NanoFold: A Compute-Efficient Benchmark for Controlled and Autonomous Research on Protein Structure Models

Chris Hayduk* [1] **Krithik Ramesh*** [2]

Abstract

Open-source AlphaFold (AF)-style systems have rapidly advanced protein structure prediction, but isolating the architectural and training choices that drive these gains remains difficult: production-scale training is computationally prohibitive outside well-resourced labs, and public corpora contain biological dependencies that can confound reduced-scale studies. We introduce **NanoFold**, a compact fixed-data benchmark for AF-style training studies, paired with a reference codebase for controlled head-to-head comparison. NanoFold defines three tracks with held-out test labels: a *limited* track for sample efficiency, a *research large* track for testing whether early gains persist under further optimization, and an *unlimited* track for best achievable performance under the fixed data budget. Splits are disjoint by MMseqs2 sequence cluster and PDB entry and stratified on structural metadata, yielding 10,000 training, 1,000 public-validation, and 1,000 sealed test chains. We verify the construction using structural features, sequence-family coverage, protein foundation model embeddings, and a randomization study over 1,000 alternative splits, finding NanoFold statistically typical and well distributed. Using OpenAI's GPT-5.5 in the Codex harness to autonomously run ablations and architecture-search experiments, we show that NanoFold is learnable but unsaturated, separates training primitives, scales predictably with budget, and supports iterative discovery beyond predefined AF-style variants, enabling transparent and reproducible architectural research at compute-accessible scale.

*Equal contribution [1]OpenAI, San Francisco, CA, USA [2]Lyra Labs, Cambridge, MA. Correspondence to: Chris Hayduk <chrishayduk@openai.com>, Krithik Ramesh <krithik@lyralabs.ai>.

Accepted at the 2026 Workshop on Generative and Agentic AI for Biology (ICML 2026)

1. Introduction

AF-style protein structure prediction has become a steady source of new publicly released systems, with OpenFold (Ahdritz et al., 2024), ESMFold (Lin et al., 2023), Boltz-1 and Boltz-2 (Wohlwend et al., 2024; Passaro et al., 2025), Chai-1 (Chai Discovery et al., 2024), Protenix (ByteDance AML AI4Science Team et al., 2025), and the ongoing OpenFold3 effort (The OpenFold3 Team, 2026) all building on AlphaFold2 and AlphaFold3 (Jumper et al., 2021; Abramson et al., 2024). These systems vary architecture, training curriculum, optimization, and data pipeline simultaneously, which makes it difficult to attribute empirical gains to specific design choices. The field is therefore accumulating systems faster than it accumulates controlled evidence about which choices generalize, even as protein structure prediction continues to anchor active programs in drug discovery (Sadybekov & Katritch, 2023), protein engineering (Huang et al., 2016; Watson et al., 2023), and structural biology (Baek et al., 2021; Jumper et al., 2021). Closing this gap requires controlled experimentation at a scale that many groups can actually access.

Two natural routes are full-scale reproduction and reduced-scale training, but both remain difficult in practice. Reproducing AF-style training from scratch is still expensive: OpenFold reports approximately 50,000 GPU hours for its AlphaFold2 reproduction (Ahdritz et al., 2024), ScaleFold reduces AF2 pretraining to roughly 10 hours of wall-clock time only by distributing across 2,080 NVIDIA H100 GPUs (Zhu et al., 2024), and production-scale biomolecular efforts such as NVIDIA's Proteina-Complexa report multi-stage runs spanning tens to hundreds of thousands of optimizer steps on 16–96 A100-80GB GPUs (Didi et al., 2026). OpenFold3-preview2 likewise remains a prerelease system whose reported training used 256 NVIDIA H100 GPUs (The OpenFold3 Team, 2026), putting multi-run attribution studies out of reach for nearly all groups. The subsampling route fails for different reasons. Public AF-style corpora contain sequence-cluster homology, repeated constructs, alternate crystal forms, and same-experiment chains, so naive random splits can leak biological signal across train and evaluation sets. Data-pruning results also show that the best selection strategy depends sharply on the data regime,

retaining hard examples when data are abundant but easy examples when data are scarce (Sorscher et al., 2022). A useful reduced benchmark therefore needs leakage-controlled splits, distributional stratification, a sealed evaluation set, and a shared training implementation. No existing resource provides this combination for small-data, small-parameter AF-style training.

NanoFold is our attempt at this combined construction. It pairs a deliberately constructed 10,000-chain training corpus, drawn from OpenProteinSet (Ahdritz et al., 2023), with public and sealed evaluation splits, an automated FoldScore evaluation protocol, and a reference AF-style training codebase. This design also makes NanoFold a practical test bed for autonomous machine learning research agents. Rather than requiring weeks of dataset construction, model implementation, and evaluation engineering before any research signal can be observed, NanoFold presents agents with a small but structured protein-folding problem whose data, training code, and scoring function are already fixed. Using autonomous research agents based on OpenAI's GPT-5.5 in Codex, we show that AF-style training learns meaningful structural representations at parameter and data scales below prior studies, and that ablation behaviors at these scales recover findings observed in production-scale models.

Contributions. We summarize our contributions as follows.

- **A fixed-data benchmark for small-scale AF-style training.** NanoFold provides a leakage-aware benchmark for studying architectural and training choices in AF-style protein structure models, organized around a 10,000-chain training set, 1,000-chain public validation set, and 1,000-chain sealed hidden set across increasing compute budgets. It also includes a reference codebase of AF-style baselines and ablations, enabling controlled comparison under matched data and matched implementation.
- **A reusable split-construction protocol for structured biological data.** NanoFold enforces MMseqs2 (Steinegger & Söding, 2017) cluster and PDB-entry disjointness, stratifies over structural metadata from CATH (Sillitoe et al., 2021), SCOPe (Chandonia et al., 2022), and ECOD (Cheng et al., 2014), and verifies split quality through a randomization study over 1,000 alternative valid splits. This provides a template for constructing reduced biological datasets without collapsing into leakage-prone or distributionally biased subsamples.
- **Empirical evidence that reduced-scale AF-style training is scientifically informative.** Using autonomous research agents based on OpenAI's GPT-5.5 in Codex, we show that AF-style models learn meaningful structural representations at parameter and data scales below prior studies, and that ablation behaviors in this regime recover findings observed in production-scale models.

2. Related Work

Community-scale blind assessment is the dominant evaluation regime for protein structure prediction, with CASP (Kryshtafovych et al., 2023; 2026) on unsolved targets, CAMEO (Haas et al., 2018) on weekly PDB releases, and PXMeter (Ma et al., 2025) providing unified evaluation of several released biomolecular predictors across curated biological-complex benchmarks. ProteinNet (AlQuraishi, 2019) provides earlier standardized data, and AlphaFold DB (Varadi et al., 2022) contributes predicted structures as a resource rather than a labeled benchmark. These resources evaluate trained models and address what gains have been achieved rather than what drives them. NanoFold inherits the sealed-test-set practice but redirects it toward attribution at scales where architectural studies are practical to run from scratch.

Public code, data, and weight releases have made AF-style training reproducible. OpenFold (Ahdritz et al., 2024) and OpenProteinSet (Ahdritz et al., 2023) provide an AlphaFold2 reproduction and redistributable corpus; Boltz-1 and Boltz-2 (Wohlwend et al., 2024; Passaro et al., 2025), Chai-1 (Chai Discovery et al., 2024), Protenix (ByteDance AML AI4Science Team et al., 2025), and OpenFold3 (The OpenFold3 Team, 2026) extend the open release model to AF3-style biomolecular modeling; and ScaleFold (Zhu et al., 2024) and MegaFold (La et al., 2025) reduce wall-clock training cost. None of these by itself provides a controlled testbed for architectural study. NanoFold builds on this infrastructure but inverts the optimization target. Existing systems work to reduce the cost of a fixed recipe; NanoFold fixes the regime so different recipes can be compared.

The integrity of such comparisons depends on the data itself, where avoiding sequence and structural leakage between training and evaluation has been a long-standing concern. Sequence clustering (Fu et al., 2012; Steinegger & Söding, 2017) and time-based PDB-deposition cutoffs, used in AlphaFold2's CASP14-cutoff training (Jumper et al., 2021) and Protenix's September 30, 2021 wwPDB cutoff (ByteDance AML AI4Science Team et al., 2025), are standard tools. Adjacent benchmark efforts including the Therapeutic Data Commons (Huang et al., 2021), MoleculeNet (Wu et al., 2018), and the Open Graph Benchmark (Hu et al., 2020) provide curated datasets with explicit, task-appropriate split and evaluation protocols; within structural biology, however, splits are usually reported as design choices rather than verified statistically. NanoFold extends this with a verification protocol combining metadata-balance audits, ESM2-based embedding coverage diagnos-

Figure 1. **NanoFold overview.** NanoFold builds a leakage-controlled OpenProteinSet-derived source pool, balances train/public/hidden splits over structural metadata, and evaluates AF-style models under fixed-data, sealed-hidden tracks.

tics, and a conditional randomization study against 1,000 alternative valid splits, on top of MMseqs2 cluster and PDB-entry disjointness. To our knowledge, this combination has not previously been applied to protein structure benchmark construction.

3. Dataset Construction

The aim of NanoFold's data construction is to make comparisons across architectural and training choices read as modeling differences rather than as artifacts of how the data was split. Achieving this with PDB-derived chains is harder than it appears, because the chains in the pool form a structured population with homologs, repeated constructs, close mutants, duplicated families, alternate crystal forms, same-complex chains, and historically biased regions of structure space. Splits drawn without care can leak close relatives across train and evaluation, bias one split over another, or concentrate training on dense redundant pockets. NanoFold's response is a leakage-controlled, stratified split over a fixed eligible pool, with sealed evaluation and statistical verification.

3.1. Source pool and eligibility

All splits draw from a single eligible source pool, so that every participant trains and evaluates on the same processable universe of chains under the same tensor schema. The pool is assembled from public OpenFold/OpenProteinSet assets (Ahdritz et al., 2023; 2024) and RCSB mmCIF coordinates (wwPDB Consortium, 2019; Burley et al., 2021; Westbrook et al., 2022), with OpenProteinSet supplying chain metadata, duplicate-chain mappings, and precomputed `uniref90_hits.a3m` MSA assets derived from UniRef90 (Suzek et al., 2015); mmCIF files supplying atomic coordinates from which atom14 labels are constructed; and CATH, SCOPe, and ECOD annotations providing structural and domain classifications for downstream balancing (Cheng et al., 2014; Sillitoe et al., 2021; Chandonia et al., 2022). Locking these assets in advance is central to the benchmark, since it gives every participant identical evolutionary input and prevents the leaderboard from becoming a contest over external retrieval or database freshness. A fixed data loader then serializes feature tensors (residue identities, MSA rows, deletion features, residue indices, masks, and zero-template tensors with $T = 0$) and label tensors (Cα coordinates and masks, atom14 positions and masks, residue indices, and resolution) per eligible chain.

Eligibility filters confine the pool to chains where the modeling task itself is well-posed under shared resources. A chain enters the pool only if it satisfies the official processability and quality gates: presence in the OpenFold chain cache, length $40 \leq L \leq 256$, monomeric status, only the 20 standard amino acids, resolution at most 3.0Å when reported, the required OpenFold feature assets, and successful atom14 projection (sufficient sequence identity, coverage, aligned fraction, and valid Cα support between the feature-side query sequence and the extracted mmCIF coordinates). These gates confine the task to a single-chain, short-to-medium-length regime where every selected example can be downloaded, represented, trained on, and scored consistently.

3.2. Split construction

To control biological dependencies, NanoFold forms units rather than sampling chains directly. Let $\mathcal{C}$ denote the eligible chains. We build a graph $G = (\mathcal{C}, E)$ with edges connecting chains that share an MMseqs2 sequence cluster

(30% identity, 80% coverage) or a PDB entry (Steinegger & Söding, 2017; wwPDB Consortium, 2019), and assign every chain in a connected component to the same split. This blocks close-homology leakage through sequence clusters and same-experiment leakage through PDB grouping, and prevents large duplicated families from receiving disproportionate influence.

Unit-level grouping prevents leakage but not distributional skew. A random allocation of units could still produce a hidden-validation set that is systematically more beta-rich, longer, or lower-resolution than training, in which case the leaderboard would measure distribution shift rather than data-efficient geometry learning. We stratify units before allocation by assigning each unit a label

$$z(u) = (\text{secondary}(u), \text{domain}(u), \\ \text{length_bin}(u), \text{resolution_bin}(u))$$

derived from the official chain annotations; these fields serve as balancing axes only and are not exposed to models or used as scoring labels. The allocator then samples units proportionally across strata to produce splits of size $|T| = 10{,}000$, $|V_{\text{pub}}| = 1{,}000$, and $|V_{\text{hid}}| = 1{,}000$.

Validation sets that are repeatedly iterated against eventually turn into implicit training signal, so NanoFold separates public and sealed hidden validation. Public train and public-validation manifests are committed and hash-locked, and hidden validation is generated privately after the public manifests are fixed: units containing any public-train or public-validation chain are removed, hidden counts are matched to the public train and validation distributions, and hidden chains are selected deterministically from a private salt. During hidden evaluation the runtime serves only feature tensors and withholds Cα coordinates, atom14 labels and masks, and resolution, and hidden manifests, labels, features, fingerprints, and locks remain maintainer-only.

