserved sequence motifs in the promoter, which can be used as binding sites for transcription factors, and then recruit other regulators and RNA polymerases to initiate the transcription of genes. Enhancers are usually located in the upstream or downstream regions of genes, which can enhance the rate of gene transcription[16]. A large number of binding sites exist in enhancers for transcription factors, which can interact with RNA polymerases and other regulatory factors to form a complex regulatory network that ultimately affects gene expression.

In addition to promoters and enhancers, some other regulatory elements exist in the non-coding regions, such as terminators and histone modifications. Terminators are located in the downstream regions of genes and can control the termination of gene transcription[17]. Histone modifications can affect the structure and stability of chromatin, which in turn affects gene expression[18]. Histone modifications include many types such as methylation, acetylation, phosphorylation, etc. These modifications can act on the histone tail to form a histone modification code, which ultimately affects the accessibility and expression level of genes. Therefore, the regulatory elements in the non-coding regions of DNA are closely related to the central dogma, and together they form the regulatory network of gene expression[14]. NGS technology can be used to determine epigenetic information such as DNA methylation and histone modifications in non-coding regions, as well as high-throughput screening of transcription factors and RNA-binding sites. This information can further help us understand the regulatory mechanism and biological function of the non-coding region. In addition to whole-genome sequencing, there are specialized sequencing methods for non-coding regions, such as RNA sequencing and Chromosome Conformation Capture (3C) technology. RNA sequencing can be used to identify and quantify non-coding RNA expression and study their function and regulatory mechanisms[19]. 3C technology can be used to capture the three-dimensional conformation of non-coding regions in chromosomes and gene transcription, providing further insights into the regulation of non-coding regions[20]. The development of sequencing technology has greatly promoted research on the function and regulatory mechanisms of DNA non-coding regions, which will help us better understand the mechanism of gene regulation and its applications in disease diagnosis and treatment. However, the annotation of DNA sequences is still far behind the new requirements, which we hope to be able to solve with machine learning. Finally, by using the existing sequencing data, the model can be generalized to any DNA sequence and obtain its functional interpretation.

3 Related Work

The intersection of machine learning and DNA processing has yielded transformative tools and models, significantly advancing our understanding of genomics. DeepSea[5] emerged as a pioneering model, utilizing a deep convolutional neural network to predict chromatin effects of noncoding variants from large-scale epigenomic data. It laid the groundwork for predictive modeling in genomics, especially in deciphering the roles of noncoding regions. Building on this, ExPecto[6] introduced a novel approach to predict the chromatin effects and associated gene expression changes resulting from noncoding genetic variants, providing crucial insights for interpreting noncoding genome-wide association studies. Transitioning to transformer-based architectures, DNABERT[10] adapted the BERT model specifically for DNA sequences, capturing contextual information within genomic data across various classification tasks. GeneBERT[11] further extended the application of transformers in genomics for gene prediction, offering an end-to-end framework that encapsulates genomic structural intricacies for accurate gene annotation. Enformer[21], with its transformer-based model trained on extensive epigenomic data, excelled in capturing long-range genomic interactions and predicting gene expression across different cell types and tissues. Collectively, these tools and models have not only enhanced predictive capabilities in genomics but also enriched our understanding of the genomic landscape, paving the way for new discoveries and applications in areas such as precision medicine and functional genomics.

4 Datasets

We propose a benchmark for six supervised learning bioinformatics tasks related to genomics. These tasks were chosen because they measure the model's ability to understand local segments of DNA (<=1000bp) from different perspectives and at different levels of difficulty, providing a wealth of challenges. In addition, these tasks have similar formulation, which usually contains the process as follows: a DNA sequence is mapped by the model to a real number or probability vector. This allows models to be compatible with all tasks to a certain extent under a uniform input, and allows fast transfer of many models from NLP to these tasks. These tasks are not only useful to measure the performance of the model, but the tasks themselves are worth caring about. Excellent performance of a model on a task can have direct applications or provide diverse biological insights. In addition, our dataset has the following features: 1) all tasks can be predicted directly from the sequence without additional domain knowledge; 2) the data is publicly available or allows us to redistribute it; 3) The dataset can be reproduced in different setting to fit in various purpose. As for the last point, it is actually quite common for DNA sequence tasks to enlarge the window length or increase the number of biological targets to increase the amount of information[10], etc., so quite a few of the datasets we list can be changed according to the needs of the user. But for comparison purposes, we can use our curated dataset.

Given the available data, we delicately processed it from its raw form into a form suitable for machine learning. For some data sets that don't contain negative samples, we add an appropriate number of negative samples. Negative samples can be chosen at random or tailored to the task. Depending on the size and nature of the data set, the data set can be split randomly or by leave-one-chromosome-out[5]. The specific division and processing is given in the task description. We categorize all tasks into sequence feature understanding and variant effect prediction. Because of task attributes, the size of dataset could be quite large. Description of data processing is in supplement Information. For each task, the dataset is described in the following ways:

- **(Definition)** A formal definition of the prediction problem, including input and output.
- **(Impact)** The importance of this problem.
- **(Principle)** The basic biology interpretation of task and how model works on this task.
- **(Metric)** The metric used to report results.
- **(Data Source)** The original source or reference of the data.
- **(Data Processing)** Data processing process, including positive and negative sample processing and data set division.

