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

0.1 Introduction

In this manuscript, the use of Deep Learning methods, and more in general of GPU accelerated optimizations, for the advance of the data analysis in Bragg Coherent Diffraction Imaging (BCDI) will be presented. However, before delving into the study's developments, I would like to share with the reader a reflection that has taken shape over the course of this PhD, serving as a kind of preface. In particular, I have come to observe that, unlike other more fundamental scientific investigations, this work originates from the practical limitations of the technique in question. It is indeed because the detectors are imperfect—unable to record flawless images due to gaps, or incapable of capturing phase information because its oscillations are too rapid—that one is compelled to manipulate the available data with sophisticated algorithms. And, as often happens in science, compensating for these technical shortcomings leads to the development of tools rooted in the most abstract realms of mathematics and information theory. How much missing information can one extract from a signal? How can it be extracted, and under what conditions? In which circumstances is it easier, and why? Thus, a fascinating world opens up not when we directly investigate the foundations of matter, but when we examine *how* we go about investigating them.

Although this manuscript is ultimately focused on the specific cases of BCDI gap inpainting and phase retrieval, I hope to convey at least a bit of the wonder and awe that comes from knowing that such applications draw their roots from far deeper, more general, complex, and abstract themes.

0.1.1 PhD objectives

As we will soon present in details, we can anticipate that BCDI is a powerful imaging-microscopy technique performed at synchrotron and XFEL facilities and for its non-invasive, high spatial resolution, investigation of the atomic structure of single crystal nano-particles it has already been proved successful in many diverse fields. In fact, the study of the internal strain distribution, defects population and morphology at the nanometer scale (typical BCDI resolution is of the order of 10 nm) under various physio-chemical environments is of crucial importance for fundamental science as well as for engineering applications [1]. Since the very first use in 2001 by Prof. Ian Robinson and coauthors [2] for the imaging of gold particles, BCDI has been employed for the analysis of materials of relevance for nanotechnology and electronics [3] as well as for Li-ion and Na-ion batteries [4, 5], catalysis [6] and recently on biological materials too [7].

However, this technique strongly relies on computer algorithms for the transformation of the acquired diffraction pattern into the real space particle shape and strain field. For this reason, in the past years, numerous efforts have been made to make the computational aspect of BCDI robust and reliable. Some of these developments will be presented later on, while here we want to highlight that the research in this field has become even more active and prosperous with the advent of machine learning (ML). Moreover, the recent upgrade to fourth-generation x-ray light source of many synchrotrons across the planet (MAX-IV - Sweden in 2017, ESRF-EBS - France and Sirius - Brazil in 2020 and others scheduled for the coming years) is boosting the crystalline nano-imaging techniques such as Bragg CDI and Bragg ptychography [8, 9]. Hence, the need for fast and optimized ways to handle the large amounts of experimental

data. It is exactly in this context that this study is framed, with the initial goal of exploring the advantages that machine learning algorithms can bring to the BCDI technique.
Specifically, the work has focused on two main tasks of primary importance for the reliability of the processed data. As briefly mentioned above and discussed more in details later, the detectors employed for x-ray imaging techniques have the main limitation of not being capable of record the phase of the impinging x-ray light, thus leaving its retrieval to computer algorithms. Moreover, these detectors are built with some non sensing

Chapter 1

Bragg Coherent Diffraction Imaging

1.1 Single crystal diffraction

1.2 Phase Problem

Chapter 2

Convolutional Neural Networks

2.1 Introduction on neural networks

2.2 Convolutional

2.3 U-Net and MSD-Net

Chapter 3

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.

3.1 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 [10, 11]. 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 0.1 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[12].

