Optimizing Human Genomics Representation via Contrastive Learning

Anonymous ACL submission

Abstract

In genomics, much like human language, raw DNA sequences are not directly interpretable by most machine learning algorithms. Consequently, it is essential to transform these sequences into a numerical format suitable for computational processing. Recent research has concentrated on developing foundational models for DNA sequences, often relying on large-scale pre-training. However, given that DNA sequences consist of a finite set of elements, we argue that it is not always necessary to pre-train models from scratch. Instead, significant improvements can be achieved through fine-tuning existing models. This study introduces a lightweight fine-tuning method aimed at enhancing genomic representations using unsupervised contrastive learning. We apply this method to DNABERT-2, a state-of-the-art genomics language model, and evaluate its performance on the publicly available GUE dataset. Our results demonstrate that this fine-tuning approach consistently surpasses the performance of the pre-existing model.

1 Introduction

The transformer (Vaswani, 2017) architecture consists of both an encoder and a decoder, yet many large language models (LLMs) achieve remarkable performance using either component independently. For instance, decoder-only models such as GPT-4 (Achiam et al., 2023) and Gemini (Team et al., 2023) have driven significant advancements in generative tasks by relying solely on the decoder. In contrast, encoder-only models like BERT (Devlin, 2018) exhibit exceptional performance in tasks related to classification and semantic representation.

Moreover, the applicability of the transformer architecture extends far beyond human language processing, proving highly effective in the analysis of genomic data, including DNA sequences. Recent research (Ji et al., 2021; Zhou et al., 2023; Dalla-Torre et al., 2023) has focused on training encoder-only BERT-based genomics models from the ground up. While these models may feature certain design variations, their overall training procedures remain largely consistent and follow a comparable structure, as outlined below:

- **Pre-training**: The process starts with unsupervised pre-training, where tokenized large-scale DNA sequences are fed into the encoder. The model is trained on tasks such as masked language modeling (MLM) (Devlin, 2018) and next sentence prediction (NSP) (Devlin, 2018), enabling it to capture meaningful sequence representations by optimizing the loss function.

- **Fine-tuning**: Following pre-training, task-specific fine-tuning is applied. The encoder's output, typically represented by the mean vector of the final hidden layer or the $[CLS]$ token, undergoes fine-tuning using either full-parameter Supervised Fine-Tuning (SFT) or parameter-efficient techniques like LoRA (Hu et al., 2021). During this stage, the loss is computed using cross-entropy (Mao et al., 2023) to adapt the model to the specific task.

- **Inference**: After fine-tuning, the model is deployed for inference on classification tasks. Given an input DNA sequence, the encoder generates a numerical representation of the sequence, which is then passed through the output layer to compute logits for each class label. These logits are processed using softmax activation or thresholding methods, depending on the task, to determine the final predictions.

However, it is important to note that pre-training a LLM from scratch is both time-consuming and resource-intensive. For instance, DNABERT (Ji et al., 2021) required approximately 25 days

