

Nucleotide Transformer: building and evaluating robust foundation models for human genomics

Received: 1 March 2023

Accepted: 19 October 2024

Published online: 28 November 2024

 Check for updates

Hugo Dalla-Torre¹, Liam Gonzalez¹, Javier Mendoza-Revilla¹, Nicolas Lopez Carranza¹, Adam Henryk Grzywaczewski², Francesco Oteri¹, Christian Dallago^{2,3}, Evan Trop¹, Bernardo P. de Almeida¹, Hassan Sirelkhatim², Guillaume Richard¹, Marcin Skwark¹, Karim Beguir¹, Marie Lopez¹✉ & Thomas Pierrot¹✉

The prediction of molecular phenotypes from DNA sequences remains a longstanding challenge in genomics, often driven by limited annotated data and the inability to transfer learnings between tasks. Here, we present an extensive study of foundation models pre-trained on DNA sequences, named Nucleotide Transformer, ranging from 50 million up to 2.5 billion parameters and integrating information from 3,202 human genomes and 850 genomes from diverse species. These transformer models yield context-specific representations of nucleotide sequences, which allow for accurate predictions even in low-data settings. We show that the developed models can be fine-tuned at low cost to solve a variety of genomics applications. Despite no supervision, the models learned to focus attention on key genomic elements and can be used to improve the prioritization of genetic variants. The training and application of foundational models in genomics provides a widely applicable approach for accurate molecular phenotype prediction from DNA sequence.

Foundation models in artificial intelligence (AI) are characterized by their large-scale nature, incorporating millions of parameters trained on extensive datasets. These models can be adapted for a wide range of subsequent predictive tasks and have profoundly transformed the AI field. Notable examples in natural language processing (NLP) include the so-called language models (LMs) BERT¹ and GPT². LMs have gained significant popularity in recent years owing to their ability to be trained on unlabeled data, creating general-purpose representations capable of solving downstream tasks. One way they achieve a comprehensive understanding of language is by solving billions of cloze tests, in which they predict the correct word to fill in the blank in a given sentence. This approach is known as masked language modeling¹. Early instances of foundation models applying this objective to biology involved training LMs on protein sequences, where they were tasked with predicting

masked amino acids in large protein sequence datasets^{3–5}. These protein LMs, when applied to downstream tasks using transfer learning, demonstrated the ability to compete with and even outperform previous methods for tasks such as predicting protein structure^{3,4} and function^{6,7}, even in data-scarce regimens⁸.

Beyond protein sequences, the dependency patterns encoded in DNA sequences play a fundamental role in understanding genomic processes, from characterizing regulatory regions to assessing the impact of individual variants within their haplotypic context. In this context, specialized deep-learning (DL) models have been trained to uncover meaningful patterns of DNA^{9–12}. For example, DL models have been used to predict gene expression from DNA sequences^{13–18}, with recent advancements combining convolutional neural networks and transformer architectures enabling the encoding of regulatory

¹InstaDeep, London, UK. ²Nvidia, Santa Clara, CA, USA. ³Technical University of Munich, Munich, Germany. ✉e-mail: m.lopez@instadeep.com; t.pierrot@instadeep.com

elements located up to 100 kilobases (kb) upstream¹⁹. The abundance of data generated by modern genomics research presents both an opportunity and a challenge. On one hand, intricate patterns of natural variability across species and populations are readily available; on the other hand, powerful DL methods capable of handling large-scale data are necessary for accurate signal extraction from unlabeled datasets. Large foundation models trained on nucleotide sequences seem to be an approach worth exploring to addressing this challenge^{20–26}.

Here, we construct robust foundation models to encode genomic sequences, named the Nucleotide Transformer (NT), and present a systematic study and benchmark to evaluate their performances. We initiated our study by building four distinct LMs of varying sizes, ranging from 500 million up to 2.5 billion parameters. These models were pre-trained on three different datasets, including the human reference genome, a collection of 3,202 diverse human genomes, and 850 genomes from various species. After training, we leveraged the representations (embeddings) from each of these models in two ways. To evaluate NT's stability in performance while adapting to various tasks, we trained each model on a diverse set of 18 genomic curated prediction tasks and compared them with three alternative DNA foundational models, alongside one state-of-the-art nonfoundational model, using a systematic tenfold cross-validation procedure. Moreover, to broaden our evaluation, we compared our best-performing model and three state-of-the-art supervised baseline models that have been optimized for the specific tasks at hand. To decipher the sequence features learned during pre-training, we explored the models' attention maps and perplexities, and conducted data dimensionality reduction on their embeddings. Additionally, we evaluated the embeddings' capacity to model the impact of functionally important genetic variants in humans, through zero-shot-based scores. Expanding upon the findings from the initial set of experiments, we developed a second set of four LMs, with decreasing sizes from 500 million to 50 million parameters, to investigate the scaling laws of such models. We successfully constructed a model that achieves the performance of our previous best model with only one-tenth the number of parameters and while doubling the perception field size.

Results

The Nucleotide Transformer model accurately predicts genomics tasks

We developed a collection of transformer-based DNA LMs, NT, which have learned general nucleotide sequence representations from 6-kb unannotated genomic data (Fig. 1a and Methods). Inspired by trends in NLP, where larger training datasets and model sizes have demonstrated improved performance²⁷, we constructed transformer models with varying parameter sizes and datasets: (1) a 500-million-parameter model trained on sequences extracted from the human reference genome ('Human ref 500M'); (2) a 500-million-parameter model ('1000G 500M') and (3) a 2.5-billion-parameter model ('1000G 2.5B') both trained on 3,202 genetically diverse human genomes²⁸; and (4) a 2.5-billion-parameter model, encompassing 850 species from diverse phyla ('Multispecies 2.5B'), including 11 model organisms (Fig. 1c and Supplementary Tables 1–4).

To evaluate the efficacy of these models in predicting varied molecular phenotypes, we curated 18 genomic datasets from publicly available resources encompassing splice site prediction tasks (GENCODE²⁹), promoter tasks (Eukaryotic Promoter Database³⁰) and histone modification and enhancer tasks (ENCODE^{31–33}), each designed to be of reasonable size to enable fast and rigorous cross-validation procedures (Fig. 1d, Supplementary Table 5 and Methods). While larger datasets for supervised models are available, this compilation of 18 genomic datasets offers a diverse and robust selection for scrutinizing the models' adaptability across diverse tasks in a statistically rigorous manner and for comparison with other DNA self-supervised foundational models. These datasets were processed into a standardized format to facilitate

experimentation and ensure reproducibility in the assessment of large LM performance (Methods). We evaluated our transformer models through two different techniques: probing and fine-tuning (Fig. 1b). 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 multilayer perceptron (MLP) composed of up to two hidden layers. In the case of fine-tuning, the LM head is substituted with either a classification or regression head, and a parameter-efficient technique is used for retraining (Methods). To ensure a fair and accurate comparison between different models, we implemented a tenfold cross-validation strategy.

To compare our pre-trained foundational model schemes with standard, supervised methods in the field, we trained different variants of the BPNet convolutional architecture⁹ from scratch on each of the 18 tasks (Methods). The BPNet architecture has been used widely in genomics and represents a very strong default architecture for modeling small-sized datasets from scratch through supervised learning. We observed strong performance across tasks for the original BPNet model (average Matthews correlation coefficient (MCC) of 0.665), which we could improve by increasing its size to 28 million parameters (average MCC of 0.683), confirming that directly supervised convolutional architectures perform very well for genomics tasks (Fig. 2a,b). We next evaluated how probing and fine-tuning of the NT models compare with these supervised baseline models on our benchmark datasets. We considered the models to be equivalent to or better than other models if the resulting two standard deviations either overlapped or were superior to the reported baseline value, respectively.

Using this criterion, NT models matched the baseline BPNet models in 5 tasks and exceeded them in 8 out of the 18 tasks through probing alone (Supplementary Fig. 1 and Supplementary Table 6) and significantly outperformed probing from raw tokens. In agreement with recent work³⁴, we observed that the best performance is both model- and layer-dependent (Supplementary Table 8). We also noted that the highest model performance is never achieved by using embeddings from the final layer, as shown in earlier work³. For instance, in the enhancer types prediction task, we observed a relative difference as high as 38% between the highest- and lowest-performing layer, indicating significant variation in learned representations across the layers (Supplementary Fig. 3). Compared to our probing strategy, our fine-tuned models either matched ($n = 6$) or surpassed ($n = 12$) out of the 18 baseline models (Fig. 2a,b and Supplementary Tables 7 and 9). Notably, fine-tuned NT models outperformed the probed models, and the larger and more diverse models consistently outperformed their smaller counterparts. These results support the necessity for fine-tuning the NT foundation models toward specific tasks to achieve superior performance. Our results also suggest 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,b). This implies that a strategy of increased sequence diversity, rather than just increased model size, may lead to improved prediction performance, particularly when computational resources are limited.

Fine-tuning has not been extensively explored in previous work³, possibly due to its demanding computational requirements. We overcame this limitation by adopting a recent parameter-efficient fine-tuning technique³⁵ which requires only 0.1% of the total model parameters (Fig. 1b and Methods). This approach allowed for faster fine-tuning on a single GPU, reduced storage needs by 1,000-fold over all fine-tuning parameters, while still delivering comparable performance. In practice, we observed that rigorous probing was slower and more computationally intensive than fine-tuning, despite the apparent simplicity of using straightforward downstream models on embeddings. This discrepancy arises from the significant impact of factors such as layer choice, downstream model selection, and

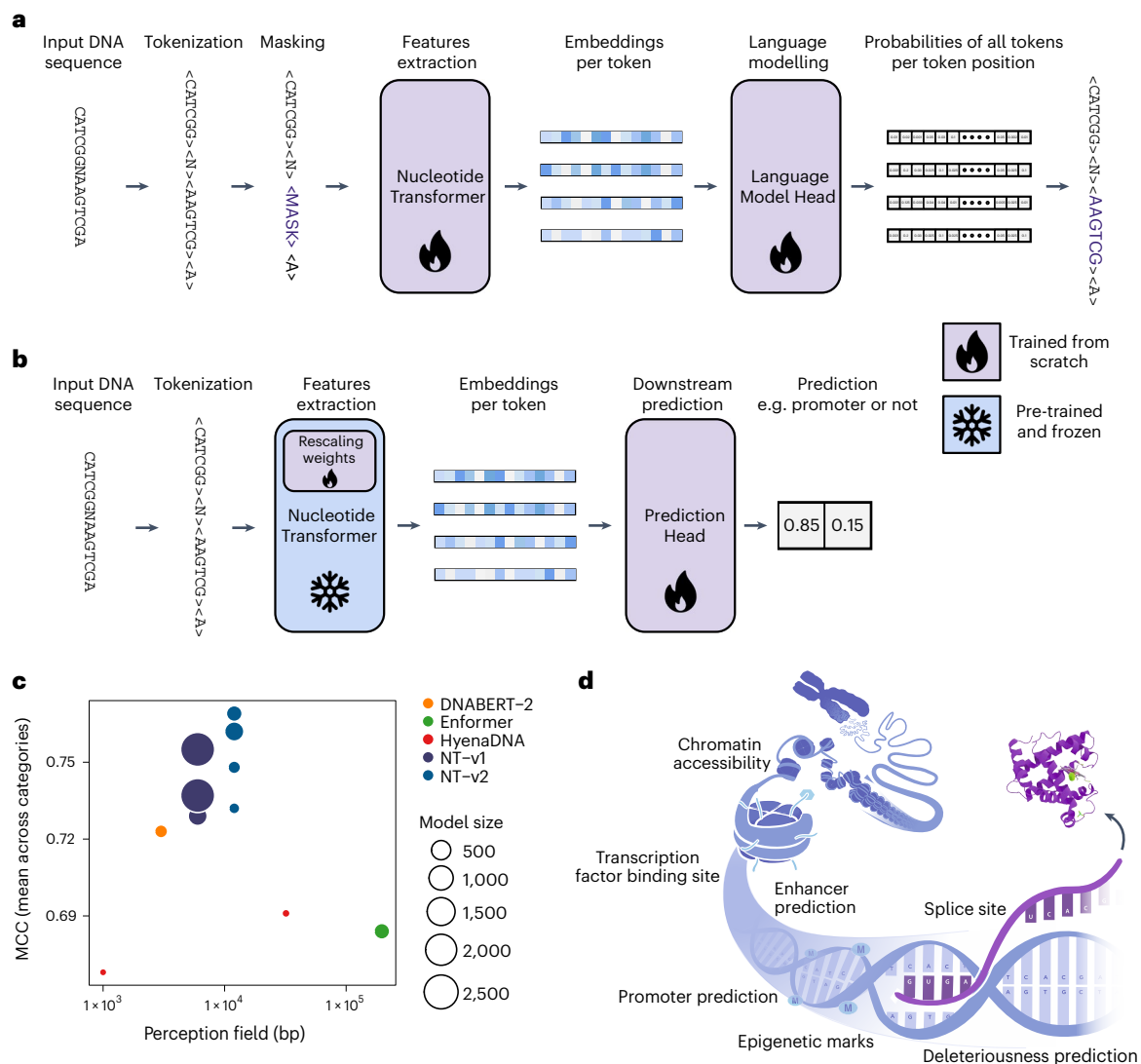


Fig. 1 | Nucleotide Transformer: effective methodology to pre-train, fine-tune, analyze and compare foundational models for genomics. a,b, Overview of NT training (a) and application for downstream genomic prediction tasks through fine-tuning (b). Downstream task prediction through probing is similar but without the rescaling weights in the NT. c, Comparison of NT models to

