NEOPEP: REINFORCEMENT LEARNING FROM MACHINE-LEARNED FORCE FIELD ENERGY FOR NON-CANONICAL PEPTIDE STRUCTURE PREDICTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Peptide therapeutics occupy a compelling middle ground between small molecules and larger proteins, and in practice, most approved peptide drugs incorporate non-canonical amino acids (ncAAs) or other non-peptidic chemistries to elicit improved physiochemical properties (Yaseen Khan et al. (2026)). The chemical space of known ncAAs numbers in the thousands, yet current state-of-the-art sequence-to-structure prediction models handle them poorly, because fewer than 0.02% of residues deposited in the Protein Data Bank (PDB) (Berman et al. (2003)), the main source of training data, correspond to ncAAs. Our central idea is that this data bottleneck can be overcome by coupling generative structure prediction with reinforcement learning (RL) from machine-learned force field (MLFF)-derived energy. We present **Neopep**, a sequence-conditioned diffusion model for ncAA-containing peptide structure prediction, first pretrained on PDB structures to establish a structural prior and then fine-tuned via RL: for ncAA-containing peptide sequences, the model proposes structures that are scored by a frozen MLFF, and this energy signal is backpropagated directly into the model to progressively lower the energy of its predictions. MLFFs have demonstrated near-quantum accuracy across much of chemical space, approacing DFT-level accuracy at a fraction of the computational cost (Kabylda et al. (2025)). This makes them substantially more accurate than classical, empirically derived force fields, which are parametrised for the canonical 20 amino acids and do not generalise to arbitrary chemistries. While Neopep is trained on a fixed set of 189 ncAAs, the underlying architecture operates at atom-level granularity and is in principle generalisable to any chemistry. We show that our model is able to produce structures of ncAA-containing peptides that achieve lower RMSD values on average to the ground truth structures, in comparison to without finetuning and other state-of-the-art structure prediction models.

1 INTRODUCTION

Peptide therapeutics are a compelling middle ground between small molecules and larger proteins. Relative to small molecules, they offer greater target specificity and lower toxicity, and relative to larger proteins, they are more stable and less costly to produce (Abdin & Kim (2024))(Wang et al. (2022)). However, the canonical vocabulary of 20 proteinogenic amino acids constrains the chemical space available for designing such peptide therapeutics. In practice, most approved peptide drugs incorporate ncAAs or non-peptidic chemistries for improved physiochemical properties (Yaseen Khan et al. (2026)). Expanding the canonical 20 amino acid vocabulary to the thousands of known non-canonical amino acids (ncAAs)- many of which occur naturally as metabolic intermediates or post-translational modifications, while others are purely synthetic- would allow for the design of novel peptide drugs with unique and improved properties (Lee & Kim (2026))(Budisa (2025)). Realizing this potential from a computational perspective incentivizes the need to accurately model the structures of peptides containing ncAAs. Current state-of-the-art (SOTA) sequence-to-structure prediction models (Abramson et al. (2024))(Discovery et al. (2024))(Passaro et al. (2025)) are able to accept ncAAs in the input sequence, but are unable to model their structures well. This is due to the fact that the Protein Data Bank (PDB), the main source of training data for these models, has

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.

very little peptides with ncAAs as less than 0.02% of the residues deposited in the PDB correspond to ncAAs (Lee & Kim (2026)). The central question then becomes, how can we navigate this data scarcity concern to effectively model peptides with ncAAs?

To address this, we turn to the physical energy function as additional training signal. This is grounded in the Boltzmann distribution, which states that: for a particular sequence, low-energy structures are more probable than high-energy ones, meaning that a peptide's native structure is a low energy one. Concretely, instead of training a generative sequence-to-structure prediction model only to reproduce the ground-truth structures we happen to have, we can train it to produce structures that are low in energy- which is our way out of the data problem. Where structural data is absent, physics is not. We obtain this oracle energy from a machine-learned force field (Unke et al. (2021)), which are trained on large quantum-chemical datasets, and now reach near-quantum accuracy across the chemical space of organic and biological molecules, with speed much faster than traditional Density Functional Theory (DFT) (Kabylda et al. (2025)). Additionally, MLFFs are not restricted to the 20 proteinogenic amino acids like classical force fields are - as they function at the atomic level and do not require residue-level parametrization. Specifically, we use SO3LR (Kabylda et al. (2025)), built on the SO3krates architecture (Frank et al. (2024)) and trained on four million molecular complexes at the PBE0+MBD level of theory. Other potentials such as MACE-OFF (Kovács et al. (2025)), AIMNet (Zubatyuk et al. (2019)) (Anstine et al. (2024)), AI2BMD (Wang et al. (2024)) are also available, however, we choose SO3LR for our purposes because of its explicit treatment of molecular charges, and thorough analyses that demonstrate its effectiveness on bio-molecular systems specifically.

Concretely, our contribution is as follows: We introduce **Neopep**, a sequence-conditioned diffusion model to predict structures of peptides with ncAAs. We first pretrain a sequence-conditioned structure predictor following the AlphaFold3 (AF3) architecture on structures from the PDB, which we subsequently finetune with an energy reward from a MLFF, following the Direct Reward Fine-Tuning (DRaFT) framework (Clark et al. (2023)).

2 Related Work

Physical energy function as training signal. The field of sequence-conditioned protein structure prediction has increasingly shifted from modeling a single static structure to sampling the full conformational ensemble a protein can adopt, aka, sampling from the equilibrium, or Boltzmann distribution. While early progress on this front came from (1) architectural advances, like the shift from AlphaFold2 (AF2) (Jumper et al. (2021)) to the diffusion-based AF3 (Abramson et al. (2024)), alongside models such as Boltz (Passaro et al. (2025)), BioEmu (Lewis et al. (2025)), the Distributional Graphformer (Zheng et al. (2024)), and AlphaFlow (Jing et al. (2024)), and (2) from data-side strategies like incorporation of molecular dynamics trajectories for additional training data (Lewis et al. (2025)), these approaches remain bottlenecked by the need for large volumes of ground-truth structural data. More recently, the field has responded to this with an energy-as-supervision paradigm, in which the physical energy function itself serves as the training signal for equilibrium sampling, in place of ground-truth structures. We see this in the Boltzmann generator framework (Noé et al. (2019)), and a recent application of it to model the equilibrium distribution of peptides with canonical amino acids (Abdin & Kim (2024)), as well as the pre-training scheme introduced in the Distributional Graphformer paper (Zheng et al. (2024)). Additionally, this energy-as-supervision paradigm appears again in the application of GFLowNets to equilibrium small-molecule conformer generation (Volokhova et al. (2024)). To our knowledge, however, these efforts have remained confined to canonical amino acid space, and has not yet been applied to address the data scarcity concerning non-canonical chemistries.

Modeling and design of proteins with ncAAs. Rarefold (Li et al. (2025)), a structure prediction model for proteins containing ncAAs, extends the AF2 architecture by treating each amino acid, canonical or noncanonical, as a distinct token. This is different from the AF3 archictecture (Abramson et al. (2024)) which represents ncAAs at atom-level granularity and canonicals at residue-level granularity. However, because each ncAA is represented by a dedicated token, RareFold is fundamentally restricted to ncAAs seen during training, that is a set of 29 ncAAs, and cannot generalize to novel ncAA chemistries. Concurrent work, NCFlow (Lee & Kim (2026)), solves a slightly narrower problem: given a fixed protein backbone and a residue position where an ncAA sits, it predicts the

3D conformation of that single ncAA's atoms, using only the ncAA's atom types and bond graph as input, making it generalizable to arbitrary ncAAs unseen during training. To compensate for the scarcity of native ncAA data in the PDB, NCFlow is pretrained on small-molecule conformers and protein-ligand complexes before finetuning on ncAA-protein environments, and demonstrates improved structure prediction accuracy over AF3-based methods. Neopep differs from both these approaches in that, we use a physical energy function to guide us towards plausible structures instead of relying on architectural modifications, while also retaining the goal of full sequence-to-structure prediction rather than single-residue. Additionaly, since our architecture, following the AF3 archictecture, operates at atom-level granularity for ncAA representations, it is generalizable in principle.

3 METHOD

Neopep is trained in two phases, illustrated in Figure 1: (1) pretraining on PDB structures to establish a structural prior (Figure 1a), and (2) fine-tuning the pretrained model with a differentiable MLFF-derived energy reward to steer predictions towards physically plausible structures of ncAA-containing peptides (Figure 1b). We describe the architecture first, then each training phase in turn. Throughout, when we refer to AF3 we mean its open-source reimplementation, Boltz (Passaro et al., 2025).

3.1 ARCHITECTURE

As shown in Figure 1a, the Neopep architecture closely follows AF3 (Abramson et al., 2024); we describe it briefly and refer the reader to that paper for full architectural details. The model takes as input a peptide sequence of length 3–50 amino acids, which may contain ncAAs and a cyclic or linear topology flag. The **input embedder** encodes each residue using per-atom features derived from the Chemical Component Dictionary (CCD) (Westbrook et al., 2014). A *pair representation* is constructed alongside a *single representation*, encoding pairwise atom (or residue) relationships. Both representations are refined by the **trunk**, which comprises an MSA module, a template module, a pairformer, and recycling steps (Figure 1a). The refined representations then condition a **diffusion network** that samples plausible all-heavy-atom structures via $T = 200$ denoising steps. Architectural modifications relative to AF3 (reduced MSA and pairformer sizes, disabled template and confidence modules) are detailed in Appendix A.