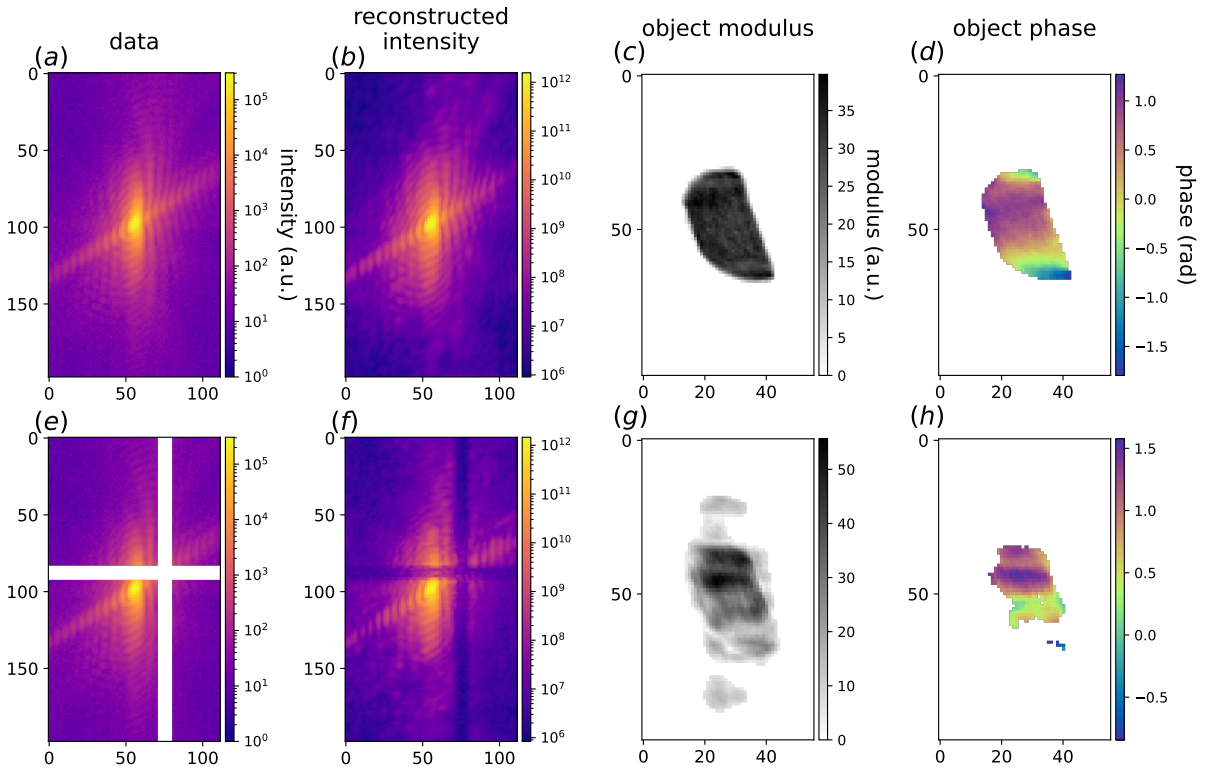


Figure 3.1: Effect of detector gaps in BCDI reconstructions (a) The central xz slice of an experimental diffraction pattern. **(b)** The same slice of the diffracted intensity calculated from the retrieved object. **(c - d)** xz slice of the modulus and phase respectively of the particle obtained from 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.

3.2 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. 3.2)

