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 representations alone match or outperform specialized methods on 11 of 18 prediction tasks, and up to 15 after fine-tuning. Despite no supervision, the transformer models learnt to focus attention on key genomic elements, including those that regulate gene expression, such as enhancers. Lastly, we demonstrate that utilizing model representations alone 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 alone.

Introduction

Foundation models in artificial intelligence (AI) refer to large models incorporating millions of parameters and 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 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 modelling [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 (DL) models were 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

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

networks (CNN) with transformer architectures enabling the encoding of regulatory elements as far as 100 kilobases (kb) upstream [14]. Modern genomics research has led to a tremendous increase in the volume of sequencing data available, from repeat measurements of single organisms (e.g., to study population variations [15]), to assays measuring multiplexed samples (e.g., to measure the biodiversity of earth's and human gut microbiomes [16, 17]). This deluge of data poses both an opportunity and a challenge; on the one hand, abundant genomic data able to uncover the intricate patterns of natural variability across species and populations is vastly available; on the other hand, powerful deep learning methods, that can operate at large scale, are required to perform accurate signal extraction from this large amount of unlabelled genomic data. Large foundation models trained on sequences of nucleotides appear to be the natural choice to seize the opportunity in genomics research, with attempts to explore different angles recently proposed [18, 19, 19, 20].

With the Nucleotide Transformer we crucially add to recent attempts to encode genomics through foundation models. We built four distinct foundation language models of different sizes, ranging from 500M up to 2.5B parameters, and pre-trained them on three different datasets encompassing the Human reference genome, a collection of 3,200 diverse human genomes, and 850 genomes from several species (Table 6). Once trained, we leveraged representations, or embeddings, of each of these models to further train downstream models on 18 genomics 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 probed the embeddings' ability 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 solutions.

Results

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

We developed a collection of transformer DNA language models, dubbed Nucleotide Transformer, able to learn general nucleotide sequence representations from unlabeled genomic data. After pre-training, model representations were used to solve domain-specific tasks with simple models based on a small number of parameters, such as logistic regression, across 18 different genomic prediction tasks (Fig. 1a) and for which baseline performance metrics were available [21, 22, 23, 24, 25] (Fig. 1 b). Assembling a set of standardized downstream tasks is paramount to evaluating and comparing language models after the self-supervised training step. Motivated by the lack of such benchmark in genomics, we first compiled a collection of 18 datasets, based on five peer-reviewed research studies, into a common format to facilitate experimentation and increase reproducibility.

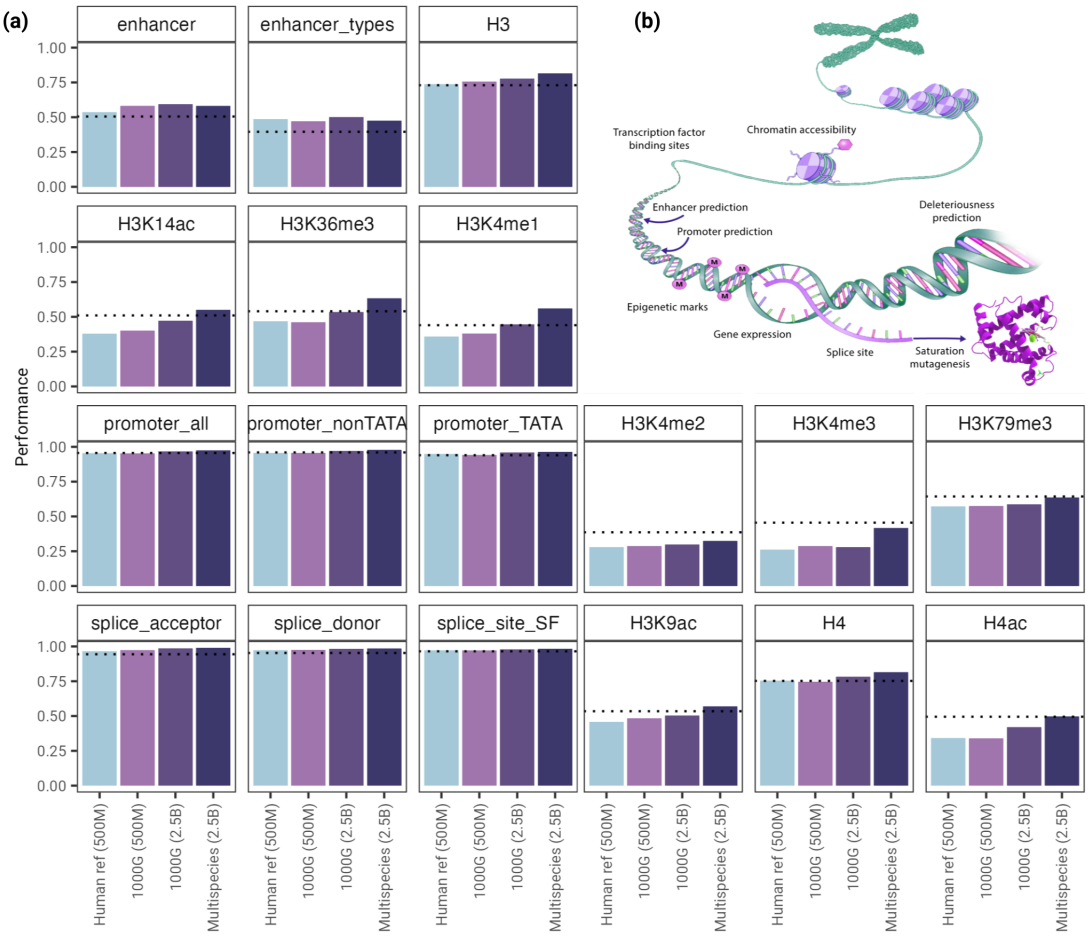


Figure 1: **2.5B Multispecies Nucleotide Transformer model beats 15 out of 18 downstream tasks using fine-tuning. a)** The downstream tasks evaluated in this study visually contextualized to the genome. **b)** Performance results on test sets across all downstream tasks for each Nucleotide Transformer model (from smallest 500M human reference genome, to largest 2.5B multi-species). Results refer to fine-tuning (probing results in Suppl. Fig. 2). To unify different metric choices reported in baselines, the term *performance* stands in for MCC, F1-score, or accuracy in order of preference and availability (details in Table 5).

The Nucleotide Transformer models were evaluated after self-supervised training through two different techniques: probing and fine-tuning. Probing refers to the use of learned LM embeddings of DNA sequences as features to simpler models for predicting genomic tasks. Specifically, we probed ten arbitrarily chosen layers of the LMs using either a logistic regression model or a small multi-layer perceptron (MLP) model composed of up to two hidden layers (Fig. 2). In agreement with recent work [26], we observe that the best performance for a given task is both model and layer dependent (Table 9). Moreover, we confirm that the best model performance is never obtained using embeddings from the final layer, as usually done 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 representation quality varies significantly across the layers of the transformer model (Suppl. Fig. 1).

Using probing, our models outperformed or matched 11 out of 18 baselines (Table 4). This number increases to 15 out of 18 after fine-tuning (Fig. 1). Even though fine-tuning was not extensively studied in previous work [5], possibly due to its high compute requirements, we overcame this limitation by relying on a recent parameter efficient fine-tuning technique [27] requiring only 0.1% of the total model parameters to be tuned. This technique allowed for faster fine-tuning on a single GPU, in addition to reducing the storage needs by 1000-fold over all fine-tuning parameters, and increasing performance. Coincidentally, while using simple downstream models on embeddings might appear faster and less compute-intensive, in practice we observed that rigorous probing was more compute-intensive compared to fine-tuning, as the choice of the layer, downstream model, and hyperparameters strongly impacted the performance. Moreover, a smaller variance in performance was observed when fine-tuning.

As an example, when classifying weak and strong enhancers using sequences alone, the combination of the best LM and downstream model reached performance (metrics in Table 5) of 0.424 before, and 0.501 after fine-tuning, thus outperforming the previous state-of-the-art baseline of 0.395 using LSTM-CNN [22] (Table 4). On promoter detection, we report performance of 0.966 and 0.974 before and after fine-tuning respectively, compared to 0.956 for the baseline [24]. For splice site detection, performance increased from 0.762 to 0.983 when fine-tuning compared to the baseline performance of 0.965 achieved by SpliceFinder [23] (Table 4). This task is also the most affected by the downstream model type, with performance increasing from 0.590 to 0.762 when considering a logistic regression as opposed to an MLP.