4.1 Sequence Feature Understanding

4.1.1 Promoter Probability Prediction: PromVary

- **(Definition)** A sequence-to-label task. In input sequence with the length of 500bp where TSS can be postioned at random site. The task is to prediction whether the sequence contains promoter region ($y_{true} \in [0, 1]$).
- **(Impact)** Promoters are regions of DNA that control the initiation of gene expression, and predicting the locations of promoters can be important for understanding gene regulation and identifying potential drug targets[15].
- **(Principle)** Promoter regions are related to some specific motifs. Some of the common motifs found in promoters include the TATA box, CAAT box, and GC box[15]. The model can detect these motifs by learning patterns of nucleotide sequences that are associated with promoter regions.
- **(Metric)** AUROC and AUPRC.
- **(Data Source)** Human promoters are from Eukaryotic Promoter Database[**?**] (EPD): 29598 sequences of human promoters.
- **(Data Processing)** All the sequence with positive label contains promoter region(-250bp to +50bp around TSS), generated by using a 500bp window centerd at -100bp from TSS and shifting it by a random offset. Negative samples are choosed randomly from genome. The number of positive samples remains equal to the number of negative samples. Of all the data, 80% is for training, 10% for validation, and 10% for testing.

4.1.2 Methylation Probability Prediction: Methyl96

- **(Definition)** (Definition) A sequence-to-float regression task. In input sequence with the length of 501bp, the task is to prediction the methylation probablity ($y_{true} \in [0, 1]$) of the center nucleotide (CpG site).
- **(Impact)** DNA methylation regulates gene expression by recruiting proteins involved in gene repression or by inhibiting the binding of transcription factor(s) to DNA. Methylation level prediction tools will are helpful when predicting gene expression level.
- **(Principle)** Methylation is regarded to be affected by certain local motifs including methylation motifs (MMs) and unmethylation motifs (UMs)[22]. The model is expected to detect corresponding motifs and their impact to center CpG site while filtering noise and pseudo-motifs in background sequences.
- **(Metric)** SpearmanR across all test samples.
- **(Data Source)** Methylation data are from 96 RRBS files selected in ENCODE[2] database.
- **(Data Processing)** The methylation probability is calculated by the number of methylated reads divided by the total number of reads. The methylation probability is represented as a value between 0 and 1, where 0 indicates completely unmethylated and 1 indicates fully methylated. The dataset is split across chromosomes, with chromosomes 12 and 13 reserved for validation, chromosome 10, 11, X and Y used for testing, and the remaining chromosomes used for training purposes.

4.1.3 Chromatin Feature Prediction: Track7878

- **(Definition)** A sequence-to-multi-label classification task, where each input sequence is mapped to multi-labels and each label represents whether the input sequence is within a peak in the Chip-seq Experiments. The given label is discrete in binary ($y_{true} \in \{0, 1\}$).
- **(Impact)** Predicting chromatin features such as DNase hypersensitivity, transcription factor (TF) binding sites, and histone modifications is important for understanding gene expression regulation and how various cellular processes are controlled[5]. DNase hypersensitivity is a measure of the accessibility of DNA to transcription factors and other DNA-binding

proteins. Transcription factors bind to specific DNA sequences to regulate gene expression. Histone modifications are chemical modifications to histone proteins, which are the proteins around which DNA is wrapped in chromatin. Different histone modifications can either promote or repress gene expression.

- **(Principle)** The accessibility of DNA is influenced by sequence features such as nucleosome positioning and DNA motifs[23]. Additionally, specific DNA motifs can be bound by transcription factors, which can recruit chromatin remodelers to open up chromatin and increase accessibility to DNase enzymes. Transcription factors bind to specific DNA sequences, or motifs, to regulate gene expression. The location and frequency of histone modifications are influenced by the underlying DNA sequence, as well as other factors such as transcription factor binding and chromatin remodeling. The model is expected to capture certain sequence feature and interaction between regions to find these signal.
- **(Metric)** AUROC and AUPRC.
- **(Data Source)** The ENCODE accession ids of all the 7878 Chip-Sequencing files were obtained from the DeepFun[24] article.
- **(Data Processing)** Using a similar processing method in DeepSea model. Each sequence is associated with 7878 binary labels which can be grouped into 3 categories according to the biological domain (DNAse I hypersensitivity site, Transcription factor binding site, Histone mark). Labels are computed by splitting the human reference genome (version hg19) into 200bp bins, discarding the ones not associated with any biological event (all the 7878 labels are 0). Each input sequence is extended to 1000bp by taking the two 400bp regions adjacent to the classified bin. The dataset contains 28000694 samples, where 96% is for training, 1% for validation, and 3% for testing.

