Electronic Supplementary Material (ESI) for Physical Chemistry Chemical Physics.


# Supplementary Materials:

# Data-Driven Generation of Mixed X-Anion Perovskites Properties

Jun-Jie Hu[1,2,4,5], Zhe-Yong Zhang[5], Guo-Xiang Zhao[1]
Qiao-Hong Li[1*], Peng Gao[2,4*], Rong-Jian Sa[3,1*]
[1]State Key Laboratory of Structural Chemistry,
Fujian Institute of Research on the Structure of Matter,
Chinese Academy of Sciences, Fuzhou, Fujian 350002, P.R. China.
[2]University of Chinese Academy of Sciences, Beijing 100049, P. R. China.
[3]Fujian Key Laboratory of Functional Marine Sensing Materials,
College of Materials and Chemical Engineering,
Minjiang University, Fuzhou, Fujian, 350108, PR China.
[4]CAS Key Laboratory of Design and Assembly of Functional Nanostructures,
Fujian Institute of Research on the Structure of Matter,
Chinese Academy of Sciences, Fuzhou, Fujian 350002, P.R. China.
[5] Center for Integrated Quantum Information Technologies (IQIT),
School of Physics and Astronomy and State Key Laboratory of Advanced
Optical Communication Systems and Networks,
Shanghai Jiao Tong University, Shanghai 200240, China
*E-mail: lqh2382@fjirsm.ac.c; E-mail: peng.gao@fjirsm.ac.cn;
E-mail: rjsa@mju.edu.cn

CONTENTS:

## The details of density functional theory (DFT) calculations

GAN was first proposed by Ian J. Goodfellow et al, which included two models: a generator G and a discriminator D. Any differentiable function was theoretically permitted by GAN. The initial algorithm of GAN was optimized according to the cross entropy error function that was also used in the WSP-GAN. These loss functions of G and D were listed here:

$$F_{loss}(D) = \log D(x_r) + \log(1 - D(x_f)) \tag{1}$$

$$F_{loss}(G) = (\log 1 - D(x_f)) \tag{2}$$

where the $x_r$ was the real data, and the $x_f$ was the synthetised data by G. Considering the running of model was optimized by the gradient descent process of Adam, we selected the opposite number of the loss function of D for our WSP-GAN. Besides, for POSCAR t, the property data calculated by density functional theory (DFT) was recorded as $x_t^F$; the syntheised data by the generator was $x_t^S$ as well. Further, we prepared a data structure of queue for the input of the discriminator. As shown in the figure S1, before the training of WSP-GAN, we performed the initialization of the input queue in the step 1 and randomly choose the n property data by first principle calculation as the element of the input. When we accepted the $x_t^F$, the input queue updated itself according to the process of step 2. While the input queue received the $x_t^S$, it deleted the first element of the queue and insert this new one at the end of this queue. Meanwhile, the queue updated by step 2 was regarded as the new n-vectors $x_r$ and that by step 3 was the n-vectors $x_f$. At last, the WSP-GAN performed its optimized processing according the following loss function:

$$F_{loss}(D) = -(\log D(x_r) + \log(1 - D(x_f)) \tag{3}$$

$$F_{loss}(G) = (\log 1 - D(x_f)) \tag{4}$$

Although we could theoretically reach the global optimization of the GAN when D was equal to 0.5, a few of studies stated that the training of GAN was unstable and they proposed some methods to improve their loss function, such as WGAN. The model of WSP-GAN were running in the kernel of jupyter notebook, and we also provided these .ipynb files in the Github.

