AtomMOF: All-Atom Flow Matching for MOF-Adsorbate Structure Prediction

Nayoung Kim Honghui Kim Sihyun Yu Minkyu Kim Seongsu Kim Sungsoo Ahn
Korea Advanced Institute of Science and Technology (KAIST)
{nayoungkim, ildmb96, sihyun.yu, minkyu-kim, seongsu.kim, sungsoo.ahn}@kaist.ac.kr

Abstract

Metal-organic frameworks (MOFs) are promising materials for applications such as direct air capture, where performance depends on how adsorbates bind within the framework. Accurately predicting MOF-adsorbate configurations is therefore important for screening candidate materials. However, traditional approaches are computationally expensive and require a known host structure, while existing generative models rely on rigid-body assumptions and do not explicitly model adsorbates. We introduce AtomMOF, a scalable all-atom flow-matching model that jointly predicts MOF-adsorbate structures from discrete building blocks and adsorbates. Built on a Diffusion Transformer with a building block-based pairwise attention bias, AtomMOF operates in an unconstrained all-atom space and exhibits clear scaling behavior. To improve the structural validity of flexible all-atom models, we also propose Feynman-Kac (FK) steering with machine-learned interatomic potentials (MLIPs). On the BW dataset, AtomMOF achieves a 85.61% relative increase in match rate and a 34.32% relative reduction in RMSD compared with prior work. MLIP-guided steering further improves validity by 28.7% and reduces formation energy error by 86.5%. On ODAC25, AtomMOF generates adsorption configurations faster than GCMC and, when combined with MLIP relaxation, identifies lower-energy configurations than those in the reference dataset.

Code: https://github.com/nayoung10/AtomMOF

1 Introduction

Metal-organic frameworks (MOFs) are porous crystalline materials with large surface areas and highly tunable chemistries. These features make them promising for applications such as gas separation [1], catalysis [2], and sensing [3]. One important application is direct air capture (DAC), which requires selectively adsorbing CO_2 from ambient air. This task is inherently challenging because atmospheric CO_2 is extremely dilute, while competing species such as H_2O can occupy adsorption sites [4]. Because adsorption performance depends on where adsorbates bind within the framework, accurately predicting MOF-adsorbate configurations is essential for discovering effective carbon capture materials.

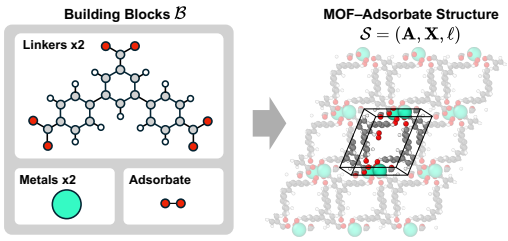


Figure 1: **MOF-adsorbate structure prediction.** Given building blocks $\mathcal{B}$, including linkers, metal nodes, and adsorbate, the goal is to predict the equilibrium structure $\mathcal{S} = (\mathbf{A}, \mathbf{X}, \boldsymbol{\ell})$ with atomic numbers $\mathbf{A}$, coordinates $\mathbf{X}$, and lattice $\boldsymbol{\ell}$.

Traditional approaches for predicting MOF-adsorbate binding configurations rely on search-based, iterative workflows. They first generate candidate adsorbate placements using grand canonical Monte Carlo [GCMC; 5], heuristics, or random sampling [6], and then relax each candidate to a local energy

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

minimum via iterative energy optimization. Although effective, these workflows are computationally expensive for MOFs, which often have hundreds of atoms per unit cell. Moreover, they require a known 3D host structure, limiting screening to existing MOF databases despite the vast space of hypothetical materials.

To overcome these challenges, deep generative models have emerged as a promising alternative. However, existing models have two key limitations. First, they often operate at a coarse-grained resolution with rigid-body assumptions [7–9] or fixed bond lengths and angles [10]. Although this simplifies the modeling task, it fails to capture variations in bond lengths, bond angles, and coordination geometries arising from local chemical environments and MOF-adsorbate interactions. Second, these methods generate the MOF structure in isolation, neglecting the adsorbates that ultimately govern performance in applications such as DAC. Because MOF-adsorbate configurations determine adsorption energies, directly predicting them accelerates materials screening.

Contribution. We introduce AtomMOF, a scalable flow-matching model for all-atom MOF-adsorbate structure prediction. Given only discrete building block identities and adsorbate types, AtomMOF directly predicts low-energy MOF-adsorbate configurations (Figure 1). Built on a unified Diffusion Transformer [DiT; 11], AtomMOF augments attention with a pairwise bias that encodes local geometry and connectivity within each building block. This architecture supports fully unconstrained all-atom generation while jointly modeling frameworks and adsorbates.

This formulation makes structure prediction more useful for MOF screening. Unlike existing MOF generative models that are structurally constrained, AtomMOF predicts all atoms in an unconstrained coordinate space and can therefore capture fine-grained variations in bond lengths, bond angles, and coordination environments. Moreover, because it predicts the MOF and adsorbate jointly, the resulting structures can be evaluated with application-relevant properties such as adsorption energy.

However, this flexibility introduces an additional challenge: unconstrained all-atom models can produce local geometric distortions, such as deformed aromatic rings [12]. To address this issue, we introduce Feynman-Kac steering with machine-learning interatomic potentials (MLIPs), a simple inference-time strategy that improves structural quality and stability without retraining. Unlike prior steering approaches that rely on hand-designed energy functions [13] or expensive multi-stage simulation pipelines [14], our method directly uses off-the-shelf MLIPs, making it easy to apply.

Our contributions are summarized as follows:

- **All-atom MOF generative framework:** We present an all-atom model for MOF structure prediction. By relaxing rigid-body assumptions, AtomMOF yields a 85.61% relative increase in match rate and a 34.32% relative reduction in RMSD on the BW dataset [15] compared to the best prior method. We also observe scaling behavior suggesting further gains with scale.
- **Feynman-Kac steering with MLIPs:** We introduce MLIP-guided Feynman-Kac steering to improve structural quality and stability in unconstrained all-atom generation. It improves MOF validity by 10.9–28.7%, structural validity by 12.2–17.3%, and reduces formation energy error by 62.7–86.5% across MLIP choices.
- **MOF-adsorbate structure prediction:** AtomMOF is the first model to jointly predict MOF-adsorbate structures. On ODAC25 [16], it generates diverse adsorption configurations with much lower wall-clock time than GCMC. When combined with MLIP relaxation, it can also find configurations with lower adsorption energies than the reference structures, which is important for accurate adsorption energy estimation [6].

2 Related work

2.1 Machine learning for crystal structure prediction

Most work on crystal structure prediction has focused on inorganic crystals. DiffCSP [17] pioneered this direction by jointly denoising lattice and coordinate information with a periodic E(3)-equivariant architecture. Subsequent works improved symmetry handling: EquiCSP [18] accounts for lattice permutation symmetry, while DiffCSP++ [19] and SymmCD [20] incorporate space-group constraints. Beyond diffusion, other generative frameworks have also been explored: FlowMM [21] uses equivariant flow matching, CrysBFN [22] applies a periodic E(3) Bayesian flow network, and OMatG [23] adapts the stochastic interpolant framework to improve performance further.

In contrast, structure prediction for MOFs is more challenging because of their modular nature and larger system sizes. Existing methods address this complexity with a rigid-body assumption: MOFFlow [7] predicts building-block rotations and translations via Riemannian flow matching, MOF-BFN [8] operates on fractional coordinates to enforce periodicity, and MOF-LLM [9] uses progressive training with a large language model. Although MOFFlow-2 [10] relaxes this assumption by incorporating torsion angles around rotatable bonds, it still keeps local geometries (bond lengths and angles) generated by cheminformatics tools fixed. In this work, we introduce an all-atom model that relaxes these constraints to enable fully flexible structure prediction.

2.2 Generative models for molecular complexes

Recent generative models have increasingly addressed interactions within molecular complexes. In protein structure prediction, AlphaFold 1 and 2 [24, 25] focused on isolated proteins, whereas AlphaFold 3 [26] extended prediction to complexes involving proteins, nucleic acids, and small molecules; subsequent models such as Boltz [13, 27], Chai-1 [28], Protenix [29], and SeedFold [30] further advanced complex modeling. Likewise, in heterogeneous catalysis, generative models explicitly capture catalyst-adsorbate interactions [31, 32]. Our approach extends MOF structure prediction in a similar spirit by jointly modeling the full MOF-adsorbate system.

3 Method

3.1 Problem formulation

In this work, we study structure prediction for MOFs. Specifically, we predict the atomic coordinates of an MOF-adsorbate system given the underlying molecular graphs of the building blocks (organic linkers and metal ions) and, optionally, the adsorbates (Figure 1).

Crystal representation. A MOF structure $\mathcal{S}$ is defined by a unit cell on a three-dimensional lattice. We represent a unit cell with N atoms as $\mathcal{S} = (\mathbf{A}, \mathbf{X}, \boldsymbol{\ell})$, where $\mathbf{A} = \{a_i\}_{i=1}^{N} \in \mathcal{A}^N$ denotes the atom types, with $\mathcal{A}$ the set of chemical elements, and $\mathbf{X} = \{x_i\}_{i=1}^{N} \in \mathbb{R}^{N\times 3}$ denotes the atomic coordinates. The lattice is given by a matrix $L = (l_1, l_2, l_3) \in \mathbb{R}^{3\times 3}$ whose columns are the lattice vectors. We parameterize it by $\boldsymbol{\ell} = (\boldsymbol{d}, \boldsymbol{\phi})$, where $\boldsymbol{d} = (a, b, c) \in \mathbb{R}^3_+$ are the edge lengths and $\boldsymbol{\phi} = (\alpha, \beta, \gamma) \in [0°, 180°]^3$ are the inter-edge angles. The corresponding infinite periodic crystal is $\{(a_n, x_n + Lm) \,\big|\, n \in [1, N],\, m \in \mathbb{Z}^3\}$, where $m = (m_1, m_2, m_3)^\top$ indexes lattice translations.

Building block representation. We represent the MOF-adsorbate system as a set of molecular graphs, denoted collectively by $\mathcal{B} = \{\mathcal{G}\}_{i=1}^{M}$, where $\mathcal{G}_i$ denotes the i-th molecular graph with atoms as vertices and bonds as edges. This set includes the MOF building blocks and, when present, the adsorbates; for bare MOFs, the adsorbate set is empty.

3.2 Variational flow matching for MOFs

In this work, we adopt variational flow matching [VFM; 33], which enables training with an L^1 objective and empirically improves convergence. VFM is a generative modeling framework that learns to transform samples from a prior distribution $p_0(\mathbf{z}_0)$ to a target data distribution q by predicting clean data from noisy observations, conditioned on the building blocks $\mathcal{B}$.

Conditional probability path. We apply VFM to jointly model the atomic coordinates $\mathbf{X}$ and lattice parameters $\boldsymbol{\ell}$; for notational simplicity, let $\mathbf{z} = (\mathbf{X}, \boldsymbol{\ell}) \in \mathbb{R}^{N\times 3} \times \mathbb{R}^6$. We define a conditional probability path $p_t(\mathbf{z} \mid \mathbf{z}_1)$ that linearly interpolates between a prior sample $\mathbf{z}_0 \sim p_0$ at $t = 0$ and a data sample $\mathbf{z}_1 \sim q$ at $t = 1$, namely $\mathbf{z}_t = (1 - t)\mathbf{z}_0 + t\mathbf{z}_1$.[1]

Training and inference. Given $\mathbf{z}_t$, VFM learns a variational posterior $q_t^\theta(\mathbf{z}_1 \mid \mathbf{z}_t)$ that approximates the true posterior $p_{1|t}(\mathbf{z}_1 \mid \mathbf{z}_t)$. Training minimizes the expected negative log-likelihood:

$$\mathcal{L}_{\text{VFM}}(\theta) = -\mathbb{E}_{t,\mathbf{z}_t,\mathbf{z}_1}\big[\log q_t^\theta(\mathbf{z}_1 \mid \mathbf{z}_t)\big], \tag{1}$$

where $t \sim \mathcal{U}[0, 1]$, $\mathbf{z}_1 \sim q$, and $\mathbf{z}_t \sim p_t(\mathbf{z} \mid \mathbf{z}_1)$. At inference time, we generate samples by solving the ordinary differential equation (ODE) $\mathrm{d}\boldsymbol{z}_t = \mathbf{v}_t(\mathbf{z}_t)\,\mathrm{d}t$, where the velocity field is given by $\mathbf{v}_t(\mathbf{z}_t) = (\boldsymbol{\mu}_t^\theta(\mathbf{z}_t) - \mathbf{z}_t)/(1 - t)$.

