
OmniGenBench: A Modular Platform for Reproducible Genomic Foundation Models Benchmarking

Heng Yang¹, Jack Cole¹, Yuan Li², Renzhi Chen³, Geyong Min¹, Ke Li¹

¹Department of Computer Science, University of Exeter, Exeter, UK

²National University of Defense Technology, Changsha, China

³Qiyuan Lab, Beijing, China

{hy345, j.cole1, g.min, k.li}@exeter.ac.uk

{liyuan22}@nudt.edu.cn {chengrenzhi1989}@gmail.com

Abstract

The code of nature, hidden in DNA and RNA genomes since the evolution of living systems, holds immense potential for impacting humans and ecosystems through genome modeling. Genomic Foundation Models (GFMs) have been proposed for genomic modeling as they hold transformative promise in genome deciphering. As GFMs scales up and reshape the landscape of AI-driven genomics, the field faces a growing need for rigorous, reproducible evaluation. We introduce OmniGenBench, a modular benchmarking platform designed to unify data, model, benchmark, and interpretability layers across GFMs. OmniGenBench enables standardized, one-command evaluation of any GFM on five benchmark suites, with seamless integration of 31+ open-source models. Through automated pipelines and community-extensible features, the platform addresses reproducibility gaps in data transparency, model interoperability, benchmark fragmentation, and black-box interpretability. OmniGenBench aspires to be a foundational infrastructure for reproducible genomic AI research, accelerating trustworthy discovery and collaborative innovation in the era of genomic-scale modeling.

1 Introduction

All living systems encode information in their genomic sequences. Just as Watson and Crick’s helical structure cracked the DNA code [1], AI is now learning the chemical language of life to decode the regulatory grammar hidden within genomes [2]. Foundation models (FMs), also known as large language models (LLMs), are generative AI systems that understand and generate human language. Prominent examples like OpenAI’s ChatGPT and Google’s Gemini have already transformed sectors like education [3, 4], entertainment [5, 6] and business [7, 8]. Analogous to LLMs learning text, genomic FMs (GFMs) learn biological insights directly from extensive genomic data [9]. The potential impact of GFMs in life sciences is substantial. McKinsey estimates that generative AI could unlock \$60–\$110bn in annual economic value within life science industries¹. Echoing the latest PCAST report in the USA², FMs offer unprecedented potential to drive a new era of digital life sciences, supercharging scientific progress, from drug discovery to super-personalized medicine.

Reproducibility Crisis FMs have sparked immense enthusiasm for their potential in genomics and the broader life sciences. However, their real-world uptake remains surprisingly slow compared to their rapid adoption in fields like natural language processing and computer vision. This slower uptake primarily arises from four critical challenges, each directly tied to *reproducibility*.

¹<https://tinyurl.com/4yhwhzfd>

²<https://tinyurl.com/2dfj68ne>

1. **Data availability** A major barrier to reproducibility in GFM research is limited access to datasets. \diamond *Lack of shared training data*: Many open-sourced GFMs do not release the exact datasets used for pre-training or fine-tuning. In our analysis of 31 GFMs (as shown in Appendix E) integrated into OmniGenBench, only 16 of them are provided with the curated datasets. This lack of transparency directly restricts reproducibility and meaningful model comparisons. \diamond *Absence of standardized task data*: Genomics modeling lacks standardized formats and central repositories for task-specific data. Consequently, researchers often rely on ad-hoc preprocessing. This results in opaque benchmarks that others cannot reliably replicate. \diamond *Limited scope of existing benchmark datasets*: Existing genomic benchmarks typically focus on narrow tasks (e.g., single data modality). Researchers thus repeatedly assemble custom datasets, causing fragmented and non-standardized evaluations.
2. **Model accessibility** Inconsistencies in model implementations hinder widespread adoption and benchmarking. \diamond *Inconsistent model implementations*: GFMs are often released with highly customized codebases, architectures, and tokenization schemes. Such variability creates interoperability barriers. For example, some GFMs use 6-mer tokenization [10], while others use byte-pair encoding [11], resulting in incompatible formats. \diamond *Lack of standardized interfaces*: Many GFMs are unavailable through common model repositories or standardized programming interfaces. Thus, integrating new models requires substantial model-specific technical efforts. For instance, the recent Plant Genome Benchmark (PGB) [12], despite its valuable datasets, lacked generic interfaces for incorporating new GFMs, limiting its utility for comparative evaluation.
3. **Benchmarking standardization** Disjointed evaluation practices prevent meaningful comparisons across models. \diamond *Inconsistent evaluation practices*: Different groups evaluate GFMs using distinct tasks and metrics. Such variation prevents fair model comparisons and complicates verification of reported results. \diamond *Barriers to universal benchmarking*: Evaluating diverse GFMs across multiple genomic tasks remains challenging due to varying data modalities (e.g., DNA vs. RNA) and incompatible model implementations. As a result, universally benchmarking ‘any model on any genomic task’ remains impractical under current fragmented approaches. \diamond *Limitations of existing benchmarks*: Current benchmark suites like Genomic Benchmarks (GB) [13] and GUE [11] narrowly focus on specific DNA classification tasks. Others benchmarks focus exclusively on niche domains (e.g., plant-specific genomics [12]). These benchmarks often operate independently, complicating evaluations across different suites. Even comprehensive benchmarks like BEACON [14] for RNA has limited portability due to specialized environments. Without standardization, researchers must repeatedly reconstruct evaluation pipelines.
4. **Interpretability** The opaque nature of GFMs limits their practical usability in biology and medicine. \diamond *Lack of mechanistic insight*: GFMs typically generate predictions without clear biological explanations. This opacity prevents reliable biological validation, causing skepticism among clinicians and biologists. \diamond *Ad-hoc interpretability practices*: Interpretability analyses, such as motif discovery or feature attribution, usually rely on manually designed, post-hoc methods. These approaches vary widely across studies, limiting consistency and reproducibility. \diamond *Inconsistent scientific conclusions*: Such inconsistent practices often yield divergent interpretations of model capabilities, even for identical tasks. The absence of routine, reproducible interpretability methods not only reduces user trust but complicates validation of biologically meaningful insights (e.g., detecting known regulatory motifs).

Our Solution OmniGenBench To explicitly address this reproducibility crisis, we introduce OmniGenBench, a unified and modular benchmarking platform. OmniGenBench consists of four core modules, each specifically designed to address one of the critical barriers identified above.

- **Data module:** OmniGenBench provides centralized access to 123 carefully curated genomic datasets, accompanied by clear documentation and standardized formats. It simplifies data reuse across pre-training and fine-tuning tasks and ensures consistency across different studies. This module also facilitates easy community sharing of genomic datasets.
- **Model module:** The platform hosts a standardized model hub featuring unified wrappers and application programming interfaces (APIs). This hub integrates 31 GFMs to date and simplifies the integration of diverse GFMs by eliminating interoperability barriers from customized code-

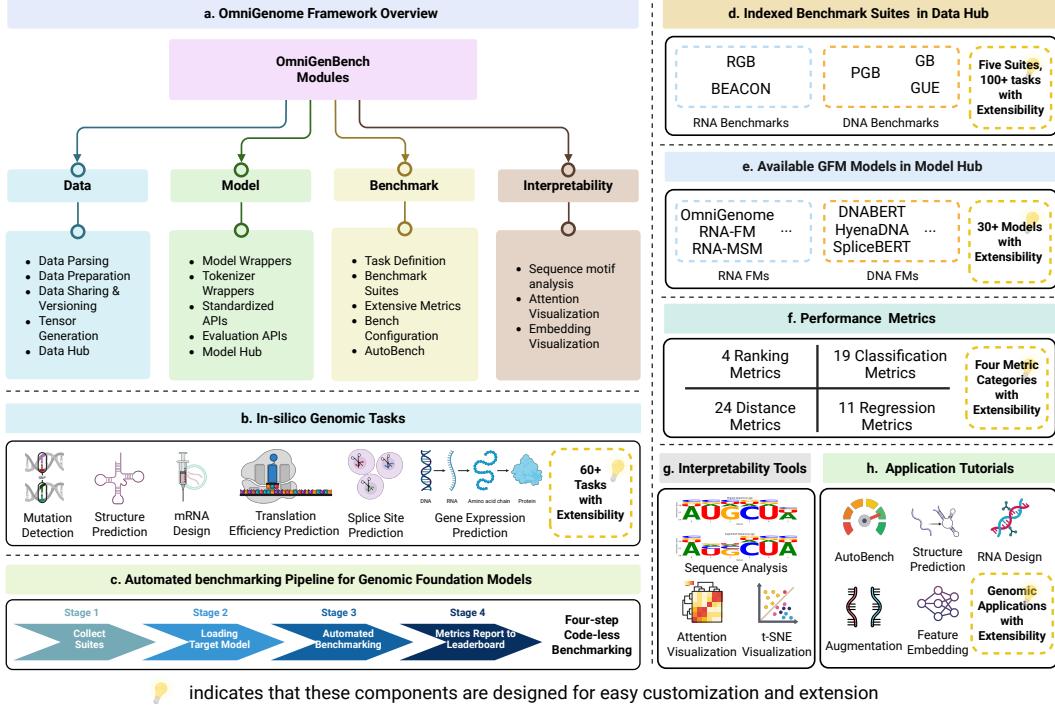


Figure 1: Overview of the OmniGenBench framework. **a)** OmniGenBench consists of four core modules covering data, model, benchmark, and interpretability aspects. **b)** The current release includes 60+ *in-silico* genomic tasks covering diverse biological processes. **c)** A four-stage code-less benchmarking pipeline that can automate the end-to-end evaluation. **d)** Five benchmark suites (containing 123+ datasets) are indexed in the Data Hub. **e)** Model Hub hosts 31+ GFMs, supporting simple deployment, to streamline applications for hosted GFMs from usually several weeks to one day. **f)** A library of four different types of 58+ evaluation metrics covering ranking, classification, regression, and distance. **g)** Interpretability tools such as sequence-level motif analysis and embedding space analysis. **h)** Eight pre-compiled pedagogical tutorials with specific applications (Appendix G).

bases, varied architectures, or incompatible tokenization methods. As a result, researchers and practitioners can significantly reduce model integration efforts.

- **Benchmark module:** OmniGenBench offers a comprehensive, automated benchmarking suite that covers diverse genomic tasks across multiple modalities (DNA and RNA). It includes 123+ datasets with 58+ metrics for integrated GFMs. Its extensible design and consistent evaluation protocols explicitly enable researchers to benchmark ‘*any model on any genomic task*’ within a unified framework. This approach ensures fair, transparent, and verifiable performance comparisons.
- **Interpretability module:** Recognizing the need to ‘*open the black box*’, OmniGenBench integrates three standardized interpretability tools into its evaluation pipelines. Researchers can routinely conduct reproducible analyses such as motif discovery and feature attribution mapping. This integration provides clear, mechanistic insights into GFM predictions, significantly enhancing model transparency and trustworthiness.

Through these interconnected components, OmniGenBench aims to serve as a foundational platform that fosters a reproducible, transparent, and collaborative ecosystem, which in turn accelerating meaningful progress in GFM research, scientific discovery, and beyond.

2 System Design of OmniGenBench

Figure 1.a gives the hierarchical structure of OmniGenBench system. Our framework comprises several clearly defined modules, each explicitly linked to one of the reproducibility challenges identified in Section 1. OmniGenBench is designed as both a **ready-to-use** and **ready-to-expand** platform. As a *ready-to-use* system, OmniGenBench enables users to apply existing GFMs directly to

specific biological tasks, such as predicting translation efficiency or performing sequence generation. As a *ready-to-expand* system, OmniGenBench provides clear APIs and comprehensive tutorials. These resources allow AI researchers, domain scientists, and stakeholders of varying expertise levels to: *i*) upload their own datasets, which are automatically standardized into formats compatible with FM training, and *ii*) easily customize or build innovative FMs guided by robust documentation and standardized protocols. To illustrate specific implementation details, we provide an auxiliary website showcasing relevant code snippets. In addition, we offer pedagogical tutorials in Appendix G, helping users practically understand and effectively interact with the platform’s core functionalities. The following paragraphs introduce the functional design of each module step by step. To elaborate on the module implementation details, we introduce the inputs, outputs and description for each module with the source code examples at this online page³.

2.1 Data Module

This module is engineered with four key functionalities to address the data availability challenge.  Standardized data parsing: OmniGenBench provides data parsing tools to convert genomic data from diverse formats (e.g., FASTA, JSON) into clearly documented and standardized datasets, each accompanied by detailed metadata.  Flexible data preparation: Researchers can easily utilize built-in capabilities for dataset preparation. These include adaptive sequence manipulation (truncation and padding), instance filtering via established tools like CD-HIT-EST⁴ to prevent label leakage in RNA-structure tasks, and sequence augmentation techniques [15]. These tools simplify the generation of researcher-curated datasets for subsequent modeling.  Community data sharing & versioning: Once curated, datasets can be versioned and shared within the centralized OmniGenBench Data Hub (see Figure 1.d). This capability directly addresses the current lack of accessible genomic training datasets and promotes broader coverage beyond existing specialized benchmarks.  Standardized tensor generation: Finally, we transform diverse genomic inputs, from existing benchmarks or community contributions, into standardized tensor data. These tensor datasets are directly compatible with downstream GFM modeling, significantly simplifying experimentation workflows.

 By standardizing data processing, simplifying dataset creation, and encouraging community sharing, this module resolves the field’s reliance on fragmented, ad-hoc preprocessing methods.

2.2 Model Module

It addresses the challenges of model accessibility and usability by standardization and simplified integration.  Unified model wrappers & wrappers: OmniGenBench offers a base model template featuring unified APIs to address interoperability barriers from diverse GFM architectures (e.g., transformer [16, 17], Hyena [18, 19, 20], Mamba [21, 22]) and customized tokenization methods. The platform employs standardized model wrappers and tokenizer wrappers. These wrappers abstract away specific architectural details, providing consistent input-output handling whether a GFM uses *k*-mer or byte-pair encoding.  Standardized APIs for core operations: The model template provides standardized, universal APIs covering critical operations. It includes three clearly defined training options: *i*) a basic native trainer, *ii*) a scalable Hugging Face trainer, and *iii*) a distributed accelerate trainer. We also standardize evaluation and inference APIs, which significantly reduce the technical effort typically needed to integrate and use diverse GFMs.  Centralized & extensible model Hub: Built on Hugging Face infrastructure, the OmniGenBench Model Hub (Figure 1.e) hosts an extensive collection of 31+ pre-trained GFMs. This includes specialized DNA models (e.g., DNABERT [10, 11], HyenaDNA [19]) and RNA models (e.g., OmniGenome [23], RNA-FM [24]). This centralized hub ensures models are readily accessible, with clearly documented interfaces. Additionally, the platform actively supports community contributions. Researchers can easily upload and share new GFMs and datasets (see Appendices C and E), further expanding the ecosystem and promoting collaborative advancement.

³<https://github.com/COLA-Laboratory/OmniGenBench/blob/master/ModuleAppendix>

⁴<https://bioinformatics.org/cd-hit/>

” By providing standardized interfaces, unified wrappers, and a centralized hub, this module directly resolves key accessibility barriers, greatly simplifying the integration, comparison, and use of diverse genomic foundation models.

2.3 Benchmark Module

This module addresses inconsistent evaluation practices and the limitations of existing disjointed benchmarks by four clearly defined functionalities.

Curated benchmark suites & Data Hub integration: The OmniGenBench Data Hub (Figure 1.d) provides unified access to five systematically curated benchmark suites, covering 123+ diverse genomic dataset. These include RNA-focused benchmarks (RGB [23], BEACON [14]) and DNA-focused benchmarks (PGB [12], GB [13], GUE [11]). This comprehensive, centralized collection, detailed further in Appendix C, facilitates broad assessments of GFM generalization beyond narrowly defined tasks. The OmniGenBench Data Hub is designed explicitly for ongoing community-driven expansion.

Flexible task definition & configuration: At the core of OmniGenBench is a robust task module. It enables clear, structured definition and configuration of genomic tasks. Researchers can explicitly specify data preprocessing steps, model architectures, loss functions, and evaluation metrics. Currently, the module includes 60+ pre-defined genomic tasks (Figure 1.b), such as RNA secondary structure prediction [25, 26, 27, 28] and mRNA design [29]. The module also supports the straightforward addition of new community-defined tasks, ensuring consistent evaluation setups across studies.

Centralized & extensible metric registry: To guarantee fair and consistent model assessments, OmniGenBench includes a dedicated metric module. This module provides a comprehensive, task-agnostic registry of over 58+ standardized metrics. Metrics span various categories, including classification and ranking (e.g., F_1 , ROC-AUC), regression (e.g., RMSE, R^2), and distance or similarity measures (e.g., cosine similarity). These metrics are largely integrated from robust libraries such as SCIKIT-LEARN⁵. Importantly, new or custom genomic metrics (e.g., structural alignment scores) can be easily integrated using task-level JSON configuration files, balancing standardization with flexibility.

Automated & reproducible evaluation workflow: Finally, the framework supports automated evaluation through streamlined task compilation, integrating data, model configurations, and evaluation metrics. The integrated AutoBench engine significantly simplifies benchmarking workflows. For example, evaluating GFMs like OmniGenome on benchmarks (e.g., RGB) can be executed in a single command, e.g., `[autobench -model OmniGenome-52M -benchmark RGB]`.

” By providing integrated benchmarks, standardized task definitions, flexible metrics, and automated benchmark workflows, OmniGenBench promotes fair comparisons, reliable result verification, and rigorous scientific inquiry within the GFM research community.

2.4 Interpretability Module

To address the transparency challenges associated with the ‘black-box’ nature of GFMs, this module provides standardized workflows and integrated tools to systematically examine and validate GFM predictions through three main functionalities (Figure 1.g).

Built-in tools for mechanistic insight: OmniGenBench includes built-in interpretability tools, eliminating the reliance on inconsistent and manually designed post-hoc analyses. Current integrated tools are broadly applicable across various GFMs and genomic tasks, including: *i*) sequence-level motif analysis for identifying learned sequence patterns (demonstrated in Section 4.1); *ii*) embedding space analysis for enabling visual exploration of how models represent genomic features (see Appendix H.1); and *iii*) Attention map visualization that reveals model focus and decision-making processes during sequence analysis (see Appendix H.2). Integrating these tools directly within the evaluation pipeline ensures their routine and consistent use.

Promoting reproducible scientific conclusions: By standardizing interpretability analyses, this module ensures the reproducibility and comparability of the biological insights derived from GFMs. Utilizing clearly documented, common methodologies (see tutorials in Appendix H), researchers can more reliably interpret and validate biological significance (e.g., detecting known regulatory motifs). This systematic approach reduces the risk of inconsistent conclusions associated with varied interpretability techniques.

Extensibility for future methods: This module is designed for extensibility, allowing for the future integration of new and emerging interpretability techniques.

⁵<https://scikit-learn.org/>

” By embedding systematic and standardized interpretability analyses directly within the GFM evaluation process, OmniGenBench enhances transparency, improves user trust, and facilitates robust, biologically meaningful insights from GFMs.

3 Overall Benchmark Results

This section presents pivotal performance of integrated GFMs benchmarked via OmniGenBench. The evaluations span four major benchmark suites: the RNA Genomic Benchmark (RGB; see Appendix C.1), the Plant Genomic Benchmark (PGB; Appendix C.2), the Genomic Understanding Evaluation (GUE; Appendix C.3), and the Genomics Benchmark (GB; Appendix C.4). For fair comparisons, data splits and primary metrics follow their original publication settings or established best practices. **Detailed numerical results and in-depth task-specific discussions are provided in Appendix C.**

3.1 State-of-the-Art (SoTA) Performance

We first quantify the SoTA achievements for 11 public GFMs across all evaluated tasks within the aforementioned benchmark suites. A model achieves SoTA by attaining the top performance on a task’s primary metric in our evaluation. This SoTA count intuitively measures cross-scenario generalization and excellence. Figure 2 summarizes these SoTA counts. OmniGenome notably leads in SoTA achievements, particularly within the RGB and PGB suites. SpliceBERT [30] and 3UTRBERT also secure several SoTAs, primarily in PGB, GUE, and GB. Conversely, models such as HyenaDNA, Caduceus [31], Agro-NT [12], RNABERT [32], RNA-MSM [33], and RNA-FM [30] did not achieve any SoTAs in our current evaluation setup.

