# The Nucleotide Transformer: Building and Evaluating Robust Foundation Models for Human Genomics

Hugo Dalla-Torre[1], Liam Gonzalez[1], Javier Mendoza-Revilla[1], Nicolas Lopez Carranza[1], Adam Henryk Grzywaczewski[2], Francesco Oteri[1], Christian Dallago[2 3], Evan Trop[1], Hassan Sirelkhatim[2], Guillaume Richard[1], Marcin Skwark[1], Karim Beguir[1], Marie Lopez[*† 1], Thomas Pierrot[*† 1]

[1]**InstaDeep** [2]**Nvidia** [3]**TUM**

**Abstract**

Closing the gap between measurable genetic information and observable traits is a longstanding challenge in genomics. Yet, the prediction of molecular phenotypes from DNA sequences alone remains limited and inaccurate, often driven by the scarcity of annotated data and the inability to transfer learnings between prediction tasks. Here, we present an extensive study of foundation models pre-trained on DNA sequences, named the Nucleotide Transformer, integrating information from 3,202 diverse human genomes, as well as 850 genomes from a wide range of species, including model and non-model organisms. These transformer models yield transferable, context-specific representations of nucleotide sequences, which allow for accurate molecular phenotype prediction even in low-data settings. We show that the sequence representations alone match or outperform specialized methods on 12 of 18 prediction tasks, and up to 15 after fine-tuning. Despite no supervision, the transformer models learned to focus attention on key genomic elements, including those that regulate gene expression, such as enhancers. Lastly, we demonstrate that utilizing model representations can improve the prioritization of functional genetic variants. The training and application of foundational models in genomics explored in this study provide a widely applicable stepping stone to bridge the gap of accurate molecular phenotype prediction from DNA sequence. Code and weights available at: https://github.com/instadeepai/nucleotide-transformer.

## Introduction

Foundation models in artificial intelligence (AI) refer to large models incorporating millions of parameters trained on vast amounts of data, which can be adapted for an array of predictive purposes. These models have deeply transformed the AI field with notable examples in natural language processing (NLP) including the so-called language models (LMs) BERT [1] and GPT-3 [2]. LMs have gained considerable popularity over the past years, owing to their capacity to be trained on unlabeled data to create general-purpose representations that can solve downstream tasks. One way they achieve a general understanding of language is by solving billions of cloze tests in which, given a sentence with some blanked-out words, they are rewarded by suggesting the correct word to fill the gap. This approach is referred to as masked language modeling [1]. Early examples of foundation models leveraging this objective in biology were trained on protein sequences by tasking LMs to uncover masked amino acids in large protein sequence datasets [3, 4, 5]. Trained protein LMs applied to downstream tasks, in what is called transfer learning, showed an aptitude to compete and even surpass previous methods for protein structure [3, 4] and function predictions [6, 7], even in data scarce regiments [8].

Beyond protein sequences, the dependency patterns embedded in DNA sequences are fundamental to the understanding of genomic processes, from the characterization of regulatory regions to the impact of individual variants in their haplotypic context. Along this line of work, specialized deep learning

[*]**Equal Supervision**
[†]**Corresponding authors**: t.pierrot@instadeep.com & m.lopez@instadeep.com

(DL) models have been trained to identify meaningful patterns of DNA, for example, to predict gene expression from DNA sequences alone [9, 10, 11, 12, 13], with recent work combining convolutional neural networks (CNN) with transformer architectures enabling the encoding of regulatory elements as far as 100 kilobases (kb) upstream [14]. The deluge of data generated by modern genomics research poses both an opportunity and a challenge. On the one hand, intricate patterns of natural variability across species and populations are vastly available; on the other hand, powerful deep learning methods, that can operate at large scale, are required to perform accurate signal extraction from unlabelled data. Large foundation models trained on sequences of nucleotides appear to be a natural choice to tackle this problem [15, 16, 16, 17].

With the Nucleotide Transformer we add to recent attempts to encode genomics through foundation models. We built four distinct language models of different sizes, ranging from 500M up to 2.5B parameters, and pre-trained them on three different datasets including the human reference genome, a collection of 3,202 diverse human genomes, and 850 genomes from several species. Once trained, we leveraged representations (i.e. embeddings) of each of these models to further train downstream models on 18 genomic prediction tasks. We then explored the models' attention maps, perplexities, and performed data dimensionality reduction on embeddings to decipher the sequence features learned during pre-training. Finally, we assessed the embeddings' ability, through zero-shot-based scores, to model the impact of functionally important genetic variants in humans. As a result, we demonstrate the benefits of building and pre-training foundational language models in genomics, across different genomic datasets and parameter sizes, with the ability to surpass specialized methods.

# Results

## The Nucleotide Transformer models outperformed or matched 15 of 18 genomic baselines after fine-tuning

We developed a collection of transformer-based DNA language models, dubbed Nucleotide Transformer, that learned general nucleotide sequence representations from 6kb unannotated genomic data (Fig. 1a; Methods). Inspired by trends in NLP, where larger training datasets and model sizes have been shown to improve performance [18], the transformer models used the following parameter sizes and datasets: (i) a 500 million parameter model trained on sequences from the human reference genome (Human ref 500M), (ii) a 500 million and (iii) a 2.5 billion parameter model trained on 3,202 genetically diverse human genomes[19] (1000G 500M and 1000G 2.5B respectively), and (iv) a 2.5 billion parameter model based on 850 species across diverse phyla (Multispecies 2.5B) among which 11 are model organisms (Fig. 1a; Supplementary Tables 1, 2, 3, 4).

To assess the ability of these models to perform molecular phenotype predictions, we compiled 18 different genomic datasets, for which baseline performance metrics were available [20, 21, 22, 23, 24], and processed them into a common format that facilitated experimentation and reproducibility (Methods). The set of 18 genomic datasets represent a diverse and challenging prediction panel to assess the performance of the models (Fig. 1b; Supplementary Table 5). Based on these genomic tasks, we evaluated the transformer models after self-supervised training through two different techniques: probing (Fig. 1c) and fine-tuning (Fig. 1d). Probing refers to the use of learned LM embeddings of DNA sequences as input features to simpler models for predicting genomic labels. Specifically, we probed ten arbitrarily chosen layers of the LMs using either a logistic regression or a small multi-layer perceptron (MLP) composed of up to two hidden layers. In the case of fine-tuning, the LM head is replaced by either a classification or regression head, and a parameter-efficient technique is used for retraining (Methods). Then, to accurately compare the performance of the transformer models with the available baselines, we used a ten-fold validation strategy. The models were considered equivalent or better than the baseline if the resulting two standard deviations overlapped or were superior to the reported baseline value.

Using this criterion, we observed performances matching or surpassing 12 of the 18 baseline models through probing only (Supplementary Fig.1 and Table 6). In agreement with recent work [25], we observed that the best performance is both model- and layer-dependent (Supplementary Table 7). We also observed that the best model performance is never obtained using embeddings from the final

layer, as shown in earlier work [5]. For example, in the H3K4me1 histone occupancy classification task, we observe a relative difference between the highest and lowest performing layer as high as 38%, indicating that the learned representations vary significantly across the layers (Supplementary Fig. 2). In comparison to our probing strategy, our fine-tuned models matched or surpassed 15 of the 18 baselines, with the largest and more diverse models constantly outperforming their smaller counterparts (Fig. 2a and Supplementary Table 8 and 6). Our results also indicate that training on a diverse dataset, represented by the Multispecies 2.5B model, outperforms or matches the 1000G 2.5B model on several tasks derived from human-based assays (Fig. 2a and b). This suggests that a strategy of increased diversity instead of increased model size may yield improved prediction performance, particularly when computational resources are limited. While the prediction performance was generally higher for the fine-tuned models, we also observed considerable variation across tasks (Supplementary Table 6). The largest improvement was observed for splice site prediction, where the difference between the best probing and best-fine-tuned models was $\sim 20\%$. Fine-tuning has not been extensively studied in previous work [5], possibly due to its high compute requirements. We overcame the limitation by relying on a recent parameter-efficient fine-tuning technique [26] requiring only 0.1% of the total model parameters (Methods). This technique allowed for faster fine-tuning on a single GPU, reduced storage needs by 1,000-fold over all fine-tuning parameters and increased performance. In practice, we found that rigorous probing was slower and more compute-intensive than fine-tuning, even though using simple downstream models on embeddings might appear straightforward. This is due to the fact that the choice of the layer, downstream model, and hyperparameters greatly influenced the performance. Moreover, a smaller variance in performance was observed during fine-tuning, thus adding to the robustness of the approach.

Given that the highest performance was obtained by the fine-tuned Multispecies 2.5B model across the 18 tasks (Fig. 2a), we applied this model to two additional genomic predictions. These included quantitatively predicting developmental and housekeeping enhancer activities from *Drosophila melanogaster* S2 cells [27] and classifying 919 chromatin profiles from a diverse set of human cells and tissues [28]. Notably, the Multispecies 2.5B model obtained performances that closely matched those of specialized deep learning models. In the case of the chromatin feature profiles for transcription factor binding, DNAse I hypersensitive sites, and histone marks the transformer model obtained area under the curve (AUC) values that were, on average, only $\sim$1% lower than those obtained by DeepSEA [28] (Fig. 2c). For the housekeeping and developmental enhancer prediction, the transformer model slightly surpassed (1%) and obtained lower (4%) correlation values, respectively (Fig. 2d), compared to those estimated by DeepSTARR [27]. Overall, these results illustrate how fine-tuned transformer models can be developed, in a computation-efficient manner, to perform state-of-the-art prediction across a variety of genomic tasks.

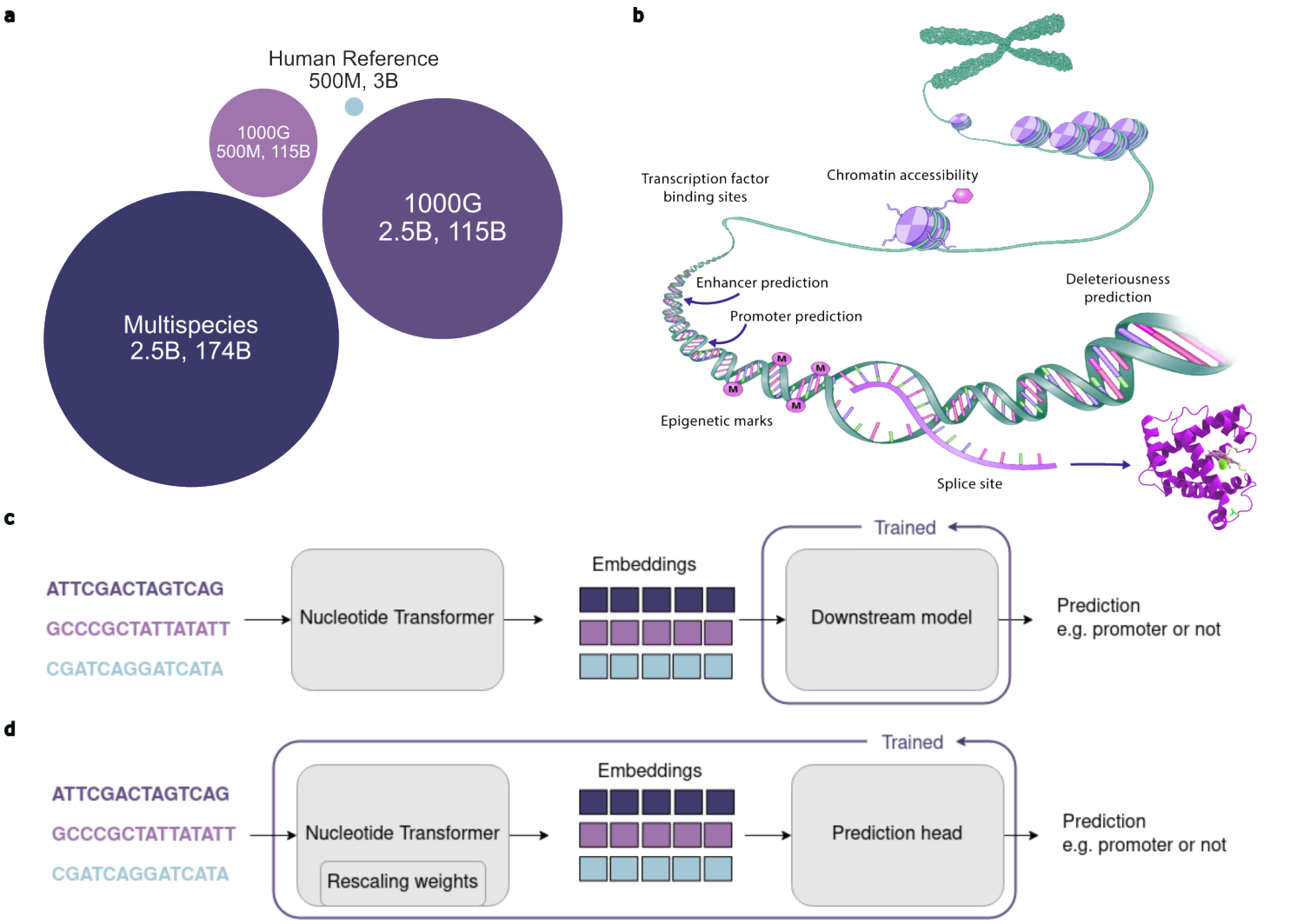



Figure 1: **The Nucleotide Transformer: a Masked Language Model trained for Genomics Prediction**. **a)** Training datasets and parameter sizes of the language models. **b)** Graphical representation of genomic features considered for prediction tasks. **c)** Overview of the Nucleotide Transformer training and application for downstream genomic prediction tasks through probing. **d)** Overview of the Nucleotide Transformer training and application for downstream genomic prediction tasks through fine-tuning.

