Improving Genomic Models via Task-Specific Self-Pretraining

Sohan Mupparapu[1] **Parameswari Krishnamurthy**[1] **Ratish Puduppully**[2]

Abstract

Pretraining DNA language models (DNALMs) on the full human genome is resource-intensive, yet often considered necessary for strong downstream performance. Inspired by recent findings in NLP and long-context modeling, we explore an alternative: self-pretraining on task-specific, unlabeled data. Using the BEND benchmark, we show that DNALMs trained with self-pretraining match or exceed the performance of models trained from scratch under identical compute. While genome-scale pretraining may still offer higher absolute performance, task-specific self-pretraining provides a practical and compute-efficient strategy for building stronger supervised baselines.

1. Introduction

DNA language models (DNALMs) (Ji et al., 2021; Dalla-Torre et al., 2024; Nguyen et al., 2023) have emerged as powerful tools for modeling and interpreting genomic sequences, drawing inspiration from advances in natural language processing (NLP). These models are typically pretrained on large-scale genomic corpora—such as the human genome.

Recent benchmarks like BEND (Marin et al., 2024) highlight that while DNALMs perform well on certain tasks, they struggle with long-range dependencies and exhibit inconsistent performance across biologically meaningful benchmarks. At the same time, findings from long-sequence modeling (Amos et al., 2024) suggest that self-pretraining—pretraining on the downstream task data itself—followed by supervised fine-tuning can outperform training from scratch, particularly for long-context tasks.

Motivated by these observations, we investigate whether self-pretraining on task-specific genomic data can yield stronger supervised models under limited compute. Using the BEND benchmark, we find that models trained with task-specific self-supervision match or exceed the performance of scratch-trained counterparts across multiple tasks. While not a substitute for genome-scale pretraining in all settings, our results show that self-pretraining is a compute-efficient strategy for improving downstream performance and building stronger supervised baselines.

This approach is particularly appealing in genomics, where sequences for a given task are abundant but labels are often scarce. By leveraging task-specific unlabeled data, self-pretraining enables models to learn domain-relevant patterns before fine-tuning, improving performance with minimal overhead.

We release the code, pretrained model and finetuned models to support reproducibility[1].

2. Related Work

A growing body of work has applied language modeling to DNA sequences, resulting in several pretrained genomic models. DNABERT (Ji et al., 2021) introduced masked language modeling on k-mer tokenized data, showing utility for motif discovery and promoter prediction. Later models like the Nucleotide Transformer (Dalla-Torre et al., 2024) and HyenaDNA (Nguyen et al., 2023) scaled to larger genomes and longer contexts, improving performance on regulatory genomics tasks. GENA-LM (Fishman et al., 2023) uses efficient Transformer variants to model full-length sequences. Caduceus (Schiff et al., 2024) further improves long-range modeling by incorporating reverse complement symmetry and scalable bidirectional blocks based on Mamba.

Beyond these pretrained models, recent work across NLP and biology has begun to question the necessity of large-scale pretraining for strong downstream performance. Krishna et al. (2023) demonstrate that language models pretrained solely on downstream data can match or exceed models trained on large corpora such as BookWiki for tasks like classification and QA. Similarly, Amos et al. (2024) show that in long-sequence modeling, self-pretraining on task data substantially narrows the gap between standard Transformers and more complex architectures like S4 (Gu et al., 2022) on Long Range Arena tasks (Tay et al., 2021),

[1] IIIT Hyderabad, India [2] IT University of Copenhagen, Denmark. Correspondence to: Ratish Puduppully <rapu@itu.dk>.

Proceedings of the Workshop on Generative AI for Biology at the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025. Copyright 2025 by the author(s).

[1] https://github.com/SohanMupparapu/dna_spt

suggesting that task-specific pretraining can serve as an effective inductive prior.

In the biological domain, Xu et al. (2025) report that foundation models (FMs) in genomics often fail to outperform well-tuned supervised baselines, despite significantly higher computational costs. Supporting this, Vishniakov et al. (2024) show that randomly initialized models can match or even outperform pretrained genomic models across tasks like enhancer and histone mark prediction. Our work builds on these insights in the context of DNA language modeling by focusing on strengthening supervised baselines through self-pretraining—an approach that is both compute-efficient and independent of large upstream corpora.

