Deep Learning for Bragg Coherent Diffraction Imaging: Detector Gap Inpainting and Phase Retrieval

Thesis

présentée et soutenue publiquement le

Pour l'obtention du titre de

Docteur de l'Université Grenoble Alpes
(mention Physique du rayonnement et de la matière condensée)

par
Matteo Masto

sous la direction de
Dr. Tobias Schülli, Dr. Vincent Favre-Nicolin, Dr. Steven Leake

Composition du Jury

XXXXXXXXXX	PR, XXXXXXXXXX	**Rapporteur**
XXXXXXXXXX	PR, XXXXXXXXXX	**Rapporteur**
XXXXXXXXXX	PR, XXXXXXXXXX	**Examinateur**
XXXXXXXXXX	PR, XXXXXXXXXX	**Examinateur**
Tobias Schülli	ESRF	**Directeur de Thèse**
Vincent Favre-Nicolin	ESRF UGA,	**Directeur de Thèse**
Steven Leake	ESRF	**Directeur de Thèse**

École Doctorale de Physique-Grenoble

Contents

Chapter 1

Introduction

1.1 Introduction

The present document is a draft of my PhD manuscript.

Chapter 2

Bragg Coherent Diffraction Imaging

2.1 Single crystal diffraction

2.2 Phase Problem

Chapter 3

Convolutional Neural Networks

3.1 Introduction on neural networks

3.2 Convolutional

3.3 U-Net and MSD-Net

Chapter 4

Deep Learning for Detector Gaps Inpainting

In this chapter, the "detector - gaps problem" in Bragg Coherent Diffraction Imaging and our approach to solve it using Deep Learning are discussed. The main state-of-the-art measures are presented briefly and the topic of image inpainting with Deep Learning is introduced. The focus will then shift to our works that led eventually to the optimal "Patching-based" approach that can also be found in the published paper entitled *"Patching-based deep learning model for the Inpainting of Bragg Coherent Diffraction patterns affected by detectors' gaps"* (https://doi.org/10.1107/S1600576724004163). The chapter is closed with some analyses of the performances of the DL models in a variety of simulated and experimental cases.

4.1 The "Gap Problem"

At time of writing, standard BCDI experiments employ pixelated photon counting detectors to acquire the diffraction patterns. These detectors can guarantee high spatial resolution, noise-free counting and fast read-out times. Two examples of these devices, currently used at the ID01 beamline are the MAXIPIX and EIGER detectors [1, 2]. These detectors are often built by tiling together several sensing chips in order to cover a larger area, and are typically bonded to an Application-Specific Integrated Circuit (ASIC) using bump bonding. This implies the presence, in the overall sensing region, of vertical and/or horizontal stripes that are not sensitive to the impinging radiation. The width of these lines varies depending on the device but normally does not exceed the equivalent of some tens of pixels. Specifically, the MAXIPIX detector, with sensing area of 516×516 pixels of $55\mu m \times 55\mu m$, is composed of four modules separated by $220\mu m$ wide gaps (equivalent of 4 pixels).

The EIGER detector instead has two types of larger gaps of 12 pixels and 38 pixels width. The detector gaps problem does not affect BCDI only, but it is shared among other x-ray techniques that deal with single photon-counting pixelated detectors and/or beamstops. We have seen in chapter 1.1 that during a BCDI scan the 2D images acquired by the detector are stacked to form a 3D array. This leads these lines to become planes of missing signal in the dataset. The problems arise when reconstructing the data affected by these gaps. In fact, these regions of non-physical zero intensity deceive the Phase Retrieval algorithms inducing the

presence of artifacts in the reconstructions[3].

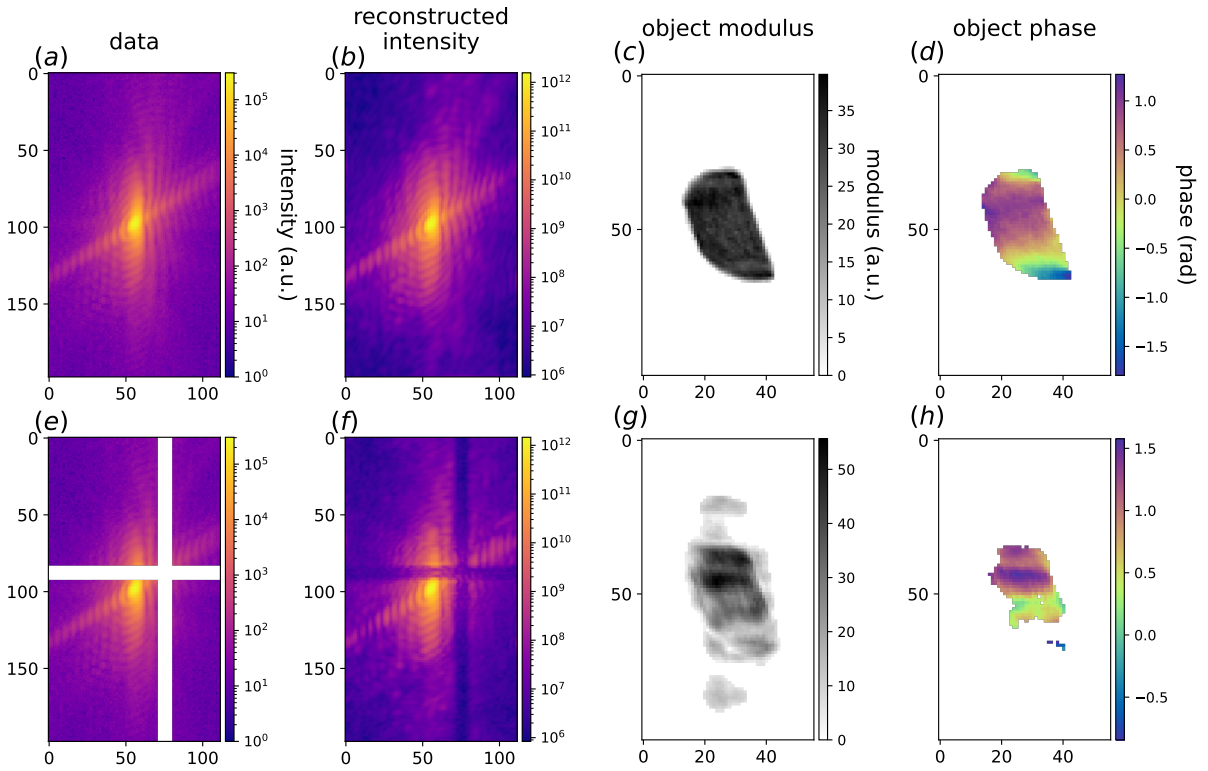


Figure 4.1: Effect of detector gaps in BCDI reconstructions (a) The central xz slice of an experimental diffraction pattern. **(b)** The same slice of the diffracted intensity calculated from the retrieved object. **(c - d)** xz slice of the modulus and phase respectively of the particle obtained from the phasing of the gap-less dataset. **(e)** Same slice as in **(a)** with an artificially added 9 pixel-wide, cross-shaped gaps to mimic the detector's ones. **(f)** The same slice of the diffracted intensity calculated from the retrieved object when not masking the gap regions. **(h - g)** xz slice of the modulus and phase respectively of the particle obtained from the phasing of the gap-affected dataset. The distortions caused by the gaps are evident.

It follows that the reliability of the reconstructions in this case is compromised as the strain distribution can be deeply affected by the artifacts. A good practice during standard BCDI experiments is to avoid the gaps by moving the detector if possible. However, this tends to be problematic for the case of high-resolution BCDI, i.e. when the diffraction pattern measurement extends to higher q-values, thus covering more than one sensing chip and necessarily crossing a gap region. Under these circumstances it becomes important to reduce the amount of artifacts deriving from the gaps.

4.2 State of the art

Here we will discuss the current strategies employed to treat the detector gaps. As someone could argue, the simplest yet not practical, solution would be to slightly move the detector sideways and acquire a second full scan with the gap hiding a different region of the same Bragg peak, and then merge the two measurements into a single gap-less one. This would more than double the acquisition time making it, de facto, never an option during standard

experiments.

The PyNX software, routinely used for the BCDI phase retrieval at ID01, allows the user to define a mask of the gap regions and ignore those pixels during the execution. In this way the quality of the reconstruction improves, but one can still notice the presence of high-frequency oscillations appearing in both object's modulus and phase. The origin of these artifacts can be found in the diffracted intensity calculated from the reconstructed particle as one can clearly see that the gap-regions is filled with non-physical high intensity (see Fig. 4.2)

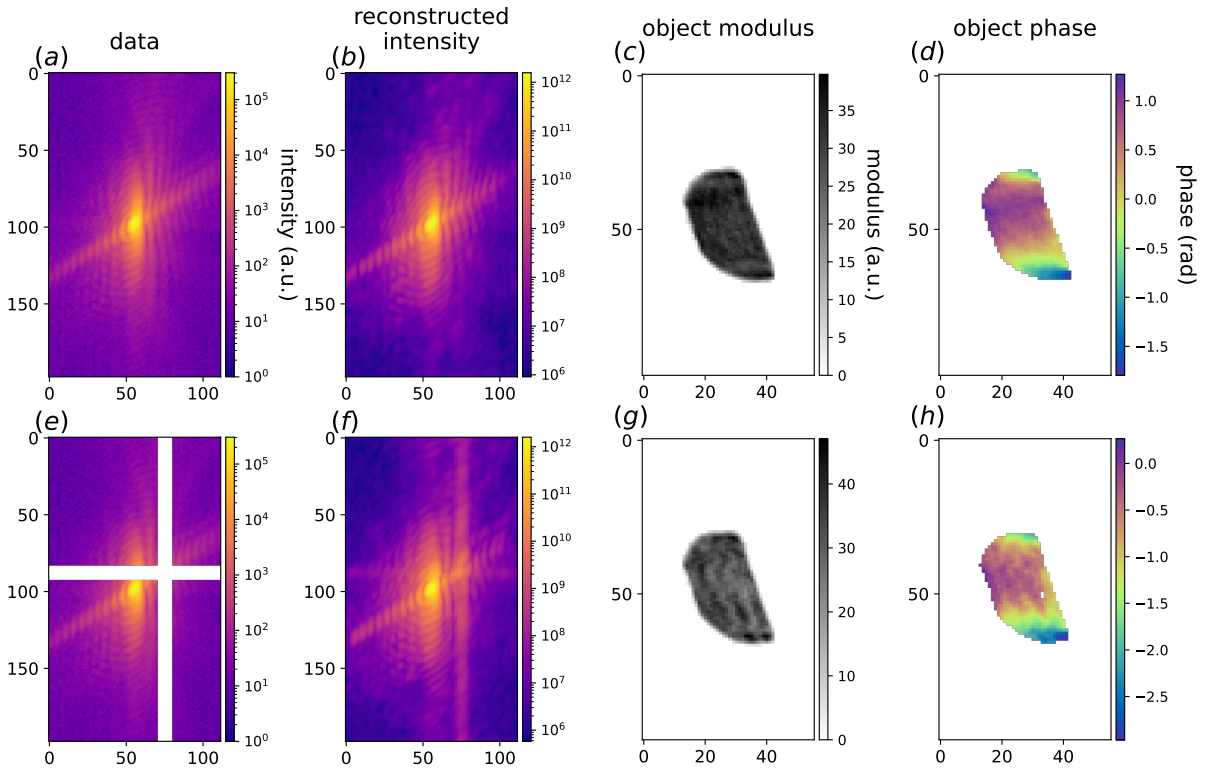


Figure 4.2: Masking the gap region during phasing (a) The central xz slice of an experimental diffraction pattern. **(b)** The same slice of the diffracted intensity calculated from the retrieved object. Comparing this figure with 4.1**(b)** one can see that when excluding the gap region from the phasing with a mask, the calculated intensity shows bright non-physical streaks instead of the gaps. **(c - d)** xz slice of the modulus and phase respectively of the particle obtained from the phasing of the gap affected data with a mask of the gap regions. Despite the much higher quality of the reconstruction, one can notice some oscillatory artifacts appearing in both the modulus and the phase of the retrieved object.

Another, more invasive, option is to *fill* these gaps with an estimate of the intensity distribution that would be there, before the phase retrieval. These tasks of filling gap in images is usually referred to as "inpainting". The following paragraph mentions the most relevant inpainting methods to give a context for our work.

4.2.1 Background on Image Inpainting Research

Computational image inpainting has been widely studied in the field of photography and imaging for many years [4, 5]. The inpainting problem can be defined as the task of utilizing

known information extractable from the image, to repair the parts where this information is missing, where for known information the colors, the textures and the semantic features are intended. In the history of image inpainting a clear-cut can be observed when deep learning methods have started to be employed. For traditional inpainting, different techniques have been explored, from the texture synthesis methods pioneered by Efros and Leung [6] to the use of PDEs as Navier-Stokes equations proposed by Bertalmio *et al.* [7] and then again from sparse representations [8] to hybrid methods combining variational and statistical methods [9]

More recently instead, Deep Learning models, headed by Convolutional Neural Networks (CNN), have taken the place of more traditional methods as they can attain higher accuracy for more complex inpainting tasks. By undergoing a training process, CNNs can "learn" to recognize and reproduce the semantic features of the training dataset, and thus leverage them during inference as additional information beside the colors and textures of the specific image to restore. As we have seen in **??**, the typical CNN architecture for image generation consists of an encoder, which retains the features of the input image and compresses them into a lower dimensional latent space, and a decoder, which is responsible for the generation of the output image starting from the latent space. The model are then trained according to a loss function that pushes the model's predictions to be close to a given ground truth reference. In some cases, the loss function can be replaced by another CNN that is trained to discriminate true images from the ones predicted by the model. These complementary networks are known as Generative Adversarial Networks (GAN), firstly proposed by Goodfellow *et al.* [10], and have also been used for image inpainting (e.g. [11]). Since reviewing the vast amount of works about CNN for image inpainting is beyond the scope of this thesis and for more information, we redirect the reader to the reviews published by Elharrouss *et al.* and Xu *et al.* [5, 12]. as well as this blog article [13]. For what concerns the application of DL based inpainting for scientific imaging, early works date back to 2018 as in the case of Sogancioglu *et al.* for x-ray human chest 2D radiographic images [14] and to 2020 for 2D microscopic images [15]. A couple of years later Tanny Chavez and coauthors published a paper comparing the performances of different CNN models for the inpainting of 2D x-ray diffraction images [16]. The work is precisely addressing the gap problem for x-ray detectors used for powder diffraction measurements and is awarding UNet and Mixed Scale Dense (MSD) models for the best performances on experimental data. The DL models outperform interpolations obtained with biharmonic functions across 7 and 17 pixel-wide gaps. This work has been of inspiration for the design of our DL model for BCDI gaps inpainting. In the same year, another work on DL based inpainting for x-ray detector gaps was published by Alfredo Bellisario and coauthors [17]. The authors tested a UNet-like model on the inpainting of 2D simulated, noiseless coherent diffraction patterns against gaps of different sizes (2 to 20 pixels) along the central row. The gaps were placed such that the center of the peak was covered, a choice that, as we will see later, yields better results than predictions on peripheral areas. To our knowledge, at the time of writing, no other works about deep learning based inpainting for X-ray detector gaps are present in the literature.

