PACKORA: SYSTEMATIC DESIGN FOR GENERATIVE MOLECULAR CRYSTAL STRUCTURE PREDICTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Molecular crystal structure prediction (CSP) is important in pharmaceuticals, agrochemicals, and organic electronics, where subtle differences in molecular conformation and packing can strongly affect material properties. We present PACKORA, a generative model for molecular CSP that jointly predicts atomic coordinates and the lattice from molecular graphs. To accommodate diverse crystal compositions and available structural information, PACKORA offers greater conditioning flexibility than existing approaches, accepting any subset of molecular conformers, stereochemical labels, and space-group information while supporting multi-component and organometallic crystals. Beyond this conditioning flexibility, PACKORA integrates effective design choices for atomistic generative modeling: cacheable pairwise refinement, effective training objectives and numerical solvers, conditioning dropout, and balanced scaling of pairwise and single representations. We also report systematic studies of architecture, training, conditioning, inference, and scaling that we conducted to make the design choices. Finally, we introduce a new evaluation suite for generative molecular CSP that separates generation and ranking, following the CCDC CSP blind test protocol. PACKORA outperforms baselines in both settings, achieving the highest coverage across all six generation benchmarks, higher experimental-form recovery, lower experimental-form ranks, and faster convergence in ranking.

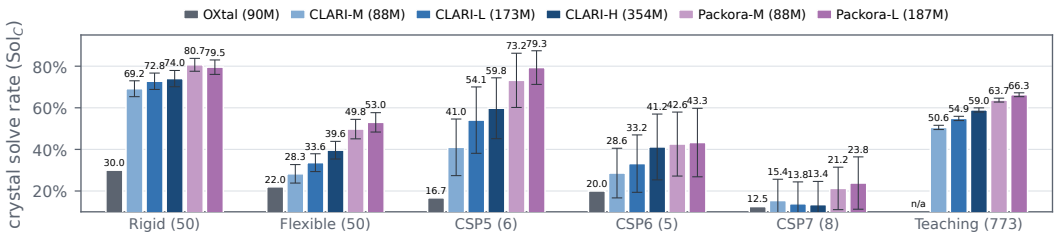


Figure 1: **Crystal solve rate with 30 candidates per target.** We report the crystal solve rate Sol_C, with parentheses indicating benchmark sizes. OXtal results are published 30-candidate scores. CLARI and PACKORA results show means and standard deviations over 5,000 bootstrap resamples of 30 from 1,000 candidates per target. PACKORA outperforms baselines across all six benchmarks.

1 INTRODUCTION

Molecular crystal structure prediction (CSP) is a key challenge in developing molecular materials for pharmaceuticals, agrochemicals, and organic electronics (Beran, 2023). The same molecule can crystallize into different polymorphs with distinct molecular conformations and packing arrangements, leading to differences in solubility, stability, bioavailability, and transport properties (Price, 2014; Jin et al., 2025; Gharakhanyan et al., 2025). Anticipating these forms has motivated decades of research on molecular CSP (Lommerse et al., 2000).

Molecular CSP typically combines candidate generation with energy-based relaxation and ranking (Hunnisett et al., 2024a;b). Traditional computational approaches generate candidate structures through random or evolutionary search (Li et al., 2018; Curtis et al., 2018) and evaluate them using density functional theory (DFT), making the evaluation of large candidate sets computationally

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

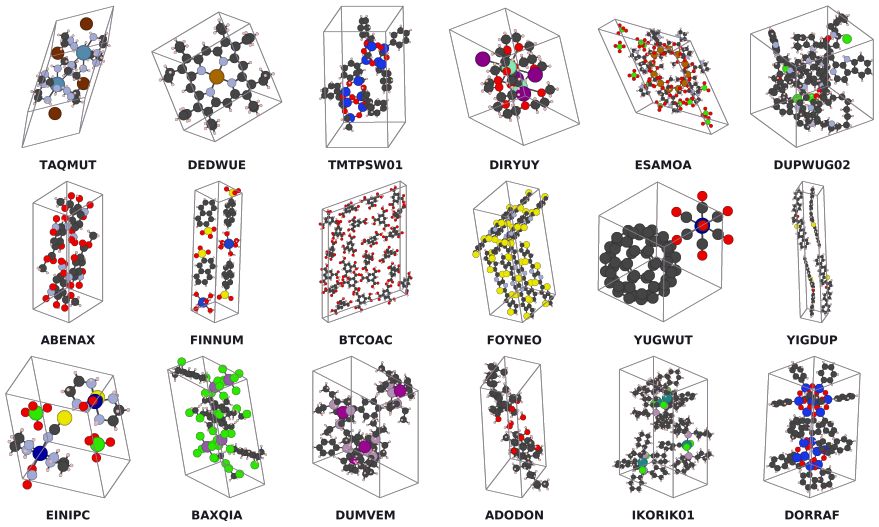


Figure 2: **Crystal structures predicted by** PACKORA**.** PACKORA predicts diverse molecular crystals across molecular sizes, chemical systems, and packing motifs. Panel labels identify CSD refcodes.

expensive. FastCSP accelerates this process by replacing DFT with a machine-learning interatomic potential (MLIP), but still relies on random candidate generation (Gharakhanyan et al., 2025).

Generative models accelerate candidate generation by learning from experimental crystal structures, but existing methods differ in their inputs, designs, and post-processing pipelines (Jin et al., 2025; Zeng et al., 2026; Subramanian et al., 2026; Lo et al., 2026). These differences make it difficult to identify which choices improve the generator itself. For practical molecular CSP, models should sample efficiently, use optional conditions when available, and achieve broad coverage within limited candidate budgets. We therefore systematically study architecture, training, conditioning, inference, and scaling for generative molecular CSP.

These studies guide the design of PACKORA. We identify cacheable pairwise refinement for efficient sampling, conditioning dropout for optional conditions, and balanced scaling of single and pair-track widths for improved target coverage. The resulting model jointly predicts atomic coordinates and the lattice from molecular graphs, supports multi-component and organometallic crystals, and optionally conditions on molecular conformers, stereochemistry, and space group. Following the seventh CSP blind test (Hunnisett et al., 2024a;b), we evaluate structure generation directly and structure ranking for end-to-end performance. PACKORA outperforms baselines across six generation benchmarks and achieves higher experimental-form recovery and better ranks on two ranking benchmarks.

Our contributions are threefold:

- We introduce PACKORA for molecular CSP that handles diverse chemical systems with flexible conditioning on molecular conformers, stereochemical labels, and space-group information.
- We conduct systematic studies of architecture, training, conditioning, inference, and scaling that demonstrate the strength of our design choices for molecular CSP.
- We introduce an evaluation suite that isolates gains in candidate generation and assesses their impact on practical CSP workflows by evaluating generator quality directly and end-to-end performance under a shared relaxation-and-ranking pipeline.

2 RELATED WORK

Search-based molecular CSP. Traditional molecular CSP searches over molecular conformations and crystal packings, then relaxes and ranks candidates by energy (Day et al., 2009; Price, 2014; Reilly et al., 2016; Beran, 2023). Search strategies include constrained random generation in Genarris (Li et al., 2018) and evolutionary search in GAtor (Curtis et al., 2018). FastCSP accelerates relaxation and ranking using a machine-learning interatomic potential (MLIP) while retaining random candidate

Table 1: **Generative molecular CSP capabilities.** ✓ and ✗ denote supported and unsupported capabilities. For conformers, ✓, ✗, and ▲ denote optional, unsupported, and required conditioning.

	Generation	Chemical/data scope		Conditioning inputs		
Method	Explicit H output	Multi-component	Organo-metallics	3D conformer	Stereo labels	Space group
OXtal (Jin et al., 2025)	✗	✓	✓	▲	✗	✗
PackFlow (Subramanian et al., 2026)	✗	✗	✗	✗	✗	✗
MolCrystalFlow (Zeng et al., 2026)	✓	✗	✗	▲	✗	✗
CLARI (Lo et al., 2026)	✓	✓	✓	✗	✗	✗
PACKORA (Ours)	✓	✓	✓	✓	✓	✓

generation (Gharakhanyan et al., 2025). However, exploring the large conformational and packing space requires evaluating many candidates, making these workflows computationally expensive.

Generative molecular CSP. Generative methods learn plausible crystal structures from experimental data. OXtal uses all-atom diffusion on lattice-free molecular crops with AlphaFold 3-based architecture (Abramson et al., 2024; Jin et al., 2025). MolCrystalFlow models molecules as rigid bodies (Zeng et al., 2026); PackFlow uses reinforcement-learning post-training with MLIP (Subramanian et al., 2026); and CLARI generates explicit unit cells with pair-bias attention, reporting a 15–30× speedup over OXtal (Lo et al., 2026). None supports independently optional conformers, stereochemical labels, and space-group conditioning within a single model. Table 1 compares their capabilities with ours. See Appendix A for related work on inorganic crystals and metal-organic frameworks.

3 METHOD

PACKORA is a flow-based model that predicts atomic coordinates and the lattice from molecular graphs, a formula-unit count Z, and optional auxiliary information. Its design follows a recipe for molecular CSP established through our systematic study of architecture, training, conditioning, inference, and scaling in Section 5.

3.1 PROBLEM FORMULATION

Crystal representation. We represent a crystal $\mathcal{M} = (\mathbf{A}, \mathbf{X}, \mathbf{L})$ by atom types $\mathbf{A} \in \mathcal{A}^N$, Cartesian coordinates $\mathbf{X} \in \mathbb{R}^{N\times 3}$, and a lattice $\mathbf{L} \in \mathbb{R}^{3\times 3}$ whose rows define the periodic translations. We canonicalize the lattice using Niggli reduction and a rotation-invariant Cholesky construction, then parameterize its lower-triangular form by an unconstrained vector $\boldsymbol{\ell} \in \mathbb{R}^6$ following Veljković et al. (2026). Details are provided in Appendix B.

Molecular crystal structure prediction (CSP). For K molecular components, let $\mathcal{G} = \{G_k\}_{k=1}^K$ denote their molecular graphs and $\mathbf{r} = (r_1, \ldots, r_K)$ their stoichiometry. Each G_k specifies the atom types, bond types, and formal charges of component k, while r_k gives its number of copies per formula unit. Given a formula-unit count Z, the crystal contains $N = Z\sum_{k=1}^K r_k N_k$ atoms, where N_k is the number of atoms in component k. We additionally allow optional conditions $\mathcal{O}$, including molecular conformers (e.g., generated with RDKit (Landrum et al., 2006)), stereochemical annotations, and a space-group label. Collecting these inputs as $\mathcal{C} = (\mathcal{G}, \mathbf{r}, Z, \mathcal{O})$, we model $p_\theta(\mathbf{X}, \boldsymbol{\ell} \mid \mathcal{C})$.

Structure generation and ranking. Given a condition $\mathcal{C}$, each method generates B candidate crystals $\{\mathcal{M}^{(i)}\}_{i=1}^B$. For structure generation, we evaluate candidates directly and count a prediction as successful if any collision-free candidate matches the experimental reference $\mathcal{M}^\star$. For structure ranking, we apply a shared relaxation pipeline and rank the B' retained crystals $(\mathcal{M}'^{(1)}, \ldots, \mathcal{M}'^{(B')})$ by increasing lattice energy. The experimental-form rank is $\mathrm{rank}(\mathcal{M}^\star) = \min\{j : \mathrm{Match}(\mathcal{M}'^{(j)}, \mathcal{M}^\star) = 1\}$, where Match indicates a structural match and lower ranks are better.

3.2 GENERATIVE FRAMEWORK

We use variational flow matching (VFM; Eijkelboom et al., 2024), which generalizes flow matching (Lipman et al., 2023) through a flexible choice of the variational posterior (Zaghen et al., 2025).

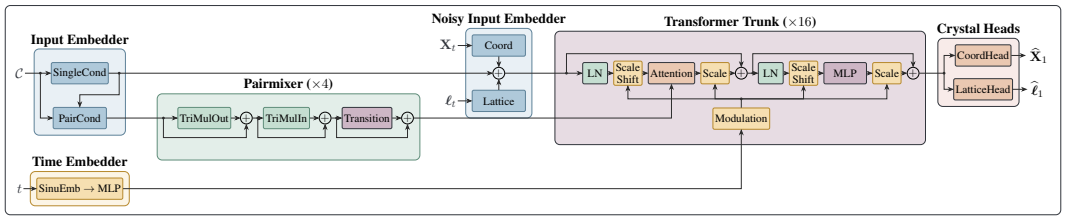


Figure 3: **Model architecture of** PACKORA**.** The condition $\mathcal{C}$ is embedded into single and pair representations, with pairs refined by PAIRMIXER (Ouyang-Zhang et al., 2025). Embeddings of noisy coordinates $\mathbf{X}_t$ and lattice $\boldsymbol{\ell}_t$ enter the single representation before a DiT trunk, which uses refined pairs as attention biases. Separate heads predict the coordinates and the lattice.

Training objective. Writing the joint coordinate and lattice state as $\mathbf{y} = (\mathbf{X}, \boldsymbol{\ell})$, let $(\mathbf{y}_1, \mathcal{C})$ denote a crystal state and its condition, and let $\mathbf{y}_0 \sim p_0$ be a prior sample. We use the linear conditional path to interpolate between the prior and data endpoints, $\mathbf{y}_t = (1-t)\mathbf{y}_0 + t\mathbf{y}_1$.

Given a noisy state $\mathbf{y}_t$, VFM learns a variational posterior $q_\theta(\mathbf{y}_1 \mid \mathbf{y}_t, t, \mathcal{C})$ that approximates the true posterior by minimizing the expected negative log-likelihood, $\mathcal{L}_{\text{VFM}}(\theta) = -\mathbb{E}_{\mathbf{y}_1,\mathbf{y}_t,t}[\log q_\theta(\mathbf{y}_1 \mid \mathbf{y}_t, t, \mathcal{C})]$. The variational family determines the form of this objective. We model the posterior as a Laplace distribution with a fixed scale, which yields L^1 loss on the predicted endpoints.

The model parameterizes the mean of the variational posterior, which is equivalent to predicting $\mathbf{y}_1$ from $\mathbf{y}_t$ and $\mathcal{C}$ as $\widehat{\mathbf{X}}_1, \widehat{\boldsymbol{\ell}}_1 = \boldsymbol{\mu}_t^\theta(\mathbf{y}_t, \mathcal{C})$. The objective simplifies to:

$$\mathcal{L}_{\text{VFM}}(\theta) = \mathbb{E}_{\mathbf{y}_1,\mathbf{y}_t,t}\left[\frac{\lambda_{\text{coord}}}{3N}\left\|\widehat{\mathbf{X}}_1 - \mathbf{X}_1\right\|_1 + \frac{\lambda_{\text{lattice}}}{6}\left\|\widehat{\boldsymbol{\ell}}_1 - \boldsymbol{\ell}_1\right\|_1\right], \tag{1}$$

where λ_{coord} and λ_{lattice} are loss weights for the coordinate and lattice components, respectively.

Sampling. We generate samples using the stochastic differential equation

$$d\mathbf{y}_t = [\mathbf{v}_t^\theta(\mathbf{y}_t) + (g_t^2/2)\mathbf{s}_t^\theta(\mathbf{y}_t)]\,dt + g_t\,d\mathbf{W}_t.$$

For the linear path, $\mathbf{v}_t^\theta(\mathbf{y}_t) = (\boldsymbol{\mu}_t^\theta(\mathbf{y}_t, \mathcal{C}) - \mathbf{y}_t)/(1-t)$ and $\mathbf{s}_t^\theta(\mathbf{y}_t) = (t\,\mathbf{v}_t^\theta(\mathbf{y}_t) - \mathbf{y}_t)/(1-t)$. Here, g_t controls stochasticity and $\mathbf{W}_t$ is a standard Wiener process.

3.3 MODEL ARCHITECTURE

As shown in Figure 3, our model $\boldsymbol{\mu}_t^\theta$ combines Pairmixer-based pair refinement (Ouyang-Zhang et al., 2025) with a Diffusion Transformer (DiT) (Peebles & Xie, 2023). It embeds the conditions, refines pair representations, incorporates the noisy crystal state, and predicts coordinates and the lattice through separate heads. See Appendix C for the complete algorithm.

Condition embedding. The input embedder maps the condition $\mathcal{C}$ to per-atom single features $\mathbf{S}_\mathcal{C} = E_{\text{single}}(\mathcal{C})$ and pairwise features $\mathbf{P}^{(0)} = E_{\text{pair}}(\mathbf{S}_\mathcal{C}, \mathcal{C})$, with learned embedding networks E_{single} and E_{pair}. Single features encode atom identity, periodic-table descriptors, formal charge, optional template coordinates, chirality, and space group. Pair features combine projected single features with intramolecular bond types, bond stereochemistry, template displacements, and distances. Availability masks and learned null states distinguish missing conditions from provided values.

