RECONCILING COARSE TOKENIZATION WITH SINGLE-NUCLEOTIDE RESOLUTION IN GENOMIC LANGUAGE MODELING

Anonymous authors
Paper under double-blind review

ABSTRACT

Genomic language models face a fundamental trade-off: coarse tokenization enables efficient long-context modeling, yet standard token-level objectives obscure single-nucleotide variation. We introduce Factorized Nucleotide Supervision (FNS), which derives position-specific nucleotide likelihoods from a distribution over k-mers through exact marginalization. FNS retains the computational efficiency of non-overlapping k-mers while providing dense nucleotide-level supervision and directly supporting variant scoring. We instantiate FNS in GENERator-v2, a family of autoregressive models for eukaryotic and prokaryotic genomes. Across training-free and supervised benchmarks, GENERator-v2 consistently improves over its predecessor, matches or exceeds substantially larger baselines, and offers up to $44\times$ faster inference. Beyond these model-level gains, our results challenge two common assumptions in GFM evaluation: supporting a longer context does not guarantee that a model benefits from the added sequence, and single-nucleotide tokenization is not required to learn nucleotide-scale biological structure.

1 INTRODUCTION

The rapid growth of high-throughput sequencing has produced an unprecedented volume of genomic data (O'Leary et al., 2015), motivating genomic foundation models (GFMs) that learn general-purpose representations directly from DNA sequence (Wang et al., 2025). Advances in self-supervised sequence modeling and Transformer architectures (Vaswani et al., 2017) have enabled a range of objectives for genome annotation, variant interpretation, and sequence design.

Early GFMs predominantly use masked language modeling (MLM) (Devlin et al., 2019); examples include DNABERT, the Nucleotide Transformer (NT), and GROVER (Ji et al., 2020; Dalla-torre et al., 2024; Sanabria et al., 2024). Although effective for sequence representation and conditional scoring, bidirectional masking does not directly support open-ended left-to-right generation. Autoregressive models instead factorize the joint sequence distribution through next-token prediction, as in HyenaDNA, GENERator, and Evo2 (Nguyen et al., 2023; Wu et al., 2025; Brixi et al., 2026). Diffusion-based genomic models provide another generative framework (Sarkar et al., 2024; Sahoo et al., 2024; Avdeyev et al., 2023).

Despite methodological differences, existing approaches face a shared challenge when scaling to genomic lengths: enabling effective conditioning over long-range functional dependencies while preserving single-nucleotide resolution and computational efficiency. Functional dependencies in genomes often span tens to hundreds of kilobases, encompassing gene structure, regulatory elements, and higher-order chromatin organization. Modeling such context at single-nucleotide resolution is computationally prohibitive for standard Transformer architectures, creating a fundamental tension between biological fidelity and scalability.

Prior work has addressed this tension primarily through architectural modifications. Convolutional or convolution-like components extend effective context length in models such as Enformer, HyenaDNA, Evo, Evo2, and AlphaGenome (Avsec et al., 2021; Nguyen et al., 2023; 2024; Brixi et al., 2026; Avsec et al., 2025), while state-space models replace explicit attention with recurrent updates that scale linearly with sequence length (Schiff et al., 2024; Ma et al., 2025). These designs support long nominal inputs, but empirical gains can saturate as context length increases (Wu et al., 2025). A maximum

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

supported input is therefore a capacity limit, not evidence that the model makes effective use of the full span, motivating complementary improvements to sequence representation and supervision.

An alternative strategy reduces the number of tokens processed by the model. Learnable tokenizers, including byte-pair encoding (Kudo & Richardson, 2018) and token-merging methods (Bolya et al., 2022), are effective with masked objectives (Zhou et al., 2023; 2025; Li et al., 2024; Qiao et al., 2024; Li et al., 2025b), but overlapping or hierarchical units can make autoregressive generation ambiguous and degrade generative performance (Wu et al., 2025). Non-overlapping k-mers avoid this ambiguity and define a simple autoregressive factorization, but standard k-mer objectives obscure single-nucleotide variation and are sensitive to tokenization phase (Li et al., 2025a).

We address this incompatibility while retaining a decoder-only Transformer and non-overlapping k-mer tokenization. Factorized Nucleotide Supervision (FNS) derives k position-specific nucleotide likelihoods from each k-mer distribution by exact marginalization, restoring explicit single-nucleotide supervision without changing the tokenizer, architecture, or output dimensionality. For eukaryotes, we pair FNS with Genome Compression Pretraining (GCP), an annotation-aware procedure that enriches long training windows for functional contexts; for compact prokaryotic genomes, we retain complete sequences. Appendix E details the construction and rationale.

Building on these components, we present GENERator-v2, a family of autoregressive genomic foundation models for eukaryotic and prokaryotic genomes. We evaluate it through training-free sequence recovery, generation efficiency, variant-effect scoring, representation analysis, and supervised fine-tuning. GENERator-v2 consistently improves over GENERator in fixed-checkpoint evaluations, matches or exceeds it under supervised fine-tuning, and matches substantially larger autoregressive baselines in key operating regimes while retaining coarse-token efficiency. Beyond these model-level gains, the experiments challenge two common assumptions in GFM evaluation. Across sequence recovery and variant scoring, longer supported windows do not consistently translate into better performance, so maximum context length alone does not establish effective context use. Meanwhile, the 6-mer models recover cleaner strand and codon-phase structure than a 1-mer baseline, showing that single-nucleotide tokenization is not required for organized nucleotide-scale representations. These findings motivate measuring long-context capability by the gains produced by added sequence and their compute cost, and representation quality by the biological structure actually learned, rather than inferring either from context length or token size alone.

2 Factorized Nucleotide Supervision (FNS)

Non-overlapping k-mer tokenization substantially improves the computational efficiency of long-context genomic language modeling by reducing sequence length by a factor of k (Wu et al., 2025). However, when combined with a standard next-token cross-entropy objective over a 4^k-sized vocabulary, this representation introduces a systematic limitation in the training signal.

Specifically, each prediction step is supervised by one categorical label. Standard cross-entropy therefore provides no label-side distinction between a k-mer that differs from the target at one position and one that differs at every position. For typical choices such as $k = 6$, this sparse target representation is poorly aligned with genomic variation, for which single-nucleotide substitutions are common and biologically meaningful. A nucleotide-aware objective can expose this structure without sacrificing the sequence-length reduction provided by k-mer tokenization.

Factorized Nucleotide Supervision (FNS) addresses these issues by redefining the training objective while preserving the computational structure of k-mer tokenization. Rather than treating each k-mer as an atomic class, FNS interprets it as a structured object composed of k nucleotides and derives nucleotide-level supervision from the same regular-token logits via probability marginalization. Importantly, FNS does not modify the model architecture, tokenizer, or output dimensionality; it operates entirely at the level of loss construction and gradient allocation. A schematic comparison with standard k-mer supervision is shown in Figure 1, with formal derivations provided in Appendix A.

2.1 Formulation

Let $\mathcal{V} = \mathcal{B}^k$ denote the regular k-mer vocabulary, where $\mathcal{B} = \{A, C, G, T\}$, and let $\mathcal{S}$ contain the special tokens. Given an input context, the model outputs a categorical distribution $p(w)$ over

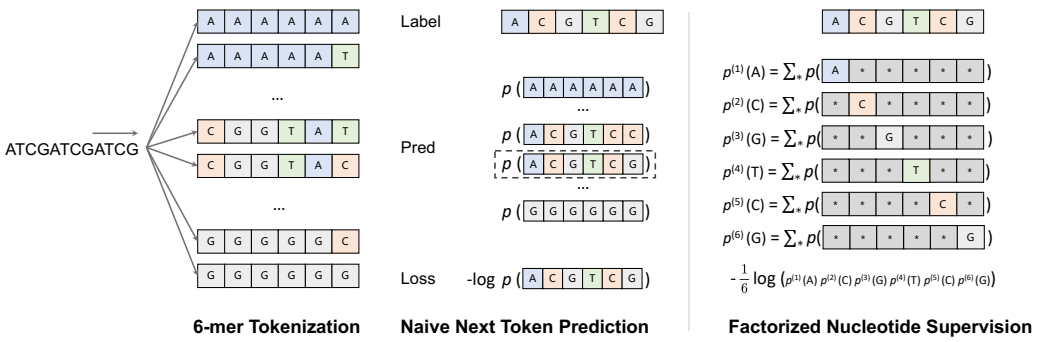

Figure 1: Factorized Nucleotide Supervision (FNS). Standard k-mer cross-entropy provides no target-side notion of nucleotide similarity among alternative tokens. FNS instead marginalizes token probabilities into position-specific nucleotide distributions and applies per-nucleotide supervision, yielding a denser signal structured by nucleotide agreement.

$\mathcal{W} = \mathcal{V} \cup \mathcal{S}$. For each nucleotide position $j \in \{1, \ldots, k\}$ and base $b \in \mathcal{B}$, FNS defines the induced marginal

$$p^{(j)}(b) = \sum_{v \in \mathcal{V}: v^{(j)} = b} p(v), \tag{1}$$

where $v^{(j)}$ denotes the j-th nucleotide of v. Given a ground-truth k-mer target $y = (y^{(1)}, \ldots, y^{(k)})$, the FNS loss is

$$\mathcal{L}_{\mathrm{FNS}} = -\sum_{j=1}^{k} \log p^{(j)}\left(y^{(j)}\right). \tag{2}$$

This objective remains fully differentiable with respect to the original output logits and rewards probability mass assigned to k-mers that agree with the target at one or more nucleotide positions. In implementation, we use the mean loss $\overline{\mathcal{L}}_{\mathrm{FNS}} = \mathcal{L}_{\mathrm{FNS}}/k$ so that its scale remains comparable across token sizes; this positive rescaling does not change the minimizers. The aggregate probability of $\mathcal{S}$ is treated as an auxiliary non-nucleotide outcome when constructing the marginals, while any unmasked special-token targets use ordinary token-level negative log-likelihood (Appendix A.3).

2.2 Optimization Signal and Inference Consistency

Under standard token-level cross-entropy, the target k-mer is the only positively supervised vocabulary item; alternatives receive no target-side structure based on whether they differ at one nucleotide or all k. FNS instead partitions the vocabulary into overlapping compatibility sets. A candidate k-mer contributes probability mass to the correct marginal at every position where its nucleotide matches the target, producing denser gradients tied to nucleotide-level agreement. The same 4^k logits are reused for all k marginals, requiring neither auxiliary prediction heads nor additional sequence positions. The additional computation consists only of reductions over already-computed vocabulary probabilities and does not change the attention pattern or sequence length.