[1] We omit conditioning on building blocks $\mathcal{B}$ for brevity.

Parameterization. We model the variational posterior $q_t^\theta(\mathbf{X}_1, \boldsymbol{\ell}_1 \mid \mathbf{X}_t, \boldsymbol{\ell}_t)$ as a Laplace distribution whose mean $\boldsymbol{\mu}_t^\theta$ is parameterized by the neural network as $\hat{\mathbf{X}}_1, \hat{\boldsymbol{\ell}}_1 = \boldsymbol{\mu}_t^\theta(\mathbf{X}_t, \boldsymbol{\ell}_t \mid \mathcal{B})$, where $\hat{\mathbf{X}}_1$ and $\hat{\boldsymbol{\ell}}_1$ are the predicted clean Cartesian coordinates and lattice parameters, respectively. Under this parameterization, Equation (1) reduces to the L^1 reconstruction loss [34]:

$$\mathcal{L}(\theta) = \mathbb{E}_{\mathcal{S}_1, \mathcal{S}_t, t}\left[\lambda_{\text{coord}} \|\hat{\mathbf{X}}_1 - \mathbf{X}_1\|_1 + \lambda_{\text{lattice}} \|\hat{\boldsymbol{\ell}}_1 - \boldsymbol{\ell}_1\|_1\right], \tag{2}$$

where $\mathcal{S}_1 = (\mathbf{A}, \mathbf{X}_1, \boldsymbol{\ell}_1) \sim q(\mathcal{S})$, $\mathcal{S}_t = (\mathbf{A}, \mathbf{X}_t, \boldsymbol{\ell}_t)$, $t \sim \mathcal{U}[0, 1]$, and λ_{coord} and λ_{lattice} are weights.

Prior distribution. We define prior distributions over the atomic coordinates $\mathbf{X}$ and lattice parameters $\boldsymbol{\ell} = (\boldsymbol{d}, \boldsymbol{\phi})$. For the coordinates, we use a centered Gaussian prior $\mathbf{X}_0 = \mathbf{P}\boldsymbol{\varepsilon}$, where $\boldsymbol{\varepsilon} \sim \mathcal{N}(0, \mathbf{I})$ and $\mathbf{P} = \mathbf{I} - \frac{1}{N}\mathbf{1}\mathbf{1}^\top$ enforces a zero center of mass. For the lattice parameters, the lengths $\boldsymbol{d}$ are sampled independently from log-normal distributions, $\boldsymbol{d} \sim \text{LogNormal}(\boldsymbol{\mu}, \boldsymbol{\sigma})$, where $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are estimated by maximum likelihood. We sample lattice angles $\boldsymbol{\phi}$ uniformly from $[60°, 120°]$ [21].

3.3 Model architecture

The denoising network, $\boldsymbol{\mu}_t^\theta(\boldsymbol{X}_t, \boldsymbol{\ell}_t | \mathcal{B})$, retains a scalable DiT backbone [11, 35] while augmenting attention with a pairwise representation that encodes local distances and connectivity within MOF building blocks, inspired by AlphaFold 3 [26]. The network comprises three stages: an *encoder* that constructs features from the building blocks and noisy inputs, a deep *trunk* that models interactions, and a *decoder* that predicts $\boldsymbol{X}_1$ and $\boldsymbol{\ell}_1$. See Figure 10 for an overview and Appendix A for pseudocode.

Encoder. The encoder initializes single-atom ($\boldsymbol{Q} \in \mathbb{R}^{N \times D_Q}$) and pairwise ($\boldsymbol{P} \in \mathbb{R}^{N \times N \times D_P}$) representations from the building blocks $\mathcal{B}$ and the noisy inputs $\boldsymbol{X}_t$ and $\boldsymbol{\ell}_t$. First, we construct initial single features $\boldsymbol{C}$ from atom types, formal charges, and local reference coordinates $\boldsymbol{X}_{\text{local}}$ derived from RDKit [36]. In parallel, we initialize $\boldsymbol{P}$ by featurizing the internal geometry of the building blocks. Specifically, we compute local displacement vectors $\boldsymbol{D}_{ij} = \boldsymbol{X}_{\text{local},i} - \boldsymbol{X}_{\text{local},j}$ and Euclidean distances $\|\boldsymbol{D}_{ij}\|$, and embed them together with bond types and block-adjacency masks indicating whether two atoms belong to the same building block. We then condition $\boldsymbol{P}$ on $\boldsymbol{C}$ using an MLP, while forming $\boldsymbol{Q}$ by adding projections of the noisy global coordinates $\boldsymbol{X}_t$ and lattice parameters $\boldsymbol{\ell}_t$ to $\boldsymbol{C}$. Finally, these features are processed by a shallow $\text{DiT}_{\text{encoder}}$, where $\boldsymbol{P}$ acts as an additive bias on the attention logits (see Algorithm 6 in Appendix A).

Trunk. The trunk performs the main interaction modeling through a deep stack of DiT blocks. We first project the encoder outputs into higher-dimensional representations $\boldsymbol{S}$ and $\boldsymbol{Z}$ using an MLP. Before the transformer pass, we update the pairwise bias $\boldsymbol{Z}$ with single-atom context via $\boldsymbol{Z}_{ij} \leftarrow \boldsymbol{Z}_{ij} + \text{Linear}(\boldsymbol{S}_i) + \text{Linear}(\boldsymbol{S}_j)$. The updated $\boldsymbol{Z}$ then serves as the structural attention bias throughout the trunk. After the trunk computation, we apply a global residual update to the encoder features: $\boldsymbol{Q} \leftarrow \boldsymbol{Q} + \text{Linear}(\boldsymbol{S}')$.

Decoder. The decoder maps the updated features $\boldsymbol{Q}$ to the denoised state $\hat{\boldsymbol{X}}_1$ and $\hat{\boldsymbol{\ell}}_1$. After a $\text{DiT}_{\text{decoder}}$, we use two parallel heads, each consisting of a LayerNorm and a linear layer, to generate the outputs. The coordinate head projects per-atom features to $\hat{\boldsymbol{X}}_1 \in \mathbb{R}^{N \times 3}$, while the lattice head applies global average pooling followed by a projection to $\hat{\boldsymbol{\ell}}_1 \in \mathbb{R}^6$.

3.4 Feynman-Kac steering with MLIP

While all-atom generative models offer high flexibility, they can also produce structural distortions and incorrect bond lengths [12]. To mitigate this issue, we introduce Feynman-Kac steering guided by a machine learning interatomic potential (MLIP), which improves the structural fidelity and stability of generated molecules at inference time. The advantage of this approach is simplicity: off-the-shelf MLIPs greatly improve structural quality without hand-designed potential functions [13] or expensive domain-specific simulations [14].

MLIP-guided Feynman-Kac steering. Intuitively, we bias toward structures that are energetically favorable under the MLIP. Concretely, we sample from an exponentially tilted distribution:

$$p_{\text{target}}(\boldsymbol{X}, \boldsymbol{\ell} \mid \mathcal{B}) \propto p_\theta(\boldsymbol{X}, \boldsymbol{\ell} \mid \mathcal{B}) \exp\left(-\lambda E(\mathcal{S})\right), \tag{3}$$

where $E(\cdot)$ is the energy estimated by the MLIP and λ controls the steering strength. In Feynman-Kac steering, a *potential* is an intermediate scoring function used to reweight and resample particles at

noisy states along the trajectory; larger values indicate particles that are more likely to yield desirable final samples [37, 14]. Different choices of potential correspond to different ways of converting the MLIP signal into intermediate resampling scores. In our experiments, we use the immediate potential, which depends only on the current timestep, unless noted otherwise. Definitions of the different potentials and the full algorithm are provided in Appendix B.

SDE reformulation. To enable particle resampling for Feynman-Kac steering, we convert the ODE induced by the learned velocity field $\mathbf{v}(\boldsymbol{X}_t, t)$ into the SDE

$$\mathrm{d}\boldsymbol{X}_t = \Big(\mathbf{v}(\boldsymbol{X}_t, t) + \tfrac{1}{2} g(t)^2\, s(\boldsymbol{X}_t, t)\Big)\mathrm{d}t + g(t)\,\mathrm{d}\boldsymbol{W}_t, \tag{4}$$

where $\boldsymbol{W}_t$ is a standard Wiener process, $g(t)$ is the diffusion coefficient [38], and $s(\boldsymbol{X}_t, t)$ is the score function. For a centered Gaussian prior $\boldsymbol{X}_0 \sim \mathcal{N}_C(\mathbf{0}, \boldsymbol{I})$, the score is given directly by $\mathbf{v}$: $s(\boldsymbol{X}_t, t) = (t\,\mathbf{v}(\boldsymbol{X}_t, t) - \boldsymbol{X}_t)/(1-t)$.

4 MOF-only structure prediction

4.1 Experimental setup

Data preprocessing. We apply a unified preprocessing pipeline to both datasets. We first use MOFid [39] to decompose each MOF into nodes, linkers, solvents, and adsorbates. We then validate the organic linkers with RDKit and adjust atomic coordinates to ensure each building block is spatially contiguous. Details for constructing node templates are provided in Appendix C.

Training and hyperparameters. We train all variants with a three-stage curriculum that progressively increases system size. The first two stages restrict the training set by maximum atom count, and the final stage fine-tunes on the full dataset. We train three model sizes: AtomMOF-S (38.4M parameters), AtomMOF-M (129M), and AtomMOF-L (491M). Full training details are provided in Appendix C, and architecture, curriculum, and deterministic prediction comparisons in Appendix D.

4.2 Match rate & RMSD

Overview. We evaluate our model on the BW dataset for MOF structure prediction from the 2D molecular graphs of building blocks $\mathcal{B}$. We use the 8:1:1 split of Kim et al. [10]. At inference time, we generate samples with an ODE solver using 10 integration steps and report results for 1 and 5 samples, evaluating the best-matching candidate for both cases.

Metrics. We assess performance using match rate (MR) and root mean squared deviation (RMSD), computed with `pymatgen`'s `StructureMatcher`. The matcher compares structures based on specified tolerances for site positions (stol), fractional lengths (ltol), and angles (angle_tol), returning 1 if the structures match within these limits and 0 otherwise. RMSD is computed only for successfully matched pairs. We use stol values of 0.5 (a typical value for CSP tasks) and 0.3 (the stricter default), ltol $= 0.3$, and angle_tol $= 10$.

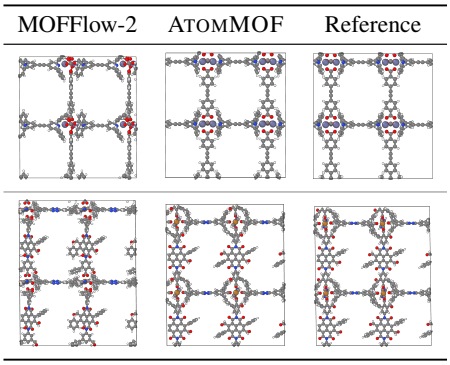


Figure 2: **Visualizations.** Structures in $2 \times 2 \times 2$ supercells. AtomMOF-M better matches the ground truth and reduces atomic clashes near metal nodes.