Pairmixer. Pairmixer (Ouyang-Zhang et al., 2025) refines pairwise representations by combining information across atom triplets through triangle multiplication. It retains Pairformer's incoming and outgoing triangle multiplication while omitting triangle attention (Abramson et al., 2024). Each block applies sequential residual updates to pair features $\mathbf{P} \in \mathbb{R}^{N\times N\times d_{\text{pair}}}$:

$$\mathbf{P} \leftarrow \mathbf{P} + \text{TriMul}_{\text{out}}(\mathbf{P}), \qquad \mathbf{P} \leftarrow \mathbf{P} + \text{TriMul}_{\text{in}}(\mathbf{P}), \qquad \mathbf{P} \leftarrow \mathbf{P} + \text{Transition}(\mathbf{P}),$$

where $\text{TriMul}_{\text{out}}$ and $\text{TriMul}_{\text{in}}$ denote outgoing and incoming triangle multiplication. The transition is a gated feed-forward network applied independently to each atom pair. Pairmixer produces $\mathbf{P}^\star$ while leaving the single representation unchanged.

Noisy-state injection. We inject the noisy crystal state as $\mathbf{S}_t = \mathbf{S}_\mathcal{C} + E_X(\mathbf{X}_t) + \text{Broadcast}(E_L(\boldsymbol{\ell}_t))$, using Fourier embeddings for coordinates (Tancik et al., 2020) and a global lattice embedding broadcast to all atoms. A sinusoidal time embedding separately conditions the transformer.

DiT trunk. Each DiT block updates the single representation with self-attention and a feed-forward network. For head h, the attention logits are $\mathbf{A}^{(h)} = \mathbf{Q}^{(h)}(\mathbf{K}^{(h)})^\top/\sqrt{d_h} + b_h(\mathbf{P}^\star)$, where d_h is the head dimension and b_h maps pair features to an attention bias. The time embedding modulates normalization and residual gates throughout the trunk (Peebles & Xie, 2023).

Prediction heads. Separate heads map normalized single features to predicted coordinates $\widehat{\mathbf{X}}_1$ and lattice parameters $\widehat{\boldsymbol{\ell}}_1$. The coordinate head linearly projects each atom's features to a 3D position and subtracts the mean position over valid atoms. The lattice head mean-pools the features over valid atoms and applies an MLP to predict the six lattice parameters.

4 Benchmark Evaluation

We evaluate structure generation (Section 4.2) directly from model predictions, and structure ranking (Section 4.3) after applying a shared relaxation-and-ranking pipeline to those predictions.

4.1 Experimental Setup

Dataset. We use the official CLARI training and validation split (Lo et al., 2026) from the Cambridge Structural Database (CSD) (Groom et al., 2016), which excludes test family and molecular-component overlap. After hydrogen completion and applying a 512-atom unit-cell limit, we retain 912,807 training and 1,047 validation structures. Preprocessing details and dataset statistics are in Appendix D.

Setup. We train PACKORA-M and PACKORA-L on eight NVIDIA H200 GPUs. Training proceeds in two stages: first on crystals with at most 300 atoms, then on crystals with up to 512 atoms (Zhou et al., 2025). We select checkpoints by validation Sol_C. At inference, we use stereochemistry and RDKit conformers (Landrum et al., 2006) when available, but omit space-group labels, which are generally unavailable in CSP blind tests (Reilly et al., 2016). We use 200-step EDM–Heun sampling (Karras et al., 2022); see Appendices E and F for settings and computational costs.

4.2 Structure Generation

Benchmarks. We evaluate on six benchmarks: OXtal Rigid and Flexible (Jin et al., 2025); CSP5, CSP6, and CSP7 (Bardwell et al., 2011; Reilly et al., 2016; Hunnisett et al., 2024b); and CLARI's CSD Teaching Subset (Battle et al., 2010; Lo et al., 2026).

Metric and protocol. Following Jin et al. (2025) and Lo et al. (2026), we report solve rate Sol_C at 30 and 1,000 candidates. A target is solved if a collision-free candidate matches at least 8 of 15 molecules with $\mathrm{RMSD}_{15} < 2$ Å under COMPACK (Chisholm & Motherwell, 2005); strict $\mathrm{Sol}_C^{15/15}$ requires all 15. We bootstrap 30-candidate coverage from each target's 1,000-candidate pool (Appendix G).

Baselines. We compare against OXtal and CLARI-M, L, and H (Jin et al., 2025; Lo et al., 2026). To ensure a consistent evaluation, we re-evaluate official CLARI checkpoints under the OXtal protocol to resolve differences in collision detection and experimental-reference selection (Appendix G). We exclude PackFlow (Subramanian et al., 2026) and MolCrystalFlow (Zeng et al., 2026) because their chemical/data scope and modeling assumptions (e.g., rigid-body generation) differ from our setting.

Results. PACKORA matches or exceeds the strongest baseline across all six benchmarks under both standard and strict criteria (Figures 1 and 4). Under the standard criterion, relative gains reach 54.5% on CSP7 and, under the strict criterion, 188.0% on CSP6. Figure 2 visualizes predictions across diverse molecules and packing motifs. Full results, empirical-Z-prior evaluation, aligned structure visualizations, and space-group conditioning results are in Appendices H, I, J and K, respectively.

4.3 Structure Ranking

Benchmarks. We evaluate two FastCSP-derived benchmarks (Gharakhanyan et al., 2025): a single-polymorph set of 23 semi-rigid and three flexible systems, each with one experimental form, and a multi-polymorph set of five semi-rigid and three flexible systems, comprising 29 forms. We exclude multi-polymorph systems whose CSD families overlap the CLARI training split; excluded refcodes are listed in Appendix L.1.

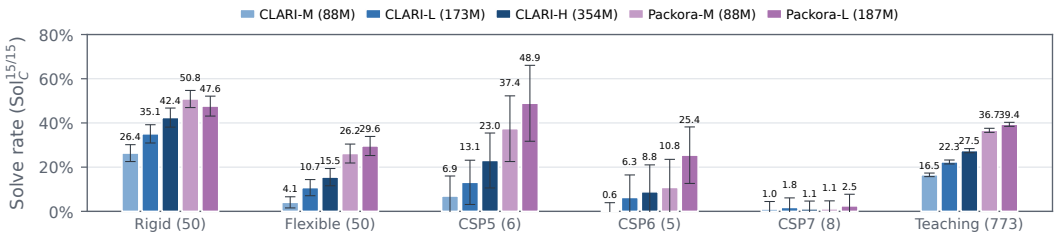
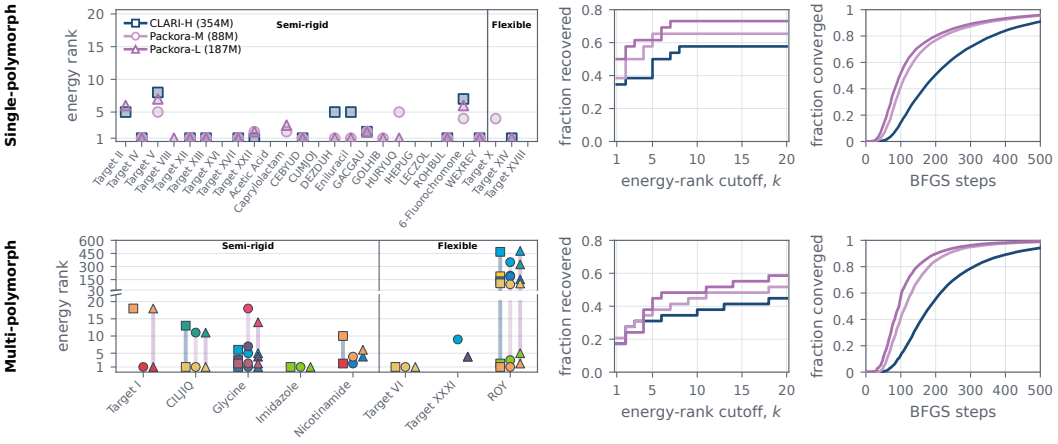


Figure 4: **Strict crystal solve rate with 30 candidates per target.** We report strict crystal solve rate, requiring all 15 molecules to match under COMPACK, with parentheses indicating benchmark sizes. Error bars show standard deviations over 5,000 bootstrap resamples of 30 from 1,000 candidates per target. PACKORA achieves the highest solve rate across all six benchmarks.



(a) Experimental-form rank (b) Recovery (c) Convergence

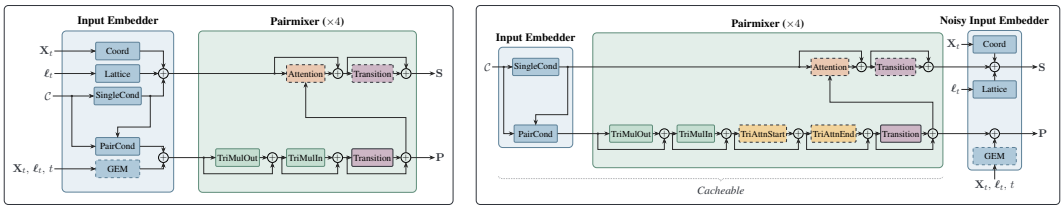
Figure 5: **Relaxation and ranking on FastCSP benchmarks.** *Top*: 26 single-polymorph targets. *Bottom*: 29 polymorphs across eight systems, with candidates pooled across CSD entries within each system. We relax 1,000 candidates per CSD entry with UMA-S-1.2 (Wood et al., 2025) and rank the retained structures by energy. Columns show **(a)** best matching energy rank (lower is better), **(b)** recovery by rank cutoff k, and **(c)** cumulative convergence by BFGS step. Marker shapes identify models; colors in the bottom-left panel distinguish polymorphs. PACKORA improves recovery and ranking while accelerating relaxation relative to CLARI-H.

Relaxation and ranking workflow. Each model generates 1,000 candidates per CSD entry. Following FastCSP, we relax candidates using UMA-S-1.2 (OMC task) (Wood et al., 2025; Gharakhanyan et al., 2026) with ASE BFGS (Larsen et al., 2017), discarding unconverged candidates and those that violate connectivity, formula-unit count, or density constraints. For multi-polymorph systems, we pool candidates across entries before energy filtering and deduplication, then rank the remaining candidates by lattice energy per formula unit. Appendix L.1 provides the filtering thresholds and definition of lattice energy.

Metrics. A candidate matches an experimental form if COMPACK matches 30 molecules with $\mathrm{RMSD}_{30} < 1$ Å (Chisholm & Motherwell, 2005). We report overall recovery, Top-k recovery for $k \in \{1, 5, 20\}$, and the mean best energy rank among recovered forms (see Appendix L).

Baselines. We compare PACKORA-M and PACKORA-L with CLARI-H. We exclude FastCSP because its random structure generator requires a substantially larger candidate budget, making relaxation prohibitively expensive within our computational budget (see Appendix L.1 for budget details).

Single-polymorph results. PACKORA improves both recovery and ranking quality. PACKORA-M and PACKORA-L recover 17 and 19 of 26 targets, respectively, compared with 16 for CLARI-H, while reducing the mean rank of recovered targets from 4.25 to 2.00 and 2.05 (Figures 5a and 5b, top). Recovery is also higher at every evaluated rank cutoff. Beyond ranking quality, PACKORA relaxes more efficiently: PACKORA-L reaches 50% convergence in 98 BFGS steps versus 198 for CLARI-H.



(a) Pre-entry: noise before Pairmixer

(b) Post-entry: noise after Pairmixer

Figure 6: **Model variants for architecture ablations. (a)** Pre-entry: noisy coordinate and lattice embeddings, together with optional geometric enhancement module (GEM) features, are injected before PAIRMIXER. **(b)** Post-entry: the condition-only INPUT EMBEDDER and PAIRMIXER are applied before noisy-state injection. Dotted outlines denote optional modules. Both variants pass the resulting single and pair representations, $\mathbf{S}$ and $\mathbf{P}$, to the same DiT trunk. The post-entry design allows condition-only computations to be cached across denoising steps.

Multi-polymorph results. PACKORA improves recovery across polymorphs while substantially accelerating relaxation. PACKORA-M and PACKORA-L recover 19 and 21 of 29 forms, respectively, compared with 18 for CLARI-H, while reducing the mean rank of recovered forms from 102.17 to 47.11 and 54.62 (Figures 5a and 5b, bottom). PACKORA recovers at least one form for all eight systems, compared with seven for CLARI-H. At 100 BFGS steps, their convergence rates reach 41.4% and 57.3%, versus 13.0% for CLARI-H. Complete results and generation and relaxation costs are provided in Appendices L and F.

5 SYSTEMATIC STUDIES FOR MOLECULAR CSP

We develop the PACKORA recipe through controlled studies spanning five areas: *Architecture:* backbone choice (Pairformer, Pairmixer, and DiT with pair bias), noisy-state entry, and optional modules (Figures 6 and 7); *Training:* time distribution, augmentation, loss, and optimization (Figure 8); *Conditioning:* template dropout for robustness to optional inputs (Figure 9a); *Inference:* sampler, solver settings, template augmentation, and autoguidance (Figure 9b); and *Scaling:* single and pair widths and their effects on per-sample success and target coverage (Figure 9c; Appendix M).

5.1 ABLATION PROTOCOL

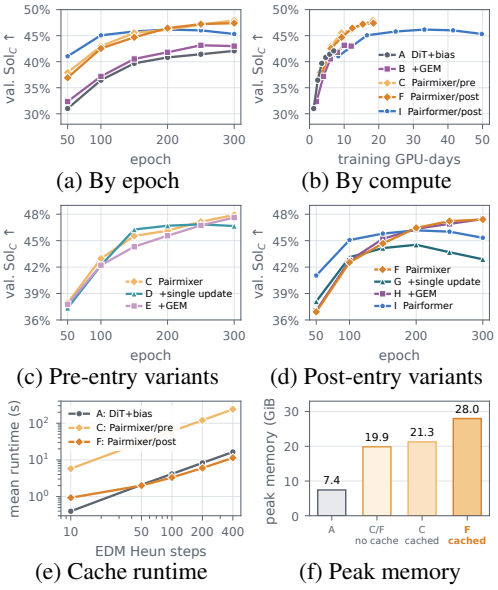
Dataset preprocessing. Following OXtal (Jin et al., 2025), we construct the ablation dataset from CSD entries deposited by May 1, 2025, excluding test CSD families and entries sharing molecular components with the test sets. Preprocessing details and dataset statistics are in Appendix N.

Setup. We train each configuration for 300 epochs on crystals with at most 300 atoms, evaluating every 50 epochs on 8,192 fixed validation crystals. We measure Sol_C using 30 candidates per target and 200-step EDM–Heun, except in inference studies that vary the sampler and evaluation budget. We vary one design choice at a time, generally selecting greedily by validation performance unless practical considerations favor an alternative.

5.2 ARCHITECTURE

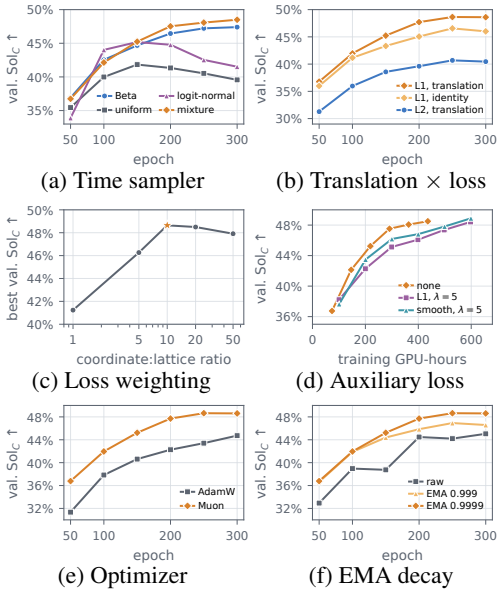
Explored designs. We compare three backbones: DiT with pair-bias attention (Lo et al., 2026), Pairformer (Abramson et al., 2024; Jin et al., 2025), and Pairmixer (Ouyang-Zhang et al., 2025). We also test optional modules, including a single-track update and the geometric enhancement module (GEM) (Veljković et al., 2026), which encodes periodic minimum-image pair geometry from the noisy structure. Finally, we compare injecting coordinate and lattice noise either before pair refinement (pre-entry; Geffner et al., 2025) or after it (post-entry; Abramson et al., 2024); the latter enables caching of condition-only pair computations.

Post-entry Pairmixer best balances accuracy and cost. Pairmixer outperforms Pairformer and DiT with pair-bias attention (0.474 vs. 0.462 vs. 0.421 with post-entry;Figures 7a to 7b). Pairformer performs better early in training but later degrades and is substantially more expensive than Pairmixer. GEM improves DiT with pair-bias attention, whereas neither GEM nor the single-track update



(a) By epoch (b) By compute (c) Pre-entry variants (d) Post-entry variants (e) Cache runtime (f) Peak memory

Figure 7: **Backbone selection, extensions, and caching.** Validation Sol_C with 30 candidates per target and 200-step EDM–Heun sampling on the $\leq$ 300-atom subset. **(a–b)** Backbones across epochs and compute (GPU days). **(c–d)** Pre- and post-entry variants with optional modules. **(e–f)** Sampling runtime and peak memory for batches of 64 300-atom structures, with caching for post-entry Pairmixer. Post-entry Pairmixer combines strong performance with moderate sampling cost.