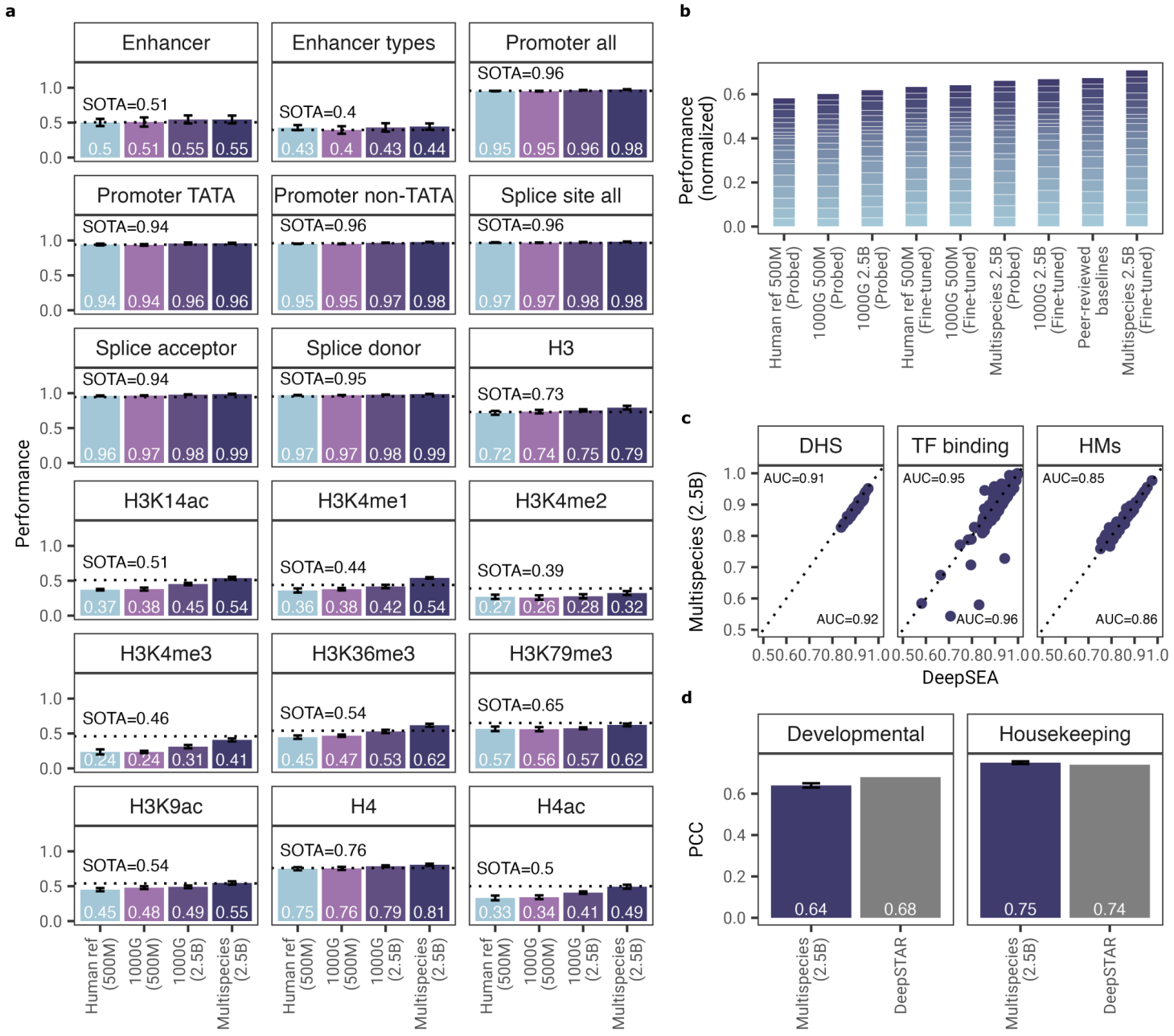



Figure 2: **The Nucleotide Transformer model matches or outperforms 15 out of 18 downstream tasks using fine-tuning. a)** Performance results across downstream tasks for fine-tuned transformer models. Error bars represent 2 SDs derived from 10-fold cross-validation. The performance metrics for the state-of-the-art (SOTA) models are shown as horizontal dotted lines. **b)** Normalized sum of performance across downstream tasks for all transformer models after probing and fine-tuning. **c)** The Multispecies 2.5B model performance on developmental and housekeeping enhancer activity predictions based on *Drosophila melanogaster* S2 cells. **d)** The Multispecies 2.5B model performance on DNase I hypersensitive sites (DHS), histone marks (HMs), and transcription factor sites predictions based on different human cells and tissues.

## The Nucleotide Transformer models learned to reconstruct human genetic variants and detect known genomic elements

To investigate the benefits of increasing the number of parameters of the models and genomic diversity of the datasets during the training, we assessed the ability of the models to reconstruct masked nucleotides. Specifically, we partitioned the human reference genome into non-overlapping 6kb sequences, tokenized the sequence into 6-mers, randomly masked a number of tokens, and estimated the proportion of tokens correctly reconstructed (Fig. 3a; Supplementary Fig. 3). We observed that the reconstruction accuracy of the Human reference 500M model showed higher median accuracy than the 1000G 500M model (median=0.202 versus 0.198, P<2.2e-16, two-sided Wilcoxon rank sum test). However, the accuracy of this model was lower than the accuracy obtained by the 1000G 2.5B model (median=0.216; P<2.2e-16, two-sided Wilcoxon rank sum test) and the 2.5B Multispecies model (median=0.219; P<2.2e-16, two-sided Wilcoxon rank sum test) (Supplementary Fig. 3), which illustrates the impact of jointly increasing the model size and diversity of the dataset. Interestingly, while the Multispecies 2.5B model showed the best reconstruction accuracy overall, a number of sequences showed much higher reconstruction accuracy for the 1000G 2.5B model (Fig. 3b), which likely points to particular characteristics of human sequences that were better learned by this model.

To better understand the representation of variant sites in DNA by the transformer language models, we then focused on polymorphisms found in human samples from diverse populations. Specifically, we measured the effectiveness of sequence reconstruction by recording the model perplexity scores over single nucleotide polymorphisms (SNPs) occurring at frequencies ranging from 1% to 100% (Methods). To also consider the impact of genomic background and genetic structure on the mutation reconstruction, we considered an independent dataset of genetically diverse human genomes, originating from 7 different meta-populations [29] (see Methods). Across SNPs frequencies, we consistently observe that the 2.5B parameter models, with median perplexity across populations ranging from 95.9±17.9 (2SD) to 145±9.3, outperform their 500M counterparts, for which the perplexity values fluctuate between 138.5±8.4 and 185.4±9.5 (Fig. 3c). The 1000G 2.5B model exhibited lower perplexity scores than the Multispecies 2.5B (median=121.5 versus 134.0, P<1.8e-12, two-sided Wilcoxon rank sum test) suggesting that this model is leveraging human genetic variability observed in the 1000G data to correctly reconstruct variants in unseen human genomes. These results are in line with the observed reconstruction accuracies over human genome sequences, and confirm the impact of model size on performance (Fig. 3b). Notably, and in contrast to typical genotype imputation methods where accuracy decreases as variant frequency decreases [30], the performance of transformer models was comparable across frequencies, indicating the potential for using these models to enhance genotype imputation even for rare variants.

To gain insights into the interpretability and the type of sequence elements that the nucleotide transformer is utilizing when making predictions, we explored different aspects of the transformer's model architecture. We first evaluated the degree to which the embeddings can capture sequence information related to different genomic elements. Specifically, we considered embeddings from sequences related to five different canonical elements of a gene's structure and visualized them with t-SNE (Supplementary Table. 9 and 5). Notably, we observe that the transformer models, without any supervision, learned to distinguish genomic sequences uniquely annotated as intergenic, intronic, coding, and UTR regions with varying degrees of ability to separate these across layers (Fig. 3d; Supplementary Fig. 5; Methods). In particular, the 500 million-sized models and those trained on less diverse sequences, showed lower separation among genomic regions, confirming again the the enhanced ability of the largest models to capture relevant genomic patterns during self-supervised training. In the case of the Multispecies 2.5B model, strongest separation at layer 1 is observed between intergenic and non-intergenic regions, followed by 5' UTR regions on layer 5, and a separation between most regions on layer 21 (Fig. 3d). The poor separation of 3'UTR regions from other elements, suggests that the model has not learned to properly distinguish this genomic element, or as previously suggested, that many of these regions might be misannotated [31]. Consistent with these observations, our probing strategy exhibited high classification performance of these elements, with accuracy values exceeding 0.78, especially for deeper layers.(Fig. 3e).

We next analyzed the transformer models through the lens of attention, as recently proposed by Vig

et al for protein sequences [32]. We computed the attention percentages across head and layers over sequences containing nine different genomic elements, including those related to gene structure and regulatory features. Formally, an attention head is thought to recognize an specific elements if its attention percentage is significantly greater than the naturally occurring frequency of the element in the pre-training dataset (see Methods). We found that the number of significant attention heads across layers varied markedly across models, with the highest number of significant attention heads observed for the Multispecies 2.5B model for CDS and introns (Supplementary Figs. 6, 7 and Table 10). A notable example includes the heads attending to enhancer sequences across models (Fig. 3f). For this regulatory element, the maximum attention percentages were highest for the largest models, reaching a value of almost 100% attention for the 1000G 2.5B model. For other genomic elements, similar patterns were also found, where the 1000G 2.5B model showed highly specialized heads showing high attention, particularly for the first layers (Supplementary Figs. 8 - 13). Altogether, these results illustrate how the transformer models have learned to recover gene structure and functional properties of genomic sequences and integrate them directly into its attention mechanism.

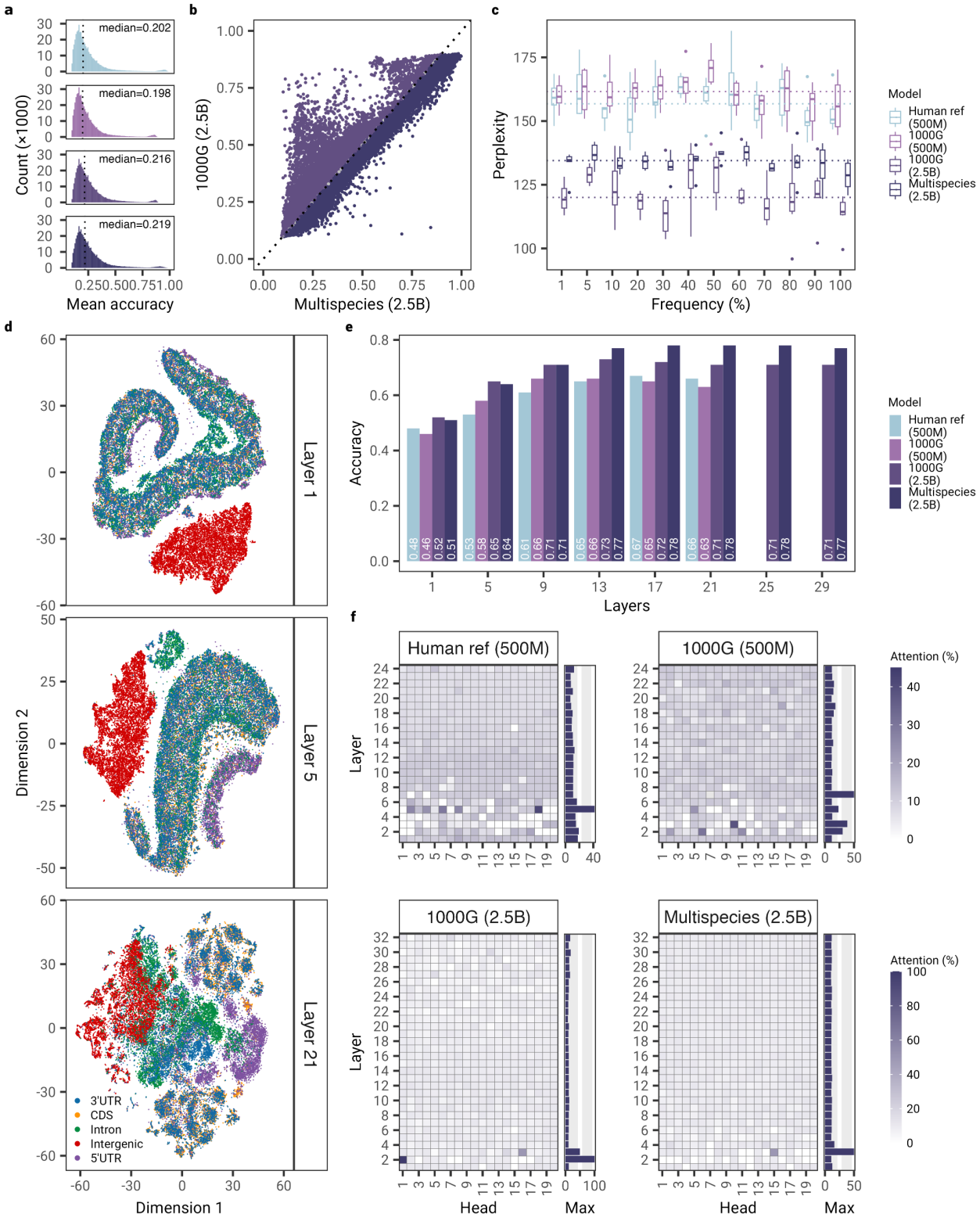



Figure 3: **The Nucleotide Transformer models acquired knowledge about genetic variations and genomic elements.** **a)** Mean reconstruction accuracy of 6kb sequences across transformer models. **b)** Comparison of reconstruction accuracies between the 1000G 2.5B and Multispecies 2.5B models. **c)** Reconstruction perplexity across allele frequencies. Perplexity values shown per frequency bin are based on 6 populations from the Human Genome Diversity Project. **d)** t-SNE projections of embeddings of 5 genomic elements from layer 1, 5, and 21 based on the Multispecies 2.5B model. **e)** Accuracy estimates based on probing to classify 5 genomic elements across layers. **f)** Attention percentages per head and layer across transformer models computed on enhancers. Barplot on the right of each tile plot shows the maximum attention percentage across all heads for a given layer.

## The Nucleotide Transformer embeddings predict the impact of mutations

Finally, we assessed the ability of the transformer models to determine the severity of different types of genetic variants and to prioritise those of functional importance. We first explored the utility of zero-shot scores (i.e. a score used to predict a class that was not seen by the model during training) by comparing their respective distribution across 10 types of genetic variants that varied in severity [33]. Specifically, we derived zero-shot scores using different aspects of the vector distances in the embeddings space as well as those based on the loss function (Fig. 4a and Methods). Encouragingly, some of these zero-shot scores showed a moderate correlation with severity across models (Supplementary Fig. 14), demonstrating how the unsupervised training alone is capturing relevant information that relates to the potential severity of genetic mutations, and the utility of testing different scores. The high variability of the correlation between scores, also suggests that distinct aspects of the embedding space may more effectively capture severity-related information. Among these scores, the cosine similarity showed the highest correlation with severity across models, with $r^2$ ranging from $-0.35$ to $-0.3$ (P-value $< 6.55e^{-186}$) (Supplementary Fig. 14). Across transformer models we show that the lowest cosine similarity scores were assigned to genetic variants that impact protein function, such as stop gained variants, as well as synonymous and missense variants (Fig. 4b). Conversely, we observe that higher scores were assigned to potentially less functionally important variants, such as intergenic variants.