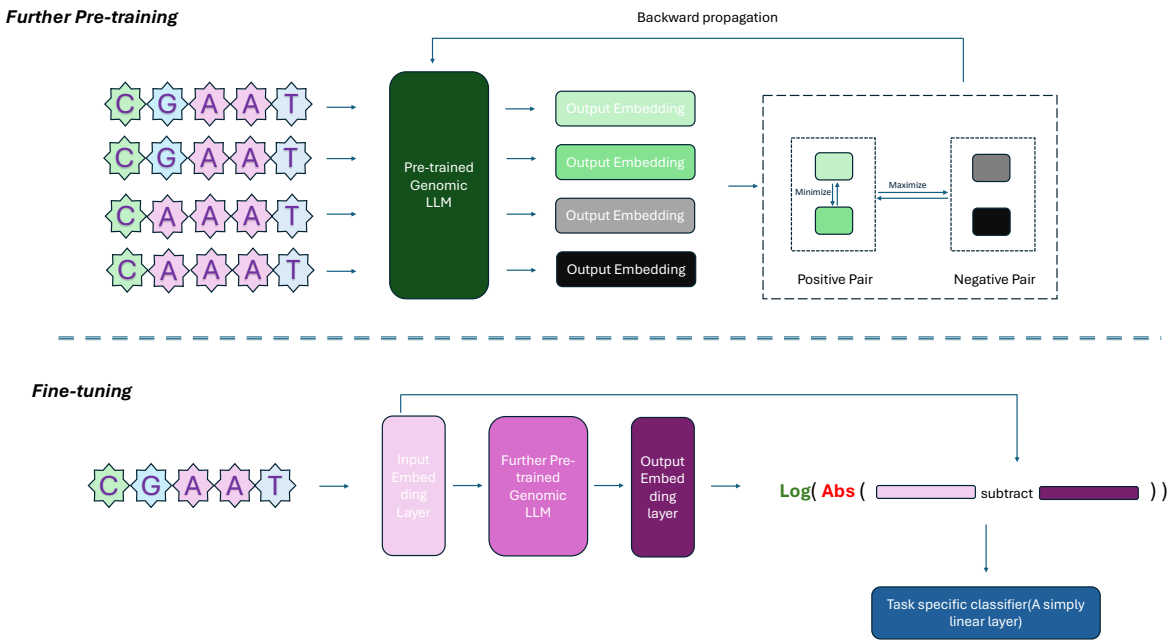

Figure 1: Our proposed method consists of two stages: the first stage outlines the procedure for further pre-training, while the second stage focuses on the inference process for downstream classification tasks. In this framework, the LLM highlighted in green represents the pre-trained genomics LLM, which is progressively optimized during the pre-training phase using a contrastive learning objective. The LLM highlighted in pink corresponds to the fully optimized version of the original genomics LLM, following the further pre-training phase.

of training on 8 NVIDIA 2080Ti GPUs, while DNABERT-2 (Zhou et al., 2023) took around 14 days on the same hardware setup. Given these substantial computational demands, this study seeks to explore whether an effective unsupervised optimization approach can be leveraged to enhance sequence representation and classification performance through the fine-tuning of an existing genomics LLM.

We propose in this study an unsupervised optimization framework utilizing contrastive learning, as illustrated in Figure 1. In this approach, contrastive learning aims to improve the learned representations by minimizing the distance between semantically similar data points and maximizing the separation between dissimilar points (Hadsell et al., 2006). Specifically, we apply the contrastive loss function (Equation 1) to DNABERT-2 during fine-tuning, optimizing the entire set of model parameters.

$$\ell_i = -\log \frac{e^{\text{sim}(\mathbf{h}_i, \mathbf{h}_i^+)/\tau}}{\sum_{j=1}^{2(N-1)} e^{\text{sim}(\mathbf{h}_i, \mathbf{h}_j)/\tau}} \quad (1)$$

Equation 1 defines the objective function for contrastive learning, where $\mathbf{h}_i$ represents the vector embedding of a sample $\mathbf{x}_i$, and $\mathbf{h}_i^+$ is the embedding of its corresponding positive sample $\mathbf{x}_i^+$. Similarly, $\mathbf{h}_j$ represents the embedding of a negative sample $\mathbf{x}_j$, which is dissimilar to $\mathbf{x}_i$. The parameter τ serves as a temperature hyperparameter, and sim denotes cosine similarity.

In line with (Chen et al., 2020), we do not explicitly select negative samples for each sample. Instead, we utilize batch processing in neural networks: a batch of **N** samples is randomly selected, and for each sample, a corresponding positive sample is generated, forming a batch of size **2N**. The contrastive learning objective is then applied across this batch, where, apart from its positive pair, all other samples in the batch are treated as negatives for a given sample.

While the identification of negative samples is straightforward using Equation 1, finding positive pairs presents a greater challenge. (Chen et al., 2020) addresses this issue in vision tasks by applying data augmentation techniques such as resizing and cropping to create positive pairs for each sample. However, such techniques are not directly applicable to natural language processing tasks, as altering a word or phrase can significantly change the meaning of a sentence. To overcome this, (Gao et al., 2021) employs random dropout(Srivastava et al., 2014) as a method for generating positive pairs, which we also adopt in our proposed ap-

proach.

Section 2 provides an overview of the background and related work, while Section 3 outlines the detailed methodology of our approach. In Section 4, we conduct an extensive experimental analysis, applying our method to DNABERT-2 across 12 clinical classification tasks.

The main contributions of this study are summarized as follows:

- We propose a refined pre-training method that replaces the traditional MLM with contrastive learning for parameter updates, providing a more efficient and targeted approach to enhancing representation learning.
- We introduce a novel DNA sequence embedding strategy, where the logarithmic scale of the absolute difference between the average sub-sequence embeddings from the final layer and those from the initial embedding layer is used. This approach, as opposed to directly using the average sub-sequence embeddings or the $[CLS]$ token from the final layer, offers a more subtle representation that captures inter-layer variations.
- We conduct a comprehensive analysis to examine how unsupervised optimization techniques, which effectively improve semantic similarity tasks, can benefit sequence classification. Furthermore, we emphasize the critical importance of sample selection in contrastive learning-based unsupervised optimization.

2 Background

Unlike human language, where sentences can be naturally decomposed into words, a DNA sequence—analogous to a sentence—lacks a standardized tokenization method for splitting itself into sub-sequences, the equivalent of words in language processing. However, a key distinction between human language and DNA sequences is that DNA is composed of only four nucleotides: **T**, **C**, **G**, and **A**. This results in a relatively small and finite vocabulary when constructing a lexicon of DNA sub-sequences. In other words, assembling a genomics LLM is inherently less complex than developing a human language LLM. For example, the largest DNABERT model, as reported in (Zhou et al., 2023), comprises only 89 million trainable parameters—approximately one-third the size of the BERT$_{large}$ model(Devlin, 2018), which contains 336 million trainable parameters.