Baselines. We compare our method against both optimization- and learning-based baselines. The optimization-based baselines include random search (RS) and evolutionary algorithms (EA), which iteratively propose candidates until an energetically favorable structure is found (implemented with `CrySPY` [40]). We also include DiffCSP, a general CSP baseline for inorganic materials, and MOF-specific baselines, namely MOFFlow [7], MOFFlow-2 [10], and MOF-BFN [8].

The original MOFFlow and MOF-BFN implementations assume access to ground-truth local coordinates, which is unrealistic in practice. For a fair comparison, we retrain both models following Kim et al. [10]. During training, we use matched coordinates obtained by aligning nodes and RDKit linkers to the ground truth; at inference time, we use node templates and direct RDKit predictions.

Table 1: **Comparison of structure prediction accuracy.** MR is the match rate, and RMSD is the root mean squared deviation. stol is the site-tolerance, where smaller values impose stricter matching. Param. is the number of parameters. **Bold** indicates the best result and underlined indicates the second-best. Random sampling and evolutionary algorithm baselines achieve 0% match rate at all tolerances with 20 samples and are omitted. AtomMOF achieves best performance, with larger gains under stricter tolerance, highlighting the importance of all-atom modeling.

		1 Sample				5 Samples			
		stol = 0.3		stol = 0.5		stol = 0.3		stol = 0.5	
Model	Param.	MR (%) ↑	RMSD ↓	MR (%) ↑	RMSD ↓	MR (%) ↑	RMSD ↓	MR (%) ↑	RMSD ↓
DiffCSP	12.3M	0.01	0.1554	0.23	0.3896	0.08	0.1299	0.87	0.3982
MOFFlow	22.5M	5.28	0.2036	21.93	0.3329	8.68	0.2039	32.71	0.3290
MOF-BFN	11.2M	6.96	0.1979	27.66	0.3234	12.11	0.1950	40.15	0.3131
MOFFlow-2	146M	8.20	0.1894	28.71	0.3094	15.98	0.1842	43.95	0.2925
AtomMOF-M	129M	14.33	0.1308	37.88	0.2542	26.67	0.1218	59.77	0.2329
AtomMOF-L	491M	**15.22**	**0.1244**	**39.82**	**0.2498**	**28.97**	**0.1168**	**62.04**	**0.2243**

Results. As shown in Table 1, AtomMOF consistently outperforms all baselines across every metric. Although MOFFlow and MOF-BFN are smaller, they predict only rigid-body transformations of building blocks rather than all-atom coordinates, which limits flexibility and reconstruction accuracy. In the 1-sample case under the stricter setting (stol $= 0.3$), at a comparable parameter count, AtomMOF-M (129M) yields a 74.76% relative increase in match rate and a 30.94% relative reduction in RMSD relative to MOFFlow-2 (146M). AtomMOF-L performs even better, yielding a 85.61% increase in match rate and a 34.32% reduction in RMSD. This advantage is evident in Figure 2, where MOFFlow-2 exhibits steric clashes around metal nodes, whereas AtomMOF faithfully reconstructs the ground-truth structures.

4.3 Property evaluation

Overview. We evaluate whether our model preserves key structural properties relevant to downstream applications such as gas storage and catalysis. For each generated structure, we measure volumetric surface area (VSA; $\mathrm{m}^2/\mathrm{cm}^3$), gravimetric surface area (GSA; m^2/g), largest cavity diameter (LCD; Å), pore limiting diameter (PLD; Å), accessible volume (AV; Å^3), unit-cell volume (UCV; Å^3), density (DST; g/cm^3), and void fraction (VF).

Metrics & baselines. We compare against MOFFlow-2 using its default sampling configuration of 50 ODE steps, while AtomMOF uses 10 integration steps. All properties are computed with Zeo++ [41]. We report the RMSE between predicted and ground-truth properties.

Table 2: **Property evaluation.** RMSE between ground-truth and generated MOFs for key properties computed via Zeo++. AtomMOF shows lower errors than MOFFlow-2, highlighting the benefit of all-atom modeling.

	VSA ↓	GSA ↓	AV ↓	UCV ↓
MOFFlow-2	312.8	443.6	485.0	427.1
AtomMOF-M	216.1	252.0	**176.4**	**148.8**
AtomMOF-L	**202.0**	**232.9**	193.3	167.3
	VF ↓	PLD ↓	LCD ↓	DST ↓
MOFFlow-2	0.0445	1.3751	1.5361	0.0312
AtomMOF-M	0.0214	0.9025	0.9594	0.0170
AtomMOF-L	**0.0197**	**0.8715**	**0.8936**	**0.0169**

Results. As shown in Table 2, both AtomMOF variants achieve lower property errors than MOFFlow-2 across all reported metrics. The largest gains are in accessible volume (AV) and unit-cell volume (UCV): relative to MOFFlow-2, AtomMOF-M reduces RMSE from 485.0 to 176.4 for AV and from 427.1 to 148.8 for UCV. AtomMOF-L performs best on the other six properties, namely VSA, GSA, VF, PLD, LCD, and density. Overall, these results suggest that all-atom modeling preserves structural properties better than torsion-based parameterizations with fixed bond constraints, and that scaling further improves most property estimates.

4.4 Scaling AtomMOF

Parameter scaling. Figure 3 shows consistent performance gains as we scale the model from 38.4M to 491M parameters. For comparison, we also train two MOFFlow-2 variants, a small model with

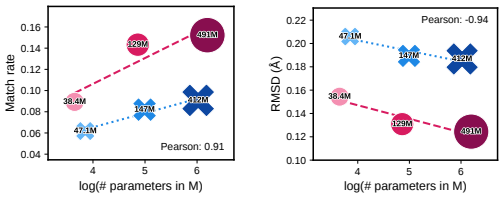


Figure 3: **Scaling behavior of AtomMOF.** Match rate (*left*) and RMSD (*right*) versus model size. AtomMOF achieves a higher match rate and lower RMSD across all model sizes than MOFFlow-2. It also exhibits better scaling behavior, with Pearson correlations near ± 0.9, indicating predictable gains with scale.

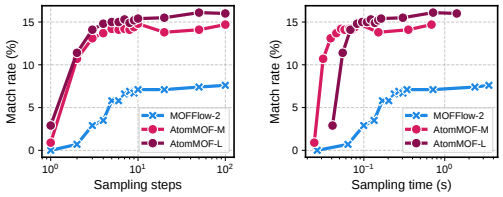


Figure 4: **Inference efficiency under scaling.** Match rate (%) versus (*left*) ODE sampling steps and (*right*) wall-clock time. AtomMOF-L outperforms AtomMOF-M at the same step or time budget, and both surpass MOFFlow-2. AtomMOF exceeds MOFFlow-2's peak match rate within a few steps and saturates by 10 steps.

47.1M parameters and a large model with 412M parameters, to assess scaling relative to AtomMOF. Both match rate and RMSD follow an approximately log-linear trend (Pearson $+0.91/-0.94$) with better scaling behavior than MOFFlow-2. All results use stol $= 0.3$, with 10 integration steps for AtomMOF and 50 for MOFFlow-2. Qualitative improvements with scale are shown in Figure 5.

Runtime scaling. Figure 4 shows how match rate (stol $= 0.3$) varies with the number of sampling steps. We evaluate 1,000 randomly sampled test structures using 1 to 100 ODE integration steps on a single A100 80GB GPU. In Figure 4 (left), AtomMOF-L outperforms AtomMOF-M at every step count, indicating higher sample efficiency per integration step. In terms of wall-clock time (Figure 4, right), AtomMOF-M performs better at very small time budgets, but AtomMOF-L becomes superior once runtime exceeds 0.0825 s. Both variants are far more efficient than MOFFlow-2: AtomMOF exceeds its peak match rate ($\sim$7.6%) in just 3 steps and largely saturates by 10 steps, reaching 15.4% for AtomMOF-L and 14.8% for AtomMOF-M, which supports our default of 10 integration steps.

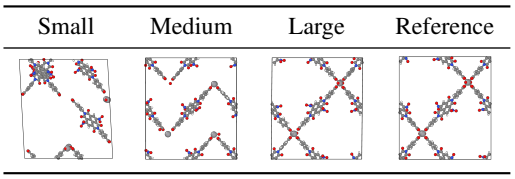


Figure 5: **Effect of model scaling on predicted structures.** From left to right, larger models produce higher-quality structures that more closely match the reference, better recovering the reference topology and reducing steric clashes.

4.5 Feynman-Kac steering with MLIP

FK steering improves stability and validity. We apply Feynman-Kac steering with four MLIPs as guidance potentials: eSEN [16], UMA [42], Orb-v3 [43], and SevenNet-Omni [44]. To avoid bias, we use held-out eSEN to evaluate ΔE_f. We set $\lambda = 2.0$, resampling interval 5, start time $t = 0.8$, and the immediate potential from Table 5, using 16 particles for eSEN and UMA and 8 for Orb and SevenNet due to compute limits. As shown in Table 3, steering improves all metrics across guidance MLIPs: ΔE_f decreases by 62.7–86.5%, `MOFChecker` validity [45] increases by 10.9–28.7%, and structural validity by 12.2–17.3%, showing robustness to the MLIP choice. See Appendix E for settings, metric definitions, compute overhead, and comparison with best-of-N, and Appendix F for extension to inorganic crystals.

Table 3: **Effect of Feynman-Kac steering with different MLIPs.** Steering improves `MOFChecker` validity (`MOF`, %) and structural validity (Struct., %) and reduces formation energy error ΔE_f (eV/atom, by eSEN).

Method	FK	`MOF` ↑	Struct. ↑	ΔE_f ↓
No steering	✗	34.16	81.70	0.126
eSEN [16]	✓	37.87	91.70	0.020
UMA [42]	✓	43.98	94.90	0.017
Orb-v3 [43]	✓	41.47	95.80	0.036
SevenNet [44]	✓	41.52	95.50	0.047

Sensitivity analysis. Figure 6 evaluates steering sensitivity on 1,000 random test samples using UMA guidance and eSEN evaluation. Increasing the number of particles and steering strength (λ) consistently improves both stability and validity. The immediate potential mode performs best, consistent with Hartman et al. [14]. Importantly, applying steering later in the denoising trajectory

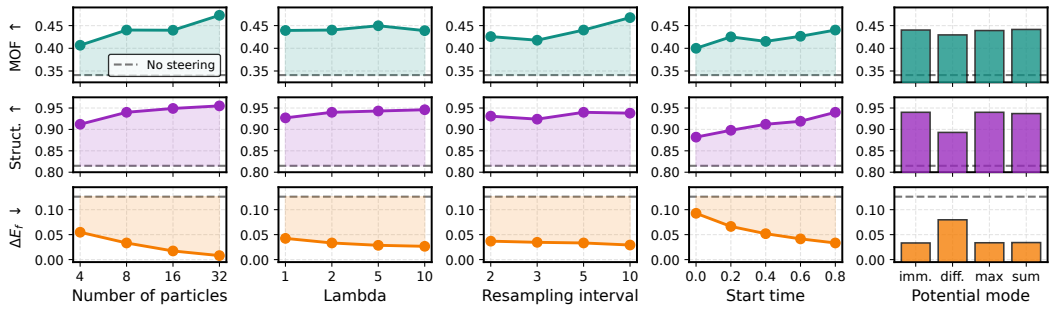


Figure 6: **Sensitivity analysis of MLIP-guided Feynman-Kac steering.** Using UMA for steering, we ablate particle count, steering strength (λ), resampling interval, start time, and potential mode, and report `MOFChecker` validity (%, *top*), structural validity (%, *middle*), and eSEN formation energy error ΔE_f (eV/atom, *bottom*). Increasing particle count and λ generally improves validity and stability. Notably, later start times perform best, leveraging accurate MLIP predictions as the structures denoise.