More explicitly, let $\mathcal{A}_j = \{v \in \mathcal{V} : v^{(j)} = y^{(j)}\}$. For any output $u \in \mathcal{W}$, the contribution of position j to the logit gradient is

$$\frac{\partial[-\log p^{(j)}(y^{(j)})]}{\partial z_u} = p(u) - \mathbf{1}[u \in \mathcal{A}_j] \frac{p(u)}{p^{(j)}(y^{(j)})}. \tag{3}$$

A k-mer agreeing with the target at several positions receives one negative target-gradient term per matching position, whereas an output with no matching positions receives only positive softmax terms.

The induced marginals also align training with downstream nucleotide-level inference. Variant-effect prediction compares reference and alternative alleles under the same marginal distribution, avoiding a post-hoc conversion from an objective trained only on exact token identity. Tasks requiring joint k-mer probabilities can still use the original output distribution, which remains available under FNS.

This flexibility is particularly relevant to single-nucleotide variants, for which the altered position can be evaluated directly rather than only through an atomic token label.

FNS is the negative log-likelihood of a derived blockwise predictor whose k nucleotide outputs are conditionally independent given a shared hidden state (Appendix A.2). This equivalence concerns the product of the induced marginals; it does not assert that the model's original joint k-mer distribution factorizes. The marginals nevertheless provide directly usable nucleotide probabilities for applications such as likelihood-ratio scoring of single-nucleotide variants (Section 5.2). Conceptually, FNS is related to multi-token prediction in language models (Liu et al., 2024), but proceeds in the opposite direction: it decomposes supervision for one coarse token into multiple biologically meaningful components.

2.3 Trade-offs with Single-Nucleotide Tokenization

The product-of-marginals interpretation introduces an explicit approximation at the level of the training objective: dependence among nucleotide positions within the next k-mer is not directly rewarded. FNS retains the full k-mer output distribution, but its loss constrains only the position-wise marginals. This trade-off restores nucleotide-level supervision while preserving the computational benefits of coarse tokenization.

Beyond this approximation, k-mer tokenization changes the granularity of attention. A 6-mer bundles six adjacent nucleotides into one query/key/value unit, so attention operates over blocks rather than individual bases. This can limit access to within-block structure at a single attention step, especially for short inputs containing few tokens. For long genomic sequences, however, reducing sequence length by a factor of six allocates more of a fixed memory and compute budget to long-range context. Our sequence-recovery experiments examine this trade-off: the 6-mer models retain explicit nucleotide-level predictions while remaining efficient and benefiting from longer contexts (Section 5.1).

3 Annotation-Aware Data Construction

Eukaryotic genomes contain extensive sequence without curated functional annotations, so uniform sampling can make weakly informative background dominate the training distribution. Genome Compression Pretraining (GCP) instead joins gene-centric and regulatory regions in genomic order with separators, leaving the objective and dense Transformer unchanged. This policy has already been empirically validated: GENERator's functional-context filtering outperforms an all-sequence control (Wu et al., 2025), and Evo2's gene-centric corpus outperforms a whole-genome control on zero-shot variant tasks (Brixi et al., 2026).

Functional-context filtering does not imply that the regions it excludes contain no useful information; it improves the training distribution's signal-to-noise ratio. Retaining every base may recover some distal or unannotated elements, but adds a much larger volume of weakly constrained background that dilutes functional patterns. A functionally enriched corpus can thus outperform an all-sequence corpus. GCP retains protein-coding genes, RNA genes, regulatory elements, and nearby regions with potential regulatory roles, reducing the RefSeq eukaryotic corpus approximately fivefold. This trade-off suits sparse eukaryotes. For compact, gene-dense prokaryotes, modest compression does not justify losing native context, so we retain complete sequences. Appendix E presents the rationale, construction, and limitations.

4 GENERator-v2

We introduce GENERator-v2, a family of autoregressive genomic foundation models trained with FNS and domain-specialized corpora. All variants use a dense Transformer decoder with non-overlapping 6-mer tokenization and a maximum supported context length of 98,304 bp. Rather than adding a specialized long-context backbone, GENERator-v2 changes how the coarse-token output is supervised: FNS converts each next-token distribution into exact position-specific nucleotide marginals. This controlled design tests whether structured supervision can recover nucleotide-level fidelity while preserving the sequence-length reduction of coarse tokenization.

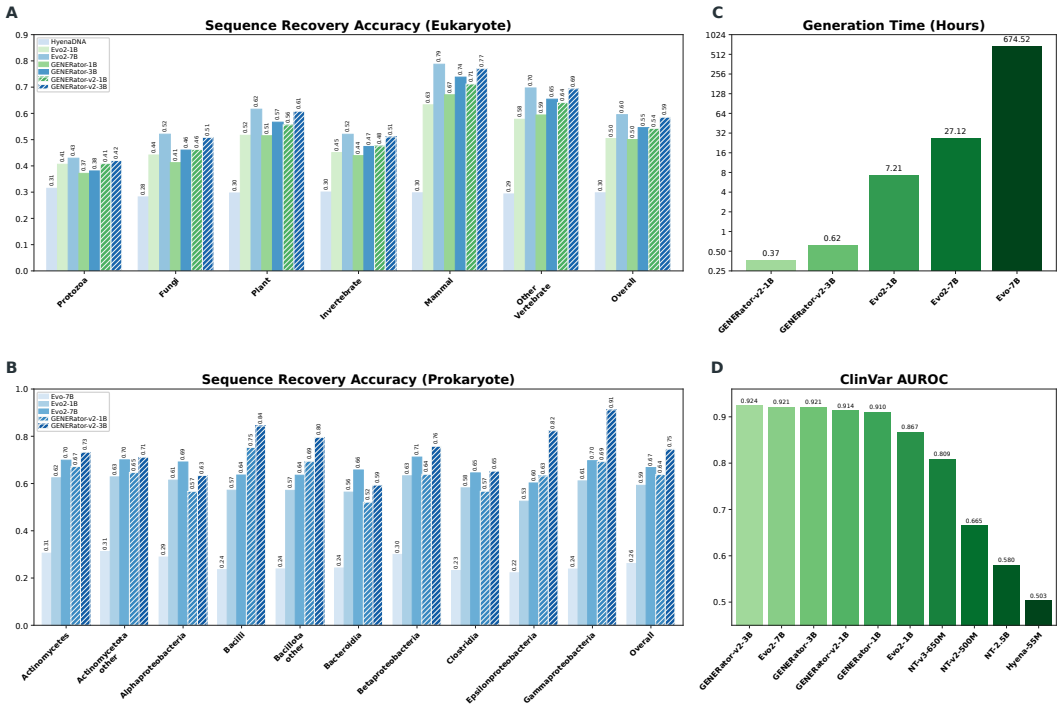

Figure 2: Training-free evaluations. (A–B) Sequence-recovery accuracy for eukaryotic and prokaryotic genomes, respectively, using 6,144-bp inputs and 30-bp continuation targets. (C) End-to-end generation time on 30,000 sequences with 6,144-bp inputs and 30-bp continuations. (D) Variant-effect prediction on ClinVar measured by AUROC.

GENERator-v2 includes two domain-specialized variants: a eukaryotic model and a prokaryotic model. For each domain, we train two scales (1.2B and 3B parameters, referred to as 1B and 3B), yielding four models in total.

FNS is applied uniformly across all four models without architectural changes, enabling the same nucleotide-aware objective to be evaluated across biological domains and parameter scales.

For eukaryotic variants, FNS is paired with Genome Compression Pretraining (GCP; Section 3), and training uses RefSeq genomes (O'Leary et al., 2015), whose curated annotations support GCP preprocessing. Prokaryotic variants retain complete sequences to preserve operonic and intergenic context and are trained on a combined RefSeq and Genome Taxonomy Database (GTDB) corpus (Parks et al., 2022). Appendix E.2 gives the domain-specific rationale, and Appendix F summarizes all model configurations (Table S1).

5 Experiments

We evaluate four aspects of GENERator-v2: training-free sequence recovery and generation efficiency, single-nucleotide variant-effect scoring, the structure of learned representations, and supervised fine-tuning on genomic classification tasks.

5.1 Sequence Recovery

We use a *sequence recovery* task (Wu et al., 2025) as a unified, training-free protocol for assessing generative accuracy and efficiency under matched input–output conditions across tokenizers and architectures. Because the task requires open-ended continuation, it directly compares only autoregressive models.

Evaluation protocol. Genomic sequences are grouped by taxonomic category. For each group, we sample 6,144-bp contexts from RefSeq and ask each model to generate the subsequent 30 bp. Inputs are constrained to end within a coding sequence while containing predominantly noncoding upstream context, providing a defined prediction target together with long-range noncoding information.

Results. Across all eukaryotic groups, GENERator-v2 consistently improves over the corresponding GENERator model (Figure 2A). GENERator-v2-1B also exceeds Evo2-1B at the same parameter scale, while GENERator-v2-3B matches Evo2-7B recovery with fewer than half the parameters and approximately $44\times$ faster generation (Figure 2A,C).

On prokaryotic datasets, GENERator-v2 likewise achieves higher aggregate recovery than Evo2 at both reported operating points: 0.64 versus 0.59 for the 1B models, and 0.75 for GENERator-v2-3B versus 0.67 for Evo2-7B (Figure 2B). Although the margin varies across bacterial clades, both GENERator-v2 models substantially outperform Evo, which approaches random recovery under this continuation protocol.

Generation efficiency. We separately measure generation time on 30,000 sequences with 6,144-bp inputs and 30-bp continuations. GENERator-v2-1B and GENERator-v2-3B complete the benchmark in 0.37 and 0.62 hours, respectively, compared with 7.21 hours for Evo2-1B and 27.12 hours for Evo2-7B—approximately $19\times$ and $44\times$ speedups at the corresponding scales. Evo-7B requires 674.52 hours and has the lowest recovery accuracy (Figure 2C).

Context length effects. The 6,144-bp input is shorter than the maximum context supported by both model families and controls evaluation cost. Results across input lengths are reported in Appendix B (Figure S1). Evo2's recovery largely plateaus beyond approximately 1,024 bp, whereas GENERator and GENERator-v2 generally continue to improve. This distinction matters: nominal context length measures how much sequence a model can ingest, not whether it can use the added sequence. We argue that long-context capability in GFMs should instead be measured by marginal task improvement as biologically relevant context is added, together with the corresponding compute cost. At matched inference cost, 6,144 GENERator tokens span 36,864 bp and yield higher recovery than Evo2-7B at 6,144 bp for several eukaryotic groups. Coarse tokenization is therefore valuable here not merely because it enlarges the nominal window, but because the added genomic span produces measurable gains.