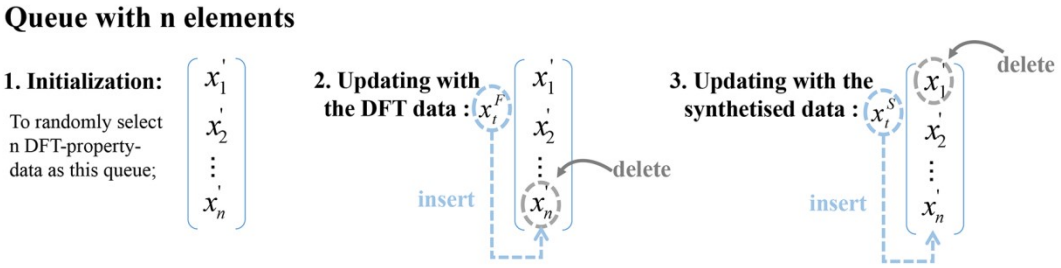



**Figure S1. The method to establish the input queue of the discriminator. The processing to build this queue includes three steps: 1. Initialization, 2. updating with the property data calculated by DFT and 3. updating with the property data synthetised by the generator. In the step 2 and 3, we delete the element in different site for updating the queue.**

## The details of density functional theory (DFT) calculations

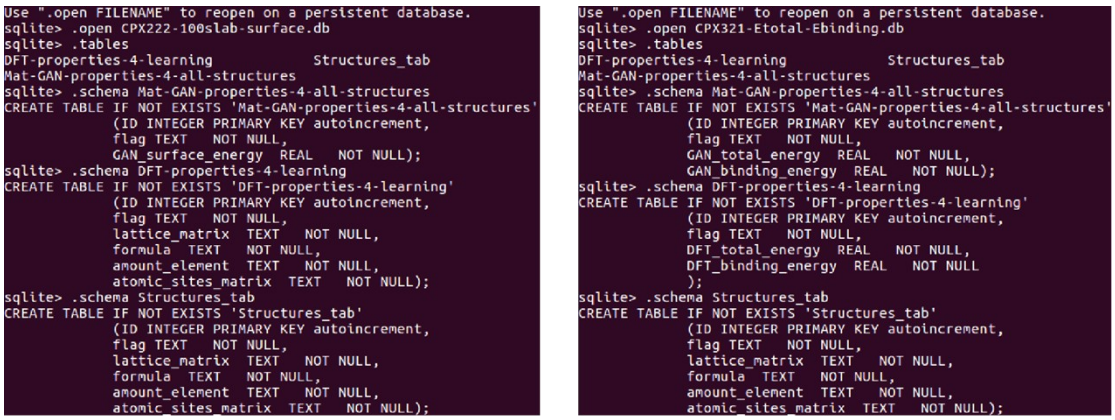
All of the density functional theory (DFT) calculations in this work were performed using the Vienna ab-initio simulation package (VASP) within the projector augmented-wave approach[52,75]. The Perdew-Burke-Ernzehof generalized gradient approximation exchange-correlation functional. A plane-wave energy cut-off of 550 eV were used for the $CsPb(Br/I)_3$ with 2*2*2 and 3*2*1 supercells and the Si-Fe-Mn alloy materials.[74]. Besides, a plane-wave energy cut-off of 400 eV were used for the slab model of $CsPb(Br/I)_3$. The 4*4*4 Monkhorst-Pack grid for the Brillouin zone sampling was used for calculation. The energy was converged to within $1*10^{-5}$ eV/atom and force within 0.01 eV/angstrom. We performed 4000 calculation for mixed-atoms $CsPb(Br/I)_3$ and Si-Fe-Mn alloy. The total energy, the binding energy, the slab energy and the surface energy of these materials were calculated.

We tackled the batch process of the running of VASP with the bash shell and python script and managed their results by the python with SQLite3[57].

## The details of SQLite3 databases

The SQLite was a light-wight database with SQL database engine. We adopted it to manage the structured data about the crystal structure and property data. For each mixed-atoms materials in figure 3 in the context, we prepared four tables to storage these property data calculated by DFT and WSP-GAN, the structure data in the POSCAR files, respectively.

