Supplementary Information for
Powder Diffraction Crystal Structure Determination Using Generative Models

Qi Li[1,2‡], Rui Jiao[3,4‡], Liming Wu[5,6], Tiannian Zhu[1,2], Wenbing Huang[5,6*], Shifeng Jin[1,2*], Yang Liu[3,4], Hongming Weng[1,2], Xiaolong Chen[1,2]

1 *The Beijing National Laboratory for Condensed Matter Physics, Institute of Physics, Chinese Academy of Sciences, Beijing 100190, China*

2 *School of Physical Sciences, University of Chinese Academy of Sciences, Beijing, 101408, China.*

3 *Dept. of Comp. Sci. & Tech., Institute for AI, Tsinghua University*

4 *Institute for AIR, Tsinghua University*

5 *Gaoling School of Artificial Intelligence, Renmin University of China*

6 *Beijing Key Laboratory of Big Data Management and Analysis Methods, Beijing, China*

‡ Those authors contribute equally.

* Corresponding author: hwenbing@ruc.edu.cn, shifengjin@iphy.ac.cn

I. Details of neural network

1.1 Details of Generative Frameworks

In this work, we apply diffusion and flow matching models in PXRDGen to jointly generate the lattice matrices (***L***) and fractional coordinates (***F**i*) of crystal structures in a condition of chemical composition (**A**) and PXRD pattern, which are adapted from DiffCSP[1] and FlowMM[2].

1.1.1 Diffusion Models

The lattice diffusion is applied via the standard DDPM[3] process, which includes a forward diffusion process and a backward generation process. The forward process gradually adds noise to L_0 towards a Gaussian prior as

$$q(L_t|L_0) = \mathcal{N}(L_t|\sqrt{\bar{\alpha}_t}L_0, (1-\bar{\alpha}_t)I),$$

where $\bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s), \beta_s \in (0,1)$ determines the standard deviation of each step.

And the backward process progressively denoises from the Gaussian prior via

$$p(L_{t-1}|M_t) = \mathcal{N}(L_{t-1}|\mu(M_t), \sigma^2(M_t)I)$$

where M_t is the intermediate state of timestep t, and the time-dependent mean and stardard deviation $\mu(M_t)$ and $\sigma^2(M_t)$ are defined by

$$\mu(M_t) = \frac{1}{\sqrt{\alpha_t}}\left(L_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\hat{\epsilon}_L(M_t, t)\right)$$

$$\sigma^2(M_t) = \beta_t \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}$$

where the denoising term $\hat{\epsilon}_L(M_t, t)$ is predicted by the model. And the training objective is defined as

$$\mathcal{L}_{L,diffusion} = \mathbb{E}_{\epsilon_L \sim \mathcal{N}(0,I), t\sim U(1,T)}[\|\epsilon_L - \hat{\epsilon}_L(M_t, t)\|_2^2]$$

The diffusion process of fractional coordinates utilizes score matching[4] on the wrapped normal distribution. The forward process is defined as

$$q(F_t|F_0) \propto \prod\nolimits_{Z\in\mathbb{Z}^{3\times N}} \exp\left(-\frac{\|F_t - F_0 + Z\|^2}{2\sigma_t^2}\right)$$

where $\sigma_t = \sigma_1 \left(\frac{\sigma_T}{\sigma_1}\right)^{\frac{t-1}{T-1}}$ is the noise scale at timestep t. The generation process is constructed via the predictor-corrector algorithm[5] as

$$F_{t-0.5} = w(F_t + (\sigma_t^2 - \sigma_{t-1}^2)\hat{\epsilon}_F + \sigma_{t-1}\epsilon_F)$$
$$F_{t-1} = w(F_{t-0.5} + \gamma \frac{\sigma_{t-1}}{\sigma_t}\hat{\epsilon}_F + \sqrt{2\gamma}\frac{\sigma_{t-1}}{\sigma_t}\epsilon'_F)$$

where $w(\cdot)$ projects fractional coordinates back to $\mathbb{R}^{[0,1)}$, and ϵ_F, ϵ'_F as Gaussian noises, $\hat{\epsilon}_F$ is yielded by the model with training objective as

$$\mathcal{L}_{F,diffusion} = \mathbb{E}_{\mathrm{F_t}\sim q(F_t|F_0), t\sim U(1,T)}\left[\lambda_t \left\|\nabla_{\mathrm{F_t}} \log q(F_t|F_0) - \hat{\epsilon}_F(M_t, t)\right\|_2^2\right]$$

