MOLTOKEN: SCALING A MOLECULAR FOUNDATION MODEL WITH UNIFIED GRAPH TOKENIZATION ACROSS CHEMICAL AND BIOLOGICAL SPACE

Anonymous authors
Paper under double-blind review

ABSTRACT

A central goal of molecular foundation modeling is to jointly model chemical and biological space. However, existing molecular foundation models often retain class-specific representations and encoding schemes even in joint modeling, while scaling behavior under a unified chemical representation remains underexplored. We introduce **MolToken**, a **Mol**ecular foundation model with unified graph **Token**ization across chemical and biological space. MolToken maps small molecules, peptides, proteins, DNA, and RNA into a unified atom-level graph space. Strictly reversible Frequency Dual Depth First Search (FDDFS) serialization and Byte Pair Encoding (BPE) yield a unified vocabulary for joint pretraining with a single autoregressive Transformer. BPE reduces the total length of FDDFS sequences by a factor of approximately 71.8, making atom-level modeling more tractable. We systematically compare joint and single-domain pre-training across model sizes and data scales. Within the evaluated range, compute-optimal model and data scales follow power laws, while optimal loss decreases approximately linearly with log compute. Out-of-sample experiments support loss prediction at substantially larger compute. Cross-domain comparisons reveal domain-dependent benefits and trade-offs of joint training and suggest that single-domain scaling cannot substitute for cross-domain data coverage. Downstream fine-tuning further demonstrates the adaptability of the unified pretrained backbone to molecular generation and design.

1 INTRODUCTION

In recent years, deep learning has been widely applied to molecular modeling and design, achieving substantial progress in drug-like molecule generation (Peng et al., 2026), molecular property prediction (Liu et al., 2021), peptide and protein design (Li et al., 2024), and nucleic acid modeling (Nguyen et al., 2023). A growing number of specialized models have been developed for molecules of different types and scales, advancing AI-assisted drug discovery and design.

However, existing methods are typically developed for specific molecular classes and rely on distinct representations. Small molecules are commonly represented as SMILES (Weininger, 1988), SELFIES (Krenn et al., 2020), or molecular graphs (Ji et al., 2024); proteins and peptides are typically modeled as amino acid sequences (Li et al., 2026); and DNA and RNA (Nguyen et al., 2023) are primarily represented as nucleotide sequences. This representational heterogeneity places different molecular classes in separate representation spaces, making it difficult for a single model to natively learn shared chemical regularities across molecular types and scales, thereby limiting the development of unified molecular foundation models.

Overcoming this representational heterogeneity to build a unified foundation model across molecular types and scales presents two key challenges. *How can a single model learn a unified representation across molecular domains?* Some existing methods integrate molecular domains through fusion of discrete structural tokens produced by multiple domain-specific encoders (Pei et al., 2026) or token-level fusion of class-specific sequence representations (Li et al., 2026), but still rely on class-specific representations and encoding schemes. *How can we quantify the benefits of joint over single-domain pre-training as scale increases?* In frameworks based on domain-specific encoding and cross-domain fusion, scaling involves multiple encoders and representation fusion, complicating controlled comparisons of scaling behavior between multi-domain and single-domain pre-training.

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.

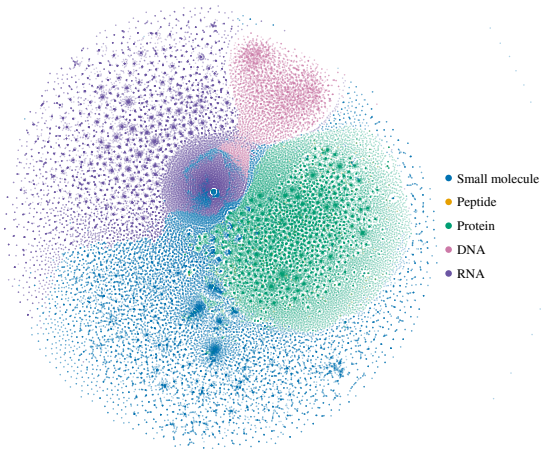


Figure 1: t-SNE visualization of molecular samples based on normalized BPE token-frequency vectors. Each point represents one sample, whose atom–level graph is serialized by FDDFS and tokenized using a BPE vocabulary shared across five molecular domains.

Whether joint pre-training offers benefits beyond scaling single-domain pre-training alone, and how these benefits vary with model and data scales, remain to be systematically investigated.

To address these challenges, we propose **MolToken**, a unified molecular foundation model spanning small molecules, peptides, proteins, DNA, and RNA. MolToken constructs a naturally unified graph tokenizer across molecular domains using atom-level graphs, strictly reversible *Frequency Dual Depth First Search* (FDDFS) serialization, and Byte Pair Encoding (BPE) (Gage, 1994). Rather than fusing representations from multiple domain-specific encoders, it jointly learns a unified vocabulary across chemical and biological space, enabling joint pre-training with a single autoregressive Transformer. As shown in Figure 1, the five molecular domains form distinct yet overlapping distributions in the unified token space, retaining domain-specific structure while exhibiting cross-domain continuity. This unified representation and modeling framework enables us to compare the scaling behavior of joint and single-domain pre-training, examining how the benefits of joint training vary with model parameter count and training data size. We further evaluate the pretrained models across diverse molecular downstream tasks to assess their cross-task transferability and generalization.

In this work, our contributions can be summarized as follows:

- **Unified Molecular Representation and Modeling.** We introduce MolToken, a molecular foundation model with unified graph tokenization across chemical and biological space. Specifically, we combine atom-level graphs, strictly reversible FDDFS serialization, and BPE tokenization to learn a vocabulary shared across molecular domains, enabling joint modeling of small molecules and biopolymers with a single autoregressive Transformer.
- **Scaling Analysis across Molecular Domains.** We systematically study the scaling behavior of joint and single-domain pre-training within a unified representation and modeling framework. We examine how model parameter count, training data size, and domain composition affect joint and domain-specific pre-training performance.
- **Empirical Findings on Compression and Scaling.** BPE reduces the total length of FDDFS sequences by a factor of approximately 71.8, making joint pre-training across molecular domains computationally feasible. The scaling relations for joint pre-training support out-of-sample prediction, while domain-specific comparisons reveal distinct scaling behavior. Within the evaluated range, our results suggest that single-domain scaling cannot substitute for cross-domain data coverage. We also achieve competitive performance on downstream tasks spanning multiple molecular domains.

2 Related Work

Molecular Representations. Small molecules are commonly represented as strings such as SMILES (Weininger, 1988) and SELFIES (Krenn et al., 2020), three-dimensional point clouds comprising atom types and coordinates, or molecular graphs with atoms as nodes and chemical bonds as edges. Peptides and proteins are typically represented by amino acid sequences, all-atom or backbone coordinates, and residue contact graphs (Jing et al., 2021). DNA and RNA are represented

using nucleotide sequences, atomic coordinates, or graphs encoding backbone connectivity and base pairing; RNA secondary structures are also commonly described using dot-bracket notation. Some methods further apply predefined rules to decompose molecules into substructures such as rings and bonds, yielding higher-level abstractions (Jin et al., 2018). These representations differ in their basic units and levels of abstraction, making it difficult to construct a vocabulary shared across molecular classes. Our method learns a shared structural vocabulary over unified atom-level graphs, combining representational consistency across molecular classes with compact sequences for macromolecules.

Molecular and Biomolecular Foundation Models. Existing models typically adopt domain-specific representations, as in Uni-Mol2 (Ji et al., 2024) for small molecules, ESM3 (Hayes et al., 2025) for proteins, and Evo 2 (Brixi et al., 2026) for genomes. PocketXMol (Peng et al., 2026) unifies three-dimensional structure prediction and generation for small molecules and peptides through atomic interactions. LOGOS (Li et al., 2026) and BioMatrix (Pei et al., 2026) extend joint modeling across classes but retain class-specific units and encoding schemes. LOGOS establishes local chemical correspondences by encoding pocket-residue side chains as SMILES, while retaining amino acid sequences and SMILES as distinct symbol systems within a shared language-model vocabulary. BioMatrix combines atom-level small-molecule tokens and residue-level protein tokens in a shared vocabulary, but their separate structural encoders and independent codebooks preserve class-specific boundaries. MolToken represents molecules from all supported domains as atom-level graphs over shared attribute spaces and jointly learns a unified vocabulary through FDDFS serialization and BPE tokenization, establishing unified molecular tokenization and naturally aligning representations across domains without requiring fusion-specific training.

Graph Foundation Models and Graph Serialization. Graph foundation models explore graph–text alignment, as in GraphTranslator (Zhang et al., 2024) and LLaGA (Chen et al., 2024), as well as sequence-based graph modeling, as in G2PT (Chen et al., 2025). Graph serialization often relies on random walks or graph traversals, such as DeepWalk (Perozzi et al., 2014) and FEuler (Guo et al., 2026), but these methods do not guarantee strict reversibility. Our method builds on Turán's (1984) reversible planar graph encoding based on spanning-tree traversal and non-tree edge records. We combine two-pass depth-first traversal (Weininger, 1988) with frequency-guided ordering in a unified serialization scheme for attributed graphs, using the same encoding rules across small molecules and biopolymers to produce strictly reversible sequences.

Neural Scaling Laws. Neural scaling laws characterize empirical relationships between model performance, parameter count, training data size, and computation. Kaplan et al. (2020) systematically study the scaling behavior of language models, while Hoffmann et al. (2022) investigate the optimal allocation of model scale and training data under fixed compute budgets through analyses including IsoFLOP curves. In the molecular domain, Uni-Mol2 (Ji et al., 2024) explores scaling in small-molecule pretraining, and Cheng et al. (2024) study compute-optimal training of protein language models under different pretraining objectives. These studies primarily focus on individual molecular classes. In contrast, our work investigates scaling behavior and compute-optimal training for joint pretraining across molecular classes under a shared representation.

3 METHOD

3.1 PRELIMINARIES

Attributed graphs. An attributed graph is a tuple $G = (V, E, x, z)$. Here V is a finite set of vertices and $E \subseteq \binom{V}{2}$ is a set of edges. A vertex may carry several attributes, each taking values in a finite discrete set $\mathcal{X}_i$, and the vertex attribute set is their product $\mathcal{X} = \prod_i \mathcal{X}_i$; the edge attribute set $\mathcal{Z} = \prod_j \mathcal{Z}_j$ is formed in the same way. The attribute functions $x : V \to \mathcal{X}$ and $z : E \to \mathcal{Z}$ assign one element of $\mathcal{X}$ or $\mathcal{Z}$ to each vertex or edge. Each element of $\mathcal{X}$ or $\mathcal{Z}$ is treated as a single symbol, so two vertices share a symbol only when they agree on every attribute. We write $\mathcal{G}$ for the set of all attributed graphs over given $\mathcal{X}$ and $\mathcal{Z}$.

Molecular graphs. Let $\mathcal{G}$ be a class of attributed undirected molecular graphs $G = (V, E, x, z)$, where vertices represent atoms and edges represent chemical bonds. The maps $x : V \to \mathcal{X}$ and $z : E \to \mathcal{Z}$ assign vertex and edge attributes, respectively. In this work, we represent small molecules, peptides, proteins, DNA, and RNA as atom-level graphs over the shared attribute spaces $\mathcal{X} = \prod_{i=1}^{6} \mathcal{X}_i$ and $\mathcal{Z} = \prod_{j=1}^{3} \mathcal{Z}_j$. The factors $\mathcal{X}_1, \ldots, \mathcal{X}_6$ are the value sets of atomic number, formal charge, aromaticity, chirality tag, degree, and total valence, respectively; $\mathcal{Z}_1, \ldots, \mathcal{Z}_3$ are those of bond type, aromaticity, and stereochemical tag, respectively.

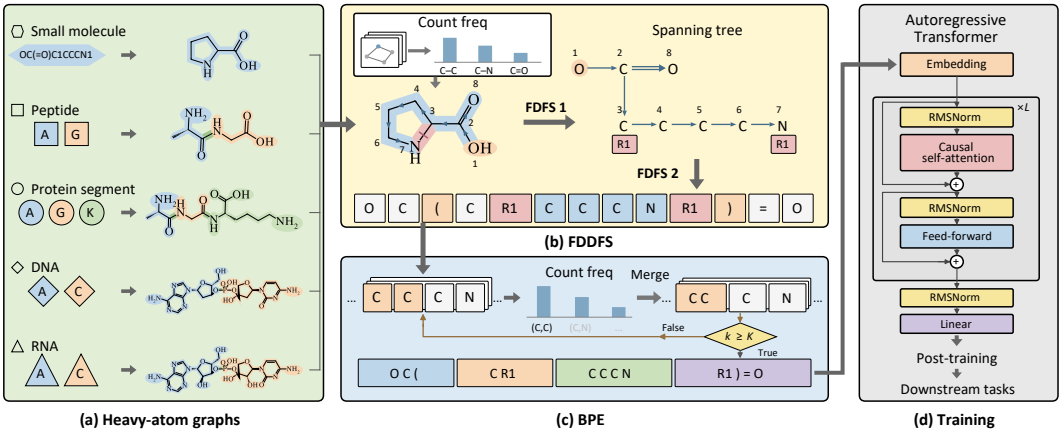


Figure 2: Overview of MolToken. (a) Molecular inputs are converted into attributed atom-level graphs, with examples from five pretraining domains. (b) In FDDFS, the first Frequency-guided DFS (FDFS) builds spanning trees using corpus frequencies; the second emits reversible sequences with branch and ring markers. (c) BPE repeatedly merges the most frequent adjacent symbol pair, yielding compact sequences and a vocabulary shared across domains. (d) A single Transformer is jointly pretrained via next-token prediction and fine-tuned for downstream tasks.

Strict reversibility. Two attributed graphs are isomorphic, written $G_1 \cong G_2$, if there exists a bijection $\phi : V_1 \to V_2$ such that $\{u, v\} \in E_1$ if and only if $\{\phi(u), \phi(v)\} \in E_2$ for all $u, v \in V_1$, and such that $x_1 = x_2 \circ \phi$ and $z_1 = z_2 \circ \hat{\phi}$, where $\hat{\phi}(\{u, v\}) = \{\phi(u), \phi(v)\}$. Given a finite alphabet Σ, a serialization encoder $\mathrm{Enc} : \mathcal{G} \to \Sigma^*$ maps a graph to a sequence and is strictly reversible if there is a decoder $\mathrm{Dec} : \Sigma^* \to \mathcal{G}$ with

$$\mathrm{Dec}\big(\mathrm{Enc}(G)\big) \cong G \qquad \text{for all } G \in \mathcal{G}, \tag{1}$$

both maps being computable in time polynomial in $|V|$ (Turán, 1984, Definition 2). Here Enc serializes a graph over $\Sigma = \mathcal{X} \dot\cup \mathcal{Z} \dot\cup \mathcal{M}$, where $\mathcal{M}$ is a finite set of additional symbols introduced by the serialization scheme.

3.2 Unified Molecular Graph Tokenization

Figure 2 summarizes the conversion of molecular inputs into attributed atom-level graphs, FDDFS sequences, and BPE tokens for autoregressive modeling. Algorithm 1 specifies the complete procedure from molecular inputs to the learned vocabulary.

Unified graph construction. Graph construction is a map $\Gamma : \mathcal{I} \to \mathcal{G}$ (Algorithm 1, line 1), where $\mathcal{I}$ comprises molecular inputs that can be converted into atom-level graphs in $\mathcal{G}$, regardless of their molecular class or original representation. Our pretraining corpus covers five domains: small molecules, peptides, proteins, DNA, and RNA. For these datasets, small molecules are parsed from SMILES with RDKit, while a biopolymer symbol sequence $(a_1, \ldots, a_n)$ is assembled from residue graphs $G_{a_1}, \ldots, G_{a_n} \in \mathcal{G}$ as

$$\Gamma\big((a_1, \ldots, a_n)\big) = \Big(\dot\bigcup_{i=1}^{n} G_{a_i}\Big) + \{e_1, \ldots, e_{n-1}\}, \tag{2}$$

where $\dot\bigcup$ is the disjoint union of graphs and e_i is the backbone edge that joins residues i and $i+1$, from carbonyl carbon to backbone nitrogen in peptides and from $\mathrm{O3'}$ to P in nucleic acids. Residue graphs, terminal-group conventions, and worked examples are given in Appendix B.1.

Frequency-guided graph serialization. Our serializer follows the spanning-tree/non-tree-edge decomposition of classical graph encodings (Turán, 1984) and introduces a frequency-guided traversal order. The resulting FDDFS procedure consists of three steps in Algorithm 1.