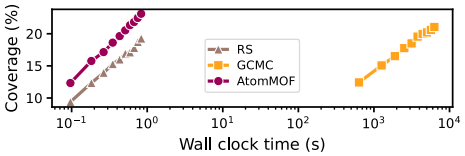


Figure 7: **Coverage vs. wall-clock time.** Coverage (%) of reference adsorption configurations versus wall-clock time (log scale) for AtomMOF, random sampling (RS), and GCMC. AtomMOF attains the highest coverage at the lowest runtime.

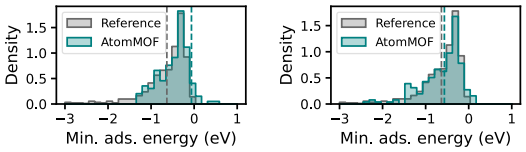


Figure 8: **Adsorption energy histogram.** Histogram of minimum adsorption energies for each MOF-adsorbate system with 1 generated sample (*left*) and 10 generated samples (*right*). Vertical lines indicate the mean. The distribution closely matches the reference.

yields greater improvements, leveraging higher accuracy of MLIPs as structures denoise. Additional details are in Appendix E.

5 MOF-adsorbate structure prediction

We evaluate AtomMOF on ODAC25 through two questions. First, under a fixed computational budget, how well does AtomMOF recover dataset adsorption configurations (Section 5.1)? Second, combied with MLIP relaxation, can it identify lower-energy configurations than those in the reference dataset (Section 5.2)? We use AtomMOF-L throughout, as it performs best in Section 4.

Data preparation & hyperparameters. We apply the preprocessing pipeline from Section 4 to local-minimum configurations from the filtered ODAC25 trajectories, yielding 153,249 training and 3,582 validation datapoints. Since the official test set is unavailable, we use the validation split for testing and randomly hold out 3,130 training datapoints for validation. Hyperparameters and curriculum adjustments are detailed in Appendix C.

5.1 Coverage of adsorption configurations

Baselines. We compare AtomMOF with 10 ODE steps and no steering against grand canonical Monte Carlo (GCMC) and random sampling (RS). Since GCMC and RS only place adsorbates, we use the AtomMOF-predicted framework and evaluate adsorbate placement accuracy. We run GCMC in `RASPA3` [46] with classical force fields. For RS, we repeatedly insert a fixed number of adsorbates and reject steric clashes, defined as any pairwise distance $d_{ij} \leq 0.8(r_i + r_j)$ using covalent radii r_i and r_j. Implementation details and baseline discussion appear in Appendix G.

Metrics. We report *coverage* (COV) [47], which measures how many reference configurations are recovered for a given MOF-adsorbate system. For a given system, let $\mathcal{D} = \{\mathcal{S}^{(n)}\}_{n=1}^{N}$ be the set of ground-truth configurations and $\hat{\mathcal{D}}$ the set of predictions. Let $\hat{\mathcal{D}}_n \subseteq \hat{\mathcal{D}}$ denote the subset of

predictions that match $\mathcal{S}^{(n)}$ under `StructureMatcher` with $\mathrm{stol} = 0.5$. We then compute the fraction of recovered structures as $\mathrm{COV} = \frac{1}{N}\sum_n \mathbb{I}[|\hat{\mathcal{D}}_n| > 0]$.

Results. Figure 7 shows that AtomMOF achieves higher coverage than both baselines under the same time budget, and is orders of magnitude faster than GCMC. AtomMOF reaches 23.11% coverage in under 1 s, whereas GCMC requires nearly 10^4 s to reach 21.06%. RS runs on a time scale comparable to AtomMOF but consistently attains lower coverage, indicating that AtomMOF learns a distribution that better matches the reference adsorption configurations. See Appendix G.2 for a complementary diversity evaluation.

5.2 Adsorption energy success rate

Sampling and relaxation. We generate 1 and 10 candidates per test datapoint. To improve stability during MLIP relaxation, we apply Feynman-Kac steering with the penalty $E_{\text{overlap}} = \sum_{i<j} \max(0, 0.8(r_i + r_j) - d_{ij})^2$, where d_{ij} is the interatomic distance and r_i and r_j are the covalent radii. We then filter remaining overlapping structures using the criterion $d_{ij} \leq 0.8(r_i + r_j)$ and relax structures with `eSEN-ODAC25-Full`.

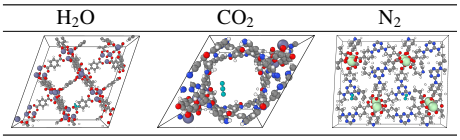


Figure 9: **Examples.** MOF-adsorbate configurations predicted by AtomMOF. Adsorbates are highlighted in teal for clarity.

Metrics. Following Lan et al. [6] and Kolluru and Kitchin [31], we report success rate. A prediction is successful if: (1) MLIP relaxation converges within 300 steps to a maximum force below 0.05 eV/Å and (2) $\varepsilon_{\min} \leq 0.1$ eV, where $\varepsilon_{\min} = \min_{\hat{\mathcal{S}} \in \hat{\mathcal{D}}} \Delta E_{\text{ads}}(\hat{\mathcal{S}}) - \min_{\mathcal{S} \in \mathcal{D}} \Delta E_{\text{ads}}(\mathcal{S})$. Because DFT relaxation and energy evaluation are prohibitively expensive for large MOF-adsorbate systems, we use MLIP-based adsorption energies as a surrogate, following prior work [48, 49]. To use a different MLIP for evaluation than for relaxation, we use UMA (`uma-s-1p2`).

Results. Table 4 shows that AtomMOF achieves success rates of 13.18% with 1 sample and 20.64% with 10 samples. It also finds configurations with lower adsorption energies than the reference ($\varepsilon_{\min} < 0$) in 6.59% and 16.40% of systems, respectively, which is important for accurate adsorption-energy estimation [6]. With 10 samples, the average minimum energy difference is 0.039 eV, indicating that the best configurations are close to the reference minimum. This agrees with Figure 8, where the distribution of minimum adsorption energies closely matches the reference. Representative adsorption configurations are shown in Figure 9.

Table 4: **Adsorption energy success rate.** Success rate and minimum energy difference ($\varepsilon_{\min}$) under UMA (`uma-s-1p2`) evaluation for 1 and 10 samples. Structures are relaxed with eSEN; average $\varepsilon_{\min}$ is computed over converged cases.

Metric	1 Sample	10 Samples
Success rate (%)	13.18	20.64
$\varepsilon_{\min} < 0$ (%)	6.59	16.40
Avg. $\varepsilon_{\min}$ (eV)	0.535	0.039

6 Conclusion

We present AtomMOF, a scalable flow-based model for MOF-adsorbate structure prediction. Unlike prior approaches with structural constraints, AtomMOF models all atoms directly, which we find necessary for high structural accuracy. We also show predictable scaling with model size and compute. To improve sampling quality, we introduce MLIP-guided Feynman-Kac steering, which increases structural stability and validity. Empirically, AtomMOF outperforms prior baselines on BW. On ODAC25, it recovers diverse adsorption configurations far faster than GCMC and provides MLIP relaxation initializations that can reach lower adsorption energies than the reference.

Limitations & future work. AtomMOF is less accurate on very large and out-of-distribution systems, such as those with unseen metal nodes or adsorbates, and it complements rather than replaces GCMC, which also yields thermodynamic quantities such as Henry constants and adsorption isotherms. Moreover, match rate cannot credit valid polymorphs absent from the references, although a MOFid [39] analysis found no such topological diversity in the BW test set. Future work includes broader data coverage, thermodynamic prediction, and polymorph-aware evaluation.

Acknowledgments and Disclosure of Funding

This work was supported by Artificial intelligence industrial convergence cluster development project funded by the Ministry of Science and ICT(MSIT, Korea) & Gwangju Metropolitan City; Advanced GPU Utilization Support Program(Beta Service) funded by the Government of the Republic of Korea (Ministry of Science and ICT); Advanced GPU Utilization Support Program funded by the Government of the Republic of Korea (Ministry of Science and ICT); Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education (RS-2025-25435147); Institute for Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (RS-2019-II190075, Artificial Intelligence Graduate School Support Program(KAIST)); National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (RS-2025-02216257); GRDC(Global Research Development Center) Cooperative Hub Program through the National Research Foundation of Korea(NRF) grant funded by the Ministry of Science and ICT(MSIT) (No. RS-2024-00436165); and Institute of Information & Communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (RS-2025-02304967, AI Star Fellowship(KAIST)).

References

[1] Qihui Qian, Patrick A Asinger, Moon Joo Lee, Gang Han, Katherine Mizrahi Rodriguez, Sharon Lin, Francesco M Benedetti, Albert X Wu, Won Seok Chi, and Zachary P Smith. MOF-based membranes for gas separations. *Chemical Reviews*, 120(16):8161–8266, 2020.

[2] JeongYong Lee, Omar K Farha, John Roberts, Karl A Scheidt, SonBinh T Nguyen, and Joseph T Hupp. Metal–organic framework materials as catalysts. *Chemical Society Reviews*, 38(5):1450–1459, 2009.

[3] Lauren E Kreno, Kirsty Leong, Omar K Farha, Mark Allendorf, Richard P Van Duyne, and Joseph T Hupp. Metal–organic framework materials as chemical sensors. *Chemical Reviews*, 112(2):1105–1125, 2012.

[4] Saptasree Bose, Debabrata Sengupta, Thomas M Rayder, Xiaoliang Wang, Kent O Kirlikovali, Ali K Sekizkardes, Timur Islamoglu, and Omar K Farha. Challenges and opportunities: Metal–organic frameworks for direct air capture. *Advanced Functional Materials*, 34(43):2307478, 2024.

[5] DJ Adams. Grand canonical ensemble monte carlo for a lennard-jones fluid. *Molecular Physics*, 29(1):307–311, 1975.

[6] Janice Lan, Aini Palizhati, Muhammed Shuaibi, Brandon M Wood, Brook Wander, Abhishek Das, Matt Uyttendaele, C Lawrence Zitnick, and Zachary W Ulissi. Adsorbml: a leap in efficiency for adsorption energy calculations using generalizable machine learning potentials. *npj Computational Materials*, 9(1):172, 2023.

[7] 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*, 2025. URL https://openreview.net/forum?id=dNT3abOsLo.

[8] 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.

[9] Mianzhi Pan, JianFei Li, Peishuo Liu, Botian Wang, Yawen Ouyang, Yiming Rong, Hao Zhou, and Jianbing Zhang. Enhancing spatial reasoning in large language models for metal-organic frameworks structure prediction. *arXiv preprint arXiv:2601.09285*, 2026.

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

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

[12] Yize Jiang, Xinze Li, Yuanyuan Zhang, Jin Han, Youjun Xu, Ayush Pandit, Zaixi Zhang, Mengdi Wang, Mengyang Wang, Chong Liu, et al. Posex: Ai defeats physics approaches on protein-ligand cross docking. *arXiv preprint arXiv:2505.01700*, 2025.

[13] Jeremy Wohlwend, Gabriele Corso, Saro Passaro, Noah Getz, Mateo Reveiz, Ken Leidal, Wojtek Swiderski, Liam Atkinson, Tally Portnoi, Itamar Chinn, et al. Boltz-1 democratizing biomolecular interaction modeling. *BioRxiv*, pages 2024–11, 2025.

[14] Erik Hartman, Jonas Wallin, Johan Malmström, and Jimmy Olsson. Controllable protein design through feynman-kac steering. *arXiv preprint arXiv:2511.09216*, 2025.