3.3. Statistical verification

Even with leakage controls and stratification in place, the committed split is one realization out of many possible valid splits drawn under the same constraints, and its specific behavior is an empirical question. Three diagnostics establish its key properties: a metadata-balance audit checks that the splits are compositionally comparable over the design fields; an embedding-space coverage analysis checks that training broadly covers the eligible pool without concentrating on dense redundant pockets; and a conditional randomization test checks that the committed split is statistically typical of valid splits drawn under the same constraints.

Metadata balance. For each categorical balancing field b, let p_S^b denote the empirical distribution of split S over the buckets of b, and $p_{\mathcal{U}}^b$ the corresponding distribution over the eligible disjoint-unit source pool. We measure split discrepancy using Jensen–Shannon divergence (JSD), total variation distance (TV), and maximum bin deviation (Appendix A.1.1). Table 1 reports the maximum discrepancy across the four audited fields; all values are small, with the largest pairwise JSD at 7.66×10^{-4} (train vs. public validation) and the largest pairwise maximum bin deviation at 0.70 percentage points. The hidden split is sealed for ranking but is not distributionally adversarial; it is matched to the public regime over the metadata axes that most directly affect difficulty and label quality.

Table 1. **Split-comparability diagnostics.** Maxima across secondary-structure class, domain-architecture class, length bin, and resolution bin.

Comparison	Max JSD ($\times 10^{-3}$)	Max TV ($\times 10^{-3}$)	Max bin deviation ($\times 10^{-3}$)
Train vs. source pool	0.012	0.68	0.48
Public val. vs. source pool	0.68	9.6	6.8
Hidden val. vs. source pool	0.44	4.4	3.4
Train vs. public val.	0.77	9.1	6.5
Train vs. hidden val.	0.57	4.6	3.1
Public vs. hidden val.	0.56	11.0	7.0

Embedding-space coverage. Metadata bins capture only the coarse structural features the field already knows to track, and two splits could match on secondary structure, length, and resolution while differing in the fine-grained sequence neighborhood structure that determines what a model actually sees during training. We embed each eligible chain with the 650M-parameter ESM2 variant (`esm2_t33_650M_UR50D`) as a mean-pooled vector $\phi(x) \in \mathbb{R}^{1280}$ and compare via cosine distance (Lin et al., 2023). Public and hidden validation are tightly matched in this space (Figure 2), with median nearest-train distances of 0.0283 and 0.0288, and 76.0% and 75.6% of examples lying within their local 50-neighbor radius of a training example; both sit closer to training than the broader non-train pool. Training is also not allocated to match raw PDB row frequencies: the densest density decile of the source pool receives only 0.09% of training chains, because the goal is broad coverage under strict leakage controls rather than reproduction of biological deposition history. Quantile breakdowns and coarse-cluster occupancy at $K = 100$ and $K = 200$ appear in Tables 3 and 4. **Conditional randomization.** The committed split is finally compared against $B = 1{,}000$ alternative valid splits drawn under the same official constraints (cluster disjointness, PDB-entry disjointness, sizes, reserve, stratification). It lies inside the central randomized range on every diagnostic and improves on the randomized median in coverage measures, with public-validation nearest-train $p_{50} = 0.0208$ versus randomized 0.0217, and $K = 100$ touched-cluster mass of 0.9082 versus randomized 0.9076. The full comparison is in Appendix A.1.2.

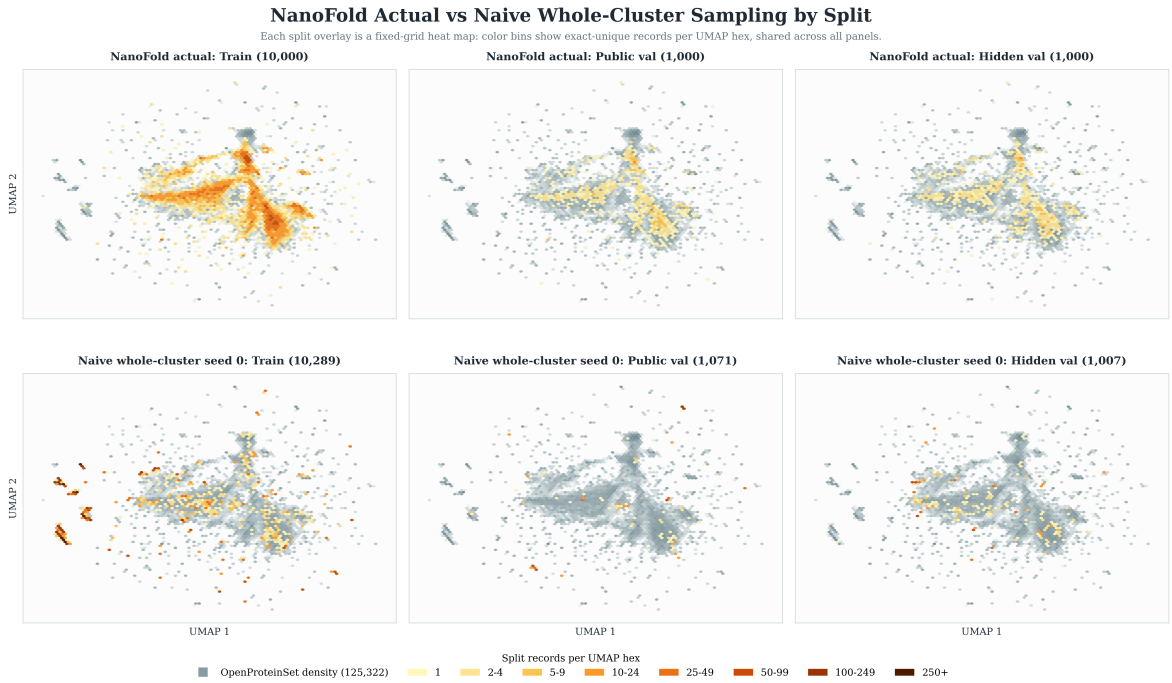


Figure 2. **Embedding-space coverage.** UMAP projection (McInnes et al., 2018) of 1280-dimensional ESM2 mean embeddings over the eligible OpenProteinSet-derived source pool. Columns compare NanoFold's actual split allocator with an equivalently sized split sampled by MMseqs2 cluster ID to produce disjoint clusters; rows show train, public-validation, and hidden-validation selections. The figure shows that, across all 3 sets, NanoFold's sampling approach provides a broader coverage of the OpenProteinSet universe, while cluster sampling is restricted to high density regions. UMAP is shown for qualitative reference; quantitative coverage is reported in Tables 3 and 4.

Together these diagnostics support the core data claim: training, public-validation, and hidden-validation are compositionally matched over biologically relevant metadata, training broadly covers the eligible pool in embedding space, and the committed split behaves like a typical draw from the official allocation procedure.

4. The NanoFold Benchmark

A fixed dataset becomes a benchmark only when paired with an evaluation protocol that asks well-posed questions. NanoFold asks three: which methods learn fastest, whether their early advantages persist under additional optimization, and what the best achievable accuracy is on the fixed corpus. Each is a separate track over the same training task, training rules, and scoring metric, differing only in compute budget and ranking rule.

4.1. Task and prediction contract

The task is to predict atom14 coordinates from sequence and the official MSA-derived features. Each chain x has length L_x, residue identities $a_{1:L_x}$, MSA features m_x, and atom14 supervision

$$y_x = \{Y_x, M_x\}, Y_x \in \mathbb{R}^{L_x \times 14 \times 3}, M_x \in \{0,1\}^{L_x \times 14},$$

where Y_x contains atom14 coordinates and M_x is the resolved-atom mask. A model f_θ takes the feature-side tensors and returns $\widehat{Y}_x = f_\theta(a_{1:L_x}, m_x) \in \mathbb{R}^{L_x \times 14 \times 3}$, supplied to the scorer as a single floating-point tensor `pred_atom14` of shape $(B, L, 14, 3)$ per batch.

4.2. Tracks and training rules

The three tracks are summarized in Table 2. The fixed-budget tracks (`limited` at 240,000 samples and `research_large` at 960,000) rank submissions by hidden FoldScore learning-curve area under the curve (AUC), rewarding methods that acquire useful geometry early; the `unlimited` track ranks by final hidden FoldScore. Official ranking is performed on the sealed hidden split, with the public validation split reserved for development.

The training rules keep the experimental object narrow. Official templates are disabled, external structure data is disallowed, external MSA generation is excluded from official runs, and models are trained from scratch on the fixed public training set. These rules isolate which architectures, losses, curricula, and biological priors most efficiently acquire transferable protein geometry under a small, fixed set of experimentally derived structures.

Table 2. **NanoFold tracks.** All tracks share the same training corpus, validation splits, prediction contract, and training rules. Fixed-budget tracks are ranked by hidden FoldScore AUC to reward early acquisition of useful geometry; the unlimited track is ranked by final hidden FoldScore.

Track	Sample budget	Rank metric	Scientific question
`limited`	240,000	Hidden FoldScore AUC	Which methods learn fastest?
`research_large`	960,000	Hidden FoldScore AUC	Do gains persist with more optimization?
`unlimited`	Unrestricted	Final hidden FoldScore	What is the best fixed-data final performance?

4.3. Scoring

FoldScore is adapted from the CASP15 tertiary-structure scoring framework (Simpkin et al., 2023), combining four families of measurement that can all be computed from the official atom14 prediction contract: global Cα fold accuracy via GDT_HA (Zemla, 2003); local atomic agreement via lDDT (Mariani et al., 2013), CAD (Olechnovic et al., 2013), and side-group and side-chain accuracy (SG, SC); backbone and dihedral consistency (BB, DipDiff); and steric plausibility via MolProbity-style clash validation (Chen et al., 2010). For a single chain,

$$\begin{aligned}\text{FoldScore} = {} & 0.25 \cdot \text{GDT_HA}_{C\alpha} \\ & + 0.09375 \cdot (\text{lDDT}_{\text{atom14}} + \text{CAD}_{aa} \\ & \quad + \text{SG} + \text{SC}) \\ & + 0.125 \cdot (\text{Clash} + \text{BB} + \text{DipDiff})\,.\end{aligned}$$

FoldScore introduces minor changes to CASP15's scoring in order to make it tractable for NanoFold. First, CASP15 ranks groups by per-target z-scores over the submitted model pool, with outlier removal and penalty clipping, rather than by an absolute per-model score. Such relative normalization would make a NanoFold score depend on the set of competing submissions and would change as new leaderboard entries or checkpoint curves are added. In addition, the CASP15 formula contains two terms outside NanoFold's prediction contract: ASE, which evaluates submitted per-residue confidence/pLDDT against observed local accuracy, and reLLG, which measures molecular-replacement utility through a specialized crystallographic assessment path. Requiring either would add submission channels or external scoring dependencies unrelated to the fixed atom14 coordinate task. FoldScore therefore keeps the CASP15 structure-derived components that are computable from `pred_atom14`, official residue identities, and atom14 masks, and renormalizes the CASP15 weights over that supported subset.

For fixed-budget tracks, models are evaluated at checkpoints indexed by cumulative samples $0 < s_1 < \cdots < s_K \leq B$, where B is the track budget. Letting F_k denote mean hidden FoldScore at checkpoint k,

$$\text{AUC}_{\text{FoldScore}} = \frac{1}{B} \sum_{k=1}^{K-1} \frac{F_k + F_{k+1}}{2} (s_{k+1} - s_k).$$

Final-score evaluation rewards models that eventually converge; the AUC ranking rewards methods that acquire structural quality early. Together with the sealed hidden split and the shared training rules, this scoring makes the leaderboard interpretable as a measurement of sample-efficient geometry learning.

5. Experiments

We use NanoFold to ask whether a deliberately small, leakage-controlled benchmark can reveal the same kinds of training behavior that motivate large-scale AF-style development. All runs train from scratch on the 10,000-chain public training split, evaluate on the 1,000-chain public-validation split, and use no hidden labels. Unless otherwise stated, runs use an effective batch size of 8, train for 30,000 optimizer steps, corresponding to 240,000 training samples, and evaluate public-validation FoldScore every 3,000 optimizer steps. Full hyperparameters and run definitions are given in Appendix A.2.

All experiments detailed below were executed, monitored, and analyzed using OpenAI's GPT-5.5 in the Codex harness. The AI agent was responsible for creating model configs, provisioning training infrastructure, launching the training runs, and then monitoring runs in a continuous loop. These autonomous experiments proved effective in interrogating AlphaFold's scaling laws and performing ablations to determine AlphaFold's core architectural features.

In addition, GPT-5.5 was leveraged to perform autoresearch using a baseline of AF2-medium on the NanoFold dataset. Over 151 experiments, the model was able to discover a set of architectural changes that improved both data efficiency and scaling performance when compared with the baseline. This experiment demonstrates that frontier AI models, when provided with the appropriate scaffolding, can act as effective researchers in AI-driven biology.

5.1. Ablation signals autonomously extracted with GPT-5.5

A benchmark intended for architecture and training studies should do more than rank finished models; it should

identify changes that can be recombined into better systems. We therefore ran medium-size ablations over three training primitives suggested by AF-style practice and by our early NanoFold runs: recycle policy, backbone-FAPE clamping, and the late fine-tuning loss phase. Figure 3 shows a compact representative set of these runs alongside the FAPE clamping sweep.

The ablations give a consistent signal. Relative to the medium AF2-default recipe, fully unclamping backbone FAPE produces a clear improvement in final public-validation FoldScore (0.484 versus 0.457). Fixed-four recycling gives a smaller but directionally useful gain (0.472). Removing the fine-tuning loss phase alone lowers FoldScore to 0.400, suggesting that the composite rank metric captures tradeoffs among global accuracy, local agreement, contact preservation, and physical plausibility.