5.2 Variant Effect Prediction

Sequence recovery does not establish whether predicted probabilities are informative for biological variation. We therefore evaluate *variant effect prediction* (VEP) (Benegas et al., 2024), testing whether nucleotide likelihoods distinguish clinically annotated reference and alternative alleles. Unlike sequence recovery, likelihood-based VEP accommodates both autoregressive and bidirectional models, allowing comparison with additional baselines such as NT.

Problem formulation. Given a reference sequence S and a single-nucleotide variant at position i with reference allele R and alternative allele A, we define the VEP score as the log-likelihood ratio

$$\mathrm{VEP}(R \to A) = \log \frac{p(s_i = R \mid \mathcal{C}_i)}{p(s_i = A \mid \mathcal{C}_i)}, \tag{4}$$

where $\mathcal{C}_i$ denotes the model-specific sequence context. Positive scores indicate that the model assigns greater probability to the reference allele; the scores are used to rank variants by predicted effect.

Nucleotide-level inference under k-mer tokenization. For models using k-mer tokenization, single-nucleotide probabilities are obtained by summing token probabilities over compatible k-mers. We apply the same marginalization procedure to every k-mer model, including baselines trained only to predict atomic tokens. The distinction is therefore not whether nucleotide marginals are available at evaluation, but whether they are native to the training objective: GENERator-v2 optimizes these same marginals through FNS, whereas other k-mer models derive them only post hoc from token logits.

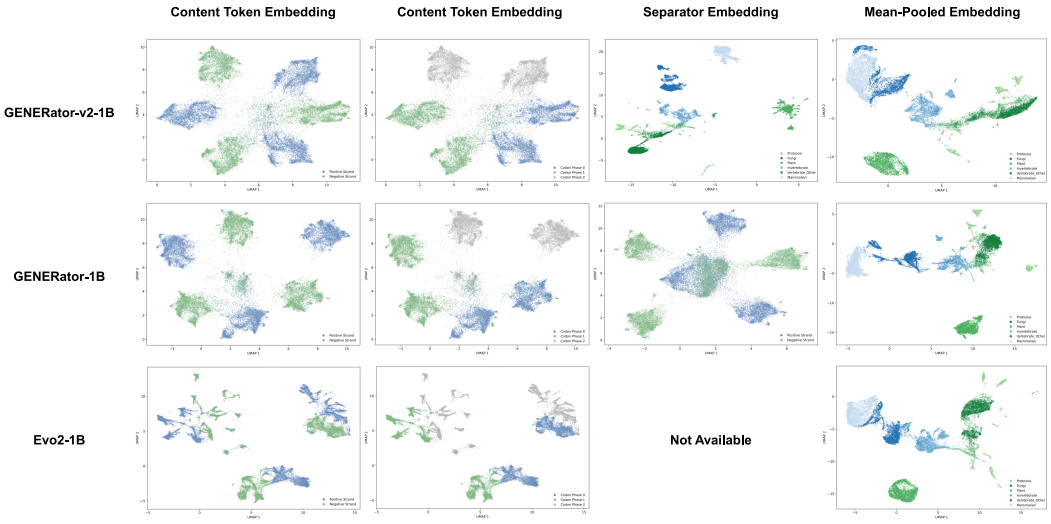

Figure 3: UMAP projections of sequence representations from GENERator-v2, GENERator, and Evo2. Rows denote models; columns denote embedding types and coloring variables. Content-token states recover two strand orientations and three codon phases; GENERator-v2 separator states instead emphasize taxonomic structure. Evo2 has no separator token, so the corresponding panel is omitted.

Evaluation protocol. We evaluate VEP on the ClinVar benchmark (Landrum et al., 2016) using AUROC and the GENERator scoring protocol (Wu et al., 2025). For bidirectional models, the variant is centered in the input window and its alleles are scored at the masked central position. For autoregressive models, the variant position is placed at the right boundary and its alleles are scored as alternative next nucleotides. We use each model's maximum supported context where computationally feasible. Evo2 is the exception: evaluation at its longer nominal windows is impractical on the full benchmark, so we retain the 8,192-bp setting used in its own evaluation (Brixi et al., 2026). All k-mer models use the nucleotide-level marginalization described above.

Results. As shown in Figure 2D, GENERator-v2 improves over GENERator at both parameter scales, from 0.910 to 0.914 AUROC at 1B and from 0.921 to 0.924 at 3B. GENERator-v2-1B also exceeds Evo2-1B (0.867), while GENERator-v2-3B attains the highest AUROC in this comparison, slightly exceeding Evo2-7B (0.921). Because every k-mer baseline receives the same beneficial marginalization at inference, the remaining advantage is consistent with training natively against nucleotide-level marginals rather than constructing them only at evaluation.

Context length and efficiency trade-offs. GENERator-v2 supports flexible accuracy–efficiency trade-offs in VEP. At the longest evaluated contexts, GENERator-v2-3B reaches 0.924 AUROC in 13.02 hours, compared with 0.921 in 13.37 hours for Evo2-7B, despite using approximately half as many parameters. At the smaller scale, GENERator-v2-1B reaches 0.914 AUROC in 6.50 hours, compared with 0.867 in 3.46 hours for Evo2-1B. NT-v3-650M uses an efficient U-Net–like architecture designed for single-base contexts up to 1 Mb (Boshar et al., 2025), yet reaches 0.809 AUROC here; support for a much larger nominal window does not translate into superior variant scoring. This echoes the recovery results in Section 5.1: context length measures what a model can ingest, whereas long-context understanding should be measured by marginal task improvement as biologically relevant context is added, together with its compute cost. Additional operating regimes are provided in Appendix C.

5.3 Embedding Representations

We analyze representation geometry rather than treating class separability alone as evidence of representation quality. Specifically, we ask whether 6-mer models recover biological structure below the token scale and whether separator states support a computation distinct from local sequence prediction.

Embedding definitions and evaluation setup. We extract three representations from pretrained models: (i) the final hidden state of the last content token; (ii) a *separator embedding*, obtained by appending <s> and extracting its hidden state; and (iii) a mean-pooled embedding over all tokens. We evaluate 30,000 genomic segments of 8,192 bp spanning six major eukaryotic taxonomic groups and visualize them with UMAP (McInnes et al., 2018), colored by species, DNA strand, or codon phase.

Local structure beyond token granularity. Content-token embeddings from GENERator and GENERator-v2 form a clean, approximately symmetric six-cluster geometry that resolves into two strand orientations and three codon phases. A 6-mer tokenizer supplies neither strand labels nor reading-frame labels. Although six is compatible with three-base codon periodicity, tokenization alone does not identify the coding frame or strand. The observed organization must therefore be learned from sequence statistics rather than imposed as an explicit labeling prior. Evo2's 1-mer embeddings contain the same biological signals but are substantially less regular and less clearly separated. Single-nucleotide tokenization is thus neither necessary nor, by itself, sufficient for learning organized nucleotide-scale structure. Mean-pooled states from all three models instead emphasize broader taxonomic variation.

Separator states and an objective shift. Separator embeddings expose a second representational hierarchy. In GENERator, they remain close to content-token geometry, retain strong local strand structure, and are included as ordinary prediction targets. GENERator-v2 instead excludes separator targets from the language-modeling loss, but the separator state remains in the causal context used to predict the next retained gene or regulatory region. Supervision therefore shifts from predicting the boundary token itself to using that boundary state for subsequent prediction: content states primarily support intra-region continuation, whereas the separator can aggregate preceding context into information useful across region boundaries. This role is functionally analogous to BERT's [CLS] token (Devlin et al., 2019), while remaining causal and trained through next-region prediction rather than a bidirectional classification objective. Consistent with this interpretation, GENERator-v2 separator embeddings suppress local strand and phase structure and organize more strongly by taxonomy than either GENERator separators or mean-pooled states.

Interpretation. These results distinguish token resolution from representational resolution. Coarse 6-mer inputs do not prevent the model from recovering two strand orientations and three codon phases, while loss masking at functional boundaries is associated with a transition from local sequence geometry to a context-summary state. The UMAP geometry does not by itself prove the proposed mechanism, but it provides direct evidence that GENERator-v2 organizes local sequence regularities and higher-level genomic context into different readouts.

5.4 Supervised Fine-Tuning and Training-Free Evaluation

Supervised fine-tuning remains an important test of whether pretrained representations can be adapted to heterogeneous labeled tasks. We evaluate the revised Nucleotide Transformer tasks (Dalla-torre et al., 2024), reporting Matthews correlation coefficient (MCC) under 10-fold cross-validation as in GENERator (Wu et al., 2025). We fine-tune GENERator-v2-1B for a scale-matched comparison. Sequences are truncated to a multiple of six and bounded by separator tokens, and the terminal separator state feeds a linear classifier. Evo2 is omitted because comparable public results are unavailable and reproducing its full fine-tuning is outside our computational scope.

GENERator-v2 matches or exceeds GENERator across the benchmark; nominally lower point estimates remain within the reported cross-validation uncertainty (Table 1). This strong fine-tuning performance is broadly consistent with its uniform gains in training-free recovery and VEP, showing that FNS-native pretraining preserves adaptability across heterogeneous tasks. Search details are in Appendix G.

Supervised fine-tuning and training-free evaluation are therefore complementary. The former measures adaptability after updating the backbone and task head; the latter isolates the capabilities of the fixed checkpoint. Task-specific optimization remains essential for downstream deployment, but it can attenuate differences between pretrained initializations and introduces sensitivity to datasets, task heads, and hyperparameters. As models scale, fine-tuning also becomes more costly and difficult

to reproduce, paralleling the shift in language-model evaluation toward zero- and few-shot protocols (Brown et al., 2020). In this setting, training-free evaluation provides scalable, reproducible evidence that complements supervised fine-tuning.

Table 1: Revised Nucleotide Transformer benchmark. Values are 10-fold mean MCC (standard error); bold and underline mark the best and second-best results.