3.2 TRAINING PHASE I: PRETRAINING A STRUCTURAL PRIOR

Training objective. The pretraining objective follows AF3 and combines a diffusion loss with an auxiliary distogram loss:

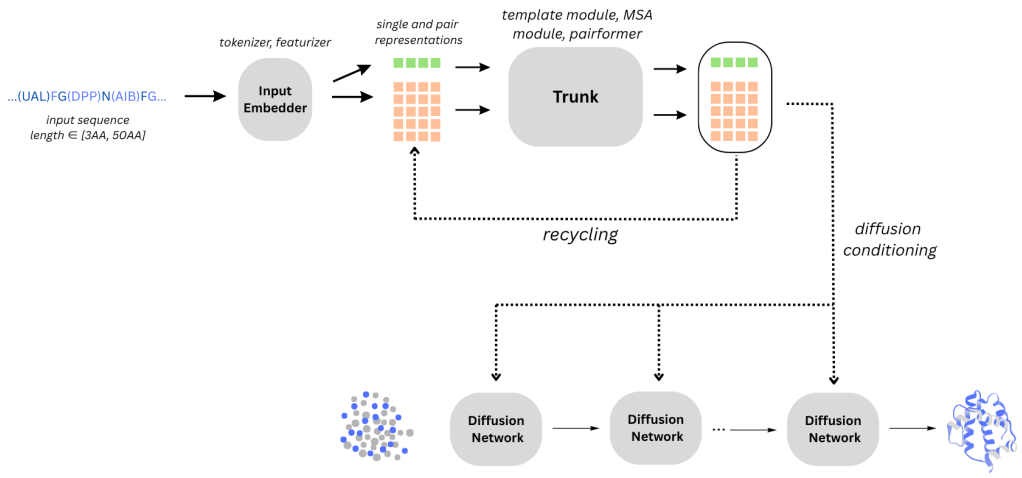
$$\mathcal{L}_{\text{pretrain}} = w_{\text{diff}}\,\mathcal{L}_{\text{diff}} + w_{\text{disto}}\,\mathcal{L}_{\text{disto}} \tag{1}$$

where $w_{\text{diff}} = 4.0$ and $w_{\text{disto}} = 0.03$.

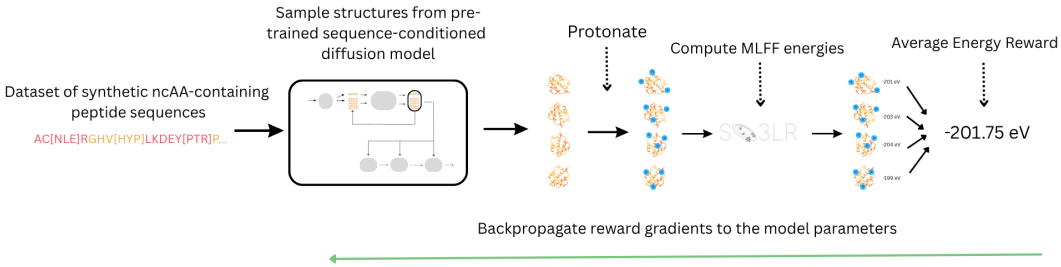
Diffusion loss. At each training step the ground-truth heavy-atom coordinates $\mathbf{x}_0$ are corrupted by Gaussian noise at a randomly sampled noise level σ following the variance-exploding (VE) schedule: $\tilde{\mathbf{x}} = \mathbf{x}_0 + \sigma\,\boldsymbol{\varepsilon}$, where $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. The network is trained via $\mathbf{x}$-prediction: it predicts the denoised coordinates $\hat{\mathbf{x}}_0$ directly, and a weighted MSE loss is computed against $\mathbf{x}_0$.

Distogram loss. An auxiliary distogram head is applied to the pair representation from the trunk to predict a discretised distribution over inter-residue distances, supervised with cross-entropy against the ground-truth distances. This loss provides direct supervision on the pair representation independently of the diffusion schedule.

Data. Following AF3, we pretrain on PDB structures with a training date cutoff of June 1, 2023; structures deposited after this date form the validation and test sets, which we further filter for sequence similarity with the training set, following AF3 protocol. Because our target application is peptides (3–50 residues), we sample *protein fragments* of this length from longer PDB chains, allowing the full PDB to be used as training data. MSA sequences are capped at 512 and template search is disabled entirely; all other data processing, clustering, and sampling follow Passaro et al. (2025). Additional details are given in Appendix B.



(a) Model architecture and pretraining (Phase 1). An input sequence is embedded and passed through the trunk, whose representations condition a diffusion network to sample peptide structures.



(b) RL fine-tuning with MLFF energy reward (Phase 2). M conformations are sampled from the pretrained model, protonated by the differentiable H adder, scored by the frozen SO3LR MLFF, and the average energy is backpropagated through the model to update its weights via gradient descent.

Figure 1: Overview of Neopep. Phase 1 (top) pretrains a Boltz-2-style structure diffusion model on PDB fragments. Phase 2 (bottom) fine-tunes the pretrained model with a differentiable MLFF energy reward via DRaFT-K.

3.3 Training Phase II: Fine-Tuning with MLFF-Derived Energy

We then fine-tune the pretrained model from phase 1 to minimize the physical energy of the generated peptide conformations. The objective is as follows:

$$\theta^* = \arg\min_\theta \ \mathbb{E}_{\mathbf{s}\sim p(\mathbf{s})}\left[\frac{1}{M}\sum_{i=1}^{M} E_{\text{SO3LR}}\left(\hat{\mathbf{x}}_\theta^{(i)}(\mathbf{s})\right)\right] \tag{2}$$

where $\hat{\mathbf{x}}_\theta^{(i)}(\mathbf{s})$ are $M = 4$ conformations sampled from the prior model for peptide sequence $\mathbf{s}$, and E_{SO3LR} is a frozen MLFF that scores these conformations. Concretely, the sequence $\mathbf{s}$ is first sampled uniformly, and is then passed as input to the model, to produce proposed structures. These proposals are then scored by the MLFF with an energy value, which is averaged over all M conformations. The model is optimized to yield the weights θ of the diffusion and trunk networks that best minimize this value. Ablations over the number of conformations $M \in \{4, 8, 12, 16\}$ and two energy pooling strategies (average vs. maximum) are reported in Appendix D. In the subsequent sections, we discuss the fine-tuning pipeline in greater detail.

Algorithm 1 DRaFT-K Fine-Tuning with SO3LR MLFF Energy Reward

Require: Pretrained model θ, SO3LR (frozen), $T = 200$, $K = 10$, $M = 4$

for each training step **do**
  Sample sequence $\mathbf{s}$, cyclic flag from PDB training set
  Precompute H placement specs from CCD (once per sequence)
  Compute $(\mathbf{s}, \mathbf{z}) \leftarrow \text{Trunk}_{\theta_{\text{trunk}}}(\mathbf{s})$
  Run steps $1, \ldots, T-K$ with **stop-gradient** $\rightarrow \mathbf{x}_{T-K}$
  Run steps $T-K+1, \ldots, T$ with gradient retained $\rightarrow \{\hat{\mathbf{x}}_T^{(i)}\}_{i=1}^{M}$
  for $i = 1$ **to** M **do**
    $\hat{\mathbf{x}}_i^{\text{full}} \leftarrow h(\hat{\mathbf{x}}_T^{(i)})$ ▷ differentiable H placement
    $E_i, \mathbf{F}_i \leftarrow \text{SO3LR}(\hat{\mathbf{x}}_i^{\text{full}}, q)$ ▷ frozen oracle MLFF energy
    $\hat{\mathbf{x}}_i^{\text{full}}.\text{backward}(-\mathbf{F}_i/M)$ ▷ inject forces as gradient
  end for
  Clip gradient norm to 1.0
  AdamW step on θ
end for

3.3.1 Synthetic Sequence Generation

At each fine-tuning step, a peptide sequence of 4–52 residues is sampled uniformly from the PDB training set used in Phase 1. Each position is then independently mutated to an ncAA with probability 0.2, drawn uniformly from the 189-ncAA support set of the differentiable hydrogen placement module (Section C.4). A cyclic or linear topology flag is assigned with probability 0.4 and 0.6, respectively. Head-to-tail peptide cyclicisation is an important property for peptide therapeutics, and AF3 natively accepts a cyclicity specification, in the form of a positional encoding, which embeds a head-to-tail proximity constraint, such that the first and last residues should be spatially close in the predicted structure. No MSA, templates, or inference-time steering are used; the model receives only the sequence with ncAAs and cyclicity specifications. Additional details are given in Appendix C.1.

3.3.2 Fine-tuning With a Differentiable Reward Specifics

Each sampled sequence is passed to the pretrained model, which generates $M = 4$ conformations $\{\hat{\mathbf{x}}_\theta^{(i)}(\mathbf{s})\}_{i=1}^{M}$ via $T = 200$ steps of diffusion sampling. These conformations are scored by SO3LR, their energies averaged, and the resulting signal used to update the model weights via gradient descent to minimize Eq. equation 2. To propagate this signal back through the model, we use the DRaFT (Directly Fine-Tuning Diffusion Models on Differentiable Rewards) framework (Clark et al., 2023), illustrated in Figure 1b and summarised in Algorithm 1.

As shown in Figure 1b and Algorithm 1, we use DRaFT-K, retaining the gradient graph through only the last $K = 10$ denoising steps to improve memory efficiency and reduce the risk of exploding gradients. Since SO3LR is a force field, it returns atomic forces $\mathbf{F}_i = -\partial E_i / \partial \hat{\mathbf{x}}_\theta^{(i)}(\mathbf{s})$ — the analytical negative gradient of the energy with respect to the final denoised coordinates — which are injected directly into the autograd graph in place of backpropagating through SO3LR itself. The full parameter gradient used in the gradient descent is then:

$$\frac{\partial \mathcal{L}}{\partial \theta} = \frac{1}{M} \sum_{i=1}^{M} \sum_{t=T-K+1}^{T} (-\mathbf{F}_i) \cdot \frac{\partial \hat{\mathbf{x}}_T^{(i)}}{\partial \mathbf{x}_t^{(i)}} \cdot \frac{\partial f_\theta(\mathbf{x}_{t-1}^{(i)}, \mathbf{s}, \mathbf{z})}{\partial \theta} \tag{3}$$

where $\mathcal{L}$ is the final SO3LR energy value averaged over the M conformations, $\hat{\mathbf{x}}_T^{(i)} = \hat{\mathbf{x}}_\theta^{(i)}(\mathbf{s})$, and $f_\theta(\mathbf{x}_{t-1}^{(i)}, \mathbf{s}, \mathbf{z}) = \mathbf{x}_t^{(i)}$, where f_θ denotes the diffusion model, which takes as input the atom coordinates at timestep $t-1$ and the single $\mathbf{s}$ and pair $\mathbf{z}$ representations from the trunk, and outputs the next denoised coordinates at timestep t. Note that we consider the transition from $\mathbf{x}_{t-1}^{(i)}$ to $\mathbf{x}_t^{(i)}$ as the denoising direction.