Most importantly, these signals compose. A model that combines fixed four recycles, fully unclamped backbone FAPE, and no fine-tuning loss phase reaches 0.504 final public-validation FoldScore, making it the best core run in this study. This is the desired behavior for NanoFold as an experimental instrument: individual ablations provide information that is predictive enough to guide a follow-up configuration, rather than merely explaining results after the fact.

NanoFold also reproduces a known AF-style sensitivity to backbone-FAPE clamping policy at the 10,000-chain scale. AlphaFold2 uses a mixed scheme in which 90% of training mini-batches clamp backbone FAPE at 10Å and 10% leave it unclamped (Jumper et al., 2021), and OpenFold later reported that samplewise rather than batchwise clamping improved convergence (Ahdritz et al., 2024); both observations were made in substantially larger regimes than NanoFold. Panel B of Figure 3 shows that the same sensitivity is visible here. The AF2-default batchwise 90/10 policy, samplewise 90/10, and samplewise 50/50 policies perform similarly by final FoldScore (0.457, 0.455, and 0.457). The deterministic soft 90/10 mixture and unclamping only after the fine-tuning boundary are lower (0.424 and 0.419), while fully unclamped backbone FAPE reaches the highest final FoldScore in the sweep (0.484). The curves separate throughout training rather than only at the final score, indicating that NanoFold is sensitive to the optimization dynamics induced by the loss itself, not just to its endpoint.

5.2. Goal Mode architecture search produces SimplexFold

In addition to autonomously probing ablations, NanoFold's structure enables AI agents to explore new architectural ideas. To test the effectiveness of autoresearch on the NanoFold dataset, we leveraged GPT-5.5 in the Codex harness. We began by training AF2-medium on NanoFold as

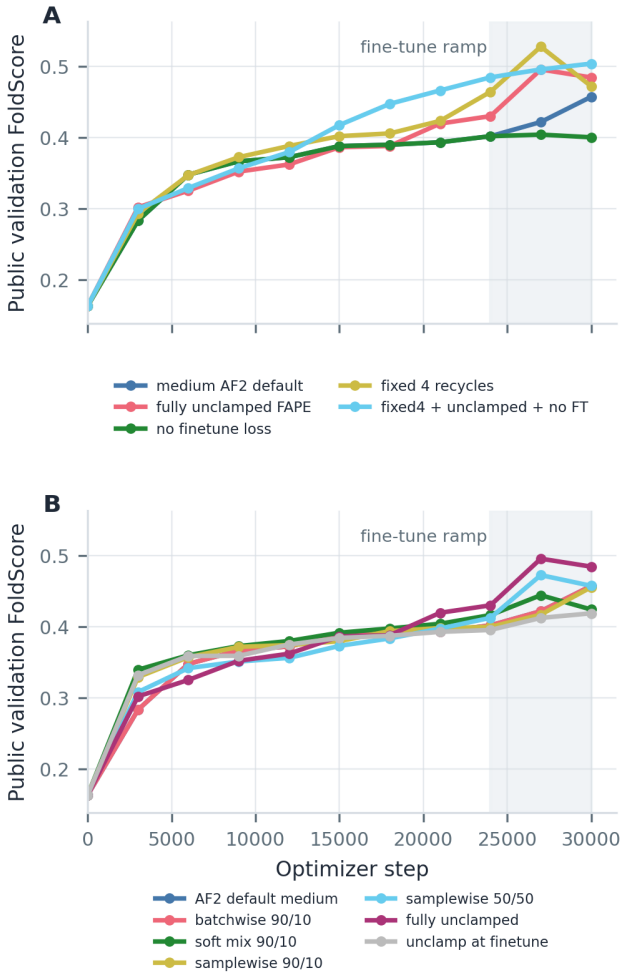


Figure 3. **NanoFold ablation signals.** Panel A shows representative medium-model ablations across recycle policy, backbone-FAPE clamping, and the late fine-tuning loss phase, with the best combined setting using fixed four-cycle recycling, fully unclamped backbone FAPE, and the initial training loss for the full run. Panel B shows that backbone-FAPE clamp policy strongly affects small-scale AF-style training. Public-validation curves compare AF2-default batchwise 90/10 clamping, deterministic soft mixing, samplewise clamping, delayed unclamping, and fully unclamped backbone FAPE. Shaded spans mark the fine-tune ramp.

an initial baseline. The goal of the AI agent was to discover novel architectural modifications to AlphaFold2 that would improve its final lDDT-Cα score.

The model was placed into an iterative loop through the Goal Mode feature; this forces the model to continue working until a specific criterion is met. Initial attempts were made to give high-level prompts to the model to improve the AF2-medium baseline against the final validation lDDT-Cα score, but this generally resulted in the model making minor changes to loss function weights and optimizer hyperparameters. On our most successful run, we seeded the model input prompt with an experimental idea direction,

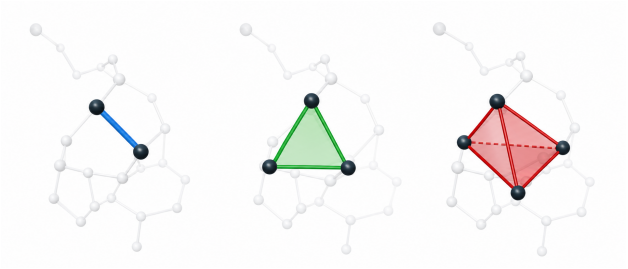


Figure 4. **SimplexFold augments pairwise protein representations with higher-order simplices.** The highlighted edge, triangular face, and tetrahedral cell illustrate the progression from 1-simplices between residue pairs to 2- and 3-simplices over residue triples and quadruples. SimplexFold uses these higher-order objects as explicit learned states for modeling local protein geometry beyond pairwise contacts.

which we dubbed **SimplexFold**. SimplexFold's aim was to be an extension of AlphaFold into higher-order topological spaces. That is, rather than only considering 1-simplices (i.e., pairs of residues) as is done in AlphaFold2's pair representation matrix, SimplexFold would now attempt to learn representations of 2-simplices (i.e., triangles of residues) and 3-simplices (i.e., tetrahedra of residues). GPT-5.5 was given this research idea along with the following prompt in Goal Mode:

> *`/goal` I'd like you to iterate on SimplexFold (parameter matched to AF2 medium) on the NanoFold dataset. We should aim for a validation lddt Calpha score that exceeds 0.7 while keeping parameters within 5% of the AF2 medium model. You can make any changes you'd like to the loss functions or architecture, but keep all changes in the spirit of the README in the SimplexFold repo. Record your high level plans in `PLAN.md`, all of your experiment ideas in `EXPERIMENTS.md`, and keep a running log of notes in `EXPERIMENTS_NOTES.md`*

The three memory files listed below provided persistent memory for the agent and enabled it to remain coherent across 151 experimental runs. The purpose of each memory file was as follows:

1. `PLAN.md`: for the agent to track the instructions and high-level goals that it placed for itself throughout the research loop.
2. `EXPERIMENT_INDEX.md`: for the agent to track which folders it was saving experimental results to and where it placed crucial scripts that it would need to reuse.
3. `RUNNING_NOTES.md`: for the agent to record which experiments it wanted to run, track the progress of running experiments, and take notes on any interesting or surprising items that arose during its autonomous loop.

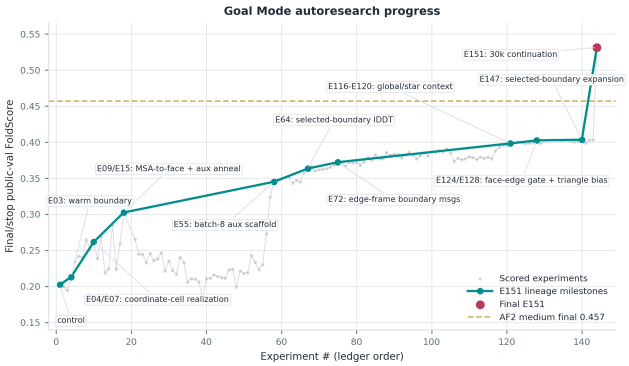


Figure 5. **Goal Mode autoresearch lineage.** The grey trace shows all scored SimplexFold experiments; the teal path follows the new-best lineage that fed E151, from the initial control through warm-boundary, coordinate-cell, MSA-to-face, selected-boundary, edge-frame, global/star-context, face-edge/triangle-bias, selected-boundary-expansion, and 30k-continuation steps. The dashed line marks the AF2-medium final public-validation FoldScore (0.457).

The autoresearch loop evaluated 151 SimplexFold experiments. Each completed run produced a scored result that the agent could compare with the current best configuration, while the persistent memory files preserved the hypotheses, implementation details, and outcomes needed to plan subsequent experiments. The agent leveraged these memory files to revisit successful configurations, continue promising branches, and discard changes that failed to improve validation performance. Figure 5 summarizes this experimental trajectory and the lineage that ultimately produced our best performing model, experiment E151.

The complete trajectory is deliberately irregular: most proposed configurations did not establish a new best result, and several promising directions produced gains only after being combined with changes discovered in earlier runs. The new-best lineage nevertheless shows sustained progress from the initial SimplexFold implementation through changes to boundary handling, coordinate and simplex representations, geometric updates, and continued training. This progression culminated in E151, the strongest configuration discovered during the Goal Mode run. We next compare that model directly with the AF2-medium baseline under the same NanoFold public-validation setting.

The final E151 configuration reaches a public-validation FoldScore of 0.533 after 30,000 optimizer steps, compared with 0.457 for the AF2-medium baseline, an absolute improvement of 0.076. It also exceeds the 0.504 FoldScore obtained by the strongest combined recipe ablation described above. The result therefore outperforms simple tweaks of the original AlphaFold2 recipe. The search instead produced a distinct architecture whose explicit higher-order representations yielded additional improvement beyond the gains recovered from recycling, FAPE, and loss-schedule

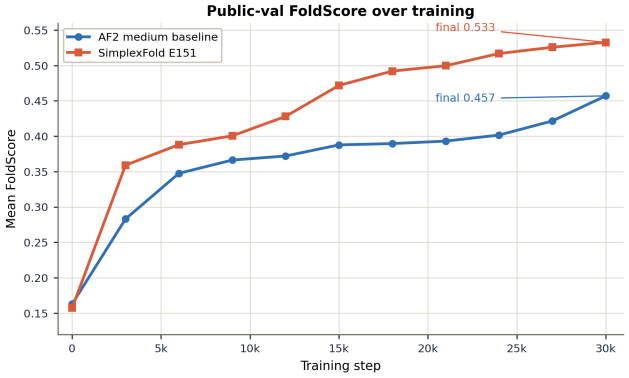


Figure 6. **Goal Mode architecture search produces SimplexFold.** A longer-running Codex Goal Mode loop used NanoFold to iterate over SimplexFold variants, continue promising branches, and reject weak ones. The final E151 SimplexFold public-validation curve reaches about 0.533 FoldScore at 30k steps, compared with 0.457 for the AF2-medium baseline under the same NanoFold public-validation setting.

changes alone.

Taken together, these experiments show that NanoFold can support autonomous research at two levels. Firstly, an agent can recover interpretable training signals through controlled ablations, and, secondly, it can use those signals within a longer-running loop to propose, reject, and refine new architectural ideas. These properties make NanoFold an ideal substrate for AI agents in biology, and its dataset construction properties provide a guide for future datasets in adjacent biological fields.

6. Discussion

NanoFold was designed to make controlled architectural comparison feasible at scales most research groups can run and AI agents can autonomously execute against, addressing a specific gap: AlphaFold-style systems improve faster than the field can attribute the improvements when production-scale training is the only available substrate. The experiments confirm the setting works as intended, with capacity, recycling depth, and recipe separating cleanly under matched conditions and ablation signals composing into stronger configurations. More notably, the design signals NanoFold returns at the 10,000-chain scale recover a production-scale sensitivity. AlphaFold2 documents a batchwise 90/10 backbone-FAPE clamping policy (Jumper et al., 2021), and OpenFold reports that changing this to samplewise clamping improves convergence (Ahdritz et al., 2024). NanoFold confirms that clamp policy matters, although it does not reproduce OpenFold's specific samplewise advantage: the samplewise and batchwise 90/10 runs are essentially tied here, while full unclamping performs best. We read this as evidence that small-scale, leakage-controlled benchmarks can carry meaningful architectural signal.

The SimplexFold autoresearch study shows why NanoFold is useful beyond static model ranking. Once the dataset, training code, and scoring function are fixed, the benchmark gives an autonomous research agent a structured environment for testing architectural hypotheses across many comparable iterations. Across 151 experiments, the strongest signal was the structure of the search process itself. Failed branches remained interpretable, promising branches could be extended, and persistent experiment memory helped turn incremental observations into a coherent sequence of design changes. The final E151 model improves over both the AF2-medium baseline and the strongest hand-composed recipe ablation, suggesting that NanoFold can support open-ended architecture search rather than only retrospective comparison among predefined variants. These results do not necessarily suggest that the E151 architecture will transfer directly to production-scale training. Rather, they show that NanoFold serves as a computationally tractable biological benchmark that can make autonomous architecture search experimentally grounded, reproducible, and measurable.