To prioritize functional variants, especially those of high pathogenicity, we delve deeper into the potential of zero-shot scores. Specifically, we assessed the ability of the models to classify genetic variants exerting regulatory effects on gene expression (i.e., expression quantitative trait loci [eQTLs]), genetic variants associated with DNA methylation variation (i.e., methylation quantitative trait loci [meQTLs]), genetic variants annotated as pathogenic in the ClinVar database, and genetic variants reported in the Human Gene Mutation Database (HGMD). Notably, the zero-shot scores obtained high classification performance, with highest AUCs across the four tasks ranging from 0.7 to 0.8 (Fig. 4c). The highest performance obtained for the Clinvar variants (AUC=0.80 for the Multispecies 2.5B model), suggests that, at least for highly pathogenic variants, zero-shot scores might be readily applicable. To formally evaluate the effectiveness of transformer models, we also made predictions based on fine-tuned models, and compared their performance against several methods. These encompassed methods that measure levels of genomic conservation, as well as scores obtained from models trained on functional features. Notably, the transformer models either slightly outperformed or closely matched the performance of the other models (Fig. 4d). The best performing models for prioritizing molecular phenotypes (i.e. eQTLs and meQTLs) were those trained on human sequences, whereas the best performing model for prioritizing pathogenic variants was based on multispecies sequences. Given that the most severely pathogenic variants tend to impact gene function due to amino acid changes, it is possible that the multispecies model is leveraging sequence variation across species to learn about the degree of conservation across sites. Our results also suggest that higher predictive power for non-coding variants such as eQTLs and meQTLs might be obtained by better-learned sequence variation derived from increasing human genetic variability. Further, when compared to zero-shot scores, the dot product showed and AUC of 0.73 and 0.71 for eQTLs and meQTLs, which were slightly higher than and equal to those obtained by the fine-tuned models, respectively. Considering that most of these genetic variants tend to lie within regulatory regions [34, 35, 36], it is probable that the transformer models, without any supervision, have learned to distinguish relevant regulatory genomic features related to gene expression and methylation variation. This is in accordance with the level of attention observed across layers and heads, especially for relevant regulatory sequences such as enhancers (Fig. 3d) and promoters (Supplementary Fig. 11), which have been shown to be enriched in meQTLs and eQTLs [34, 35, 36]. Overall, these results illustrate how DNA-based transformer models can help reveal and contribute to understanding the potential biological consequences of variants associated with molecular phenotypes and disease.

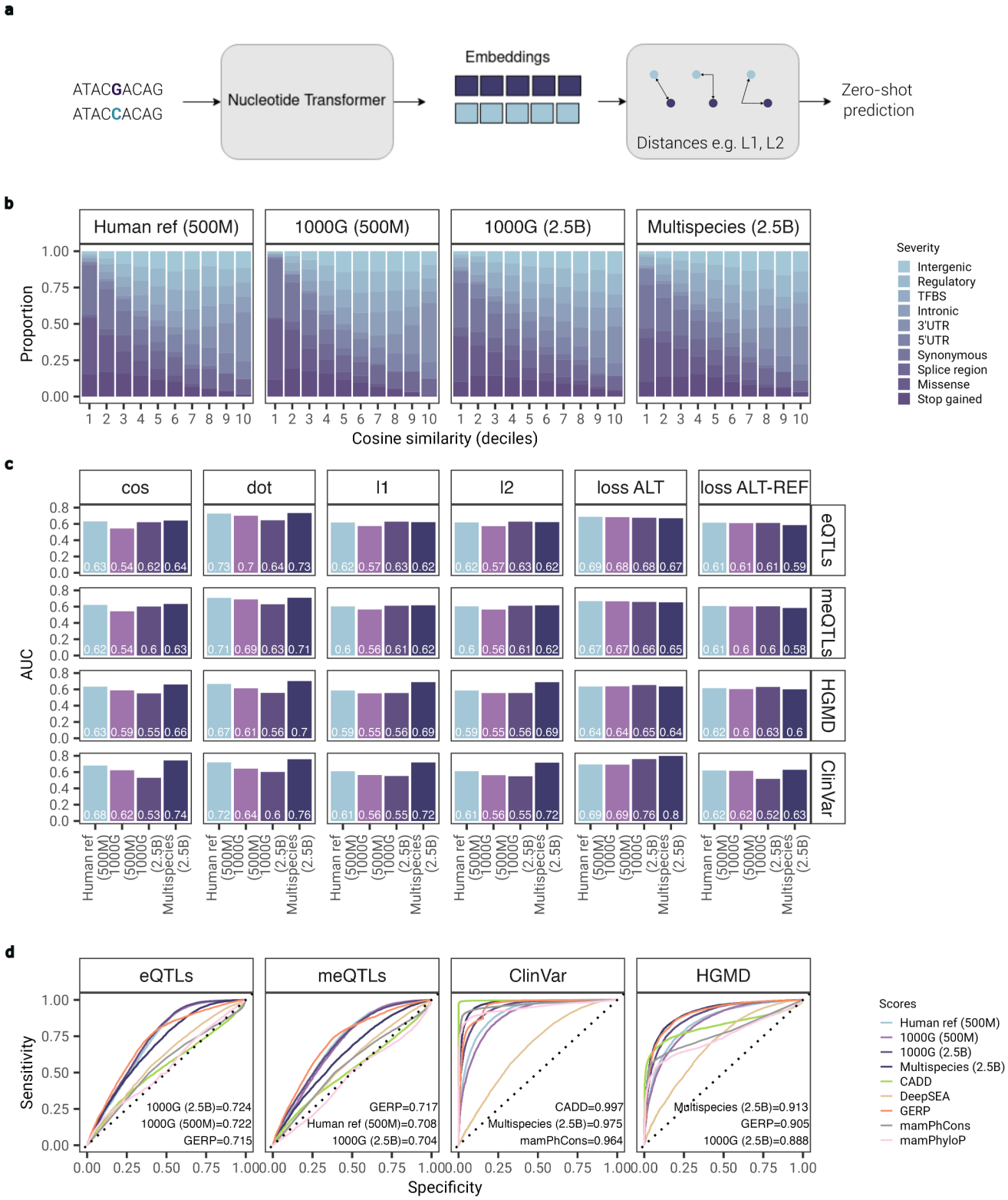



Figure 4: **Prioritizing functional genetic variants. a)** Overview of the Nucleotide Transformer application of zero-shot predictions. **b)** Proportion of variant consequence terms across deciles based on the cosine similarity metric across models. The consequence terms are shown in order of severity (less severe to more severe) as estimated by Ensembl. **c)** Comparison of zero-shot predictions for prioritizing functional variants based on different distance metrics. **d)** Comparison of fine-tuned models and available methods for prioritizing functional variants based on GRASP eQTLs and meQTLs, ClinVar, and HGMD annotated mutations. Model performance is measured with the area under the receiver operating characteristic curve (AUC). The AUC for the three best-performing models is shown.

## Discussion

To our knowledge, this study represents the first attempt to investigate the impact of different datasets to pre-train equally-sized transformer models on DNA sequences. Our results, based on distinct genomic prediction tasks, demonstrate that both intra- (i.e. when training on multiple genomes of a single species) and inter-species (i.e., on genomes across different species) variability play an important factor in driving accuracy across tasks (Fig. 2). The models trained on genomes from different species perform better on many human prediction tasks, even when compared to those trained exclusively on human sequences. This may indicate that transformer models based on different species learned to capture genomic features that are likely of functional importance across species and therefore generalise better in a variety of human-based prediction tasks. Based on this finding, we predict that future studies might benefit from leveraging genetic variability across species.

The transformer models trained in this study ranged from 500M up to 2.5B parameters, which is five times larger than DNABert [15] and ten times larger than the Enformer [14] models. As previously shown in NLP research [18], our results on genomic prediction tasks confirmed that increasing model size yields better performance. In order to train the models with the largest parameter sizes, we required a total of 128 GPUs across 16 compute nodes for 28 days. Significant investments were made to engineer efficient training routines that took full advantage of the infrastructure, highlighting the need for both specialized infrastructure and dedicated software solutions. Once trained, however, these models can be used for inference at a relatively low cost.

Previous work based on language models trained on biological data (mainly protein sequences) evaluated downstream performance exclusively probing the last transformer layer [5], most likely due to its perceived ease of use, relatively good performance, and low computational complexity. In this study, we aimed to maximise downstream accuracy through computationally expensive, rigorous probing of different transformer layers, downstream models, and hyperparameter sweeps. We observed best-probing performance for intermediate transformer layers (Supplementary Fig. 1), aligning with recent work in computational biology [25] and common practice in NLP [37]. Through probing only, the Multispecies 2.5B model achieved better performance for 8 out of 18 tasks compared to the baseline. In addition, we explored a recent downstream fine-tuning technique that introduces a small number of trainable weights in the transformer, providing a relatively fast and weight-efficient fine-tuning procedure ($IA^3$ [26]). Compared to the extensive probing exercise, this technique yielded better results using fewer compute resources, confirming that downstream model engineering can lead to performance improvements [38]. The usage of this technique makes fine-tuning competitive with probing from an operational perspective, both for training and inference.

Through different analyses related to the transformer architecture, we showed that the models learned to recognize key regulatory genomic elements, as demonstrated through the analysis of attention maps, embedding spaces, and probability distributions. Important regulatory elements that control gene expression, such as enhancers and promoters [34, 35, 36], were detected by all models in several heads and layers. We also observed that each model contained at least one layer that produced embeddings that clearly separated five of the genomic elements analyzed. As self-supervised training allowed for the detection of these elements, we expect that this approach can be leveraged to refine or detect novel genomic elements in the future.

We demonstrated that transformer models can match and outperform other variant effect and deleteriousness prediction methods. In addition to developing supervised transformer models, we also showed the utility of zero-shot based scores, particularly for non-coding effect prediction. Since these zero-shot based scores can be obtained solely from genomic sequences, we encourage similar applications for non-human organisms, particularly those with poor functional annotation.

While our models have an attention span limited to 6kb, the recently developed Enformer model [14] suggested that increasing the perception field up to 200kb is necessary to capture long-range dependencies in the human genome. The authors argued that these are needed to accurately predict gene expression, which is controlled by distal regulatory elements, usually located far greater than 10-20kb away from the transcription starting site. Processing such large inputs is intractable with the standard

transformer architecture, due to the quadratic scaling of self-attention with respect to the sequence length. The Enformer model addressed this by passing sequences through convolution layers to reduce the dimensions of the inputs to transformer layers. However, this choice hindered language modeling. Based on our results, we suggest that building transformer models with the ability to model language that can work with long inputs, up to 200kbp or more, is a promising direction for the field. Techniques such as sparse attention [39, 40] could be investigated to overcome computational complexity limitations. At a moment where multi-omics data are growing quickly, we ultimately expect that the methodology presented here will lead to a boost in the adoption, development, and improvement of large language foundation models in genomics.

## Data availability

The Nucleotide Transformer pre-training sequences were obtained from publicly available resources. The 1000 Genomes Project sequences were obtained from $http://ftp.1000genomes.ebi.ac.uk/vol1/ftp/data_collections/1000G_2504_high_coverage/$, and the human and multispecies reference genomes from $https://ftp.ncbi.nlm.nih.gov/genomes/refseq/$. Gene annotations were obtained from GENCODE ($https://www.gencodegenes.org/$) and Ensembl databases ($https://www.ensembl.org$). Variants effects predictions were obtained using the Variant Effect Prediction (VEP) API from Ensembl ($https://www.ensembl.org/info/docs/tools/vep/index.html$). Pathogenic and regulatory variants were extracted from ClinVar ($https://ftp.ncbi.nlm.nih.gov/pub/clinvar/vcf_GRCh38/$), the Genome-Wide Repository of Associations Between SNPs and Phenotypes (GRASP) ($https://grasp.nhlbi.nih.gov$), and The Human Gene Mutation Database (HGMD) (public version 2020.4) through Ensembl Biomart. The training and test datasets used in the downstream tasks were acquired from repositories that were referenced in the corresponding publications (see Supplementary Table 5).

## Code availability

Model weights of the four pre-trained transformer models as well as inference code are available for research purposes at: https://github.com/instadeepai/nucleotide-transformer.

## Acknowledgements

We thank members of the Rostlab, particularly Tobias Olenyi, Ivan Koludarov, and Burkhard Rost for constructive discussions that helped identify interesting research directions. We also thank Maša Roller for helpful discussion and commenting on the manuscript. Furthermore, we would like to express our gratitude to all volunteers who generously provided their biological data, as well as to the researchers who deposited experimental data in public databases, the researchers who maintain these databases, and those who make analytical and predictive methods available to the scientific community.

# References


[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[2] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[3] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, *et al.*, "Highly accurate protein structure prediction with alphafold," *Nature*, vol. 596, no. 7873, pp. 583–589, 2021.

[4] A. Rives, J. Meier, T. Sercu, S. Goyal, Z. Lin, J. Liu, D. Guo, M. Ott, C. L. Zitnick, J. Ma, *et al.*, "Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences," *Proceedings of the National Academy of Sciences*, vol. 118, no. 15, p. e2016239118, 2021.

[5] A. Elnaggar, M. Heinzinger, C. Dallago, G. Rihawi, Y. Wang, L. Jones, T. Gibbs, T. Feher, C. Angerer, M. Steinegger, *et al.*, "Prottrans: towards cracking the language of life's code through self-supervised deep learning and high performance computing," *arXiv preprint arXiv:2007.06225*, 2020.

[6] M. Littmann, M. Heinzinger, C. Dallago, T. Olenyi, and B. Rost, "Embeddings from deep learning transfer go annotations beyond homology," *Scientific reports*, vol. 11, no. 1, pp. 1–14, 2021.

