npj | computational materials

Article

Published in partnership with the Shanghai Institute of Ceramics of the Chinese Academy of Sciences

https://doi.org/10.1038/s41524-026-02140-8

Symmetry-aware Bayesian flow networks for crystal generation

Laura Ruple[1], Luca Torresi[1,2], Henrik Schopmans[1] & Pascal Friederich[1,2] ✉

The discovery of new crystalline materials is essential to scientific and technological progress. However, traditional trial-and-error approaches are inefficient due to the vast search space. Recent advancements in machine learning have enabled generative models to predict new stable materials by incorporating structural symmetries and to condition the generation on desired properties. In this work, we introduce SymmBFN, a novel symmetry-aware Bayesian Flow Network (BFN) for crystalline material generation that accurately reproduces the distribution of space groups found in experimentally observed crystals. SymmBFN substantially improves efficiency, generating stable structures at least one order of magnitude faster than the next-best method, at similar or even superior quality. Furthermore, we demonstrate its capability for property-conditioned generation, enabling the design of materials with tailored properties. Our findings establish BFNs as an effective tool for accelerating the discovery of crystalline materials.

The discovery and design of novel materials are essential for advancing technologies in areas such as energy, electronics, and sustainability. Recent developments in machine learning are reshaping the field of materials research in diverse and transformative ways[1]. Machine learning models have been applied to enhance conventional screening approaches[2,3]. At the same time, advancements in generative modeling show great potential for accelerating materials design by proposing novel and realistic candidates for crystal structures with targeted properties[4,5].

An early demonstration of the effectiveness of generative models for crystalline materials is CDVAE[6], which employs a variational autoencoder to generate composition, lattice parameters, and the number of atoms in the unit cell. Then, the atom coordinates are randomly initialized and iteratively denoised using score-matching[7]. Subsequent works[8–10] have replaced the autoencoder with a purely diffusion-based approach, jointly diffusing lattice parameters, fractional coordinates, and atom types. This approach effectively captures crystal geometries as a whole, leading to an improved quality of the generated structures. Various other generative modeling techniques have been applied to crystals, including Riemannian flow matching[11], large language models[12–14], and normalizing flows[15]. While these models have demonstrated the capability to generate novel and stable structures, an important aspect that has often been neglected is the incorporation of space group symmetry. A large proportion of the samples generated with models such as DiffCSP[8] and CDVAE belong to the low-symmetry space group P1, which is rarely observed in nature. To alleviate this issue, models that explicitly incorporate space group symmetries into the generation process have been recently proposed along several complementary directions. One class of approaches enforces explicit symmetry constraints during generation: DiffCSP++[16] extends diffusion-based generation by incorporating lattice and Wyckoff constraints directly into the sampling process, while SGEquiDiff[17] and Space Group Conditional Flow Matching[18] similarly condition the generative dynamics on a target space group to ensure that atomic coordinates evolve on symmetry-consistent manifolds. Second, a growing family of methods adopts symmetry-reduced, discrete representations based on Wyckoff positions: CrystalFormer[19] and WyFormer[20] employ autoregressive transformers over symmetry-aware tokens, WyckoffDiff[21] applies diffusion in this discrete symmetry-adapted space. Collectively, these approaches move beyond unconstrained atom-wise generation by operating directly in symmetry-adapted representations. A third class focuses on modeling the asymmetric unit jointly with symmetry operations, as exemplified by SymmCD[22], a diffusion model which generates only the minimal set of representative atoms together with the corresponding group actions required to reconstruct the full crystal.

Diffusion-based methods have proven effective in predicting realistic materials. However, they present two main limitations when applied to crystal generation: First, handling heterogeneous variable types (continuous atomic coordinates and lattice parameters, alongside categorical atom types and site symmetries) typically requires additional modeling choices or representations, which can complicate the formulation. A second drawback is that generating high-quality samples requires a significant number of integration steps, which results in high computational costs.

Bayesian Flow Networks (BFN) are a novel class of generative models proposed by Graves et al.[23], which generate samples through an

[1]Institute for Anthropomatics and Robotics, Karlsruhe Institute of Technology, Karlsruhe, Germany. [2]Institute of Nanotechnology, Karlsruhe Institute of Technology, Karlsruhe, Germany. ✉e-mail: pascal.friederich@kit.edu

iterative procedure similar to the reverse process used in diffusion. Unlike diffusion models, BFNs operate directly on the parameters of the data distribution, rather than its noisy samples. This formulation provides a principled way to handle heterogeneous variable types (continuous, discrete, and categorical) within a single framework, without requiring separate modeling components or discretization schemes. BFNs have been recently applied to the domain of three-dimensional molecular generation[24], where they demonstrated a superior trade-off between efficiency and quality, achieving a significant speedup over diffusion models. CrysBFN[25] recently introduced a periodic Bayesian flow network for crystal generation by extending Bayesian flow models to non-Euclidean manifolds through an entropy conditioning mechanism designed to capture periodic symmetries in crystal structures.

In this study, we present SymmBFN, a novel adaptation of BFNs for the generation of crystalline materials using a symmetry-aware representation of crystal structures first introduced by Levy et al.[22]. Our approach provides a simple and efficient formulation for jointly modeling distributions over fractional coordinates, atom types, unit cell parameters, and site symmetries. Specifically, the proposed method (i) provides a principled and efficient framework for modeling heterogeneous variables within a single formulation, (ii) accurately reproduces the distribution of space groups observed in real-world materials, and (iii) achieves a speed improvement in the generation process of up to two orders of magnitude compared to state-of-the-art diffusion- and flow-based methods. This efficiency gain arises from (a) the BFN formulation, which in the context of crystal structure generation, requires fewer iterative denoising steps and (b) the use of a symmetry-aware representation that reduces the effective dimensionality of the generation space. Additionally, we develop a method to condition the generation process on additional desired properties. Our results demonstrate the potential of BFNs as a tool for accelerating the design of crystalline materials with targeted properties. In contrast to CrysBFN, SymmBFN includes the Euclidean invariances by operating in a canonical representative subspace. This removes the need to reformulate BFNs for non-Euclidean data, as the periodicity is implicitly enforced by the unique canonical representation[26]. As a result, our model avoids the complexity of extending BFNs to non-Euclidean spaces while leveraging a compact representation that enhances sampling efficiency and more faithfully captures the symmetries inherent in real-world materials.