Parameter scaling and increasing data diversity improve performance

Motivated by trends in natural language processing (NLP), where increasing the volume and diversity of the datasets, as well as the size of the transformer model yields better results [28], we trained large models from 500M up to 2.5B parameters on datasets of increasing size. We first constructed a pre-training dataset with sequences from the Human reference genome to establish our baseline LM (see Methods). We prepared additional training sets with sequences from 3,200 genetically diverse human genomes [15], spanning 27 populations across the world (Table 2), which increased the data size but also included 98% redundancy amongst samples. Finally, we developed a custom multispecies dataset composed of reference sequences of 850 species (Fig. 2a; Table 6; see Methods). We observed that scaling the model size and adding diversity to the dataset sequences increases the average performances of the downstream tasks (Fig. 2c).

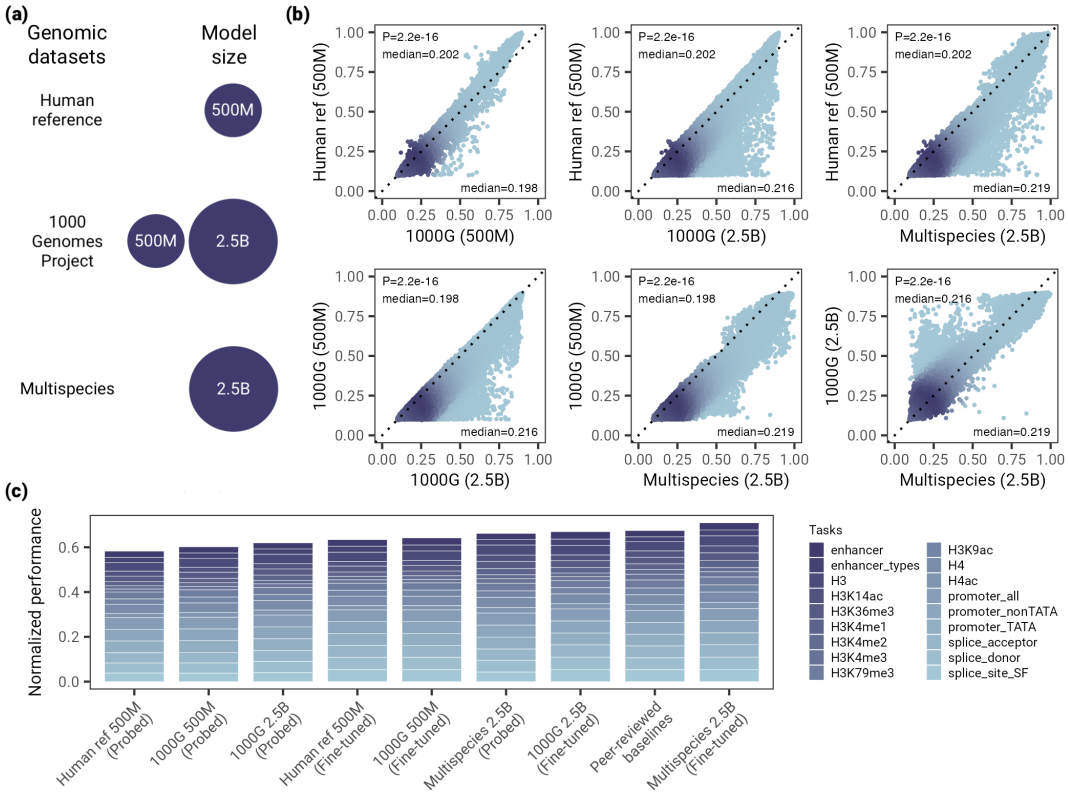


Figure 2: **Increasing model size and data diversity improves downstream performance and increases token reconstruction accuracy.** **a)** Visual representation of models parameter sizes across datasets (larger circles indicate larger models). **b)** Pairwise comparison of token reconstruction accuracy across models. P-values refer to a two-sided Wilcoxon signed rank test between models. Median values for the two models compared are shown. **c)** Normalized sum of performance across 18 downstream tasks for all Nucleotide Transformer models after probing and fine-tuning. Normalized sum is obtained by summing performances over tasks for each model and dividing by the number of tasks.

As a benchmark, the 500M parameter model trained on the Human reference genome achieved a compound performance (in absolute best values) of 0.634 across tasks, whereas a model with the same number of parameters trained on the 1000G dataset achieved a performance of 0.641. When increasing the size of the model from 500M to 2.5B parameters on the 1000G dataset, compound performance increased to 0.669. While adding intra-species diversity slightly helped the overall performance on the designated tasks, adding inter-species diversity at a fixed model scale yielded better results. Indeed, the 2.5B model on the multispecies dataset yields a compound performance of 0.709. Only this combination of largest model size (2.5B), inter-species diversity (multispecies dataset) and fine-tuning resulted in the compound performance of 0.709 surpassing baselines at 0.674 (Fig. 2c).

To further investigate the benefits of increasing the number of parameters of the models and genomic diversity of the datasets, 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. 2b). We observed that the reconstruction accuracy of the 500M Human reference model showed a higher median accuracy than the 500M 1000G model (median=0.202 versus 0.198, P<2.2e-16, two-sided Wilcoxon rank sum test); however, it was lower than the accuracy obtained by the 2.5B 1000G model (median=0.216) and the 2.5B Multispecies model (median=0.219), which illustrates the impact of jointly increasing the model size and diversity of the dataset. Interestingly, while the 2.5B Multispecies model showed the best reconstruction accuracy overall, a number of sequences showed much higher reconstruction accuracy for the 2.5B 1000G model, which likely points to particular characteristics of human sequences that were better learned by this model.

The Nucleotide Transformer models learnt to detect known genomic elements and human genetic variation

To gain insights into the sequence elements that the nucleotide transformer is utilizing when making predictions, we explored different aspects of the transformer language model architecture. First, we evaluated the ability of the models to reconstruct tokens containing natural variation present in human populations based on their haplotypic background (Fig. 3a). We measure the effectiveness of sequence reconstruction by recording the model perplexity scores over single nucleotide polymorphisms (SNPs) occurring at 1%, 5%, 10% and 50% frequencies. To evaluate 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). SNP tokens are centered in sample-specific 6kb sequences. The perplexity is computed at the token position after masking it. A lower perplexity value is indicative of more accurate sequence reconstruction. Across all populations and SNPs frequencies, we consistently observe that the 2.5B parameter models, with median perplexity ranging from 48.4±4.1 (2SD) to 95.4±7.2, outperform their 500M counterparts, for which the perplexity values fluctuate between 69.1±5.2 and 118.2±8.2. These results are in line with the observed reconstruction accuracies over human genome sequences, and confirm the impact of model size on performance.