Discussion. The distribution of SoTA achievements in Figure 2 offers initial insights into GFM capabilities. OmniGenome’s superiority, especially in RGB, suggests its pre-training techniques are well-suited for RNA tasks and generalize effectively. The absence of SoTAs for models like HyenaDNA, Caduceus, Agro-NT, RNABERT, RNA-MSM, and RNA-FM in this evaluation (while they may be strong in their original contexts) could indicate that their pre-training objectives (e.g., standard masked language modeling without genomics-specific knowledge) are less effective for the complex tasks in these suites (e.g., RNA secondary structure prediction). This observation may underscore the importance of GFMs incorporating more structure-aware [23], multi-species pre-training strategies for specialized genomic tasks. However, SoTA counts are aggregate measures and do not capture task-level nuances or the specific strengths of individual GFMs, which are further explored in Appendix C.

3.2 Rank-Based Performance

To provide a more nuanced view of where each GFM excels or struggles, we present rank-normalized radar charts. For each benchmark suite, an individual radar chart is generated for each GFM, where axes represent distinct task categories. A model’s performance on each axis is its average rank (rank 1 = best). Smaller, more centrally-focused polygons indicate stronger, more balanced performance. The average rank of each model within a suite is also noted. Due to space constraints, we illustrate the RGB radar charts (Figure 3) in this section; comprehensive radar charts for all suites are in Appendix B.1.

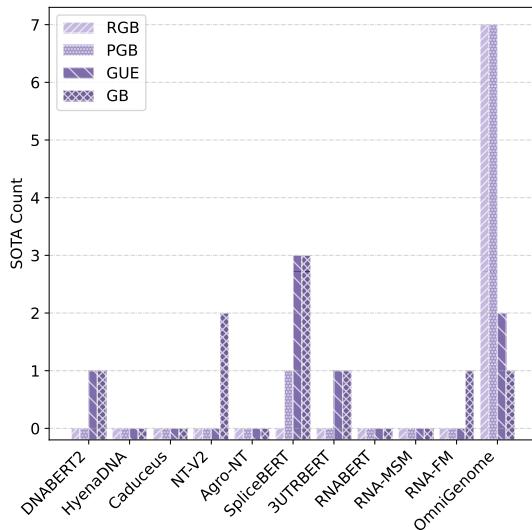


Figure 2: State-of-the-Art (SoTA) achievements of public GFMs across tasks within the four primary benchmark suites.

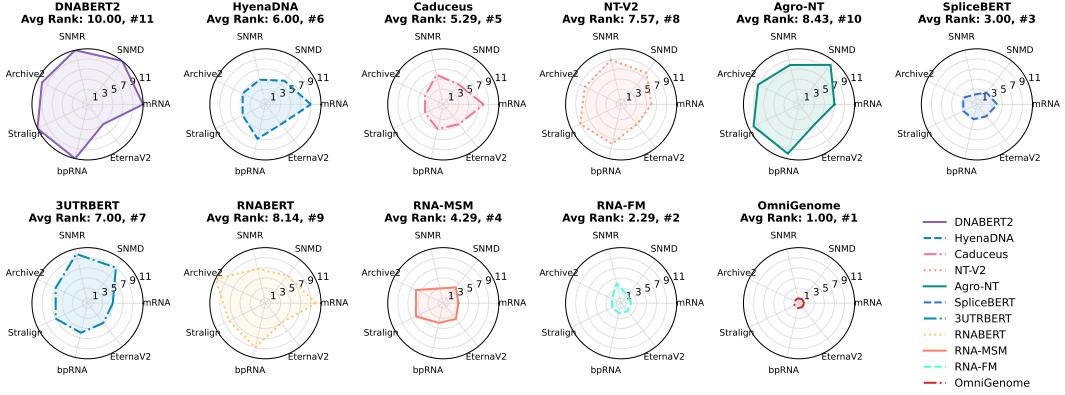


Figure 3: Rank-based radar charts comparing eleven GFMs on the RGB suite. Each small plot represents a model, with axes corresponding to different RNA task categories. Lower ranks (closer to the center) indicate better performance. The average rank for each model on RGB is displayed above its plot.

Discussion. Focusing on the RGB suite (Figure 3), OmniGenome (avg. rank 1.00) demonstrates clear dominance, its radar polygon nearly collapsing to the center. RNA-FM (avg. rank 2.29) follows as a strong runner-up, with SpliceBERT [30] (avg. rank 3.00) also showing competitive performance. DNA-centric models like DNABERT-2 (avg. rank 10.00) exhibit significantly larger polygons, underscoring their weaker RNA task transferability. Structure-aware models, particularly OmniGenome and RNA-FM, excel in structure-related task categories such as secondary structure prediction (e.g., the SSP task).

Broader observations from the comprehensive radar charts across all four benchmark suites (see Figure 5) reveal several key trends regarding GFM behaviours:

- OmniGenome leverages RNA structure-aware pre-training, significantly boosting its performance on structure modeling tasks like secondary structure prediction and RNA design. Its pre-training on multi-species plant genomes also translates to strong performance in PGB on tasks such as PolyA site prediction. Notably, OmniGenome demonstrates robust generalization capabilities even on out-of-domain benchmarks (GUE and GB), where pre-training data has less direct overlap with downstream tasks. This suggests that incorporating structural information into pre-training is a promising direction for future GFM research.
- RNA-FM, another RNA-focused model, also achieves commendable results on the RGB suite, particularly in structure prediction tasks. However, its generalization ability appears less pronounced, with suboptimal performance on out-of-domain benchmarks like PGB. Its reliance on structural information for RNA also means it is not directly applicable to DNA sequence modeling using the same framework.
- SpliceBERT, a DNA-centric GFM, excels in DNA-related tasks such as DNA sequence modeling within GUE and GB. However, its performance can be inconsistent across different tasks, possibly due to the specificity of its training data, indicating a need for broader pre-training data for more robust, generalizable DNA models.
- Models employing single nucleotide tokenization (SNT) generally exhibit strong performance across both fine-grained RNA modeling and broader DNA sequence tasks, showcasing a degree of flexibility. However, SNT can lead to longer effective sequence lengths, potentially increasing computational demands and impacting modeling efficiency for very long sequences.
- Generative models like HyenaDNA and Caduceus generally show modest performance across most benchmarks in this evaluation. This may reflect the current developmental stage of generative GFMs, whose sequence understanding capabilities may not yet match those of discriminative models optimized for specific tasks. We anticipate that continued advancements in generative GFM architectures will lead to improved performance in precise sequence understanding and generation.

In summary, these rank-based visualizations effectively highlight model-specific strengths and weaknesses across diverse genomic task categories beyond what aggregate SoTA scores reveal. This detailed insight is crucial for informed GFM selection for specific downstream applications and for guiding future GFM development strategies, including choices regarding pre-training data, model architecture, and tokenization schemes.

4 Interpretability Analysis of Genomic Foundation Models

While quantitative performance metrics are essential for evaluating GFMs, they often fall short of addressing critical concerns regarding the trustworthiness and biological relevance of the model predictions. Therefore, this section presents an interpretability case study focused on sequence motif preservation to assess the fidelity of GFMs when used for sequence augmentation. Further interpretability case studies, including feature embedding analysis and attention mechanism inspection, are detailed in Appendix H.

4.1 Sequence Motif Preservation in GFM-based Augmentation

Motivation. A critical question for the practical application of GFMs in sequence analysis is their ability to preserve evolutionarily conserved contexts encoded within biological sequences. Wet-lab scientists often express skepticism regarding whether sequences generated or augmented by these models authentically reflect native sequence characteristics, such as those captured in a Multiple Sequence Alignment (MSA). This study investigates whether GFMs, when used for sequence augmentation, can faithfully reproduce the position-specific motifs characteristic of conserved RNA families.

Experimental Design. To evaluate this fidelity, we employ visual inspection of sequence logos complemented by quantitative, information-theoretic scores. We selected two distinct, well-characterized RNA families from the Rfam database [28] for this study: RF02914 (DUF805 motif)⁶ and RF02913 (pemK motif)⁷. These families represent conserved RNA structures initially discovered through bioinformatics methods and possess curated seed alignments representing conserved primary sequences and, implicitly, structural information. The core methodology employs GFMs in a Masked Language Modeling (MLM) task to generate augmented sequences. The experimental steps were as follows:

1. **Sequence Selection:** Sequences were drawn from the seed MSA of each RNA family.
2. **Masking:** In each selected sequence, 15% of nucleotides were randomly masked.
3. **Prediction and Augmentation:** Four GFMs capable of single nucleotide prediction, OmniGenome, SpliceBERT [30], RNAFM [24], and RNA-MSM [33], were used to predict the masked nucleotides, generating ten augmented variants per original sequence.
4. **Aggregation:** The collection of augmented sequences for each GFM was treated as a new MSA group for comparison against the original seed MSA.
5. **Visualization:** Sequence logos were generated from the augmented MSAs using Logo-maker⁸ to visualize position-specific nucleotide frequencies.
6. **Quantification:** The distributional similarity between the nucleotide frequencies of the GFM-augmented MSA subset and the original seed MSA subset (evaluated at unmasked positions for direct comparison, and across the entire sequence for overall fidelity) was quantified using **Jensen-Shannon Divergence** (D_{JS}) (lower values indicate higher similarity) and **Cosine Similarity** (higher values indicate greater similarity).

The visual and quantitative results for the RF02914 and RF02913 families are presented in Figure 4.

Results. As shown in Figure 4, the fidelity of motif preservation varies notably across the evaluated GFMs. OmniGenome achieves the best performance, with minimal divergence from native distributions (D_{JS} of 0.0163 and 0.0082; Cosine Similarity of 0.9960 and 0.9981, for RF02914 and RF02913 respectively), producing sequence logos nearly indistinguishable from the original seeds. SpliceBERT shows moderate fidelity (D_{JS} up to 0.0454 for RF02913, Cosine Similarity down to

⁶<https://rfam.org/family/RF02914>

⁷<https://rfam.org/family/RF02913>

⁸<https://logomaker.readthedocs.io/en/latest/>

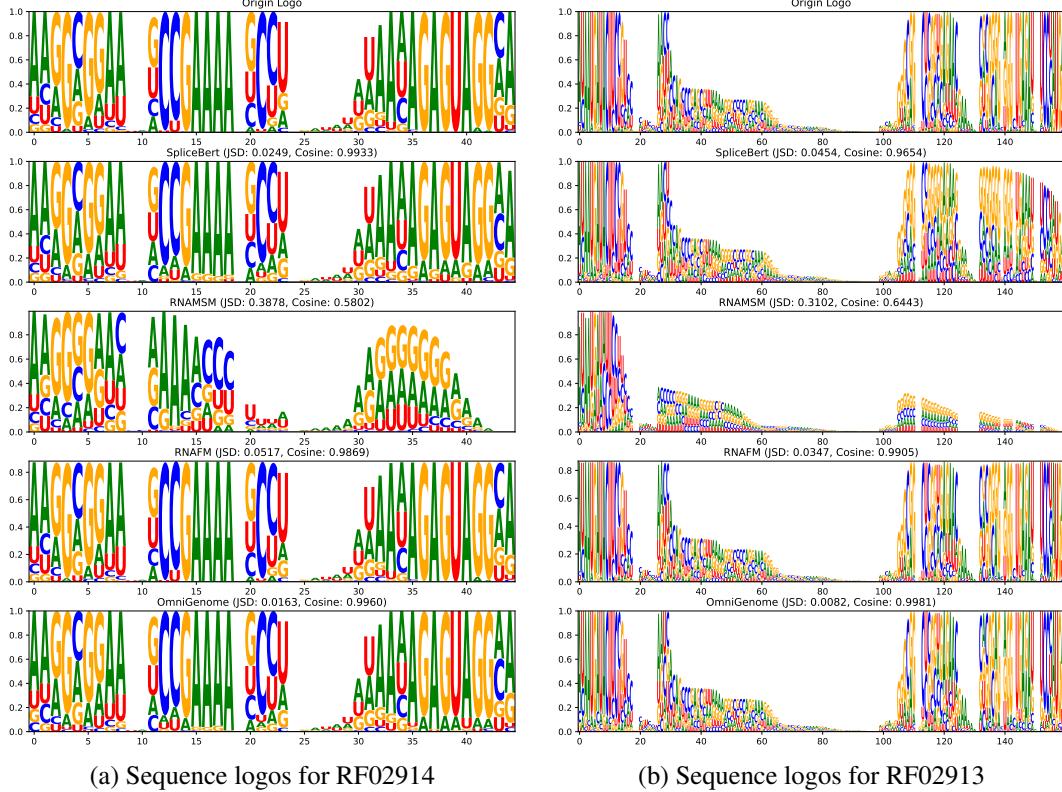


Figure 4: Sequence logo comparison for RNA families after MLM-based augmentation with different GFMs.

0.9654 for RF02913), retaining core features but with some over-smoothing evident in less conserved regions. RNAFM performs similarly or slightly worse than SpliceBERT in this context, with D_{JS} values between 0.0347–0.0517 and Cosine Similarities around 0.9869–0.9905 across the two families, reflecting reasonable but imperfect reconstruction. In stark contrast, RNA-MSM diverges significantly from the native patterns (D_{JS} of 0.3102–0.3878; Cosine Similarity of 0.5802–0.6443), substantially distorting conserved motifs and introducing artifacts, likely due to its architectural design or specific pre-training objectives not being optimized for this type of fine-grained motif preservation.

Conclusion. OmniGenome demonstrates a strong ability to preserve the position-specific motifs of conserved RNA families, indicating that its pre-training objective (potentially incorporating sequence-structure alignment information) effectively captures and respects the complex dependencies inherent in RNA sequences. This comparative analysis underscores the critical role of model architecture and pre-training strategies in a GFM’s capacity to generate biologically plausible sequence variants. Such sequence-centric interpretability analyses, focused on motif fidelity, provide a crucial framework for validating the reliability of GFMs as sequence augmentation engines, identifying their potential biases, and guiding the selection of appropriate models for downstream tasks that depend on high sequence fidelity, ultimately informing the development of future GFMs that better capture the nuanced features of biological sequences.

5 Conclusion

We present OmniGenBench, a modular, extensible platform that consolidates data hubs, model repositories, automated benchmarking, and interpretability tools into a cohesive ecosystem. By enabling reproducible evaluation and streamlined application across diverse genomic tasks, OmniGenBench provides an infrastructure backbone for accelerating research, standardization, and real-world deployment of GFMs. This initiative marks a step forward in maturing genome-scale AI from isolated prototypes to robust, interpretable, and scalable systems for computational biology.

References

- [1] James Watson and Francis Crick. Molecular structure of nucleic acids: A structure for deoxyribose nucleic acid. *Nature*, 171:737–738, 1953.
- [2] Lin Tang. Large models for genomics. *Nature Methods*, 20:1868, 2023.
- [3] Nature Editorial. Why teachers should explore chatgpt’s potential. *Nature*, 623:15, 2023. doi: 10.1038/d41586-023-03505-5.
- [4] Ibrahim Adeshola and Adeola Praise Adepoju. The opportunities and challenges of chatgpt in education. *Interactive Learning Environments*, 32(10):6159–6172, 2024.
- [5] Zhihan Lv. Generative artificial intelligence in the metaverse era. *Cognitive Robotics*, 3: 208–217, 2023.
- [6] Vinay Chamola, Gaurang Bansal, Tridib Kumar Das, Vikas Hassija, Siva Sai, Jiacheng Wang, Sherali Zeadally, Amir Hussain, Fei Richard Yu, Mohsen Guizani, et al. Beyond reality: The pivotal role of generative ai in the metaverse. *IEEE Internet of Things Magazine*, 7(4):126–135, 2024.
- [7] David C. Edelman and Mark Abraham. Generative AI will change your business. here’s how to adapt. *Harvard Business Review*, April 2023. URL <https://hbr.org/2023/04/generative-ai-will-change-your-business-heres-how-to-adapt>.
- [8] A Shaji George and AS Hovan George. A review of chatgpt ai’s impact on several business sectors. *Partners universal international innovation journal*, 1(1):9–23, 2023.
- [9] Micaela E Consens, Cameron Dufault, Michael Wainberg, Duncan Forster, Mehran Karimzadeh, Hani Goodarzi, Fabian J Theis, Alan Moses, and Bo Wang. Transformers and genome language models. *Nature Machine Intelligence*, pages 1–17, 2025.
- [10] Yanrong Ji, Zhihan Zhou, Han Liu, and Ramana V Davuluri. Dnabert: pre-trained bidirectional encoder representations from transformers model for dna-language in genome. *Bioinformatics*, 37(15):2112–2120, 2021.
- [11] Zhihan Zhou, Yanrong Ji, Weijian Li, Pratik Dutta, Ramana V. Davuluri, and Han Liu. DNABERT-2: efficient foundation model and benchmark for multi-species genome. *CoRR*, abs/2306.15006, 2023. doi: 10.48550/ARXIV.2306.15006. URL <https://doi.org/10.48550/arXiv.2306.15006>.
- [12] Javier Mendoza-Revilla, Evan Trop, Liam Gonzalez, Maša Roller, Hugo Dalla-Torre, Bernardo P de Almeida, Guillaume Richard, Jonathan Caton, Nicolas Lopez Carranza, Marcin Skwark, et al. A foundational large language model for edible plant genomes. *Communications Biology*, 7(1):835, 2024.
- [13] Katarína Grešová, Vlastimil Martinek, David Čechák, Petr Šimeček, and Panagiotis Alexiou. Genomic benchmarks: a collection of datasets for genomic sequence classification. *BMC Genomic Data*, 24(1):25, 2023.
- [14] Yuchen Ren, Zhiyuan Chen, Lifeng Qiao, Hongtai Jing, Yuchen Cai, Sheng Xu, Peng Ye, Xinzhu Ma, Siqi Sun, Hongliang Yan, Dong Yuan, Wanli Ouyang, and Xihui Liu. BEACON: benchmark for comprehensive RNA tasks and language models. *CoRR*, abs/2406.10391, 2024. doi: 10.48550/ARXIV.2406.10391. URL <https://doi.org/10.48550/arXiv.2406.10391>.
- [15] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*, pages 4171–4186. Association for Computational Linguistics, 2019.
- [16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017. URL <https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547de91fb053c1c4a845aa-Abstract.html>.