We can decompose $\partial \mathcal{L} / \partial \theta$ into two parts: The gradient w.r.t the diffusion module weights θ_{diff}, and the gradient w.r.t the trunk weights θ_{trunk}. The gradient w.r.t. the diffusion weights θ_{diff} follows

directly from Eq. 3 with $\partial f_\theta / \partial \theta_{\text{diff}}$ in place of $\partial f_\theta / \partial \theta$. For the gradient w.r.t. the trunk weights θ_{trunk}: the trunk runs once per training step to produce $(\mathbf{s}, \mathbf{z})$, which are shared across all K retained denoising steps and all M conformations. The gradient w.r.t. $(\mathbf{s}, \mathbf{z})$ therefore accumulates from all $K \times M$ calls:

$$\frac{\partial \mathcal{L}}{\partial (\mathbf{s}, \mathbf{z})} = \frac{1}{M} \sum_{i=1}^{M} \sum_{t=T-K+1}^{T} (-\mathbf{F}_i) \cdot \frac{\partial \hat{\mathbf{x}}_T^{(i)}}{\partial \mathbf{x}_t^{(i)}} \cdot \frac{\partial f_\theta(\mathbf{x}_{t-1}^{(i)}, \mathbf{s}, \mathbf{z})}{\partial (\mathbf{s}, \mathbf{z})} \tag{4}$$

$$\frac{\partial \mathcal{L}}{\partial \theta_{\text{trunk}}} = \frac{\partial \mathcal{L}}{\partial (\mathbf{s}, \mathbf{z})} \cdot \frac{\partial (\mathbf{s}, \mathbf{z})}{\partial \theta_{\text{trunk}}} \tag{5}$$

3.3.3 Differentiable Hydrogen Placement

The gradient chain in Eq. 3 terminates at the heavy-atom coordinates $\hat{\mathbf{x}}_T^{(i)}$ predicted by the diffusion model. However, SO3LR is an all-atom force field - it requires hydrogen atom positions, which the diffusion model does not predict. Standard structure protonation tools (e.g., RDKit, OpenBabel, OpenMM) are all either non-differentiable, and/or are parametrized for only the 20 canonical amino acids. We therefore implement a differentiable hydrogen placement module h in PyTorch, inspired by the approach of differentiable hydrogen placement for canonical amino acids in SideChainNet King & Koes (2021), and extend it to a set of 189 ncAAs such that $\hat{\mathbf{x}}_T^{\text{full},(i)} = h\big(\hat{\mathbf{x}}_T^{(i)}\big)$ is a differentiable function of the heavy-atom coordinates. The 189 supported residues were selected by first identifying the most frequently occurring non-canonical residues across protein chains in the PDB, then incrementally expanding the set as new ncAA types were encountered during subsequent test set curation.

The placement is split into a one-time preprocessing step and a differentiable runtime step. In preprocessing, the hydrogens that need to be placed for a given residue are given by the CCD Westbrook et al. (2014). Each hydrogen to be placed is assigned a *placement rule*: a set of reference heavy atoms and a closed-form geometric construction chosen according to the local bonding environment of its parent atom (e.g., hydrogens can assume any of the geometries which constrains their placement: methyl, methylene, tetrahedral, trigonal, etc). We define placement rules for each of the 20 canonical and 189 non-canonical residues, hard-coding how each hydrogen in that residue must be positioned via differentiable operations. At runtime, hydrogens are placed according to the rules specified for their residue. We note that this hard-coding inherently limits the number of ncAAs that the model can train on, so we aim to focus efforts on making the hydrogen placement generalizable in the future. The full construction rules for each of the hydrogen placement geometries are given in Appendix C.4.

This extends the full gradient chain from Eq. 3 to:

$$\frac{\partial \mathcal{L}}{\partial \theta} = \frac{1}{M} \sum_{i=1}^{M} \sum_{t=T-K+1}^{T} (-\mathbf{F}_i) \cdot \frac{\partial \hat{\mathbf{x}}_i^{\text{full}}}{\partial \hat{\mathbf{x}}_T^{(i)}} \cdot \frac{\partial \hat{\mathbf{x}}_T^{(i)}}{\partial \mathbf{x}_t^{(i)}} \cdot \frac{\partial f_\theta(\mathbf{x}_{t-1}^{(i)}, \mathbf{s}, \mathbf{z})}{\partial \theta} \tag{6}$$

where $\partial \hat{\mathbf{x}}_i^{\text{full}} / \partial \hat{\mathbf{x}}_T^{(i)}$ is the Jacobian of the hydrogen placement module- connecting the all-atom energy gradient $(-\mathbf{F}_i)$ to the heavy-atom output of the diffusion model.

4 Experiments

4.1 Test Set Curation

There is currently no established benchmark for ncAA peptide structure prediction. We therefore curate our own evaluation set from structures deposited in the PDB with release dates after January 1, 2024, ensuring temporal separation from our training cutoff of June 1, 2023 and our validation window (June 1, 2023–January 1, 2024).

Non-canonical peptide set (34 peptides). We downloaded 45,579 structures from the RCSB and extracted all protein chains of length 4–52 residues. Sequences are then subsequently filtered via the following pipeline: (*i*) removal of sequences that are similar to the training set via MMseqs2

(Steinegger & Söding, 2017)- any chain sharing $\geq$40% sequence identity over $\geq$80% query coverage is excluded; (*ii*) removal of chains whose only non-standard residues are trivial terminal caps (ACE, NH2), unresolved (UNK) positions, or positions whose CCD codes are classified as non-polymer ligands. From the remaining polymer-only ncAA pool we retain 34 chains for which every non-canonical residue is supported by our differentiable H-placement module (Section C.4).

Canonical peptide set (24 peptides). We additionally curate a canonical-only set which contain chains from the same post-2024 release window that contain no non-canonical residues, in order to monitor potential regression on standard amino acids after fine-tuning. The same MMseqs2 filter is applied.

4.2 Baselines

We compare against three baselines. **Boltz-2** (Passaro et al., 2025) is the publicly released checkpoint of the model our pretraining stage is based on; it serves as a measure of how well a general-purpose structure predictor handles ncAA chemistry without any ncAA-specific adaptation. **Protenix-v1** (Zhang et al., 2026b), (Team et al., 2025), Zhang et al. (2026a) is a state-of-the-art open-source AlphaFold-3-style predictor; 5 of the 34 ncAA test peptides contain residues not in Protenix's CCD vocabulary and are excluded from its evaluation (n=29). **Pretrained (Neopep Phase 1)** is our own model after Phase 1 pretraining only, before any RL fine-tuning.

All models run in single-sequence mode (no MSA). We generate 25 structure predictions per test peptide for all models.

4.3 Evaluation Metrics

We report the Cα RMSD (**CA-RMSD**) after superposition of the predicted chain onto the ground-truth. Because our fine-tuning objective minimises a force-field energy rather than a direct structural loss, the model can produce a diverse ensemble of low-energy conformers that may not cluster tightly around the crystal structure. We therefore report the **minimum CA-RMSD** over 25 predictions (the closest the model can get to the ground truth) and additionally the **average CA-RMSD**. We also report the minimum heavy-atom RMSD (**FA-RMSD**) after superposition.

4.4 Results

Table 1 summarises the overall CA-RMSD averaged across each test subset. Per-peptide breakdowns are provided in Appendix D.1.

Table 1: CA-RMSD (Å) averaged over all test peptides (mean $\pm$ std across peptides, min CA-RMSD is computed over the average of all the min CA-RMSDs across the test peptides). Bold indicates the best value in each column. Protenix evaluated on n=29 non-canonical peptides (5 excluded due to missing CCD support); all other models evaluated on n=34.

	Non-canonical (n=34)		**Canonical** (n=24)	
Model	**Min CA-RMSD**	**Avg CA-RMSD**	**Min CA-RMSD**	**Avg CA-RMSD**
Boltz-2	4.448 $\pm$ 1.595	9.047 $\pm$ 5.522	2.744 $\pm$ 2.605	3.438 $\pm$ 2.955
Protenix-v1†	2.258 $\pm$ 1.320	2.845 $\pm$ 1.437	3.005 $\pm$ 3.022	3.738 $\pm$ 3.039
Neopep Pretrained	2.271 $\pm$ 0.837	3.236 $\pm$ 0.958	3.046 $\pm$ 2.765	3.847 $\pm$ 3.105
Neopep Finetuned	**1.565** $\pm$ 0.957	**2.546** $\pm$ 1.160	**2.211** $\pm$ 1.927	3.768 $\pm$ 3.078

†Evaluated on n=29 non-canonical peptides.

Non-canonical peptides. Neopep fine-tuned achieves the lowest average Min CA-RMSD (1.565 Å) among all evaluated models. The large gap between Boltz-2 (4.448 Å) and the pretrained Neopep model (2.271 Å) suggests that the pretraining strategy of training on PDB fragments is beneficial for establising a good structural prior for peptides. The further improvement from the energy fine-tuned model confirms that the SO3LR energy reward steers the diffusion model towards physically plausible conformations of ncAA-containing peptides. The same ordering holds

Table 2: Minimum heavy-atom RMSD (FA-RMSD, Å) averaged over all test peptides (mean ± std across peptides). Bold indicates the best value in each column. Protenix evaluated on n=29 non-canonical peptides; all other models on n=34.