Results

Metrics

Following prior work[6,8,11], we use several property-based metrics to benchmark our proposed model against existing approaches. First, the Wasserstein distances between the test set and 1000 generated structures for the distributions of the density ρ and the distribution of the number of unique elements in the unit cell. Following the work of Levy et al.[22], we also compute the Jensen-Shannon distance between the space group distribution of 1000 generated structures and that of the test set, to determine whether the models accurately capture the real-world distribution of space groups. We classify the space groups with pymatgen's `SpacegroupAnalyzer`[27] using a tolerance of 0.1. The most informative metric for de novo generation is the stability of the generated structures. For de novo generation, we aim to generate structures that are stable (S), unique within the generated set of structures (U), and novel with respect to the training dataset (N). The S.U.N. rate, a metric introduced by Zeni et al.[9], represents the proportion of generated structures that meet these criteria. In this work, all stability metrics are consistently evaluated using CHGNet[28], as not all prior works report DFT-based evaluations, leading to systematically different absolute values compared to DFT-based results. We include a more detailed explanation of these metrics in Section A of the Supplementary Information.

Finally, to evaluate the model efficiency, we introduce two novel cost metrics that quantify the average computational time required to generate a stable and S.U.N. material, respectively. More details on these metrics are in Section A of the Supplementary Information.

Dataset

All models discussed in this work, including our proposed method, were trained on the MP-20 dataset, a subset of the Materials Project database[29]. This dataset comprises 40,476 crystal structures with up to 20 atoms per unit cell. We adopt the 60-20-20 train-validation-test split initially introduced by Xie et al.[6] and subsequently used in all other studies we compare against. The formation energy per atom and bandgap, used for the property-conditioned generation, is computed with M3GNet[30] for each structure in the dataset. We further evaluate our model on the Perov-5[6,31] and MPTS-52[32] datasets. Perov-5 comprises 18,928 Perovskite materials, each with five atoms per unit cell, sharing a common structure but differing in composition across 56 elements. Most structures in Perov-5 are unstable; therefore, stability and S.U.N. metrics are not applicable for models trained on this dataset. MPTS-52 is a more challenging subset of the Materials Project, containing up to 52 atoms per cell. It comprises 40,476 samples split chronologically into 27,380 training, 5000 validation, and 8096 test structures.

CSP

We also trained our model on the crystal structure prediction (CSP) task, a simplified variant of the de novo generation. Since our symmetry-aware model requires sampling a space group at the beginning of generation, the CSP task differs slightly for SymmBFN compared to the baselines that do not incorporate crystal symmetries explicitly. More details and results for CSP are in Section A of the Supplementary Information and Supplementary Table 1.

De novo generation

We benchmark SymmBFN using the metrics described above, comparing it against several state-of-the-art models for de novo crystal generation (see Table 1). These models include DiffCSP[8] and FlowMM[11], both of which utilize the standard unit cell representation, as well as DiffCSP++[16], SymmCD[22], and CrystalFormer[19], which support space group-conditioned generation. Of these models, only DiffCSP and CrysBFN have been trained on the Perov-5 dataset, while SymmCD is the only baseline trained on MPTS-52. To ensure a fair comparison on larger systems, we additionally trained CrysBFN on MPTS-52 and selected the number of sampling steps that achieved optimal performance. The Steps column reports the number of sampling steps that are used to generate a sample for each method. The reported Wasserstein distances are taken directly from the respective original publications. For DiffCSP, DiffCSP ++, and CrystalFormer, we computed the other metrics using structures generated from the published checkpoints, while for FlowMM and CrysBFN, we used structures provided directly by the authors. All stability and S.U.N. metrics are evaluated using 10,000 generated structures per model to ensure a fair and consistent comparison across methods. Following Levy et al.[22], we also evaluate our model with the same metrics by sampling only from the 10 most common space groups in the MP-20 dataset, enabling a more balanced comparison with methods that do not account for crystal symmetries, while still covering a substantial portion of the data distribution.

SymmBFN proves to be competitive with the other generative models on all property metrics. Notably, only DiffCSP++, SymmCD, CrystalFormer, and SymmBFN are capable of accurately modeling crystal symmetries, as evidenced by the Jensen-Shannon distance between the space group distributions of the generated structures and the test set (We note that this distance is not zero even for these models, as remaining differences arise from discrepancies between the training and test set distributions). This underscores the importance of incorporating crystal symmetries into the generation process. SymmBFN achieves competitive results with only 100 sampling steps, demonstrating exceptional sampling efficiency compared to other generative models. For the MPTS-52 dataset, the more compact asymmetric unit cell representation proves particularly useful in substantially reducing the size of the required computational graph compared to methods that do not leverage symmetries. On this dataset, SymmBFN

Table 1 | Results on the de novo generation: property metrics

Dataset	Method	Steps	Property ↓			Stability ↑	S.U.N. ↑
			d_ρ	d_{elem}	d_G	(%)	(%)
Perov-5	DiffCSP	1000	0.111	0.013	-	-	-
	CrysBFN	1000	0.073	**0.010**	-	-	-
	SymmBFN (ours)	100	**0.070**	0.028	-	-	-
MP-20	DiffCSP	1000	0.350	0.340	0.444	10.29	7.88
	DiffCSP++	1000	0.235	0.375	0.160	11.03	8.29
	SymmCD	1000	0.23	0.40	0.164	9.34	6.86
	FlowMM	500	**0.075**	0.079	0.545	9.00	6.84
	CrysBFN	1000	0.207	0.163	0.309	**15.71**	**8.81**
	CystalFormer	-	0.173	0.166	0.153	12.31	8.28
	SymmBFN (ours)	100	0.083	**0.061**	**0.149**	12.11	7.24
MP-20 (10 SGs)[a]	SymmCD	1000	0.53	0.16	**0.469**	10.92	7.59
	SymmBFN (ours)	100	**0.290**	**0.054**	0.471	**15.33**	**8.89**
MPTS-52	SymmCD	1000	0.844	**0.317**	0.274	5.97	4.62
	CrysBFN	100	2.677	1.160	0.618	6.14	4.50
	SymmBFN (ours)	100	**0.694**	0.423	**0.252**	**10.14**	**6.12**

[a]Space groups with the numbers 2, 12, 14, 62, 63, 139, 166, 194, 221 and 225.
Best results are shown in bold and second-best results are underlined.

