Probing and Lightweight Fine-Tuning of DNA Foundation Models

Runpeng He runpengh@usc.edu
Jiaxuan Cao caojiaxu@usc.edu
Nan Huang nanhuang@usc.edu
Allen Tran avtran@usc.edu
Natalie Kao kaon@usc.edu
Zuge Li zugeli@usc.edu

May 2026

1 Background Information

Recent advances in deep learning have led to the development of DNA foundation models that learn contextual representations from biological sequences by treating nucleotides as tokenized inputs [5]. Models such as DNABERT, DNABERT-2, Nucleotide Transformer, and HyenaDNA use different architectures and scaling strategies to capture sequence dependencies and regulatory patterns across different context lengths [5, 8, 1, 7].

Although embeddings from these pretrained models can be used directly for downstream tasks with lightweight classifiers, it is still not fully clear how much biological information is already encoded in frozen representations and how much task-specific adaptation is needed for better performance. To study this, we evaluate pretrained DNA models on three genomic classification tasks from Genomic Benchmarks: human non-TATA promoter prediction, human enhancer prediction, and Drosophila enhancer prediction [3]. These tasks allow us to test model performance across different regulatory elements and species.

In this project, we combine probing and fine-tuning approaches. Probing keeps the pretrained backbone frozen and tests whether its learned embeddings already contain useful biological signals, while fine-tuning adapts the model parameters to each downstream task. We further explore several adaptation strategies, including embedding layer selection, sequence length variation, alternative downstream classifiers, and LoRA-based lightweight fine-tuning [4]. This framework allows us to examine how model architecture, scale, and adaptation strategy jointly affect performance across promoter and enhancer classification tasks.

2 Related Work

2.1 DNABERT

DNABERT [5] is a transformer-based model designed to learn representations of DNA sequences by capturing contextual relationships between nucleotides. Unlike traditional approaches that treat individual nucleotides independently, DNABERT introduces k-mer tokenization, where sequences are divided into overlapping subsequences to better capture local patterns and biological motifs.

The model follows a self-supervised pretraining strategy similar to BERT, allowing it to learn general representations from large-scale genomic data. These pretrained representations can then be adapted to downstream tasks such as promoter prediction with relatively small labeled datasets. DNABERT provides the foundational idea of applying transformer-based language models to genomic sequences and demonstrates the effectiveness of contextual embeddings for biological tasks.

2.2 DNABERT-2

Zhou et al. [8] builds on DNABERT and addresses several of its key limitations. While DNABERT relies on k-mer tokenization and is trained only on the human reference genome, DNABERT-2 introduces a more flexible and efficient design.

One major improvement is the use of Byte Pair Encoding (BPE) instead of k-mers. This avoids the overlapping token issue in DNABERT and results in more compact and efficient representations of DNA sequences. DNABERT-2 also removes the fixed input length constraint by adopting ALiBi positional encoding, allowing the model to generalize to longer sequences. In addition, DNABERT-2 is trained on large-scale multi-species genomic data, enabling it to capture more diverse biological patterns beyond the human genome. It also incorporates more efficient attention mechanisms, making training significantly faster and more scalable.

Overall, DNABERT-2 improves both the efficiency and generalization ability of DNABERT, making it a stronger and more practical model for downstream genomic tasks.

2.3 Nucleotide Transformer

The Nucleotide Transformer (NT) Dalla-Torre et al. [1] is a family of transformer-based DNA language models trained on large-scale genomic data using a masked language modeling objective. Similar to BERT, the model learns contextual representations of DNA sequences by predicting masked tokens, allowing it to capture meaningful patterns in nucleotide sequences.

The pretrained representations can then be transferred to downstream tasks such as promoter classification, either through probing with lightweight classifiers or through fine-tuning. The paper shows that these representations are effective across a range of genomic prediction tasks.

Another important aspect of NT is its training data. The models were pretrained on different genomic datasets, including the human reference genome, a collection of diverse human genomes, and a multispecies dataset. This design allows the study to examine how model scale and data diversity affect downstream performance.

2.4 HyenaDNA

HyenaDNA[7] is a decoder-only genomic foundational model pretrained on the human reference genome, designed for processing long-range DNA sequence modeling at a single-nucleotide level using next-token prediction and causal masking. Unlike previous transformer-based models such as DNABERT2 and Nucleotide Transformer that rely on self-attention mechanisms, HyenaDNA replaces attention with Hyena operators, a class of subquadratic operators based on implicit convolutions that scale as O(N log N) in sequence length rather than O(N^2). This allows the model to process context lengths of up to 1 million tokens, up to 500x longer than previous dense-attention transformer models, with faster training and lower computational cost.