4.3 Model design: 2D case

On the heels of the last mentioned works we have started to tackle the detector gaps problem for BCDI using CNNs. For simplicity, we started off with 2D case, using simulated diffraction patterns and inpainting randomly placed vertical gaps of different width. First, we created a

training set of simulated data, composed of pairs of gap-affected images and corresponding gap-free ground truths, then built a U-Net-like model and trained it in a supervised fashion.

4.3.1 Dataset creation

The creation of training datasets of simulated 2D BCDI patterns for both the gap-inpainting and phase retrieval tasks has followed the procedure described in this paragraph.
In first place, once chosen the size of the array, a randomly shaped polygon is created in the center using `scipy.spatial.ConvexHull` function. This guarantees the object to have a compact support with homogeneous electron density as assumed for BCDI. Subsequently, a random phase field of the same size with variable phase range and correlation length is generated thus the complete complex object is formed. In order to make the object more realistic a Gaussian filter and Gaussian random noise are applied to the object's modulus, so to smoothen the edges and simulate real cases respectively. At this point the object is resized to the shape required to match the chosen oversampling ratio and the 2D Discrete Fourier Transform is computed. As last stage, Poisson noise is added to the diffraction patterns with different magnitudes to simulate various X-ray flux conditions.
Datasets contain a number of diffraction patterns in the order of thousands and for each of them the random variables are different as well as the oversampling ratios. In the datasets for the training of phase retrieval models, the reciprocal space phase corresponding to each diffraction pattern is saved as well and used as ground truth label. For inpainting tasks a randomly located vertical gap mask was created and applied to the intensity data. In some cases cross-shaped gaps were added instead to simulate the experimental condition of the Bragg peak in the vicinity of the corner of the sensing area. The size of the gaps was chosen to be consistent across the dataset and four different cases were studied (3px, 6px, 9px, 12px).

4.3.2 2D Model design

The 2D model that we have implemented is a U-Net that takes in input batches of 32 simulated BCDI patterns affected by both vertical and cross-shaped gaps. Each diffraction pattern is transformed into logarithmic scale to enhance the spatial features and then normalized between 0 and 1. This last passage is proven to be convenient to any DL model [18]. Regarding the logarithmic transformation, it is important to notice that in order to avoid problems for zero intensity values, the $\log(I + 1)$ was taken. The shape of each image was chosen to be of 128×128 pixels. The inputs go through five convolutional blocks inside each of which a convolutional layer, a Leaky ReLU activation function and a MaxPooling operation are applied. The tensor's dimensionality is so reduced down to 2×2 while the channel dimension is brought up to 768 filters while the kernel size is kept at 3×3. In this first model we directly pass this $(32, 2, 2, 768)$ tensor to the decoder that, mirroring the encoder, is composed of five blocks inside each of which there is a transposed convolutional layer that upsamples the feature maps (stride = 2) and a Leaky ReLU activation function. We also implemented skip connections connecting each encoder block to its corresponding shape-like decoder block. This measure has proven to be beneficial for the information flow between encoder-decoder [19]. The last activation layer of the model is a sigmoid function that guarantees an output bounded between 0 and 1

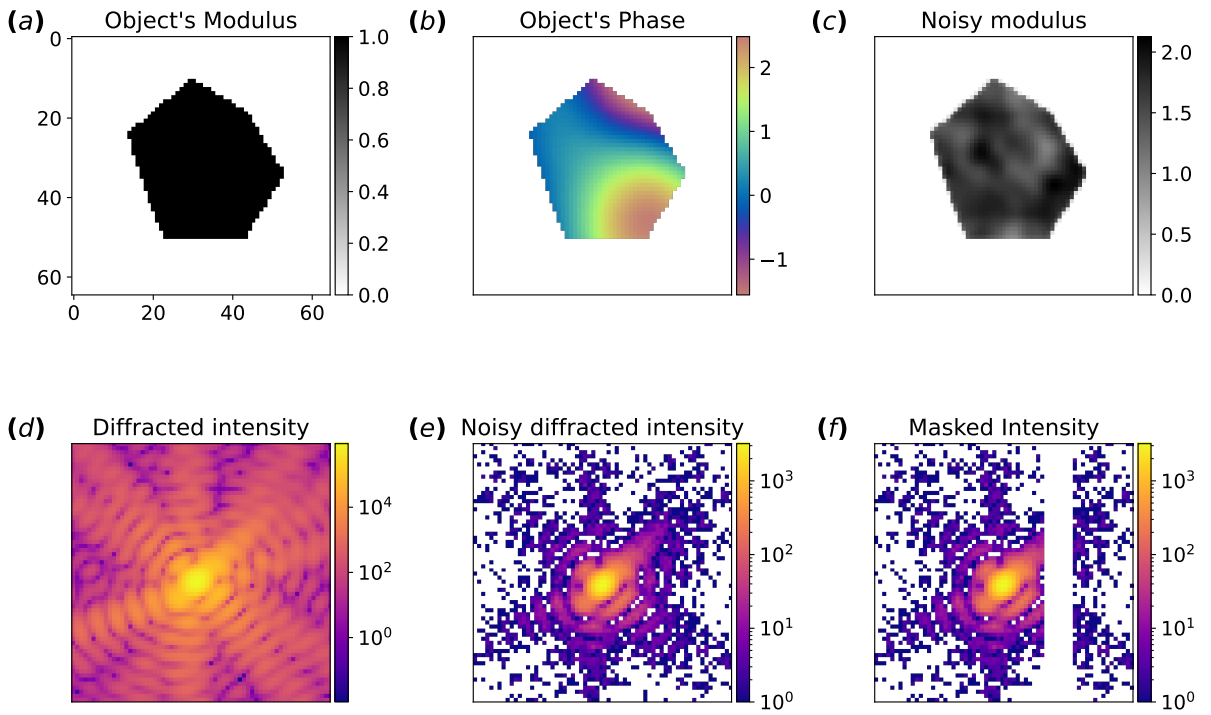


Figure 4.3: Steps for the simulation of a single 2D diffraction pattern (a) Simulated modulus of a 2D object with random shape and compact support. **(b)** Simulated object's phase **(c)** Object's modulus after smoothening the edges and adding random Gaussian noise. **(d)** Squared modulus of the Fourier Transform of the complex object (in log scale). The object is first padded with zeros to match the chosen oversampling ratio. **(e)** Poisson noise is added to the simulated diffracted intensity. **(f)** A 6 pixel-wide vertical gap is added to the diffracted intensity at a random position.

In the first place we utilized a simple Mean Squared Error (MSE) as cost function inside the gap region only, training the model on 12'000 diffraction patterns over 10 epochs, with ADAM optimizer and a learning rate of 10^{-4}. We have tested the Mean Absolute Error (MAE) and the Structural Similarity Index Measure (SSIM) [20] as well afterwards and compared the results after the same training. Here in Fig. 4.4 we report the comparisons for the 9 pixel-wide gap on a test simulated diffraction pattern. The accuracy scores were calculated using the Pearson Correlation Coefficient (PCC).

$$PCC = \frac{\sum_{i\in\text{gap}}(\mathbf{I}_i^{\text{true}} - \langle\mathbf{I}^{\text{true}}\rangle)(\mathbf{I}_i^{\text{pred}} - \langle\mathbf{I}^{\text{pred}}\rangle)}{\sqrt{\sum_{i\in\text{gap}}(\mathbf{I}_i^{\text{true}} - \langle\mathbf{I}^{\text{true}}\rangle)^2}\sqrt{\sum_{i\in\text{gap}}(\mathbf{I}_i^{\text{pred}} - \langle\mathbf{I}^{\text{pred}}\rangle)^2}}, \tag{4.1}$$

Where $\mathbf{I}$ is the intensity inside the gap.

In the light of these results, we have decided to discard the MAE metric and adopt instead the sum of MSE and SSIM. At last, another term computing the MSE between the *gradients* of the ground truth and predicted intensity inside the gap region was added in the definitive loss function.

Once established what we considered the best loss function, we have explored different models. Following the work of Chavez *et al.* mentioned above ([16]), we considered a Mixed-

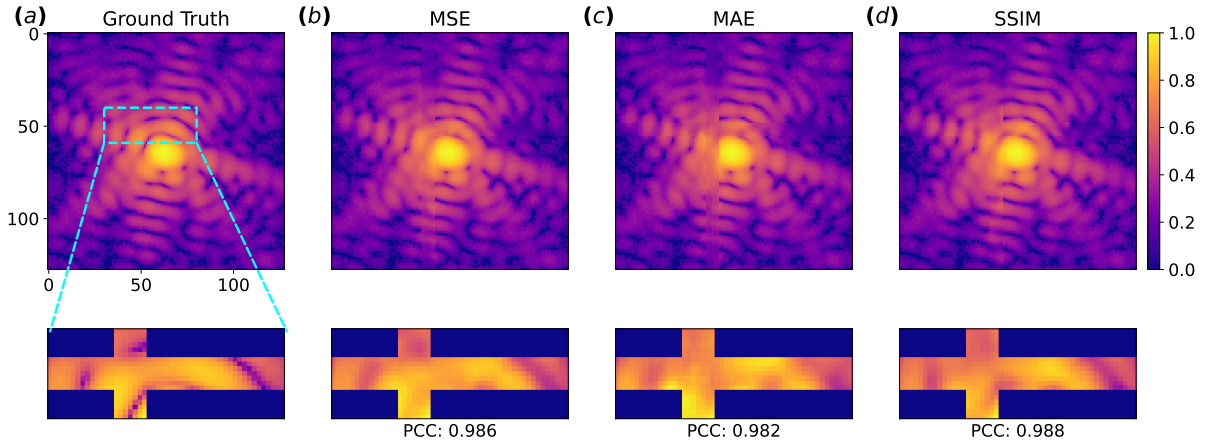


Figure 4.4: Comparison of different losses Results on a test simulated diffraction pattern for the inpainting of a 9 pixel-wide cross-shaped gap produced by the same UNet model trained for 10 epochs with different loss functions. **(a)** Shows the ground truth. **(b)** The prediction of the model trained with the MSE, **(c)** with the MAE, **(d)** with the SSIM. Corresponding accuracy scores calculated with the Pearson Correlation Coefficient (PCC) are shown as well. While MAE fails to recover the oscillations, SSIM yields better results.

Scale Dense Network (MSD-Net). The advantage of this type of networks is the significant reduction of trainable parameters, and the use of *dilated* convolutions with respect to U-Net ones. While the former property guarantees faster trainings and lower chances of overfitting, the latter enhances the capture of long-range correlations. Moreover, in a MSD-Net, the image's spatial dimensions are kept constant throughout the whole network as no downsampling nor upsampling is operated. The MSD-Net that we have used consists of sequential blocks in each of which the input is transformed by two different convolutional layers with growing dilation rates. Each output of the convolutional layers is concatenated to the input feature map and the result is passed to the following block. While the kernel size is kept constant to $3 \times 3 \times 3$ pixels the dilation rate increases linearly from 1 to 30. The last layer is a sigmoid function as well as for the U-Net. The total number of trainable parameters is in the order of 320'000, two orders of magnitude lower than the U-Net.
In order to combine the hierarchical dimensionality reduction of the U-Net with the fine-features capturing of the MSD-Net we have implemented a modified U-Net that adopts dilated convolutions inside the first three encoder blocks. In particular, they return the input tensor concatenated with the outputs of four dilated convolutional layers computed from the input. Dilation rates of (16,8,4,2), (10,5,3,1) and (5,3,2,1) were chosen respectively. As the MaxPooling operation down-samples the feature maps into smaller sizes, we limited the dilated convolutions to the first three blocks. Moreover, we utilized them in the encoder layers only as they are mostly used for feature extraction [21]. The characteristics of each model are summarized in form of pseudo - code in Table 4.1.

The three different models have been trained with a combined loss function (MSE + SSIM + MSE on the gradients) on the same training dataset for 10 epochs each. The results showed poor performances of the MSD-Net with respect to the two U-Nets. Slightly higher accuracy was achieved by our modified U-Net.

We conclude here the introductory studies on 2D simulated data. These preliminary tests

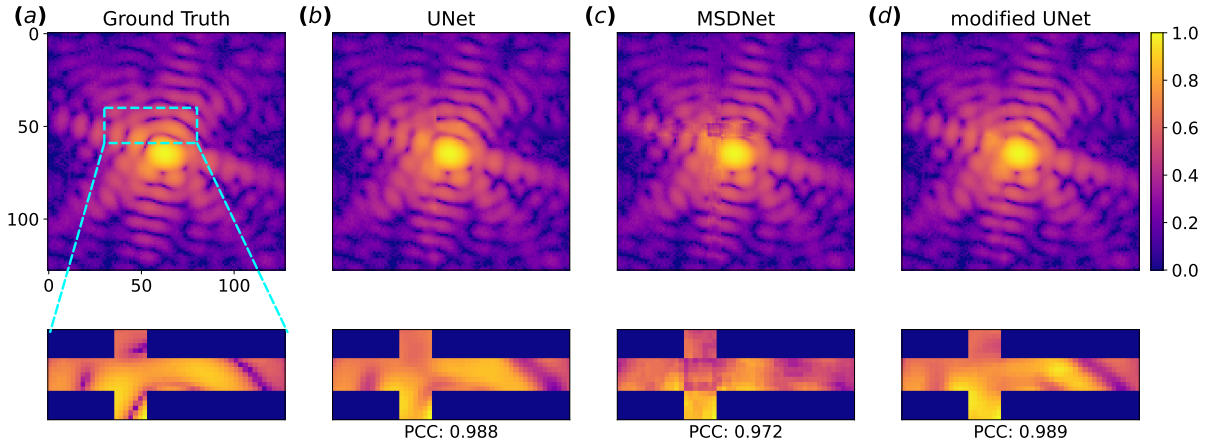


Figure 4.5: Comparison of different models Results on a test simulated diffraction pattern for the inpainting of a 9 pixel-wide cross-shaped gap using three different models trained with the same loss function. **(a)** Shows the ground truth. **(b)** The prediction of the U-Net, **(c)** of the MSD-Net, **(d)** of the modified U-Net. Corresponding accuracy scores calculated with the Pearson Correlation Coefficient (PCC) are shown as well.

served to get familiar with the different DL architectures and loss functions and select the optimal choices for the inpainting of BCDI detector gaps.

4.3.3 Accuracy VS Gap position

Before moving to the 3D case, it is worth spending a few words on the assessment of the DL model upon different conditions. We leave the assessment of the prediction accuracies against the gap size for the 3D case and will focus instead on two other evaluations. Namely, the accuracy as function of the position of the gap inside the diffraction pattern and the as a function of the oversampling ratio. For the first test we have simulated 150 2D diffraction patterns from random particles shapes, random oversampling ratios and Poisson noise intensity. For each diffraction pattern we have then placed a vertical 9 pixel-wide gap at all positions from left to right, computed the DL prediction and corresponding accuracy score when compared to the ground truth. The accuracy was again calculated with the Pearson Correlation Coefficient. We have then averaged this score for each gap position, over the 150 diffraction patterns and plotted the result as a function of the gap position. Fig. 4.6 shows the resulting curve that clearly highlights that the model performs better regions with high intensity. This can be qualitatively explained with different reasons: (i) central regions have larger features both because of the nature of the Bragg peak, and because of the lower noise level. This makes it easier for the model as it reduces the complexity of the prediction. (ii) As we move away from the center of the Bragg peak, the Signal to Noise Ratio (SNR) decreases, along with the *density of signal*. High accuracy scores in these regions would require the model to be able to predict noise correctly which is by definition impossible as it is an uncorrelated random process. One could argue that the accuracy curve would follow the statistical distribution of the gap positions inside the DL model training dataset. However, each mask has been applied at a position drawn from a discrete uniform probability function spanning in the full data size, thus we exclude this hypothesis.