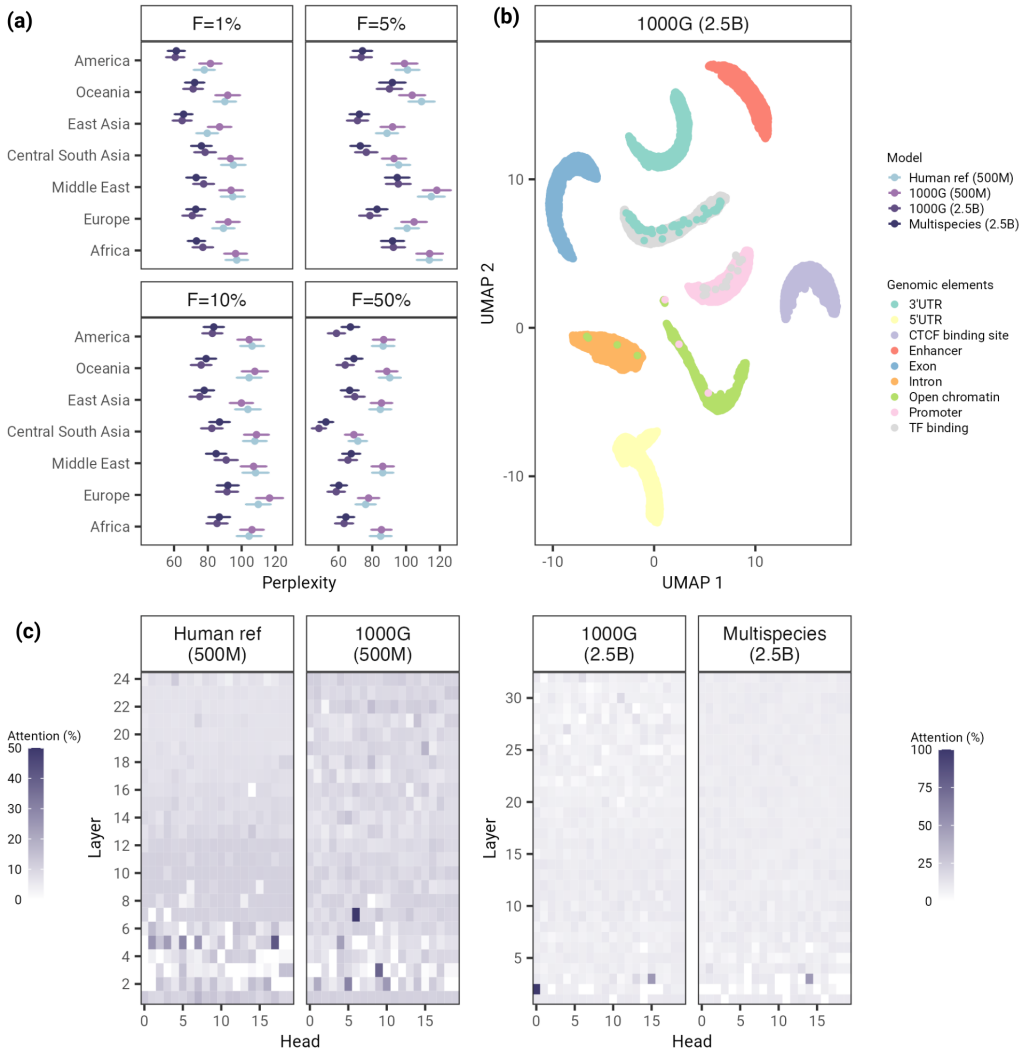


Figure 3: **The Nucleotide Transformer models acquired knowledge about genomic elements and human genetic variation. a)** Reconstruction perplexity of the Nucleotide Transformer models on SNPs at 1%, 5%, 10% and 50% frequency across worldwide human populations. **b)** U-MAP projections of embeddings of nine regulatory elements from layer four of the 2.5B model trained on the 1000G dataset. **c)** Attention percentages per head and layer for all Nucleotide Transformer models computed on enhancers.

Next, we sought to determine whether the model embeddings encode representations of the different genomic sequence features. We considered a total of 90,000 sequences annotated by Ensembl [30] as: "5' UTR", "3' UTR", "exon", "intron", "enhancer", "promoter", "CTCF binding site", "open chromatin", and "transcription factor binding sites". For each sequence category, we examined 10,000 sequences of length 6kb, where the element is randomly positioned in the sequence. We computed embeddings at three arbitrary layers for all Nucleotide Transformer models and averaged them only over the genomic element indexes. The resulting data was projected through Uniform Manifold Approximation and Projection (UMAP) to lower dimensions. We report the visualization of the embeddings obtained at one layer over two UMAP components of the 2.5B parameters model trained over the 1000G dataset (Fig. 3b). We observe that the "enhancers", "5' UTR regions", "CTCF binding sites", and "exons"sequence clusters are perfectly separated from the other genomic features. Interestingly, sequences corresponding to "open chromatin"and "introns"features show limited overlap, while "promoters", "TF binding sites"and "3' UTR regions"exhibit substantial co-clustering, probably owing to the versatility of sequence function in regulatory portions of the genome. We observed similar trends for all models, but not in all layers (Suppl. Fig. 13 and Suppl. Fig. 14), confirming the layers' specialization, as well as the models' ability to capture genomic patterns during self-supervised training.

Finally, to understand whether the model encodes for genomic features, we computed the attention percentages for all heads and layers of the Nucleotide Transformer models over the same nine genomic features, following the methodology introduced in previous work [31]. An attention head is thought to recognize some specific elements if its attention percentage is significantly greater than the naturally occurring frequency of the element in the pre-training dataset (Suppl. Fig. 12). We performed two proportions z-test to determine whether any of these elements were detected due to chance across models and heads, and corrected for multiple testing using Bonferroni. Of 480 heads for the 500M models, with a significance level of 0.05, we observed that 27 and 24 heads specialized in the detection of enhancers for the 500M Human reference and 1000G models. In the case of the larger 2.5B models with 640 heads, 28 and 26 detected enhancers for either the 1000G and multispecies datasets. Interestingly, maximum attention percentages over enhancers were higher for the larger models, with values reaching up to almost 100% attention for the 2.5B 1000G model (Fig. 3c). Similar results were observed for all nine genomic sequence annotations (Suppl. Fig. 3 -11), with a maximum number of 117 out of 640 heads detecting introns for the 2.5B Multispecies model. Altogether, these results demonstrate that DNA sequences are adequately recapitulated by language models, both at the scale of large genomic regions and individual variants in diverse populations. Remarkably, custom representations can be obtained for any region of the genome with mutations which could provide substantial improvement in assessing the pathogenicity of mutations in their haplotypic context.

The Nucleotide Transformer embeddings predict the impact of mutations

Finally, we assessed the ability of our transformer models to differentiate between different types of genetic variants, and to prioritise those of functional importance. In order to do so, we tested divergence between alternative allele and reference embeddings for mutations of interest using four metrics (see Methods; dubbed "divergence score").

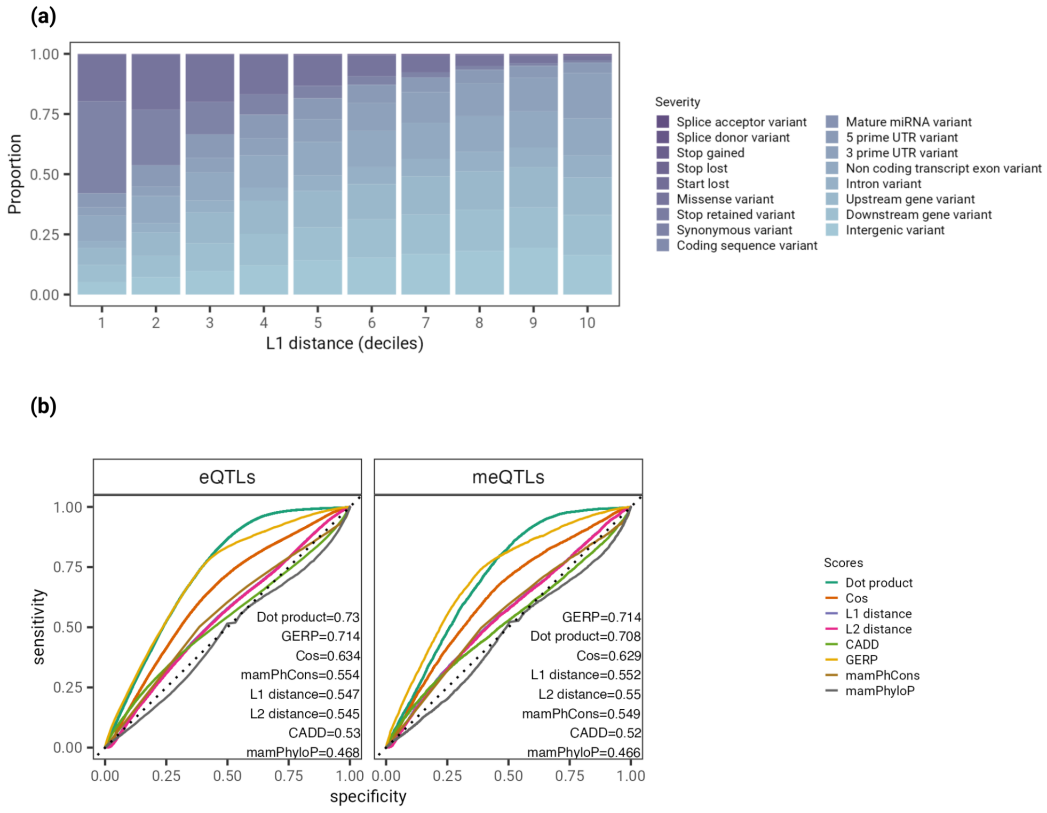


Figure 4: **Prioritizing functional genetic variants based solely on DNA sequence information. a)** Proportion of variant consequence terms across deciles based on the L1 distance metric for the Multispecies (2.5B) model. The consequence terms are shown in order of severity (more severe to less severe) as estimated by Ensembl. **b)** Comparison of different distance metrics for the Multispecies (2.5B) model to other methods for prioritizing functional variants based on GRASP eQTLs and meQTLs, against 1000 Genomes Project common SNPs. Model performance is measured with the area under the receiver operating characteristic curve (AUC).