In the following, we listed the contents of SQLite for CPX321, CPX222, SFM and (100) CPX in the Figure S3. These .sql files were available in our Github.



```
Use ".open FILENAME" to reopen on a persistent database.
sqlite> .open CPX222-100slab-surface.db
sqlite> .tables
DFT-properties-4-learning           Structures_tab
Mat-GAN-properties-4-all-structures
sqlite> .schema Mat-GAN-properties-4-all-structures
CREATE TABLE IF NOT EXISTS 'Mat-GAN-properties-4-all-structures'
            (ID INTEGER PRIMARY KEY autoincrement,
            flag TEXT   NOT NULL,
            GAN_surface_energy  REAL   NOT NULL);
sqlite> .schema DFT-properties-4-learning
CREATE TABLE IF NOT EXISTS 'DFT-properties-4-learning'
            (ID INTEGER PRIMARY KEY autoincrement,
            flag TEXT   NOT NULL,
            lattice_matrix  TEXT   NOT NULL,
            formula  TEXT   NOT NULL,
            amount_element  TEXT   NOT NULL,
            atomic_sites_matrix  TEXT   NOT NULL);
sqlite> .schema Structures_tab
CREATE TABLE IF NOT EXISTS 'Structures_tab'
            (ID INTEGER PRIMARY KEY autoincrement,
            flag TEXT   NOT NULL,
            lattice_matrix  TEXT   NOT NULL,
            formula  TEXT   NOT NULL,
            amount_element  TEXT   NOT NULL,
            atomic_sites_matrix  TEXT   NOT NULL);
```

(a)

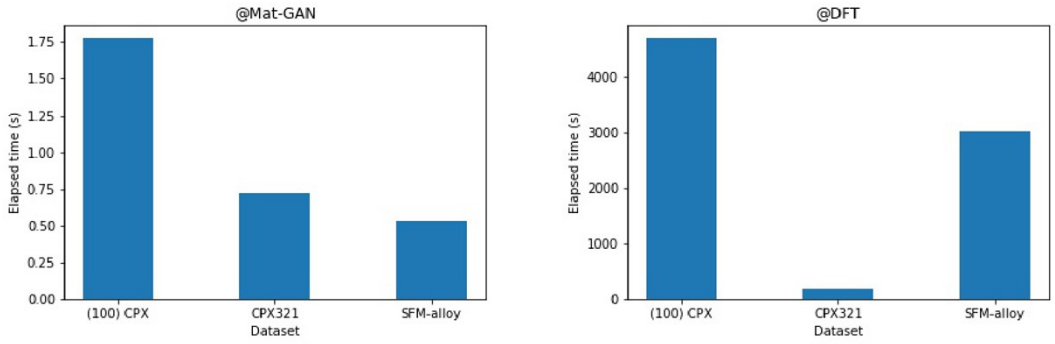
```
Use ".open FILENAME" to reopen on a persistent database.
sqlite> .open CPX321-Etotal-Ebinding.db
sqlite> .tables
DFT-properties-4-learning           Structures_tab
Mat-GAN-properties-4-all-structures
sqlite> .schema Mat-GAN-properties-4-all-structures
CREATE TABLE IF NOT EXISTS 'Mat-GAN-properties-4-all-structures'
            (ID INTEGER PRIMARY KEY autoincrement,
            flag TEXT   NOT NULL,
            GAN_total_energy  REAL   NOT NULL,
            GAN_binding_energy  REAL   NOT NULL);
sqlite> .schema DFT-properties-4-learning
CREATE TABLE IF NOT EXISTS 'DFT-properties-4-learning'
            (ID INTEGER PRIMARY KEY autoincrement,
            flag TEXT   NOT NULL,
            DFT_total_energy  REAL   NOT NULL,
            DFT_binding_energy  REAL   NOT NULL
            );
sqlite> .schema Structures_tab
CREATE TABLE IF NOT EXISTS 'Structures_tab'
            (ID INTEGER PRIMARY KEY autoincrement,
            flag TEXT   NOT NULL,
            lattice_matrix  TEXT   NOT NULL,
            formula  TEXT   NOT NULL,
            amount_element  TEXT   NOT NULL,
            atomic_sites_matrix  TEXT   NOT NULL);
```

(b)

**Figure S2. The schema of SQLite3. The CPX222, CPX321 and SFM have the same tables of databases. That of 100 CPX replaces the total energy and binding energy with the properties related to surface energy. The label of flag is used to query data across tables.**

## The computational resource of generative model and DFT calculations

In this section, we provided the computational time of each structure by VASP and WSP-GAN (Figure S4) and the information of the hardware (Table S1) to perform the calculation of VASP (cluster) and WSP-GAN (PC).