Rather than using k-mer tokenization as in traditional transformer DNA models, HyenaDNA operates at a single nucleotide resolution, treating each base as an individual token. This preserves fine-grained positional information that k-mer aggregation may obscure, enabling precise sequence modeling across both short and long genomic contexts. The model is available in multiple scales on Hugging Face, differing in context length to balance performance, efficiency, and task-specific needs.

2.5 Genomic Benchmarks

Grešová et al. [3] introduced *Genomic Benchmarks*, a standardized collection of genomic sequence classification datasets that makes it easier to compare models across different tasks. In this project, we use three datasets from the benchmark: Human non-TATA promoters, Human enhancers (Cohn), and Drosophila enhancers (Stark).

The Human non-TATA promoters dataset, formally `human_nontata_promoters`, is used for human non-TATA promoter prediction. Each sequence is 251 bp long and covers the region around the transcription start site. The task is binary classification: the model must distinguish promoter sequences from negative samples, which are constructed from random fragments of human genes located after the first exon.

To make the evaluation broader, we also in-

clude two enhancer prediction datasets: Human enhancers (Cohn), formally `human_enhancers_cohn`, and Drosophila enhancers (Stark), formally `drosophila_enhancers_stark`. These datasets test whether the model can recognize enhancer regions, not just promoters. They are also formulated as binary classification tasks, where the model predicts whether a given DNA sequence is an enhancer or a non-enhancer/background sequence.

Using these three datasets gives us a more complete view of model performance. Instead of evaluating only one type of regulatory element, we test both promoter and enhancer prediction. We also include data from both human and Drosophila, which helps us examine whether probing, full fine-tuning, and lightweight fine-tuning methods can generalize across different regulatory tasks and species.

3 Methodology

We implement DNABERT2, Nucleotide Transformer, and HyenaDNA as pretrained models. Using these models allows us to compare different representation learning approaches and assess how model architecture and pretraining strategies influence performance on downstream classification.

3.1 DNABERT2

3.1.1 Probe

Dataset and Preprocessing. For DNABERT-2 model, we use the `human_nontata_promoters` dataset from the Genomic Benchmarks collection [3]. Each DNA sequence is tokenized using the DNABERT-2 tokenizer. We set the maximum sequence length to 512 and apply padding. An attention mask is used so that the model can ignore padded positions.

Pretrained Model. For DNABERT-2 [8], we load the pre-trained checkpoint from Hugging Face [9] and freeze all model parameters. No fine-tuning is applied to any layer of the model. This setup allows us to evaluate the quality of the pre-trained representations directly, without the influence of task-specific fine-tuning.

Embedding Extraction. To evaluate how information is represented across different layers of DNABERT-2, we extract hidden-state embeddings from multiple layers and apply three pooling methods to obtain sequence-level representations for downstream probing.

We extract embeddings from multiple parts of the model: the embedding layer (layer 0), several transformer layers, the average of the last four layers.

DNABERT-2 does not behave exactly like standard BERT models. It removes padding tokens internally, which leads to two problems. First, we cannot directly obtain intermediate representations. Instead, we use forward hooks to capture the hidden states of each layer. Second, some layer outputs are flattened. We therefore implement a re-padding step to restore them back to the usual shape before applying pooling.

We then apply three pooling strategies:

- **Mean pooling**: averages all non-padding token embeddings using the attention mask
- **Max pooling**: takes the element-wise maximum over tokens while ignoring padding positions
- **CLS pooling**: uses the first token embedding as a sequence-level representation

This design is motivated by findings in both NLP and genomics that intermediate layers often capture the most task-relevant features, while the final layer may be overly specialized for the pre-training objective [1, 2].

Evaluation Metrics We evaluate performance using accuracy, F1 score, and Matthews Correlation Coefficient (MCC).

Accuracy measures the overall proportion of correctly classified sequences and gives a general sense of performance. However, it does not distinguish between different types of errors and can be less informative when classes are not perfectly balanced.

F1 score captures the trade-off between false positives and false negatives by combining precision and recall into a single metric. This makes it more suitable for comparing different model configurations in our setting.

MCC provides a more comprehensive evaluation by taking all entries of the confusion matrix into account and is generally considered a robust metric even when class distributions are uneven. It is also commonly used in genomic benchmarks such as DNABERT-2 and Nucleotide Transformer [8, 1].

3.1.2 Fine-Tuning

While the probing in 3.1.1 evaluates the quality of *frozen* DNABERT-2 representations on a single promoter task, it leaves two important questions: (i) how much performance can be gained when the backbone itself is allowed to adapt under different parameter-efficient strategies, and (ii) whether the conclusions generalize beyond the human non-TATA promoter setting. To answer both, we extend the DNABERT-2 evaluation along three axes simultaneously: *three datasets*, *four adaptation strategies*, and *three pooling functions*.

