Ultrafast Bragg Coherent Diffraction Imaging of Epitaxial Thin Films using Deep Complex-valued Neural Networks

Xi Yu[1*], Longlong Wu[2†], Yuewei Lin[1], Jiecheng Diao[3], Jialun Liu[3], Jörg Hallmann[4], Ulrike Boesenberg[4], Wei Lu[4], Johannes Möller[4], Markus Scholz[4], Alexey Zozulya[4], Anders Madsen[4], Tadesse Assefa[2,5], Emil S. Bozin[2], Yue Cao[6], Hoydoo You[6], Dina Sheyfer[6], Stephan Rosenkranz[6], Samuel D. Marks[7], Paul G. Evans[7], David A. Keen[8], Xi He[2,9], Ivan Božović[2], Mark P.M. Dean[2], Shinjae Yoo[1], Ian K. Robinson[2,3‡]

[1]Computational Science Initiative, Brookhaven National Laboratory, Upton, NY 11973, USA

[2]Condensed Matter Physics and Materials Science Department, Brookhaven National Laboratory, Upton, NY 11973, USA

[3]London Centre for Nanotechnology, University College London, London WC1H 0AH, UK

[4]European X-Ray Free-Electron Laser Facility, Holzkoppel 4, 22869 Schenefeld, Germany

[5]Linac Coherent Light Source, SLAC National Accelerator Laboratory, Menlo Park, CA 94025

[6]Materials Science Division, Argonne National Laboratory, Lemont IL 60439, USA

[7]Materials Science & Engineering, University of Wisconsin, Madison, WI 53706, USA

[8]ISIS Facility, Rutherford Appleton Laboratory, Harwell Campus, Didcot, Oxfordshire OX11 0QX, UK

[9]Department of Chemistry, Yale University, New Haven CT 06520, USA.

*Correspondence authors' e-mails: *xyu1@bnl.gov, †lwu@bnl.gov, ‡irobinson@bnl.gov*

SUPPLEMENTARY NOTE 1: Complex-valued CNN architecture

Layer	Output Shape
CConv2d	[64,32,64,64,2]
CLeakReLU	[64,32,64,64,2]
CConv2d	[64,32,64,64,2]
CLeakReLU	[64,32,64,64,2]
CMaxPool2d	[64,32,32,32,2]
CConv2d	[64,64,32,32,2]
CLeakReLU	[64,64,32,32,2]
CConv2d	[64,64,32,32,2]
CLeakReLU	[64,64,32,32,2]
CMaxPool2d	[64,64,16,16,2]
CConv2d	[64,128,16,16,2]
CLeakReLU	[64,128,16,16,2]
CConv2d	[64,128,16,16,2]
CLeakReLU	[64,128,16,16,2]
CMaxPool2d	[64,128,8,8,2]
CConv2d	[64,256,8,8,2]
CLeakReLU	[64,256,8,8,2]
CConvTrans2d	[64,128,16,16,2]
CConv2d	[64,128,16,16,2]
CLeakReLU	[64,128,16,16,2]
CConv2d	[64,128,16,16,2]
CLeakReLU	[64,128,16,16,2]
CConvTrans2d	[64,64,32,32,2]
CConv2d	[64,64,32,32,2]
CLeakReLU	[64,64,32,32,2]
CConv2d	[64,64,32,32,2]
CLeakReLU	[64,64,32,32,2]
CConvTrans2d	[64,32,64,64,2]
CConv2d	[64,1,64,64,2]

Supplementary Table 1 Complex-valued CNN architecture. The size of the input is [64,1,64,64,2], where the first dimension represents the batch size, the second dimension represents the coherent diffraction pattern's intensity channel, the third and fourth dimension represent the height and width of the image, and the last dimension represents the real and imaginary part. We first downsample the input from 64×64 to 8×8 using the encoder and then up-sample to its original size using the decoder.

SUPPLEMENTARY NOTE 2: Implementation Details

All machine learning model training experiments are implemented on a single V100 GPU. More details are reported in Supplementary Table 2.

	Details
Parameters of cosine annealing scheduler	T_max=500, $lr\_\min = 0.0001$
Weight initialization	xavier_uniform (gain=1)
Values for the ADAM optimizer	$\beta_1 = 0.9$ $\beta_2 = 0.999$
Batch size	64 for simulated data and 1 for experimental data
Activation function of the intermediate layer	Leaky ReLu with $negative\ slope = 0.2$.
CUDA versions and driver	Python: 3.8.13 PyTorch: 1.8.0 Torchvision: 0.9.0 CUDA: 11.1 CUDNN: 8005