(a) Corpus frequency statistics. Over the constructed graph corpus $\mathcal{D}$, we compute $f_N : \mathcal{X} \to \mathbb{N}_0$, $f_E : \mathcal{Z} \to \mathbb{N}_0$, and $f_T : \mathcal{X} \times \mathcal{Z} \times \mathcal{X} \to \mathbb{N}_0$, counting vertex attributes, edge attributes, and directed atom–bond–atom triples, respectively (line 2). Each undirected edge contributes both orientations to f_T. We collect these statistics as $\mathcal{F} = (f_N, f_E, f_T)$, which is shared across molecular domains.

Algorithm 1 Unified molecular graph tokenization

Require: training molecules $\mathcal{I}_{\text{tr}} \subseteq \mathcal{I}$, number of merges K
Ensure: $\mathcal{F}$, vocabulary $\mathcal{W}$, merge rules $\mathcal{R}$
1: $\mathcal{D} \leftarrow \{\Gamma(m) : m \in \mathcal{I}_{\text{tr}}\}$
2: $\mathcal{F} \leftarrow (f_N, f_E, f_T)$; ▷ counts over $\mathcal{D}$
3: **for** $G = (V, E, x, z) \in \mathcal{D}$ **do**
4: $\quad \pi_v \leftarrow \text{sort}^{\downarrow}_{\kappa_v} N(v), \quad \forall v \in V$ ▷ neighbors $N(v)$ by equation 3
5: $\quad A \leftarrow \varnothing$; $E_T \leftarrow \varnothing$ ▷ visited vertices, tree edges
6: $\quad$ **for** $Q \in \mathcal{Q}(G)$ **do** ▷ connected components
7: $\quad\quad r_Q \leftarrow \arg\max_{v \in Q} \big(f_N(x(v)), |N(v)|\big)$ ▷ get start vertex as the tree root
8: $\quad\quad$ FIRST(r_Q)
9: $\quad$ **end for**
10: $\quad S_G \leftarrow$ SECOND($G, E_T, (r_Q)_Q, (\pi_v)_v$) ▷ Second FDFS
11: **end for**
12: $(\mathcal{W}, \mathcal{R}) \leftarrow \text{BPE}\big(\{S_G\}_{G \in \mathcal{D}}, K\big)$
13: **return** $(\mathcal{F}, \mathcal{W}, \mathcal{R})$

14: **function** FIRST(v) ▷ DFS based on a given neighbor traversal order
15: $\quad A \leftarrow A \cup \{v\}$
16: $\quad$ **for** $u \in \pi_v$ with $u \notin A$ **do**
17: $\quad\quad E_T \leftarrow E_T \cup \{\{v, u\}\}$; FIRST($u$)
18: $\quad$ **end for**
19: **end function**

(b) Traversal order and first FDFS. For $G = (V, E, x, z)$, let $N(v)$ denote the neighbors of vertex v and $\mathcal{Q}(G)$ denote the connected components of G. Each neighbor $u \in N(v)$ is assigned the atom–bond–atom triple priority

$$\kappa_v(u) = f_T\big(x(v), z(\{v, u\}), x(u)\big). \tag{3}$$

By sorting $N(v)$ in decreasing order of κ_v, we obtain the visiting order π_v (line 4). In each component $Q \in \mathcal{Q}(G)$, the starting vertex r_Q maximizes $\big(f_N(x(v)), \deg(v)\big)$ lexicographically, where $\deg(v) = |N(v)|$ (line 7). Starting from component root r_Q of each connected component, the first FDFS follows the prescribed neighbor visitation order π_v, recursively visiting the unvisited neighbors of each vertex v in this order (line 16). This traversal produces a spanning forest $T = (V, E_T)$ covering all vertices. The tree edges E_T are those traversed when first discovering new vertices, and the remaining edges $E \setminus E_T$ are non-tree edges.

(c) Second FDFS and sequence emission. The second FDFS traverses the spanning forest using the same neighbor orders and emits the graph sequence $S_G = \text{Enc}(G)$ (line 10), following the graph encoding scheme of Turán (1984). During this traversal, each non-tree edge is represented by a pair of matching ring labels at its two endpoints; for example, the $R1$ symbols in Fig. 2(b) mark the corresponding non-tree connection. These labels record edges that are not explicitly traversed in the tree. Branching structures are marked by parentheses at the beginning and end of each branch, while the attribute symbols of visited vertices are emitted along the traversal. These symbols encode the complete graph structure as S_G together. For graphs with multiple connected components, the traversal starts from each component root r_Q, with separators inserted between components. Details of the serialization and decoding procedures and theoretical analysis are provided in Appendix B.

BPE tokenization. In line 12 of Algorithm 1, BPE initializes the vocabulary with the base symbols used in FDDFS sequences and iteratively merges the most frequent pair of adjacent symbols. The resulting vocabulary $\mathcal{W}$ and ordered merge rules $\mathcal{R}$ are shared across domains, encoding recurring fragments as variable-length graph tokens to reduce sequence length. Figure 3 traces successive merges from a common chemical root into amino-acid, nucleotide, and ring fragments. In molecular context, these tokens correspond to familiar chemical or biological motifs as well as data-driven combinations whose boundaries differ from predefined building blocks.

3.3 PRETRAINING AND SCALING LAWS

Autoregressive pretraining. MolToken uses a decoder-only Transformer trained with per-token negative log-likelihood. Let $\mathcal{B} = \{\mathbf{t}^{(j)}\}_{j=1}^{B}$ be a training batch, where $\mathbf{t}^{(j)} = (t_1^{(j)}, \ldots, t_{n_j}^{(j)})$ denotes

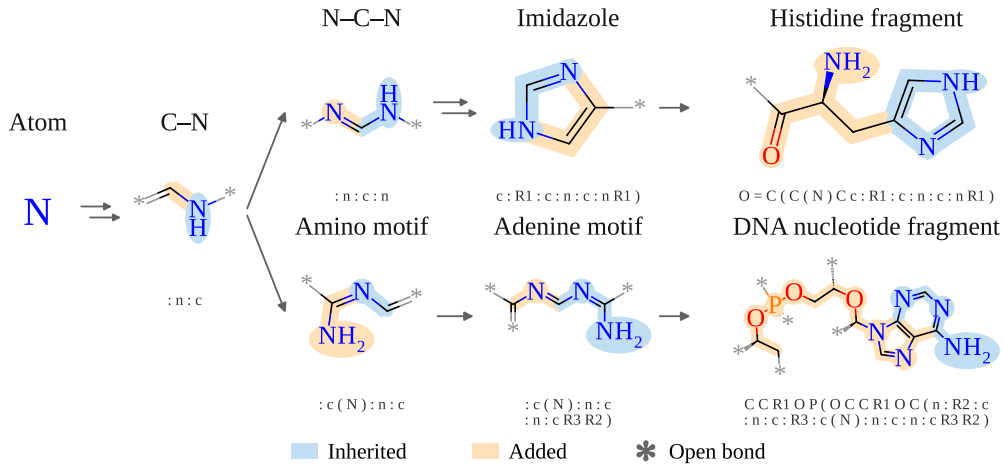


Figure 3: Hierarchical BPE merges from an atom into amino-acid, nucleotide, and ring fragments.

a token sequence. For a model with parameters θ, the loss is

$$\mathcal{L}(\theta;\mathcal{B}) = -\frac{1}{\sum_{j=1}^{B} n_j} \sum_{j=1}^{B} \sum_{i=1}^{n_j} \log p_\theta\left(t_i^{(j)} \mid t_1^{(j)}, \ldots, t_{i-1}^{(j)}\right), \tag{4}$$

All scaling analyses use the same token-averaged negative log-likelihood.

Cross-domain scaling. For a fixed model family and training protocol, $L(N, D)$ denotes the loss of a model with N parameters trained on D tokens. We evaluate it under four settings, written $s \in \{\texttt{1-to-1}, \texttt{m-to-m}, \texttt{1-to-m}, \texttt{m-to-1}\}$ and differing in which domains are trained on and which are evaluated: *one-to-one*, training and evaluation within the same domain; *many-to-many*, joint training with joint evaluation; *one-to-many*, single-domain training with joint evaluation; and *many-to-one*, joint training with evaluation on a single domain. Each setting gives a separate loss surface $L_s(N, D)$.

Compute-optimal scaling. Following the IsoFLOP framework of Hoffmann et al. (2022), we estimate the optimal allocation of model parameters and training tokens under a fixed compute budget $C = 6ND$ (Kaplan et al., 2020). For each setting,

$$\left(N_s^*(C), D_s^*(C)\right) \in \underset{N,D:\,6ND=C}{\arg\min} L_s(N, D), \qquad L_s^*(C) = L_s\left(N_s^*(C), D_s^*(C)\right). \tag{5}$$

We fit the optimal scales with power laws and the frontier loss with a semi-logarithmic form,

$$N_s^*(C) \approx K_{N,s} C^{a_s}, \qquad D_s^*(C) \approx K_{D,s} C^{b_s}, \qquad L_s^*(C) \approx u_s \log_{10} C + v_s, \tag{6}$$

where $K_{N,s}, K_{D,s}$ are scale coefficients, a_s, b_s are allocation exponents, and u_s, v_s are loss-model parameters. The compute constraint forces $a_s + b_s = 1$. Parameters are fitted separately for each setting and used to predict the allocation and loss at compute budgets excluded from fitting. Appendix D gives the estimation procedure and uncertainty quantification.

4 Experiments

We first analyze the sequence compression and context utilization enabled by BPE across molecular domains. Under this shared tokenization, we investigate the predictability of joint-pretraining scaling and compare domain-specific scaling behavior between joint and single-domain pretraining. We further examine whether scaling single-domain pretraining can compensate for limited molecular-domain coverage. Finally, we evaluate downstream adaptation across molecular tasks, with detailed protocols and complete results provided in Appendix F.

4.1 Experimental Setup

Datasets. The pretraining corpus covers small molecules, peptides, proteins, DNA, and RNA, comprising approximately 3.23×10^8 molecular graphs and 9.90×10^9 BPE tokens. All domains use the shared representation and unified vocabulary introduced in Section 3. This corpus size is distinct from the cumulative training token count, which includes repeated passes over the data. Domain composition, data processing, and test splits are detailed in Appendix A.1.

Table 1: Compression and sequence length across molecular domains. Counts and lengths exclude EOS and PAD. M denotes 10^6 and B denotes 10^9.

Domain	Records	Corpus tokens (B)		Compression Ratio	BPE length (tokens)			
	(M)	FDDFS	BPE	(×)	Mean	Median	P90	P99
Small molecule	92.452	5.274	0.924	5.71	10.00	8	16	32
Peptide	0.844	0.372	0.016	22.73	19.41	13	41	47
Protein	108.336	125.434	5.428	23.11	50.10	42	79	161
DNA	23.561	201.054	1.332	150.94	56.53	43	77	79
RNA	98.209	378.643	2.202	171.93	22.43	19	35	67

Models and training. All experiments use the same decoder-only Transformer family. The joint-pretraining scaling suite spans approximately 5.98M–458.08M parameters under a fixed optimization configuration. Following the definitions of parameter count N, cumulative training tokens D, and compute budget C in Section 3.3, we conduct IsoFLOP analyses by varying model size and training tokens at fixed compute. Complete model configurations, training hyperparameters, and fitting procedures are provided in Appendices A.2 and D.

Evaluation. We evaluate $L_s(N, D)$ under the four settings defined in Section 3: joint pretraining with joint ($s = \texttt{m-to-m}$) or domain-specific ($s = \texttt{m-to-1}$) evaluation, and single-domain pretraining with same-domain ($s = \texttt{1-to-1}$) or joint ($s = \texttt{1-to-m}$) evaluation. All losses are token-averaged negative log-likelihoods. Joint loss aggregates domain losses in proportion to the number of tokens included in loss computation for each domain. We compare the `1-to-1` and `m-to-1` frontiers using $N_s^*(C)$, $D_s^*(C)$, and $L_s^*(C)$ defined in equation 5. For joint pretraining, we report both total compute C and the target-domain compute share $r_k C$, where r_k is domain k's share of joint-training tokens. The corresponding target-domain token count is $r_k D_s^*(C)$; this rescaling leaves model size $N_s^*(C)$ and loss $L_s^*(C)$ unchanged. Loss estimation procedures and data-repetition effects are detailed in Appendix D.

4.2 Experimental Results

4.2.1 Tokenization Analysis

Sequence compression and context utilization. BPE substantially compresses molecular graph sequences, allowing a fixed context window to cover more atomic-level structure. It merges frequent subsequences in FDDFS representations, including recurring backbone and local connectivity patterns, while preserving exact graph reconstruction. Across the full corpus, BPE reduces the token count from 7.108×10^{11} to 9.903×10^9, corresponding to a corpus-level compression ratio of $71.78\times$. Table 1 shows larger reductions for biopolymers, particularly DNA and RNA, and a substantial narrowing of the differences in mean sequence length across domains. For the filtered and windowed pretraining records, the 99th-percentile BPE length in every domain falls below the 511-token content budget. By reducing the number of autoregressive predictions required to represent the same molecular graph, this compression makes atom-level modeling more tractable for biopolymers and supports joint pretraining across all domains.

4.2.2 Scaling and Predictability of Joint Pretraining

Compute-optimal scaling. Figure 4(a) shows that optimal loss decreases with compute, while the optimal configurations shift toward larger models and more training tokens. For joint pretraining, the compute-optimal allocation exhibits approximately balanced growth between model capacity and training data, with the fitted exponents indicating a near-equal allocation of additional compute to the two factors. The model-size scaling relation provides a strong fit to the observed optima ($R^2 = 0.987$; Figure 4(b)), and optimal loss decreases approximately linearly with $\log_{10} C$ ($R^2 = 0.993$; Figure 4(c)). Across the `m-to-m`, `1-to-1`, and `m-to-1` settings, the fitted scaling relations show similar compute-optimal trends, with differences in the allocation patterns across training settings (Table D.1). Uncertainty estimates and test-set robustness checks are also provided in Appendix D.

Out-of-sample prediction. Using the fitted `m-to-m` loss relation summarized in Table D.1, we predict a loss of 4.0370 for the 0.55B model at approximately $37.5\times$ the largest fitting budget. The observed loss is 4.0348, giving an absolute prediction error of approximately 2.24×10^{-3} (Figure 4(c)). This experiment evaluates out-of-sample loss prediction at a configuration selected using the fitted scaling relations. Detailed prediction procedures and error analysis are provided in Appendix D.5.

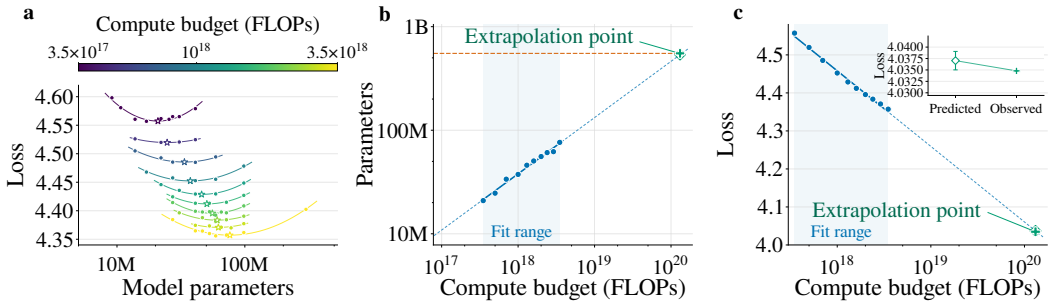


Figure 4: Joint-pretraining scaling and out-of-sample prediction. (a) IsoFLOP curves; (b) compute-optimal model size and (c) optimal loss. The extrapolated fit in (c) matches the observed loss of a 0.55B-parameter model trained at $37.5\times$ the largest fitting budget (absolute error 2.24×10^{-3}).

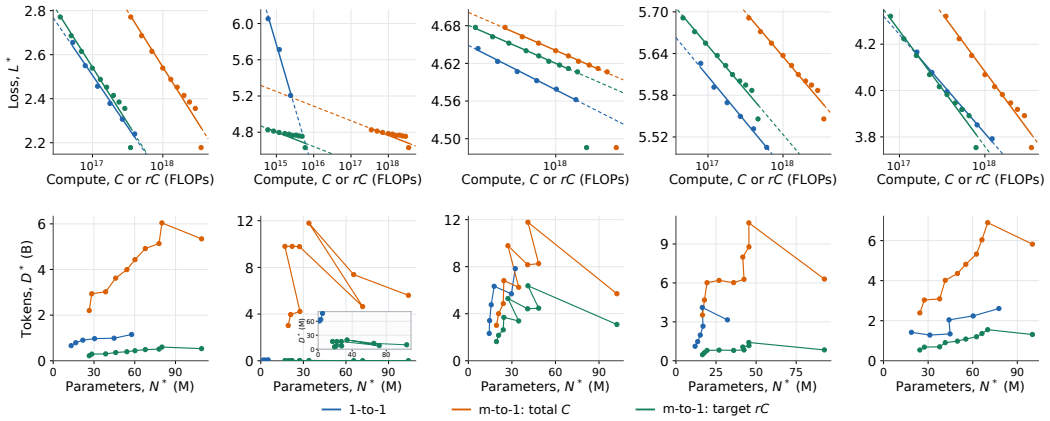


Figure 5: Single-domain and joint-pretraining frontiers for small molecules, peptides, proteins, DNA, and RNA (left to right). Top: loss frontiers. Bottom: optimal model–data configurations. Joint training uses total $(C, D_s^*(C))$ or target-domain-normalized $(r_k C, r_k D_s^*(C))$ accounting.