	U-Net	MSD-Net	Unet_mod

block1	`def encoder_block(x_input, num_filters, ker):`
	`  s = Conv2D(num_filters, ker,' leaky_relu')(x_input)`
	`  x = MaxPool2D(2)(s)`
	`  return x, s`
		`def MSD_block(x, in_channels, dilations,kernel_size=3):`
		`  if isinstance(dilations, int):`
		`  dilations = [(j % 10) + 1 for j in range(dilations)]`
		`  out_channels = in_channels + len(dilations)`
		`  for d in dilations:`
		`  x1 = Conv2D(out_channels //2,kernel_size,1, dilation_rate= dilation, 'same', ' leaky_relu')(x)`
		`  x = tf.concat([x1,x] ,axis = -1)`
		`  return x, out_channels`
			`def encoder_block_mod(x_input, ker, num_filters, rate):`
			`  f = num_filters // 4`
			`  s = tf.concat([x_input] + [ Conv2D(f, ker, dilation_rate=r, ' leaky_relu')(x_input) for r in rate], axis=-1)`
			`  return MaxPool2D(2)(s), s`

block2	`def decoder_block(x_input, num_filters, ker, skip_input = None):`
	`  if skip_input is not None:`
	`  x_input = Concatenate()([ x_input, skip_input])`
	`  x = Conv2DTranspose( num_filters, ker, strides=2, 'leaky_relu') (x_input)`
	`  return x`
			`def decoder_block(x_input, num_filters, ker, skip_input = None):`
			`  if skip_input is not None:`
			`  x_input = Concatenate()([ x_input, skip_input])`
			`  x = Conv2DTranspose( num_filters, ker, strides=2, 'leaky_relu') (x_input)`
			`  return x`

body	`x, s1 = encoder_block(inputs, 48,3)`
	`x, s2 = encoder_block(x, 96,3)`
	`x, s3 = encoder_block(x, 192,3)`
	`x, s4 = encoder_block(x, 384,3)`
	`x, s5 = encoder_block(x, 768,3)`
	`x = Conv2D(1536,3, ' leaky_relu')(x)`
	`x = decoder_block(x,768,3)`
	`x = decoder_block(x,384,3, s5)`
	`x = decoder_block(x, 192,3,s4)`
	`x = decoder_block(x, 96,3,s3)`
	`x = decoder_block(x, 48,3,s2)`
	`x = Conv2D(24,5,'leaky_relu') (x)`
	`x = Conv2D(12,5,'leaky_relu') (x)`
	`x = Conv2D(6,5,'leaky_relu')( x)`
	`out = Conv2D(1,5,'sigmoid')(x )`
		`x,out_ch = MSD_block(inputs ,1,[1,2])`
		`x,out_ch = MSD_block(x,out_ch ,[3,4])`
		`...`
		`...`
		`x,out_ch = MSD_block(x,out_ch ,[31,32])`
		`out = Conv2D(1,3,'sigmoid')(x )`
			`x, s1 = encoder_block_mod( inputs,3,48,[16,8,4,2])`
			`x, s2 = encoder_block_mod(x ,3, 96,[10,5,3,1])`
			`x, s3 = encoder_block_mod(x ,3, 192,[5,3,2,1])`
			`x, s4 = encoder_block(x, 384 ,3)`
			`x, s5 = encoder_block(x, 768, 3)`
			`x = Conv2D(1536,3,'leaky_relu ')(x)`
			`x = decoder_block(x,768,3)`
			`x = decoder_block(x,384,3,s5)`
			`x = decoder_block(x,192,3,s4)`
			`x = decoder_block(x,96,3,s3)`
			`x = decoder_block(x,48,4,s2)`
			`x = Concatenate()([x, s1])`
			`x = Conv2D(24,5,'leaky_relu') (x)`
			`x = Conv2D(12,5,'leaky_relu') (x)`
			`x = Conv2D(6,5,'leaky_relu')( x)`
			`out = Conv2D(1,3,'sigmoid')(x )`

parameters	31,827,673	322,458	32,652,337

Table 4.1: Comparison of Unet, MSDNet, and Unet_mod components.

To conduct the second test, we simulated 150 diffraction patterns for the same particle varying gradually the oversampling ratio between 2 and 6. For each image we have then applied a 9 pixel-wide vertical gap at all X positions and performed the DL prediction. The accuracy scores have been averaged for each prediction in the same image and plotted against

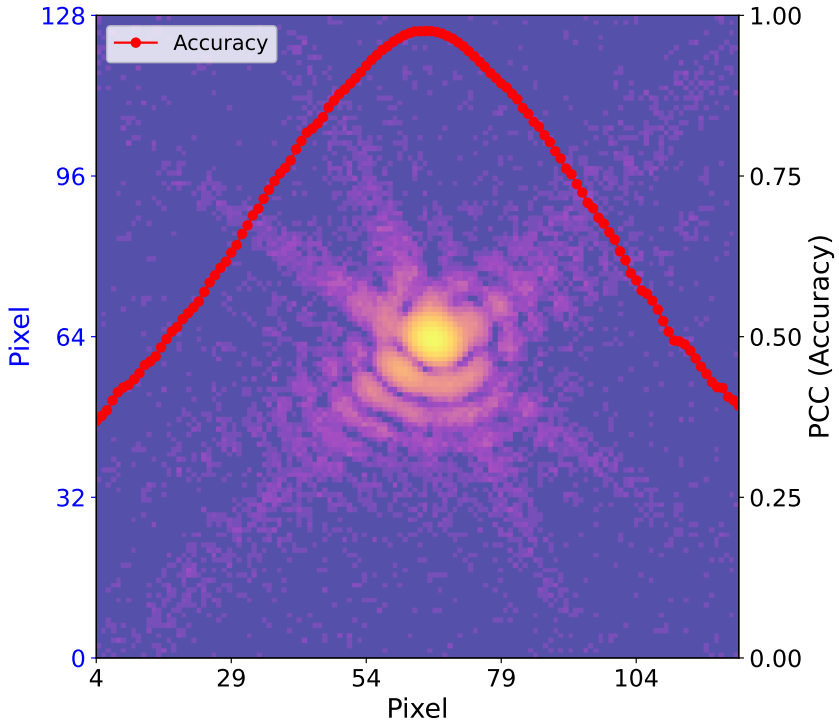


Figure 4.6: (Accuracy VS Gap position) Average Pearson Correlation Coefficient calculated over 150 9 pixel-wide vertical predicted gaps for each position of the gap inside the diffraction patterns. The model shows higher accuracies for high intensity regions.

the oversampling (Fig. 4.7). As expected from the above considerations, the model performs better for larger oversampling ratios, because of the bigger size of the features with respect to the gap width and because of the more uniform *density of signal*. About this last concept, it is worth clarifying that, for a given particle, the total amount of intensity in the diffraction patterns is in principle constant regardless of the oversampling ratio as it is fixed by Parseval theorem. However, if the size of the dataset is kept fixed for different oversamplings, the effect is the same of a zoom lens that increases or reduces the field of view. Therefore, while for low oversampling ratios the full peak is recorded, for higher ones the peak is cropped, and less intensity is present in the image. This effect, coupled with the typical radial intensity decay of Bragg peaks and the presence of Poisson noise, makes largely oversampled BCDI patterns having a smaller and more uniform *density of signal*, intended as the amount of information per pixel. On the contrary, for low oversampling ratio the *density of signal* is less uniform as it is high inside bright regions (lot of information concentrated in few pixels) and low in noisy regions far from the peak. It follows that in order to properly assess the accuracy against the oversampling ratio one should consider diffraction patterns over the same extent in Q-space, thus changing the size of the images. This more accurate evaluation was carried out for the 3D case and can be found in the next section.

4.4 3D case - Patching approach

When considering the 3D case, and especially the experimental conditions, there are a few practical issues that needs to be overcome. In fact, experimental BCDI datasets that are more

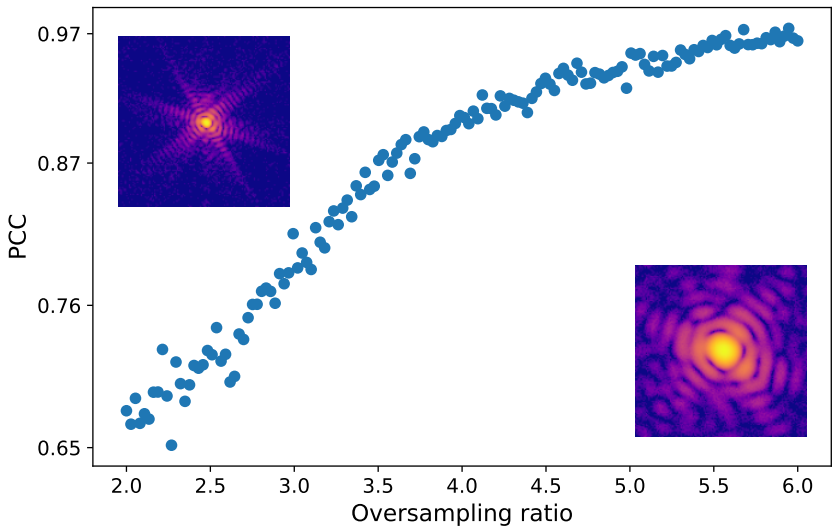


Figure 4.7: (Accuracy VS Oversampling ratio) Average Pearson Correlation Coefficient calculated over 280 9 pixel-wide vertical predicted gaps for each position of the gap inside the diffraction patterns. The model shows higher accuracies for high intensity regions.

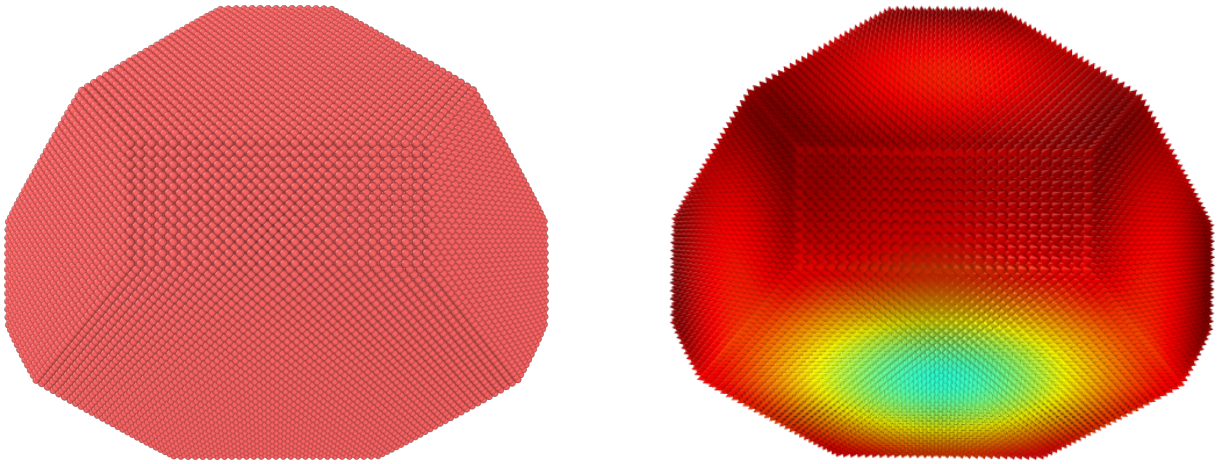
often affected by detector's gaps are necessarily large datasets (e.g. $512 \times 512 \times N_{\text{rocking\_steps}}$). Training a U-Net like model for 3D images of that size is overly expensive in terms of computing memory and time. Moreover, a common problem with this type of architectures is that the size of the images they can process is fixed by the first initialization. This means that one would need to resize, via binning or interpolation, the experimental datasets to the shape accepted by the DL model, and back to the original shape after the inpainting. Besides the impracticality, these operations are not recommended as they induce further modification and information loss to the original data. For these reasons we have opted for a patching approach that loosens these constraints while preserving sufficiently high accuracies.

The patching method exploits the regularity of the oscillations within BCDI datasets. The periodicity of the fringes in reciprocal space, peculiar property of this coherent diffraction technique, can be observed by eye and in many cases makes the prediction inside a gap region intuitively possible starting from just a few neighboring pixels. In our case we have decided to work with 32 pixel-sided cubic sub-volumes (patches from now on) cropped out of entire diffraction patterns. Among the "GPU-friendly" tensor sizes [22] we opted for 32 as good trade-off between amount of contained information and computing power required for training and inference.

4.4.1 Dataset creation

The training dataset consists of 50% patches from experimental data and 50% from simulated data. The experimental measurements were acquired at the ID01 beamline of the ESRF during different beamtimes on different particles. Namely, (i) Pt particles dewetted on sapphire and YSZ (yttria–stabilized zirconia) with Winterbottom shape, measured under various temperatures and gas conditions, (ii) Pd and PdCe particles on glassy carbon, with Wulff shape, measured in an electrochemical environment following hydrogen loading. (iii) Ni particles on sapphire

undergoing changes during CO_2 adsorption and (iv) cubic $CaCO_3$ particles on glassy carbon. The synthetic diffraction patterns were instead simulated in three steps. The first step consisted in the creation of simulated 3D particles of different shapes (Winterbottom, Wulff, Cubic, Octahedral and random) using pre-existing scripts developed by Dr. Dupraz and Dr. Bellec [23]. These codes allow the user to construct a cubic FCC crystal of a given element, taking into account the inter-atomic potential, the atomic mass and the lattice parameter. The final particle is finally obtained by "cutting" off atomic planes along given (or random) directions, depending on the chosen shape. We have simulated only Gold nano-particles and this is, in first approximation, equivalent to any generic element as a different lattice parameter would just shift the Bragg peak to a different position in reciprocal space, with no significant alterations of the diffraction pattern. Each particle's configuration is then automatically saved in a LAMMPS-readable file. In a second stage, we perform energy relaxation using LAMMPS software for Molecular Dynamics. This step induces small displacements to the perfect lattice, especially near the surface. In the last stage, the 3D diffraction pattern of a selected Bragg reflection is computed using PyNX scattering package [24]. This software, optimized for GPU acceleration, produces a 3D representation of a selected Bragg peak. It is then possible to adjust the parameters that control the oversampling ratio, the size of the array in which the Bragg peak is centered and the rotation of the Q-space. In our case we simulated 128 pixel-size cubic diffraction patterns and, in order to augment the training dataset, we did it for various oversampling ratios (from 2 to 5) and different rotations for each particle. As we have seen in Chapter (ref to introduction), in the kinematic scattering approximation the energy of the incident X-ray does not alter the diffraction pattern if not as a "zooming" factor. Thus, in our case we don't need to explicitly account for different energies as we already vary the oversampling ratio. Before taking portions of these simulated BCDI patterns, we added Poisson noise randomly scaling the λ parameter for each image.



