GENATATOR: DE NOVO GENE ANNOTATION WITH DNA LANGUAGE MODEL

Aleksei Shmelev[*], Artem Shadskiy[*], Yuri Kuratov, Olga Kardymon and Veniamin Fishman

AIRI, Moscow, Russia
Institute of Cytology and Genetics, Novosibirsk, Russia
Moscow Institute of Physics and Technology, Dolgoprudny, Russia
Sirius University, Sochi, Russia
HSE University, Moscow, Russia
minja-f@yandex.com, kardymon@airi.net

Mikhail Burtsev

London Institute for Mathematical Sciences, London, UK
mb@lims.ac.uk

ABSTRACT

Inference of gene structure and location based on genome sequences, also known as *de novo* gene annotation, is a critical first step in biological research. However, rules of encoding gene structure in the DNA sequence are complex and poorly understood, often necessitating the use of costly transcriptomic data to achieve accurate gene annotation. Here, we present GENATATOR — Genome Annotator Using the GENA DNA Language Model — an advanced machine learning tool for inferring gene annotations directly from DNA sequences. Unlike previous approaches that rely on explicitly defined gene segmentation rules derived from protein-coding sequences, GENATATOR learns how to infer gene structure directly from the data. This enables GENATATOR to perform correct segmentation for previously untraceable class of non-coding transcripts and identify subset of protein-coding genes missed by other models, achieving top performance in the gene segmentation benchmarks. Finally, with in-depth analysis of GENATATOR's model embeddings and predictions, we reveal how DNA language models utilize memory to learn the biological rules underlying gene encoding.

1 INTRODUCTION

The rapid evolution of DNA sequencing technologies, such as third-generation sequencing and Hi-C, has led to an exponential growth in the availability of genome assemblies across the tree of life. This genomic data is invaluable for fundamental research, biotechnology, and biomedicine. Yet, raw DNA sequences alone are insufficient for most applications, requiring genome annotation — identification of specific functional elements in the genome. Among various genomic annotations, genes annotation is the most important, since it identifies genes and reveals their structural elements, which is critical for almost all downstream applications.

Genes, which are DNA segments templating RNA synthesis (transcription), are categorized as protein-coding or non-coding. Protein-coding genes produce RNA for translation, while non-coding genes, with long non-coding RNA (lncRNA) genes being particularly prevalent in eukaryotes and serve key regulatory roles Ferrer & Dimitrova (2024), yield functional RNA. Both types include exons and introns, but mature RNA retains only exons due to intron excision Shine et al. (2024), and in protein-coding genes, exons are split into the CDS and UTRs (5'- and 3'-UTR), framing gene segmentation as a multi-label classification problem.

[*]These authors contributed equally to this work.

Existing gene annotation tools are designed to detect statistical patterns typical for protein-coding genes, such as the presence of a reading frame—sequences of 3-nucleotide blocks that encode peptides Gabriel et al. (2024). Therefore, these methods are not applicable to lncRNA and other non-coding genes identification. Furthermore, this limits the ability of the tools to detect protein-coding genes that have undergone mutations resulting in disrupted protein reading frames. Although these genes may no longer encode functional proteins, identifying them is essential for understanding genomic and evolutionary processes Cheetham et al. (2020).

In this study, we demonstrate that DNA language models are effective at addressing the gene segmentation problem. By optimizing training dataset, model architecture, training regimes, and hyperparameters, we have developed GENATATOR, a novel tool based on DNA language models for *ab initio* genome annotation. GENATATOR surpasses previous solutions, inferring the structures of both protein-coding and non-coding genes. Our analysis reveals how GENATATOR learns DNA grammar and use recurrent memory in processing of long genomic sequences.

2 Related work

Historically, *ab initio* gene annotation has relied on Hidden Markov Models (HMMs) to capture protein-coding gene features Stanke et al. (2004), but these models, explicitly setting model parameters to fit protein-coding grammar limits identification of UTRs, non-coding, and mutated genes. Recent advances using deep neural networks (DNNs) have led to models such as Helixer, a CNN that classifies sequences into Intergenic, UTR, CDS, and Intron categories Stiehler et al. (2020), and Tiberius, which combines trainable HMM layers with CNNs to achieve state-of-the-art protein-coding annotation Gabriel et al. (2024). However, their reliance on protein-coding grammar and limited input lengths (up to 10Kb in Tiberius) pose challenges given human genes typically average 30Kb and can exceed 1Mb. In contrast, transformer-based DNA language models (DNA-LMs) have emerged as potent tools, matching or surpassing previous methods in detecting splice sites, promoters, polyA signals and other crucial functional elements Fishman et al. (2025); Dalla-Torre et al. (2024); Marchal (2024); Zhou et al. (2023). Notably, SegmentNT, a refined variant of the Nucleotide Transformer, has demonstrated encouraging results in gene annotation tasks de Almeida et al. (2024).

Among the available models, we focus on the recently developed GENA-LMs, since they are able to process exceptionally long DNA sequences Fishman et al. (2025). GENA-LMs' capacity for handling sequence lengths is on par with the scale of human genes Fishman et al. (2025), making GENA-LMs particularly suited for in-depth gene annotation tasks.

GENA-LMs comprise a family of DNA language models that vary in complexity, with parameter counts ranging from $\sim$110M for the base to $\sim$336M for the large model. All GENA-LMs utilize a BERT-based transformer architecture, accommodating input lengths from 512 to 4096 BPE tokens, which translates to approximately 4kb to 32Kb of DNA nucleotides. For processing longer inputs, GENA-LMs can be augmented with the Recurrent Memory Transformer (RMT) Bulatov et al. (2022). RMT handles extended inputs by sequentially processing them in chunks or segments, utilizing dedicated memory tokens to facilitate the transfer and retention of information across segments. Benchmarking studies of RMT Bulatov et al. (2024); Kuratov et al. (2024a) have shown its efficacy in managing exceptionally long sequences up to 10M tokens. When integrated with GENA-LMs, RMT improves performance in various biological applications Kuratov et al. (2024b). Therefore, we decided that GENA-LM is an optimal starting point for gene annotator development.

3 GENATATOR Gene Annotation Model

We started with the fine-tuning base version of GENA-LM (110M parameters; input length 512 tokens, $\sim$ 4 Kb) (Fig. 1A). This initial model was fine-tunned for segmentation of human genes. For this task, we defined five classes: 5'-UTR, exon, intron, 3'-UTR, and CDS. The BPE tokens in our model contain multiple nucleotides (8-9 in average), which can span across different classes, thus necessitating the assignment of multiple labels per token. Additionally, a single gene can be processed differently within a cell, leading to the production of multiple alternative RNA molecules, or transcripts. Therefore, one genomic sequence can be assigned with different class labels in samples derived from alternative transcripts.