(a) (b)

**Figure S3. The Elapsed time of each structure in the different dataset for WSP-GAN and DFT. The Elapsed time of WSP-GAN is small than two seconds. Meanwhile, that of VASP was beyond 100 seconds and even over 4000 seconds.**

**Table S1 the Performance Comparison Between PC and Cluster.**

| | Personal Computer(PC) | Cluster |
|---|---|---|
| **Architecture** | x86_64 | x86_64 |
| **CPU op-mode(s)** | 32-bit,64-bit | 32-bit,64-bit |
| **Byte Order** | LittleEndian | LittleEndian |
| **CPU(s)** | 4 | 32 |
| **On-line CPU(s) list** | 0-3 | 0-31 |
| **Thread(s) per core** | 2 | 1 |
| **Core(s) per socket** | 2 | 16 |
| **Socket(s)** | 1 | 2 |
| **NUMA node(s)** | 1 | 2 |
| **Vendor ID** | AuthenticAMD | GenuineIntel |
| **CPU family** | 21 | 6 |
| **Model** | 48 | 85 |
| **Model name** | AMDA8PRO-7600BR7,10ComputeCores4C+6G | Intel(R)Xeon(R)Gold6129CPU@2.30GHz |
| **Stepping** | 1 | 4 |
| **CPU MHz** | 1651.364 | 1399.945 |
| **CPU max MHz** | 3100 | —— |
| **CPU min MHz** | 1400 | —— |
| **BogoMIPS** | 6188.23 | 4605.03 |
| **Virtualization** | AMD-V | VT-x |
| **L1d cache** | 16K | 32K |
| **L1i cache** | 96K | 32K |

| | | |
|---|---|---|
| **L2 cache** | 2048K | 1024K |
| **L3 cache** | —— | 22528K |
| **NUMA node0 CPU(s)** | 0-3 | 0-15 |
| **NUMA node1 CPU(s)** | —— | 16-31 |

## The PXRD data of Mixed X-anion perovskite by Pymatgen

These results of powder X-ray diffraction (PXRD) about different crystals were simulated by the Pymatgen package with CuKa radiation. The PXRD contained the information of different crystal plane. Each peak of the PXRD stood for a crystal face and was a vector with a diffraction angle ($\theta^* = 2\theta$) and intensity (I). As shown in figure S3, we gave out the sample PXRD data for $CsPb(Br/I)_3$ in the 3*2*1 supercell and 2*2*2 supercell, the slab model of $CsPb(Br/I)_3$ and Si-Fe-Mn alloy. These PXRD data could reflect these change about the arrangement of mixed-atoms.

For different materials, we established the Crystal Plane Graph Network from the PXRD data. Besides, this structural descriptor owned a significant role in the learning of the supervised GCN and unsupervised WSP-GAN. We should choose the peaks that changed significantly when the arrangement of mixed-atoms altered.

We used the inner product of peaks as the element of this matrix, which corresponds to the Laplacian matrix. This matrix also stood for the graph networks, which was a set of vertices and edges. As shown in the figure S5, we gave out the sample of PXRD of these materials involved in our works. Besides, the completed information of the PXRD in figure S5 were listed in the Table S2.