Table 2 | Results on the de novo generation: cost metrics

Dataset	Method	Steps	Time	Stability Cost ↓	SUN Cost ↓
			Seconds/ Sample	Seconds	Seconds
MP-20	DiffCSP	1000	0.482	4.869	6.117
	DiffCSP++	1000	1.573	14.261	18.975
	SymmCD	1000	0.514	5.503	7.920
	FlowMM	500	0.275	2.957	3.667
	CrysBFN	1000	0.414	2.635	4.699
	CrystalFormer	-	0.688	5.589	8.309
	SymmBFN (ours)	100	**0.007**	**0.059**	**0.097**
MP-20 (10 SGs)[a]	SymmCD	1000	0.514	4.707	6.772
	SymmBFN (ours)	100	**0.007**	**0.046**	**0.079**
MPTS-52	SymmCD	1000	0.790	13.232	17.099
	CrysBFN	100	0.096	1.564	2.133
	SymmBFN (ours)	100	**0.008**	**0.079**	**0.131**

[a]Space groups with the numbers 2, 12, 14, 62, 63, 139, 166, 194, 221 and 225.
Best results are shown in bold.

shows a clear improvement over SymmCD and CrysBFN in terms of stability and S.U.N. rate.

In Table 2, we present metrics regarding the computational cost of SymmBFN and the baselines. SymmBFN's computational cost is substantially lower than that of any other method. This makes our approach ideal for large-scale materials discovery screening studies. The speed advantage of our model can be attributed to the smaller computational graph, its simpler mathematical framework, and its sampling efficiency. An evaluation of the impact of varying the number of sampling steps on network performance and additional details on the selected hyperparameter values are provided in Section A of the Supplementary Information and Supplementary Tables 2 and 3. For completeness, we include additional metrics based on heuristics used in earlier methods in Supplementary Table 4. While some diffusion- and flow-based methods achieve higher scores on these heuristic validity metrics, we note that they are less reflective of synthesizability than stability and S.U.N. metrics, on which SymmBFN performs competitively. In Supplementary Fig. 1, we show examples of structures generated by SymmBFN.

Property-conditioned generation

We evaluate our property-conditioned model using the formation energy per atom, defined as the energy required to form a crystal structure from its constituent elements, normalized by the number of atoms in the unit cell[29]. In a second experiment, we also condition the model on the bandgap, the energy difference between the valence and conduction bands, which governs a material's electrical conductivity. For these experiments, we use the same hyperparameters as the model without property conditioning. During generation, we specify three different target values for the formation energy per atom: one at the mode, one in the tail, and one outside of the distribution of training set values, to demonstrate the ability of the model to generate structures across diverse target values. We do the same for the bandgap; however, the dataset distribution is highly imbalanced, with most structures having a bandgap value of 0.0. To address this, we specify three target values during generation: one at 0 (the mode), and two others with only a few examples in the dataset. For each target, we generate 1000 samples and relax them using the CHGNet neural network. For all metastable structures ($E_{\text{hull}} \leq 0.1$), we then calculate the formation energy per atom or bandgap using M3GNet[30]. We provide the stability and metastability rates for each experiment in Supplementary Tables 5 and 6. The results, shown in Fig. 1, indicate that the model reliably generates structures aligned with the specified target properties. As the target values move further from the well-represented regions of the training distribution, the mean property of the generated structures tends to deviate more from the target, accompanied by increased variance. Nonetheless, even for sparsely represented targets, such as a formation energy of −4eV/atom, the model remains effective at proposing stable structures with the desired property. Similarly, despite the pronounced imbalance in the bandgap distribution, the model demonstrates strong performance, successfully generating structures with specified bandgap values, including those with limited representation in the training data.