4.1.4 Gene Expression Prediction: GeneExp

- **(Definition)** A sequence-to-float regression task. The input is the 40kb length sequence centering at TSS and the ouput is its log fold change value ($y_{true} \in R$).
- **(Impact)** Directly predicting gene expression level from DNA sequence is a crucial step in understanding the relationship between genetic variation and gene expression. The expression level of a gene is determined by various factors such as regulatory elements, epigenetic modifications, and transcription factors. The DNA sequence itself can provide important information about the potential effects of genetic variants on gene expression. One of the main advantages of predicting gene expression directly from DNA sequence is that it allows us to identify functional genetic variants that affect gene expression levels.
- **(Principle)** DNA sequence can affect gene expression through a variety of mechanisms. Regulatory elements are regions of DNA that control gene expression by binding to transcription factors and other regulatory proteins. Variations in the DNA sequence of these regulatory elements can alter their binding affinity for these proteins, leading to changes in gene expression[25]. Epigenetic modifications, such as DNA methylation and histone modifications, can also affect gene expression by altering the accessibility of DNA to regulatory proteins. Changes in the DNA sequence can alter the pattern of epigenetic modifications, leading to changes in gene expression. The model is expected to understand regulatory elements, epigenetic modifications and their interaction to predict gene expression level.
- **(Metric)** SpearmanR across all genes.
- **(Data Source)** Expression data are from Expecto's training data[6].
- **(Data Processing)** Using a similar processing method in Expecto model. Each sequence is centered at TSS with 218 float numbers as label to representing its expression level (log RPKM) in different tissue or cell-type. The dataset is split across chromosomes, with chromosomes X and Y reserved for validation, chromosome 8 used for testing, and the remaining chromosomes used for training purposes.

4.2 Variant Effect Prediction

4.2.1 Causal SNP Prediction: Causal SNP

- **(Definition)** A paired-sequence-to-label classification task. The input is two sequences, which differs at centered(single mutation postion). The given label is discrete in binary ($y_{true} \in \{0, 1\}$). 1 represents the according mutation has significant variant effect at certain tissue or cell type, and vise versa.
- **(Impact)** Causal SNP prediction refers to the identification of single nucleotide polymorphisms (SNPs) that are causally associated with a specific trait or disease[25]. By identifying the specific SNPs that are causally related to a disease, researchers and clinicians can develop targeted therapies that are tailored to an individual's unique genetic makeup. Also, Causal SNP prediction can also help in understanding the biological mechanisms underlying diseases, by identifying the genes and pathways that are involved in disease development.
- **(Principle)** When a sequence mutates, some sequence and functional features of the chromosome change. We hope the model can accurately predict the features of the mutant sequence. By testing the model's predictions of significant and non-significant variations across multiple tissues, we can evaluate the quality of the DNA sequence representation and the predicted chromosome features learned by the model.
- **(Metric)** AUROC and AUPRC.
- **(Data Source)** High-probability causal variant data are from GTEx[26] v8 release.
- **(Data Processing)** We use likely causal variants (causal probability > 0.9, as determined by the fine-mapping model DAP-G[27] in GTEx release) as positive sample and likely spurious eQTLs (causal probability < 0.01) as negative samples. For each tissue, the negative samples are randomly sampled and the number of negative samples is twice that of positive samples. Of all the data, 70% is for training, 10% for validation, and 20% for testing.

4.2.2 In Silico MPRA prediction: MPRAProm

- **(Definition)** A sequence-to-float regression task. For a single sample, the input is a sequence with single mutation and the ouput is its Log2 variant expression effect ($y_{true} \in R$).
- **(Impact)** MPRA (massively parallel reporter assay) prediction is a method used to predict the impact of non-coding genetic variants on gene expression levels. It involves inserting a library of DNA sequences containing a variant of interest into a plasmid vector and then transfecting it into cells[28]. The resulting gene expression levels can then be compared to those of the wild-type sequence to determine the effect of the variant. The significance of in silico MPRA lies in its ability to identify and characterize regulatory elements on a genome-wide scale. By analyzing large datasets of in silico MPRA results, researchers can gain insights into the mechanisms of gene regulation and the genetic basis of complex traits and diseases.
- **(Principle)** Mutations in promoter and enhancer regions have a considerable probability to change the expression level of corresponding genes, and the prediction of mutation effects in this region can test the accuracy of the model to recognize the function of untrained sequences.
- **(Metric)** SpearmanR across all mutations.
- **(Data Source)** This dataset includes 10 promoter sequences from source[8]. Selected elements were limited up to 600 base pairs (bp) for technical reasons related to the mapping of variants to barcodes by subassembly.
- **(Data Processing)** Due to the unequal length of the experimental measured sequences, the following steps were taken to ensure that the input data has the same length: Find the center based on the range of the experimentally measured sequences; 2) Extend 700bp on both sides of the center; 3) Finally, obtain a sequence with a length of 1400bp. This ensures that

the input sequence contains at least 600bp of sequence from the center and has sufficient contextual length. The dataset was partitioned as follows: 1) positions on the sequence were randomly selected, of which 60% were used for training, 20% for validation, and 20% for testing; 2) For mutation data at any position, if the mutation type is SNP, it will be retained, and other mutation types (inDel) will be discarded, with a maximum of three samples per position.

5 Experimental Setup

As a baseline, we evaluated two popular neural architectures. One of them is the traditional convolutional network architecture commonly used in the bioinformatics community; The other is the transformer architecture recently used in deep learning. Since the application of transformer-based architectures to DNA-related tasks is still in its infancy, we explore a limited number of common issues, such as empirical selection of some parameters. Our experiments focus on producing numerical representations of DNA sequences and can be used to provide task-specific features for classification versus regression, among others. Therefore, our model structure is as simple as possible with common.