[15] Peter G Boyd, Arunraj Chidambaram, Enrique García-Díez, Christopher P Ireland, Thomas D Daff, Richard Bounds, Andrzej Gładysiak, Pascal Schouwink, Seyed Mohamad Moosavi, M Mercedes Maroto-Valer, et al. Data-driven design of metal-organic frameworks for wet flue gas co2 capture. *Nature*, 576(7786):253–256, 2019. doi: 10.24435/materialscloud:2018.0016/v3.

[16] Anuroop Sriram, Logan M Brabson, Xiaohan Yu, Sihoon Choi, Kareem Abdelmaqsoud, Elias Moubarak, Pim de Haan, Sindy Löwe, Johann Brehmer, John R Kitchin, et al. The Open DAC 2025 dataset for sorbent discovery in direct air capture. *arXiv preprint arXiv:2508.03162*, 2025.

[17] 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.

[18] Peijia Lin, Pin Chen, Rui Jiao, Qing Mo, Cen Jianhuan, Wenbing Huang, Yang Liu, Dan Huang, and Yutong Lu. Equivariant diffusion for crystal structure prediction. In *Forty-first International Conference on Machine Learning*, 2024.

[19] Rui Jiao, Wenbing Huang, Yu Liu, Deli Zhao, and Yang Liu. Space group constrained crystal generation. In *The Twelfth International Conference on Learning Representations*, 2024.

[20] Daniel Levy, Siba Smarak Panigrahi, S'ekou-Oumar Kaba, Qiang Zhu, Kin Long Kelvin Lee, Mikhail Galkin, Santiago Miret, and Siamak Ravanbakhsh. SymmCD: Symmetry-preserving crystal generation with diffusion models. In *The Thirteenth International Conference on Learning Representations*, 2025.

[21] 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.

[22] 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.

[23] 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.

[24] Andrew W Senior, Richard Evans, John Jumper, James Kirkpatrick, Laurent Sifre, Tim Green, Chongli Qin, Augustin Žídek, Alexander WR Nelson, Alex Bridgland, et al. Improved protein structure prediction using potentials from deep learning. *Nature*, 577(7792):706–710, 2020.

[25] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, 2021.

[26] 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.

[27] Saro Passaro, Gabriele Corso, Jeremy Wohlwend, Mateo Reveiz, Stephan Thaler, Vignesh Ram Somnath, Noah Getz, Tally Portnoi, Julien Roy, Hannes Stark, et al. Boltz-2: Towards accurate and efficient binding affinity prediction. *BioRxiv*, 2025.

[28] Chai Discovery team, Jacques Boitreaud, Jack Dent, Matthew McPartlon, Joshua Meier, Vinicius Reis, Alex Rogozhonikov, and Kevin Wu. Chai-1: Decoding the molecular interactions of life. *BioRxiv*, pages 2024–10, 2024.

[29] ByteDance AML AI4Science Team, Xinshi Chen, Yuxuan Zhang, Chan Lu, Wenzhi Ma, Jiaqi Guan, Chengyue Gong, Jincai Yang, Hanyu Zhang, Ke Zhang, et al. Protenix-advancing structure prediction through a comprehensive AlphaFold 3 reproduction. *BioRxiv*, pages 2025–01, 2025.

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

[31] Adeesh Kolluru and John R Kitchin. Adsorbdiff: Adsorbate placement via conditional denoising diffusion. *arXiv preprint arXiv:2405.03962*, 2024.

[32] Minkyu Kim, Nayoung Kim, Honghui Kim, and Sungsoo Ahn. CatFlow: Co-generation of slab-adsorbate systems via flow matching. *arXiv preprint arXiv:2602.05372*, 2026.

[33] 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.

[34] 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.

[35] Chaitanya K. Joshi, Xiang Fu, Yi-Lun Liao, Vahe Gharakhanyan, Benjamin Kurt Miller, Anuroop Sriram, and Zachary W. Ulissi. All-atom diffusion transformers: Unified generative modelling of molecules and materials. In *International Conference on Machine Learning*, 2025.

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

[37] Raghav Singhal, Zachary Horvitz, Ryan Teehan, Mengye Ren, Zhou Yu, Kathleen McKeown, and Rajesh Ranganath. A general framework for inference-time scaling and steering of diffusion models. In *Forty-second International Conference on Machine Learning*, 2025.

[38] Nanye Ma, Mark Goldstein, Michael S Albergo, Nicholas M Boffi, Eric Vanden-Eijnden, and Saining Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *European Conference on Computer Vision*, pages 23–40. Springer, 2024.

[39] Benjamin J Bucior, Andrew S Rosen, Maciej Haranczyk, Zhenpeng Yao, Michael E Ziebel, Omar K Farha, Joseph T Hupp, J Ilja Siepmann, Alán Aspuru-Guzik, and Randall Q Snurr. Identification schemes for metal–organic frameworks to enable rapid search and cheminformatics analysis. *Crystal Growth & Design*, 19(11):6682–6697, 2019.

[40] Tomoki Yamashita, Shinichi Kanehira, Nobuya Sato, Hiori Kino, Kei Terayama, Hikaru Sawahata, Takumi Sato, Futoshi Utsuno, Koji Tsuda, Takashi Miyake, et al. Cryspy: a crystal structure prediction tool accelerated by machine learning. *Science and Technology of Advanced Materials: Methods*, 1(1):87–97, 2021.

[41] Thomas F Willems, Chris H Rycroft, Michaeel Kazi, Juan C Meza, and Maciej Haranczyk. Algorithms and tools for high-throughput geometry-based analysis of crystalline porous materials. *Microporous and Mesoporous Materials*, 149(1):134–141, 2012.

[42] 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.

[43] Benjamin Rhodes, Sander Vandenhaute, Vaidotas Šimkus, James Gin, Jonathan Godwin, Tim Duignan, and Mark Neumann. Orb-v3: atomistic simulation at scale. *arXiv preprint arXiv:2504.06231*, 2025.

[44] Jaesun Kim, Jinmu You, Yutack Park, Yunsung Lim, Yujin Kang, Jisu Kim, Haekwan Jeon, Suyeon Ju, Deokgi Hong, Seung Yul Lee, et al. Optimizing cross-domain transfer for universal machine learning interatomic potentials. *Nature Communications*, 2026.

[45] Kevin Maik Jablonka. MOFChecker, June 2023. URL https://github.com/kjappelbaum/mofchecker.

[46] YA Ran, S Sharma, SRG Balestra, Z Li, S Calero, TJH Vlugt, RQ Snurr, and D Dubbeldam. Raspa3: A monte carlo code for computing adsorption and diffusion in nanoporous materials and thermodynamics properties of fluids. *The Journal of Chemical Physics*, 161(11), 2024.

[47] Yuyang Wang, Ahmed A. Elhag, Navdeep Jaitly, Joshua M. Susskind, and Miguel Ángel Bautista. Swallowing the bitter pill: Simplified scalable conformer generation. In *Forty-first International Conference on Machine Learning*, 2024.

[48] Vaidotas Simkus, Anders Christensen, Steven Bennett, Ian Johnson, Mark Neumann, James Gin, Jonathan Godwin, and Benjamin Rhodes. Mofasa: A step change in metal-organic framework generation. *arXiv preprint arXiv:2512.01756*, 2025.

[49] Zicheng Liu, Ben Fei, and Di Huang. Pro-mof: Policy optimization with universal atomistic models for controllable mof generation. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=BIzrFlp0hv.

[50] 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.

[51] 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.

[52] Alexander Stukowski. Visualization and analysis of atomistic simulation data with OVITO-the Open Visualization Tool. *MODELLING AND SIMULATION IN MATERIALS SCIENCE AND ENGINEERING*, 18(1), JAN 2010. ISSN 0965-0393. doi: {10.1088/0965-0393/18/1/015012}.

[53] Stephen L Mayo, Barry D Olafson, and William A Goddard. Dreiding: a generic force field for molecular simulations. *Journal of Physical Chemistry*, 94(26):8897–8909, 1990.

[54] Anthony K Rappé, Carla J Casewit, KS Colwell, William A Goddard III, and W Mason Skiff. Uff, a full periodic table force field for molecular mechanics and molecular dynamics simulations. *Journal of the American Chemical Society*, 114(25):10024–10035, 1992.

[55] Jeffrey J Potoff and J Ilja Siepmann. Vapor–liquid equilibria of mixtures containing alkanes, carbon dioxide, and nitrogen. *AIChE Journal*, 47(7):1676–1682, 2001.

[56] Ling Zhang and J Ilja Siepmann. Direct calculation of henry's law constants from gibbs ensemble monte carlo simulations: nitrogen, oxygen, carbon dioxide and methane in ethanol. *Theoretical Chemistry Accounts*, 115(5):391–397, 2006.

[57] Hans W Horn, William C Swope, and Jed W Pitera. Characterization of the tip4p-ew water model: Vapor pressure and boiling point. *The Journal of chemical physics*, 123(19), 2005.

[58] Thomas A Manz and David S Sholl. Chemically meaningful atomic charges that reproduce the electrostatic potential in periodic and nonperiodic materials. *Journal of Chemical Theory and Computation*, 6(8):2455–2468, 2010.

[59] Guobin Zhao and Yongchul G Chung. Pacman: a robust partial atomic charge predicter for nanoporous materials based on crystal graph convolution networks. *Journal of chemical theory and computation*, 20(12):5368–5380, 2024.

[60] Akhila Ram, Juno Nam, Aaron J Havens, Benjamin Kurt Miller, and Rafael Gomez-Bombarelli. Diffusion sampling of adsorbate configurations on catalyst surfaces. In *ICML 2026 AI for Science Workshop*, 2026.

[61] Albert P Bartók, Risi Kondor, and Gábor Csányi. On representing chemical environments. *Physical Review B—Condensed Matter and Materials Physics*, 87(18):184115, 2013.

APPENDIX

A Model architecture **16**

B MLIP-guided Feynman-Kac steering algorithm **19**

C Training details **20**

D Additional ablations and controls **22**
D.1 Deterministic versus generative prediction 22

E Experimental details for Feynman-Kac steering **23**
E.1 Metric definitions 23
E.2 Comparison with best-of-N reranking 23
E.3 Compute overhead of FK steering 23
E.4 Sensitivity analysis 23

F MLIP-guided FK steering for inorganic crystal structure prediction **26**

G Experimental details for MOF-adsorbate structure prediction **26**
G.1 Baseline methods and implementation details 26
G.2 Diversity of adsorption configurations 26

H Broader impacts **27**

A Model architecture

Here we describe the detailed implementation of our denoiser, $\boldsymbol{\mu}_t^\theta(\mathbf{X}_t, \boldsymbol{\ell}_t \mid \mathcal{B})$ (Algorithm 1). As shown in Figure 10, the architecture consists of three stages: (i) an *encoder* that featurizes the building blocks $\mathcal{B}$ into single-atom and pairwise representations (Algorithm 2); (ii) a *trunk* that performs the main computation over atomic interactions (Algorithm 3); and (iii) a *decoder* that maps the transformed features back to per-atom outputs to predict $\hat{\mathbf{X}}_1$ and $\hat{\boldsymbol{\ell}}_1$ (Algorithm 4).

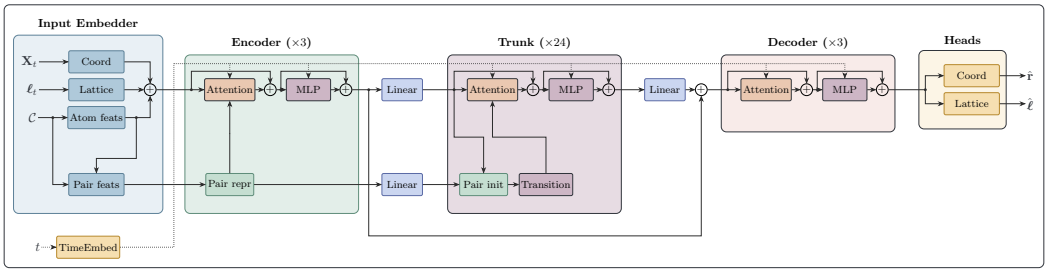


Figure 10: **AtomMOF architecture.** Building-block features and noisy coordinates and lattice parameters are processed by an encoder, trunk, and decoder to predict the clean structure. Pairwise representations condition attention, while timestep embeddings condition the network blocks.