other foundational genomics models in terms of perception field size, number of parameters and performance over our benchmark made of 18 curated downstream tasks. d, Graphical representation of genomic features considered for downstream tasks (adapted from elsewhere⁴⁸).

hyperparameters on performance. Additionally, fine-tuning exhibited a smaller variance in performance, enhancing the robustness of the approach. Overall, this general approach is versatile and adaptable to various tasks without requiring adjustments to the model architecture or hyperparameters. This stands in contrast to supervised models, which typically feature distinct architectures and necessitate training from scratch for each task.

Finally, we aimed to evaluate the potential for large language DNA models to compete with robust baselines trained in a supervised manner using extensive datasets and optimized architectures. To do so, we applied the Multispecies 2.5B model to three additional genomic prediction tasks, which encompassed the classification of 919 chromatin profiles from a diverse set of human cells and tissues¹⁰, predicting canonical splice acceptor and donor sites across the whole human genome³⁶, and predicting developmental and housekeeping enhancer activities from *Drosophila melanogaster* S2 cells¹² (Methods). Remarkably, the Multispecies 2.5B model achieved performance levels closely aligned with those of specialized DL models, despite no additional changes or optimizations of its original fine-tuning architecture.

For instance, in the case of classifying chromatin feature profiles, we obtained area under the curve (AUC) values that were, on average, only approximately ~1% lower than those achieved by DeepSEA (Fig. 2c). Regarding the prediction of whether each position in a pre-mRNA transcript is a splice donor, splice acceptor or neither, we adapted the NT model to deliver nucleotide-level splice site predictions and achieved a top-k accuracy of 95% and a precision-recall AUC of 0.98 (Fig. 2d). Notably, our 2.5B 6-kb-context model matched the performance of the state-of-the-art SpliceAI-10k³⁶, which was trained on 15-kb input sequences, in addition to other splicing baselines; and outperformed SpliceAI when tested on 6-kb input sequences. Finally, in the case of housekeeping and developmental enhancer prediction, our model slightly surpassed (1%) and obtained lower (4%) correlation values, respectively (Fig. 2e), when compared to those of DeepSTARR¹². Across these three different tasks, we also conducted a comparison between our parameter-efficient fine-tuning and full-model fine-tuning (training all the parameters of the whole model to optimize its performance on a specific task or dataset). Of note, we observed no significant improvement in chromatin and splicing predictions, and only a modest

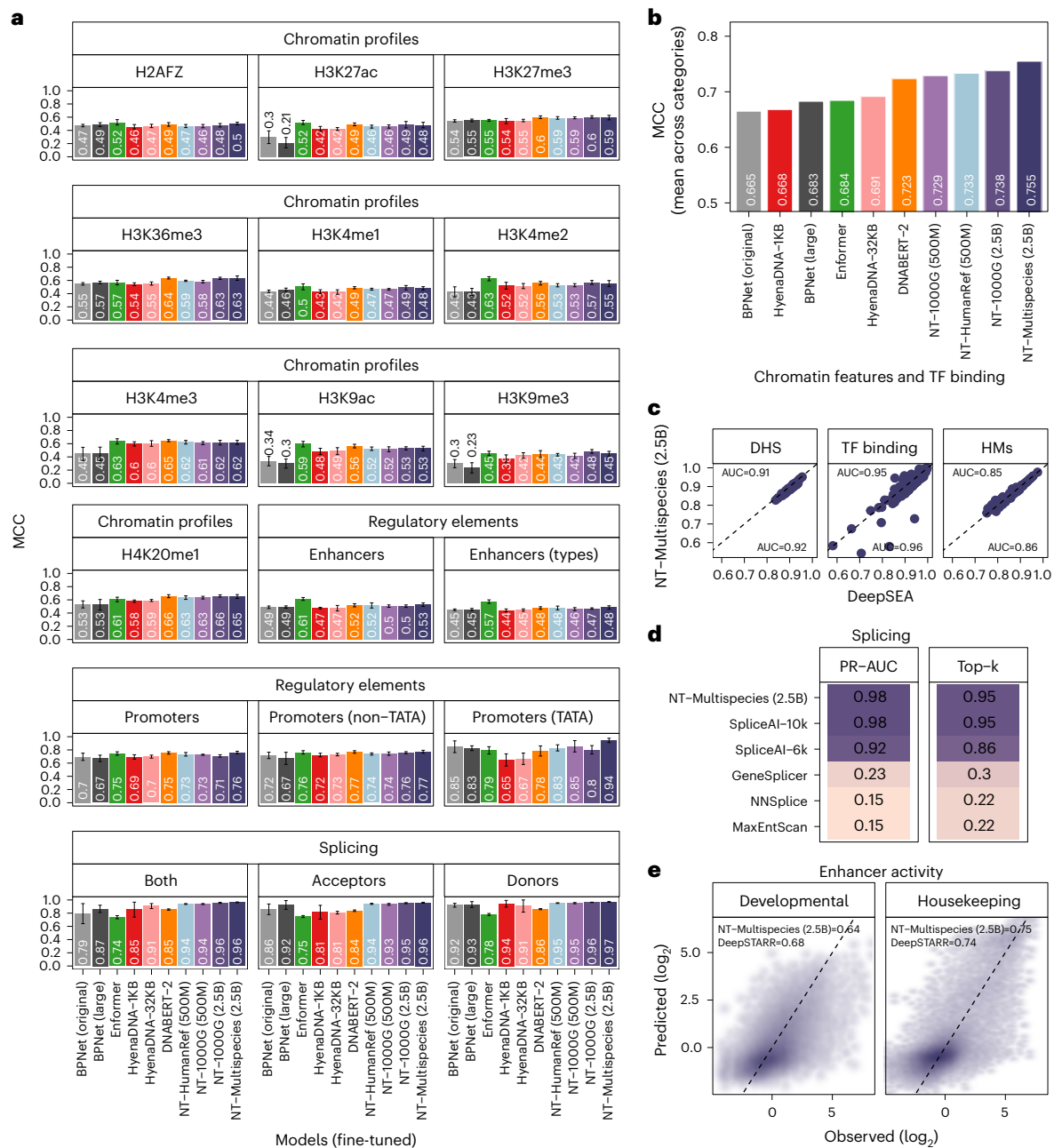


Fig. 2 | The Nucleotide Transformer model accurately predicts diverse genomics tasks after fine-tuning. **a**, Performance results across downstream tasks for fine-tuned NT models as well as HyenaDNA, DNABERT and Enformer pre-trained models based on the MCC. We also trained BPNNet models from scratch for comparison (original, 121,000 parameters; large, 28 million parameters). Data are presented as mean MCC $\pm 2 \times$ s.d. from the tenfold cross-validation procedure ($n = 10$ for each point). **b**, Normalized mean of MCC performance across downstream tasks (divided by category) for all LMs after fine-tuning. **c**, The Multispecies 2.5B model performance on DNase I hypersensitive sites

(DHSs), histone marks (HMs) and TF site predictions from different human cells and tissues compared to the baseline DeepSEA model. Each dot represents the receiver operating characteristic (ROC) AUC for a different genomic profile. The average AUC per model is labeled. **d**, The Multispecies 2.5B model performance on predicting splice sites from the human genome, compared to the SpliceAI and other splicing models. **e**, The Multispecies 2.5B model performance on developmental and housekeeping enhancer activity predictions from *Drosophila melanogaster* S2 cells, compared to the baseline DeepSTARR model.

3% enhancement in enhancer activity predictions (Supplementary Fig. 2), supporting the use of our efficient fine-tuning approach. Altogether, our extensive benchmarks and results demonstrate the flexibility and performance of NT as a general approach to tackle many diverse genomics tasks with high accuracy.

Benchmark of genomics foundational models

We compared the NT models to other genomics foundational models: DNABERT-2 (ref. 23), HyenaDNA (1-kb and 32-kb-context length;²⁵) and

the Enformer (using it as an alternative architecture for a pre-trained model;¹⁹; Fig. 2a,b and Methods). We excluded DNABERT-1 from this comparison as it can only handle a maximum input length of 512 bp and thus could not be used for most tasks²⁰. To ensure fair comparisons, all models underwent fine-tuning and evaluation using the same protocol across the 18 downstream tasks (Methods). When compared to DNABERT-2, HyenaDNA-32-kb and Enformer, our Multispecies 2.5B model achieved the highest overall performance across tasks (Fig. 2a,b and Table 9). Still, Enformer achieved the best performance

on enhancer prediction and some chromatin tasks, demonstrating that it can be a strong DNA foundation model. Our model exhibited better performance than every other model on all promoter and splicing tasks. Notably, despite HyenaDNA being pre-trained on the human reference genome, our Multispecies 2.5B model matched ($n = 7$) or outperformed ($n = 11$) it in all 18 tasks, highlighting the advantage of pre-training on a diverse set of genome sequences. We have established an interactive leaderboard containing results for all models across each task to facilitate comparisons (https://huggingface.co/spaces/InstaDeepAI/nucleotide_transformer_benchmark). This represents an extensive benchmark of foundational genomics models and should serve as a reference for the development of further LMs in genomics (Fig. 1c).

Detection of known genomic elements in an unsupervised manner

To gain insights into the interpretability and understand the type of sequence elements that the NT models use when making predictions, we explored different aspects of their architecture. First, we assessed the extent to which the embeddings can capture sequence information associated with five genomic elements (Supplementary Tables 7 and 10). We observed that the NT models, without any supervision, learned to distinguish genomic sequences that were uniquely annotated as intergenic, intronic, coding and untranslated regions (UTRs), albeit with varying degrees of proficiency across different layers (Fig. 3a, Supplementary Fig. 7 and Methods). In particular, the 500-M-sized models and those trained on less-diverse sequences, exhibited lower separation among genomic regions, reinforcing 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, the strongest separation at layer 1 was observed between intergenic and genic regions, followed by 5' UTR regions on layer 5, and a separation between most regions on layer 21 (Fig. 3a). The limited separation of 3' UTR regions from other elements suggests that the model has not fully learned to distinguish this type of element, or as previously suggested, that many of these regions might be mis-annotated³⁷. Consistent with these observations, our supervised probing strategy demonstrated high classification performance for these elements, with accuracy values exceeding 0.78, especially for deeper layers (Fig. 3b). This demonstrated that the NT models have learned to detect known genomic elements within their embeddings in an unsupervised manner, which can be harnessed for efficient downstream genomics task predictions.