Supplementary Table 2. Implementation details for ML model training

SUPPLEMENTARY NOTE 3: Complex-valued Convolution and Activation Function

We first introduce the complex convolutional operation. Let $I = I_\mathrm{r} + jI_\mathrm{i}$, as a complex-valued input, where I_r and I_i are the real and imaginary parts of I. Given a complex-valued convolutional filter $k = k_\mathrm{r} + jk_\mathrm{i}$, the complex-valued convolutional operation based on a complex-valued input and a complex-valued filter can be described as follows:

$$I * k = (I_\mathrm{r} + jI_\mathrm{i}) * (k_\mathrm{r} + jk_\mathrm{i}) = (I_\mathrm{r} * k_\mathrm{r} - I_\mathrm{i} * k_\mathrm{i}) + j(I_\mathrm{r} * k_\mathrm{i} + I_\mathrm{i} * k_\mathrm{r}). \quad (1)$$

Here, we can split a complex-valued convolutional operation into two separate real-valued convolutions (*i.e.*, $k_\mathrm{r}, k_\mathrm{i}$), and calculate the output based on Eq. 1. Similarly, for complex-valued transpose convolution operation, we can just replace k_r and k_i as real-valued transpose convolution operation. Figure 2b in the main text illustrates the complex-valued convolutional operation.

In real-valued neural networks, rectified linear unit (ReLU) is the most widely used activation function. However, for a complex signal, the real or imaginary component could be negative in some cases. Accordingly, we employ a Leaky ReLU activation function, which has a small slope for negative values on both real and imaginary parts. The definition of Leaky ReLU and complex Leaky ReLU are shown as follows:

$$LeakyReLU(x) = \begin{cases} x, \; x \geq 0 \\ negative\ slope\ \times x, \; otherwise, \end{cases} \tag{2}$$

$$CLeakyReLU(z) = LeakyReLU(z_{\mathrm{r}}) + j * LeakyReLU(z_{\mathrm{i}}), \tag{3}$$

where $z = z_{\mathrm{r}} + j * z_{\mathrm{i}}$, and we set $negative\ slope = 0.2$.

The definition of complex-valued Max-pooling becomes:

$$CMaxpool(z) = Maxpool(z_{\mathrm{r}}) + j * Maxpool(z_{\mathrm{i}}). \tag{4}$$

SUPPLEMENTARY NOTE 4: Definition of χ^2 error and SSIM

We calculated the χ^2 error for the modulus of the diffraction pattern in reciprocal space defined as:

$$\chi^2 = \frac{\sum\left(\sqrt{I_{\mathrm{e}}} - \sqrt{I_{\mathrm{m}}}\right)^2}{\sum I_{\mathrm{m}}}, \tag{5}$$

where I_{e} is the reconstructed X-ray diffraction intensity and I_{m} is the true or experimental diffraction intensity.

The SSIM index between two image arrays $\boldsymbol{x}$ and $\boldsymbol{y}$ is defined as:

$$\mathrm{SSIM} = \frac{\left(2\mu_x\mu_y + c_1\right)\left(2\sigma_{xy} + c_2\right)}{\left(\mu_x^2 + \mu_y^2 + c_1\right)\left(\sigma_x^2 + \sigma_y^2 + c_2\right)}, \tag{6}$$

where $\mu_{\boldsymbol{x}}$ and $\mu_{\boldsymbol{y}}$ denote the pixel sample mean of $\boldsymbol{x}$ and $\boldsymbol{y}$, respectively. $\sigma_{\boldsymbol{x}}^2$, $\sigma_{\boldsymbol{y}}^2$, and $\sigma_{\boldsymbol{xy}}$ represent the variance of $\boldsymbol{x}$ and variance of $\boldsymbol{y}$**,** and covariance of $\boldsymbol{x}$ and $\boldsymbol{y}$, respectively. c_1 and c_2 are $(k_1L)^2$

and $(k_2L)^2$, respectively, where L is the dynamic range of pixel values. We used the default values with $k_1 = 0.01$ and $k_2 = 0.03$.

SUPPLEMENTARY NOTE 5: Training process and Quantitative evaluation

Figure 3a and b of the main text demonstrate the prediction amplitude and phase of simulated data. To further investigate the capability of complex-valued neural networks, we generate two types of simulated data based on phase-domain structures in real space, with different levels of complexity. The first has 15 domains with positions on a grid (see Ground truth column in Fig. 3a), and the second has about 50 domains with random positions (see Ground truth column in Fig. 3b). The phases of each domain are randomly generated over [-π,π]. The training process is as follows. We first generate 150,000 simulated data with different domain structures and then split the data into training, validation, and test sets. We utilize the same hyperparameters (e.g., batch size, epochs, learning rate, and optimizer) for both R-CNN and C-CNN. The learning curve for C-CNN on the validation set during the training is shown in Supplementary Figure 2.

Based on Supplementary Table 3, it is clear that C-CNN outperforms R-CNN in both types of domain architecture. Additionally, we examine the robustness of the C-CNN model by adding Gaussian white noise to the original diffraction patterns at different levels. Supplementary Fig. 2 shows the phase and amplitude SSIMs for varying noise levels. When predicting phase and amplitude, the C-CNN is robust to input noise levels of 5dB and 8dB for the two types, respectively.