(a) (a) **(b) (b)**

Figure 4.8: (a) Simulated Au particle with Winterbottom shape (134114 atoms).**(b)** Atomic displacement field of the same particle after LAMMPS energy relaxation. It is evident the typical distribution at the interface with the substrate.

At this point, we proceeded with the extraction of sub-volumes taken at *pseudo*-random locations inside each 3D pattern. The selection in fact was not totally random as we favored the extraction of sub-volumes from the outer regions, far from the center of the peak. There are mainly two reasons for this choice, namely (i) compensate the inherent uneven accuracy score against the position of the gap (see Fig.4.6) by increasing the training data far from the

center and (ii) emulate as much as possible the experimental conditions, in which unavoidable gaps are typically far from the center of the peak. For each sub-volume a 3D mask of the gap was created for different gap sizes (3,6,9,12 pixel-wide). The gap was placed vertically, in the center, along the third dimension, resulting in a "empty slab". Cross-shaped gaps were also included in the training dataset, with a population ratio of 1:5 compared to vertical gaps. They were created by adding a horizontal gap at a random height to an existing vertical gap. The final training dataset consisted of 30'000 $32 \times 32 \times 32$ sub-volumes created as described above.

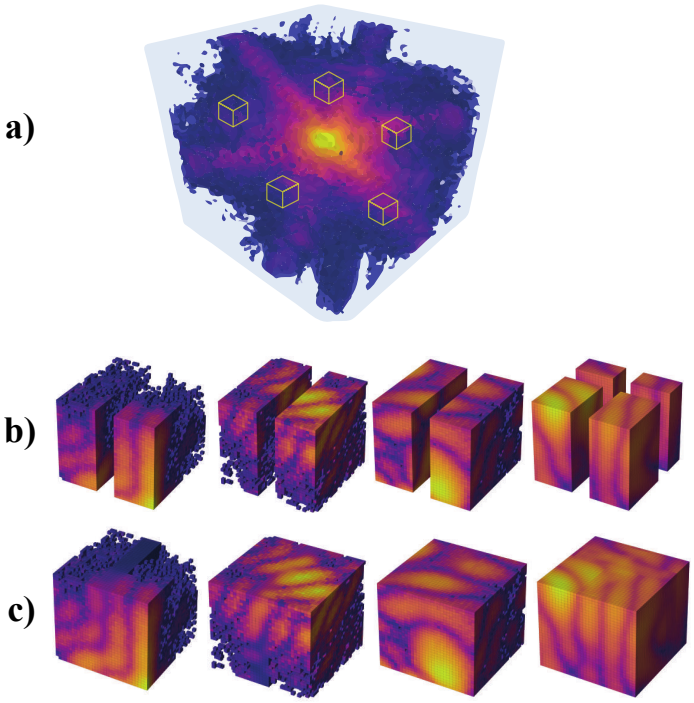


Figure 4.9: Schematic of the sub-volumes extraction. a) The 3D BCDI diffraction pattern and the sub-volumes. **b)** $32 \times 32 \times 32$ pixel-size sub-volumes with 9 pixel-wide vertical and cross-shaped gaps. **c)** Same sub-volumes with the DL inpainted gaps.

4.5 3D model architecture

The DL architecture used for the 3D patching inpainting is illustrated in Fig. 4.10. Given the reduced size of the inputs, the encoder this time is composed of four blocks only, in each of which there are convolutional layers and max pooling layers. The feature map is thus reduced to a $2 \times 2 \times 2 \times 478$ tensor before being passed to the decoder. Notice that, as introduced above in section Sec.4.3, we have employed dilated convolutions in the first two blocks to enhance the extraction long-range correlated features. After four decoder blocks we have put three simple convolutional layers with 24,12 and 6 channels respectively, in order to restore the possible smoothing effect of the decoder. Same as in the 2D model, the last activation function is a sigmoid that ensures the output to be in the range (0,1). The model contains 2'770'000 trainable parameters, significantly less than the 2D models working on full size patterns.
The training was performed loading batches of 32 images at the time over 100 epochs using ADAM optimizer [25]. We initialized the optimizer with a learning rate of 10^{-3} and decreased it progressively with the `ReduceLROn-Plateau` callback feature available in Tensorflow.

In order to exploit at maximum the training dataset we left only 4% and 2.5% of the whole dataset for validation and testing respectively.

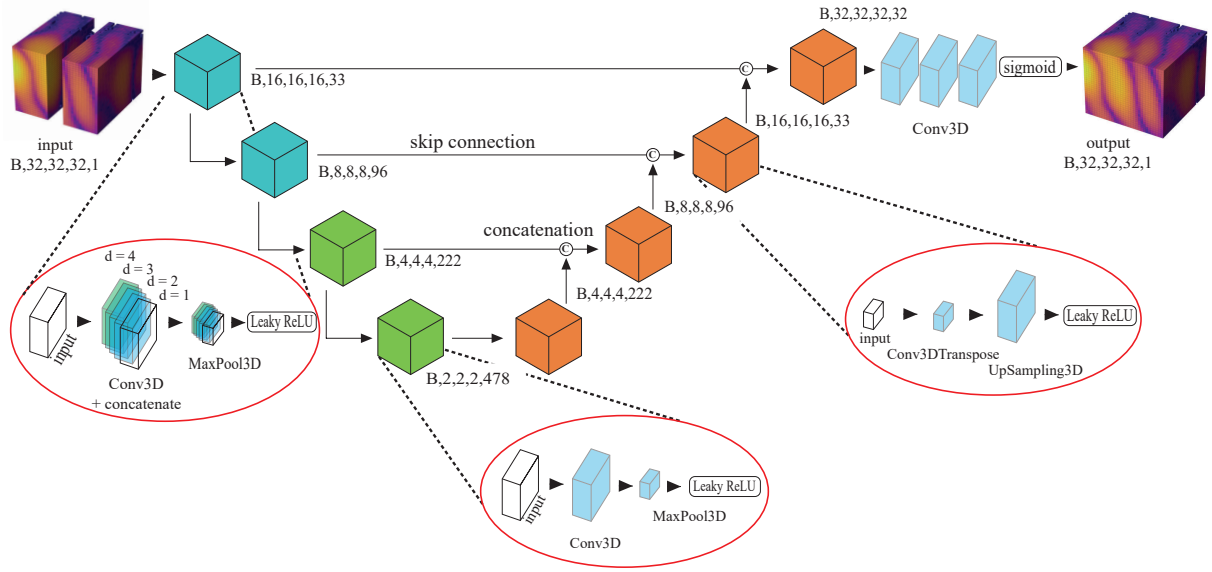


Figure 4.10: Schematic of the 3D model architecture The model uses a modified U-Net structure. In the first two encoder blocks (highlighted by the left red circle), dilated convolutions are applied where the original input is concatenated with its convolutions at various dilation rates (*d = 4,3,2,1*) prior to the MaxPooling operation. The input consists of small gap-affected portions, grouped into batches of 32 (B). These portions (top left) are progressively processed by the encoder until they are reduced to a $2 \times 2 \times 2$ pixel-size feature map. In the decoder, each building block (represented as orange cubes) receives as input the concatenation of the output from the previous block and the matching output from the encoder block of the same size. The final result (top right) is a batch of inpainted versions of the input portions.

4.6 Results in detector space

In this section we will present the results of our DL model on both simulated and experimental diffraction patterns. In the next section we will move instead to the results in real space, therefore focusing more on the reduction of the artifacts in the reconstructed objects.

Once completed the training of the model we have first tested it on portions taken from the test dataset. It is possible to qualitatively observe that the model works equally well for both simulated and experimental data (see Figs. 4.11 - 4.12). From a first visual assessment we can also confirm that low noise regions with larger features are better restored than others as previously stated in Sec. 4.7. Another curious effect that we can observe, is the "smoothening" of features around noisy areas (see first column in Fig. 4.11 and last column in Fig. - 4.12). In fact, the "grainy" aspect of these regions is caused by Poisson noise which cannot be predicted by the DL model as it is uncorrelated. In those regions the DL performs therefore a sort of average that "smoothens" the features and acts like a denoiser. This effect has been already studied in the literature and exploited for denoising applications like the Noise2Void model [26].

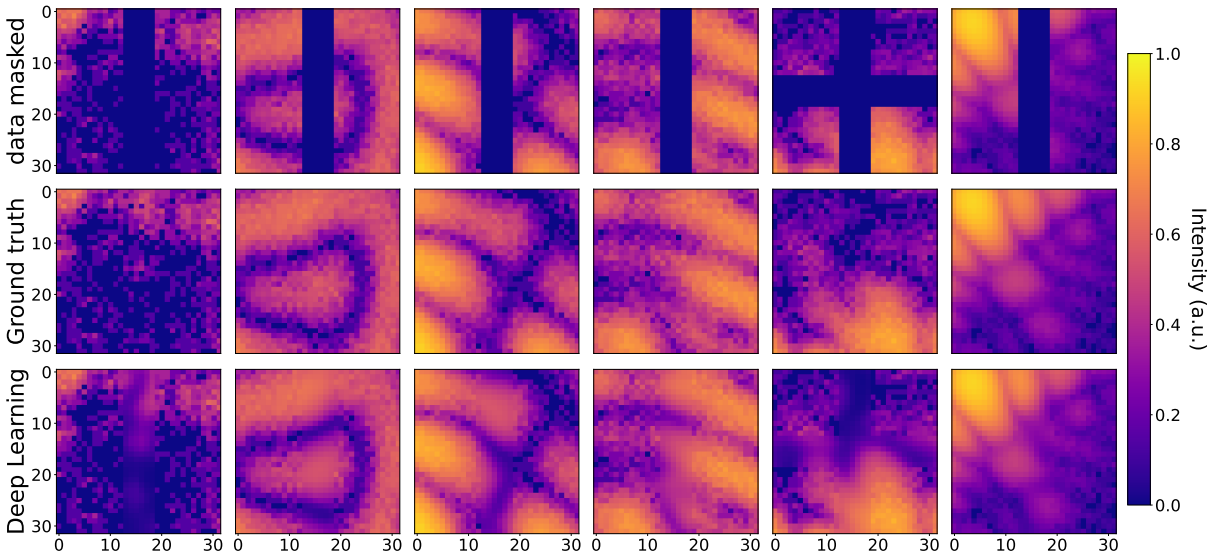


Figure 4.11: Results on portions of test simulated data. Central slices of portions taken from the simulated test dataset. Masked input with 6 pixel-wide gap in the first row, corresponding ground truth and DL inpainted in second and third row respectively.

4.6.1 Full gap inpainting

For the inpainting of a gap inside a full 3D BCDI pattern it is sufficient to apply repeatedly the DL model on sub-volumes cropped such that the gap plane lies vertical in the center of the array perpendicularly to the third dimension. Each sub- volume needs to be preprocessed exactly in the same way described above, i.e. transformed into logarithmic scale and normalized between 0 and 1. Moreover, it is advised to apply a mask on the gap, to match exactly the gap width the model has been trained with. One can then proceed along the gap moving forward one pixel at the time, compute the inpainted gap and average the prediction over the overlapping pixels with the previous predictions. By doing this, potential errors are averaged out and the accuracy of the prediction is maximized. However, for large datasets this can be time-consuming. For example, for a $128 \times 128 \times 128$ pixel-size diffraction pattern with a cross-shaped gap the time needed to compute the full inpainting amounts to 11 minutes (using a NVIDIA Tesla V100-SXM2 GPU with 32GB RAM). However, it is possible to increase the step size to significantly reduce the computing time without affecting excessively the accuracy (see Fig.4.14). We have proven that the amount of time for the full inpainting follows a power law (see Fig. 4.13a) and the accuracy starts dropping significantly for more than 5 pixels skipped at the time (see Fig. 4.13b).

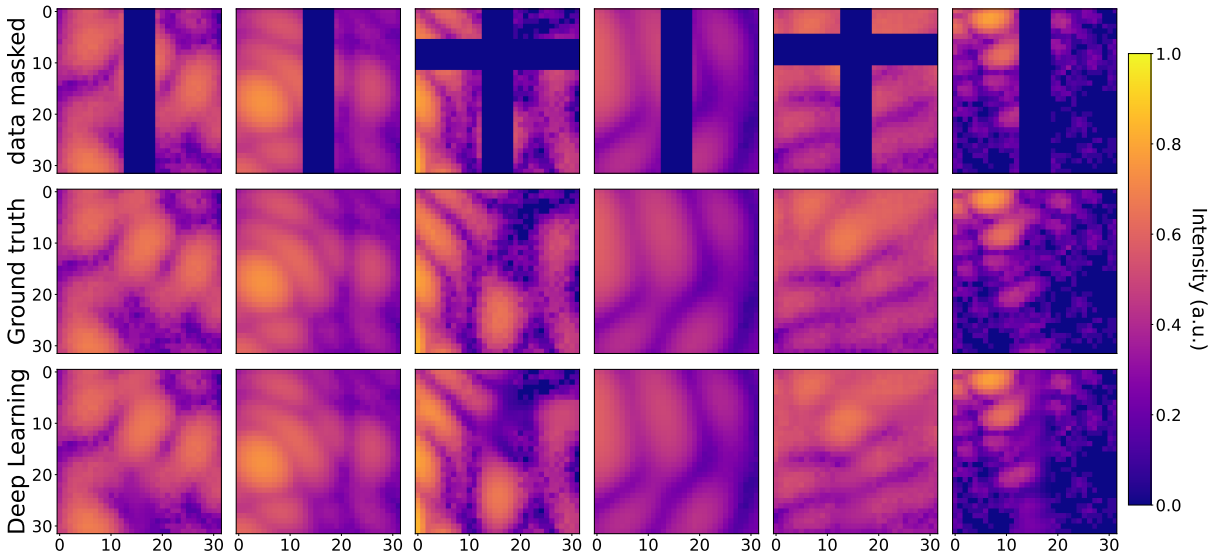
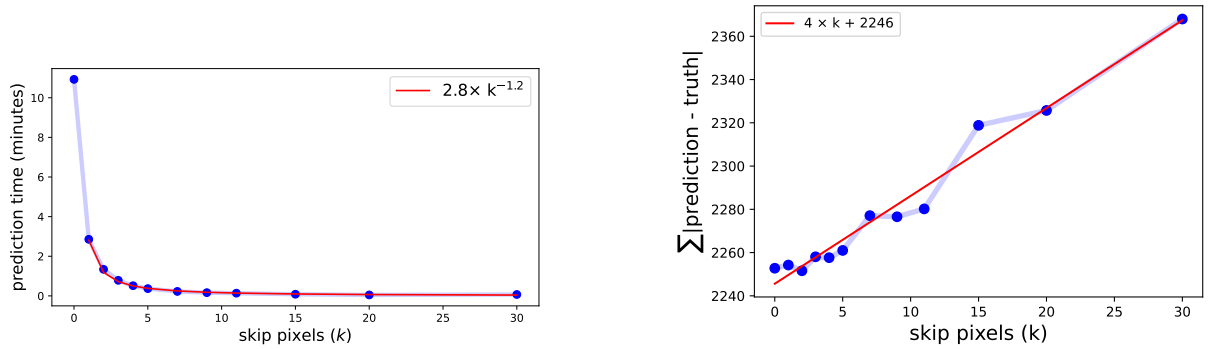


Figure 4.12: Results on portions of test experimental data. Central slices of portions taken from the experimental test dataset. Masked input with 6 pixel-wide gap in the first row, corresponding ground truth and DL inpainted in second and third row respectively.