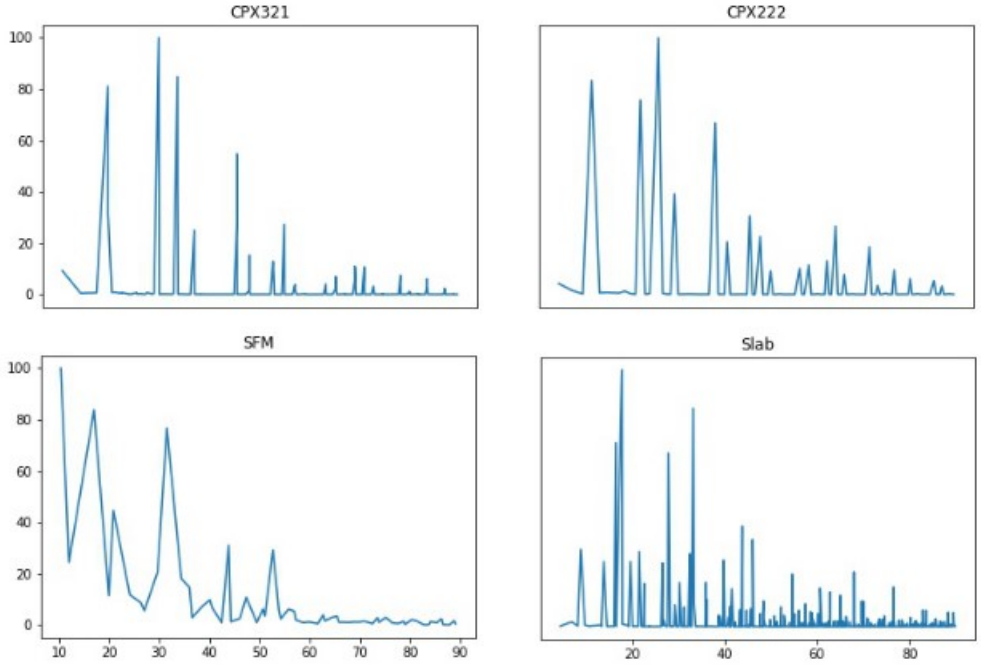



**Figure S4. The Powder X-Ray Diffraction Simulated by Pymatgen. There are the PXRD results of $CsPb(Br/I)_3$ in the 3*2*1 supercell (CPX321), $CsPb(Br/I)_3$ in the 2*2*2 supercell (CPX222), Si-Fe-Mn alloy and the (100) slab model of $CsPb(Br/I)_3$, respectively.**

## Structural descriptors for generative model: The Gram matrix

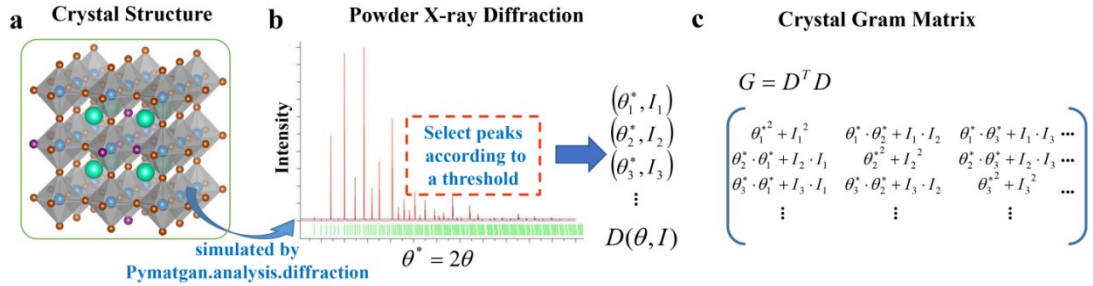
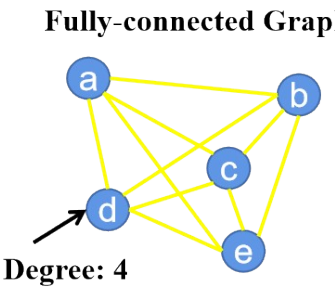



**Figure S5. Schematic representing the PXRD data of crystal by the format of the Gram matrix.} Perovskite material is given as an example with the auto-workflow to establish the structural descriptor with its crystal structure and POSCAR file as inputs. (a) The structural example of perovskite materials. (b) The powder X-ray diffraction (PXRD) data of the structure in (a). (c) The Crystal Gram Matrix generated by PXRD data.**

A graph structured data was defined as a non empty set of vertices and a set of edges. The order of a graph (N) was the amount of vertices in this set; the size of the graph (S) was defined as the number of edges; the degree of a vertex (D) was equal to the amount of edges incident with a vertex. Herein, the graph was converted to matrices. The degree matrix ($M_D$) was a diagonal matrix that contained the degree of each vertex in a graph. The adjacency matrix ($M_A$) recorded the information of each edges. Besides, The Laplacian matrix was define as $M_D - M_A$. These were shown in the figure S6.