Model	**Min FA-RMSD, Non-canonical** (n=34)	**Min FA-RMSD, Canonical** (n=24)
Boltz-2	5.818 ± 1.458	4.035 ± 2.550
Protenix-v1[†]	3.563 ± 1.553	4.331 ± 2.922
Neopep Pretrained	3.735 ± 0.804	4.037 ± 2.800
Neopep Finetuned	**3.194** ± 0.944	**3.456** ± 1.917

[†]Evaluated on n=29 non-canonical peptides.

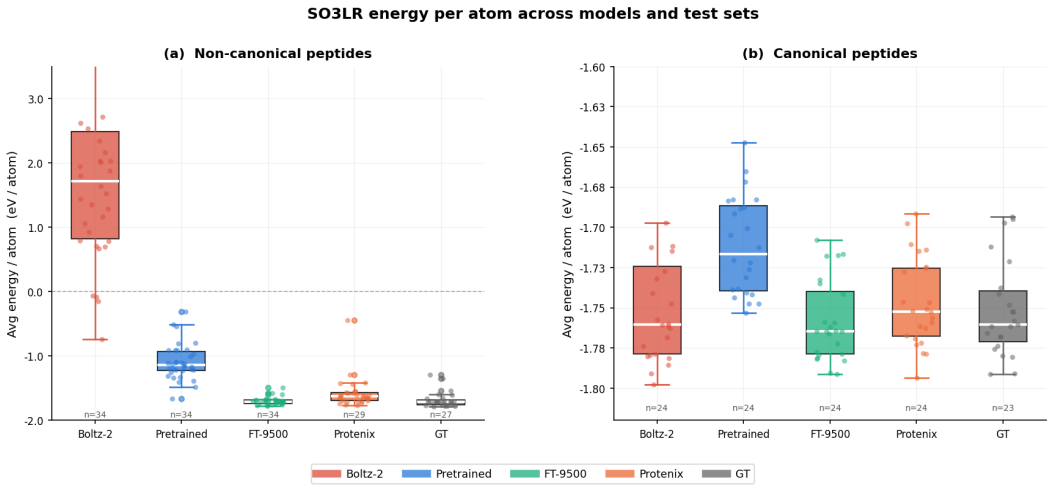


Figure 2: Average SO3LR energy per atom (eV/atom) over 25 predictions per peptide, across all peptides for the non-canonical (left) and canonical (right) test sets. GT = crystal structure energy. Some atoms in the ground truth for some of the test peptides were missing, so for a fair comparison of the energies, we only include ground truth peptides that have the same number of atoms as the predictions (i.e., everything is fully resolved).

for heavy-atom FA-RMSD (Table 2): Neopep fine-tuned achieves a Min FA-RMSD of 3.194 Å, compared to 3.735 Å for the pretrained model, 3.563 Å for Protenix-v1, and 5.818 Å for Boltz-2 on the non-canonical test set. Additionally, figure 2 shows the SO3LR computed average energy per atom over 25 predictions per peptide, for the non-canoncial and canoncial test sets, across the same models. We highlight the performance of our finetuned model in that it is able to predict more lower energy conformers, while also matching the eV/atom of the ground truth structrures. This is essentially a sanity check, to ensure that our fine-tuning objective worked as intended.

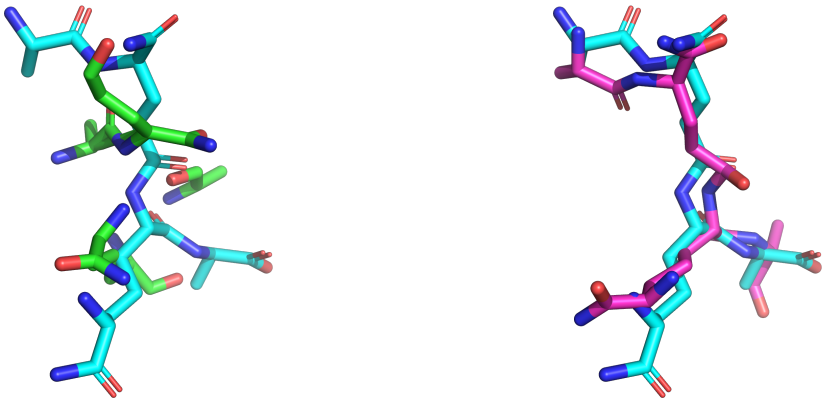
Canonical peptides. Fine-tuning does not degrade performance on canonical sequences: Neopep fine-tuned achieves the best average Min CA-RMSD on the canonical set (2.211 Å), outperforming both Boltz-2 (2.744 Å) and Protenix-v1 (3.005 Å). Neopep fine-tuned achieves a Min FA-RMSD of 3.456 Å, compared to 4.035 Å for Boltz-2, 4.037 Å for the pretrained model, and 4.331 Å for Protenix-v1 (Table 2). Additionally, we look at the SO3LR computed average energy per atom over 25 predictions per peptide in figure 2. Similar to the results for the non-canonical test set, our finetuned model outputs structures that are either matching in energy to the respective ground truth, or are slightly better. Again, we note that this is essentially a sanity check to ensure that our training objective works as intended.

4.5 Case Studies

To qualitatively illustrate the effect of MLFF energy-guided fine-tuning, we examine two non-canonical peptides from the test set where the pretrained and fine-tuned models produce noticeably

different structures. For each peptide we show the minimum-RMSD prediction from each model overlaid on the crystal structure, rendered with PyMOL (Schrödinger, LLC).

As shown in Figures 3, and 4 the pretrained-only model produces a conformation that appears to not be aligned well to the ground truth, and pymol was unable to infer some bonds, which is why the structure looks like it has ”breaks” in it. We then examine a structure after energy tuning, and observe that it appears to be fully connected, which means that pymol correctly inferred bonds from atomic positions, which is reflected in the very low CA-RMSD.



(a) Pretrained (CA-RMSD = 2.46 Å (b) Fine-tuned (CA-RMSD = 0.27 Å)

Figure 3: **8pxy_B** - predicted structure (green, pink) overlaid on the ground truth structure (blue). Left: Neopep pretrained model. Right: Neopep fine-tuned (FT-9500). Fine-tuning corrects ’breaks’ in structures, allowing pymol to better infer bonds from better placed atoms.



(a) Pretrained (CA-RMSD = 1.7 Å) (b) Fine-tuned (CA-RMSD = 0.72 Å)

Figure 4: **22es_F** - predicted structure (pink, blue) overlaid on the ground truth structure. Left: Neopep pretrained model. Right: Neopep fine-tuned (FT-9500). Fine-tuning corrects ’breaks’ in structure, allowing pymol to better infer bonds from better placed atoms.

AI USE STATEMENT

In this work, we used generative AI tools to aid in the writing and planning of the paper contents, especially getting lateX-style equations. AI was also use to write code scripts to automate processes. For instance, a script to search through a database, or a script to write hydrogen placement functions for each of our parametrized 189 ncAAs given a list of rules to follow. We do not however, use AI to prove any sort of mathematical claims or for the research ideation process. Wherever we did use LLM assistance (when writing the paper, for isntance) we made sure to thoroughly check its outputs and understand everything before deciding to include it. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

Our models leverage already-public data for training, such as the Protein Databank. As such, we feel that we do not have any instances where the paper submission raises questions in the code of Ethics.

REPRODUCIBILITY STATEMENT

We provide experiments in section 4, detailing how we curated our datasets and which baselines we used. The appendix has additional details on our augmentations to the Boltz2/AF3 architecture, pretraining, and finetuning, as well as ablations. Additionally, we attach an anonymous zip file with this submission with key scripts/code important to this project. As of now, the zip file is simply a folder of relevant scripts, but we hope to release it with thorough instructions and information on how data processing pretraining and finetuning should be run.

ACKNOWLEDGMENTS

Acknowledgments are omitted to preserve anonymity during review.

REFERENCES

Osama Abdin and Philip M. Kim. Direct conformational sampling from peptide energy landscapes through hypernetwork-conditioned diffusion. *Nature Machine Intelligence*, 6:775–786, 2024. doi: 10.1038/s42256-024-00860-4. URL https://doi.org/10.1038/s42256-024-00860-4.

Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J. Ballard, Joshua Bambrick, Sebastian W. Bodenstein, David A. Evans, Chia-Chun Hung, Michael O'Neill, David Reiman, Kathryn Tunyasuvunakool, Zachary Wu, Akvilė Žemgulytė, Eirini Arvaniti, Charles Beattie, Ottavia Bertolli, Alex Bridgland, Alexey Cherepanov, Miles Congreve, Alexander I. Cowen-Rivers, Andrew Cowie, Michael Figurnov, Fabian B. Fuchs, Hannah Gladman, Rishub Jain, Yousuf A. Khan, Caroline M. R. Low, Kuba Perlin, Anna Potapenko, Pascal Savy, Sukhdeep Singh, Adrian Stecula, Ashok Thillaisundaram, Catherine Tong, Sergei Yakneen, Ellen D. Zhong, Michal Zielinski, Augustin Žídek, Victor Bapst, Pushmeet Kohli, Max Jaderberg, Demis Hassabis, and John M. Jumper. Accurate structure prediction of biomolecular interactions with alphafold 3. *Nature*, 630: 493–500, 2024. doi: 10.1038/s41586-024-07487-w. URL https://doi.org/10.1038/s41586-024-07487-w.

Dylan Anstine, Roman Zubatyuk, and Olexandr Isayev. Aimnet2: A neural network potential to meet your neutral, charged, organic, and elemental-organic needs. 2024. doi: 10.26434/chemrxiv-2023-296ch-v2. URL https://doi.org/10.26434/chemrxiv-2023-296ch-v2.

Helen Berman, Kim Henrick, and Haruki Nakamura. Announcing the worldwide protein data bank. *Nature Structural amp; Molecular Biology*, 10:980–980, 2003. doi: 10.1038/nsb1203-980. URL https://doi.org/10.1038/nsb1203-980.

Nediljko Budisa. Introduction: "noncanonical amino acids". *Chemical Reviews*, 125:1659–1662, 2025. doi: 10.1021/acs.chemrev.5c00065. URL https://doi.org/10.1021/acs.chemrev.5c00065.