NanoFold's coverage is deliberately narrow. It targets single chains in the 40–256 residue range under the official MSA and chain quality filters, and excludes template ingestion, external MSA retrieval, multi-chain complexes, ligands, and conformational ensembles; it is also not a substitute for CASP or production-scale benchmarks such as PXMeter. The hidden split is compositionally comparable to the public split, so observed gaps reflect modeling differences and not distribution shift. We hope these boundaries point to natural extensions: multi-chain and ligand variants under the same verification methodology, adversarial hidden splits as a robustness test, AF3-scale variants of the construction, and the verified-split protocol as a backbone for related protein-structure benchmarks where biological dependencies in PDB-derived data make naive splits unreliable. The training corpus, evaluation infrastructure, and reference codebase are publicly released to support these directions.

7. Conclusion

NanoFold provides shared methodological infrastructure for AlphaFold-style research. NanoFold ties verified dataset construction to sealed evaluation tracks, reference training code, and automatic scoring, giving researchers a common basis for comparing modeling choices and giving AI agents a tractable setting for iterative protein-structure experiments. We hope it accelerates attribution and architecture search in protein structure prediction, and that its construction methodology informs benchmark design across computational biology where leakage and biological dependency remain persistent obstacles.

Acknowledgments

We thank Lyra Labs for providing the compute to run these experiments. We would also like to thank our anonymous reviewers for their time and constructive feedback, which greatly helped us improve the clarity and quality of our manuscript.

Software and Data

The training corpus, evaluation infrastructure, and reference codebase are publicly released to support the directions discussed in this paper.

Impact Statement

This paper presents work whose goal is to advance reproducible machine learning benchmark design for protein structure prediction. Potential societal consequences are primarily those associated with improved scientific tooling for structural biology; we do not identify specific additional societal impacts requiring separate highlighting.

References

Abramson, J., Adler, J., Dunger, J., Evans, R., Green, T., Pritzel, A., Ronneberger, O., Willmore, L., Ballard, A. J., Bambrick, J., et al. Accurate structure prediction of biomolecular interactions with AlphaFold 3. *Nature*, 630 (8016):493–500, 2024. doi: 10.1038/s41586-024-07487-w.

Ahdritz, G., Bouatta, N., Kadyan, S., Jarosch, L., Berenberg, D., Fisk, I., Watkins, A. M., Ra, S., Bonneau, R., and AlQuraishi, M. OpenProteinSet: Training data for structural biology at scale. In *Advances in Neural Information Processing Systems 36*, pp. 4597–4609, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/0eb82171240776fe19da498bef3b1abe-Abstract-Datasets_and_Benchmarks.html.

Ahdritz, G., Bouatta, N., Floristean, C., Kadyan, S., Xia, Q., Gerecke, W., O'Donnell, T. J., Berenberg, D., Fisk, I., Zanichelli, N., et al. OpenFold: retraining AlphaFold2 yields new insights into its learning mechanisms and capacity for generalization. *Nature Methods*, 21(8):1514–1524, 2024. doi: 10.1038/s41592-024-02272-z.

AlQuraishi, M. ProteinNet: a standardized data set for machine learning of protein structure. *BMC Bioinformatics*, 20(1):311, 2019. doi: 10.1186/s12859-019-2932-0.

Baek, M., DiMaio, F., Anishchenko, I., Dauparas, J., Ovchinnikov, S., Lee, G. R., Wang, J., Cong, Q., Kinch, L. N., Schaeffer, R. D., et al. Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, 373(6557):871–876, 2021. doi: 10.1126/science.abj8754.

Burley, S. K., Bhikadiya, C., Bi, C., Bittrich, S., Chen, L., Crichlow, G. V., Christie, C. H., Dalenberg, K., Di Costanzo, L., Duarte, J. M., et al. RCSB protein data bank: powerful new tools for exploring 3d structures of biological macromolecules for basic and applied research and education in fundamental biology, biomedicine, biotechnology, bioengineering and energy sciences. *Nucleic Acids Research*, 49(D1):D437–D451, 2021. doi: 10.1093/nar/gkaa1038.

ByteDance AML AI4Science Team, Chen, X., Zhang, Y., Lu, C., Ma, W., Guan, J., Gong, C., Yang, J., Zhang, H., Zhang, K., Wu, S., Zhou, K., Yang, Y., Liu, Z., Wang, L., Shi, B., Shi, S., and Xiao, W. Protenix: Advancing structure prediction through a comprehensive AlphaFold3 reproduction. *bioRxiv*, 2025. doi: 10.1101/2025.01.08.631967. URL https://www.biorxiv.org/content/early/2025/01/11/2025.01.08.631967.

Chai Discovery, Boitreaud, J., Dent, J., McPartlon, M., Meier, J., Reis, V., Rogozhnikov, A., and Wu, K. Chai-1: Decoding the molecular interactions of life. *bioRxiv*, 2024. doi: 10.1101/2024.10.10.615955. URL https://www.biorxiv.org/content/early/2024/10/11/2024.10.10.615955.

Chandonia, J.-M., Guan, L., Lin, S., Yu, C., Fox, N. K., and Brenner, S. E. SCOPe: improvements to the structural classification of proteins–extended database to facilitate variant interpretation and machine learning. *Nucleic Acids Research*, 50(D1):D553–D559, 2022. doi: 10.1093/nar/gkab1054.

Chen, V. B., Arendall, W. B., Headd, J. J., Keedy, D. A., Immormino, R. M., Kapral, G. J., Murray, L. W., Richardson, J. S., and Richardson, D. C. MolProbity: all-atom structure validation for macromolecular crystallography. *Acta Crystallographica Section D: Biological Crystallography*, 66(1):12–21, 2010. doi: 10.1107/S0907444909042073.

Cheng, H., Schaeffer, R. D., Liao, Y., Kinch, L. N., Pei, J., Shi, S., Kim, B.-H., and Grishin, N. V. ECOD: an evolutionary classification of protein domains. *PLoS Computational Biology*, 10(12):e1003926, 2014. doi: 10.1371/journal.pcbi.1003926.

Didi, K., Zhang, Z., Zhou, G., Reidenbach, D., Cao, Z., Cha, S., Geffner, T., Dallago, C., Tang, J., Bronstein, M. M., Steinegger, M., Kucukbenli, E., Vahdat, A., and Kreis, K. Scaling atomistic protein binder design with generative pretraining and test-time compute. In *The Fourteenth*

International Conference on Learning Representations, 2026. URL https://openreview.net/forum?id=qmCpJtFZra.

Fu, L., Niu, B., Zhu, Z., Wu, S., and Li, W. CD-HIT: accelerated for clustering the next-generation sequencing data. *Bioinformatics*, 28(23):3150–3152, 2012. doi: 10.1093/bioinformatics/bts565.

Haas, J., Barbato, A., Behringer, D., Studer, G., Roth, S., Bertoni, M., Mostaguir, K., Gumienny, R., and Schwede, T. Continuous automated model evaluation (CAMEO) complementing the critical assessment of structure prediction in CASP12. *Proteins: Structure, Function, and Bioinformatics*, 86(S1):387–398, 2018. doi: 10.1002/prot.25431.

Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M., and Leskovec, J. Open graph benchmark: Datasets for machine learning on graphs. In *Advances in Neural Information Processing Systems 33*, pp. 22118–22133, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/fb60d411a5c5b72b2e7d3527cfc84fd0-Abstract.html.

Huang, K., Fu, T., Gao, W., Zhao, Y., Roohani, Y. H., Leskovec, J., Coley, C. W., Xiao, C., Sun, J., and Zitnik, M. Therapeutics data commons: Machine learning datasets and tasks for drug discovery and development. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, 2021. URL https://openreview.net/forum?id=8nvgnORnoWr.

Huang, P.-S., Boyken, S. E., and Baker, D. The coming of age of de novo protein design. *Nature*, 537(7620): 320–327, 2016. doi: 10.1038/nature19946.

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., et al. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, 2021. doi: 10.1038/s41586-021-03819-2.

Kabsch, W. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A*, 32(5): 922–923, 1976. doi: 10.1107/S0567739476001873.

Kryshtafovych, A., Schwede, T., Topf, M., Fidelis, K., and Moult, J. Critical assessment of methods of protein structure prediction (CASP)–round XV. *Proteins: Structure, Function, and Bioinformatics*, 91(12):1539–1549, 2023. doi: 10.1002/prot.26617.

Kryshtafovych, A., Schwede, T., Topf, M., Fidelis, K., and Moult, J. Progress and bottlenecks for deep learning in computational structure biology: CASP round XVI. *Proteins: Structure, Function, and Bioinformatics*, 94(1): 5–14, 2026. doi: 10.1002/prot.70076.

La, H., Gupta, A., Morehead, A., Cheng, J., and Zhang, M. MegaFold: Efficient training of next-generation 3d attention protein models on cross-platform GPUs. *arXiv preprint arXiv:2506.20686*, 2025. doi: 10.48550/arXiv.2506.20686. URL https://arxiv.org/abs/2506.20686. Version 2, revised 13 June 2026.

Lin, Z., Akin, H., Rao, R., Hie, B., Zhu, Z., Lu, W., Smetanin, N., Verkuil, R., Kabeli, O., Shmueli, Y., et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637): 1123–1130, 2023. doi: 10.1126/science.ade2574.

Ma, W., Liu, Z., Yang, J., Lu, C., Zhang, H., and Xiao, W. From dataset curation to unified evaluation: Revisiting structure prediction benchmarks with PXMeter. *bioRxiv*, 2025. doi: 10.1101/2025.07.17.664878. URL https://www.biorxiv.org/content/early/2025/07/22/2025.07.17.664878.

Mariani, V., Biasini, M., Barbato, A., and Schwede, T. lDDT: a local superposition-free score for comparing protein structures and models using distance difference tests. *Bioinformatics*, 29(21):2722–2728, 2013. doi: 10.1093/bioinformatics/btt473.

McInnes, L., Healy, J., and Melville, J. UMAP: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018. doi: 10.48550/arXiv.1802.03426. URL https://arxiv.org/abs/1802.03426.

Olechnovic, K., Kulberkyte, E., and Venclovas, C. CAD-score: a new contact area difference-based function for evaluation of protein structural models. *Proteins: Structure, Function, and Bioinformatics*, 81(1):149–162, 2013. doi: 10.1002/prot.24172.

Passaro, S., Corso, G., Wohlwend, J., Reveiz, M., Thaler, S., Somnath, V. R., Getz, N., Portnoi, T., Roy, J., Stark, H., Kwabi-Addo, D., Beaini, D., Jaakkola, T., and Barzilay, R. Boltz-2: Towards accurate and efficient binding affinity prediction. *bioRxiv*, 2025. doi: 10.1101/2025.06.14.659707. URL https://doi.org/10.1101/2025.06.14.659707.

Sadybekov, A. V. and Katritch, V. Computational approaches streamlining drug discovery. *Nature*, 616(7958): 673–685, 2023. doi: 10.1038/s41586-023-05905-z.

Sillitoe, I., Bordin, N., Dawson, N., Waman, V. P., Ashford, P., Scholes, H. M., Pang, C. S. M., Woodridge, L., Rauer, C., Sen, N., Abbasian, M., Le Cornu, S., Lam, S. D., Berka, K., Varekova, I. H., Svobodova, R., Lees, J., and

Orengo, C. A. CATH: increased structural coverage of functional space. *Nucleic Acids Research*, 49(D1):D266–D273, 2021. doi: 10.1093/nar/gkaa1079.

Simpkin, A. J., Mesdaghi, S., Sánchez Rodríguez, F., Elliott, L., Murphy, D. L., Kryshtafovych, A., Keegan, R. M., and Rigden, D. J. Tertiary structure assessment at CASP15. *Proteins: Structure, Function, and Bioinformatics*, 91 (12):1616–1635, 2023. doi: 10.1002/prot.26593.

Sorscher, B., Geirhos, R., Shekhar, S., Ganguli, S., and Morcos, A. S. Beyond neural scaling laws: beating power law scaling via data pruning. In *Advances in Neural Information Processing Systems 35*, pp. 19523–19536, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/7b75da9b61eda40fa35453ee5d077df6-Abstract-Conference.html.

Steinegger, M. and Söding, J. MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature Biotechnology*, 35(11):1026–1028, 2017. doi: 10.1038/nbt.3988.

Suzek, B. E., Wang, Y., Huang, H., McGarvey, P. B., Wu, C. H., and UniProt Consortium. UniRef clusters: a comprehensive and scalable alternative for improving sequence similarity searches. *Bioinformatics*, 31(6):926–932, 2015. doi: 10.1093/bioinformatics/btu739.

The OpenFold3 Team. OpenFold3-preview2 technical report. https://portal.openfold.omsf.io/reports/of3p2_technical_report.pdf, 2026. Research preview technical report.

Varadi, M., Anyango, S., Deshpande, M., Nair, S., Natassia, C., Yordanova, G., Yuan, D., Stroe, O., Wood, G., Laydon, A., et al. AlphaFold protein structure database: massively expanding the structural coverage of protein-sequence space with high-accuracy models. *Nucleic Acids Research*, 50(D1):D439–D444, 2022. doi: 10.1093/nar/gkab1061.

Watson, J. L., Juergens, D., Bennett, N. R., Trippe, B. L., Yim, J., Eisenach, H. E., Ahern, W., Borst, A. J., Ragotte, R. J., Milles, L. F., et al. De novo design of protein structure and function with RFdiffusion. *Nature*, 620 (7976):1089–1100, 2023. doi: 10.1038/s41586-023-06415-8.

Westbrook, J. D., Young, J. Y., Shao, C., Feng, Z., Guranovic, V., Lawson, C. L., Vallat, B., Adams, P. D., Berrisford, J. M., Bricogne, G., et al. PDBx/mmCIF ecosystem: Foundational semantic tools for structural biology. *Journal of Molecular Biology*, 434(11):167599, 2022. doi: 10.1016/j.jmb.2022.167599.