- [17] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023.
- [18] Michael Poli, Stefano Massaroli, Eric Nguyen, Daniel Y. Fu, Tri Dao, Stephen Baccus, Yoshua Bengio, Stefano Ermon, and Christopher Ré. Hyena hierarchy: Towards larger convolutional language models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 28043–28078. PMLR, 2023. URL <https://proceedings.mlr.press/v202/poli23a.html>.
- [19] Eric Nguyen, Michael Poli, Marjan Faizi, Armin W. Thomas, Callum Birch-Sykes, Michael Wornow, Aman Patel, Clayton M. Rabideau, Stefano Massaroli, Yoshua Bengio, Stefano Ermon, Stephen A. Baccus, and Christopher Ré. Hyenadna: Long-range genomic sequence modeling at single nucleotide resolution. *CoRR*, abs/2306.15794, 2023. doi: 10.48550/ARXIV.2306.15794. URL <https://doi.org/10.48550/arXiv.2306.15794>.
- [20] Eric Nguyen, Michael Poli, Matthew G Durrant, Brian Kang, Dhruva Katrekar, David B Li, Liam J Bartie, Armin W Thomas, Samuel H King, Garyk Brixi, et al. Sequence modeling and design from molecular to genome scale with evo. *Science*, 386(6723):eado9336, 2024.
- [21] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *CoRR*, abs/2312.00752, 2023. doi: 10.48550/ARXIV.2312.00752. URL <https://doi.org/10.48550/arXiv.2312.00752>.
- [22] Yair Schiff, Chia-Hsiang Kao, Aaron Gokaslan, Tri Dao, Albert Gu, and Volodymyr Kuleshov. Caduceus: Bi-directional equivariant long-range DNA sequence modeling. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL <https://openreview.net/forum?id=mk3A5IUdn8>.
- [23] Heng Yang and Ke Li. Omnidrome: Aligning RNA sequences with secondary structures in genomic foundation models. *CoRR*, abs/2407.11242, 2024. doi: 10.48550/ARXIV.2407.11242. URL <https://doi.org/10.48550/arXiv.2407.11242>.
- [24] Jiayang Chen, Zhihang Hu, Siqi Sun, Qingxiong Tan, Yixuan Wang, Qinze Yu, Licheng Zong, Liang Hong, Jin Xiao, Tao Shen, et al. Interpretable rna foundation model from unannotated data for highly accurate rna structure and function predictions. *bioRxiv*, pages 2022–08, 2022.
- [25] Zhen Tan, Yinghan Fu, Gaurav Sharma, and David H Mathews. Turbofold ii: Rna structural alignment and secondary structure prediction informed by multiple homologs. *Nucleic acids research*, 45(20):11570–11581, 2017.
- [26] Padideh Danaee, Mason Rouches, Michelle Wiley, Dezhong Deng, Liang Huang, and David Hendrix. bprna: large-scale automated annotation and analysis of rna secondary structure. *Nucleic acids research*, 46(11):5381–5394, 2018.
- [27] David H Mathews. How to benchmark rna secondary structure prediction accuracy. *Methods*, 162:60–67, 2019.
- [28] Ioanna Kalvari, Eric P Nawrocki, Nancy Ontiveros-Palacios, Joanna Argasinska, Kevin Lamkiewicz, Manja Marz, Sam Griffiths-Jones, Claire Toffano-Nioche, Daniel Gautheret, Zasha Weinberg, et al. Rfam 14: expanded coverage of metagenomic, viral and microrna families. *Nucleic Acids Research*, 49(D1):D192–D200, 2021.
- [29] Kizzmekia S Corbett, Darin K Edwards, Sarah R Leist, Olubukola M Abiona, Seyhan Boyoglu-Barnum, Rebecca A Gillespie, Sunny Himansu, Alexandra Schäfer, Cynthia T Ziwawo, Anthony T DiPiazza, et al. Sars-cov-2 mrna vaccine design enabled by prototype pathogen preparedness. *Nature*, 586(7830):567–571, 2020.
- [30] Ken Chen, Yue Zhou, Maolin Ding, Yu Wang, Zhixiang Ren, and Yuedong Yang. Self-supervised learning on millions of pre-mrna sequences improves sequence-based rna splicing prediction. *bioRxiv*, pages 2023–01, 2023.

- [31] Yair Schiff, Chia-Hsiang Kao, Aaron Gokaslan, Tri Dao, Albert Gu, and Volodymyr Kuleshov. Caduceus: Bi-directional equivariant long-range dna sequence modeling. *arXiv preprint arXiv:2403.03234*, 2024.
- [32] Kengo Sato, Manato Akiyama, and Yasubumi Sakakibara. Rna secondary structure prediction using deep learning with thermodynamic integration. *Nature communications*, 12(1):941, 2021.
- [33] Yikun Zhang, Mei Lang, Juhong Jiang, Zhiqiang Gao, Fan Xu, Thomas Litfin, Ke Chen, Jaswinder Singh, Xiansong Huang, Guoli Song, et al. Multiple sequence alignment-based rna language model and its application to structural inference. *Nucleic Acids Research*, 52(1):e3–e3, 2024.
- [34] Žiga Avsec, Roman Kreuzhuber, Johnny Israeli, Nancy Xu, Jun Cheng, Avanti Shrikumar, Abhimanyu Banerjee, Daniel S Kim, Thorsten Beier, Lara Urban, et al. The kipoi repository accelerates community exchange and reuse of predictive models for genomics. *Nature biotechnology*, 37(6):592–600, 2019.
- [35] Frederic Runge, Karim Farid, Jorg KH Franke, and Frank Hutter. Rnabench: A comprehensive library for in silico rna modelling. *bioRxiv*, pages 2024–01, 2024.
- [36] Zicheng Liu, Jiahui Li, Siyuan Li, Zelin Zang, Cheng Tan, Yufei Huang, Yajing Bai, and Stan Z Li. Genbench: A benchmarking suite for systematic evaluation of genomic foundation models. *arXiv preprint arXiv:2406.01627*, 2024.
- [37] Jacob West-Roberts, Joshua Kravitz, Nishant Jha, Andre Cornman, and Yunha Hwang. Diverse genomic embedding benchmark for functional evaluation across the tree of life. *bioRxiv*, pages 2024–07, 2024.
- [38] The Galaxy Community. The galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2022 update. *Nucleic Acids Research*, 50(W1):W345–W351, 2022.
- [39] Pascal Notin, Aaron Kollasch, Daniel Ritter, Lood Van Niekerk, Steffanie Paul, Han Spinner, Nathan Rollins, Ada Shaw, Rose Orenbuch, Ruben Weitzman, et al. Proteingym: Large-scale benchmarks for protein fitness prediction and design. *Advances in Neural Information Processing Systems*, 36, 2024.
- [40] Christian Dallago, Jody Mou, Kadina E Johnston, Bruce J Wittmann, Nicholas Bhattacharya, Samuel Goldman, Ali Madani, and Kevin K Yang. Flip: Benchmark tasks in fitness landscape inference for proteins. *bioRxiv*, pages 2021–11, 2021.
- [41] Minghao Xu, Zuobai Zhang, Jiarui Lu, Zhaocheng Zhu, Yangtian Zhang, Ma Chang, Runcheng Liu, and Jian Tang. Peer: a comprehensive and multi-task benchmark for protein sequence understanding. *Advances in Neural Information Processing Systems*, 35:35156–35173, 2022.
- [42] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A A Kohl, Andrew J Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishabh Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, 2021. doi: 10.1038/s41586-021-03819-2.
- [43] Richard Evans, Michael O'Neill, Alexander Pritzel, Natasha Antropova, Andrew Senior, Tim Green, Augustin Žídek, Russ Bates, Sam Blackwell, Jason Yim, Olaf Ronneberger, Sebastian Bodenstein, Michal Zielinski, Alex Bridgland, Anna Potapenko, Andrew Cowie, Kathryn Tunyasuvunakool, Rishabh Jain, Ellen Clancy, Pushmeet Kohli, John Jumper, and Demis Hassabis. Protein complex prediction with alphafold-multimer. *bioRxiv*, 2021. doi: 10.1101/2021.10.04.463034. URL <https://www.biorxiv.org/content/early/2021/10/04/2021.10.04.463034>.
- [44] Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J Ballard, Joshua Bambrick, et al. Accurate structure prediction of biomolecular interactions with alphafold 3. *Nature*, pages 1–3, 2024.

- [45] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Sal Candido, et al. Language models of protein sequences at the scale of evolution enable accurate structure prediction. *BioRxiv*, 2022: 500902, 2022.
- [46] Yanrong Ji, Zhihan Zhou, Han Liu, and Ramana V. Davuluri. DNABERT: pre-trained bidirectional encoder representations from transformers model for dna-language in genome. *Bioinform.*, 37(15):2112–2120, 2021.
- [47] 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. The nucleotide transformer: Building and evaluating robust foundation models for human genomics. *bioRxiv*, pages 2023–01, 2023.
- [48] 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*, pages 2024–03, 2024.
- [49] Fan Yang, Wenchuan Wang, Fang Wang, Yuan Fang, Duyu Tang, Junzhou Huang, Hui Lu, and Jianhua Yao. scbert as a large-scale pretrained deep language model for cell type annotation of single-cell rna-seq data. *Nat. Mac. Intell.*, 4(10):852–866, 2022. doi: 10.1038/S42256-022-00534-Z. URL <https://doi.org/10.1038/s42256-022-00534-z>.
- [50] Manato Akiyama and Yasubumi Sakakibara. Informative rna base embedding for rna structural alignment and clustering by deep representation learning. *NAR genomics and bioinformatics*, 4(1):lqac012, 2022.
- [51] Ning Wang, Jiang Bian, Yuchen Li, Xuhong Li, Shahid Mumtaz, Linghe Kong, and Haoyi Xiong. Multi-purpose rna language modelling with motif-aware pretraining and type-guided fine-tuning. *Nature Machine Intelligence*, pages 1–10, 2024.
- [52] Logan Hallee, Nikolaos Rafailidis, and Jason P Gleghorn. cdsbert-extending protein language models with codon awareness. *bioRxiv*, 2023.
- [53] Yanyi Chu, Dan Yu, Yupeng Li, Kaixuan Huang, Yue Shen, Le Cong, Jason Zhang, and Mengdi Wang. A 5' utr language model for decoding untranslated regions of mrna and function predictions. *Nature Machine Intelligence*, pages 1–12, 2024.
- [54] Yuning Yang, Gen Li, Kuan Pang, Wuxinhao Cao, Xiangtao Li, and Zhaolei Zhang. Deciphering 3'utr mediated gene regulation using interpretable deep representation learning. *bioRxiv*, pages 2023–09, 2023.
- [55] Xi Wang, Ruichu Gu, Zhiyuan Chen, Yongge Li, Xiaohong Ji, Guolin Ke, and Han Wen. Uni-rna: universal pre-trained models revolutionize rna research. *bioRxiv*, pages 2023–07, 2023.
- [56] Guillaume Richard, Bernardo P de Almeida, Hugo Dalla-Torre, Christopher Blum, Lorenz Hexemer, Priyanka Pandey, Stefan Laurent, Marie P Lopez, Alexander Laterre, Maren Lang, et al. Chatnt: A multimodal conversational agent for dna, rna and protein tasks. *bioRxiv*, pages 2024–04, 2024.
- [57] Haopeng Yu, Heng Yang, Wenqing Sun, Zongyun Yan, Xiaofei Yang, Huakun Zhang, Yiliang Ding, and Ke Li. An interpretable rna foundation model for exploring functional rna motifs in plants. *Nature Machine Intelligence*, 6(12):1616–1625, 2024.
- [58] Jeehyung Lee, Wipapat Kladwang, Minjae Lee, Daniel Cantu, Martin Azizyan, Hanjoo Kim, Alex Limpaecher, Snehal Gaikwad, Sungroh Yoon, Adrien Treuille, et al. Rna design rules from a massive open laboratory. *Proceedings of the National Academy of Sciences*, 111(6): 2122–2127, 2014.
- [59] Weizhong Li and Adam Godzik. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics*, 22(13):1658–1659, 2006.

- [60] Stephen F Altschul, Warren Gish, Webb Miller, Eugene W Myers, and David J Lipman. Basic local alignment search tool. *Journal of molecular biology*, 215(3):403–410, 1990.
- [61] Ronny Lorenz, Stephan H Bernhart, Christian Höner zu Siederdissen, Hakim Tafer, Christoph Flamm, Peter F Stadler, and Ivo L Hofacker. Viennarna package 2.0. *Algorithms for molecular biology*, 6:1–14, 2011.
- [62] Carlos Outeiral and Charlotte M. Deane. Codon language embeddings provide strong signals for use in protein engineering. *Nature Machine Intelligence*, 6:170–179, 2024. doi: 10.1038/s42256-024-00791-0.
- [63] Weijie Yin, Zhaoyu Zhang, Liang He, Rui Jiang, Shuo Zhang, Gan Liu, Xuegong Zhang, Tao Qin, and Zhen Xie. Ernie-rna: An rna language model with structure-enhanced representations. *bioRxiv*, pages 2024–03, 2024.
- [64] Heng Yang and Ke Li. Mp-rna: Unleashing multi-species rna foundation model via calibrated secondary structure prediction. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 5278–5296, 2024.
- [65] Rafael Josip Penić, Tin Vlašić, Roland G. Huber, Yue Wan, and Mile Šikić. Rinalmo: General-purpose rna language models can generalize well on structure prediction tasks. *arXiv*, 2024.
- [66] Carnegie Endowment for International Peace. Mitigating risks from gene editing and synthetic biology: Global governance priorities. *Carnegie Endowment*, 2024. URL <https://carnegieendowment.org/research/2024/10/mitigating-risks-from-gene-editing-and-synthetic-biology-global-governance-priorities>.
- [67] iScience. Safety by design: Biosafety and biosecurity in the age of synthetic biology. *iScience*, 2023. URL <https://www.cell.com/iscience/fulltext/S2589-0042%2823%2900242-0>.
- [68] American Society of Human Genetics. The economic impact and functional applications of human genetics and genomics, 2021. URL <https://www.ashg.org/wp-content/uploads/2021/05/ASHG-TEconomy-Impact-Report-Final.pdf>.
- [69] Frontiers in Bioengineering and Biotechnology. Safety risks and ethical governance of biomedical applications of synthetic biology. *Frontiers*, 2023. URL <https://www.frontiersin.org/journals/bioengineering-and-biotechnology/articles/10.3389/fbioe.2023.1292029/full>.
- [70] IUCN. Genetic frontiers for conservation: An assessment of synthetic biology and biodiversity conservation. 2019. URL <https://portals.iucn.org/library/sites/library/files/documents/2019-012-En.pdf>.
- [71] ScienceDirect. Ethical framework on risk governance of synthetic biology. *ScienceDirect*, 2023. URL <https://www.sciencedirect.com/science/article/pii/S2588933823000201>.
- [72] PMC. Genomic medicine on the frontier of precision medicine. *PMC*, 2022. URL <https://pmc.ncbi.nlm.nih.gov/articles/PMC9167337/>.
- [73] MDPI. Innovations in genomics and big data analytics for personalized medicine and health care: A review. *International Journal of Molecular Sciences*, 2022. URL <https://www.mdpi.com/1422-0067/23/9/4645>.
- [74] Wired. Combining ai and crispr will be transformational. *Wired*, 2024. URL <https://www.wired.com/story/combining-ai-and-crispr-will-be-transformational>.
- [75] Liebert. Revolutionizing biological science: The synergy of genomics in agriculture. *Liebert*, 2023. URL <https://www.liebertpub.com/doi/10.1089/omi.2023.0197>.

Table of Contents in Appendices

A	Related Works	16
A.1	Benchmarking Platforms and Tools for Genomic Models	16
A.2	Evolution of Genomic Foundation Models	16
B	Extended Overall Benchmark Results	17
B.1	Rank-based Performance	17
C	Integrated Benchmark Suites in OmniGenBench	19
C.1	RNA Genomic Benchmark (RGB)	19
C.2	Plant Genomic Benchmark	19
C.3	Genomic Understanding Evaluation	20
C.4	Genomic Benchmarks	21
C.5	BEACON Benchmark	22
C.6	Data Filtering in Benchmarking	23
D	Detailed Benchmark Performance Report	23
D.1	Evaluation Settings in Benchmarking	23
D.2	Evaluation GFMs in Benchmarks	24
D.3	RNA Genomic Benchmark (RGB)	25
D.4	Plant Genomic Benchmark (PGB)	26
D.5	Genomic Understanding Evaluation (GUE)	27
D.6	Genomic Benchmarks (GB)	27
D.7	BEACON Results	28
D.8	Overall Discussion	28
E	Integrated Genomic Foundation Models in OmniGenBench	29
F	Public Leaderboard	31
G	Tutorials	31
G.1	Automated Benchmarking with AutoBench	31
G.2	Fine-tuning for RNA Secondary Structure Prediction	33
G.3	Zero-Shot RNA Secondary Structure Prediction	35
G.4	Generating RNA Embeddings	35
G.5	Computational RNA Sequence Design	36
G.6	Sequence Augmentation via Masked Language Modeling	37
H	Interpretability Cases for Genomic Foundation Models	38
H.1	Feature-Embedding Analysis	38
H.2	Attention Representation Inspection	40
H.3	Development Environment	43
I	Ethical Considerations	43
J	Societal Impact	43
K	Limitations	44

A Related Works

The development of robust platforms for GFMs intersects with advancements in genomic benchmarking tools and the evolution of GFMs themselves.

A.1 Benchmarking Platforms and Tools for Genomic Models

Effective evaluation is crucial for advancing genomic models. Several benchmarking tools and platforms have emerged, yet most exhibit limitations in scope, extensibility, or GFM-specific support.

Genomic Benchmarking Suites. Early efforts like Kipoi [34] focused on standardizing access to trained models for genomic sequence analysis, primarily classic predictive models rather than modern GFM, and offered limited benchmarking capabilities. More recent suites have targeted specific modalities or tasks. For instance, RNABench [35] provides benchmarks for RNA-centric tasks like secondary structure prediction but lacks comprehensive support for evaluating diverse pre-trained GFMs. GenBench [36] offers a modular framework for DNA sequence evaluation but does not extend to RNA and can be challenging for users not deeply familiar with its architecture. BEACON [14] is a notable recent effort for RNA foundation models, introducing several RNA evaluation datasets. However, our experience indicates that its complex environment setup can hinder model portability and benchmarking scalability. DEGB [37] evaluates genomic embeddings for both nucleic acids and amino acids but is restricted by the small scale of its benchmarks and does not support the evaluation of GFMs in downstream application contexts. Broader platforms like Galaxy [38] democratize pipeline execution but lack the specific abstractions and integrated GFM support necessary for streamlined GFM research. Commercial cloud platforms (e.g., DNAnexus, Seven Bridges) provide computational infrastructure but typically treat GFMs as external entities rather than integrated components.

Protein Language Model Benchmarks. The protein language modeling domain has also seen the development of specialized benchmarks, such as ProteinGym [39] for fitness prediction, FLIP [40] for sequence-function relationships, and PEER [41] for diverse protein-related tasks. While these tools are valuable in their specific area, their focus is on protein sequences and associated tasks, and they are not directly applicable to the unique challenges of DNA and RNA GFM benchmarking.

Limitations of Existing Genomic Benchmarks. A significant gap remains: existing tools often provide narrow task coverage (e.g., DNA-only or RNA-only), lack extensibility for new datasets and models, or do not support the comprehensive end-to-end evaluation of GFMs, from pre-training to downstream application and interpretability. The diverse architectures and tokenization strategies of GFMs further complicate direct comparisons using these fragmented tools. This landscape underscores the need for a unified and extensible platform like OmniGenBench, which is designed to address these limitations by offering broad support for both DNA and RNA GFMs, flexible benchmark integration, and ease of use for the genomics community.

A.2 Evolution of Genomic Foundation Models

The application of foundation models to biological sequences, inspired by successes in natural language processing (NLP), has rapidly advanced, though progress in DNA and RNA modeling has historically lagged behind protein modeling (e.g., AlphaFold [42, 43, 44], ESM [45]).

DNA Foundation Models. Early DNA GFMs adapted NLP architectures. DNABERT [46] applied the BERT [15] architecture to DNA, with DNABERT2 [11] later enhancing performance by adopting byte-pair encoding (BPE) over k-mer tokenization. Subsequently, models like Nucleotide Transformers V2 (NT-V2) [47], AgroNT [12] (focused on plant DNA), and SegmentNT [48] explored scaling to billions of parameters, achieving strong results on various DNA understanding tasks. However, specialized models like AgroNT have shown limited transferability to other modalities like RNA. To handle the long-sequence nature of genomes, auto-regressive models such as HyenaDNA [19] and Evo [20] have also been introduced, emphasizing long-range dependency modeling.

RNA Foundation Models. RNA GFM development has faced challenges due to the relative scarcity of large-scale, annotated RNA datasets. Initial models like scBERT [49] (for single-cell RNA), RNABERT [50], RNA-FM [24], RNA-MSM [33], and RNAErnie [51] were often trained on smaller databases. Some GFMs target specific RNA types, such as coding sequences (CDSBERT [52]), 5'UTRs (5UTR-LM [53]), 3'UTRs (3UTRBERT [54]), or precursor mRNAs (SpliceBERT [30]), which can limit their generalizability across the diverse RNA landscape. While models like Uni-RNA [55] have reported strong performance due to large-scale pre-training, their closed-source nature restricts comparative analysis and community adoption.

Multimodal and Conversational Genomic Agents. More recently, models like ChatNT [56] have emerged as multimodal conversational agents capable of assisting with tasks across DNA, RNA, and

protein sequences. These tools aim to integrate various AI capabilities to facilitate broader research in genomics and proteomics.

Positioning of OmniGenBench. Despite the proliferation of GFMs, their practical application and comparative evaluation are hindered by the lack of a unified platform that can accommodate diverse model architectures, data types, and downstream tasks. **OmniGenBench** aims to fill this void by providing the necessary infrastructure to integrate, benchmark, and apply these varied GFMs effectively, thereby fostering a more cohesive and rapidly advancing GFM ecosystem.