Kevin Clark, Paul Vicol, Kevin Swersky, and David J Fleet. Directly fine-tuning diffusion models on differentiable rewards, 2023. URL https://arxiv.org/abs/2309.17400.

Chai Discovery, Jacques Boitreaud, Jack Dent, Matthew McPartlon, Joshua Meier, Vinicius Reis, Alex Rogozhnikov, and Kevin Wu. Chai-1: Decoding the molecular interactions of life. 2024. doi: 10.1101/2024.10.10.615955. URL https://doi.org/10.1101/2024.10.10.615955.

J. Thorben Frank, Oliver T. Unke, Klaus-Robert Müller, and Stefan Chmiela. A euclidean transformer for fast and stable machine learned force fields. *Nature Communications*, 15, 2024. doi: 10.1038/s41467-024-50620-6. URL https://doi.org/10.1038/s41467-024-50620-6.

Bowen Jing, Bonnie Berger, and Tommi Jaakkola. Alphafold meets flow matching for generating protein ensembles, 2024. URL https://arxiv.org/abs/2402.04845.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A. A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W. Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. Highly accurate protein structure prediction with alphafold. *Nature*, 596:583–589, 2021. doi: 10.1038/s41586-021-03819-2. URL https://doi.org/10.1038/s41586-021-03819-2.

Adil Kabylda, J. Thorben Frank, Sergio Suarez Dou, Almaz Khabibrakhmanov, Leonardo Medrano Sandonas, Oliver T. Unke, Stefan Chmiela, Klaus-Robert Müller, and Alexandre Tkatchenko. Molecular simulations with a pretrained neural network and universal pairwise force fields. *ChemRxiv*, 2025(0611), 2025. doi: 10.26434/chemrxiv-2024-bdfr0-v3. URL https://chemrxiv.org/doi/abs/10.26434/chemrxiv-2024-bdfr0-v3.

Jonathan Edward King and David Ryan Koes. ¡scp¿sidechainnet¡/scp¿ : An ¡scp¿all-atom¡/scp¿ protein structure dataset for machine learning. *Proteins: Structure, Function, and Bioinformatics*, 89:1489–1496, 2021. doi: 10.1002/prot.26169. URL https://doi.org/10.1002/prot.26169.

Dávid Péter Kovács, J. Harry Moore, Nicholas J. Browning, Ilyes Batatia, Joshua T. Horton, Yixuan Pu, Venkat Kapil, William C. Witt, Ioan-Bogdan Magdău, Daniel J. Cole, and Gábor Csányi. Mace-off: Short-range transferable machine learning force fields for organic molecules. *Journal of the American Chemical Society*, 147:17598–17611, 2025. doi: 10.1021/jacs.4c07099. URL https://doi.org/10.1021/jacs.4c07099.

Jin Sub Lee and Philip M Kim. Design of peptides with noncanonical amino acids using flow matching. *Bioinformatics*, 42, 2026. doi: 10.1093/bioinformatics/btag654. URL https://doi.org/10.1093/bioinformatics/btag654.

Sarah Lewis, Tim Hempel, José Jiménez-Luna, Michael Gastegger, Yu Xie, Andrew Y. K. Foong, Victor García Satorras, Osama Abdin, Bastiaan S. Veeling, Iryna Zaporozhets, Yaoyi Chen, Soojung Yang, Adam E. Foster, Arne Schneuing, Jigyasa Nigam, Federico Barbero, Vincent Stimper, Andrew Campbell, Jason Yim, Marten Lienen, Yu Shi, Shuxin Zheng, Hannes Schulz, Usman Munir, Roberto Sordillo, Ryota Tomioka, Cecilia Clementi, and Frank Noé. Scalable emulation of protein equilibrium ensembles with generative deep learning. *Science*, 389, 2025. doi: 10.1126/science.adv9817. URL https://doi.org/10.1126/science.adv9817.

Qiuzhen Li, Diandra Daumiller, Fanglei Zuo, Harold Marcotte, Qiang Pan-Hammarström, and Patrick Bryant. Rarefold: Structure prediction and design of proteins with noncanonical amino acids. 2025. doi: 10.1101/2025.05.19.654846. URL https://doi.org/10.1101/2025.05.19.654846.

Frank Noé, Simon Olsson, Jonas Köhler, and Hao Wu. Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. *Science*, 365, 2019. doi: 10.1126/science.aaw1147. URL https://doi.org/10.1126/science.aaw1147.

Saro Passaro, Gabriele Corso, Jeremy Wohlwend, Mateo Reveiz, Stephan Thaler, Vignesh Ram Somnath, Noah Getz, Tally Portnoi, Julien Roy, Hannes Stark, David Kwabi-Addo, Dominique Beaini, Tommi Jaakkola, and Regina Barzilay. Boltz-2: Towards accurate and efficient binding affinity prediction. 2025. doi: 10.1101/2025.06.14.659707. URL https://doi.org/10.1101/2025.06.14.659707.

Schrödinger, LLC. The PyMOL molecular graphics system, version 3.0. URL https://www.pymol.org.

Martin Steinegger and Johannes Söding. Mmseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature Biotechnology*, 35:1026–1028, 2017. doi: 10.1038/nbt.3988. URL https://doi.org/10.1038/nbt.3988.

ByteDance AML AI4Science Team, Xinshi Chen, Yuxuan Zhang, Chan Lu, Wenzhi Ma, Jiaqi Guan, Chengyue Gong, Jincai Yang, Hanyu Zhang, Ke Zhang, Shenghao Wu, Kuangqi Zhou, Yanping Yang, Zhenyu Liu, Lan Wang, Bo Shi, Shaochen Shi, and Wenzhi Xiao. 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.

Oliver T. Unke, Stefan Chmiela, Huziel E. Sauceda, Michael Gastegger, Igor Poltavsky, Kristof T. Schütt, Alexandre Tkatchenko, and Klaus-Robert Müller. Machine learning force fields. *Chemical Reviews*, 121:10142–10186, 2021. doi: 10.1021/acs.chemrev.0c01111. URL https://doi.org/10.1021/acs.chemrev.0c01111.

Alexandra Volokhova, Michał Koziarski, Alex Hernández-García, Cheng-Hao Liu, Santiago Miret, Pablo Lemos, Luca Thiede, Zichao Yan, Alán Aspuru-Guzik, and Yoshua Bengio. Towards equilibrium molecular conformation generation with gflownets. *Digital Discovery*, 3:1038–1047, 2024. doi: 10.1039/d4dd00023d. URL https://doi.org/10.1039/d4dd00023d.

Lei Wang, Nanxi Wang, Wenping Zhang, Xurui Cheng, Zhibin Yan, Gang Shao, Xi Wang, Rui Wang, and Caiyun Fu. Therapeutic peptides: current applications and future directions. *Signal Transduction and Targeted Therapy*, 7, 2022. doi: 10.1038/s41392-022-00904-4. URL https://doi.org/10.1038/s41392-022-00904-4.

Tong Wang, Xinheng He, Mingyu Li, Yatao Li, Ran Bi, Yusong Wang, Chaoran Cheng, Xiangzhen Shen, Jiawei Meng, He Zhang, Haiguang Liu, Zun Wang, Shaoning Li, Bin Shao, and Tie-Yan Liu. Ab initio characterization of protein molecular dynamics with ai2bmd. *Nature*, 635:1019–1027, 2024. doi: 10.1038/s41586-024-08127-z. URL https://doi.org/10.1038/s41586-024-08127-z.

John D. Westbrook, Chenghua Shao, Zukang Feng, Marina Zhuravleva, Sameer Velankar, and Jasmine Young. The chemical component dictionary: complete descriptions of constituent molecules in experimentally determined 3d macromolecules in the protein data bank. *Bioinformatics*, 31:1274–1278, 2014. doi: 10.1093/bioinformatics/btu789. URL https://doi.org/10.1093/bioinformatics/btu789.

Muhammad Yaseen Khan, Touseef Nawaz, Muhammad Sajid Hamid Akash, and Adnan Amin. Pharmaceutical peptides: From synthesis and mechanistic pharmacology to future biologic therapeutics. *Pharmaceuticals*, 19(6), 2026. ISSN 1424-8247. doi: 10.3390/ph19060811. URL https://www.mdpi.com/1424-8247/19/6/811.

Yuxuan Zhang, Chengyue Gong, Jinyuan Sun, Jiaqi Guan, Milong Ren, Song Xue, Hanyu Zhang, Wenzhi Ma, Zhenyu Liu, Xinshi Chen, and Wenzhi Xiao. Protenix-v2: Broadening the reach of structure prediction and biomolecular design. *bioRxiv*, 2026a. doi: 10.64898/2026.04.10.717613. URL https://www.biorxiv.org/content/early/2026/04/11/2026.04.10.717613.

Yuxuan Zhang, Chengyue Gong, Hanyu Zhang, Wenzhi Ma, Zhenyu Liu, Xinshi Chen, Jiaqi Guan, Lan Wang, Yanping Yang, Yu Xia, and Wenzhi Xiao. Protenix-v1: Toward high-accuracy open-source biomolecular structure prediction. *bioRxiv*, 2026b. doi: 10.64898/2026.02.05.703733. URL https://www.biorxiv.org/content/early/2026/02/22/2026.02.05.703733.1.

Shuxin Zheng, Jiyan He, Chang Liu, Yu Shi, Ziheng Lu, Weitao Feng, Fusong Ju, Jiaxi Wang, Jianwei Zhu, Yaosen Min, He Zhang, Shidi Tang, Hongxia Hao, Peiran Jin, Chi Chen, Frank Noé, Haiguang Liu, and Tie-Yan Liu. Predicting equilibrium distributions for molecular systems with deep learning. *Nature Machine Intelligence*, 6:558–567, 2024. doi: 10.1038/s42256-024-00837-3. URL https://doi.org/10.1038/s42256-024-00837-3.