Datasets In addition to `human_nontata_promoters`, we include two enhancer-classification tasks from the same Genomic Benchmarks collection: `human_enhancers_cohn` and `drosophila_enhancers_stark`. Both are binary classification tasks, but they differ from the promoter task in sequence length and species: human enhancer sequences average several hundred base pairs, while Drosophila enhancers average roughly two kilobases. For all three datasets, we tokenize with the DNABERT-2 BPE tokenizer and truncate or pad to a fixed maximum of 512 BPE tokens. We verified empirically that on `drosophila_enhancer_stark` fewer than 4% of sequences exceed this length after BPE tokenization, so the effect of truncation on this dataset is negligible and a uniform 512-token budget keeps tokenized inputs directly comparable across all three datasets and all four fine-tuning strategies. This three-dataset suite covers regulatory-element classes and one cross-species transfer setting, which lets us test whether each adaptation strategy generalizes beyond a single task.

Fine-Tuning Strategies We compare four strategies that span the trainable-parameter spectrum from $\sim$0.17% (head only) to $\sim$32% (last-four-layer unfreezing).

Baseline. The entire DNABERT-2 backbone is frozen and only the MLP head is trained ($\sim$0.17% trainable). This is the end-to-end counterpart of the MLP probe and serves as the floor against which all adaptation strategies are measured.

Unfreeze last 4 layers. The last four transformer blocks are unfrozen alongside the MLP head, with the rest of the backbone (embeddings and the first eight layers) kept frozen. The rationale is that upper layers encode more task-specific and less general representations, making them the most beneficial layers to adapt.

Progressive unfreeze. To reduce the risk of catastrophic forgetting that can arise from simultaneously unfreezing multiple layers, we adopt a gradual unfreezing schedule. At the start of training (epoch 0), only the last layer is trainable; at epochs 2, 4, and 6, we additionally unfreeze the next-deepest layer, reaching the same final last-four-layer configuration by epoch 6. The optimizer is rebuilt whenever the trainable parameter set changes so that newly unfrozen weights inherit a fresh AdamW state, which stabilizes training relative to immediately unfreezing the full block. This approach allows the classification head to first stabilize before allowing deeper layers to adapt.

LoRA. We follow the LoRA configuration from the DNABERT-2 paper [8]: rank $r = 8$, scaling factor $\alpha = 16$, dropout 0.05, no bias adaptation. Together with the MLP head, this yields approximately 1.29% trainable parameters. The LoRA parameters are trained with a backbone learning rate of 1×10^{-4}.

Pooling Ablation The probing experiments showed that the choice of pooling function (Mean, Max, CLS) interacts strongly with layer depth in the frozen model. We therefore re-run each of the adaptation strategies above with each of the three pooling functions on each of the three datasets, holding all other hyperparameters fixed. The pooled representation is computed from the last hidden state of the backbone in all cases, so the ablation isolates the effect of pooling choice from the effect of layer choice. This produces a full $4 \times 3 \times 3 = 36$-cell evaluation grid, which allows us to determine whether its

reported performance depends on a particular favorable combination.

3.2 Nucleotide Transformer

We compare two parameter-efficient adaptation strategies for the Nucleotide Transformer (NT) 500M model: a *frozen probe*, where the pretrained backbone is kept fixed and a logistic regression classifier is trained on top of mean-pooled embeddings, and *LoRA fine-tuning*, where small low-rank adapter matrices are inserted into the attention projections of every transformer block while the backbone weights remain frozen. This isolates how much performance can be gained from lightweight backbone adaptation versus relying on the pretrained representations alone, and complements the DNABERT-2 evaluation along the parameter-efficient axis.

3.2.1 Datasets and Preprocessing

We use three binary classification tasks from the Genomic Benchmarks collection [3]: `human_nontata_promoters`, `human_enhancers_cohn`, `drosophila_enhancers_stark`. The provided train/test split is preserved. For LoRA we further hold out 10% of the training set as a stratified validation split (`random_state` $=$ 42) for model selection; for the probe, the logistic regression model is fit on the entire training set without an additional validation split. Sequences are tokenized with the Nucleotide Transformer tokenizer, which uses non-overlapping 6-mers, and truncated or padded to a fixed token budget. An attention mask is generated alongside each sequence so that padded positions are excluded from pooling.

3.2.2 Pretrained Model

We load the 500M-parameter Nucleotide Transformer checkpoint (`InstaDeepAI/nucleotide-transformer-500m-human-ref`) from Hugging Face. In both arms the backbone weights are frozen: in the probe arm no backbone parameter is updated at all, and in the LoRA arm only the inserted adapter matrices and the classification head receive gradient updates. The probe arm therefore reflects the quality of the pretrained representations directly, while the LoRA arm captures the marginal benefit of lightweight task-specific adaptation [1].