Next, we conducted an analysis of the models through the lens of attention to comprehend which sequence regions are captured and utilized by the attention layers³⁸. We computed the attention percentages across each model head and layer for sequences containing nine different types of genomic elements related to gene structure and regulatory features (Fig. 3c). In a formal sense, an attention head is considered to recognize specific elements when its attention percentage significantly exceeds the naturally occurring frequency of that element in the pre-training dataset (Methods). For example, a percentage of 50% implies that, on average over the human genome, 50% of the attention of that particular head is directed toward the type of element of interest. By applying this approach to each type of element across approximately 10,000 different 6-kb sequences, where the element can be situated at various positions and accounts for between 2–11% of the sequence (Supplementary Table 10), we discovered that attention is distinctly focused on specific types of genomic elements across its diverse heads and layers (Fig. 3d and Supplementary Figs. 8–16). 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 introns (117 out of 640 heads), exons ($n = 72$) and transcription factor (TF) binding sites ($n = 74$) (Supplementary Figs. 8 and 9 and Supplementary Table 12), despite the relatively small proportion of sequences containing exons and TF motifs. Regarding enhancers, the maximum attention percentages

were highest for the largest models, with the 1000G 2.5B model, for instance, achieving nearly 100% attention (Supplementary Fig. 15). Similar patterns were also observed for other genomic elements such as 3' UTR, promoters and TF binding sites, where the 1000G 2.5B model showed highly specialized heads with high attention, particularly in the first layers (Supplementary Figs. 8–16).

To gain deeper insights into the pre-trained NT Multispecies 2.5B model at higher resolution (focusing on more-local sequence features), we examined token probabilities across different types of genomics elements as a metric of sequence constraints and importance learned by the model. Specifically, we calculated the six-mer token probabilities (based on masking each token at a time) for every 6-kb window in chromosome 22. Our findings revealed that, in addition to repetitive elements, that are well reconstructed by the model as expected, the pre-trained model learned a variety of gene structure and regulatory elements. These included acceptor and donor splicing sites, polyA signals, CTCF binding sites and other genomic elements (Supplementary Fig. 17a–d). Furthermore, we compared our token predictions with an experimental saturation mutagenesis splicing assay of exon 11 of the gene *MST1R* (data from Braun et al.³⁹). This analysis revealed a significant correlation between the experimental mutation effects and the token predictions made by the Multispecies 2.5B model (Pearson correlation coefficient (PCC) = 0.44; Supplementary Fig. 17e). The model not only captured constraints at different splicing junctions but also identified a region in the middle of the second intron crucial for the splicing of this exon. These results serve as a robust validation of the biological knowledge acquired by the NT model during unsupervised pre-training.

Last, the Multispecies 2.5B model, which had been fully fine-tuned on the DeepSTARR enhancer activity data, was examined to determine whether it had learned about TF motifs and their relative importance specifically for enhancer activity. We used a dataset of experimental mutation of hundreds of individual instances of five different TF motif types across hundreds of enhancer sequences¹² and evaluated the model's accuracy in predicting these mutation effects. In comparison with the state-of-the-art enhancer activity DeepSTARR model, our model achieved similar performance for four TF motifs and demonstrated superior performance for the Dref motif (Supplementary Fig. 18). Collectively, these results illustrate how the NT models have acquired the ability to recover gene structure and functional properties of genomic sequences and integrate them directly into its attention mechanism. This encoded information should prove helpful in evaluating the significance of genetic variants.

The pre-trained embeddings predict the impact of mutations

Additionally, we evaluated the NT models' ability to assess the severity of various genetic variants and prioritize those with functional significance. We first investigated the use of zero-shot scores, which are scores used to predict classes that were not seen by the model during training. Specifically, we computed zero-shot scores using different aspects of vector distances in the embedding space, as well as those derived from the loss function, and compared their distribution across ten different types of genetic variants that varied in severity⁴⁰ (Fig. 4a and Methods). Encouragingly, several of these zero-shot scores exhibited a moderate correlation with severity across the models (Supplementary Fig. 19). This illustrates how unsupervised training alone captured relevant information related to the potential severity of genetic mutations and highlighted the utility of evaluating different scoring methods. The high variability in correlation between scores also suggests that distinct aspects of the embedding space may more effectively capture information related to severity. Among these scores, cosine similarity exhibited the highest correlation with severity across models, with r^2 values ranging from -0.35 to -0.3 ($P < 6.55 \times 10^{-186}$; Supplementary Fig. 19). Across models, the lowest cosine similarity scores were assigned to genetic variants affecting protein function, such as stop-gained variants,

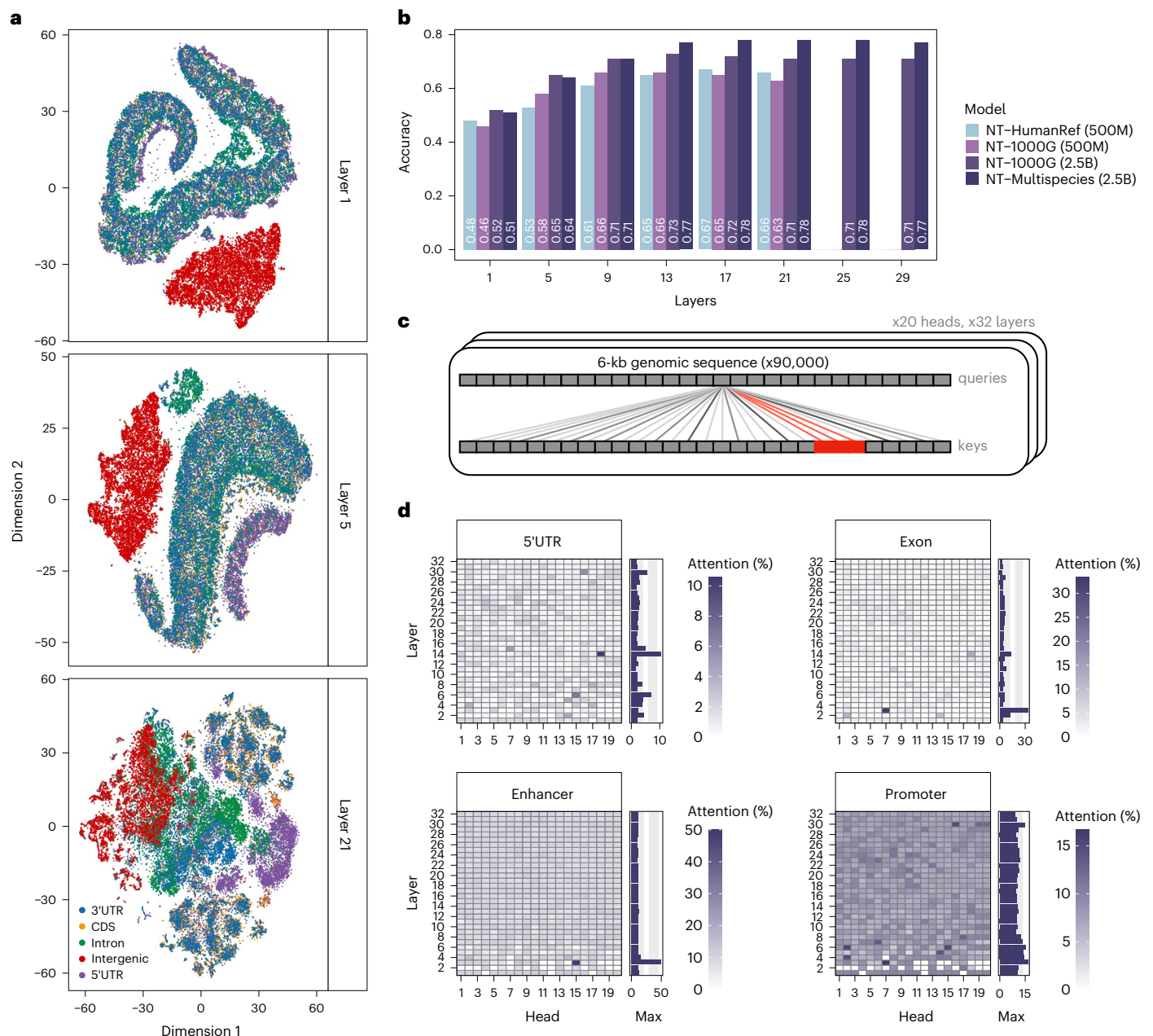


Fig. 3 | The Nucleotide Transformer models acquired knowledge about genomic elements. a, t-SNE projections of embeddings of five genomic elements from layer 1, 5 and 21 based on the Multispecies 2.5B model. **b**, Accuracy estimates based on probing to classify five genomic elements across layers. **c**, Schematic describing the evaluation of attention levels at a given genomic

element. **d**, Attention percentages per head and layer across the Multispecies 2.5B model computed on 5' UTR, exon, enhancer and promoter regions. Barplot on the right of each tile plot shows the maximum attention percentage across all heads for a given layer.

as well as synonymous and missense variants (Fig. 4b). Conversely, we noted that higher scores were assigned to potentially less functionally important variants, such as intergenic variants, highlighting its potential use to capture the effect of the severity of genetic variants.

Next, we also explored the potential of zero-shot scores to prioritize functional variants as well as those with pathogenic effects. Specifically, we assessed the ability of the models to classify genetic variants that influence gene expression regulation (expression quantitative trait loci (eQTLs)), genetic variants linked to DNA methylation variations (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). The zero-shot scores demonstrated high classification performance, with the 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. Last, and to more formally evaluate the effectiveness of these models, we also made predictions based on fine-tuned models and compared their performance against several methods. These methods encompassed those measuring levels of genomic conservation, as well as scores obtained from models trained on functional features. Notably, the fine-tuned models either slightly outperformed or closely matched the performance of the other models (Fig. 4d and Supplementary Fig. 20). The best-performing models for prioritizing molecular phenotypes (eQTLs and meQTLs) were those trained on human sequences, whereas the best-performing model for