	15 grid domains		50 random domains	
	C-CNN	R-CNN	C-CNN	R-CNN
Amp SSIM	**0.9954±0.0012**	0.9282±0.0026	**0.8891±0.0009**	0.5130±0.0010
Phase SSIM	**0.9200±0.0024**	0.6343±0.0063	**0.7075±0.0026**	0.5614±0.0021
χ^2 error	**0.0677±0.0006**	0.2460±0.0066	**0.1167±0.0006**	0.2802±0.0026

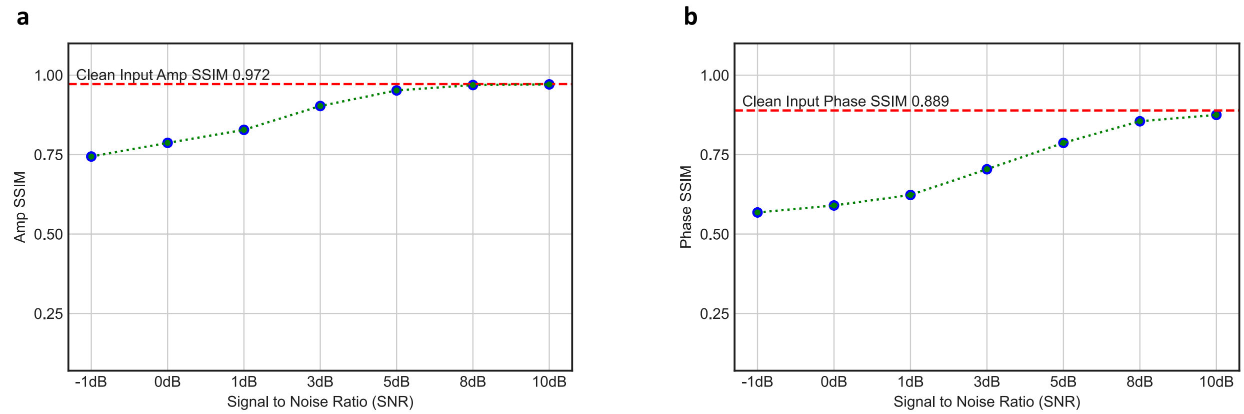
Supplementary Table 3 Mean and standard deviation of phase and amplitude SSIM in real space and χ^2 error in diffraction pattern space on different test set with complex-valued and real-valued neural networks. Larger is better for the SSIM criterion, and smaller is better for the χ^2 error. The best performance is highlighted.

SUPPLEMENTARY NOTE 6: Reproducibility

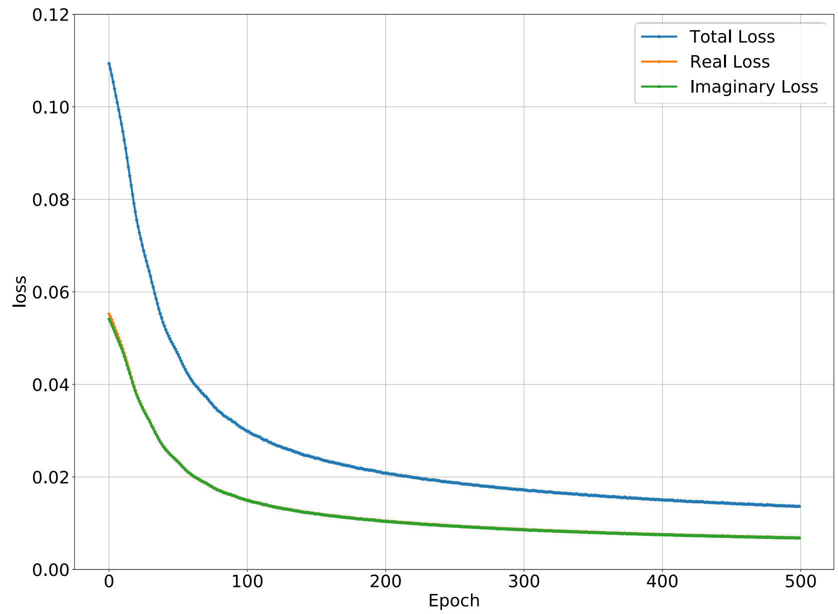
To quantify reproducibility, we obtained the results in real space on one test sample by training the model from scratch five times with different seeds. Specifically, we calculated the mean and standard deviation of pair-wise correlation, and χ^2 errors within 5 trails on one test sample, as shown in Supplementary Table 4. According to Supplementary Table 2, C-CNN performed an average correlation of 0.96 on predicted amplitude and 0.002 standard derivations of χ^2 errors with five trials, indicating high reproducibility. However, there is uncertainty regarding the predicted phase, because of arbitrary offsets propagating from the training. Nevertheless, the conclusion in Supplementary Figure 3 can still be trusted because it is based on highlighted amplitude regions in real space**.**

	mean $\pm$ std	minimum	maximum
Amplitude	0.961 ± 0.017	0.950	0.971
Phase	0.884 ± 0.121	0.851	0.949
χ^2 errors	0.036 ± 0.002	0.035	0.038

