Journal Name

Crossmark

RECEIVED
dd Month yyyy

REVISED
dd Month yyyy

PAPER

Enhancing Crystal Optimization at the Latent Descriptor Space

Osman Goni Ridwan[1], Hongfei Xue[2], Gilles Frapper[3] and Qiang Zhu[1,4,*]

[1]Department of Mechanical Engineering and Engineering Science, University of North Carolina at Charlotte, Charlotte, NC 28223, USA

[2]Department of Computer Science, University of North Carolina at Charlotte, Charlotte, NC 28223, USA

[3]Applied Quantum Chemistry Group, Poitiers University-CNRS, Poitiers 86073, France

[4]North Carolina Battery Complexity, Autonomous Vehicle and Electrification (BATT CAVE) Research Center, Charlotte, NC 28223, USA

*Author to whom any correspondence should be addressed.

E-mail: oridwan@charlotte.edu, qzhu8@charlotte.edu

Keywords: crystal structure optimization, latent space, conditional VAE, machine learning, materials discovery

Abstract

Recent advances in deep generative models have opened new avenues for crystal structure prediction and materials discovery. However, generating large-unit-cell crystals with novel topologies while enforcing a target local environment remains challenging. Energy-based relaxation is computationally expensive at scale and does not reliably drive structures toward desired bonding motifs. We present a targeted-design refinement framework that couples a symmetry-conditioned variational autoencoder (CVAE) with a differentiable SO(3) power spectrum objective to steer candidates toward a specified local environment while preserving crystallographic constraints. The central contribution is an end-to-end, GPU-differentiable pipeline that performs batch-wise optimization directly on symmetry-reduced free variables, backpropagating through symmetry-consistent reconstruction and descriptor evaluation. This GPU-accelerated implementation improves throughput by approximately fivefold compared to our previous CPU workflow, while yielding comparable outcomes. For challenging cases where representation-space updates stall, we introduce conditional latent-space refinement via the trained decoder, followed by re-optimization in representation space within an iterative workflow. This dual-level refinement strategy approximately doubles the number of low-energy and unique structures that satisfy the target design criteria. More broadly, this approach can be extended to other target local environments, compositions, and large-unit-cell systems, providing a scalable route to goal-directed crystal optimization for machine learning-driven materials design.

Crystal structures are commonly stored in standardized formats such as CIF, which explicitly enumerate all atomic positions. For learning and optimization at scale, we adopt a compact tabular representation that exploits space-group symmetry to reduce dimensionality while remaining fully reconstructible. Each structure is encoded as a fixed-length vector comprising: (i) the space-group number (`spg`); (ii) the six lattice parameters $(a, b, c, \alpha, \beta, \gamma)$; and (iii) up to eight symmetry-inequivalent Wyckoff sites, each specified by a Wyckoff position index and the fractional coordinates of the corresponding representative atom in the asymmetric unit.

Because the number of occupied Wyckoff sites varies across structures, we cap the maximum at eight and pad unused entries. This yields a 39-column representation: $1 + 6 + 8 \times 4$ (space group, lattice parameters, and Wyckoff-sites).

The compression achieved by symmetry-aware encoding is substantial. For example, a 288-atom carbon structure in space group 229 ($Im\bar{3}m$) with cubic lattice ($a = b = c = 14.96\,\text{Å}$) would require 871 variables in an explicit Cartesian encoding ($1 + 6 + 288 \times 3$). In contrast, the full 288-atom

basis is generated from only three symmetry-inequivalent Wyckoff sites (96l), reducing the representation to 19 variables ($1 + 6 + 3 \times 4$).

We construct the training set from the SACADA (SAMARA Carbon Allotrope Database) collection [1], which contains 154 experimentally known or hypothetical sp^2-bonded carbon allotropes. By default, crystal structures are presented in the highest possible symmetry group. But to increase diversity while preserving crystallographic validity, we apply subgroup augmentation [2]: for each parent structure we enumerate possible subgroups combinations and enumerate the symmetry operations of the wyckoff sites. This procedure yields 63 115 structures spanning a broad range of space groups and Wyckoff multiplicities.

3.2 Conditional VAE

In the previous LEGOxtal approach, state-of-the-art variational autoencoder (VAE) models [3] were trained on the full dataset containing both continuous variables (lattice parameters and fractional coordinates) and discrete variables (space-group labels and Wyckoff positions). In this work, we instead adopt a Conditional Variational Autoencoder (CVAE) [4] architecture, in which the generative process is explicitly conditioned on discrete crystallographic symmetry information. This design enables targeted exploration and optimization of the latent space while keeping the discrete symmetry fixed, allowing the continuous degrees of freedom (cell parameters and Wyckoff fractional coordinates) to adapt toward a desired local environment.