(a) Time sampler (b) Translation × loss (c) Loss weighting (d) Auxiliary loss (e) Optimizer (f) EMA decay

Figure 8: **Training recipe selection.** Validation Sol_C with 30 candidates per target and 200-step EDM–Heun sampling. **(a)** Time sampling distribution. **(b)** Translation augmentation and L^1/L^2 loss. **(c)** Coordinate-to-lattice loss weighting. **(d)** Auxiliary pair loss. **(e)** Optimizer. **(f)** EMA decay. Beta-uniform time sampling, translation augmentation, L^1 loss with 10:1 weighting, Muon, and EMA decay 0.9999 perform best; auxiliary pair loss offers little benefit.

improves Pairmixer (Figures 7c to 7d). Pre-entry Pairmixer slightly outperforms post-entry Pairmixer (0.479 vs. 0.474), but caching the post-entry design yields a 20.1× reduction in 200-step sampling time in a synthetic $[64, 300]$ benchmark, with peak memory increasing from 21.3 to 28.0 GiB (Figures 7e to 7f). We therefore select post-entry Pairmixer without additional modules. Appendix M reports parameter counts and GFLOPs for each variant.

5.3 Training

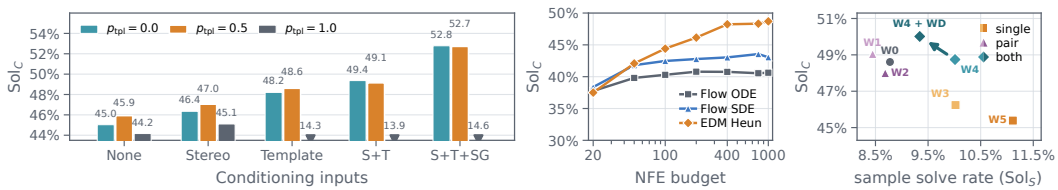
Explored designs. We compare uniform, logit-normal, Beta, and Beta-uniform time sampling (Geffner et al., 2025); random translation augmentation (Gruver et al., 2024); and L^1/L^2 loss under different coordinate-to-lattice loss weights. We also test auxiliary losses on periodic pair distances using L^1 (Lo et al., 2026) or smooth-LDDT (Abramson et al., 2024), both with coefficient $\lambda = 5$ (see Appendix M.2 for definitions). Finally, we compare AdamW (Loshchilov & Hutter, 2019) with Muon (Liu et al., 2025), and raw with EMA parameters.

Endpoint loss and time sampling matter most. Beta-uniform time sampling performs best, and translation augmentation improves Sol_C from 0.466 to 0.486. L^1 endpoint regression outperforms L^2 (0.486 vs. 0.407), with a 10:1 coordinate-to-lattice loss-weight ratio performing best. We omit the auxiliary pair loss because the best smooth-LDDT setting improves only by 0.004 at 1.37× the training GPU-hours. Muon outperforms AdamW, and EMA decay 0.9999 performs best (Figure 8).

5.4 Conditioning

Explored designs. We vary template-coordinate dropout $p_{\mathrm{tpl}} \in \{0, 0.5, 1\}$ across five inference settings: no optional inputs, stereochemistry, templates, both, and both plus space group.

Partial template dropout performs best overall. We use $p_{\mathrm{tpl}} = 0.5$, which gives the highest mean Sol_C across settings (0.487, versus 0.483 and 0.264 for $p_{\mathrm{tpl}} = 0$ and 1). It also performs best without



(a) Conditioning (b) Inference (c) Scaling

Figure 9: **Conditioning, inference, and scaling.** Validation performance with 30 candidates per target. **(a)** Sol_C under different template-dropout rates across conditioning inputs (S: stereochemistry; T: template coordinates; SG: space group); triangles indicate values below the plotted range. **(b)** Sol_C for different samplers at matched function-evaluation budgets (NFE). **(c)** Per-sample success Sol_S versus target coverage Sol_C for scaling variants; W0 and W4 + WD (10^{-2}) correspond to PACKORA-M and PACKORA-L. Template dropout $p_{\mathrm{tpl}} = 0.5$, EDM–Heun, and balanced width scaling with weight decay perform best for conditioning, inference, and scaling, respectively.

optional inputs (0.459 versus 0.450 and 0.442 for $p_{\mathrm{tpl}} = 0$ and 1), while staying within 0.003 of $p_{\mathrm{tpl}} = 0$ when templates are provided (Figure 9a).

5.5 INFERENCE

Explored designs. We compare Euler, Euler–Maruyama, and EDM–Heun (Karras et al., 2022), while varying the EDM time-schedule parameter ρ. We also test template augmentation at inference (rotation and translation, rotation only, or none) and autoguidance (Karras et al., 2024), using earlier checkpoints as degraded models (Geffner et al., 2025).

Solver choice matters most. At 400 NFE, EDM–Heun achieves $\mathrm{Sol}_C = 0.482$, versus 0.430 for Euler–Maruyama and 0.408 for Euler. Increasing to 1000 NFE adds only 0.005, so we use 400 NFE (Figure 9b). Performance is similar for $\rho \in [5, 10]$; we use $\rho = 7$ following Karras et al. (2022). Neither reduced template augmentation nor autoguidance improves performance, so we retain rotation and translation without autoguidance (Appendix M.3).

5.6 SCALING

Explored designs. We vary single-track width d_s and pair-track width d_p, holding other settings fixed. Starting from W0 with $(d_s, d_p) = (512, 128)$, we increase d_p to 192 or 256 (W1, W2), d_s to 768 or 1024 (W3, W5), or both to $(768, 192)$ (W4). We evaluate scaling using per-sample success Sol_S and 30-candidate target coverage Sol_C. Full configurations are provided in Appendix M.4.

Scaling either width alone is insufficient. Increasing only the single width improves Sol_S but eventually reduces Sol_C, suggesting more repeatable success on already-solvable targets rather than broader coverage. Increasing only the pair width yields no consistent improvement in either metric. Neither consistently improves the trade-off across sampling budgets (Figure 9c; Appendix M.4).

Balanced scaling offers a favorable success–coverage trade-off. Increasing both single and pair widths (W4) recovers target coverage while preserving the single-sample success. Adding weight decay of 10^{-2} further improves coverage at k=30, at some cost to success at k=1. We prioritize coverage because downstream ranking cannot recover structures absent from the candidate set. We therefore use W0 as PACKORA-M and W4 with 10^{-2} weight decay as PACKORA-L (Figure 9c).

6 CONCLUSION

We present PACKORA, a flexible all-atom generative model for molecular CSP that supports multi-component and organometallic crystals with optional conditions. Controlled studies of architecture, training, conditioning, inference, and scaling yield a design recipe that combines strong generation performance with efficient sampling. Evaluated separately on structure generation and ranking, PACKORA achieves stronger coverage, higher experimental-form recovery, better ranks, and faster relaxation than the baselines across six generation and two ranking benchmarks.

AI USE STATEMENT

We used OpenAI's ChatGPT and Codex to assist with implementation throughout the molecular crystal structure prediction pipeline, including data processing, model implementation, training, and evaluation, and to assist with interpreting experimental results. We also used these tools to write plotting code, prepare figures, and draft and revise manuscript text. All activities followed specific instructions reflecting the authors' ideas and intentions. We did not use generative AI tools to develop the method, design experiments, formulate mathematical claims, or construct proofs. All AI-assisted work was thoroughly reviewed by the authors: generated code was manually checked, and figures, manuscript text, and suggested interpretations were carefully examined. The authors take full responsibility for the final content of this work.

ETHICS STATEMENT

This work could be misused to support the design of harmful compounds. We encourage responsible use and independent validation of predictions. The Cambridge Structural Database (CSD) is proprietary; access to the underlying data remains subject to CCDC licensing terms.

REPRODUCIBILITY STATEMENT

We provide the source code for dataset preprocessing, training, inference, and evaluation, together with dataset splits specified by CSD refcodes, in the anonymous repository at https://anonymous.4open.science/r/packora-7878/. Architecture pseudocode is provided in Appendix C; dataset preprocessing and split details in Appendices D and N; training and sampling hyperparameters in Appendix E; and evaluation protocols in Appendices G and L.1. These materials support reproduction of both the benchmark experiments and ablation studies. Reproducing the entire workflow requires licensed access to the CSD Python API.

REFERENCES

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(8016):493–500, 2024. doi: 10.1038/s41586-024-07487-w.

David A. Bardwell, Claire S. Adjiman, Yelena A. Arnautova, Ekaterina Bartashevich, Stephan X. M. Boerrigter, Doris E. Braun, Aurora J. Cruz-Cabeza, Graeme M. Day, Raffaele G. Della Valle, Gautam R. Desiraju, et al. Towards crystal structure prediction of complex organic compounds: a report on the fifth blind test. *Acta Crystallographica Section B: Structural Science, Crystal Engineering and Materials*, 67(6):535–551, 2011. doi: 10.1107/S0108768111042868.

Gary M. Battle, Gregory M. Ferrence, and Frank H. Allen. Applications of the Cambridge Structural Database in chemical education. *Journal of Applied Crystallography*, 43(5):1208–1223, 2010. doi: 10.1107/S0021889810024155.

Gregory J. O. Beran. Frontiers of molecular crystal structure prediction for pharmaceuticals and functional organic materials. *Chemical Science*, 14(46):13290–13312, 2023. doi: 10.1039/D3SC03903J.

James Alexander Chisholm and Sam Motherwell. COMPACK: a program for identifying crystal structure similarity using distances. *Journal of Applied Crystallography*, 38(1):228–231, 2005. doi: 10.1107/S0021889804027074.

Farren Curtis, Xiayue Li, Timothy Rose, Álvaro Vázquez-Mayagoitia, Saswata Bhattacharya, Luca M. Ghiringhelli, and Noa Marom. GAtor: A first-principles genetic algorithm for molecular crystal structure prediction. *Journal of Chemical Theory and Computation*, 14(4):2246–2264, 2018. doi: 10.1021/acs.jctc.7b01152.

Graeme M. Day, Timothy G. Cooper, Aurora J. Cruz-Cabeza, Katie E. Hejczyk, Herman L. Ammon, Stephan X. M. Boerrigter, Jonathan S. Tan, Raffaele G. Della Valle, Elisabetta Venuti, K. V. J. Jose, et al. Significant progress in predicting the crystal structures of small organic molecules—a report on the fourth blind test. *Acta Crystallographica Section B: Structural Science*, 65(2):107–125, 2009. doi: 10.1107/S0108768109004066.

Floor Eijkelboom, Grigory Bartosh, Christian Andersson Naesseth, Max Welling, and Jan-Willem van de Meent. Variational flow matching for graph generation. *Advances in Neural Information Processing Systems*, 37:11735–11764, 2024.

Tomas Geffner, Kieran Didi, Zuobai Zhang, Danny Reidenbach, Zhonglin Cao, Jason Yim, Mario Geiger, Christian Dallago, Emine Kucukbenli, Arash Vahdat, and Karsten Kreis. Proteina: Scaling flow-based protein structure generative models. In *International Conference on Learning Representations (ICLR)*, 2025.

Vahe Gharakhanyan, Yi Yang, Luis Barroso-Luque, Muhammed Shuaibi, Daniel S. Levine, Kyle Michel, et al. FastCSP: Accelerated molecular crystal structure prediction with universal model for atoms. *arXiv preprint arXiv:2508.02641*, 2025.

Vahe Gharakhanyan, Luis Barroso-Luque, Yi Yang, Muhammed Shuaibi, Kyle Michel, Daniel S Levine, Misko Dzamba, Xiang Fu, Meng Gao, Xingyu Liu, et al. Open molecular crystals 2025 (omc25) dataset and models. *Scientific Data*, 13(1):354, 2026.

Colin R. Groom, Ian J. Bruno, Matthew P. Lightfoot, and Suzanna C. Ward. The Cambridge Structural Database. *Acta Crystallographica Section B: Structural Science, Crystal Engineering and Materials*, 72(2):171–179, 2016. doi: 10.1107/S2052520616003954.

Ralf W Grosse-Kunstleve, Nicholas K Sauter, and Paul D Adams. Numerically stable algorithms for the computation of reduced unit cells. *Acta Crystallographica Section A: Foundations of Crystallography*, 60(1):1–6, 2004.

Nate Gruver, Anuroop Sriram, Andrea Madotto, Andrew Gordon Wilson, C. Lawrence Zitnick, and Zachary Ulissi. Fine-tuned language models generate stable inorganic materials as text. In *International Conference on Learning Representations*, 2024.

Philipp Höllmer, Thomas Egg, Maya Martirossyan, Eric Fuemmeler, Zeren Shui, Amit Gupta, Pawan Prakash, Adrian Roitberg, Mingjie Liu, George Karypis, et al. Open materials generation with stochastic interpolants. In *Forty-second International Conference on Machine Learning*, 2025.

Lily M. Hunnisett, Nicholas Francia, Jonas Nyman, Nathan S. Abraham, Srinivasulu Aitipamula, Tamador Alkhidir, Mubarak Almehairbi, Andrea Anelli, Dylan M. Anstine, John E. Anthony, et al. The seventh blind test of crystal structure prediction: structure ranking methods. *Acta Crystallographica Section B: Structural Science, Crystal Engineering and Materials*, 80(6):548–574, 2024a. doi: 10.1107/S2052520624008679.

Lily M. Hunnisett, Jonas Nyman, Nicholas Francia, Nathan S. Abraham, Claire S. Adjiman, Srinivasulu Aitipamula, Tamador Alkhidir, Mubarak Almehairbi, Andrea Anelli, Dylan M. Anstine, et al. The seventh blind test of crystal structure prediction: structure generation methods. *Acta Crystallographica Section B: Structural Science, Crystal Engineering and Materials*, 80(6):517–547, 2024b. doi: 10.1107/S2052520624007492.

Rui Jiao, Wenbing Huang, Peijia Lin, Jiaqi Han, Pin Chen, Yutong Lu, and Yang Liu. Crystal structure prediction by joint equivariant diffusion. *Advances in Neural Information Processing Systems*, 36, 2024.

Rui Jiao, Hanlin Wu, Wenbing Huang, Yuxuan Song, Yawen Ouyang, Yu Rong, Tingyang Xu, Pengju Wang, Hao Zhou, Wei-Ying Ma, et al. MOF-BFN: Metal-organic frameworks structure prediction via bayesian flow networks. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Emily Jin, Andrei Cristian Nica, Mikhail Galkin, Jarrid Rector-Brooks, Kin Long Kelvin Lee, Santiago Miret, Frances H. Arnold, Michael Bronstein, Avishek Joey Bose, Alexander Tong, and Cheng-Hao Liu. OXtal: An all-atom diffusion model for organic crystal structure prediction. *arXiv preprint arXiv:2512.06987*, 2025.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 26565–26577, 2022.

Tero Karras, Miika Aittala, Tuomas Kynkaanniemi, Jaakko Lehtinen, Timo Aila, and Samuli Laine. Guiding a diffusion model with a bad version of itself. *arXiv preprint arXiv:2406.02507*, 2024.

Nayoung Kim, Seongsu Kim, and Sungsoo Ahn. Flexible MOF generation with torsion-aware flow matching. In *Advances in Neural Information Processing Systems*, 2025a. URL https://openreview.net/forum?id=cLJfumTWLI.

Nayoung Kim, Seongsu Kim, Minsu Kim, Jinkyoo Park, and Sungsoo Ahn. MOFFlow: Flow matching for structure prediction of metal-organic frameworks. In *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=dNT3abOsLo.

Nayoung Kim, Honghui Kim, Sihyun Yu, Minkyu Kim, Seongsu Kim, and Sungsoo Ahn. AtomMOF: All-atom flow matching for MOF-adsorbate structure prediction. *arXiv preprint arXiv:2602.07351*, 2026. URL https://arxiv.org/abs/2602.07351.

I. Křivý and B. Gruber. A unified algorithm for determining the reduced (Niggli) cell. *Acta Crystallographica Section A*, 32(2):297–298, 1976. doi: 10.1107/S0567739476000636.

Greg Landrum et al. Rdkit: Open-source cheminformatics, 2006.

Ask Hjorth Larsen, Jens Jørgen Mortensen, Jakob Blomqvist, Ivano E Castelli, Rune Christensen, Marcin Dułak, Jesper Friis, Michael N Groves, Bjørk Hammer, Cory Hargus, et al. The atomic simulation environment: a python library for working with atoms. *Journal of Physics: Condensed Matter*, 29(27):273002, 2017.