Roman Zubatyuk, Justin S. Smith, Jerzy Leszczynski, and Olexandr Isayev. Accurate and transferable multitask prediction of chemical properties with an atoms-in-molecules neural network. *Science Advances*, 5, 2019. doi: 10.1126/sciadv.aav6490. URL https://doi.org/10.1126/sciadv.aav6490.

A Appendix

Appendix Contents.

A Architecture Details
- Model Configuration

B Pretraining Details
- Dataset
- Training Configuration

C Fine-tuning Details
- Synthetic Sequence Generation
- Optimization Hyperparameters
- Supported Non-Canonical Amino Acids
- Hydrogen Placement Geometry Rules

D Ablations

E Limitations

A Architecture Details

A.1 Model Configuration

Table 3 summarizes the key hyperparameters of the Neopep model. We follow the AF3/Boltz architecture but reduce the model size relative to the full Boltz-2 configuration to allow more efficient training.

Table 3: Neopep model hyperparameters

Hyperparameter	Neopep	Boltz-2 (reference)
Single representation dim d_s	384	384
Pair representation dim d_z	128	128
MSA dim d_m	64	64
MSA blocks	2	4
Pairformer blocks	12	48
Diffusion steps T	200	200

MSA module. We include a lightweight MSA module with d_m=64 and 2 MSA blocks (vs. 4 in Boltz-2). During pretraining, the MSA module processes up to `max_seqs`= 512 sequences. During fine-tuning, the MSA module is disabled (`max_seqs`= 0), and the model receives only the single input sequence with no alignment.

Template module. We disable the template module entirely (`use_templates=False`). No template embedder is instantiated and no template features are computed at any stage of training or inference.

Confidence and affinity modules. The confidence and affinity head modules present in Boltz-2 are not used. These heads were removed to reduce memory footprint and because our training objective does not require predicted confidence scores or affinity.

B Pretraining Details

B.1 Dataset

Our pretraining dataset is drawn from the Protein Data Bank (PDB), following the same structure processing and filtering methods outlined in the Boltz source code and paper Passaro et al. (2025). Specifically, we use a training cutoff date of June 1, 2023, leaving structures deposited after this date for our validation and test sets.

Because our target application is peptides (3–50 residues), we train on *protein fragments* of length 3–50 residues sampled from longer PDB chains. Each fragment is sampled by choosing a contiguous window of the desired length uniformly at random from a sampled protein chain, ensuring the window does not cross chain breaks. This allows us to make use of all proteins in the PDB. MSA coverage is reduced relative to full proteins; we therefore cap the number of MSA sequences at 512 and disable template search entirely. All other data processing, clustering, and sampling follow Passaro et al. (2025). Figure 5 shows the diffusion loss curves during pre-training.

B.2 Training Configuration

Table 4: Pretraining hyperparameters

Hyperparameter	Value
Effective batch size	128
Peak learning rate	1.8×10^{-3}
LR schedule	AF3: linear warmup 1,000 steps, decay $\times 0.95$ every 50,000 steps after step 50,000
Training steps	$\sim$8,400 optimizer steps
GPUs	4
GPU type	H100

Training follows the AF3 scheme with no concept of epochs; the training distribution is sampled continuously.

C Fine-tuning Details

C.1 Synthetic Sequence Generation

At each fine-tuning step, a peptide sequence is drawn from the PDB training set used in Phase 1 via the following procedure:

1. **Chain selection.** A single protein chain of length 4–52 residues is sampled uniformly from a pre-filtered chain index built from the training manifest. Each entry in the index

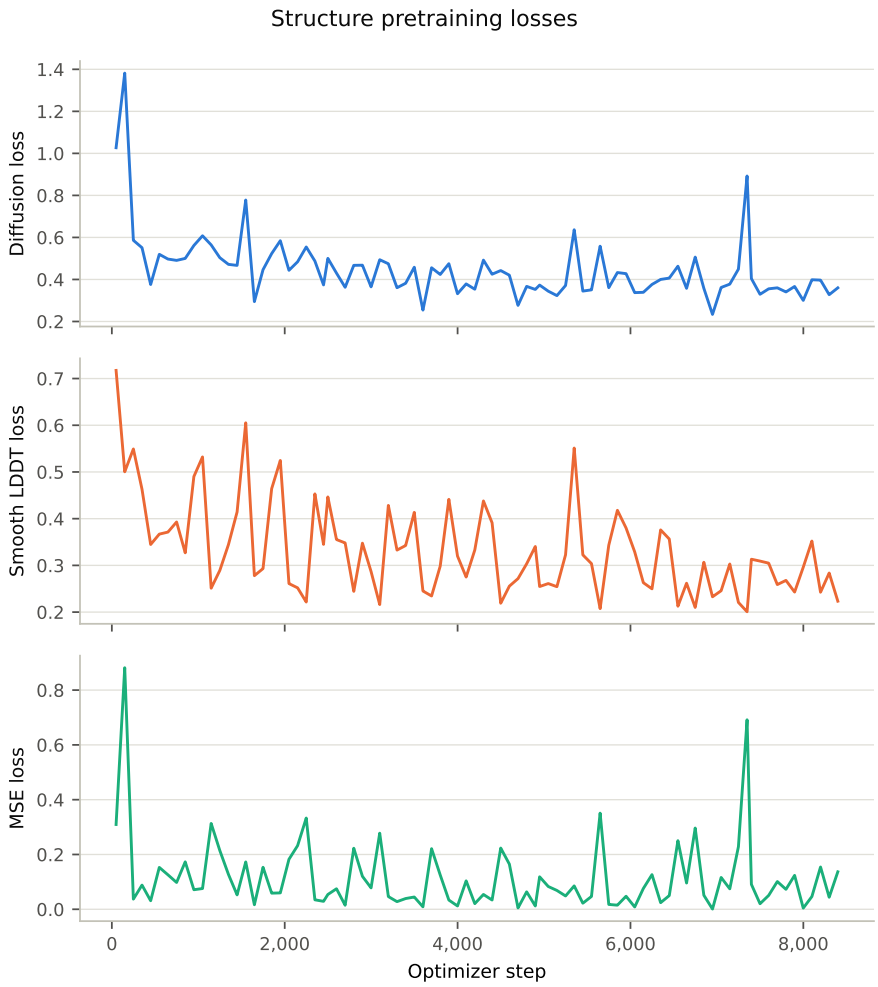


Figure 5: Training losses over ˜8,400 optimizer steps of structure pretraining. **Top:** diffusion loss. **Middle:** smooth LDDT loss. **Bottom:** coordinate MSE loss.

is a (PDB_id, chain_idx) pair that has already passed the length filter, so every draw is guaranteed valid without rejection sampling.

2. **Cap stripping.** Terminal capping residues (N-terminal ACE, C-terminal NH2) are removed before any further processing.

3. **ncAA substitution.** Non-standard residues already present in the sampled chain that are not in the 189-ncAA support set are replaced with a residue drawn uniformly from the support set. Each remaining position (whether a canonical or already-supported ncAA) is then independently mutated to a uniformly drawn ncAA with probability $p_{\text{ncAA}} = 0.2$.

4. **Cyclicity assignment.** The peptide is designated cyclic with probability $p_{\text{cyc}} = 0.4$, and linear with probability 0.6.

5. **YAML encoding.** The resulting residue list and cyclicity flag are encoded into a Boltz-compatible YAML input. Canonical residues are encoded by their one-letter code; ncAAs are encoded as CCD three-letter codes in the `modifications` field. The MSA field is set to `empty`, so no alignment is used.

The 189 supported ncAAs are defined by the set `SUPPORTED_NCAAS` in the differentiable hydrogen placement module (Section C.4).

C.2 Optimization Hyperparameters

Table 5: Fine-tuning hyperparameters

Hyperparameter	Value
Optimizer	AdamW
Learning rate	10^{-5}
Weight decay	0.1
(β_1, β_2)	$(0.9,\ 0.999)$
Gradient clip (norm)	1.0
DRaFT-K	10
Diffusion steps T	200
Conformations M	4
Trunk recycles N_r	3
Random seed	42
GPU	H100 ($1\times$)
Training Steps	9500 steps

The pretrained checkpoint is loaded and the model is fine-tuned end-to-end on both the diffusion and trunk weights, with SO3LR frozen throughout; each step processes one sequence with $M{=}4$ conformations.

Energy pooling. We use average pooling over the M conformations as the default. We also evaluate a max-pooling variant, $\mathcal{L} = \max_i E_{\mathrm{SO3LR}}(\hat{\mathbf{x}}_\theta^{(i)}(\mathbf{s}))$, which focuses the gradient on the worst-energy conformation; results are reported in Section D.

SO3LR inputs and charge computation. SO3LR (Kabylda et al., 2025) is an equivariant machine-learned force field that takes as input an *ASE* `Atoms` object specifying atomic positions, element symbols, and a scalar net charge, and returns the potential energy E (in eV) together with atomic forces $\mathbf{F} = -\partial E/\partial \mathbf{x}$ (in eV/Å). Because SO3LR is charge-aware, we compute an approximate net integer charge for each peptide at pH 7.4 using a lookup table of side-chain formal charges (`_RESIDUE_CHARGES` in the differentiable H-placement module). Only charged side-chain groups are accounted for. The resulting integer charge is passed to SO3LR via `atoms.info["charge"]` at each forward call. SO3LR is kept frozen throughout all fine-tuning; no parameters of the force field are updated.

C.3 Supported Non-Canonical Amino Acids

The differentiable hydrogen placement module is parametrised for the following 189 ncAAs (CCD three-letter codes). Only residues in this set may appear in fine-tuning sequences; canonical amino acids are always supported in addition. The 189 supported ncAA types were assembled in two stages. We first identified the most frequently occurring non-canonical residues across protein chains in the PDB and hand-specified differentiable H-placement rules for those residues. The set was then expanded incrementally as new ncAA types were encountered during curation of our test set.