4.2.3 Domain-Specific Scaling and Joint Pretraining

Target-domain data efficiency and trade-offs. Joint pretraining reduces the target-domain data needed to reach a given loss, most clearly for peptides. For peptides, which account for only 0.17% of joint-training tokens, joint pretraining attains domain-specific loss at or below the single-domain frontier (top row of Figure 5) while using fewer target-domain tokens $r_k D$ than single-domain pretraining (bottom row), with the remaining training data drawn from other domains. Small molecules and RNA show the same pattern over part of the compute range. The size of this benefit differs across domains: for proteins and DNA, the single-domain frontier lies below the target-domain-normalized joint frontier over their overlapping compute range (top row). Details of the target-domain-normalized accounting are given in Appendix E.3.

Domain-specific scaling. The `1-to-1` and `m-to-1` fits in Table D.1 show decreasing optimal loss and power-law growth in compute-optimal model size across all five molecular domains over the evaluated range. Complete IsoFLOP curves and fitting diagnostics are provided in Appendix E.1.

4.2.4 Molecular-Domain Diversity and Joint Modeling

Joint performance of single-domain models. With more training compute, single-domain models achieve lower in-domain loss, while joint loss improves little and can even increase. The `1-to-1` fits show that the optimal in-domain loss decreases with compute in all five domains (Table D.1). In contrast, joint evaluation (`1-to-m`) shows that increasing compute for single-domain pretrained models yields only limited reductions in joint loss (Figure 6). In some configurations, joint loss even increases as compute grows (RNA panel of Figure 6): for the 41M-parameter RNA model, raising the compute budget from 2.4×10^{17} to 8.1×10^{17} FLOPs reduces its in-domain loss from 4.084 to 3.858 but increases its joint loss from 15.706 to 17.844 (Table E.5).

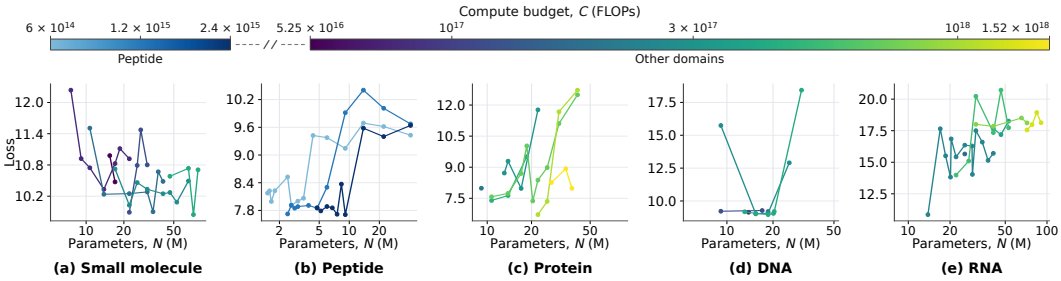


Figure 6: Joint loss of single-domain pretrained models. Panels indicate training domains; colors denote compute budgets. Increasing the compute budget does not consistently reduce joint loss.

Table 2: Results on three representative downstream datasets. The best scores are shown in **bold**, the second-best are underlined. A dash ("—") indicates task inputs not supported by the baseline.

Model	MOSES				RAbD	TCR External-13	
	Validity↑	Unique↑	Novelty↑	FCD/TestSF↓	AAR (%)↑	Rec. $\geq 90\%$ ↑	GIANA↑
VAE	0.9767	0.9984	0.6949	**0.5670**	—	—	—
DiGress	0.8572	**1.000**	0.9503	1.193	—	—	—
DisCo-GT	0.8834	**1.000**	**0.9773**	1.438	—	—	—
RefineGNN	—	—	—	—	34.81	—	—
TCRT5	—	—	—	—	—	**3.462**	**8.000**
TCRT5 FT	—	—	—	—	—	3.077	6.462
BioMatrix-1.7B-SFT	0.9980	0.9990	0.8377	1.950	23.71	—	—
BioMatrix-4B-SFT	**0.9994**	0.9988	0.7949	1.950	21.22	—	—
MolToken-0.55B	0.9766	0.9992	0.8402	0.5963	**38.28**	2.692	2.231
MolToken-1.44B	0.9855	0.9992	0.8206	0.5850	37.69	3.154	3.077

Limitation of class-restricted pretraining. Within the evaluated range, scaling single-domain pretraining cannot compensate for restricted molecular-domain coverage, because the joint loss of single-domain models doesn't scale with compute. In most panels of Figure 6, the curves for larger compute budgets do not lie consistently below those for smaller budgets. By contrast, under joint pretraining on five domains, the IsoFLOP curves in Figure 4(a) shift downward as compute grows.

4.2.5 Downstream Evaluation

We evaluate MolToken through supervised fine-tuning using its shared molecular representation, vocabulary, and pretrained backbone. Table 2 presents representative comparisons with VAE (Polykovskiy et al., 2020), DiGress (Vignac et al., 2023), and DisCo-GT (Xu et al., 2024) for unconditional small-molecule generation on MOSES. We compare with RefineGNN (Jin et al., 2022) for antibody CDR-H3 completion on RAbD (Adolf-Bryfogle et al., 2018), and with TCRT5 and TCRT5 FT (Karthikeyan et al., 2025) for pMHC-conditioned CDR3β generation on TCR External-13. BioMatrix (Pei et al., 2026) is also included on MOSES and RAbD. Overall, MolToken achieves competitive downstream performance across molecular generation and sequence design. Experimental configurations, adaptation protocols, data-overlap analyses, and complete results for all downstream tasks are provided in Appendix F.

5 Conclusion

We introduced MolToken, a molecular foundation model that unifies small molecules and biopolymers through atom-level graphs, strictly reversible FDDFS serialization, and shared BPE tokenization. Scaling laws fitted at small compute budgets predict the optimal model size and loss, and we verify the predicted loss at 37.5 times the largest fitting budget. Cross-domain analyses reveal domain-dependent benefits and trade-offs of joint training and suggest that, within the evaluated range, scaling single-domain pretraining cannot substitute for broader molecular-domain coverage. We also achieve competitive performance on downstream tasks spanning multiple molecular domains. Future work will extend MolToken to accept natural-language conditions within the same framework. We further plan to improve the parallel efficiency of BPE tokenization, scale up pretraining, and broaden downstream evaluation.

AI USE STATEMENT

AI tools assisted manuscript organization, drafting, and language editing. The authors retain full responsibility for the accuracy and integrity of this work.

ETHICS STATEMENT

Applications of MolToken should include safeguards against chemical and biological misuse. Generated molecules require independent assessment of safety and suitability before experimental or practical use.

REFERENCES

Jared Adolf-Bryfogle, Oleks Kalyuzhniy, Michael Kubitz, Brian D. Weitzner, Xiaozhen Hu, Yumiko Adachi, William R. Schief, and Roland L. Dunbrack, Jr. RosettaAntibodyDesign (RAbD): A general framework for computational antibody design. *PLOS Computational Biology*, 14(4): e1006112, 2018. doi: 10.1371/journal.pcbi.1006112. URL https://doi.org/10.1371/journal.pcbi.1006112.

Garyk Brixi, Matthew G Durrant, Jerome Ku, Mohsen Naghipourfar, Michael Poli, Gwanggyu Sun, Greg Brockman, Daniel Chang, Alison Fanton, Gabriel A Gonzalez, et al. Genome modelling and design across all domains of life with evo 2. *Nature*, 652(8112):1349–1361, 2026.

Nathan Brown, Marco Fiscato, Marwin H. S. Segler, and Alain C. Vaucher. GuacaMol: Benchmarking models for de novo molecular design. *Journal of Chemical Information and Modeling*, 59(3):1096–1108, March 2019. doi: 10.1021/acs.jcim.8b00839. URL https://doi.org/10.1021/acs.jcim.8b00839.

Runjin Chen, Tong Zhao, Ajay Jaiswal, Neil Shah, and Zhangyang Wang. Llaga: Large language and graph assistant. *arXiv preprint arXiv:2402.08170*, 2024.

Xiaohui Chen, Yinkai Wang, Jiaxing He, Yuanqi Du, Soha Hassoun, Xiaolin Xu, and Li-Ping Liu. Graph generative pre-trained transformer. *arXiv preprint arXiv:2501.01073*, 2025.

Xingyi Cheng, Bo Chen, Pan Li, Jing Gong, Jie Tang, and Le Song. Training compute-optimal protein language models. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 69386–69418. Curran Associates, Inc., 2024. doi: 10.52202/079017-2216. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/8066ae1446b2bbccb5159587cc3b3bcc-Paper-Conference.pdf.

Lucas Ferreira DaSilva, Simon Senan, Judith F. Kribelbauer-Swietek, Zain Munir Patel, Lithin Karmel Louis, Aniketh Janardhan Reddy, Sameer Gabbita, Jonathan D. Rosen, Zach Nussbaum, César Miguel Valdez Córdova, Aaron Wenteler, Noah Weber, Tin M. Tunjic, Martino Mansoldo, Talha Ahmad Khan, Gue-Ho Hwang, Vincent Gardeux, David T. Humphreys, Cameron Smith, Matei Bejan, Peter Bromley, Will Connell, Bart Deplancke, Michael I. Love, Emily S. Wong, Wouter Meuleman, and Luca Pinello. Designing synthetic regulatory elements using the generative AI framework DNA-Diffusion. *Nature Genetics*, 58(1):180–194, January 2026. doi: 10.1038/s41588-025-02441-6. URL https://doi.org/10.1038/s41588-025-02441-6.

Philip Gage. A new algorithm for data compression. *C Users Journal*, 12(2):23–38, 1994.

Zeyuan Guo, Enmao Diao, Cheng Yang, and Chuan Shi. Graph tokenization for bridging graphs and transformers. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=jCctxI1BGF.

Thomas Hayes, Roshan Rao, Halil Akin, Nicholas J. Sofroniew, Deniz Oktay, Zeming Lin, Robert Verkuil, Vincent Q. Tran, Jonathan Deaton, Marius Wiggert, Rohil Badkundri, Irhum Shafkat, Jun Gong, Alexander Derry, Raul S. Molina, Neil Thomas, Yousuf A. Khan, Chetan Mishra,

Carolyn Kim, Liam J. Bartie, Matthew Nemeth, Patrick D. Hsu, Tom Sercu, Salvatore Candido, and Alexander Rives. Simulating 500 million years of evolution with a language model. *Science*, 387(6736):850–858, 2025. doi: 10.1126/science.ads0018. URL https://www.science.org/doi/abs/10.1126/science.ads0018.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. An empirical analysis of compute-optimal large language model training. *Advances in neural information processing systems*, 35:30016–30030, 2022.

Xiaohong Ji, Zhen Wang, Zhifeng Gao, Hang Zheng, Linfeng Zhang, Guolin Ke, et al. Exploring molecular pretraining model at scale. *Advances in Neural Information Processing Systems*, 37: 46956–46978, 2024.

Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction tree variational autoencoder for molecular graph generation. In *International conference on machine learning*, pp. 2323–2332. PMLR, 2018.

Wengong Jin, Jeremy Wohlwend, Regina Barzilay, and Tommi Jaakkola. Iterative refinement graph neural network for antibody sequence-structure co-design. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=LI2bhrE_2A.

Bowen Jing, Stephan Eismann, Patricia Suriana, Raphael J. L. Townshend, and Ron Dror. Learning from protein structure with geometric vector perceptrons, 2021. URL https://arxiv.org/abs/2009.01411.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Dhuvarakesh Karthikeyan, Sarah N. Bennett, Amy G. Reynolds, Benjamin G. Vincent, and Alex Rubinsteyn. Conditional generation of real antigen-specific T cell receptor sequences. *Nature Machine Intelligence*, 7(9):1494–1509, 2025. doi: 10.1038/s42256-025-01096-6. URL https://doi.org/10.1038/s42256-025-01096-6.

Mario Krenn, Florian Häse, AkshatKumar Nigam, Pascal Friederich, and Alan Aspuru-Guzik. Self-referencing embedded strings (selfies): A 100% robust molecular string representation. *Machine Learning: Science and Technology*, 1(4):045024, 2020.

Jiahan Li, Chaoran Cheng, Zuofan Wu, Ruihan Guo, Shitong Luo, Zhizhou Ren, Jian Peng, and Jianzhu Ma. Full-atom peptide design based on multi-modal flow matching. *arXiv preprint arXiv:2406.00735*, 2024.

Mingyang Li, Yurou Liu, Jieping Ye, Bing Su, Ji-Rong Wen, and Zheng Wang. Speaking the language of science: Toward a general-purpose generative foundation model for the natural sciences. *arXiv preprint arXiv:2606.16905*, 2026.

Shengchao Liu, Hanchen Wang, Weiyang Liu, Joan Lasenby, Hongyu Guo, and Jian Tang. Pre-training molecular graph representation with 3d geometry. *arXiv preprint arXiv:2110.07728*, 2021.

Niklas Muennighoff, Alexander M. Rush, Boaz Barak, Teven Le Scao, Aleksandra Piktus, Nouamane Tazi, Sampo Pyysalo, Thomas Wolf, and Colin Raffel. Scaling data-constrained language models. *Journal of Machine Learning Research*, 26(53):1–66, 2025. URL https://jmlr.org/papers/v26/24-1000.html.

Eric Nguyen, Michael Poli, Marjan Faizi, Armin Thomas, Michael Wornow, Callum Birch-Sykes, Stefano Massaroli, Aman Patel, Clayton Rabideau, Yoshua Bengio, et al. Hyenadna: Long-range genomic sequence modeling at single nucleotide resolution. *Advances in neural information processing systems*, 36:43177–43201, 2023.

Qizhi Pei, Zhimeng Zhou, Yi Duan, Yiyang Zhao, Wei Li, Han Guo, Liang He, Chengping Li, Chang-Yu Hsieh, Conghui He, et al. Biomatrix: Towards a comprehensive biological foundation model spanning the modality matrix of sequences, structures, and language. *arXiv preprint arXiv:2606.22138*, 2026.

Xingang Peng, Ruihan Guo, Fenglin Guo, Ziyi Wang, Jiayu Sun, Jiaqi Guan, Yinjun Jia, Yan Xu, Yanwen Huang, Muhan Zhang, et al. Unified modeling of 3d molecular generation via atomic interactions with pocketxmol. *Cell*, 189(7):1904–1922, 2026.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 701–710, 2014.

Daniil Polykovskiy, Alexander Zhebrak, Benjamin Sanchez-Lengeling, Sergey Golovanov, Oktai Tatanov, Stanislav Belyaev, Rauf Kurbanov, Aleksey Artamonov, Vladimir Aladinskiy, Mark Veselov, Artur Kadurin, Simon Johansson, Hongming Chen, Sergey Nikolenko, Alán Aspuru-Guzik, and Alex Zhavoronkov. Molecular sets (MOSES): A benchmarking platform for molecular generation models. *Frontiers in Pharmacology*, 11:565644, 2020. doi: 10.3389/fphar.2020.565644. URL https://doi.org/10.3389/fphar.2020.565644.

Shunsuke Sumi, Michiaki Hamada, and Hirohide Saito. Deep generative design of RNA family sequences. *Nature Methods*, 21(3):435–443, March 2024. doi: 10.1038/s41592-023-02148-8. URL https://doi.org/10.1038/s41592-023-02148-8.

György Turán. On the succinct representation of graphs. *Discrete Applied Mathematics*, 8(3):289–294, 1984. ISSN 0166-218X. doi: https://doi.org/10.1016/0166-218X(84)90126-4. URL https://www.sciencedirect.com/science/article/pii/0166218X84901264.

Clement Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. DiGress: Discrete denoising diffusion for graph generation. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=UaAD-Nu86WX.

Xinyou Wang, Zaixiang Zheng, Fei Ye, Dongyu Xue, Shujian Huang, and Quanquan Gu. Diffusion language models are versatile protein learners. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 52309–52333. PMLR, 2024. URL https://proceedings.mlr.press/v235/wang24ct.html.