3. Tasks in BEND

We evaluate on four tasks from the BEND benchmark (Marin et al., 2024) (see Appendix Table 3 for an overview).

Gene Finding. A multiclass sequence labeling task where each nucleotide is labeled as exon, intron, splice site (donor/acceptor, forward/reverse strand), or noncoding. Derived from GENCODE (Frankish et al., 2021) annotations, this task evaluates both local and long-range understanding of gene structure. Performance is measured with multiclass MCC (Gorodkin, 2004).

Chromatin Accessibility Prediction. A multilabel classification task where models predict DNase I hypersensitivity across 125 human cell types from 512 bp sequences. Labels are derived from ENCODE DNase-seq data (ENCODE Project Consortium, 2012), following preprocessing from Kelley et al. (2016). Each label indicates whether the chromatin is open in a specific cell type. Evaluation is based on average AUROC across labels.

Histone Modification Prediction. A multilabel task predicting the presence of 18 histone modifications from 512 bp sequences in the K562 cell line. Labels are based on ENCODE ChIP-seq data for 11 histone marks across 18 replicates (ENCODE Project Consortium, 2012). Each label indicates whether a particular mark is bound to the sequence. AUROC is averaged across all histone marks.

CpG Methylation Prediction. A multilabel classification task to predict the methylation status of CpG sites across 7 human cell lines using 512 bp windows centered on CpG sites. Labels are derived from ENCODE bisulfite sequencing data (ENCODE Project Consortium, 2012). AUROC is averaged over all cell lines.

4. Methods

4.1. Model and Pretraining Setup

We use a residual convolutional neural network (CNN) as a shared encoder across tasks. The architecture, inspired by ResNet (He et al., 2016) and the BEND benchmark (Marin et al., 2024), consists of 30 convolutional layers (kernel size 9) with 512 hidden channels. Dilation doubles with each layer (reset every 6 layers, max 32), and we use GELU activation and LayerNorm.

During self-supervised pretraining, we attach a masked language modeling (MLM) head to the encoder and train on unlabeled DNA sequences. Inputs are one-hot encoded over a vocabulary of size $V = 7$ (`A`, `C`, `G`, `T`, `N`, `[MASK]`, `[PAD]`). Tokens are masked independently with probability $p_{\text{mlm}} = 0.15$, using the standard 80/10/10 replacement strategy. The MLM loss is computed as $\mathcal{L}_{\text{MLM}} = -\sum_{i:m_i=1} \log P_\theta(x_i \mid \widetilde{x})$, where m_i indicates masked positions and $\widetilde{x}$ is the corrupted input.

4.2. Supervised Fine-Tuning and Training Configuration

After pretraining, we replace the MLM head with a task-specific predictor consisting of a two-layer CNN (stride 1, ReLU activation) and a linear output layer. The model is fine-tuned end-to-end on downstream tasks. We use cross-entropy loss for single-label classification (e.g., gene finding) and binary cross-entropy loss for multi-label tasks (e.g., chromatin accessibility, histone modifications, CpG methylation).

All models are implemented in PyTorch and trained on a mix of L40S and A100 GPUs, using code adapted from the BEND benchmark. Pretraining is done on 4,780 gene-finding samples over 20 epochs. Fine-tuning is performed for 10 epochs on gene finding and 5 epochs on the other tasks. The latter are trained for fewer epochs to reduce compute, as they have larger training sets and longer per-epoch runtimes.

4.3. Pretraining Corpus and Evaluation

We use gene-finding sequences for pretraining and reuse the resulting encoder across all tasks. These sequences are the longest in the BEND benchmark (1,433–14,000 bp), enabling modeling of long-range dependencies. Although labels are not used during pretraining, the sequences cover biologically structured regions such as exons, introns, and splice sites—features closely linked to regulatory activity and downstream task relevance.

Table 1. Performance of models trained from scratch, with self-pretraining (SPT), and SPT with CRF. CRF improves gene finding by modeling label transitions.