	Enformer (252M)	DNABERT-2 (117M)	HyenaDNA (55M)	NT (2.5B)	NT-v2 (500M)	Caduceus-Ph (8M)	Caduceus-PS (8M)	GROVER (87M)	GENERator (1.2B)	GENERator-v2 (1.2B)
H2AFZ	0.522 (0.019)	0.490 (0.013)	0.455 (0.015)	0.503 (0.010)	0.524 (0.008)	0.417 (0.016)	0.501 (0.013)	0.509 (0.013)	0.529 (0.009)	**0.533 (0.005)**
H3K27ac	0.520 (0.015)	0.491 (0.010)	0.423 (0.017)	0.481 (0.020)	0.488 (0.013)	0.464 (0.018)	0.464 (0.022)	0.489 (0.023)	**0.546 (0.015)**	0.544 (0.012)
H3K27me3	0.552 (0.007)	0.599 (0.010)	0.541 (0.018)	0.593 (0.016)	0.610 (0.006)	0.547 (0.010)	0.561 (0.036)	0.600 (0.008)	0.619 (0.008)	**0.623 (0.003)**
H3K36me3	0.567 (0.017)	0.637 (0.007)	0.543 (0.010)	0.635 (0.016)	0.633 (0.015)	0.543 (0.009)	0.602 (0.008)	0.585 (0.008)	0.650 (0.006)	**0.659 (0.006)**
H3K4me1	0.504 (0.021)	0.490 (0.008)	0.430 (0.014)	0.481 (0.012)	0.490 (0.017)	0.411 (0.012)	0.434 (0.030)	0.468 (0.011)	0.504 (0.010)	**0.518 (0.006)**
H3K4me2	**0.626 (0.015)**	0.558 (0.013)	0.521 (0.024)	0.552 (0.022)	0.552 (0.013)	0.480 (0.013)	0.526 (0.035)	0.558 (0.012)	0.607 (0.010)	0.616 (0.011)
H3K4me3	0.635 (0.019)	0.646 (0.008)	0.596 (0.015)	0.618 (0.015)	0.627 (0.020)	0.588 (0.020)	0.611 (0.015)	0.634 (0.011)	0.653 (0.008)	**0.657 (0.014)**
H3K9ac	**0.593 (0.020)**	0.564 (0.013)	0.484 (0.022)	0.527 (0.017)	0.551 (0.016)	0.514 (0.014)	0.518 (0.018)	0.531 (0.014)	0.570 (0.017)	0.569 (0.008)
H3K9me3	0.453 (0.016)	0.443 (0.025)	0.375 (0.026)	0.447 (0.018)	0.467 (0.044)	0.435 (0.019)	0.455 (0.019)	0.441 (0.017)	0.509 (0.013)	**0.517 (0.018)**
H4K20me1	0.606 (0.016)	0.655 (0.011)	0.580 (0.009)	0.650 (0.014)	0.654 (0.011)	0.572 (0.012)	0.590 (0.020)	0.634 (0.006)	**0.670 (0.006)**	0.669 (0.004)
Enhancer	**0.614 (0.010)**	0.517 (0.011)	0.475 (0.006)	0.527 (0.012)	0.575 (0.023)	0.480 (0.008)	0.490 (0.009)	0.519 (0.009)	0.594 (0.013)	0.580 (0.007)
Enhancer type	**0.573 (0.013)**	0.476 (0.009)	0.441 (0.010)	0.484 (0.012)	0.541 (0.013)	0.461 (0.009)	0.459 (0.011)	0.481 (0.009)	0.547 (0.017)	0.546 (0.010)
Promoter all	0.745 (0.012)	0.754 (0.009)	0.693 (0.016)	0.761 (0.009)	0.780 (0.012)	0.707 (0.017)	0.722 (0.014)	0.721 (0.011)	**0.795 (0.005)**	**0.795 (0.005)**
Promoter non-TATA	0.763 (0.012)	0.769 (0.009)	0.723 (0.013)	0.773 (0.010)	0.785 (0.009)	0.740 (0.012)	0.746 (0.009)	0.739 (0.018)	**0.801 (0.005)**	**0.801 (0.006)**
Promoter TATA	0.793 (0.026)	0.784 (0.036)	0.648 (0.044)	0.944 (0.016)	0.919 (0.028)	0.868 (0.023)	0.853 (0.034)	0.891 (0.041)	0.950 (0.009)	**0.956 (0.013)**
Splice acceptor	0.749 (0.007)	0.837 (0.006)	0.815 (0.049)	0.958 (0.003)	**0.965 (0.004)**	0.906 (0.015)	0.939 (0.012)	0.812 (0.012)	0.964 (0.003)	0.964 (0.003)
Splice site all	0.739 (0.011)	0.855 (0.005)	0.854 (0.053)	0.964 (0.003)	0.968 (0.003)	0.941 (0.006)	0.942 (0.012)	0.849 (0.015)	0.966 (0.003)	**0.970 (0.002)**
Splice donor	0.780 (0.007)	0.861 (0.004)	0.943 (0.024)	0.970 (0.002)	0.976 (0.003)	0.944 (0.026)	0.964 (0.010)	0.842 (0.009)	0.977 (0.002)	**0.978 (0.002)**

6 Conclusion

Factorized Nucleotide Supervision (FNS) reframes the apparent conflict between coarse tokenization and nucleotide resolution as a supervision problem. By optimizing exact nucleotide marginals while retaining the full k-mer distribution, GENERator-v2 consistently improves over its predecessor, matches substantially larger models in training-free evaluations, and offers up to $44\times$ faster inference. Supervised fine-tuning further confirms that these gains preserve adaptability. More broadly, the experiments challenge two common assumptions: supporting longer inputs does not establish that a model benefits from the added sequence, and using single-nucleotide tokens does not guarantee more biologically organized nucleotide-scale representations. Long-context and representation quality should therefore be measured from task gains and learned biological structure, not inferred from window size or token granularity alone.

Important limitations remain. FNS retains blockwise attention and does not directly supervise within-block joint dependence, leaving room for objectives that combine joint and marginal signals. GCP depends on human-curated annotations, potentially biasing training toward known biology, while concatenation discards native physical distances. Better data construction should retain functional enrichment while reducing curation bias and preserving genomic geometry; broader validation across organisms and measured effects also remains necessary.

We release GENERator-v2 as a fully open resource, including model weights, training data, and code. Although it is not a final solution, its ability to match or exceed leading genomic models in key settings at a fraction of the inference cost provides a solid, reproducible foundation for further development and downstream use.

REFERENCES

Pavel Avdeyev, Chenlai Shi, Yuhao Tan, Kseniia Dudnyk, and Jian Zhou. Dirichlet diffusion score model for biological sequence generation, 2023. URL https://arxiv.org/abs/2305.10699.

Žiga Avsec, Vikram Agarwal, Daniel Visentin, Joseph R Ledsam, Agnieszka Grabska-Barwinska, Kyle R Taylor, Yannis Assael, John Jumper, Pushmeet Kohli, and David R Kelley. Effective gene expression prediction from sequence by integrating long-range interactions. *Nature methods*, 18 (10):1196–1203, 2021.

Žiga Avsec, Natasha Latysheva, Jun Cheng, Guido Novati, Kyle R Taylor, Tom Ward, Clare Bycroft, Lauren Nicolaisen, Eirini Arvaniti, Joshua Pan, et al. AlphaGenome: advancing regulatory variant effect prediction with a unified DNA sequence model. *bioRxiv*, pp. 2025–06, 2025.

Gonzalo Benegas, Carlos Albors, Alan J Aw, Chengzhong Ye, and Yun S Song. GPN-MSA: an alignment-based DNA language model for genome-wide variant effect prediction. *bioRxiv*, pp. 2023–10, 2024.

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. *arXiv preprint arXiv:2210.09461*, 2022.

Sam Boshar, Benjamin Evans, Ziqi Tang, Armand Picard, Yanis Adel, Franziska K Lorbeer, Chandana Rajesh, Tristan Karch, Shawn Sidbon, David Emms, Javier Mendoza-Revilla, Fatimah Al-Ani, Evan Seitz, Yair Schiff, Yohan Bornachot, Ariana Hernandez, Marie Lopez, Alexandre Laterre, Karim Beguir, Peter Koo, Volodymyr Kuleshov, Alexander Stark, Bernardo P de Almeida, and Thomas Pierrot. A foundational model for joint sequence-function multi-species modeling at scale for long-range genomic prediction. *bioRxiv*, pp. 2025.12.22.695963, 2025. doi: 10.64898/2025.12.22.695963.

Garyk Brixi, Matthew G. Durrant, Jerome Ku, et al. Genome modelling and design across all domains of life with Evo 2. *Nature*, 652:1349–1361, 2026. doi: 10.1038/s41586-026-10176-5.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901, 2020.

Hugo Dalla-torre, Liam Gonzalez, Javier Mendoza Revilla, Nicolás López Carranza, Adam Henryk Grzywaczewski, Francesco Oteri, Christian Dallago, Evan Trop, Hassan Sirelkhatim, Guillaume Richard, Marcin J. Skwark, Karim Beguir, Marie Lopez, and Thomas Pierrot. The Nucleotide Transformer: Building and evaluating robust foundation models for human genomics. *bioRxiv*, 2024.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*, 2019.

Yanrong Ji, Zhihan Zhou, Han Liu, and Ramana V. Davuluri. DNABERT: pre-trained bidirectional encoder representations from transformers model for DNA-language in genome. *bioRxiv*, 2020.

Taku Kudo and John Richardson. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Conference on Empirical Methods in Natural Language Processing*, 2018.

Chih-Horng Kuo, Nancy A. Moran, and Howard Ochman. The consequences of genetic drift for bacterial genome complexity. *Genome Research*, 19(8):1450–1454, 2009. doi: 10.1101/gr.091785.109.

Melissa J Landrum, Jennifer M Lee, Mark Benson, Garth Brown, Chen Chao, Shanmuga Chitipiralla, Baoshan Gu, Jennifer Hart, Douglas Hoffman, Jeffrey Hoover, et al. ClinVar: public archive of interpretations of clinically relevant variants. *Nucleic acids research*, 44(D1):D862–D868, 2016.

Qiuyi Li, Wei Wu, Yiheng Zhu, Fuli Feng, Jieping Ye, and Zheng Wang. GENERanno: A genomic foundation model for metagenomic annotation. *bioRxiv*, pp. 2025–06, 2025a.

Siyuan Li, Zedong Wang, Zicheng Liu, Di Wu, Cheng Tan, Jiangbin Zheng, Yufei Huang, and Stan Z Li. VQDNA: Unleashing the power of vector quantization for multi-species genomic sequence modeling. *arXiv preprint arXiv:2405.10812*, 2024.