5.1 Tokenization and Encoding

Unlike natural language, DNA sequences cannot be "naturally" split into words, but character-level input is quite adequate for DNA sequence processing. In the usual convolutional neural networks dealing with DNA, such as DeepSea and previous work, AGCT four letters are converted to one-hot codes, such as A for [1, 0, 0, 0], G for [0, 1, 0, 0]... , N is denoted by [0, 0, 0, 0]. N-length sequences can be converted into $N \times 4$ size matrices in this way and then fed into a convolutional network for encoding. Character-level input is not necessarily very suitable for the Transformer architecture. There are some common approaches for word segmentation in Transformer architectures[29, 10], while the computational cost also needs to be considered. With fewer tokens, the model runs faster, reducing computational pressure. In order to balance performance and cost simultaneously, we collate and propose four possible schemes. It is important to note that there are some empirical choices of hyperparameters, we have conducted some experiments and determined the best hyperparameters as possible. Illustrated in Figure11.

- **K-mer**: This is a method of forming words by combining k adjacent characters, with a step size of 1 each time. For example, the sequence ATGGCTC will be converted into three tokens in the case of 5-mer: ATGGC, TGGCT, and GGCTC. Each token segmented by k-mers will be directly mapped to the corresponding hidden state by the embedding layer. In this case, an N-length sequence will result in N tokens. We choose k=5 because increasing k will lead to negligible performance improvement from previous work[10] and our experiments, but will cause the model's parameter size to increase rapidly.
- **Byte Pair Encoding**: BPE[30] is a type of dictionary-based compression, where the most common substrings in the data are replaced with a single symbol or token. Each token segmented by BPE will be directly mapped to the corresponding hidden state by the embedding layer. We use the human reference genome hg38 to train a BPE tokenizer with a vocabulary size of 16k. In this case, an N-length sequence will result in approximately N/7 tokens.
- **K-patch**: Also a method of forming words by combining k adjacent characters, but with a step size of k. For example, the sequence ATGGCTCG will be converted into two tokens in the case of 4-patch: ATGG and CTCG. Each token segmented by k-patch will be mapped to a vector by the embedding layer. Taking an N-length sequence as an example, assuming the model has a dimension of 512, an input matrix of size $\frac{N}{k} \times 512$ will be obtained at this step. This matrix will go through a convolutional layer and finally result in the corresponding hidden state. In this case, an N-length sequence will result in approximately N/k tokens.

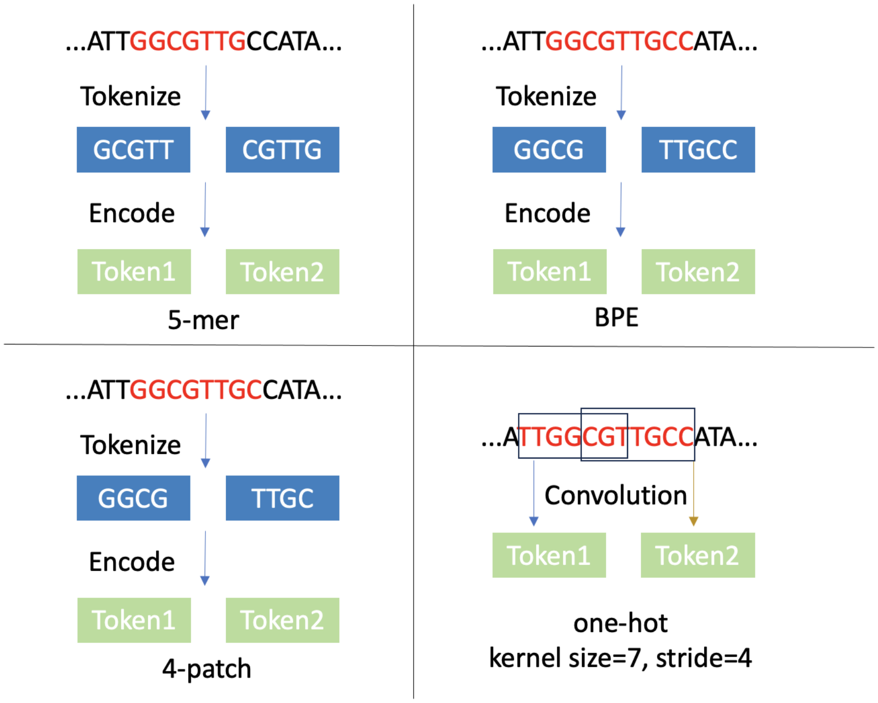

Figure 1: Illustrations of original input to encoded vector.

Choosing k=4 and a kernel size of 7 for the convolutional layer (allowing each initial token to carry approximately 28bp of information, which can be compared with the onehot method).

- **One-hot**: The DNA sequence is converted into one-hot encoding and then passed through a convolutional layer and a max-pooling layer to obtain the corresponding hidden state. In this case, the number of tokens obtained from an N-length sequence depends on the stride of the convolutional layer and the max-pooling layer. Choosing a convolutional layer stride of 1 and a kernel size of 25; and a max-pooling layer stride of 4.

In addition to the encoding of the input, position encoding is also important for the Transformer architecture. The performance of rotary position encoding[31], relative position encoding[32], and absolute position encoding was measured on the promoter detection dataset.