1.1.2 Flow Matching Models

As the lattices are defined in the Euclidean space, the flow path can be simply constructed as

$$q(L_t|L_0) = \mathcal{N}(L_t|(1-t)L_0, tI).$$

The training objective is defined as

$$\mathcal{L}_{L,flow} = \mathbb{E}_{\epsilon_L \sim \mathcal{N}(0,I), t\sim U(1,T)}[\|\epsilon_L - \hat{\epsilon}_L(M_t, t)\|_2^2].$$

The generative ODE process follows

$$L_{t-\Delta t} = L_t - \frac{\hat{\epsilon}_L - L_t}{1-t}\Delta t$$

For the fractional coordinates, we first sample F_1 from the uniform distribution on the torus, and connect the shortest path from the data F_0 to F_1 via the logarithmic map $\log_{\mathrm{F_0}} F_1 = w(F_1 - F_0 + 0.5) - 0.5$. We also reduce the overall translation to acquire the final target as $v_F(F_0, F_1) = \log_{F_0} F_1 - Mean(\log_{F_0} F_1)$. The training objective on coordinates is

$$\mathcal{L}_{F,flow} = \mathbb{E}_{t\sim U(1,T), F_1\sim U(0,1)}[\|\hat{\epsilon}_F(M_t, t) - v_F(F_0, F_1)\|_2^2].$$

Inspired by FlowMM[2], we apply the anti-annealing strategy during the generative ODE as

$$F_{t-\Delta t} = F_t + (1+10t)\hat{\epsilon}_F \Delta t.$$

1.2 Details of Backbone Models

1.2.1 Decoder in Generating Structures in PXRDGen

We adapt CSPNet in DiffCSP[1] as the backbone decoder model, which is a graph neural network to jointly model $\boldsymbol{L}$ and $\boldsymbol{F_i}$. It takes atom types $f_{atom}(a_i)$, time embeddings $f_{pos}(t)$, and XRD features f_{XRD} from the XRD encoder as inputs, and acquires the initial atom representations via multi-layer perceptron (MLP) as $h_i^{(0)} = \rho\left(f_{atom}(a_i), f_{pos}(t), f_{XRD}\right)$. The message passing mechanism is modeled as

$$m_{ij}^{(s)} = \phi_m\left(h_i^{(s-1)}, h_j^{(s-1)}, L^{\top}L, \psi_{FT}(f_j - f_i)\right),$$

$$m_i^{(s)} = \sum\nolimits_{j=1}^{N} m_{ij}^{(s)},$$

$$h_i^{(s)} = h_i^{(s-1)} + \phi_h\left(h_i^{(s-1)} + m_i^{(s)}\right),$$

where ϕ_h, ϕ_m are MLPs, and ψ_{FT} is the Fourier transformation. For the final output layer, we have

$$\hat{\epsilon}_L = L\phi_L\left(\frac{1}{N}\sum\nolimits_{i=1}^{N} h_i^{(S)}\right),$$

$$\hat{\epsilon}_F[:, i] = \phi_F\left(h_i^{(S)}\right),$$

where ϕ_L, ϕ_F are MLPs.

1.2.2 Decoder in Generating Lattice in CellNet of PXRDGen

We adopt diffusion model to generate $\boldsymbol{L}$ in the CellNet, which means that the $\boldsymbol{L}$ can be generated given only the XRD data. It takes time embeddings $f_{pos}(t)$, XRD features f_{XRD} from the XRD encoder as inputs, and is modeled as

$$\hat{\epsilon}_L = L\phi_L\left(f_{pos}(t), f_{XRD}\right),$$

where ϕ_L is MLP.