B Extended Overall Benchmark Results

B.1 Rank-based Performance

To provide a more nuanced view of where each GFM excels or struggles, we present rank-normalized radar charts in Figure 5. For each benchmark suite, an individual radar chart is generated for each of the baseline GFMs. The axes of these charts represent distinct task categories within the respective suite. A model’s performance on each axis is its average rank for tasks in that category (rank 1 = best). Consequently, a smaller and more centrally-focused polygon for a model indicates stronger and more balanced performance across the task categories of that benchmark suite. The average rank of each model across all task categories within a suite is also noted on its respective plot.

Results. The rank-normalized radar charts (Figure 5) reveal distinct GFM performance profiles across the four benchmark suites. Each GFM’s individual radar plot, with its average rank displayed, visualizes strengths and weaknesses across task categories (axes). In these plots, smaller and more central polygons indicate superior, balanced performance across the task categories of that benchmark suite. The average rank of each model across all task categories within a suite is also noted on its respective plot. For RGB in Figure 5a, OmniGenome (avg. rank 1.00) demonstrates clear dominance with a near-central polygon. RNA-FM (avg. rank 2.29) is a strong second, followed by SpliceBERT (avg. rank 3.00). DNA-centric models like DNABERT-2 (avg. rank 10.00) exhibit larger polygons, underscoring weaker RNA task transferability. Structure-aware models, particularly OmniGenome and RNA-FM, excel in structure-related task categories (e.g., SSP axis). As for the PGB in Figure 5b, OmniGenome (avg. rank 1.12) again shows exceptional generalization with a compact plot, notably on PolyAP and ProStrP axes, despite no specific plant pre-training. SpliceBERT (avg. rank 3.75) and the specialized NT-v2 (avg. rank 4.12) are also top performers. HyenaDNA (avg. rank 6.38) and Caduceus (avg. rank 6.12) have larger, skewed plots, suggesting variable performance across PGB task categories. For the GUE in Figure 5c, performance is more varied. OmniGenome (avg. rank 3.00) and SpliceBERT (avg. rank 2.86) show the best overall balance. DNABERT-2 (avg. rank 3.71) and NT-V2 (avg. rank 5.86) are also highly competitive. Specific strengths are visible, e.g., SpliceBERT excels on the Human SSP axis, DNABERT-2 on Mouse TF-M. OmniGenome, while strong overall, displays a comparatively larger rank on the Mouse TF-M axis, indicating a specific performance gap. Within the GB in Figure 5d, OmniGenome (avg. rank 2.78) leads with a compact radar. NT-V2 (avg. rank 4.11), RNA-FM (avg. rank 3.33), and SpliceBERT (avg. rank 3.67) follow closely. DNABERT-2’s plot highlights its strength on the HCE axis but relative weakness on DRE/HRE, potentially due to tokenizer differences. SpliceBERT and NT-V2 show strong performance on their respective favored task categories (e.g., DOW/HNP and DME/DRE).

These granular radar visualizations effectively highlight model-specific strengths and weaknesses across diverse genomic task categories beyond what aggregate scores reveal. This detailed insight is crucial for informed GFM selection for downstream tasks and for guiding future model development.

C Integrated Benchmark Suites in OmniGenBench

There are five benchmark collections in this paper: RGB, PGB, GUE, GB, and BENCON, with 123 tasks⁹ complied in total.

⁹We define a task as a downstream task compiled from with a dataset. For example, the tasks in RGB are compiled from the datasets in RGB.

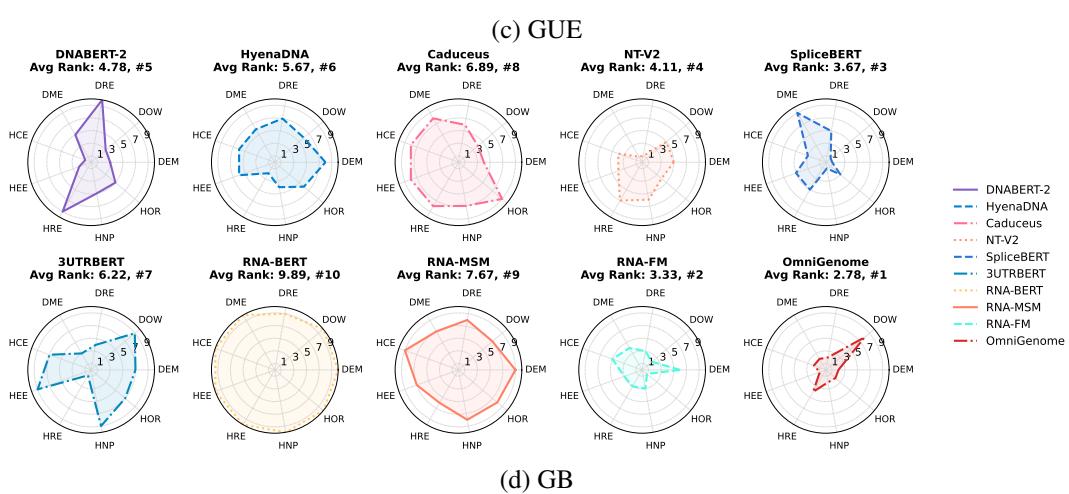
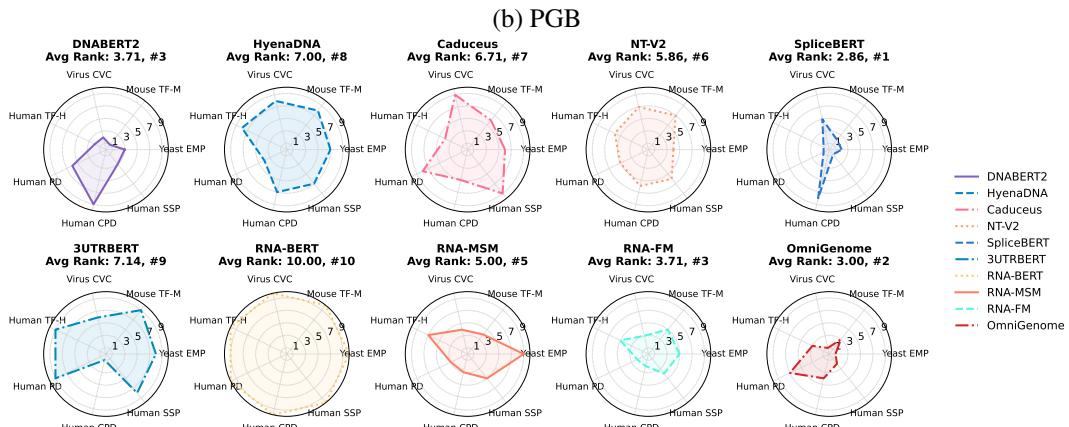
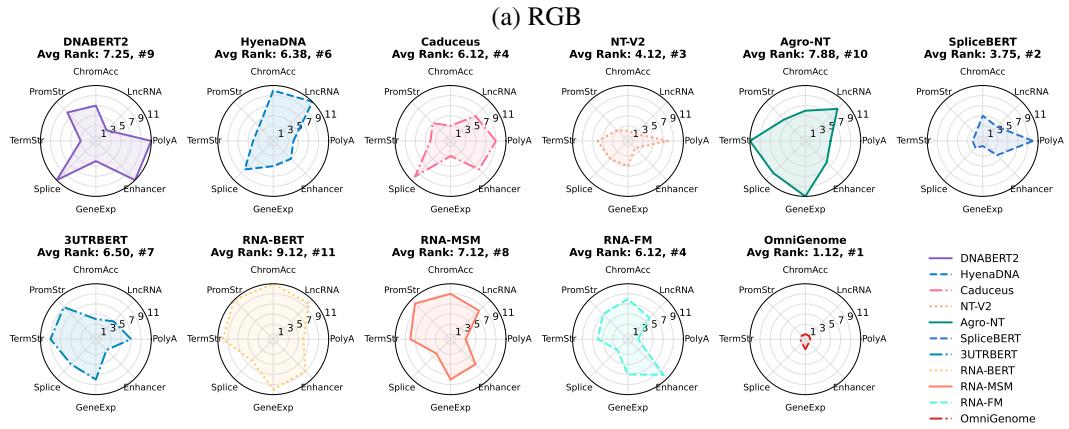
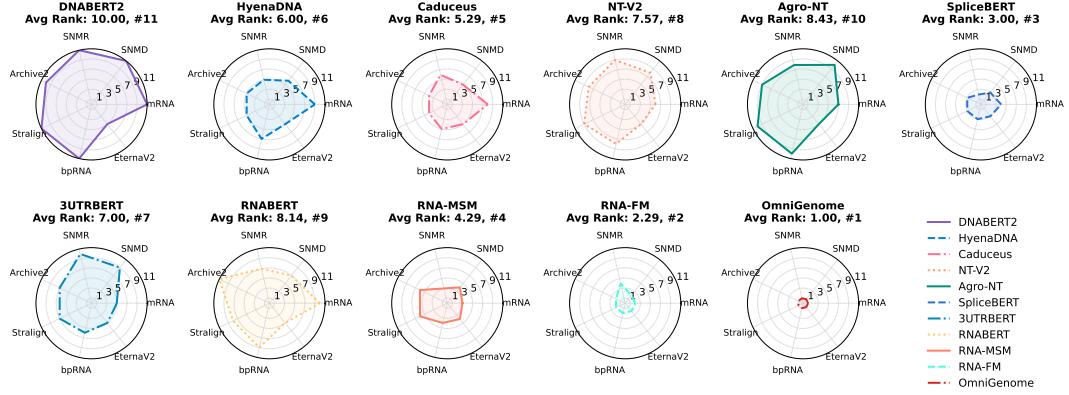


Figure 5: Rank-based radar charts comparisons between GFMs across four genomic benchmark suites.

Table 1: Statistics of RGB subtasks. ‘‘Cls’’ = classification, ‘‘Reg’’ = regression. ‘—’ denotes values not explicitly reported in source papers.

Task	Type	#Train/Val/Test	Classes	Metric	Len. (max/mean)	Source
SNMD	Token Cls	8 000/1 000/1 000	2	AUC	200/200	This work
SNMR	Token Cls	8 000/1 000/1 000	4	macro-F1	200/200	This work
mRNA	Token Reg	1 735/193/192	—	RMSE	107/107	Kaggle
bpRNA-1m	Token Cls	10 814/1 300/1 305	3	macro-F1	512/134	[26]
ArchiveII	Token Cls	2 278/285/285	3	macro-F1	500/151	[27]
RNAStralign	Token Cls	17 483/2 186/2 185	3	macro-F1	500/142	[25]
EternaV2	Seq Cls	20 430/3 607/3 607*	2	Accuracy	180/101	[58]
Region-Ara	Token Cls	12 838/1 604/1 604	3	macro-F1	1024/1024	[57]
Region-Rice	Token Cls	11 412/1 426/1 426	3	macro-F1	1024/1024	[57]
TE-Ara	Seq Cls	9 644/1 206/1 206	2	macro-F1	500/500	[57]
TE-Rice	Seq Cls	8 102/1 013/1 013	2	macro-F1	500/500	[57]

C.1 RNA Genomic Benchmark (RGB)

RGB now comprises **12** single-nucleotide (SN)-level tasks spanning mutation analysis, RNA degradation, secondary-structure prediction, *de-novo* RNA design, plant-specific genic-region annotation, and translation-efficiency (TE) modelling. Sequences longer than 512 nt ($< 3\%$ of bpRNA/ArchiveI-II/Stralign) are discarded, leaving a length range of 107-512 nt, which is sufficient for most RNA-understanding workloads. The benchmark therefore probes both cross-species generalisation (human, plant, virus) and fine-grained SN-level reasoning. Brief task synopses follow; full statistics appear in Table 1.

- **SN Mutation Detection (SNMD)** Binary token classification that flags mutated positions in synthetic plant RNAs (up to ten random SNVs per sequence). Loss: cross-entropy. (eg, used to screen deleterious variants).
- **SN Mutation Repair (SNMR)** Four-way token classification (A/U/C/G) that proposes the corrected base at each mutated position; shares the SNMD splits.
- **mRNA Degradation Rate Prediction (mRNA)** Token-level regression (MSE) on the Stanford *OpenVaccine* dataset that measures in-line hydrolysis of 102-130 nt constructs.
- **RNA Secondary-Structure Prediction** bpRNA-1m [26], ArchiveII [27], and Stralign [25]: three-label (‘‘(’’, ‘‘.’’, ‘‘)’’) token classification; metric = macro-F1.
- **RNA Design (EternaV2)** Sequence-level accuracy on *EternaV2*, which contains $\sim 27\text{k}$ player-designed RNAs with desired secondary structures. Models must predict whether a candidate folds into the target structure, mirroring practical design pipelines.
- **Genic-Region Classification** Two plant species from PlantRNA-FM [57]:
 - **Region-Ara** (*Arabidopsis*)
 - **Region-Rice** (*Oryza sativa*)

Token-level 3-way classification of 5’UTR, CDS, 3’UTR; metric = macro-F1. These tasks test whether GFM capture plant-specific transcript organisation [57].

- **Translation-Efficiency Prediction** Again using PlantRNA-FM data [57]:
 - **TE-Ara** (*Arabidopsis*)
 - **TE-Rice** (*Rice*)

Sequence-level binary classification (high vs. low TE) on 5'-UTRs; metric = AUC. Successful models must integrate subtle sequence cues governing ribosome loading [57].

Table 2 shows the virtual examples of different datasets in RGB. Please refer to our supplementary materials to find the datasets for more details.

C.2 Plant Genomic Benchmark

PGB [12] provides a comprehensive suite of datasets designed to evaluate and improve the predictive capabilities of GFM in plant biology. This benchmark, as shown in Table 3, encompasses a range

Table 2: The virtual input and output examples in the four benchmarks. The “...” represents the sequences that are omitted for better presentation and the red color indicates the wrong prediction in classification tasks. In the mRNA dataset, all single nucleotides have three values to predict. Note that “T” and “U” can be regarded as the same symbol in RNA sequences and depend on different datasets.

Genome Type	Dataset	Column	Examples
RNA	SNMD	Input Sequence	G A G T A ... T T G A G
		True Label	0 0 1 0 0 ... 0 0 1 0 0
		Prediction	0 0 0 0 0 ... 0 0 1 0 0
	SNMR	Input Sequence	T A C G A ... C T G A T
		True Label	T A C A A ... G T A A T
		Prediction	T A C A A ... C T G A T
	mRNA	Input Sequence	G G ... A C
		True Label	[0.1,0.3,0.2] [0.8,0.4,0.1] ... [0.9,0.4,0.3] [0.5,0.2,0.6]
		Prediction	[0.1,0.3,0.2] [0.8,0.4,0.1] ... [0.9,0.4,0.3] [0.5,0.2,0.6]
	bpRNA	Input Sequence	G G C G A ... C U U U U
		True Label	((.)))
		Prediction	(((. . . .))))
	Classification	Input Sequence	A T C G A ... T A G
		True Label	1
		Prediction	0
DNA	Regression	Input Sequence	G C C A T ... G C T
		True Label	2.56
		Prediction	2.45

Table 3: The genomic tasks in the Plant Genomic Benchmark. This table briefly enumerates each task by name, the number of datasets available, the type of classification or regression analysis required, the range of sequence lengths, and the total number of samples in each dataset. “Cls.” indicates classification. Please find the dataset details of PGB in Agro-NT.

Task	# of datasets	Task Type	Total # of examples	# of classes	Metric	Sequence length
Polyadenylation	6	Classification	738,918	2	macro F1	400
Splice site	2	Classification	4,920,835	2	macro F1	398
LncRNA	2	Classification	58,062	6	macro F1	101 – 6000
Promoter strength	2	Regression	147,966	—	RMSE	170
Terminator strength	2	Regression	106,818	—	RMSE	170
Chromatin accessibility	7	Multi-label Cls.	5,149,696	9-19 (multi-label)	macro F1	1,000
Gene expression	6	Multi-variable Reg.	206,358	—	RMSE	6,000
Enhancer region	1	Classification	18,893	2	macro F1	1,000

of critical genomic tasks, including binary classification, single and multi-variable regression, and multi-label classification, addressing various aspects of plant genomics such as RNA processing, gene expression, and chromatin accessibility. By integrating diverse genomic tasks, the PGB aims to facilitate advanced research and development in plant genomics, offering a robust platform for the assessment and enhancement of model performance across different plant species. To obtain a detailed description of PGB, please refer to Agro-NT [12].

C.3 Genomic Understanding Evaluation

GUE [11] serves as a DNA genomic benchmark, encompassing 36 datasets across nine crucial genome analysis tasks applicable to a variety of species. Similar to PGB and GB, it is used for evaluating the generalizability of OmniGenBench on DNA genome benchmarking. To thoroughly assess the capabilities of genome foundation models across sequences of varying lengths, tasks have been chosen with input lengths spanning from 70 to 10,000. The brief statistics for each dataset included in the GUE benchmark are displayed in Table 4, and the task descriptions are available in Zhou et al. [11]. Due to resource limitations, we do not include large-scale FMs in this benchmark, e.g., Agro-NT. Besides, all reported scores are on a randomly stratified 10 k-sample subset per split; hence they are *not* directly comparable to the original GUE leaderboard.

C.4 Genomic Benchmarks

GB is also a DNA-oriented FM benchmark suite, which can be used for generalizability evaluation of OmniGenome. It contains a well-curated collection of datasets designed for the classification

Table 4: Statistics of tasks in the GUE, these details can be found in Section B.2. from Zhou et al. [11].

Task	Metric	Datasets	Training	Validation	Testing
Core Promoter Detection	macro F1	tata	4,904	613	613
		notata	42,452	5,307	5,307
		all	47,356	5,920	5,920
Promoter Detection	macro F1	tata	4,904	613	613
		notata	42,452	5,307	5,307
		all	47,356	5,920	5,920
Transcription Factor Prediction (Human)	macro F1	wgEncodeEH000552	32,378	1,000	1,000
		wgEncodeEH000606	30,672	1,000	1,000
		wgEncodeEH001546	19,000	1,000	1,000
		wgEncodeEH001776	27,497	1,000	1,000
		wgEncodeEH002829	19,000	1,000	1,000
Splice Site Prediction	macro F1	reconstructed	36,496	4,562	4,562
Transcription Factor Prediction (Mouse)	macro F1	Ch12Nrf2\iggrab	6,478	810	810
		Ch12Zrf384hpa004051\iggrab	5,395	674	674
		MeIJun\iggrab	2,620	328	328
		MelMafkDm2p5dStd	1,904	239	239
		MelNelf\iggrab	15,064	1,883	1,883
Epigenetic Marks Prediction	macro F1	H3	11,971	1,497	1,497
		H3K14ac	26,438	3,305	3,305
		H3K36me3	29,704	3,488	3,488
		H3K4me1	25,341	3,168	3,168
		H3K4me2	24,545	3,069	3,069
		H3K4me3	29,439	3,680	3,680
		H3K79me3	23,069	2,884	2,884
		H3K9ac	22,224	2,779	2,779
		H4	11,679	1,461	1,461
		H4ac	27,275	3,410	3,410
Covid Variant Classification	macro F1	Covid	77,669	7,000	7,000
Enhancer Promoter Interaction	macro F1	GM12878	10,000	2,000	2,000
		HeLa-S3	10,000	2,000	2,000
		HUVEC	10,000	2,000	2,000
		IMR90	10,000	2,000	2,000
		K562	10,000	2,000	2,000
		NHEK	10,000	2,000	2,000
Species Classification	macro F1	fungi	8,000	1,000	1,000
		virus	4,000	500	500

of genomic sequences, focusing on regulatory elements across multiple model organisms. This collection facilitates robust comparative analysis and development of genomic FMs. The task names in the original repository are complex, we abbreviate the names as follows:

- DEM corresponds to "Demo Coding vs Intergenomic Seqs"
- DOW is for "Demo Human or Worm"
- DRE represents "Drosophila Enhancers Stark"
- HCE is short for "Human Enhancers Cohn"
- HEE denotes "Human Enhancers Ensembl"
- HRE abbreviates "Human Ensembl Regulatory"
- HNP shortens "Human Nontata Promoters"
- HOR is an abbreviation for "Human Ocr Ensembl"
- DME simplifies "Dummy Mouse Enhancers Ensembl"

The brief statistics for each dataset included in the GUE benchmark are displayed in Table 4. Similar to GUE, we run the evaluation on a subset of GB, where for each task we randomly select at most 10k samples from the original splits, e.g., training, testing and validation (if any) sets.