David Weininger. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of chemical information and computer sciences*, 28(1):31–36, 1988.

Zhe Xu, Ruizhong Qiu, Yuzhong Chen, Huiyuan Chen, Xiran Fan, Menghai Pan, Zhichen Zeng, Mahashweta Das, and Hanghang Tong. Discrete-state continuous-time diffusion for graph generation. In *Advances in Neural Information Processing Systems*, volume 37, pp. 79704–79740, 2024. doi: 10.52202/079017-2531. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/91813e5ddd9658b99be4c532e274b49c-Abstract-Conference.html.

Mengmei Zhang, Mingwei Sun, Peng Wang, Shen Fan, Yanhu Mo, Xiaoxiao Xu, Hong Liu, Cheng Yang, and Chuan Shi. Graphtranslator: Aligning graph model to large language model for open-ended tasks. In *Proceedings of the ACM Web Conference 2024*, pp. 1003–1014, 2024.

Appendix

A Data and Pretraining Details

All pretraining experiments use the same heavy-atom graph representation, FDDFS serialization rules, and BPE vocabulary. Model size, training volume, and domain composition vary across experimental settings.

A.1 Pretraining corpus

Composition. The corpus comprises small molecules, peptides, proteins, DNA, and RNA. Normalization yields 440,160,253 records; cleaning and deduplication retain 324,754,740 records, of which 323,402,911 can be converted to the supported heavy-atom graphs. Table A.1 lists their sources and counts. For long biological sequences, a record is a sequence window rather than an entire macromolecule.

Table A.1: Pretraining sources and numbers of successfully constructed graphs.

Domain	Sources	Records
Small molecule	ChEMBL 36; PubChem	92,452,053
Peptide	PepBenchmark; SwissProt-derived sequences	844,090
Protein	UniRef50; SwissProt	108,336,213
DNA	RefSeq bacterial, archaeal, fungal, and viral sequences	23,561,409
RNA	RNAcentral; Rfam	98,209,146
Total		323,402,911

Cleaning and windowing. Small molecules are converted to canonical isomeric SMILES with RDKit. We retain single-component molecules with 3–128 heavy atoms and supported elements. Biological sequences are uppercased, and T is converted to U in RNA. Protein windows contain 64, 128, or 256 residues, with half-window strides and at most eight windows per source sequence. Peptides contain 5–80 residues, with at most one window per source sequence. DNA windows contain 128 or 256 nucleotides with stride 256. RNA windows contain 64, 128, or 256 nucleotides, with half-window strides and at most 16 windows per source sequence. No single base may constitute more than 0.8 of a DNA or RNA window. The supplementary Rfam sources comprise RF00001, RF00005, RF00017, and RF00026.

Exact normalized sequences are deduplicated within each domain, including duplicates across sources in that domain. Identical letter strings in different domains are not treated as identical inputs: for example, A denotes different chemical units in proteins and DNA. This procedure performs exact sequence deduplication.

Tokenizer construction and model-training splits. The frequency guide and BPE vocabulary are constructed from the five-domain unlabeled corpus and then frozen. FDDFS frequency statistics cover all 323,402,911 graphs, including records originally tagged as training, validation, and test during preprocessing: 320,192,388, 1,606,385, and 1,604,138 records, respectively.

Joint pretraining holds out one sequence in every 1,000 according to sequence index and uses the remaining records for gradient updates. The complete corpus contains 9,902,656,060 BPE content tokens, or 10,226,058,971 tokens after appending EOS to each record. The training portion contains 10,215,832,909 tokens before length truncation. In compute accounting, D counts cumulative training input tokens, including EOS and repeated presentations but excluding PAD. Single-domain training and cross-domain evaluation use separately fixed domain splits, described in Appendix E.4.

A.2 Architecture and optimization

Compute resources. Model training in this study used a total of 24 NVIDIA H200 GPUs.

Architecture. All models are randomly initialized decoder-only Transformers with causal self-attention, grouped-query attention, RMSNorm, and rotary positional embeddings. Input embeddings and output projections share weights. The pretraining vocabulary contains 17,207 entries. Table A.2 lists the 20 architectures used in the joint-pretraining scaling experiments and the two downstream backbones. Parameter count N includes embeddings, with shared weights counted once.

Table A.2: Twenty joint-pretraining scaling architectures and the two downstream backbones (last two rows). Attention head dimension is 128 throughout.

Parameters (M)	d_{model}	d_{FFN}	Layers	Q heads	KV heads
5.979136	256	768	2	2	1
7.553024	256	768	4	2	1
9.126912	256	768	6	2	1
10.700800	256	768	8	2	1
13.848576	256	768	12	2	1
16.996352	256	768	16	2	1
22.097664	384	1152	9	3	1
25.539840	384	1152	11	3	1
27.260928	384	1152	12	3	1
30.703104	384	1152	14	3	1
41.029632	384	1152	20	3	1
46.571520	512	1536	12	4	2
52.865024	512	1536	14	4	2
59.158528	512	1536	16	4	2
65.452032	512	1536	18	4	2
71.745536	512	1536	20	4	2
78.039040	512	1536	22	4	2
98.168832	768	2304	12	6	3
300.773376	1024	3072	18	16	8
458.080256	1024	3072	28	16	8
552.473088	1024	3072	34	16	8
1444.649984	2048	6144	28	16	8

Both downstream backbones also use RoPE with base 10^6, attention dropout 0, and RMSNorm epsilon 10^{-6}, and are randomly initialized.

Table A.3: Optimization settings for joint-pretraining scaling experiments, shared across model sizes.

Setting	Value
Optimizer	AdamW; $(\beta_1, \beta_2) = (0.9, 0.95)$; $\epsilon = 10^{-8}$
Peak learning rate	2×10^{-4}
Schedule	3% linear warmup, then cosine decay to 10% of the peak
Weight decay	0.1
Gradient clipping	Global norm 1.0
Effective batch	6,144 sequences
Maximum context	512 tokens
Precision and seed	BF16; 42

Sequence organization and loss. Each molecular sequence retains at most 511 BPE content tokens, followed by EOS and padding to length 512. Following the notation in equation 4, $\mathbf{t}^{(j)} = (t_1^{(j)}, \ldots, t_{n_j}^{(j)})$ denotes the non-padding token sequence, where n_j includes EOS. Block-diagonal causal attention prevents attention across sample boundaries. Autoregressive supervision covers positions $i = 1, \ldots, n_j$, including EOS, while PAD positions are excluded from the loss. For a batch $\mathcal{B}$ of B sequences, $\mathcal{L}(\theta; \mathcal{B})$ is computed by summing the negative log-likelihood over all supervised positions and dividing by $\sum_{j=1}^{B} n_j$, as defined in equation 4. Natural logarithms give the loss in nats/token. Test loss is computed for a fixed model using this objective, while training loss is estimated from terminal training logs as described in Appendix D.1. Length statistics are provided in Appendix C.3.

Compute budgets. Training FLOPs are estimated by $C = 6ND$, following Hoffmann et al. (2022). Training stops at a discrete optimizer step when the input-token budget is reached. Fits use the target compute budget, with $D^* = C/(6N^*)$. This approximation omits some attention computation and implementation overhead.

Larger pretrained models. The 0.55B and 1.44B models contain 552,473,088 and 1,444,649,984 parameters, respectively. Both consume 39,571,656,240 training input tokens. The former is configured from the compute-optimal relation and used for the loss-extrapolation test in Figure 4(c); the

latter supports the downstream capacity comparison. Their final training losses are 4.034795 and 3.897028. Both training volumes remain below the four-pass cap of approximately 40.863B tokens.

A.3 Training–evaluation settings and domain shares

The four settings and their loss sources are described in Appendix D.1. The domain shares r_k used in Figure 5 are pooled from logged joint-training input-token counts, including EOS and excluding PAD (Table A.4). Test-mixture weights use effective prediction positions.

Table A.4: Domain shares of joint-pretraining input tokens.

Domain	Small molecule	Peptide	Protein	DNA	RNA
r_k	9.9410%	0.1685%	54.1374%	13.2561%	22.4970%

For domain k, we map a joint-training configuration as

$$(C, N, D, L) \longmapsto (r_k C, N, r_k D, L). \tag{A.1}$$

B Graph Construction and FDDFS Subroutines

The construction map Γ converts a molecular input into an attributed graph $G = (V, E, x, z)$, which FDDFS serializes into base symbols. Encoding and decoding preserve the stored node attributes, edge attributes, and connectivity; Figure 2(a,b) illustrates the pipeline. Algorithm 1 in the main text provides the overall procedure and the first traversal, First. Here we specify deterministic ordering, expand the omitted Second routine, and give its inverse decoder.

B.1 Unified heavy-atom graphs

All five domains share the node-attribute space $\mathcal{X}$ and edge-attribute space $\mathcal{Z}$, rather than using domain-specific atom alphabets. A complete attribute tuple defines one base symbol (Table B.1). Graphs are simple and undirected, potentially with multiple connected components. Hydrogen atoms are not separate nodes; three-dimensional coordinates and noncovalent contacts are not represented.

Table B.1: Shared graph attributes. Tuple order is fixed.

Object	Attributes
Node $x(v)$	Atomic number, formal charge, aromaticity, chirality tag, degree, total valence
Edge $z(e)$	Bond type, aromaticity, stereochemical tag

Small molecules. RDKit parses SMILES into heavy atoms and bonds, from which the attributes in Table B.1 are extracted. Multiple retained connected components are separated by a component delimiter in one sequence.

Peptides and proteins. The 20 standard L-amino acids are expanded using RDKit-derived residue templates and assembled directly as graphs, without a full-sequence SMILES intermediate. Consecutive residues are linked by a peptide bond between the preceding carbonyl carbon and the following backbone nitrogen. The first residue uses a free-N-terminal template, while internal residues use chain-internal templates. Monomer terminal oxygen atoms are removed during assembly, and one OXT is added at the chain terminus. Thus, each residue graph G_{a_i} reflects its chain position; free-monomer terminal groups are not retained at every residue.

DNA and RNA. Nucleotide templates distinguish monomers, first, internal, and last positions. Consecutive units are linked through O3′ and P. DNA uses ACGT and deoxyribose; RNA uses ACGU and ribose. No additional phosphate caps are added at the 5′ or 3′ ends, so a standard linear chain of length l has $l - 1$ linking phosphorus atoms. Windows are constructed independently, with no bonds between different windows.

Table B.2: Short-chain construction examples. Peptides and proteins share amino-acid templates.

Domain	Sequence	Heavy atoms	Undirected bonds
Peptide/protein	AG	10	9
DNA	AC	37	41
RNA	AC	39	43

B.2 Frequency guide and deterministic ordering

The frozen guide $\mathcal{F} = (f_N, f_E, f_T)$ counts node attributes, edge attributes, and directed atom–bond–atom patterns. Each undirected edge contributes a pattern in both directions. Counts are pooled across domains, rather than normalized separately within each domain.

Let $\ell(x(v))$ be a fixed atom label derived from the attributes, and let $\iota(v)$ be the input vertex index. A component root maximizes the lexicographic key

$$K_{\mathrm{root}}(v) = \big(f_N(x(v)), \deg(v), \ell(x(v)), -\iota(v)\big). \tag{B.1}$$

Components are ordered by decreasing root key. The key for a neighbor u of v is

$$\begin{aligned} K_v(u) = \big(&f_T(x(v), z(\{v,u\}), x(u)),\ f_E(z(\{v,u\})),\\ &f_N(x(u)),\ \ell(x(u)),\ -\iota(u)\big). \end{aligned} \tag{B.2}$$

Local-pattern frequency is the primary criterion; the remaining terms deterministically break ties.

Branches and the main continuation. The neighbor visiting order is

$$\pi_v = \mathrm{sort}^{\downarrow}_{K_v} N(v). \tag{B.3}$$

The first FDFS scans neighbors in this order and skips visited vertices. The second FDFS preserves the resulting tree-child order, enclosing all but the last child subtree in parentheses and treating the last child as an unparenthesized continuation.

B.3 Second FDFS: ring events and sequence emission

The default edge attribute is $z_0 = \arg\max_z f_E(z)$, which is $(1, 0, 0)$ for an ordinary single bond in these experiments. Nodes are always emitted. Edges with attribute z_0 omit their edge symbol; all others emit the complete edge tuple. Table B.3 specifies readable symbols; the model processes the corresponding integer IDs.

Table B.3: FDDFS grammar. Ring indices are occupied only while open.

Symbol	Meaning
$\mathrm{Atom}[x]$	Create a vertex with attribute x
$\mathrm{Bond}[z]$	Set a nondefault attribute for the next tree edge or ring opening
(,)	Enter a branch; return to its parent
R_j	Open or close non-tree edge index j
SEP	End a component without adding a cross-component edge

Complete encoder. Algorithm B.1 combines the first traversal of Algorithm 1 with Second in Algorithm B.2. It uses the keys K_{root} and K_v defined in the preceding subsection. Let $\rho = (r_1, \ldots, r_c)$ be the ordered component roots, where $c = |\mathcal{Q}(G)|$. The visited set A, tree-edge set E_T, discovery map τ, and ordered child lists $\chi = (\chi_v)_{v \in V}$ are shared by all calls to First within one invocation of Encode; the neighbor lists $(\pi_v)_{v \in V}$ remain fixed. The map τ is initially empty and records one-based discovery indices. We use $\langle\rangle$ for an empty sequence and $\|$ for concatenation. Iteration over a sequence follows its stored order. The main-text algorithm uses a compact interface for the second traversal; the complete encoder below explicitly records the discovery order τ and ordered tree-child lists χ and passes them, together with $\mathcal{F}$ and z_0, to Second.

Non-tree edges. Each non-tree edge opens at its earlier-discovered endpoint and closes at its later-discovered endpoint. Opening allocates the smallest free index, emits any nondefault edge attribute, and emits R_j. Closing emits the same R_j and frees that index. At a vertex, all openings precede all closings; each group uses frequency-priority order. Algorithm B.2 constructs opening and closing

Algorithm B.1 ENCODE: FDDFS graph serialization

Require: Supported graph $G = (V, E, x, z)$; fixed guide $\mathcal{F} = (f_N, f_E, f_T)$
Ensure: FDDFS base-symbol sequence $S_G = \mathrm{Enc}(G)$

function ENCODE($G, \mathcal{F}$)
  $z_0 \leftarrow \arg\max_{z \in \mathcal{Z}} f_E(z)$
  $A \leftarrow \varnothing$; $E_T \leftarrow \varnothing$; $\tau \leftarrow \varnothing$
  for all $v \in V$ **do**
    $\pi_v \leftarrow \mathrm{sort}^{\downarrow}_{K_v} N(v)$
    $\chi_v \leftarrow \langle\rangle$
  end for
  for all $Q \in \mathcal{Q}(G)$ **do**
    $r_Q \leftarrow \arg\max_{v \in Q} K_{\mathrm{root}}(v)$
  end for
  $\rho \leftarrow \mathrm{sort}^{\downarrow}_{K_{\mathrm{root}}} \{r_Q : Q \in \mathcal{Q}(G)\}$
  for $i = 1, \ldots, |\rho|$ **do**
    FIRST(r_i)
  end for
  $S_G \leftarrow$ SECOND($G, E_T, \rho, \tau, \chi, \mathcal{F}, z_0$)
  return S_G
end function

procedure FIRST(v)
  $A \leftarrow A \cup \{v\}$; $\tau(v) \leftarrow |A|$
  for $u \in \pi_v$ **do**
    if $u \notin A$ **then**
      $E_T \leftarrow E_T \cup \{\{v, u\}\}$
      $\chi_v \leftarrow \chi_v \,\|\, \langle u \rangle$
      FIRST(u)
    end if
  end for
end procedure

Algorithm B.2 SECOND: construct ring events and serialize components

Require: $G = (V, E, x, z)$, E_T, ρ, τ, χ, $\mathcal{F}$, z_0
Ensure: FDDFS base-symbol sequence S_G

function SECOND($G, E_T, \rho, \tau, \chi, \mathcal{F}, z_0$)
  $O_v \leftarrow \varnothing$, $C_v \leftarrow \varnothing$ for all $v \in V$
  for all $e = \{u, w\} \in E \setminus E_T$ **do**
    if $\tau(u) > \tau(w)$ **then**
      $(u, w) \leftarrow (w, u)$
    end if
    $O_u \leftarrow O_u \cup \{e\}$; $C_w \leftarrow C_w \cup \{e\}$
  end for
  for all $v \in V$ **do**
    $O_v \leftarrow \mathrm{sort}^{\downarrow}_{e \mapsto K_v(\mathrm{other}(e,v))} O_v$
    $C_v \leftarrow \mathrm{sort}^{\downarrow}_{e \mapsto K_v(\mathrm{other}(e,v))} C_v$
  end for
  $S \leftarrow \langle\rangle$; $\eta \leftarrow \varnothing$
  for $i = 1, \ldots, |\rho|$ **do**
    if $i > 1$ **then**
      $S \leftarrow S \| \langle \mathrm{SEP} \rangle$
    end if
    EMIT(r_i) ▷ Algorithm B.3
  end for
  return S
end function

lists O_v, C_v, and Algorithm B.3 emits the ordered subtrees. Here $\mathrm{other}(e, v)$ is the other endpoint of e, and $\|$ denotes concatenation. The active map $\eta : e \mapsto j$, output S, graph, and ordered forest are shared by recursive calls. Deleting $\eta[e]$ frees j. Excluding a non-tree edge from the tree traversal does not discard it: its attributes and endpoints are retained by paired ring markers.