	Gene finding	**Chromatin accessibility**	**Histone modification**	**CpG Methylation**
scratch	0.38	0.87	0.76	0.89
SPT	0.50	0.87	0.77	0.94
SPT + CRF	0.64	–	–	–

5. Results

We evaluate whether self-supervised pretraining on task-specific data (*SPT*) improves downstream performance compared to training from scratch (*scratch*). We begin with a compute-matched comparison using the same ResNet architecture, and then benchmark *SPT* against expert methods and genome-pretrained models from the BEND benchmark.

Compute-Matched Comparison. Table 1 shows that *SPT* consistently achieves equal or better performance than *scratch* under identical compute. Gains are especially notable for gene finding and CpG methylation, with absolute improvements of 12 and 5 points, respectively. For chromatin accessibility and histone modification, both models perform similarly—suggesting that these tasks are already well-solved by supervised training, leaving limited room for additional improvement via pretraining. Notably, even a shallow CNN (Table 2, row "CNN") performs competitively on histone modification, indicating that this task may be relatively insensitive to architectural depth or pretraining.

Improving Gene Finding via Sequence Labeling. In the BEND benchmark, gene finding is framed as a per-base classification task. To explore whether modeling label dependencies improves performance, we augment our *SPT* model with a neural linear-chain Conditional Random Field (CRF) (Lample et al., 2016; Lafferty et al., 2001), drawing inspiration from the structured HMM-based approach used in AUGUSTUS (Rabiner, 1989; Stanke & Waack, 2003). The CRF layer captures transition dynamics across sequence positions, promoting globally coherent predictions (e.g., valid exon–intron boundaries). Using the `torchcrf` implementation,[2] we observe a substantial improvement: MCC increases from 0.50 to 0.64 (Table 1). This gain suggests that gene boundaries and exon-intron transitions benefit from structured prediction, which enforces global label consistency—something per-position classifiers cannot capture. CRFs provide an inductive bias toward valid biological transitions, which may be especially helpful in low-data settings.

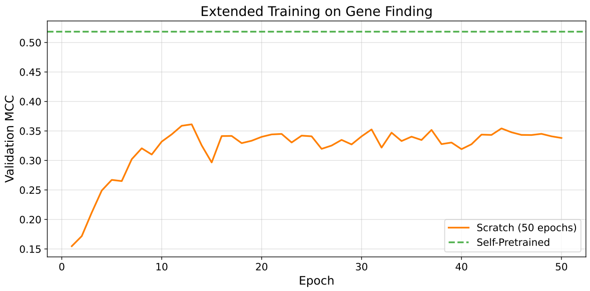

Figure 1. Validation MCC on gene finding. The scratch model plateaus early and does not match the self-pretrained model.

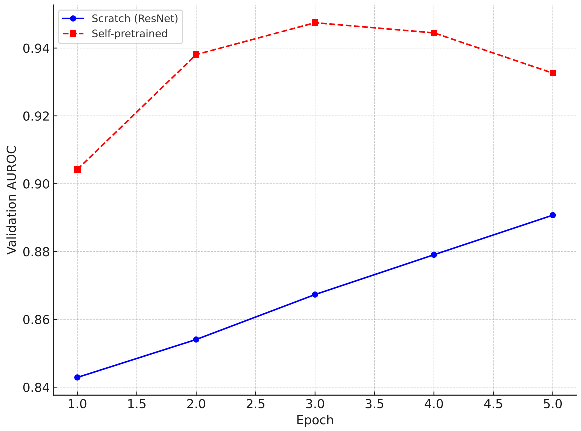

Figure 2. Validation AUROC on the CpG methylation task over training epochs.

Extended Training Comparison. To assess whether longer training improves performance for *scratch* models, we train the *scratch* model on gene finding for 50 epochs—substantially more than the 10 used in our earlier comparison. Our *SPT* model, by contrast, uses 20 epochs of unsupervised pretraining followed by supervised fine-tuning. As shown in Figure 1, the *scratch* model improves initially but plateaus around epoch 13 and fails to match the performance of the *SPT* model. A dashed horizontal line indicates the best validation MCC achieved by the *SPT* model, highlighting the efficiency of self-pretraining over prolonged supervised training.

We now focus on CpG methylation, where *SPT* shows strong gains as well.