Algorithm 1 Denoiser $\boldsymbol{\mu}_t^\theta(\mathbf{X}_t, \boldsymbol{\ell}_t \mid \mathcal{B})$

Input: Noisy inputs $\boldsymbol{X}_t, \boldsymbol{\ell}_t$, Time t, building block $\mathcal{B}$.
Output: Denoised updates $\hat{\mathbf{X}}_1, \hat{\boldsymbol{\ell}}_1$.
 # 1. Encoding
$\boldsymbol{Q}, \boldsymbol{P} \leftarrow \text{Encoder}(\boldsymbol{X}_t, \boldsymbol{\ell}_t, t, \mathcal{B})$
 # 2. Projection
$\boldsymbol{S} \leftarrow \text{Linear}(\boldsymbol{Q})$
$\boldsymbol{Z} \leftarrow \text{Linear}(\boldsymbol{P})$
 # 3. Trunk
$\boldsymbol{S}' \leftarrow \text{Trunk}(\boldsymbol{S}, \boldsymbol{Z}, t)$
 # 4. Residual projection
$\boldsymbol{Q} \leftarrow \boldsymbol{Q} + \text{Linear}(\boldsymbol{S}')$
 # 5. Decoding
$\hat{\mathbf{X}}_1, \hat{\boldsymbol{\ell}}_1 \leftarrow \text{Decoder}(\boldsymbol{Q}, t)$
Return: $\hat{\mathbf{X}}_1, \hat{\boldsymbol{\ell}}_1$

Algorithm 2 Encoder

Input: Noisy coordinates $\boldsymbol{X}_t$, noisy lattice $\boldsymbol{\ell}_t$, time t, building block $\mathcal{B}$.
Output: Single representation $\boldsymbol{Q}$, pair representation $\boldsymbol{P}$.
 # 1. Feature embedding
Concatenate $\mathcal{B}$'s RDKit-initialized $\boldsymbol{X}_{\text{local}}$, formal charges, and atom types into $\boldsymbol{f}_{\text{atom}}$.
Initialize single representation: $\boldsymbol{C} \leftarrow \text{LinearNoBias}(\boldsymbol{f}_{\text{atom}})$
 # 2. Pairwise representation construction
Compute pairwise distances $\boldsymbol{D}_{ij} \leftarrow \boldsymbol{X}_{\text{local},i} - \boldsymbol{X}_{\text{local},j}$ and norms $\|\boldsymbol{D}_{ij}\|$.
Construct block adjacency mask $\boldsymbol{v}_{ij}$.
Initialize pair representation $\boldsymbol{P}$:
$\boldsymbol{P} \leftarrow \text{Embed}_{\text{pos}}(\boldsymbol{D}) + \text{Embed}_{\text{dist}}(\|\boldsymbol{D}\|) + \text{Embed}_{\text{bond}}(\mathcal{B}_{\text{bond}})$.
$\boldsymbol{P} \leftarrow \boldsymbol{P} \odot \boldsymbol{v} + \text{Embed}_{\text{mask}}(\boldsymbol{v}) \odot \boldsymbol{v}$.
 # 3. Condition integration
Add single context to pairs:
$\boldsymbol{P}_{ij} \leftarrow \boldsymbol{P}_{ij} + \text{MLP}_Q(\boldsymbol{C}_i) + \text{MLP}_K(\boldsymbol{C}_j) + \text{MLP}_{\text{pair}}(\boldsymbol{P}_{ij})$.
Add noise conditioning to single rep $\boldsymbol{Q}$:
$\boldsymbol{Q} \leftarrow \boldsymbol{C} + \text{Linear}(\boldsymbol{X}_t) + \text{Linear}(\boldsymbol{\ell}_t)$.
 # 4. Encoding
$\boldsymbol{Q}, \boldsymbol{P} \leftarrow \text{DiT}_{\text{encoder}}(\boldsymbol{Q}, \boldsymbol{P}, t)$.
Return: $\boldsymbol{Q}, \boldsymbol{P}$

Algorithm 3 Trunk

Input: Single representation $\boldsymbol{S}_{\text{in}}$, pair representation $\boldsymbol{Z}_{\text{in}}$, time t.
Output: Updated single representation $\boldsymbol{S}'$.
 # 1. Initialization
$\boldsymbol{S} \leftarrow \text{Linear}_{\text{init}}(\boldsymbol{S})$
$\boldsymbol{Z}_{\text{bias}} \leftarrow \text{Linear}_1(\boldsymbol{S}) \oplus \text{Linear}_2(\boldsymbol{S})^T$
$\boldsymbol{Z} \leftarrow \boldsymbol{Z} + \boldsymbol{Z}_{\text{bias}}$
 # 2. Normalization
$\boldsymbol{S} \leftarrow \text{Linear}(\text{LayerNorm}(\boldsymbol{S}))$, $\boldsymbol{Z} \leftarrow \text{Linear}(\text{LayerNorm}(\boldsymbol{Z}))$
 # 3. Transformer Pass
$\boldsymbol{S}_{\text{out}} \leftarrow \text{DiT}_{\text{trunk}}(\boldsymbol{S}, \boldsymbol{Z}, t)$
Return: $\boldsymbol{S}'$

Algorithm 4 Decoder

Input: Single representation $\boldsymbol{Q}$, time t.
Output: Coordinate prediction $\hat{\mathbf{X}}_1$, lattice prediction $\hat{\boldsymbol{\ell}}_1$.
 # 1. Feature decoding
$\boldsymbol{Q}_{\text{dec}} \leftarrow \text{DiT}_{\text{decoder}}(\boldsymbol{Q}, t)$
 # 2. Coordinate projection
$\hat{\mathbf{X}}_1 \leftarrow \text{Linear}(\text{LayerNorm}(\boldsymbol{Q}_{\text{dec}}))$
$\boldsymbol{Q}_{\text{global}} \leftarrow \text{Average}(\boldsymbol{Q}_{\text{dec}})$
$\hat{\boldsymbol{\ell}}_1 \leftarrow \text{Linear}(\text{LayerNorm}(\boldsymbol{Q}_{\text{global}}))$
Return: $\hat{\mathbf{X}}_1, \hat{\boldsymbol{\ell}}_1$

Algorithm 5 Diffusion Transformer (DiT) Architecture

Input: Input $\boldsymbol{S}$, time t, (optional) bias $\boldsymbol{Z}$.
Output: Updated $\boldsymbol{S}$.
 # 1. Time conditioning
$\boldsymbol{c} \leftarrow \text{MLP}(\text{SinusoidalEmbedding}(t))$
for $l = 1 \dots L$ **do**
  # 2. Adaptive modulation (AdaLN)
  $\boldsymbol{\gamma}_{1,2}, \boldsymbol{\beta}_{1,2}, \boldsymbol{\alpha}_{1,2} \leftarrow \text{Chunk}(\text{Linear}(\text{SiLU}(\boldsymbol{c})), 6)$
  # 3. Self-attention block
  $\boldsymbol{S}_{\text{norm}} \leftarrow \text{LayerNorm}(\boldsymbol{S}) \cdot (1 + \boldsymbol{\gamma}_1) + \boldsymbol{\beta}_1$
  $\boldsymbol{S} \leftarrow \boldsymbol{S} + \boldsymbol{\alpha}_1 \cdot \text{Attn}(\boldsymbol{S}_{\text{norm}}, \text{bias} = \boldsymbol{Z})$
  # 4. Feed-forward block
  $\boldsymbol{S}_{\text{norm}} \leftarrow \text{LayerNorm}(\boldsymbol{S}) \cdot (1 + \boldsymbol{\gamma}_2) + \boldsymbol{\beta}_2$
  $\boldsymbol{S} \leftarrow \boldsymbol{S} + \boldsymbol{\alpha}_2 \cdot \text{MLP}(\boldsymbol{S}_{\text{norm}})$
end for
Return: $\boldsymbol{S}$

Algorithm 6 Attention with pairwise bias

Input: Input $\boldsymbol{X}$, (optional) pairwise bias $\boldsymbol{Z}$.
Output: Attention output $\boldsymbol{Y}$
$\mathsf{Q}, \mathsf{K}, \mathsf{V} \leftarrow \text{Linear}_{q,k,v}(\boldsymbol{X})$
$\boldsymbol{A} \leftarrow \text{softmax}\left(\frac{\mathsf{Q}\mathsf{K}^\top}{\sqrt{D_k}} + \text{Linear}_{\text{bias}}(\boldsymbol{Z})\right)\mathsf{V}$
$\boldsymbol{Y} \leftarrow \text{Linear}_{\text{out}}(\boldsymbol{A})$
Return: $\boldsymbol{Y}$

B MLIP-guided Feynman-Kac steering algorithm

In this section, we present the details of the MLIP-guided Feynman–Kac (FK) steering algorithm. The key idea is to bias the sampling trajectory towards low-energy structures by maintaining an ensemble of K particles, updating their weights at intermediate steps using a potential, and resampling the particles according to these weights. As summarized in Table 5, the potential can depend on the current state, consecutive states, or the trajectory history. Algorithm 7 details the full FK procedure.

Table 5: **Overview of potential functions.** Here, $\mathcal{E}_t := E(\hat{\mathcal{S}}_1)$ denotes the potential energy of the clean state predicted at time t. Δt refers to the resampling interval and N_t denotes the number of steps elapsed up to time t.

Potential	**Domain**	**Definition** $G_t(\cdot)$
Difference	$\mathcal{S}_{t-\Delta t}, \mathcal{S}_t$	$\exp\left(\lambda(\mathcal{E}_{t-\Delta t} - \mathcal{E}_t)\right)$
Immediate	$\mathcal{S}_t$	$\exp\left(-\lambda \mathcal{E}_t\right)$
Max	$\{\mathcal{S}_\tau\}_{\tau\in[0,t]}$	$\exp\left(-\lambda \min_{\tau\le t} \mathcal{E}_\tau\right)$
Sum	$\{\mathcal{S}_\tau\}_{\tau\in[0,t]}$	$\exp(-\frac{\lambda}{N_t} \sum_{\tau\le t} \mathcal{E}_\tau)$