Wohlwend, J., Corso, G., Passaro, S., Getz, N., Reveiz, M., Leidal, K., Swiderski, W., Atkinson, L., Portnoi, T., Chinn, I., Silterra, J., Jaakkola, T., and Barzilay, R. Boltz-1: Democratizing biomolecular interaction modeling. *bioRxiv*, 2024. doi: 10.1101/2024.11.19.624167. URL https://doi.org/10.1101/2024.11.19.624167.

Wu, Z., Ramsundar, B., Feinberg, E. N., Gomes, J., Geniesse, C., Pappu, A. S., Leswing, K., and Pande, V. MoleculeNet: a benchmark for molecular machine learning. *Chemical Science*, 9(2):513–530, 2018. doi: 10.1039/C7SC02664A.

wwPDB Consortium. Protein data bank: the single global archive for 3d macromolecular structure data. *Nucleic Acids Research*, 47(D1):D520–D528, 2019. doi: 10.1093/nar/gky949.

Zemla, A. LGA: a method for finding 3d similarities in protein structures. *Nucleic Acids Research*, 31(13):3370–3374, 2003. doi: 10.1093/nar/gkg571.

Zhu, F., Nowaczynski, A., Li, R., Xin, J., Song, Y., Marcinkiewicz, M., Eryilmaz, S. B., Yang, J., and Andersch, M. ScaleFold: Reducing AlphaFold initial training time to 10 hours. *arXiv preprint arXiv:2404.11068*, 2024. doi: 10.48550/arXiv.2404.11068. URL https://arxiv.org/abs/2404.11068.

A. Technical appendices and supplementary material

A.1. Supplementary Dataset Details

A.1.1. Diagnostic formulas

This appendix gives the exact diagnostics used in Section 3.3. For a categorical metadata field with bucket set $\mathcal{B}$, let $p_S(b)$ be the empirical fraction of chains in split S that fall in bucket $b \in \mathcal{B}$, and let $p_R(b)$ be the corresponding reference distribution, either the eligible source pool or another split. We report Jensen–Shannon divergence,

$$\mathrm{JSD}(p_S, p_R) = \frac{1}{2}\mathrm{KL}(p_S \,\|\, m) + \frac{1}{2}\mathrm{KL}(p_R \,\|\, m), \qquad m = \frac{1}{2}(p_S + p_R),$$

with zero-probability terms omitted in the usual way. We also report total variation distance,

$$\mathrm{TV}(p_S, p_R) = \frac{1}{2}\sum_{b\in\mathcal{B}} |p_S(b) - p_R(b)|,$$

and maximum bin deviation,

$$\Delta_{\max}(p_S, p_R) = \max_{b\in\mathcal{B}} |p_S(b) - p_R(b)|.$$

The embedding-space diagnostics use cosine distance in the 1280-dimensional ESM2 mean-embedding space. For chain x, the nearest-train distance is

$$r_T(x) = \min_{t\in T} d_{\cos}(\phi(x), \phi(t)),$$

where T is the public training split and $\phi(x)$ is the mean-pooled ESM2 embedding. To normalize for local density, let $\rho_k(x)$ be the distance from x to its kth nearest neighbor in the eligible source pool, excluding x itself. The density-normalized gap is

$$g_{T,k}(x) = \frac{r_T(x)}{\rho_k(x)}.$$

Coverage at a local radius is reported as

$$C_k(S;\alpha) = \frac{1}{|S|}\sum_{x\in S} \mathbf{1}\{r_T(x) \le \alpha\rho_k(x)\},$$

with $\alpha \in \{1, 2\}$ in the reported tables. Coarse-region coverage clusters ESM2 embeddings into K regions and reports both the fraction of clusters touched by a split and the fraction of source-pool mass contained in touched clusters.

A.1.2. Conditional randomization protocol

The randomization analysis compares the committed public split against alternative valid splits drawn under the same constraints. Each randomized split preserves the official candidate universe, chain eligibility filters, unit-level assignment, PDB-entry disjointness, MMseqs2 cluster disjointness, target train and public-validation sizes, hidden-sized reserve, and the same stratification fields used by the committed allocator. For a statistic Q, the reported empirical percentile is

$$\widehat{P}_{\le}(Q_{\mathrm{obs}}) = \frac{1}{B}\sum_{b=1}^{B} \mathbf{1}\{Q_b \le Q_{\mathrm{obs}}\}, \qquad B = 1000.$$

This is a conditional randomization check, not a claim that NanoFold is an IID sample from all protein space. The null distribution is the set of valid splits under the same construction rules. The purpose is to verify that the committed split is not an unusually easy, unusually hard, or unusually unbalanced draw from its own design procedure.

Conditional randomization. Finally, the committed NanoFold split is compared against alternative valid splits drawn under the same official constraints. We generate $B = 1{,}000$ alternative public train and public-validation splits respecting the same candidate universe, sequence-cluster disjointness, PDB-entry disjointness, train and validation sizes, hidden-sized reserve, and stratification fields. For a diagnostic statistic Q, the empirical percentile $\widehat{P}_{\le} = \frac{1}{B}\sum_b \mathbf{1}\{Q_b \le Q_{\mathrm{obs}}\}$ places the committed split within the distribution of valid randomized alternatives. Table 5 reports the comparison. Across every diagnostic, the committed split lies inside the central randomized range and improves on the randomized median in coverage measures, with nearest-train p50 of 0.0208 versus randomized 0.0217, p95 of 0.0580 versus 0.0627, public-validation coverage at ρ_{50} of 0.6550 versus 0.6416, and $K = 100$ touched-cluster mass of 0.9082 versus 0.9076.

Table 3. **Nearest-train coverage in ESM2 embedding space.** Cosine distances in the 1280-dimensional ESM2 mean-embedding space, with density-normalized gap $g_{T,50}(x) = r_T(x)/\rho_{50}(x)$.

	Nearest-train distance			Density-normalized gap ($k = 50$)		
Set	p50	p90	p95	p50	Within ρ_{50}	Within $2\rho_{50}$
Non-train pool	0.0348	0.0851	0.1050	1.090	0.414	0.731
Public validation	0.0283	0.0584	0.0771	0.894	0.760	0.996
Hidden validation	0.0288	0.0666	0.0844	0.897	0.756	0.996

Table 4. **Coarse ESM2 cluster occupancy.** Fraction of clusters touched by each split, and source-pool mass covered, at $K = 100$ and $K = 200$ regions.

	$K = 100$		$K = 200$	
Split	Clusters	Mass	Clusters	Mass
Train	0.850	0.867	0.830	0.826
Public validation	0.660	0.779	0.590	0.699
Hidden validation	0.650	0.788	0.590	0.707

A.1.3. Standard disjoint-cluster baseline

The whole-cluster baseline in Figure 2 is a deliberately simpler alternative to the official allocator. It samples disjoint MMseqs/PDB-connected units into train and validation splits without the official density-aware and structural balancing procedure. This baseline is useful because it satisfies the same high-level leakage constraint but does not actively control how limited split budgets are distributed across the eligible protein universe. The figure overlays both procedures on the same UMAP coordinates and fixed-grid hex bins. Gray points show the broader OpenProteinSet/PDB universe, while colored hexes show exact-unique records per UMAP bin for the selected split. The comparison demonstrates that disjointness alone is insufficient: a standard whole-cluster procedure can satisfy leakage rules while leaving visibly different density and coverage patterns.

A.2. Paper Experiment Protocol

A.2.1. Scope of the reported experiments

The experiments reported in the paper are paper-only public train/public validation studies. They are not official leaderboard submissions and do not use the sealed hidden validation labels. This distinction matters: the experiments are intended to demonstrate that NanoFold exposes interpretable training signals, not to claim competition performance. All reported learning curves in the core paper use the public validation split.

The core manuscript figures use the following analysis panels:

- **Scaling:** completed tiny and medium minAlphaFold2 runs under the corrected AF2-style default training recipe.
- **FAPE policy:** medium-model backbone-FAPE clamp/unclamp policy ablations.
- **Recycling policy:** medium-model recycling-depth and fixed-vs-sampled recycling ablations.
- **Representative ablations:** a compact cross-ablation panel containing the medium default, fully unclamped FAPE, no fine-tune loss, fixed-four recycling, and the best combined setting.

Other exploratory and diagnostic runs are not included in the core production paper figures described here.

A.2.2. Common data and training setup

All core training runs use the same public NanoFold train and validation splits:

$$|T| = 10{,}000, \qquad |V_{\text{pub}}| = 1{,}000.$$

Table 5. **Conditional randomization diagnostics.** Committed public split compared with 1,000 alternative valid splits sampled under the same official constraints. Lower is better for JSD and nearest-train distance; higher is better for coverage measures.

Metric	Observed	Random p05	Random p50	Random p95
Public max JSD ($\times 10^{-3}$)	0.70	0.42	0.42	0.42
Public val. nearest-train p50	0.0208	0.0209	0.0217	0.0224
Public val. nearest-train p95	0.0580	0.0580	0.0627	0.0683
Public val. coverage at ρ_{50}	0.6550	0.6176	0.6416	0.6657
Non-train coverage at ρ_{50}	0.5497	0.5434	0.5481	0.5519
$K = 100$ touched-cluster mass	0.9082	0.8933	0.9076	0.9138

The hidden validation split is disabled for these paper-only runs. Inputs are cropped to 256 residues, with random crops during training and center crops during validation. Training MSA rows are sampled randomly; validation uses the top MSA rows. Unless explicitly ablated, runs use MSA depth 192, extra MSA depth 64, batch size 1, gradient accumulation over 8 microbatches, Adam with learning rate 10^{-3}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-6}$, no weight decay, global gradient clipping at 0.1, and no mixed precision.

Every core run is trained for 30,000 optimizer steps. With batch size 1 and 8-step gradient accumulation, this corresponds to

$$30{,}000 \times 8 = 240{,}000$$

training samples. We evaluate public validation every 3,000 optimizer steps and save checkpoints at the same interval. The checkpoint set used for curves is

$$0, 3000, 6000, 9000, 12000, 15000, 18000, 21000, 24000, 27000, 30000.$$

All curves in the core paper are therefore directly comparable in optimizer-step and sample-budget units.

Table 6 summarizes the shared configuration. All stochastic runs use seed 0.

Table 6. **Shared training configuration for core public-validation experiments.**

Setting	Value
Training chains	10,000 public training chains
Validation chains	1,000 public-validation chains
Hidden validation	Not used in paper-only experiments
Crop length	256 residues
Training crop	Random contiguous crop
Validation crop	Center crop
MSA rows	192 primary MSA rows, 64 extra MSA rows
MSA sampling	Random rows during training, top rows during validation
Microbatch size	1 chain
Gradient accumulation	8 microbatches
Effective batch size	8 chains per optimizer step
Optimizer	Adam
Learning rate	10^{-3}
Adam coefficients	$\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-6}$
Weight decay	0
Gradient clipping	Global norm clipped to 0.1
Mixed precision	Disabled
Training length	30,000 optimizer steps, or 240,000 samples
Evaluation cadence	Every 3,000 optimizer steps
Warmup	333 optimizer steps
Learning-rate decay	Factor 0.95 at step 16,695
Default fine-tune phase	Starts at step 26,088; ramps over 2,250 steps
Default fine-tune LR scale	0.5 after the fine-tune boundary

A.2.3. Model profiles

The experiments use minAlphaFold2 profiles derived from AlphaFold-style components but reduced to make repeated small-data training feasible. Table 7 summarizes the profiles that appear in the core figures. The medium profile is the default ablation size. The tiny profile is included only in the scaling panel.

Table 7. **minAlphaFold2 profiles used in core figures.** Parameter counts are measured by the NanoFold runner.

Profile	Parameters	Evoformer blocks	Default max recycles
tiny	88,898	1	1
medium	3,106,642	4	2

The medium profile uses MSA channel 128, single channel 192, pair channel 64, four Evoformer blocks, a four-layer structure module, and eight IPA heads. The tiny profile uses MSA channel 32, single channel 32, pair channel 16, one Evoformer block, a two-layer structure module, and four IPA heads.

A.2.4. Learning-rate and fine-tuning schedule

The schedule follows the AlphaFold two-stage recipe (Jumper et al., 2021; Ahdritz et al., 2024) scaled to the 30,000-step paper budget. For the default schedule, the initial objective is used until step 26,088. Fine-tuning begins at step 26,088 and ramps over 2,250 steps, becoming fully active at step 28,338. The learning rate warms up for 333 steps, decays by a factor of 0.95 at step 16,695, and is multiplied by 0.5 after the fine-tuning boundary. The no-fine-tune ablation sets the fine-tune start to the final step and keeps the initial loss active throughout all optimizer updates.

A.2.5. Training loss

The initial loss is NanoFold's AlphaFold-inspired supervised objective, adapted from AlphaFold2 and OpenFold (Jumper et al., 2021; Ahdritz et al., 2024). The displayed weights describe NanoFold's implementation rather than a term-by-term transcription of the cited objectives; in particular, NanoFold logs and weights intermediate backbone-FAPE and torsion terms separately.

$$\begin{aligned}\mathcal{L}_{\text{init}} = 0.5\,\mathcal{L}_{\text{aux-bbFAPE}} + 0.5\,\mathcal{L}_{\text{allatomFAPE}} + \mathcal{L}_{\text{torsion}} \\ + 0.3\,\mathcal{L}_{\text{distogram}} + 2.0\,\mathcal{L}_{\text{maskedMSA}} + 0.01\,\mathcal{L}_{\text{pLDDT}}.\end{aligned}$$

Here $\mathcal{L}_{\text{aux-bbFAPE}}$ is the per-structure-module-iteration backbone FAPE, $\mathcal{L}_{\text{allatomFAPE}}$ is the final all-atom FAPE term, $\mathcal{L}_{\text{torsion}}$ includes the torsion-angle and angle-normalization terms, $\mathcal{L}_{\text{distogram}}$ is the Cβ-Cβ distance-bin cross entropy, $\mathcal{L}_{\text{maskedMSA}}$ is the BERT-style MSA reconstruction loss, and $\mathcal{L}_{\text{pLDDT}}$ is the pLDDT confidence-head loss.