Siyuan Li, Kai Yu, Anna Wang, Zicheng Liu, Chang Yu, Jingbo Zhou, Qirong Yang, Yucheng Guo, Xiaoming Zhang, and Stan Z Li. MergeDNA: Context-aware genome modeling with dynamic tokenization through token merging. *arXiv preprint arXiv:2511.14806*, 2025b.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. DeepSeek-V3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

Michael Lynch. Streamlining and simplification of microbial genome architecture. *Annual Review of Microbiology*, 60:327–349, 2006. doi: 10.1146/annurev.micro.60.080805.142300.

Michael Lynch and John S. Conery. The origins of genome complexity. *Science*, 302(5649): 1401–1404, 2003. doi: 10.1126/science.1089370.

Mingqian Ma, Guoqing Liu, Chuan Cao, Pan Deng, Tri Dao, Albert Gu, Peiran Jin, Zhao Yang, Yingce Xia, Renqian Luo, et al. HybriDNA: A hybrid transformer-mamba2 long-range DNA language model. *arXiv preprint arXiv:2502.10807*, 2025.

Leland McInnes, John Healy, and James Melville. UMAP: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

Eric Nguyen, Michael Poli, Matthew G. Durrant, Armin W. Thomas, Brian Kang, Jeremy Sullivan, Madelena Y Ng, Ashley Lewis, Aman Patel, Aaron Lou, Stefano Ermon, Stephen A. Baccus, Tina Hernandez-Boussard, Christopher Ré, Patrick D. Hsu, and Brian L. Hie. Sequence modeling and design from molecular to genome scale with Evo. *bioRxiv*, 2024.

Eric D 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. *ArXiv*, 2023.

Nuala A. O'Leary, Mathew W. Wright, James Rodney Brister, Stacy Ciufo, Diana Haddad, Richard McVeigh, Bhanu Rajput, Barbara Robbertse, Brian Smith-White, et al. Reference sequence (RefSeq) database at NCBI: current status, taxonomic expansion, and functional annotation. *Nucleic Acids Research*, 44:D733 – D745, 2015.

Donovan H Parks, Maria Chuvochina, Christian Rinke, Aaron J Mussig, Pierre-Alain Chaumeil, and Philip Hugenholtz. GTDB: an ongoing census of bacterial and archaeal diversity through a phylogenetically consistent, rank normalized and complete genome-based taxonomy. *Nucleic acids research*, 50(D1):D785–D794, 2022.

Lifeng Qiao, Peng Ye, Yuchen Ren, Weiqiang Bai, Chaoqi Liang, Xinzhu Ma, Nanqing Dong, and Wanli Ouyang. Model decides how to tokenize: Adaptive DNA sequence tokenization with mxDNA. *Advances in Neural Information Processing Systems*, 37:66080–66107, 2024.

Michelle Rönspies, Solmaz Khosravi, Ondřej Helia, Alessandro Valisi, Jiří Fajkus, Miloslava Fojtová, Andreas Houben, and Holger Puchta. CRISPR-Cas-mediated heritable chromosome fusions in Arabidopsis. *Science*, 390(6775):843–848, 2025. doi: 10.1126/science.adz8505.

Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models, 2024. URL https://arxiv.org/abs/2406.07524.

Melissa Sanabria, Jonas Hirsch, Pierre M. Joubert, and Anna R. Poetsch. DNA language model GROVER learns sequence context in the human genome. *Nat. Mac. Intell.*, 6:911–923, 2024.

Anirban Sarkar, Ziqi Tang, Chris Zhao, and Peter K Koo. Designing DNA with tunable regulatory activity using discrete diffusion. *bioRxiv*, 2024. doi: 10.1101/2024.05.23.595630. URL https://www.biorxiv.org/content/early/2024/05/24/2024.05.23.595630.

Yair Schiff, Chia-Hsiang Kao, Aaron Gokaslan, Tri Dao, Albert Gu, and Volodymyr Kuleshov. Caduceus: Bi-directional equivariant long-range DNA sequence modeling. *ArXiv*, abs/2403.03234, 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Yihui Wang, Zhiyuan Cai, Qian Zeng, Yihang Gao, Jiarui Ouyang, Yingxue Xu, Shu Yang, Sunan He, Yuxiang Nie, Yu Cai, et al. Genomic Touchstone: Benchmarking genomic language models in the context of the central dogma. *bioRxiv*, pp. 2025–06, 2025.

Wei Wu, Qiuyi Li, Mingyang Li, Kun Fu, Fuli Feng, Jieping Ye, Hui Xiong, and Zheng Wang. GENERator: a long-context generative genomic foundation model. *arXiv preprint arXiv:2502.07272*, 2025.

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. *ArXiv*, abs/2306.15006, 2023.

Zhihan Zhou, Robert Riley, Satria Kautsar, Weimin Wu, Rob Egan, Steven Hofmeyr, Shira Goldhaber-Gordon, Mutian Yu, Harrison Ho, Fengchen Liu, et al. GenomeOcean: An efficient genome foundation model trained on large-scale metagenomic assemblies. *bioRxiv*, 2025.

A FACTORIZED NUCLEOTIDE SUPERVISION (FNS)

This appendix provides formal definitions, theoretical results, and implementation details supporting Section 2. In particular, we show that the FNS objective is the negative log-likelihood of a derived predictor for the next k nucleotides under a shared conditioning state. This statement concerns the induced nucleotide marginals, not a factorization of the model's original joint distribution over k-mers.

A.1 SETUP AND NOTATION

Let $\mathcal{B} = \{A, C, G, T\}$ denote the nucleotide alphabet, let $\mathcal{V} = \mathcal{B}^k$ denote the regular k-mer vocabulary, and let $\mathcal{S}$ denote the set of special tokens. The full output vocabulary is $\mathcal{W} = \mathcal{V} \cup \mathcal{S}$. At autoregressive step t, the model produces logits $\mathbf{z}_t \in \mathbb{R}^{|\mathcal{W}|}$, inducing a categorical distribution

$$p_t(w) \triangleq \Pr(W_t = w \mid h_t) = \mathrm{softmax}(\mathbf{z}_t)_w, \quad w \in \mathcal{W},$$

where h_t denotes the conditioning state (e.g., the decoder hidden state determined by $x_{<t}$).

Each k-mer token $v \in \mathcal{V}$ corresponds to an ordered tuple of nucleotides $v = (v^{(1)}, \ldots, v^{(k)})$ with $v^{(j)} \in \mathcal{B}$. For nucleotide position $j \in \{1, \ldots, k\}$ and base $b \in \mathcal{B}$, we define the induced nucleotide marginal

$$p_t^{(j)}(b) \triangleq \Pr(B_{t,j} = b \mid h_t) = \sum_{v \in \mathcal{V} : v^{(j)} = b} p_t(v), \tag{5}$$

and introduce an auxiliary outcome $\perp$ with

$$p_t^{(j)}(\perp) \triangleq \sum_{s \in \mathcal{S}} p_t(s). \tag{6}$$

Thus, $p_t^{(j)}$ is a valid categorical distribution over $\mathcal{B}_\perp = \mathcal{B} \cup \{\perp\}$. If the output has no special tokens, the $\perp$ outcome has zero mass and the distribution reduces to the usual four-way nucleotide marginal.

Given a ground-truth k-mer target $y_t = (y_t^{(1)}, \ldots, y_t^{(k)}) \in \mathcal{V}$, the FNS per-step loss is defined as

$$\mathcal{L}_{\mathrm{FNS}}(t) \triangleq -\sum_{j=1}^{k} \log p_t^{(j)}\left(y_t^{(j)}\right). \tag{7}$$

The implementation minimizes $\overline{\mathcal{L}}_{\mathrm{FNS}}(t) = \mathcal{L}_{\mathrm{FNS}}(t)/k$. The two objectives have identical minimizers; the mean form keeps the loss scale comparable across token lengths.

A.2 FORMAL RESULTS

Definition A.1 (Blockwise 1-mer predictor)**.** Fix a step t with conditioning state h_t. A *blockwise 1-mer predictor* outputs k categorical distributions $\{q_t^{(j)}(\cdot)\}_{j=1}^k$ over $\mathcal{B}_\perp$ and defines a joint distribution over the next block $\tilde{Y}_t = (b_1, \ldots, b_k) \in \mathcal{B}_\perp^k$ via conditional independence:

$$\Pr(\tilde{Y}_t = (b_1, \ldots, b_k) \mid h_t) = \prod_{j=1}^{k} q_t^{(j)}(b_j). \tag{8}$$

This corresponds to predicting k consecutive 1-mer tokens in parallel, conditioned on the same h_t.

Theorem A.2 (FNS equals blockwise 1-mer negative log-likelihood)**.** *Let $p_t(w) = \mathrm{softmax}(\mathbf{z}_t)_w$ be the model distribution over the full vocabulary $\mathcal{W}$ at step t. Define induced nucleotide marginals $p_t^{(j)}(\cdot)$ by Eq. equation 5. Consider a blockwise 1-mer predictor with per-position distributions set to*

$$q_t^{(j)}(b) \equiv p_t^{(j)}(b), \quad \forall j \in \{1, \ldots, k\},\ b \in \mathcal{B}_\perp.$$

Then for any ground-truth k-mer target $y_t = (y_t^{(1)}, \ldots, y_t^{(k)}) \in \mathcal{B}^k$, the FNS loss equals the negative log-likelihood of predicting the next k 1-mer tokens under the blockwise 1-mer predictor:

$$\mathcal{L}_{\mathrm{FNS}}(t) = -\log \Pr(\tilde{Y}_t = y_t \mid h_t). \tag{9}$$

Proof. By Definition A.1 and Eq. equation A.2,

$$-\log \Pr(\tilde{Y}_t = y_t \mid h_t) = -\log \prod_{j=1}^{k} q_t^{(j)}(y_t^{(j)}) = -\sum_{j=1}^{k} \log q_t^{(j)}(y_t^{(j)}) = -\sum_{j=1}^{k} \log p_t^{(j)}(y_t^{(j)}). \quad (10)$$

The right-hand side is exactly the FNS loss by Eq. equation 7, proving Eq. equation 9. □

Implication for nucleotide-level supervision. Theorem A.2 shows that, although the model predicts one token at each step, the objective explicitly supervises the nucleotide outcome at each of the k positions. It does not imply that the original distribution $p_t(w)$ equals the product of its marginals, nor does it remove the blockwise attention granularity introduced by k-mer tokenization.