Xiayue Li, Farren S. Curtis, Timothy Rose, Christoph Schober, Álvaro Vázquez-Mayagoitia, Karsten Reuter, Harald Oberhofer, and Noa Marom. Genarris: Random generation of molecular crystal structures and fast screening with a Harris approximation. *The Journal of Chemical Physics*, 148 (24):241701, 2018. doi: 10.1063/1.5014038.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Jingyuan Liu, Jianlin Su, Xingcheng Yao, Zhejun Jiang, Guokun Lai, Yulun Du, Yidao Qin, Weixin Xu, Enzhe Lu, Junjie Yan, Yanru Chen, Huabin Zheng, Yibo Liu, Shaowei Liu, Bohong Yin, Weiran He, Han Zhu, Yuzhi Wang, Jianzhou Wang, Mengnan Dong, Zheng Zhang, Yongsheng Kang, Hao Zhang, Xinran Xu, Yutao Zhang, Yuxin Wu, Xinyu Zhou, and Zhilin Yang. Muon is scalable for llm training, 2025.

Alston Lo, Luka Mucko, Austin H. Cheng, Andy Cai, Alastair J. A. Price, Wojciech Matusik, and Alán Aspuru-Guzik. Fast organic crystal structure prediction with unit cell flow matching. *arXiv preprint arXiv:2606.03199*, 2026.

J. P. M. Lommerse, W. D. S. Motherwell, H. L. Ammon, J. D. Dunitz, A. Gavezzotti, D. W. M. Hofmann, F. J. J. Leusen, W. T. M. Mooij, S. L. Price, B. Schweizer, et al. A test of crystal structure prediction of small organic molecules. *Acta Crystallographica Section B: Structural Science*, 56 (4):697–714, 2000. doi: 10.1107/S0108768100004584.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Xiaoshan Luo, Zhenyu Wang, Qingchang Wang, Xuechen Shao, Jian Lv, Lei Wang, Yanchao Wang, and Yanming Ma. CrystalFlow: A flow-based generative model for crystalline materials. *Nature Communications*, 16:9267, 2025. doi: 10.1038/s41467-025-64364-4.

Benjamin Kurt Miller, Ricky TQ Chen, Anuroop Sriram, and Brandon M Wood. Flowmm: Generating materials with riemannian flow matching. In *Forty-first International Conference on Machine Learning*, 2024.

Shyue Ping Ong, William Davidson Richards, Anubhav Jain, Geoffroy Hautier, Michael Kocher, Shreyas Cholia, Dan Gunter, Vincent Chevrier, Kristin A. Persson, and Gerbrand Ceder. Python materials genomics (pymatgen): A robust, open-source python library for materials analysis. *Computational Materials Science*, 68:314–319, 2013. doi: 10.1016/j.commatsci.2012.10.028.

Jeffrey Ouyang-Zhang, Pranav Murugan, Daniel J. Diaz, Gianluca Scarpellini, Richard Strong Bowen, Nate Gruver, Adam Klivans, Philipp Krähenbühl, Aleksandra Faust, and Maruan Al-Shedivat. Triangle multiplication is all you need for biomolecular structure representations. *arXiv preprint arXiv:2510.18870*, 2025.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4195–4205, 2023.

Sarah L. Price. Predicting crystal structures of organic compounds. *Chemical Society Reviews*, 43: 2098–2111, 2014. doi: 10.1039/C3CS60279F.

Nicholas Rego and David Koes. 3Dmol.js: molecular visualization with WebGL. *Bioinformatics*, 31 (8):1322–1324, 2015. doi: 10.1093/bioinformatics/btu829.

Anthony M. Reilly, Richard I. Cooper, Claire S. Adjiman, Saswata Bhattacharya, A. Daniel Boese, Jan Gerit Brandenburg, Peter J. Bygrave, Rita Bylsma, Josh E. Campbell, Roberto Car, et al. Report on the sixth blind test of organic crystal structure prediction methods. *Acta Crystallographica Section B: Structural Science, Crystal Engineering and Materials*, 72(4):439–459, 2016. doi: 10.1107/S2052520616007447.

Akshay Subramanian, Elton Pan, Juno Nam, Maurice Weiler, Shuhui Qu, Cheol Woo Park, Tommi S. Jaakkola, Elsa Olivetti, and Rafael Gomez-Bombarelli. PackFlow: Generative molecular crystal structure prediction via reinforcement learning alignment. *arXiv preprint arXiv:2602.20140*, 2026.

Richard A. Sykes, Natalie T. Johnson, Christopher J. Kingsbury, Jürgen Harter, Andrew G. P. Maloney, Isaac J. Sugden, Suzanna C. Ward, Ian J. Bruno, Stewart A. Adcock, Peter A. Wood, Patrick McCabe, Alexandru A. Moldovan, Francis Atkinson, Ilenia Giangreco, and Jason C. Cole. What has scripting ever done for us? The CSD Python application programming interface (API). *Journal of Applied Crystallography*, 57(4):1235–1250, 2024. doi: 10.1107/S1600576724005934.

Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Tin Hadži Veljković, Joshua Rosenthal, Ivor Lončarić, and Jan-Willem van de Meent. Crystalite: A lightweight transformer for efficient crystal modeling. *arXiv preprint arXiv:2604.02270*, 2026.

Shuzhe Wang, Jagna Witek, Gregory A Landrum, and Sereina Riniker. Improving conformer generation for small rings and macrocycles based on distance geometry and experimental torsional-angle preferences. *Journal of chemical information and modeling*, 60(4):2044–2058, 2020.

Brandon M Wood, Misko Dzamba, Xiang Fu, Meng Gao, Muhammed Shuaibi, Luis Barroso-Luque, Kareem Abdelmaqsoud, Vahe Gharakhanyan, John R Kitchin, Daniel S Levine, et al. Uma: A family of universal models for atoms. In *Advances in Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=SvopaNxYWt.

Hanlin Wu, Yuxuan Song, Jingjing Gong, Ziyao Cao, Yawen Ouyang, Jianbing Zhang, Hao Zhou, Wei-Ying Ma, and Jingjing Liu. A periodic bayesian flow for material generation. In *The Thirteenth International Conference on Learning Representations*, 2025.

Tian Xie, Xiang Fu, Octavian-Eugen Ganea, Regina Barzilay, and Tommi S. Jaakkola. Crystal diffusion variational autoencoder for periodic material generation. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=03RLpj-tc_.

Olga Zaghen, Floor Eijkelboom, Alison Pouplin, Cong Liu, Max Welling, Jan-Willem van de Meent, and Erik J Bekkers. Riemannian variational flow matching for material and protein design. *arXiv preprint arXiv:2502.12981*, 2025.

Cheng Zeng, Harry W. Sullivan, Thomas Egg, Maya M. Martirossyan, Philipp Höllmer, Jirui Jin, Richard G. Hennig, Adrian Roitberg, Stefano Martiniani, Ellad B. Tadmor, and Mingjie Liu. MolCrystalFlow: Molecular crystal structure prediction via flow matching. *arXiv preprint arXiv:2602.16020*, 2026.

Yi Zhou, Chan Lu, Yiming Ma, Wei Qu, Fei Ye, Kexin Zhang, Lan Wang, Minrui Gui, and Quanquan Gu. Seedfold: Scaling biomolecular structure prediction. *arXiv preprint arXiv:2512.24354*, 2025.

APPENDIX

A Related work **16**

B Crystal representation **16**

C Model architecture **17**

D Dataset statistics **20**

E Training and sampling settings **21**

F Computational requirements **23**
F.1 Training 23
F.2 Generation and relaxation 24

G Structure generation evaluation **25**

H Structure generation results **26**

I Empirical Z sampling **28**

J Structure overlays **30**

K Space-group conditioning **32**

L Structure ranking results **33**
L.1 Evaluation protocol 33
L.2 Recovery and ranking results 33

M Ablation results **36**
M.1 Architecture 36
M.2 Training 36
M.3 Inference 36
M.4 Scaling 37

N Ablation dataset preprocessing **38**

A Related work

Generative CSP. A related line of work studies crystal generation and structure prediction for inorganic materials. CDVAE combines a variational autoencoder with diffusion-based coordinate generation, whereas DiffCSP jointly diffuses fractional coordinates and lattice conditioned on a given composition (Xie et al., 2022; Jiao et al., 2024). Subsequent approaches formulate periodic crystal generation using flow matching, as in FlowMM and CrystalFlow (Miller et al., 2024; Luo et al., 2025); stochastic interpolants, as in OMatG (Höllmer et al., 2025); or Bayesian flows over periodic variables (Wu et al., 2025).

Generative CSP has also been extended to metal-organic frameworks (MOFs), where metal nodes and organic linkers are treated as modular building blocks. MOFFlow (Kim et al., 2025b) and MOF-BFN (Jiao et al., 2025) generate lattices and rototranslations of rigid building blocks using Riemannian and Bayesian flows, respectively, while MOFFlow-2 (Kim et al., 2025a) additionally generates building blocks and models torsions. AtomMOF (Kim et al., 2026) removes the rigid-body assumption by directly generating MOF and MOF–adsorbate structures at all-atom resolution.

B Crystal representation

Lattice canonicalization. Since the same lattice can be represented by different basis vectors and global rotations, we canonicalize the cell in two steps. First, we Niggli-reduce (Křivý & Gruber, 1976; Grosse-Kunstleve et al., 2004) each crystal to obtain a reduced basis $\widetilde{\mathbf{L}}$, reducing the ambiguity among equivalent lattice bases. Second, we remove the remaining global rotational freedom by forming the rotation-invariant Gram matrix $\widetilde{\mathbf{L}}\widetilde{\mathbf{L}}^\top$ and taking its unique lower-triangular Cholesky factor with positive diagonal:

$$\mathbf{L} = \mathrm{chol}\left(\widetilde{\mathbf{L}}\widetilde{\mathbf{L}}^\top\right) = \begin{pmatrix} e^{\ell_1} & 0 & 0 \\ \ell_2 & e^{\ell_3} & 0 \\ \ell_4 & \ell_5 & e^{\ell_6} \end{pmatrix}.$$

We further parameterize the canonical cell by an unconstrained vector $\boldsymbol{\ell} = (\ell_1, \ldots, \ell_6) \in \mathbb{R}^6$ following Veljković et al. (2026).

Chemical specification. For K molecular components, r_k denotes the number of copies of molecular graph G_k in one formula unit. For example, a binary 2:1 cocrystal has $\mathbf{r} = (2, 1)$. If component G_k contains N_k atoms, a cell with Z formula units contains $N = Z \sum_{k=1}^{K} r_k N_k$ atoms.

C MODEL ARCHITECTURE

Notation. This section specifies the PACKORA forward pass in algorithmic form. Indices i, j, k denote atoms, and h denotes an attention head. We use $\mathbf{S}$ and $\mathbf{P}$ for single and pair representations, respectively. Symbols E, W, and MLP denote learned embedding functions, linear maps, and multilayer perceptrons, respectively, while Norm denotes normalization. Split and Concat partition and concatenate feature channels, respectively. We use σ for the sigmoid function, ϕ for the configured activation function, $\odot$ for elementwise multiplication, and d_h for the attention-head dimension.

Overall forward pass. Algorithm 1 summarizes the complete forward pass: the model embeds the condition, refines pair features, injects the noisy crystal state, applies the DiT trunk, and predicts coordinate and lattice.

Algorithm 1 Model forward pass

Require: Noisy state $(\mathbf{X}_t, \boldsymbol{\ell}_t)$, time t, condition $\mathcal{C}$, and valid-atom mask $\mathbf{m}$
Ensure: Posterior mean $(\widehat{\mathbf{X}}_1, \widehat{\boldsymbol{\ell}}_1)$

```
# 1. Encode the condition and refine pair features
```
$(\mathbf{S}_{\mathcal{C}}, \mathbf{P}) \leftarrow \textsc{ConditionEmbed}(\mathcal{C}, \mathbf{m})$
for each Pairmixer block **do**
  $\mathbf{P} \leftarrow \textsc{PairmixerBlock}(\mathbf{P}, \mathbf{m})$
end for
$\mathbf{P}^{\star} \leftarrow \mathbf{P}$

```
# 2. Inject the noisy state
```
$(\mathbf{S}, \mathbf{c}_t) \leftarrow \textsc{NoisyStateEmbed}(\mathbf{S}_{\mathcal{C}}, \mathbf{X}_t, \boldsymbol{\ell}_t, t)$

```
# 3. Propagate and decode
```
for each DiT block **do**
  $\mathbf{S} \leftarrow \textsc{DiTBlock}(\mathbf{S}, \mathbf{c}_t, \mathbf{P}^{\star}, \mathbf{m})$
end for
$(\widehat{\mathbf{X}}_1, \widehat{\boldsymbol{\ell}}_1) \leftarrow \textsc{PredictionHeads}(\mathbf{S}, \mathbf{m})$
return $(\widehat{\mathbf{X}}_1, \widehat{\boldsymbol{\ell}}_1)$

Condition embedding. For Algorithm 2, $\mathbf{a}_i$ contains the atomic number, periodic-table descriptors, formal charge, and chirality of atom i. Template coordinates are denoted by $\widetilde{\mathbf{x}}_i$, with availability indicator m_{T}. Molecular membership, bond type, and bond stereochemistry are denoted by u_i, τ_{ij}, and ζ_{ij}, respectively; unavailable categorical conditions use learned null categories. The global condition contains the space-group embedding. Bond, stereochemistry, and template features are restricted to atom pairs within the same molecular component, and stereochemistry features are zero for nonbonded pairs.

Algorithm 2 Condition embedding

Require: Condition $\mathcal{C}$ and valid-atom mask $\mathbf{m}$
Ensure: Condition-only single representation $\mathbf{S}_{\mathcal{C}}$ and ordered pair representation $\mathbf{P}$

```
# 1. Embed atom-level and global conditions
```
$\mathbf{S}_{\mathcal{C},i} \leftarrow E_{\mathrm{atom}}(\mathbf{a}_i) + m_{\mathrm{T}} E_{\mathrm{temp}}(\widetilde{\mathbf{x}}_i) + m_i E_{\mathrm{global}}(\mathcal{C})$

```
# 2. Construct intramolecular pair features
```
$M_{ij}^{\mathrm{mol}} \leftarrow m_i m_j \mathbf{1}[u_i = u_j]$
$\boldsymbol{\delta}_{ij} \leftarrow \widetilde{\mathbf{x}}_i - \widetilde{\mathbf{x}}_j$, $\rho_{ij} \leftarrow (1 + \|\boldsymbol{\delta}_{ij}\|_2^2)^{-1}$
$\overline{\mathbf{P}}_{ij} \leftarrow W_Q \mathbf{S}_{\mathcal{C},i} + W_K \mathbf{S}_{\mathcal{C},j} + M_{ij}^{\mathrm{mol}} \big[E_{\mathrm{bond}}(\tau_{ij}) + E_{\mathrm{stereo}}(\zeta_{ij}) + m_{\mathrm{T}}(W_\delta \boldsymbol{\delta}_{ij} + W_\rho \rho_{ij}) \big]$.

```
# 3. Refine each pair independently
```
$\mathbf{P}_{ij} \leftarrow \overline{\mathbf{P}}_{ij} + \mathrm{MLP}_P\big(\mathrm{Norm}_P(\overline{\mathbf{P}}_{ij})\big)$
return $(\mathbf{S}_{\mathcal{C}}, \mathbf{P})$

Pairmixer block. Algorithm 3 updates only the pair representation using outgoing and incoming triangle multiplication followed by a gated pair transition. The mask $\mathbf{m}$ restricts all pair operations to valid atoms.

Algorithm 3 Pairmixer block

Require: Pair representation $\mathbf{P}$ and valid-atom mask $\mathbf{m}$
Ensure: Updated pair representation $\mathbf{P}$

$M_{ij} \leftarrow m_i m_j$

`# 1. Aggregate information over outgoing and incoming triangles`

$\mathrm{Contract}_{\mathrm{out}}(\mathbf{A},\mathbf{B})_{ij} \leftarrow \sum_k \mathbf{A}_{ik} \odot \mathbf{B}_{jk}, \qquad \mathrm{Contract}_{\mathrm{in}}(\mathbf{A},\mathbf{B})_{ij} \leftarrow \sum_k \mathbf{A}_{ki} \odot \mathbf{B}_{kj}$

for d in the ordered sequence $(\mathrm{out}, \mathrm{in})$ **do**

$\overline{\mathbf{P}} \leftarrow \mathrm{Norm}_d(\mathbf{P})$

$[\mathbf{A},\mathbf{B}] \leftarrow \mathrm{Split}\big(W_p^d \overline{\mathbf{P}} \odot \sigma(W_g^d \overline{\mathbf{P}})\big) \odot \mathbf{M}$

$\mathbf{U} \leftarrow \mathrm{Contract}_d(\mathbf{A},\mathbf{B})$

$\mathbf{P} \leftarrow \mathbf{P} + W_o^d \,\mathrm{Norm}'_d(\mathbf{U}) \odot \sigma(W_{go}^d \overline{\mathbf{P}})$

end for

`# 2. Mix feature channels within each pair`

$\overline{\mathbf{P}} \leftarrow \mathrm{Norm}_{\mathrm{tr}}(\mathbf{P})$

$\mathbf{P} \leftarrow \mathbf{P} + W_{\mathrm{tr}}\big[\mathrm{SiLU}(W_a \overline{\mathbf{P}}) \odot W_b \overline{\mathbf{P}}\big]$

return $\mathbf{P}$

Noisy-state and time embedding. In Algorithm 4, $\boldsymbol{\Omega}$ and $\boldsymbol{\eta}$ are fixed random Fourier frequencies and phases for encoding the coordinates, while $\boldsymbol{\omega}$ contains the fixed frequencies used for the sinusoidal time embedding.