DNABERT utilizes overlapping K-mers to construct sub-sequences, while Nucleotide Transformer (NT) employs a similar approach but segments the sequence into non-overlapping sections. In the case of K-mers, the original sequence is divided into sub-sequences of equal length, K. Non-overlapping K-mers partition the sequence without redundancy; in contrast, with overlapping K-mers, each segment shares some bases with its adjacent K-mer. DNABERT-2, on the other hand, implements Byte-Pair Encoding (BPE)(Sennrich et al., 2016), which learns a finite vocabulary of variable-length tokens based on the co-occurrence frequency of DNA sub-sequences.

Although tokenization methods may differ among various genomics LLMs, their outputs are generally similar, as they all employ BERT-based encoders. Given a single DNA sequence as input, the output from the last hidden layer has the shape $[[size_token_seq, out_dim], [out_dim]]$, where the first component is a 2D matrix containing the vector representation of each input sub-sequence, and the second component is the vector representation of the $[CLS]$ token. To obtain the representation of the entire DNA sequence, one can either compute the mean vector from the first component (*Mean-Pooling*) or use the $[CLS]$ vector from the second component. Previous research (Reimers and Gurevych, 2019) has shown that the *Mean-Pooling* vector consistently outperforms the $[CLS]$ token in semantic similarity tasks. Additionally, (Feng et al., 2024) demonstrates that *Mean-Pooling* yields superior performance in DNA sequence classification tasks compared to using the default $[CLS]$ vector. Therefore, this study employs the *Mean-Pooling* vector from the last hidden layer for all experimental analyses.

Using *Mean-Pooling*, we can derive a fixed-length vector representation regardless of the input DNA sequence length. However, a key challenge arises in optimizing model parameters due to the anisotropy problem, which is commonly observed in language representations from pretrained LLMs (Ethayarajh, 2019). Anisotropy refers to the phenomenon where output embeddings are concentrated in a narrow region of the vector space, limiting their expressiveness and ability to represent diverse semantic information. To address this, (Li et al., 2020) introduces BERT_flow, an invertible

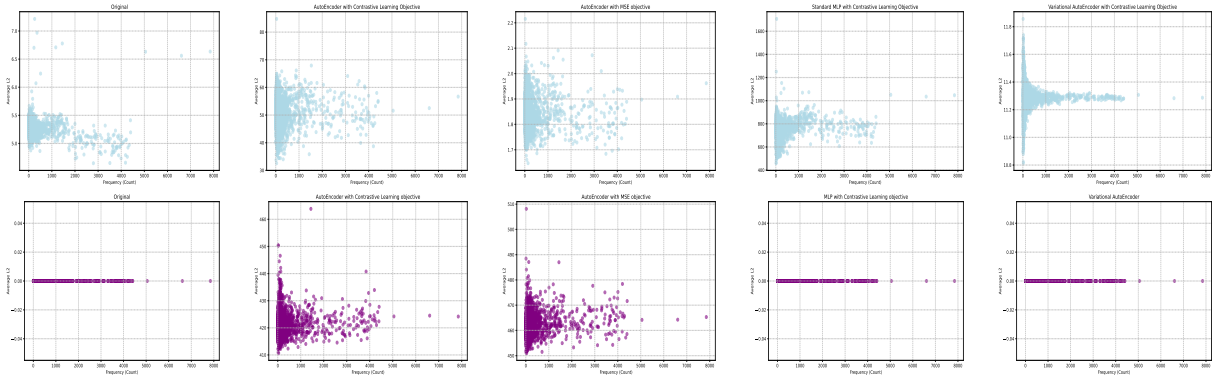

Figure 2: The average L2 distance of sampling DNA sub-sequences and sparsity under different selected models. the plots coloured by blue present the average L2 distance and the plots coloured by purple present the sparsity.

non-linear transformation that maps embeddings from a complex distribution to a standard normal distribution using normalizing flows(Papamakarios et al., 2021). Similarly, (Su et al., 2021) applies a linear transformation to map embeddings into an isotropic distribution. Both methods are unsupervised optimization techniques aimed at alleviating the anisotropy problem and enhancing the expressiveness of the sequence embeddings.