The continuous crystallographic features are transformed using a Gaussian Mixture Model (GMM) [5] representation. Each scalar variable is encoded by a K-component categorical indicator together with a standardized residual, yielding a transformed representation $\tilde{X}$. This GMM-based encoding mitigates strong multi-modality in crystallographic parameters and improves representational smoothness, which is critical for stable VAE training and downstream latent-space optimization. The discrete symmetry information, consisting of the space-group index and Wyckoff site labels, is converted into a one-hot condition vector U.

The encoder network takes $\tilde{X}$ as input and outputs the parameters $(\mu, \log \sigma^2)$ of a Gaussian latent distribution. Latent variables are sampled using the reparameterization trick,

$$z = \mu + \sigma \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I).$$

In parallel, the condition vector U is processed by a multilayer perceptron to produce a 128-dimensional condition embedding e_c. The latent variable z is concatenated with e_c and passed to the decoder, which reconstructs the GMM logits and residuals $\hat{\tilde{X}}$. The training objective consists of a combination of reconstruction and regularization terms. Cross-entropy loss [6] is applied to the discrete GMM component assignments, while the continuous residuals are optimized using negative log-likelihood. These reconstruction losses are combined with a Kullback–Leibler divergence regularization term [7] to enforce a smooth latent prior and promote generalizable latent representations.

At inference time, a latent variable z is sampled from the prior, and the condition embedding e_c is computed for a specified symmetry configuration. The decoder generates $\hat{\tilde{X}}$, which is mapped back to the physical crystallographic parameters $\hat{X}$ via the inverse GMM transform, yielding structures that are consistent with the imposed space-group and Wyckoff constraints.

3.3 SO(3) Descriptor and Local-Environment Loss

To guide crystal structure optimization toward desired bonding motifs, we require a quantitative measure of similarity between local atomic environments that is invariant to global rotations and translations. We adopt the SO(3) power spectrum descriptor introduced by Bartók *et al.* [8], which encodes both radial and angular information of an atom's local neighborhood in a rotation-invariant form. Unlike simple coordination metrics or purely angular order parameters, the SO(3) descriptor provides a compact yet expressive fingerprint of local geometry that is differentiable with respect to atomic positions.

In practice, the descriptor is computed by expanding a Gaussian-smeared neighbor density around each atom in a combined radial and spherical harmonic basis, followed by forming rotation-invariant power spectrum components. Here, $n_{\max}$ controls the radial resolution of the descriptor, $\ell_{\max}$ sets the maximum angular momentum and thus the angular resolution, and r_{cut} defines the local neighborhood radius considered around each atom. For our application, we use $n_{\max} = 2$, $\ell_{\max} = 4$, and $r_{\text{cut}} = 2\,\text{Å}$, yielding $L = 15$ descriptor components per site. Further mathematical details are provided in the Supplementary Information.