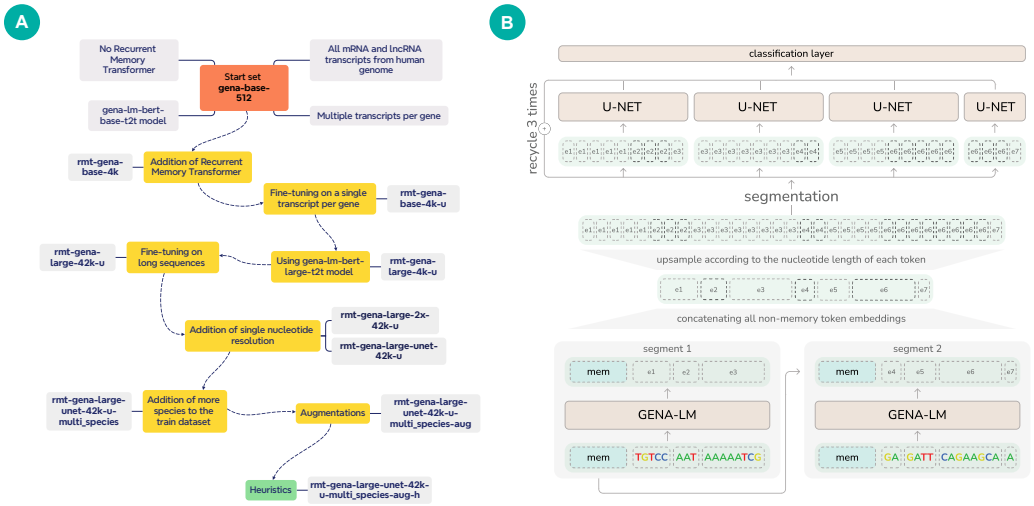

Figure 1: Incremental updates of dataset, architecture, and training pipeline result in high-quality gene annotation model, GENATATOR. A. Key modifications during GENATATOR development. B. The scheme of the final GENATATOR architecture.

To evaluate the model, we measure precision, recall, and f1-score at the interval (exon or CDS) level (Fig. 2A and Appendix C). In this analysis, a continuous set of tokens positive for exon or CDS label is designated as exon or CDS interval, respectively. A predicted interval is considered a true positive only if there is complete reciprocal overlap between the target and predicted intervals. Additionally, we employ gene-level metrics, considering a prediction to be a true positive when the sequence of predicted exons exactly matches any transcript of the target gene. In addition, we use AUC as per-token metrics (Appendix C).

The baseline `gena-base-512` model showed poor performance at the exon and gene levels (Fig. 2A). Specifically, the model successfully identified only a small subset of protein-coding exons, failed to report any lncRNA exons, and was unable to completely reconstruct any genes.

The base version of GENA-LM, with input size of 512 BPE tokens, provides limited context. However, this can be significantly expanded with RMT. We trained the model with RMT to handle sequences up to 4096 BPE tokens in length, equivalent to 8 segments or about 32 Kb of DNA nucleotides. The model was specifically trained to process each sample from the gene start and proceed no further than 4096 BPE tokens. This approach, which incorporates a broader genomic context, markedly improved all model metrics (Fig. 2A, `rmt-gena-base-4k`), indicating that global context is beneficial for better performance.

Despite these improvements, a substantial fraction of exons was still not accurately identified. Comparing model predictions with ground truth, we noted that transcripts of a single gene often share similar structures but can differ in the start and end positions of reading, as well as the inclusion or exclusion of alternative exons within the gene body. We found that nearly 47% of our training dataset consisted of samples that had identical sequences of BPE tokens as input to the model, but different target labels. To address this issue, we filtered the dataset to represent each gene with the only transcript having the longest cumulative exon length. The training was carried out using the same setup as in the previous models and show that using on unique transcripts significantly improved the accuracy of the model (Fig. 2A, `rmt-gena-base-4k-u`).

We started our experiments with the base 110M model, and replacing it with 336M GENA-LM large (`rmt-gena-large-4k-u`) led to further improvements across all metrics. Although the best-performing version of the smaller model, `rmt-gena-base-4k-u`, was only able to accurately segment 4 genes, the larger model, `rmt-gena-large-4k-u`, correctly segmented 44 genes: 27 protein coding and 17 encoding lncRNAs, representing about 5% of all genes in the validation dataset.

We next decided to further scale the length of inputs processed by the model, extending the training fragments up to 250Kb (80 RMT segments). Guided by the curriculum learning approach Bulatov

et al. (2022), we progressively increased the input lengths in our training, starting with 8 segments (4 Kb), moving to 32 segments (128 Kb), and finally reaching 80 segments (250 Kb), while also incorporating shortened inputs at each training step. As shown in Fig. 2A, each increment in context length considerably enhanced the model's performance (models `rmt-gena-large-4k-u`, `rmt-gena-large-16k-u`, `rmt-gena-large-42k-u`), particularly improving the predictions of long transcripts. This aligns with the established understanding of long-range dependencies in splicing machinery Statello et al. (2021). Furthermore, the increase in transcript chunk lengths used for training also enlarges the training dataset, further contributing to improved model performance.

In DNA sequence, shifting the boundary of an exon by just one nucleotide can lead to a reading frame shift, altering the entire downstream protein sequence. Given that multiple tokens may contain boundaries of different DNA functional elements, it is critically important for the model to perform gene segmentation at a nucleotide resolution. For example, while the `rmt-gena-large-42k-u` model can correctly segment 106 genes at token-resolution, the accuracy significantly drops when these predictions are converted to nucleotide resolution—with only 23 of them remaining correctly segmented (Fig. 2B).

Inspired by the SegmentNT architecture, which integrates a DNA language model with a U-NET head de Almeida et al. (2024), we adapted our model to nucleotide-level resolution by upsampling the embeddings produced for each token. We merge the upsampled embeddings with learnable embeddings of individual DNA nucleotides, followed by processing through a nucleotide-level classification head. We evaluated two distinct architectures for the nucleotide-level classification head: a transformer-based head (`rmt-gena-large-2x-42k-u`) and a U-NET-based head (`rmt-gena-large-unet-42k-u`). For these models, and all subsequent iterations, we report performance metrics at nucleotide resolution. Additionally, we include the BUSCO score Manni et al. (2021), which is common in genomics and reflects the recall rate for a subset of evolutionary-conserved protein-coding genes.