We first considered all segregating 1000 Genomes Project SNPs from chromosome 22 and investigated the distribution of these embedding-based divergence scores across 17 variant categories (e.g. stop gained, missense, intergenic). For the four models, the scores among the different categories were significantly different (P-value = 2.22e−16, K-S rank sum test). The Multispecies (2.5B) model using the L1 (Manhattan) distance showed the strongest ability to differentiate between categories (Fig. 4a). Notably, scores computed from this model at coding variants, i.e. those with a potential functional impact on protein function, showed significantly lower scores compared to intergenic variants (median of 675 and 622 for missense and synonymous variants, respectively, versus median of 780 for intergenic variants, P-value = 2.22e−16, two-sided Wilcoxon rank sum test). This illustrates how embedding-based distances alone encodes information about variant functional categories. Nonetheless, we also found that other types of variants, including those with high potential disruption on protein, such as stop gain mutations, showed similar scores compared to intergenic variants (P-value >0.05, two-sided Wilcoxon rank sum test), which suggests that this score is not necessarily associated with mutation severity.

Lastly, we investigated the ability of the divergence scores to prioritize functional variants, and compared these against scores that measure levels of genomic conservation, as well as a score that leverages both conservation and genomic functional features (including genic effects, regulatory element annotations, among others). Specifically, we assessed the ability of the divergence scores to classify genetic variants exerting regulatory effects on gene expression (i.e., expression quantitative trait loci [eQTLs]) and genetic variants associated with DNA methylation variation (i.e., methylation quantitative trait loci [meQTLs]) (Fig. 4b). As the scores computed from the Multispecies (2.5B) model showed the best ability to differentiate variant categories, we relied on divergence scores only from this model. Notably, the performance of divergence scores surpassed that of previous scores in prioritizing eQTLs (dot score product, AUC=0.73) and achieved an AUC close to the best-performing score in prioritizing meQTLs (dot score product, AUC=0.708 versus GERP score, AUC=0.714). Overall, these results illustrate how divergence scores, extracted solely from embeddings of DNA sequences, can help reveal and contribute to understanding the potential biological consequences of variants associated with a disease or phenotype.

Discussion

Evolution informs representations: models trained on genomes from different species top charts. We explored the impact of different datasets to pre-train equally sized transformer models. 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 driving accuracy across tasks (Fig. 2). Notably, the models trained on genomes coming from different species perform well on categorical human genomics downstream tasks, as well as on human variant prediction, even when compared to models trained exclusively on the human genome (Fig. 2). This could indicate that the genome LMs capture a signal of evolution so fundamental across species that it better generalises to shared functions.

Model scale drives performance. The Nucleotide Transformer models trained ranged from 500 million up to 2.5 billion parameters, which is five times larger than DNABert [18] and ten times larger than the Enformer [14]. As has been the case in NLP [28], results in the genomic space confirmed that increasing model size yields better performance. Training the largest parameter model 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, benefiting the research community as a whole.

Diverse datasets, large models and fine-tuning: best results require it all. Previous work in biological LM development evaluated downstream performance exclusively probing the last transformer layer [5], most likely due to its perceived ease of use, relative good performance and low computational complexity. As in this study we sought to push the limits of downstream accuracy, we performed computationally expensive, rigorous probing of different transformer layers, downstream models and

hyperparameter sweeps. We observed best probing performance for intermediate transformer layers (Suppl. Fig. 2), aligning with recent work in computational biology [26] and common practice in NLP [32], suggesting that even for existing biological-related LMs new highs could be reached. Through probing, the 2.5B multispecies Nucleotide Transformer model achieved better performance for 8 out of 18 tasks compared to the baseline. We additionally explored a recent downstream fine-tuning technique that introduces a small number of trainable weights in the transformer for weight-efficient fine-tuning (IA^3 [27]). Compared to the extensive probing exercise, this technique yielded better results using fewer compute resources and rose the number of tasks outperforming or matching baselines to 15, confirming that downstream model engineering can yield performance improvements [33]. The usage of this technique makes fine-tuning competitive with probing from an operational perspective, both for training and inference.

Genomics insights are captured during training despite no supervision. The Nucleotide Transformer models learned insights about key regulatory genomic elements through attention, as demonstrated through the analysis of attention maps, embedding spaces, and probability distributions. Elements such as enhancers and promoters were detected by all models, and at several heads and layers. We also observed that each model contained at least one layer that produced embeddings which 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 unravel new elements and effects in the future. While our models have an attention span limited to 6kbp, the recently developed Enformer model [14] suggested that increasing the perception field up to 200kbp is necessary to capture long-range dependencies, which are needed to accurately solve tasks such as gene expression prediction. 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 modelling. 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 [34, 35] could be investigated to overcome compute complexity limitations.

Acknowledgments

We thank members of the Rostlab, particularily Tobias Olenyi, Ivan Koludarov and Burkhard Rost for constructive discussions that helped identify interesting research directions. Furthermore, we extend gratitude to all those who deposit experimental data in public databases, to those who maintain these databases, and those who make analytical and predictive methods freely available.

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

[16] S. Nayfach, S. Roux, R. Seshadri, D. Udwary, N. Varghese, F. Schulz, D. Wu, D. Paez-Espino, I.-M. Chen, M. Huntemann, *et al.*, "A genomic catalog of earth's microbiomes," *Nature biotechnology*, vol. 39, no. 4, pp. 499–509, 2021.

[17] P. J. Turnbaugh, R. E. Ley, M. Hamady, C. M. Fraser-Liggett, R. Knight, and J. I. Gordon, "The human microbiome project," *Nature*, vol. 449, no. 7164, pp. 804–810, 2007.

[18] 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.

[19] 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.

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

[21] 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.

[22] 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.

[23] 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.

[24] 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.

[25] 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.

[26] 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?,"

[27] 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.

[28] 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.

[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] B. L. Aken, S. Ayling, D. Barrell, L. Clarke, V. Curwen, S. Fairley, J. Fernandez Banet, K. Billis, C. García Girón, T. Hourlier, *et al.*, "The ensembl gene annotation system," *Database*, vol. 2016, 2016.

[31] 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.

[32] 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.

[33] 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.

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

[35] 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.

[36] 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.

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

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

[39] 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.

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

[41] 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.

[42] 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.

[43] 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.

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 [36]. 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 [37]. 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 3. 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 [38] with a learning rate schedule, and standard values for exponential decay rates and epsilon constants, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and ϵ=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 [39] 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 [40] 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 is reported in Table 9. 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 [27]. 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 γ 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.

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 [27]. 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 [41], 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 6, 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 7).

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 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

[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/

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 [21]. Nucleosome occupancy values in these ten datasets were obtained with Chip-Chip experiments [42] 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 [24], 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 [22] 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 [22] showed that training performance was increased by augmenting the dataset with 6000 synthetic enhancers and 6000 synthetic non-enhancers produced through a generative model.

[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

A.4.4 Splice site prediction

We used two datasets to evaluate splice site prediction. First, we downloaded the dataset used by SpliceFinder [23], 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 [25]. 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 Metrics

Due to discrepancies in metric choices by baselines methods considered, different metrics were used to compare downstream results for different tasks (Table 5). In general, MCC was preferred over F1-score, which was preferred over accuracy. Thus, the term *performance* will be used as an umbrella term throughout the text to refer to either MCC, F1-score, or to accuracy as listed in Table 5.

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 θ. Within a nucleotide sequence $\mathbf{s}$ of interest, we masked tokens using one of two strategies (i.e. we replaced the tokens at these positions by 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 positions retrieved. The loss function $l(\theta, \mathbf{s})$ and the accuracy $acc(\theta, \mathbf{s})$ are defined as follows:

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

where $\mathcal{P}_{\text{masked}}$ is the set of masked positions and $\mathcal{V}$ 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 variant consequence annotations we used the Variant Effect Prediction (VEP) software on all bi-allelic single nucleotide polymorphisms (SNPs) present on chromosome 22 from the 1000 Genomes dataset. After filtering out genetic variants that were annotated to more than one consequence (e.g. those annotated as stop gained and splice site variants), we obtained a final dataset of 72,788 genetic variants across 17 consequence categories.