(Li et al., 2020) highlights that word embeddings in human language models can be influenced by word frequency, which inspired us to investigate whether this phenomenon also extends to genomic language models. To explore this, we randomly sampled batches of DNA sequences and processed them through DNABERT-2, extracting subsequence embeddings—analogous to words—from the last hidden layer. We then computed their L2 norms and sparsity (i.e., the number of zero-valued positions) to assess the impact of sequence frequency on the embeddings. In addition to examining the embeddings from DNABERT-2, we evaluated the effect of three models built on top of it: an autoencoder (AE) (Michelucci, 2022), a variational autoencoder (VAE) (Kingma et al., 2019), and a multilayer perceptron (MLP) with a contrastive learning objective. Our goal was to determine whether these models could mitigate any potential frequency-related bias in the embeddings. As shown in Figure 2, while the original DNABERT-2 embeddings exhibit subtle differences between high- and low-frequency sequences, neither the AE nor the VAE effectively addresses these disparities. In contrast, the MLP with a contrastive learning objective significantly reduces the discrepancy while preserving the sparsity properties of the embeddings. This suggests that contrastive learning may help rectify frequency-related bias in genomic language models.

The preceding discussion focused on methods for extracting fixed-length vector representations and the optimization techniques applied to them. Next, we turn to the critical question of data selection for model training. As demonstrated by (Sun et al., 2019), in-domain pre-training has been shown to significantly enhance model classification performance. In alignment with this approach, DNABERT-2 adopts a similar strategy by conducting further pre-training using the training data from all downstream classification tasks, optimized with the MLM loss. While the consensus in machine learning is that models should not access labeled test data during training, the fairness of utilizing unlabeled test data for unsupervised optimization remains an open question. (Alex et al., 2021) advocates for pre-training on unlabeled test data, and (Dubey, 2024) performed an empirical study on human language models, suggesting that there is no conclusive evidence to deem the use of unlabeled test data as unfair. However, several challenges arise when attempting to justify the model's results when unlabeled test data is involved in the pre-training process:

- If the final classification performance shows consistent improvement, it is difficult to determine whether this improvement stems from legitimate learning or from unintended information leakage from the test data.
- If the final classification performance does not show consistent improvement, the validity of utilizing this approach becomes questionable.

Consequently, in this study, we exclude unlabeled test data from the pre-training process. In-

stead, we employ public human genomics data (hg38) (for Biotechnology Information , NCBI), along with the original training data from downstream classification tasks, to conduct further pre-training. This approach ensures a more transparent and robust evaluation of the model's performance, eliminating concerns of potential data leakage or bias.

3 Proposed Method

The proposed method introduces an unsupervised further pre-training optimization based on contrastive learning. As illustrated in Figure 1, the overall training framework comprises two stages: **Further Pre-training** and **Fine-tuning**. The objective of the further pre-training stage is to capture meaningful representations of DNA sequences in an unsupervised manner, while the fine-tuning stage focuses on extracting the most salient information to enhance performance on downstream classification tasks. The training procedure is outlined as follows:

1. Utilizing a pre-trained genomics LLM, the model is set to training mode to activate random dropout, allowing for stochastic variations in the output.

2. The unlabeled training data is randomly shuffled, and a batch containing **N** data points is fed into the LLM. Each data point corresponds to an individual DNA sequence.

3. Each DNA sequence within the batch is processed through the LLM twice, generating two similar, yet distinct, vector representations due to the stochastic effects of random dropout.

4. Upon completion of the previous step, the total number of vector representations in the batch increases to **2N**. The even-numbered positions in the batch correspond to the original vector representations, while the subsequent odd-numbered positions contain the corresponding positive pairs.

5. The model computes the contrastive learning objective (Equation 1) **N** times per batch and returns the average loss, which is then used for backpropagation.

Figure 1 provides an illustrative example of the proposed process. For instance, if the batch size is set to 2, the two sequences are fed into the pre-trained LLM twice, resulting in an output tensor with a shape of $[4, out_dim]$, achieved through Mean-Pooling. The loss is then computed in two stages: in the first stage, positions 0 and 1 are treated as a positive pair, while positions 2 and 3 are considered negative pairs; in the second stage, positions 2 and 3 become the positive pair, with positions 0 and 1 acting as negative pairs. The final loss is calculated by averaging the losses from both stages. In this study, backward propagation is applied to all model parameters directly, without the use of parameter-efficient techniques such as LoRA (Hu et al., 2021).

After the **Further Pre-training** phase concludes, the training proceeds to the next stage, where the LLM focuses on learning task-specific information. (Arora et al., 2017) proposes a method for generating sentence embeddings by computing a weighted average of word vectors and removing the projection of this average onto its first singular vector. Inspired by this approach, the log-scale absolute difference between the *Mean-Pooling* of the initial input embeddings and the last hidden layer is calculated to enhance the impact of the self-attention mechanism in the LLM, as shown in the **Fine-tuning** stage in Figure 1. This study does not apply weighting to sub-sequence embeddings, which is typically done to mitigate the effect of high-frequency sub-sequences. However, since the usage frequency of most DNA sub-sequences is observed to be relatively uniform in this case, weighting is deemed unnecessary.

4 Experiments

An experimental analysis is now presented to assess the effectiveness of the proposed method in achieving consistent improvements over existing LLMs. Section 4.1 provides a description of the downstream classification dataset, while Section 4.2 details the hardware configuration utilized for the experiments. Section 4.3 outlines the training setup, including the software libraries and hyper-parameter configurations employed. Finally, Section 4.4 presents the quantitative comparisons and performance evaluations conducted throughout the study.