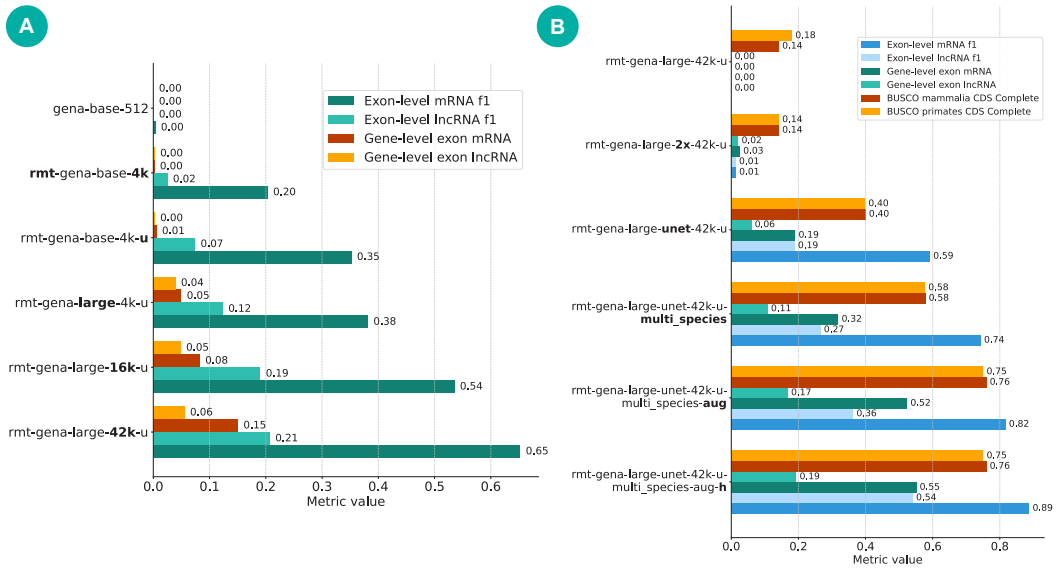

Figure 2: (A) Context length, dataset structure, and model size are essential for gene segmentation task. Models comparison are for token-level classification, all labels coerced to the token-resolution before computing metrics. (B) Improved architecture allows gene segmentation at nucleotide resolution, significantly enhancing gene annotation performance. The bar chart compares models with all metrics provided at nucleotide resolution. For `rmt-gena-large-42k-u`, predictions made at token-resolution were converted to nucleotide resolution as a baseline of token-level predictions. For `rmt-gena-large-unet-42k-u-multi_species-aug-h` we manually removed predicted exons with non-canonical splice sites.

Benchmarking results show that `rmt-gena-large-2x-42k-u` model can't identify exon-intron boundaries when they fall within a single token (Fig. 2B). In contrast, the model incorporating a U-NET architecture (`rmt-gena-large-unet-42k-u`) demonstrates substantially better performance in these cases (Fig. 2B). The addition of the U-NET head not only enhances the resolution of the model but also allows detecting exons that were missed by token-resolution model. At nucleotide resolution, this model identifies 130 genes, showing a significant improvement over the 106 genes identified at token-resolution (Fig. 2B).

Comparison of train and validation metrics (mean AUC across all classes: 0.98 for train, 0.65 for validation suggests that the quantity of training data is a bottleneck. To address this, we decided to expand our training dataset to include genomes from an additional 38 mammalian species, incorporating all chromosomes from these organisms. The validation dataset remained unchanged. By training this enhanced model with RMT augmentation, we achieved significant improvements across all metrics (Fig. 2B, `rmt-gena-large-unet-42k-u-multi_species`). This approach enabled the model to identify 221 genes (23% of the validation dataset).

We next applied several augmentations to increase model performance. First, we used "recycling" mechanism, motivated by models such as AlphaFold and AlphaFold2 that merge intermediate embeddings with original inputs in repeated cycles of inference Jumper et al. (2021); Yang et al. (2023); Koehler et al. (2024). Since U-NET produces outputs with the same dimensionality as its inputs, we were able to feed the U-NET output embeddings back into it up to three times, merging them each time with the embeddings from our RMT module. Second, we apply augmentation based on gene directionality. We note that genes are distributed randomly across two DNA strands, therefore approximately half of the genes are presented in the reverse orientation, from the transcription stop signal to the transcription start signal, during training phase. Using the model's capacity to analyze genes in both directions, we process original and reverse-complement of input sequences and average obtained predictions. These augmentations have demonstrated notable improvements in model performance, as detailed in Fig. 2B (`rmt-gena-large-unet-42k-u-multi_species-aug`).

Also, analysis of predictions and embeddings of memory tokens reveals approximately 88% compliance with NMD rules and accurate learning of gene type, orientation and exon-intron structure (Appendix E).

In summary, applying GENA-LM large model, augmented with recurrent memory to the optimized multispecies dataset, using training with specific schedule and hyperparameters, and incorporating test-time augmentation, we have developed GENATATOR - a robust model capable of high-precision gene annotation tasks. GENATATOR identify protein coding exons with f1 = 0.82, lncRNA exons with f1 = 0.36, and correctly segments 36% of genes in the validation dataset. Moreover, during the analysis of GENATATOR errors, we filtered out predicted exons with non-canonical boundary dinucleotides, which improved accuracy up to 40% (Appendix D).

4 BENCHMARKING GENATATOR AGAINST OTHER GENE-ANNOTATION TOOLS

We evaluated the performance of the GENATATOR model in comparison with state-of-the-art models that include HMMs (AUGUSTUS, Stanke et al. (2004)), hybrid DNN+HMM models (Tiberius, Gabriel et al. (2024)), and DNA-LMs (SegmentNT, de Almeida et al. (2024)).

Initially, we assessed the models' ability to recognize exons and CDS (Table 1). In the exon identification challenge, GENATATOR demonstrated superior performance, achieving an f1-score of 0.81 (0.89 for protein-coding genes and 0.57 for lncRNA). In contrast, the Tiberius model, while excelling in CDS prediction, showed limitations in identifying non-coding regions within exons. This specificity likely contributes to its lower overall performance in the exon-level benchmark. For CDS prediction, although GENATATOR outperformed AUGUSTUS and SegmentNT with higher recall, it was less effective than Tiberius, with f1-scores of 0.78 (GENATATOR) and 0.86 (Tiberius). This gap in performance is because GENATATOR often fails to precisely locate start and stop codons—the initial and terminal trinucleotides of the CDS, although correctly identify overall exon-intron structure of the gene.

In addition, we evaluated the performance of each model at detecting entire genes. Here, a gene is considered detected if at least one transcript of the gene is correctly predicted, with all its exons

Table 1: Exon- and CDS-level f1-score (see Appendix B for more information on metrics) and gene-level (metrics calculated for protein-coding genes and lncRNA genes on holdout gene set from human chromosome 20) segmentation benchmarks.

	exon- and CDS-level				gene-level			
	mRNA		lncRNA	allRNA	mRNA		lncRNA	allRNA
	exon	CDS	exon	exon	exon	CDS	exon	exon
GENATATOR	**0.8877**	0.7753	**0.5407**	**0.8114**	**55%**	1%	**19%**	**39%**
SegmentNT	0.4231	0.1620	0.0074	0.1627	18%	0%	0%	10%
AUGUSTUS	0.6345	0.7211	–	–	2%	19%	–	–
Tiberius	0.7179	**0.8593**	–	–	0%	**69%**	–	–

accurately identified. According to our benchmarks, GENATATOR successfully identified 55% of protein-coding genes and 19% of lncRNA genes in the validation dataset, surpassing SegmentNT in both categories.