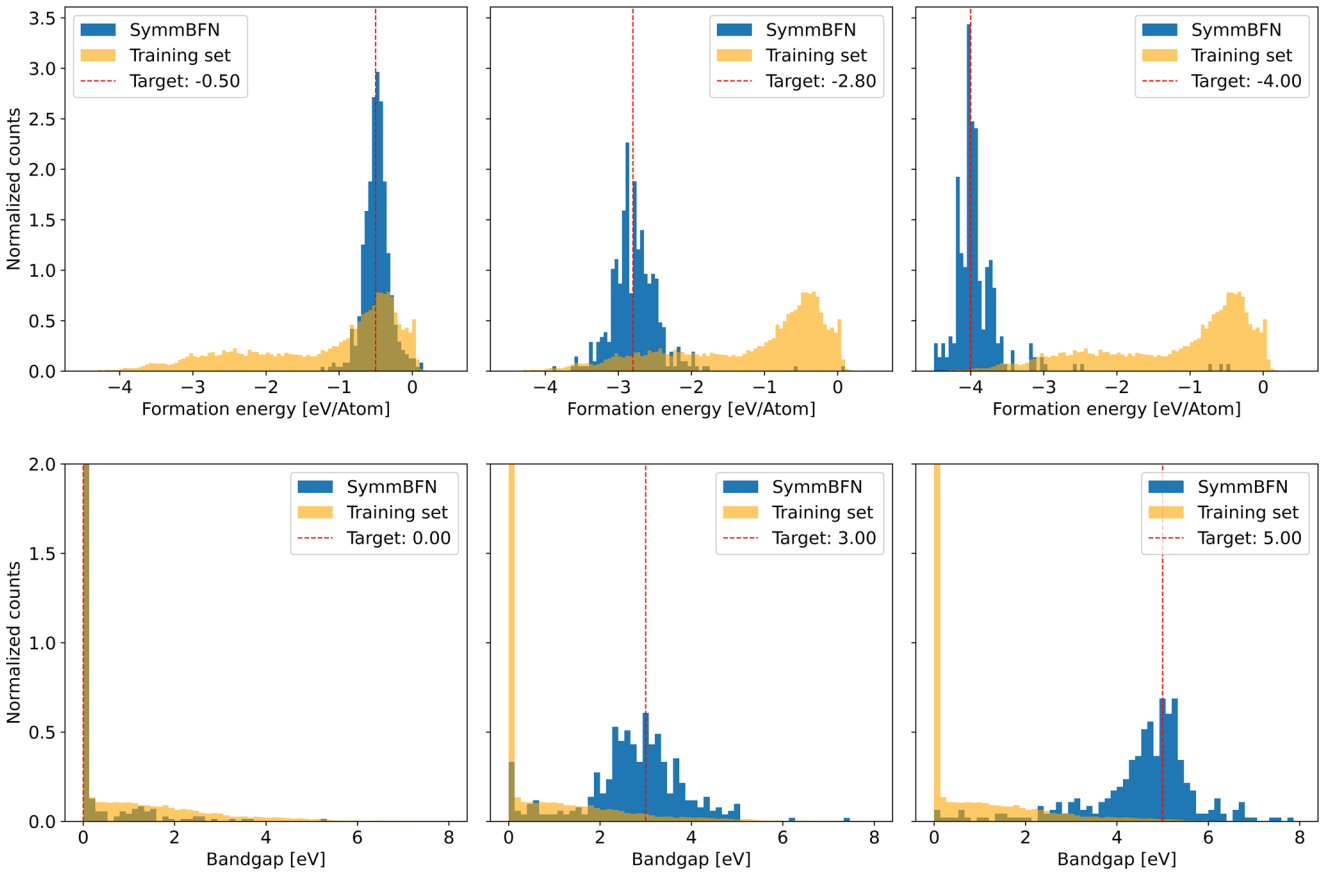

Fig. 1 | Results for the property-conditioned generation for three different target values. The histograms show the distributions of the formation energy per atom and bandgap of the generated structures in blue and of the training set in orange. The dashed red line represents the target of the generation. Due to the overrepresentation of values at 0.0 eV in the training set, the y-axis of the bandgap plots is cut at 2 for better visibility.

Ablation study

To quantify the computational benefits of each component of SymmBFN—(i) the reduced, symmetry-aware representation and (ii) the Euclidean BFN formulation—we performed an ablation study comparing SymmBFN with two models that operate on the full crystal structure: (i) a baseline variant of SymmBFN that uses the same Euclidean BFN but processes all atoms in the conventional cell, and (ii) CrysBFN, which also operates on the full structure but employs a non-Euclidean BFN formulation (see Table 3).

The Euclidean formulation alone yields a consistent ~1.75× speed-up over CrysBFN's non-Euclidean BFN, while the asymmetric-unit representation offers gains that grow with system size, exceeding 6× faster inference for large MPTS-52 structures. Correspondingly, SymmBFN exhibits a scaling factor of only 1.14× from MP-20 to MPTS-52, compared to 3.31× for the Baseline and 3.39× for CrysBFN, making the combination of symmetry-aware representation and Euclidean BFN formulation substantially more efficient in scaling to larger crystal structures.

Table 3 | Sample generation time (ms) and scaling factor from MP-20 to MPTS-52 for 100 generation steps

Method	MP-20	MPTS-52	Scaling
SymmBFN	7	8	1.14×
Baseline	16	53	3.31×
CrysBFN	28	95	3.39×

Discussion

In this work, we introduced SymmBFN, a novel Bayesian flow network for the generation of crystal structures. By explicitly incorporating the crystal symmetries into the generation process, we can generate crystals more consistent with those naturally observed. Furthermore, by allowing conditioning on specific target properties, SymmBFN facilitates the discovery of structures tailored to desired applications. In contrast to prior approaches based on diffusion models, the BFN framework enables the combination of all target variables, including the site symmetry groups and elements of the individual atoms, into a unified framework. SymmBFN achieves competitive results for the generation of stable and novel structures while offering a substantial speedup, more than 40 times faster than previous generative models. This establishes BFNs as an effective framework for crystal generation, eliminating the sampling bottleneck of previous approaches. The demonstrated efficiency and versatility of SymmBFN position it as a promising tool for accelerating materials design. It enables the generation of stable, property-targeted structures with reduced computational costs, making large-scale screening studies possible. Future work will focus on extending SymmBFN to multi-property conditioning for properties relevant to practical applications and on exploring experimental synthesis, as successful synthesis remains the ultimate goal and cannot be fully captured by computational metrics alone.

Methods

SymmBFN employs BFNs to generate crystals constrained to any specified space group by modeling only the asymmetric unit instead of the whole unit cell, resulting in a smaller computational graph and inherent enforcement of the Euclidean invariances. SymmBFN models in a single framework the lattice parameters, the site symmetries, and the atoms in the asymmetric unit, and then reconstructs the entire unit cell in a post-processing step. The reduced, asymmetric unit representation is described in more detail in Section B of the Supplementary Information and Supplementary Fig. 2.

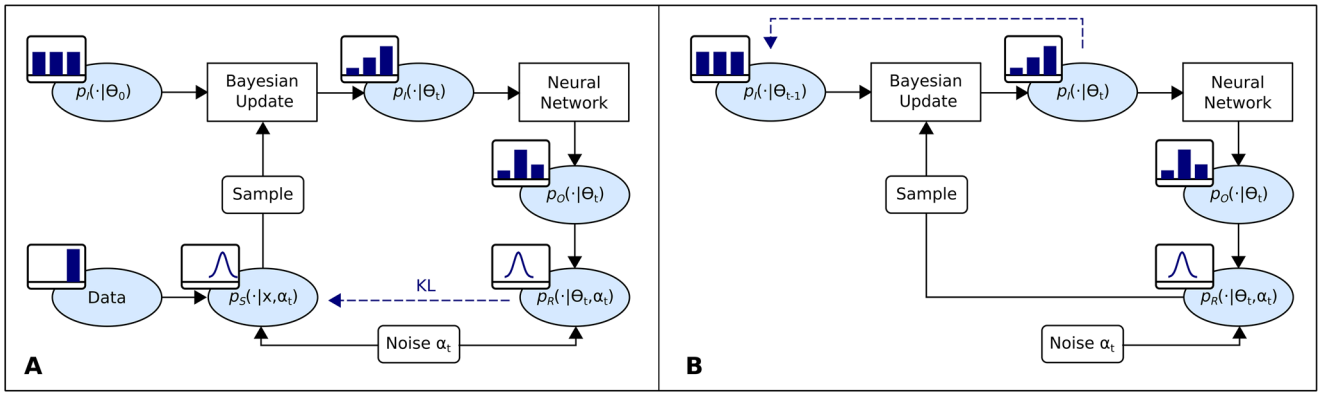

Fig. 2 | Bayesian Flow Network. Illustration of BFN training procedure at the left (**A**) and of the sampling procedure at the right (**B**).