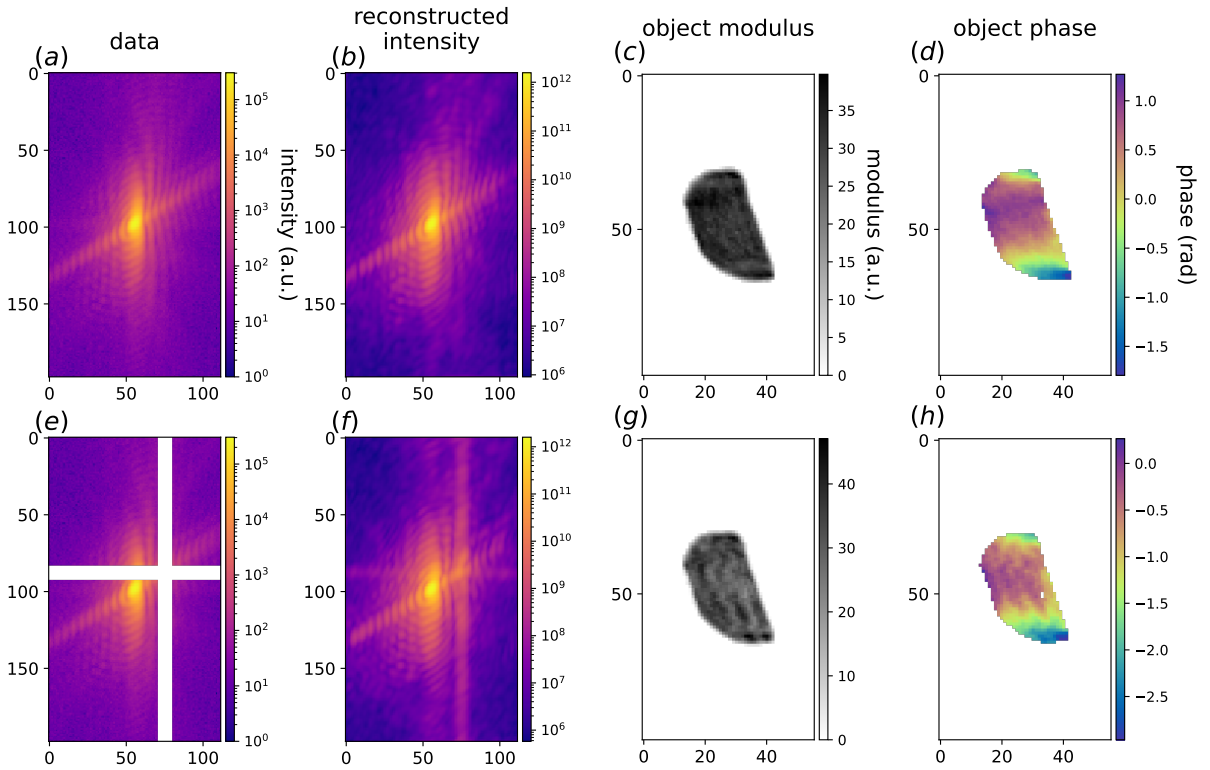


Figure 3.2: Masking the gap region during phasing (a) The central xz slice of an experimental diffraction pattern. **(b)** The same slice of the diffracted intensity calculated from the retrieved object. Comparing this figure with 3.1**(b)** one can see that when excluding the gap region from the phasing 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.

3.2.1 Background on Image Inpainting Research

Computational image inpainting has been widely studied in the field of photography and imaging for many years [13, 14]. 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 [15] to the use of PDEs as Navier-Stokes equations proposed by Bertalmio *et al.* [16] and then again from sparse representations [17] to hybrid methods combining variational and statistical methods [18]

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.* [19], and have also been used for image inpainting (e.g. [20]). 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.* [14, 21]. as well as this blog article [22]. 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 [23] and to 2020 for 2D microscopic images [24]. 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 [25]. 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 [26]. 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.

3.3 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.

3.3.1 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).

3.3.2 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 [27]. 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 [28]. The last activation layer of the model is a sigmoid function that guarantees an output bounded between 0 and 1