4.1 Dataset

This study focuses on optimizing human genomics by selecting 12 downstream classification tasks

Model	TF0	TF1	TF2	TF3	TF4	**TF-Avg**
DNABERT-2$_{SFT}$	66.41	73.05	65.77	59.78	76.01	**68.20**
DNABERT-2$_{GapLog+SFT}$ ♣	69.49↑	73.29↑	64.63↓	60.41↑	76.77↑	**68.92**↑
DNABERT-2$_{simCL+SFT}$ ♣	70.11↑	73.41↑	61.78↓	60.99↑	77.53↑	**68.76**↑
DNABERT-2$_{simCL+GapLog+SFT}$ ♣	70.28↑	73.81↑	65.6↓	60.97↑	78.23↑	**69.78**↑
DNABERT-2$_{simCL+GapLog+SFT}$ ♡	70.18↑	73.39↑	65.09↓	61.84↑	76.56↑	**69.41**↑
DNABERT-2$_{simCL+GapLog+SFT}$ ◇	72.45↑	77.03↑	65.20↓	60.62↑	76.41↑	**70.34**↑

Table 1: The results for the Transcription Factor Prediction (Human) task, with MCC used as the evaluation metric.

Model	Splice	PD70_ALL	PD70_TATA	PD70_NOTATA	PD300_ALL	PD300_TATA	PD300_NOTATA	**Avg**
DNABERT-2$_{SFT}$	84.75	65.78	68.77	54.95	85.20	93.05	62.95	**73.64**
DNABERT-2$_{GapLog+SFT}$ ♣	84.49↓	66.94↑	69.05↑	60.19↑	84.79↓	93.10↑	66.06↑	**74.95**↑
DNABERT-2$_{simCL+SFT}$ ♣	85.26↑	64.70↓	68.87↑	63.49↑	85.52↑	93.10 ↑	66.06↑	**75.29**↑
DNABERT-2$_{simCL+GapLog+SFT}$ ♣	85.63↑	67.16↑	68.99↑	63.78↑	86.67↑	93.33↑	66.58↑	**76.02**↑
DNABERT-2$_{simCL+GapLog+SFT}$ ♡	86.43↑	67.05↑	68.09↓	55.00↑	87.34↑	94.09↑	69.02↑	**75.29**↑
DNABERT-2$_{simCL+GapLog+SFT}$ ◇	86.70↑	67.59↑	69.68↑	69.87↑	86.86↑	93.90 ↑	69.30↑	**77.70**↑

Table 2: The results for the Splice Site Prediction task, Promoter Detection tasks, and Core Promoter Detection tasks are presented. In these experiments, ♡ indicates that the training data for **Further Pre-training** was sampled from public human genomics (hg38), ♣ signifies that the data was sampled from all downstream training datasets, and ◇ denotes that the data was drawn from domain-specific training sets. The best-performing model for each task is highlighted in purple, with ↑ indicating performance improvement over the baseline and ↓ denoting a decline in performance compared to the baseline. The MCC is used as the evaluation metric.

from the GUE benchmark (Zhou et al., 2023). These include five Transcription Factor Binding Site Prediction tasks (Human), aimed at identifying the binding sites of Transcription Factors (TF); one Splice Site Prediction task (Human), which predicts splice donor and acceptor sites to identify locations in the genome where alternative splicing occurs; three Promoter Detection tasks (Human), focused on identifying proximal promoter regions, key sequences that initiate transcription; and three Core Promoter Detection tasks (Human), which specifically target the prediction of core promoter regions. The input sequence lengths range from 70 to 400 nucleotides, and most of these tasks are framed as binary classification problems.

4.2 Training Details

DNABERT-2 is selected as the base genomics LLM, with parameter updates applied in two stages. The first stage, **Further Pre-training**, involves unsupervised optimization. For this stage, three types of data are utilized: (1) random samples from the general human genomic dataset (hg38), (2) random samples from all downstream training datasets, and (3) random samples from domain-specific training data. The second stage, **Fine-tuning**, performs supervised optimization using domain-specific training data.

All experiments were conducted on a Linux server equipped with an NVIDIA RTX A4500 GPU with 20,470 MiB of memory and a 32-core Intel Xeon W5-3435X processor. All models are fine-tuned using the AdamW optimizer (Loshchilov and Hutter, 2019). The evaluation metrics for downstream tasks include top-1 accuracy (Acc), F1-score (F1), and the Matthews Correlation Coefficient (MCC). All experiments are implemented with PyTorch and the Transformers library (Wolf et al., 2020). A fixed random seed is set to ensure deterministic results across all experiments. We applied the identical configuration as (Zhou et al., 2023) and used their script on https://github.com/MAGICS-LAB/DNABERT_2 for **Fine-tuning**, with only one modification: replacing the output embedding by applying *Mean-Pooling* with *GapLog*, as the original DNABERT-2 used *[CLS]*. For **Further Pre-training**, we set the learning rate to $5e-5$ and the batch size to 64.