Bayesian flow networks

BFNs[23] generate data through an iterative process that combines the strengths of Bayesian inference and deep learning. This process (see Fig. 2) involves transforming the parameters of a distribution representing the data from an uninformative prior to increasingly confident posteriors, alternating between two steps: (i) The parameters of a set of independent distributions p_I, each representing a variable, are updated using Bayesian inference. (ii) These updated parameters are passed to a neural network, incorporating contextual information to output a joint distribution p_O. The mathematical framework of BFNs is described in Section B of the Supplementary Information.

Incorporating symmetries

To ensure that the probability distribution over generated crystals is invariant under the actions of the Euclidean group $\mathbb{E}$, which includes both global rotations and translations of the crystal, we map all elements of the orbit $\{g \cdot x | g \in \mathbb{E}\}$ of a given structure x to a unique canonical representative[22,26]:

(i) Rotational invariance: We model the atom coordinates using fractional coordinates with respect to unit cell axes $\mathbf{l}_i$, $\widetilde{\mathbf{x}} = \sum_{i=1}^{3} x_i \mathbf{l}_i \in \mathbb{R}^3$. We model the lattice using the 6 values k_i, $i = 1,\ldots6$ (see Section B of the Supplementary Information). These parameters define a canonical lattice by removing arbitrary rotational degrees of freedom. This projection to a canonical lattice ensures that our model operates within a rotationally invariant subspace. Accordingly, the Bayesian flow network is defined only over these canonical lattice configurations.

(ii) Periodic translation invariance: We additionally want our model to be invariant with respect to periodically wrapped shifts of all atomic fractional coordinates, which corresponds to a shift of the whole crystal structure. For a given set of fractional coordinates, we consider the orbit under the group of lattice-periodic shifts and select a unique representative using the PyXtal library[33]. In PyXtal, the selection of fractional coordinates among duplicated atoms within a Wyckoff site is deterministic and follows the conventions set by the International Tables for Crystallography. This guarantees that configurations related by such translations are treated identically. Instead of using the fractional coordinates of the whole unit cell directly, we operate on the asymmetric unit, making our algorithm more efficient and more reliable in generating symmetric structures.

By operating exclusively within these canonical subspaces, we ensure that the generator is invariant under both rotations and periodic translations, $p(x) = p(g \cdot x), \forall g \in \mathbb{E}$.

Fractional coordinates

We apply the BFN instantiation for continuous data on the fractional coordinates $\mathbf{x}$ of the atoms in the asymmetric unit. The *input distribution* is set as an isotropic Gaussian $p_I(\mathbf{x}|\boldsymbol{\theta}^{\mathbf{x}}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \rho^{-1}\boldsymbol{I})$ with parameters $\boldsymbol{\theta}^{\mathbf{x}} = \{\boldsymbol{\mu}, \rho\}$, and the prior in $t = 0$ is initialized as $\boldsymbol{\theta}_0^{\mathbf{x}} = \{\boldsymbol{0}, 1\}$. Similarly, the *sender distribution* takes the form of an isotropic Gaussian: $p_S(\mathbf{y}|\mathbf{x};\alpha) = \mathcal{N}(\mathbf{y}|\mathbf{x}, \alpha^{-1}\boldsymbol{I})$. Assuming both the *input* and *sender distribution* are isotropic Gaussians, Graves et al.[23] derive the *Bayesian update functionh* as

$$\rho_i = \rho_{i-1} + \alpha, \boldsymbol{\mu}_i = \frac{\boldsymbol{\mu}_{i-1}\rho_{i-1} + \mathbf{y}\alpha}{\rho_i}. \tag{1}$$

Marginalizing over $\mathbf{y} \sim \mathcal{N}(\mathbf{y}|\mathbf{x}, \alpha^{-1}\boldsymbol{I})$ yields the *Bayesian update distribution* for continuous data

$$p_U(\boldsymbol{\theta}_i|\boldsymbol{\theta}_{i-1}, \mathbf{x}; \alpha) = \mathcal{N}\left(\boldsymbol{\mu}_i | \frac{\alpha\mathbf{x} + \boldsymbol{\mu}_{i-1}\rho_{i-1}}{\rho_i}, \frac{\alpha}{\rho_i^2}\boldsymbol{I}\right). \tag{2}$$

The *Bayesian update* can be extended to continuous time by introducing the accuracy schedule $\beta(t) = \int_{t'=0}^{t} \alpha(t')dt'$. The accuracy schedule for continuous data is defined as $\beta(t) = \sigma_{\mathbf{x}}^{-2t} - 1$, where $\sigma_{\mathbf{x}}$ is the empirically chosen standard deviation of the input distribution at $t = 1$. With $\gamma(t) = \frac{\beta(t)}{1+\beta(t)} = 1 - \sigma_{\mathbf{x}}^{2t}$, the *Bayesian flow distribution* is $p_F(\boldsymbol{\theta}^{\mathbf{x}}|\mathbf{x}; t) = p_U(\boldsymbol{\theta}^{\mathbf{x}}|\boldsymbol{\theta}_0^{\mathbf{x}}, \mathbf{x}, \beta(t))$, which for continuous data can be derived to be

$$p_F(\boldsymbol{\theta}^{\mathbf{x}}|\mathbf{x}; t) = \mathcal{N}\left(\boldsymbol{\mu}|\gamma(t)\mathbf{x}, \gamma(t)(1-\gamma(t))\boldsymbol{I}\right). \tag{3}$$

The neural network is trained to predict an estimate $\widehat{\boldsymbol{\epsilon}}(\boldsymbol{\theta}^{\mathbf{x}}, t)$ of the Gaussian noise vector $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ which was used to generate the input $\boldsymbol{\mu}$ provided to the network. Given this estimate, the *output distribution* is

$$p_O(\mathbf{x}|\boldsymbol{\theta}^{\mathbf{x}}; t) = \delta(\mathbf{x} - \widehat{\mathbf{x}}(\boldsymbol{\theta}^{\mathbf{x}}, t)) \text{ , where } \widehat{\mathbf{x}}(\boldsymbol{\theta}^{\mathbf{x}}, t) = \frac{\boldsymbol{\mu}}{\gamma(t)} - \sqrt{\frac{1-\gamma(t)}{\gamma(t)}}\widehat{\boldsymbol{\epsilon}}(\boldsymbol{\theta}^{\mathbf{x}}, t). \tag{4}$$