3.2.3 Frozen Probe

For each tokenized sequence, the frozen backbone is run in evaluation mode with `output_hidden_states=True` and the final hidden state is mean-pooled across non-padding tokens to obtain a single hidden-size-dimensional sequence-level representation. Embeddings are extracted in mini-batches of size 16 at a maximum sequence length of 64 tokens.[1] A logistic regression classifier is then trained on the pooled embeddings using a scikit-learn pipeline that standardizes features and fits a regularized logistic regression model with $C = 1.0$ and a maximum of $2{,}000$ optimization iterations. Because the frozen backbone produces deterministic embeddings, no validation set or early stopping is required, and the trained probe is evaluated directly on the held-out test split.

3.2.4 LoRA Fine-Tuning

Adapter Configuration. We attach LoRA adapters [4] to the *query*, *key*, and *value* projection matrices of every self-attention block in the Nucleotide Transformer backbone using the Hugging Face PEFT library with `TaskType.FEATURE_EXTRACTION`. We use rank $r = 8$, scaling factor $\alpha = 16$, adapter dropout 0.1, and no bias adaptation. All non-LoRA parameters of the backbone are explicitly frozen; only the LoRA matrices and a newly initialized two-class linear head (preceded by a dropout layer with $p = 0.1$) receive gradient updates. Sequences are tokenized to a maximum length of 256 tokens for this arm to give the adapters access to longer DNA context during training.

Training Setup. On top of the mean-pooled final hidden state, the linear head produces logits for the two classes and is trained with cross-entropy loss. We use AdamW with learning rate 5×10^{-4} and

[1] NT's 6-mer tokenization yields roughly 42 tokens for a 251 bp promoter sequence, so 64 tokens is a safe budget for `human_nontata_promoters`; longer datasets are truncated to the same budget.

weight decay 1×10^{-4}, a per-device batch size of 8, and gradient accumulation over 2 steps for an effective batch size of 16. We train for 5 epochs with a linear learning-rate schedule and a warmup ratio of 10%, applying gradient clipping with maximum norm 1.0. After each epoch we evaluate on the validation split and retain the checkpoint with the best validation macro F1; this checkpoint is used for the final test-set evaluation.

3.2.5 Evaluation Metrics

For both arms we report accuracy, F1, and ROC-AUC on the held-out test set. Accuracy gives a coarse measure of overall classification quality, F1 captures the precision–recall trade-off and is more informative when class balance shifts across datasets, and ROC-AUC summarizes the classifier's ranking quality independently of any specific decision threshold. Together, these three metrics let us compare frozen-probe and LoRA performance along complementary axes.

3.3 HyenaDNA

For HyenaDNA [7], we evaluate three binary classification datasets from Genomic Benchmarks: `human_nontata_promoters`, `human_enhancers_cohn`, and `drosophila_enhancers_stark`. These datasets cover both promoter and enhancer prediction tasks. Each sequence is tokenized with the HyenaDNA tokenizer and padded or truncated to a fixed length. The original training split is divided into training and validation sets using a stratified split, while the original test split is reserved for final evaluation.

We use pretrained HyenaDNA checkpoints from LongSafari [6], including `LongSafari/hyenadna-small-1k-seqlen-hf` and `LongSafari/hyenadna-small-32k-seqlen-hf`. The 1k model is used for shorter input settings, while the 32k model is used for longer-context experiments. A lightweight classification head is added on top of the backbone. We apply mean pooling over the final hidden states, excluding padding tokens with the attention mask, and pass the pooled representation to the classifier.

HyenaDNA is evaluated using the same adaptation strategies used in the rest of the project, including LoRA-based fine-tuning, last-layer unfreezing, progressive unfreezing, and vanilla full fine-tuning. We also include majority-class and uniform-random baselines for comparison. Models are trained with cross-entropy loss and evaluated using accuracy, macro F1 score, and MCC. The final test result is reported from the checkpoint with the best validation macro F1 score.

4 Result

4.1 DNABERT2

4.1.1 Probe

Overall Performance The best linear probe results come from layer 6 with mean pooling (F1 $\approx$ 0.854, accuracy $\approx$ 0.852, and MCC $\approx$ 0.714). The best MLP probe results come from layer 3 with mean pooling (F1 $\approx$ 0.928, accuracy $\approx$ 0.924, and MCC $\approx$ 0.851).

The improvements are approximately 8.6% in F1, 8.4% in accuracy, and 19.2% in MCC, suggesting that the learned representations are not purely linearly separable.

Effect of Layer Depth Performance varies depending on which layer is used. For the linear probe, performance improves as we move deeper into the model and peaks around layer 6, followed by a slight decline. For the MLP probe, the best performance appears earlier (around layer 3). This suggests that useful features are already present in early layers but are not easily captured by a linear model.

Linear vs MLP The gap between MLP and linear probe performance illustrates how complex the learned representations are across layers. The largest gap occurs at the embedding layer, indicating that the raw embeddings already contain useful information but are not linearly separable. The gap decreases and reaches its minimum around layer 6, suggesting that intermediate layers produce more linearly separable features. However, the gap increases again in later layers, indicating that representations become more specialized and less linearly separable.