After the fine-tuning ramp is fully active, the fine-tuning objective is

$$\mathcal{L}_{\text{ft}} = \mathcal{L}_{\text{init}} + 1.0\,\mathcal{L}_{\text{viol}} + 0.01\,\mathcal{L}_{\text{exp}} + 0.1\,\mathcal{L}_{\text{PAE}},$$

where $\mathcal{L}_{\text{viol}}$ is the structural violation loss, $\mathcal{L}_{\text{exp}}$ is the experimentally resolved atom loss, and $\mathcal{L}_{\text{PAE}}$ is the predicted-aligned-error/TM head loss. During the ramp interval, the runner blends the initial and fine-tuning losses and logs the ramp weight. All component losses, weighted component losses, train loss, validation loss, public validation lDDT-Cα, RMSD-Cα, and FoldScore components are saved when available.

A.2.6. FAPE clamp/unclamp policies

Let $\mathcal{L}^{\text{clamp}}_{\text{bbFAPE}}$ denote backbone FAPE with the 10Å distance clamp and $\mathcal{L}^{\text{unclamp}}_{\text{bbFAPE}}$ denote the unclamped backbone FAPE. The implementation writes the active clamp weight as w and evaluates

$$w\,\mathcal{L}^{\text{clamp}}_{\text{bbFAPE}} + (1-w)\,\mathcal{L}^{\text{unclamp}}_{\text{bbFAPE}}.$$

Side-chain/all-atom FAPE remains clamped. The core paper compares the following policies:

- **AF2 default / batchwise 90/10:** training samples a scalar $w \in \{0, 1\}$ per optimizer step with $\Pr(w = 1) = 0.9$; validation uses the deterministic expectation $w = 0.9$.

- **Soft mix 90/10:** training and validation always use $w = 0.9$, directly optimizing the expectation of the clamped/unclamped mixture.
- **Samplewise 90/10 and 50/50:** training samples w independently per example with clamped probability 0.9 or 0.5; validation uses the corresponding deterministic expectation.
- **Fully unclamped:** training and validation use $w = 0$.
- **Unclamp at fine-tune:** training is fully clamped before the fine-tuning boundary and fully unclamped after the fine-tuning boundary.

A.2.7. Recycling policies

The AF2-style default samples the number of training recycles uniformly from $1, \ldots, N_{\max}$ and evaluates deterministically at $N_{\max}$. Thus the medium default samples 1 or 2 recycles during training and evaluates at 2 recycles. The recycling sweep sets $N_{\max} \in \{2, 4, 6, 8\}$ with the same sampled training rule and fixed-max evaluation rule. The explicit fixed-four ablation trains and evaluates with exactly 4 recycles, without train-time recycle sampling.

A.2.8. Core run definitions

Table 8 gives the full experimental definition for every run used in the core public-validation learning-curve figures. "Sampled" recycling means that the number of training recycles is sampled uniformly from $1, \ldots, N_{\max}$, while validation uses $N_{\max}$. "Fixed" means the same recycle count is used during both training and validation. Unless noted otherwise, the fine-tune schedule is the default two-stage schedule in Appendix A.2.4.

Table 8. **Run definitions for the core ablation figures.**

Run	Profile	Training recycles	Validation recycles	Backbone FAPE / loss schedule
tiny AF2 default	tiny	sampled 1	1	batchwise 90/10; default fine-tune
medium AF2 default	medium	sampled 1..2	2	batchwise 90/10; default fine-tune
FAPE batchwise 90/10	medium	sampled 1..2	2	batchwise 90/10; default fine-tune
FAPE soft mix 90/10	medium	sampled 1..2	2	deterministic 90/10 mix; default fine-tune
FAPE samplewise 90/10	medium	sampled 1..2	2	samplewise 90/10; default fine-tune
FAPE samplewise 50/50	medium	sampled 1..2	2	samplewise 50/50; default fine-tune
FAPE fully unclamped	medium	sampled 1..2	2	fully unclamped; default fine-tune
FAPE unclamp at fine-tune	medium	sampled 1..2	2	clamped before fine-tune, unclamped after
one cycle	medium	fixed 1	1	batchwise 90/10; default fine-tune
sampled 1..4 recycles	medium	sampled 1..4	4	batchwise 90/10; default fine-tune
sampled 1..6 recycles	medium	sampled 1..6	6	batchwise 90/10; default fine-tune
sampled 1..8 recycles	medium	sampled 1..8	8	batchwise 90/10; default fine-tune
fixed 4 recycles	medium	fixed 4	4	batchwise 90/10; default fine-tune
no fine-tune loss	medium	sampled 1..2	2	batchwise 90/10; initial loss for all steps
fixed4 + unclamped + no FT	medium	fixed 4	4	fully unclamped; initial loss for all steps

A.3. Core Experimental Results

Table 9 lists all runs used by the core public-validation learning-curve figures. The final public validation lDDT-Cα and RMSD-Cα values are the last points plotted in the manuscript curves. We report lDDT-Cα and RMSD-Cα because these quantities are available for every core run and because the manuscript learning curves use lDDT-Cα consistently across all ablations.

Interpretation of the core runs. The scale panel shows that NanoFold is learnable and not saturated: medium improves over tiny under the same data and sample budget. The FAPE panel isolates the largest objective-level signal in these experiments. Fully unclamped backbone FAPE substantially improves both lDDT-Cα and RMSD-Cα, while partial unclamping gives intermediate behavior. The recycling panel shows a modest but coherent benefit from using more recycles, with fixed-four recycling outperforming sampled-four under this small-data setup. The representative panel combines these observations and shows that the strongest observed setting among the core experiments is the medium model with fixed four-cycle recycling, fully unclamped backbone FAPE, and no fine-tuning-loss phase.

Table 9. **Core paper public-validation runs.** All runs use 30,000 optimizer steps and 240,000 training samples. Wall time is measured by the Runpod H100 runner.

Run	Main difference from medium default	lDDT-Cα	RMSD-Cα	H100 h
tiny AF2 default	tiny profile, max recycle 1	0.235	25.35	10.02
medium AF2 default	medium profile, sampled 1..2 recycles, batchwise FAPE	0.293	19.56	15.75
FAPE batchwise 90/10	AF2-default comparator	0.293	19.56	16.55
FAPE soft mix 90/10	deterministic 90/10 FAPE mixture	0.286	18.13	16.43
FAPE samplewise 90/10	per-example stochastic 90/10 FAPE	0.305	18.36	17.97
FAPE samplewise 50/50	per-example stochastic 50/50 FAPE	0.385	12.71	16.84
FAPE fully unclamped	unclamped backbone FAPE throughout	0.450	10.54	15.72
FAPE unclamp at fine-tune	unclamp only after fine-tune boundary	0.319	15.85	16.63
one cycle	no iterative recycling	0.270	20.28	10.09
sampled 1..4 recycles	max/eval recycle 4, sampled train recycles	0.291	20.03	19.53
sampled 1..6 recycles	max/eval recycle 6, sampled train recycles	0.302	19.10	22.04
sampled 1..8 recycles	max/eval recycle 8, sampled train recycles	0.313	16.39	23.47
fixed 4 recycles	fixed train/eval 4 recycles	0.324	14.54	19.28
no fine-tune loss	initial objective for all 30k steps	0.413	12.66	17.90
fixed4 + unclamped + no FT	fixed 4 recycles, fully unclamped FAPE, no fine-tune	0.534	7.74	23.38

A.4. Supplementary lDDT and loss-component trajectories

The main text reports FoldScore curves because FoldScore is the benchmark-level aggregate metric. For transparency, this appendix also reports the corresponding public-validation lDDT-Cα curves and the per-run training loss-component trajectories for the ablations discussed in Section 5. The lDDT-Cα panels use the same checkpoint set as the FoldScore panels. The loss-component panels show the logged supervised objective terms for each run, which helps distinguish early optimization effects from changes that appear near the fine-tuning boundary or final checkpoint.

A.5. Qualitative Structure Examples

The two manuscript structure figures are generated from public-validation chains only. Predictions are Kabsch-aligned (Kabsch, 1976) to the ground-truth structure using valid Cα atoms before rendering. Ground truth is colored gray, minAlphaFold2 tiny is colored pink, and minAlphaFold2 medium is colored blue.

RMSD-selected triptych. The triptych in Figure 14 selects three public-validation chains using aligned Cα RMSD: one where both tiny and medium have comparatively low RMSD, one where medium has low RMSD and tiny has high RMSD, and one where both have high RMSD. The selected chains are 2k6i_A, 2v66_E, and 4abx_D.

Secondary-structure representatives. The secondary-structure figure in Figure 15 is not a best-RMSD panel. It is a representative visualization panel. We first used the split metadata bucket, then used PyMOL DSS assignment on the ground-truth PDBs to choose examples whose rendered cartoons visibly match the intended class. The selected public-validation chains are 2x7a_A (alpha), 6jqy_D (alpha/beta), 4glm_C (beta), 1aqs_A (coil/sparse), and 5w9f_A (mixed low-confidence).

A.6. Compute and Artifact Accounting

All core runs were executed on Runpod pods with NVIDIA H100 80GB HBM3 GPUs, PyTorch 2.8.0+cu128, CUDA 12.8, and cuDNN 9.10.2. The 15 unique runs in Table 9 consumed 261.6 measured H100-hours. At the observed H100 price of $2.99/hour, the core-figure runs cost approximately $782. Additional exploratory runs, including full-scale restarts, SimplexFold, Muon, interaction ablations, and legacy controls, were excluded from the core figures. Across complete tracked exploratory runs, measured wall time summed to approximately 375.1 H100-hours.

The largest individual core medium run was the sampled 1..8 recycle experiment at 23.47 H100-hours, which cost $70.18. The shortest core run was the tiny scaling run at 10.02 H100-hours, which cost $29.96.

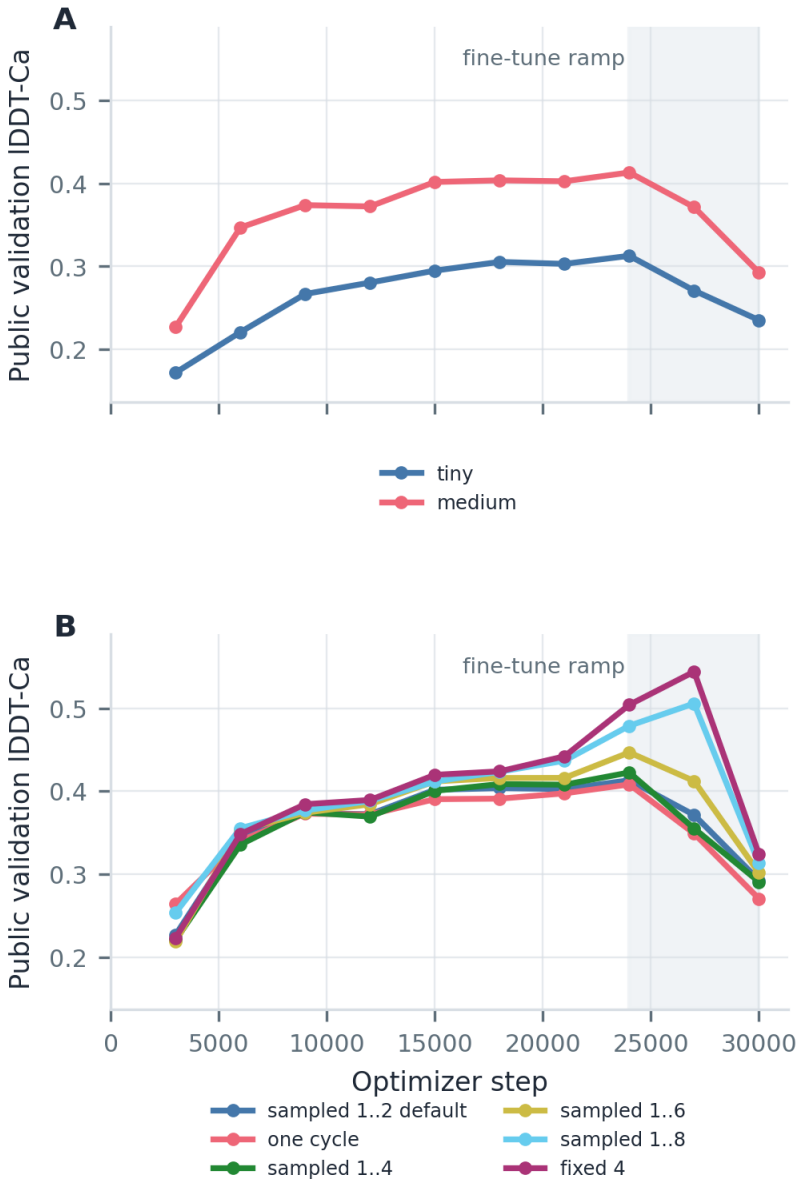
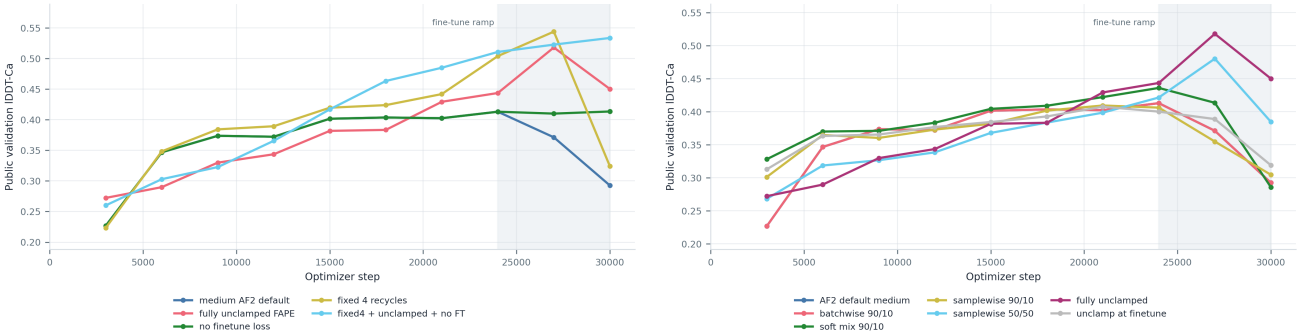


Figure 7. **Supplementary public-validation lDDT-Cα curves for scaling and recycling.** These are the lDDT-Cα counterparts to the FoldScore scaling and recycling curves in the main text: Panel A shows model scaling, and Panel B shows recycling-policy ablations. The shaded span marks the fine-tune ramp.