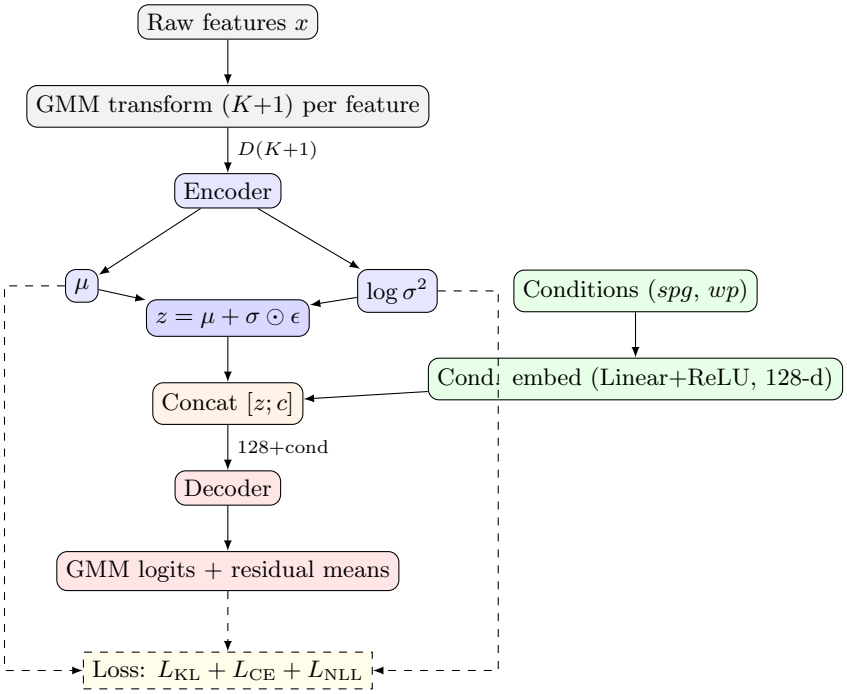

Figure 1. Architecture of the Conditional VAE with DiffGMM data transformation. The encoder processes GMM-transformed continuous features to produce latent variables μ and $\log \sigma^2$, while discrete conditions (space group and Wyckoff positions) are embedded separately and concatenated with the sampled latent z before decoding.

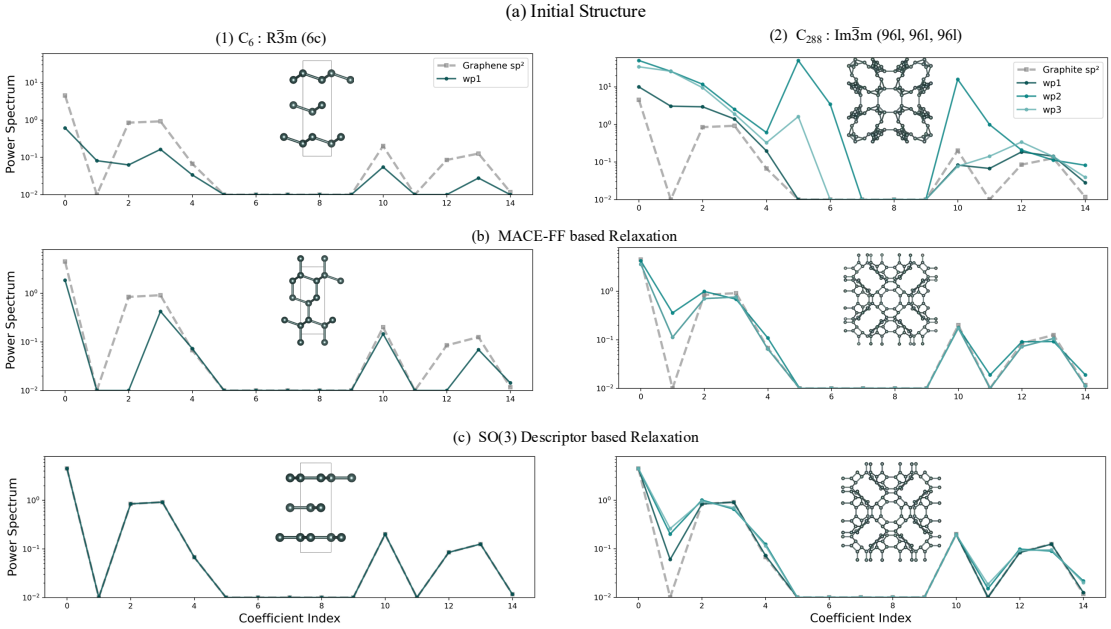

Figure 2. SO(3) power spectrum comparison for CVAE-generated carbon structures before and after refinement. Shown are the initial structures, results after energy-based relaxation using MACE-FF, and results after SO(3) descriptor-based optimization, together with the graphite sp^2 reference (gray dashed line).

Each generated crystal structure is evaluated by computing SO(3) descriptors for all symmetry-inequivalent Wyckoff sites and comparing them to a reference descriptor extracted from graphite (space group 194), which represents the target planar sp^2 bonding environment with coordination number CN=3. The per-structure loss is defined as the mean-squared deviation between the computed and reference descriptors,

$$\ell_i = \frac{1}{W_i L} \sum_{j=1}^{W_i} \sum_{k=1}^{L} (P_{ijk} - p_{\mathrm{ref},k})^2, \tag{1}$$

where W_i is the number of Wyckoff sites in structure i, P_{ijk} denotes the descriptor component, and p_{ref} is the graphite reference fingerprint.

The effectiveness of the SO(3) descriptor as a goal-directed refinement objective is illustrated in Fig. 2 for two representative CVAE-generated structures: a small cell C_6 structure (space group 166, Wyckoff 6*c*) and a large 288-atom structure (space group 229, three 96*l* sites). In both cases, the initial generated structures exhibit substantial deviations from the graphite reference power spectrum (gray dashed lines), indicating that their local atomic environments differ significantly from the target motif.