A practical implication is that extending Neopep to new chemistries is straightforward: adding a new ncAA requires only writing one additional placement rule (specifying the hydrogen geometry for that residue type), and running further RL fine-tuning steps with the expanded residue pool. The 189-residue set listed below should therefore be seen as a starting point rather than a limit of the approach.

```
004, 00E, 0A1, 0W6, 28J, 2TL, 33X, 3EG, 3FG, 4HH, 4J2, 4WQ, 5CW, 66E,
6ZS, 7C9, 7T2, A1A5S, A1A5T, A1A8T, A1AL8, A1AMB, A1AMC, A1AT6, A1AT7,
A1B6Z, A1BC0, A1CCQ, A1CCR, A1CCS, A1CHA, A1CLH, A1CZ7, A1D5E, A1D5P,
A1EAK, A1ECM, A1EQM, A1EUR, A1I4F, A1I67, A1I8Q, A1I8R, A1ICB, A1ICC,
A1IFI, A1IJP, A1IKE, A1IUO, A1J0G, A1J16, A1JND, A1L3I, A1L4D, A1L7K,
A1L9X, A1MFG, ABA, AC5, ACA, AIB, ALO, ALY, B3S, BAL, BB8, BMT, CGU, CHG,
```

```
CSD, CSO, CSS, CSX, D4P, DAB, DAL, DAM, DAR, DAS, DGL, DGN, DI8, DIL,
DLE, DLY, DPN, DPP, DPR, DSG, DSN, DTH, DTR, DTY, DVA, EEP, FME, G5G,
GHP, HIC, HJV, HPE, HRG, HY3, HYP, IAE, IAS, IC0, IIL, IML, JGO, JXR,
JYC, KBE, KCR, KJW, LE1, LTU, M70, MAA, MEA, MHS, MLE, MLU, MLY, MLZ,
MVA, MYN, N2C, N9P, NAL, NCY, NH2, NIY, NLE, NVA, NZC, O7G, OCS, OIC,
OMY, OMZ, ORD, ORN, OUK, OUR, PCA, PHA, PIV, PPN, PQ4, PTR, QDD, QOL,
QQB, QUJ, QUK, QVE, QVS, SAC, SAR, SEP, SFE, SLL, SNC, SOQ, TBG, TIH,
TPO, TY5, TYS, UAL, UKD, UN1, URL, UX8, WEH, WFP, WLU, WPA, WVL, WWB,
                X60, XRW, YC3, YNM, YYA, ZAE, ZGL, ZY9
```

C.4 Hydrogen Placement Geometry Rules

The differentiable hydrogen placement module, takes inspiration from the differentiable hydrogen adder in King & Koes (2021), and assigns each hydrogen atom a *placement rule* determined by its local bonding environment. Six rules are defined; all operations are implemented in PyTorch and are fully differentiable with respect to the heavy-atom coordinates. In each rule, C denotes the heavy parent atom, $R_1, R_2, \ldots$ denote its bonded reference heavy atoms, d is the target C–H bond length, $\hat{\cdot}$ denotes ℓ_2 normalisation, and $\times$ denotes the cross product.

Rule 1: Bisector (sp^2, 2 reference atoms). Used for aromatic, carbonyl, and planar amine hydrogens. The hydrogen lies in the plane spanned by R_1–C–R_2, along the bisector of the supplementary angle:

$$\mathbf{b} = 2C - R_1 - R_2, \qquad \mathbf{h} = C + d\,\hat{\mathbf{b}}. \tag{7}$$

Rule 2: Methine (sp^3, 2 reference atoms). Used for CH groups where the hydrogen lies opposite the bisector of the R_1–C–R_2 angle, placing it in a roughly tetrahedral geometry. The construction is identical to Rule 1:

$$\mathbf{b} = 2C - R_1 - R_2, \qquad \mathbf{h} = C + d\,\hat{\mathbf{b}}. \tag{8}$$

The distinction from the bisector rule is purely in which bonding environment triggers it (sp^3 single-bond context vs. sp^2 double-bond context).

Rule 3: Tetrahedral fourth (sp^3_4th, 3 reference atoms). Used for CH groups in a tetrahedral environment with 3 known substituents. The hydrogen is placed opposite the centroid of the three reference atoms:

$$\mathbf{b} = 3C - R_1 - R_2 - R_3, \qquad \mathbf{h} = C + d\,\hat{\mathbf{b}}. \tag{9}$$

Rule 4: Methylene (geminal pair, 2 reference atoms). Used for CH_2 groups. A local orthonormal frame is constructed from the bisector direction $\mathbf{u}$ and the plane normal $\mathbf{v}$:

$$\mathbf{u} = \widehat{(R_1 - C) + (R_2 - C)}, \tag{10}$$

$$\mathbf{v} = \widehat{(R_1 - C) \times (R_2 - C)}. \tag{11}$$

The two geminal hydrogens are placed symmetrically about the C–$\mathbf{u}$ axis:

$$\mathbf{h}_{1,2} = C + d(\sin\alpha\,\mathbf{u} \pm \cos\alpha\,\mathbf{v})\,, \tag{12}$$

where α is the half-opening angle derived from the ideal H–C–H bond angle for the local geometry.

Rule 5: Methyl (cone, 3 hydrogens). Used for CH_3 groups. Three hydrogens are placed on a cone of half-angle θ around the C–R_1 bond axis. An orthonormal frame $(\mathbf{a}, \mathbf{m}, \mathbf{n})$ is constructed with $\mathbf{a} = \widehat{C - R_1}$ as the axis:

$$\mathbf{h}_k = C + d\big(\cos\theta\,\mathbf{a} + \sin\theta\big(\cos\phi_k\,\mathbf{m} + \sin\phi_k\,\mathbf{n}\big)\big)\,, \quad \phi_k = \phi_0 + \tfrac{2\pi(k-1)}{3}, \quad k = 1, 2, 3, \tag{13}$$

where ϕ_0 is determined by the dihedral of the first hydrogen relative to a reference atom on R_1, and θ is the ideal tetrahedral cone half-angle.

Rule 6: NERF. The Natural Extension Reference Frame (NERF) rule handles hydrogens whose placement requires a dihedral angle ϕ relative to a third reference atom A, in addition to the bond length d and bond angle θ. Given ordered reference atoms A, B, C (with the hydrogen bonded to C), an orthonormal local frame is constructed:

$$\mathbf{bc} = \widehat{C - B}, \tag{14}$$

$$\mathbf{n} = \widehat{(B - A)} \times \mathbf{bc}, \tag{15}$$

$$\mathbf{m} = \mathbf{n} \times \mathbf{bc}. \tag{16}$$

The hydrogen position is then:

$$\mathbf{h} = C + d\big(-\cos\theta\,\mathbf{bc} + \sin\theta\big(\cos\phi\,\mathbf{m} + \sin\phi\,\mathbf{n}\big)\big)\,. \tag{17}$$

All parameters (d, θ, ϕ) are precomputed from the CCD geometry at preprocessing time and stored as a `HPlacementSpec` dataclass.

D Ablations

Number of conformations M. At each fine-tuning step, M heavy-atom conformations are sampled from the diffusion model, protonated, and scored by SO3LR; their energies are averaged to form the reward signal. Larger M reduces gradient variance at the cost of $M\times$ more MLFF evaluations per step.

Figure 6 shows energy per atom over 1,200 training steps for $M \in \{4, 8, 12, 16\}$. All four settings converge to approximately -1.50 to -1.54 eV/atom, with no statistically meaningful difference in final energy. The rolling means are nearly indistinguishable after step $\sim$600, indicating that the variance reduction from larger M does not translate into a better optimum within this training budget. We adopt $M = 4$ as the default.

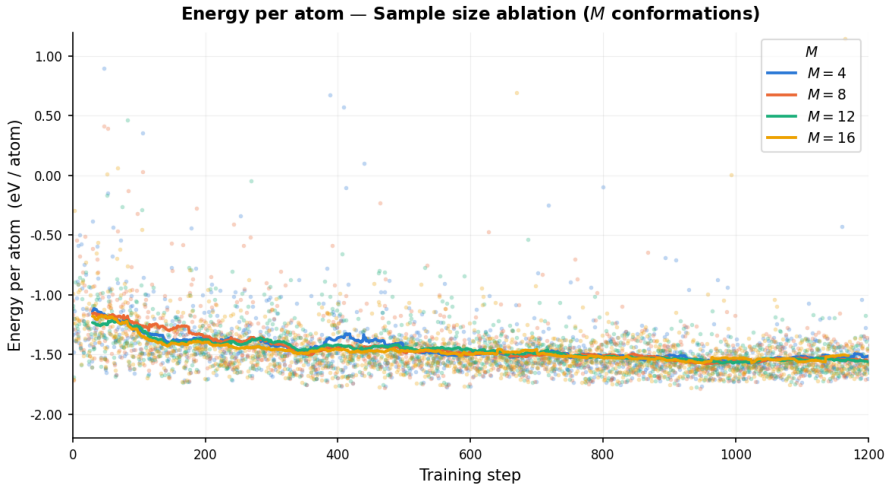


Figure 6: Energy per atom (eV/atom, lower is better) during RL fine-tuning for $M \in \{4, 8, 12, 16\}$ sampled conformations. Dots show per-step values; solid lines show a 60-step rolling mean. All variants converge to the same final energy, motivating the choice of $M = 4$.

Energy pooling: average vs. maximum. During RL fine-tuning, $M = 4$ conformations are sampled per peptide and scored by SO3LR; the per-conformation energies must be aggregated into a single reward signal for the gradient update. We compare two strategies: average pooling ($\bar{E} = \frac{1}{M}\sum_i E_i$) and max-pooling ($\bar{E} = \max_i E_i$), which focuses the gradient on the highest-energy (least stable) conformation found.

Figure 7 shows the energy per atom (eV/atom) over [1,200] fine-tuning steps for both variants. Both methods drive the mean energy from approximately -1.1 eV/atom to -1.5 eV/atom over the course of training. The two strategies converge to similar final energies, with virtually no difference in training performance. We adopt average pooling as the default.