The estimates $\widehat{\mathbf{x}}(\boldsymbol{\theta}^{\mathbf{x}}, t)$ of the fractional coordinates are then wrapped around the interval $[0, 1)^3$ by applying the modulus operation. Finally, the loss function is obtained as

$$L^{\infty}(\mathbf{x}) = -ln\sigma_{\mathbf{x}}\mathbb{E}_{\substack{t \sim U(0,1), \\ p_F(\boldsymbol{\theta}^{\mathbf{x}}|\mathbf{x};t)}} \frac{\left\|\mathbf{x} - \widehat{\mathbf{x}}(\boldsymbol{\theta}^{\mathbf{x}}, t)\right\|^2}{\sigma_{\mathbf{x}}^{2t}}. \tag{5}$$

Atom types

Consistent with the categorical nature of atom types, we apply the BFN instantiation designed for discrete data. We represent the atom type of each of the D atoms in the asymmetric cell as $\mathbf{a} = \left(a^{(1)}, \ldots, a^{(D)}\right) \in \{1, K\}^D$, with K being the highest atomic number in the dataset. The *input distribution* for the atom types is defined as the categorical distribution $p_I(\mathbf{a}|\boldsymbol{\theta}^{\mathbf{a}}) = \prod_{d=1}^{D} \theta_{a^{(d)}}^{(d)}$ with parameters $\boldsymbol{\theta}^{\mathbf{a}} = \left(\theta_1^{(1)}, \theta_2^{(1)}, \ldots, \theta_K^{(D)}\right) \in \mathbb{R}^{KD}$,

where $\theta_k^{(d)}$ is the probability for atom d to be of type k. The prior is set to the uninformative uniform distribution, where all atom types are assigned equal probability $1/K$.

The *sender distribution* is then defined as

$$p_S(\mathbf{y}|\mathbf{a};\alpha) = \mathcal{N}\left(\mathbf{y}|\alpha\left(K\mathbf{e}_\mathbf{a} - \mathbf{1}\right), \alpha K\boldsymbol{I}\right), \tag{6}$$

where $\mathbf{e}_\mathbf{a} = \left(\mathrm{e}_1^{(1)}, \ldots, \mathrm{e}_K^{(D)}\right) \in \mathbb{R}^{KD}$ and $\mathrm{e}_k^{(d)} = \delta_{ka^{(d)}}$. With the *Bayesian update function* for discrete data $h(\boldsymbol{\theta}_{i-1}, \mathbf{y}, \alpha) = \frac{e^\mathbf{y}\boldsymbol{\theta}_{i-1}}{\sum_{k=1}^K e^{\mathbf{y}_k}(\boldsymbol{\theta}_{i-1})_k}$, the *Bayesian update distribution* $p_U(\boldsymbol{\theta}^\mathbf{a}|\boldsymbol{\theta}_{i-1}^\mathbf{a}, \mathbf{a}; \alpha)$ is given by

$$\mathbb{E}_{\mathcal{N}\left(\mathbf{y}|\alpha\left(K\mathbf{e}_\mathbf{a}-\mathbf{1}\right),\alpha K\boldsymbol{I}\right)}\delta\left(\boldsymbol{\theta}^\mathbf{a} - \frac{e^\mathbf{y}\boldsymbol{\theta}_{i-1}^\mathbf{a}}{\sum_{k=1}^K e^{\mathbf{y}_k}(\boldsymbol{\theta}_{i-1}^\mathbf{a})_k}\right). \tag{7}$$

With the accuracy schedule defined as $\beta(t) = t^2\beta(1)$, where $\beta(1)$ is a hyperparameter, the *Bayesian flow distribution* for discrete data $p_F(\boldsymbol{\theta}^\mathbf{a}|\mathbf{a}; t)$ is given by

$$\mathbb{E}_{\mathcal{N}\left(\mathbf{y}|\beta(t)\left(K\mathbf{e}_\mathbf{a}-\mathbf{1}\right),\beta(t)K\boldsymbol{I}\right)}\delta\left(\boldsymbol{\theta}^\mathbf{a} - \mathrm{softmax}\left(\mathbf{y}\right)\right). \tag{8}$$

Given the network output $\widehat{\boldsymbol{\epsilon}}_\mathbf{a} = \Psi(\boldsymbol{\theta}^\mathbf{a}, t)$, the *output distribution* can be obtained as

$$p_O(\mathbf{a}|\boldsymbol{\theta}^\mathbf{a}; t) = \prod_{d=1}^D p_O^{(d)}(a^{(d)}|\boldsymbol{\theta}^\mathbf{a}; t), \text{ with } p_O^{(d)}(k|\boldsymbol{\theta}^\mathbf{a}; t) = \left(\mathrm{softmax}(\widehat{\boldsymbol{\epsilon}}_\mathbf{a}^{(d)})\right)_k. \tag{9}$$

Finally, the loss function for discrete data and continuous accuracy schedule $\beta(t)$ is defined as

$$L^\infty(\mathbf{a}) = K\beta(1)\mathbb{E}_{t\sim U(0,1),\ \boldsymbol{\theta}^\mathbf{a}\sim p_F(\boldsymbol{\theta}^\mathbf{a}|\mathbf{a},t)} \| \mathbf{e}_\mathbf{a} - \widehat{\mathbf{e}}(\boldsymbol{\theta}^\mathbf{a}, t)\|^2, \tag{10}$$

where

$$\widehat{\mathbf{e}}(\boldsymbol{\theta}^\mathbf{a}, t) = \left(\widehat{\mathbf{e}}^{(1)}(\boldsymbol{\theta}^\mathbf{a}, t), \ldots, \widehat{\mathbf{e}}^{(D)}(\boldsymbol{\theta}^\mathbf{a}, t)\right), \widehat{\mathbf{e}}^{(d)}(\boldsymbol{\theta}^\mathbf{a}, t) = \sum_{k=1}^K p_O^{(d)}(k|\boldsymbol{\theta}^\mathbf{a}; t)\mathbf{e}_k. \tag{11}$$

Site symmetry groups

For each atom in the asymmetric cell, we generate the site symmetry group by having the model output the index of one of the 13 possible symmetry operations for each of the 15 axes. The model is trained with the categorical BFN for the symmetry operations to generate $\mathbf{S} = \left(s^{(1)}, s^{(2)}, \ldots, s^{(15)}\right) \in \{1, 13\}^{15}$ for each node. The BFN distributions for symmetry operations are defined similarly to those for atomic numbers.