Unlike GENATATOR, tools like Tiberius, which focus on CDS, can not detect non-coding parts of exons (UTRs). To address this, we introduced a specialized challenge for protein-coding genes: a gene is considered detected only if at least one transcript of the gene has all CDS segments correctly predicted. Under this criterion, which focus on CDS identification rather than exon recognition, GENATATOR's gene-level metrics show a significant decrease, mirroring its lower performance in the CDS category. Tiberius, which identified 375 genes and holds the highest CDS-based score. However, this performance is below GENATATOR's exon-based gene segmentation, allowing to identify 385 genes. Highest performance of GENATATOR stems from its capacity to identify lncRNA, which are completely missed by Tiberius, and improved detection of protein-coding genes compared to SegmentNT. These results highlight GENATATOR as the most comprehensive gene segmentation model available, although it exhibits some limitations in its ability to accurately predict CDS segments.

The common metric for assessing the completeness of genome annotation is BUSCO Manni et al. (2021). To compute BUSCO, the predicted exon-intron structure of a gene is used to generate an amino acid sequence, which is then compared to a set of proteins that are specific to a particular taxonomy group. The results of BUSCO are presented as a number of proteins that were identified from a selected dataset. These proteins are divided into two categories: Complete and Fragmented, where fragmented proteins have some segments missing. For the mammalian BUSCO dataset, GENATATOR fully identifies 77% of the genes. In this set, GENATATOR outperforms all models except for Tiberius, which identifies 86% of the genes (Table 2). Similar results were obtained for the primates BUSCO dataset.

Table 2: BUSCO completeness computed on hold-out gene set (human chromosome 20). C — complete, F — fragmented.

	MAMMALIA				PRIMATES			
	EXON		CDS		EXON		CDS	
	C	F	C	F	C	F	C	F
GENATATOR	212	35	209	39	300	48	307	49
SEGMENTNT	167	47	55	53	239	62	81	76
AUGUSTUS	194	27	192	30	278	46	279	54
TIBERIUS	**236**	7	**236**	7	**356**	6	**356**	6
REFERENCE	275	3	275	3	409	4	409	4

Finally, we compared the sets of genes reported by all models (Fig. 5A and Fig. 5B in Appendix F). We found that each model has specific set of genes that can not be correctly segmented by any other model. Within protein-coding genes, GENATATOR and Tiberius display the largest subsets of such genes (47 and 93, respectively). For a complete gene set that includes both protein-coding and lncRNA, 131 genes were correctly segmented by GENATATOR only, while Tiberius processed 93 genes unavailable by other models, AUGUSTUS and SegmentNT - 3 genes.

Overall, these benchmarks show that GENATATOR provides correct segmentation for the largest subset of genes, outperforming all other models.

5 CONCLUSIONS

In this work, we developed GENATATOR, a state-of-the-art tool for gene annotation. We demonstrated that GENATATOR identifies a large set of genes missed by other models and uniquely enables the simultaneous annotation of both lncRNA and protein-coding genes. Our analysis highlights key factors contributing to GENATATOR's efficiency: utilizing long-range contextual information; increasing model size; bridging the gap between token- and nucleotide-level resolution; incorporating multispecies data during model training.

We note that DNA segmentation is essential for numerous biological applications beyond gene annotation. The identification of promoters, enhancers, polycomb-repressed regions, transcription factor binding sites (Mitra et al., 2024), other regulatory sequences, and chromatin states (Ernst & Kellis, 2017), topologically-associated domains and compartments (Belokopytova & Fishman, 2021), lamina-associated domains (Briand & Collas, 2020), and replication domains (Zhao et al. (2017)) — among other genomic challenges — can all be formulated as segmentation tasks. We believe that the architectures developed in this work can be successfully applied to address these challenges in the future, opening new perspectives in genomic applications of DNA language models.

REFERENCES

Polina Belokopytova and Veniamin Fishman. Predicting genome architecture: challenges and solutions. *Frontiers in genetics*, 11:617202, 2021.

Nolwenn Briand and Philippe Collas. Lamina-associated domains: peripheral matters and internal affairs. *Genome biology*, 21:1–25, 2020.

Aydar Bulatov, Yury Kuratov, and Mikhail Burtsev. Recurrent memory transformer. *Advances in Neural Information Processing Systems*, 35:11079–11091, 2022.

Aydar Bulatov, Yuri Kuratov, Yermek Kapushev, and Mikhail Burtsev. Beyond attention: Breaking the limits of transformer context length with recurrent memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 17700–17708, 2024.

Seth W Cheetham, Geoffrey J Faulkner, and Marcel E Dinger. Overcoming challenges and dogmas to understand the functions of pseudogenes. *Nature Reviews Genetics*, 21(3):191–201, 2020.

Hugo Dalla-Torre, Liam Gonzalez, Javier Mendoza-Revilla, Nicolas Lopez Carranza, Adam Henryk Grzywaczewski, Francesco Oteri, Christian Dallago, Evan Trop, Bernardo P de Almeida, Hassan Sirelkhatim, et al. Nucleotide transformer: building and evaluating robust foundation models for human genomics. *Nature Methods*, pp. 1–11, 2024.

Bernardo P de Almeida, Hugo Dalla-Torre, Guillaume Richard, Christopher Blum, Lorenz Hexemer, Maxence Gélard, Javier Mendoza-Revilla, Priyanka Pandey, Stefan Laurent, Marie Lopez, et al. Segmentnt: annotating the genome at single-nucleotide resolution with dna foundation models. *bioRxiv*, pp. 2024–03, 2024.

Jason Ernst and Manolis Kellis. Chromatin-state discovery and genome annotation with chromhmm. *Nature protocols*, 12(12):2478–2492, 2017.

J. Ferrer and N. Dimitrova. Transcription regulation by long non-coding rnas: mechanisms and disease relevance. *Nature Reviews Molecular Cell Biology*, 25:396–415, 2024.

Veniamin Fishman, Yuri Kuratov, Aleksei Shmelev, Maxim Petrov, Dmitry Penzar, Denis Shepelin, Nikolay Chekanov, Olga Kardymon, and Mikhail Burtsev. Gena-lm: a family of open-source foundational dna language models for long sequences. *Nucleic Acids Research*, 53(2):gkae1310, 2025.

Lars Gabriel, Felix Becker, Katharina J Hoff, and Mario Stanke. Tiberius: end-to-end deep learning with an hmm for gene prediction. *Bioinformatics*, 40(12):btae685, 2024.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *nature*, 596(7873):583–589, 2021.

Gregor Koehler, Tassilo Wald, Constantin Ulrich, David Zimmerer, Paul F Jaeger, Jörg KH Franke, Simon Kohl, Fabian Isensee, and Klaus H Maier-Hein. Recyclenet: Latent feature recycling leads to iterative decision refinement. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 810–818, 2024.