Algorithm 7 Feynman–Kac Steering with MLIP

Input: Denoiser $\boldsymbol{\mu}_\theta$, MLIP $E(\cdot)$.
Hyperparams: Steering strength λ, particles K, resampling interval $\Delta\tau$.
Initialize: Sample $\mathcal{S}_0^{(k)} \sim p_0$ for $k = 1, \dots, K$.
Initialize: History of energies $\mathcal{E}_{\text{hist}}^{(k)} \leftarrow \emptyset$ for all k.
Define time discretization $\{t_i\}_{i=0}^{T}$ with $t_0 = 0, t_T = 1$.
for $i = 0$ **to** $T-1$ **do**
  $t \leftarrow t_i$, $\Delta t \leftarrow t_{i+1} - t_i$.
  # 1. Predict clean state
  $\hat{\mathcal{S}}_1^{(k)} \leftarrow \boldsymbol{\mu}_t^\theta(\boldsymbol{X}_t^{(k)}, \boldsymbol{\ell}_t^{(k)} \mid \mathcal{B})$ for all k.
  Compute current energies: $\mathcal{E}_{\text{curr}}^{(k)} \leftarrow E(\hat{\mathcal{S}}_1^{(k)})$.
  # 2. Resample (Feynman–Kac)
  if $t \ge t_{\text{start}}$ **and** $i \bmod \Delta\tau = 0$ **then**
    Compute weights w_k using G_t (see Table 5).
    e.g., for difference: $w_k \leftarrow \exp(\lambda(\mathcal{E}_{t-\Delta t}^{(k)} - \mathcal{E}_t^{(k)}))$.
    Normalize weights: $\tilde{w}_k \leftarrow w_k / \sum_{j=1}^{K} w_j$.
    Resample indices $\{j_k\}_{k=1}^{K} \sim \text{Multinomial}(\{\tilde{w}_k\})$.
    Update particles: $\mathcal{S}_t^{(k)} \leftarrow \mathcal{S}_t^{(j_k)}$, $\mathcal{E}_{\text{curr}}^{(k)} \leftarrow \mathcal{E}_{\text{curr}}^{(j_k)}$.
  end if
  Update history: $\mathcal{E}_{\text{hist}}^{(k)} \leftarrow (\mathcal{E}_{\text{hist}}^{(k)}, \mathcal{E}_{\text{curr}}^{(k)})$
  # 3. SDE update (for all k)
  # Step lattice:
  $\mathbf{v}_{t,\boldsymbol{\ell}}^{(k)} \leftarrow (\hat{\boldsymbol{\ell}}_1^{(k)} - \boldsymbol{\ell}_t^{(k)})/(1-t)$
  $\boldsymbol{\ell}_{t+\Delta t}^{(k)} \leftarrow \boldsymbol{\ell}_t^{(k)} + \mathbf{v}_{t,\boldsymbol{\ell}}^{(k)} \Delta t$
  # Step coordinates:
  $\mathbf{v}_{t,\boldsymbol{X}}^{(k)} \leftarrow (\hat{\boldsymbol{X}}_1^{(k)} - \boldsymbol{X}_t^{(k)})/(1-t)$
  $s_t^{(k)} \leftarrow (t\mathbf{v}_{t,\boldsymbol{X}}^{(k)} - \boldsymbol{X}_t^{(k)})/(1-t)$, sample $\boldsymbol{z}^{(k)} \sim \mathcal{N}_C(\mathbf{0}, \boldsymbol{I})$, compute $g(t) \leftarrow \sqrt{(1-t)/2}$
  $\boldsymbol{X}_{t+\Delta t}^{(k)} \leftarrow \boldsymbol{X}_t^{(k)} + (\mathbf{v}_{t,\boldsymbol{X}}^{(k)} + \frac{1}{2} g(t)^2 s_t^{(k)})\Delta t + g(t)\sqrt{\Delta t}\, \boldsymbol{z}^{(k)}$
end for
Return: $\{\mathcal{S}_1^{(k)}\}_{k=1}^{K}$

C Training details

Metal node template construction. For the BW dataset, node-template construction is simple because the dataset contains only 9 recurring node types: for each type, we align all training-set instances with the Kabsch algorithm and average the aligned coordinates, yielding one canonical template per node type. For ODAC25, this averaging strategy is not appropriate because metal-node geometries are substantially more diverse. Instead, we keep the training set geometries themselves and build a dictionary from node SMILES to all matching node structures, covering 883 unique nodes. At inference time, we use this dictionary only to initialize node coordinates by randomly sampling one training-set geometry with the same SMILES. Nearly all validation nodes are covered by this dictionary (472/473), and the rare missing cases are initialized from a Gaussian distribution.

Training setup. We train all model variants with dynamic batching based on a squared-atom limit (N^2_{max}) to handle the high variability in atomic counts. We optimize the model with AdamW and a linear warmup scheduler, and improve convergence and stability with a multi-stage curriculum that progressively increases the maximum number of atoms used in each stage; this curriculum is shared across AtomMOF-S, AtomMOF-M, and AtomMOF-L. Architectural hyperparameters are listed in Table 6, optimization settings in Table 7, and the curriculum schedule in Table 8.

For ODAC25, we reuse the BW hyperparameters, increasing the curriculum atom-count limit by 100 atoms to improve dataset coverage.

Compute resources. Training used distributed data parallelism on 8 GPUs. AtomMOF-S and AtomMOF-L were trained on NVIDIA B200 GPUs with approximately 180 GB memory per GPU, while AtomMOF-M was trained on NVIDIA RTX Pro 6000 Blackwell Max-Q GPUs with 96 GB memory per GPU. Table 9 reports the wall-clock time and GPU-hour estimate for each primary training run; reproducing the released primary checkpoints requires approximately 3,490 GPU-hours in total, excluding preprocessing, inference/evaluation, and exploratory or failed runs.

License. This work uses the BW [15] and ODAC25 [16] datasets. BW uses a Creative Commons Attribution license; ODAC25 uses the FAIR Chemistry License v1, which permits use, redistribution, and derivative works subject to the license terms, attribution requirements, applicable law, and the FAIR Chemistry Acceptable Use Policy.

Softwares. Existing software assets used in this work include RDKit [36], pymatgen [50], Zeo++ [41], RASPA3 [46], ASE [51], and OVITO [52].

Table 6: Hyperparameters for AtomMOF-S, AtomMOF-M, and AtomMOF-L.

Parameter	AtomMOF-S	AtomMOF-M	AtomMOF-L
Encoder & Decoder			
Single dim.	256	256	256
Pairwise dim.	128	128	128
Depth	3	3	3
Heads	4	4	4
Trunk			
Single dim.	256	512	1024
Pairwise dim.	128	256	512
Depth	24	24	24
Heads	8	8	16
Total params.	38.4M	129M	491M
Batch limit (N^2_{max})			
Pretraining	3.0×10^6	1.0×10^6	1.0×10^6
Fine-tuning 1	3.0×10^6	1.0×10^6	1.0×10^6
Fine-tuning 2	2.5×10^6	8.0×10^5	8.0×10^5

Table 7: Training and optimization hyperparameters.

Parameter	Value
Loss weights	
Coordinate weight	1.0
Lengths weight	1.0
Angle weight	0.01
Optimizer (AdamW)	
Learning rate	1×10^{-4}
Weight decay	0.0
Scheduler (linear warmup)	
Base LR	0.0
Warmup steps	10,000

Table 8: **Curriculum learning schedule.** Maximum atom count and training epochs for each stage, shared across AtomMOF-S, AtomMOF-M, and AtomMOF-L.

Setting	Pretraining	Fine-tuning 1	Fine-tuning 2
Maximum atom count			
BW	200	300	No limit
ODAC25	300	400	No limit
Epochs	500	100	500

Table 9: Compute resources for primary AtomMOF training runs. GPU-hours are computed as wall-clock hours $\times$ 8 GPUs.

Experiment	Stage	Wall-clock	GPU-hours
AtomMOF-S (BW)	Pretraining	23.0 h	184
	Fine-tuning 1	7.2 h	58
	Fine-tuning 2	22.3 h	178
	Total	52.5 h	420
AtomMOF-M (BW)	Pretraining	26.7 h	214
	Fine-tuning 1	8.3 h	66
	Fine-tuning 2	83.6 h	669
	Total	118.6 h	949
AtomMOF-L (BW)	Pretraining	35.5 h	284
	Fine-tuning 1	11.5 h	92
	Fine-tuning 2	45.0 h	360
	Total	92.0 h	736
AtomMOF-L (ODAC25)	Pretraining	54.3 h	434
	Fine-tuning 1	19.1 h	153
	Fine-tuning 2	99.8 h	798
	Total	173.1 h	1,385

D Additional ablations and controls

We evaluate building-block features, curriculum training, and deterministic prediction using AtomMOF-M on BW at stol $= 0.3$ and 0.5.

Model architecture ablations. We ablate pairwise representations and bond topology, training each variant for 500 epochs on structures with at most 200 atoms (83.5% of training data). Removing pairwise representations reduces match rate by 25.8% at stol $= 0.3$ (Table 10). Removing bond topology reduces it by only 7.9%, suggesting that atom types and pairwise distances retain some connectivity information.

Curriculum training. We retrain AtomMOF-M without curriculum on the full dataset for a comparable number of steps ($\sim$863K). Curriculum improves all reported metrics (Table 11), but the model without it still outperforms MOFFlow-2, suggesting that the architecture contributes most.

Table 10: **Ablation on building block feature initialization.** Removing the pairwise representation degrades performance most, while bond topology has a smaller effect.

	Match rate (%) ↑	
Configuration	stol $= 0.3$	stol $= 0.5$
AtomMOF-M	11.18	30.80
− Pairwise representation	8.29	26.60
− Bond topology	10.30	30.07

Table 11: **Ablation on curriculum training.** Curriculum learning improves match rate and reduces RMSD at both tolerances.

	stol $= 0.3$		stol $= 0.5$	
Configuration	**MR (%) ↑**	**RMSD ↓**	**MR (%) ↑**	**RMSD ↓**
Without curriculum	12.88	0.1360	36.43	0.2615
With curriculum	14.33	0.1308	37.88	0.2542

D.1 Deterministic versus generative prediction

We compare generative and deterministic AtomMOF-M variants with the same architecture, optimizer, learning-rate warmup, and batching strategy. The deterministic model directly regresses coordinates and lattice parameters in one step. It uses canonical building-block coordinates because random orientation augmentation prevented accurate prediction in our experiments; the generative model retains augmentation and iterative sampling.

Table 12 supports the generative formulation in two respects: it achieves higher single-sample accuracy at both tolerances, and additional sampling further improves match rate and RMSD, whereas repeated deterministic predictions remain identical. A third advantage is compatibility with inference-time steering: FK steering improves physical properties without retraining the model (Section 4.5).

Table 12: **Deterministic versus generative prediction on BW.** Both models use the AtomMOF-M architecture. Deterministic predictions are identical across repeated evaluations. RMSD is computed over successfully matched structures.

stol	Model	MR@1 (%) ↑	RMSD@1 (Å) ↓	MR@5 (%) ↑	RMSD@5 (Å) ↓
0.3	Deterministic	4.00	0.1709	4.00	0.1709
	Generative	**11.18**	**0.1347**	**21.78**	**0.1274**
0.5	Deterministic	17.74	0.3047	17.74	0.3047
	Generative	**30.80**	**0.2622**	**49.46**	**0.2397**

E Experimental details for Feynman-Kac steering

This section provides additional details on the Feynman-Kac steering experiments in Section 4.5. All results use ATOMMOF-M and 200 SDE integration steps. For the MLIPs, we use `eSEN-ODAC25-Full` for eSEN, `uma-s-1p2` for UMA, `orb-v3-conservative-inf-omat` for Orb, and SevenNet-Omni with task `odac23`.

E.1 Metric definitions

We define three metrics: `MOFChecker` validity, structure validity, and formation energy error.

1. **`MOFChecker` validity [45]** (%): an MOF is valid if it passes all `MOFChecker` checks, which include (but are not limited to) the presence of key elements (e.g., C/H and a metal), no atomic overlaps, chemically reasonable coordination environments (e.g., around C and N), sufficient porosity, no unphysically large charges, and no isolated molecules.
2. **Structure validity** (%): a predicted crystal structure is considered valid if (i) all pairwise interatomic distances are at least 0.5 Å, and (ii) the unit-cell volume is at least 0.1 Å^3.
3. **Formation energy error** (eV/atom): the difference between the formation energy of the predicted structure and that of the ground-truth structure, measuring predicted stability relative to the ground truth (negative values indicate a more stable prediction):

$$\Delta E_f \;:=\; \Delta E_f(\hat{\mathcal{S}}) - \Delta E_f(\mathcal{S}). \tag{5}$$

E.2 Comparison with best-of-N reranking

We compare FK steering with best-of-N reranking using K particles or independent samples, respectively. Best-of-N selects the lowest-energy unsteered sample under `uma-s-1p2`; FK uses the same MLIP for steering, and both methods are evaluated with `eSEN-ODAC25-Full`. As shown in Table 13, FK improves formation energy and structural validity over best-of-N. It is more effective for larger MOFs ($N \geq 300$), with structural-validity gains 2.33–2.53 percentage points larger and reductions in ΔE_f 3.1–10.1 times greater than on the full evaluation set.