4.3 Quantitative Analysis

Table 1 and 2 present the results of the quantitative analysis. The baseline model, **DNABERT-2**$_{\text{SFT}}$, is trained using standard supervised fine-tuning without the incorporation of **Further Pre-training**. In contrast, **DNABERT-2**$_{\text{GapLog + SFT}}$ repre-

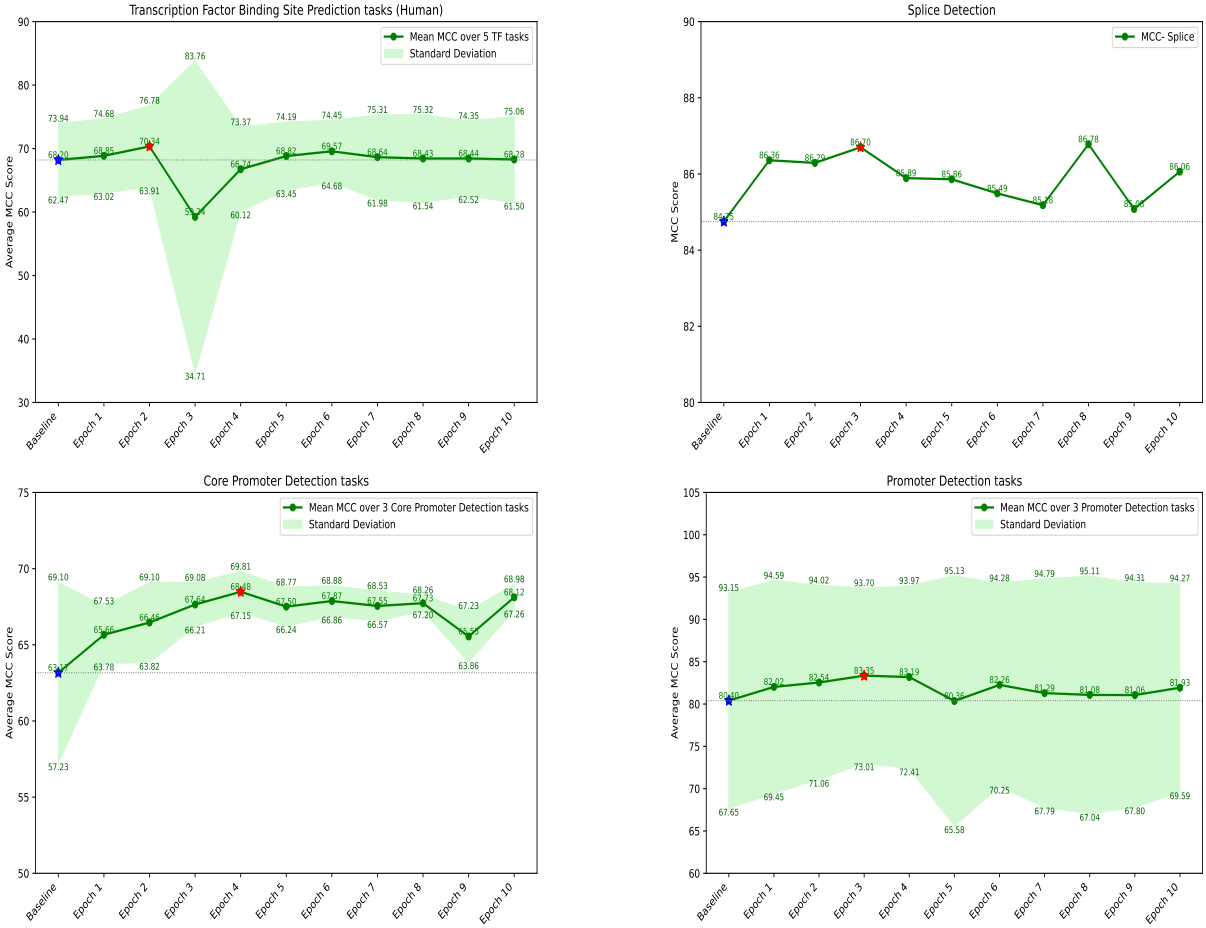

Figure 3: Performance of four **DNABERT-2**$_{simCL+GapLog+SFT}$ ◇ models, ⋆ denotes the baseline performance and ⋆ denotes the best model. The evaluation metric is MCC.

sents a model where **Fine-tuning** is applied without **Further Pre-training**, and serves as part of the ablation study to evaluate the impact of the proposed method. Another model, **DNABERT-2**$_{\text{simCL + SFT}}$, incorporates **Further Pre-training** alongside standard supervised fine-tuning, also included in the ablation study. The models **DNABERT-2**$_{\text{simCL + GapLog + SFT}}$ ♡, **DNABERT-2**$_{\text{simCL + GapLog + SFT}}$ ♣, and **DNABERT-2**$_{\text{simCL + GapLog + SFT}}$ ◇ follow identical training protocols, combining both **Further Pre-training** and **Fine-tuning** phases. These models are evaluated to determine the effectiveness of the proposed approach in comparison to the baseline and ablation variants.