Yuri Kuratov, Aydar Bulatov, Petr Anokhin, Ivan Rodkin, Dmitry Sorokin, Artyom Sorokin, and Mikhail Burtsev. Babilong: Testing the limits of llms with long context reasoning-in-a-haystack. *arXiv preprint arXiv:2406.10149*, 2024a.

Yuri Kuratov, Aleksei Shmelev, Veniamin Fishman, Olga Kardymon, and Mikhail Burtsev. Recurrent memory augmentation of gena-lm improves performance on long dna sequence tasks. In *ICLR 2024 Workshop on Machine Learning for Genomics Explorations*, 2024b.

Mosè Manni, Matthew R Berkeley, Mathieu Seppey, and Evgeny M Zdobnov. Busco: assessing genomic data quality and beyond. *Current Protocols*, 1(12):e323, 2021.

Iris Marchal. Evo learns biological complexity from the molecular to genome scale. *nature biotechnology*, 42(12):1793–1793, 2024.

Raktim Mitra, Jinsen Li, Jared M. Sagendorf, Yibei Jiang, Ari S. Cohen, Tsu-Pei Chiu, Cameron J. Glasscock, and Remo Rohs. Geometric deep learning of protein–dna binding specificity. *Nature Methods*, 21(9):1674–1683, Sep 2024. ISSN 1548-7105. doi: 10.1038/s41592-024-02372-w. URL https://doi.org/10.1038/s41592-024-02372-w.

M. Shine, J. Gordon, and L. et al Schärfen. Co-transcriptional gene regulation in eukaryotes and prokaryotes. *Nature Reviews Molecular Cell Biology*, 25:534–554, 2024.

Mario Stanke, Rasmus Steinkamp, Stephan Waack, and Burkhard Morgenstern. Augustus: a web server for gene finding in eukaryotes. *Nucleic acids research*, 32(suppl_2):W309–W312, 2004.

L. Statello, CJ. Guo, and LL. et al Chen. Gene regulation by long non-coding rnas and its biological functions. *Nature Reviews Molecular Cell Biology*, 22:96–118, 2021.

Felix Stiehler, Marvin Steinborn, Stephan Scholz, Daniela Dey, Andreas PM Weber, and Alisandra K Denton. Helixer: cross-species gene annotation of large eukaryotic genomes using deep learning. *Bioinformatics*, 36(22-23):5291–5298, 2020.

F. Supek, B. Lehner, and R.G.H. et al Lindeboom. To nmd or not to nmd: Nonsense-mediated mrna decay in cancer and other genetic diseases. *Trends in Genetics*, 37:657–668, 2021.

Zhenyu Yang, Xiaoxi Zeng, Yi Zhao, and Runsheng Chen. Alphafold2 and its applications in the fields of biology and medicine. *Signal Transduction and Targeted Therapy*, 8(1):115, 2023.

Peiyao A Zhao, Juan Carlos Rivera-Mulia, and David M Gilbert. Replication domains: genome compartmentalization into functional replication units. *DNA replication: from old principles to new discoveries*, pp. 229–257, 2017.

Zhihan Zhou, Yanrong Ji, Weijian Li, Pratik Dutta, Ramana Davuluri, and Han Liu. Dnabert-2: Efficient foundation model and benchmark for multi-species genome. *arXiv preprint arXiv:2306.15006*, 2023.

APPENDIX A. DATASET PREPARATION, MODEL TRAINING AND ARCHITECTURE DETAILS

The dataset was constructed using the human genome assembly GCF_009914755.1. Chromosomes 8, 20 and 21 were designated as the validation set, but only chromosome 20 was used to compute final metrics for computational efficiency. We did not use a separate test set. The dataset contains all mRNA and lncRNA genes, and all sequences were exclusively from the forward strand. Only transcripts with a length of up to 250 Kb were included.

Below we provide details of modifications in dataset, training regime or architecture for specific models:

1. For models with name containing "-u" (unique transcript models), we filtered the dataset selecting one representative transcript per gene, choosing the longest transcript available.
2. For the multispecies dataset, we processed data for 39 species (38 plus human) using the same strategy as for human samples. The list of species is provided in Table 3. It's important to note that only the human genome is fully assembled, therefore samples from other species containing 'N' characters (indicating unknown sequences) were excluded.
3. In the case of the `gena-base-512` model, if the input length exceeded the model's receptive field, we randomly selected a subsequence of the transcript. For other models, inputs were split into chunks equal to the model's receptive field; next, either the first chunk was fed into the model (for models without RMT augmentation), or a specific number of chunks were processed consecutively, starting from the first (for RMT models). During validation, we processed samples in 512-token segments without overlaps, starting from the first segment and up to the sample length.
4. In initial trials with base-size models, each sample was augmented with 2 Kb of intergenic sequence; however, this was found to be inefficient. In subsequent experiments with large-size models, we used samples that precisely matched gene boundaries.
5. All large-size models were trained using flash attention support to improve computational efficiency.
6. To train models at nucleotide-resolution, we utilized embeddings generated from the token-resolution model just before the classification layer, excluding memory tokens, CLS, and SEP tokens. Each token's embedding was replicated (upsampled) according to the number of nucleotides it represents. We formed so-called nucleotide embeddings, each corresponding to one of the four nucleotides and matching the dimensionality of the model-derived embeddings.

   For the transformer head setup, these nucleotide embeddings were then element-wise summed with the upsampled token embeddings and fed into a transformer head. As the transformer head, we employed the GENA-LM bert-large-t2t language model checkpoint, which processes segments of 512 embeddings from start to end of the transcript without overlaps. Additionally, this model includes a classification layer on top, outputting probabilities for each class at every nucleotide position. Thus, the resultant model consists of two transformers: the first initialized from the GENA-LM language model checkpoint and fine-tuned on the task of token-level gene segmentation using DNA sequence with RMT; the second is also initialized from a GENA-LM checkpoint but is finetuned on the task of predicting nucleotide-level gene segmentation based on summed outputs from the first model and DNA nucleotide sequence embeddings, without RMT. When fine-tuning a nucleotide-level head, we freeze the token-level part of the model.

   For models employing a U-NET head, we adopted a similar training strategy as with the transformer heads but with few differences. Instead of summing, we concatenated the embeddings from the token-level model with the nucleotide embeddings. Additionally, we trained the U-NET head together with the token-level model. The smaller size of the U-NET head allows computationally efficient training, enabling simultaneous optimization of both components. The input size for the U-NET in the original implementation (`rmt-gena-large-unet-42k-u-`**`multi_species`**) model was set equal to the longest input (250 Kb). However, to make training more efficient, the input size was adjusted to 8192 bp for all subsequent models.