Algorithm 4 Noisy-state and time embedding

Require: Condition-only single representation $\mathbf{S}_{\mathcal{C}}$, noisy coordinates $\mathbf{X}_t$, noisy lattice $\boldsymbol{\ell}_t$, and time t
Ensure: DiT input $\mathbf{S}$ and time condition $\mathbf{c}_t$

`# 1. Embed noisy coordinates into the single track`

$\boldsymbol{\Phi}_i \leftarrow \sqrt{2}\cos(\boldsymbol{\Omega} \odot \mathbf{X}_{t,i} + \boldsymbol{\eta})$ ▷ fixed random Fourier features

$\mathbf{e}_i^X \leftarrow \mathrm{MLP}_X(\mathrm{vec}(\boldsymbol{\Phi}_i))$

`# 2. Embed and broadcast the noisy lattice`

$\mathbf{e}^L \leftarrow \mathrm{MLP}_L(\boldsymbol{\ell}_t)$

$\mathbf{S}_i \leftarrow \mathbf{S}_{\mathcal{C},i} + \mathbf{e}_i^X + \mathbf{e}^L$ ▷ $\mathbf{e}^L$ is broadcast over atoms

`# 3. Embed flow time`

$\boldsymbol{\psi}(t) \leftarrow [\cos(\boldsymbol{\omega} t), \sin(\boldsymbol{\omega} t)]$ ▷ fixed sinusoidal frequencies

$\mathbf{c}_t \leftarrow \mathrm{MLP}_t(\boldsymbol{\psi}(t))$

return $(\mathbf{S}, \mathbf{c}_t)$

DiT block. In Algorithm 5, $\boldsymbol{\beta}$, $\boldsymbol{\alpha}$, and $\boldsymbol{\gamma}$ denote the adaptive shift, scale, and residual-gate vectors, respectively. Subscripts A and F identify attention and feed-forward modulation parameters.

Algorithm 5 Pair-biased DiT block

Require: Single representation $\mathbf{S}$, time condition $\mathbf{c}_t$, refined pair representation $\mathbf{P}^\star$, and mask $\mathbf{m}$
Ensure: Updated single representation $\mathbf{S}$

`# 1. Generate time-dependent modulation`

$[\boldsymbol{\beta}_A, \boldsymbol{\alpha}_A, \boldsymbol{\gamma}_A, \boldsymbol{\beta}_F, \boldsymbol{\alpha}_F, \boldsymbol{\gamma}_F] \leftarrow W_c \phi(\mathbf{c}_t)$

$\mathbf{U}_i \leftarrow (1 + \boldsymbol{\alpha}_A) \odot \mathrm{Norm}_1(\mathbf{S}_i) + \boldsymbol{\beta}_A$

$\widetilde{\mathbf{U}}_i \leftarrow \mathrm{Norm}_A(\mathbf{U}_i)$

`# 2. Apply self-attention with pair bias`

$\mathbf{q}_i^h \leftarrow \mathrm{Norm}_q(W_q^h \widetilde{\mathbf{U}}_i)$, $\mathbf{k}_i^h \leftarrow \mathrm{Norm}_k(W_k^h \widetilde{\mathbf{U}}_i)$, $\mathbf{v}_i^h \leftarrow W_v^h \widetilde{\mathbf{U}}_i$

$b_{ij}^h \leftarrow W_P^h \mathrm{Norm}_P(\mathbf{P}_{ij}^\star)$

$a_{ij}^h \leftarrow \mathrm{softmax}_j\left(\frac{\langle \mathbf{q}_i^h, \mathbf{k}_j^h \rangle}{\sqrt{d_h}} + b_{ij}^h - \infty(1 - m_j)\right)$

$\mathbf{o}_i^h \leftarrow \sum_j a_{ij}^h \mathbf{v}_j^h$

$\mathbf{o}_i \leftarrow W_o\left[\sigma(W_g \widetilde{\mathbf{U}}_i) \odot \mathrm{Concat}_h(\mathbf{o}_i^h)\right]$

$\mathbf{S}_i \leftarrow \mathbf{S}_i + \boldsymbol{\gamma}_A \odot \mathbf{o}_i$

`# 3. Apply the gated feed-forward update`

$\mathbf{H}_i \leftarrow (1 + \boldsymbol{\alpha}_F) \odot \mathrm{Norm}_2(\mathbf{S}_i) + \boldsymbol{\beta}_F$

$\mathbf{f}_i \leftarrow W_{\text{down}}[\phi(W_{\text{gate}} \mathbf{H}_i) \odot W_{\text{up}} \mathbf{H}_i]$

$\mathbf{S}_i \leftarrow \mathbf{S}_i + \boldsymbol{\gamma}_F \odot \mathbf{f}_i$

return $\mathbf{S}$

Prediction heads. Algorithm 6 maps the final single representation to centered Cartesian coordinates and a pooled lattice prediction over valid atoms.

Algorithm 6 Coordinate and lattice prediction heads

Require: Final single representation $\mathbf{S}$ and valid-atom mask $\mathbf{m}$
Ensure: Coordinate endpoint $\widehat{\mathbf{X}}_1$ and lattice endpoint $\widehat{\boldsymbol{\ell}}_1$

$N \leftarrow \sum_i m_i$

`# 1. Predict centered atomic coordinates`

$\mathbf{r}_i \leftarrow \mathrm{Linear}_X(\mathrm{Norm}_X(\mathbf{S}_i))$

$\widehat{\mathbf{X}}_{1,i} \leftarrow \mathbf{r}_i - \frac{1}{N} \sum_j m_j \mathbf{r}_j$

`# 2. Pool atom features and predict the lattice`

$\bar{\mathbf{s}} \leftarrow \frac{1}{N} \sum_i m_i \mathrm{Norm}_L(\mathbf{S}_i)$

$\widehat{\boldsymbol{\ell}}_1 \leftarrow \mathrm{MLP}_L(\bar{\mathbf{s}})$

return $(\widehat{\mathbf{X}}_1, \widehat{\boldsymbol{\ell}}_1)$

D DATASET STATISTICS

We use the official CLARI split (Lo et al., 2026) from the Cambridge Structural Database (CSD) (Groom et al., 2016). To prevent leakage, CLARI excludes all entries from test refcode families and training structures sharing an RDKit-sanitizable molecular component with more than seven heavy atoms with the test sets, then holds out 1,000 refcode families for validation.

We report statistics for the CLARI training and validation splits used in our benchmark experiments (Section 4). Our preprocessing adds missing hydrogens using the CSD Python API (Sykes et al., 2024) before applying the 512-atom limit. This reduces the training split from 917,014 to 912,807 structures and the validation split from 1,048 to 1,047. We report both statistics for completeness.

Table 2: **Size statistics for the CLARI training and validation splits.** Statistics are reported before and after applying the 512-atom unit-cell limit following hydrogen completion.

	Unfiltered				$\leq$ 512 atoms			
Property	Min	Mean	Median	Max	Min	Mean	Median	Max
Training								
Atoms / unit cell	5	203.65	180	3,558	5	201.71	180	512
Molecules / unit cell	1	5.42	4	246	1	5.35	4	160
Atoms / molecule	1.06	48.96	42.00	682.00	1.06	48.89	42.00	512.00
Formula-unit count Z	1	3.34	4	40	1	3.33	4	40
Validation								
Atoms / unit cell	10	205.95	185	617	10	205.56	184	512
Molecules / unit cell	1	5.51	4	99	1	5.42	4	52
Atoms / molecule	5.00	48.42	42.10	242.00	5.00	48.46	42.20	242.00
Formula-unit count Z	1	3.39	4	16	1	3.39	4	16

We additionally report chemical composition, molecular flexibility, and the availability of optional conditioning information. Similar to Jin et al. (2025), we label a structure as rigid if every molecular component contains at most three rotatable bonds and as flexible if any component contains more than three. This provides a simple heuristic rather than a complete description of molecular flexibility. Table 3 summarizes these properties before and after the 512-atom filtering.

Table 3: **Composition, flexibility, and conditioning availability in the CLARI splits.** Entries report the number of structures before and after applying the 512-atom unit-cell limit.

	Training		Validation	
Property	Unfiltered	$\leq$ 512 atoms	Unfiltered	$\leq$ 512 atoms
Composition				
Organic	491,219	490,475	573	573
Organometallic	425,795	422,332	475	474
Molecular flexibility				
Rigid	194,017	193,620	226	226
Flexible	722,997	719,187	822	821
Conditioning availability				
Template	598,141	596,447	683	683
Stereochemistry	917,014	912,807	1,048	1,047
Space group	917,012	912,805	1,048	1,047

E TRAINING AND SAMPLING SETTINGS

We summarize the hyperparameters used for PACKORA-M and PACKORA-L. Unless noted, all settings are shared between the two model sizes.

Architecture. Table 4 specifies model capacity and backbone configuration; the two models differ only in width, head count, and parameter count.

Table 4: **Architecture hyperparameters.** Model capacity and backbone configuration for PACKORA-M and PACKORA-L.

Parameter	PACKORA-M	PACKORA-L
Capacity		
Parameters	88M	187M
Single width	512	768
Pair width	128	192
Attention heads	8	12
Backbone		
Pairmixer blocks	4	4
DiT blocks	16	16
MLP ratio	4	4
QK normalization	On	On
Feed-forward network	SwiGLU	SwiGLU
Dropout	0	0

Training. Table 5 summarizes the flow-matching, optimization, conditioning, and checkpoint-selection settings.

Table 5: **Training hyperparameters.** Flow-matching, optimization, conditioning, and model-selection settings. All settings are shared between PACKORA-M and PACKORA-L unless noted.

Parameter	Value
Flow matching	
Priors	Centered Gaussian for $\mathbf{X}$; standard Gaussian for $\boldsymbol{\ell}$
Interpolant	Linear ($x_t = (1-t)x_0 + tx_1$)
Time distribution	$0.98\ \mathrm{Beta}(1.9, 1.0) + 0.02\ \mathrm{Uniform}(0, 1)$
Objective	L^1 coordinate–lattice loss
Loss weights	Coordinates 10; lattice 1
Optimization	
Optimizer	Muon for hidden matrices; AdamW otherwise
Learning rate	10^{-4}
Weight decay	0 (PACKORA-M); 10^{-2} (PACKORA-L)
Muon settings	Momentum 0.95; Nesterov; 5 Newton–Schulz steps
AdamW settings	$\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$
Schedule	10,000-step linear warmup from zero
Gradient clipping	Global norm, 1.0
EMA decay	0.9999
Conditioning and augmentation	
Condition dropout	Template 0.5; stereochemistry 0.5; space group 0.9
Augmentation	Crystal translation; template rotation and translation
Runtime and selection	
Precision	bfloat16 mixed precision
Checkpoint interval	Every 50 epochs
Checkpoint selection	Highest validation Sol_C

Curriculum and batching. We use two-stage training with fixed-shape batches to enable `torch.compile`. Each structure is assigned to the smallest atom-count bucket that contains it and padded to that bucket size.

Table 6: **Two-stage training curriculum and batching.** Atom-count buckets and batch sizes used for fixed-shape training across eight GPUs.

Parameter	Value
Stage 1	
Maximum atoms	300
Bucket caps	32, 64, 96, 128, 160, 192, 224, 256, 288, 300
Per-GPU batch	16 for every bucket
Global batch	128
Stage 2	
Maximum atoms	512
Additional bucket caps	320, 352, 384, 416, 448, 480, 512
Per-GPU batch	16 ($\leq$ 300), 14 (320), 11 (352), 9 (384), 8 (416), 7 (448), 6 (480), 5 (512)
Global batch	128, 112, 88, 72, 64, 56, 48, and 40, respectively

Sampling. Both model sizes use the same sampling and conditioning configuration for benchmark generation.

Table 7: **Sampling hyperparameters.** Sampling and conditioning configuration used for benchmark generation.

Parameter	Value
Numerical solver	
Solver	EDM–Heun with stochastic churn
Noise schedule	Karras
Sampling steps	200
Sigma range	$\sigma_{\min} = 0.002$, $\sigma_{\max} = 80$
Schedule exponent	$\rho = 7$
Churn settings	$S_{\text{churn}} = 60$, $S_{\min} = 0$, $S_{\max} = 999$, $S_{\text{noise}} = 1.003$
Generation protocol	
Weights	EMA
Conditioning	Template and available stereochemistry on; space group off
Template augmentation	Random rotation and translation
Candidates per target	1,000
Random seed	42

F COMPUTATIONAL REQUIREMENTS

We report the training, generation, and relaxation costs of PACKORA. Training is a one-time cost, while generation and relaxation recur for each new molecular system. Inference costs follow the candidate budgets and relaxation protocol in Section 4.3.

F.1 TRAINING

Benchmark models. Table 8 reports the training costs of PACKORA-M and PACKORA-L across both curriculum stages: 107.1 and 157.8 hours on eight H200 GPUs, respectively. Durations are measured from recorded launch times to checkpoint timestamps, including overhead within each run but excluding the gap between stages.

Ablations. After fixing the Pairmixer architecture, eight representative 300-epoch training runs required an average of 54.2 hours on eight NVIDIA H200 GPUs (433.9 GPU-hours), with a range of 52.3–54.7 hours. These comprise the baseline, EMA-0.999, AdamW, four alternative coordinate-to-lattice loss ratios, and the L^2 objective.

Table 8: **Training costs.** Wall time is measured in hours; GPU-hours equal unrounded wall time multiplied by eight H200 GPUs. The ablation row averages eight distinct runs.

Model / study	Stage	Epochs	Wall time (h)	GPU-hours
PACKORA-M	$\leq$ 300 atoms	250	51.9	415.1
	$\leq$ 512 atoms	160	55.2	441.6
	Total	410	107.1	856.7
PACKORA-L	$\leq$ 300 atoms	400	104.1	833.0
	$\leq$ 512 atoms	120	53.6	429.1
	Total	520	157.8	1,262.1
Ablations	Training	300	54.2	433.9

Potential reductions. Our training runs use compiled networks and mixed precision. Specialized triangle-multiplication kernels, such as those provided by NVIDIA cuEquivariance,[1] could further reduce training time but were not used in our experiments.

[1] https://docs.nvidia.com/cuda/cuequivariance/api/generated/cuequivariance_torch.triangle_multiplicative_update.html

F.2 GENERATION AND RELAXATION

Protocol and timing. We report computational costs on eight NVIDIA H200 GPUs. Generation uses 200-step EDM–Heun for PACKORA and 20-step Heun for CLARI-H (Lo et al., 2026). Relaxation uses UMA-S-1.2 with the OMC task (Wood et al., 2025; Gharakhanyan et al., 2026) and ASE BFGS (Larsen et al., 2017), with one structure relaxed at a time per GPU. We attempt relaxation for all generated benchmark candidates. For the single-polymorph benchmark, we estimate generation time for 26,000 candidates by scaling recorded generation times in proportion to candidate count. Multi-polymorph generation and both benchmarks' relaxation times are reconstructed from recorded job timestamps. These timings include startup and compilation-cache effects. Combined time sums generation and relaxation, excluding inter-stage idle time and subsequent evaluation.

Table 9: **Generation and relaxation costs and recovery.** Each method generates 1,000 candidates per benchmark target or CSD entry and attempts relaxation for all candidates: 26,000 for single-polymorph and 28,000 for multi-polymorph. Single-polymorph generation times are proportional estimates. Bold indicates the lowest time or highest recovery within each benchmark.

Method	Generation (min) ↓	Relaxation (h) ↓	Combined (h) ↓	Recovered ↑
Single-polymorph				
CLARI-H	**1.61**	16.82	16.84	16/26
PACKORA-M	14.42	11.42	11.66	17/26
PACKORA-L	14.25	**10.22**	**10.46**	**19/26**
Multi-polymorph				
CLARI-H	**1.23**	16.79	16.81	18/29
PACKORA-M	9.25	10.61	10.77	19/29
PACKORA-L	8.42	**8.93**	**9.07**	**21/29**

Single-polymorph benchmark. For the 26 benchmark targets, estimated combined time is 11.66 hours for PACKORA-M and 10.46 hours for PACKORA-L, compared with 16.84 hours for CLARI-H. This corresponds to approximately 31% and 38% less time, while recovery increases from 16/26 targets to 17/26 and 19/26, respectively.