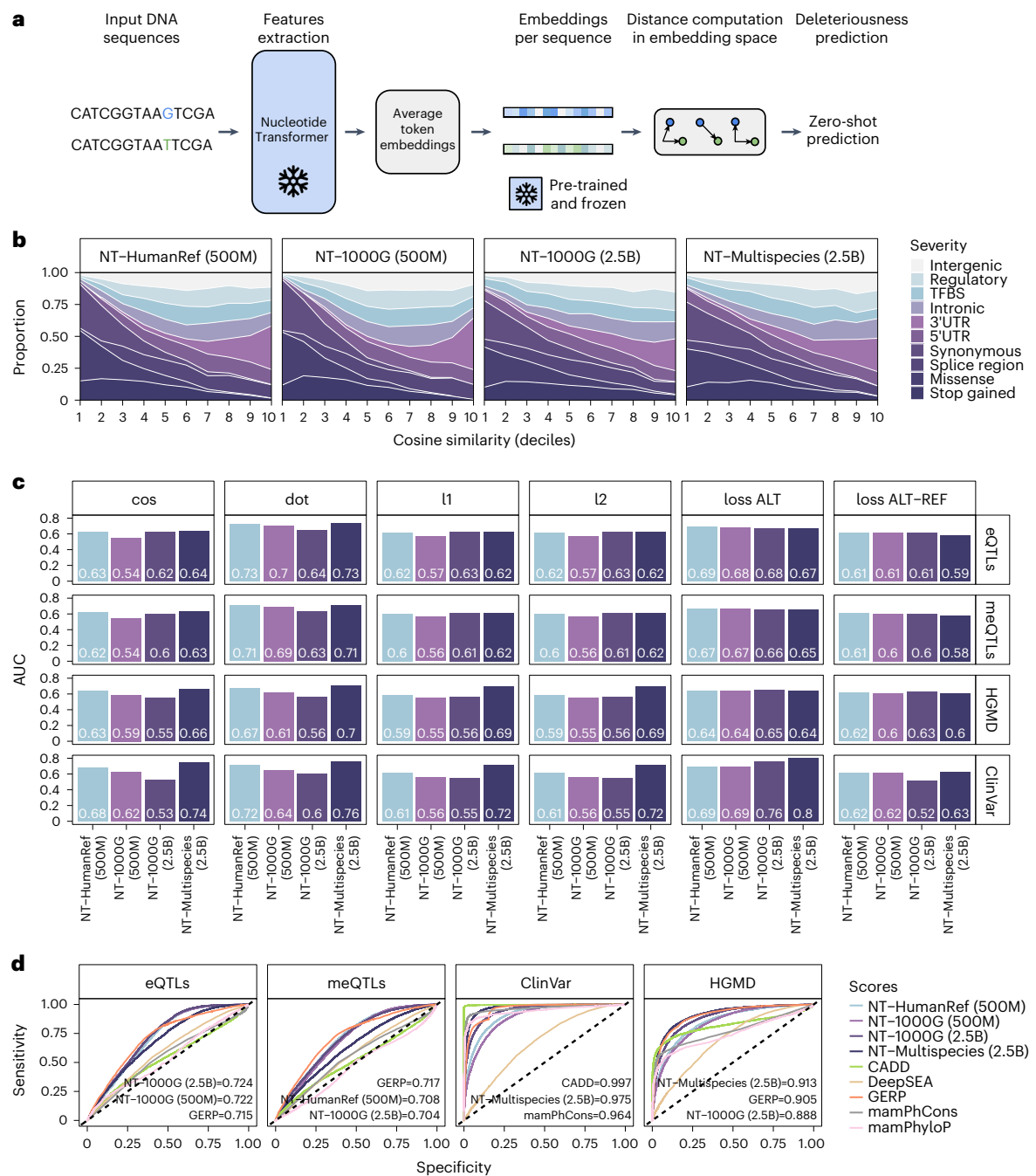


Fig. 4 | Prioritizing functional genetic variants. **a**, Overview of the NT application of zero-shot predictions. **b**, Proportion of variant consequence terms across deciles based on the cosine similarity metric across models. The consequence terms are shown in order of severity (less severe to more 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 ROC AUC. The AUC for the three best-performing models is shown.

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 leveraged sequence variation across species to learn about the degree of conservation across sites. Our results also suggest that higher predictive power for noncoding variants such as eQTLs and meQTLs, could be achieved by better-learned sequence variation derived from increased human genetic variability. Moreover, when compared to zero-shot scores, the dot-product yielded AUC values of 0.73 and 0.71 for eQTLs and meQTLs, respectively, slightly surpassing or matching those obtained by the fine-tuned models. Given that most of these

genetic variants tend to reside within regulatory regions^{41–43}, it is probable that the models, without any supervision, have learned to distinguish relevant regulatory genomic features associated with 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. 3a) and promoters (Supplementary Fig. 13), which have been shown to be enriched in meQTLs and eQTLs^{41–43}. Overall, these results illustrate how DNA-based transformer models can help reveal and contribute to understanding the potential biological implications of variants linked to molecular phenotypes and disease.

Model optimization for cost-effective predictions in genomics

Finally, we explored the potential for optimizing our top-performing models by incorporating contemporary architectural advancements and extending the training duration. We developed four new NT models (NT-v2) with varying number of parameters ranging from 50 million to 500 million, and introduced a series of architectural enhancements (Supplementary Table 1 and Methods). These include the incorporation of rotary embeddings, the implementation of swiGLU activations, and the elimination of MLP biases and dropout mechanisms, in line with the latest research⁴⁴. Additionally, we expanded the context length to cover 12 kb to accommodate longer sequences and capture more distant genomic interactions. We extended the training duration of the 250M- and 500M-parameter models to encompass 1 trillion tokens, aligning with recent recommendations in the literature⁴⁵ (Fig. 5a). After pre-training on the same multispecies dataset, all four NT-v2 models underwent fine-tuning and evaluation across the same set of 18 downstream tasks, with their results compared to those of the four initial NT models (Fig. 5b and Supplementary Tables 7 and 9).

We observed that the 50M-parameter NT-v2 model achieved similar performance than our two NT 500-million-parameter models as well as the 2.5B-parameter model trained on the 1000 Genomes dataset. This demonstrates that the synergy of a superior pre-training dataset, coupled with advancements in training techniques and architecture, can lead to a remarkable 50-fold reduction in model parameters while simultaneously enhancing performance (Fig. 5c). Indeed, the NT-v2 250M- and 500M-parameter model managed to achieve improved performance over the 2.5B-parameter multispecies model while maintaining a significantly leaner parameter count and doubling the perception field. Of particular interest is the NT-v2 250M-parameter model, which attained the best performance on our benchmark (average MCC of 0.769) while being ten-times smaller than the 2.5B-parameter model (Fig. 5c).

To gain further insights into the need for longer pre-training, we conducted a systematic evaluation of the NT-v2 models' performance in function of the number of tokens seen during pre-training (Fig. 5b). This showed that the 250M-parameter model has a small improvement over the 500M model only after training for 900 billion tokens. In summary, the NT-v2 models, equipped with a context length of 12 kb, are suitable for deployment on cost-effective accelerators due to their compact sizes. Consequently, they offer an economically viable and practical alternative for users seeking to leverage cutting-edge foundational models in their downstream applications.

As a relevant use case, with an established well-performing model, we assessed the advantage of the longer context length of the NT-v2 models by evaluating the 500M model on the SpliceAI splicing task. We evaluated the 500M model as this model showed the highest performance on splicing-related tasks (Supplementary Tables 7 and 9). We adapted our classification head to predict nucleotide-level probabilities of being a splicing donor, acceptor or none (Fig. 5d and Methods). Compared to the NT 6-kb models, our NT-v2 500M 12-kb model improved performance by 1% to a top-k accuracy of 96% and a precision-recall AUC of 0.98 (Fig. 5e). This performance surpasses that of the state-of-the-art SpliceAI-10K³⁶, which was trained on 15-kb input sequences. It is worth noting that we did not attempt to optimize our model architectures specifically for the splicing prediction task; instead, we applied a similar fine-tuning approach as used for other downstream tasks, with adjustments in the classification head to yield nucleotide-level predictions. Further architectural refinements tailored to specific tasks like splicing are likely to enhance performance. In summary, these results affirmed the utility and effectiveness of both NT v1 and v2 models for a wide range of genomics tasks, requiring minimal modifications and computing power while achieving high accuracy.

Discussion

This study represents an attempt to investigate the impact of different datasets used to pre-train equally-sized transformer models on

DNA sequences. Our results, based on distinct genomic prediction tasks, demonstrate that both intra-species (when training on multiple genomes of a single species) and inter-species (on genomes across different species) variability significantly influences accuracy across tasks (Figs. 1c and 2a). Models trained on genomes from different species outperformed those trained exclusively on human sequences in most of the considered human prediction tasks. This suggests that transformer models trained on diverse species have learned to capture genomic features that likely have functional importance across species, thus enabling better generalization in various human-based prediction tasks. Based on this finding, we anticipate that future studies may benefit from leveraging genetic variability across species, including determining the best ways to sample this variability. Another interesting avenue would be to explore different ways of encoding intra-species variability. Our approach of mixing sequences from all individual genomes resulted in limited improvements, and thus suggests that it might not be that trivial to leverage genomes from different individuals when most of them are shared.

The transformer models trained in this study ranged from 50 million up to 2.5 billion parameters, which is 20 times larger than DNABERT-2 (ref. 23) and ten-times larger than the Enformer¹⁹ backbone models. As previously demonstrated in NLP research²⁷, our results on the 18 genomic prediction tasks confirmed that increasing model size leads to consistently improved performance. To train the models with the largest parameter sizes, we utilized a total of 128 GPUs across 16 compute nodes for 28 days. Significant investments were made in engineering efficient training routines that fully utilized the infrastructure, underscoring the importance of both specialized infrastructure and dedicated software solutions. Once trained, however, these models can be used for inference at a relatively low cost, and we provide notebooks to apply these models to any downstream task of interest, thereby facilitating further research.

Previous work, which was based on LMs trained on biological data (primarily protein sequences), has evaluated downstream performance mostly by probing the last transformer layer⁵. This choice was likely driven by its perceived ease of use, relatively good performance and low computational complexity. In this study, our aim was to evaluate downstream accuracy through computationally intensive and thorough probing of different transformer layers, downstream models and hyperparameter sweeps. We observed that the best probing performance was achieved with intermediate transformer layers (Supplementary Fig. 1), which aligns with recent work in computational biology³⁴ and common practice in NLP⁴⁶. Through probing alone, the Multispecies 2.5B model outperformed the BpNet baseline model for 8 out of 18 tasks. Additionally, we explored a recent downstream fine-tuning technique that introduces a small number of trainable weights into the transformer. This approach provides a relatively fast and resource-efficient fine-tuning procedure with minor differences compared to full-model fine-tuning (IA³)³⁵. Notably, this fine-tuning approach only requires 0.1% of the total number of parameters, allowing even our largest models to be fine-tuned in under 15 min on a single GPU. In comparison to the extensive probing exercise, this technique yielded superior results while using fewer compute resources, confirming that downstream model engineering can lead to performance improvements⁴⁷. The use of this technique makes fine-tuning competitive with probing from an operational perspective, both for training and inference, while achieving improved performance.

While supervised models with optimized architectures and extensive training on large datasets continue to demonstrate superior performance, the DNA language approach remains competitive with these models in a diverse range of tasks encompassing areas such as histone modifications, splicing sites, and regulatory elements characterization. Moreover, the NT model represents a versatile approach that can be seamlessly tailored to a diverse array of tasks for both human and nonhuman species. The value of our unsupervised pre-training