7. For the model *rmt-gena-large-unet-42k-u-multi_species-**aug*** that recycle inputs supplementing each cycle with embeddings from the U-NET output obtained in previous cycle (so-called approach using iterations), we use following training scheme. We start with a checkpoint of the model trained without iterations. Resulting U-NET outputs are added to the embeddings that were used as U-NET inputs, and the result is used as input for second iteration of U-NET. We train this 2-iteration model using a short context of 32 Kb. Next, we extend context to 64 Kb with 2 iterations, and finally train on 64 Kb with 3 iterations. Increasing the number of iterations further does not improve results.

   Base models fine-tuning was always conducted starting from pre-trained GENA-LM. When training large models, we start from the best model checkpoint obtained on the previous gene annotation training iteration. We train all models without freezing any layers, except for transformer-based nucleotide-level model (see above).
8. Each model was trained on 8 NVIDIA A100 GPUs for 2–3 days.

Table 3: List of genomic assemblies used to create the multispecies training dataset. Assembly names correspond to the annotation and genome names. The annotation files have been received by the NCBI Eukaryotic Genome Annotation Pipeline.

Assembly	Species
GCF_000952055.2	Aotus nancymaae
GCF_002263795.3	Bos taurus
GCF_000767855.1	Camelus bactrianus
GCF_000002285.3	Canis lupus familiaris
GCF_000151735.1	Cavia porcellus
GCF_001604975.1	Cebus imitator
GCF_000283155.1	Ceratotherium simum simum
GCF_000276665.1	Chinchilla lanigera
GCF_000260355.1	Condylura cristata
GCF_002940915.1	Desmodus rotundus
GCF_000151885.1	Dipodomys ordii
GCF_002288905.1	Enhydra lutris kenyon
GCF_000308155.1	Eptesicus fuscus
GCF_000002305.2	Equus caballus
GCF_018350175.1	Felis catus
GCF_000247695.1	Heterocephalus glaber
GCF_009914755.1	Homo sapiens
GCF_000236235.1	Ictidomys tridecemlineatus
GCF_000280705.1	Jaculus jaculus
GCF_000001905.1	Loxodonta africana
GCF_001458135.1	Marmota marmota
GCF_000165445.2	Microcebus murinus
GCF_000317375.1	Microtus ochrogaster
GCF_000001635.26	Mus musculus
GCF_900095145.1	Mus pahari
GCF_002201575.1	Neomonachus schauinslandi
GCF_000292845.1	Ochotona princeps
GCF_000260255.1	Octodon degus
GCF_000321225.1	Odobenus rosmarus divergens
GCF_009806435.1	Oryctolagus cuniculus
GCF_000181295.1	Otolemur garnettii
GCF_016772045.2	Ovis aries
GCF_000956105.1	Propithecus coquereli
GCF_003327715.1	Puma concolor
GCF_036323735.1	Rattus norvegicus
GCF_000235385.1	Saimiri boliviensis boliviensis
GCF_000181275.1	Sorex araneus
GCF_000003025.6	Sus scrofa
GCF_000243295.1	Trichechus manatus latirostris

APPENDIX B. MODELS SCORING AND BENCHMARKING

PROCESSING PREDICTIONS

For models with token-level resolution, thresholds of 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, and 0.9 were defined to filter predictions. Predictions that passed each threshold were used to calculate metrics. The threshold with the best results was then selected.

For models with nucleotide resolution and SegmentNT, each nucleotide was assigned the class with the highest value from the comparison group. The comparison group is specific to each class: for the exon class, it includes exon and intron; for the CDS class, it includes CDS, intron, 5'UTR, and 3'UTR.

BENCHMARKING

Predictions were obtained by feeding the model with nucleotide sequences of transcripts (for interval-level and BUSCO) or genes (for gene-level). SegmentNT is not designed to process very long sequences, so for this model, the gene sequence was split into non-overlapping 50 kb segments.

It is important to note that some models are not capable of predicting lncRNA (Tiberius and AUGUSTUS), therefore only sequences of protein-coding genes or mRNA were provided to them. SegmentNT can predict only the exon class, so metrics for the CDS class were obtained by subtracting predictions of 5'UTR and 3'UTR from exon predictions. Finally, GENATATOR is capable of predicting both exons and CDS, so for this model, metrics were calculated across all classes for all genes and transcripts.

INTERVAL-LEVEL METRICS

To evaluate the accuracy of exon prediction for each model, sequences of a single transcript per gene were provided (the transcripts with the maximum total exon length were selected).

GENE-LEVEL

Each model generated predictions based on the gene sequences. Exon-level or CDS-level analysis was then performed, comparing predictions to each known transcript of each gene. If there is a transcript with complete and reciprocal overlap between predicted and exons and known exons, the gene was considered to be identified. CDS analysis was performed similarly.

BUSCO

Based on the predictions of each model, the nucleotide sequences of the genes were obtained for analysis. After performing the translation operation, the corresponding proteins were obtained and the longest of them was selected. The strand for translation was determined either directly if model outputs it explicitly (Tiberius and AUGUSTUS), or based on the predicted classes 5'UTR and 3'UTR, using the formula: $(FirstU5 - FirstU3) - (LastU5 - LastU3)$, where $FirstU5$ is the cumulative probability of 5'-UTR class preidction in the first 50 bases, $LastU3$ is the cumulative probability of 3'-UTR class prediction in the last 50 bases, and etc. (for SegmentNT and GENERATOR). Subsequently, the set of obtained proteins was analyzed using BUSCO.

NMD

Predictions for the longest transcripts of each protein-coding gene on human chromosome 20 were used as data for analysis.

APPENDIX C. ADDITIONAL MODEL METRICS

Table 4: Comparison of GENA-based gene segmentation models for token-level classification task. All labels were coerced to the token-resolution before computing metrics.

		gena-base-512	rmt-gena-base-4k	rmt-gena-base-4k-u	rmt-gena-large-4k-u	rmt-gena-large-16k-u	rmt-gena-large-42k-u
PR AUC		0.2983	0.6019	0.5940	0.6277	0.6421	0.6481
Exon-level mRNA	precision	0.0024	0.1274	0.2491	0.2930	0.4367	0.5700
	recall	0.0089	0.4988	0.5997	0.5458	0.6869	0.7617
	f1	0.0036	0.2030	0.3520	0.3813	0.5351	0.6521
Exon-level lncRNA	precision	0.0000	0.0147	0.0472	0.0823	0.1390	0.1519
	recall	0.0000	0.0822	0.1703	0.2393	0.2998	0.3241
	f1	0.0000	0.0249	0.0740	0.1234	0.1900	0.2069
Gene-level exon mRNA		0.0000	0.0018	0.0055	0.0495	0.0824	0.1502
Gene-level exon lncRNA		0.0000	0.0023	0.0023	0.0392	0.0484	0.0553

Table 5: Comparison of GENA-based gene segmentation models at nucleotide-resolution. All metrics provided at nucleotide resolution. For `rmt-gena-large-42k-u` predictions obtained at token-resolution were coerced to nucleotide resolution providing baseline of the token-level prediction.