Corollary A.3 (SNP likelihood ratio under blockwise 1-mer equivalence)**.** *Let $y, y' \in \mathcal{B}^k$ differ only at position j. Under the blockwise 1-mer interpretation in Theorem A.2,*

$$\log \frac{\Pr(\tilde{Y}_t = y' \mid h_t)}{\Pr(\tilde{Y}_t = y \mid h_t)} = \log \frac{p_t^{(j)}(y'^{(j)})}{p_t^{(j)}(y^{(j)})}. \quad (11)$$

Proof. By Eq. equation 8,

$$\frac{\Pr(\tilde{Y}_t = y' \mid h_t)}{\Pr(\tilde{Y}_t = y \mid h_t)} = \prod_{\ell=1}^{k} \frac{p_t^{(\ell)}(y'^{(\ell)})}{p_t^{(\ell)}(y^{(\ell)})}. \quad (12)$$

All factors cancel except $\ell = j$ since $y'^{(\ell)} = y^{(\ell)}$ for $\ell \neq j$. □

A.3 Algorithmic Details

Algorithm 1 summarizes FNS for k-mer–tokenized DNA. It sums token probabilities into nucleotide marginals, averages the k position-wise losses, and uses token-level negative log-likelihood for any special targets retained by the training setup. Masked positions, including padding and GENERator-v2 separator targets, do not contribute. Regular and special losses are combined by their numbers of valid target positions.

Algorithm 1 Factorized Nucleotide Supervision (FNS) Loss

Input: logits $\mathbf{Z} \in \mathbb{R}^{B\times T\times V}$, integer labels $\mathbf{Y} \in \{1, \ldots, V\}^{B\times T}$, token length k, nucleotide index map Idx, special-token IDs $\mathcal{S}$
Output: mean scalar loss $\overline{\mathcal{L}}_{\mathrm{FNS}}$
Shift $\mathbf{Z}$ and $\mathbf{Y}$ for autoregressive prediction
Mask invalid label positions in $\mathbf{Y}$
$\mathcal{M}_{\mathrm{reg}} \leftarrow$ valid positions with non-special k-mer labels
$\mathcal{M}_{\mathrm{spec}} \leftarrow$ valid positions with special-token labels
Initialize $\mathcal{L}_{\mathrm{bp}} \leftarrow 0$, $\mathcal{L}_{\mathrm{tok}} \leftarrow 0$
$\mathbf{P} \leftarrow \mathrm{softmax}(\mathbf{Z})$
if $\mathcal{M}_{\mathrm{reg}} \neq \varnothing$ **then**
  for $j = 1$ **to** k **do**
    $\mathbf{y}^{(j)} \leftarrow \mathsf{Idx}[\mathbf{Y}, j]$ on $\mathcal{M}_{\mathrm{reg}}$
    Construct marginal probabilities $\mathbf{P}^{(j)} \in \mathbb{R}^{B\times T\times 4}$ by summing $\mathbf{P}$ over regular k-mers sharing the j-th nucleotide
    $\mathcal{L}_{\mathrm{bp}} \leftarrow \mathcal{L}_{\mathrm{bp}} - \mathrm{mean}_{\mathcal{M}_{\mathrm{reg}}} \log \mathbf{P}^{(j)}[\mathbf{y}^{(j)}]$
  end for
  $\mathcal{L}_{\mathrm{bp}} \leftarrow \mathcal{L}_{\mathrm{bp}}/k$
end if
if $\mathcal{M}_{\mathrm{spec}} \neq \varnothing$ **then**
  $\mathcal{L}_{\mathrm{tok}} \leftarrow -\mathrm{mean}_{\mathcal{M}_{\mathrm{spec}}} \log \mathbf{P}[\mathbf{Y}]$
end if
Return:

$$\overline{\mathcal{L}}_{\mathrm{FNS}} \leftarrow \frac{|\mathcal{M}_{\mathrm{reg}}|\,\mathcal{L}_{\mathrm{bp}} + |\mathcal{M}_{\mathrm{spec}}|\,\mathcal{L}_{\mathrm{tok}}}{|\mathcal{M}_{\mathrm{reg}}| + |\mathcal{M}_{\mathrm{spec}}|} \quad (13)$$

B SEQUENCE RECOVERY

This section provides additional analysis of the sequence-recovery experiments in Section 5.1, focusing on input length, context scaling, and computational trade-offs.

Input length selection. The 6,144-bp input used in the main comparison is shorter than the maximum context of GENERator, GENERator-v2, and Evo2. We select it to control evaluation cost while comparing recovery at an identical nucleotide span; the separate timing benchmark quantifies the resulting compute difference.

Effect of input length on recovery accuracy. Figure S1 reports recovery over shorter and longer inputs. Under this protocol, Evo2 improves little beyond approximately 1,024 bp, whereas GENERator and GENERator-v2 generally continue to improve as additional context is provided.

With 6,144 GENERator tokens (36,864 bp), GENERator-3B has inference time comparable to Evo2-7B at 6,144 bp and higher recovery for other vertebrates, invertebrates, and plants. This comparison illustrates the additional nucleotide span available to a 6-mer model at a similar inference cost.

Implications for long-context modeling. This plateau challenges the use of a maximum supported window as evidence of long-context capability. A nominal window states what a model can ingest; useful long-context understanding requires measurable gains when additional biologically relevant sequence is exposed. The result on one recovery task does not imply that Evo2 cannot exploit long context on every task, but it shows why context claims require scaling curves rather than a single advertised limit. We therefore suggest reporting marginal performance gain as a function of input span, its compute and memory cost, and its consistency across taxa and tasks. By these criteria, coarse tokenization here translates a fixed token budget into context that is not only accessible but empirically useful.

B.1 RESULTS ACROSS INPUT LENGTHS

The complete context-length sweeps supporting the analysis above are shown in Figure S1.

C VARIANT EFFECT PREDICTION

This section provides additional analysis of the variant-effect prediction (VEP) experiments in Section 5.2, focusing on context length and accuracy–efficiency trade-offs.

Operating regimes and context length selection. Where computationally feasible, we use the maximum context supported by each model rather than force identical input lengths across tokenizers. GENERator-v2 uses its full 16,384-token window (98,304 bp). Evo2 is the exception: scoring the full ClinVar benchmark at its longer nominal windows is impractical, so we retain the 8,192-bp setting used in the Evo2 evaluation (Brixi et al., 2026). Because the evaluated spans differ, we report both accuracy and wall-clock time rather than treating this as a matched-context comparison.

Accuracy and efficiency under matched inference budgets. Under this protocol, GENERator-v2-3B and Evo2-7B require similar inference time on ClinVar (13.02 versus 13.37 hours) and reach 0.924 and 0.921 AUROC, respectively, although GENERator-v2-3B has approximately half as many parameters. At the smaller scale, GENERator-v2-1B requires 6.50 hours versus 3.46 hours for Evo2-1B and reaches 0.914 versus 0.867 AUROC. These points characterize the observed accuracy–efficiency frontier; they do not control context length independently of model family.

Accuracy–efficiency trade-offs. We also evaluate GENERator-v2 with an 8,192-bp context. GENERator-v2-1B completes inference in 0.36 hours with 0.913 area under the precision–recall curve (AUPRC), and GENERator-v2-3B requires 0.68 hours with 0.928 AUPRC. These complementary precision–recall results show that shorter contexts substantially reduce inference cost while retaining useful variant-ranking performance.

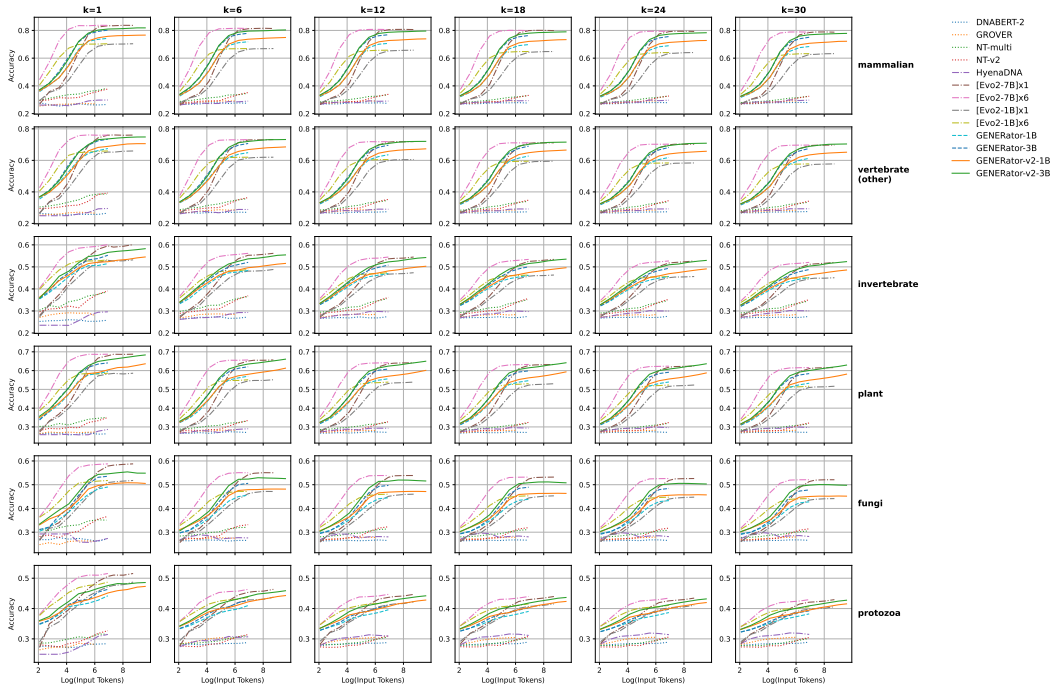

Figure S1: Sequence-recovery accuracy as a function of input length. Each panel corresponds to one taxonomic group and continuation length; the horizontal axis reports the number of input tokens on a logarithmic scale. Because Evo2 uses single-nucleotide tokens whereas GENERator and GENERator-v2 use 6-mers, we report two Evo2 configurations: ×1 uses the displayed number of tokens for comparison at matched inference cost, whereas ×6 uses six times as many tokens to match the input sequence length, or nucleotide span, of the 6-mer models. Curves for GENERator and GENERator-v2 show mean recovery under the common protocol described in Section 5.1.

Implications. These results provide two practical operating regimes: longer contexts maximize accuracy, whereas shorter contexts reduce cost while retaining much of the performance. Reporting both avoids conflating model quality with a single context-length choice.

D Embedding Representations

This section provides additional details on the extraction and interpretation of sequence embeddings used in Section 5.3, clarifying differences across embedding types and model architectures.