1.3 Details of training neural networks

All the networks are implemented in Pytorch and Pytorch_Lightning framework, and trained on NVIDIA A100-PCIE-40GB GPUs.

In contrastive learning module, the batch size is set as 256, and the Adam optimizer is employed with a learning rate of 1e-3 for XRDEncoder-CNN and 1e-4 for XRDEncoder-T, with a CosineAnnealingLR scheduler. The training epoch is set as 400.

In crystal structure or lattice generation, the batch size is set as 256, and the Adam optimizer is employed with a learning rate of 1e-3, with a ReduceLROnPlateau scheduler. The training epoch is set as 1000 for both diffusion model and flow model. Diffusion model samples crystal structures or lattices in 1000 steps, while flow model can generate structures in less steps, here 200 steps are chosen.

References

1. Jiao, R. *et al.* Crystal structure prediction by joint equivariant diffusion. *arXiv preprint arXiv:2309.04475* (2023).
2. Miller, B. K., Chen, R. T. Q., Sriram, A. & Wood, B. M. FlowMM: Generating Materials with Riemannian Flow Matching. Preprint at http://arxiv.org/abs/2406.04713 (2024).
3. Ho, J., Jain, A. & Abbeel, P. Denoising Diffusion Probabilistic Models. Preprint at http://arxiv.org/abs/2006.11239 (2020).
4. Song, Y. & Ermon, S. Generative Modeling by Estimating Gradients of the Data Distribution. Preprint at http://arxiv.org/abs/1907.05600 (2020).
5. Song, Y. *et al.* Score-Based Generative Modeling through Stochastic Differential Equations. Preprint at http://arxiv.org/abs/2011.13456 (2021).

II. Figures and Tables

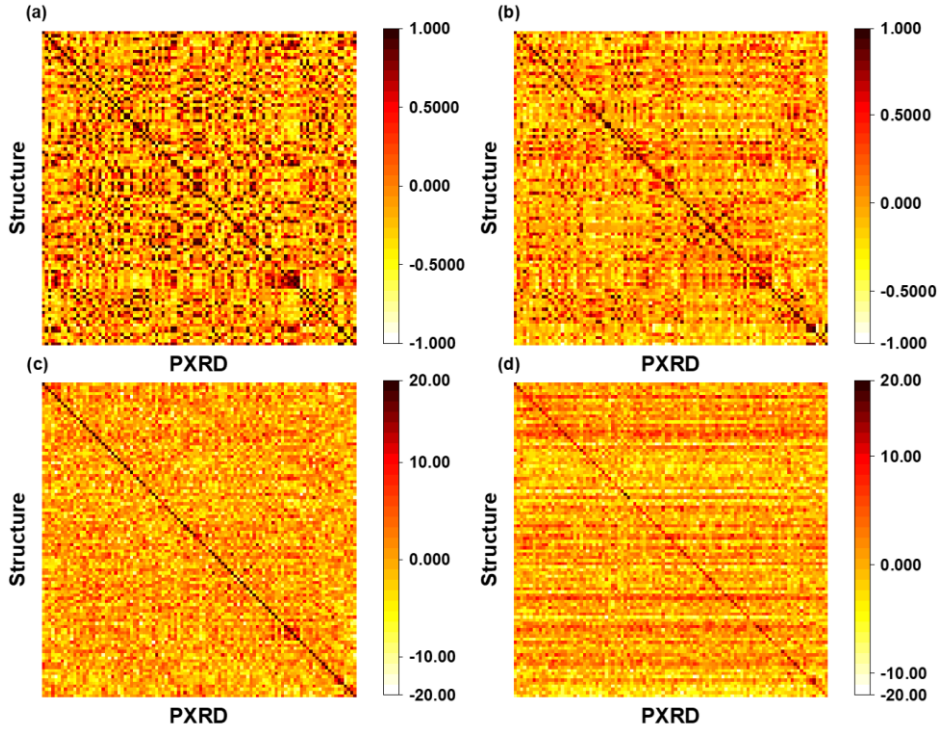

Figure S1. Visualization of the pre-trained contrastive learning module results. Panels (a)-(d) display the heatmap of similarity metrics for a random selection of 100 samples from the test dataset using XRDEncoder-T (t=1.0), XRDEncoder-CNN (t=1.0), XRDEncoder-T (t=0.05), and XRDEncoder-CNN (t=0.05) respectively. Darker colors indicate higher matches.