We compare two distinct refinement strategies: (i) conventional energy-based relaxation using the MACE-mp-0 interatomic potential, which minimizes the total potential energy, and (ii) SO(3) descriptor-based optimization, which directly minimizes the mean-squared deviation from the target local-environment fingerprint defined in Eq. 1. These two approaches optimize fundamentally different objectives and are therefore complementary rather than directly equivalent.

For the C_6 structure, energy-based relaxation converges to a lower-energy configuration with tetrahedral coordination, corresponding to an sp^3-like local environment, despite achieving a favorable total energy. In contrast, descriptor-based optimization preserves the imposed symmetry while driving the structure toward planar trigonal coordination (CN=3), consistent with the graphite reference fingerprint. This highlights that energy minimization alone does not guarantee convergence toward a desired bonding motif when multiple local minima are present.

For the larger C_{288} structure, MACE-based relaxation lowers the energy to -8.518 eV/atom but retains noticeable discrepancies in the SO(3) power spectrum, indicating incomplete convergence to the target sp^2 local environment. Descriptor-based optimization, by construction, yields near-alignment with the graphite reference across all symmetry-inequivalent Wyckoff sites, while maintaining a comparable final energy of -8.480 eV/atom. Importantly, the descriptor-driven optimization reaches this state significantly faster (27.8 s versus 6.2 min on the same CPU hardware), as it bypasses explicit force evaluations and instead exploits a differentiable, local-environment objective.

We emphasize that SO(3) descriptor-based optimization is not intended to replace energy minimization, but rather to provide a fast, physically motivated refinement step that steers generative model outputs toward a specified local bonding motif. In the full workflow, descriptor-based refinement serves as a pre-screening and motif-alignment stage, after which energy-based relaxation and ranking are applied to assess thermodynamic stability.

3.4 Differentiable Descriptor-Driven Optimization

3.4.1 Batch-wise Representation-Space Optimization Traditional crystal structure optimization for large candidate sets faces substantial computational bottlenecks. Our previous LEGOxtal framework processed each structure independently on CPU using Python Multiprocessing feature: from the tabular representation (lattice parameters, space group, Wyckoff sites), we transformed structures to Cartesian coordinates sequentially, computed neighbor lists, evaluated radial density functions with spherical Bessel and harmonic projections, and finally calculated the SO(3) power spectrum deviation from the target. This sequential, per-structure workflow suffered from severe load imbalance—when a complex or large structure consumed excessive computation time, it stalled the entire batch queue, blocking subsequent structures from advancing. In addition, optimization on CPU relied on either derivative-free updates (e.g. Nelder–Mead) or gradient-based routines that required finite-difference gradients in our setting (e.g. L-BFGS-B), which are not naturally suited to GPU batching or integration with differentiable model components. Also, the batch size was limited by available CPU cores and so could not scale to thousands of structures.

To overcome these limitations, we reformulate the problem as a GPU-accelerated batch tensor operation leveraging automatic differentiation [9]. The key innovation is making the optimization pipeline differentiable, enabling automatic differentiation to provide gradients through the reconstruction and SO(3) descriptor evaluation for thousands of structures in parallel. The

Algorithm 1 Batch-wise representation-space pre-relaxation on GPU

Require: batch rows $\{x_i\}$ (containing spg, wps, and $\mathbf{R}$), symmetry constructor WP, descriptor module f, reference spectrum p_{ref}, steps T

Encode batch: $(\mathrm{spg}, \mathrm{wps}, \mathbf{R}) \leftarrow \{x_i\}$ ▷ $\mathbf{R}$: symmetry-reduced free variables
Initialize $\mathbf{R}$ as learnable and set up a gradient-based optimizer
Initialize per-sample state (best loss, plateau counter, and a scaling factor)
for $t = 1$ to T **do**
    Reconstruct batch geometry from $(\mathrm{spg}, \mathrm{wps}, \mathbf{R})$ using WP
    Compute SO(3) descriptors for all structures in the batch: $P \leftarrow f(\cdot)$
    Compute per-sample losses $\{\ell_i\}$ using Eq. 1 (masking padded/unused Wyckoff entries)
    Backpropagate $\ell = \sum_i \ell_i$ to obtain gradients with respect to $\mathbf{R}$
    Apply per-sample gradient conditioning (scaling on plateau and gradient clipping)
    Optimizer update on $\mathbf{R}$ and clamp normalized variables to $[0, 1]$
end for
return optimized $\mathbf{R}$ and final losses $\{\ell_i\}$