		rmt-gena-large-42k-u	rmt-gena-large-2x-42k-u	rmt-gena-large-unet-42k-u	rmt-gena-large-unet-42k-u-multi_species	rmt-gena-large-unet-42k-u-multi_species-aug	rmt-gena-large-unet-42k-u-multi_species-aug-h	Reference
PR AUC		-	0.6457	0.6470	0.6793	0.6894	0.6894	
Exon-level mRNA	precision	-	0.0097	0.4785	0.6614	0.7616	0.8943	
	recall	-	0.0227	0.7748	0.8480	0.8813	0.8813	
	f1	-	0.0135	0.5917	0.7431	0.8171	0.8877	
Exon-level lncRNA	precision	-	0.0083	0.1259	0.1981	0.2795	0.5662	
	recall	-	0.0316	0.3767	0.4043	0.5174	0.5174	
	f1	-	0.0131	0.1887	0.2659	0.3629	0.5407	
Gene-level exon mRNA		-	0.0256	0.1905	0.3187	0.5238	0.5513	
Gene-level exon lncRNA		-	0.0207	0.0599	0.1083	0.1682	0.1935	
BUSCO mammalia CDS	Complete	39	39	110	159	209	209	275
	Fragmented	42	46	61	54	39	39	3
BUSCO primates CDS	Complete	74	58	163	236	307	307	409
	Fragmented	41	46	61	67	49	49	4

APPENDIX D. ANALYSIS OF GENATATOR MODEL ERRORS TO IMPROVE GENE ANNOTATION

We next focus on understanding the primary causes underlying erroneous predictions of GENATATOR, starting from false-positive (FP) predictions within the exon class, specifically the incorrect insertion of exon sequences into intronic regions (Fig. 3A and Fig. 3B). A prevalent error was the mislabeling of complete introns shorter than 500 bp as exons. This indicates that the model learned exon length distribution (exons are typically small, around 200-300 bp in length) and does not recognize short intervals as introns. Beyond that, we observed no significant correlation between FP probability and intron length or proximity to exon-intron boundaries. Additionally, precision slightly declines with distance from the gene start, reflecting the underrepresentation of longer genes in the training dataset (Fig. 3C and Fig. 3D).

Next, we analyzed precision and recall stratifying exon-intron boundaries by their flanking dinucleotides. Consistent with prior observations, there is a pronounced overrepresentation of AG dinucleotides at left (exon $\rightarrow$ intron) boundaries and GC dinucleotides at right (intron $\rightarrow$ exon) boundaries (Fig. 3E and Fig. 3F). Both precision and recall show a correlation with the frequency of the boundary dinucleotides, pointing to the importance of having sufficient training samples to learn the grammar of intron boundaries. Although the frequency of predicted boundaries at each dinucleotide generally reflects the true distribution, we identified samples where dinucleotides flanking predicted boundaries never occur at boundary positions in the actual data. By explicitly excluding exons flanked by these ”illegal” dinucleotides, model performance improved, enabling correct seg-

mentation of 385 genes, which represents about 40% of the dataset. We use this improved model in the following benchmarking experiments.

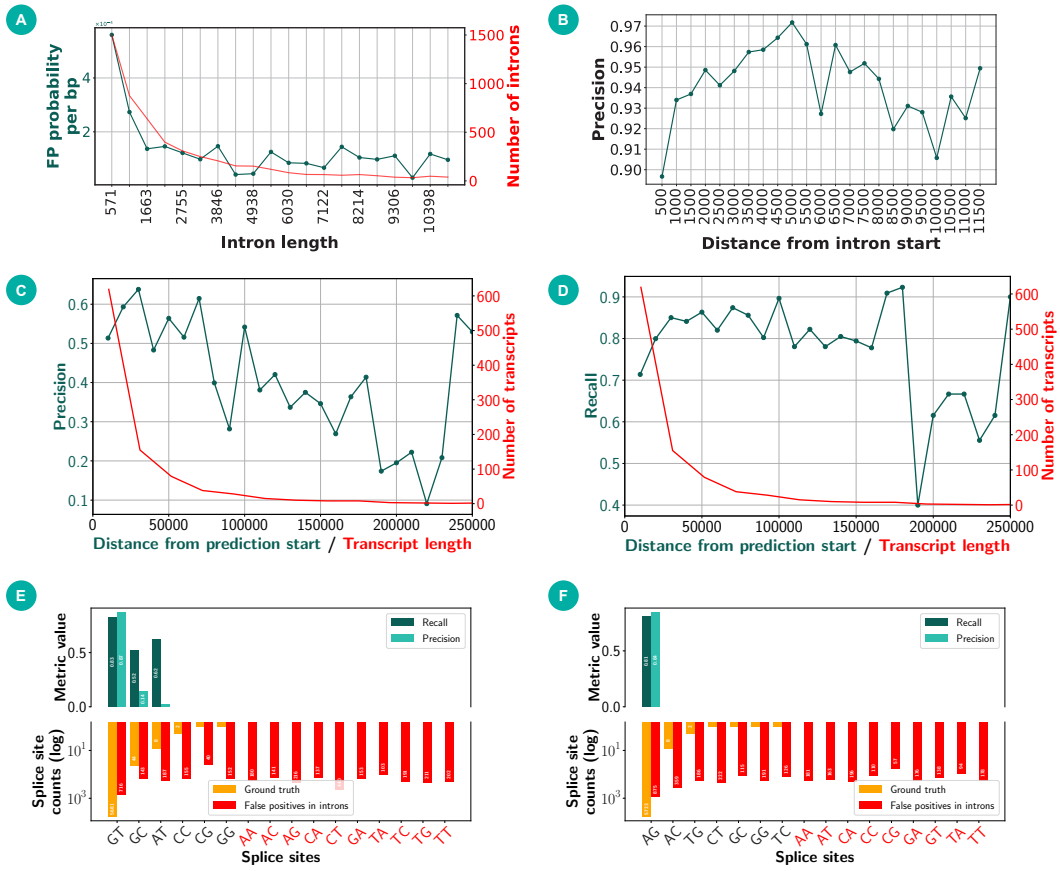

Figure 3: GENATATOR error analysis provides insights into potential tweaks for improving gene annotation. A-D: Performance metrics as a function of intron length (A), distance from exon-intron boundary (B), and distance from gene sequence start (C-D). B aggregates intron sequences located at specific distance from exon-intron boundary. In A and B the distribution is cropped at the 90th percentile, in C and D at 250Kb. E and F: Precision and recall at predicted intron-exon boundaries, stratified by flanking dinucleotide, separately for left (E) and right (F) intron boundary, with the distribution of targets shown in red and orange.

APPENDIX E. LEARNING TRANSCRIPTION RULES AND RETAINING GENE PROPERTIES USING MEMORY

In our further analysis we study how GENATATOR learns the genomic grammar underlying gene segmentation.