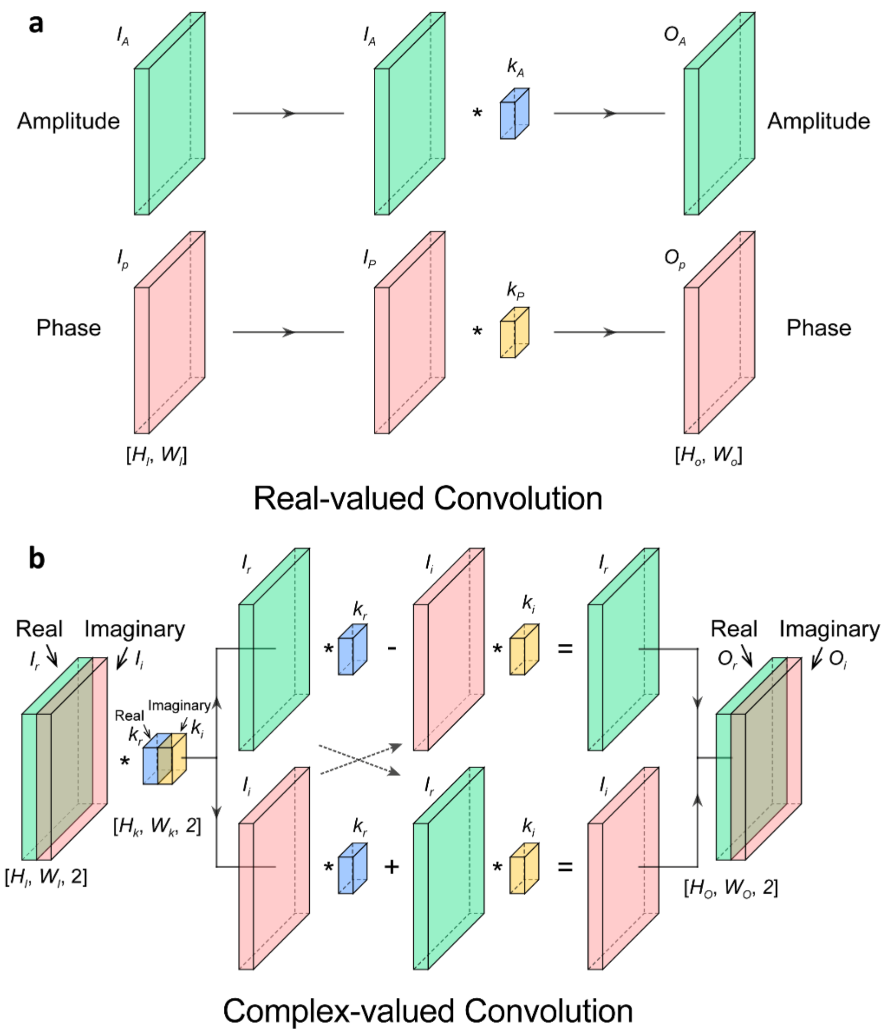
Supplementary Table 4 Mean and standard deviation of real space amplitude/phase pair-wise correlation coefficients, and χ^2 errors of diffraction pattern on one test sample with 5 trails.