workflow proceeds as follows: First, from the tabular representation, we identify the minimal set of free variables (reduced parameters) for each structure based on its space group symmetry constraints. We refer to this symmetry-reduced parameterization as the *representation space*. For instance, cubic systems require only one lattice parameter instead of six, and special Wyckoff positions fix certain fractional coordinates. We encode these free variables as a learnable tensor $\mathbf{R} \in \mathbb{R}^{B\times N}$ (with padding and masking for unused entries), and the SO(3) loss is evaluated for all structures in parallel. Automatic differentiation then provides gradients of the descriptor loss with respect to $\mathbf{R}$, enabling efficient gradient-based updates at scale. We sort structures by space group before batching. Structures sharing the same space group have similar Wyckoff-site options and free-variable layouts, which reduces padding in the fixed-length representation and improves GPU memory efficiency.

In our implementation, symmetry expansion is expressed via precomputed index maps that enable fully vectorized reconstruction on GPU. For each batch, we store generator coordinates for the occupied Wyckoff sites along with mapping tensors that (i) associate each free variable in $\mathbf{R}$ with the corresponding generator coordinates and (ii) enumerate the symmetry operations needed to expand each generator into the full atomic basis. Using $(\mathrm{spg}, \mathrm{wps}, \mathbf{R})$ and these maps, we reconstruct batched lattice matrices and Cartesian coordinates without Python-side loops, and pass them to the SO(3) descriptor module.

With this formulation, reconstruction, descriptor evaluation, and loss computation are executed as a single batched tensor program on GPU. Automatic differentiation then provides gradients with respect to $\mathbf{R}$ through the entire pipeline, enabling efficient gradient-based refinement of thousands of structures in parallel. This removes the need for per-structure finite-difference estimates and makes the pre-relaxation stage compatible with modern differentiable learning and optimization workflows.

A key issue arises in GPU batch-wise optimization that is not present in CPU-based per-structure runs: samples at different stages of convergence can interfere when their gradients are aggregated into a single batched update. In practice, structures that are already close to the target (small loss) may stop improving because their gradients are dominated by a few hard samples with large loss. To mitigate this, we use a gradient-based optimizer with per-structure conditioning: gradients are clipped independently per sample, and each sample's effective step size is adaptively reduced when its loss stagnates. This prevents a small number of difficult cases from dominating the batch update and improves convergence across heterogeneous structures.

3.4.2 Conditional Latent-Space Refinement Representation-space pre-relaxation (Alg. 1) performs local, symmetry-preserving updates of the free variables $\mathbf{R}$. After this stage, a substantial fraction of samples can remain *targeted-design-invalid*, meaning they do not satisfy the CN=3 criterion. This indicates that local updates in $\mathbf{R}$ are sometimes insufficient to reach the target sp^2 manifold from the current configuration.

To handle these hard cases, we refine structures by optimizing the CVAE latent variables z while keeping the discrete symmetry condition fixed, $C = (\mathrm{spg}, \mathrm{wps})$. The refinement objective is unchanged: we minimize the SO(3) loss in Eq. 1. The key difference from representation-space updates is *how* the structure is modified. In representation space, we directly perturb the free variables $\mathbf{R}$ (lattice degrees of freedom and Wyckoff free coordinates), which typically produces

Algorithm 2 Conditional latent-space refinement

Require: invalid index set $\mathcal{I}$ with conditions $\{C_i\}$, trained CVAE decoder, symmetry constructor WP, descriptor module f, reference spectrum p_{ref}, steps T

Initialize latent variables $\{z_i\}$ for $i \in \mathcal{I}$
for $t = 1$ to T **do**
  Decode continuous parameters under $\{C_i\}$ and map back to $\mathbf{R}$
  Reconstruct batched geometries via WP and compute SO(3) descriptors: $P \leftarrow f(\cdot)$
  Compute per-sample losses $\{\ell_i\}$ using Eq. 1
  Backpropagate the summed loss to obtain gradients with respect to $\{z_i\}$
  Apply the same per-sample conditioning as Alg. 1 and update $\{z_i\}$
end for
return decoded structures for $\mathcal{I}$

local, coordinate-wise changes. In latent refinement, updates are applied through the decoder mapping $D_\theta(z, C)$ that generates the continuous crystal parameters under a fixed condition. Because the decoder is trained to model correlations among these parameters, small changes in z can induce coordinated adjustments across lattice and Wyckoff-site coordinates, rather than changing each variable independently. This coupling provides an effective mechanism to move candidates between distinct structural modes that may be difficult to reach using local steps in $\mathbf{R}$. In our implementation, we optimize z directly under the trained decoder and do not add extra prior regularization beyond the CVAE training objective.