For ♡, the training data was drawn from the public human genomics dataset hg38, consisting of 1,000 randomly selected DNA sequences of length 100 from each chromosome. For ♣, the training data was sampled from the training sets of all 12 classification tasks, with 2,000 randomly selected DNA sequences per task. In the case of ◇, domain-specific data was used, with 3,000 DNA sequences selected per task within each domain. Only tasks within the same domain shared the same **Further Pre-training** LLM. The 12 tasks were grouped into four domains: Transcription Factor Prediction, Splice Site Prediction, Promoter Detection, and Core Promoter Detection. As a result, four distinct **DNABERT-2**$_{\text{simCL + GapLog + SFT}}$ ◇ models were trained, one for each domain.

As shown in Table 1, the proposed method, **DNABERT-2**$_{\text{simCL + GapLog + SFT}}$, outperforms the baseline model in four out of five classification tasks for Transcription Factor Prediction, with the ◇ model demonstrating significant improvement over the baseline (Avg: 70.34 vs. 68.20). This result is consistent with expectations, as the ◇ model was trained on domain-specific data, thereby enhancing its performance in the corresponding tasks. Table 2 presents the classification results for Splice Site Prediction and Core Promoter Detection, where **DNABERT-2**$_{\text{simCL + GapLog + SFT}}$ consistently surpasses the baseline across all seven tasks, with the

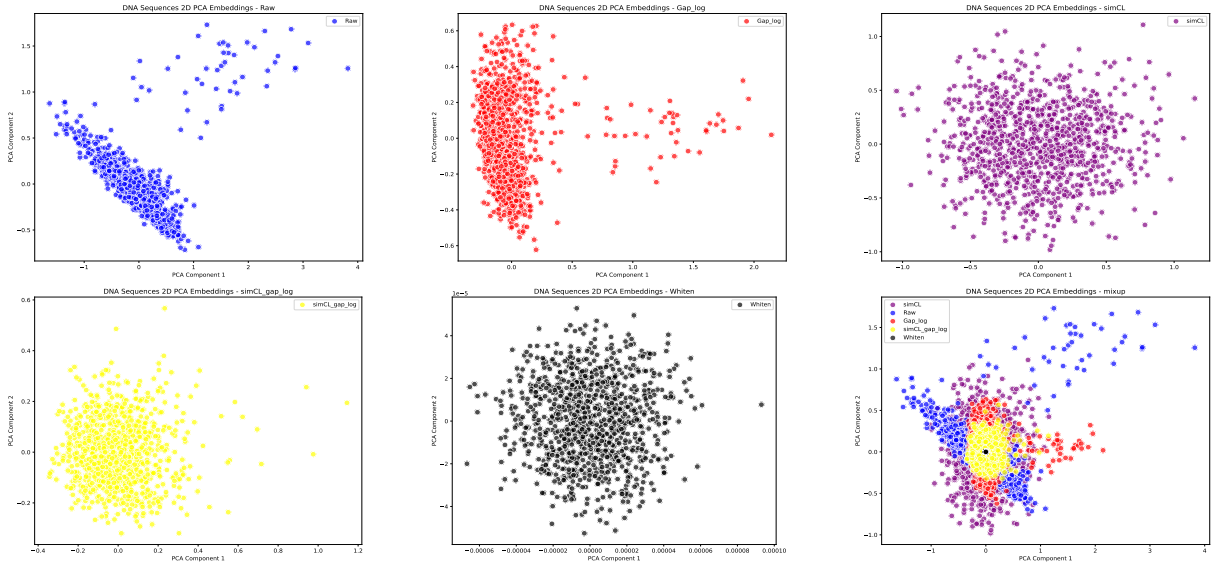

Figure 4: The distribution of vector representations of DNA sequences, generated by the LLMs

◇ model achieving the highest scores in most cases.

4.4 Impact of Over-Training

(Zhou et al., 2023) also employs further pre-training to incorporate domain knowledge using MLM. However, their model exhibits signs of overfitting, as the pre-trained model performs significantly worse than the version without further pre-training on all Human Transcription Factor Prediction tasks. This underperformance is likely a result of over-training, as their further pre-training stage utilizes the entire training data from the downstream tasks.