5.2 Neural Network

5.2.1 Convolution Models

For baseline CNN models, we referred to the DeepSea and Basset[33] models to design. Specifically, we mapped DNA sequences to a two-dimensional matrix representation using one-hot encoding, and the first layer of the network scanned the entire sequence's position weight matrix (PWM). After the matrix convolution of the sequence, we activated and applied a max pooling layer to reduce the number of parameters and achieve left or right shift invariance of the sequence. Subsequent convolutional layers operate on the output of the previous layer. After several activated convolutional layers, there are two ways to obtain the final sequence representation. One is to flatten the remaining hidden states and obtain the final sequence representation through a fully connected layer, denoted as CNN+FC; the other is to use an LSTM to process the remaining hidden states, denoted as CNN+LSTM. This sequence representation can be used for specific task classification or regression.

5.2.2 Transformer Models

For the Transformer, the most common quadratic complexity structure was used, similar to BERT, and the CLS token was used to obtain the sequence representation. The difference is that the gated GELU[34] activation function is used in the FFN module to achieve better performance. The entire model includes 4 Transformer layers with 512 hidden units and 8 attention heads in each layer with gated relu activation in FFN layer.

5.3 Pre-training

Due to limitations in the model structure, we only performed Masked Language Model[29] pre-training on Transformer models. We generated training data in the human genome hs1[35] by directly non-overlapping splitting and trained both the forward and reverse strands simultaneously. The pre-training sequence length was 1024, and the model was pre-trained for 10 epochs with a batch size of 128. The learning rate of the model increased from 0 to 1e-4 in the first 5k steps. Generally, the model reaches the plateau of the loss curve after 4-5 epochs. As the human reference genome is not very large, pre-training is very fast. During pre-training, the mask rate is a crucial parameter. It is worth noting that in the original DNABERT, because the 6-mer segmentation method requires six consecutive masked tokens, setting a mask rate of 15% actually results in a mask rate of only 0.025. In our setting, the mask rate is set to 30%, and preliminary experimental results show that this performs better than a mask rate of 15% in various segmentation methods.

5.4 Supervised Training

For each task, we fixed a supervised architecture and made the model as simple as possible. We did not perform hyperparameter tuning or significant architecture optimization because the main goal was to compare the feature extraction techniques of basic models on short sequences. In addition, according to the settings of the previous tasks, the supervised sequence representation of any DNA sequence shorter than 200bp (excluding context) can be directly obtained from the model trained in Task3. For a DNA sequence longer than 200bp, it is necessary to segment the sequence into segments with a length of 200bp or less, independently feed them into the model trained in Task3, and then concatenate them to obtain the sequence representation. Therefore, sequences of different lengths will have vector representations of different dimensions. The obtained sequence representations below will automatically apply the above process. Details of every task are listed in Appendix.

6 Results and Discussion

6.1 Position Encoding

We tested the performance of position encoding on two slightly different datasets. The task was to detect promoters, but in one dataset, the position of TSS in the positive examples was fixed, while in the other dataset, the TSS sequence in the positive examples was randomly shifted around sequence center. This allows us to test two important functions of position encoding in DNA tasks: 1) the ability to distinguish tokens in different positions; 2) the ability to guide attention flow based on the pattern of sequences in different positions. The fixed TSS position makes the task easier because the model only needs to focus on some fixed position sequence patterns, which is tested by DNABERT. The test results are shown in the table1. In both tasks, regardless of whether the position of TSS is fixed or not, the performance of absolute position encoding is relatively poor, while the performance of relative position encoding and rotational position encoding is similar. Because the implementation of relative position encoding requires training and additional computational overhead, while rotational position encoding does not require training, it may have better generalization ability. We chose rotational position encoding for further experiments.

Table 1: Performance of different position encoding on two datasets. PromFix is for TSS is fixed and PromVary is for TSS can vary in samples.

Dataset	Postion Encoding	AUROC	AUPR
PromFix	Rotary	0.979	0.982
	Relative	**0.980**	**0.983**
	Absolute	0.971	0.975
PromVary	Rotary	**0.945**	**0.897**
	Relative	0.942	0.891
	Absolute	0.935	0.885

Table 2: Performance of different tokenization on two datasets.

Dataset	Postion Encoding	AUPR	AUROC
PromVary	BPE	0.885	0.935
	one-hot	0.898	0.945
	4-patch	0.896	0.940
	5-mer	**0.902**	**0.946**
DeepSea	BPE	0.299	0.907
	one-hot	0.340	0.925
	4-patch	**0.357**	**0.935**
	5-mer	0.352	0.931