Multi-polymorph benchmark. Each method generates 28,000 candidates across 28 CSD entries and attempts relaxation for all of them. Combined time is 10.77 and 9.07 hours for PACKORA-M and PACKORA-L, compared with 16.81 hours for CLARI-H: approximately 36% and 46% less time, while recovering 19 and 21 experimental forms rather than 18.

Practical implication. Under the shared relaxation protocol, PACKORA's additional generation time is more than offset by shorter relaxation, yielding lower combined cost and higher experimental-form recovery on both benchmarks.

G STRUCTURE GENERATION EVALUATION

This section details the evaluation protocol for structure generation. We formalize the crystal solve rate Sol_C (Jin et al., 2025), including COMPACK (Chisholm & Motherwell, 2005) matching, collision detection, and experimental-reference selection. We also summarize the differences between the released OXtal and CLARI evaluators that motivate our re-evaluation of CLARI.

Definition of crystal solve rate. Let $c \in \{1, \ldots, C\}$ index targets and $s \in \{1, \ldots, n_c\}$ index generated candidates for target c. The crystal-level solve rate is

$$\mathrm{Sol}_C = \frac{1}{C} \sum_{c=1}^{C} \max_{1 \le s \le n_c} \mathrm{sol}_{c,s}, \tag{2}$$

where $\mathrm{sol}_{c,s}$ indicates whether candidate $x_{c,s}$ solves target c:

$$\mathrm{sol}_{c,s} = \mathbf{1}\left[\mathrm{col}_{c,s} = 0 \ \wedge \ n_{15}(x_{c,s}, t^{\star}_{c,s}) \ge 8 \ \wedge \ r_{15}(x_{c,s}, t^{\star}_{c,s}) < 2\,\text{Å}\right], \tag{3}$$

where $\mathrm{col}_{c,s}$ is the collision indicator; n_{15} and r_{15} denote the number of matched molecules and RMSD, respectively, returned by COMPACK at packing size 15; and $t^{\star}_{c,s}$ is the selected experimental reference.

Intuitively, a candidate is collision-free ($\mathrm{col}_{c,s} = 0$) if all nonbonded heavy-atom pairs are separated beyond the collision threshold. Specifically,

$$\mathrm{col}_{c,s} = \mathbf{1}\left[\exists (i,j) : r_i^{\mathrm{vdW}} + r_j^{\mathrm{vdW}} - d_{ij}^{(c,s)} \ge 0.7\,\text{Å}\right], \tag{4}$$

where $d_{ij}^{(c,s)}$ is the distance between a nonbonded heavy-atom pair (i, j), and r_i^{vdW} and r_j^{vdW} are their van der Waals radii.

For targets with multiple experimental references, we select the best-matching reference as $t^{\star}_{c,s}$. Let $T(c)$ denote the reference set for target c and define

$$P_{c,s} = \{t \in T(c) : n_{15}(x_{c,s}, t) \ge 8\}. \tag{5}$$

OXtal selects

$$t^{\star}_{c,s} = \begin{cases} \arg\min_{t \in P_{c,s}} r_{15}(x_{c,s}, t), & P_{c,s} \neq \varnothing, \\ \arg\max_{t \in T(c)} n_{15}(x_{c,s}, t), & P_{c,s} = \varnothing. \end{cases} \tag{6}$$

Candidate budgets and bootstrapping. We evaluate coverage at budgets of 30 and 1,000 candidates. Following CLARI (Lo et al., 2026), we estimate 30-candidate coverage by drawing 5,000 resamples of 30 candidates from each target's fixed pool of 1,000 and averaging the resulting Sol_C. The strict metric $\mathrm{Sol}_C^{15/15}$ requires all 15 molecules to match, retaining the same collision and RMSD criteria.

Differences between the OXtal and CLARI evaluators. The released CLARI evaluator differs from the OXtal evaluator in two aspects: (1) collision detection, including the treatment of intermolecular collisions, and (2) selection of the best experimental reference for targets with multiple reference structures. For a consistent comparison, we therefore re-evaluate CLARI-M, CLARI-L, and CLARI-H using the OXtal definitions.

H STRUCTURE GENERATION RESULTS

This appendix expands the structure-generation results in Section 4.2 across candidate budgets and evaluation criteria. At 1,000 candidates per target, Figure 10 reports the standard crystal solve rate, while Figure 11 reports the strict solve rate, requiring all 15 molecules to match under COMPACK. These full-pool results complement the 30-candidate comparisons in Figures 1 and 4. For the 30-candidate budget, Table 10 lists both solve rates with resampling standard deviations, and Table 11 provides the corresponding collision, packing, and recovery metrics alongside the standard solve rate.

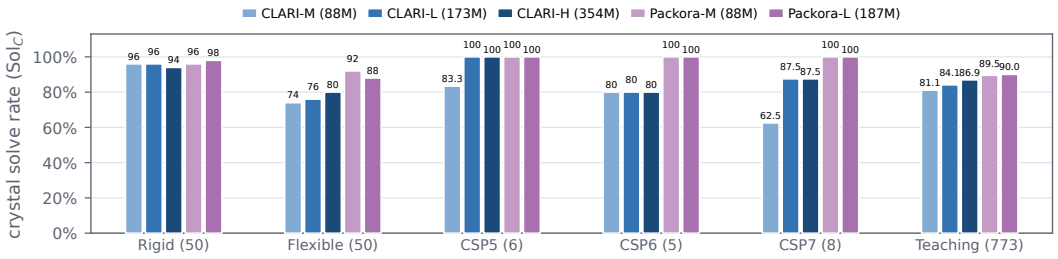


Figure 10: **Crystal solve rate with 1,000 candidates per target.** We report Sol_C, with parentheses indicating benchmark sizes. PACKORA achieves the highest solve rate across all six benchmarks.

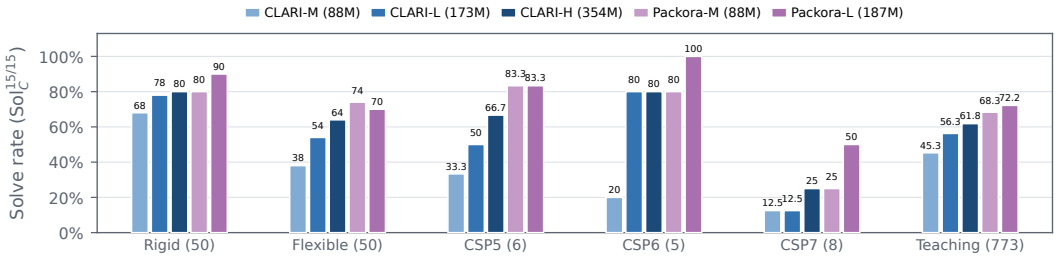


Figure 11: **Strict crystal solve rate with 1,000 candidates per target.** We report $\mathrm{Sol}_C^{15/15}$, requiring all 15 molecules to match under COMPACK, with parentheses indicating benchmark sizes. PACKORA achieves the highest solve rate across all six benchmarks.

Table 10: **Structure-generation results with 30 candidates per target.** We report standard Sol_C and strict $\mathrm{Sol}_C^{15/15}$. CLARI and PACKORA results are mean ± standard deviation over 5,000 bootstrap resamples of 30 from 1,000-candidate pools; other baselines are reported as provided in Jin et al. (2025). **Bold** and underline denote the best and second-best generative results, respectively.

Method	Rigid (50)	Flexible (50)	CSP5 (6)	CSP6 (5)	CSP7 (8)	Teaching (773)
Sol_C						
A-Transformer	0.060	0.000	0.000	0.000	0.000	–
AssembleFlow	0.000	0.000	0.000	0.000	0.000	–
OXtal	0.300	0.220	0.167	0.200	0.125	–
CLARI-M	0.692 ± 0.038	0.283 ± 0.044	0.410 ± 0.136	0.286 ± 0.119	0.154 ± 0.102	0.506 ± 0.010
CLARI-L	0.728 ± 0.039	0.336 ± 0.043	0.541 ± 0.160	0.332 ± 0.138	0.138 ± 0.105	0.549 ± 0.010
CLARI-H	0.740 ± 0.039	0.396 ± 0.042	0.598 ± 0.146	0.412 ± 0.158	0.134 ± 0.112	0.590 ± 0.010
PACKORA-M	**0.807** ± 0.031	0.498 ± 0.047	0.732 ± 0.130	0.426 ± 0.154	0.212 ± 0.102	0.637 ± 0.009
PACKORA-L	0.795 ± 0.035	**0.530** ± 0.047	**0.793** ± 0.081	**0.433** ± 0.165	**0.238** ± 0.126	**0.663** ± 0.009
$\mathrm{Sol}_C^{15/15}$						
CLARI-M	0.264 ± 0.038	0.041 ± 0.025	0.069 ± 0.091	0.006 ± 0.033	0.010 ± 0.034	0.165 ± 0.008
CLARI-L	0.351 ± 0.041	0.107 ± 0.037	0.131 ± 0.100	0.063 ± 0.102	0.018 ± 0.043	0.223 ± 0.009
CLARI-H	0.424 ± 0.043	0.155 ± 0.039	0.230 ± 0.124	0.088 ± 0.122	0.011 ± 0.036	0.275 ± 0.009
PACKORA-M	**0.508** ± 0.039	0.262 ± 0.043	0.374 ± 0.149	0.108 ± 0.127	0.011 ± 0.036	0.367 ± 0.009
PACKORA-L	0.476 ± 0.045	**0.296** ± 0.043	**0.489** ± 0.172	**0.254** ± 0.128	**0.025** ± 0.053	**0.394** ± 0.009

Table 11: **Full metrics at a 30-candidate budget.** We report per-sample collision, packing, and recovery rates (Col_S, Pac_S, Rec_S) and crystal-level packing, recovery, and solve rates (Pac_C, Rec_C, Sol_C), following the definitions in Jin et al. (2025). Other baselines use their exact 30-candidate pools, while PACKORA-M and PACKORA-L report means over 5,000 seeded draws of 30 candidates from their 1,000-candidate pools. CLARI is omitted because only Sol_C is available.

Dataset	Method	$\mathrm{Col}_S \downarrow$	$\mathrm{Pac}_S \uparrow$	$\mathrm{Pac}_C \uparrow$	$\mathrm{Rec}_S \uparrow$	$\mathrm{Rec}_C \uparrow$	$\mathrm{Sol}_C \uparrow$
Rigid	A-Transformer	0.731	0.015	0.060	0.033	0.120	0.060
	AssembleFlow	0.524	0.001	0.040	0.211	0.760	0.000
	OXtal	0.011	0.873	1.000	0.737	0.960	0.300
	PACKORA-M	0.036	0.433	0.876	0.860	0.978	0.807
	PACKORA-L	0.028	0.405	0.872	0.847	0.979	0.795
Flexible	A-Transformer	0.900	0.001	0.020	0.000	0.000	0.000
	AssembleFlow	0.883	0.000	0.000	0.021	0.140	0.000
	OXtal	0.097	0.291	0.900	0.048	0.400	0.220
	PACKORA-M	0.030	0.099	0.578	0.346	0.879	0.498
	PACKORA-L	0.031	0.099	0.605	0.371	0.920	0.530
CSP5	A-Transformer	0.833	0.000	0.000	0.000	0.000	0.000
	AssembleFlow	0.717	0.000	0.000	0.150	0.500	0.000
	OXtal	0.006	0.667	0.833	0.572	0.833	0.167
	PACKORA-M	0.023	0.251	0.808	0.784	0.915	0.732
	PACKORA-L	0.018	0.328	0.797	0.792	0.902	0.793
CSP6	A-Transformer	0.967	0.000	0.000	0.000	0.000	0.000
	AssembleFlow	0.800	0.000	0.000	0.073	0.200	0.000
	OXtal	0.013	0.660	1.000	0.160	0.600	0.200
	PACKORA-M	0.070	0.177	0.607	0.559	0.851	0.426
	PACKORA-L	0.052	0.193	0.661	0.578	0.902	0.433
CSP7	A-Transformer	0.950	0.000	0.000	0.000	0.000	0.000
	AssembleFlow	0.808	0.000	0.000	0.063	0.250	0.000
	OXtal	0.021	0.483	0.875	0.129	0.375	0.125
	PACKORA-M	0.178	0.041	0.396	0.251	0.656	0.212
	PACKORA-L	0.174	0.035	0.469	0.278	0.681	0.238
Teaching	PACKORA-M	0.057	0.281	0.704	0.536	0.842	0.637
	PACKORA-L	0.050	0.288	0.730	0.554	0.859	0.663

I EMPIRICAL Z SAMPLING

Unlike the supplied-Z benchmarks, this setting samples the unknown formula-unit count Z from a training-set prior.

Prior construction. We group molecular components by exact component SMILES in the 912,807 CLARI training structures retained after the 512-atom filter. For component multiplicities n_k, reduced stoichiometry r_k, and c_z training structures with $Z = z$, we define

$$Z = \gcd(n_1, \ldots, n_K), \qquad r_k = n_k/Z, \qquad \widehat{p}_{\text{train}}(Z = z) = \frac{c_z}{\sum_{z'} c_{z'}}.$$

The prior has support

$$\{1, \ldots, 10, 12, 14, 15, 16, 18, 20, 24, 26, 27, 28, 30, 32, 34, 40\}.$$

Sampling. For each target, let $N_{\text{fu}} = \sum_k r_k N_k$ be the atoms per formula unit, where N_k is the atom count of component k. We truncate and renormalize the prior to $N_{\text{fu}} Z \leq 4096$. For each candidate, we independently sample Z, replicate the formula unit, and pass the resulting composition to PACKORA. All other benchmark settings are unchanged.

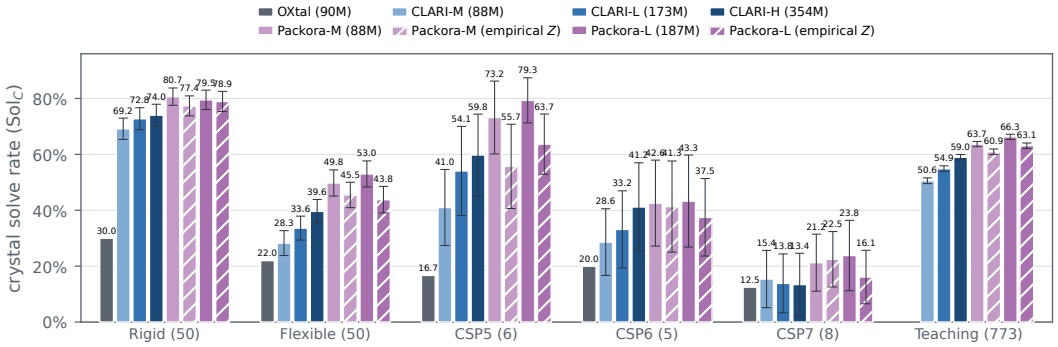


Figure 12: **Crystal solve rate with empirical Z sampling at 30 candidates per target.** We report Sol_C, with parentheses indicating benchmark sizes. Solid bars reproduce Figure 1; hatched bars use independently sampled empirical Z for each PACKORA candidate. OXtal results are published 30-candidate scores. CLARI and PACKORA results show means and standard deviations over 5,000 bootstrap resamples of 30 from 1,000 candidates per target.

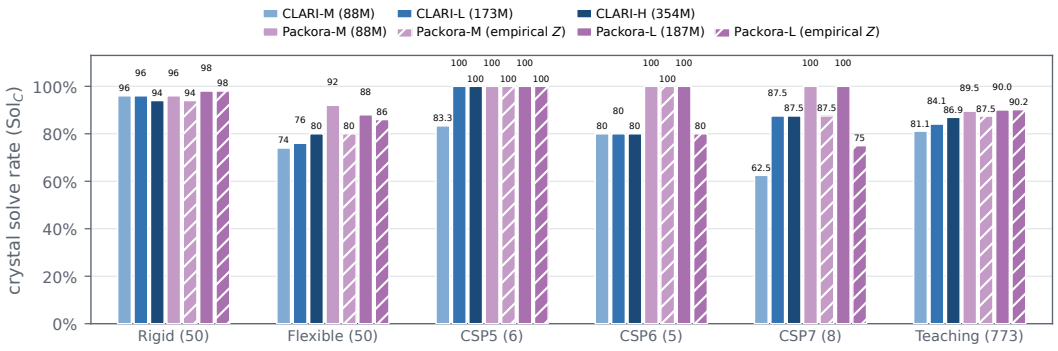


Figure 13: **Crystal solve rate with empirical Z sampling at 1,000 candidates per target.** We report Sol_C, with parentheses indicating benchmark sizes. Solid bars reproduce Figure 10; hatched bars use independently sampled empirical Z for each PACKORA candidate.

Results. Figures 12 and 13 compare standard solve rates at 30 and 1,000 candidates; Figures 14 and 15 report strict matching. Both model sizes remain competitive with supplied Z. Figure 16 compares the two settings across candidate budgets on Rigid and Flexible targets.