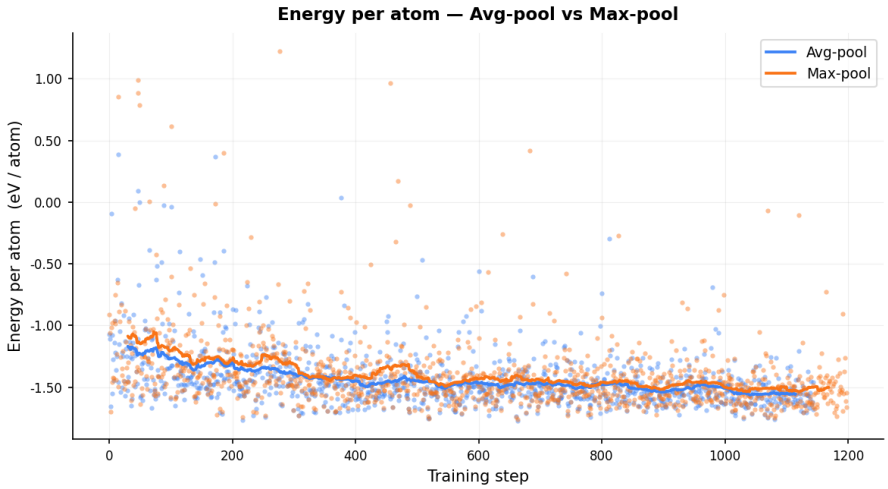


Figure 7: Energy per atom (eV/atom, lower is better) during RL fine-tuning for average pooling (blue) and max-pooling (orange) over $M = 4$ sampled conformations. Dots show per-step values; solid lines show a rolling mean.

D.1 Per-Peptide Results: Non-Canonical Test Set

Table 6 reports per-peptide minimum CA-RMSD, minimum FA-RMSD, and average CA-RMSD (mean $\pm$ std over 25 predictions) for all 34 non-canonical test peptides. Dashes indicate peptides not supported by Protenix. Bold marks the best (lowest) value in each metric row.

E Limitations

A key assumption throughout this work is the exclusion of explicit solvent. Solvation effects are well-known to influence biomolecular structure, yet incorporating them into our training scheme was infeasible for two reasons: the limited compute available to us, and the additional complexity introduced by the stochasticity of solvent configurations — a single peptide can adopt many valid solvation shells, and capturing this effect faithfully would require averaging over an ensemble of such configurations during training. We therefore restrict our structures to include hydrogen atoms and leave the incorporation of solvation effects to future work.

We also observed that the RL-based energy fine-tuning depends critically on the quality of the structural prior. When the candidate structures generated by the pretrained model are severely flawed, the RL mechanism has little to build on, and fine-tuning fails to recover a physically reasonable geometry. In other words, the energy-guided RL stage is only as strong as the prior it refines. We note that this conclusion is based on qualitative inspection of a small number of pretrained versus fine-tuned structures; we did not conduct systematic ablations or quantitative experiments to characterize this failure mode, and we leave a rigorous analysis to future work.

Lastly, since our model was trained via RL on only a fixed number of ncAAs, we cannot guarantee generalizability across the space of all possible ncAAs, though the architecture permits this in principle, as it operates on atom-level granularity. We leave future work on building a more generalizable model that can generalize to unseen ncAAs during training.

Table 6: Per-peptide RMSD results (Å) for the non-canonical test set (n=34). Min = best of 25 predictions; Avg = mean ± SD over 25 predictions. Protenix evaluated on n=29 (5 peptides marked — not supported).

	Boltz-2			**Pretrained**			**FT-9500**			**Protenix**		
Peptide	Min CA	Min FA	Avg CA	Min CA	Min FA	Avg CA	Min CA	Min FA	Avg CA	Min CA	Min FA	Avg CA
22es_F	5.273	6.542	16.12±9.12	1.698	2.951	2.30±0.36	**0.722**	**2.259**	**1.78**±0.45	1.660	2.996	2.25±0.29
24hr_I	4.373	6.052	7.10±1.35	2.382	3.444	3.36±0.69	**2.361**	**3.919**	**2.97**±0.37	—	—	—
26nl_G	4.226	6.612	7.83±2.19	3.267	3.932	3.72±0.30	**1.088**	**3.306**	**2.28**±0.78	—	—	—
8fvm_C	4.233	5.937	6.22±1.54	2.609	4.595	3.70±0.31	**1.669**	**3.359**	3.47±0.69	2.171	4.119	**3.01**±0.31
8g82_A	5.703	6.971	9.30±2.12	2.899	4.536	4.06±0.64	1.790	3.891	2.59±0.40	**0.189**	**0.625**	**0.33**±0.09
8i82_C	5.491	7.581	14.86±6.05	1.996	3.637	2.77±0.39	**1.621**	**3.215**	**2.70**±0.73	2.130	2.561	2.84±0.16
8p94_U	2.317	3.034	3.33±0.75	1.341	2.545	1.95±0.32	**0.507**	**1.660**	**1.40**±0.52	1.001	2.315	1.53±0.33
8pxy_B	3.783	5.026	32.49±23.44	2.461	3.798	3.07±0.44	**0.270**	**1.836**	**1.30**±0.54	1.877	2.602	2.34±0.22
8r3h_F	1.583	2.949	3.31±0.84	1.226	2.356	2.20±0.36	**0.581**	**1.722**	**1.20**±0.42	1.365	2.965	1.71±0.09
8r7t_C	7.947	7.833	10.59±1.40	1.935	2.730	2.77±0.42	1.697	3.300	2.53±0.46	**0.488**	**1.034**	**0.97**±0.58
8s4l_A	4.310	4.982	5.71±0.77	3.032	4.172	3.66±0.32	2.978	4.032	3.54±0.47	**2.434**	**3.340**	**2.48**±0.06
8s8n_A	4.556	5.158	6.27±1.14	3.665	4.986	4.10±0.23	3.363	4.632	4.33±0.69	**1.812**	**2.361**	**2.45**±0.35
8u78_A	2.859	4.327	4.60±1.03	2.816	3.914	5.64±0.93	2.700	4.053	4.53±0.81	**1.768**	**2.630**	**2.14**±0.17
8v4b_A	8.583	8.696	10.76±1.50	4.482	5.586	6.31±0.64	**3.665**	**5.375**	**5.74**±1.08	4.866	6.633	5.74±0.45
8w6z_B	3.334	4.498	6.06±1.27	2.508	4.198	3.16±0.39	**1.929**	**3.627**	**2.56**±0.34	2.071	3.839	2.56±0.18
8xip_F	4.013	6.402	6.84±1.86	1.220	3.252	1.96±0.50	**0.360**	**2.623**	**1.12**±0.35	2.170	4.881	3.29±0.52
8xp8_C	4.696	5.398	19.30±15.93	1.712	2.277	2.66±0.51	0.839	1.812	1.86±0.46	**0.751**	**1.348**	**0.91**±0.07
8zcj_F	4.173	6.289	8.20±2.89	1.416	3.490	2.63±0.57	**0.537**	**2.340**	**1.05**±0.28	1.274	2.738	1.45±0.08
9bbj_B	5.378	6.957	8.41±2.24	1.973	3.955	2.70±0.56	**0.632**	**2.751**	**1.69**±0.35	1.469	2.522	4.16±0.72
9bic_A	2.976	3.753	4.61±1.07	**1.699**	3.454	**2.52**±0.58	1.768	**3.333**	2.88±0.76	2.419	4.170	2.85±0.36
9din_B	6.374	6.926	8.65±2.68	1.475	3.980	2.59±0.44	**1.118**	**3.682**	**1.85**±0.27	4.250	5.349	4.50±0.11
9fbi_C	2.334	4.134	4.37±0.82	1.457	3.007	2.32±0.50	**0.572**	**2.678**	**1.06**±0.22	2.813	3.570	3.26±0.26
9hty_D	2.909	3.891	5.85±1.41	1.355	2.557	2.87±0.55	**0.654**	**1.663**	**1.58**±0.59	2.408	3.022	3.60±0.45
9ik8_F	5.560	6.956	8.40±2.14	1.592	4.302	2.98±0.76	**0.729**	**2.843**	**1.15**±0.24	1.020	2.923	1.18±0.10
9ivc_E	4.358	5.707	5.86±1.21	2.036	3.547	2.95±0.34	1.533	3.260	2.30±0.37	**1.108**	**2.753**	**1.89**±0.47
9p4h_B	5.140	7.233	8.03±2.63	2.079	4.396	2.71±0.28	**1.099**	**3.350**	**1.70**±0.22	2.459	4.794	2.58±0.05
9qb4_CA	4.392	6.695	13.11±8.72	1.846	3.970	3.05±0.58	**1.552**	**2.858**	**2.45**±0.60	3.441	5.171	4.41±0.20
9qdu_B	5.664	6.660	12.94±10.32	2.600	3.479	**3.82**±0.69	**2.938**	**3.790**	3.63±0.58	—	—	—
9rfw_HA	4.451	5.688	10.83±10.23	2.757	4.616	3.67±0.51	**2.694**	**4.389**	3.96±0.59	3.716	5.185	**4.41**±0.41
9sab_F	2.864	5.640	6.85±2.03	4.014	3.856	4.83±0.38	**1.497**	**3.038**	**2.69**±0.63	—	—	—
9upj_C	7.021	8.351	10.42±1.64	2.627	3.498	2.97±0.23	**1.770**	**3.494**	**2.50**±0.45	2.085	4.222	2.49±0.16
9upk_C	5.248	6.129	9.33±1.97	2.533	3.931	2.95±0.38	**1.680**	**2.752**	**2.45**±0.43	5.138	5.926	5.47±0.13
9v8f_B	**2.296**	**3.690**	**5.26**±1.12	3.327	5.234	4.04±0.50	3.368	5.168	4.62±0.70	5.127	6.735	5.72±0.25
9w2h_C	2.822	5.129	5.81±2.29	1.181	2.803	3.01±1.04	**0.924**	**2.586**	3.12±0.97	—	—	—