Table 5: The brief statistics of datasets reported in the genomic benchmark [13].

Task	# of Sequences	# of Classes	Class Ratio	Median Length	Standard Deviation
DME	1,210	2	1.0	2,381	984.4
DEM	100,000	2	1.0	200	0.0
DOW	100,000	2	1.0	200	0.0
DRE	6,914	2	1.0	2,142	285.5
HCE	27,791	2	1 : 9	500	0.0
HEE	154,842	2	1 : 5	269	122.6
HRE	289,061	3	1.2	401	184.3
HNP	36,131	2	1.2	251	0.0
HOR	174,456	2	1.0	315	108.1

C.5 BEACON Benchmark

To address the lack of standardized benchmarks for RNA foundation models, **BEACON** (BEnchmark for Comprehensive RNA Task and Language Models) was proposed as the first large-scale and multi-faceted benchmark tailored for RNA understanding. It includes 13 well-curated tasks spanning three major domains: *Structural Analysis*, *Functional Studies*, and *Engineering Applications*. Comprising over 967,000 sequences ranging from 23 to 1,182 nucleotides in length, BEACON provides a diverse landscape to evaluate RNA-based Genomic Foundation Models (GFMs). Notably, pre-trained RNA language models have surpassed previous task-specific state-of-the-art (SOTA) performance on 8 of the 13 tasks.

According to [14], tasks are grouped as follows, detailed data statistics, metrics, and sources are listed in Table 6.

- **Structural Analysis Tasks:** These tasks probe secondary and tertiary RNA structure, which is fundamental to molecular function and therapeutic applications.
 - **Secondary Structure Prediction (SSP):** Predicts paired (stems) and unpaired (loops, bulges) regions using data from the bpRNA-1m dataset. Evaluated via F1 score.
 - **Contact Map Prediction (CMP):** Identifies nucleotide pairs in spatial proximity ($<8\text{\AA}$). Evaluation metric: Top- L precision.
 - **Distance Map Prediction (DMP):** Predicts pairwise nucleotide distances from 3D structure data. Metric: R^2 .
 - **Structural Score Imputation (SSI):** Imputes missing experimental structural scores (e.g., icSHAPE signals). Evaluated using R^2 .
- **Functional Studies Tasks:** These evaluate the regulatory and functional roles of RNA in gene expression and biological processes.
 - **Splice Site Prediction (SPL):** Classifies each nucleotide as donor, acceptor, or neither. Evaluated with Top- k accuracy.
 - **APA Isoform Prediction (APA):** Predicts usage ratios of alternative polyadenylation sites in the 3' UTR. Metric: R^2 .
 - **Non-coding RNA Function Classification (ncRNA):** Classifies ncRNAs into categories (e.g., miRNA, lncRNA). Metric: sequence-level accuracy.
 - **Modification Prediction (Modif):** Predicts presence of 12 RNA modification types. Metric: AUC.
 - **Mean Ribosome Loading (MRL):** Estimates translation efficiency of mRNA sequences. Metric: R^2 .
- **Engineering Applications Tasks:** These highlight RNA's role in synthetic biology and therapeutic engineering.
 - **Vaccine Degradation Prediction (VDP):** Predicts nucleotide-level degradation rates of vaccine candidates. Evaluated by Mean Columnwise RMSE (MCRMSE).
 - **Programmable RNA Switches (PRS):** Predicts ON/OFF states of synthetic conformational switches. Metric: R^2 .
 - **CRISPR On-Target Prediction (CRI-On):** Estimates gene-editing efficiency at intended target sites. Metric: weighted Spearman correlation.
 - **CRISPR Off-Target Prediction (CRI-Off):** Predicts editing efficiency at unintended loci. Uses same metric as CRI-On.

Table 6: Summary of tasks in the BEACON benchmark. "Cls" = classification; "Reg" = regression. Sources as cited in Ren et al. [14].

Task	Train/Val/Test	Metric	Task Type	Level	Max/Mean Length	Source
<i>Structural Analysis</i>						
SSP	10,814 / 1,300 / 1,305	F1	Multi-label Cls	Nucleotide	499 / 133.8	bpRNA
CMP	188 / 23 / 80	Top- <i>L</i> Precision	Multi-label Cls	Nucleotide	960 / 110.3	RNAcontact
DMP	188 / 23 / 80	R^2	Regression	Nucleotide	960 / 110.3	RNAcontact
SSI	14,049 / 1,756 / 3,095	R^2	Regression	Nucleotide	100 / 100	StructureImpute
<i>Functional Studies</i>						
SPL	144,628 / 18,078 / 16,505	Top- <i>k</i> ACC	Multi-class Cls	Nucleotide	100 / 100	SpliceAI
APA	145,463 / 33,170 / 49,755	R^2	Regression	Sequence	186 / 186	APARENT
ncRNA	5,679 / 650 / 2,400	ACC	Multi-class Cls	Sequence	1,182 / 158.4	GENCODE+Rfam
Modif	304,661 / 3,599 / 1,200	AUC	Multi-label Cls	Sequence	101 / 101	MultiRM
MRL	76,319 / 7,600 / 7,600	R^2	Regression	Sequence	100 / 61.5	Optimus
<i>Engineering Applications</i>						
VDP	2,155 / 245 / 629	MCRMSE	Multi-label Reg	Nucleotide	130 / 118.5	OpenVaccine
PRS	73,227 / 9,153 / 9,154	R^2	Multi-label Reg	Sequence	148 / 148	Angenent-Mari
CRI-On	1,453 / 207 / 416	Spearman Corr	Regression	Sequence	23 / 23	DeepCRISPR
CRI-Off	14,223 / 2,032 / 4,064	Spearman Corr	Regression	Sequence	23 / 23	DeepCRISPR

C.6 Data Filtering in Benchmarking

The pertaining involves RNA sequences and structures prediction, we take the data and annotation leakage problem seriously.

- To avoid structure annotation leakage of downstream benchmarks, the secondary structure predictors for all FMs were randomly initialized for fair comparisons, which means the pre-trained structure predictor of OmniGenBench was not used in benchmarks, except for zero-shot SSP experiments. Please find the source codes for details.
- To reduce sequence leakage caused by evolutionary conservative sequences across multiple species, we use the ch-hit-est tool to calculate the sequence similarity between sequences from the OneKP database and downstream tasks. We adopt the similarity threshold of 80% for ch-hit-est [59] to eliminate sequences whose homogeneous sequences appeared in the OneKP database. Subsequently, we exploit the blastn [60] tool to query potentially leaked sequences in downstream benchmark datasets and further alleviate the data leakage problem. The e-value has been set to 1 for rigorous sequence filtering.

D Detailed Benchmark Performance Report

D.1 Evaluation Settings in Benchmarking

In this experiment, we carefully selected a set of key hyperparameters to optimize model performance. Below are the main hyperparameter settings along with detailed explanations:

- **Dropout:** To prevent the model from overfitting during training, we set the Dropout value to 0.1, meaning that no random neuron dropout is applied during training. This choice was made based on our consideration of model stability and generalization ability.
- **Learning Rate:** We set the learning rate to 2e-5, which is a relatively small value to ensure stable convergence, especially in complex training tasks. A smaller learning rate helps to avoid drastic fluctuations during the training process, leading to more precise optimization.
- **Weight Decay:** We applied a weight decay of 0.01 to control model complexity and prevent overfitting. Weight decay is a regularization technique that effectively constrains the growth of model parameters, maintaining the model's generalization capability.
- **Adam Optimizer:** We used the Adam optimizer with its parameters set to $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The Adam optimizer combines the benefits of momentum and adaptive learning rates, accelerating convergence and adapting to different gradient changes, thereby improving the efficiency and effectiveness of model training.
- **Learning Rate Scheduler:** We opted for a linear decay learning rate scheduler, allowing the learning rate to gradually decrease during training. This strategy helps the model make smaller adjustments as it approaches the optimal solution, ensuring a better convergence outcome.

Table 7: Statistics of RNA and DNA foundation-model baselines. Pre-training data sizes come verbatim from each paper and are therefore not directly comparable.

Model	Tokenisation	#Params	Pre-train Size	Data Source	Species	Sequence Type
DNABERT-2	BPE	117 M	32.5 B tokens	1000 Genomes	Human+135 sp.	DNA
NT-V2-100M	k-mer	96 M	300 B tokens	Multi-source DNA	850 sp.	DNA
HyenaDNA-Large	SNT	47 M	3.2 B tokens	Human GRCh38	Human	DNA
Caduceus	SNT	1.9 M	35 B tokens	Human GRCh38	Human	DNA
Agro-NT-1B	k-mer	985 M	472 B tokens	Ensembl Plants	48 plants	DNA
SpliceBERT	SNT	19 M	2 M seqs	UCSC pre-mRNA	Vertebrates	pre-mRNA
RNA-BERT	SNT	0.5 M	4 069 families	Rfam	Multi-sp.	ncRNA
RNA-MSM	SNT	96 M	4 069 families	Rfam (MSA)	Multi-sp.	ncRNA
RNA-FM	SNT	96 M	23 M seqs	RNAcentral	Multi-sp.	ncRNA
3UTRBERT	k-mer	86 M	20 k seqs	GENCODE UTR	Human	mRNA 3'UTR
OmniGenome	SNT	186 M	54.2 B tokens	OneKP	1124 plants	mRNA/CDS/UTR

- **Batch Size:** The batch size was set to 8. This relatively small batch size helps to efficiently train the model within limited memory resources, particularly when handling large-scale data, enabling a balance between model performance and computational resource usage. For long-sequence tasks (≥ 4 kb) we reduce batch to 1-2 to fit 24 GB GPUs; epochs are capped at 10.
- **# of Epochs:** We set the number of training epochs to 20. This setting ensures that the model can fully learn the features within the data while avoiding the negative effects of overtraining.
- **Early Stopping:** We implemented an early stopping mechanism, terminating the training early if the validation performance does not improve for 5 consecutive epochs. This mechanism effectively prevents model overfitting and saves training time.

It is important to note that for different tasks, some hyperparameter settings may be adjusted. To obtain accurate experimental results, please refer to the detailed parameter configurations in the compiled dataset specific to each task.

D.2 Evaluation GFMs in Benchmarks

To evaluate existing GFMs on the four integrated benchmark suites (RGB, PGB, GUE, GB), we adapted each public checkpoint to the `OmniGenBench` interface and report their results in Section 3. Table 7 summarizes key statistics for all baselines.¹⁰

ViennaRNA secondary-structure annotations were *optionally* supplied for structure-aware variants of `OmniGenBench`, but are **not** required by baseline models.

We exclude models without publicly-visible code or checkpoints (e.g. Uni-RNA, 5UTR-LM) and briefly *summarize* each included FM below; see the original publications for full method details.

- ViennaRNA [61]. ViennaRNA is a comprehensive genomic analysis tool that includes a diverse set of interfaces, such as RNAFold¹¹ and RNAInverse¹² design. ViennaRNA serves as the baseline for RNA structure prediction and RNA design in our experiments.
- DNABERT2 [11]. DNABERT2 is one of the latest DNA FMs which improves the performance of DNABERT. The main modification of DNABERT2 is the tokenization method, which was changed to BPE from k-mers.
- HyenaDNA [19]. HyenaDNA is an autoregressive FM optimized for long-range genome data processing. HyenaDNA is based on the Hyena convolution architecture and capable of handling sequences up to 1M bases in length.
- Caduceus [31]. Caduceus¹³ is an advanced DNA language model built on the MambaDNA architecture, designed to address challenges in genomic sequence modeling, such as long-range token interactions and reverse complementarity (RC).

¹⁰SNT’ denotes *single-nucleotide tokenisation*, one character per base.

¹¹<https://www.tbi.univie.ac.at/RNA/RNAfold.1.html>

¹²<https://www.tbi.univie.ac.at/RNA/RNAinverse.1.html>

¹³https://huggingface.co/kuleshov-group/caduceus-ps_seqlen-131k_d_model-256_n_layer-16

- Nucleotide Transformer (NT) V2 [47]. The NT FMs were trained on DNA data, including the human reference genome and multi-species DNA sequences. They aim to capture the complex patterns within nucleotide sequences for various genome modeling applications.
- Agricultural Nucleotide Transformer (Agro-NT) [12]. Agro-NT is a large-scale DNA FM (1B parameters) akin to the Nucleotide Transformers but with a focus on plant DNA.
- SpliceBERT [30]. It was trained on 2M precursor messenger RNA (pre-mRNA) and specialised in RNA splicing of pre-mRNA sequences.
- 3UTRBERT [54]. This model was trained on 20k 3'UTRs for 3'UTR-mediated gene regulation tasks. It uses k-mers tokenization instead of SNT. RNA-BERT [50]. RNA-BERT is a BERT-style model pre-trained on a large corpus of non-coding RNA sequences. It uses masked language modeling (MLM) as its primary training objective. The model is designed to predict RNA structural alignments and can be fine-tuned for various RNA sequence classification and regression tasks
- RNA-MSM [33] RNA-MSM is an unsupervised RNA language model based on multiple sequence alignment (MSA). It is the first model of its kind to produce embeddings and attention maps that directly correlate with RNA secondary structure and solvent accessibility. RNA-MSM is particularly effective for tasks involving evolutionary relationships in RNA sequences.
- RNA-FM [24] RNA-FM is a BERT-based RNA foundation model trained on a vast dataset of non-coding RNA sequences. The model excels in predicting RNA structure and function by leveraging masked language modeling (MLM) during pre-training. RNA-FM's training data is sourced from the RNACentral database, providing it with extensive knowledge across diverse RNA species.
- **OmniGenBench.** OmniGenBench is the RNA genome FM that advocates the importance of sequence-structure alignment. Moreover, it is the first FM which addressed the *in-silico* RNA design task.
- **OmniGenome:** A FM dedicated to RNA genome modeling. This model leverages the computation-based structure to enhance the genome modeling ability and achieves impressive performance on both RNA and DNA genomes.

D.3 RNA Genomic Benchmark (RGB)

The RGB comprises seven challenging single-nucleotide level RNA modeling tasks, designed to evaluate models' fine-grained capabilities in understanding RNA sequences, such as predicting RNA structures. These tasks include mRNA degradation rate prediction, single-nucleotide modification detection (SNMD), single-nucleotide modification regression (SNMR), and RNA secondary structure prediction tasks such as Archive2, Stralign, and bpRNA. Additionally, EternaV2 evaluates models on RNA design tasks.

Table 8 presents the performance of various GFMs on the RGB tasks. Overall, OmniGenome achieves the best performance across all tasks, highlighting its exceptional capability in RNA structure modeling. This superior performance can be attributed to OmniGenome's integration of structural information into its modeling process, which is particularly beneficial for tasks requiring secondary structure prediction.

Table 8: The performance of OmniGenBench and baseline models on the RGB, with results averaged based on five random seeds. “N.A.” means not available for predictive tasks.

Model	mRNA	SNMD	SNMR	Archive2	Stralign	bpRNA	EternaV2
	RMSE	AUC	F1	F1	F1	F1	Accuracy
ViennaRNA	N.A.	N.A.	N.A.	73.99	74.09	65.03	33
MXFold2	N.A.	N.A.	N.A.	90.09	97.01	64.99	N.A.
Ufold	N.A.	N.A.	N.A.	89.78	95.76	78.38	N.A.
DNABERT2	0.8158	49.94	15.86	55.73	64.09	33.77	0
HyenaDNA	0.8056	53.32	39.80	71.18	91.24	57.43	0
Caduceus	0.8026	57.01	39.59	74.37	92.28	59.76	0
NT-V2	0.7826	50.49	26.01	68.36	83.18	56.95	0
Agro-NT	0.7830	49.99	26.38	62.81	72.54	46.87	0
SpliceBERT	0.7340	58.11	46.44	79.89	93.81	71.59	3
3UTRBERT	0.7772	50.02	24.01	68.62	88.55	57.90	0
RNABERT	0.8087	51.32	29.14	24.66	83.68	47.96	0
RNA-MSM	0.7321	57.86	45.22	68.72	91.15	64.44	2
RNA-FM	0.7297	59.02	42.21	82.55	95.07	78.16	4
OmniGenome	0.7121	64.13	52.44	91.89	98.21	83.18	84

In particular, OmniGenome significantly outperforms other models on the mRNA degradation rate prediction task, achieving an RMSE of 0.7121, compared to the second-best RMSE of 0.7297 by RNA-FM. Similarly, for the SNMD task, OmniGenome achieves an AUC of 64.13, surpassing the second-best score of 59.02 by RNA-FM. These results indicate that OmniGenome effectively captures single-nucleotide level variations, which are crucial in RNA function and regulation. Furthermore, in the secondary structure prediction tasks (Archive2, Stralign, bpRNA), OmniGenome demonstrates superior performance, highlighting its proficiency in modeling RNA secondary structures. This can be attributed to OmniGenome’s incorporation of structural context during pretraining, which enhances its ability to understand and predict RNA folding patterns. One limitation observed is that models not specifically designed for RNA tasks, such as DNABERT2 and HyenaDNA, perform poorly on RNA-specific tasks. This underscores the importance of tailoring GFM to the specific characteristics of RNA sequences.

In summary, the RGB results highlight the critical role of structural modeling in RNA genomics and demonstrate the effectiveness of OmniGenome in capturing complex RNA features. Future GFMs may benefit from incorporating structural information to enhance performance on RNA-related tasks.

D.4 Plant Genomic Benchmark (PGB)

The PGB comprises DNA-based tasks focused on plant biology. The sequences in PGB contain up to 6,000 bases, presenting challenges for models in handling long genomic sequences. Table 9 summarizes the performance of various GFMs on the PGB tasks. Resembling the results of RGB, OmniGenome achieves top-tier performance across most tasks, even though it was only trained on RNA. This suggests that OmniGenome generalizes well to DNA-based tasks, likely due to shared sequence motifs and structural similarities between RNA and DNA.

In the PolyA task, OmniGenome achieves an F1 score of 87.55, outperforming the second-best model, RNA-FM, which achieves 84.94. Similarly, for the LncRNA task, OmniGenome attains an F1 score of 77.96, significantly higher than the second-best score of 73.08 by NT-V2. OmniGenome excels in the Splice Site prediction task, achieving an F1 score of 98.41, surpassing the second-best score of 96.45 by SpliceBERT. This suggests that OmniGenome effectively captures sequence motifs important for splicing, which is crucial in gene expression regulation. These results indicate that GFMs incorporating structural context, like OmniGenome, can generalize effectively across different genomic modalities (RNA and DNA) and species (plants). The strong performance of OmniGenome on DNA-based tasks suggests that structural modeling enhances the understanding of genomic sequences beyond the specific type of nucleic acid. However, it’s also observed that some models specifically designed for DNA tasks, such as NT-V2 and SpliceBERT, perform competitively on certain tasks. This underscores the importance of task-specific pretraining and the potential benefits of integrating both sequence and structural information in GFMs.

In summary, the PGB results highlight the potential for cross-modal generalization in GFMs and the value of incorporating structural context to enhance performance on diverse genomic tasks.

Table 9: Performance of open-source GFM_s on PGB, where the results are re-implemented based on our evaluation protocol. ‘‘PolyA’’ stands for Polyadenylation, ‘‘Chrom Acc’’ for Chromatin Accessibility, ‘‘Prom Str’’ for Promoter Strength, ‘‘Term Str’’ for Terminator Strength, ‘‘Splice’’ for Splice Site, ‘‘Gene Exp’’ for Gene Expression, and ‘‘Enh Reg’’ for Enhancer Region.