[7] C. Marquet, M. Heinzinger, T. Olenyi, C. Dallago, K. Erckert, M. Bernhofer, D. Nechaev, and B. Rost, "Embeddings from protein language models predict conservation and variant effects," *Human genetics*, vol. 141, no. 10, pp. 1629–1647, 2022.

[8] M. Littmann, M. Heinzinger, C. Dallago, K. Weissenow, and B. Rost, "Protein embeddings and deep learning predict binding residues for various ligand classes," *Scientific Reports*, vol. 11, Dec. 2021.

[9] G. Eraslan, Ž. Avsec, J. Gagneur, and F. J. Theis, "Deep learning: new computational modelling techniques for genomics," *Nature Reviews Genetics*, vol. 20, no. 7, pp. 389–403, 2019.

[10] J. Zhou, C. L. Theesfeld, K. Yao, K. M. Chen, A. K. Wong, and O. G. Troyanskaya, "Deep learning sequence-based ab initio prediction of variant effects on expression and disease risk," *Nature Genetics*, vol. 50, pp. 1171–1179, July 2018.

[11] D. R. Kelley, "Cross-species regulatory sequence activity prediction," *PLOS Computational Biology*, vol. 16, p. e1008050, July 2020.

[12] D. R. Kelley, Y. A. Reshef, M. Bileschi, D. Belanger, C. Y. McLean, and J. Snoek, "Sequential regulatory activity prediction across chromosomes with convolutional neural networks," *Genome Research*, vol. 28, pp. 739–750, Mar. 2018.

[13] V. Agarwal and J. Shendure, "Predicting mRNA abundance directly from genomic sequence using deep convolutional neural networks," *Cell Reports*, vol. 31, p. 107663, May 2020.

[14] Ž. Avsec, V. Agarwal, D. Visentin, J. R. Ledsam, A. Grabska-Barwinska, K. R. Taylor, Y. Assael, J. Jumper, P. Kohli, and D. R. Kelley, "Effective gene expression prediction from sequence by integrating long-range interactions," *Nature methods*, vol. 18, no. 10, pp. 1196–1203, 2021.

[15] Y. Ji, Z. Zhou, H. Liu, and R. V. Davuluri, "Dnabert: pre-trained bidirectional encoder representations from transformers model for dna-language in genome," *Bioinformatics*, vol. 37, no. 15, pp. 2112–2120, 2021.

[16] M. T. Zvyagin, A. Brace, K. Hippe, Y. Deng, B. Zhang, C. O. Bohorquez, A. Clyde, B. Kale, D. Perez-Rivera, H. Ma, *et al.*, "Genslms: Genome-scale language models reveal sars-cov-2 evolutionary dynamics.," *bioRxiv*, 2022.

[17] C. Outeiral and C. M. Deane, "Codon language embeddings provide strong signals for protein engineering," *bioRxiv*, 2022.

[18] J. W. Rae, S. Borgeaud, T. Cai, K. Millican, J. Hoffmann, F. Song, J. Aslanides, S. Henderson, R. Ring, S. Young, *et al.*, "Scaling language models: Methods, analysis & insights from training gopher," *arXiv preprint arXiv:2112.11446*, 2021.

[19] . G. P. Consortium *et al.*, "A global reference for human genetic variation," *Nature*, vol. 526, no. 7571, p. 68, 2015.

[20] T. H. Phaml, D. H. Tran, T. B. Ho, K. Satou, and G. Valiente, "Qualitatively predicting acetylation and methylation areas in dna sequences," *Genome Informatics*, vol. 16, no. 2, pp. 3–11, 2005.

[21] Q. Geng, R. Yang, and L. Zhang, "A deep learning framework for enhancer prediction using word embedding and sequence generation," *Biophysical Chemistry*, vol. 286, p. 106822, 2022.

[22] R. Wang, Z. Wang, J. Wang, and S. Li, "Splicefinder: ab initio prediction of splice sites using convolutional neural network," *BMC bioinformatics*, vol. 20, no. 23, pp. 1–13, 2019.

[23] M. Oubounyt, Z. Louadi, H. Tayara, and K. T. Chong, "Deepromoter: robust promoter predictor using deep learning," *Frontiers in genetics*, vol. 10, p. 286, 2019.

[24] N. Scalzitti, A. Kress, R. Orhand, T. Weber, L. Moulinier, A. Jeannin-Girardon, P. Collet, O. Poch, and J. D. Thompson, "Spliceator: Multi-species splice site prediction using convolutional neural networks," *BMC bioinformatics*, vol. 22, no. 1, pp. 1–26, 2021.

[25] F.-Z. Li, A. P. Amini, K. K. Yang, and A. X. Lu, "Pretrained protein language model transfer learning: is the final layer representation what we want?,"

[26] H. Liu, D. Tam, M. Muqeeth, J. Mohta, T. Huang, M. Bansal, and C. Raffel, "Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning," *arXiv preprint arXiv:2205.05638*, 2022.

[27] B. P. de Almeida, F. Reiter, M. Pagani, and A. Stark, "Deepstarr predicts enhancer activity from dna sequence and enables the de novo design of synthetic enhancers," *Nature Genetics*, vol. 54, no. 5, pp. 613–624, 2022.

[28] J. Zhou and O. G. Troyanskaya, "Predicting effects of noncoding variants with deep learning–based sequence model," *Nature methods*, vol. 12, no. 10, pp. 931–934, 2015.

[29] A. Bergström, S. A. McCarthy, R. Hui, M. A. Almarri, Q. Ayub, P. Danecek, Y. Chen, S. Felkel, P. Hallast, J. Kamm, H. Blanché, J.-F. Deleuze, H. Cann, S. Mallick, D. Reich, M. S. Sandhu, P. Skoglund, A. Scally, Y. Xue, R. Durbin, and C. Tyler-Smith, "Insights into human genetic variation and population history from 929 diverse genomes," *Science*, vol. 367, Mar. 2020.

[30] S. Rubinacci, O. Delaneau, and J. Marchini, "Genotype imputation using the positional burrows wheeler transform," *PLoS genetics*, vol. 16, no. 11, p. e1009049, 2020.

[31] G. Benegas, S. S. Batra, and Y. S. Song, "Dna language models are powerful zero-shot predictors of non-coding variant effects," *bioRxiv*, pp. 2022–08, 2022.

[32] J. Vig, A. Madani, L. R. Varshney, C. Xiong, R. Socher, and N. F. Rajani, "Bertology meets biology: interpreting attention in protein language models," *arXiv preprint arXiv:2006.15222*, 2020.

[33] W. McLaren, L. Gil, S. E. Hunt, H. S. Riat, G. R. Ritchie, A. Thormann, P. Flicek, and F. Cunningham, "The ensembl variant effect predictor," *Genome biology*, vol. 17, no. 1, pp. 1–14, 2016.

[34] T. Lappalainen, M. Sammeth, M. R. Friedländer, P. A. 't Hoen, J. Monlong, M. A. Rivas, M. Gonzalez-Porta, N. Kurbatova, T. Griebel, P. G. Ferreira, *et al.*, "Transcriptome and genome sequencing uncovers functional variation in humans," *Nature*, vol. 501, no. 7468, pp. 506–511, 2013.

[35] G. Consortium, "The gtex consortium atlas of genetic regulatory effects across human tissues," *Science*, vol. 369, no. 6509, pp. 1318–1330, 2020.

[36] U. Võsa, A. Claringbould, H.-J. Westra, M. J. Bonder, P. Deelen, B. Zeng, H. Kirsten, A. Saha, R. Kreuzhuber, S. Yazar, *et al.*, "Large-scale cis-and trans-eqtl analyses identify thousands of genetic loci and polygenic scores that regulate blood gene expression," *Nature genetics*, vol. 53, no. 9, pp. 1300–1310, 2021.

[37] A. Rogers, O. Kovaleva, and A. Rumshisky, "A primer in bertology: What we know about how bert works," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 842–866, 2020.

[38] H. Stärk, C. Dallago, M. Heinzinger, and B. Rost, "Light attention predicts protein location from the language of life," *Bioinformatics Advances*, vol. 1, no. 1, p. vbab035, 2021.

[39] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The long-document transformer," *arXiv preprint arXiv:2004.05150*, 2020.

[40] M. Zaheer, G. Guruganesh, K. A. Dubey, J. Ainslie, C. Alberti, S. Ontanon, P. Pham, A. Ravula, Q. Wang, L. Yang, *et al.*, "Big bird: Transformers for longer sequences," *Advances in Neural Information Processing Systems*, vol. 33, pp. 17283–17297, 2020.

[41] A. Wang, Y. Pruksachatkun, N. Nangia, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "Superglue: A stickier benchmark for general-purpose language understanding systems," 2019.

[42] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," *arXiv preprint arXiv:1606.08415*, 2016.

[43] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[44] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

[45] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," *Advances in neural information processing systems*, vol. 24, 2011.

[46] M. Byrska-Bishop, U. S. Evani, X. Zhao, A. O. Basile, H. J. Abel, A. A. Regier, A. Corvelo, W. E. Clarke, R. Musunuri, K. Nagulapalli, *et al.*, "High-coverage whole-genome sequencing of the expanded 1000 genomes project cohort including 602 trios," *Cell*, vol. 185, no. 18, pp. 3426–3440, 2022.

[47] P. Dmitry K, H. Christopher T, L. Stuart, C. Megan, M. H. Nancy, L. Tong Ihn, B. George W, W. Kimberly, R. P Alex, H. Elizabeth, Z. Julia, L. Fran, G. David K, and Y. Richard A, "Genome-wide map of nucleosome acetylation and methylation in yeast," *Cell*, vol. 122, pp. 517–27, 2005.

[48] R. Leslie, C. J. O'Donnell, and A. D. Johnson, "Grasp: analysis of genotype–phenotype results from 1390 genome-wide association studies and corresponding open access database," *Bioinformatics*, vol. 30, no. 12, pp. i185–i194, 2014.

[49] M. J. Landrum, J. M. Lee, M. Benson, G. R. Brown, C. Chao, S. Chitipiralla, B. Gu, J. Hart, D. Hoffman, W. Jang, *et al.*, "Clinvar: improving access to variant interpretations and supporting evidence," *Nucleic acids research*, vol. 46, no. D1, pp. D1062–D1067, 2018.

[50] P. D. Stenson, M. Mort, E. V. Ball, M. Chapman, K. Evans, L. Azevedo, M. Hayden, S. Heywood, D. S. Millar, A. D. Phillips, *et al.*, "The human gene mutation database (hgmd®): optimizing its use in a clinical diagnostic or research setting," *Human genetics*, vol. 139, pp. 1197–1207, 2020.

# A Methods

## A.1 Models

Language models (LMs) have been primarily developed within Natural Language Processing (NLP) to model spoken languages [1, 2]. A LM is a probability distribution over sequences of tokens (often words), i.e. given any sequence of words, an LM will return the probability for that sentence to exist. LMs gained in popularity thanks to their ability to leverage large unlabeled datasets to generate general-purpose representations that can solve downstream tasks even when little supervised data is available [41]. One technique to train LMs tasks models to predict the most likely tokens at masked positions in a sequence, often referred to as masked language modelling (MLM). Motivated by results obtained with MLM in the field of protein research [4, 5], where proteins are considered as sentences and amino-acids as words, we apply MLM to train language models transformers in genomics, considering sequences of nucleotides as sentences and k-mers (with k=6) as words. Transformers are a class of deep learning models that achieved breakthroughs in machine learning fields including NLP and computer vision. They consist of an initial embedding layer that transform positions in the input sequence into an embedding vector, followed by stack of self-attention layers that sequentially refine these embedding. The main technique to train language models transformers with MLM is called Bidirectional Encoder Representations from Transformers (BERT) [1]. In BERT, all positions in the sequence can attend to each other allowing the information to flow in both directions, which is essential in the context of DNA sequences. During training, the final embedding of the network is fed to a language model head that transforms it into a probability distribution over the input sequence.

### A.1.1 Architecture

All our models follow an encoder-only transformer architecture. An embedding layer transforms sequences of tokens into sequences of embeddings. Positional encodings are then added to each embedding in the sequence to provide the model with positional information. We use a learnable positional encoding layer that accepts a maximum of 1000 tokens. The embeddings are then processed by a transformer layer stack. Each transformer layer transforms its input through a layer normalisation layer followed by a multi-head self-attention layer. The output of the self-attention layer is summed with the transformer layer input through a skip connection. The result of this operation is then passed through a new layer normalisation layer and a two layer perceptron with GELU activations [42]. The number of heads, the embedding dimension, the number of neurons within the perceptron hidden layer and the total number of layers for each model can be found in Table 1. During self-supervised training, the embeddings returned by the final layer of the stack are transformed by a language model head into a probability distribution over the existing tokens at each position in the sequence.

### A.1.2 Training

The models are trained following the BERT methodology [1]. At each training step a batch of tokenized sequences is sampled. The batch size is adapted to available hardware and model size. We conducted all experiments on clusters of A100 GPUs, and took batches of sizes 14 and 2 sequences to train the 500M and 2.5B parameters models, respectively. Within a sequence, of a subset of 15% of tokens, 80% are replaced by a special mask [MASK] token. For training runs on the Human reference genome and multispecies datasets, an additional 10% of the 15% subset of tokens are replaced by randomly selected standard tokens (i.e. any token different from the class [CLS], pad [PAD] or mask [MASK] token), as was done in BERT. For training runs on the 1000G dataset, we skipped this additional data augmentation, as the added noise was greater than the natural mutation frequency present in the human genome. For each batch, the loss function was computed as the sum of the cross-entropy losses, between the predicted probabilities over tokens and the ground truth tokens, at each selected position. Gradients were accumulated to reach an effective batch size of 1M tokens per batch. We used the Adam optimizer [43] with a learning rate schedule, and standard values for exponential decay rates and epsilon constants, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon$=1e-8. During a first warmup period, the learning rate was increased linearly between 5e-5 and 1e-4 over 16k steps before decreasing following a square root decay until the end of training.