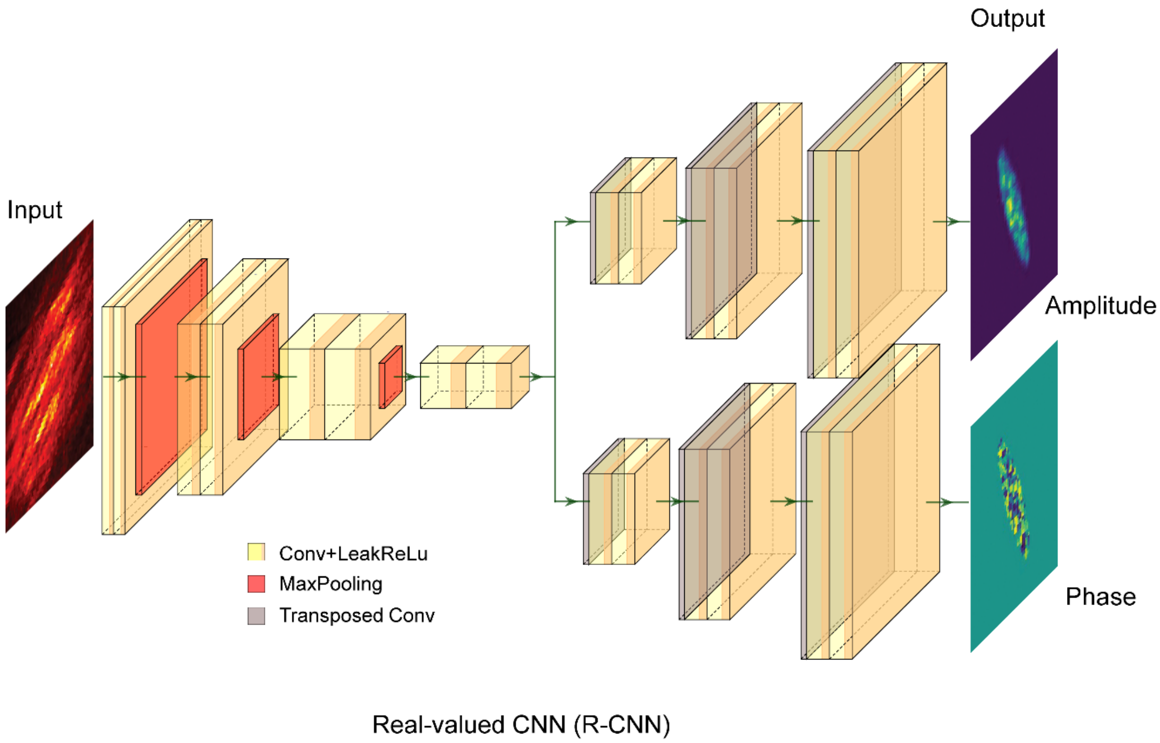
Supplementary Fig. 1 The SSIM of **a** amplitude and **b** phase in real space on the 15 grid domains simulated data by adding different levels of noise.



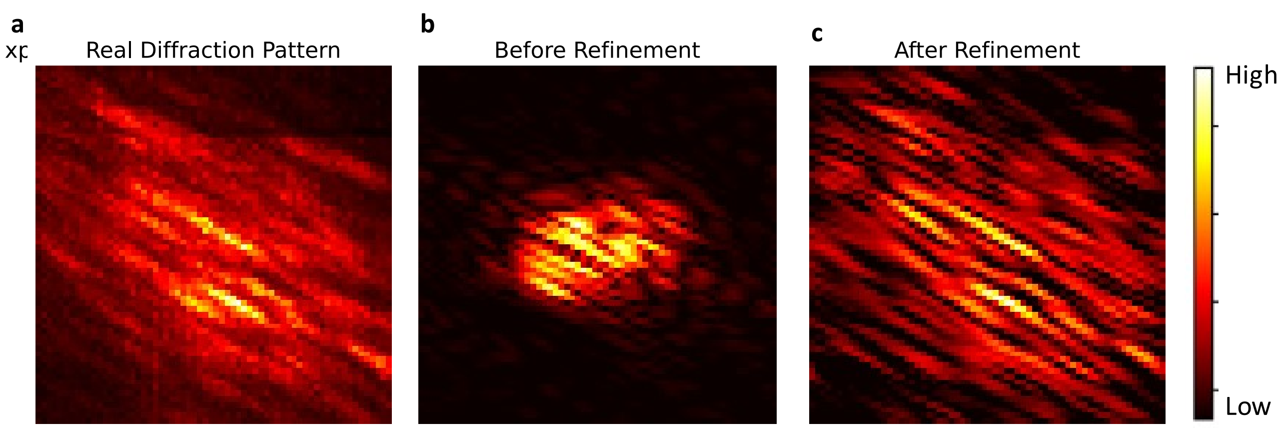
Supplementary Fig. 2 Total loss, real and imaginary loss on the validation set during the training process.