As a positive set of functional genetic variants 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 [43] with a P-value

[7] https://git.unistra.fr/nscalzitti/spliceator/-/tree/master/Data/Datasets

$<10^{-12}$. For each of these two 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 GRASP dataset, and that were within 100kb of the associated variants. After these filters, we retained a total of 82,489 and 11,910 genetic variants for the eQTLs and meQTLs SNP datasets, respectively.

To predict the potential effect of genetic variants using our models, for each SNP, we obtained a 6kbp sequence centered on the SNP of interest based on the Human reference genome. We created two sequences, one carrying the reference allele and a second carrying the alternative allele at the SNP position. We then computed four different similarity scores to capture different aspects of the vector distances in the embedding space based on: (i) the L1 distance (Manhattan), (ii) the L2 distance (Euclidean), (iii) the cosine similarity as well as (iv) the dot-product (not normalized cosine similarity) between both embedding vectors.

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 [31]. For a genomic element, we define the indicator function over tokens $f(i)$ that equals 1 if one or several nucleotide 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 μ is a confidence threshold.

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 6000kbp extracted from the Human reference genome. The average proportion of tokens belonging to each element can be found in Table 8. For each sequence, the position of the feature within the sequence was sampled uniformly during the dataset creation. As suggested in previous work [31], 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 8). 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 bonferonni correction to the significance level, α, of 0.05. We computed z-scores and associated p-value for each head in every model for every genomic element as follow:

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

where $\hat{p}_1$ represents the proportion of attention above μ 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 μ, 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.

B Tables

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**

Table 1: Normalized summed downstream performance. These values are obtained through summing performances over all tasks for each model and dividing by the number of tasks.

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

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

	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

Table 3: Models hyperparameters.

	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_SF	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_nonTATA	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

Table 4: Downstream performance per task for all models and baselines. For probed models, the reported performance corresponds to the best layer and bets downstream model.

	Num train sequences	Num test sequences	Max sequence length in bp	Reported Metric	Baseline Name
H3K4me3	25953	2884	500	MCC	SVM [21]
H3K4me2	27614	3069	500	MCC	SVM [21]
H3K36me3	31392	3488	500	MCC	SVM [21]
H3K9ac	25003	2779	500	MCC	SVM [21]
splice_donor	19775	2198	600	F1	SpliceFinder [23]
splice_site_SF	27000	3000	400	Acc	SpliceFinder [23]
H4ac	30685	3410	500	MCC	SVM [21]
H3K4me1	28509	3168	500	MCC	SVM [21]
enhancer	14968	400	200	MCC	LSTM-CNN [22]
enhancer_types	14968	400	200	MCC	LSTM-CNN [22]
H4	13140	1461	500	MCC	SVM [21]
splice_acceptor	19961	2218	600	F1	SpliceFinder [23]
H3K79me3	25953	2884	500	MCC	SVM [21]
promoter_nonTATA	47767	5299	300	F1	DeePromoter [24]
promoter_all	53276	5920	300	F1	DeePromoter [24]
H3K14ac	29743	3305	500	MCC	SVM [21]
H3	13468	1497	500	MCC	SVM [21]
promoter_TATA	5509	621	300	F1	DeePromoter [24]

Table 5: Downstream Tasks Meta-Data.

Class	Number of species	Number of nucleotides (B)
Bacteria	667	17.1
Fungi	45	2.3
Invertebrate	39	20.8
Protozoa	10	0.5
Mammalian Vertebrate	31	69.8
Other Vertebrate	57	63.4

Table 6: Distribution of the genomes present 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

Table 7: Model organisms selected to be included in the multispecies dataset.

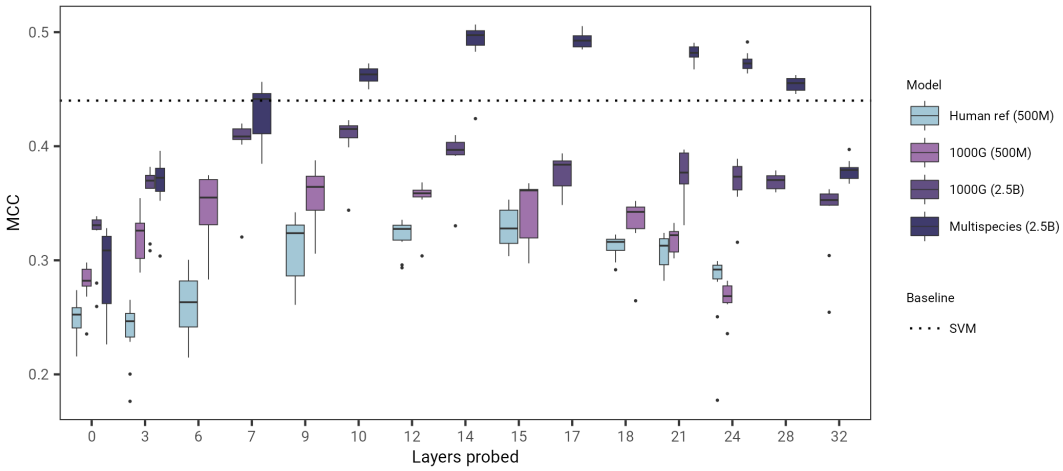
	Percentage of sequences containing this element	Mean Percentage of Sequence Length belonging to this element
Enhancer	10.18%	11.09%
Open_chromatin	11.32%	6.16%
3' UTR	11.37%	4.30%
TF_binding	11.28%	6.38%
Intron	10.45%	11.08%
Exon	11.87%	3.18%
5'UTR	11.84%	1.99%
Promoter	10.95%	9.78%
CTCF Binding Site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

Table 8: Genomic element percentage of considered dataset and length's percentage of input sequence (6kbp). This corresponds to the dataset used for attention maps and embedding spaces visualization experiments.

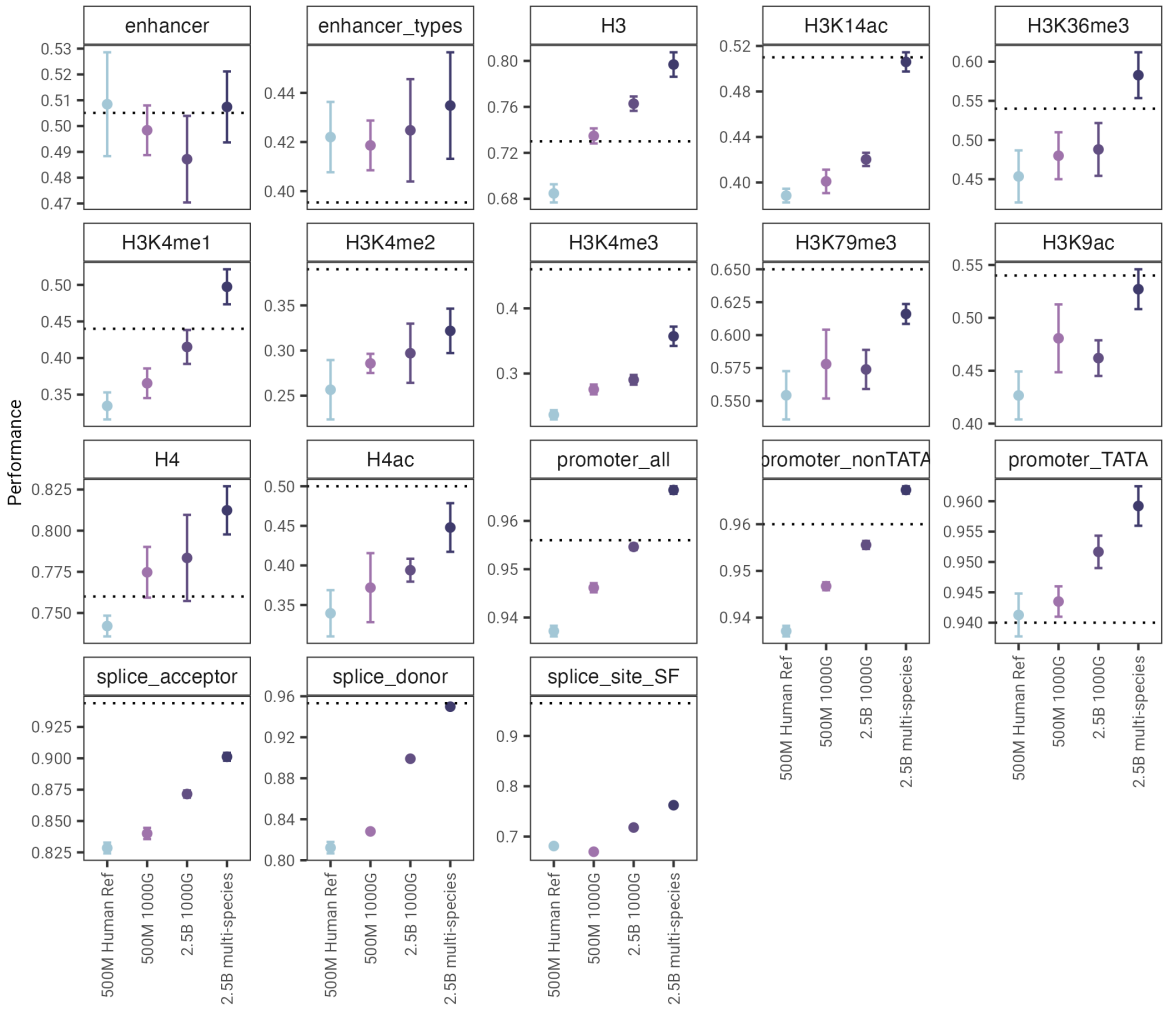
	Model	Layer probed	Number of hidden layers	Hidden layer dimension	Batch size	Learning rate
H3K4me3	2.5B MS	17	1	256	500	1e-4
H3K4me2	2.5B MS	10	1	256	500	1e-4
H3K36me3	2.5B MS	17	1	256	500	1e-4
H3K9ac	2.5B MS	10	1	256	500	1e-4
splice_site_SF	2.5B MS	14	1	512	650	6.1e-5
H4ac	2.5B MS	17	1	256	500	1e-4
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
polyA_human	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