Table 1: Full pooling ablation on the test set: accuracy, F1, and MCC for each (method × pooling) cell on the three Genomic Benchmarks tasks. Best value per dataset in bold; best value per method-row underlined.

		Mean pooling			**Max pooling**			**CLS pooling**		
Dataset	**Method**	Acc	F1	MCC	Acc	F1	MCC	Acc	F1	MCC
Promoters (human)	Frozen baseline	0.850	0.850	0.716	0.801	0.801	0.626	0.826	0.826	0.670
	Unfreeze last 4	0.942	0.942	0.884	**0.956**	**0.956**	**0.912**	0.929	0.929	0.862
	Progressive unfreeze	0.948	0.948	0.896	0.946	0.946	0.894	0.920	0.920	0.845
	LoRA r=8	0.942	0.942	0.886	0.944	0.944	0.889	0.940	0.940	0.880
Enhancers (human)	Frozen baseline	0.749	0.749	0.500	0.714	0.713	0.429	0.734	0.734	0.469
	Unfreeze last 4	0.761	0.761	0.522	0.758	0.758	0.515	0.755	0.755	0.509
	Progressive unfreeze	0.760	0.760	0.521	0.753	0.753	0.505	0.748	0.748	0.497
	LoRA r=8	**0.766**	**0.766**	**0.532**	0.761	0.760	0.522	0.765	0.764	0.532
Enhancers (drosophila)	Frozen baseline	0.720	0.708	0.478	0.699	0.692	0.417	0.728	0.722	0.477
	Unfreeze last 4	0.735	0.724	0.509	**0.811**	**0.811**	**0.624**	0.768	0.765	0.554
	Progressive unfreeze	0.729	0.719	0.492	0.768	0.765	0.549	0.761	0.755	0.547
	LoRA r=8	0.792	0.792	0.584	0.743	0.734	0.519	0.765	0.765	0.532

4.1.2 Fine-tuning

Effect of Fine-Tuning All three fine-tuning methods substantially improve over the frozen baseline across all datasets. On the two human tasks, the three strategies produce nearly identical results: F1 clusters around 0.942–0.948 on promoters and 0.760–0.766 on human enhancers. Given this equivalence, LoRA is the most practical choice, achieving comparable performance while training only 1.29% of total parameters versus approximately 32% for the unfreezing methods.

On Drosophila, differences between methods become more pronounced. Since DNABERT-2 was pretrained primarily on mammalian genomes, its representations are less well-adapted to insect sequences, making backbone adaptation more critical. Partial unfreezing achieves the best overall result of F1 = 0.811, outperforming LoRA (F1 = 0.792) and the frozen baseline (F1 = 0.708) by a clear margin.

Effect of Pooling Strategy The pooling strategy interacts with the fine-tuning method in a task-dependent manner.

For the frozen baseline and LoRA, mean pooling consistently performs best or on par across all three datasets, confirming that aggregating over all token embeddings yields more robust sequence-level representations.

Partial unfreezing, however, benefits considerably from max pooling. By directly modifying transformer layers, partial unfreezing encourages the backbone to produce sharper, more localized activations at motif-relevant positions. Max pooling then selects these peak signals rather than diluting them through averaging. It is particularly pronounced on Drosophila, where the model must adapt more substantially from its pretraining distribution to represent insect-specific regulatory patterns.

CLS pooling consistently yields the weakest results across methods and datasets, confirming that the first token embedding is a less informative summary than pooling over the full sequence in this setting.

4.2 Nucleotide Transformer

We compare two adaptation strategies for the Nucleotide Transformer 500M model: a frozen probe, where the pretrained backbone is kept fixed and only a classifier head is trained, and LoRA fine-tuning, where a small number of low-rank trainable parameters are added to the model. The overall comparison is shown in Figure 1, and the ROC curves are shown in Figure 2.

Overall, LoRA fine-tuning consistently improves performance over the frozen probe baseline across all three datasets. On Human non-TATA promoters,

LoRA gives the strongest performance, improving accuracy from 0.829 to 0.859, F1 score from 0.818 to 0.852, and ROC-AUC from 0.904 to 0.931. This suggests that the pretrained Nucleotide Transformer already captures useful promoter-related representations, while LoRA further adapts these representations to the downstream classification task.

For Human enhancers (Cohn), the accuracy improvement is small, increasing from 0.720 to 0.722. However, the F1 score improves more clearly, from 0.715 to 0.750, and ROC-AUC increases from 0.801 to 0.820. This means that LoRA does not greatly change the total number of correct predictions, but it improves the balance of the classifier and its ability to separate enhancer and non-enhancer sequences.