For a fixed condition C, we solve

$$\min_z \ \ell\big(\mathcal{S}(z; C)\big), \tag{2}$$

where $\mathcal{S}(z; C)$ denotes the structure obtained by decoding under C (followed by the inverse GMM transform and the same symmetry-consistent reconstruction used in the representation-space stage), and ℓ is the SO(3) loss from Eq. 1.

Latent refinement is used as a targeted recovery step: it is applied only after rep-space pre-relaxation and is followed by rep-space re-optimization (Section 3.5 describes the later energy screening stage).

3.4.3 Iterative Dual-Level Refinement Strategy We use an iterative dual-level workflow that alternates fast representation-space optimization with latent-space refinement for the remaining targeted-design-invalid samples (Fig. 3).

Run 1 applies representation-space optimization to all N candidates and appends valid (CN=3) outputs to $\mathcal{D}_{\mathrm{valid}}$. The remaining samples form the invalid set $\mathcal{I}_1$. For rounds $r = 2, \ldots, R_{\max}$, we apply latent refinement to all samples in $\mathcal{I}_{r-1}$ under their fixed discrete conditions C, decode the updated latents, and run representation-space optimization again. Newly valid (CN=3) structures are added to $\mathcal{D}_{\mathrm{valid}}$, and the rest define $\mathcal{I}_r$. We restart latent variables each round and carry forward only the symmetry condition C and the index set of invalid samples.

3.5 Energy-Based Screening and Database Construction

After descriptor-driven optimization, we apply a post-processing and screening pipeline to construct the final ranked database. First, we retain only those structures that satisfy the target sp^2 coordination criterion (CN=3), yielding N_{valid} candidates. Next, we characterize each candidate using CrystalNets.jl [10] to determine its topological net label and structural dimensionality. We then remove duplicates using this topology–dimensionality signature, which provides an efficient first-pass uniqueness filter.

For thermodynamic screening, we relax the deduplicated candidates with the MACE interatomic potential [11] and compute their energies. After relaxation, we perform a second uniqueness check based on the full atomic structure using pymatgen's `StructureMatcher` (lattice and atomic positions) [12], using the following matching tolerances: a maximum site distance of 0.3 Å, a maximum relative lattice parameter difference of 20%, and a maximum lattice angle difference of 5°. The resulting set, N_{final}, consists of structures that are following target local motif, unique, and energetically screened. We store these candidates in the energy-ranking database together with their net topology and dimensionality metadata.

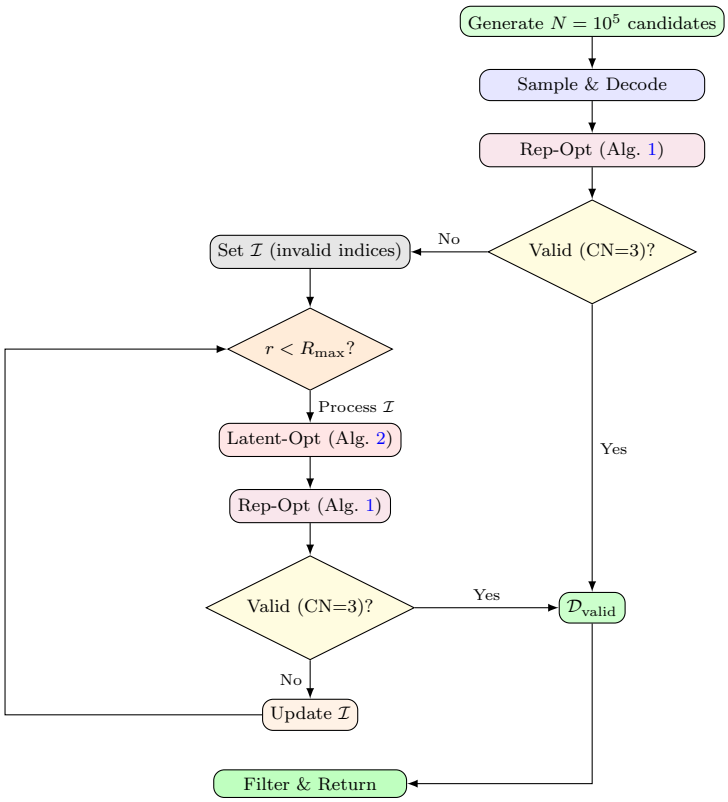

Figure 3. Iterative dual-level refinement workflow used in this study. Motif-valid structures (CN=3) are collected in $\mathcal{D}_{\text{valid}}$; remaining samples ($\mathcal{I}$) undergo latent refinement followed by re-optimization for up to $R_{\max}$ rounds.