Embedding types. We consider three types of sequence embeddings throughout this work: (i) *content token embeddings*, obtained from the hidden state of the final content token in the input sequence; (ii) *separator embeddings*, obtained from the hidden state of a special separator token <s> appended to the sequence; and (iii) *mean-pooled embeddings*, obtained by averaging hidden states across all tokens in the sequence.

Embedding extraction for GENERator and GENERator-v2. For both GENERator and GENERator-v2, embeddings are extracted from the final hidden layer of the Transformer decoder, and <s> marks boundaries between retained regions in their functional-sequence streams. The separator-specific difference is in the objective: GENERator includes separator tokens as ordinary prediction targets, whereas GENERator-v2 masks those targets from the language-modeling loss. The separator state in GENERator-v2 nevertheless remains in the causal context and receives learning signal through predictions in the following region. It can therefore aggregate the preceding region into a boundary representation useful for predicting the next retained gene or regulatory element. This mechanism motivates, but does not by itself prove, the context-summary interpretation of the UMAP geometry.

Embedding extraction for Evo2. Unlike GENERator and GENERator-v2, Evo2 does not employ an explicit separator or end-of-sequence token in its vocabulary. Accordingly, we do not construct separator embeddings for Evo2, and comparisons are restricted to content-token embeddings and mean-pooled sequence embeddings.

Evo2 uses a 24-layer StripedHyena2 backbone (Brixi et al., 2026). In our extraction runs, representations become more structured through layer 21, whereas layers 22–24 show activation magnitudes on the order of 10^{10} and do not yield interpretable clusters. We therefore use layer 21, the deepest numerically stable layer in our pipeline, for all Evo2 embedding analyses.

E GENOME COMPRESSION PRETRAINING (GCP)

This appendix describes the annotation-aware preprocessing summarized in Section 3. GCP is a data transformation rather than an architectural modification: it packs selected annotated regions into long training streams while retaining their genomic order.

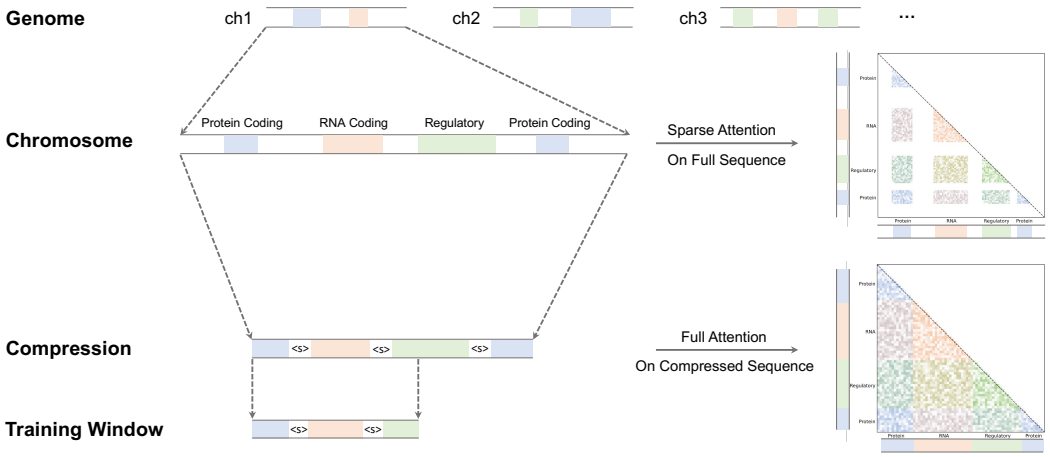

Figure S2: Genome Compression Pretraining (GCP). Selected gene-centric and regulatory regions are concatenated in genomic order, with separator tokens marking region boundaries. Windows sampled from the resulting stream contain a higher proportion of annotated sequence while retaining standard dense self-attention.

E.1 MOTIVATION AND CONSTRUCTION

Uniform sampling from eukaryotic chromosomes makes sequence without annotated functional content dominate the training distribution. Restricting pretraining to selected intervals increases annotation density, but individual intervals are generally much shorter than the model context. GCP concatenates these intervals before sampling training windows.

GCP was empirically validated in GENERator: gene-centric functional-sequence pretraining outperforms an all-sequence control across downstream tasks even though the latter attains lower pretraining loss (Wu et al., 2025). Evo2 independently corroborates this policy: its genic-window–enriched mixture outperforms a whole-genome control on BRCA1, ClinVar, and splice-variant evaluations (Brixi et al., 2026). These results do not imply that excluded bases are uniformly dispensable; they show that functional-context enrichment can improve downstream capability even with less raw sequence.

For each chromosome, we select protein-coding genes (including coding sequence, introns, and untranslated regions), RNA genes, regulatory elements, promoters, and selected flanking or intervening regions with potential regulatory roles. We sort the retained intervals by genomic coordinate, tokenize each independently, and concatenate them with separator tokens. Separators are visible to the model but masked from the language-modeling loss. This conservative selection prioritizes information retention over maximal compaction and reduces the RefSeq eukaryotic corpus from approximately 2×10^{12} to 4×10^{11} bp. Figure S2 illustrates the pipeline.

The resulting stream preserves the order of retained regions but not their original base-pair distances. Fixed-length windows can consequently contain multiple functional units while using the same dense-attention architecture and autoregressive objective. This procedure depends on curated annotations and may bias training toward known biology. We use it only where the expected gain in information density is large: for sparse eukaryotic genomes. Prokaryotic genomes are already compact and gene dense, so analogous filtering offers less compression while risking loss of operonic and intergenic context; the prokaryotic models therefore use uncompressed RefSeq and GTDB sequences.

E.2 Why Genomic Redundancy Is Expected

A common objection to annotation-aware filtering is that unannotated sequence can still be functional. This objection is correct but does not imply that uniform sampling is optimal. GCP requires only that functional information is distributed nonuniformly: omitting some unannotated signal can be beneficial when it removes a much larger volume of weakly constrained background and thereby improves the signal-to-noise ratio of the training distribution. The following toy model makes the evolutionary-search asymmetry explicit.

Suppose viability requires K sequence modules with lengths $\ell_1, \ldots, \ell_K$ and total functional length $\ell = \sum_{i=1}^{K} \ell_i$. A genome of length $L \geq \ell$ preserves their order but permits arbitrary gaps before, between, and after the modules. Writing these gap lengths as nonnegative integers $g_0, \ldots, g_K$ gives $\sum_{i=0}^{K} g_i = L - \ell$. Hence the number of ordered layouts is

$$A(L, \ell, K) = \binom{L - \ell + K}{K}. \tag{14}$$

Proposition E.1 (Expected number of viable modular embeddings)**.** *Assume an independent uniform nucleotide background, and let p_i be the probability that a random segment of length ℓ_i realizes an acceptable instance of module i. If X_L counts ordered, non-overlapping embeddings of all K modules in a random length-L sequence, then*

$$\mathbb{E}[X_L] = A(L, \ell, K) \prod_{i=1}^{K} p_i. \tag{15}$$

For a perfectly compact sequence ($L = \ell$), this reduces to $\mathbb{E}[X_\ell] = \prod_i p_i$.

Proof. By the stars-and-bars argument, Equation 14 counts the admissible gap vectors. Associate an indicator with each layout that equals one when every prescribed module is realized. Its expectation is $\prod_i p_i$ because the module intervals do not overlap. Summing these indicators and applying linearity of expectation yields Equation 15. No independence between different candidate layouts is required. □

For exact module sequences, $p_i = 4^{-\ell_i}$ and the compact expectation is $4^{-\ell}$. With $\ell = 1{,}000$, $K = 100$, and $L = 100{,}000$, the compact value is approximately $10^{-602.1}$, whereas the layout factor is $\binom{99{,}100}{100} \approx 10^{341.6}$. Thus, even with module order fixed, added background increases the expected number of valid embeddings by more than 341 orders of magnitude. If module order is unconstrained, the multiplicity can increase by up to a further factor of $100! \approx 10^{158.0}$. For rare, weakly overlapping embeddings, $\Pr(X_L \geq 1) \approx \mathbb{E}[X_L]$; in general, $\Pr(X_L \geq 1) \leq \mathbb{E}[X_L]$, so we use the exact expectation rather than presenting the approximation as a probability identity.

This calculation is not a literal model of evolution: genomes are modified locally by mutation, duplication, recombination, transposition, selection, and drift rather than sampled uniformly in one step. Its implication is nevertheless sharp. Requiring every nucleotide to be indispensable confines viability to an extremely small and mutationally fragile part of sequence space, whereas modular function embedded in weakly constrained sequence creates many layouts and neutral or near-neutral neighbors. Perfect compactness is possible as a formal construction, but it is not a realistic null model for complex eukaryotic genomes.

Population genetics explains why this asymmetry differs across domains of life. The efficacy of selection against a weakly deleterious insertion scales with effective population size N_e: effects much smaller than approximately $1/N_e$ are dominated by drift. Classic comparative estimates

place long-term N_e for many prokaryotes and multicellular eukaryotes several orders of magnitude apart, reflecting differences in census size, reproductive contribution, and life history (Lynch & Conery, 2003). Large microbial populations can therefore remove weakly costly excess DNA more efficiently, whereas weaker selection in many multicellular eukaryotes permits introns, mobile elements, duplications, and longer intergenic regions to persist (Lynch, 2006). Generation time contributes to this contrast, but the relevant quantity is long-term N_e, not lifecycle alone. The trend is not universal—severe reductions in bacterial N_e, particularly in host-associated lineages, can instead drive gene loss and genome reduction (Kuo et al., 2009)—but the broad domain-level difference motivates our asymmetric preprocessing: GCP has high potential leverage on sparse eukaryotic chromosomes, while complete-sequence training better preserves the already dense context of prokaryotic genomes.

Experimental chromosome engineering provides a complementary illustration of robustness. CRISPR-mediated arm fusions converted the native ten-chromosome karyotype of *Arabidopsis thaliana* into two distinct eight-chromosome lines; both lines were fertile and showed no detected phenotypic or transcriptional differences from wild type, although crosses back to wild type had reduced fertility and altered recombination (Rönspies et al., 2025). This result demonstrates tolerance to large-scale genome reorganization, not that every displaced base is nonfunctional. Together, the combinatorial, population-genetic, and empirical arguments support the premise behind GCP: filtering need not imply that excluded sequence is useless, because a small loss of unannotated signal can be outweighed by a much larger reduction in weakly constrained background. Our conservative $5\times$ eukaryotic compaction retains protein-coding genes, RNA genes, regulatory elements, and nearby regions with potential regulatory roles; the prokaryotic pipeline retains complete genomes.