(a) (a) **(b) (b)**

Figure 4.13: (a) Full inpainting time for a 6 pixel-wide cross-shaped gap on a $128 \times 128 \times 128$ pixel-size diffraction pattern as function of the amount of pixels skipped between patch DL predictions along the gap. **(b)** Sum of the absolute errors as function of the skipped pixels.

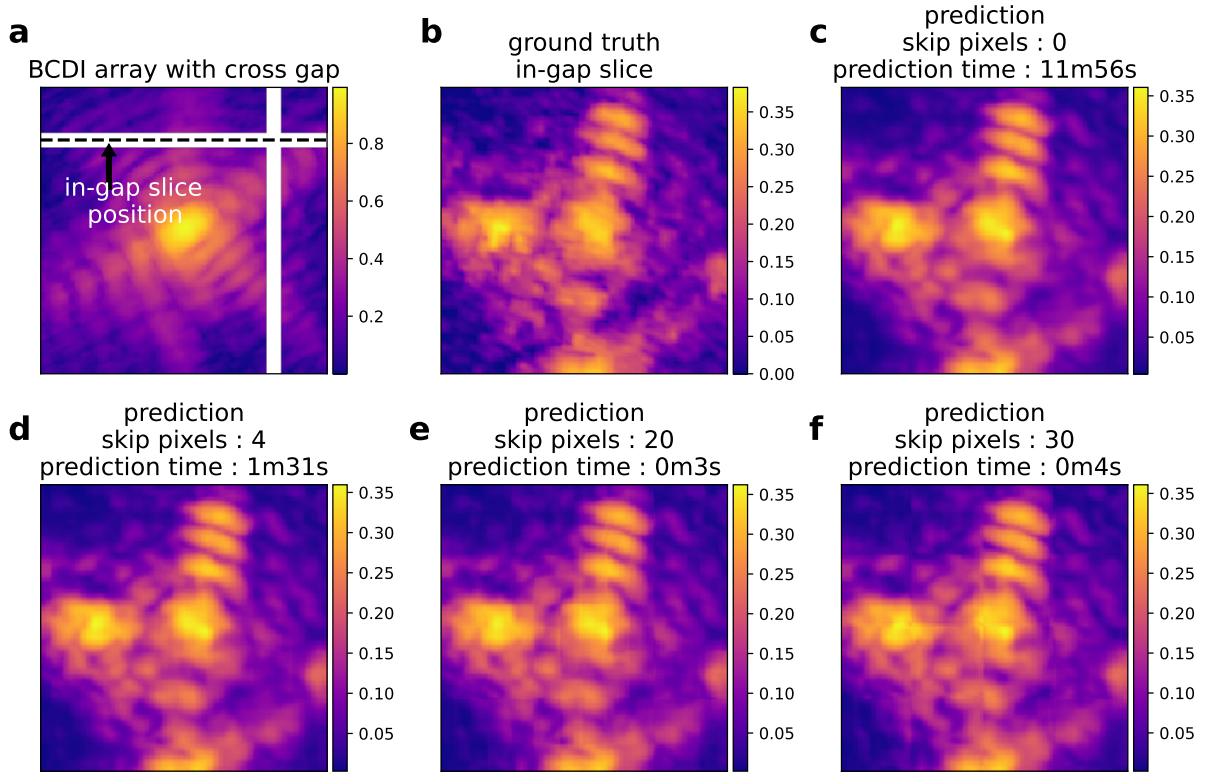
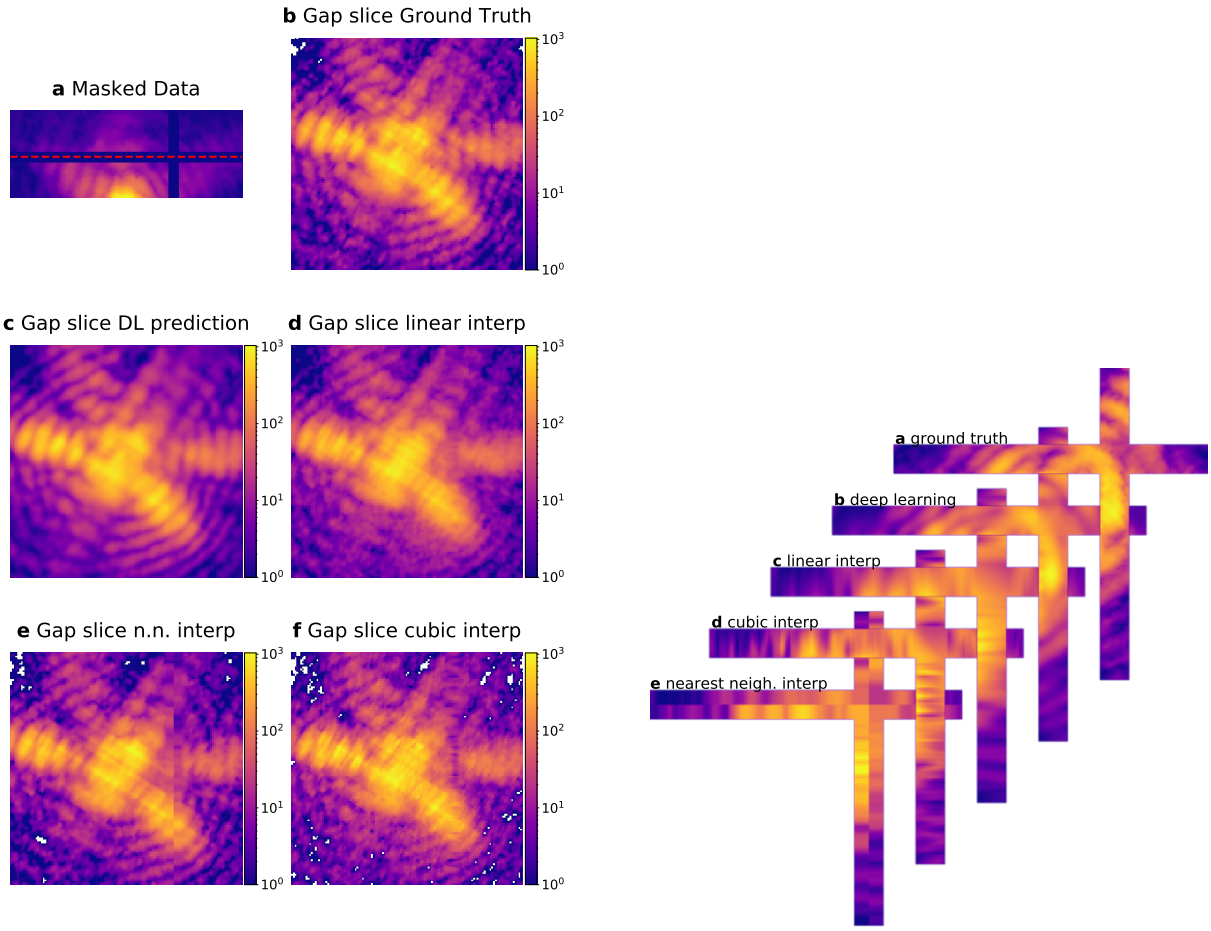


Figure 4.14: Full inpainting of an experimental BCDI pattern for different amounts of skipped pixels. **a** slice of the diffraction pattern perpendicular to the gap plane.**b** Ground truth intensity inside the gap. **c-d-e-f** In-gap prediction with 0, 4, 20 and 30 skipped pixels respectively, with corresponding execution time. Skipping 4 pixels is a good trade off between time and accuracy.

4.7 Performances assessment

In order to assess the performances of our DL model with respect to other inpainting methods, we tested it against conventional interpolation methods. Specifically, we have taken an experimental BCDI pattern with a 6 pixel-wide cross-shaped gap and compared the inpainting results of our DL model with (i) linear interpolation (ii) cubic interpolation (iii) nearest-neighbor interpolation. These techniques allow for a quick estimation of the intensity distribution inside the gaps but fail to recover fine features (see Figs. 4.15). In particular, we can notice in the *in-gap slice* (Fig. 4.15a) that linear interpolation for instance doesn't retrieve correctly the space curvature of the fringes while nearest neighbor and cubic interpolations show artifacts in correspondence of the perpendicular gap. When considering the central slice perpendicular to the gap planes (along the rocking curve dimension) we can notice even more how the DL model outperforms conventional interpolations (Fig. 4.15b).



(a) (a) (b) (b)

Figure 4.15

Similarly to the 2D case above, we have also evaluated the performances of the model against the amount of intensity inside the sub-volume and against the oversampling ratio. We repeated the test for different gap widths, namely 3,6,9,12 pixel-wide, using vertical gaps placed in the center of each portion. For the first performance assessment we have considered a full simulated $128 \times 128 \times 128$ pixel-size BCDI pattern and randomly cropped out of it 1000 portions. We have then applied a vertical gap in the center of each portion for different gap sizes and then computed the prediction with the corresponding DL model. The intensity (in pixel counts) inside each sub- volume was then summed and the obtained values for the 1000 samples were binned into 20 classes for better visualization. The accuracy scores, calculated with the PCC, were then averaged inside each bin class. The results are displayed in Fig. 4.16. As expected from what discussed above for the 2D case, better accuracy scores are obtained for portions containing larger amount of signals, where noise levels are lower and the features of the diffraction pattern are more visible. Moreover, the plot logically shows that smaller gaps are generally better recovered, but it is worth noticing that the accuracy spread across different gap sizes widens for noisy portions and narrows down as the amount of signal increases. These trends suggest that DL models are overall robust to different gap sizes especially for high intensity regions, which are eventually the most important ones as they contribute the most during to the reconstruction.

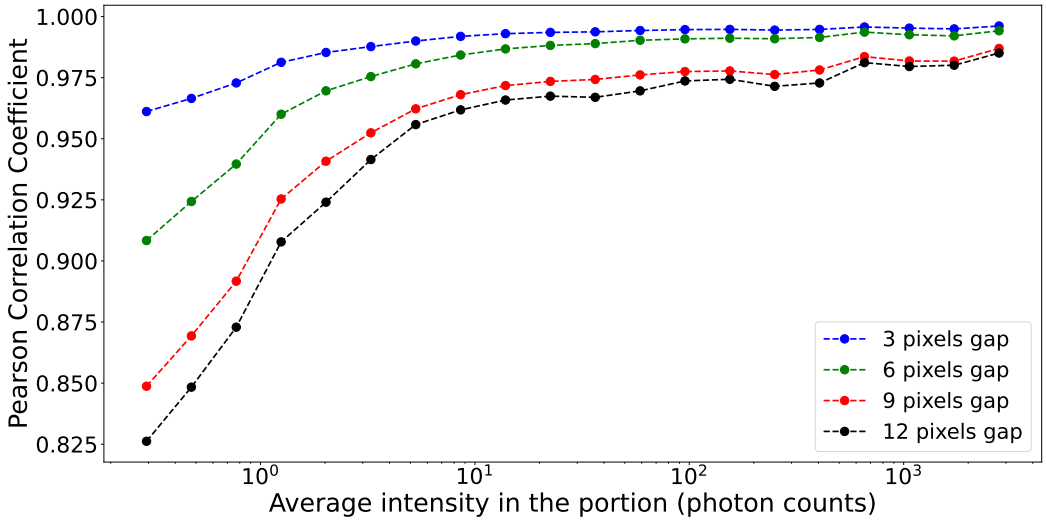


Figure 4.16: Accuracy scores (PCC) of the DL patching model

The last test concerns the study of the accuracy for different oversampling ratios. As anticipated above for the 2D case, to carry out properly this evaluation, one should consider the same diffraction pattern extending to the same equivalent Q-space value for each oversampling ratio. This in practice is done reducing increasing the dq per pixel as decreasing the oversampling ratio, resulting in a smaller size of the overall BCDI pattern. In our particular case we have simulated the same BCDI pattern for oversampling ratios spanning from 2 to 7. For each oversampling ratio, a vertical gap mask was applied to the whole BCDI array and the DL prediction was calculated with no-skip pixel (see Sec. 4.6.1). The gap was then shifted laterally and this procedure was repeated until the whole BCDI array was predicted using our model, thus leading to a full BCDI predicted image. The PCC was then calculated using the whole BCDI array for different oversampling ratios and model gap sizes. The results are displayed in Fig. 4.17. As expected, the predictions are more accurate for large oversampling ratios and small gap sizes (i.e., large oscillation periods relative to the gap width).

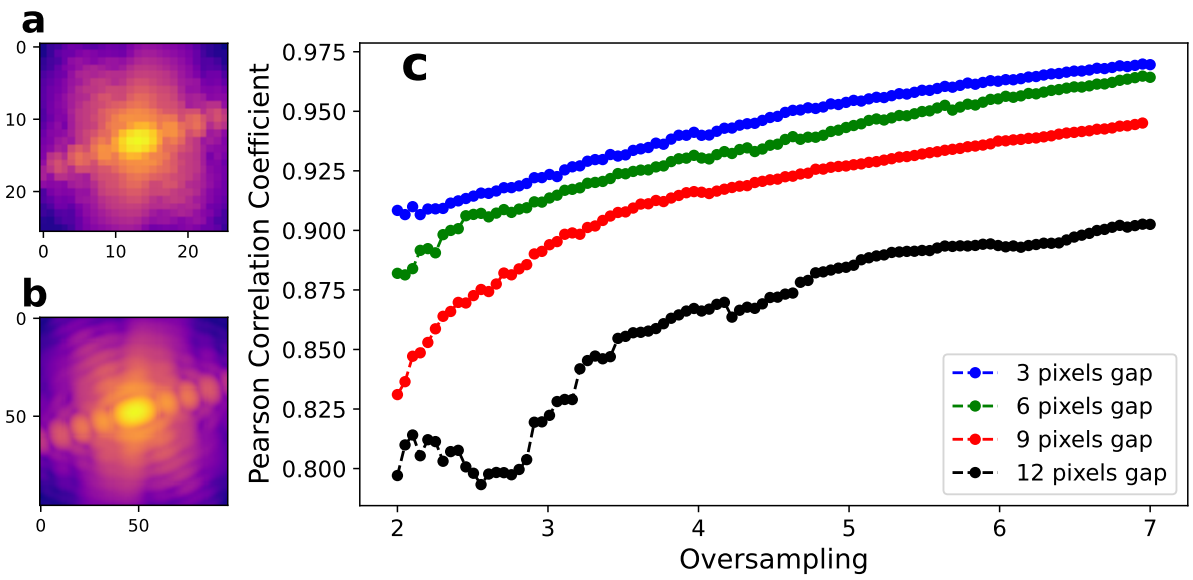


Figure 4.17: Accuracy scores (PCC) of the DL patching model against the oversampling ratio.