6.2 Tokenization

Two datasets of different sizes were selected to test different tokenization methods applied to DNA sequences. One dataset is the promoter detection task without fixed TSS as mentioned earlier, and the other is the Chromatin Feature Prediction task based on the training data used by the DeepSea[5] model, which consists of 4,863,024 chromatin profiles (4,400,000 training, 8,000 validation, and 455,024 test) with 919 labels (690 transcription factor (TF) binding sites, 125 DNase marks, and 104 Histone marks). Only the AUPR and AUROC obtained from the column representing TF were reported in the results. TF binding events are more concentrated in peak measurement compared to histone marks; moreover, TF binding also relies more on local sequence features, which satisfies our need for the model to obtain short sequence features[33]. From the results, it can be seen that the least suitable method among all the tokenization methods is the BPE algorithm. Despite its efficiency in compressing the number of tokens, it did not perform well in either of the two tasks. This may be because there are no natural delimiters in DNA, and slightly different DNA sequences may be split into very dissimilar token lists by the BPE algorithm, making it difficult for the model to understand the sequence features. One-hot tokenization is the most commonly used tokenization method in DNA, and it performed consistently well in both tasks, making it a stable baseline. In addition, retaining the convolutional layer also enables the visualization of convolutional kernels using conventional methods, making the model more interpretable. The 5-mers method performed well, and its convergence during training can be observed to be very fast with the number of epochs. However, its drawback is that it does not reduce the number of tokens at all, resulting in a significant increase in required memory and computation time when increasing sequence length, which leads to resource waste. The 4-patch method achieved the best results in the Chromatin Feature Prediction task and retained properties similar to the one-hot method: it not only visualizes convolutional kernels but also reduces the number of tokens, making training faster. Considering the training performance of different methods on the two tasks, we choose the 4-patch method for subsequent experiments.

6.3 Pre-training

Pre-training is a method that can achieve good models with less annotated data. After reading a large amount of sequence data, the model captures specific patterns in the sequences and can quickly respond to features in another domain[29]. In NLP, unsupervised pre-training has significantly

Table 3: Performance of different mask rate setting on two datasets.

Mask Rate	Methyl96(SpearmanR/MSE)	PromVary(AUROC)
0.1	0.595/0.055	0.941
0.2	0.604/0.052	0.944
0.3	**0.607/0.052**	0.947
0.4	0.605/0.053	**0.948**
0.5	0.598/0.056	0.946
0.6	0.600/0.054	0.944

Table 4: Performance of transfer learning from different source models.

Methyl96(SpearmanR/MSE)	From Scratch	Pretrain	Track7878
few shot(1% train)	0.219/0.857	0.240/0.839	**0.242/0.840**
all(100% train)	0.592/0.055	**0.607/0.052**	0.598/0.049

improved performance in many tasks with insufficient data. We first use two tasks, PromVary and Methyl96, to determine a reasonable masking rate. The reason for selecting these two tasks is that the amount of data for these two tasks is not very large, which satisfies the assumption of transfer learning, and the training can start from any initialized model. The results are shown in the table. For our setup, it can be seen that the best masking rate is not the previously assumed 15%, but can be higher under the 4-patch tokenization method. We chose a well-performing masking rate of 0.3 as the setting for subsequent experiments. Different best mask ratio for different model is also found by previous work[36]. It is necessary to explore the best mask rate changing to another model setting. In addition, we compared the performance difference between unsupervised and supervised pre-training models. We created a smaller dataset using the methylation dataset, with the training set being 1% of the original training set, and the validation and test sets remaining unchanged. The supervised pre-training model refers to the model trained on Track7878. Since the essence of Track7878 can be regarded as recognizing motifs with various functions, the model is expected to have the ability to quickly recognize new motifs. Finally, we compared the improvement in performance with the randomly initialized model. The results are shown in the table. It can be observed that unsupervised pre-training has comparable improvement on the small dataset compared to supervised pre-training, and even greater improvement on the dataset trained with all data. Pre-training on the human reference genome is a method with superior positive impact on the task.

6.4 Baseline for all tasks

We trained three models on all tasks. One of them is a second-order attention model based on the experiment described above, which uses rotary positional encoding and 4-patch tokenization. The other two are baseline models designed based on CNN. All their results are listed in the table. It can be seen from the table that the attention model with these settings outperforms the previously popular convolutional model in all tasks. However, the two models based on different sequence representations of the CNN model have comparability on different tasks. And the superiority of the two neural structures cannot be determined by a single task.

7 Conclusion

In this paper, we introduce a new benchmark for evaluating the ability of neural models to understand the functional semantics of genomic sequences. The purpose of this benchmark is to narrow the gap between the bioinformatics and NLP communities, and to provide a standardized and reliable tool for evaluating the state-of-the-art neural techniques on bioinformatics tasks. This benchmark is based on existing public data, which has been organized and formatted into six datasets that we define as many supervised learning tasks and provide some task guidance. These datasets are easy

Table 5: Performance of 4 setting for all tasks.

Dataset (Metrics)	Attention	CNN + FC	CNN + LSTM
PromVary (AUROC/AUPR)	**0.954/0.905**	0.939/0.880	0.942/0.887
Methyl96 (SpearmanR)	**0.607**	0.576	0.572
Track7878 (AUPR/AUROC)	**0.401/0.887**	0.391/0.870	0.395/0.874
CausalSNP (AUPR/AUROC)	**0.610/0.721**	0.605/0.716	0.606/0.718
GeneExp (SpearmanR)	**0.843**	0.835	0.835
MPRAProm (SpearmanR)	**0.389**	0.377	0.370

to use and can be solved by NLP experts without any additional knowledge in the field of biology. And our experiments also demonstrate the necessity of a multitask benchmark. In addition, we also conduct a preliminary exploration of the application of popular attention structures in NLP to DNA sequences. We experiment with tokenization methods and pre-training settings, and find a better DNA tokenization method than those previously used in Transformer models, which enables models to be trained and perform better in less time. By testing on our proposed benchmark, our attention model performs better than the popular convolutional model in the bioinformatics community.