The improvement is also clear on Drosophila enhancers (Stark). LoRA increases accuracy from 0.638 to 0.682, F1 score from 0.630 to 0.658, and ROC-AUC from 0.684 to 0.727. Although this dataset remains the most difficult among the three, the gain suggests that LoRA helps the model adapt better to enhancer prediction in a different species.

Figure 2 further supports these results. In all three datasets, the LoRA ROC curve is above the frozen probe curve, showing better discrimination between positive and negative samples. The Human promoter task has the highest ROC-AUC for both methods, while Drosophila enhancer prediction has the lowest ROC-AUC, suggesting that cross-species enhancer prediction is more challenging for the model.

In summary, LoRA fine-tuning provides consistent gains over frozen probing while keeping the adaptation lightweight. The improvement is especially clear in F1 score and ROC-AUC, showing that LoRA improves not only raw accuracy but also the model's ability to distinguish regulatory from non-regulatory sequences.

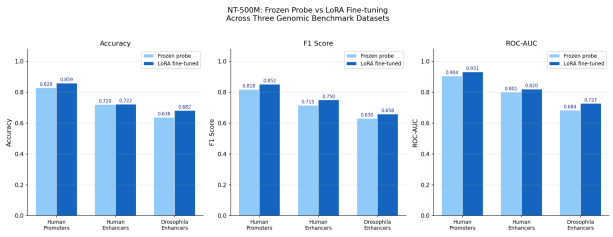

Figure 1: Performance comparison between frozen probing and LoRA fine-tuning on three Genomic Benchmarks datasets.

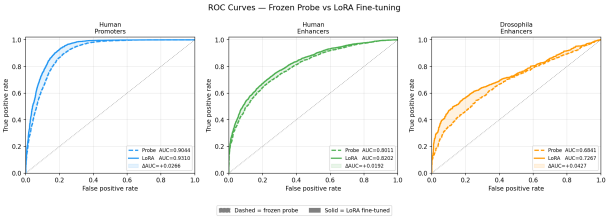

Figure 2: ROC curves comparing frozen probing and LoRA fine-tuning on Human promoters, Human enhancers (Cohn), and Drosophila enhancers (Stark).

4.3 HyenaDNA

4.3.1 Overall Performance

Tables 2 and 3 report the fine-tuning results for HyenaDNA Tiny-1K and Small-32K respectively across three Genomic Benchmarks datasets using accuracy, macro F1 score, and MCC as evaluation metrics. Overall, both models benefit from backbone adaptation, but the degree of improvement and the best-performing strategy vary by dataset and model scale.

On `human_nontata_promoters`, both models achieve the strongest results with progressive unfreezing. Tiny-1K progressive unfreezing reaches an F1 of 0.909 and MCC of 0.818, while Small-32K Last 4 layers + LoRA achieves a comparable F1 of 0.907 and MCC of 0.818. The frozen backbone setting is notably weaker for Tiny-1K (F1 of 0.768) compared to Small-32K No unfreezing + LoRA (F1 of 0.874), suggesting that the larger model's pretrained representations are more directly transferable to promoter classification without backbone adaptation.

On `human_enhancers_cohn`, both models achieve moderate performance across all settings. For Tiny-1K, LoRA performs best with an F1 of 0.728, while for Small-32K vanilla full fine-tuning edges ahead at F1 of 0.733. The small gap between strategies on both models suggests that this task does not strongly require deep backbone adaptation, and that the pretrained representations already capture relevant enhancer patterns.

On `drosophila_enhancers_stark`, the two models diverge most clearly. Tiny-1K's frozen

Table 2: HyenaDNA Tiny-1K Fine-Tuning results across three Genomic Benchmarks datasets. Best result per dataset is shown in **bold**.

Dataset	Setting	Accuracy	F1 (macro)	MCC
human_nontata_promoters	Frozen Backbone	0.768	0.768	0.549
	Unfreeze Last Block	0.815	0.815	0.636
	Progressive Unfreezing	**0.910**	**0.909**	**0.818**
	LoRA	0.856	0.856	0.718
human_enhancers_cohn	Frozen Backbone	0.664	0.663	0.331
	Unfreeze Last Block	0.691	0.691	0.383
	Progressive Unfreezing	**0.722**	**0.721**	**0.445**
	LoRA	0.728	0.728	0.457
drosophila_enhancers_stark	Frozen Backbone	0.500	0.349	0.000
	Unfreeze Last Block	0.521	0.462	0.065
	Progressive Unfreezing	**0.673**	**0.668**	**0.355**
	LoRA	0.651	0.645	0.317

backbone collapses entirely to the majority-class baseline (MCC of 0.000), while Small-32K's frozen + LoRA setting also fails (F1 of 0.333). However, Small-32K recovers more strongly with progressive + LoRA (F1 of 0.709, MCC of 0.421) compared to Tiny-1K progressive unfreezing (F1 of 0.668, MCC of 0.355). This gap reflects the capacity advantage of Small-32K on the harder cross-species task, where its four blocks and higher hidden dimension provide more representational flexibility for adapting to Drosophila enhancer sequences.