Figure 3 shows the performance of **DNABERT-2**$_{\text{simCL + GapLog + SFT}}$ ◇ over the course of the **Further Pre-training** phase at each epoch. Notably, the model achieves peak performance in a relatively short number of epochs, with three tasks reaching optimal results within just 3 epochs, and one task by epoch 4. These findings suggest that a large amount of data may not be required for effective **Further Pre-training**, and extended training could have adverse effects, such as overfitting.

4.5 Impact of Further Pre-training on Data Distribution

To better understand the impact of our proposed **Further Pre-training** on LLM outputs, we selected all DNA sequences from the test set of the downstream task TF_0. These sequences were processed through both the pre-trained models and the raw model, with Pinciple Component Analysis (PCA) applied to project the high-dimensional vectors into a 2D space. Notably, none of the models compared in this section underwent fine-tuning.

Figure 4 illustrates the distribution of DNA sequences across the compared models. In the raw model, **DNABERT-2**, the data points are widely dispersed, forming a diagonal clustering pattern along a linear trend. In contrast, the *Gap_log* model produces tightly clustered points with a clear linear separation, primarily aligned along the y-axis. The *simCL* model, however, distributes the data more uniformly in a spherical pattern, reducing directional bias, while the *simCL_Gap_log* model also achieves a spherical distribution but with greater density. It is important to note that increased density does not necessarily imply better performance. For instance, while (Su et al., 2021) transforms data into a highly concentrated distribution, our experimental results suggest that this dense configuration does not consistently lead to performance improvements.

5 Conclusion

This study presents an optimization framework for genomic LLMs that combines unsupervised **Further Pre-training** with supervised **Fine-tuning**. Empirical analysis demonstrates that this approach substantially improves the performance of genomic LLMs on downstream tasks. Moreover, the study investigates the impact of overtraining and examines how this optimization affects LLMs from a sampling distribution perspective.

6 Limitations

While the proposed method consistently achieves superior performance in downstream classification tasks, there is still room for improvement, and certain limitations must be acknowledged. The current approach involves two full-parameter update stages for the genomic LLM: unsupervised **Further Pre-training** and supervised **Fine-tuning**. To assess its effectiveness in low-data scenarios, we conducted an experiment using **DNABERT-2**$_{\text{simCL + GapLog + SFT}}$ ♣, fine-tuning the model with only 10% and 5% of the training data across the classification tasks. The average MCC across 12 tasks was 62.64 and 55.45, respectively, which represents a significant decline compared to the model trained with the full dataset (72.42). These findings suggest that the current method may not be optimally suited for few-shot classification, highlighting an area for future research.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Neel Alex, Eli Lifland, Lewis Tunstall, Abhishek Thakur, Pegah Maham, C Jess Riedel, Emmie Hine, Carolyn Ashurst, Paul Sedille, Alexis Carlier, et al. 2021. Raft: A real-world few-shot text classification benchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *5th International Conference on Learning Representations, ICLR 2017-Conference Track Proceedings*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Hugo Dalla-Torre, Liam Gonzalez, Javier Mendoza-Revilla, Nicolas Lopez Carranza, Adam Henryk Grzywaczewski, Francesco Oteri, Christian Dallago, Evan Trop, Bernardo P de Almeida, Hassan Sirelkhatim, et al. 2023. The nucleotide transformer: Building and evaluating robust foundation models for human genomics. *BioRxiv*, pages 2023–01.

Jacob Devlin. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Kush Dubey. 2024. Evaluating the fairness of task-adaptive pretraining on unlabeled test data before few-shot text classification. *arXiv preprint arXiv:2410.00179*.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China. Association for Computational Linguistics.

Haonan Feng, Lang Wu, Bingxin Zhao, Chad Huff, Jianjun Zhang, Jia Wu, Lifeng Lin, Peng Wei, and Chong Wu. 2024. Benchmarking dna foundation models for genomic sequence classification. *bioRxiv*.

National Center for Biotechnology Information (NCBI). 2024. Ncbi genome dataset: Gcf_000001405.26.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910.

Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

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

Diederik P Kingma, Max Welling, et al. 2019. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9119–9130, Online. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Anqi Mao, Mehryar Mohri, and Yutao Zhong. 2023. Cross-entropy loss functions: theoretical analysis

and applications. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Umberto Michelucci. 2022. An introduction to autoencoders. *arXiv preprint arXiv:2201.03898*.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. 2021. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Jianlin Su, Jiarun Cao, Weijie Liu, and Yangyiwen Ou. 2021. Whitening sentence representations for better semantics and faster retrieval. *arXiv preprint arXiv:2103.15316*.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *Chinese computational linguistics: 18th China national conference, CCL 2019, Kunming, China, October 18–20, 2019, proceedings 18*, pages 194–206. Springer.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

A Vaswani. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

Zhihan Zhou, Yanrong Ji, Weijian Li, Pratik Dutta, Ramana V Davuluri, and Han Liu. 2023. Dnabert-2: Efficient foundation model and benchmark for multi-species genomes. In *The Twelfth International Conference on Learning Representations*.