Lattice

We apply the BFN instantiation for continuous data on the lattice vector representation $\mathbf{k} \in \mathbb{R}^6$[16] (for more details see Section B of the Supplementary Information and Supplementary Table 7). To comply with the space group constraint, after the Bayesian updates and network calls, we introduce a masking step. Otherwise, the distributions are defined analogously to the fractional coordinate generation.

Neural network

For our framework, we employ the graph neural network architecture proposed by Jiao et al.[8], based on the EGNN model[34]. The network $\Psi(\boldsymbol{\mu}_\mathbf{k}, \boldsymbol{\mu}_\mathbf{x}, \boldsymbol{\theta}^\mathbf{a}, \boldsymbol{\theta}^\mathbf{S}, t, G)$ predicts the scores for the output distributions after N message-passing layers on a fully-connected graph. More details about the implementation can be found in Section B of the Supplementary Information. It is worth noting that since SymmBFN utilizes a canonical reference system for crystal structures, namely the unit cell axes, there is no need to enforce equivariance within the neural network[22].

Sampling

To generate new structures, we first sample the space group G and, conditioned on G, the number of atoms in the asymmetric unit from the dataset distribution. Then, starting with prior parameters $\boldsymbol{\theta}_0$, the sample is generated in n steps with times $t_i = i/n$ by iteratively sampling y from $p_R(\cdot|\boldsymbol{\theta}_i; t_i, \alpha_i)$-i.e., sampling a structure prediction $\mathbf{c}'$ from $p_O(\cdot|\boldsymbol{\theta}_i, t_i)$ and then $\mathbf{y}$ from $p_S\left(\cdot|\mathbf{c}', \alpha_{i+1}\right)$-and then setting $\boldsymbol{\theta}_{i+1} = h(\boldsymbol{\theta}_i, \mathbf{y})$. The final sample $\mathbf{c}$ is drawn from $p_O(\cdot|\boldsymbol{\theta}_n, 1)$. $\mathbf{c}$ includes the vector $\mathbf{k}$ for the lattice representation, which is multiplied with the basis matrices to obtain the lattice $\mathbf{L}$ in its matrix form, as explained in Section B of the Supplementary Information. The sample also encodes the atoms in the asymmetric unit $\mathbf{a}$, their site symmetry groups $\mathbf{S}$, and their positions in fractional coordinates $\mathbf{x}$. The complete unit cell is reconstructed from the asymmetric unit representation using the following procedure, based on the work of Levy et al.[22]. First, we identify the point groups that are subgroups of G and most closely match $\mathbf{S}$ by minimizing the Frobenius norm of their differences. Given these site symmetries and the predicted fractional coordinates $\mathbf{x}$ for each atom in the asymmetric unit, we map $\mathbf{x}$ to the closest Wyckoff positions $\mathbf{x}'$ using the `search_closest_wp` function from the PyXtal library[33]. Finally, the complete unit cell is obtained by replicating the representatives according to their Wyckoff positions, as implemented in the PyXtal library.

Property-conditioned generation

The SymmBFN architecture can be adapted to generate crystal structures with desired properties. To enable conditioning on a scalar property, we modify the neural network to incorporate the desired value T as an additional input: $\Psi(\boldsymbol{\mu}_\mathbf{k}, \boldsymbol{\mu}_\mathbf{x}, \boldsymbol{\theta}^\mathbf{a}, \boldsymbol{\theta}^\mathbf{S}, T, t, G)$. The target T is represented using a sinusoidal positional encoding $f_{\mathrm{pos}}(T)$, which is concatenated with the input features of each node. The rest of the BFN framework operates as in the model without property conditioning. During training, the neural network receives the target values from the dataset, while during generation, it is provided with the desired target.

Data availability

All datasets used in this study are publicly available and can be accessed via our GitHub repository at: github.com/aimat-lab/symmbfn.

Code availability

The code used for model training and evaluation is publicly available at: github.com/aimat-lab/symmbfn.

Received: 20 January 2026; Accepted: 7 May 2026;
Published online: 19 May 2026

References

1. Mobarak, M. H. et al. Scope of machine learning in materials research—a review. *Appl. Surf. Sci. Adv.* **18**, 100523 (2023).
2. Merchant, A. et al. Scaling deep learning for materials discovery. *Nature* **624**, 80–85 (2023).
3. Schmidt, J. et al. Large-scale machine-learning-assisted exploration of the whole materials space. *arXiv preprint* https://doi.org/10.48550/arXiv.2210.00579 (2022).
4. Metni, H. et al. Generative models for crystalline materials. *Adv. Mater.* **38**, e23620 (2026).
5. Park, H., Li, Z. & Walsh, A. Has generative artificial intelligence solved inverse materials design? *Matter* **7**, 2355–2367 (2024).
6. Xie, T., Fu, X., Ganea, O.-E., Barzilay, R. & Jaakkola, T. S. Crystal diffusion variational autoencoder for periodic material generation. In *International Conference on Learning Representations*. https://doi.org/10.48550/arXiv.2110.06197 (2022).