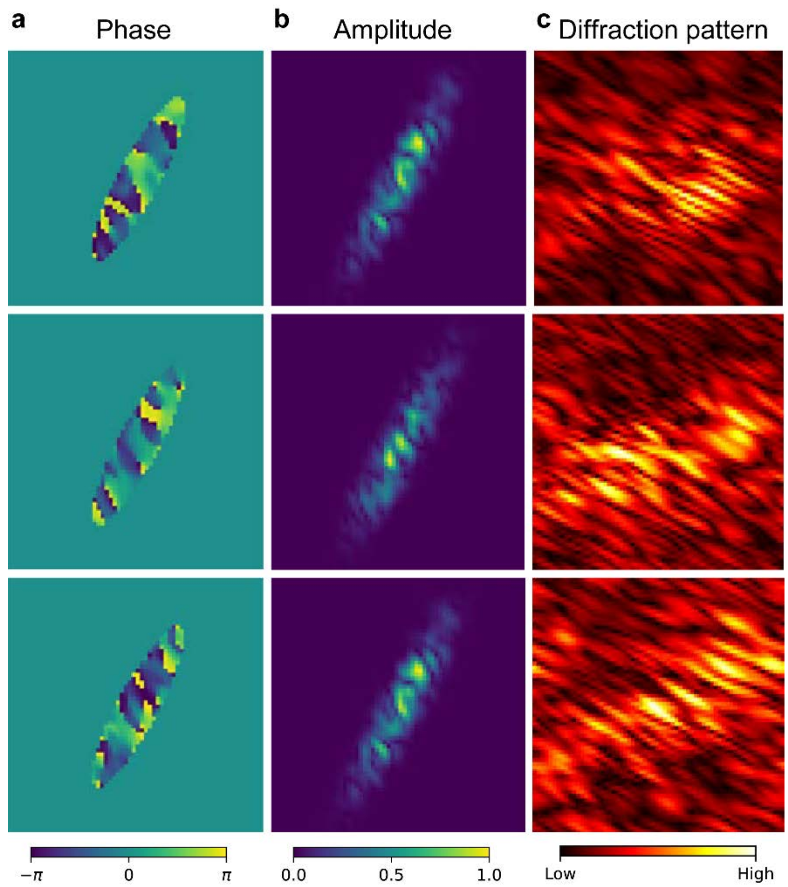
Supplementary Fig. 3 a Real-valued convolutional layer in R-CNN, where I_A and I_P denote input amplitude and phase, and O_A and O_P denote output amplitude and phase, respectively. **b** Complex-valued Convolutional layer in C-CNN, where I_r and I_i denote input real and imaginary, and O_r and O_i denote output amplitude and phase, respectively.



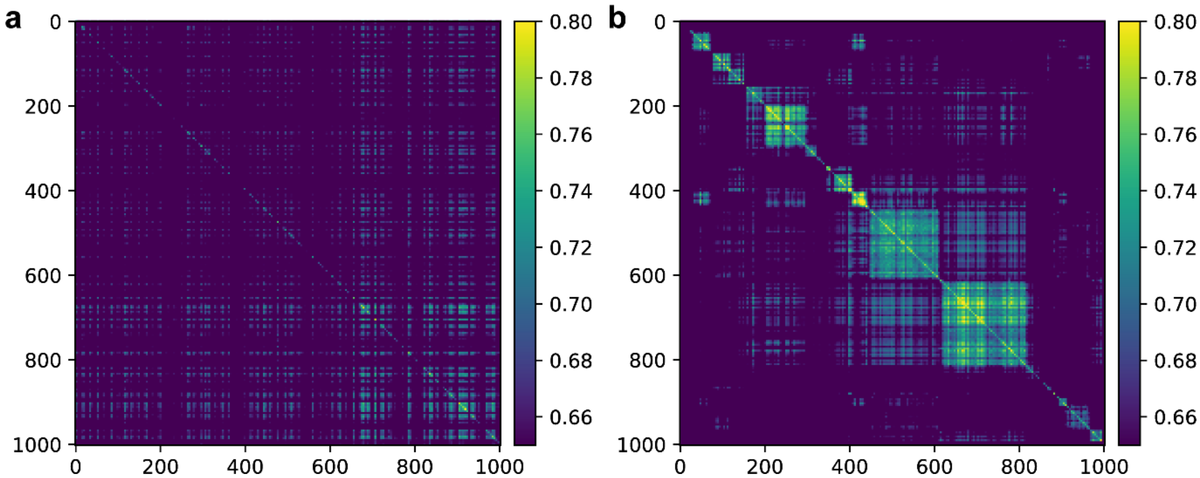
Supplementary Fig. 4 The architecture of R-CNN. The R-CNN has one real input (one channel) and two branches of real outputs (i.e., phase, and amplitude). Since the outputs of the two branches are independent, for these models, there is no connection between the amplitude and phase channels.