4.8 Results in real space

In this section we will discuss the effects of DL inpainting on the reconstructed objects for both simulated and experimental data. In particular, we will assess, both qualitatively and quantitatively, the gap induced artifacts in the modulus, phase and strain fields of the reconstructions and their reduction thanks to the DL inpainting. To carry out these analyses we have taken an experimental BCDI dataset acquired at the ID01 beamline of the ESRF and already exploited by Carnis *et al.* in 2019 for similar studies on gap-induced artifacts [3]. The dataset corresponds to the BCDI pattern around the (**111**) peak of a Pt tetrahextrahedral (THH) particle (400 nm in size). Similarly to what the authors did, we have kept the modulus of the reconstructed object and set the real space phase to zero, making it our reference ground-truth object **O**. This measure helps us to highlighting the gap induced artifacts on the phase and strain fields as we have a zero-phase reference to compare the results with. We have then calculated the corresponding diffraction pattern with the fast Fourier transform (FFT) obtaining a complex diffracted amplitude $\mathbf{A} = FFT(\mathbf{O})$. A cross-shaped gap was subsequently applied to **A** and the corresponding object $\mathbf{O}_{gap}$ was calculated with the inverse FFT. From the same gapped **A**, the intensity $\mathbf{I} = |\mathbf{A}|^2$ was also "inpainted" using our DL model and corresponding object $\mathbf{O}_{DL}$ was calculated with the inverse FFT as well, using the ground truth reciprocal space phase. We have repeated the procedure for four different gap sizes (3, 6, 9, 12 px-wide) matching exactly the cases mentioned in the work of Carnis and coauthors. Figure Fig.4.18 shows the projection along the rocking curve axis (XY slice in this case) of the ground truth diffracted intensity, the gapped and the DL inpainted ones for the 9 pixel-wide gap case.

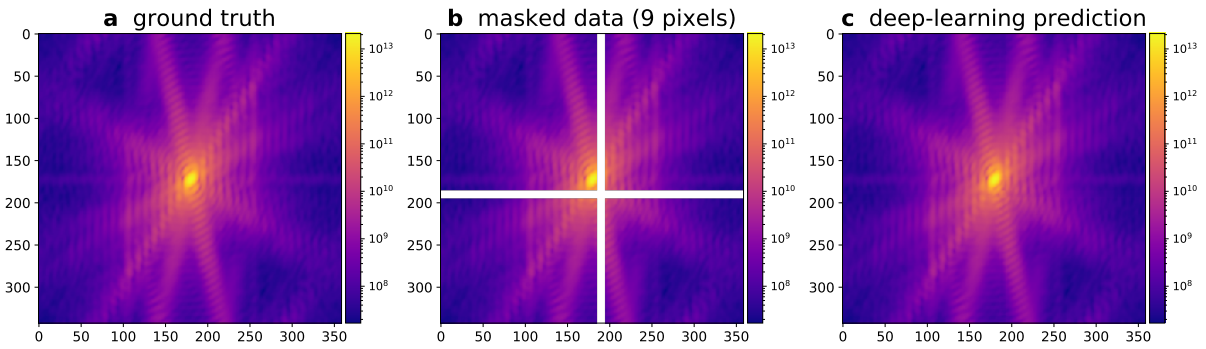


Figure 4.18: Projections along the rocking curve axis of the studied diffraction pattern in log scale. a Ground truth pattern obtained from the $|FFT(\mathbf{O})|^2$. **b** Pattern with a 9 pixel-wide cross shaped gap. The position close to the center of the peak is experimentally unlikely but here it allows us to enhance the artifacts in the reconstructions. **c** Corresponding DL inpainted BCDI pattern. It is visible the presence of aliasing due to the FFT calculation rather than the more correct kinematic sum. This effect is however not relevant for the scope of these analyses.

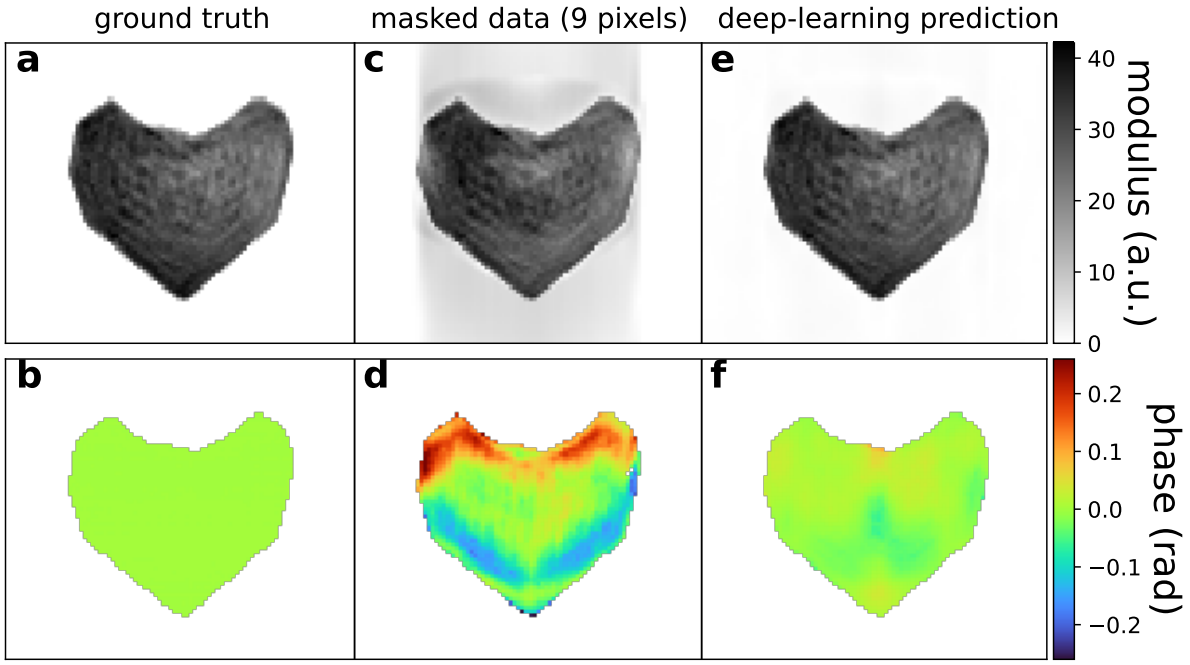


Figure 4.19: Reconstructed objects.a-b Ground truth modulus and phase. **c-d** Modulus and phase of $\mathbf{O}_{gap}$. **e-f** Modulus and phase of $\mathbf{O}_{DL}$

Figure Fig.4.19 illustrate instead the central (YZ) slice of the reconstructed objects for the three cases. It is evident that while $\mathbf{O}_{gap}$ shows significant abnormalities in both modulus and phase, $\mathbf{O}_{DL}$ is much closer to the ground truth. In particular one can notice that the gap plane, horizontal in the YZ plane, induces artifacts along its perpendicular direction. The result is indeed a stripe of non-zero modulus outside the support and, most importantly, an overall phase variation of 0.4 radians along the vertical direction. This phase variation results in an error of ± 7 pm in the lattice displacement field for the 111 Pt reflection, with more intensity around the surface. These artefacts are particularly problematic in the cases of (electro-)catalytic

experiments [27] or in situ gas experiments [28]; Kim et al., 2018; Abuin et al., 2019; Kawaguchi et al., 2019; Dupraz et al., 2022), where the particle's surface is primarily involved in the reaction and one could follow the process by monitoring the evolution of the strain in that region. As one could expect the artifacts become more severe as the gap size increases. Fig.4.20 depicts the phases of $\mathbf{O}_{gap}$ and $\mathbf{O}_{DL}$ for the different gap sizes considered in this study while Fig.4.21 the strain distribution in the XY plane.

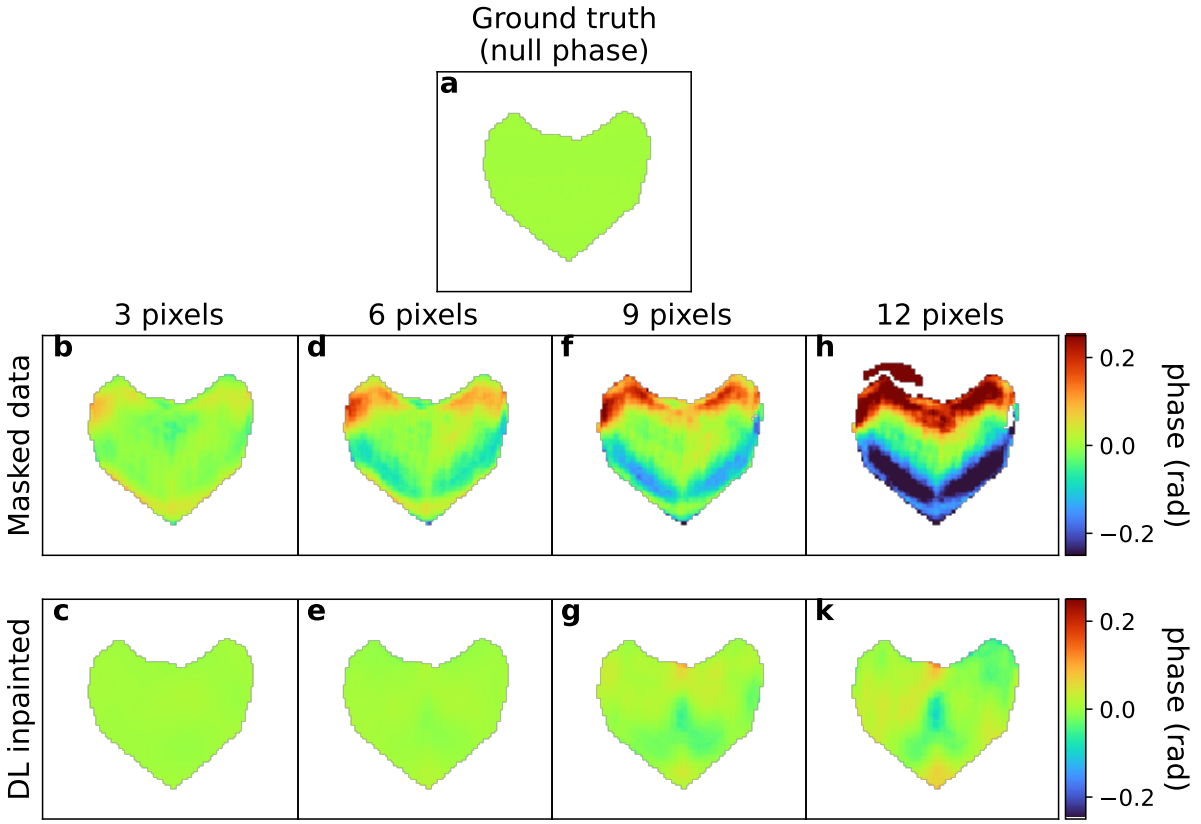


Figure 4.20: Artifacts on the phase of $\mathbf{O}_{gap}$ for different gap sizes, and phases of the corresponding $\mathbf{O}_{DL}$

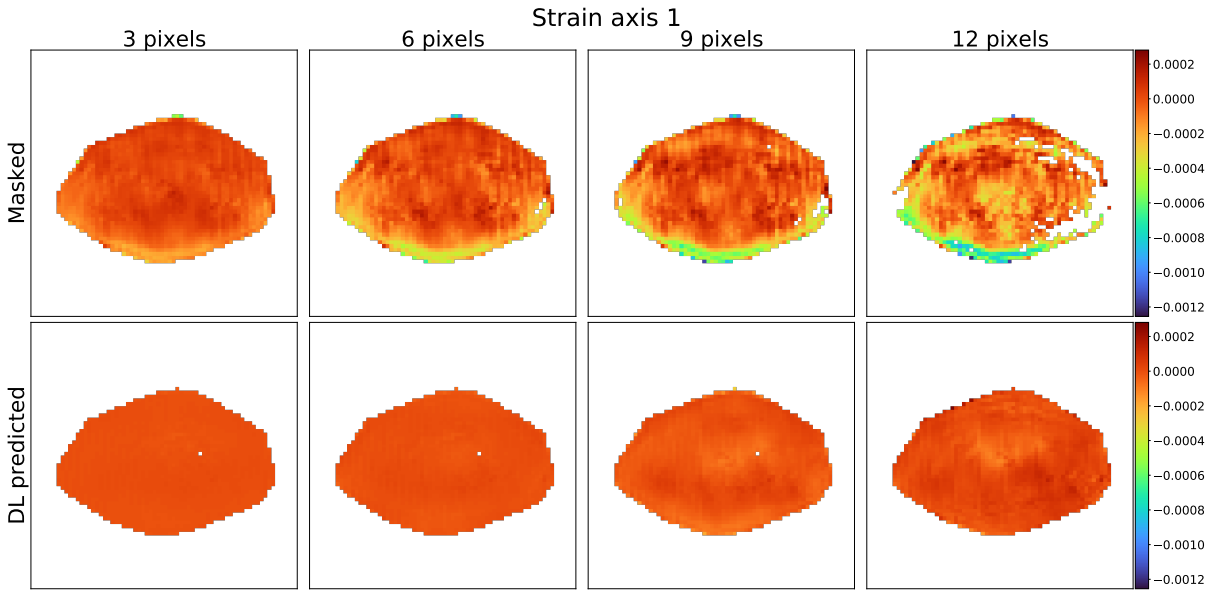


Figure 4.21: Strain distribution in the central XZ slice of $\mathbf{O}_{gap}$ for different gap sizes and corresponding results for $\mathbf{O}_{DL}$

The deviation from the ground truth zero value of the retrieved strain for both $\mathbf{O}_{gap}$ and $\mathbf{O}_{DL}$ can be measured with the root mean squared error (RMSE) across all the different gap sizes. This calculation was already proposed in the aforementioned work of Carnis and coauthors in which the results were plotted in Fig.4. We have reproduced a similar figure adding the results of our DL model (Fig.4.22). The trend of the strain RMSE resembles indeed the curve showed in [3], increasing significantly with the gap size while the DL equivalent curve lies below, dampening the error of approximately a factor 5. Moreover, the strain artifacts induced by the gaps shift the average strain from the zero value as shown in Fig.4.23 whereas our DL model maintains the average strain around zero.

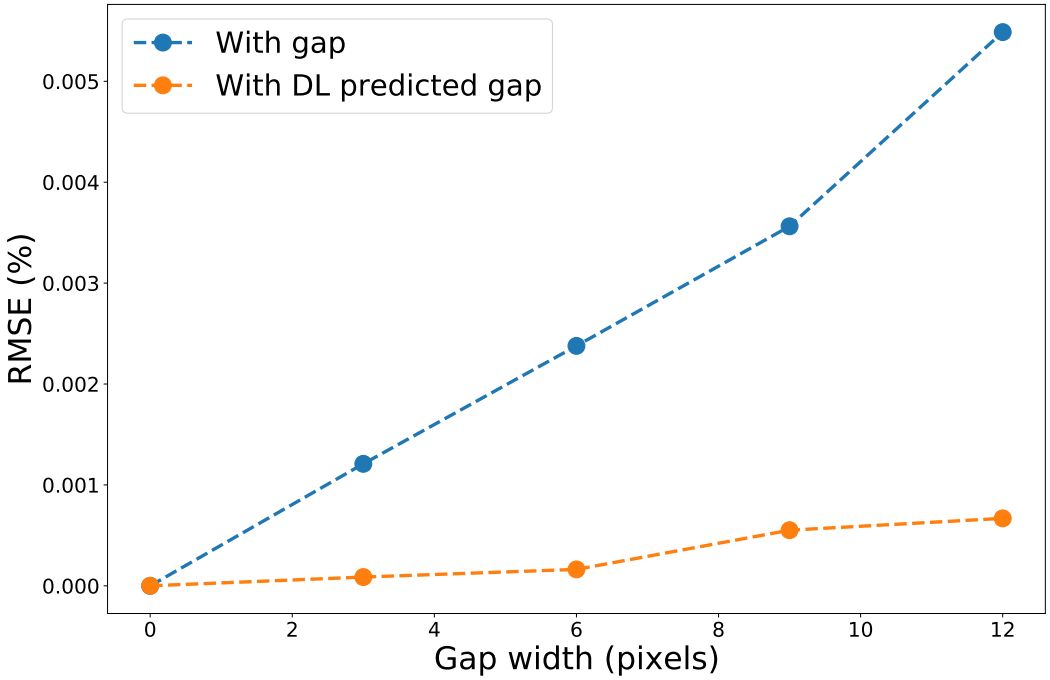


Figure 4.22: RMSE of the strain field versus gap size. For both cases of masked and DL inpainted diffraction patterns. For all gap sizes, the DL inpainted diffraction patterns yield a smaller error.