Model	PolyA		LncRNA		Chrom Acc	Prom Str	Term Str	Splice	Gene Exp	Enhancer
	F1	F1	F1	RMSE	RMSE	F1	RMSE	F1		
DNABERT2	41.35	72.55	61.49	0.99	0.24	45.34	14.78	36.40		
HyenaDNA	83.11	58.21	52.20	0.88	0.26	90.28	14.79	66.17		
Caduceus	70.89	68.40	64.53	0.91	0.26	78.51	14.72	60.83		
NT-V2	71.26	73.08	65.71	0.81	0.27	95.05	14.79	73.89		
Agro-NT	78.89	67.24	63.27	0.94	0.78	88.45	15.56	62.83		
SpliceBERT	65.23	71.88	63.62	0.75	0.22	96.45	14.70	69.71		
3UTRBERT	76.48	70.75	63.71	1.04	0.36	94.44	14.87	71.67		
RNA-BERT	78.54	61.99	48.94	1.81	0.38	94.45	14.89	57.61		
RNA-MSM	84.25	67.49	53.52	1.28	0.28	95.49	14.87	61.45		
RNA-FM	84.94	68.75	54.92	0.95	0.27	95.95	14.83	57.14		
OmniGenome	87.55	77.96	67.69	0.59	0.18	98.41	14.71	79.77		

D.5 Genomic Understanding Evaluation (GUE)

The GUE is a multi-species benchmark like RGB and PGB, but focuses on the non-plant genomes. The sequences in GUE range in length and complexity, providing a robust assessment of GFM_s’ abilities to generalize across species and genomic tasks. Table 10 presents the performance of various GFM_s on the GUE tasks. While OmniGenome does not achieve the highest performance on all tasks, it consistently delivers competitive results, demonstrating strong cross-species generalization despite being primarily trained on RNA data.

Table 10: Performance of open-source GFM_s on GUE, where the results are re-implemented based on our evaluation protocol. The performance for each task is the average macro F1 score in all sub-datasets.

Model	Yeast EMP	Mouse TF-M	Virus CVC	Human TF-H	Human PD	Human CPD	Human SSP
	F1						
DNABERT-2	75.85	86.23	58.23	81.80	90.17	82.57	85.21
HyenaDNA	73.08	73.44	27.59	77.62	91.19	84.31	83.34
Caduceus	73.49	78.18	27.49	79.56	89.13	85.09	81.82
NT-V2	74.93	78.10	32.71	79.12	90.87	84.70	84.13
SpliceBERT	77.66	84.97	47.17	82.77	92.24	83.96	93.81
3UTRBERT	71.89	71.46	34.84	74.85	82.37	90.51	81.95
RNA-BERT	60.14	59.83	21.08	67.48	79.87	76.25	44.75
RNA-MSM	64.99	79.15	51.81	78.72	91.28	85.42	84.24
RNA-FM	74.41	78.24	52.22	79.27	92.18	86.05	84.76
OmniGenome	78.51	84.72	64.41	81.73	90.04	85.22	90.39

In the Yeast EMP task, OmniGenome achieves the highest F1 score of 78.51, slightly outperforming SpliceBERT of 77.66. For the Virus CVC task, OmniGenome also achieves the best performance with an F1 score of 74.72, indicating its strong ability to model viral genomic sequences. However, for tasks like Human TF-H and Human SSP, models like SpliceBERT and DNABERT2 achieve higher scores. This suggests that these models may be better optimized for human genomic sequences or specific tasks like splice site prediction. The results on GUE highlight the challenges in developing GFM_s that generalize across different species and genomic tasks. While OmniGenome demonstrates strong cross-species performance, there is variability depending on the specific task and species. These findings suggest that combining the strengths of different GFM_s or developing ensemble methods could be a fruitful direction for future research. Additionally, incorporating more diverse training data and task-specific fine-tuning may enhance the performance of GFM_s across a broader range of tasks.

D.6 Genomic Benchmarks (GB)

GB is a collection of DNA genome datasets aimed at evaluating the performance of models on sequence classification tasks involving regulatory elements such as promoters, enhancers, and open chromatin regions across different species including humans, mice, and roundworms. Table 11 shows

the performance of various GFMs. The tasks are denoted by their species and regulatory elements, and the acronyms are explained in Appendix C.4.

Table 11: Performance of open-source GFMs on GB, where the results are re-implemented based on our evaluation protocol.. The performance (macro F1) for each task is the average macro F1 score across all sub-datasets.

Model	DEM	DOW	DRE	DME	HCE	HEE	HRE	HNP	HOR
	F1								
DNABERT-2	92.67	95.17	43.77	77.21	75.58	80.66	78.14	85.80	68.03
HyenaDNA	88.21	94.13	70.11	76.44	70.38	79.58	96.33	85.99	67.03
Caduceus	92.13	94.74	72.03	75.61	70.20	76.47	79.16	84.36	63.17
NT-V2	91.66	94.32	78.20	81.72	71.98	79.85	93.30	85.30	68.53
SpliceBERT	94.72	96.42	72.29	74.70	73.50	79.60	95.23	89.57	68.89
3UTRBERT	89.50	90.22	74.35	80.14	70.23	76.33	98.47	82.49	66.78
RNA-BERT	76.56	62.17	50.11	60.79	66.69	63.29	46.57	73.80	56.59
RNA-MSM	79.38	93.71	54.13	75.90	69.79	78.07	94.87	84.28	63.93
RNA-FM	91.53	95.49	74.77	79.74	71.62	80.03	95.72	87.14	69.38
OmniGenome	94.16	93.49	77.17	80.34	73.51	82.23	95.66	87.87	68.97

In the DEM and DOW tasks, SpliceBERT achieves the highest F1 scores, with OmniGenome closely following in DEM and RNA-FM in DOW. For the DRE task, NT-V2 achieves the best performance with an F1 score of 78.20, with OmniGenome performing closely. In the HEE task, OmniGenome attains the highest F1 score of 82.23, surpassing the second-best score of 80.66 by DNABERT-2. This indicates OmniGenome’s effectiveness in modeling human enhancer regions. From a global perspective, these results demonstrate that while different GFMs excel in specific tasks, OmniGenome consistently performs well across various genomic benchmarks, highlighting its versatility. The performance variations across models suggest that task-specific features and training data significantly impact model efficacy. A limitation observed is that GFMs primarily trained on RNA data, like RNA-BERT and RNA-MSM, lost on DNA-based tasks. This underscores the importance of training data relevance and the potential need for multimodal pretraining strategies.

In conclusion, the GB results emphasize the need for GFMs that can generalize across different genomic tasks and species. Integrating structural information, as done in OmniGenome, appears to enhance model performance on complex genomic tasks.

D.7 BEACON Results

We have completed data organization and the compilation of benchmark tasks. However, we are currently unable to provide results because the experimental outcomes cannot be reproduced. Our next steps involve verifying the custom implementation of the evaluation metrics and checking the integrity of the dataset. Additionally, there are still missing dataset in the BEACON benchmark. We have submitted an issue to the authors and are awaiting a response.

D.8 Overall Discussion

Our comprehensive evaluation across four genomic benchmarks reveals that OmniGenome consistently achieves top-tier performance, particularly excelling in tasks that involve structural modeling of RNA sequences. The integration of structural information in OmniGenome enhances its ability to capture complex sequence features, which is advantageous across diverse genomic tasks. While OmniGenome demonstrates strong performance even on DNA-based tasks, models specifically tailored to certain tasks or species, such as SpliceBERT and DNABERT2, sometimes outperform OmniGenome in those specific contexts. This suggests that task-specific or species-specific pretraining can provide benefits, and there is potential for combining the strengths of different models.

The absence of results for certain models on some benchmarks (e.g., RNA-BERT, RNA-MSM, and RNA-FM on GUE) highlights the challenges in benchmarking GFMs across diverse datasets. Differences in model architectures, pretraining data, and tokenization strategies can impact a model’s applicability to specific tasks. Future work should focus on developing unified evaluation protocols and improving the interoperability of GFMs. An important consideration is the need for detailed descriptions of the models evaluated, including their architectures, pretraining data, and key features. This information is crucial for understanding the factors contributing to their performance and for reproducing results.

Overall, our comprehensive benchmarking highlights the importance of integrating structural information into GFMs and suggests that models capable of capturing both sequence and structural features offer improved performance across a range of genomic tasks. This work provides valuable insights for the development of next-generation GFMs and underscores the need for continued efforts in benchmarking to drive advancements in genomic modeling.

E Integrated Genomic Foundation Models in OmniGenBench

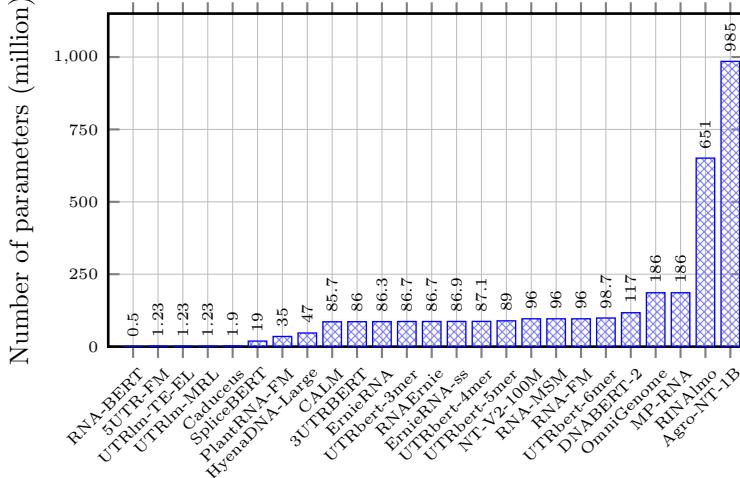


Figure 6: Overview of GFM parameter scales adapted in **OmniGenBench**. The models span from approximately 0.5 million to nearly 1 billion parameters, showcasing the scalability of the benchmarking framework. Our own model, OmniGenome, is included with 186 million parameters.

As illustrated in Figure 6, **OmniGenBench** has been utilized to benchmark a diverse array of GFMs (26 types with 31 models in total) that vary significantly in scale. The tested models range from compact architectures, such as RNA-BERT (0.5 million parameters) and Caduceus (1.9 million parameters), to substantially larger models like DNABERT-2 (117 million), our own OmniGenome (186 million), and Agro-NT-1B, which approaches 1 billion (985 million) parameters. This breadth demonstrates the capability of **OmniGenBench** to handle and rigorously evaluate GFMs across a wide spectrum of complexities and sizes, which is crucial for comprehensive and fair benchmarking.

The **OmniGenBench** platform is designed for continuous growth, with ongoing efforts to incorporate new and emerging GFMs. The following list details the models currently integrated, categorized broadly by scale and primary application domain. Each entry provides key characteristics and relevant citations.

- RNA-BERT. (0.5 M; MLM on 4,069 ncRNA families) A pioneering lightweight BERT encoder, RNA-BERT introduced nucleotide-level masking for effective ncRNA representation [50].
- 5UTR-FM. (1.23 M; 18M UTRs, auxiliary structure labels) This foundation model targets the regulatory grammar of **5'-UTRs**. It excels at predicting translation efficiency by leveraging extensive UTR pre-training and structural information [53].
- UTR-LM (TE-EL).(1.23 M) A variant of the UTR-LM series, specifically focused on co-predicting Translation Efficiency and mRNA Expression Level from 5'-UTR sequences [53].
- UTR-LM (MRL).(1.23 M) Another UTR-LM variant, this model is trained to predict Mean Ribosome Load, serving as a proxy for global protein output based on 5'-UTR features [53].
- Caduceus. (1.9 M; MambaDNA for human chromosomes) The first **MambaDNA** state-space model, Caduceus introduces reverse-complement (RC) equivariant gating, enabling efficient processing of long human chromosomal sequences [31].
- SpliceBERT. (19 M; 2M vertebrate pre-mRNA) Pre-trained on a vast dataset of vertebrate pre-mRNA, SpliceBERT achieves state-of-the-art splice-site detection using single-nucleotide tokenization [30].
- PlantRNA-FM. (35 M; *Arabidopsis* & 20+ crops, sequence & structure) An interpretable encoder designed for plant systems, PlantRNA-FM jointly leverages sequence and structural information to predict translation control mechanisms in *Arabidopsis* and over 20 other crop species [57].

- HyenaDNA-L. (47 M; Autoregressive, Hyena filters) This autoregressive DNA language model employs Hyena operators, allowing it to handle sequences up to 1 Mb with remarkable memory efficiency [19].
- CALM. (85.7 M; Codon-aware, ESM-based) The Codon-aware Language Model (CALM) adapts ESM layers to operate on nucleotide triplets (codons), thereby improving the analysis of protein-coding regions [62].
- 3UTRBERT. (86 M; BERT for 3'-UTRs) An encoder focusing on 3'-UTRs, 3UTRBERT provides family-aware embeddings crucial for understanding post-transcriptional regulation tasks [54].
- UTRbert-3,4,5,6mer. (86–87 M) This suite comprises four 3UTRBERT checkpoints, each utilizing a different k-mer tokenization strategy (3-mer to 6-mer). They are valuable for investigating the impact of tokenization granularity on UTR analysis [54].
- ErnieRNA. (86.3 M; Structure-enhanced, base-pair priors) An RNA language model enhanced with structural information, ErnieRNA injects base-pair priors during masked language modeling (MLM) pre-training [63].
- RNAErnie. (86.7 M; Motif-aware, motif-level masking) This model employs motif-aware pre-training, performing random masking at the *motif* level to better capture *cis*-regulatory elements within RNA sequences [51].
- ErnieRNA-ss. (86.9 M) A fine-tuned variant of ErnieRNA, specializing in RNA secondary-structure (ss) annotation tasks.
- NT-V2. (96 M; 300B DNA tokens, 850 species, hybrid k-mer) The second-generation **Nucleotide Transformer**, NT-V2 was trained on an extensive dataset of 300 billion DNA tokens from 850 species, utilizing a hybrid k-mer tokenization approach [47].
- RNA-MSM. (96 M; MSA-based evolutionary context) The first RNA language model to leverage multiple-sequence alignments (MSAs), RNA-MSM couples evolutionary context with token predictions for enhanced understanding [33].
- RNA-FM. (96 M; 23M ncRNA sequences) A large-scale encoder pre-trained on 23 million non-coding RNA sequences, RNA-FM has demonstrated superior performance over traditional tools like RNAfold for structure prediction tasks [24].
- UTRbert-6mer. (98.7 M) The checkpoint within the UTRbert family employing the most granular 6-mer tokenization, particularly effective for discovering long motifs in UTRs.
- DNABERT-2. (117 M; BPE, 32B tokens, 136 species) An upgrade to the classic DNABERT, DNABERT-2 uses byte-pair encoding (BPE) and was pre-trained on 32 billion tokens from 136 species [11].
- OmniGenome. (186 M; Multi-modal plant RNA/DNA, 54B tokens) Our flagship multi-modal foundation model, OmniGenome is pre-trained on 54 billion plant RNA and DNA tokens. It demonstrates state-of-the-art (SoTA) coverage across multiple benchmark suites (see Sec. 3).
- MP-RNA. (186 M; Multi-Phyla, explicit structure pre-training) The Multi-Phyla RNA foundation model (MP-RNA) incorporates explicit structure information during pre-training, achieving over 40% performance gains on RGB tasks [64].
- RiNALMo. (651 M; 36M ncRNA sequences, Flash-Attention-2) Currently the *largest* publicly available RNA language model, RiNALMo (650 M parameters) was trained on 36 million ncRNA sequences and utilizes Flash-Attention-2 for efficiency [65].
- Agro-NT-1B. (985 M; Billion-scale DNA LM for 48 plant genomes) A billion-parameter scale DNA language model, Agro-NT-1B is specialized for 48 edible plant genomes and inherits the k-mer vocabulary from NT-V2 [12].

Ongoing Integration and Adaptability. The integration of new GFM_s into **OmniGenBench** is a streamlined process, facilitated by its flexible adapter interfaces. For *most* models, adaptation primarily involves configuring two main components: the tokenizer and the model’s forward prediction function. This typically requires only minor code modifications (often \approx 100 lines), enabling a rapid turnaround time of **2–3 days** for onboarding new GFM_s. Work is continuously in progress to incorporate the latest advancements in the field; preliminary wrappers for several new models have already passed our unit-test suite and are awaiting large-scale benchmark evaluations. This ensures **OmniGenBench** remains a current and comprehensive platform for GFM assessment.

F Public Leaderboard

To promote transparency and reproducibility in genomic foundation modeling, we have released a dynamic, publicly accessible leaderboard alongside this manuscript. The current interface, shown in Figure 7, is built on our OmniGenBench platform and provides the following key features:

- **Multi-Suite Coverage:** Users can switch among the four major benchmark suites (RGB, PGB, GUE, GB) via the top-level tabs, allowing direct comparison of model performance on distinct task collections.
- **Interactive Filtering:** A flexible search bar accepts model names or keywords, while sidebar controls enable filtering by model type (pretrained vs. fine-tuned), numeric precision (e.g., bfloat16), and model scale (parameter count slider).
- **Customizable Metrics Display:** Checkboxes let users choose which columns to display—rank, task-specific metrics (e.g., mRNA RMSE, SNMD AUC, SNMR F1, ArchiveII F1, bpRNA F1, RNAStralign F1), as well as model metadata such as architecture, hub availability, and license.
- **Real-Time Ranking Updates:** Underlying OmniGenBench pipelines automatically re-compute ranks whenever new submissions are processed, ensuring that the leaderboard reflects the latest community contributions.
- **Submission Portal:** A “Submit here!” button links to a standardized result-upload API, where contributors package their evaluation outputs along with a Docker image or YAML environment spec. Automated sanity checks validate input format and metric integrity before the new entry is incorporated.

We are actively extending the leaderboard to integrate the latest GFMs—particularly those published after our initial release—and to refine the user experience. Planned enhancements include:

- **Expanded Task Coverage:** Addition of emerging genomic benchmarks (e.g., Plant Genomic Benchmark, epigenetic marking tasks) and support for multi-omics datasets.
- **Drill-Down Analytics:** Clickable cells will reveal per-task performance distributions, confidence intervals, and training details (e.g., data splits, seed settings).
- **Versioning and Provenance:** Each model entry will record a Git SHA and timestamp, enabling exact replication of results and rollback to prior evaluations.
- **Programmatic Access:** A RESTful API will allow users to query leaderboard data for downstream analysis, visualization, or integration into continuous-integration workflows.

By providing this living, community-driven resource, we aim to accelerate progress in genomic foundation models and foster an open benchmarking culture that underscores fairness, rigor, and collaborative innovation.

G Tutorials

This appendix provides practical examples demonstrating the usage of the OmniGenBench framework for various genomic tasks. These tutorials illustrate core functionalities and highlight the framework’s utility in addressing key challenges in computational genomics, moving beyond basic model application to enabling robust research workflows. The examples correspond to the Jupyter notebooks available in the project’s repository¹⁴.

G.1 Automated Benchmarking with AutoBench

Evaluating and comparing the performance of the rapidly growing number of Genomic Foundation Models (GFMs) is crucial for advancing the field, yet it presents significant challenges due to variations in datasets, tasks, and evaluation protocols. The AutoBench module within OmniGenBench addresses this by providing a standardized and automated pipeline for rigorous GFM assessment. This tutorial demonstrates its configuration and execution.

First, initialize the AutoBench class, specifying the root directory of the desired benchmark suite (e.g., RGB for RNA Genome Benchmark, GUE for general DNA tasks), the GFM identifier (local path or Hugging Face name), and the target compute device.

¹⁴<https://github.com/yangheng95/OmniGenBench/tree/master/examples>

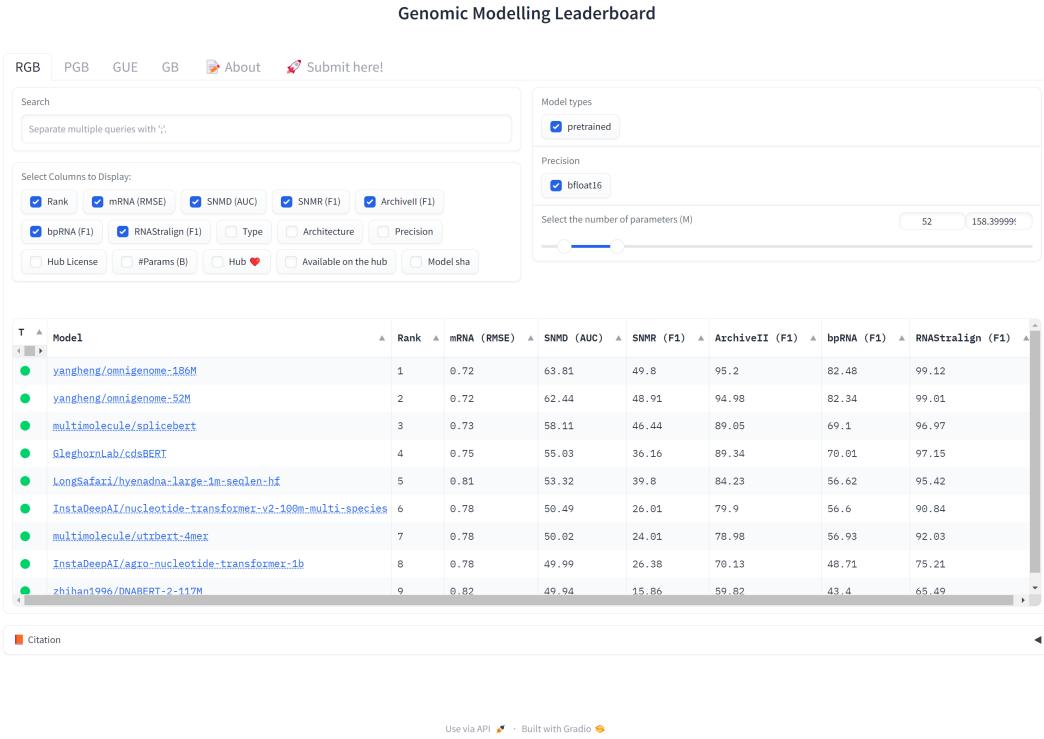


Figure 7: Screenshot of the interactive web interface for the public leaderboard, illustrating suite selection, filtering controls, and customizable metric display.