**Degree matrix ($M_D$)**

$$\begin{pmatrix} V(vertex\_a) & 0 & 0 & 0 & 0 \\ 0 & V(vertex\_b) & 0 & 0 & 0 \\ 0 & 0 & V(vertex\_c) & 0 & 0 \\ 0 & 0 & 0 & V(vertex\_d) & 0 \\ 0 & 0 & 0 & 0 & V(vertex\_e) \end{pmatrix}$$

**Adjacency matrix ($M_A$)**

$$\begin{pmatrix} 0 & V(edge\_ba) & V(edge\_ca) & V(edge\_da) & V(edge\_ea) \\ V(edge\_ab) & 0 & V(edge\_cb) & V(edge\_db) & V(edge\_eb) \\ V(edge\_ac) & V(edge\_bc) & 0 & V(edge\_dc) & V(edge\_ec) \\ V(edge\_ad) & V(edge\_bd) & V(edge\_cd) & 0 & V(edge\_ed) \\ V(edge\_ae) & V(edge\_be) & V(edge\_ce) & V(edge\_de) & 0 \end{pmatrix}$$

**Figure S6. The Diagram of Graph, Degree Matrix and Adjacency Matrix.**

## The details of supervised CNN generator(CNN-Gen) and weakly-supervised perovskite GAN(WSP-GAN)

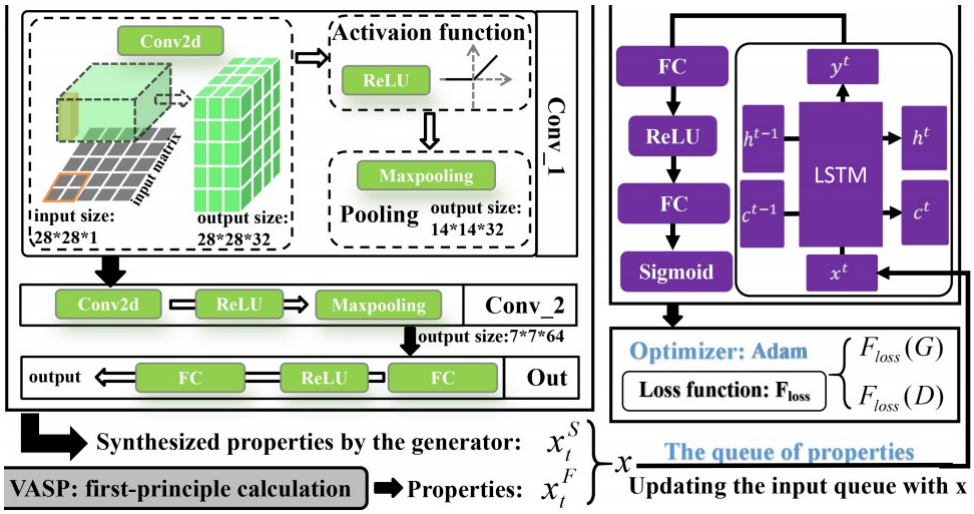



**Figure S7. The architecture of weakly-supervised perovskite GAN. The convolutional neural networks(Conv2d) and fully-connected neural networks(FC) in the generator(WSP-Gen); the scheme of input property queues; the long short term memory in the discriminator.**