E.3 Setup and Notation

Nucleotide alphabet and sequences. Let $\mathcal{B} = \{A, C, G, T\}$ denote the nucleotide alphabet. A DNA sequence is a finite string $S = (b_1, \ldots, b_n) \in \mathcal{B}^n$.

Reference genomes and annotations. A reference genome G consists of chromosomes $\{C\}$. Each chromosome C has an associated DNA sequence $\mathrm{seq}(C)$ and length $|C|$. Functional annotations are represented as intervals

$$\mathcal{A}(C) = \{(s_i, e_i, \tau_i)\},$$

where $0 \le s_i < e_i \le |C|$ and τ_i denotes the annotation type.

Non-overlapping k-mer tokenization. Let k denote the k-mer size. A non-overlapping tokenization with offset $\delta \in \{0, \ldots, k-1\}$ maps $S = (b_1, \ldots, b_n)$ to

$$\mathrm{Tok}_\delta(S) = ((b_{\delta+1:\delta+k}), (b_{\delta+k+1:\delta+2k}), \ldots),$$

discarding incomplete k-mers at the boundaries.

Model and hidden states. We consider an autoregressive decoder parameterized by θ. Given previous tokens $\mathbf{x}_{<t}$, the model produces a hidden state $h_t \in \mathbb{R}^d$ and logits $\mathbf{z}_t \in \mathbb{R}^{4^k}$ over the k-mer vocabulary.

Separator token. <s> denotes a special token used to mark boundaries between functional regions. Separator tokens are included in the input but masked from the language-modeling loss.

E.4 Tokenization-Shift Augmentation

Non-overlapping k-mer tokenization is inherently phase-sensitive: shifting the tokenization offset δ by one nucleotide yields a completely different sequence of token IDs. When combined with long-context training, this raises a natural concern: does the choice of tokenization phase affect which nucleotide positions receive supervision during training?

We use all k offsets during training. The result below establishes balanced nucleotide coverage under this augmentation: away from boundaries, every nucleotide is supervised once per offset. It does not assert that predictions or losses are identical across offsets.

E.4.1 NUCLEOTIDE-LEVEL LOSS INDUCED BY FNS

Let $S = (b_1, \ldots, b_n) \in \mathcal{B}^n$ be a DNA sequence. For a fixed offset $\delta \in \{0, \ldots, k-1\}$, let $\mathrm{Tok}_\delta(S)$ produce m_δ non-overlapping k-mer tokens. Under FNS, each token contributes k nucleotide-level negative log-likelihood terms. We define the total nucleotide-level loss induced by offset δ as

$$\mathcal{L}_\delta(S) \triangleq \sum_{i=1}^{m_\delta} \sum_{j=1}^{k} \ell_{\delta,\, \delta+(i-1)k+j}, \tag{16}$$

where $\ell_{\delta,t}$ is the FNS negative log-probability assigned to nucleotide b_t when the sequence is tokenized at offset δ. The offset is retained in the notation because the tokenization and conditioning state can change the prediction.

E.4.2 BALANCED COVERAGE ACROSS TOKENIZATION SHIFTS

Because different offsets δ cover different subsets of nucleotide positions, the per-offset losses $\mathcal{L}_\delta(S)$ are not identical. However, in training protocols that exhaustively cycle through all k offsets (e.g., one offset per epoch), the relevant quantity is the *combined* nucleotide-level supervision across offsets.

Proposition E.2 (Balanced nucleotide coverage under exhaustive shifts)**.** *Let $S = (b_1, \ldots, b_n) \in \mathcal{B}^n$ and $k \geq 1$. For each offset $\delta \in \{0, \ldots, k-1\}$, define $\mathcal{L}_\delta(S)$ as in Eq. equation 16. Let I_δ be the nucleotide positions covered by complete k-mers at offset δ, and let $D_t = \{\delta : t \in I_\delta\}$. Then*

$$\sum_{\delta=0}^{k-1} \mathcal{L}_\delta(S) = \sum_{t=1}^{n} \sum_{\delta \in D_t} \ell_{\delta,t}. \tag{17}$$

Every interior position $t \in \{k, \ldots, n-k+1\}$ satisfies $|D_t| = k$. If the sequence is padded on both sides by at least $k-1$ symbols before tokenization and losses on padding are masked, then $|D_t| = k$ for every original position t.

Proof. Fix an offset δ and define the set of nucleotide positions covered by complete k-mers under δ as

$$I_\delta \triangleq \{t \in \{1, \ldots, n\} :\ t = \delta + (i-1)k + j \text{ for some } i \in \{1, \ldots, m_\delta\},\ j \in \{1, \ldots, k\}\}.$$

By definition, $\mathcal{L}_\delta(S) = \sum_{t \in I_\delta} \ell_{\delta,t}$. Summing over offsets yields

$$\sum_{\delta=0}^{k-1} \mathcal{L}_\delta(S) = \sum_{t=1}^{n} \sum_{\delta \in D_t} \ell_{\delta,t}, \tag{18}$$

which proves Eq. equation 17 by exchanging finite sums. For any interior position, a complete k-mer contains that position at every offset, so $D_t = \{0, \ldots, k-1\}$ and $|D_t| = k$. Only positions within $k-1$ bases of a boundary can be omitted at some offsets. Padding supplies complete boundary-crossing k-mers for every offset; masking padded positions then retains exactly k loss terms for each original nucleotide. □

Interpretation. Exhaustive shifts prevent any interior nucleotide from being systematically omitted and balance the number of supervision terms across phases. Because $\ell_{\delta,t}$ can vary with δ, Proposition E.2 establishes coverage balance, not phase-invariant predictions.

E.5 IMPLICATIONS OF GENOME COMPRESSION

Context utilization and effective span. Compressed streams are not contiguous representations of the original chromosomes. A length-L window can contain multiple selected regions in genomic order even when those regions are far apart in physical distance, increasing the amount of annotated sequence in each model input. The trade-off is that exact inter-region distances and omitted sequence are unavailable to the model.

Tokenization-shift augmentation under FNS. Non-overlapping k-mer tokenization is phase sensitive: shifting the offset changes every token boundary. We cycle through all k offsets during training so that each interior nucleotide is supervised equally often at every phase. Proposition E.2 establishes this coverage property but does not guarantee identical predictions across phases.

Relationship to sparse attention. GCP changes the distribution of input sequence rather than the attention operator. All retained tokens participate in dense self-attention, whereas omitted sequence cannot influence the representation. It is therefore complementary to sparse-attention methods and makes a different trade-off: greater annotation density at the cost of genomic contiguity and physical-distance information.

Separator representations. The inserted separator creates a repeated boundary state whose hidden representation can summarize preceding retained context and condition the next region. The taxonomic organization observed in separator embeddings (Section 5.3) supports its role as a high-level readout within compressed streams.

E.6 Algorithmic Details

Algorithm 2 describes the construction of compressed training sequences used in GCP. Training windows of fixed length L tokens are sampled from $\mathbf{X}_C$ using random offsets or a fixed stride, and loss terms corresponding to separator tokens are masked.

Algorithm 2 Genome Compression Pretraining (GCP)

Input: reference-genome collection $\mathcal{G}$, chromosome annotations $\mathcal{A}(C)$, tokenizer Tok_δ, separator token <s>
Output: compressed token streams $\{\mathbf{X}_C\}$
for each genome $G \in \mathcal{G}$ **do**
  for each chromosome $C \in G$ **do**
    Filter $\mathcal{A}(C)$ by annotation type and length
    Sort intervals by genomic coordinate
    Initialize empty token list $\mathbf{X}_C$
    for each interval (s_i, e_i, τ_i) **do**
      Extract $S_i \leftarrow \mathrm{seq}(C[s_i : e_i])$
      Tokenize $\mathbf{x}_i \leftarrow \mathrm{Tok}_\delta(S_i)$
      Append $\mathbf{x}_i$ to $\mathbf{X}_C$
      Append separator <s> to $\mathbf{X}_C$
    end for
  end for
end for
Return: $\{\mathbf{X}_C\}$

F Model Configurations

Table S1 summarizes the domains, parameter scales, pretraining strategies, and data sources used by GENERator and GENERator-v2.

Table S1: Configurations of GENERator and GENERator-v2. All models use the same Transformer decoder, non-overlapping 6-mer tokenization, and a maximum context of approximately 98 kb.

Model	Domain	Parameters	FNS	GCP	Data source
GENERator	Eukaryote	1.2B / 3B	✗	✓	RefSeq
GENERator-v2	Eukaryote	1.2B / 3B	✓	✓	RefSeq
GENERator-v2	Prokaryote	1.2B / 3B	✓	✗	RefSeq + GTDB

G Fine-Tuning Hyperparameters

For the fine-tuning evaluation in Section 5.4, we search learning rates in $\{1 \times 10^{-5}, 2 \times 10^{-5}, 5 \times 10^{-5}\}$ and batch sizes in $\{32, 64, 128, 256\}$. We retain the pretraining optimizer settings ($\beta_1 = 0.9$, $\beta_2 = 0.95$, weight decay $= 0.1$), use 100 linear warmup steps followed by learning-rate reduction

on a validation plateau, and apply early stopping with patience 5. Table S2 reports the selected hyperparameters for reproducibility.

Table S2: Selected GENERator-v2 fine-tuning hyperparameters for the revised Nucleotide Transformer tasks.

Task	Learning rate	Batch size
H2AFZ	1×10^{-5}	32
H3K27ac	2×10^{-5}	32
H3K27me3	1×10^{-5}	32
H3K36me3	1×10^{-5}	128
H3K4me1	2×10^{-5}	64
H3K4me2	2×10^{-5}	64
H3K4me3	2×10^{-5}	64
H3K9ac	1×10^{-5}	32
H3K9me3	2×10^{-5}	32
H4K20me1	1×10^{-5}	64
Enhancer	2×10^{-5}	32
Enhancer type	2×10^{-5}	32
Promoter all	2×10^{-5}	256
Promoter non-TATA	1×10^{-5}	32
Promoter TATA	1×10^{-5}	32
Splice acceptor	2×10^{-5}	32
Splice site all	2×10^{-5}	32
Splice donor	2×10^{-5}	64

AI USE STATEMENT

In this work, generative AI tools were used to assist with manuscript editing, including improvements to language, organization, clarity, and formatting. All AI-assisted revisions were reviewed by the authors, who verified the scientific claims, citations, and reported results and take full responsibility for the final content.