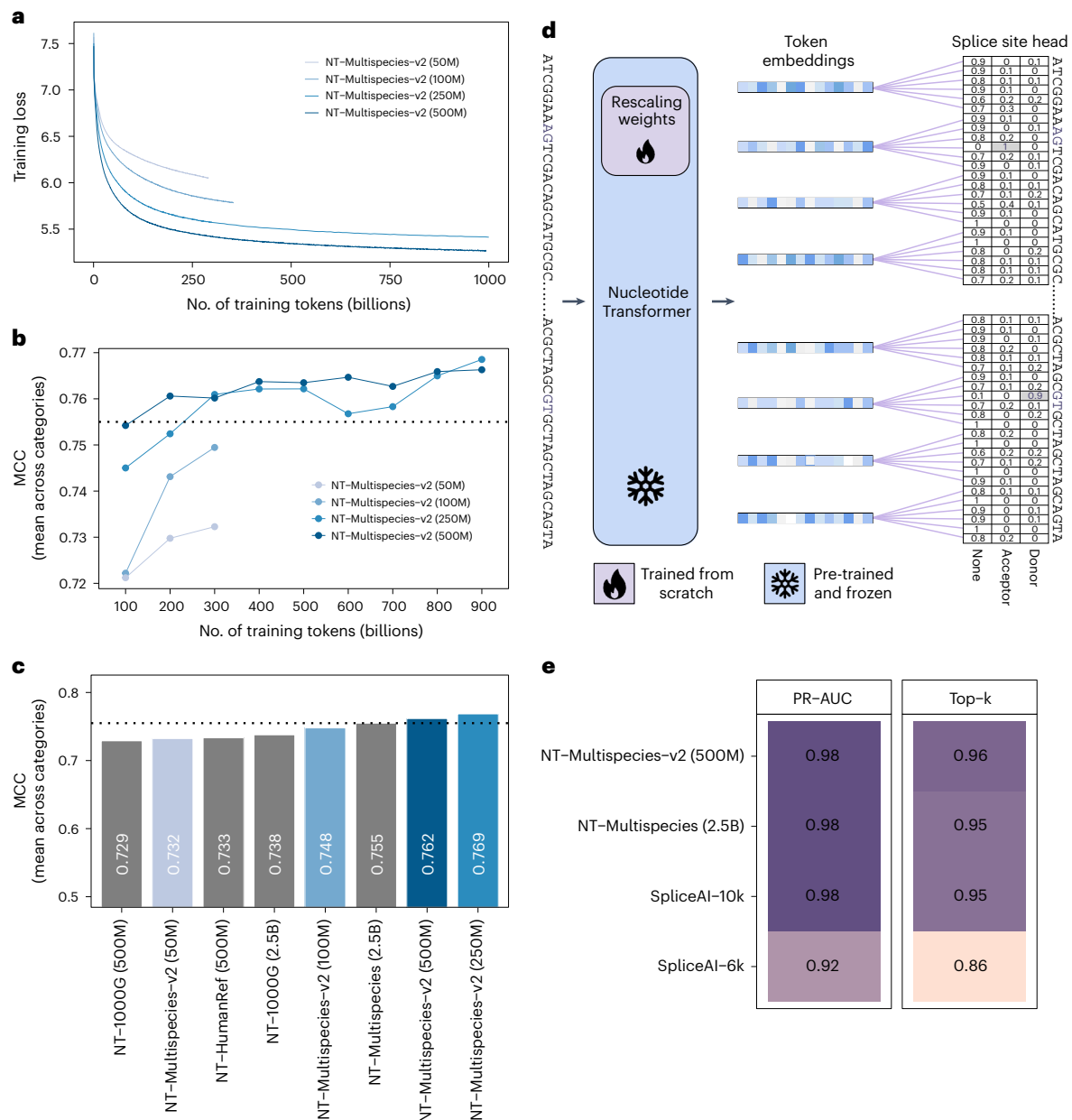


Fig. 5 | Efficient model architecture allows matching performance while strongly reducing the number of model parameters. a, NT-v2 models loss value evolution during training as a function of the number of tokens seen so far. **b**, Normalized mean of MCC performance for NT-v2 models as a function of the number of tokens seen during pre-training. **c**, Normalized mean of MCC performance across downstream tasks (divided by category) for all NT (gray)

and -v2 (blue shades) NT models after fine-tuning. Black dashed line represents the performance of the NT 2.5B Multispecies model. **d**, Overview of the NT fine-tuning on nucleotide-level splice site prediction task. Pre-trained weights and weights trained from scratch are highlighted. **e**, The Multispecies v2 500M model performance on predicting splice sites from the human genome, compared to its 2.5B counterpart and SpliceAI.

approach becomes particularly evident when dealing with smaller datasets, where training supervised models from scratch typically yields limited results. Recognizing the pivotal role of foundational genomics models in the field, we have conducted an extensive comparison and benchmarking study, evaluating our model against four distinct pre-trained models with different architectures: DNABERT-2 (ref. 23), HyenaDNA-1 kb, HyenaDNA-32 kb (ref. 25) and Enformer¹⁹. This robust benchmark will serve as a reference point for the development of future LMs in genomics.

Through various analyses related to the transformer architecture, we have demonstrated that the models have acquired the ability not only to reconstruct genetic variants (Supplementary Note) but also to recognize key regulatory genomic elements. This property is

demonstrated throughout the analysis of attention maps, embedding spaces, token reconstruction and probability distributions. Crucial regulatory elements governing gene expression, such as enhancers and promoters^{41–43}, were consistently detected by all models across multiple heads and layers. Additionally, we observed that each model contained at least one layer that produced embeddings clearly distinguishing the five genomic elements analyzed. Given that self-supervised training facilitated the detection of these elements, we anticipate that this approach can be harnessed for the characterization or discovery of novel genomic elements in future research.

We have demonstrated that transformer models can match and even outperform other methods for predicting variant effects and deleteriousness. In addition to developing supervised transformer

models, we have also showcased the utility of zero-shot-based scores, especially for predicting noncoding variant effects. Given that these zero-shot-based scores can be derived solely from genomic sequences, we encourage their application in nonhuman organisms, especially those with limited functional annotation.

Last, we have demonstrated the potential of enhancing the model architecture to achieve a dual benefit of reducing model size and improving performance. Our subsequent series of NT-v2 models have a context length of 12 kb. To put this in perspective, this length is 24× and 4× larger than the respective average context lengths of DNABERT-1 (500 bp) and DNABERT-2 (3 kb). These advanced models not only exhibit improved downstream performance but also offer the advantage of being suitable for execution and fine-tuning on economical hardware. This advantage arises from their compact size, complemented by the utilization of the efficient fine-tuning techniques that we have introduced.

While our models have an attention span that remains limited, the recently developed Enformer model¹⁹ suggested that increasing the perception field up to 200 kb 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 10–20 kb 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 addresses this issue by passing sequences through convolution layers to reduce input dimensions before reaching the transformer layers. However, this choice hampers its effectiveness in language modeling. On the other hand, the recent HyenaDNA models²⁵ were trained with perception fields up to 1 million bp, but our benchmark analyses in downstream tasks show that the performance of these models quickly deteriorates when the perception field used during training increases. Based on our results, we suggest that developing transformer models with the ability to handle long inputs while maintaining high performance on shorter ones is a promising direction for the field. In an era where multi-omics data are rapidly expanding, we ultimately anticipate that the methodology presented here, along with the available benchmarks and code, will stimulate the adoption, development and enhancement of large foundational LMs in genomics.

Online content

Any methods, additional references, Nature Portfolio reporting summaries, source data, extended data, supplementary information, acknowledgements, peer review information; details of author contributions and competing interests; and statements of data and code availability are available at <https://doi.org/10.1038/s41592-024-02523-z>.

References

- Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K. Bert: pre-training of deep bidirectional transformers for language understanding. Preprint at <https://arxiv.org/abs/1810.04805> (2018).
- Brown, T. et al. Language models are few-shot learners. *Adv. Neural Inf. Process. Syst.* **33**, 1877–1901 (2020).
- Jumper, J. et al. Highly accurate protein structure prediction with alphafold. *Nature* **596**, 583–589 (2021).
- Rives, A. et al. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proc. Natl Acad. Sci. USA* **118**, e2016239118 (2021).
- Elnaggar, A. et al. ProtTrans: towards cracking the language of life's code through self-supervised deep learning and high performance computing. *IEEE Trans. Pattern Anal. Mach. Intell.* **44**, 7112–7127 (2022).
- Littmann, M., Heinzinger, M., Dallago, C., Olenyi, T. & Rost, B. Embeddings from deep learning transfer go annotations beyond homology. *Sci. Rep.* **11**, 1–14 (2021).
- Marquet, C. et al. Embeddings from protein language models predict conservation and variant effects. *Hum. Genet.* **141**, 1629–1647 (2022).
- Littmann, M., Heinzinger, M., Dallago, C., Weissenow, K. & Rost, B. Protein embeddings and deep learning predict binding residues for various ligand classes. *Sci. Rep.* **11**, 23916 (2021).
- Avsec, Ž. et al. Base-resolution models of transcription-factor binding reveal soft motif syntax. *Nat. Genet.* **53**, 354–366 (2021).
- Zhou, J. & Troyanskaya, O. G. Predicting effects of noncoding variants with deep learning–based sequence model. *Nat. Methods* **12**, 931–934 (2015).
- Mateo, L. J., Sinnott-Armstrong, N. & Boettiger, A. N. Tracing dna paths and rna profiles in cultured cells and tissues with orca. *Nat. Protoc.* **16**, 1647–1713 (2021).
- de Almeida, B. P., Reiter, F., Pagani, M. & Stark, A. DeepSTARR predicts enhancer activity from dna sequence and enables the de novo design of synthetic enhancers. *Nat. Genet.* **54**, 613–624 (2022).
- Eraslan, G., Avsec, Ž., Gagneur, J. & Theis, F. J. Deep learning: new computational modelling techniques for genomics. *Nat. Rev. Genet.* **20**, 389–403 (2019).
- Zhou, J. et al. Deep learning sequence-based ab initio prediction of variant effects on expression and disease risk. *Nat. Genet.* **50**, 1171–1179 (2018).
- Kelley, D. R. Cross-species regulatory sequence activity prediction. *PLOS Comput. Biol.* **16**, e1008050 (2020).
- Kelley, D. R. et al. Sequential regulatory activity prediction across chromosomes with convolutional neural networks. *Genome Res.* **28**, 739–750 (2018).
- Agarwal, V. & Shendure, J. Predicting mRNA abundance directly from genomic sequence using deep convolutional neural networks. *Cell Rep.* **31**, 107663 (2020).
- Chen, K. M., Wong, A. K., Troyanskaya, O. G. & Zhou, J. A sequence-based global map of regulatory activity for deciphering human genetics. *Nat. Genet.* **54**, 940–949 (2022).
- Avsec, Ž. et al. Effective gene expression prediction from sequence by integrating long-range interactions. *Nat. Methods* **18**, 1196–1203 (2021).
- Ji, Y., Zhou, Z., Liu, H. & Davuluri, R. V. Dnabert: pre-trained bidirectional encoder representations from transformers model for DNA-language in genome. *Bioinformatics* **37**, 2112–2120 (2021).
- Zvyagin, M. T. et al. Genslms: genome-scale language models reveal SARS-CoV-2 evolutionary dynamics. Preprint at *bioRxiv* <https://doi.org/10.1101/2022.10.10.511571> (2022).
- Outeiral, C. & Deane, C. M. Codon language embeddings provide strong signals for protein engineering. *Nat. Mach. Intell.* **6**, 170–179 (2024).
- Zhou, Z. et al. Dnabert-2: efficient foundation model and benchmark for multi-species genome. in *Proceedings of the Twelfth International Conference on Learning Representations* <https://openreview.net/pdf?id=oMLQB4EZE1> (ICLR, 2024).
- Fishman, V. et al. Gena-lm: A family of open-source foundational models for long dna sequences. Preprint at *bioRxiv* <https://doi.org/10.1101/2023.06.12.544594> (2023).
- Nguyen, E. et al. Hyenadna: Long-range genomic sequence modeling at single nucleotide resolution. in *37th Conference on Neural Information Processing Systems* <https://openreview.net/pdf?id=ubzNoJjOKj> (NeurIPS, 2023).
- Mendoza-Revilla, J. et al. A foundational large language model for edible plant genomes. *Commun. Biol.* **7**, 835 (2024).
- Rae, J. W. et al. Scaling language models: methods, analysis & insights from training gopher. Preprint at <https://arxiv.org/abs/2112.11446> (2021).
- Consortium, G. P. et al. A global reference for human genetic variation. *Nature* **526**, 68 (2015).