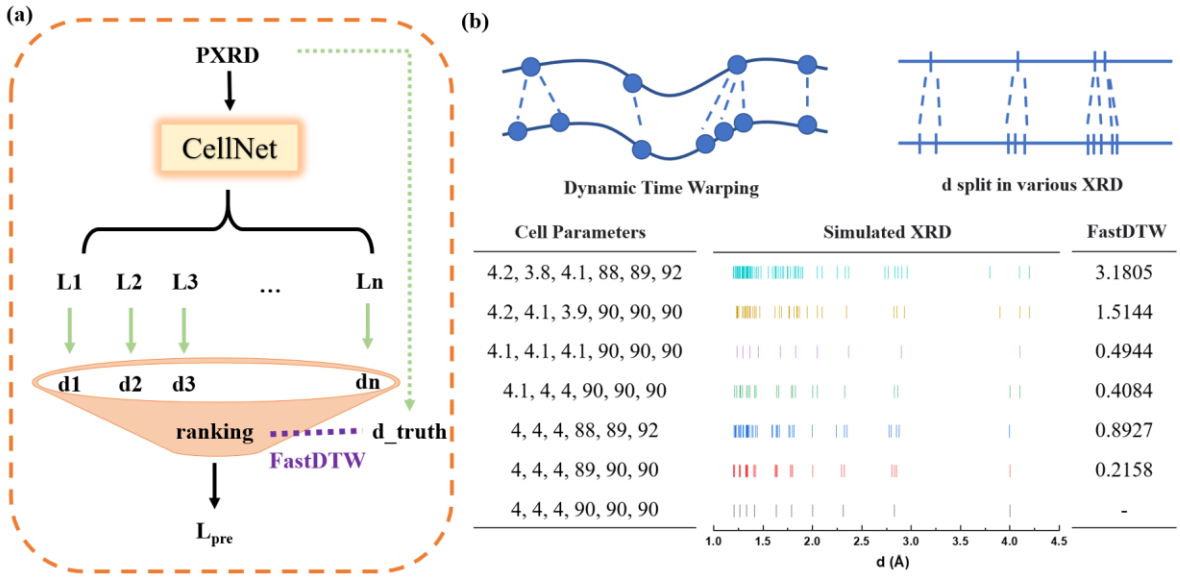

Figure S2. CellNet in PXRDGen. (a) Screen the best ***L*** from multiple ***L*** generated by CellNet. (b) The demonstration of FastDTW. Dynamic Time Warping is used to describe the similarity of different lengths of sequences. This scenario frequently occurs in XRD patterns when minor alterations in cell parameters result in variations in peak numbers, as illustrated by the provided examples. Therefore, utilizing FastDTW to determine the optimal ***L*** from multiple options is a logical approach.