We first examined the model's capacity to detect gene directionality. Input sequence may represent genes oriented from transcription start to end or *vice versa*. Gene orientation is reflected by the arrangement of UTRs: the 5'-UTR is proximal to the transcription start site, whereas the 3'-UTR is located at the transcription termination site.

To validate the model's ability to determine strand orientation, we developed a strand-definition score calculated as follows: $(FirstU5 - FirstU3) - (LastU5 - LastU3)$, where $FirstU5$ is the cumulative probability of 5'-UTR classpredictionn in the first 50 bp, $LastU3$ - sampe for 3'-UTR class in the last 50 bp, and etc. Sequences yielding positive scores were considered to be in the forward orientation, whereas negative scores indicated a reverse orientation. This metric allows correctly determining gene orientation with 95% accuracy, confirming the model's capacity to differentiate gene orientation based on the input sequence.

Next, we evaluated whether GENATATOR has learned biological rules related to translation grammar. Nonsense-mediated decay (NMD) is a crucial cellular mechanism that targets transcripts containing non-functional protein-coding sequence Supek et al. (2021). NMD machinery identifies these transcripts by the presence of premature translation termination signal, i.e. STOP-codon located more than 50 nucleotides upstream from the last exon-exon junction. Consequently, the CDS of protein-coding genes is typically terminated within the last two exons. Our analysis revealed that approximately 88.4% (483 out of 546) of all predicted gene segmentations comply with the NMD rules, suggesting that GENATATOR has effectively internalized the NMD grammar (Fig. 4A). This demonstrates the model's capability to incorporate complex biological rules governing gene expression and regulation.

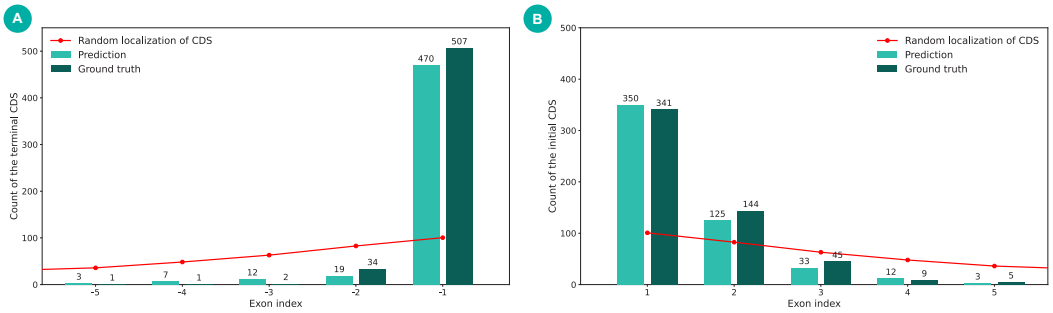

Figure 4: GENATATOR learns translation grammar. Location of last (A) and first (B) CDS nucleotide depending on exon index. Red line shows the distribution obtained by randomly selecting exon index.

In contrast to the terminal CDS, initial CDS positions are not constrained by NMD rules and can occur in any exon. The predicted distribution of initial CDS (Fig. 4B) closely matches biological patterns observed in ground truth transcripts, indicating that the model successfully captures biological dependencies between exons and CDS.

GENATATOR's ability to learn biological signals crucially depends on its capacity to utilize information from proximal gene segments analyzing distal chunks of the sequences. This includes retaining knowledge about the direction and type (coding or lncRNA) of the transcript for accurate processing. In the GENATATOR model, we use RMT augmentation, where memory tokens serve as placeholders to retain important information between sequence chunks. To investigate whether the memory tokens receive and store information about gene direction and type, we trained a gradient boosting classifier predicting these gene features based on memory tokens embeddings extracted either from the first or last chunks of gene sequences. The results provided in the Table 6 show

Table 6: F1-score of gradient boosting classifier on memory token embeddings.

	GENE ORIENTATION		GENE TYPE		LAST STATE
MEMORY ID	FIRST	LAST	FIRST	LAST	ALL
MEAN	0.9273	0.9387	0.8630	0.9064	0.7847
STD	0.0214	0.0032	0.0156	0.0198	0.0132

that memory tokens acquire information about gene direction and type immediately from the first segment and retain it until the last gene segment.

Correctly arranging gene elements in their specific order is essential for resolving gene structure. For instance, the sequence of exons and introns should not be interrupted by untranslated regions in the middle, and the coding sequence should be continuous and flanked by non-coding UTRs. Tools like Tiberius use HMM with explicitly defined transition probabilities above the DNN predictions to ensure this order. In contrast, models utilizing RMT can preserve the correct arrangement of gene elements by remembering the class of the previous segment's end. This memory is then utilized to constrain the potential states at the start of the next segment, ensuring that predictions follow valid structure. To find whether memory tokens store information about the class of the last token in a segment, we again employ classifier on memory token embeddings, predicting if the last token of a segment belongs to exon or intron class. Our results (Table 6) confirm that memory retains information about the state of the last tokens in the segment, supporting the function of RMT in structuring gene elements.

APPENDIX F. BENCHMARKING GENATATOR AGAINST OTHER GENE-ANNOTATION TOOLS

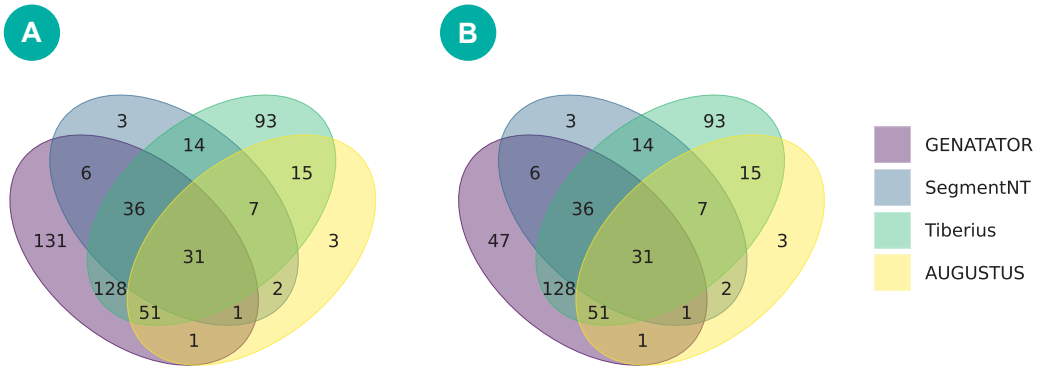

Figure 5: GENATATOR provides the most complete set of genes. Overlap of genes identified using different models. Data is shown for protein-coding and lncRNA genes together (A), and for protein-coding genes only (B). The number of protein-coding genes is 301, 100, 111 and 375 for GENATATOR, SegmentNT, AUGUSTUS and Tiberius, respectively. When accounting for lncRNA genes, GENATATOR's total gene count increases to 385. For other models, the count of genes remains constant.