Table 9: Hyperparameters used to train the model with the best cross-validation performance for each of the downstream tasks, 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 hyper-parameters.

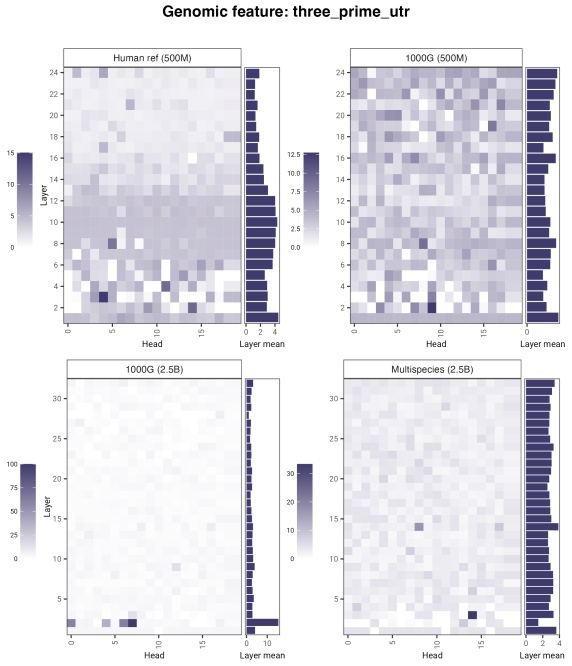
C Supplementary Figures



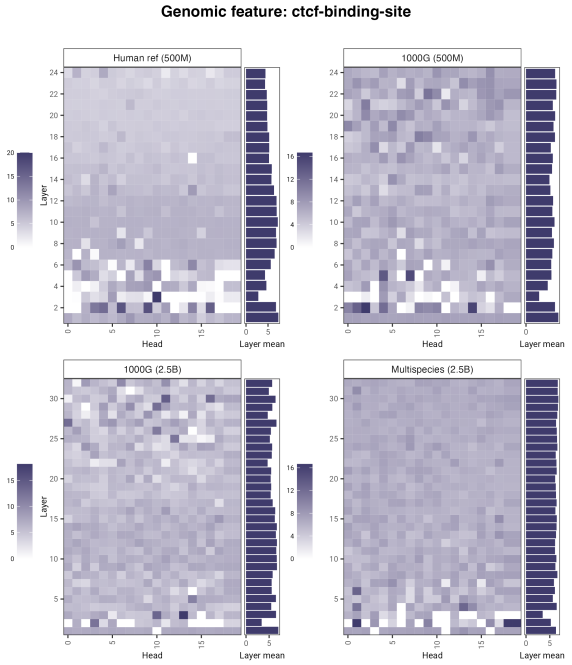
Supplementary Figure 1: For each pre-trained model, the embeddings are extracted from 10 layers of the transformer when taking the H3K4me1 dataset as input. We tuned the hyperparameters of an MLP on the train set with 10 fold cross-validation and selected the 10 models corresponding to the best performing set of hyperparameters to evaluate them on the test set. Their performances are displayed by each whisker box. Median performance on the test dataset varies significantly across the layers as, when using the 2.5B Multispecies model, embeddings produced by the layer 14 produce a median MCC score of 0.50 against 0.38 for the final layer, a 30% increase in performance.



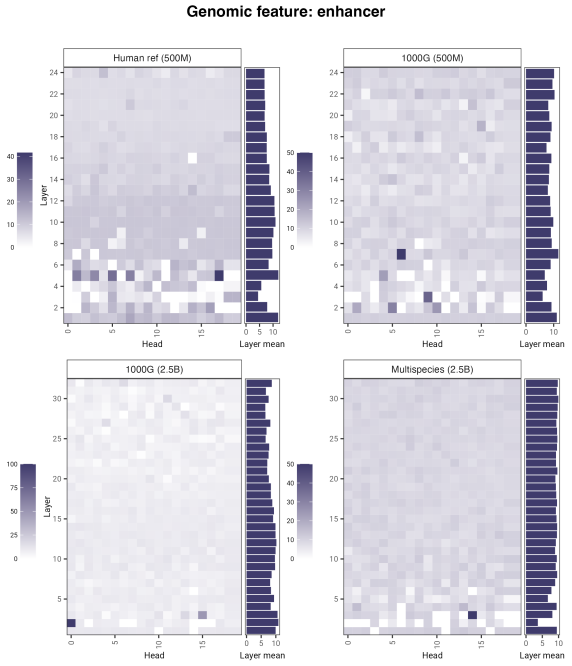
Supplementary Figure 2: Performance results on test sets across all downstream tasks for each Nucleotide Transformer model (from smallest 500M Human reference genome, to largest 2.5B Multi-species) during probing. Best layers and best downstream models are used for each model and task. The error bars represent 1 std for the 10-folds validation.



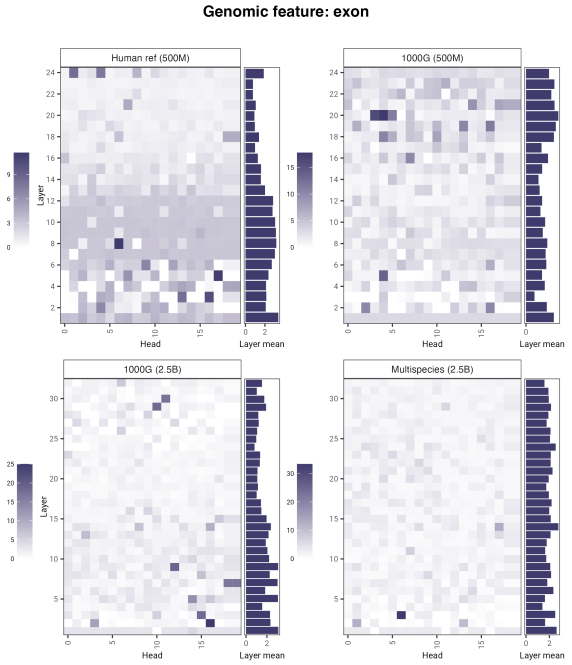
Supplementary Figure 3: Nucleotide Transformer models attention percentages per head computed for the 3' UTR regions.