(a) Representative ablations

(b) Backbone-FAPE clamping policies

Figure 8. **Supplementary public-validation lDDT-Cα curves for objective ablations.** The ranking is consistent with the FoldScore curves: fully unclamped backbone FAPE, removing the fine-tuning loss phase, and the combined fixed-four/unclamped/no-fine-tune setting produce the strongest lDDT-Cα gains. Shaded spans mark the fine-tune ramp.

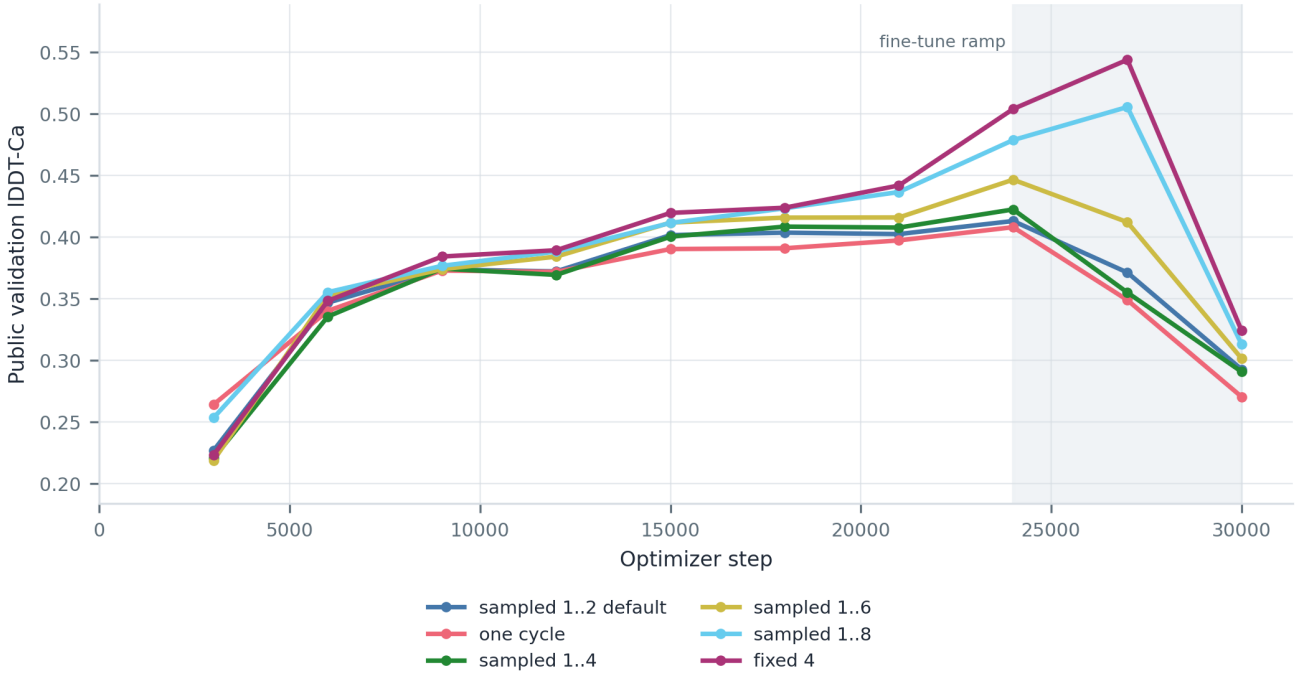
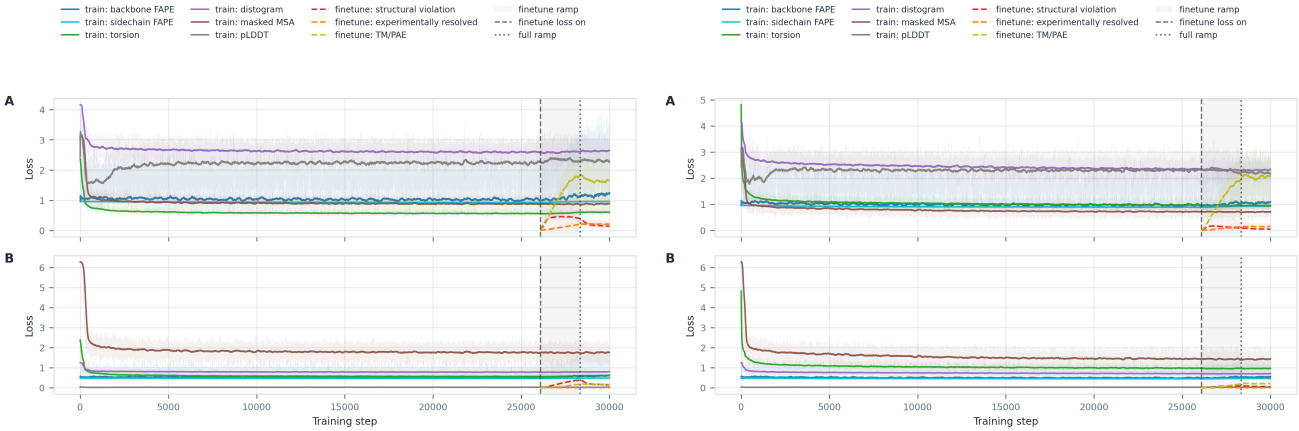


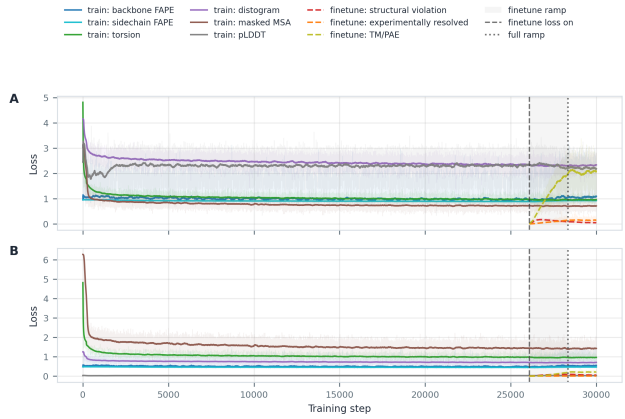
Figure 9. **Supplementary public-validation lDDT-Cα curves for recycling policy.** Recycling depth affects lDDT-Cα in the same direction as FoldScore, with the fixed-four recycle setting outperforming sampled-four recycling under the 240,000-sample public-validation study. The shaded span marks the fine-tune ramp.



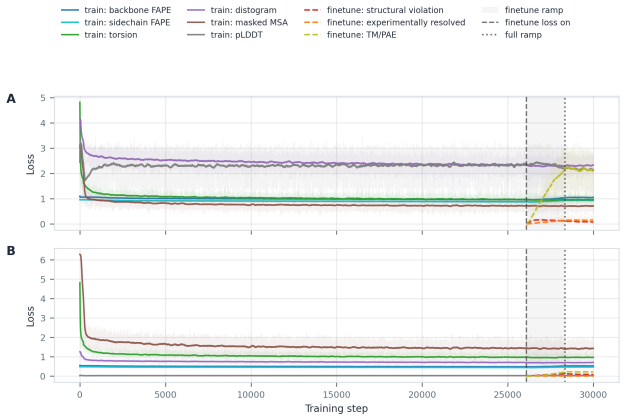
(a) Tiny AF2 default

(b) Medium AF2 default

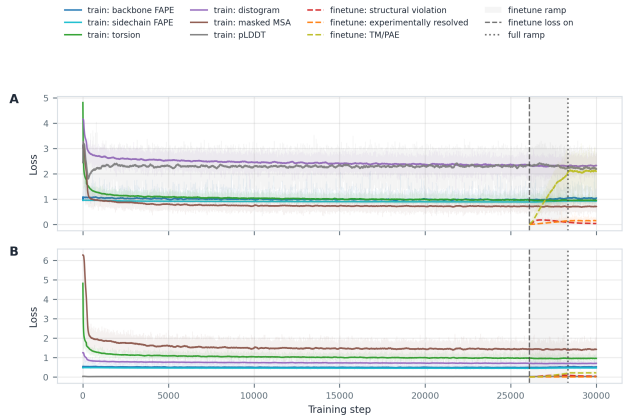
Figure 10. **Loss-component trajectories for the scaling runs.** In each subfigure, Panel A shows raw reported components and Panel B shows weighted contributions. Both profiles use the same AF2-style default recipe; the medium profile converts the same fixed data budget into stronger validation structure learning.



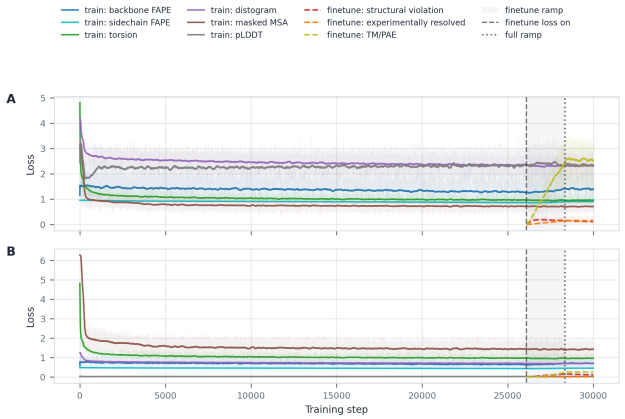
(a) Batchwise 90/10



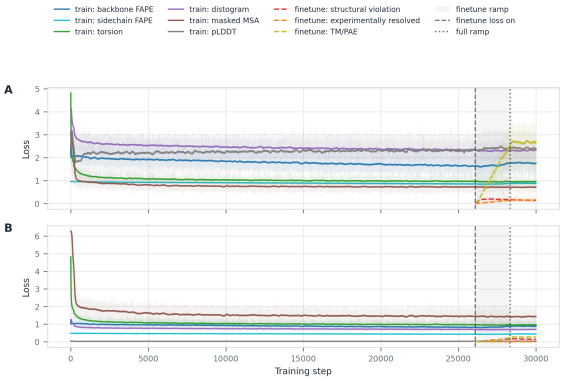
(b) Soft mix 90/10



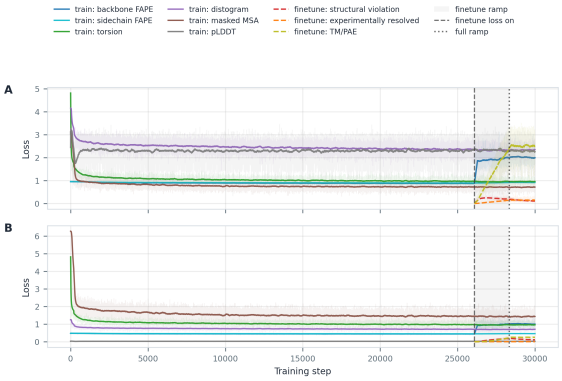
(c) Samplewise 90/10



(d) Samplewise 50/50

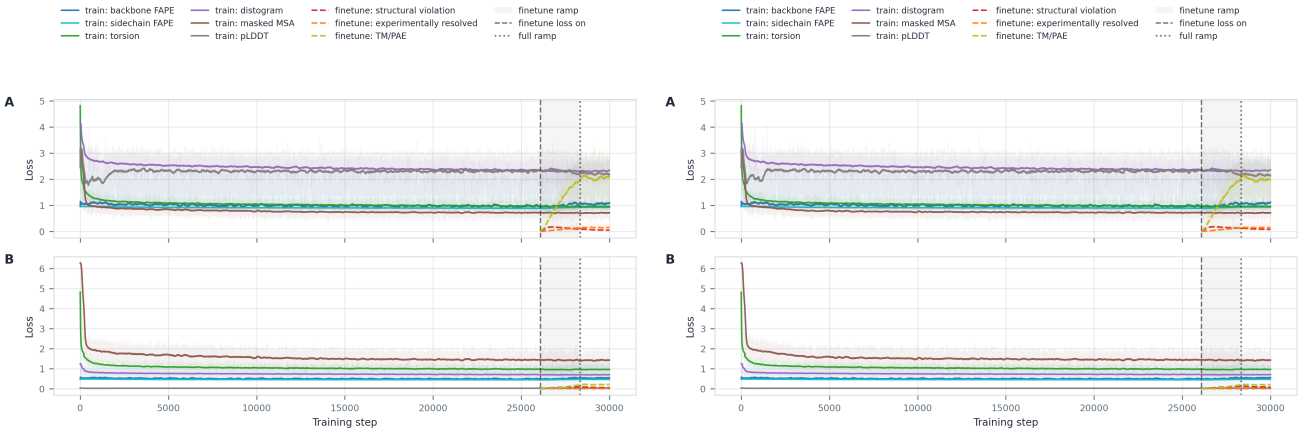


(e) Fully unclamped



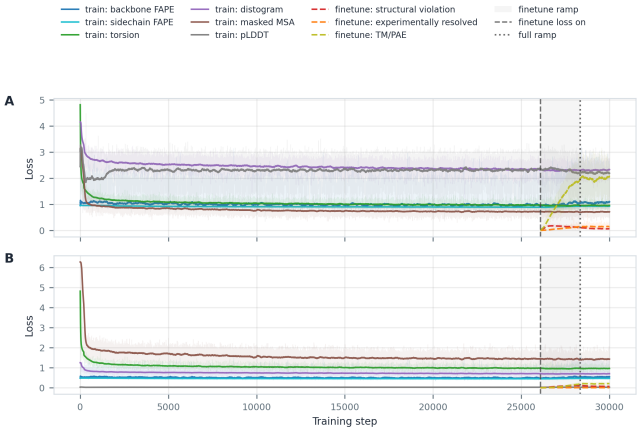
(f) Unclamp at fine-tune

Figure 11. **Loss-component trajectories for backbone-FAPE clamping policies.** In each subfigure, Panel A shows raw reported components and Panel B shows weighted contributions. These panels show how the clamped/unclamped backbone-FAPE choice changes the supervised objective components underlying the validation curves in Figures 8 and 3.