7. Song, Y. & Ermon, S. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems* 32. https://doi.org/10.48550/arXiv.1907.05600 (2019).
8. Jiao, R. et al. Crystal structure prediction by joint equivariant diffusion. In *Thirty-seventh Conference on Neural Information Processing Systems*. https://doi.org/10.48550/arXiv.2309.04475 (2023).
9. Zeni, C., Pinsler, R. & Zügner, D. A generative model for inorganic materials design. *Nature* **639**, 624–632 (2025).
10. Yang, S. et al. Scalable Diffusion for Materials Generation. In *Thirty-seventh Conference on Neural Information Processing Systems AI for Science Workshop.* https://doi.org/10.48550/arXiv.2311.09235 (2024).
11. Miller, B. K., Chen, R. T. Q., Sriram, A. & Wood, B. M. FlowMM: Generating materials with Riemannian flow matching. In *Forty-first International Conference on Machine Learning* https://doi.org/10.48550/arXiv.2406.04713 (2024).
12. Antunes, L. M., Butler, K. T. & Grau-Crespo, R. Crystal structure generation with autoregressive large language modeling. *Nat. Commun* **15**, 10570 (2024).
13. Gruver, N. et al. Fine-tuned language models generate stable inorganic materials as text. In *The Twelfth International Conference on Learning Representations*. https://doi.org/10.48550/arXiv.2402.04379 (2024).
14. Sriram, A., Miller, B. K., Chen, R. T. Q. & Wood, B. M. FlowLLM: Flow matching for material generation with large language models as base distributions. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. https://doi.org/10.48550/arXiv.2410.23405 (2024).
15. Wirnsberger, P. et al. Normalizing flows for atomic solids. *Mach. Learn. Sci. Technol.* **3**, 025009 (2022).
16. Jiao, R., Huang, W., Liu, Y., Zhao, D. & Liu, Y. Space group constrained crystal generation. In *The Twelfth International Conference on Learning Representations*. https://doi.org/10.48550/arXiv.2402.03992 (2024).
17. Chang, R. et al. Space group equivariant crystal diffusion. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*. https://doi.org/10.48550/arXiv.2505.10994 (2025).
18. Puny, O., Lipman, Y. & Miller, B. K. Space group conditional flow matching. https://doi.org/10.48550/arXiv.2509.23822 (2025).
19. Cao, Z., Luo, X., Lv, J. & Wang, L. Space group informed transformer for crystalline materials generation. *Sci. Bull.* **70**, 3522–3533 (2025).
20. Kazeev, N. et al. Wyckoff transformer: generation of symmetric crystals. In *Forty-second International Conference on Machine Learning*. https://doi.org/10.48550/arXiv.2503.02407 (2025).
21. Kelvinius, F. E. et al. Wyckoffdiff—a generative diffusion model for crystal symmetry. In *Forty-second International Conference on Machine Learning*. https://doi.org/10.48550/arXiv.2502.06485 (2025).
22. Levy, D. et al. SymmCD: symmetry-preserving crystal generation with diffusion models. In *The Thirteenth International Conference on Learning Representations*. https://doi.org/10.48550/arXiv.2502.03638 (2025).
23. Graves, A., Srivastava, R. K., Atkinson, T. & Gomez, F. Bayesian flow networks. *arXiv preprint* https://doi.org/10.48550/arXiv.2308.07037 (2023).
24. Song, Y. et al. Unified generative modeling of 3d molecules with Bayesian flow networks. In *The Twelfth International Conference on Learning Representations*. https://doi.org/10.48550/arXiv.2403.15441 (2023).
25. Wu, H. et al. A periodic Bayesian flow for material generation. In *The Thirteenth International Conference on Learning Representations*. https://doi.org/10.48550/arXiv.2502.02016 (2025).
26. Kaba, S.-O., Mondal, A. K., Zhang, Y., Bengio, Y. & Ravanbakhsh, S. Equivariance with learned canonicalization functions. https://doi.org/10.48550/arXiv.2211.06489 (2023).
27. Ong, S. P. et al. Python Materials Genomics (pymatgen): a robust, open-source Python library for materials analysis. *Comput. Mater. Sci.* **68**, 314–319 (2013).
28. Deng, B. et al. Chgnet as a pretrained universal neural network potential for charge-informed atomistic modelling. *Nat. Mach. Intell.* **5**, 1031–1041 (2023).
29. Jain, A. et al. Commentary: The materials project: a materials genome approach to accelerating materials innovation. *APL Mater.* **1**, 011002 (2013).
30. Chen, C. & Ong, S. P. A universal graph deep learning interatomic potential for the periodic table. *Nat. Comput. Sci.* **2**, 718–728 (2022).
31. Castelli, I. E. et al. New cubic perovskites for one- and two-photon water splitting using the computational materials repository. *Energy Environ. Sci.* **5**, 9034–9043 (2012).
32. Baird, S. G., Sayeed, H. M., Montoya, J. & Sparks, T. D. matbench-genmetrics: a Python library for benchmarking crystal structure generative models using time-based splits of materials project structures. *J. Open Source Softw.* **9**, 5618 (2024).
33. Fredericks, S., Parrish, K., Sayre, D. & Zhu, Q. Pyxtal: a Python library for crystal structure generation and symmetry analysis. *Comput. Phys. Commun.* **261**, 107810 (2021).
34. Satorras, V. G., Hoogeboom, E. & Welling, M. E(n) equivariant graph neural networks. https://doi.org/10.48550/arXiv.2102.09844 (2022).

Acknowledgements

L.R. acknowledges funding by the German Research Foundation (DFG) through the collaborative research center CRC 1249 "N-Heteropolycycles as Functional Materials" (SFB 1249, Project C13). L.T. acknowledges support by the Federal Ministry of Education and Research (BMBF) under Grant No. 01DM21002A (FLAIM). H.S. acknowledges financial support by the German Research Foundation (DFG) through the Research Training Group 2450 "Tailored Scale-Bridging Approaches to Computational Nanoscience." P.F. acknowledges funding by the German Research Foundation (DFG) under Germany's Excellence Strategy via the Excellence Cluster "3D Matter Made to Order" (3DMM2O, EXC-2082/1-390761711) and by the Federal Ministry of Education and Research (BMBF) under Grant No. 01DM21001B (German-Canadian Materials Acceleration Center). The authors acknowledge support by the state of Baden-Württemberg through bwHPC.

Author contributions

L.R. contributed to conceptualization, methodology, implementation, experiments, and writing of the manuscript. L.T. contributed to conceptualization and writing. All authors contributed to writing and discussions. P.F. contributed to conceptualization, supervised the project and was responsible for project administration and funding acquisition.

Funding

Open Access funding enabled and organized by Projekt DEAL.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41524-026-02140-8.

Correspondence and requests for materials should be addressed to Pascal Friederich.