Overall, progressive unfreezing consistently performs well across all three datasets, particularly on the more challenging Drosophila task, where gradual adaptation appears to preserve pretrained representations while still allowing sufficient task-specific learning. LoRA provides a competitive lightweight alternative, especially on the human datasets, though it struggles on Drosophila without any direct backbone weight updates. The frozen backbone setting performs adequately on the human tasks but fails on the cross-species enhancer task, indicating that the pretrained HyenaDNA Tiny-1K representations do not fully transfer to Drosophila enhancer sequences without adaptation.

4.3.2 Effect of Model Scale

Comparing the two models reveals a consistent pattern: Small-32K outperforms Tiny-1K across all three datasets and most settings, but the margin depends on task difficulty. On the relatively easy promoter task, the performance gap is small — Tiny-1K progressive unfreezing (F1 of 0.909) nearly matches Small-32K Last 4 layers + LoRA (F1 of 0.907). On the harder Drosophila enhancer task, the gap widens considerably, with Small-32K achieving an MCC of 0.421 compared to Tiny-1K's 0.355 under their respective best settings. This suggests that model scale becomes increasingly important as task difficulty increases, particularly for cross-species generalization.

4.3.3 Effect of Fine-Tuning Strategy

The results show that no single fine-tuning strategy performs best across all datasets. LoRA-based methods are competitive with vanilla full fine-tuning, especially on `human_nontata_promoters`, where both Last 4 layers + LoRA and Progressive + LoRA outperform full fine-tuning. This suggests that updating a smaller set of task-specific parameters can be enough when the pretrained representation already transfers well to the target task.

The main difference appears on the more difficult enhancer tasks. On `human_enhancer`

Table 3: HyenaDNA Small-32k fine-tuning results across three Genomic Benchmarks datasets. The best result for each dataset is shown in bold.

Dataset	Setting	Accuracy	F1	MCC
human_nontata_promoters	Majority-class baseline	0.544	0.352	0.000
human_nontata_promoters	Uniform-random baseline	0.502	0.501	0.003
human_nontata_promoters	No unfreezing + LoRA	0.875	0.874	0.748
human_nontata_promoters	Last 4 layers + LoRA	**0.908**	**0.907**	**0.818**
human_nontata_promoters	Progressive + LoRA	0.905	0.904	0.808
human_nontata_promoters	Vanilla full FT	0.893	0.893	0.791
human_enhancers_cohn	Majority-class baseline	0.500	0.333	0.000
human_enhancers_cohn	Uniform-random baseline	0.495	0.494	-0.011
human_enhancers_cohn	No unfreezing + LoRA	0.725	0.724	0.451
human_enhancers_cohn	Last 4 layers + LoRA	0.730	0.730	0.459
human_enhancers_cohn	Progressive + LoRA	0.726	0.726	0.451
human_enhancers_cohn	Vanilla full FT	**0.733**	**0.733**	**0.466**
drosophila_enhancers_stark	Majority-class baseline	0.500	0.333	0.000
drosophila_enhancers_stark	Uniform-random baseline	0.498	0.498	-0.005
drosophila_enhancers_stark	No unfreezing + LoRA	0.500	0.333	0.000
drosophila_enhancers_stark	Last 4 layers + LoRA	0.693	0.691	0.391
drosophila_enhancers_stark	Progressive + LoRA	**0.710**	**0.709**	**0.421**
drosophila_enhancers_stark	Vanilla full FT	0.500	0.333	0.000

s_cohn, full fine-tuning gives the highest score, but the LoRA-based methods remain close. On drosophila_enhancers_stark, Progressive + LoRA performs best, while both No unfreezing + LoRA and vanilla full fine-tuning stay near baseline. This suggests that immediately updating either too few parameters or the full backbone may not be ideal for this task. Gradually unfreezing the backbone provides a more stable adaptation path.

Overall, the fine-tuning results suggest that progressive unfreezing is most useful when the target task requires stronger adaptation, while simpler LoRA-based fine-tuning can work well when the pretrained HyenaDNA features already align with the downstream dataset.

5 Conclusion / Future Work

Overall, our results show that lightweight adaptation can substantially improve pretrained DNA foundation models, but the best strategy depends on both the model family and the dataset. LoRA is consistently useful for Nucleotide Transformer and competitive for DNABERT-2 and HyenaDNA on human tasks. However, cross-species enhancer prediction is more difficult and often requires stronger backbone adaptation, such as partial or progressive unfreezing. These findings suggest that frozen representations contain useful biological information, but task-specific adaptation remains important, especially when the downstream task differs from the model's pretraining distribution.