References

[1] Clifford A Meyer and X Shirley Liu. Identifying and mitigating bias in next-generation sequencing methods for chromatin biology. *Nature Reviews Genetics*, 15(11):709–721, 2014.

[2] Yunhai Luo, Benjamin C Hitz, Idan Gabdank, Jason A Hilton, Meenakshi S Kagda, Bonita Lam, Zachary Myers, Paul Sud, Jennifer Jou, Khine Lin, et al. New developments on the encyclopedia of dna elements (encode) data portal. *Nucleic acids research*, 48(D1):D882–D889, 2020.

[3] Stephen T Sherry, M-H Ward, M Kholodov, J Baker, Lon Phan, Elizabeth M Smigielski, and Karl Sirotkin. dbsnp: the ncbi database of genetic variation. *Nucleic acids research*, 29(1):308–311, 2001.

[4] Shamil R Sunyaev, Warren C Lathe Iii, Vasily E Ramensky, and Peer Bork. Snp frequencies in human genes: an excess of rare alleles and differing modes of selection. *Trends in Genetics*, 16(8):335–337, 2000.

[5] Jian Zhou and Olga G Troyanskaya. Predicting effects of noncoding variants with deep learning–based sequence model. *Nature methods*, 12(10):931–934, 2015.

[6] Jian Zhou, Chandra L Theesfeld, Kevin Yao, Kathleen M Chen, Aaron K Wong, and Olga G Troyanskaya. Deep learning sequence-based ab initio prediction of variant effects on expression and disease risk. *Nature genetics*, 50(8):1171–1179, 2018.

[7] Wolf Reik and Wendy Dean. Dna methylation and mammalian epigenetics. *Electrophoresis*, 22(14):2838–2843, 2001.

[8] Martin Kircher, Chenling Xiong, Beth Martin, Max Schubach, Fumitaka Inoue, Robert JA Bell, Joseph F Costello, Jay Shendure, and Nadav Ahituv. Saturation mutagenesis of twenty disease-associated regulatory elements at single base-pair resolution. *Nature communications*, 10(1):3583, 2019.

[9] Yu Li, Chao Huang, Lizhong Ding, Zhongxiao Li, Yijie Pan, and Xin Gao. Deep learning in bioinformatics: Introduction, application, and perspective in the big data era. *Methods*, 166:4–21, 2019.

[10] Yanrong Ji, Zhihan Zhou, Han Liu, and Ramana V Davuluri. Dnabert: pre-trained bidirectional encoder representations from transformers model for dna-language in genome. *Bioinformatics*, 37(15):2112–2120, 2021.

[11] Shentong Mo, Xi Fu, Chenyang Hong, Yizhen Chen, Yuxuan Zheng, Xiangru Tang, Zhiqiang Shen, Eric P Xing, and Yanyan Lan. Multi-modal self-supervised pre-training for regulatory genome across cell types. *arXiv preprint arXiv:2110.05231*, 2021.

[12] Denis Thieffry and Sahotra Sarkar. Forty years under the central dogma. *Trends in biochemical sciences*, 23(8):312–316, 1998.

[13] Roger P Alexander, Gang Fang, Joel Rozowsky, Michael Snyder, and Mark B Gerstein. Annotating non-coding regions of the genome. *Nature Reviews Genetics*, 11(8):559–571, 2010.

[14] Anandakumar Shanmugam, Arumugam Nagarajan, and Shanmughavel Pramanayagam. Non-coding dna–a brief review. *Journal of Applied Biology and Biotechnology*, 5(5):42–47, 2017.

[15] Yehuda M Danino, Dan Even, Diana Ideses, and Tamar Juven-Gershon. The core promoter: At the heart of gene expression. *Biochimica et Biophysica Acta (BBA)-Gene Regulatory Mechanisms*, 1849(8):1116–1131, 2015.

[16] Elizabeth M Blackwood and James T Kadonaga. Going the distance: a current view of enhancer action. *Science*, 281(5373):60–63, 1998.

[17] Sankar Adhya and Max Gottesman. Control of transcription termination. *Annual review of biochemistry*, 47(1):967–996, 1978.

[18] Craig L Peterson and Marc-André Laniel. Histones and histone modifications. *Current Biology*, 14(14):R546–R551, 2004.

[19] Fatih Ozsolak and Patrice M Milos. Rna sequencing: advances, challenges and opportunities. *Nature reviews genetics*, 12(2):87–98, 2011.

[20] Satish Sati and Giacomo Cavalli. Chromosome conformation capture technologies and their impact in understanding genome function. *Chromosoma*, 126:33–44, 2017.

[21] Žiga Avsec, Vikram Agarwal, Daniel Visentin, Joseph R Ledsam, Agnieszka Grabska-Barwinska, Kyle R Taylor, Yannis Assael, John Jumper, Pushmeet Kohli, and David R Kelley. Effective gene expression prediction from sequence by integrating long-range interactions. *Nature methods*, 18(10):1196–1203, 2021.

[22] Mengchi Wang, Vu Ngo, and Wei Wang. Deciphering the genetic code of dna methylation. *Briefings in bioinformatics*, 22(5):bbaa424, 2021.