Table 13: **FK steering versus best-of-N at matched sample counts.** Each entry reports best-of-N / FK, with the better value in bold. K denotes the number of samples or particles. Results cover 1,000 randomly sampled BW test structures.

Data	K	MOF validity (%) ↑	Structural validity (%) ↑	ΔE_f (eV/atom) ↓
All samples	8	43.10 / **44.30**	93.50 / **94.40**	0.02128 / **0.01924**
	16	**45.80** / 45.40	94.90 / **95.60**	0.00989 / **0.00560**
$N \geq 300$	8	9.68 / 9.68	80.65 / **83.87**	0.11992 / **0.09925**
	16	6.45 / **9.68**	83.87 / **87.10**	0.09898 / **0.08562**

E.3 Compute overhead of FK steering

This section quantifies the additional inference-time cost of FK steering. We report inference-only compute for 1,000 BW test structures using the same checkpoint, 200 sampling steps, one sample per structure, and 8 NVIDIA H200 GPUs. As shown in Table 14, unsteered sampling takes 87 s, whereas FK steering increases runtime to 418 s at $K = 4$ and 2,682 s at $K = 32$, corresponding to 0.93-5.96 GPU-hours and a 4.80-30.83× slowdown.

This overhead is expected because FK steering maintains K particles per sample and periodically evaluates an external MLIP potential for resampling. Since batching was kept largely fixed, the overhead could likely be reduced by adjusting batch sizes to fit the memory.

E.4 Sensitivity analysis

This section summarizes the hyperparameter settings used in the Feynman-Kac steering sensitivity analysis (Section 4.5). To isolate the effect of each hyperparameter, we fix a default configuration

Table 14: **Inference-time compute for FK steering on BW.** Results cover 1,000 test structures with the same checkpoint, 200 sampling steps, one sample per structure, and 8 NVIDIA H200 GPUs.

Setting	K	**Wall-clock (s)**	**GPU-hours**	**Samples/sec**	**Slowdown**
No steering	–	87	0.19	11.494	1.00×
FK steering	4	418	0.93	2.392	4.80×
	8	786	1.75	1.272	9.03×
	16	1511	3.36	0.662	17.37×
	32	2682	5.96	0.373	30.83×

and vary one parameter at a time while keeping all others unchanged. The default settings are listed in Table 15. For completeness, we report the numerical results underlying Figure 6 in Table 17.

Table 15: **Default configurations for Feynman-Kac steering sensitivity analysis.** The hyperparameters we consider are the number of particles (K), steering strength (λ), resampling interval ($\Delta\tau$), steering start time t_{start}, and the potential [14].

Parameter	**Value**
K	8
λ	2.0
$\Delta\tau$	5
t_{start}	0.8
Potential	Immediate

To evaluate the robustness of the method, we conducted ablation studies by varying individual hyperparameters. Table 16 summarizes the specific values tested for each parameter.

Table 16: **Hyperparameter variations for sensitivity analysis.** For each experiment, one parameter was varied according to the values below, while all others remained at their default values.

Parameter	**Varied Values**
K	4, 16
λ	1.0, 5.0, 10.0
$\Delta\tau$	2, 5, 10
t_{start}	0, 0.20, 0.40, 0.60, 0.80
Potential	Difference, Max, Sum

Table 17: Numerical results for the FK steering sensitivity analysis (Figure 6).

Parameter	**Value**	**`MOFChecker` validity** (%)	**Structure validity** (%)	ΔE_f (eV/atom)
K	4	40.67	91.20	0.0548
	8	44.02	94.00	0.0333
	16	43.98	94.90	0.0173
	32	47.27	95.50	0.0080
λ	1.0	43.93	92.70	0.0425
	2.0	44.02	94.00	0.0333
	5.0	44.97	94.30	0.0286
	10.0	43.89	94.60	0.0266
$\Delta\tau$	2	42.58	93.10	0.0369
	3	41.78	92.40	0.0346
	5	44.02	94.00	0.0333
	10	46.76	93.80	0.0292
t_{start}	0.0	40.00	88.20	0.0927
	0.2	42.52	89.80	0.0664
	0.4	41.52	91.20	0.0519
	0.6	42.65	91.90	0.0415
	0.8	44.02	94.00	0.0333
Potential	Immediate	44.02	94.00	0.0333
	Difference	42.95	89.30	0.0797
	Max	43.91	94.00	0.0337
	Sum	44.15	93.70	0.0341

F MLIP-guided FK steering for inorganic crystal structure prediction

In principle, MLIP-guided FK steering applies to any flow matching model whose generative ODE can be converted to an SDE. To demonstrate generalization of our method beyond MOFs, we apply MLIP-guided FK steering to OMatG [23], an inorganic CSP model that operates with an SDE formulation. We use the released `Linear-SDE-Gamma` checkpoints for MP-20 and MPTS-52 with the default 310 integration steps, and use UMA (`uma-s-1p2`) as the guidance MLIP. FK hyperparameters match our main experiments: 16 particles, $\lambda = 2.0$, immediate potential, resampling interval of 5, and start time $t = 0.8$.

Table 18 shows that MLIP-guided FK steering consistently improves both metrics: structure validity increases by $+0.68\%$ (MP-20) and $+8.81\%$ (MPTS-52), and ΔE_f decreases by 72.8% and 68.1%, respectively. These results suggest that MLIP-guided FK steering generalizes across diverse generative architectures and material types.

Table 18: **FK steering for OMatG.** Steering OMatG [23] with UMA [42] improves both structure validity and formation energy error on MP-20 and MPTS-52.

Dataset	Method	Validity (%) ↑	ΔE_f (eV/atom) ↓
MP-20	OMatG	99.26	1.4975
	OMatG + FK	99.93	0.4074
MPTS-52	OMatG	91.37	4.2787
	OMatG + FK	99.42	1.3652

G Experimental details for MOF-adsorbate structure prediction

G.1 Baseline methods and implementation details

GCMC implementation. For MOF atoms, we use a mixture of Dreiding [53] and UFF [54]; for adsorbates, we use TraPPE for CO_2, O_2, and N_2 [55, 56], and TIP4P-ew for H_2O [57]. Electrostatics use DDEC6 charges [58] predicted by `PACMAN` [59] for MOF atoms and are computed with Ewald summation using a force tolerance of 10^{-6} kcal/mol/Å; all pairwise interactions use a 12.8 Å cutoff, and simulations are run at 298 K. During Monte Carlo sampling, translation, rotation, and reinsertion moves are attempted with equal probability. For each adsorbate type and loading, we run 10,000 initialization cycles followed by 10,000 production cycles and take the final production configuration as the representative MOF-adsorbate structure.

Atomic overlap criterion for RS. For RS, we reject insertions that create steric clashes, defined as any MOF-adsorbate or inter-adsorbate atom pair under periodic boundary conditions with minimum-image distance $d_{ij} \leq 0.8\,(r_i + r_j)$, where r_i and r_j are ASE covalent radii [51].

Learning-based baselines for ODAC25. Because our work is the first to model full MOF-adsorbate systems, there is no direct learning-based baseline for this task. We nevertheless attempted to adapt MOFFlow-2 to ODAC25, but found that it fails due to the complexity of experimental MOFs. ODAC25 contains 887 metal-node types with 1,384,427 variants, including oxo, hydroxo, water, and other bound species, whereas the hypothetical BW dataset contains only 9 metal-node types. Since these bound atoms vary across metal nodes, fixed-template models such as MOFFlow-2, MOFFlow, and MOF-BFN cannot generate or correct their internal all-atom geometry. Accordingly, the adapted MOFFlow-2 model achieves only 0.244% 10-sample coverage on ODAC25, further motivating all-atom generative modeling for realistic MOF-adsorbate structures.

G.2 Diversity of adsorption configurations

Coverage measures the recovery of reference adsorption configurations, whereas diversity quantifies variation among generated samples. Adapting the clustering-based metric of Ram et al. [60], we count C distinct clash-free clusters among $K = 10$ configurations and define diversity as $100C/K\%$. Repeated identical clash-free predictions yield 10% diversity, while entirely distinct clash-free predictions yield 100%. We represent nearby framework environments using normalized periodic SOAP descriptors [61] within 6 Å of each adsorbate atom, followed by complete-linkage clustering at

a distance threshold of 0.10 to distinguish adsorption sites and orientations. We use the atomic-overlap criterion described above and average within each MOF–adsorbate system, then across systems.

Table 19 shows that AtomMOF achieves substantially higher diversity than the deterministic reference, indicating that its predictions do not collapse to a single configuration. However, higher diversity does not necessarily imply greater practical effectiveness, since distinct but irrelevant configurations can increase this metric. For example, RS achieves nearly maximal diversity but the lowest coverage: 19.27%, compared with 21.06% for GCMC and 23.11% for AtomMOF. Moreover, GCMC and RS select configurations from substantially larger candidate pools. Together, diversity and coverage indicate that AtomMOF generates varied configurations while recovering more reference-supported adsorption modes.

Table 19: **Diversity of adsorption configurations on ODAC25.** Results use 10 samples per input. The deterministic reference is analytical: repeated identical clash-free predictions produce one cluster.

Method	Diversity (%)
Deterministic	10.00
AtomMOF	78.51
GCMC	89.31
RS	99.14

H Broader impacts

Our work may help accelerate screening of MOFs for applications such as direct air capture by reducing the cost of evaluating candidate materials. However, the method could also be misused to screen materials for harmful purposes, and incorrect predictions could waste computational or experimental resources if used without validation. We therefore view AtomMOF as a research tool for prioritization rather than a substitute for physical verification; its predictions should be followed by experiments, especially on large or out-of-distribution systems where reliability is lower.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the main contributions, and the experiments evaluate them. See the abstract, Section 1, Section 4, and Section 5.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper explicitly discusses the main limitations in Section 6.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: This paper does not contain formal theorems or proofs.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper gives preprocessing, splits, metrics, inference settings, architecture, training details, and ODAC25/GCMC evaluation details, with code publicly available at `https://github.com/nayoung10/AtomMOF`. See Section 4.1, Appendix A, Appendix C, and Appendix G.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We release code and reproduction instructions at https://github.com/nayoung10/AtomMOF. The datasets used in this work are publicly available, and the code includes preprocessing and experiment scripts needed to reproduce the main results. See Section 4.1 and Appendix C for dataset access, preprocessing, and training details.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: The main text gives splits, baselines, sampling, and metrics, and the appendix gives optimizer, schedule, hyperparameters, and preprocessing details. See Section 4.1, Section 5, Appendix C, and Appendix G.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: We do not report error bars or significance tests since each run is expensive, following prior work.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Details on compute resources is provided in Appendix C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We reviewed the NeurIPS Code of Ethics and are not aware of any departure from it.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: A positive impact is faster screening of MOFs for applications such as direct air capture; possible downsides include misuse for harmful materials screening and wasted experimental effort from incorrect predictions. See Section 1 and Appendix H.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The release covers MOF code, checkpoints, and structure data, not high-risk generative systems or scraped human data. See Section 5 and https://github.com/nayoung10/AtomMOF.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The appendix states the licenses for the datasets used in this work. See Appendix C.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.

- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We release code, preprocessed data, model checkpoints, and documentation at `https://github.com/nayoung10/AtomMOF`. See also Appendix C.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This work does not involve crowdsourcing or human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This work does not involve crowdsourcing or human subjects, so IRB approval was not required.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are not part of the proposed method; they appear only in related work. See Section 2 and Section 3.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.