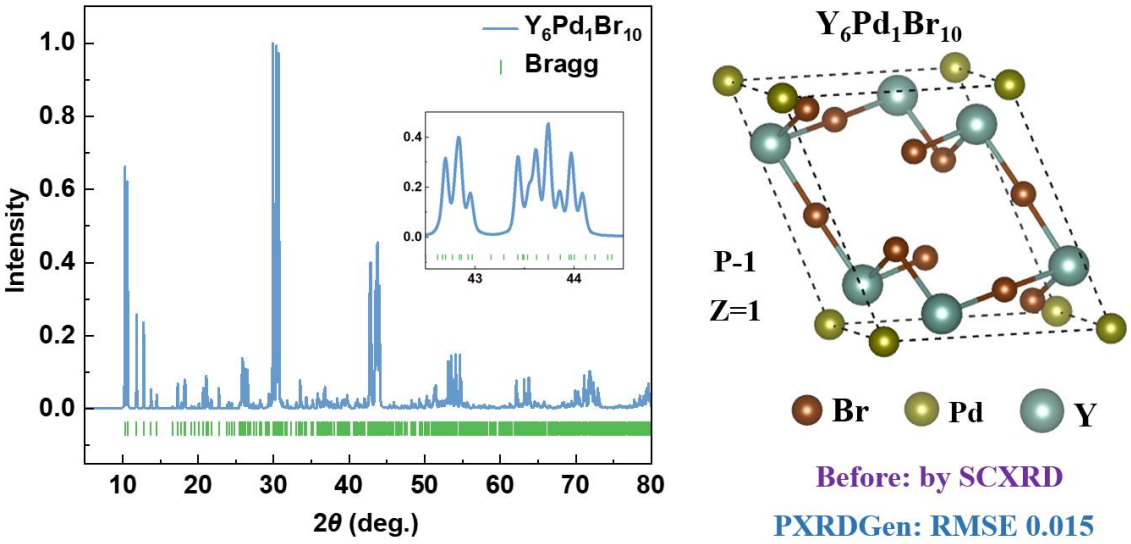

Figure S3. Examples of PXRDGen in Solving Challenging Structures. Demonstrating the application of PXRDGen in resolving peak overlap to determine the structure of $Y_6Pd_1Br_{10}$.