29. Harrow, J. et al. GENCODE: the reference human genome annotation for the encode project. *Genome Res.* **22**, 1760–1774 (2012).
30. Meylan, P., Dreos, R., Ambrosini, G., Groux, R. & Bucher, P. Epi in 2020: enhanced data visualization and extension to ncRNA promoters. *Nucleic Acids Res.* **48**, D65–D69 (2020).
31. ENCODE. An integrated encyclopedia of dna elements in the human genome. *Nature* **489**, 57–74 (2012).
32. The ENCODE Project Consortium. Expanded encyclopaedias of dna elements in the human and mouse genomes. *Nature* **583**, 699–710 (2020).
33. Meuleman, W. et al. Index and biological spectrum of human DNase I hypersensitive sites. *Nature* **584**, 244–251 (2020).
34. Li, F.-Z., Amini, A. P., Yang, K. K. & Lu, A. X. Pretrained protein language model transfer learning: is the final layer representation what we want? in *Machine Learning for Structural Biology Workshop* (NeurIPS, 2022).
35. Liu, H. et al. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. in *36th Conference on Neural Information Processing Systems* (NeurIPS, 2022).
36. Jaganathan, K. et al. Predicting splicing from primary sequence with deep learning. *Cell* **176**, 535–548 (2019).
37. Benegas, G., Batra, S. S. & Song, Y. S. DNA language models are powerful zero-shot predictors of non-coding variant effects. *Proc. Natl Acad. Sci. USA* **120**, e2311219120 (2023).
38. Vig, J. et al. BERTology meets biology: interpreting attention in protein language models. in *Proceedings of the International Conference on Learning Representations 2021* <https://openreview.net/pdf?id=YWtLZvLmud7> (ICLR, 2021).
39. Braun, S. et al. Decoding a cancer-relevant splicing decision in the ron proto-oncogene using high-throughput mutagenesis. *Nat. Commun.* **9**, 3315 (2018).
40. McLaren, W. et al. The Ensembl variant effect predictor. *Genome Biol.* **17**, 1–14 (2016).
41. Lappalainen, T. et al. Transcriptome and genome sequencing uncovers functional variation in humans. *Nature* **501**, 506–511 (2013).
42. Consortium, G. The gtex consortium atlas of genetic regulatory effects across human tissues. *Science* **369**, 1318–1330 (2020).
43. Vösa, U. et al. Large-scale cis- and trans-eQTL analyses identify thousands of genetic loci and polygenic scores that regulate blood gene expression. *Nat. Genetics* **53**, 1300–1310 (2021).
44. Chowdhery, A. et al. PaLM: scaling language modeling with pathways. *J. Mach. Learn. Technol.* **24**, 11324–11436 (2021).
45. Hoffmann, J. et al. Training compute-optimal large language models. in *36th Conference on Neural Information Processing Systems* https://proceedings.neurips.cc/paper_files/paper/2022/file/c1e2faff6f588870935f114ebe04a3e5-Paper-Conference.pdf (NeurIPS, 2022).
46. Rogers, A., Kovaleva, O. & Rumshisky, A. A primer in BERTology: what we know about how bert works. *Trans. Assoc. Comput. Linguist.* **8**, 842–866 (2020).
47. Stärk, H., Dallago, C., Heinzinger, M. & Rost, B. Light attention predicts protein location from the language of life. *Bioinform. Adv.* **1**, vbab035 (2021).
48. Zou, J. et al. A primer on deep learning in genomics. *Nat. Genetics* **51**, 12–18 (2019).

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by-nc-nd/4.0/>.

© The Author(s) 2024

Methods

Models

LMs have been primarily developed within NLP to model spoken languages^{1,2}. An LM is a probability distribution over sequences of tokens (often words) that is 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 few supervised data are available⁴⁹. One technique to train LM task models to predict the most likely tokens at masked positions in a sequence is often referred to as masked language modeling (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 LMs transformers in genomics, considering sequences of nucleotides as sentences and k-mers (with $k = 6$) as words. Transformers are a class of DL 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 embeddings. The main technique to train LM 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.

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 1,000 tokens. We used six-mer tokens as a trade-off between sequence length (up to 6 kb) and embedding size, and because it achieved the highest performance when compared to other token lengths. The token embeddings are then processed by a transformer layer stack. Each transformer layer transforms its input through a layer normalization 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 normalization layer and a two-layer perceptron with GELU activations⁵⁰. 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 Supplementary 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.

Our second version NT-v2 models include a series of architectural changes that proved more efficient: instead of using learned positional embeddings, we use rotary embeddings⁵¹, which are used at each attention layer; we use gated linear units with swish activations without bias, making NLPs more efficient. These improved models also accept sequences up to 2,048 tokens, leading to a longer context window of 12 kb.

Training. The models are trained following BERT methodology¹. 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' parameter models, respectively. Within a sequence, of a subset of 15% of tokens, 80% are replaced by a special masking ('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 (any token different from the class ('CLS'), padding ('PAD') or MASK token), as was carried out 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 1 million tokens per batch. We used the Adam optimizer⁵² with a learning rate schedule and standard values for exponential decay rates and epsilon constants, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 1 \times 10^{-8}$. During a first warmup period, the learning rate was increased linearly between 5×10^{-5} and 1×10^{-4} over 16,000 steps before decreasing following a square root decay until the end of training.

We slightly modified the hyperparameters of our NT-v2 models: the optimizer and learning rate schedule are kept the same; however, we increased batch size to 512 (1 million tokens per batch). Inspired by Chinchilla scaling laws⁴⁵, we also trained our NT-v2 models for longer duration compared to other DL models. Specifically, we pre-trained our NT-v2 50M- and 250M-parameters models for 300 billion tokens, while our '250M' and '500M' parameters models were trained for up to 1 trillion tokens to understand the scaling laws at play. In comparison, the NT-v1 2.5B-parameters models were trained for 300 billion tokens and their 500M counterparts were trained for 50 billion tokens. In the end we used the following model checkpoints for the NT-v2 models: checkpoint 300 billion tokens for the '50M' and '100M' models, checkpoint 800 billion tokens for the '250M' model and checkpoint 900 billion tokens for the '500M' model.

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-learn⁵³ and an MLP. 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 tenfold 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 solver⁵⁴ 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 Supplementary Table 8. 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.

As a baseline, we evaluated the performance of a logistic regression model that takes as input the tokenized sequence, before passing the tokens through the transformer layers. Using the raw tokenized sequences as input yielded much better performance than using a vector where the token ids were one-hot encoded and passed through a pooling layer (summing or averaging, over the sequence length axis).

Fine-tuning. In addition to probing our models through embedding extraction at various layers, we also performed parameter-efficient fine-tuning through the IA³ technique³⁵. 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 $l_{ff} \in \mathbb{R}^{d_{ff}}$, 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 $(l_{ff} \gamma(W_1 x)) W_2$, where γ is the feed-forward network nonlinearity and $\odot$ represents the element-wise multiplication. This adds a total of $L(d_k + d_v + d_{ff})$ 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 as just a fraction of parameters needed updating. Similarly, it alleviated storage requirements, needing to create space for just 0.1% new parameters over 500 million and 2.5 billion for each downstream task using traditional fine-tuning. For instance, for the 2.5B-parameter models, the weights represent 9.5 GB. Considering 18 downstream tasks, classical fine-tuning would have required $9.5 \times 18 = 171$ GB, 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, which contains different shuffles with 90% and 10% of the initial set. For each split, the model was fine-tuned for 10,000 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 3×10^{-3} . Other optimizer parameters were maintained from training regimens. Each model is fine-tuned for 10,000 steps for each task. These hyperparameters were selected as they led to promising results in the field of NLP³⁵. Diverging hyperparameter choices did not yield significant gains in our experiments. We have also compared this approach with fine-tuning from a randomly initialized checkpoint.

Comparison with supervised BPNet baseline. As baseline supervised models, we trained different variants of the BPNet convolutional architecture⁹ from scratch on each of the 18 tasks. We tested the original architecture (121,000 parameters), a large one (28 million) and an extra-large (113 million) one. For each, the hyperparameters were manually adjusted to yield the best performance on the validation set. We implemented a tenfold cross-validation strategy to measure the performance of each of the 18 models per architecture.

Comparison with published pre-trained genomics models. We compared the fine-tuned performance of NT models on the 18 downstream tasks with three different pre-trained models: DNABERT-2 (ref. 23), HyenaDNA (1-kb and 32-kb-context length;²⁵) and Enformer¹⁹. We excluded DNABERT-1 from this comparison as it can only handle a maximum input length of 512 bp and thus could not be used for most tasks²⁰. We ported the architecture and trained weights of each model to our code framework and performed parameter-efficient fine-tuning

on the transformer part of every model as described above, using the same cross-validation scheme for a fair comparison. All results can be visualized in an interactive leaderboard (https://huggingface.co/spaces/InstaDeepAI/nucleotide_transformer_benchmark). Only for HyenaDNA we performed full fine-tuning due to the incompatibility of our parameter-efficient fine-tuning approach with the model architecture.

Note that Enformer has been originally trained in a supervised fashion to solve chromatin and gene expression tasks¹⁹. For the sake of benchmarking, we reused the provided model torso as a pretrained model for our benchmark, which is not the intended and recommended use of the original paper; however, we think this comparison is interesting to highlight the differences between self-supervised and supervised learning for pretraining and observe that the Enformer is indeed a very competitive baseline even for tasks that differ from gene expression. Still, we note that Enformer has been pretrained originally using different data splits and thus its performance in our benchmark evaluation might be inflated due to potential data leakage.

Pre-training datasets

The Human reference genome dataset. The Human reference dataset was constructed by considering all autosomal and sex chromosomes sequences from the reference assembly GRCh38/hg38 (https://www.ncbi.nlm.nih.gov/assembly/GCF_000001405.26) and reached a total of 3.2 billion nucleotides.

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 (http://ftp.1000genomes.ebi.ac.uk/vol1/ftp/data_collections/1000G_2504_high_coverage/working/20201028_3202_phased/) from the 1000 Genomes (1000G) project⁵⁵, which aims at recording genetic variants occurring at a frequency of at least 1% in the human population. The dataset contained 3,202 high-coverage human genomes, originating from 27 geographically structured populations of African, American, East Asian and European ancestry as detailed in Supplementary 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 125 million mutations, 111 million and 14 million of which are single-nucleotide polymorphisms (SNPs) and indels, respectively.

The Multispecies dataset. The selection of genomes for the Multispecies dataset was primarily based on two factors: (1) the quality of available reference genomes and (2) diversity among the species used. The genomes chosen for this dataset were selected from the ones available at the Reference Sequence (RefSeq) collection of NCBI (<https://www.ncbi.nlm.nih.gov/>). To ensure a diverse set of genomes, we randomly selected one genome at the genus level from each of the main groupings available in RefSeq (archaea, fungi, vertebrate_mammalian, vertebrate_other, etc.). However, due to the large number of available bacterial genomes, we opted to include only a random subset of them. 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 Supplementary 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 (Supplementary Table 4).

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 the alphabet the $4^6 = 4,096$ possible six-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 4,104 tokens. To tokenize an input sequence, the tokenizer will start with a class token and then convert the sequence starting from the left, matching six-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 were split into overlapping chunks of 6,100 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 was randomly sampled between 0 and 100, and the sequence was then tokenized from this nucleotide until 1,000 tokens was reached. The number of epochs was determined depending on the dataset so that the model processed a total of 300 billion 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, were sampled at each step. Then, for each sampled chunk, an individual from the 1000G dataset was randomly selected, and if that individual carried mutations at the positions and chromosome corresponding to that chunk, these mutations were introduced into the sequence and the corresponding tokens were replaced. This data-processing technique ensured uniform sampling both over the genome and over the individuals during training, as well as enabled us to efficiently store only mutations for each individual, instead of full genomes.

Hardware. All models were trained on the Cambridge-1 Nvidia super-computer 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 (https://jax.readthedocs.io/en/latest/_autosummary/jax.pmap.html), which relies on the NCCL (<https://developer.nvidia.com/nccl>) protocol to handle communications between nodes and devices, and observed an almost linear decrease of the training time with respect to the number of GPUs available. The 500M-parameter models were trained on a single node for a day, whereas the 2.5B-parameter models required the whole cluster for 28 days to be trained. The NT-v2 models with varying number of parameters ranging from 50 million to 500 million were similarly trained on a single node for a single day. 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 were 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 min for the 500M-parameter models and 50 min 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 2 days to compute. Then, 760,000 downstream models were fit on a cluster of 3,000 CPUs, requiring 2.5 days.