### A.1.3 Probing

We refer to probing the assessment of the quality of the model embeddings to solve downstream tasks. After training, for each task, we probe each layer of the model and compare several downstream methods to evaluate in depth the representations capabilities of the model. In other words, given a dataset of nucleotide sequences for a downstream task, we compute and store the embeddings returned by ten layers of the model. Then, using the embeddings of each individual layer as inputs, we trained several downstream models to solve the downstream task. We tested logistic regression with the default hyperparameters from scikit-learn [44] and a multi-layer peceptron. As we observed that the choice of hyperparameters, such as the learning rate, the activation function and the number of layers per hidden layer impacted final performance, we also ran hyperparameters sweeps for each downstream model. We used a ten fold validation scheme, where the training dataset was split ten times in a training and validation set, that contain different shuffles with 90% and 10% of the initial set. For a given set of hyperparameters, ten models were trained over the ten splits, and their validation performances were averaged. This procedure is run 100 times with a Tree-structured Parzen Estimator solver [45] guiding the search over the hyperparameters space, before evaluating the best performing set of models on the test set. Therefore, for each downstream task, for ten layers of each pre-trained model, the performance on the test set is recorded at the end of the hyperparameters search. The hyperparameters of the best performing probe across the pre-trained models and their layers are reported in Table 7. This probing strategy resulted in 760,000 downstream models trained, which provides detailed analysis into various aspects of training and using LMs, such as the role of different layers on downstream task performance.

### A.1.4 Fine-tuning

In addition to probing our models through embedding extraction at various layers, we also performed parameter-efficient fine-tuning through the IA$^3$ technique [26]. Using this strategy, the language model head is replaced by either a classification or regression head depending on the task at hand. The weights of the transformer layers and embedding layers are frozen and new, learnable weights are introduced. For each transformer layer, we introduced three learned vectors $l_k \in \mathbb{R}^{d_k}$, $l_v \in \mathbb{R}^{d_v}$ and $l_{\text{ff}} \in \mathbb{R}^{d_{\text{ff}}}$, which were introduced in the self-attention mechanism as:

$$\text{softmax}\left(\frac{Q(l_k) \odot K^T}{\sqrt{d_k}}\right)(l_v \odot V)$$

and in the position-wise feed-forward networks as $(l_{\text{ff}}\gamma\,(W_1 x))\,W_2$, where $\gamma$ is the feed-forward network nonlinearity, and $\odot$ represents the element-wise multiplication. This adds a total of $L\,(d_k + d_v + d_{\text{ff}})$ new parameters, where $L$ is the number of transformer layers. We refer to these learnable weights as *rescaling* weights. The intuition is that during fine-tuning these weights will weigh the transformer layers to improve the final representation of the model on a downstream task, so that the classification/regression head can more accurately solve the problem. As we observed layer specialization during probing, we speculate that this fine-tuning technique will similarly select layers with greater predictive ability for particular tasks.

In practice, the number of additional parameters introduced by rescaling weights and the classification/regression head weights represented approximately 0.1% of the total number of weights of the model. This increased fine-tuning speed since just a fraction of parameters needed updating. Similarly, it alleviated storage requirements, needing to create space for just 0.1% new parameters over 500M and 2.5B for each downstream task using traditional fine-tuning. For instance, for the 2.5B parameters models, the weights represent 9.5GB. Considering 18 downstream tasks, classical fine-tuning would have required $9.5 \times 18 = 171$ GBs, whereas parameter-efficient fine tuning required only 171 MB.

Like in the probing scheme, the training dataset was split ten times in a training and validation set, that contain different shuffles with 90% and 10% of the initial set. For each, split, the model was fine-tuned for 10k steps and parameters yielding the highest validation score were then used to evaluate the model on the test set. We used a batch size of eight and the Adam optimizer with a learning rate of 3e-3. Other optimizer parameters were maintained from training regiments. Each model is fine-tuned for 10k steps for each task. These hyperparameters were selected as they led to promising

results in the field of NLP [26]. Diverging hyperparameter choices did not yield significant gains in our experiments.

## A.2 Datasets

### A.2.1 The Human reference genome dataset

The Human reference dataset was constructed by considering all autosomal and sex chromosomes sequences from reference assembly GRCh38/hg38 [1] and reached a total of 3.2 billion nucleotides.

### A.2.2 The 1000G dataset

To inform the model on naturally occurring genetic diversity in humans, we constructed a training dataset including genetic variants arising from different human populations. Specifically, we downloaded the variant calling format (VCF) files[2] from the 1000 Genomes project [46], which aims at recording genetic variants occurring at a frequency of at least 1% in the human population. The dataset contained 3202 high-coverage human genomes, originating from 27 geographically structured populations of African, American, East Asian, and European ancestry as detailed in Table 2, making up a total of 20.5 trillion nucleotides. Such diversity allowed the dataset to encode a better representation of human genetic variation. To allow haplotype reconstruction in the FASTA format from the VCF files, we considered the phased version of the data, which corresponded to a total of 125M mutations, 111M and 14M of which are single nucleotide polymorphisms (SNPs) and indels, respectively.

### A.2.3 The Multispecies dataset

To build a dataset that encompassed a large and diverse set of genomes, we first parsed the genomes available on NCBI[3], before arbitrarily selecting only one species from each genus. Plant and virus genomes were not taken into account, as their regulatory elements differ from those of interest in this work. The resulting collection of genomes was downsampled to a total of 850 species, whose genomes add up to 174 billion nucleotides. The final contribution of each class, in terms of number of nucleotides, to the total number of nucleotides in the dataset, displayed in Table 3, is the same as in the original collection parsed from NCBI. Finally, we enriched this dataset by selecting several genomes that have been heavily studied in the literature (Table 4).

## A.3 Data preparation

Once the FASTA files of each genome / individual were collected, they were assembled into one unique FASTA file per dataset that was then pre-processed before training. During this data processing phase, all nucleotides other than A,T,C,G were replaced by N. A tokenizer was employed to convert strings of letters to sequences of tokens. The tokenizer used as alphabet the $4^6 = 4096$ possible 6-mer combinations obtained by combining A,T,C,G, as well as five extra tokens to represent stand-alone A,T,C,G and N. It also included three special tokens, namely the padding [pad], masking [mask] and the beginning of sequence (also called class; [CLS]) token. This adds to a vocabulary of 4104 tokens. To tokenize an input sequence, the tokenizer will start with a class token and then convert the sequence starting from the left, matching 6-mer tokens when possible, or falling back on the stand-alone tokens when needed (for instance when the letter N is present or if the sequence length is not a multiple of 6).

For the multispecies and Human reference dataset, genomes are split into overlapping chunks of 6100 nucleotides, each sharing the first and last 50 nucleotides with the previous and last chunk, respectively. As a data augmentation exercise, for each epoch and chunk, a starting nucleotide index is randomly sampled between 0 and 100, and the sequence is then tokenized from this nucleotide until 1000 tokens is reached. The number of epochs was determined depending on the dataset so that the model processed a total of 300B tokens during training. At each step, a batch of sequences sampled randomly within

[1] https://www.ncbi.nlm.nih.gov/assembly/GCF_000001405.26

[2] http://ftp.1000genomes.ebi.ac.uk/vol1/ftp/data_collections/1000G_2504_high_coverage/working/20201028_3202_phased/

[3] https://www.ncbi.nlm.nih.gov/

the epoch set was fed to the model. For the 1000G dataset, batches of sequences from the Human reference genome, prepared as specified above, are sampled at each step. Then, for each sampled chunk, an individual from the 1000G dataset is randomly selected, and if that individual carries mutations at the positions and chromosome corresponding to that chunk, these mutations are introduced into the sequence, and the corresponding tokens replaced. This data processing technique ensured uniform sampling both over the genome and over the individuals during training, as well as enabled to efficiently store only mutations for each individual, instead of full genomes.

#### A.3.1 Hardware

All models were trained on the Cambridge-1 Nvidia supercomputer system, using 16 nodes, each equipped with eight A100 GPUs, leading to a total of 128 A100 GPUs used. During training, model weights were replicated on each GPU, while batches were sharded across GPUs. Gradients were computed on each shard and accumulated before being averaged across devices and backpropagated. We relied on the jax library[4] that relied on the NCCL[5] protocol to handle communications between nodes and devices, and observed almost linear decrease of the training time with respect to the number of GPU available. The 500M parameters models were trained on a single node for a day, while the 2.5B models models required the whole cluster for 28 days to be trained.

All fine-tuning runs were performed on a single node with eight A100 GPUs. As for the training runs, the models weights were replicated and batches distributed across GPUs. As we used a batch size of eight for fine-tuning, each GPU processed a single sample before averaging the gradients and applying them. On average, a fine-tuning run lasted 20 minutes for the 500M parameter models, and 50 minutes for the 2.5B parameter models.

For the probing experiments, all embeddings (for all sequences in all downstream tasks, for selected layers of each model) were computed and stored on a single node with eight A100 GPUs, requiring two days to compute. Then, 760,000 downstream models were fit on a cluster of 3000 CPUs, requiring 2.5 days.

### A.4 Downstream tasks

#### A.4.1 Epigenetic marks prediction

We downloaded the dataset[6] of epigenetic marks identified in the yeast genome, namely acetylation and metylation nucleosome occupancies [20]. Nucleosome occupancy values in these ten datasets were obtained with Chip-Chip experiments [47] and further processed into positive and negative observations to provide epigenetic training data for the following histone marks: H3, H4, H3K9ac, H3K14ac, H4ac, H3K4me1, H3K4me2, H3K4me3, H3K36me3 and H3K79me3.

#### A.4.2 Promoter sequence prediction

We built a dataset of promoter sequences to evaluate the capabilities of the model to identify promoter motifs. Following the DeePromoter method [23], we considered sequences of 300 base pairs (bp) genome-wide, selected to span 249bp upstream and 50bp downstream of transcription start sites. This resulted in 29,597 promoter regions, 3,065 of which were TATA-box promoters. For each promoter region sample, a negative sample (non-promoter sequence) with matching length was constructed by splitting the promoter sequence in 20 sub-sequences and randomly selecting and shuffling 12 of them, while keeping the other 8 intact. The final dataset is then composed of 59,194 sequences.

#### A.4.3 Enhancer sequence prediction

We used a single dataset presented prior [21] to evaluate the capacity of the transformer models to provide an accurate representation of enhancer sequences. The original dataset included 742 strong enhancers, 742 weak enhancers and 1484 non-enhancers, but previous work [21] showed that training

[4] https://jax.readthedocs.io/en/latest/_autosummary/jax.pmap.html
[5] https://developer.nvidia.com/nccl
[6] http://www.jaist.ac.jp/~tran/nucleosome/members.htm

performance was increased by augmenting the dataset with 6000 synthetic enhancers and 6000 synthetic non-enhancers produced through a generative model.

#### A.4.4 Splice site prediction

We used two datasets to evaluate splice site prediction. First, we downloaded the dataset used by SpliceFinder [22], which was composed of donor, acceptor, and non-splice sites, containing sequences detected in human genes. Each sequence was 400 nucleotides long and contained either a splicing site in the center (i.e. an acceptor or donor site), or a non-splicing site. Second, to leverage a more diverse set of splicing sites, we also downloaded the training dataset[7] of the Spliceator model [24]. Contrary to the SpliceFinder dataset, this set was based primarily on the G3PO database, which included sequences from 147 phylogenetically diverse organisms (ranging from protists to primates, including humans). All sequences were 600bp and included either a splicing site at the center (i.e. an acceptor or donor site), or a non-splicing site. Following the Spliceator study, we only included sequences that were part of the balanced 'Gold Standard' dataset (referred to as 'GS_1' in the study).

#### A.4.5 Chromatin Profiles Prediction

We used the dataset[8] compiled in Zhou et al. 2015 [28] for chromatin profiles prediction. The dataset is composed of 2.4 million sequences, each of size 1000 nucleotides, and associated with 919 chromatin features. These include 690 transcription factor (TF), 125 DNAse, and 104 histone features. The model is trained simultaneously on the 919 classification tasks, with 919 independent classification heads, and a loss taken as the average of the cross entropy losses. Since each label is highly unbalanced and is composed mostly of negative samples, the losses associated with positive samples are upscaled by a factor of 8. Contrarily to the DeepSEA method [28], which trained two models independently, one on the forward sequences and one on the corresponding reverse-complementary, and evaluated the average of their predictions, the model presented here was trained only on the forward sequences.

#### A.4.6 Enhancer Activity

We used the enhancer activity dataset [9] released by in P. de Almeida et al. 2022 [27]. The dataset is composed of 484,052 housekeeping and developmental enhancer sequences, each of size 249 nucleotides, with associated log fold change expression values. Following the methodology used in [3], we chose to treat this regression task as a multi-label classification problem. Specifically, each label $y$ was discretized over a set of 50 values $(b_i)_{i\in[1,50]}$, evenly spaced between the minimum and maximum value. For each label, the model predicts the normalized weights $(w_i)_{i\in[1,50]}$ such that

$$y = \sum_{i=1}^{50} w_i b_i.$$

#### A.4.7 Performance Metrics

The baselines considered in this study used different performance metrics, as each study used datasets that varied in size and number of positive/negative observations (Table 5). In order to benchmark the transformer models presented here, we decided to use the following ranking across performance metrics: The Matthews Correlation Coefficient (MCC) was preferred over the F1-scores, which was preferred over accuracy. The term *performance*, which is used throughout the text, should therefore be considered as an umbrella term to refer to the performance used by each baseline.

### A.5 Additional Performance Analysis

#### A.5.1 Reconstruction accuracy and perplexity

We studied how pre-trained models could reconstruct masked tokens. We considered a trained language model with parameters $\theta$. Within a nucleotide sequence $\mathbf{s}$ of interest, we masked tokens using one of

[7] https://git.unistra.fr/nscalzitti/spliceator/-/tree/master/Data/Datasets
[8] http://deepsea.princeton.edu/media/code/deepsea_train_bundle.v0.9.tar.gz
[9] https://zenodo.org/record/5502060#.Y9qO7hzMLUc

two strategies (i.e. we replaced the tokens at these positions with the mask token [MASK]). We either masked the central token of the sequence only, or we masked randomly 15% of the tokens within the sequence. The masked sequence is then fed to the model and the probabilities over tokens at each masked position are retrieved. The loss function $l(\theta, \mathbf{s})$ and the accuracy $acc(\theta, \mathbf{s})$ are defined as follows:

$$\begin{cases} l(\theta, \mathbf{s}) = \sum_{i \in \mathcal{P}_{\text{masked}}} \sum_{\text{tok} \in \mathcal{V}} \log p(\theta, i, \text{tok}) \cdot \mathbf{1}(\text{tok} = \mathbf{s}(i)) \\ acc(\theta, \mathbf{s}) = \frac{1}{|\mathcal{P}_{\text{masked}}|} \sum_{i \in \mathcal{P}_{\text{masked}}} \mathbf{1}\left(\underset{\text{tok} \in \mathcal{V}}{\text{argmax}}(\log p(\theta, i, \text{tok})) = \mathbf{s}(i)\right) \end{cases}$$

where $\mathcal{P}_{\text{masked}}$ is the set of masked positions and $\mathcal{V}$ is the vocabulary, i.e. the set of all existing tokens. The perplexity is usually defined in the context of autoregressive generative models. Here, we rely on an alternative definition used in Rives [4], and define it as the exponential of the loss function computed over the masked positions:

$$\text{perplexity}(\theta, \mathbf{s}) = 2^{l(\theta, \mathbf{s})}. \tag{1}$$

Perplexity measures how well a model can reconstruct masked positions, and is a more refined measure than accuracy, as it does also account for magnitude. In contrast to accuracy, lower perplexity suggests better reconstruction ability, thus better performance.