Algorithm B.3 EMIT: emit a vertex, ring markers, and ordered subtrees

Require: Current vertex v; shared state from Algorithm B.2
Ensure: Append the subtree rooted at v to S and update η

procedure EMIT(v)
  $S \leftarrow S \| \langle \mathrm{Atom}[x(v)] \rangle$
  for $e \in O_v$ in list order **do**
    $j \leftarrow \min\big(\mathbb{N}_{>0} \setminus \mathrm{range}(\eta)\big)$ ▷ Smallest free index
    $\eta[e] \leftarrow j$; EMITBOND($z(e)$)
    $S \leftarrow S \| \langle R_j \rangle$
  end for
  for $e \in C_v$ in list order **do**
    $j \leftarrow \eta[e]$; $S \leftarrow S \| \langle R_j \rangle$; **delete** $\eta[e]$
  end for
  $d \leftarrow |\chi_v|$
  for $i = 1, \ldots, d$ **do**
    $u \leftarrow \chi_v[i]$
    if $i < d$ **then**
      $S \leftarrow S \| \langle (\rangle$
    end if
    EMITBOND($z(\{v, u\})$); EMIT(u)
    if $i < d$ **then**
      $S \leftarrow S \| \langle) \rangle$
    end if
  end for
end procedure

procedure EMITBOND(z)
  if $z \neq z_0$ **then**
    $S \leftarrow S \| \langle \mathrm{Bond}[z] \rangle$
  end if
end procedure

Since all openings precede closings at a vertex, an index closed at that vertex can be reused only at a later vertex. Traversing children in χ_v order preserves the discovery order τ, so each non-tree edge opens before it closes.

Example. Consider a tree path $A - B - C$ with a branch D at B and a non-tree edge $A - C$. With default attributes on all edges, one encoding is

$$\mathrm{Atom}[x_A]\ R_1\ \mathrm{Atom}[x_B]\ (\mathrm{Atom}[x_D])\ \mathrm{Atom}[x_C]\ R_1.$$

The first R_1 records A, and the second restores edge $A - C$ at C. Parentheses attach D to B. A nondefault attribute on $A - C$ is emitted immediately before its opening marker.

B.4 DECODING AND REVERSIBILITY

The decoder maintains the current vertex, a branch stack, and an active ring table. A node symbol creates a vertex and connects it to the current vertex using the pending edge attribute, defaulting to z_0. Parentheses save and restore the current vertex. A ring marker first stores an endpoint and edge attribute, then closes that edge on its next occurrence and deletes the stored entry. A component separator clears the current parent.

Algorithm B.4 gives the inverse for a complete valid encoding. NEWVERTEX creates an attributed vertex and returns its index; ADDEDGE adds an attributed undirected edge. Variable p stores the pending edge attribute, B is the branch stack, and H stores open ring endpoints.

Proposition B.1 (Strict reversibility of FDDFS)**.** *Fix the symbol interpretation, default edge attribute, and traversal rules. For any finite simple undirected graph G whose attributes and active ring indices lie within the supported vocabulary, the complete encoding satisfies*

$$\mathrm{Dec}(\mathrm{Enc}(G)) \cong G. \tag{B.4}$$

The isomorphism preserves node attributes, edge attributes, and adjacency.

Proof. The spanning forest visits every vertex once, and the second traversal emits its complete attribute once. This induces a bijection between original vertices and decoded vertices created in

Algorithm B.4 DECODE: reconstruct an attributed graph from FDDFS

Require: Complete valid sequence S_G; default edge attribute z_0
Ensure: Reconstructed graph $G' \cong G$

function DECODE(S_G, z_0)
  $G' \leftarrow (\varnothing, \varnothing, \varnothing, \varnothing)$; $v \leftarrow \perp$; $p \leftarrow z_0$
  $B \leftarrow \langle\rangle$; $H \leftarrow \varnothing$
  for $t \in S_G$ from left to right **do**
    if $t = \mathrm{Atom}[x]$ **then**
      $u \leftarrow$ NEWVERTEX(G', x)
      if $v \neq \perp$ **then**
        ADDEDGE(G', v, u, p)
      end if
      $v \leftarrow u$; $p \leftarrow z_0$
    else if $t = \mathrm{Bond}[z]$ **then**
      $p \leftarrow z$
    else if $t =$ (**then**
      PUSH(B, v)
    else if $t =$) **then**
      $v \leftarrow$ POP(B)
    else if $t = R_j$ **then**
      if $j \notin \mathrm{dom}(H)$ **then**
        $H[j] \leftarrow (v, p)$
      else
        $(u, z) \leftarrow H[j]$; ADDEDGE(G', u, v, z)
        delete $H[j]$
      end if
      $p \leftarrow z_0$
    else if $t = \mathrm{SEP}$ **then**
      $v \leftarrow \perp$; $p \leftarrow z_0$
    end if
  end for
  return G'
end function

emission order. Each tree edge is recovered from a parent–child relation, with its attribute uniquely given by an explicit symbol or the fixed default. The branch stack restores the correct parent for every subtree. Each non-tree edge opens at one endpoint and closes at the other; its index is not reused while active, uniquely recovering its endpoints and attribute. Component delimiters preserve components without adding cross-component edges. Under the induced vertex bijection, all attributes and edges are therefore identical. □

Tie-breaking uses input vertex indices, so isomorphic graphs may produce different, equally decodable sequences. The guarantee is reconstruction up to attributed-graph isomorphism, not a unique canonical string. Syntactically invalid generated sequences are rejected.

Length and complexity. For n vertices, m undirected edges, c components, b_{br} parenthesized branches, and m_{nd} nondefault edges,

$$|S_G| = n + 2(m - n + c) + 2b_{\mathrm{br}} + m_{\mathrm{nd}} + (c - 1). \tag{B.5}$$

The terms count node symbols, ring markers, parentheses, nondefault edge symbols, and component delimiters. With fixed frequency lookups, adjacency sorting costs $O(\sum_v \deg(v) \log \deg(v))$. Including component and ring-event ordering, the overall time complexity is

$$O((n + m) \log(n + m)). \tag{B.6}$$

The two traversals and decoding scan are linear, and storage is $O(n + m)$. The frozen vocabulary supports 249 reusable ring indices. Inputs outside the supported attributes or ring capacity are not encoded. Reversibility applies to complete sequences, not arbitrary truncated prefixes.

C TOKENIZATION AND REPRESENTATION ANALYSIS

BPE merges adjacent FDDFS symbols into variable-length tokens. All molecular domains share one vocabulary and one ordered set of merge rules. Merging changes sequence segmentation but not the expanded base-symbol sequence.

C.1 VOCABULARY AND MERGE RULES

We use the notation of Algorithm 1: $\mathcal{W}$ is the vocabulary and $\mathcal{R}$ is the ordered merge list. The initial alphabet contains atom attributes, nondefault bond attributes, and branch, ring, and component-boundary symbols. Each iteration counts adjacent symbol pairs, selects a most frequent pair (a, b), introduces a new symbol c with rule $(a, b) \mapsto c$, and replaces nonoverlapping occurrences from left to right. Ties are broken lexicographically by integer symbol pairs. Neither counting nor replacement crosses a molecular sample boundary.

Learning first applies a 70-merge prefix selected by the subset-stability experiment in Appendix C.2, then continues on the complete five-domain corpus. The final codebook is truncated by a pair-frequency threshold of 80,000, retaining 16,621 merges. The last retained count is 80,001, and the next recorded candidate has count 79,993. Table C.1 gives the frozen vocabulary. These frequencies count pairs when selected during learning, not token usage after final segmentation.

Table C.1: FDDFS/BPE vocabulary. Model-special symbols are distinct from graph-grammar symbols.

Symbol category	Count
Atom-attribute tuples	326
Nondefault edge-attribute tuples	5
Reusable ring markers	249
Branch parentheses and component delimiter	3
BPE merge symbols	16,621
Content vocabulary	17,204
Model-special symbols: PAD, BOS, EOS	3
Total model vocabulary	17,207

The five nondefault edge attributes are $(2, 0, 0)$, $(2, 0, 2)$, $(2, 0, 3)$, $(3, 0, 0)$, and $(4, 1, 0)$, covering double bonds with different stereochemical tags, triple bonds, and aromatic bonds. Ordinary single bonds use the omitted default attribute. Thus, the number of edge-attribute symbols is not the number of chemical bond orders.

Each learned symbol stores its two constituents. Recursively expanding the ordered rules recovers the FDDFS sequence; for a complete tokenized graph t,

$$\mathrm{Dec}\big(\mathrm{Expand}_{\mathcal{R}}(t)\big) \cong G. \tag{C.1}$$

BPE expansion followed by FDDFS decoding therefore preserves attributed-graph reconstruction.

C.2 MERGE-PREFIX STABILITY AND MEMORY REQUIREMENTS

Motivation. The complete FDDFS corpus contains 710,777,441,079 base tokens. Incremental BPE maintains sequences as doubly linked lists and stores occurrence locations, counts, and candidates for adjacent pairs. A merge updates only affected adjacencies rather than rescanning the corpus. Memory depends on token and occurrence-index counts, not only vocabulary size. In the implementation, linked-list nodes occupy 8 or 16 bytes and occurrence entries occupy 6 or 8 bytes, in addition to hash tables, dynamic-array capacity, and priority queues. Building these structures directly over the uncompressed corpus is therefore memory-intensive.

We first learn a stable high-frequency prefix on a subset, then stream through the full corpus and apply it to individual sequences. This step does not build a corpus-wide occurrence index. The subsequent learner constructs fresh indices and recomputes full-corpus frequencies on the shortened sequences. The prefix changes the initial segmentation for this stage without dropping samples or merging across their boundaries.

Quantitative choice of prefix length. FDDFS token files are shuffled with seed 20260620 to construct nested subsets at approximately 0.1%, 1%, and 10% of the token budget. Each subset undergoes 100 merges with minimum frequency 1, using the same integer IDs and tie-breaking rule. Table C.2 reports subset sizes and recorded memory. Process counts differ across subsets; the reported peak is for the coordinating process (rank 0), not the sum across processes.

Agreement is assessed by comparing the ordered pair (a, b) selected at each iteration, not by requiring raw counts to match across subset sizes. The 1% and 10% subsets agree for 71 consecutive

Table C.2: Merge-prefix experiments. Token counts are in billions; memory is the recorded rank-0 peak.

Target fraction	Tokens (B)	Records	Processes	Peak memory (GB)
0.1%	0.715	342,972	2	14.4
1%	7.109	3,213,065	4	58.9
10%	71.081	32,401,715	8	261.7

merges and first diverge at merge 72. The 0.1% subset agrees with each larger subset for only 21 merges. We select the first 70 rules from the 10% subset, within the common prefix of the two larger subsets. Table C.3 uses one-based iteration numbers.

Table C.3: Agreement in ordered merge choices across subset sizes.

Subset pair	Common prefix	First differing merge
0.1% vs. 1%	21	22
0.1% vs. 10%	21	22
1% vs. 10%	71	72

This verifies iteration-wise consistency of the selected prefix between the 1% and 10% scales and provides empirical support for reuse. It does not guarantee invariant merge order at arbitrary corpus scales. After applying the fixed prefix, each subsequent iteration aggregates integer pair counts over the complete shortened corpus and performs the globally selected nonoverlapping replacement.

Full-corpus precompression. All 323,402,911 records are retained. Applying the 70 rules reduces the token count from 710,777,441,079 to 64,883,465,080, a $10.95\times$ shortening or approximately 90.87% reduction. This measures the sequence volume entering index construction, not a measured proportional reduction in peak memory. The final codebook retains another 16,551 merges, for 16,621 in total, and the final content sequences contain 9,902,656,060 tokens. The remaining vocabulary is learned from the complete corpus, not solely from a subset.

C.3 Compression and length-statistic conventions

Domain-wise compression and length distributions are reported in the main-text Table 1. They use the same complete graph corpus, containing 332,121,946,865 heavy atoms and 365,256,017,512 undirected bonds. Without EOS, PAD, or training-length truncation, compression is

$$\rho = \frac{\sum_i |S_i|}{\sum_i |t_i|}. \tag{C.2}$$

Domain statistics restrict these sums to the corresponding records; they do not average per-sample or per-domain ratios. Appending EOS to each BPE sequence changes the corpus-wide ratio from 71.78 to 69.51. Length quantiles use linear interpolation at position $(n-1)q$ for n records and quantile level q. Counts are displayed in millions or billions, while ratios are computed from unrounded integer totals.

C.4 Structural fragments and cross-domain vocabulary

Figure 3 traces two paths in the frozen codebook. An aromatic C–N fragment merges into an imidazole motif and then a histidine-related fragment. Another branch forms an adenine-related local motif and combines it with sugar–phosphate fragments. A single arrow represents one merge; paired arrows omit intermediate nodes spanning two merges. Gray attachment points indicate fragment boundaries in the reference molecule and are not part of the token.

Token boundaries need not coincide with complete biological building blocks. The illustrated imidazole fragment covers the five-membered ring in the reference histidine, whereas the histidine-related fragment covers 10 of its 11 heavy atoms. The DNA fragment covers parts of two adjacent nucleotides in the reference sequence CA, containing 3 and 16 atoms from the respective units. The adenine-related motif is also partial rather than the complete fused-ring system. Fragment names describe molecular context, not additional functional annotations.

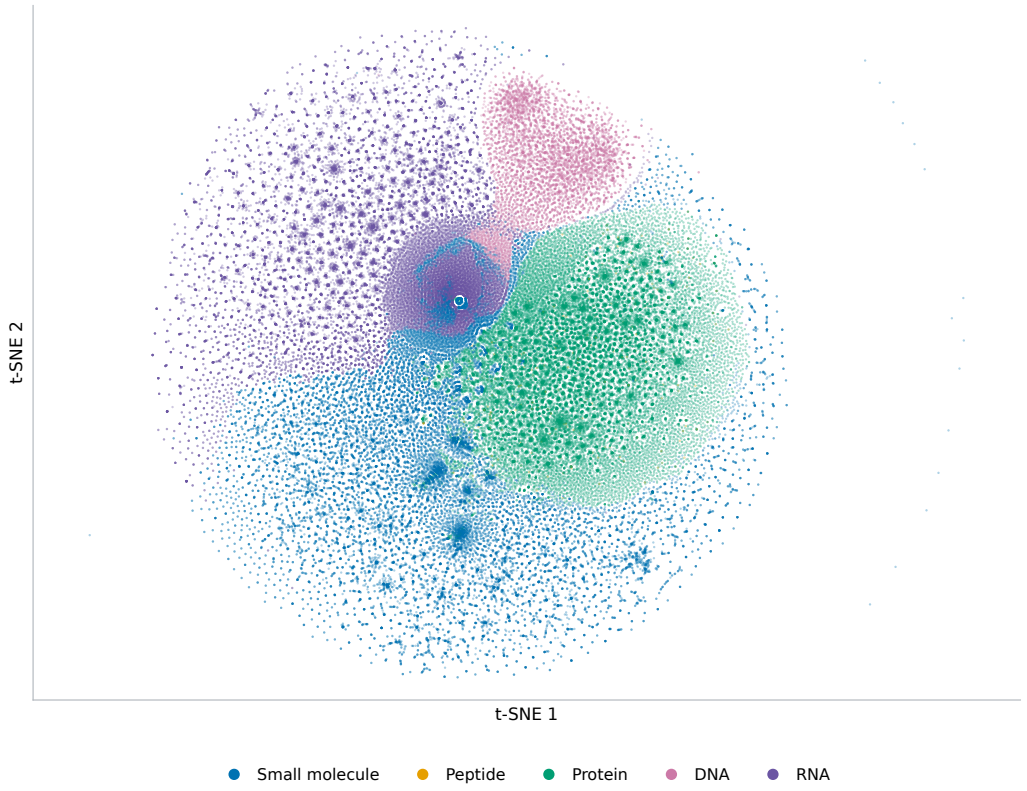


Figure C.1: Token-frequency visualization across five domains. Each point is one sequence, sampled in proportion to domain record counts. The small peptide population reflects its lower record share.