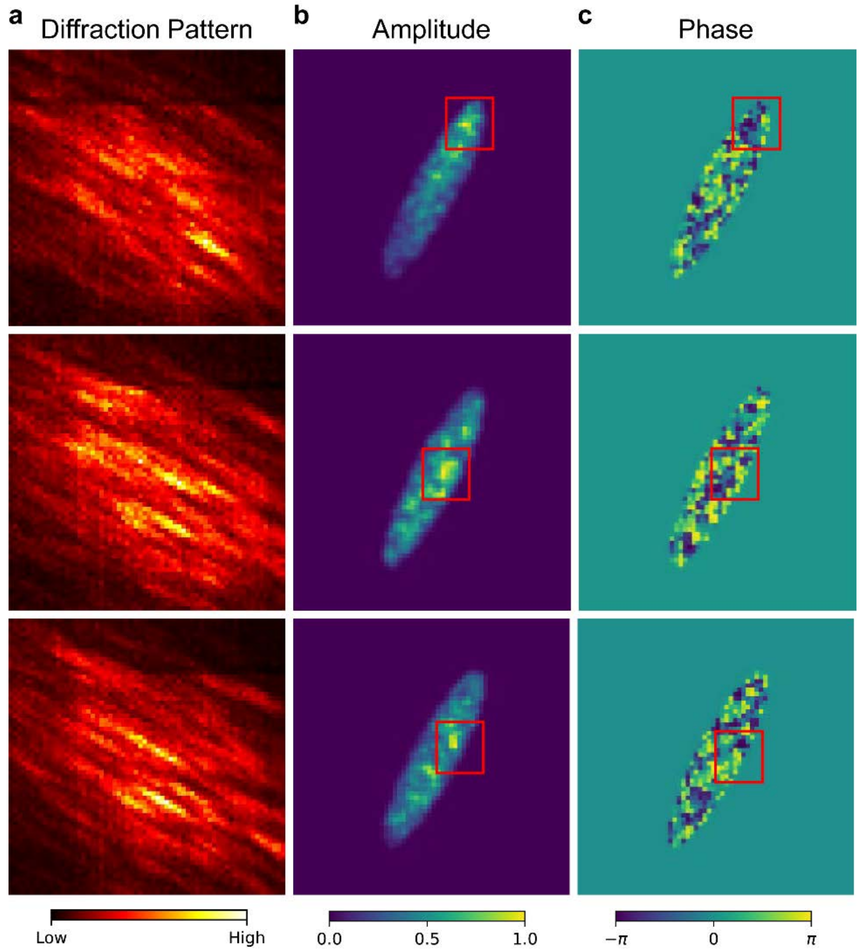
Supplementary Fig. 5 a real experimental diffraction pattern **b** before refinement **c** after refinement**.** The diffraction pattern from the pre-trained model on simulated dataset before refinement has similar stripes but is still far from the real diffraction pattern. However, there is considerable improvement with refined model by continuing unsupervised training on the experimental dataset.



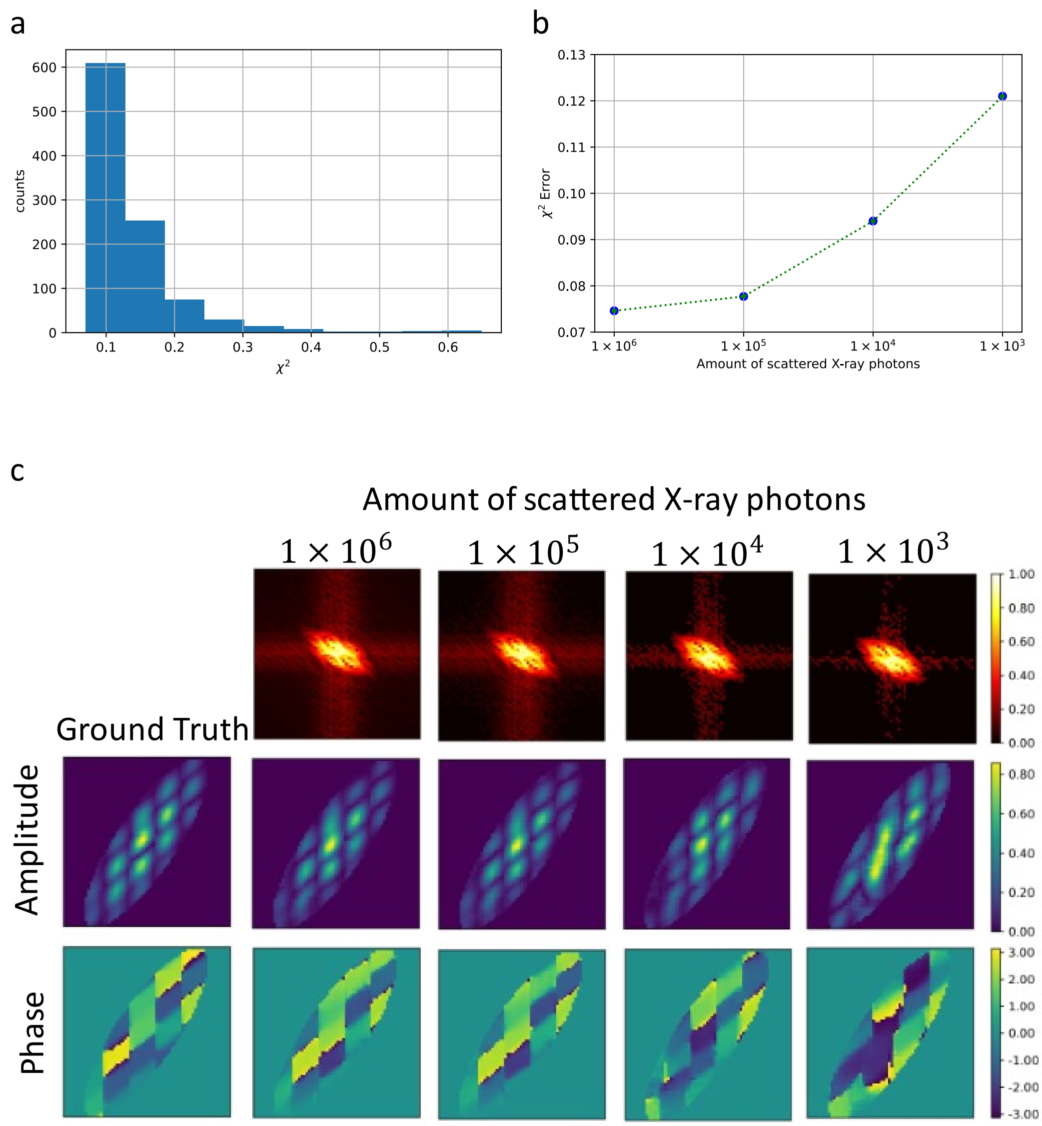
Supplementary Fig. 6 Synthetic data samples for pre-trained model on the real XFEL experimental data. a synthetic phase**. b** synthetic amplitude**. c** synthetic diffraction pattern**.** We chose the Gaussian support as the ellipsoid shape with semi-major axis 18 pixels and semi-minor axis 4 pixels, and a rotation of 28 degrees. As can be seen, the synthetic diffractions are similar to the real XFEL experimental data, and we use such synthetic data to train a model as the initiation and continue training on real XFEL experimental data.