#### A.5.2 Functional variant prioritization

To obtain genetic variants with varying levels of associated severity we used the Variant Effect Prediction (VEP) software [33] and annotated sequences across the human genome. Specifically, we randomly sampled sequences throughout the human genome, and kept genetic variants within those sequences annotated to any of the following classes: "intron variant", "intergenic variant", "regulatory region variant", "missense variant", "3 prime UTR variant", "synonymous variant", "TF binding site variant", "5 prime UTR variant", "splice region variant", and "stop gained variant". After keeping only single nucleotide polymorphisms and filtering out variants annotated to more than one consequence (e.g. those annotated as stop gained and splice variants), we obtained a final dataset composed of 920 genetic variants per class.

As a positive set of functional genetic variants we compiled SNPs from four different resources. We used SNPs associated to gene expression (i.e. expression quantitative trait loci [eQTLs]) and to methylation variation (i.e., meQTLs) from the Genome-Wide Repository of Associations Between SNPs and Phenotypes (GRASP) database [48] with a P-value $<10^{-12}$, SNPs with "likely pathogenic "annotations from ClinVar [49] and SNPs reported in The Human Gene Mutation Database (HGMD) (public version 2020.4) [50]. After these filters, we retained a total of 80,590, 11,734, 70,224, and 14,626 genetic variants for the eQTLs, meQTLs, ClinVar, and HGMD SNP datasets, respectively. For each of these four datasets, we then constructed a set of negative variants based on SNPs from the 1000 Genomes Project with a minor allele frequency (MAF) >5%, that did not overlap with any variant reported in the dataset tested, and that were within 100kb of the associated variants, resulting in four balanced datasets.

To compute zero-shot based scores for a given site of interest we did the following: For each SNP, we obtained a 6,000 bp sequence centered on the SNP of interest based on the human reference genome. We then, created two sequences, one carrying the reference allele and a second carrying the alternative allele at the SNP position. We then computed several zero-shot scores that capture different aspects of the vector distances in the embedding space between those two sequences, namely: the L1 distance (Manhattan), (ii) the L2 distance (Euclidean), (iii) the cosine similarity, and (iv) the dot-product (not normalized cosine similarity). We also computed the loss of the alternative allele and the difference in the loss between the sequence carrying the alternative and reference alleles, as two additional zero-shot scores. In the case of functional variants, in addition to zero-shot scores, we also fine-tuned the transformer models to classify positive and negative variants. We employed a similar strategy as the one previously described, with the primary difference being that the training and test sets were divided by chromosomes and strictly kept non-overlapping. Specifically, we divided the 22 chromosomes into 5

sets and sequentially used each of them as a test set and the 4 others as a training set. By fine-tuning on the training set, we could derive probabilities of being a positive variant for each sequence in the test set. We use those probabilities as a score for each SNP.

To compare these predictions against other methods, we randomly sampled 10,000 positive and negative SNPs from each of the four datasets. We then used the Combined Annotation Dependent Depletion tool (version GRCh38-v1.6) to compute CADD, GERP, phastCons, and phyloP scores. The DeepSEA scores were computed using the Beluga model available at: https://hb.flatironinstitute.org/sei/. The score considered was the "disease impact score"reported for each SNP.

### A.5.3 Attention Maps Analysis

We analysed how attention maps gathered from the pre-trained models capture key genomic elements. We followed a methodology proposed in previous work [32]. For a genomic element, we define the indicator function over tokens $f(i)$ that equals 1 if one or several nucleotides within token $i$ belong to the element, and 0 otherwise. We computed the average proportion of attention focused on that genomic element aggregated over a dataset of nucleotide sequences $\mathbf{X}$ as:

$$p_\alpha(f) = \sum_{\mathbf{x}\in\mathbf{X}} \sum_i \sum_j f(i)\mathbf{1}\left(\alpha(i,j) < \mu\right)$$

where $\alpha(i,j)$ is the attention coefficient between tokens $i$ and tokens $j$ defined such that $\sum_i \alpha_{i,j} = 1$ and $\mu$ is a confidence threshold.

As an additional analysis, we also use perform probing using genomic analysis as labels: we extract sequence embedding for several layers of each model and fit a logistic regression to separate the 5 genomic elements.
We computed the values of $p_\alpha(f)$ for all the heads and all the layers of all models, and considered nine elements ( "5' UTR", "3' UTR", "exon", "intron", "enhancer", "promoter", "CTCF binding site", "open chromatin", and "transcription factor binding sites"). We perform these analyses over a dataset made of 90,000 sequences, 10,000 per feature, of length 6k bp extracted from the Human reference genome. The average proportion of tokens belonging to each element can be found in Table 9. For each sequence, the position of the feature within the sequence was sampled uniformly during the dataset creation. As suggested in previous work [32], we selected a confidence threshold $\mu = 0.3$ for all experiments.

We considered that a feature is captured by an attention head if the quantity $p_\alpha(f)$ is significantly greater than the natural occurring frequency of the feature within the dataset (Table 9). To validate this, we conducted a two proportion z-test with the null hypothesis as the natural frequency of the feature, and the alternate hypothesis as $p_\alpha(f)$. The total number of heads of each model is used as a Bonferroni correction to the significance level, $\alpha$, of 0.05. We computed z-scores and associated p-value for each head in every model for every genomic element as follows:

$$Z = \frac{\hat{p}_1 - \hat{p}_2}{\sqrt{\hat{p}\left(1-\hat{p}\right)\left(\frac{1}{n_1} + \frac{1}{n_2}\right)}}$$

where $\hat{p}_1$ represents the proportion of attention above $\mu$ associated with each genomic element, $\hat{p}_2$ represents the proportion of the sequence occupied by the genomic element, $n_1$ is the total number of sequence positions with attention above $\mu$, $n_2$ is the total number of sequence positions. Attention heads with p-values below the Bonferonni corrected significance level are considered to be significant.

| | 2.5B 1000G | 2.5B Multispecies | 500M 1000G | 500M Human Ref |
|---|---|---|---|---|
| alphabet | k-mers | k-mers | k-mers | k-mers |
| k_for_kmers | 6 | 6 | 6 | 6 |
| num_warmup_updates | 16000 | 16000 | 16000 | 16000 |
| warmup_init_lr | 0.00005 | 0.00005 | 0.00005 | 0.00005 |
| warmup_end_lr | 0.0001 | 0.0001 | 0.0001 | 0.0001 |
| training_set_proportion | 0.95 | 0.95 | 0.95 | 0.95 |
| masking_ratio | 0.15 | 0.15 | 0.15 | 0.15 |
| masking_prob | 0.8 | 0.8 | 0.8 | 0.8 |
| random_token_prob | 0.0 | 0.1 | 0.0 | 0.1 |
| alphabet_size | 4105 | 4105 | 4105 | 4105 |
| prepend_bos | True | True | True | True |
| append_eos | False | False | False | False |
| attention_heads | 20 | 20 | 20 | 20 |
| embed_dim | 2560 | 2560 | 1280 | 1280 |
| ffn_embed_dim | 10240 | 10240 | 5120 | 5120 |
| num_layers | 32 | 32 | 24 | 24 |
| token_dropout | True | True | True | True |
| num_parameters | 2547800585 | 2547800585 | 485729545 | 485729545 |
| dataset | 1000G | Multispecies | 1000G | Human Reference |

Supplementary Table 1: Models hyperparameters.

# B Supplementary Tables

| Population name | Population code | Number of individuals |
|---|---|---|
| African Ancestry SW | ASW | 74 |
| African Caribbean | ACB | 116 |
| Bengali | BEB | 131 |
| British | GBR | 91 |
| CEPH | CEU | 179 |
| Colombian | CLM | 132 |
| Dai Chinese | CDX | 93 |
| Esan | ESN | 149 |
| Finnish | FIN | 99 |
| Gambian Mandinka | GWD | 178 |
| Gujarati | GIH | 103 |
| Han Chinese | CHB | 103 |
| Iberian | IBS | 157 |
| Japanese | JPT | 104 |
| Kinh | KHV | 2 |
| Kinh Vietnamese | KHV | 120 |
| Luhya | LWK | 99 |
| Mende | MSL | 99 |
| Mexican Ancestry | MXL | 97 |
| Peruvian | PEL | 122 |
| Puerto Rican | PUR | 139 |
| Punjabi | PJL | 146 |
| Southern Han Chinese | CHS | 163 |
| Tamil | STU | 114 |
| Telugu | ITU | 107 |
| Toscani | TSI | 107 |
| Yoruba | YRI | 178 |

Supplementary Table 2: Number of individuals per population in the 1000G dataset.

| Class | Number of species | Number of nucleotides (B) |
|---|---|---|
| Bacteria | 667 | 17.1 |
| Fungi | 46 | 2.3 |
| Invertebrate | 39 | 20.8 |
| Protozoa | 10 | 0.5 |
| Mammalian Vertebrate | 31 | 69.8 |
| Other Vertebrate | 57 | 63.4 |

Supplementary Table 3: Genomes in the multispecies dataset.

| Class | Selected Species |
|---|---|
| Bacteria | Escherichia coli |
| Fungi | Saccharomyces cerevisiae |
| Invertebrate | Caenorhabditis elegans, Drosophila melanogaster |
| Protozoa | Plasmodium vivax, Plasmodium falciparum |
| Mammalian Vertebrate | Homo sapiens, Mus musculus, Rattus norvegicus |
| Other Vertebrate | Danio rerio, Xenopus tropicalis |

Supplementary Table 4: Model organisms genomes in the multispecies dataset.

| | Num train sequences | Num test sequences | Max sequence length in bp | Reported Metric | Baseline Name |
|---|---|---|---|---|---|
| H3K4me3 | 25953 | 2884 | 500 | MCC | SVM [20] |
| H3K4me2 | 27614 | 3069 | 500 | MCC | SVM [20] |
| H3K36me3 | 31392 | 3488 | 500 | MCC | SVM [20] |
| H3K9ac | 25003 | 2779 | 500 | MCC | SVM [20] |
| Splice donor | 19775 | 2198 | 600 | F1 | SpliceFinder [22] |
| Splice site all | 27000 | 3000 | 400 | Acc | SpliceFinder [22] |
| H4ac | 30685 | 3410 | 500 | MCC | SVM [20] |
| H3K4me1 | 28509 | 3168 | 500 | MCC | SVM [20] |
| Enhancer | 14968 | 400 | 200 | MCC | LSTM-CNN [21] |
| Enhancer types | 14968 | 400 | 200 | MCC | LSTM-CNN [21] |
| H4 | 13140 | 1461 | 500 | MCC | SVM [20] |
| Splice acceptor | 19961 | 2218 | 600 | F1 | SpliceFinder [22] |
| H3K79me3 | 25953 | 2884 | 500 | MCC | SVM [20] |
| Promoter non-TATA | 47767 | 5299 | 300 | F1 | DeePromoter [23] |
| Promoter all | 53276 | 5920 | 300 | F1 | DeePromoter [23] |
| H3K14ac | 29743 | 3305 | 500 | MCC | SVM [20] |
| H3 | 13468 | 1497 | 500 | MCC | SVM [20] |
| Promoter TATA | 5509 | 621 | 300 | F1 | DeePromoter [23] |

Supplementary Table 5: Downstream tasks metadata.

| | 2.5B MS Probed | 500M 1000G Probed | 2.5B 1000G Probed | 500M HR Probed | 2.5B MS Finetuned | 500M 1000G Finetuned | 2.5B 1000G Finetuned | 500M HR Finetuned | Peer-Reviewed Baselines |
|---|---|---|---|---|---|---|---|---|---|
| H3K4me3 | 0.357 | 0.276 | 0.290 | 0.237 | 0.421 | 0.288 | 0.281 | 0.263 | **0.460** |
| H3K4me2 | 0.322 | 0.286 | 0.297 | 0.257 | 0.326 | 0.288 | 0.300 | 0.281 | **0.390** |
| H3K36me3 | 0.583 | 0.480 | 0.488 | 0.454 | **0.632** | 0.461 | 0.533 | 0.467 | 0.540 |
| Splice donor | 0.950 | 0.828 | 0.899 | 0.812 | **0.984** | 0.975 | 0.982 | 0.972 | 0.953 |
| H3K9ac | 0.527 | 0.481 | 0.462 | 0.427 | **0.575** | 0.488 | 0.508 | 0.462 | 0.540 |
| Splice site all | 0.762 | 0.670 | 0.718 | 0.681 | **0.983** | 0.971 | 0.978 | 0.972 | 0.965 |
| H4ac | 0.448 | 0.372 | 0.394 | 0.340 | **0.501** | 0.342 | 0.423 | 0.344 | 0.500 |
| H3K4me1 | 0.497 | 0.366 | 0.415 | 0.335 | **0.559** | 0.379 | 0.445 | 0.358 | 0.440 |
| Enhancer | 0.507 | 0.498 | 0.487 | 0.508 | 0.580 | 0.580 | **0.593** | 0.535 | 0.505 |
| Enhancer types | 0.435 | 0.419 | 0.425 | 0.422 | 0.474 | 0.469 | **0.500** | 0.485 | 0.395 |
| H4 | 0.812 | 0.775 | 0.783 | 0.742 | **0.822** | 0.752 | 0.789 | 0.762 | 0.760 |
| Splice acceptor | 0.901 | 0.840 | 0.872 | 0.828 | **0.990** | 0.972 | 0.985 | 0.965 | 0.944 |
| H3K79me3 | 0.616 | 0.578 | 0.574 | 0.554 | 0.642 | 0.579 | 0.592 | 0.577 | **0.650** |
| Promoter non-TATA | 0.967 | 0.947 | 0.956 | 0.937 | **0.977** | 0.955 | 0.969 | 0.956 | 0.960 |
| Promoter all | 0.966 | 0.946 | 0.955 | 0.937 | **0.974** | 0.953 | 0.966 | 0.954 | 0.956 |
| H3K14ac | 0.506 | 0.401 | 0.420 | 0.388 | **0.550** | 0.399 | 0.471 | 0.377 | 0.510 |
| H3 | 0.797 | 0.735 | 0.763 | 0.685 | **0.814** | 0.756 | 0.776 | 0.737 | 0.730 |
| Promoter TATA | 0.959 | 0.943 | 0.952 | 0.941 | **0.964** | 0.941 | 0.958 | 0.948 | 0.940 |