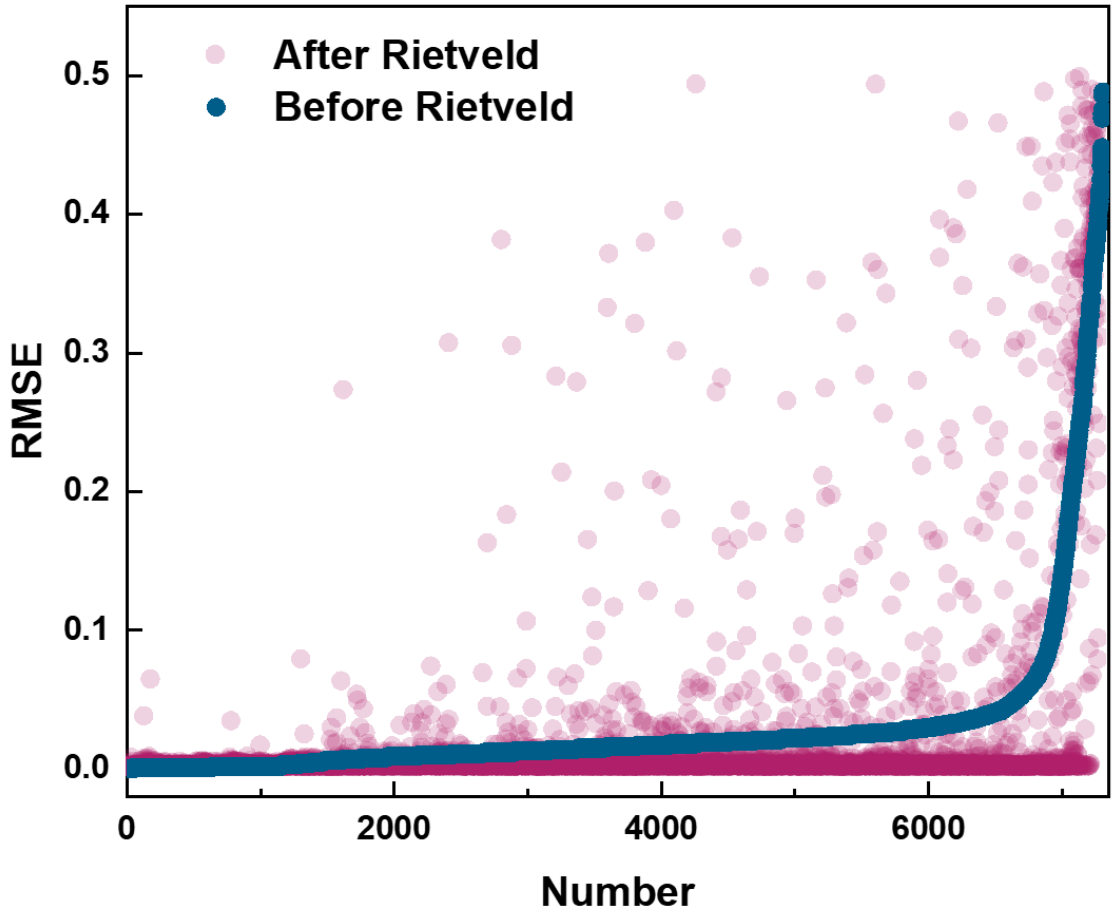

Figure S4. Comparison of RMSE values before and after Rietveld refinement using the flow-CNN-Ltruth module in the MP-20 dataset.

Table S1. Diffusion model combined with various pre-trained XRD encoders in MP-20 dataset.

Diffusion + pretrained XRD encoder	fix	Sample 1		Sample 20	
		Match rate (%)	RMSE	Match rate (%)	RMSE
origin *	-	52.42	0.0639	77.69	0.0504
origin + ground truth	-	67.74	0.0700	87.68	0.0431
XRDEncoder-T w/o CL	-	50.93	0.0605	78.29	0.0505
XRDEncoder-T (t=1.0)	yes	52.43	0.0569	78.92	0.0478
XRDEncoder-T (t=1.0)	no	50.45	0.0646	78.33	0.0516
XRDEncoder-T (t=0.05)	yes	60.09	0.0620	81.87	0.0457
XRDEncoder-T (t=0.05)	no	49.44	0.0618	78.07	0.0540
XRDEncoder-CNN w/o CL	-	58.76	0.0555	80.36	0.0422
XRDEncoder-CNN (t=1.0)	yes	55.59	0.0582	78.80	0.0463
XRDEncoder-CNN (t=1.0)	no	66.01	0.0601	84.04	0.0402
XRDEncoder-CNN (t=0.05)	yes	56.85	0.0634	78.73	0.0502
XRDEncoder-CNN (t=0.05)	no	65.56	0.0617	83.87	0.0422

* The origin result of DiffCSP shows 51.49% match rate, 0.0631 RMSE in one sample, and 77.93% match rate, 0.0492 RMSE in 20 samples. Our test result is slightly better than the origin version.

Table S2. Flow model combined with various pre-trained XRD encoders in MP-20 dataset.

Flow + pretrained XRD encoder	fix	Sample 1		Sample 20	
		Match rate (%)	RMSE	Match rate (%)	RMSE
origin *	-	58.25	0.0675	80.23	0.0540
origin + ground truth	-	76.33	0.0746	87.97	0.0454
XRDEncoder-T w/o CL	-	59.20	0.0674	80.17	0.0534
XRDEncoder-T (t=1.0)	yes	60.78	0.0660	80.85	0.0517
XRDEncoder-T (t=1.0)	no	59.99	0.0662	79.88	0.0525
XRDEncoder-T (t=0.05)	yes	67.84	0.0724	83.63	0.0473
XRDEncoder-T (t=0.05)	no	60.13	0.0732	80.13	0.0538
XRDEncoder-CNN w/o CL	-	68.47	0.0724	84.16	0.0476
XRDEncoder-CNN (t=1.0)	yes	60.32	0.0682	81.27	0.0538
XRDEncoder-CNN (t=1.0)	no	68.07	0.0711	85.02	0.0495
XRDEncoder-CNN (t=0.05)	yes	60.86	0.0761	81.45	0.0570
XRDEncoder-CNN (t=0.05)	no	68.68	0.0707	85.37	0.0489

* This version is inspired and adapted by FlowMM, and the training epoch is set to 1000. When training in 2000 epochs, it shows 61.80% match rate, 0.0605 RMSE in one sample, and 79.18% match rate, 0.0472 RMSE in 20 samples. Although the performance in one sample increases, the ability in 20 samples dose not gain more. So the training epoch of flow is set to 1000 to save time and to keep align with diffusion model.