In this work, we evaluated DNA models on binary classification tasks across three Genomic Benchmarks datasets. Extending the evaluation to multi-class genomic tasks, such as regulatory element type classification or splice site prediction, would help assess how well the observed patterns generalize to more complex prediction settings.

In addition, the partial unfreezing experiments used a fixed configuration of the last four layers. A more systematic search over the number of unfrozen layers could reveal which parts of the model contribute most to task-specific adaptation, and whether comparable performance can be achieved by unfreezing fewer layers.

6 Team Contribution

Runpeng He: Implemented and evaluated LoRA fine-tuning for the Nucleotide Transformer model across three Genomic Benchmarks datasets. I han-

dled the full experimental pipeline, including data preprocessing, model training, evaluation, and visualization. I also compared frozen probing with lightweight fine-tuning strategies and analyzed their performance differences using accuracy, F1 score, and ROC-AUC.

Jiaxuan Cao: Implemented the DNABERT-2 probing pipeline, including hook-based hidden-state extraction across all 13 layers (with a custom repadding step to handle DNABERT-2's unpadded attention) and both linear and MLP probes. Implemented LoRA fine-tuning and extended the evaluation from a single promoter task to three Genomic Benchmarks datasets.

Nan Huang: Built the DNABERT-2 probing pipeline foundation, including model and dataset loading, tokenization, pooling strategy definitions, and linear probe training. Implemented the partial unfreezing fine-tuning pipeline, serving as a shared template for the partial unfreezing fine-tuning experiments, and extended DNABERT-2 evaluation across three Genomic Benchmarks datasets.

Allen Tran: Worked on freeze-probing the nucleotide model and reporting results such as accuracy, F1, and ROC-AUC across the Genomic Benchmark datasets.

Natalie Kao: Implemented and evaluated lightweight fine-tuning of the HyenaDNA Tiny-1k model across three genomic benchmark datasets from preprocessing, training, to evaluation. Also conducted linear and MLP probing experiments on subset of data to understand structure of pretrained embeddings. Contributed to the report by writing the background section, HyenaDNA related work and experimental results on Tiny-1k model.

Zuge Li: Implemented and evaluated the HyenaDNA small-32k model for long-context genomic sequence classification. Ran fine-tuning experiments on Genomic Benchmarks datasets, collected test accuracy, F1, and MCC results, and organized the results into tables and heatmap visualizations. Contributed to the HyenaDNA methodology and result sections, focusing on the 32k model setup and performance analysis.

References

[1] Hugo Dalla-Torre, Liam Gonzalez, Javier Mendoza-Revilla, Nicolas Lopez Carranza, Adam Henryk Grzywaczewski, Francesco Ober, et al. The nucleotide transformer: Building and evaluating robust foundation models for human genomics. *bioRxiv*, 2023. doi: 10.1101/2023.01.11.523679.

[2] Haonan Feng, Lang Wu, Bingxin Zhao, Chad Huff, Jianjun Zhang, Jia Wu, Lifeng Lin, Peng Wei, and Chong Wu. Benchmarking dna foundation models for genomic and genetic tasks. *Nature Communications*, 2025. doi: 10.1038/s41467-025-65823-8. URL https://doi.org/10.1038/s41467-025-65823-8.

[3] Katarína Grešová, Vlastimil Martinek, David Čechák, Petr Šimeček, and Panagiotis Alexiou. Genomic benchmarks: a collection of datasets for genomic sequence classification. *BMC Genomic Data*, 2023. doi: 10.1186/s12863-023-01123-8. URL https://doi.org/10.1186/s12863-023-01123-8.

[4] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://arxiv.org/abs/2106.09685.

[5] 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, August 2021. doi: 10.1093/bioinformatics/btab083. URL https://doi.org/10.1093/bioinformatics/btab083.

[6] LongSafari. Hyenadna models. https://huggingface.co/collections/LongSafari/hyenadna-models, 2023.

[7] Eric Nguyen, Michael Poli, Marjan Faizi, Armin Thomas, Callum Birch-Sykes, Michael Wornow, Aman Patel, Clayton Rabideau, Stefano Massaroli, Yoshua Bengio, Stefano Ermon,

Stephen A. Baccus, and Chris Ré. Hyenadna: Long-range genomic sequence modeling at single nucleotide resolution. https://arxiv.org/abs/2306.15794, 2023.

[8] Zhihan Zhou, Yanrong Ji, Weijian Li, Pratik Dutta, Ramana V. Davuluri, and Han Liu. Dnabert-2: Efficient foundation model and benchmark for multi-species genomes. *arXiv preprint arXiv:2306.15006*, 2023. URL https://arxiv.org/abs/2306.15006.

[9] Zhihan Zhou, Yanrong Ji, Weijian Li, Pratik Dutta, Ramana V. Davuluri, and Han Liu. Dnabert-2-117m. https://huggingface.co/zhihan1996/DNABERT-2-117M, 2023.