4 Results

4.1 Evaluation Metrics

We assess model performance using the following metrics:

$N_{\text{valid\_env}}$ Number of candidates that satisfy the target local-environment criterion (CN=3) after descriptor-driven optimization.

N_{unique} Number of unique structures after removing duplicates based on topology information and pymatgen's `StructureMatcher` after MACE relaxation.

$N_{\text{low\_E}}$ Number of unique structures whose MACE-relaxed energy lies within 0.55 eV/atom of the graphite reference (taken as the ground state).

4.2 Baseline Performance of CVAE Generation

	Base VAE	**Conditional VAE**
N_{Total}	100000	100000
N_{Valid_env}	18248	23372
N_{Unique}	4878	5695
N_{Low_E}	508	571

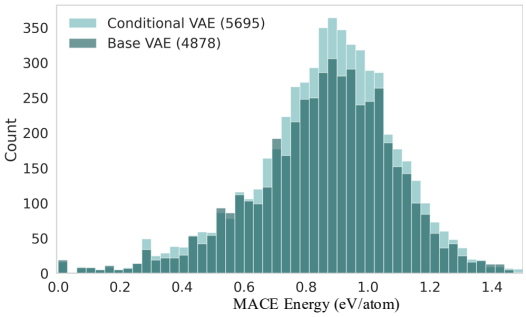

Figure 4. Baseline generation comparison between a VAE and a CVAE. Both models generate 100k candidates and are evaluated using the same post-processing pipeline (CN=3 motif check, deduplication, and MACE energy screening). *Left:* Summary counts of motif-valid structures ($N_{\text{valid\_env}}$), unique structures (N_{unique}), and low-energy candidates ($N_{\text{low\_E}}$). *Right:* MACE-relaxed energy distributions (eV/atom) shown as *counts* for the screened unique structures; both models use identical binning and exhibit strongly overlapping energy profiles.

We compare the conditional VAE (CVAE) against a baseline VAE using the same post-processing pipeline (Section 3.5). To make the comparison as controlled as possible, we

generate $100k$ candidates from each model and reuse the same condition multiset $C = (\text{spg}, \text{wps})$ when sampling the CVAE, so both models are evaluated under identical symmetry constraints.

Figure 4 shows that conditioning improves downstream yield. Relative to the baseline VAE, the CVAE produces more valid (CN=3) structures (23,372 vs 18,248), retains more unique candidates after deduplication (5,695 vs 4,878), and yields more low-energy structures after MACE screening (571 vs 508).

The MACE energy histograms (eV/atom) for the screened unique structures overlap strongly across the full range, suggesting that the two models produce broadly similar energy profiles after screening.

4.3 Runtime Scalability of Batch-wise Pre-relaxation

We nextbenchmark the throughput of the proposed GPU-batched descriptor based optimization in the representation space R against our previous CPU-based implementation in LEGOxtal. As shown in Fig. 5, the PyTorch-GPU pipeline achieves approximately a 5× reduction in wall-clock time (∼5 min to ∼1 min per 1k candidates), enabling high-throughput descriptor-guided refinement for large generative candidate pools.

	CPU-SciPy	**GPU-PyTorch**
Device	AMD 96-core	Nvidia H100
Optimisation	Nelder–Mead, L-BFGS-B	AdamW
Differentiability	No	End-to-end (autodiff)
Batching	Python multiprocessing	GPU vectorised batching
Time (1k samples)	∼5 min	∼1 min
$N_{\text{valid\_env}}$	18,248	12,874
N_{unique}	4,878	4,817
$N_{\text{low\_E}}$	508	453

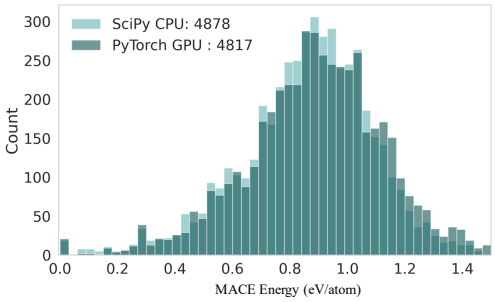

Figure 5. GPU acceleration of representation-space pre-relaxation. *Left:* CPU–SciPy (prior LEGOxtal implementation) versus GPU–PyTorch (this work), including device, optimizer, batching strategy, throughput (time per 1k candidates), and post-screening counts. *Right:* MACE-relaxed energy histograms (eV/atom) for the screened unique structures, shown as counts with identical binning.