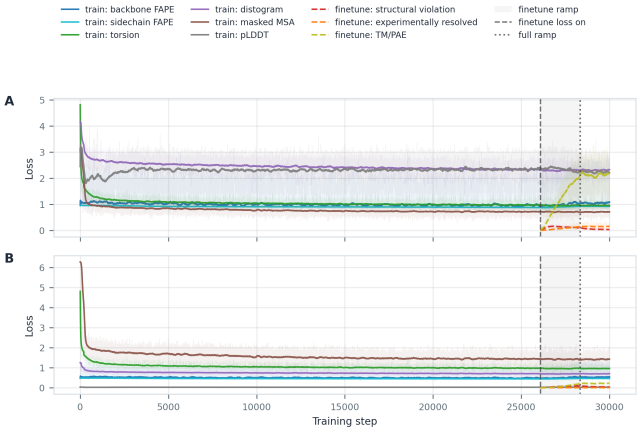


(a) Sampled 1..2 recycles

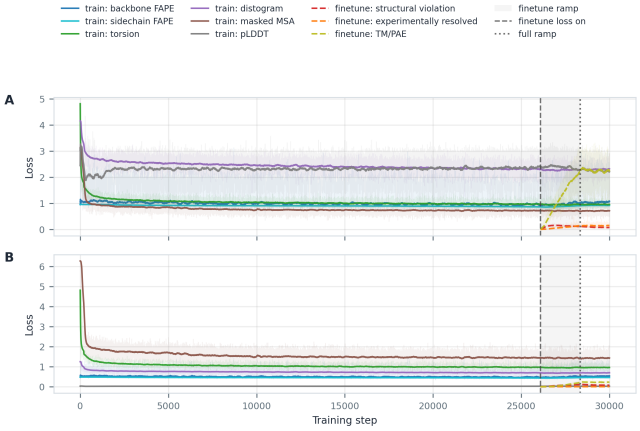
(b) One cycle



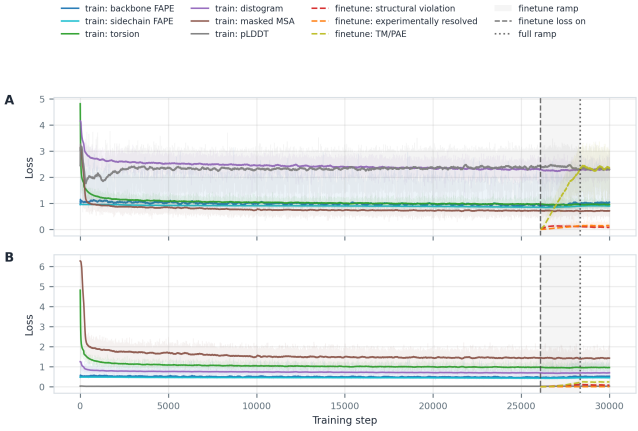
(c) Sampled 1..4 recycles



(d) Sampled 1..6 recycles

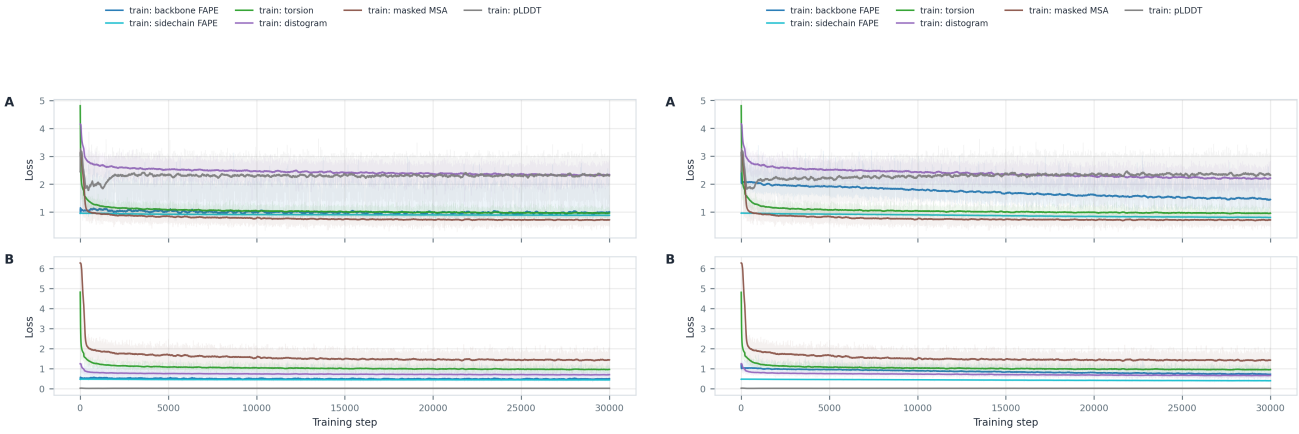


(e) Sampled 1..8 recycles



(f) Fixed 4 recycles

Figure 12. **Loss-component trajectories for recycling-policy ablations.** In each subfigure, Panel A shows raw reported components and Panel B shows weighted contributions. The loss traces complement the validation lDDT-Cα curves by showing the supervised terms for the same medium-model recycle sweep.



(a) No fine-tune loss

(b) Fixed4 + unclamped + no FT

Figure 13. **Loss-component trajectories for the loss-schedule and combined ablations.** In each subfigure, Panel A shows raw reported components and Panel B shows weighted contributions. The no-fine-tune run keeps the initial objective active for all 30,000 optimizer steps, while the combined run pairs that schedule with fixed-four recycling and fully unclamped backbone FAPE.

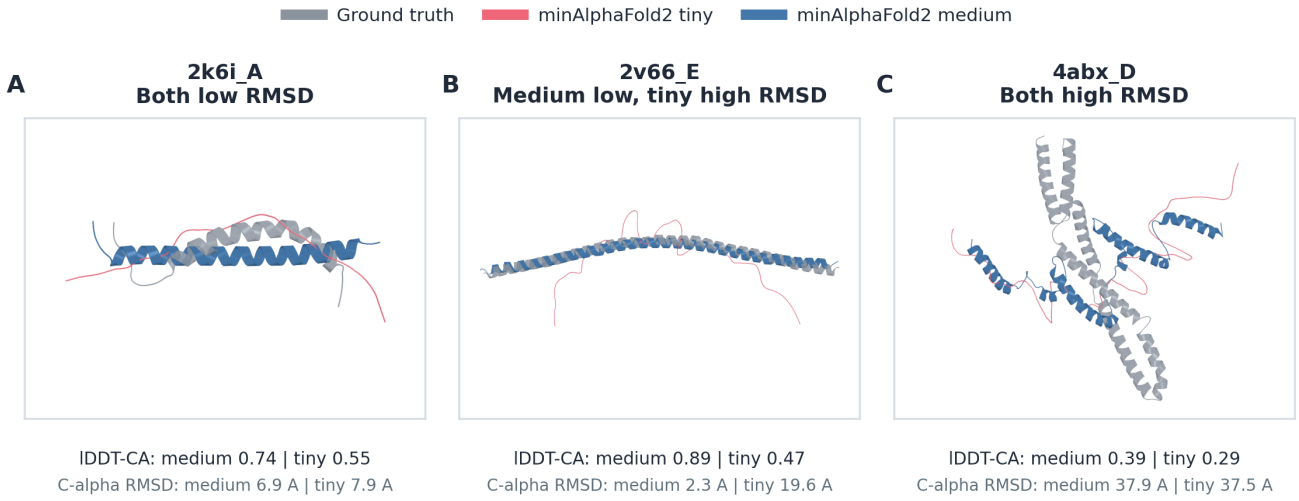


Figure 14. **RMSD-selected public-validation structure examples.** Ground truth, tiny prediction, and medium prediction are aligned on valid Cα atoms.

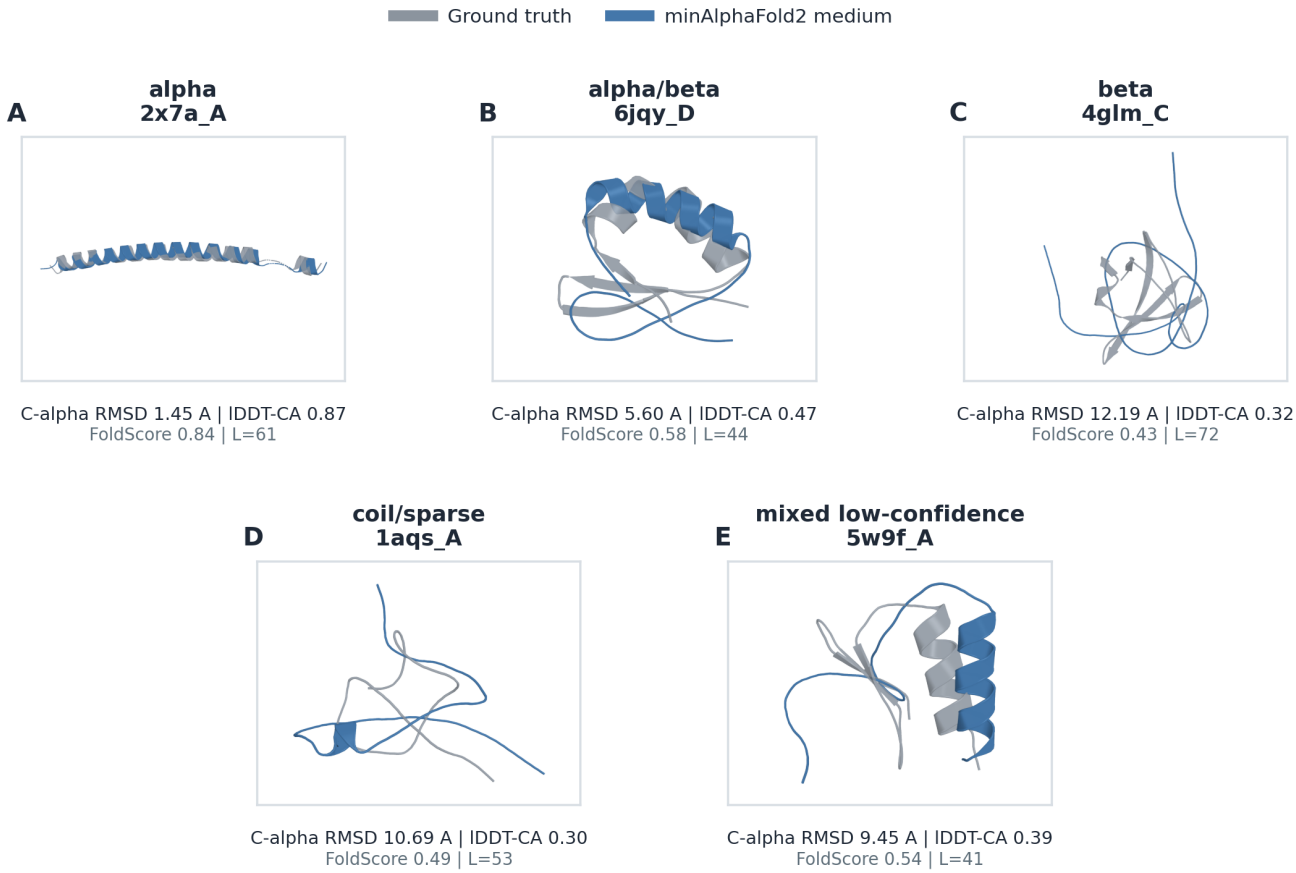


Figure 15. **Representative public-validation examples by secondary-structure split bucket.** Examples are chosen for visual representativeness within the split bucket rather than for best RMSD.