Nucleotide Transformer downstream tasks

Epigenetic marks prediction. Histone ChIP-seq data for ten histone marks in the K562 human cell line were obtained from ENCODE³¹ (<https://www.encodeproject.org/>). We downloaded bed narrow-Peak files with the following identifiers: H3K4me3 (ENCFF706WUF), H3K27ac (ENCFF544LXB), H3K27me3 (ENCFF323WOT), H3K4me1 (ENCFF135ZLM), H3K36me3 (ENCFF561OUZ), H3K9me3 (ENCFF963GZJ), H3K9ac (ENCFF891CHI), H3K4me2 (ENCFF749KLQ), H4K20me1 (ENCFF909RKY) and H2AFZ (ENCFF213OTI). For each dataset, we selected 1-kb genomic sequences containing peaks as positive examples and all 1-kb sequences not overlapping peaks as negative examples.

Promoter sequence prediction. We built a dataset of promoter sequences to evaluate the capabilities of the model to identify promoter motifs. We downloaded all human promoters from the Eukaryotic Promoter Database³⁰ (<https://epd.expasy.org/epd/>), spanning 49 bp upstream and 10 bp downstream of transcription start sites (https://epd.expasy.org/ftp/epdnew/H_sapiens/006/Hs_EPDnew_006_hg38.bed). This resulted in 29,598 promoter regions, 3,065 of which were TATA-box promoters (using the motif annotation at https://epd.expasy.org/ftp/epdnew/H_sapiens/006/db/promoter_motifs.txt). We selected 300-bp genomic sequences containing promoters as positive examples and all 300-bp sequences not overlapping promoters as negative examples. These positive and negative examples were used to create three different binary classification tasks: presence of any promoter element (Promoter all), a promoter with a TATA-box motif (Promoter TATA) or a promoter without a TATA-box motif (Promoter non-TATA).

Enhancer sequence prediction. Human enhancer elements were retrieved from ENCODE's SCREEN database (<https://screen.wenglab.org/>)³². Distal and proximal enhancers were combined. Enhancers were split in tissue-specific and tissue-invariant based on the vocabulary from Meuleman et al.³³ (<https://www.meuleman.org/research/dhsindex/>). Enhancers overlapping regions classified as tissue-invariant were defined as that, while all other enhancers were defined as tissue-specific. We selected 400-bp genomic sequences containing enhancers as positive examples and all 400-bp sequences not overlapping enhancers as negative examples. We created a binary classification task for the presence of enhancer elements in the sequence (Enhancer) and a multilabel prediction task with labels being tissue-specific enhancer, tissue-invariant enhancer or none (Enhancer types).

Splice site prediction. We obtained all human annotated splice sites from GENCODE²⁹ V44 gene annotation. Annotations were filtered to exclude level-3 transcripts (automated annotation), so that all training data were annotated by a human. We used `extract_splice_sites.py` from HISAT2 (ref. 56) (https://github.com/DaehwanKimLab/hisat2/blob/master/hisat2_extract_splice_sites.py) to extract respective splice site annotations. We selected 600-bp genomic sequences containing a splice acceptor or donor site in the center as positive examples and all 600-bp sequences not overlapping splice sites as negative examples. We used these sequences to create three different tasks to evaluate splice prediction: a multilabel prediction task with labels being acceptor, donor or none (Splice site all); a binary classification task for the prediction of splice acceptors (Splice acceptor); and a binary classification task for the prediction of splice donors (Splice donor).

Dataset splits and performance evaluation. Model training and performance evaluation were performed on different sets of chromosomes from the human genome. Namely, sequences from chromosomes 20 and 21 were used for the test and the remainder were used for training the different models. Sequences with Ns were removed. We balanced each dataset by subsampling the negative examples to

the same number of positive examples. To obtain small-sized datasets that can be used to benchmark any new design choice or model quickly, we further randomly subsampled examples to a maximum of 30,000 for training and 3,000 for validation and testing (see details in Supplementary Table 5).

We used a tenfold cross-validation scheme to evaluate each model, where the training set is split into ten folds, and each time we used nine of those parts for training and reserved one-tenth for validation and selecting the final checkpoint. We repeated this procedure ten times per model, each time reserving a different tenth for validation and evaluating the final performance on the same held-out test set. We used the median performance across the ten folds as the final performance metric of each model on a given task.

For a consistent performance evaluation across tasks, we used the binary or multi-class MCC as the metric as it is robust to uneven label ratios. For a final comparison across models, we calculated for each model the mean MCC across the three different categories of task (chromatin profiles, regulatory elements and splicing), where for each category we used the median MCC across tasks.

Additional downstream tasks

Chromatin profile prediction. We used the DeepSEA dataset (http://deepsea.princeton.edu/media/code/deepsea_train_bundle.v0.9.tar.gz) compiled by Zhou et al.¹⁰ for chromatin profile prediction. The dataset is composed of 2.4 million sequences, each of size 1,000 nucleotides, and associated with 919 chromatin features. These include 690 TF, 125 DNase and 104 histone features. As in the original publication, our model was trained simultaneously on the 919 classification tasks, with 919 independent classification heads, and a loss was taken as the average of the cross-entropy losses. As each label is highly unbalanced and is composed mostly of negative samples, the losses associated with positive samples are upsampled by a factor of 8. Contrary to the DeepSEA method¹⁰, 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.

SpliceAI benchmark. We used the scripts available at the Illumina Base-space platform (<https://basespace.illumina.com/projects/66029966/>) to reproduce the training dataset presented in SpliceAI³⁶. In brief, this training dataset was constructed using GENCODE V24lift37 annotations and RNA-seq data from the GTEx cohort, focusing solely on splice site annotations from the principal transcript. The training dataset comprised annotations from genes located on chromosomes 2, 4, 6, 8 and 10–22, as well as chromosomes X and Y, while annotations from genes on the remaining chromosomes, which were not paralogs, constituted the test dataset. Each sequence produced by this pipeline had a length of 15,000 bp, with the central 5,000 bp containing the sites to be predicted. Additional details about the construction of the training dataset can be found in the original publication. We adapted this original SpliceAI dataset to be able to run our models by reducing the sequence length to 6,000 bp (for NT-v1 model) and 12,000 bp (for NT-v2 model), reducing the flanking contexts but keeping the central 5,000 bp. We also removed sequences that contained Ns. When comparing against SpliceAI on the first dataset, which we refer to as SpliceAI-6k, we appended 9,000 ‘N’ nucleotides as a flanking sequence as SpliceAI is based on a model with a 15,000-bp input. When comparing against SpliceAI on the second dataset, we report the performance presented in the original publication, which, compared to this dataset, included a sequence length of 15,000 bp instead of 12,000 bp.

This task is a multilabel classification for each of the input sequence’s nucleotides similar to SpliceAI³⁶ (Fig. 5d). From each embedding outputted by the transformer model, a head predicts, for each of the six nucleotides represented by the token embedding, three label probabilities: splice acceptor, splice donor or none. The head is a simple

classification layer that predicts 18 classes (three labels for each of the six nucleotides). To ensure that each embedding is associated with a six-mer, the sequences were cut so that their length was divisible by 6. Furthermore, all sequences with Ns were removed from both training and test set, which represented a negligible portion of the data. Note that if we were to use a Byte Pair Encoding tokenizer such as DNABERT-2 (ref. 23), the number of nucleotides represented by each embedding would vary and make nucleotide-level prediction tasks substantially trickier to implement.

Enhancer activity prediction. We used the DeepSTARR enhancer activity dataset released by de Almeida et al.¹² (<https://zenodo.org/record/5502060#.Y9qO7hzMLUc>). The dataset is composed of 484,052 DNA sequences of size 249 nucleotides, each measured for their quantitative enhancer activity toward a developmental or a housekeeping promoter. We added two independent regression heads to our models to predict both enhancer activities in simultaneous. Following the methodology used elsewhere³, we chose to treat this regression task as a multilabel 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

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

Additional performance analysis

t -SNE projections of embeddings. t -distributed stochastic neighbor embedding (t -SNE) was used to reduce NT inner embeddings to two-dimensional vectors to visualize the separation of different genomic elements. NT embeddings for each genomic element were computed at several transformer layer outputs and the mean embeddings computed across the sequence locations corresponding to the element were calculated. These mean embeddings were then passed as input into a t -SNE reducer object with default parameters from the sklearn Python package³³.

Reconstruction accuracy and perplexity. We studied how pretrained 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 (we replaced the tokens at these positions with the masking (MASK) token). 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 (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 Rives⁴, and define it as the exponential of the loss function computed over the masked positions:

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

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.

Reconstruction of tokens in different genomics elements. We also performed this token-reconstruction approach across the full chromosome 22 in 6-kb windows. We only kept windows without Ns. For each window masked each token at a time, recovering the predicted probability for the original token in the sequence. We display these scores as a WashU Epigenome Browser session (<https://shorturl.at/jov28>). To obtain average token probabilities across different types of genomic elements, we retrieved gene annotation regions from Ensembl (exons, introns, splice acceptors and donors, 5' UTR, 3' UTR, transcription start site and transcription end site) (<https://www.ensembl.org/info/data/ftp/index.html>), polyA signal sites from GENCODE (<https://www.gencodegenes.org/human/>) and regulatory elements from ENCODE (enhancers, promoters and CTCF-bound sites from the SCREEN database) (https://api.wenglab.org/screen_v13/fdownloads/GRCh38-ccREs.bed).

Functional variant prioritization. To obtain genetic variants with varying levels of associated severity we used Variant Effect Prediction (VEP) software⁴⁰ 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 SNPs and filtering out variants annotated to more than one consequence (for example 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 with gene expression (eQTLs) and to meQTLs from the GRASP database⁵⁷ with a P value $< 10^{-12}$, SNPs with 'likely pathogenic' annotations from ClinVar⁵⁸ and SNPs reported in HGMD (public v.2020.4)⁵⁹. 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 1000G Project with a minor allele frequency $> 5\%$ that did not overlap with any variant reported in the dataset tested and that were within 100 kb 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 captured different aspects of the vector distances in the embedding space between those two sequences, namely: (1) the L1 distance (Manhattan); (2) the L2 distance (Euclidean); (3) the cosine similarity; and (4) 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 to the one previously described, with the primary difference being that the training and test sets were divided by chromosomes and strictly kept as nonoverlapping. Specifically, we divided the 22 chromosomes into five sets and sequentially used each of them as a test set and the four others as a training set. By fine-tuning the training set we could derive the probabilities of being a positive variant for each sequence in the test set. We used 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 (v.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.

Attention maps analysis. We analyzed how attention maps gathered from the pretrained models capture key genomic elements. We followed a methodology proposed in previous work³⁸. For a genomic element, we defined 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, in one attention head, aggregated over a dataset of nucleotide sequences $\mathbf{X}$ as:

$$p_a(f) = \frac{1}{|\mathbf{X}|} \sum_{\mathbf{x} \in \mathbf{X}} \frac{\sum_i \sum_j f(i) \mathbf{1}(\alpha(i, j) > \mu)}{\sum_i \sum_j \mathbf{1}(\alpha(i, j) > \mu)}$$

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 $p_a(f)$ 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 6 kb extracted from the Human reference genome. The average proportion of tokens belonging to each element can be found in Supplementary Table 10. For each sequence, the position of the feature within the sequence was sampled uniformly during the dataset creation. As suggested in previous work³⁸, we selected a confidence threshold $\mu = 0.3$ for all experiments.

We considered that a feature was captured by an attention head if the quantity $p_a(f)$ was significantly greater than the natural occurring frequency of the feature within the dataset (Supplementary Table 10). 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 $p_a(f)$. The total number of heads of each model was used as a Bonferroni correction to the significance level (α) of 0.05. We computed z-scores and associated P values for each head in every model for every genomic element as follows:

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

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 and n_2 is the total number of sequence positions. Attention heads with P values below the Bonferroni-corrected significance level were considered to be significant.

Prediction of important TF motif instances from DeepSTARR data.