[23] Patrik D'haeseleer. What are dna sequence motifs? *Nature biotechnology*, 24(4):423–425, 2006.

[24] Guangsheng Pei, Ruifeng Hu, Peilin Jia, and Zhongming Zhao. Deepfun: a deep learning sequence-based model to decipher non-coding variant effect in a tissue-and cell type-specific manner. *Nucleic acids research*, 49(W1):W131–W139, 2021.

[25] Vivian G Cheung and Richard S Spielman. The genetics of variation in gene expression. *Nature genetics*, 32(4):522–525, 2002.

[26] Darren J Burgess. Reaching completion for gtex. *Nature reviews Genetics*, 21(12):717–717, 2020.

[27] Yeji Lee, Francesca Luca, Roger Pique-Regi, and Xiaoquan Wen. Bayesian multi-snp genetic association analysis: Control of fdr and use of summary statistics. *bioRxiv*, 2018.

[28] Anat Kreimer, Haoyang Zeng, Matthew D Edwards, Yuchun Guo, Kevin Tian, Sunyoung Shin, Rene Welch, Michael Wainberg, Rahul Mohan, Nicholas A Sinnott-Armstrong, et al. Predicting gene expression in massively parallel reporter assays: A comparative study. *Human mutation*, 38(9):1240–1250, 2017.

[29] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[30] Yusuxke Shibata, Takuya Kida, Shuichi Fukamachi, Masayuki Takeda, Ayumi Shinohara, Takeshi Shinohara, and Setsuo Arikawa. Byte pair encoding: A text compression scheme that accelerates pattern matching. 1999.

[31] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021.

[32] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*, 2020.

[33] David R Kelley, Jasper Snoek, and John L Rinn. Basset: learning the regulatory code of the accessible genome with deep convolutional neural networks. *Genome research*, 26(7):990–999, 2016.

[34] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[35] Sergey Nurk, Sergey Koren, Arang Rhie, Mikko Rautiainen, Andrey V Bzikadze, Alla Mikheenko, Mitchell R Vollger, Nicolas Altemose, Lev Uralsky, Ariel Gershman, et al. The complete sequence of a human genome. *Science*, 376(6588):44–53, 2022.

[36] Alexander Wettig, Tianyu Gao, Zexuan Zhong, and Danqi Chen. Should you mask 15% in masked language modeling? *arXiv preprint arXiv:2202.08005*, 2022.

A Supervised Traning Setting

A.1 Task1: Promoter Probability Prediction

Similar to the method used in DNABERT, sequence features are simply fed into a classifier. The classifier consists of a fully connected layer, an activation layer, and another fully connected layer, which is finally activated by a sigmoid function to obtain a probability value between 0 and 1, representing the probability that the model considers the sequence to be a promoter sequence. The threshold used for judgment is determined by the validation set. The loss function used is binary cross-entropy.

A.2 Task2: Methylation Probability Prediction

Sequence features are fed into a classifier. The classifier consists of a fully connected layer, an activation layer, and another fully connected layer, which is finally activated by a sigmoid function to obtain a probability vector between 0 and 1. Each dimension represents the probability value of methylation in the corresponding tissue or cell type. The loss function used is binary cross-entropy.

A.3 Task3: Chromatin Feature Prediction

Sequence features are fed into a classifier. The classifier consists of a fully connected layer, an activation layer, and another fully connected layer, which is finally activated by a sigmoid function to obtain a probability vector between 0 and 1. Each dimension represents whether there are peaks measured by Chip-seq experiments in the corresponding tissue or cell type. The threshold used for judgment is determined by the validation set. Actually, the more meaningful aspect of this task is to enable the model to produce supervised sequence representations for use in other tasks. To solve the problem of extremely unbalanced positive and negative labels, we used Focal loss to make training easier.

A.4 Task4: Gene Expression Prediction

Task 3's pre-trained backbone model is required. Since the single input is a sequence of about 40kb near the TSS, a sequence representation of $200 \times model_size$ dimensions can be obtained. Then this two-dimensional representation is fed into a randomly initialized transformer model (upper model), and the sequence representation obtained by this model is fed into a regression head as described above. There is no need to add an activation function at the end of the output, and a real-valued vector can be obtained directly, with each dimension representing the expression level of the corresponding gene in the corresponding tissue or cell type. The upper model also contains 4 transformer layers with 512 hidden units and 8 attention heads in each layer with relu activation in the FFN layer. The loss function is huber loss.

A.5 Task5: Casual SNP Prediction

Task 3's pre-trained backbone model is required. For any given SNP mutation, extending its sequence to the same input length as Task3 and inputting it together with the original sequence into the backbone model can obtain two vector representations, which are concatenated into a single vector as the feature input for the next model. Then a simple random forest classifier is implemented to determine whether the mutation has a significant impact on a specific tissue. The loss function is binary cross-entropy.

A.6 Task6: In Silico MPRA prediction

Task 3's pre-trained backbone model is required. For any mutation sequence, by feeding it into the backbone model along with the original sequence, two sequence-representing vectors can be obtained and concatenated into one vector as the input feature of the next model. Then, a simple XGBoost regressor is implemented to obtain changes in gene expression levels caused by mutations. The loss function is mean squared error (MSE).