Training Dynamics. Figure 2 shows validation AUROC over training epochs. The performance gap between *SPT* and *scratch* appears from the first epoch and persists throughout, demonstrating the benefits of task-specific self-pretraining.

[2] https://github.com/rikeda71/TorchCRF

Table 2. Performance on selected BEND benchmark tasks. We compare expert methods, supervised baselines, genome-pretrained DNA LMs, and our self-pretrained (*SPT*) models. *SPT + CRF*† augments the gene finding model with a linear-chain CRF decoder. Best results for each task are shown in bold.

	Gene finding	**Chromatin accessibility**	**Histone modification**	**CpG Methylation**
Expert method	0.80 Augustus	0.85 Basset	0.74 Basset	0.93 Basset
Fully supervised				
ResNet	0.46	-	-	-
CNN	0.00	0.75	0.76	0.84
Pre-trained				
ResNet-LM	0.36	0.82	0.77	0.87
AWD-LSTM	0.05	0.69	0.74	0.81
NT-H	0.41	0.74	0.76	0.88
NT-MS	**0.68**	0.79	0.78	0.92
NT-1000G	0.49	0.77	0.77	0.89
NT-V2	0.64	0.80	0.76	0.91
DNABERT	0.20	0.85	**0.79**	0.91
DNABERT-2	0.43	0.81	0.78	0.90
GENA-LM BERT	0.52	0.76	0.78	0.91
GENA-LM BigBird	0.39	0.82	0.76	0.91
HyenaDNA large	0.35	0.84	0.76	0.91
HyenaDNA tiny	0.10	0.78	0.76	0.86
GROVER	0.28	0.82	0.77	0.89
SPT	0.50	**0.87**	0.77	**0.94**
SPT + CRF†	0.64	–	–	–

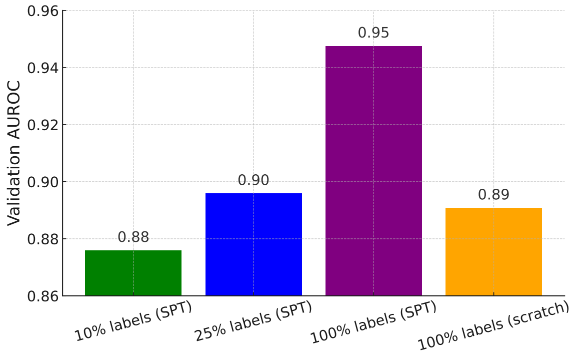

Figure 3. Maximum validation AUROC on CpG methylation using different fractions of labeled data. The 25% self-pretrained (SPT) model outperforms the 100% scratch model, while the 10% SPT model performs slightly below.

Sample Efficiency To assess whether self-pretraining improves sample efficiency, we train the *SPT* model on the CpG methylation task using varying fractions of the labeled training data: 10%, 25%, and 100%. The self-pretrained encoder is kept fixed across all fine-tuning settings. As shown in Figure 3, the *SPT* model trained with 25% of the data already outperforms the *scratch* model trained on 100%. The 10% variant performs slightly below *scratch* but still shows strong performance. These results highlight the value of self-pretraining for enabling accurate predictions in low-data regimes.

Comparison to Benchmark Models. We also compare our *SPT* model to expert methods and genome-pretrained DNA language models from the BEND benchmark (Marin et al., 2024). These include expert tools (e.g., Augustus (Stanke & Waack, 2003), Basset (Kelley et al., 2016)), supervised CNNs and ResNets, and large-scale pretrained models such as DNABERT (Ji et al., 2021), Nucleotide Transformer (Dalla-Torre et al., 2024), GENA-LM (Fishman et al., 2023), and HyenaDNA (Nguyen et al., 2023).

Table 2 reproduces BEND benchmark results and adds our *SPT* and *SPT + CRF* models. Despite using no genome-scale pretraining, *SPT* outperforms all supervised-from-scratch baselines (even when baselines are trained for up to 100 epochs) and matches or exceeds many genome-pretrained models. On the CpG methylation task, it achieves the highest AUROC overall. On gene finding, further gains are possible by treating the task as structured sequence labeling: adding a linear-chain CRF layer (*SPT + CRF*) boosts performance to 0.64 MCC.