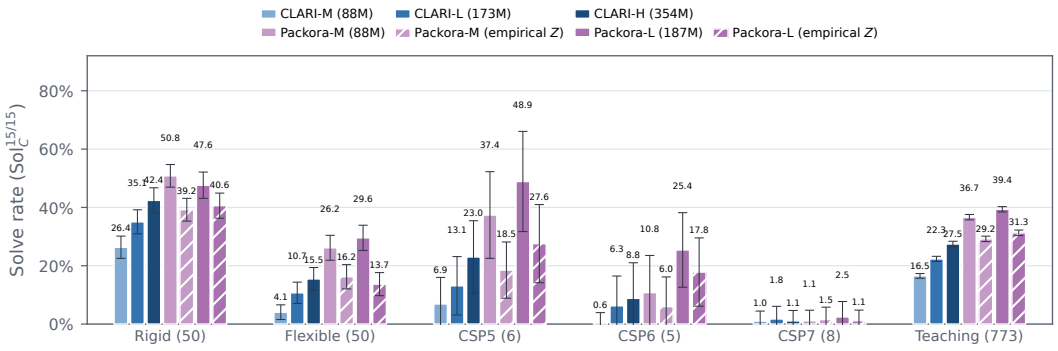


Figure 14: **Strict crystal solve rate with empirical Z sampling at 30 candidates per target.** We report $\mathrm{Sol}_C^{15/15}$, requiring all 15 molecules to match under COMPACK, with parentheses indicating benchmark sizes. Solid bars reproduce Figure 4; hatched bars use independently sampled empirical Z for each PACKORA candidate. Bars and error bars show means and standard deviations over 5,000 bootstrap resamples of 30 from 1,000 candidates per target.

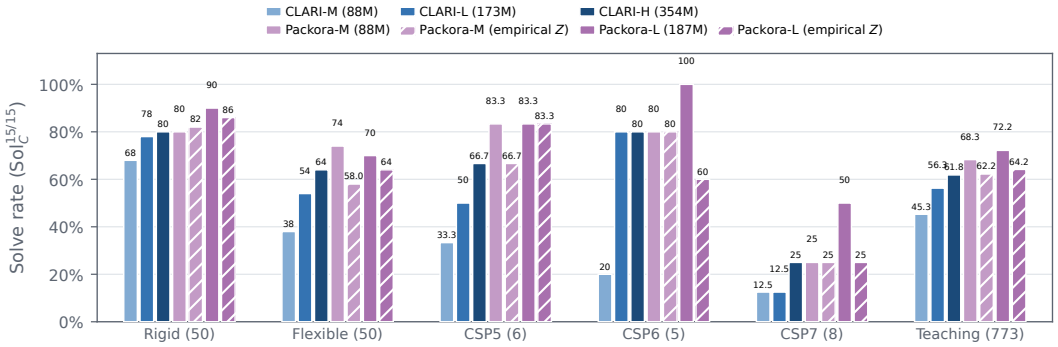


Figure 15: **Strict crystal solve rate with empirical Z sampling at 1,000 candidates per target.** We report $\mathrm{Sol}_C^{15/15}$, requiring all 15 molecules to match under COMPACK, with parentheses indicating benchmark sizes. Solid bars reproduce Figure 11; hatched bars use independently sampled empirical Z for each PACKORA candidate.

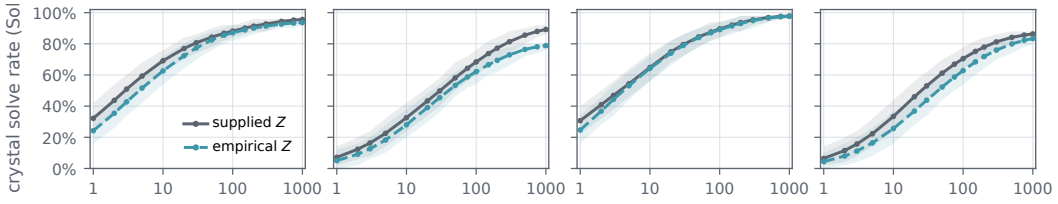


Figure 16: **Effect of empirical unit-cell Z sampling across candidate budgets.** Curves show mean Sol_C for PACKORA-M and PACKORA-L on the Rigid and Flexible benchmarks over 5,000 resamples, and shaded bands show the 2.5th–97.5th percentiles. The gray solid lines denote supplied Z, and the teal dotted lines denote empirical Z. The two settings remain close across candidate budgets.

J STRUCTURE OVERLAYS

We visualize 18 representative strict matches by overlaying PACKORA-L predictions with their experimental references. Experimental structures are shown as wider gray sticks and PACKORA-L predictions as narrower green sticks; hydrogen atoms are omitted for clarity. The overlays are rendered with py3Dmol using the 3Dmol.js WebGL backend (Rego & Koes, 2015).

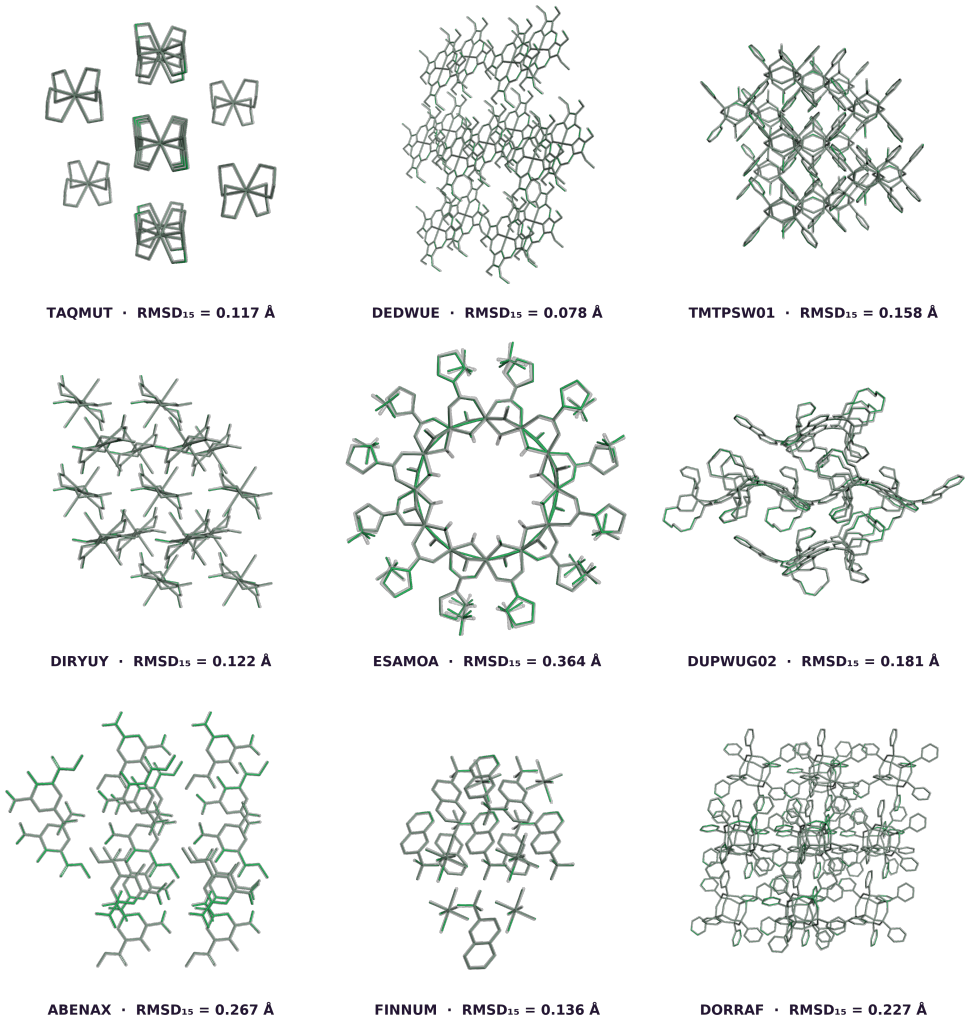


Figure 17: **Structure overlays for PACKORA-L predictions.** Experimental structures are shown in gray and PACKORA-L predictions in green. Labels give CSD refcodes and RMSD_{15}.

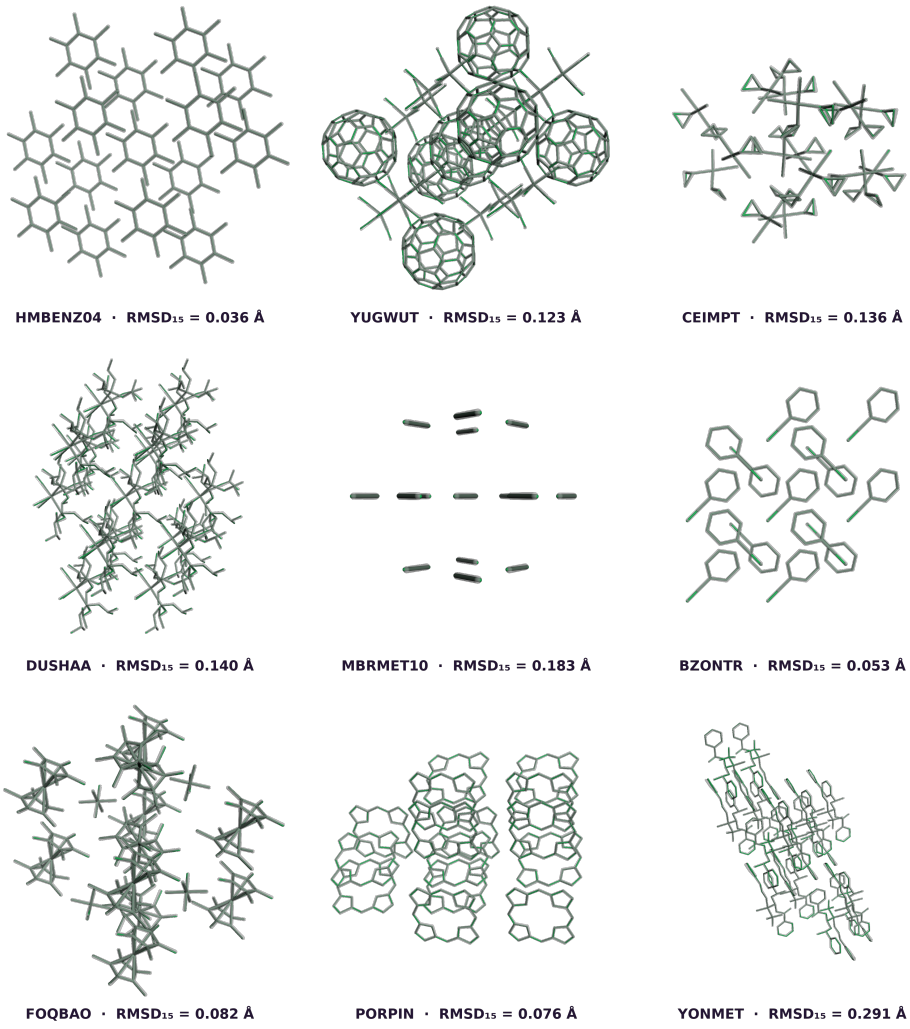


Figure 18: **Structure overlays for** PACKORA**-L predictions.** Experimental structures are shown in gray and PACKORA-L predictions in green. Labels give CSD refcodes and RMSD_{15}.

K SPACE-GROUP CONDITIONING

We test known space-group conditioning on the six benchmarks in Section 4.2; the main setting omits this label.

Protocol. We use the main PACKORA-M and PACKORA-L checkpoints and sampling setup, adding the target space-group label to templates and stereochemistry with conditioning weight 1. We retain supplied Z and generate 1,000 candidates per target from seeds 42–49 and evaluate a 30-candidate budget by bootstrap resampling. Evaluation follows Appendix G.

Results. Figures 19 and 20 compare standard and strict solve rates with and without space-group conditioning at 30 candidates per target. At 30 candidates, strict solve rates increase on all six benchmarks for both model sizes. On CSP5, they rise from 37.4% to 56.1% for PACKORA-M and from 48.9% to 66.6% for PACKORA-L; on Teaching, they rise from 36.7% to 43.5% and from 39.4% to 46.2%, respectively. Standard 30-candidate solve rates also increase on all six benchmarks for PACKORA-L, while PACKORA-M shows mixed changes.

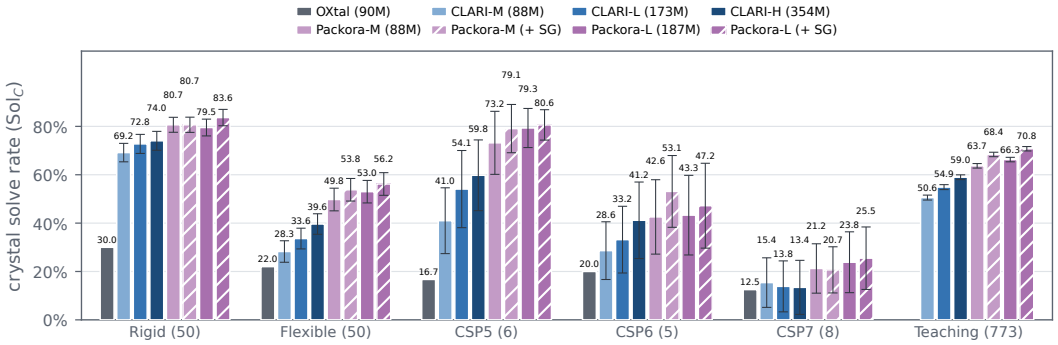


Figure 19: **Crystal solve rate with space-group conditioning at 30 candidates per target.** We report Sol_C, with parentheses indicating benchmark sizes. Solid bars reproduce Figure 1; hatched bars use the target space-group label in addition to templates and stereochemistry. OXtal results are published 30-candidate scores. CLARI and PACKORA results show means and standard deviations over 5,000 bootstrap resamples of 30 from 1,000 candidates per target.

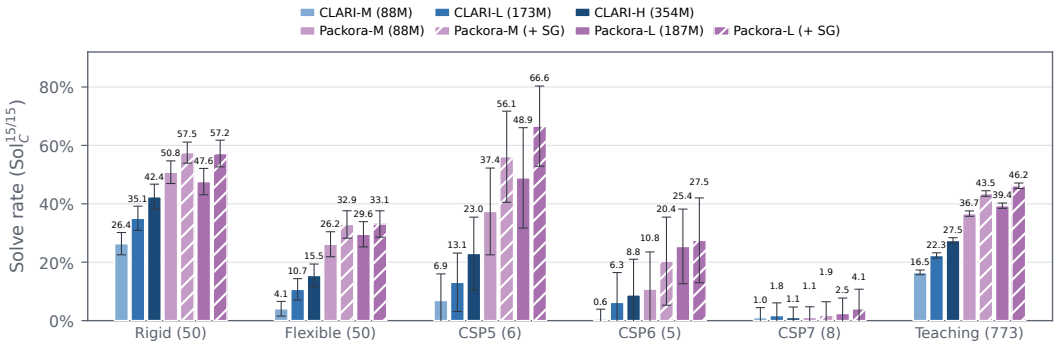


Figure 20: **Strict crystal solve rate with space-group conditioning at 30 candidates per target.** We report $\mathrm{Sol}_C^{15/15}$, requiring all 15 molecules to match under COMPACK, with parentheses indicating benchmark sizes. Solid bars reproduce Figure 4; hatched bars use the target space-group label in addition to templates and stereochemistry. Bars and error bars show means and standard deviations over 5,000 bootstrap resamples of 30 from 1,000 candidates per target.

L STRUCTURE RANKING RESULTS

We provide the evaluation protocol, summary metrics, and per-form results for the FastCSP structure-ranking benchmarks in Section 4.3.

L.1 EVALUATION PROTOCOL

Benchmark exclusions. We exclude BEDMIG, UNOGIN, BISMEV, and BIYSEH from the multi-polymorph benchmark because their CSD families appear in the CLARI training split.

Relaxation and filtering. Each model generates 1,000 candidates per CSD entry. We relax candidates using UMA-S-1.2 with the OMC task (Wood et al., 2025; Gharakhanyan et al., 2026) and ASE BFGS (Larsen et al., 2017), with a maximum-force threshold of 0.02 eV/Å and at most 500 optimization steps. We discard candidates that fail to converge, change molecular connectivity or Z, or have densities outside 0.5–3.0 g/cm^3. We retain structures within 10 kJ/mol of the minimum energy and deduplicate equivalent relaxed structures. For each multi-polymorph system, candidates from all constituent CSD entries are pooled before energy filtering, deduplication, and ranking, giving all experimental forms a shared energy ordering.

Lattice energy. We rank the remaining unique candidates by increasing lattice energy (Gharakhanyan et al., 2025). For candidate $\mathcal{S}_i = (\mathbf{A}_i, \mathbf{X}_i, \mathbf{L}_i)$, the lattice energy per formula unit is $E_{\text{latt}}(\mathcal{S}_i) = E_{\text{UMA}}(\mathcal{S}_i)/Z_i - E_{\text{mol}}$, where Z_i is the number of formula units in the cell and E_{mol} is the isolated-component reference energy for one formula unit.