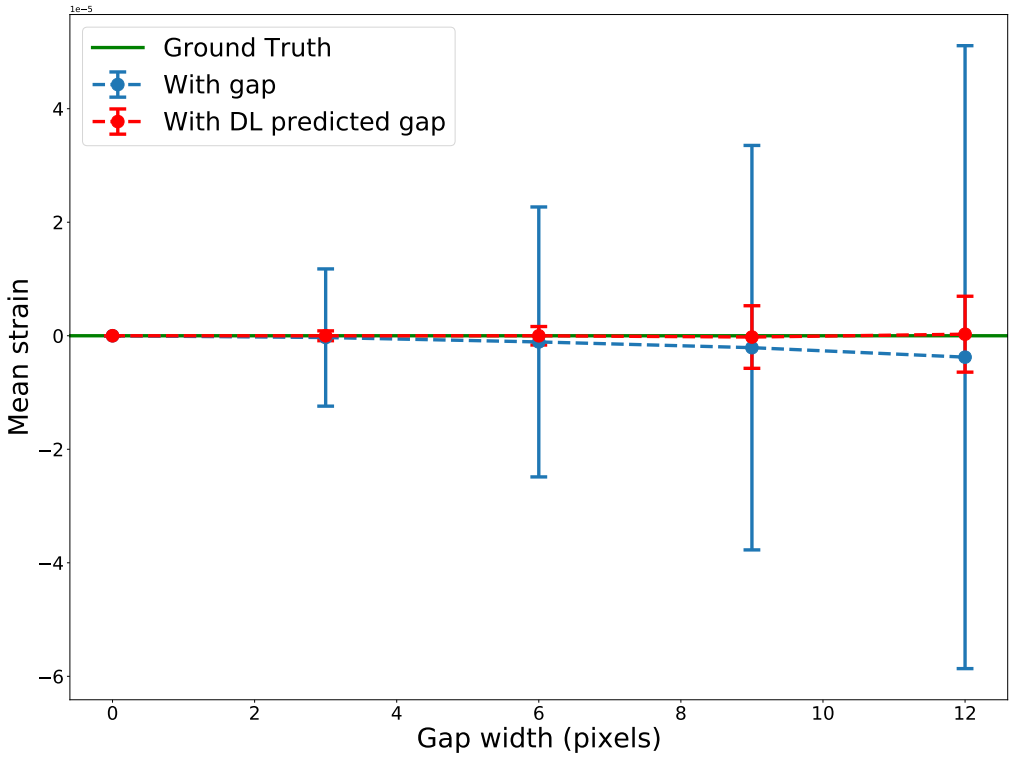


Figure 4.23: RMSE of the strain field versus gap size.

4.8.1 DL inpainting for high resolution BCDI

As anticipated in the introduction to the chapter, many of the cases in which a BCDI dataset is inevitably affected by a detector gap is the so-called "high-resolution" BCDI. The acquired data here extend for large Q ranges in all directions resulting in ROIs of several pixels. This can imply that parts of the diffracted signal crosses a region on the detector with a vertical or horizontal gap, thus needing for gap-inpainting. It is then convenient to use a patching approach as treating the full volume would be computationally too expensive. Moreover, any binning or interpolation to smaller sizes will induce information loss, as well not advised.
An example of high-resolution BCDI dataset of this type is the one we have used so far from the work of Carnis and coauthors. The original dataset is indeed a large ($256 \times 300 \times 300$ pixels) array that contains a cross-shaped, 6 pixel-wide gap. Here we show how the artifacts can change depending on the type of masking of the gaps is chosen during the phasing and how our DL model can outperform these methods. A common approach, when using PyNX software, is to mask the gap such that those pixels don't contribute during the phasing and are left free to evolve (**a**). Moreover, one could mask only near the intensity streaks affected by the gap (**b**) or simply leave the gap with zeros and remove the contribution of the gap voxels during the whole phasing (**c**). These strategies have been used during the phasing of the cited Pt diffraction pattern and the results in object space compared with what obtained from the DL inpainted pattern. The results, illustrated in Figs. 4.24 - 4.25, show that the amount

of oscillatory artifacts progressively decreases as we go from method **a** to **c**, proving the DL inpainting to be the optimal method among them.

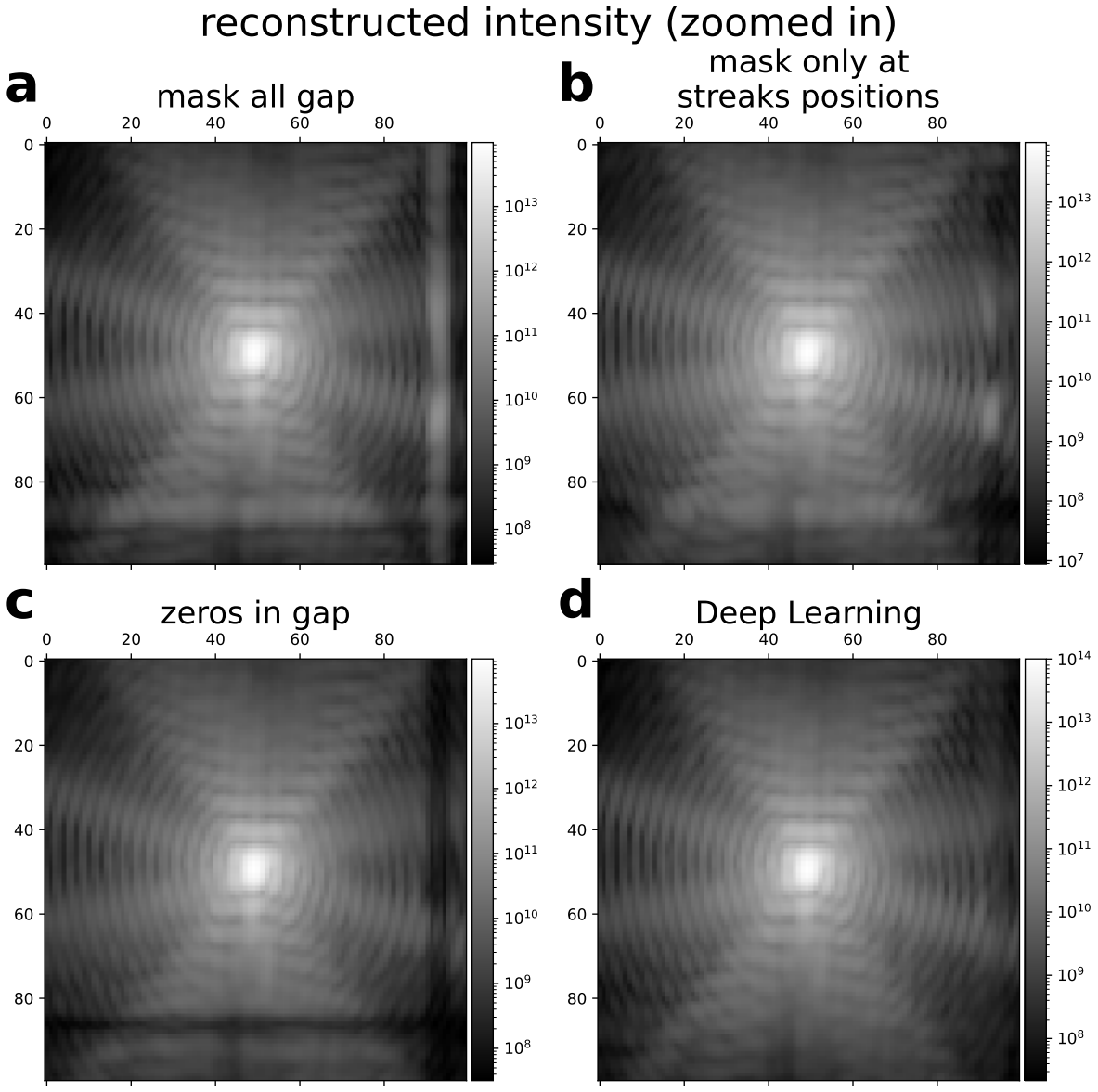
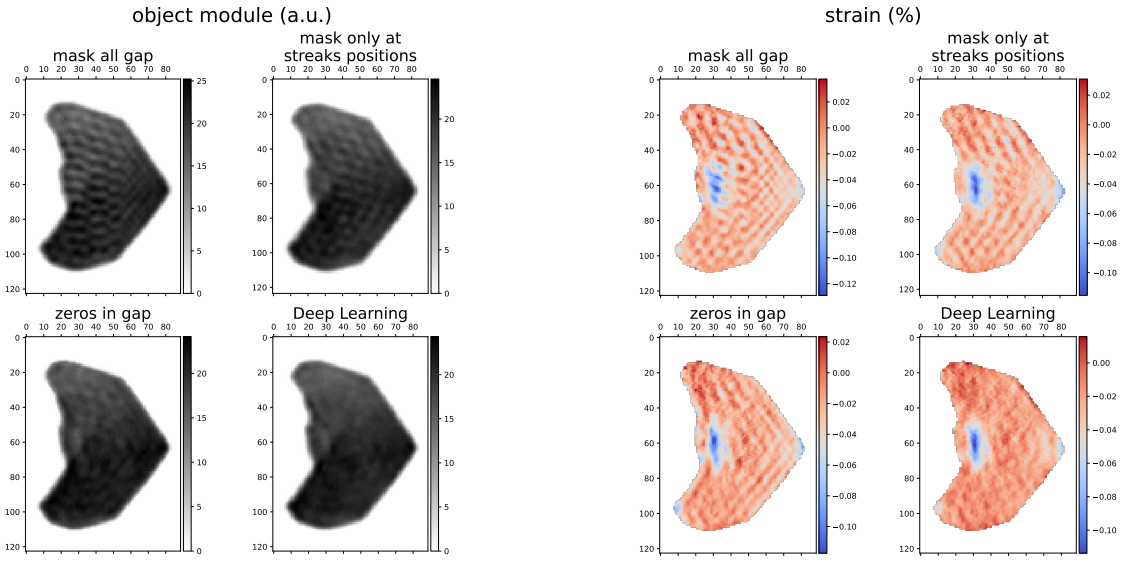


Figure 4.24: Zoom on the projection along the rocking curve axis of the Pt THH BCDI pattern calculated from the reconstructed object obtained with PyNX software using **a** a mask on the gaps, **b** a mask on the streaks only, leaving zeros inside the gaps **c** and inpainting the gaps with our DL model.



(a) (a) **(b) (b)**

Figure 4.25: Modulus and strain of the object's reconstructions for each case mentioned above. The oscillatory artifacts are smallest for the object obtained after DL inpainting.

4.9 Fine-tuning

For those cases in which the DL model does not yield satisfactory results when inpainting a new experimental BCDI pattern we have thought about a fine-tuning of the model to improve the accuracy of the prediction. This fine-tuning is enabled by the patching approach as it consists of a secondary short training of the general model on a small dataset made of portions extracted from the new BCDI pattern to be inpainted. In particular, after loading the gap affected BCDI pattern we have randomly cropped 6400 portions out of it, paying attention not to include the gap region. We have then trained the model for the corresponding gap width for 5 epochs. Biasing the model to the fit the features of that specific diffraction pattern (oversampling ratio, particle shape, noise level, fringes shape) we could obtain better result on the real gap. An example is shown in Fig.4.9. There, the general DL model was not able to predict the fringes with the correct periodicity inside the gap. After the fine-tuning instead, the model properly recovers the fringes improving the accuracy. This fine-tuning technique wants to be a further example of the advantages of using a patching approach and its usage depends on the user judgement on the quality of the general model inpainting.

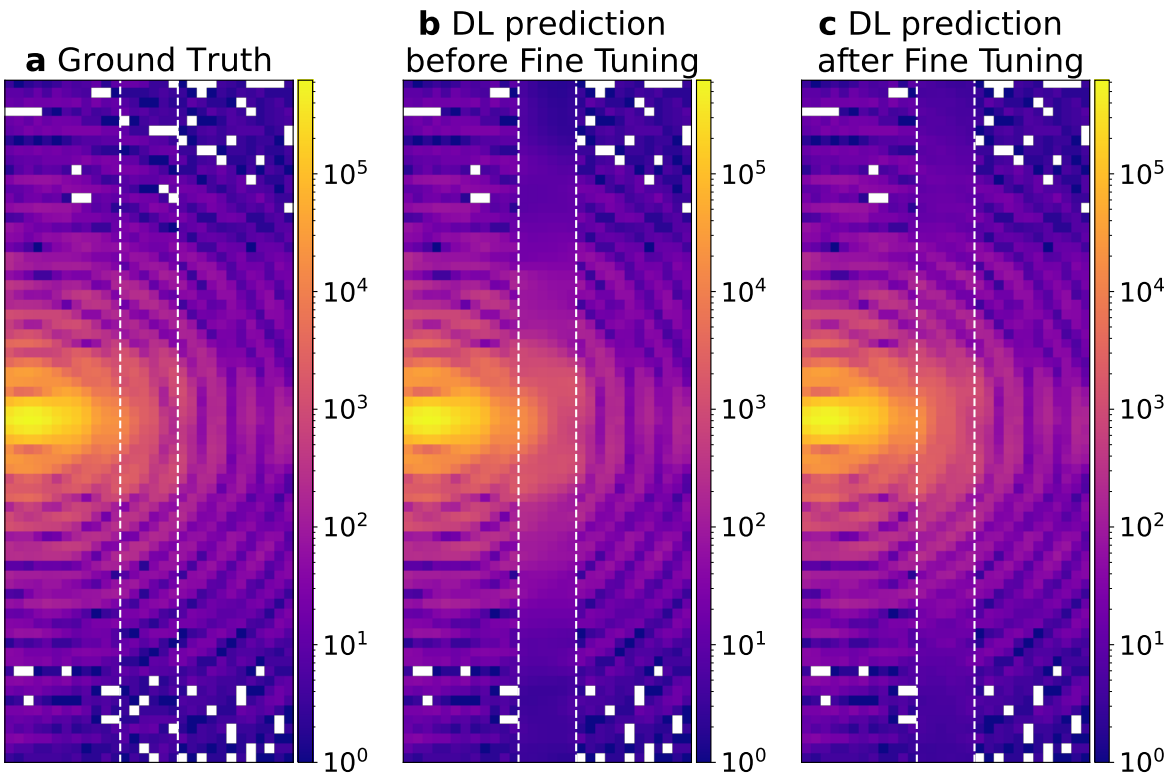


Figure 4.26: Example of improved accuracy after fine-tuning of the DL model. The fringes are better recovered after 5 epochs of fine-tuning.