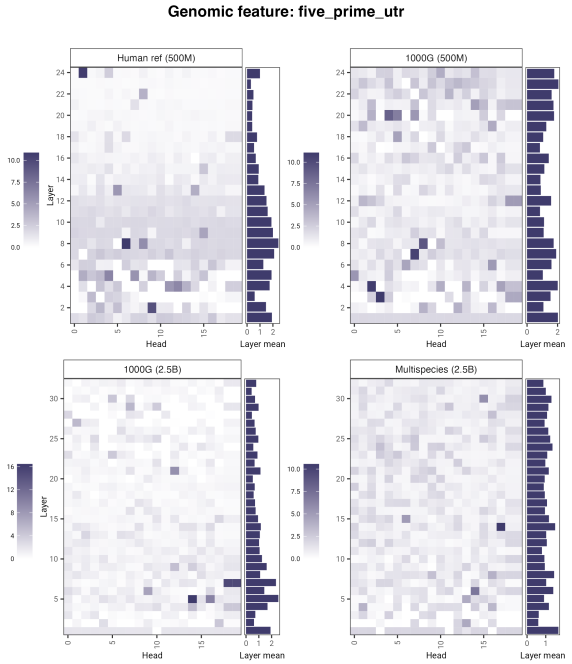
Supplementary Figure 4: Nucleotide Transformer models attention percentages per head computed for CTCF binding sites.



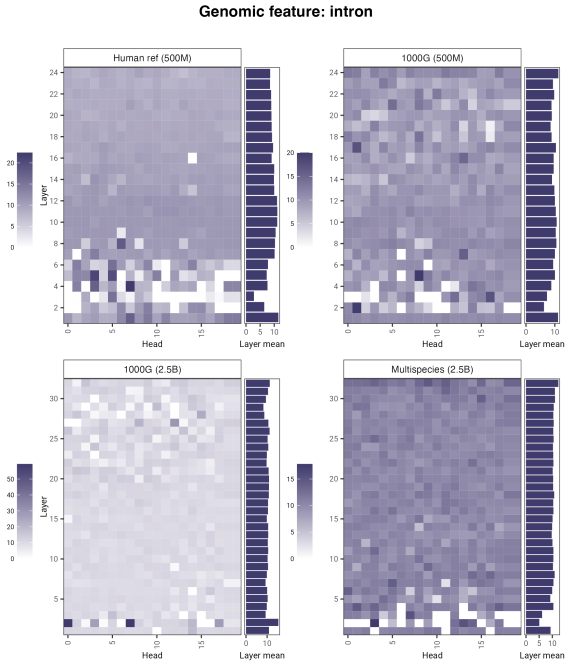
Supplementary Figure 5: Nucleotide Transformer models attention percentages per head computed for the enhancers.



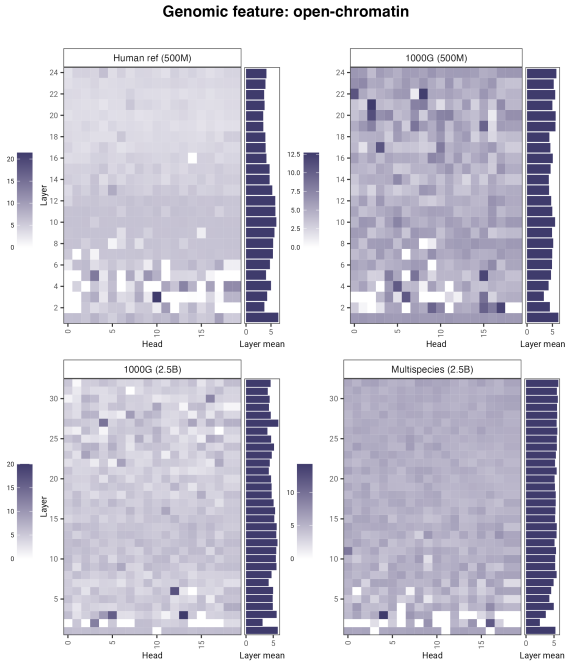
Supplementary Figure 6: Nucleotide Transformer models attention percentages per head computed for the exons.



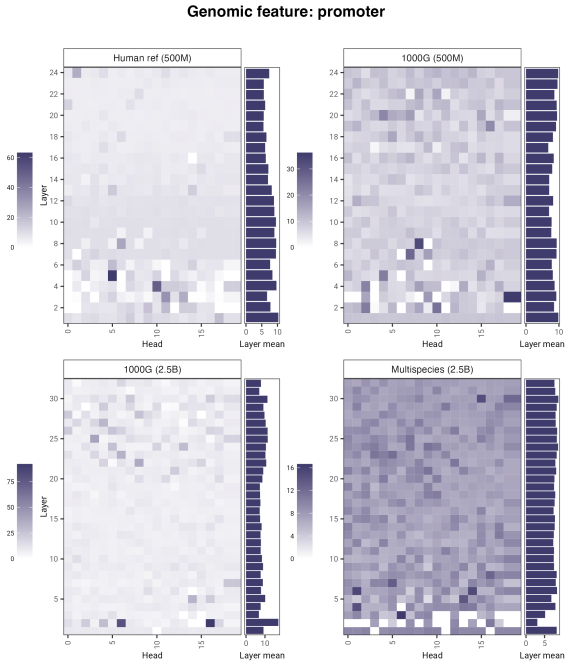
Supplementary Figure 7: Nucleotide Transformer models attention percentages per head computed for the 5' UTR regions.



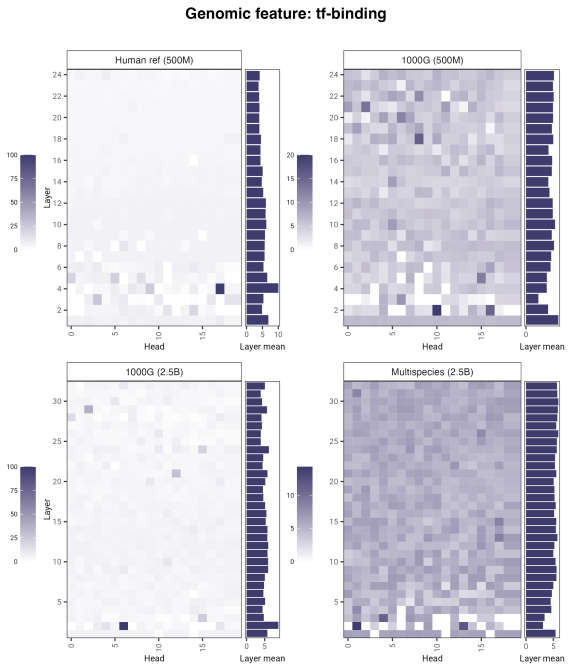
Supplementary Figure 8: Nucleotide Transformer models attention percentages per head computed for the introns.



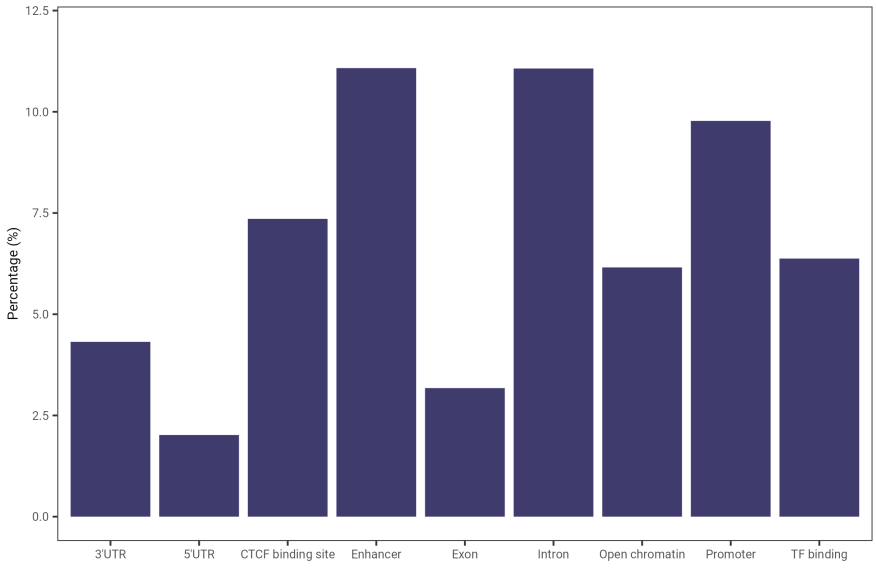
Supplementary Figure 9: Nucleotide Transformer models attention percentages per head computed for the open chromatin.