C.5 Two-dimensional visualization of token composition

We sample 100,000 training sequences without replacement in proportion to domain record counts: 28,587 small molecules, 261 peptides, 33,499 proteins, 7,286 DNA sequences, and 30,367 RNA sequences. Each full content sequence is represented by a normalized 17,204-dimensional token-frequency vector, without EOS or TF–IDF weighting.

t-SNE uses Euclidean distance, perplexity 30, seed 42, and 1,500 iterations. PCA initializes the embedding but does not reduce the input feature dimension. Domain labels are used only for coloring. Figure C.1 shows local patterns in token composition under the shared vocabulary.

D Scaling Estimation and Joint-Pretraining Results

For each training–evaluation setting s, we first estimate the optimal model size at fixed compute and then fit $N_s^*(C)$, $D_s^*(C)$, and $L_s^*(C)$ across budgets. Figure 4 presents the joint-pretraining curves and extrapolation results. As in the main text, we use *loss* to denote autoregressive negative log-likelihood.

D.1 Loss Definition and Estimation

Loss objective. All scaling experiments use autoregressive negative log-likelihood, reported in nats/token; equation 4 defines the loss and supervised positions. Many-to-many and many-to-one use terminal training loss; one-to-one and one-to-many use test loss measured on fixed test sets with the budget-end model.

Test splits and mixture weighting. Downstream SFT splits used for early stopping and checkpoint selection remain validation sets. Joint pretraining uses the periodic holdout in Appendix A.1; single-domain and one-to-many experiments use the fixed domain splits in Appendix E.4. Scaling

Table D.1: Fitted parameters of the scaling relations in equation 6.

Domain	Model scale		Data scale		Loss	
	a_s	$\log_{10} K_{N,s}$	b_s	$\log_{10} K_{D,s}$	u_s	v_s
Multi-to-multi ($s =$ `m-to-m`)						
Mixture	0.5430	−2.1959	0.4570	1.4177	−0.1988	8.0359
Single-to-single ($s =$ `1-to-1`)						
Small molecule	0.7526	−5.4861	0.2474	4.7079	−0.4683	10.4670
Peptide	0.8265	−5.9857	0.1735	5.2076	−1.4068	26.8708
Protein	0.4435	−0.5771	0.5565	−0.2011	−0.0899	6.1970
DNA	0.3970	0.3386	0.6030	−1.1168	−0.1306	7.8273
RNA	0.6385	−3.6437	0.3615	2.8655	−0.4260	11.4839
Multi-to-single ($s =$ `m-to-1`)						
Small molecule	0.5650	−2.5077	0.4350	1.7295	−0.4421	10.5098
Peptide	0.4475	−0.5698	0.5525	−0.2083	−0.0780	6.1937
Protein	0.4982	−1.4985	0.5018	0.7204	−0.0755	5.9996
DNA	0.5714	−2.8415	0.4286	2.0634	−0.1123	7.6592
RNA	0.5328	−1.9627	0.4672	1.1846	−0.4508	12.2026

C is measured in FLOPs, N in parameters, and D in training input tokens. For `m-to-1`, C and D refer to the full training mixture. No scaling relation is fitted for `1-to-m`.

runs use budget-end models. The single-domain model-size grids were supplemented using completed runs' test losses to cover both sides of the minima.

For one-to-many evaluation, we pool the five domains' NLL sums and prediction-token counts for each model:

$$L_{\text{all,test}}(\theta) = \frac{\sum_k T_k L_{k,\text{test}}(\theta)}{\sum_k T_k}. \tag{D.1}$$

All evaluated models share this test mixture and denominator; Table E.4 lists T_k.

Online estimation and data repetition. Chinchilla uses smoothed training loss as an unbiased estimate of test loss in the infinite-data regime (Hoffmann et al., 2022). We use terminal training loss to analyze joint-training loss frontiers and report the available training–test pairs below. For data repetition, *Scaling Data-Constrained Language Models* observes that up to four passes over repeated data can achieve similar test loss to training on more unique data at fixed compute (Muennighoff et al., 2025). We define D as cumulative input-token presentations.

Paired training and test measurements. Training logs and complete in-domain test loss are available for 146 single-domain runs. This comparison estimates training loss from logs covering the final 10% of optimizer steps, weighted by effective prediction tokens.

For run i, define $\Delta_i = L_{i,\text{test}} - \widehat{L}_{i,\text{on}}$. With equal weight per run, we report

$$\overline{\Delta} = \frac{1}{n}\sum_i \Delta_i, \qquad \text{MAE}_\Delta = \frac{1}{n}\sum_i |\Delta_i|. \tag{D.2}$$

Across the 146 pairs, the signed mean is -0.000795 nats/token and the mean absolute difference is 0.003004 nats/token. Table D.2 lists domain-level results, paired-run coverage, and training-pass ranges.

D.2 Experimental budgets and within-budget optima

Joint pretraining uses the final analysis set underlying the main-text figures, the architectures in Table A.2, and ten budgets:

$$C \in \{0.35, 0.5, 0.7, 1.0, 1.3, 1.6, 2.0, 2.4, 2.9, 3.5\} \times 10^{18} \text{ FLOPs}. \tag{D.3}$$

Within budget C_j, let $q_i = \log_{10} N_i$ and y_i be the corresponding terminal loss. Observations satisfying $y_i - \min_r y_r \le 0.05$ are retained; if fewer than three remain, all observations at that budget are used. Least squares fits

$$\widehat{L}_j(q) = \gamma_{0,j} + \gamma_{1,j} q + \gamma_{2,j} q^2. \tag{D.4}$$

Table D.2: Existing single-domain online–test pairs. Differences are test minus online, in nats/token, with equal weight per run. Pass ranges cover only paired runs and use cumulative input tokens divided by domain training-input volume.

Training domain	Paired/all runs	Signed mean	Mean absolute	Pass range
Small molecule	32/49	$+0.000542$	0.003134	0.51–2.74
Peptide	41/41	-0.003172	0.005027	0.14–9.71
Protein	27/45	-0.000023	0.000373	0.41–1.92
DNA	14/41	$+0.001161$	0.001412	0.72–3.85
RNA	32/51	-0.000592	0.003198	0.43–1.93
Total	146/227	-0.000795	0.003004	0.14–9.71

For $\gamma_{2,j} > 0$, the vertex gives

$$q_j^* = -\frac{\gamma_{1,j}}{2\gamma_{2,j}}, \qquad N_j^* = 10^{q_j^*},$$
$$D_j^* = \frac{C_j}{6N_j^*}, \qquad L_j^* = \widehat{L}_j(q_j^*). \tag{D.5}$$

These are continuous fitted optima, not additional trained models. The quadratic is a local estimator, not an assumed global loss surface at arbitrary model and data scales.

Table D.3: IsoFLOP vertices underlying the main joint-pretraining curves.

C (10^{18} FLOPs)	N^* (M)	D^* (B)	L^*
0.35	20.905	2.790	4.557310
0.50	24.670	3.378	4.519640
0.70	33.676	3.464	4.485439
1.00	37.447	4.451	4.452312
1.30	45.924	4.718	4.428747
1.60	50.475	5.283	4.412010
2.00	55.694	5.985	4.395639
2.40	60.810	6.578	4.383515
2.90	62.053	7.789	4.370839
3.50	76.364	7.639	4.357202

Figure 4(a) shows the curves; Table D.3 lists their vertices.

D.3 Across-budget fits

Ordinary least squares on the vertices estimates

$$\log_{10} N_s^*(C) = a_s \log_{10} C + c_{N,s}. \tag{D.6}$$

The compute constraint gives

$$b_s = 1 - a_s, \qquad c_{D,s} = -\log_{10} 6 - c_{N,s}. \tag{D.7}$$

Thus $a_s + b_s = 1$ and the data exponent is not independently estimated. Optimal loss follows the empirical semi-log relation $L_s^*(C) = u_s \log_{10} C + v_s$; u_s is a slope against log compute, not a power-law exponent of $\log L$.

The joint-pretraining fits used in the main text are

$$\log_{10} N^*(C) = 0.5430459 \log_{10} C - 2.1958587, \tag{D.8}$$
$$\log_{10} D^*(C) = 0.4569541 \log_{10} C + 1.4177074, \tag{D.9}$$
$$L^*(C) = -0.1987741 \log_{10} C + 8.0359430. \tag{D.10}$$

The corresponding coefficients of determination are 0.987126, 0.981915, and 0.992998. Substitute FLOPs for C, parameter count for N, and input-token count for D. Embeddings are included in N, with tied weights counted once.

D.4 Empirical loss-variation scale

Five groups of repeated configurations provide an empirical scale of loss variation. For the selected paired repeats,

$$\widehat{\sigma} = \frac{\text{mean}\,|\Delta L|}{\sqrt{2}} = 0.00202. \tag{D.11}$$

The out-of-sample checks below express prediction errors relative to this scale. It is not the standard deviation of online–test differences.

D.5 Out-of-sample prediction

High-compute loss. The 0.55B model consumes 1.3117365×10^{20} FLOPs, approximately 37.48 times the largest fitting budget. Predicted online loss is 4.037036 and observed loss is 4.034795, a difference of -0.00224161, or approximately $1.11\widehat{\sigma}$ (Figure 4(c)). Its size is selected from the fitted relation, and training presents approximately 3.87 passes of input tokens. This tests extrapolation of terminal online loss at the selected configuration.

E Domain-Wise Scaling and Cross-Domain Comparisons

Figure 5 compares single-domain and joint-training loss frontiers and model–data configurations. This section provides complete within-budget curves, allocation coefficients, and test settings. Loss sources and their scope are specified in Appendix D.1.

E.1 Complete single-domain results

Single-domain pretraining comprises 227 runs and 27 fixed-budget curves. Each observation evaluates the complete in-domain test set. Unlike the local loss filtering used in many-to-many analysis, each single-domain quadratic uses all observations at its budget. A supported vertex requires at least five observations, $R^2 \geq 0.8$, at least two distinct parameter sizes on each side of both the observed minimum and fitted vertex, and at most one grid-position difference between the fitted and observed minima.

Table E.1: Single-domain experimental coverage and vertex support.

Domain	Runs	Budgets	Supported vertices
Small molecule	49	6	6
Peptide	41	3	2
Protein	45	6	5
DNA	41	6	4
RNA	51	6	5
Total	227	27	22

The five weakly supported vertices occur at protein budget 4, DNA budgets 4 and 5, RNA budget 2, and peptide budget 3. The first four fail the R^2 threshold; the peptide curve has a discrepancy between fitted and observed minimum locations. Table E.2 uses all 27 budget vertices. Figure E.1 shows the complete curves, with open markers for weakly supported vertices.

Table E.2: One-to-one scaling coefficients. The allocation exponents are a_s, b_s; u_s is the loss slope against $\log_{10} C$.

Domain	a_s	b_s	R^2_N	R^2_D	u_s	R^2_L
Small molecule	0.7526	0.2474	0.9938	0.9457	-0.4683	0.9920
Peptide	0.8265	0.1735	0.9951	0.8994	-1.4068	0.9874
Protein	0.4435	0.5565	0.8563	0.9036	-0.0899	0.9956
DNA	0.3970	0.6030	0.7640	0.8820	-0.1306	0.9893
RNA	0.6385	0.3615	0.9371	0.8268	-0.4260	0.9949

The domain exponents a_s range from 0.3970 to 0.8265. Peptide experiments cover only three budgets, and the cap for some supplementary configurations is relaxed to ten data passes. Their expo-

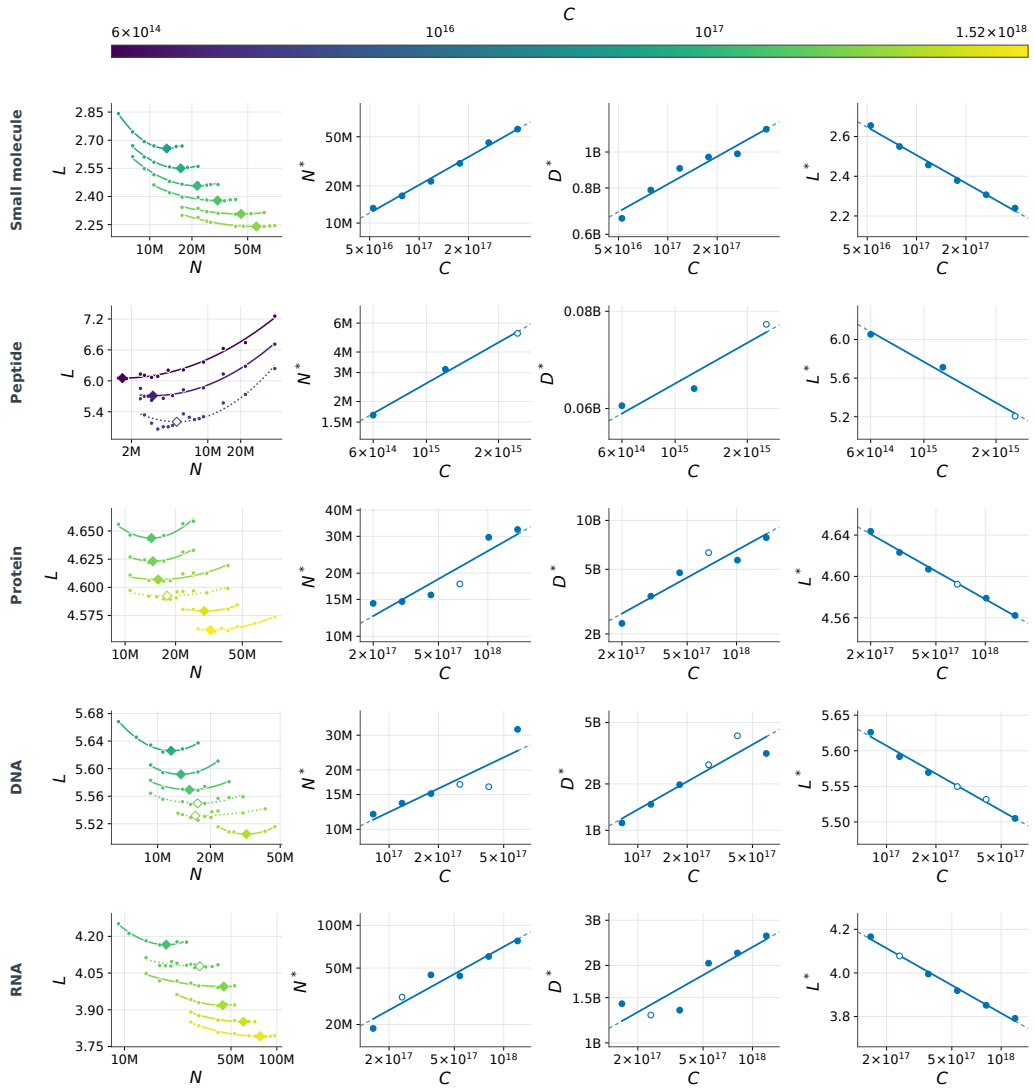


Figure E.1: Single-domain IsoFLOP curves and budget optima. Rows: small molecule, peptide, protein, DNA, RNA. Columns: within-budget curves, optimal model size, optimal training tokens, optimal loss. Solid lines denote fits, dashed lines extrapolation, and open vertices weak support.

nent should therefore be interpreted alongside the narrower budget range and data repetition. These experiments have no independent training replicates; R^2 describes fit quality only.

E.2 Domain-specific optima under joint pretraining

For each jointly pretrained model, we extract five domain-specific online loss estimates and locate optima at the same total compute budget. Here $D_k^*(C) = C/(6N_k^*(C))$ is the total mixed-domain training volume of the model selected for domain k's objective, not the training volume of domain k alone. Different domains can select different models at a given budget; these vertices are not five simultaneous optima of one model.

Across-budget allocation fits retain curves that are convex, have a vertex within the observed parameter range, and have an observed minimum away from the grid boundary.

Figure E.2 shows the corresponding curves. Excluding the peptide subset, model-size exponents range from 0.4982 to 0.5714, a narrower range than in single-domain training.

Table E.3: Domain-specific scaling coefficients under joint pretraining.