These results demonstrate that task-specific self-pretraining can serve both as a strong supervised baseline and a practical alternative to genome-scale pretraining—especially under compute constraints.

6. Conclusion

We demonstrate that self-supervised pretraining on task-specific genomic data—using only unlabeled sequences from the gene-finding task—can improve downstream performance compared to models trained from scratch. Across tasks in the BEND benchmark, self-pretraining consistently yields stronger results under matched compute budgets. While it may not always outperform genome-wide pretraining in absolute terms, our approach provides a practical and compute-efficient alternative that establishes a stronger supervised baseline for genomic modeling.

Impact Statement

This paper presents work whose goal is to improve the efficiency and accessibility of machine learning for genomics by exploring task-specific self-pretraining. While our methods may have downstream applications in health and biological research, we do not foresee any immediate ethical or societal risks that require specific attention. There are many potential broader impacts of machine learning in genomics, none of which we feel must be specifically highlighted here.

References

Amos, I., Berant, J., and Gupta, A. Never train from scratch: Fair comparison of long-sequence models requires data-driven priors. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=PdaPky8MUn.

Dalla-Torre, H., Gonzalez, L., Romero, J., et al. The nucleotide transformer: Building and evaluating robust foundation models for human genomics. *Nature Methods*, 21:123–134, 2024. doi: 10.1038/s41592-024-02523-z. URL https://www.nature.com/articles/s41592-024-02523-z.

ENCODE Project Consortium. An integrated encyclopedia of dna elements in the human genome. *Nature*, 489 (7414):57–74, September 2012. ISSN 1476-4687. doi: 10.1038/nature11247.

Fishman, V., Kuratov, Y., Shmelev, A., Petrov, M., Penzar, D., Shepelin, D., Chekanov, N., Kardymon, O., and Burtsev, M. Gena-lm: A family of open-source foundational dna language models for long sequences. *bioRxiv*, 2023. doi: 10.1101/2023.06.12.544594. URL https://www.biorxiv.org/content/10.1101/2023.06.12.544594v1.

Frankish, A., Diekhans, M., Jungreis, I., Lagarde, J., Loveland, J. E., Mudge, J. M., Sisu, C., Wright, J. C., Armstrong, J., Barnes, I., Berry, A., Bignell, A., Boix, C., Carbonell Sala, S., Cunningham, F., Di Domenico, T., Donaldson, S., Fiddes, I. T., García Girón, C., Gonzalez, J. M., Grego, T., Hardy, M., Hourlier, T., Howe, K. L., Hunt, T., Izuogu, O. G., Johnson, R., Martin, F. J., Martínez, L., Mohanan, S., Muir, P., Navarro, F. C. P., Parker, A., Pei, B., Pozo, F., Riera, F. C., Ruffier, M., Schmitt, B. M., Stapleton, E., Suner, M.-M., Sycheva, I., Uszczynska-Ratajczak, B., Wolf, M. Y., Xu, J., Yang, Y. T., Yates, A., Zerbino, D., Zhang, Y., Choudhary, J. S., Gerstein, M., Guigó, R., Hubbard, T. J. P., Kellis, M., Paten, B., Tress, M. L., and Flicek, P. GENCODE 2021. *Nucleic Acids Research*, 49(D1):D916–D923, January 2021. ISSN 0305-1048. doi: 10.1093/nar/gkaa1087. URL https://doi.org/10.1093/nar/gkaa1087.

Gorodkin, J. Comparing two k-category assignments by a k-category correlation coefficient. *Computational Biology and Chemistry*, 28(5-6):367–374, December 2004. ISSN 1476-9271. doi: 10.1016/j.compbiolchem.2004.09.006. URL https://www.sciencedirect.com/science/article/pii/S1476927104000799.

Gu, A., Goel, K., and Re, C. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=uYLFoz1vlAC.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

Ji, Y., Zhou, Z., Liu, H., and Davuluri, R. V. Dnabert: pre-trained bidirectional encoder representations from transformers model for dna-language in genome. *Bioinformatics*, 37(15):2112–2120, 2021. doi: 10.1093/bioinformatics/btab083. URL https://academic.oup.com/bioinformatics/article/37/15/2112/6128680.