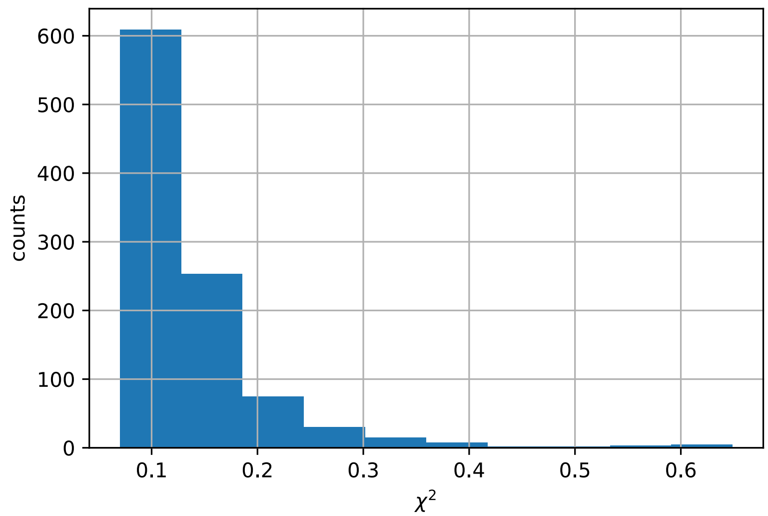
Supplementary Fig. 7 2D correlation coefficient heatmaps of real experimental XFEL data. We calculate correlation coefficient for these 1002 average images and group them into 20 different clusters. The pre-processing real XEFL data is obtained by averaging each cluster. We first calculate the correlations between 1002 average images shown in **a**, and then utilize the hierarchical clustering method to find hierarchy within the data and order the data in clusters. In specific, we employ the dissimilarity matrix computed by $d(x,y) = 1 - |\rho_{x,y}|$. After computing the dissimilarity matrix, we can group data hierarchically according to their dissimilarity, we visualize it in **b** and obtain 20 different clusters with threshold equal to 0.7.



Supplementary Fig. 8 Predicted amplitude and phase in real space for three different real experimental XFEL test data. a the experimental XFEL diffraction pattern samples**.** The red rectangle highlights the large value regions of the **b** predicted amplitude and the corresponding regions of **c** predicted phase. We observe the movement of large value regions in the predicted amplitude, but they are not related in the phase, which indicates that movement of the beam rather than internal sample fluctuations.



Supplementary Fig. 9 Performance of the C-CNN model on different amounts of scattered x-ray photons. **a** χ^2 errors *vs.* dose levels on simulated data. **b** Amplitude and phase perdition of a test simulated sample with different dose levels. We represent the χ^2 on the simulated data with different amounts of scattered X-ray photons in **a** and a test sample in **b**. As can be seen, our model achieves comparable results with 10^5 scattered photons and further lowering the number of scattered photons will degrade performance considerably.



Supplementary Fig. 10 Histogram of χ^2 in total 1002 real experimental diffraction patterns. We tested our trained model on a total of 1002 pulse-train consensus diffraction patterns of real experimental data. The histogram of χ^2 of the images shows that about 60% samples achieve χ^2 around the 0.1. However, the trained model achieves about 0.04 χ^2 in the average images because the input diffraction patterns are noisy and have some outliers which will degrade the performance.