Chapter 5

Deep Learning for Phase Retrieval

We enter now the core topic of the thesis. Most of the efforts during this PhD have been dedicated to the study of the Phase Problem for Bragg Coherent Diffraction Imaging using Deep Learning based approaches. Here we will discuss the main steps of this journey, starting off from the analysis of the most relevant works in literature and concluding with our final version of a DL model for highly strained particles. The latter has become the subject of an article, currently in preparation, entitled "*Phase Retrieval of Highly Strained Bragg Coherent Diffraction Patterns with Supervised Convolutional Neural Network*". The process that led to the final version of the model will be unraveled, and particular attention will be given to elucidating the key steps and the critical issues encountered along the way.

5.1 State of the art

In this paragraph we will focus on the state of the art for what concerns the Phase Retrieval of BCDI diffraction patterns with deep-learning, tensor-computation and automatic differentiation methods. Conventional phase retrieval iterative algorithms are discussed in the introduction chapter as well as other approaches.
Given the relatively new development of neural networks and more specifically even more recent for BCDI phase retrieval we will try to give a chronological broad overview over many of the main works in the literature pointing out strengths and weaknesses. The first work pioneering the field is "Real-time coherent diffraction inversion using deep generative networks" published by Cherukara *et. al* in 2018 [29]. The paper presents two CNNs for the phase retrieval of small (32×32 pixels) 2D simulated BCDI patterns, one predicting the support and the other the phase. A U-Net like architecture with encoder-decoder was implemented, and the model was trained for just 10 epochs in a supervised fashion with a cross-entropy loss function (see Appendix). The results show an excellent agreement between prediction and ground truth also in presence of relatively strong phases. The potential of this new approach for phase retrieval becomes immediately clear when considering the drastic reduction of computational time and resources needed for the model inference. Once the model is trained the reconstruction can be obtained within few milliseconds on a desktop machine. In 2020 Scheinker and Pokharel proposed another approach [30] that employs a CNN model for 3D diffraction patterns. The fundamental difference is that the object's support is defined by its surface only, as it is assumed to be *compact* and *homogeneous* inside. Moreover, the surface is parametrized by spherical

harmonics and the DL model is trained to predict 28 of the first even coefficients of the spherical harmonics. The model architecture is therefore essentially different since, while the encoder is just transposed to a 3D one, the decoder is replaced by a flattening and dense layer with 28 different classes as output. The model shows good performances on both simulated and experimental data, marking the first DL-based approach capable of real 3D BCDI phase retrieval. In the same year, Wu and coauthors [31] opted for an architecture made of a single encoder and two identical decoders for the prediction of amplitude and phase of single crystals from the central slice of the BCDI pattern. They conducted the study on simulated data and tested it on one experimental case as well. What is evident from their work is the winning combination of DL prediction and iterative refinement. The speed and generalization capabilities of the CNN allows for fast and good estimations of the object's support and phase. In addition, the precise and well established iterative methods can bring this initial guess to a more polished and accurate solution in fewer cycles than without DL prediction. This successful combined approach has been later adopted in other works, ours included. In 2021 two important works were published. First, Chan *et al.* in [32] extended the encoder/2-decoders architecture to the 3D case. In their work they first created a "physics-informed" training set obtained building particles by clipping planes from a cubic FCC structure of atomic positions, relaxing them with LAMMPS software for molecular dynamics and computing the BCDI pattern around the (111) Bragg peak. The procedure is very similar to the one adopted by Lim *et al.* in [23] and described above in Section 4.4.1. Training the CNN on a restricted set of such created BCDI patterns biases the predictions towards physically meaningful particles. Although the authors managed to successfully test their model on an experimental BCDI pattern, the small size ($32 \times 32 \times 32$ pixels) of the images accepted by the CNN was not yet enough for proper experimental use. It's with the work of Wu *et al.* [33] published in the same year, which lifts the size to 64 pixel-sided cubes, that the model can be tested on several experimental cases. Their CNN model maintains the encoder/2-decoders architecture for a simultaneous prediction of the object's amplitude and phase and explores for the first time the unsupervised training for refinement as well. The authors claim that this approach is able to achieve better reconstruction quality with respect to current state-of-the-art iterative algorithms in use. The year after, Yao and coauthors published AutoPhaseNN [34], again an encoder/2-decoders architecture that completely trained in an unsupervised manner. This approach is beneficial as it doesn't require datasets labeled with a ground truth, which means that experimental data can be directly used in the training set. Another advantage is that it overcomes the limitation of simulating an enough diverse population of samples, capable of constituting a comprehensive distribution of real cases. AutoPhaseNN is trained to predict an object the diffracted intensity of which matches the observed one according to a normalized Mean Absolute Error metric. The model is shown to work on simulated data as well as on experimental data and once more the winning method lies in the combination of DL prediction and iterative refinement. AutoPhaseNN has marked a milestone in the BCDI data analysis, attaining 10X to 100X phase retrieval speed up with reduced efforts for the model training. Although of different nature, it is worth mentioning the work of Zhuang and coauthors [35] in which two CNNs are used in the "deep image prior" (DIP) framework. DIP [36] typically implies the use of a CNN for an enhanced representation of an image, often to solve inverse problems like super-resolution, denoising and inpainting. However, it differs from classical deep learning as there is no training dataset but a fit of the target problem exploiting the parameters of the convolutional layers and the efficient gradient descent provided by the automatic differentiation. In their work, Zhuang *et al.* formulated the more general far-field phase retrieval problem as an optimization problem and considered the phase symmetries that affect this class of solutions (see Introduction chapter). Their work

employs two DIPs, one for the modulus and one for the phase, and successfully manages to reconstruct simulated objects even in presence of strong phases. A last interesting contribution is the work of Yu and *et al.* [37]. In this paper the authors propose a DL model that computes complex convolutions, handling real and imaginary parts of the complex tensor in a single passage through the convolutional block. Complex convolutional layers are claimed to be better at preserving the physical connection between real and imaginary parts inside the complex object. The model is used for the phase retrieval of experimental 2D diffraction patterns, for which an unsupervised refinement is used as well.
Before proceeding with our study, we summarize the main features of the mentioned works into a table, and we will comment about the different choices.

5.2 Reciprocal space phasing

5.3 Phase symmetries breaking

5.4 Model design

5.5 Results on 2D case

5.6 Results on 3D case

5.7 Refinement with iterative algorithms

5.8 Experimental results

Chapter 6

Automatic Differentiation for BCDI Phase Retrieval

Chapter 7

Conclusions

Appendix A

Additional Data and Methods

Appendix B

Appendix

Bibliography

1. Ponchut, C. *et al. MAXIPIX, a fast readout photon-counting X-ray area detector for synchrotron applications* in *Journal of Instrumentation* **6.** Issue: 1 ISSN: 17480221 (Jan. 2011).
2. Johnson, I. *et al.* Eiger: a single-photon counting x-ray detector. *Journal of Instrumentation* **9,** C05032. https://dx.doi.org/10.1088/1748-0221/9/05/C05032 (May 2014).
3. Carnis, J. *et al.* Towards a quantitative determination of strain in Bragg Coherent X-ray Diffraction Imaging: artefacts and sign convention in reconstructions. *Scientific Reports* **9.** Publisher: Nature Research. ISSN: 20452322 (Dec. 1, 2019).
4. Elharrouss, O., Almaadeed, N., Al-Maadeed, S. & Akbari, Y. Image Inpainting: A Review. *Neural Processing Letters* **51,** 2007–2028. ISSN: 1573-773X. http://dx.doi.org/10.1007/s11063-019-10163-0 (Dec. 2019).
5. Jam, J. *et al.* A comprehensive review of past and present image inpainting methods. *Computer Vision and Image Understanding* **203,** 103147. ISSN: 1077-3142. https://www.sciencedirect.com/science/article/pii/S1077314220301661 (2021).
6. Efros, A. & Leung, T. *Texture synthesis by non-parametric sampling* in *Proceedings of the Seventh IEEE International Conference on Computer Vision* **2** (1999), 1033–1038 vol.2.
7. Bertalmio, M., Bertozzi, A. & Sapiro, G. *Navier-stokes, fluid dynamics, and image and video inpainting* in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001* **1** (2001), I–I.
8. Mairal, J., Elad, M. & Sapiro, G. Sparse Representation for Color Image Restoration. *IEEE Transactions on Image Processing* **17,** 53–69 (2008).
9. Allene, C. & Paragios, N. *Image Renaissance Using Discrete Optimization* in *18th International Conference on Pattern Recognition (ICPR'06)* **3** (2006), 631–634.
10. Goodfellow, I. J. *et al. Generative Adversarial Networks* 2014. arXiv: 1406.2661 [stat.ML]. https://arxiv.org/abs/1406.2661.
11. Jiang, Y., Xu, J., Yang, B., Xu, J. & Zhu, J. Image Inpainting Based on Generative Adversarial Networks. *IEEE Access* **8,** 22884–22892 (2020).
12. Xu, Z. *et al.* A Review of Image Inpainting Methods Based on Deep Learning. *Applied Sciences* **13.** ISSN: 2076-3417. https://www.mdpi.com/2076-3417/13/20/11189 (2023).
13. Li, C.-T. *10 Papers You Must Read for Deep Image Inpainting* Accessed: 2025-03-03. 2020. https://towardsdatascience.com/10-papers-you-must-read-for-deep-image-inpainting-2e41c589ced0/.
14. Sogancioglu, E., Hu, S., Belli, D. & van Ginneken, B. *Chest X-ray Inpainting with Deep Generative Models* 2018. arXiv: 1809.01471 [cs.GR]. https://arxiv.org/abs/1809.01471.
15. MA, B. *et al.* Deep learning-based automatic inpainting for material microscopic images. *Journal of Microscopy* **281,** 177–189. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/jmi.12960. https://onlinelibrary.wiley.com/doi/abs/10.1111/jmi.12960 (2021).

16. Chavez, T., Roberts, E. J., Zwart, P. H. & Hexemer, A. A comparison of deep-learning-based inpainting techniques for experimental X-ray scattering. *Journal of Applied Crystallography* **55.** Publisher: International Union of Crystallography (IUCr), 1277–1288. ISSN: 1600-5767. https://scripts.iucr.org/cgi-bin/paper?S1600576722007105 (Oct. 1, 2022).

17. Bellisario, A., Maia, F. R. & Ekeberg, T. Noise reduction and mask removal neural network for X-ray single-particle imaging. *Journal of Applied Crystallography* **55.** Publisher: International Union of Crystallography, 122–132. ISSN: 16005767 (Feb. 1, 2022).

18. Lecun, Y., Bottou, L., Orr, G. & Müller, K.-R. Efficient BackProp (Aug. 2000).

19. Li, H., Xu, Z., Taylor, G., Studer, C. & Goldstein, T. Visualizing the Loss Landscape of Neural Nets. arXiv: 1712.09913. http://arxiv.org/abs/1712.09913 (Dec. 28, 2017).

20. Wang, Z., Bovik, A., Sheikh, H. & Simoncelli, E. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing* **13,** 600–612 (2004).

21. Yu, F. & Koltun, V. *Multi-Scale Context Aggregation by Dilated Convolutions* 2016. arXiv: 1511.07122 [cs.CV]. https://arxiv.org/abs/1511.07122.

22. Blog, N. D. *Optimizing GPU Performance with Tensor Cores* Accessed: 2025-03-25. 2020. https://developer.nvidia.com/blog/optimizing-gpu-performance-tensor-cores/.

23. Lim, B. *et al.* A convolutional neural network for defect classification in Bragg coherent X-ray diffraction. *npj Computational Materials* **7.** Publisher: Nature Research. ISSN: 20573960. arXiv: 2106.16179 (Dec. 1, 2021).

24. Favre-Nicolin, V., Coraux, J., Richard, M.-I. & Renevier, H. Fast computation of scattering maps of nanostructures using graphical processing units. *Journal of Applied Crystallography* **44,** 635–640. https://doi.org/10.1107/S0021889811009009 (June 2011).

25. Kingma, D. P. & Ba, J. *Adam: A Method for Stochastic Optimization* 2017. arXiv: 1412.6980 [cs.LG]. https://arxiv.org/abs/1412.6980.

26. Krull, A., Buchholz, T.-O. & Jug, F. *Noise2Void - Learning Denoising from Single Noisy Images* 2019. arXiv: 1811.10980 [cs.CV]. https://arxiv.org/abs/1811.10980.

27. Atlan, C. *et al.* Imaging the strain evolution of a platinum nanoparticle under electrochemical control. *Nature Materials* **22.** Publisher: Nature Research, 754–761. ISSN: 14764660 (June 1, 2023).

28. Ulvestad, A. *et al.* ARTICLE Avalanching strain dynamics during the hydriding phase transformation in individual palladium nanoparticles. www.nature.com/naturecommunications (2015).

29. Cherukara, M. J., Nashed, Y. S. & Harder, R. J. Real-time coherent diffraction inversion using deep generative networks. *Scientific Reports* **8.** Publisher: Nature Publishing Group. ISSN: 20452322 (Dec. 1, 2018).

30. Scheinker, A. & Pokharel, R. Adaptive 3D convolutional neural network-based reconstruction method for 3D coherent diffraction imaging. *Journal of Applied Physics* **128.** Publisher: American Institute of Physics Inc. ISSN: 10897550. arXiv: 2008.10094 (Nov. 14, 2020).

31. Wu, L., Juhas, P., Yoo, S. & Robinson, I. Complex imaging of phase domains by deep neural networks. *IUCrJ* **8,** 12–21. ISSN: 20522525 (2021).

32. Chan, H. *et al.* Rapid 3D nanoscale coherent imaging via physics-aware deep learning. *Applied Physics Reviews* **8.** Publisher: American Institute of Physics Inc. ISSN: 19319401. arXiv: 2006.09441 (June 1, 2021).

33. Wu, L. *et al.* Three-dimensional coherent X-ray diffraction imaging via deep convolutional neural networks. *npj Computational Materials* **7.** Publisher: Nature Research. ISSN: 20573960. arXiv: 2103.00001 (Dec. 1, 2021).

34. Yao, Y. *et al.* AutoPhaseNN: unsupervised physics-aware deep learning of 3D nanoscale Bragg coherent diffraction imaging. *npj Computational Materials* **8.** Publisher: Nature Research. ISSN: 20573960. arXiv: 2109.14053 (Dec. 1, 2022).

35. Zhuang, Z., Yang, D., Hofmann, F., Barmherzig, D. A. & Sun, J. Practical Phase Retrieval Using Double Deep Image Priors. *ArXiv* **abs/2211.00799.** https://api.semanticscholar.org/CorpusID:253255048 (2022).

36. Ulyanov, D., Vedaldi, A. & Lempitsky, V. Deep Image Prior. *International Journal of Computer Vision* **128,** 1867–1888. ISSN: 1573-1405. http://dx.doi.org/10.1007/s11263-020-01303-4 (Mar. 2020).

37. Yu, X. *et al.* Ultrafast Bragg coherent diffraction imaging of epitaxial thin films using deep complex-valued neural networks. *npj Computational Materials* **10.** Publisher: Nature Research. ISSN: 20573960 (Dec. 1, 2024).

Annexes

Appendix A

Appendix