Benchmark Preparation

```

from omnigenome import AutoBench
import autocuda

# Define benchmark root and model path
root = 'RGB' # Example: RNA Genome Benchmark suite
model_name_or_path = 'anonymous8/OmniGenome-52M'
device = autocuda.auto_cuda() # Automatically select available GPU

# Initialize AutoBench
auto_bench = AutoBench(
    bench_root=root,
    model_name_or_path=model_name_or_path,
    device=device,
    overwrite=True,
)

```

Next, execute the benchmark run using the `run` method. While AutoBench uses predefined, community-accepted configurations for each task to ensure fair comparison, users can override specific hyperparameters (e.g., `epochs`, `batch_size`, `seeds`) for experimental purposes or resource adaptation.

Automated Benchmarking

```
# Define run-specific parameters (optional override)
batch_size = 8
epochs = 10
seeds = [42, 43, 44] # Multiple seeds for robustness assessment

# Run the benchmark across all tasks in the specified suite
auto_bench.run(epochs=epochs, batch_size=batch_size, seeds=seeds)
```

The results are systematically logged, providing detailed metrics for each task across different random seeds. This automated approach significantly reduces manual effort, enhances reproducibility, and facilitates transparent comparison of GFM capabilities, which is essential for identifying state-of-the-art models and guiding future development.

G.2 Fine-tuning for RNA Secondary Structure Prediction

RNA secondary structure (SSP), the pattern of base pairings within an RNA molecule, is fundamental to its function, stability, and interactions with other molecules. Predicting SSP accurately from the primary sequence remains a key challenge in computational biology. This tutorial demonstrates how to leverage the power of pre-trained GFMs by fine-tuning them for this specific token-level prediction task using `OmniGenBench`.

The process begins by importing necessary framework components and initializing a task-specific model (`OmniGenomeModelForTokenClassification`) built upon a chosen GFM backbone. A suitable tokenizer, potentially specialized for nucleotides (`OmniSingleNucleotideTokenizer`), is also required.

SSP Model Initialization

```
from omnigenome import (
    OmniGenomeDatasetForTokenClassification, ClassificationMetric,
    OmniTokenizer, OmniGenomeModelForTokenClassification,
    Trainer, ModelHub
)
import torch
import autocuda

# Define model path and task-specific label mapping (dot-bracket)
model_name_or_path = "anonymous8/OmniGenome-186M"
label2id = {": 0, ")": 1, ".": 2}

# Initialize tokenizer and model with classification head
tokenizer = OmniTokenizer.from_pretrained(model_name_or_path)
ssp_model = OmniGenomeModelForTokenClassification(
    model_name_or_path, tokenizer=tokenizer, label2id=label2id
)
device = autocuda.auto_cuda()
ssp_model.to(device)
```

Datasets containing sequences and their corresponding known structures (e.g., from bpRNA) are loaded using `OmniGenomeDatasetForTokenClassification`. This class handles tokenization and alignment of labels to tokens, preparing the data for training.

Training Preparation

```
# Define dataset paths and parameters
train_file = "Archive2/train.json"
# ... define test_file, valid_file ...
max_length = 512
batch_size = 8

# Load datasets (assuming JSON format: {"seq": "...", "label": "..."})
train_set = OmniGenomeDatasetForTokenClassification(
    data_source=train_file, tokenizer=tokenizer,
    label2id=label2id, max_length=max_length
)
# ... load valid_set, test_set ...
# Create DataLoaders
train_loader = torch.utils.data.DataLoader(
    train_set, batch_size=batch_size, shuffle=True
)
# ... create valid_loader, test_loader ...
```

Appropriate evaluation metrics (e.g., F1-score, Matthews correlation coefficient for structured prediction) are selected using `ClassificationMetric`. The `Trainer` class orchestrates the fine-tuning process, managing the training loop, optimization, evaluation, and checkpointing.

RNA Structure Prediction Fine-tuning

```
# Define metrics suitable for structure prediction
compute_metrics = [
    # Base accuracy
    ClassificationMetric().accuracy_score,
    # F1 is often key for SSP
    ClassificationMetric(average="macro").f1_score,
    # MCC useful for imbalanced classes
    ClassificationMetric().matthews_corrcoef,
]

# Setup optimizer and Trainer
learning_rate = 2e-5; weight_decay = 1e-5
optimizer = torch.optim.AdamW(
    ssp_model.parameters(),
    lr=learning_rate,
    weight_decay=weight_decay
)
epochs = 10; seed = 42
trainer = Trainer(
    model=ssp_model, train_loader=train_loader,
    eval_loader=valid_loader, test_loader=test_loader,
    batch_size=batch_size, epochs=epochs,
    optimizer=optimizer, compute_metrics=compute_metrics,
    seed=seed, device=device
)

# Run training and evaluation
metrics = trainer.train()
```

This fine-tuning approach allows the GFM to specialize its learned representations for the nuances of RNA folding, often achieving higher accuracy than traditional physics-based or purely statistical methods, especially for complex or novel structures. The trained model can be saved, shared via the ModelHub, and used for inference on new RNA sequences.

G.3 Zero-Shot RNA Secondary Structure Prediction

While fine-tuning yields high accuracy, GFMs pre-trained on vast sequence datasets (including structural information implicitly or explicitly) may possess inherent capabilities for tasks like SSP even without specific fine-tuning. This "zero-shot" capability is valuable for rapid analysis or when task-specific labeled data is limited. This tutorial demonstrates zero-shot SSP prediction using `OmniGenBench`.

Load a pre-trained model potentially suitable for this task (e.g., one trained with objectives related to structure or using relevant datasets). Crucially, the model class should provide a high-level inference method, such as `fold`, for direct structure prediction.

SSP Model Initialization

```
import torch
import autocuda
from transformers import AutoTokenizer
from omnigenome import OmniGenomeForTokenClassification

# Load a potentially zero-shot capable pre-trained model
model_path = "anonymous8/OmniGenome-186M" # Example model path
ssp_model = OmniGenomeForTokenClassification.from_pretrained(model_path)
ssp_model.to(autocuda.auto_cuda())
ssp_model.eval()
tokenizer = AutoTokenizer.from_pretrained(model_path)
```

Utilize the model's specialized inference method (`fold`) to directly predict the secondary structure from a sequence string. This abstracts away the tokenization, prediction, and decoding steps involved in typical token classification inference.

Zero-Shot RNA Secondary Structure Prediction

```
# Example RNA sequence
sequence = "GAAAAAAAAGGGGAGAAUCCCGCCCGAAAGGGGCCAAAGGGC"

# Predict structure directly using the model's high-level API
# This relies on the model class
# having implemented this specific functionality
predicted_structure_list = ssp_model.fold(sequence)
if predicted_structure_list:
    print("Zero-shot predicted structure:", predicted_structure_list[0])
```

Zero-shot prediction offers significant advantages in speed and data requirements compared to fine-tuning. While its accuracy might be lower than a fine-tuned model, it provides a powerful baseline and is useful for large-scale exploratory analysis. Comparing zero-shot results with established algorithms like ViennaRNA (as shown in the notebook) helps gauge the GFM's intrinsic structural understanding gained during pre-training.

G.4 Generating RNA Embeddings

GFMs learn rich, context-dependent representations of sequences. Extracting these representations as fixed-size vectors (embeddings) provides a powerful way to encode biological sequences for various downstream machine learning tasks, often capturing more nuanced information than traditional sequence similarity metrics. This tutorial demonstrates embedding generation using `OmniGenomeModelForEmbedding`.

Initialize the embedding model using a pre-trained GFM backbone. The choice of backbone influences the nature of the learned representations.

Embedding Model Initialization

```
from omnigenome import OmniGenomeModelForEmbedding

# Initialize using a pre-trained model. The model's architecture
# dictates the embedding properties.
model_name = "anonymous8/OmniGenome-186M" # Example GFM
embedding_model = OmniGenomeModelForEmbedding(model_name)
```

The model can encode single sequences or batches, producing vector representations (typically derived from the hidden states of the model, e.g., the CLS token or mean-pooled token states).

RNA Embedding

```
# Example RNA sequences
rna_sequences = ["AUGGCUACG", "CGGAUACGGC", "UGGCCAAGUC"]

# Encode a batch of sequences
# Output shape depends on pooling strategy
# (e.g., [batch_size, hidden_dim])
embeddings = embedding_model.batch_encode(rna_sequences)

# Encode a single sequence
embedding = embedding_model.encode_single_sequence("AUGGCUACG")
```

These embeddings encapsulate learned features and can be used directly for tasks like similarity search (calculating cosine similarity between embedding vectors often provides a measure of functional or structural similarity), sequence clustering, or as input features for simpler downstream classifiers or regressors. The ability to represent complex biological sequences as dense vectors is a cornerstone application of foundation models in genomics.

Embedding Similarity Calculation

```
# Save/load embeddings for later use
embedding_model.save_embeddings(rna_embeddings, "rna_embeddings.pt")
loaded_embeddings = embedding_model.load_embeddings("rna_embeddings.pt")

# Compute similarity based on embeddings
similarity = embedding_model.compute_similarity(
    loaded_embeddings[0],
    loaded_embeddings[1]
)
print(f"Embedding-based similarity: {similarity:.4f}")
```

G.5 Computational RNA Sequence Design

Designing RNA sequences (a.k.a., Inverse Design) that adopt a specific target secondary structure is an important challenge in synthetic biology and RNA therapeutics (e.g., mRNA vaccine optimization). This inverse folding problem involves searching the vast sequence space for candidates meeting structural constraints. This tutorial demonstrates how OmniGenBench can facilitate this using GFMs [23], potentially combined with optimization algorithms.

Initialize a specialized model class, `OmniGenomeModelForRNADesign`, which might incorporate structure prediction capabilities or fitness functions suitable for design.

RNA Inverse Design

```
from omnigenome import OmniGenomeModelForRNADesign

# Initialize the RNA design model
# This might wrap a GFM with structure prediction capabilities
# or a design-specific objective
model = OmniGenomeModelForRNADesign("anonymous8/OmniGenome-186M")
```

Define the target structure and execute the design process using `run_rna_design`. This method likely employs an optimization strategy (like the genetic algorithm mentioned in the notebook) guided by a fitness function that assesses how well a candidate sequence folds into the target structure (perhaps evaluated internally using the GFM or an external tool like ViennaRNA). Parameters control the search process.

```
# Define target structure (dot-bracket notation)
target_structure = "(((....)))"

# Run RNA design using an optimization algorithm (e.g., Genetic Algorithm)
# Parameters like mutation_ratio, num_population,
# num_generation tune the search
best_sequences = model.run_rna_design(
    structure=target_structure,
    mutation_ratio=0.5,      # Rate of mutation in GA
    num_population=100,      # Size of the sequence population per generation
    num_generation=100       # Number of optimization iterations
)

# Output the best candidate sequences found
print("Designed RNA sequences:", best_sequences)
```

This computational approach allows exploring sequence possibilities guided by learned models of sequence-structure relationships, potentially identifying novel solutions for synthetic biology applications faster than experimental screening or simpler computational methods. Verifying the designs by folding the output sequences with standard tools is a common validation step.

G.6 Sequence Augmentation via Masked Language Modeling

Training robust deep learning models often requires large and diverse datasets. When real data is limited, augmentation techniques can generate synthetic variations to improve model generalization. This tutorial demonstrates sequence augmentation using a Masked Language Model (MLM) approach within `OmniGenBench`, leveraging a GFM's understanding of sequence patterns to create plausible variants.

Initialize `OmniGenomeModelForAugmentation` with a pre-trained MLM (like a BERT-style GFM). Key parameters are the ‘noise_ratio’ (fraction of sequence tokens to mask) and ‘instance_num’ (how many augmented versions to generate per input).

Augmentation Model Initialization

```
from omnigenome import OmniGenomeModelForAugmentation

# Initialize the augmentation model using a pre-trained MLM
model = OmniGenomeModelForAugmentation(
    model_name_or_path="anonymous8/OmniGenome-186M",
    noise_ratio=0.2,      # Mask 20%
    max_length=1026,     # Max sequence length
    instance_num=3        # Generate 3 variants per original sequence
)
```

Apply augmentation to single sequences or process entire datasets from files. The model masks tokens randomly based on the ‘noise_ratio’ and uses the MLM to predict replacements, generating new sequence instances.

RNA Sequence Augmentation based on MLM

```
\begin{verbatim}
# Augment a single DNA/RNA sequence
sequence = "ATCTTGCATTGAAG" # Example sequence
augmented_sequences = model.augment_sequence(sequence)
# Output will be a list if instance_num > 1
print(f"Augmented variants: {augmented_sequences}")

# Augment all sequences in a JSON file and save to a new file
input_file = "test.json" # Assumes {"seq": "..."} per line
output_file = "augmented_sequences.json"
model.augment_from_file(input_file, output_file)
print(f"Augmented dataset saved to {output_file}")

```

This MLM-based augmentation is considered more sophisticated than simple random mutations, as the GFM fills masked positions based on learned sequence context, potentially generating more biologically plausible or challenging variations for model training, thus enhancing robustness against sequence variability or noise.

H Interpretability Cases for Genomic Foundation Models

To assess whether GFMs learn biologically relevant representations and internal mechanisms, we conduct three interpretability case studies: the sequence motif preservation analysis in Section 4.1, feature-embedding analysis for antimicrobial resistance prediction Appendix H.1, and attention inspection for RNA secondary structure Appendix H.2.

H.1 Feature-Embedding Analysis

Motivation. For tasks such as antimicrobial resistance (AMR) prediction, a GFM should ideally map resistant (ARG) and non-resistant (non-ARG) sequences to distinct regions in its representation space. Visualizing this latent geometry allows us to gauge whether pre-training already encodes functional cues related to AMR and how much additional separation task-specific fine-tuning provides.

Experimental Design. We collated DNA sequence data from key ARG databases (CARD, MEGARes, ResFinder), ensuring sequences were labeled as ARG or Non-ARG. For each of the three RNA language models, OmniGenome, RNA-MSM, and RNA-FM, we extracted sequence representations under three distinct model states: random initialization, after pre-training, and after task-specific fine-tuning on AMR prediction. We use RNA models for this analysis because we aim to study the ability of GFMs to learn cross-modal genomic representations (i.e., utilizing RNA GFM to decipher DNA modelling). Specifically, the last hidden state representation for each sequence was obtained and mean-pooled across the sequence dimension to generate a fixed-length high-dimensional embedding vector.

To evaluate the models’ ability to discriminate between ARG and Non-ARG classes, these high-dimensional embeddings were subjected to density-based clustering using DBSCAN. The quality of the resulting clusters was quantitatively assessed using the Silhouette Coefficient (SC, higher is better, indicating better separation and cohesion) and the Davies-Bouldin Index (DB, lower is better, indicating more distinct and compact clusters). For visualization of the high-dimensional embedding space, we first applied Principal Component Analysis (PCA) for initial dimensionality reduction, followed by t-SNE (perplexity = 30, 1,000 iterations) to project the data into a 2-D space, allowing for intuitive comparison of the representational capabilities of the different models and training stages. It is important to note that the SC and DB scores are calculated on the original high-dimensional embeddings before t-SNE projection, as t-SNE can distort global structure.

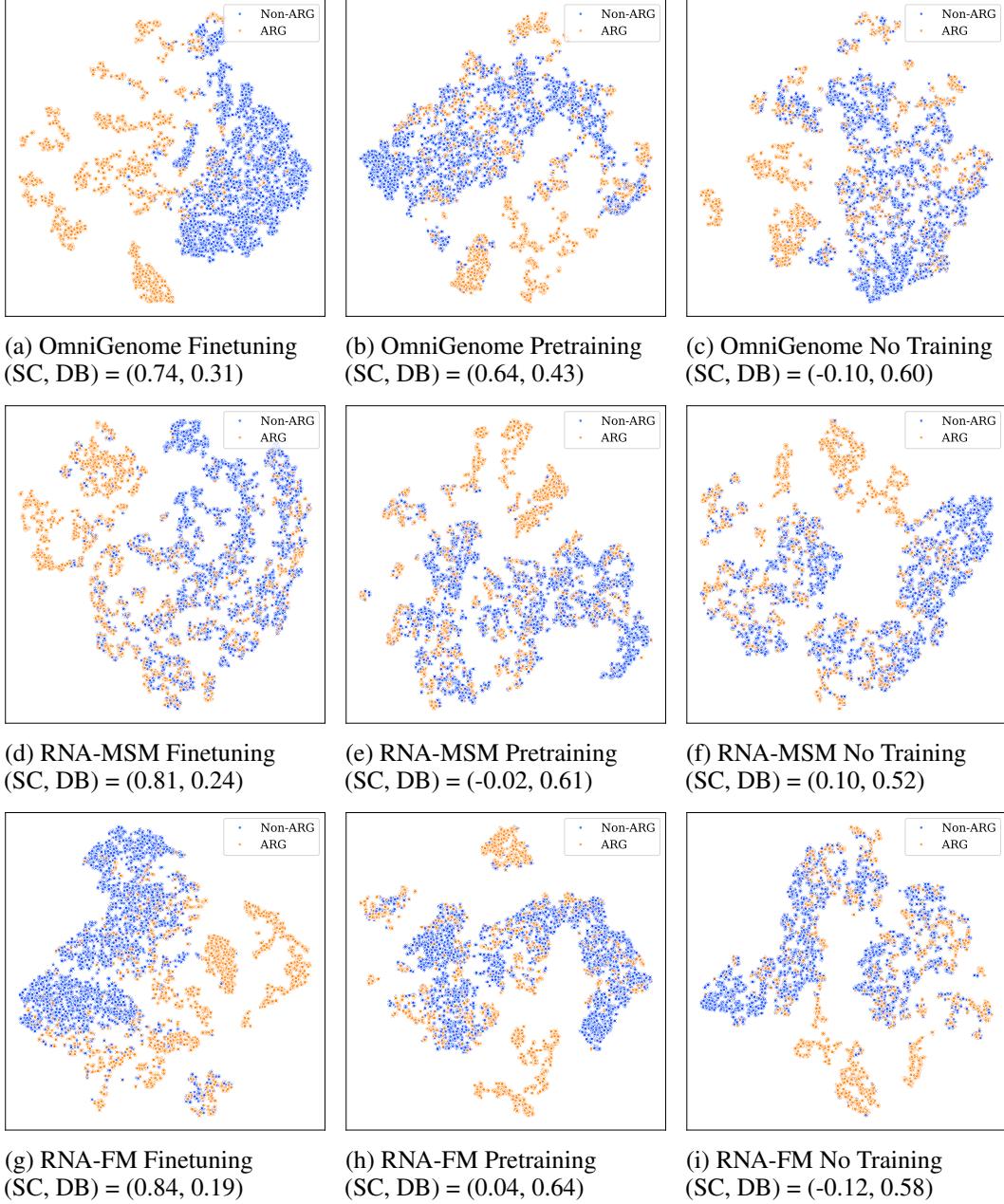


Figure 8: t-SNE visualization of GFM-derived sequence embeddings for antimicrobial resistance gene (ARG) classification. Each point represents a DNA fragment embedded by the corresponding model and training stage, colored by its class: orange for ARG and blue for Non-ARG. Models shown are OmniGenome (top row, a-c), RNA-MSM (middle row, d-f), and RNA-FM (bottom row, g-i), each under three training regimes: task fine-tuning, pre-training only, and random initialization (no training). Quantitative clustering scores (Silhouette coefficient, Davies-Bouldin index) are displayed below each subplot. Clearer clustering and separation are observed in finetuned and pretrained models, with OmniGenome exhibiting the most discriminative embeddings.