Domain	Vertices	a_s	b_s	u_s	R_N^2	R_L^2
Small molecule	10	0.564974	0.435026	-0.442127	0.991557	0.994598
Peptide	4	0.447480	0.552520	-0.078048	0.988031	0.995911
Protein	8	0.498211	0.501789	-0.075485	0.954691	0.997835
DNA	10	0.571420	0.428580	-0.112323	0.953617	0.994661
RNA	10	0.532794	0.467206	-0.450838	0.988513	0.988771

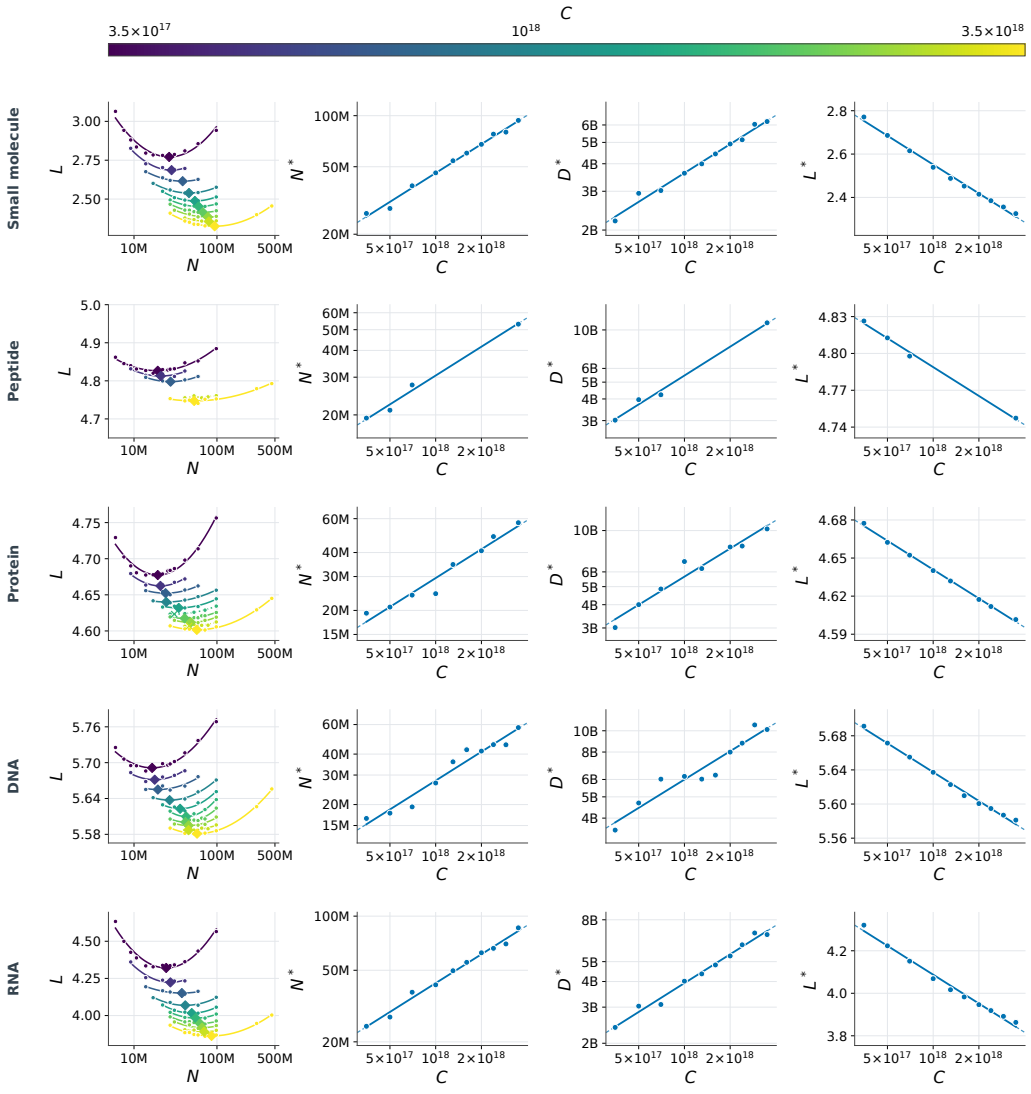


Figure E.2: Joint-pretraining IsoFLOP curves and domain-specific optima, with the same layout as Figure E.1. Here C and D^* count total mixed-domain compute and training tokens.

E.3 Loss frontiers and target-domain data efficiency

Resource accounting. The two joint-training frontiers in Figure 5 use the same configurations, expressed with total resources or target-domain shares under equation A.1. In the N–D panel, they have identical N coordinates and their vertical coordinates differ by r_k. In the loss–compute panel, losses are identical and compute coordinates are multiplied by r_k.

Coverage. Across the ten joint budgets, small molecules, peptides, proteins, DNA, and RNA have 10, 4, 8, 10, and 10 interior quadratic vertices, respectively. Other budgets are represented by observed grid minima, without treating a boundary minimum as an established continuous optimum.

Its orange and green markers are the same configuration under two accounting scales.

These frontiers correspond to their respective optimal model configurations. Single-domain training uses test loss and joint training uses training loss, as described in Appendix D.1. Target-domain resource axes use $r_k C$ and $r_k D$; the joint run's total compute budget remains C.

E.4 Joint evaluation after single-domain pretraining

Figure 6 evaluates a fixed five-domain test set using prediction-token weighting:

$$L_{\text{all}}(\theta) = \frac{\sum_k T_k L_k(\theta)}{\sum_k T_k}. \tag{E.1}$$

Table E.4 specifies record counts and denominators.

Table E.4: Fixed joint test set and evaluated models by training domain. All models use the same complete test objective.

Domain	Test records	Prediction tokens	Evaluated models
Small molecule	462,035	4,614,732	32
Peptide	4,167	81,620	35
Protein	540,806	27,093,028	24
DNA	117,276	6,625,302	14
RNA	491,528	11,022,322	31
Total	1,615,812	49,437,004	136

The evaluation covers 136 of the 227 single-domain models. Lines connect measured observations only.

Table E.5: Increasing RNA training compute at fixed model size, 41,034,752 parameters.

C (FLOPs)	In-domain test loss	Joint test loss
2.4×10^{17}	4.083636	15.706239
8.1×10^{17}	3.858084	17.844395

For the fixed 41.0M model, more RNA training reduces in-domain loss from 4.0836 to 3.8581 while increasing joint loss from 15.7062 to 17.8444 (Table E.5). Both are measured on fixed test sets.

F Downstream Adaptation and Evaluation

Downstream experiments initialize from the 0.55B and 1.44B multidomain pretrained models and fine-tune by task, retaining the common graph representation and tokenizer.

F.1 Tasks and datasets

Table F.1 summarizes task inputs and outputs. MolToken always generates discrete graph tokens. Three-dimensional computations, such as structure prediction, are post-generation evaluation tools rather than model inputs or outputs.

Validation splits are drawn from each task's training source for early stopping and model selection. Table F.2 reports fine-tuning sizes.

MOSES uses all 1,584,663 official training molecules. Of 1,273,104 GuacaMol training records; the remainder are split for training and validation. Protein generation selects UniRef50 sequences of exactly 100, 200, 300, or 400 residues. Representation and context screening retain 508,738 records, including an additional held-out set of 25,416. Another 858 sequences exceed the full 512-token context and two exceed ring-marker capacity; these are excluded from training.

Table F.1: Downstream conditions and generation targets.

Dataset/evaluation	Condition	Target
MOSES	None	Small molecule
GuacaMol	None	Small molecule
RAbD	Antibody sequence flanking CDR-H3; target length	CDR-H3 sequence
TCR External-13/Top-20	pMHC	CDR3β sequence
Rfam-17	RNA family	RNA sequence
DPLM protein evaluation	Target residue count	Protein sequence
DNA-Diffusion	Cell type	200-bp DNA
PepBench-Gen	Target residue count	5–80-aa peptide

Table F.2: Fine-tuning data sizes. Validation is used for model selection only.

Task	Training records	Validation records
MOSES	1,568,791	15,872
GuacaMol	1,260,235	12,856
RAbD	8,150	496
TCR	308,251	16,184
Rfam-17	298,364	63,938
Protein generation	459,233	24,089
DNA generation	37,292	
PepBench-Gen	49,077	10,520

The length-screened Rfam-17 collection has 529,585 records, of which 426,248 are accepted by the fixed representation. Beyond the training and validation sets in the table, 63,946 records are held out. Two RF01960 training targets are shortened from 1,808 to 1,801 nt to reduce their contexts from 514 to 512 tokens; evaluation references retain their original lengths. Released DNA-Diffusion assets contain 11,968 sequences of length 200 bp for each of four cell categories, with chr1/chr2 held out in the released split.

Pretraining and some downstream datasets share source databases. Novelty is measured against the reference collection specified for each metric. PepBench-Gen evaluates distribution modeling under exposure to the source collection.

F.2 Conditioning interface and fine-tuning

Input representation. Molecular conditions use the same FDDFS/BPE encoding and directly enter the model's token embeddings; multiple molecules are joined with component delimiters. A frozen MPNet encoder maps text conditions to 768-dimensional token-level states. Two linear layers with GELU project these states into the model embedding space, with a learned scale for embedding-norm matching. The projected sequence is a continuous prefix, not a single pooled vector.

Five fixed domain-type embeddings identify the target domain; no task-specific vocabulary expansion is used. Added embeddings are initialized from the mean of the original embedding distribution, and input/output weights remain tied. The target sequence is

$$[\mathrm{TYPE}_k, \mathrm{BOS}, t_1, \ldots, t_m, \mathrm{EOS}].$$

Conditions, type markers, BOS, and PAD are excluded from supervision; only target content and EOS contribute to loss. The combined condition and target context is at most 512 tokens.

Two-stage training. Alignment freezes the pretrained Transformer and optimizes conditioning projections and domain-type embeddings. Full fine-tuning starts from the matching Alignment checkpoint with the best validation loss and unfreezes the Transformer and projections, while keeping the text encoder frozen. Each stage runs for at most 100 epochs, with early stopping after five consecutive epochs without an improvement of at least 0.001 in validation loss. Final evaluation uses the best Full checkpoint, not the stopping-epoch checkpoint or a test-selected model.

Learning rates are 10^{-4} for Alignment and 5×10^{-5} for Full, with weight decay 0.1, 3% linear warmup followed by cosine decay to 10% of peak, and gradient-norm clipping at 1.0. Each epoch retains its final short batch.

Reporting. We distinguish published results, new samples from released models, and rescoring of released predictions. Unless stated otherwise, MolToken uses the predetermined sampling seed 42. Variation over sampling runs is not variation over independently trained models. The two capacities share task definitions and selection rules.

F.3 SMALL-MOLECULE DISTRIBUTION LEARNING

Generation and molecular decoding. BPE expansion and FDDFS decoding reconstruct attributed graphs, which are converted to RDKit molecules and canonical SMILES under fixed valence and hydrogen-assignment rules. Conversion preserves generated heavy atoms and connectivity. Invalid outputs remain in the request denominator.

F.3.1 MOSES

We use the official MOSES procedure (Polykovskiy et al., 2020), generating 30,000 raw samples per model and evaluating against Test and the scaffold-separated TestSF. Validity is the fraction of all requests that yield valid molecules. Unique@K measures canonical uniqueness among the first K valid outputs. Novelty compares unique valid generated molecules with the training reference. Filters is the fraction of valid molecules passing the official chemical filters.

FCD is the Fréchet distance between ChemNet feature distributions:

$$\mathrm{FD}(A,B) = \|\mu_A - \mu_B\|_2^2 + \mathrm{tr}\Big[\Sigma_A + \Sigma_B - 2(\Sigma_A^{1/2}\Sigma_B\Sigma_A^{1/2})^{1/2}\Big]. \tag{F.1}$$

SNN averages each generated molecule's maximum fingerprint Tanimoto similarity to the reference set. Frag and Scaf are cosine similarities between BRICS-fragment and Bemis–Murcko-scaffold count distributions, respectively. Internal diversity is

$$\mathrm{IntDiv}_p(G) = 1 - \left[\frac{1}{|G|^2}\sum_{g,g'\in G}\mathrm{Tan}(g,g')^p\right]^{1/p}, \qquad p \in \{1,2\}, \tag{F.2}$$

where G contains valid outputs. Differences in logP, SA, QED, and MW distributions use one-dimensional Wasserstein distance. Except for validity, metrics follow the official valid-sample handling rules.

Table F.3: MOSES validity, uniqueness, novelty, and internal diversity. † denotes published means.

Model	Valid.	Uniq.@1k	Uniq.@10k	Filters	Novelty	IntDiv_1	IntDiv_2
CharRNN†	0.9748	1.0000	0.9994	0.9943	0.8419	0.8562	0.8503
VAE†	0.9767	1.0000	0.9984	0.9970	0.6949	0.8558	0.8498
JTN-VAE†	1.0000	1.0000	0.9996	0.9760	0.9143	0.8551	0.8493
BioMatrix-1.7B-SFT	0.9980	1.0000	0.9990	0.9120	0.8377	0.8667	0.8608
BioMatrix-4B-SFT	0.9994	1.0000	0.9988	0.9112	0.7949	0.8660	0.8602
MolToken-0.55B	0.9766	1.0000	0.9992	0.9792	0.8402	0.8576	0.8515
MolToken-1.44B	0.9855	0.9997	0.9992	0.9808	0.8206	0.8576	0.8516

MolToken-0.55B and MolToken-1.44B achieve mean Test FCDs of 0.1117 and 0.1009, and mean TestSF FCDs of 0.5963 and 0.5850, respectively. Internal diversity is close to published baselines, whereas novelty is lower than CharRNN and JTN-VAE.

F.3.2 GUACAMOL

Distribution learning uses GuacaMol 0.5.5 (Brown et al., 2019). Each metric requests 10,000 samples and retains the official retries for obtaining valid or unique sets. The full five-metric evaluation can therefore require more than 10,000 raw draws.

The FCD score is $\exp(-0.2\,\mathrm{FCD})$, with larger values better. The KL score averages $\exp(-\mathrm{KL})$ across the official descriptor comparisons, using their prescribed continuous and discrete estimators. Validity, uniqueness, and novelty follow their respective native sampling procedures. MolToken-0.55B and MolToken-1.44B obtain FCD scores of 0.8951 and 0.8992, with KL scores of 0.9786 and 0.9812. Their novelty remains below several published baselines. The training-set random sampler is a distributional reference, not a model capable of novel generation.

Table F.4: MOSES distribution similarity. Lower FCD and higher SNN, Frag, and Scaf are better. † denotes published results.

Model	Reference	FCD	SNN	Frag	Scaf
CharRNN†	Test	0.0732	0.6015	0.9998	0.9242
CharRNN†	TestSF	0.5204	0.5649	0.9983	0.1101
VAE†	Test	0.0990	0.6257	0.9994	0.9386
VAE†	TestSF	0.5670	0.5783	0.9984	0.0588
JTN-VAE†	Test	0.3954	0.5477	0.9965	0.8964
JTN-VAE†	TestSF	0.9382	0.5194	0.9947	0.1009
BioMatrix-1.7B-SFT	Test	1.5591	0.5583	0.9918	0.8542
BioMatrix-1.7B-SFT	TestSF	1.9500	0.5306	0.9906	0.2134
BioMatrix-4B-SFT	Test	1.5424	0.5675	0.9934	0.8490
BioMatrix-4B-SFT	TestSF	1.9501	0.5369	0.9925	0.2021
MolToken-0.55B	Test	0.1117	0.5997	1.0000	0.9216
MolToken-0.55B	TestSF	0.5963	0.5645	1.0000	0.1353
MolToken-1.44B	Test	0.1009	0.6048	1.0000	0.9251
MolToken-1.44B	TestSF	0.5850	0.5669	1.0000	0.1190

Table F.5: MOSES property-distribution Wasserstein distances; lower is better.

Model	logP	SA	QED	MW
BioMatrix-1.7B-SFT	0.4519	0.1687	0.0956	35.5553
BioMatrix-4B-SFT	0.4291	0.1583	0.0948	35.9346
MolToken-0.55B	0.1146	0.0175	0.0145	9.3647
MolToken-1.44B	0.0999	0.0176	0.0137	8.5464

Table F.6: GuacaMol distribution learning; higher is better. † denotes published results. The random sampler is a training-distribution reference.

Model	Validity	Uniqueness	Novelty	KL score	FCD score
Random sampler†	1.0000	0.9970	0.0000	0.9980	0.9290
SMILES LSTM†	0.9590	1.0000	0.9120	0.9910	0.9130
Graph MCTS†	1.0000	1.0000	0.9940	0.5220	0.0150
AAE†	0.8220	1.0000	0.9980	0.8860	0.5290
ORGAN†	0.3790	0.8410	0.6870	0.2670	—
VAE†	0.8700	0.9990	0.9740	0.9820	0.8630
BioMatrix-1.7B-SFT	0.9986	0.9998	0.9350	0.8979	0.6159
BioMatrix-4B-SFT	0.9993	0.9996	0.8855	0.8886	0.5922
MolToken-0.55B	0.9381	0.9996	0.9612	0.9786	0.8951
MolToken-1.44B	0.9513	0.9999	0.9485	0.9812	0.8992

F.4 ANTIBODY CDR-H3 COMPLETION

Data and task. We use all 58 RAbD test cases released in the RefineGNN repository (Jin et al., 2022). The paper describes a 60-case collection, so we rerun RefineGNN using the authors' released weights and native sampling procedure on the same 58 cases. Table F.7 reports this rerun alongside MolToken.