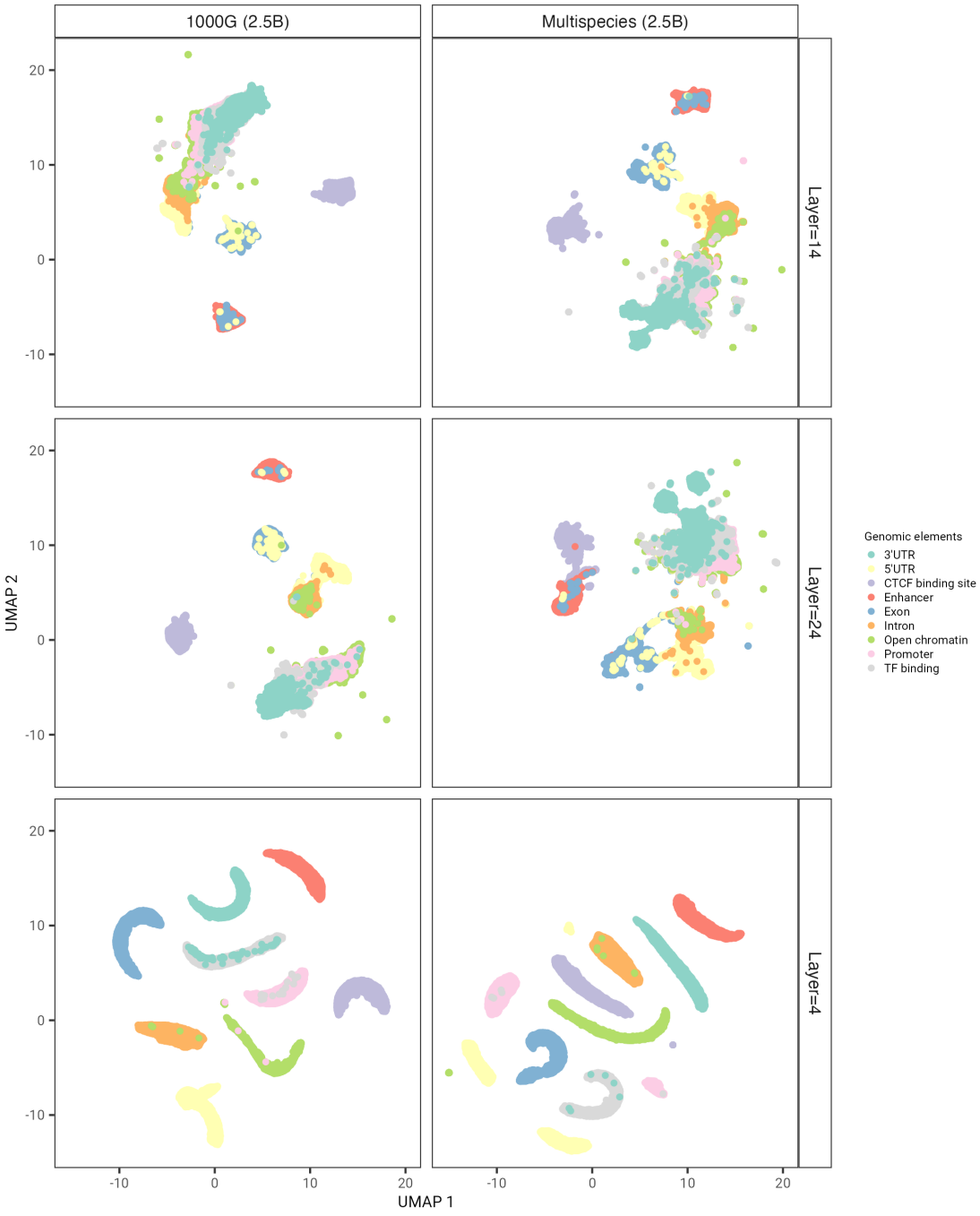
Supplementary Figure 10: Nucleotide Transformer models attention percentages per head computed for the promoters.



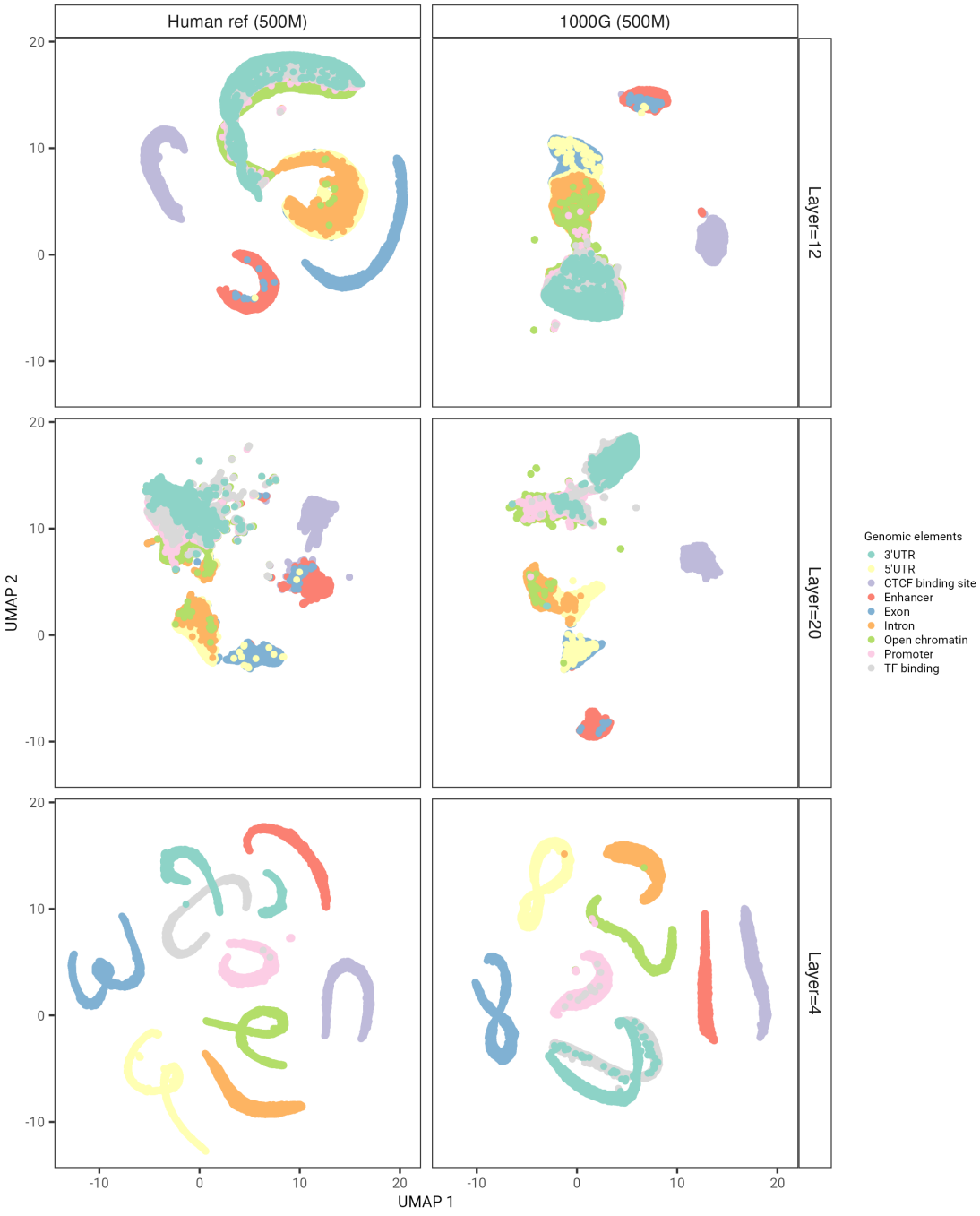
Supplementary Figure 11: Nucleotide Transformer models attention percentages per head computed for the transcription factor binding sites.



Supplementary Figure 12: 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 8.



Supplementary Figure 13: U-MAP projections of embeddings of nine regulatory elements from layers 4, 14 and 24 for the two Nucleotide Transformer 2.5B parameters models.



Supplementary Figure 14: U-MAP projections of embeddings of nine regulatory elements from layers 4, 14 and 24 for the two Nucleotide Transformer 500M parameters models.