In addition to the computational speedup, we compare the evaluation metrics to ensure that the GPU implementation yields comparable screening outcomes. The number of screened unique structures is nearly identical between the two implementations ($N_{\text{unique}} = 4878$ for CPU–SciPy versus 4817 for GPU–PyTorch), and the corresponding MACE energy histograms (eV/atom; right panel) show strong overlap across the full range. There is, however, a noticeable difference in the number of valid (CN=3) structures ($N_{\text{valid\_env}} = 18248$ for CPU–SciPy versus 23372 for GPU–PyTorch). We attribute this to differences in optimization dynamics: independent CPU runs use SciPy's L-BFGS-B with finite-difference gradients, while the GPU implementation employs batched Adam updates with automatic differentiation. Since the final sets of unique screened structures and their energy distributions are highly similar, this difference is acceptable for our purposes. Overall, the results demonstrate that the proposed GPU implementation substantially improves throughput while maintaining a comparable post-screening energy profile.

4.4 Iterative Refinement Performance

5 Conclusions

Acknowledgments

Funding

Data availability

Author contributions

References

[1] Hoffmann R, Kabanov A A, Golov A A and Proserpio D M 2016 *Angew. Chem. Int. Ed.* **55** 10962–10976

[2] Zhu Q, Kang B and Parrish K 2022 *MRS communications* **12** 686–691

[3] Patki N, Wedge R and Veeramachaneni K 2016 The synthetic data vault *IEEE International Conference on Data Science and Advanced Analytics (DSAA)* pp 399–410

Cumulative run	N_{Valid_env}	N_{Unique}	N_{Low_E}	Time (h)
1	15 076	4 806	412	1.66
1–2	29 837	7 581	675	4.36
1–3	40 426	9 647	872	6.66
1–4	48 308	11 546	1 014	8.56

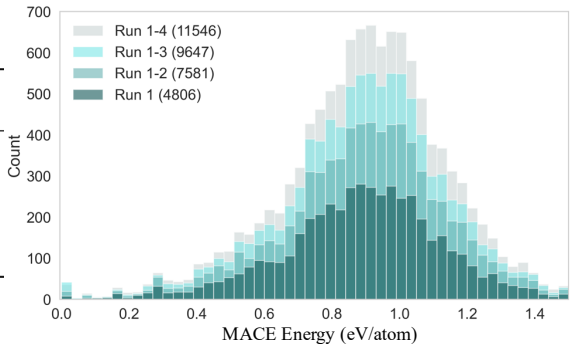

Figure 6. Iterative refinement performance and energy distributions. Left: Cumulative metrics showing progressive improvement in valid structures (CN=3), unique structures, and low-energy candidates. Right: MACE energy distributions showing cumulative growth of validated structures across refinement runs.

[4] Harvey W, Naderiparizi S and Wood F 2021 *arXiv preprint arXiv:2102.12037*

[5] Xuan G, Zhang W and Chai P 2001 Em algorithms of gaussian mixture model and hidden markov model *Proceedings 2001 international conference on image processing (Cat. No. 01CH37205)* vol 1 (IEEE) pp 145–148

[6] Mao A, Mohri M and Zhong Y 2023 Cross-entropy loss functions: Theoretical analysis and applications *International conference on Machine learning* (pmlr) pp 23803–23828

[7] Kingma D P and Welling M 2013 *arXiv preprint arXiv:1312.6114*

[8] Bartók A P, Kondor R and Csányi G 2013 *Phys. Rev. B* **87** 184115

[9] Paszke A, Gross S, Chintala S, Chanan G, Yang E, DeVito Z, Lin Z, Desmaison A, Antiga L and Lerer A 2017

[10] Zoubritzky L and Coudert F X 2022 *SciPost Chem.* **1** 005

[11] Batatia I, Kovacs D P, Simm G N C, Ortner C and Csanyi G 2022 MACE: Higher order equivariant message passing neural networks for fast and accurate force fields *Advances in Neural Information Processing Systems* ed Oh A H, Agarwal A, Belgrave D and Cho K URL https://openreview.net/forum?id=YPpSngE-ZU

[12] Ong S P, Richards W D, Jain A, Hautier G, Kocher M, Cholia S, Gunter D, Chevrier V L, Persson K A and Ceder G 2013 *Computational Materials Science* **68** 314–319