MolToken is conditioned on cropped antibody sequences flanking CDR-H3 and the requested residue count, and generates the entire CDR-H3 without structural coordinates. The 8,646 official training-source records are grouped by exact cropped sequence and split with seed 42 into 8,150 training and 496 internal validation records. The longest complete encoded context is 102 tokens. We retain the released split.

Sampling and ranking. MolToken generates 10,000 raw candidates per case at temperature 1 and top-$p = 1$, without top-k truncation, using free EOS generation and sampling seed 7. Candidates are ranked by the complete BPE-path negative log probability, including EOS, divided by the requested residue count. The top 100 per case yield 5,800 selected candidates. Duplicates and invalid outputs remain, and ranking uses only model probabilities. RefineGNN retains its structurally supervised, fixed-length amino-acid generation model and native probability ranking.

Amino-acid recovery. We apply the revised prefix-L AAR uniformly to the fixed candidate sets. For reference length L_j, only the first L_j original positions of a valid output are compared; missing positions and invalid outputs contribute zero, and overlong tails are ignored. If $I(x)$ indicates complete decoding to a valid amino-acid sequence,

$$\mathrm{AAR}_{\mathrm{prefix}} = \frac{\sum_{j=1}^{58} \sum_{m=1}^{100} I(x_{jm}) \sum_{i=1}^{\min(|x_{jm}|, L_j)} \mathbf{1}\{x_{jm,i} = y_{j,i}\}}{\sum_{j=1}^{58} 100 L_j}. \tag{F.3}$$

The denominator is fixed at 75,400 and matches are counted at their original positions. All evaluated models use the same scoring function, with candidates ranked before scoring. For RefineGNN's valid, prescribed-length outputs, this definition equals its native AAR. Table F.7 also reports validity and exact-length rates.

Table F.7: Results on the 58 released RAbD cases (%). AAR uses the prefix-L adaptation. Raw validity divides by 580,000 requests; selected validity and exact-length rates divide by 5,800 slots. RefineGNN is rerun on the same collection with released weights; its raw validity was not retained and is left blank.

Model	Adapted AAR	Raw valid.	Selected valid.	Selected exact length
RefineGNN	34.8064	—	100.0000	100.0000
BioMatrix-1.7B-SFT	23.7082	87.6666	83.7414	3.5690
BioMatrix-4B-SFT	21.2188	25.0002	37.9310	1.7241
MolToken-0.55B	38.2772	64.6903	96.2069	47.8793
MolToken-1.44B	37.6923	67.2264	98.0517	46.3448

Results. MolToken-0.55B and MolToken-1.44B recover 28,861 and 28,420 positions, yielding prefix-L AARs of 38.2772% and 37.6923%, respectively. RefineGNN recovers 26,244 positions on the same collection, with an AAR of 34.8064%.

F.5 PMHC-CONDITIONED TCR GENERATION

The condition is pMHC and the target is a CDR3β sequence, without structural coordinates. We use the data and scoring methods released with TCRT5 (Karthikeyan et al., 2025), reporting External-13 and auxiliary Top-20 evaluations.

External-13. We generate 1,000 ordered outputs for each of 13 pMHC groups. MolToken uses 1,000-beam decoding with at most 25 BPE tokens and a no-repeat 5-gram constraint. The constraint applies to BPE tokens and is not identical to an amino-acid-token constraint.

Exact match counts outputs identical to any reference in the group, including repeated hits as separate slots. Recovery selects a nearest same-length reference when available and computes positional recovery; otherwise, the released implementation uses $1 - d_{\text{edit}}/|r|$. We count outputs with recovery at least 90%. GIANA 4.1 jointly clusters generated and reference sequences in CDR3-only mode and counts generated outputs assigned to clusters containing a reference.

Table F.8: TCR External-13 hit counts: mean ± sample standard deviation across groups (total hits). Higher is better. Released denotes rescored released predictions; resampled denotes new outputs from the released model.

Model	Source	Exact	Recovery $\geq 90\%$	GIANA
TCRT5 FT	Resampled	0.231 ± 0.599 (3)	3.077 ± 4.769 (40)	6.462 ± 14.104 (84)
TCRT5	Released	0.154 ± 0.376 (2)	3.462 ± 5.577 (45)	8.000 ± 14.306 (104)
GRATCR	Released	0.000 ± 0.000 (0)	0.538 ± 1.941 (7)	0.000 ± 0.000 (0)
ER-TRANSFORMER	Released	0.000 ± 0.000 (0)	0.077 ± 0.277 (1)	0.077 ± 0.277 (1)
ER-TRANSFORMER+	Released	0.077 ± 0.277 (1)	1.923 ± 2.100 (25)	2.154 ± 4.200 (28)
MolToken-0.55B	This work	0.231 ± 0.599 (3)	2.692 ± 3.860 (35)	2.231 ± 4.952 (29)
MolToken-1.44B	This work	0.154 ± 0.376 (2)	3.154 ± 3.648 (41)	3.077 ± 3.639 (40)

Means and sample standard deviations are computed over hit counts in the 13 groups, not percentages. Each ER baseline has 12,956 released outputs; the other methods have 13,000. Missing outputs are not replenished. The two MolToken models decode 12,958 and 12,985 valid sequences; failures remain non-hits. GIANA proximity does not establish experimentally verified binding.

Top-20. Each group uses 300 beams to retain 100 outputs, giving 2,000 outputs in total, plus one greedy output per group. All methods use the same ordered collection of 68,416 references. Char-BLEU is character-level corpus BLEU for the greedy outputs. Precision@100, Recall@100, and F1@100 are computed within groups and averaged; recall divides by the smaller of the group reference count and 100. Diversity is the fraction of distinct sequences among all 2,000 slots. Perplexities under different tokenization units are not compared across models. The 1.44B model improves

Table F.9: Auxiliary TCR Top-20 evaluation. P, R, and F1 denote Precision@100, Recall@100, and F1@100. Lower edit distance is better. † denotes author-reported results.

Model	Char-BLEU	P	R	F1	Edit dist.	Recovery	Diversity
TCRT5 FT (50k)	0.9479	0.0905	0.0905	0.0905	1.7140	0.8492	0.6530
TCRT5 FT (48k)†	0.9643	0.0905	0.0905	0.0905	1.7070	0.8502	0.6500
MolToken-0.55B	0.8483	0.0445	0.0445	0.0445	2.2945	0.8008	0.8750
MolToken-1.44B	0.8810	0.0560	0.0560	0.0560	2.1930	0.8119	0.8045

BLEU, F1, and recovery over 0.55B but remains below TCRT5. Sequence diversity decreases from 0.8750 to 0.8045.

F.6 RNA-FAMILY-CONDITIONED GENERATION

Rfam-17 follows the 17-family evaluation scope and released covariance models of RfamGen (Sumi et al., 2024). We request 1,000 sequences per family, conditioned on family ID. Infernal 1.1.3 computes bit scores against the corresponding model with `cmalign --notrunc --sfile`. Scores are normalized by the family gathering threshold (GA) from Rfam 14.7:

$$S_{\text{rel}}(x, k) = \frac{\text{bit}(x, \text{CM}_k)}{\text{GA}_k}. \tag{F.4}$$

For each family, we compute the mean, median, and fraction with $S_{\text{rel}} > 1$, then average over the 17 families equally. Passing rates use both scored-output and all-request denominators; coverage is the scored fraction of requests.

Baselines use released family scores.

Table F.10: Equal-family averages on Rfam-17. † denotes aggregates of released scores. GA/scored uses the indicated denominators. Blanks denote unavailable results.

Model	Rel. mean	Rel. median	GA/scored
RfamGen†	6.7585	6.7678	99.0392%
GCVAE†	5.9575	5.9871	99.2011%
GVAE†	0.2731	0.2306	14.9934%
CVAE†	0.0848	0.0042	10.1108%
MolToken-0.55B	5.0443	5.3020	74.7561%
MolToken-1.44B	5.1659	5.2994	74.2669%

Table F.11: MolToken-0.55B by RNA family, with 1,000 requests per family.

Family	Scored/request	Rel. mean	Rel. median	GA/request	GA/scored
RF00001	955/1000	1.7555	1.7661	805/1000 (80.50%)	84.29%
RF00003	916/1000	1.4603	1.6771	715/1000 (71.50%)	78.06%
RF00004	854/1000	2.1656	2.4598	669/1000 (66.90%)	78.34%
RF00017	819/1000	1.0404	1.0209	489/1000 (48.90%)	59.71%
RF00029	905/1000	0.8757	1.0470	497/1000 (49.70%)	54.92%
RF00059	940/1000	0.9905	1.1362	608/1000 (60.80%)	64.68%
RF00100	409/1000	3.2551	3.6171	305/1000 (30.50%)	74.57%
RF00163	983/1000	1.1912	1.2276	815/1000 (81.50%)	82.91%
RF00174	778/1000	1.7194	2.2250	536/1000 (53.60%)	68.89%
RF00177	34/1000	30.4673	31.4598	33/1000 (3.30%)	97.06%
RF00230	821/1000	0.6222	0.6300	54/1000 (5.40%)	6.58%
RF00906	975/1000	1.3087	1.3274	862/1000 (86.20%)	88.41%
RF00994	938/1000	1.0533	1.1003	695/1000 (69.50%)	74.09%
RF01960	49/1000	33.7396	35.2640	47/1000 (4.70%)	95.92%
RF02271	974/1000	1.1698	1.1774	846/1000 (84.60%)	86.86%
RF04021	977/1000	1.3181	1.3368	819/1000 (81.90%)	83.83%
RF04088	896/1000	1.6196	1.6620	822/1000 (82.20%)	91.74%

F.7 LENGTH-CONDITIONED PROTEIN GENERATION

Following the DPLM evaluation design (Wang et al., 2024), we request 40 sequences at each length of 100, 200, 300, and 400 residues. MolToken uses a textual length condition and free EOS, whereas DPLM fixes length through the number of masked positions. We report exact-length rate, successful-folding coverage, and structure quality separately.

ESMFold predicts structures. pLDDT is first averaged over each sequence's residues, then within each length group, then equally over the four groups. Inner-TM uses fast TM-align, normalized by the first chain's length, and averages ordered pairs either including or excluding self-pairs. Lower inner-TM indicates greater diversity among predicted structures, not higher quality of each structure.

Table F.12: Protein generation with 40 requests at each of four lengths. Inner-TM is shown with and without self-pairs.

Model/reference	Fold coverage	Exact length	pLDDT	$\mathrm{TM}_{\mathrm{self}}$	$\mathrm{TM}_{\mathrm{no\ self}}$
DPLM150M	100.000%	100.00%	81.8768	0.2929	0.2748
MolToken-0.55B	66.250%	18.75%	35.1518	0.2942	0.2655
MolToken-1.44B	76.875%	27.50%	34.3455	0.2937	0.2694
Natural UniRef50	100.000%	100.00%	63.7940	0.2556	0.2365

Table F.13: pLDDT reported in the DPLM paper. The last column is the equal-length-group mean.

Model	100 aa	200 aa	300 aa	400 aa	Mean
DPLM150M	73.31	84.30	84.82	86.90	82.3325
DPLM650M	74.00	85.61	85.91	88.16	83.4200
DPLM3B	77.78	86.16	87.39	90.06	85.3475

MolToken pLDDT is measured only on successfully folded outputs and is lower than DPLM's. Coverage and exact-length rates are shown separately. Natural UniRef50 sequences serve as a data reference. Published values and newly generated results appear in separate tables.

F.8 CELL-TYPE-CONDITIONED DNA GENERATION

DNA-Diffusion designs 200-bp regulatory sequences conditioned on cell type (DaSilva et al., 2026). MolToken requests 1,000 outputs each for GM12878, K562, and HepG2 at temperature 1 and top-$p = 1$, without top-k restriction, with at most 128 BPE tokens and sampling seed 20260902. Strict validity requires successful decoding, an ACGT-only alphabet, and exactly 200 bp.

Table F.14: DNA sequence validity over all requests, and uniqueness within valid outputs.

Model	Strict 200-bp validity	Uniqueness
MolToken-0.55B	39.0%	100.0%
MolToken-1.44B	37.9%	100.0%

Table F.14 reports sequence validity. Table F.15 lists cell-specific motif-distribution results, with uncompleted entries left blank.

Table F.15: Motif-distribution comparison under the DNA-Diffusion protocol ($1 - \mathrm{Gini}$). † denotes released results from Figure 7c of the original paper.

Model/reference	GM12878	K562	HepG2
DNA-Diffusion†	0.5283	0.5465	0.5204
Training reference†	0.5896	0.5688	0.5960
MolToken-0.55B			
MolToken-1.44B			

F.9 LENGTH-CONDITIONED PEPTIDE GENERATION

PepBench-Gen is constructed from PepBenchmark sources to evaluate distribution modeling of 5–80-aa peptides; it is not the dataset's official generation leaderboard. Target lengths are sampled once from the training distribution. Each MolToken model and an amino-acid-frequency baseline generate 10,000 sequences for each seed in $\{42, 43, 44\}$. The baseline samples residues independently at the requested length; MolToken uses a length prompt and free EOS.

Validity requires graph decoding, normal EOS termination, the standard 20-amino-acid alphabet, and the specified length range. Uniqueness and novelty use valid outputs, with novelty measured against both the fine-tuning training set and the full amino-acid source collection of 109,513,498 sequences. Nearest-reference distance is

$$d_{\mathrm{near}}(g, \mathcal{R}) = \min_{r \in \mathcal{R}} \frac{d_{\mathrm{Lev}}(g, r)}{\max(|g|, |r|)}. \tag{F.5}$$

Internal diversity is the mean pairwise normalized edit distance over at most 2,000 outputs sampled without replacement.

Feature-distribution metrics use ESM2-t6-8M layer-6 representations averaged over actual residues, excluding special symbols, yielding 320 features per sequence. ESM-FBD uses equation F.1. Precision and recall measure feature coverage with $k = 3$ nearest-neighbor balls. Pseudo-perplexity masks residues individually:

$$\mathrm{pPPL}(x) = \exp\left[-\frac{1}{|x|}\sum_i \log p(x_i \mid x_{\setminus i})\right], \tag{F.6}$$

then averages over valid outputs. Amino-acid composition and length distributions use base-2 Jensen–Shannon divergence, without the square root.

Table F.16: PepBench-Gen means and sample standard deviations over three sampling runs. All ESM metrics use valid outputs. Distances, diversity, and novelty do not establish biological activity.

Metric	AA-frequency baseline	MolToken-0.55B	MolToken-1.44B
Validity	1.0000 ± 0.0000	0.7926 ± 0.0014	0.8073 ± 0.0041
Uniqueness	1.0000 ± 0.0001	0.9969 ± 0.0009	0.9959 ± 0.0006
Novelty / SFT train	1.0000 ± 0.0001	0.9730 ± 0.0005	0.9634 ± 0.0010
Nearest distance / SFT train	0.5576 ± 0.0015	0.4800 ± 0.0007	0.4722 ± 0.0024
Novelty / AA source	1.0000 ± 0.0001	0.9688 ± 0.0009	0.9591 ± 0.0012
Nearest distance / AA source	0.5025 ± 0.0015	0.4361 ± 0.0008	0.4294 ± 0.0025
Diversity	0.8599 ± 0.0005	0.8737 ± 0.0010	0.8732 ± 0.0009
ESM-FBD	2.5033 ± 0.0177	0.2378 ± 0.0022	0.2252 ± 0.0052
ESM-pPPL	21.3898 ± 0.0393	18.0448 ± 0.0168	18.0181 ± 0.0618
Precision	0.8646 ± 0.0065	0.7839 ± 0.0051	0.7924 ± 0.0034
Recall	0.5022 ± 0.0179	0.8689 ± 0.0036	0.8656 ± 0.0030
AA JSD	0.000039 ± 0.000014	0.001527 ± 0.000114	0.001307 ± 0.000099
Length JSD	0.001491 ± 0.000173	0.009015 ± 0.000098	0.009589 ± 0.000657
Exact-length rate	1.0000 ± 0.0000	0.5607 ± 0.0040	0.5548 ± 0.0074

Reported means and sample standard deviations summarize three sampling runs from fixed models. MolToken has lower feature-distribution distances than the frequency baseline but lower validity. The 1.44B model does not improve novelty or recall over 0.55B.