Results. Figure 8 displays the t-SNE visualizations of sequence embeddings for the three models across the three training stages. The Silhouette Coefficient (SC) and Davies-Bouldin Index (DB) for each condition are reported directly on the subplots.

- **OmniGenome (Finetuned)** (Figure 8a) demonstrates excellent class separation, forming distinct and compact clusters for ARG (orange) and Non-ARG (blue) sequences. This is quantitatively supported by a high SC of 0.74 and a low DB index of 0.31, indicating that fine-tuning has enabled the model to effectively learn AMR-specific discriminative features.
- **OmniGenome (Pretrained)** (Figure 8b) already exhibits substantial separation between the ARG and Non-ARG classes (SC = 0.64, DB = 0.43). This strong performance post-pre-training suggests that OmniGenome’s general pre-training regimen on diverse RNA/DNA sequences successfully captures latent signals relevant to antimicrobial resistance, providing a significant head-start for downstream tasks.
- **OmniGenome (Random)** (Figure 8c), as anticipated, shows no meaningful separation, with intermingled ARG and Non-ARG sequences and poor clustering metrics (SC = -0.10, DB = 0.60).
- **RNA-MSM (Finetuned)** (Figure 8d) yields well-separated clusters after fine-tuning, with strong quantitative scores (SC = 0.81, DB = 0.24), comparable to RNA-FM and outperforming OmniGenome on these metrics.
- **RNA-MSM (Pretrained)** (Figure 8e) demonstrates minimal class separation in its pretrained state (SC = -0.02, DB = 0.61), suggesting its pre-training does not inherently capture AMR-relevant features as effectively as OmniGenome.
- **RNA-MSM (Random)** (Figure 8f) shows some slight tendency for ARG sequences to cluster but overall poor metrics (SC = 0.10, DB = 0.52).
- **RNA-FM (Finetuned)** (Figure 8g) surprisingly achieves the best quantitative clustering among all finetuned models (SC = 0.84, DB = 0.19), forming very distinct clusters. This suggests that while its pre-training might lack strong AMR-related signals, its architecture is highly amenable to learning discriminative features when provided with task-specific supervision.
- **RNA-FM (Pretrained)** (Figure 8h) shows very little inherent structure related to AMR classification (SC = 0.04, DB = 0.64), indicating its pre-training does not effectively separate these classes.
- **RNA-FM (Random)** (Figure 8i) also shows poor separation (SC = -0.12, DB = 0.58).

Conclusion. The analysis of feature embeddings reveals distinct learning dynamics across the GFMs. Task-specific fine-tuning significantly improves the ability of all models to separate ARG and Non-ARG sequences, with RNA-MSM and RNA-FM achieving particularly high Silhouette Coefficients and low Davies-Bouldin Index scores post-finetuning, indicating very well-defined clusters. Notably, OmniGenome’s pre-training phase already instills a strong capability to distinguish between these classes, as evidenced by its superior SC and DB scores in the pretrained state compared to RNA-MSM and RNA-FM. This suggests that OmniGenome’s pre-training strategy is more effective at capturing inherent, functionally relevant genomic signals for AMR even before task-specific adaptation. While RNA-MSM and RNA-FM show excellent clustering after fine-tuning, their pretrained embeddings do not exhibit the same level of innate class separation as OmniGenome. These findings underscore the importance of both robust pre-training objectives for capturing generalizable biological signals and the capacity of models to adapt effectively during fine-tuning. The geometry of these learned representations serves as an insightful diagnostic for GFM capabilities.

H.2 Attention Representation Inspection

Motivation. RNA function is governed by its secondary structure, i.e. the set of canonical base pairs that form stems, loops and, occasionally, long-range (pseudoknot-like) contacts. If a GFM truly internalises these constraints, high self-attention weights should coincide with the known base pairs. Visualising attention maps therefore offers an intuitive lens on the *reasoning path* learned by a model.

Experimental design. We analyse two RNA sequences from the bpRNA-1m benchmark:

Example 1

```
Sequence: GCGCCAAGGGUGGCACGCCGGUCAGCGAGGUUUCGCCACCGCGUCUUUGUC
Structure: .....((((((.....))))....)).....
```

Example 2

```
Sequence: UGAAAGACACGGGUAGUGAGUAUGGAUUUUCAUCUCUAUUCGUGUCUUUC
Structure: .(((((((.(((.(((..(.....))))...)).))))))))
```

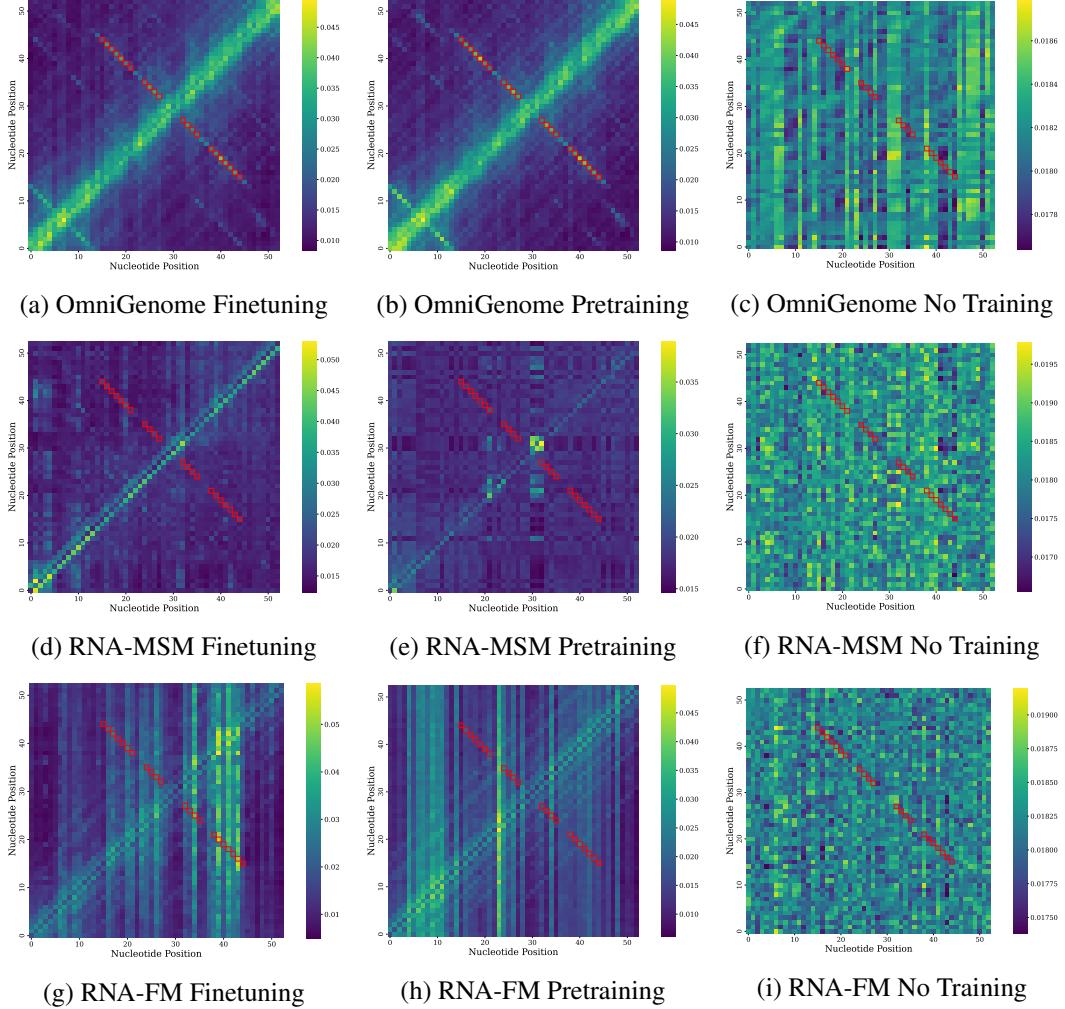


Figure 9: Attention maps (Example 1) for RNA sequences from the bpRNA-1m benchmark. Each heatmap shows the average self-attention weights in the last encoder layer. Red squares denote ground-truth base pairs from the reference RNA secondary structure. OmniGenome (top row) demonstrates strong alignment with biologically meaningful contacts after pretraining and finetuning, whereas RNA-MSM (middle row) shows weaker or scattered patterns. Randomly initialized models (rightmost column) exhibit noisy or structure-less attention.

For three GFMs, **OmniGenome**, **RNA-MSM**, and **RNA-FM**, we plot the average final-layer attention under three checkpoints: *random*, *pre-trained* (no task labels), and *fine-tuned* on an RNA task. Red squares mark ground-truth base pairs (contact distance $\leq 8 \text{ \AA}$). The resulting heat-maps are shown in Figure 9 and Figure 10.

Results.

- **OmniGenome.** Even *before* task supervision (panels b) the model attends strongly to both local stems and the long off-diagonal contacts in Example 2, indicating that its structure-aware pre-training already encodes canonical pairing rules. Fine-tuning (panels a) further sharpens these tracks, producing attention patterns that visually trace the reference contact map.
- **RNA-MSM.** The random checkpoint is essentially uniform noise (panels f). Pre-training (panels e) introduces sparse, patchy focus—some stem residues are highlighted, but many true pairs receive little weight. Fine-tuning (panels d) strengthens the main diagonal stems of Example 1, yet long-range contacts remain weak or absent, suggesting limited inductive bias for global structure.

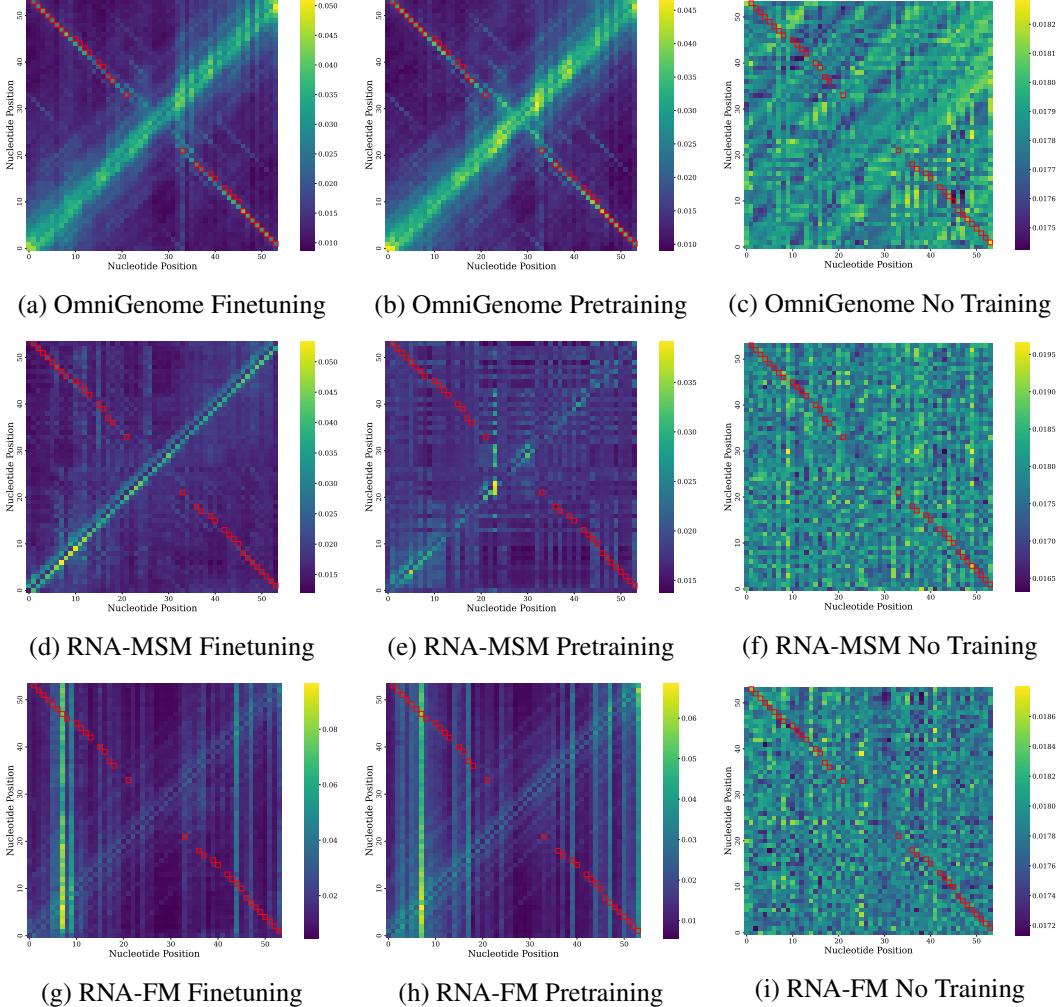


Figure 10: Attention maps (Example 2) for another RNA sequence from the bpRNA-1m dataset. As in Figure 9, each attention map is overlaid with ground-truth contacts (red squares). OmniGenome (top row) again attends to meaningful structural regions, even in pretraining alone, while RNA-MSM (middle row) requires fine-tuning to weakly capture stems. Both models with random initialization fail to highlight any coherent secondary structure.

- **RNA-FM.** Pre-training (panels h) yields clearer vertical/horizontal stripes than RNA-MSM, hinting at moderate structural awareness. After fine-tuning (panels g) the model emphasises most stem pairs and begins to highlight a subset of distal contacts, though the signal is less continuous than OmniGenome’s and occasionally spreads to unpaired regions.
- **Random baselines.** All three architectures with random parameters (panels c, f, i) display near-uniform, uninformative attention, confirming that the structure-aligned patterns above are learned rather than architectural artefacts.

Conclusion. Attention inspection corroborates our other interpretability findings. OmniGenome’s explicit, structure-aware pre-training guides the model to focus on native base pairs even without downstream labels, and fine-tuning refines this latent alignment into relatively precise contact patterns. RNA-FM benefits from pre-training but retains some off-target noise; RNA-MSM relies more heavily on labelled data and still struggles with long-range interactions. In practice, GFM’s whose pre-training embeds secondary-structure priors—such as OmniGenome, and to a lesser extent RNA-FM—offer more biologically faithful internal reasoning and are preferable for structure-sensitive RNA applications.

H.3 Development Environment

The benchmark experiments based on `OmniGenBench` were conducted on a dedicated Linux computation node, equipped with 2 NVIDIA RTX 4090 GPUs. For distributed model training, we employed version 4.44.0 of the Transformers library alongside version 0.28.3 of the Accelerate library. Our implementation framework of choice for `OmniGenBench` was PyTorch, specifically version 2.1.0. The ViennaRNA version is 2.6.4 in our experiments. While some existing code was adapted for the modules within `OmniGenBench`, the majority of the codebase, such as genomic sequences preprocessing, model pre-training, objective functions, and experiments, was meticulously crafted from scratch.

I Ethical Considerations

The advancement GFM, including powerful systems like AlphaFold and other large-scale models (e.g., Evo), necessitates a careful examination of their ethical implications. While GFM offer unprecedented capabilities for generating, predicting, and interpreting genomic sequences, thereby accelerating breakthroughs in genetic engineering, synthetic biology, and therapeutic development, they also introduce significant ethical challenges that require proactive governance.

We realize some key ethical dimensions in the work, including:

- **Biosecurity and Dual-Use Risks:** The capacity of GFM to design and manipulate genetic material at scale, while beneficial for research, also presents dual-use risks. There is a potential for misuse by malicious actors to create synthetic pathogens or other biological threats, thereby impacting global bio-safety. Addressing this requires robust safety protocols, stringent access controls for particularly powerful model capabilities, and mechanisms for auditing model usage and research outputs [66, 67].
- **Equity in Access and Benefit Sharing:** The development and deployment of GFM could exacerbate existing inequalities. While open-sourcing models like OmniGenome promote transparency and broader access, there is a risk that well-resourced entities (e.g., large pharmaceutical corporations) will disproportionately benefit, potentially monopolizing discoveries or creating treatments with prohibitive costs. This could widen global health disparities and limit access for researchers and patients in lower-resource settings. Intellectual property frameworks also need careful consideration to ensure fair benefit-sharing [68, 69].
- **Environmental Impact and Ecological Stability:** The enhanced ability to engineer organisms could lead to unintended ecological consequences, such as the disruption of natural ecosystems, loss of biodiversity, or the emergence of invasive or harmful synthetic species. Furthermore, the significant computational resources required for training and deploying large-scale GFM contribute to an increased carbon footprint, an environmental cost that must be carefully weighed against the scientific and societal benefits [70, 71].

J Societal Impact

The societal impact of GFM is substantial, with applications ranging from personalized medicine to environmental management.

Positive Societal Contributions

- **Advancements in Healthcare and Personalized Medicine:** GFM can significantly accelerate biomedical research by enabling more accurate prediction of disease risk, identification of novel biomarkers, and the design of targeted therapies. This can lead to breakthroughs in precision medicine, tailoring treatments to individual genetic profiles and improving outcomes for complex diseases, including cancer and rare genetic disorders [72, 73].
- **Innovations in Agriculture and Food Security:** In agriculture, GFM can contribute to the development of crops with enhanced yields, improved nutritional content, and greater resilience to pests, diseases, and climate change. This has the potential to bolster global food security and promote sustainable agricultural practices [74, 75].

- **Environmental Science and Conservation:** GFM can aid in understanding ecosystem dynamics, tracking biodiversity, and developing strategies for conservation and bioremediation by analyzing environmental DNA and RNA [70].

Challenges and Mitigation Strategies

- **Risk of Exacerbating Disparities:** A primary concern is that unequal access to GFM technologies and the insights they generate could widen socio-economic and global health divides. Entities with substantial computational resources and technical expertise may gain a significant advantage, potentially leading to an inequitable distribution of benefits [68, 69].
- **Ecological Considerations:** While offering benefits for agriculture and environmental science, the application of GFM-driven genetic modifications requires cautious oversight to prevent unintended negative impacts on ecological balance and biodiversity [70, 71].

K Limitations

The GFM benchmarking may not reflect the accurate performance in biology reality, we attribute the limitations of benchmarking to two major aspects:

- **Lack of *in-vivo* Data:** One of the critical limitations of GFMs lies in the absence of *in-vivo* verified genome data. While GFMs perform well in *in-silico* environments, where computational models and simulations are used to predict biological processes, these models are rarely validated against *in-vivo* data, which refers to experimental data obtained from living organisms. This presents a significant challenge for accurately translating model predictions to real-world biological applications. To be more specific, the complexity of biological systems, including interactions within cells, tissues, and organisms, often introduces variables that are not fully captured in computational simulations. For example, gene regulation, environmental factors, and cellular responses to genetic modifications may behave differently in living organisms than predicted by models trained on *in-silico* data. As a result, GFMs might not fully capture the biological complexity, leading to discrepancies between predicted and actual outcomes.
- **Model Scale Constraints:** The second major limitation is the model scales in benchmarking. As GFMs become larger and more sophisticated, their performance improves, but this scaling comes at a significant cost. Training as well as benchmarking large-scale GFMs requires immense computational resources, including high-performance GPUs or TPUs, massive memory allocation, and extensive storage for datasets. The cost of acquiring and maintaining this infrastructure can be prohibitive for many research institutions or companies, limiting access to cutting-edge GFMs.

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [\[Yes\]](#)

Justification:

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [\[Yes\]](#)

Justification:

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [\[NA\]](#)

Justification:

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification:

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification:

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification:

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: The evaluation scale is tremendous for repreating in GFMs

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [\[Yes\]](#)

Justification:

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [\[Yes\]](#)

Justification:

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [\[Yes\]](#)

Justification:

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to

generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification:

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification:

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. New assets

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification:

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. Crowdsourcing and research with human subjects

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification:

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. Institutional review board (IRB) approvals or equivalent for research with human subjects

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification:

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. Declaration of LLM usage

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorosity, or originality of the research, declaration is not required.

Answer: [NA]

Justification:

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.