LSTM belonged to the Recurrent Neural Network (RNN). Herein, it was adopted as the core layer of the discriminator and depended on the module of "torch.nn.LSTM". For each input queue, these inner parameters of LSTM were calculated:

$$i_t = \sigma(W_{ii}x_t + b_{ii} + W_{hi}h_{t-1} + b_{hi}) \quad (5)$$

$$f_t = \sigma(W_{if}x_t + b_{if} + W_{hf}h_{t-1} + b_{hf}) \quad (6)$$

$$g_t = \tanh(W_{ig}x_t + b_{ig} + W_{hg}h_{t-1} + b_{hg}) \quad (7)$$

$$o_t = \sigma(W_{io}x_t + b_{io} + W_{ho}h_{t-1} + b_{ho}) \quad (8)$$

$$c_t = f_t \otimes c_{t-1} + i_t \otimes g_t \quad (9)$$

$$h_t = o_t \otimes \tanh(c_t) \quad (10)$$

where $h_t$ was the hidden state of updating queue t, $c_t$ was the cell state of updating queue t, $x_t$ was the input of updating queue t, $h_{t-1}$ is the hidden state of the layer of updating queue t-1or the initial hidden state, and $i_t$, $f_t$, $g_t$, $o_t$ are the input, forget, cell, and output gates, respectively. $\sigma$ is the sigmoid function, and $\otimes$ was the Hadamard product. The $h_t$ and $c_t$ were used to the processing of forward function in the LSTM.

Herein, the LSTM improved the ability to grab data distribution from the training process of the input queue. Even if the training of WSP-GAN was performed with 3-5 samples, the effective gradient was calculated for the optimization of WSP-GAN. The performance of WSP-GAN also depended on the equilibrium between the generating ability of G and the discriminating ability of D.

Conv2d was the module of convolution layers in the torch. Herein, we listed the details of the parameters of Conv2d, the output value of this layer with input size (N, $C_{in}$, H, W) and output (N, $C_{out}$, $H_{out}$, $W_{out}$) could be described as:

$$out(N_i, C_{out_j}) = bias(C_{out_j}) + \sum_{k=0}^{Ci_n - 1} wight(C_{out_j}, k) \oplus input(N_i, k) \tag{11}$$

where $\oplus$ was the valid 2D cross-correlation operator, N was the batch size, C denoted a number of channels, H was the height of input planes, and W was the width. The Conv2d function owned these input parameters: in_channels, out_channels, kernel_sizes, stride, padding, and so on, which could be found in the python file of WSP-GAN in our github. “Stride” was a single number or a tuple and controlled the stride of the cross-correlation; “padding” controlled the amount of implicit zero-padding on both sides for padding number of points for each dimension.

In our works, the CNN layers worked as the core to deal with the PXRD data of crystal structures. Benefiting from the invariability of translation and rotation of the CNN, it was added into the supervised GCN and WSP-GAN.