Supplementary Table 6: Downstream performance per task for all models and baselines. For probed models, the reported performance corresponds to the best layer and best downstream model.

| | Model | Layer probed | Number of hidden layers | Hidden layer dimension | Batch size | Learning rate |
|---|---|---|---|---|---|---|
| H3K4me3 | 2.5B MS | 17 | 1 | 256 | 500 | 1e-4 |
| H3K4me2 | 2.5B MS | 10 | 1 | 256 | 500 | 1e-4 |
| H3K36me3 | 2.5B MS | 17 | 1 | 256 | 500 | 1e-4 |
| H3K9ac | 2.5B MS | 10 | 1 | 256 | 500 | 1e-4 |
| Splice site all | 2.5B MS | 14 | 1 | 512 | 650 | 6.1e-5 |
| H4ac | 2.5B MS | 17 | 1 | 256 | 500 | 1e-4 |
| H3K4me1 | 2.5B MS | 14 | 2 | 512 | 500 | 1e-4 |
| Enhancer | 500M HR | 18 | 2 | 512 | 500 | 1e-4 |
| H4 | 2.5B MS | 28 | 1 | 256 | 500 | 1e-4 |
| Splice acceptor | 2.5B MS | 10 | 1 | 512 | 500 | 1e-4 |
| H3K79me3 | 2.5B MS | 14 | 2 | 512 | 650 | 1.8e-4 |
| Promoter non-TATA | 2.5B MS | 10 | 1 | 512 | 500 | 1e-4 |
| Promoter all | 2.5B MS | 10 | 1 | 512 | 500 | 1e-4 |
| H3K14ac | 2.5B MS | 14 | 1 | 512 | 500 | 1e-4 |
| H3 | 2.5B MS | 14 | 1 | 256 | 500 | 1e-4 |
| Promoter TATA | 2.5B MS | 10 | 1 | 512 | 500 | 1e-4 |

Supplementary Table 7: Hyperparameters used to train the model with the best cross-validation performance for each of the downstream task, along with the model and the layer used to compute the corresponding embeddings. The best probe method for the tasks **Enhancer types** and **Splice donor** was the logistic regression with default hyperparameters.

| Model Name | Normalized summed downstream performance |
|---|---|
| 500M Human Ref Probed | 0.582 |
| 500M 1000G Probed | 0.602 |
| 2.5B 1000G Probed | 0.619 |
| 500M Human Ref Finetuned | 0.634 |
| 500M 1000G Finetuned | 0.641 |
| 2.5B Multispecies Probed | 0.661 |
| 2.5B 1000G Finetuned | 0.669 |
| Peer-Reviewed Baselines | 0.674 |
| 2.5B Multispecies Finetuned | **0.709** |

Supplementary Table 8: Normalized summed downstream performance. These values are obtained by summing performances over all tasks for each model and dividing by the number of tasks.

| | Percentage of sequences containing this element | Mean Percentage of Sequence Length belonging to this element |
|---|---|---|
| Enhancer | 10.18% | 11.09% |
| Open chromatin | 11.32% | 6.16% |
| 3' UTR | 11.37% | 4.30% |
| TF binding | 11.28% | 6.38% |
| Intron | 10.45% | 11.08% |
| Exon | 11.87% | 3.18% |
| 5'UTR | 11.84% | 1.99% |
| Promoter | 10.95% | 9.78% |
| CTCF Binding Site | 10.69% | 7.36% |

Supplementary Table 9: Proportions of the genomic elements in the dataset used for attention maps and embedding space visualization experiments.

| | CDS | CTCF | Enhancer | Exon | 5'UTR | Intron | Open chromatin | Promoter | TF | 3'UTR | No. heads |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Human ref (500M) | 40 | 26 | 27 | 33 | 37 | 37 | 27 | 33 | 26 | 37 | 480 |
| 1000G (500M) | 52 | 27 | 24 | 48 | 41 | 40 | 26 | 55 | 27 | 49 | 480 |
| Multispecies (2.5B) | 89 | 35 | 36 | 72 | 42 | 117 | 44 | 50 | 74 | 51 | 640 |
| 1000G (2.5B) | 38 | 32 | 28 | 40 | 33 | 61 | 40 | 55 | 45 | 54 | 640 |

Supplementary Table 10: Number of significant attention heads capturing gene features and regulatory elements. Note that the number of total heads varies between the 500M and 2.5B models.

# C Supplementary Figures

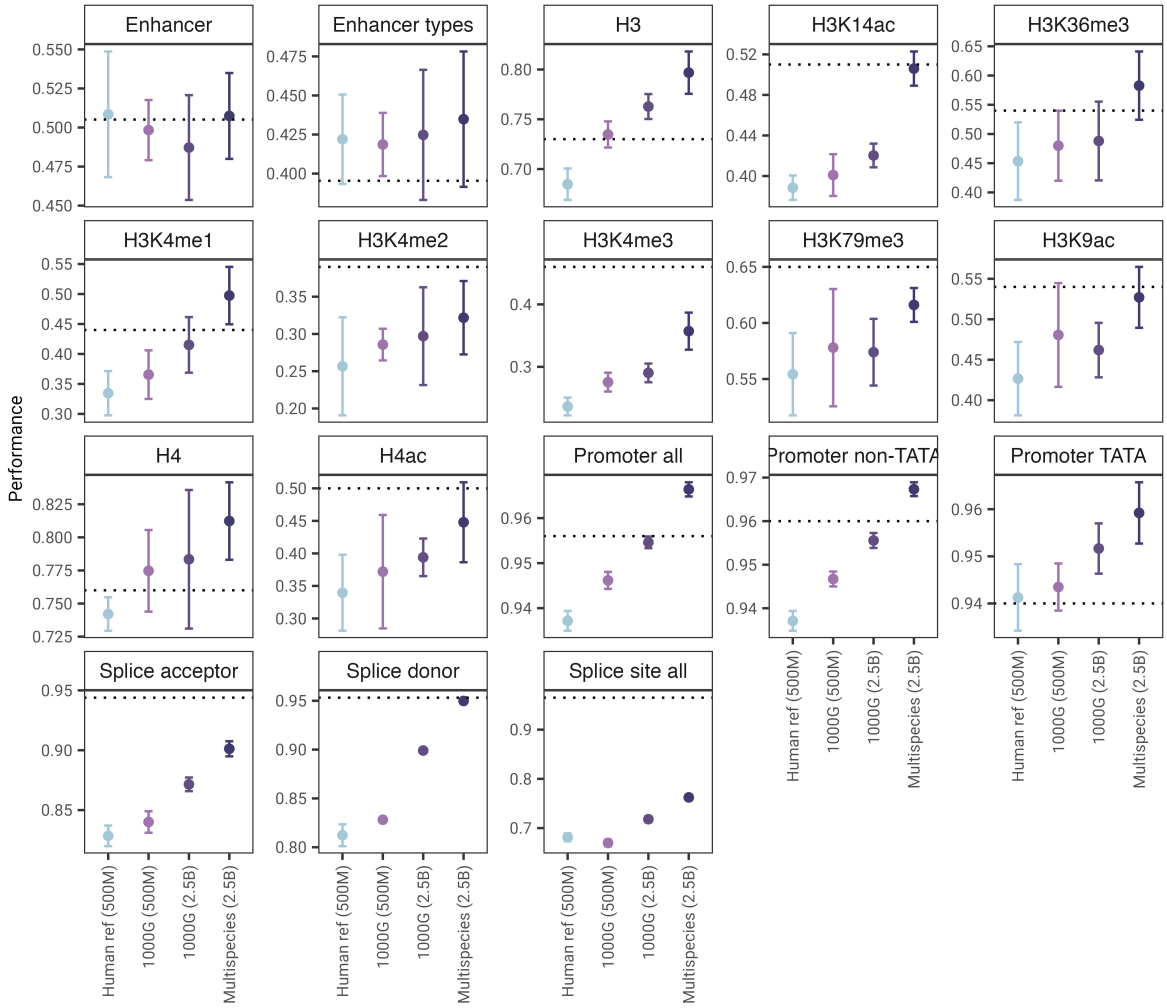
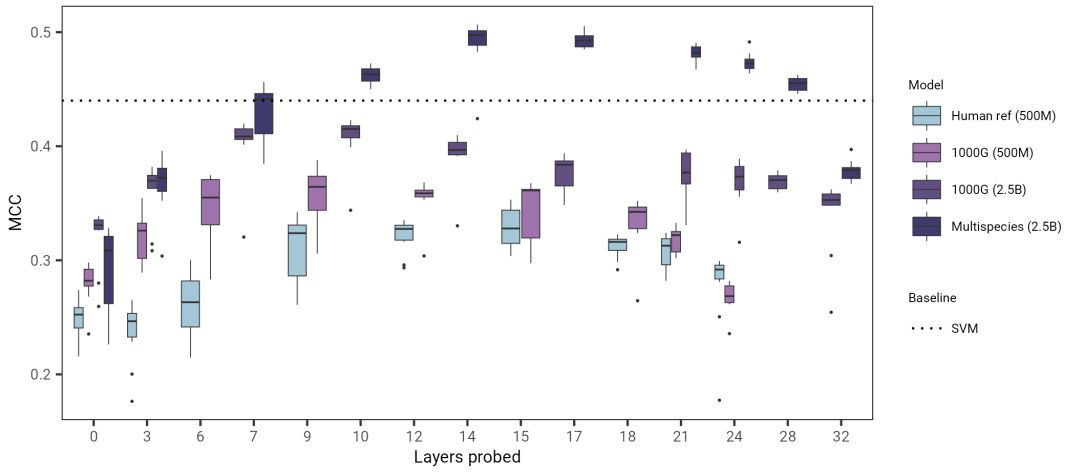
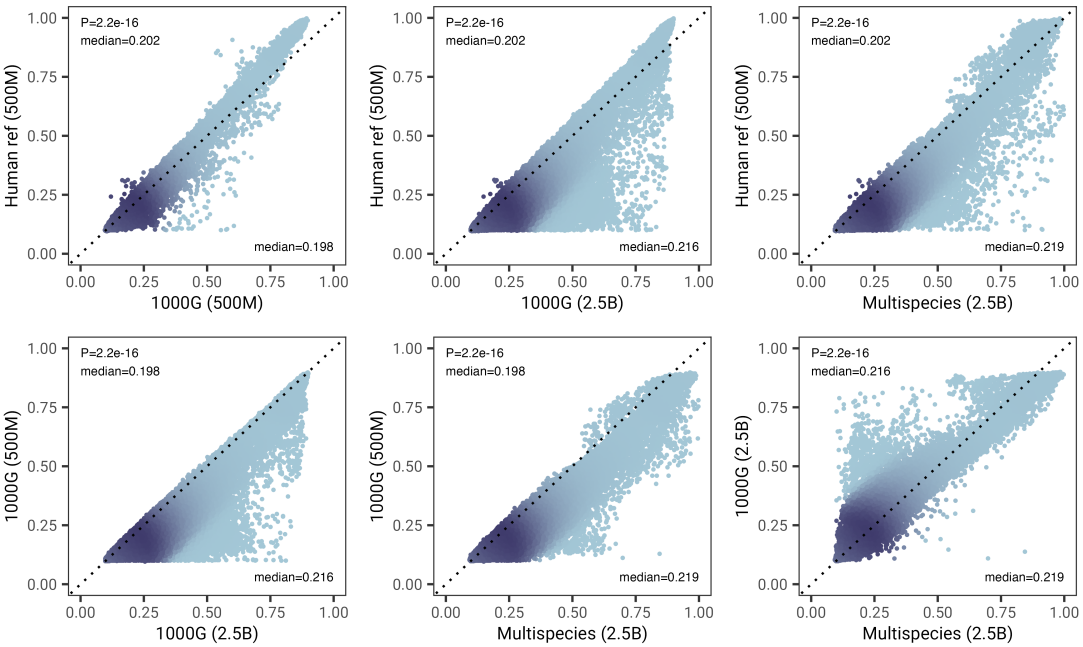


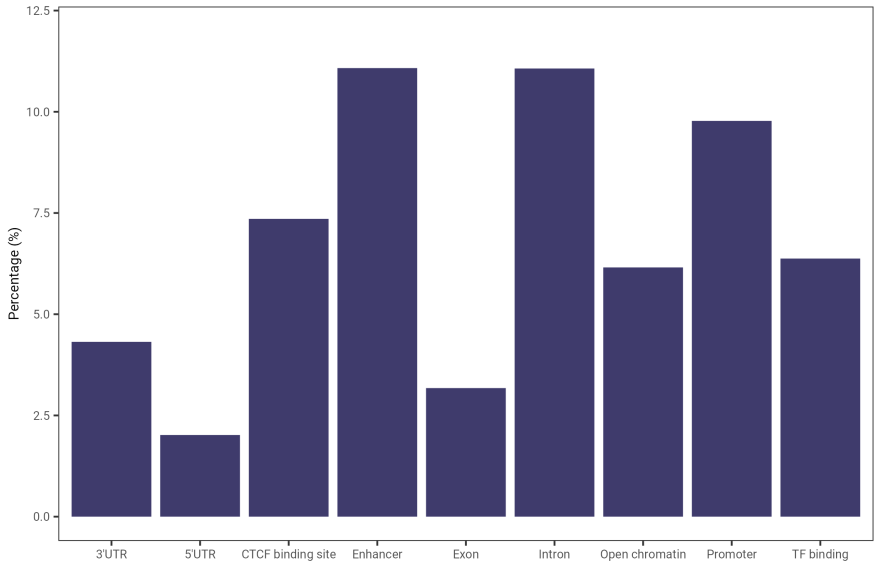

Figure 1: Probing performance results on test sets across downstream tasks for each Nucleotide Transformer model. The performances of the best layers and best downstream models are shown. The error bars represent 2 std from the 10-fold cross-validation procedure.