Table S3. Results of CellNet in lattice generation in MP-20 dataset.

Cell parameters	MAPE （Mean Absolute Percentage Error）/ %					
	$\boldsymbol{L}_1$	$\boldsymbol{L}_{20}$	$\boldsymbol{L}_{100}$	$\boldsymbol{L}_{200}$	$\boldsymbol{L}_{500}$	$\boldsymbol{L}_{1000}$
a	6.3792	3.4658	2.6598	2.4584	2.0846	1.8750
b	4.8907	2.7106	2.1520	1.9053	1.6789	1.4781
c	6.0874	2.7942	2.1202	1.8803	1.5651	1.3780
α	7.4524	1.5136	0.7798	0.6213	0.4341	0.3424
β	5.3369	1.2756	0.7475	0.5939	0.4158	0.3440
γ	6.6226	1.4259	0.7547	0.5318	0.3810	0.2828
V	12.8206	6.9605	5.3762	4.9635	4.1894	3.5727

Cell parameters	RMSE（Root Mean Square Error）					
	$\boldsymbol{L}_1$	$\boldsymbol{L}_{20}$	$\boldsymbol{L}_{100}$	$\boldsymbol{L}_{200}$	$\boldsymbol{L}_{500}$	$\boldsymbol{L}_{1000}$
a	0.6555	0.4276	0.3505	0.3398	0.2959	0.2729
b	0.6274	0.4265	0.3579	0.3203	0.2906	0.2615
c	2.5196	0.8553	0.7079	0.6652	0.5221	0.4625
α	14.5805	4.9690	3.2510	2.6570	2.2032	1.7897
β	10.9401	4.2835	3.1762	2.6958	2.0387	1.8461
γ	13.9864	5.1898	3.4696	2.6341	2.0887	1.5966
V	61.2992	42.7034	33.6405	34.0600	28.9132	26.5617

$\boldsymbol{L}_N$ refers to screen out the best lattice from N predicted lattice from CellNet, and the screening method is to calculate the MAPE between the predicted cell and the target cell, which represents the upper capability of the CellNet.

Table S4. Performance of flow-CNN module given the predicted $\boldsymbol{L}$ by CellNet.

Generate structures given $\boldsymbol{L}$	Screen Method	One sample Match rate (%)	One sample RMSE
flow-CNN	-	68.68	0.0707
flow-CNN + $\boldsymbol{L}_{\text{truth}}$	-	75.32	0.0726
flow-CNN + $\boldsymbol{L}_{1}$	Compare with $\boldsymbol{L}_{\text{truth}}$	61.96	0.0694
flow-CNN + $\boldsymbol{L}_{20}$	Compare with $\boldsymbol{L}_{\text{truth}}$	71.05	0.0735
flow-CNN + $\boldsymbol{L}_{100}$	Compare with $\boldsymbol{L}_{\text{truth}}$	72.40	0.0743
flow-CNN + $\boldsymbol{L}_{200}$	Compare with $\boldsymbol{L}_{\text{truth}}$	72.92	0.0746
flow-CNN + $\boldsymbol{L}_{500}$	Compare with $\boldsymbol{L}_{\text{truth}}$	73.71	0.0752
flow-CNN + $\boldsymbol{L}_{1000}$	Compare with $\boldsymbol{L}_{\text{truth}}$	74.46	0.0744
flow-CNN + $\boldsymbol{L}_{20}$	FastDTW	67.80	0.0687
flow-CNN + $\boldsymbol{L}_{100}$	FastDTW	69.40	0.0680
flow-CNN + $\boldsymbol{L}_{200}$	FastDTW	69.57	0.0707
flow-CNN + $\boldsymbol{L}_{500}$	FastDTW	70.24	0.0693
flow-CNN + $\boldsymbol{L}_{1000}$	FastDTW	70.39	0.0707

$\boldsymbol{L}_{\text{N}}$ represents to screen out the best lattice from N predicted lattice from CellNet.