Kelley, D. R., Snoek, J., and Rinn, J. L. Basset: learning the regulatory code of the accessible genome with deep convolutional neural networks. *Genome Research*, 26(7):990–999, July 2016. ISSN 1549-5469. doi: 10.1101/gr.200535.115.

Krishna, K., Garg, S., Bigham, J., and Lipton, Z. Downstream datasets make surprisingly good pretraining corpora. In Rogers, A., Boyd-Graber, J., and Okazaki, N. (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12207–12222, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.682. URL https://aclanthology.org/2023.acl-long.682/.

Lafferty, J., McCallum, A., and Pereira, F. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, pp. 282–289, 2001.

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., and Dyer, C. Neural architectures for named entity recognition. In Knight, K., Nenkova, A., and Rambow, O. (eds.), *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 260–270, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1030. URL https://aclanthology.org/N16-1030/.

Marin, F. I., Teufel, F., Horlacher, M., Madsen, D., Pultz, D., Winther, O., and Boomsma, W. BEND: Benchmarking DNA language models on biologically meaningful tasks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=uKB4cFNQFg.

Nguyen, E., Poli, M., Faizi, M., Thomas, A. W., Sykes, C. B., Wornow, M., Patel, A., Rabideau, C., Massaroli, S., Bengio, Y., Ermon, S., Baccus, S. A., and Ré, C. Hyenadna: Long-range genomic sequence modeling at single nucleotide resolution. In *Advances in Neural Information Processing Systems*, volume 36, pp. 12345–12356, 2023. URL https://papers.neurips.cc/paper_files/paper/2023/file/86ab6927ee4ae9bde4247793c46797c7-Paper-Conference.pdf.

Rabiner, L. R. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

Schiff, Y., Kao, C.-H., Gokaslan, A., Dao, T., Gu, A., and Kuleshov, V. Caduceus: Bi-directional equivariant long-range dna sequence modeling. *arXiv preprint arXiv:2403.03234*, 2024. URL https://arxiv.org/abs/2403.03234.

Stanke, M. and Waack, S. Gene prediction with a hidden markov model and a new intron submodel. *Bioinformatics (Oxford, England)*, 19 Suppl 2:ii215–ii225, October 2003. ISSN 1367-4811. doi: 10.1093/bioinformatics/btg1080.

Tay, Y., Dehghani, M., Abnar, S., Shen, Y., Bahri, D., Pham, P., Rao, J., Yang, L., Ruder, S., and Metzler, D. Long range arena : A benchmark for efficient transformers. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=qVyeW-grC2k.

Vishniakov, K., Viswanathan, K., Medvedev, A., Kanithi, P. K., Pimentel, M. A., Rajan, R., and Khan, S. Genomic foundationless models: Pretraining does not promise performance. *bioRxiv*, 2024. doi: 10.1101/2024.12.18.628606. URL https://doi.org/10.1101/2024.12.18.628606.

Xu, Z., Gupta, R., Cheng, W., Shen, A., Shen, J., Talwalkar, A., and Khodak, M. Specialized foundation models struggle to beat supervised baselines. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=J7NmKXF5Ll.

Appendix

Table 3. Overview of four BEND benchmark tasks. Adapted from Marin et al. (2024).

Task	Type	# Samples	Length range	Evaluation	Metric	Source
Gene finding	Nucleotide-wise Multiclass (9)	5,976	1,433–14,000 bp	4780/597/597	MCC	GENCODE (Frankish et al., 2021)
Chromatin accessibility	Sequence-wise Multilabel (125)	2,005,617	512 bp	1,354,042/ 279,422/ 372,153	AUROC	ENCODE (ENCODE Project Consortium, 2012)
Histone modification	Sequence-wise Multilabel (18)	612,081	512 bp	70,801/ 120,567/ 743,095	AUROC	ENCODE (ENCODE Project Consortium, 2012)
CpG methylation	Sequence-wise Multilabel (7)	959,039	512 bp	109,717/ 106,227	AUROC	ENCODE (ENCODE Project Consortium, 2012)