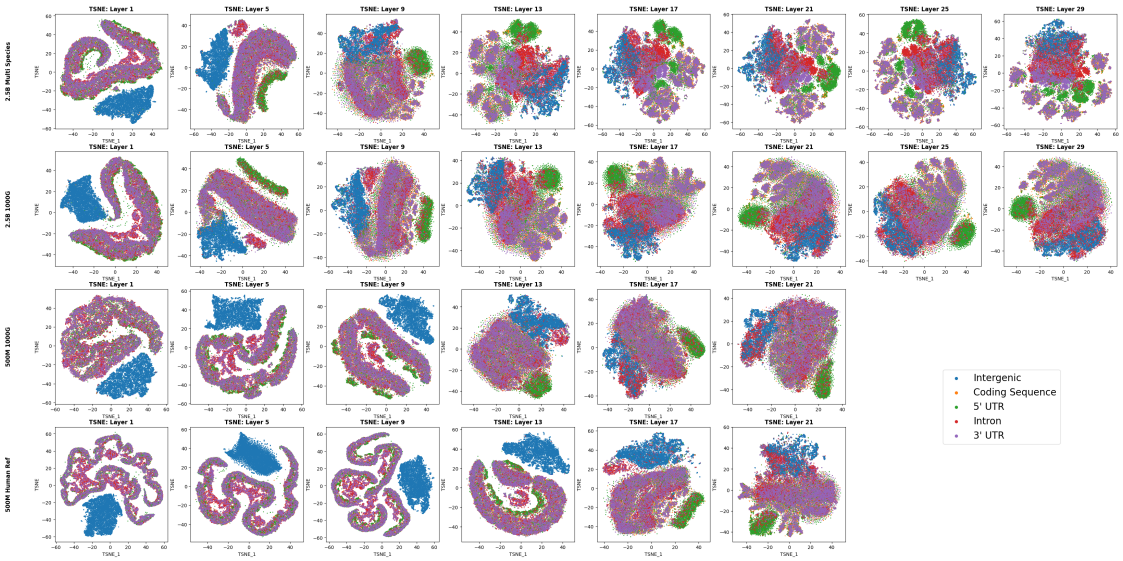

Supplementary Figure 2: Probing performance across layers for the H3K4me1 prediction task. Boxplots show the MCC values of 10 layers based on 10 fold cross-validation experiments.



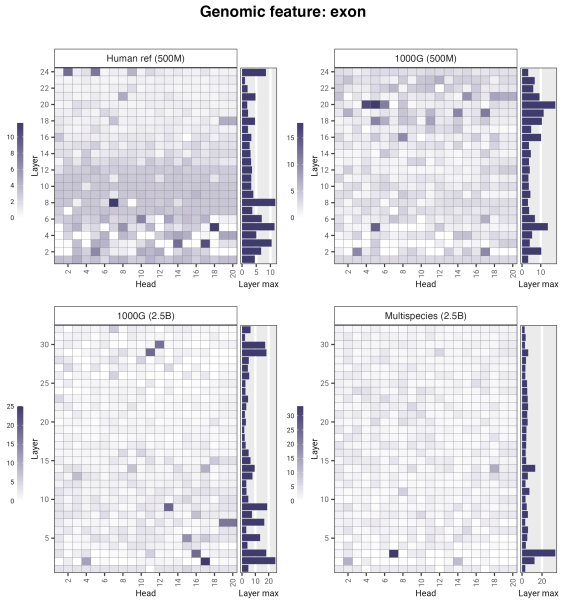

Supplementary Figure 3: Pairwise comparison of token reconstruction accuracy across transformer models. P-values refer to a two-sided Wilcoxon signed rank test between models. Median values for the two models compared are shown.



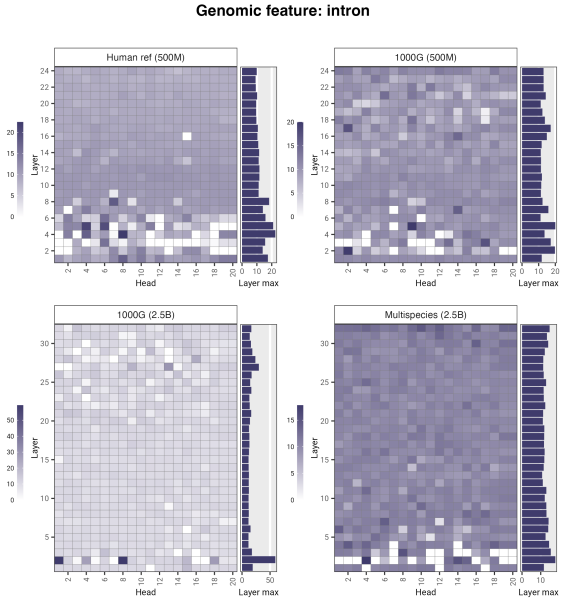

Supplementary Figure 4: Genomic element length's percentage of input sequence (6kbp). This corresponds to the dataset used for attention maps and embedding spaces visualization experiments. Details in Table 9.



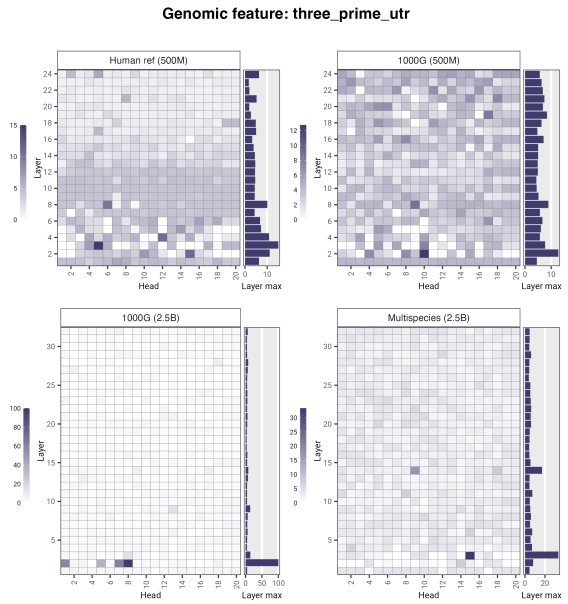

Supplementary Figure 5: t-SNE projections of embeddings of 5 genomic elements across transformer models and layers.



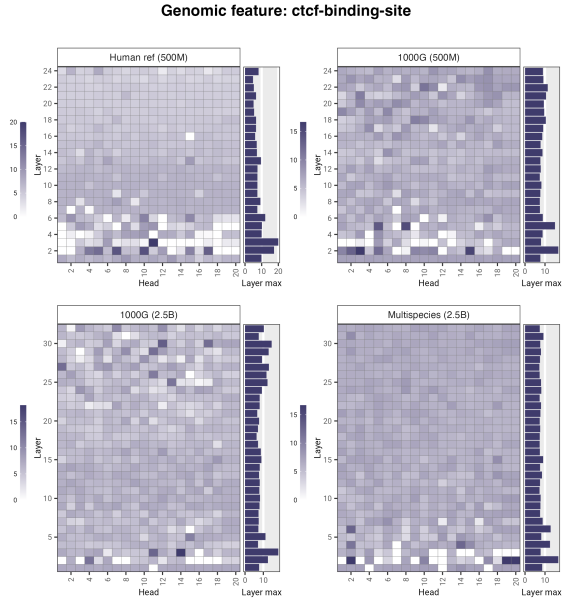

Supplementary Figure 6: Nucleotide Transformer models' attention percentages per head computed for exons.



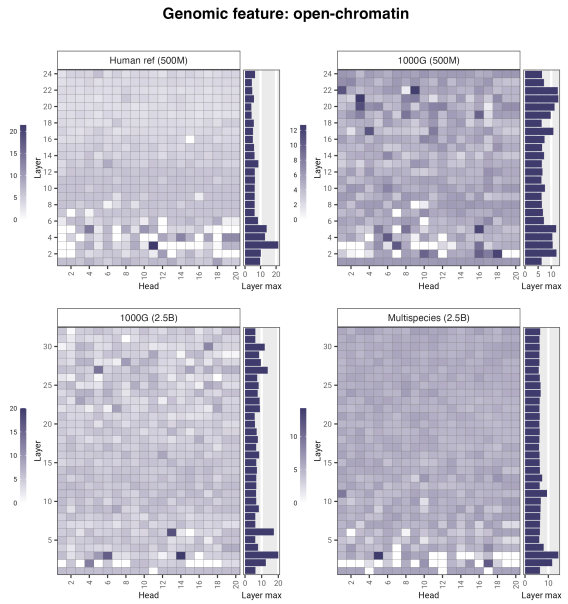

Supplementary Figure 7: Nucleotide Transformer models' attention percentages per head computed for introns.



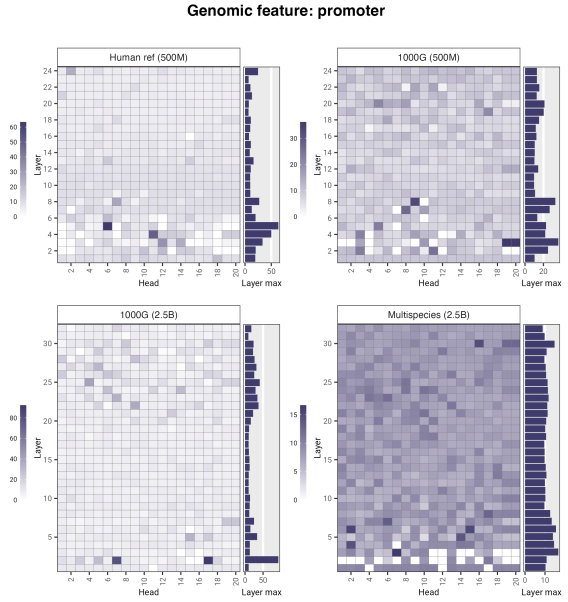

Supplementary Figure 8: Nucleotide Transformer models' attention percentages per head computed for 3' UTR regions.



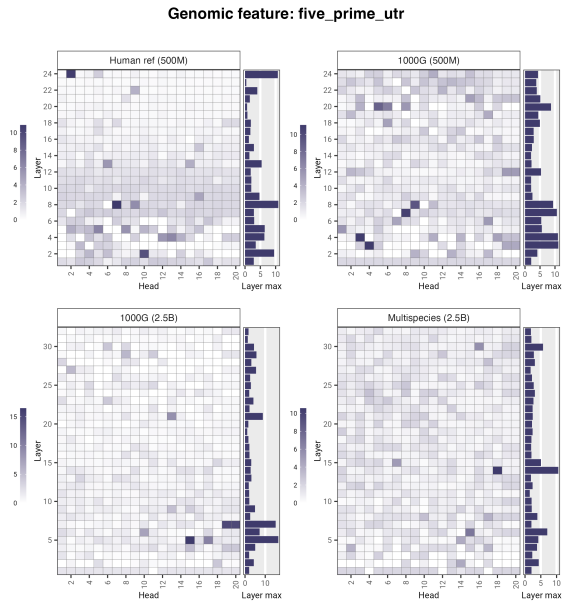

Supplementary Figure 9: Nucleotide Transformer models' attention percentages per head computed for CTCF binding sites.



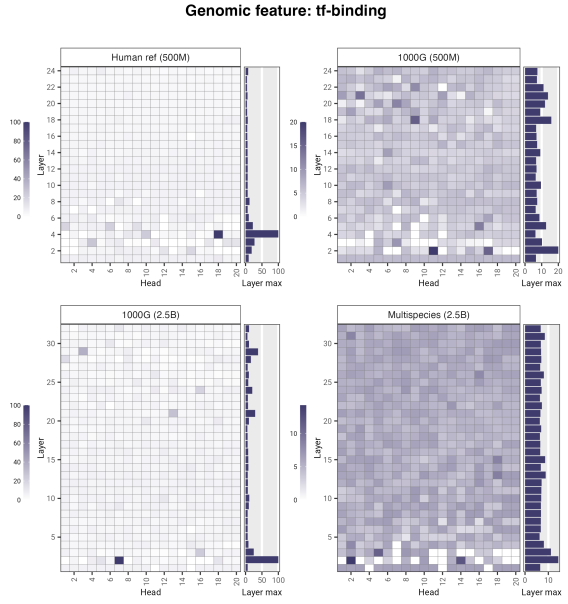

Supplementary Figure 10: Nucleotide Transformer models' attention percentages per head computed for open chromatin.



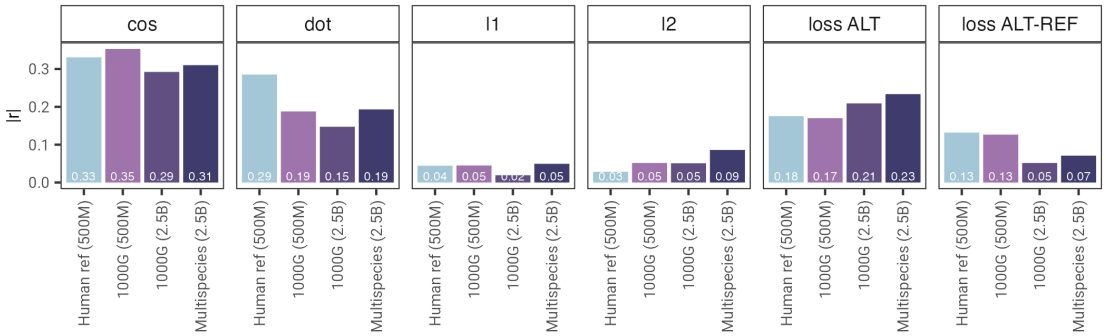

Supplementary Figure 11: Nucleotide Transformer models' attention percentages per head computed for promoters.




Supplementary Figure 12: Nucleotide Transformer models' attention percentages per head computed for 5' UTR regions.




Supplementary Figure 13: Nucleotide Transformer models' attention percentages per head computed for transcription factor binding sites.




Supplementary Figure 14: Performance of zero-shot based scores for SNPs annotations based on Ensembl Variant Effect Prediction (VEP). Performance is based on the Pearson correlation between the scores and severity as estimated by Ensembl.