We retrieved experimental mutagenesis data from the DeepSTARR dataset¹², where individual TF motif instances are mutated and their impact is measured in the activity of developmental and housekeeping enhancers. We assessed the performance of the fully fine-tuned NT 2.5B Multispecies model (as it was the model with the highest test set performance) to predict the contribution of each TF motif instance by predicting the activity of the wild-type and respective motif-mutant sequence and calculating their $\log_2$ fold change. We compared our predicted mutation effects with the ones predicted by the original method DeepSTARR and the experimentally derived $\log_2$ fold changes.

Reporting summary

Further information on research design is available in the Nature Portfolio Reporting Summary linked to this article.

Data availability

The NT pre-training sequences were obtained from publicly available resources. The 1000G 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 were from <https://ftp.ncbi.nlm.nih.gov/genomes/refseq/>. Gene annotations were obtained from the GENCODE (<https://www.encodegenes.org/>) and Ensembl databases (<https://www.ensembl.org/>). Variant effects predictions were obtained using the 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/), GRASP (<https://grasp.nhlbi.nih.gov>) and HGMD (public v.2020.4) through Ensembl Biomart. The NT downstream tasks were curated from the following publicly available resources: ENCODE^{31–33} (<https://www.encodeproject.org/>), <https://screen.wenglab.org/> and <https://www.meuleman.org/research/dhsindex/>), the Eukaryotic Promoter Database³⁰ (<https://epd.expasy.org/epd/>) and GENCODE²⁹ (Supplementary Table 5). We made available HuggingFace versions of our multispecies and human pre-training datasets (<https://huggingface.co/collections/InstaDeepAI/nucleotide-transformer-65099cddde13ff96230f2e592>) as well as our downstream tasks benchmark (https://huggingface.co/datasets/InstaDeepAI/nucleotide_transformer_downstream_tasks_revised). We have established an interactive leaderboard (https://huggingface.co/spaces/InstaDeepAI/nucleotide_transformer_benchmark) containing results for all models across each task to facilitate comparisons. We have also created an interactive browser session with the pre-trained model token probabilities across the full chromosome 22 on the WashU Epigenome Browser at <https://shorturl.at/jov28>.

Code availability

Model code and weights of the pre-trained transformer models as well as inference code in Jax are available for research purposes at <https://github.com/instadeepai/nucleotide-transformer>. HuggingFace versions of the models in PyTorch can be found at <https://huggingface.co/collections/InstaDeepAI/nucleotide-transformer-65099cddde13ff96230f2e592>. Example notebooks are available on HuggingFace at <https://huggingface.co/docs/transformers/notebooks#pytorch-bio>.

References

49. Wang, A. et al. Superglue: a stickier benchmark for general-purpose language understanding systems. in *33rd Conference on Neural Information Processing Systems* https://papers.nips.cc/paper_files/paper/2019/file/4496bf24afe7fab6f046bf4923da8de6-Paper.pdf (NeurIPS, 2019).
50. Hendrycks, D. & Gimpel, K. Gaussian error linear units (gelus). Preprint at <https://arxiv.org/abs/1606.08415> (2016).
51. Su, J. et al. Roformer: enhanced transformer with rotary position embedding. *Neurocomputing* **568**, 127063 (2024).
52. Kingma, D. P. & Ba, J. Adam: a method for stochastic optimization. Preprint at <https://arxiv.org/abs/1412.6980v5> (2015).
53. Pedregosa, F. et al. Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011).
54. Bergstra, J., Bardenet, R., Bengio, Y. & Kégl, B. Algorithms for hyper-parameter optimization. in *Advances in Neural Information Processing Systems 24* https://papers.nips.cc/paper_files/paper/2011/file/86e8f7ab32cfd12577bc2619bc635690-Paper.pdf (NeurIPS, 2011).
55. Byrska-Bishop, M. et al. High-coverage whole-genome sequencing of the expanded 1000 genomes project cohort including 602 trios. *Cell* **185**, 3426–3440 (2022).
56. Kim, D., Paggi, J. M., Park, C., Bennett, C. & Salzberg, S. L. Graph-based genome alignment and genotyping with hisat2 and hisat-genotype. *Nat. Biotechnol.* **37**, 907–905 (2019).
57. Leslie, R., O'Donnell, C. J. & Johnson, A. D. GRASP: analysis of genotype–phenotype results from 1390 genome-wide association studies and corresponding open access database. *Bioinformatics* **30**, i185–i194 (2014).
58. Landrum, M. J. et al. Clinvar: improving access to variant interpretations and supporting evidence. *Nucleic Acids Res.* **46**, D1062–D1067 (2018).
59. Stenson, P. D. et al. The Human Gene Mutation Database (HGMD®): optimizing its use in a clinical diagnostic or research setting. *Hum. Genet.* **139**, 1197–1207 (2020).

Acknowledgements

We thank members of the Rost laboratory, particularly T. Olenyi, I. Koludarov and B. Rost for constructive discussions that helped identify interesting research directions. We also thank M. 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.

Author contributions

M.L. and T.P. conceived the research idea. H.D.-T., L.G., J.M.-R., A.H.G., F.O., C.D., E.T., B.P.d.A., H.S. and G.R. performed the analyses. N.L.C., M.S., K.B., M.L. and T.P. provided advice on study design and analyses. J.M.-R., B.P.d.A., M.L. and T.P. wrote the paper with input from all co-authors.

Competing interests

H.D.-T., L.G., J.M.-R., N.L.C., F.O., E.T., B.P.d.A., G.R., M.S., K.B., M.L. and T.P. are employees of InstaDeep. A.H.G. and C.D. are employees of Nvidia.

Additional information

Supplementary information The online version contains supplementary material available at <https://doi.org/10.1038/s41592-024-02523-z>.

Correspondence and requests for materials should be addressed to Marie Lopez or Thomas Pierrot.

Peer review information *Nature Methods* thanks the anonymous reviewers for their contribution to the peer review of this work. Primary Handling Editor: Lin Tang, in collaboration with the *Nature Methods* team. Peer reviewer reports are available.

Reprints and permissions information is available at www.nature.com/reprints.

Reporting Summary

Nature Portfolio wishes to improve the reproducibility of the work that we publish. This form provides structure for consistency and transparency in reporting. For further information on Nature Portfolio policies, see our [Editorial Policies](#) and the [Editorial Policy Checklist](#).

Statistics

For all statistical analyses, confirm that the following items are present in the figure legend, table legend, main text, or Methods section.

n/a	Confirmed
☐	☒ The exact sample size (<i>n</i>) for each experimental group/condition, given as a discrete number and unit of measurement
☒	☐ A statement on whether measurements were taken from distinct samples or whether the same sample was measured repeatedly
☐	☒ The statistical test(s) used AND whether they are one- or two-sided <i>Only common tests should be described solely by name; describe more complex techniques in the Methods section.</i>
☐	☒ A description of all covariates tested
☐	☒ A description of any assumptions or corrections, such as tests of normality and adjustment for multiple comparisons
☐	☒ A full description of the statistical parameters including central tendency (e.g. means) or other basic estimates (e.g. regression coefficient) AND variation (e.g. standard deviation) or associated estimates of uncertainty (e.g. confidence intervals)
☐	☒ For null hypothesis testing, the test statistic (e.g. <i>F</i> , <i>t</i> , <i>r</i>) with confidence intervals, effect sizes, degrees of freedom and <i>P</i> value noted <i>Give P values as exact values whenever suitable.</i>
☒	☐ For Bayesian analysis, information on the choice of priors and Markov chain Monte Carlo settings
☒	☐ For hierarchical and complex designs, identification of the appropriate level for tests and full reporting of outcomes
☐	☒ Estimates of effect sizes (e.g. Cohen's <i>d</i> , Pearson's <i>r</i>), indicating how they were calculated

Our web collection on [statistics for biologists](#) contains articles on many of the points above.

Software and code

Policy information about [availability of computer code](#)

Data collection	python 3.9, pandas 1.5.2, torch 2.0.1
Data analysis	Model code and weights of the pre-trained transformer models as well as inference code in Jax are available for research purposes at https://github.com/instadeepai/nucleotide-transformer . HuggingFace versions of the models, in PyTorch, can be found at https://huggingface.co/InstaDeepAI . Example note- books are available on HuggingFace at https://huggingface.co/docs/transformers/notebooks#pytorch-bio .

For manuscripts utilizing custom algorithms or software that are central to the research but not yet described in published literature, software must be made available to editors and reviewers. We strongly encourage code deposition in a community repository (e.g. GitHub). See the Nature Portfolio [guidelines for submitting code & software](#) for further information.

Data

Policy information about [availability of data](#)

All manuscripts must include a [data availability statement](#). This statement should provide the following information, where applicable:

- Accession codes, unique identifiers, or web links for publicly available datasets
- A description of any restrictions on data availability
- For clinical datasets or third party data, please ensure that the statement adheres to our [policy](#)

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.encodegenes.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 Nucleotide Transformer downstream tasks were curated from the following publicly available resources: ENCODE (<https://www.encodeproject.org/>, <https://screen.wenglab.org/>, <https://www.meuleman.org/research/dhsindex/>), the Eukaryotic Promoter Database (<https://epd.expasy.org/epd/>) and GENCODE. We make available HuggingFace versions of our multispecies and human pre-training datasets (<https://huggingface.co/collections/InstaDeepAI/nucleotide-transformer-65099cdde13ff96230f2e592>) as well as our downstream tasks benchmark (https://huggingface.co/datasets/InstaDeepAI/nucleotide_transformer_downstream_tasks_revised). We have established an interactive leaderboard containing results for all models across each task to facilitate comparisons, available at https://huggingface.co/spaces/InstaDeepAI/nucleotide_transformer_benchmark. We have also created an interactive browser session with the pre-trained model token probabilities across the full chromosome 22 on the WashU Epigenome Browser at <https://shorturl.at/jov28>.

Research involving human participants, their data, or biological material

Policy information about studies with [human participants or human data](#). See also policy information about [sex, gender \(identity/presentation\), and sexual orientation](#) and [race, ethnicity and racism](#).

Reporting on sex and gender	N/A
Reporting on race, ethnicity, or other socially relevant groupings	N/A
Population characteristics	N/A
Recruitment	N/A
Ethics oversight	N/a

Note that full information on the approval of the study protocol must also be provided in the manuscript.

Field-specific reporting

Please select the one below that is the best fit for your research. If you are not sure, read the appropriate sections before making your selection.

☒ Life sciences ☐ Behavioural & social sciences ☐ Ecological, evolutionary & environmental sciences

For a reference copy of the document with all sections, see [nature.com/documents/nr-reporting-summary-flat.pdf](https://www.nature.com/documents/nr-reporting-summary-flat.pdf)

Life sciences study design

All studies must disclose on these points even when the disclosure is negative.

Sample size	We used the 1000 Genomes Project database which contains 3202 high-coverage human genomes, originating from 27 geographically structured populations of African, East Asian and European ancestry. The multispecies database contained 850 species' whole genomes retrieved from NCBI.
Data exclusions	No data were excluded from the analyses.
Replication	We implemented a 10-fold cross-validation strategy in our benchmark.
Randomization	We splatted sequences into training and test held-out sets.
Blinding	Not applicable, we used held-out test sets.

Reporting for specific materials, systems and methods

We require information from authors about some types of materials, experimental systems and methods used in many studies. Here, indicate whether each material, system or method listed is relevant to your study. If you are not sure if a list item applies to your research, read the appropriate section before selecting a response.

Materials & experimental systems

Methods

n/a	Involved in the study
☒	☐ Antibodies
☒	☐ Eukaryotic cell lines
☒	☐ Palaeontology and archaeology
☒	☐ Animals and other organisms
☒	☐ Clinical data
☒	☐ Dual use research of concern
☒	☐ Plants

n/a	Involved in the study
☒	☐ ChIP-seq
☒	☐ Flow cytometry
☒	☐ MRI-based neuroimaging

Plants

Seed stocks

N/A

Novel plant genotypes

N/A

Authentication

N/A