## The loss function and error of quantum Auto-Encoder networks

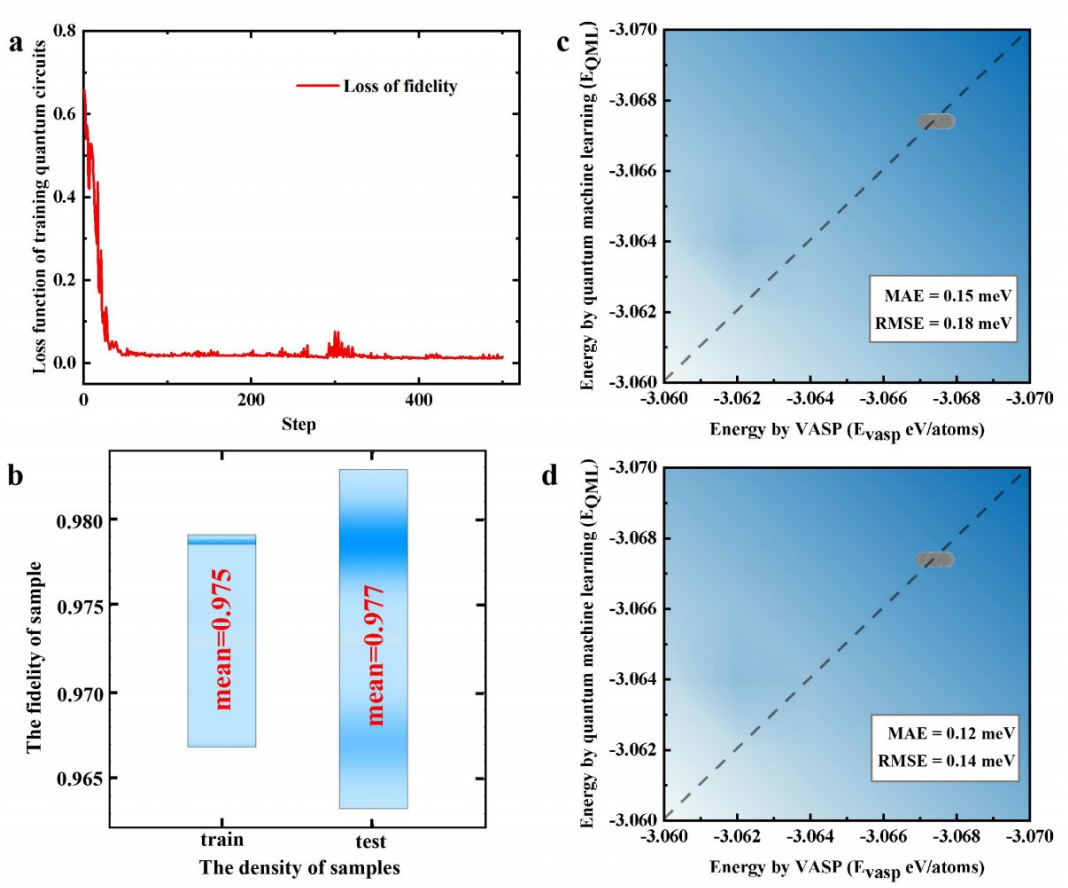



**Figure S8. (a) the Loss function curve of quantum AutoEncoder with quantum CNN as encoder network; (b) the density distribution of the fidelity of samples; (c) and (d) the comparison chart of prediction results and sample labels at the test-set and train-set, respectively.**

## The t-SNE based visualization for the results of SVM regression model with linear and quantum kernel.

The details about SVM regression model(SVR) and SVR with quantum kernel(QSVR) could be found via the link:

https://github.com/HuByHu/PCCP-code-2022

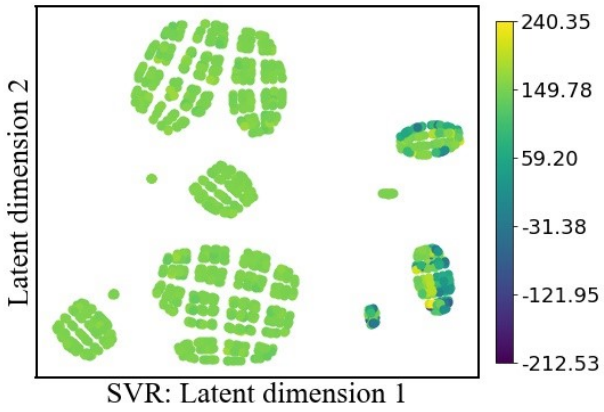



**Figure S9. A The t-SNE based visualization for the results of SVM regression model with linear kernel.**

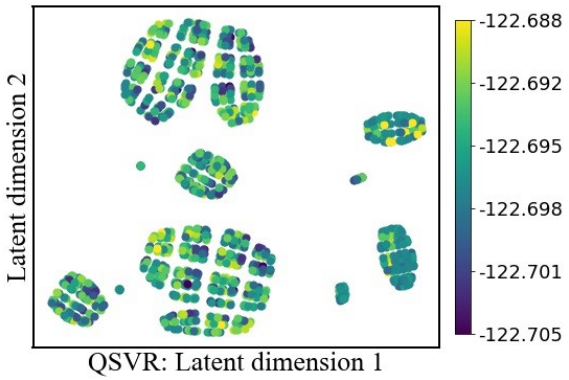



**Figure S9. B The t-SNE based visualization for the results of SVM regression model with quantum kernel.**