FastCSP candidate budget. FastCSP's random structure generator uses approximately 53,000–300,000 raw structures per target, with 10,000–176,000 retained for relaxation after deduplication (Gharakhanyan et al., 2025), compared with 1,000 candidates per CSD entry in our evaluation. We therefore exclude FastCSP as a baseline because of its substantially larger search budget.

L.2 RECOVERY AND RANKING RESULTS

As in the seventh CSP blind test (Hunnisett et al., 2024a), we identify matches to experimental forms and report their ranks. Recovered and Top-k report the fractions of all experimental forms matched at any rank and within the first k ranks, respectively. Mean rank averages the best matching rank over recovered forms only; unrecovered forms have no matching rank. Since recovered subsets can differ across methods, interpret mean rank alongside recovery and Top-k. Metrics cover 26 single-polymorph targets or 29 forms across eight multi-polymorph systems.

Tables 12 and 13 summarize recovery and ranking for the single- and multi-polymorph benchmarks, respectively. Tables 14 and 15 report the corresponding per-form energy ranks and RMSD_{30} values of the best matching candidates; NR denotes no recovered match.

Table 12: **FastCSP single-polymorph recovery and ranking.** Results cover 26 targets with 1,000 candidates per target. **Bold** marks the best result.

Method	Recovered ↑	Top-1 ↑	Top-5 ↑	Top-20 ↑	Mean rank ↓
CLARI-H	16/26	9/26	13/26	15/26	4.25
PACKORA-M	17/26	10/26	**17/26**	17/26	**2.00**
PACKORA-L	**19/26**	**13/26**	16/26	**19/26**	2.05

Table 13: **FastCSP multi-polymorph recovery and ranking.** Results cover 29 experimental forms across eight systems, with 1,000 candidates per CSD entry and a shared energy ordering within each system. **Bold** marks the best result.

Method	Recovered ↑	Top-1 ↑	Top-5 ↑	Top-20 ↑	Mean rank ↓
CLARI-H	18/29	5/29	9/29	13/29	102.17
PACKORA-M	19/29	**6/29**	11/29	15/29	**47.11**
PACKORA-L	**21/29**	5/29	**13/29**	**17/29**	54.62

Table 14: **Complete FastCSP single-polymorph results.** Each method reports the energy rank and RMSD_{30} (Å) of the best candidate matching the experimental form. NR denotes no recovery, with the corresponding RMSD_{30} unavailable.

		CLARI-H		PACKORA-M		PACKORA-L	
Target	Subset	Rank	RMSD_{30}	Rank	RMSD_{30}	Rank	RMSD_{30}
Target II	Semi-rigid	5	0.55	NR	–	6	0.58
Target IV	Semi-rigid	1	0.16	1	0.14	1	0.18
Target V	Semi-rigid	8	0.33	5	0.18	7	0.12
Target VIII	Semi-rigid	NR	–	NR	–	1	0.22
Target XII	Semi-rigid	1	0.25	1	0.25	1	0.28
Target XIII	Semi-rigid	1	0.09	1	0.10	1	0.09
Target XVI	Semi-rigid	NR	–	NR	–	NR	–
Target XVII	Semi-rigid	1	0.12	1	0.14	1	0.18
Target XXII	Semi-rigid	1	0.17	2	0.19	2	0.25
Acetic Acid	Semi-rigid	NR	–	NR	–	NR	–
Caprylolactam	Semi-rigid	NR	–	2	0.15	3	0.11
CEBYUD	Semi-rigid	1	0.14	1	0.14	1	0.14
CUMJOJ	Semi-rigid	NR	–	NR	–	NR	–
DEZDUH	Semi-rigid	5	0.44	1	0.06	1	0.04
Eniluracil	Semi-rigid	5	0.63	1	0.11	1	0.10
GACGAU	Semi-rigid	2	0.12	2	0.12	2	0.17
GOLHIB	Semi-rigid	NR	–	1	0.10	1	0.09
HURYUQ	Semi-rigid	27	0.13	5	0.09	1	0.15
IHEPUG	Semi-rigid	NR	–	NR	–	NR	–
LECZOL	Semi-rigid	NR	–	NR	–	NR	–
ROHBUL	Semi-rigid	1	0.08	1	0.05	1	0.04
6-Fluorochromone	Semi-rigid	7	0.16	4	0.72	6	0.25
WEXREY	Semi-rigid	1	0.45	1	0.19	1	0.45
Target X	Flexible	NR	–	4	0.16	NR	–
Target XIV	Flexible	1	0.38	NR	–	1	0.18
Target XVIII	Flexible	NR	–	NR	–	NR	–

Table 15: **Complete FastCSP multi-polymorph results.** Each method reports the energy rank and RMSD_{30} (Å) of the best candidate matching each experimental form. NR denotes no recovery, with the corresponding RMSD_{30} unavailable.

			CLARI-H		PACKORA-M		PACKORA-L	
System	Refcode	Form	Rank	RMSD_{30}	Rank	RMSD_{30}	Rank	RMSD_{30}
Target I	XULDUD01	Monoclinic	NR	–	1	0.297	1	0.267
Target I	XULDUD	Orthorhombic	18	0.308	NR	–	18	0.270
CILJIQ	CILJIQ	Form I	1	0.200	1	0.252	1	0.256
CILJIQ	CILJIQ01	Form II	13	0.165	11	0.118	11	0.117
Glycine	GLYCIN16	gamma	6	0.191	5	0.183	5	0.187
Glycine	GLYCIN20	alpha	1	0.128	1	0.115	1	0.121
Glycine	GLYCIN32	beta	3	0.177	7	0.191	4	0.194
Glycine	GLYCIN67	delta	2	0.279	2	0.280	2	0.281
Glycine	GLYCIN68	epsilon	NR	–	18	0.393	14	0.379
Imidazole	IMAZOL06	alpha	1	0.220	1	0.174	1	0.284
Imidazole	IMAZOL25	beta	NR	–	NR	–	NR	–
Nicotinamide	NICOAM03	alpha	NR	–	2	0.894	4	0.842
Nicotinamide	NICOAM07	epsilon	2	0.234	NR	–	NR	–
Nicotinamide	NICOAM17	iota	10	0.781	4	0.449	6	0.585
Target VI	UJIRIO	Form I	1	0.147	1	0.224	1	0.185
Target VI	UJIRIO05	Form III	NR	–	NR	–	NR	–
Target XXXI	ZEHFUR	Form B	NR	–	9	0.374	NR	–
Target XXXI	ZEHFUR02	Form A major	NR	–	NR	–	4	0.204
Target XXXI	ZEHFUR02	Form A minor	NR	–	NR	–	4	0.204
ROY	QAXMEH01	Y	NR	–	NR	–	NR	–
ROY	QAXMEH12	YT04	NR	–	NR	–	NR	–
ROY	QAXMEH81	R	2	0.446	3	0.473	5	0.565
ROY	QAXMEH03	OP	188	0.121	195	0.247	NR	–
ROY	QAXMEH32	ON	131	0.597	187	0.605	153	0.570
ROY	QAXMEH04	YN	883	0.555	NR	–	NR	–
ROY	QAXMEH05	ORP	1	0.354	1	0.154	2	0.120
ROY	QAXMEH59	PO13	108	0.211	96	0.176	106	0.165
ROY	QAXMEH53	Y04	NR	–	NR	–	325	0.423
ROY	QAXMEH60	Y19	468	0.203	350	0.134	479	0.190

M Ablation results

We report parameter counts, GFLOPs, and validation results for the architecture variants and width-scaling configurations evaluated in Sections 5.2 and 5.6, along with auxiliary training-loss definitions for Section 5.3 and additional inference results for Section 5.5.

M.1 Architecture

We report the parameter count, GFLOPs, and best validation Sol_C for each architecture variant.

Table 16: **Architecture ablation with model size and compute.** We report parameter count and GFLOPs for each architecture variant, together with its best validation Sol_C under the shared ≤ 300-atom, 30-candidate protocol with 200-step EDM–Heun sampling. Config. (F) is the selected cacheable post-entry design.

Config.	Entry	TriMul	TriAtt	SU	GEM	$\mathrm{Sol}_C \uparrow$	Params (M)	GFLOPs
(A)	–	–	–	–	–	0.421	85.98	41.2
(B)	–	–	–	–	✓	0.432	86.01	45.7
(C)	pre	✓	–	–	–	0.479	87.56	379.6
(D)	pre	✓	–	✓	–	0.469	105.40	391.8
(E)	pre	✓	–	–	✓	0.476	87.59	384.2
(F)	post	✓	–	–	–	0.474	87.56	379.6
(G)	post	✓	–	✓	–	0.445	105.40	391.8
(H)	post	✓	–	–	✓	0.474	87.59	384.2
(I)	post	✓	✓	✓	–	0.462	106.07	621.1

M.2 Training

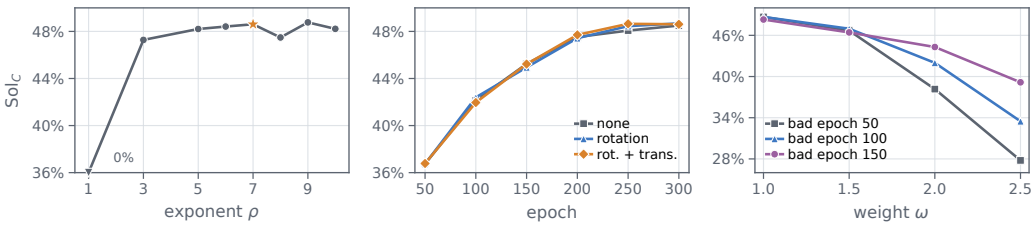
Auxiliary pair losses. For the training ablations in Section 5.3, we test auxiliary loss on periodic pair distances. Let $\mathcal{P} = \{(i,j) : i \neq j,\ d_{ij} < 15\,\text{Å}\}$, where d_{ij} and $\hat{d}_{ij}$ denote target and predicted minimum-image distances. We consider an L^1 loss following CLARI (Lo et al., 2026) and a periodic adaptation of AlphaFold 3 smooth-LDDT (Abramson et al., 2024):

$$\mathcal{L}^{L^1} = \frac{1}{|\mathcal{P}|} \sum_{(i,j)\in\mathcal{P}} \left|\hat{d}_{ij} - d_{ij}\right|, \qquad \mathcal{L}^{\text{smooth}} = 1 - \frac{1}{4|\mathcal{P}|} \sum_{(i,j)\in\mathcal{P}} \sum_{\tau\in\{0.5,1,2,4\}} \sigma\Big(\tau - \left|\hat{d}_{ij} - d_{ij}\right|\Big),$$

where σ is the sigmoid function and distances are in angstroms.

M.3 Inference

Figure 21 reports discretization, template-augmentation, and autoguidance sweeps supporting the inference choices in Section 5.5.



(a) EDM ρ sweep (b) Template aug. (c) Autoguidance

Figure 21: **Additional inference ablations.** Validation Sol_C for **(a)** EDM–Heun discretization exponent ρ at 400 NFE (failed $\rho = 1$ at the lower boundary; star: selected $\rho = 7$); **(b)** template augmentation during inference; and **(c)** autoguidance using checkpoints from epochs 50, 100, and 150. Reduced template augmentation and autoguidance provide no improvement.

M.4 SCALING

Table 17 reports model configurations, parameter counts, GFLOPs, and validation Sol_S and Sol_C at epoch 300 for each width-scaling variant; Figure 22 shows the corresponding epoch sweeps.

Table 17: **Width-scaling ablation.** All rows keep depth, Pairmixer placement, conditioning, validation split, and sampler fixed while varying the single width d_s and pair width d_p. Sol_S and Sol_C are evaluated at epoch 300 with 30 candidates per target and 200-step EDM–Heun sampling (400 NFE). GFLOPs denote analytical denoiser-forward costs at $N{=}300$. Teal rows mark the selected width configurations: W0 defines PACKORA-M, while W4 defines the width of PACKORA-L before applying weight decay as selected in Figure 9c.

ID	d_s	d_p	Heads	Params (M)	GFLOPs	$\mathrm{Sol}_S \uparrow$	$\mathrm{Sol}_C \uparrow$
W0	512	128	8	87.56	379.6	0.088	0.486
W1	512	192	8	89.65	762.6	0.084	0.490
W2	512	256	8	92.54	1287.2	0.087	0.480
W3	768	128	12	184.79	421.0	0.100	0.462
W4	768	192	12	186.91	804.7	0.100	0.487
W5	1024	128	16	330.79	483.8	0.111	0.454

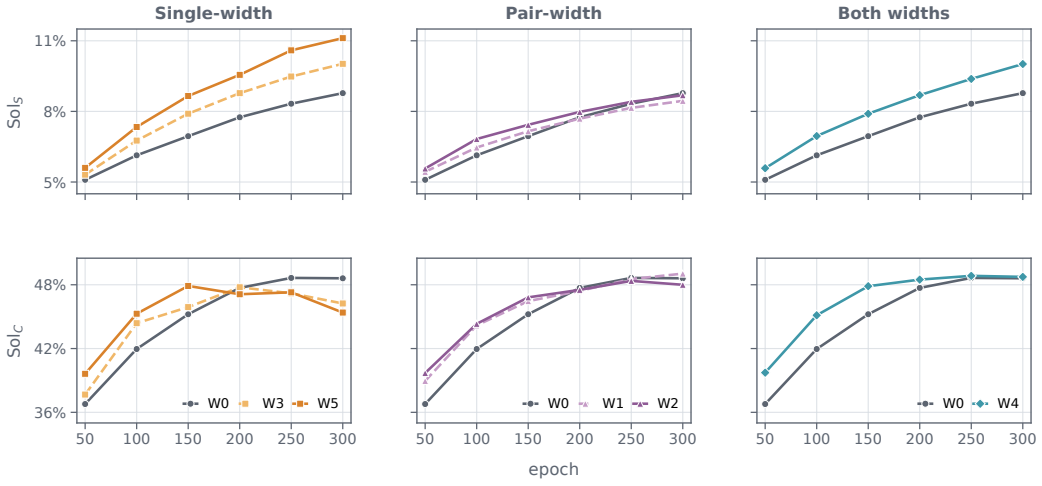


Figure 22: **Width-scaling epoch sweeps.** Columns compare single-width (W0, W3, W5), pair-width (W0, W1, W2), and balanced scaling (W0, W4). Top and bottom rows report Sol_S and 30-candidate Sol_C, respectively, using 200-step EDM–Heun sampling. Scaling either width alone yields no consistent improvement across both metrics.

N Ablation dataset preprocessing

This section describes the dataset preprocessing used for our ablation studies in Section 5. Because CLARI's public split refcodes were not available at the time, we constructed the dataset independently. The preprocessing consists of four stages: filtering, structure processing, benchmark exclusion, and deduplication. Table 18 reports the number of structures remaining after each stage.

Filtering. Following OXtal (Jin et al., 2025), we use CSD entries deposited by May 1, 2025 and retain structures with three-dimensional coordinates, an R-factor of at most 9%, single-crystal X-ray diffraction at ambient pressure, a non-polymeric structure, and a known space group. We Niggli-reduce each crystal, add missing hydrogens with the CSD Python API (Sykes et al., 2024), and apply the 512-atom limit.

Structure processing. We infer missing bond types and standardize aromatic and delocalized bonds with the CSD Python API (Sykes et al., 2024). For each crystal, we store Cartesian coordinates, the unit cell, atomic numbers, bonds and bond types, formal charges, stereochemical annotations, and the space-group label. We generate an independent 3D conformer for each molecular component using RDKit ETKDGv3 (Landrum et al., 2006; Wang et al., 2020). If conformer generation fails, we retain the crystal and mark the template as unavailable.

Benchmark exclusion. To prevent leakage, we exclude structures overlapping with the OXtal Rigid and Flexible benchmarks (Jin et al., 2025), CSP5–7 (Bardwell et al., 2011; Reilly et al., 2016; Hunnisett et al., 2024b), and the CSD Teaching Subset (Battle et al., 2010; Lo et al., 2026). We first remove entries whose six-letter CSD refcode family matches a benchmark family. We then remove entries containing a molecular component found in the benchmark set, using the CSD-provided component SMILES. Following CLARI (Lo et al., 2026), we exclude recognized solvents and components with fewer than eight heavy atoms from this overlap check.

Deduplication. We group the remaining entries by six-letter CSD refcode family and cluster equivalent structures using pymatgen StructureMatcher (Ong et al., 2013) with `ltol`$= 0.2$, `stol`$= 0.3$, and `angle_tol`$= 5^\circ$. Within each cluster, we retain the entry with the lowest reported R-factor.

Table 18: **Ablation dataset preprocessing counts.** We report the number of structures remaining after each stage.

Stage	# Structures
Raw CSD entries	1,451,367
Filtering and structure preprocessing	833,033
Benchmark-family exclusion	831,018
Benchmark-component exclusion	798,459
Deduplication	766,391
Training split	751,023
Validation split	15,368