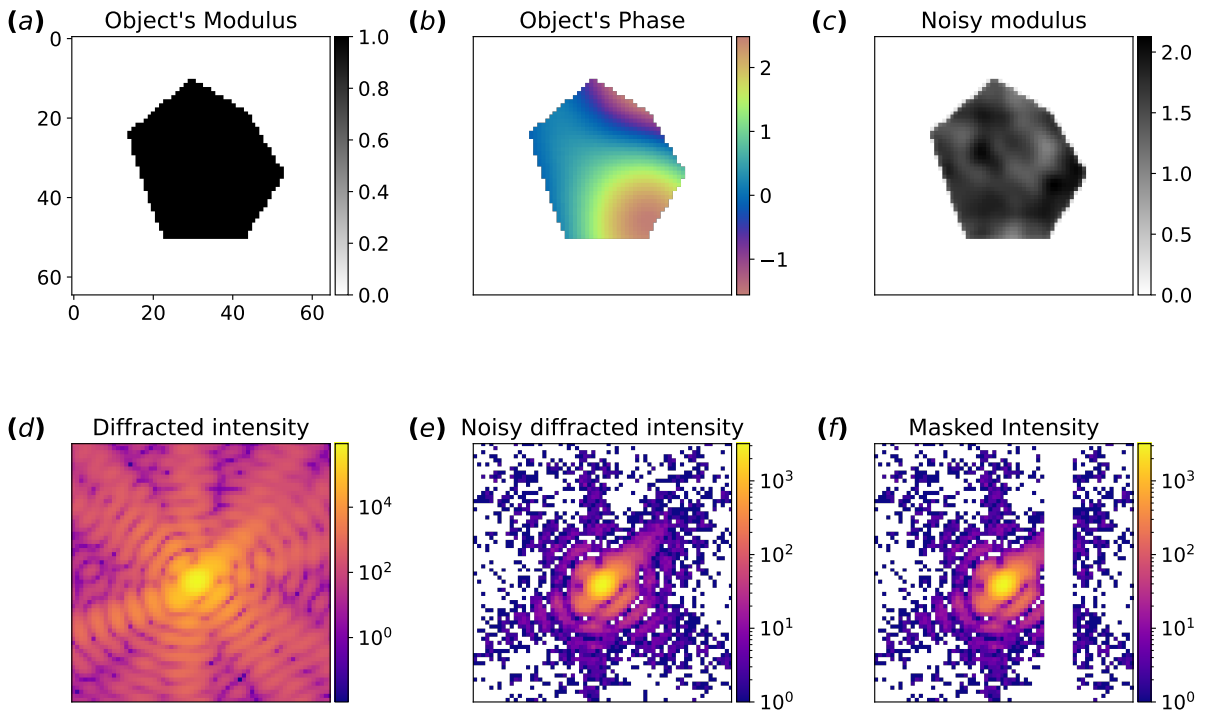


Figure 3.3: Steps for the simulation of a single 2D diffraction pattern (a) Simulated modulus of a 2D object with random shape and compact support. **(b)** Simulated object's phase **(c)** Object's modulus after smoothening the edges and adding random Gaussian noise. **(d)** Squared modulus of the Fourier 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) [29] as well afterwards and compared the results after the same training. Here in Fig. 4.11 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).

$$PCC = \frac{\sum_{i\in\text{gap}}(\mathbf{I}_i^{\text{true}} - \langle\mathbf{I}^{\text{true}}\rangle)(\mathbf{I}_i^{\text{pred}} - \langle\mathbf{I}^{\text{pred}}\rangle)}{\sqrt{\sum_{i\in\text{gap}}(\mathbf{I}_i^{\text{true}} - \langle\mathbf{I}^{\text{true}}\rangle)^2}\sqrt{\sum_{i\in\text{gap}}(\mathbf{I}_i^{\text{pred}} - \langle\mathbf{I}^{\text{pred}}\rangle)^2}}, \tag{3.1}$$

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 ([25]), we considered a Mixed-

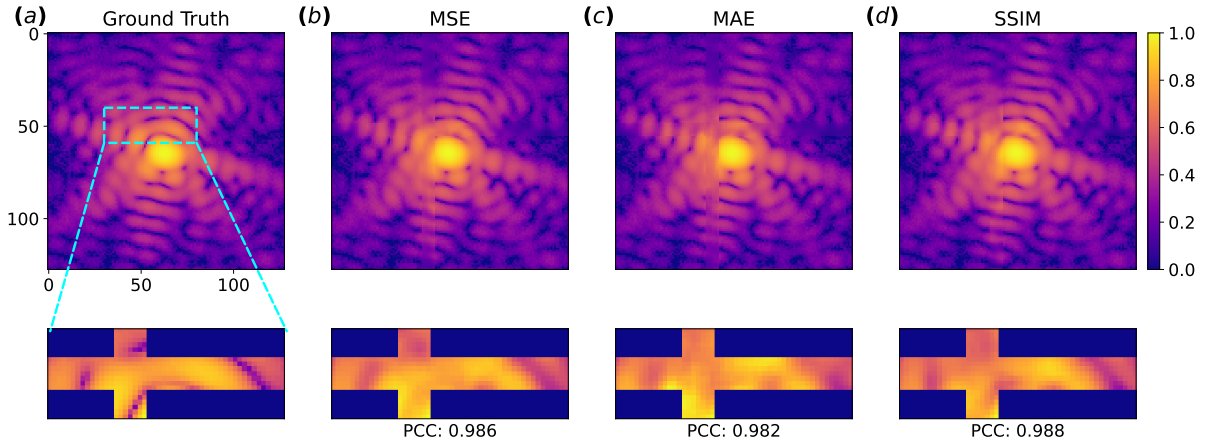


Figure 3.4: Comparison of different losses Results on a test simulated diffraction pattern for the inpainting of a 9 pixel-wide cross-shaped gap produced by the same UNet model trained for 10 epochs with different loss functions. **(a)** Shows the ground truth. **(b)** The prediction of the model trained 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 [30]. The characteristics of each model are summarized in form of pseudo - code in Table 3.1.

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

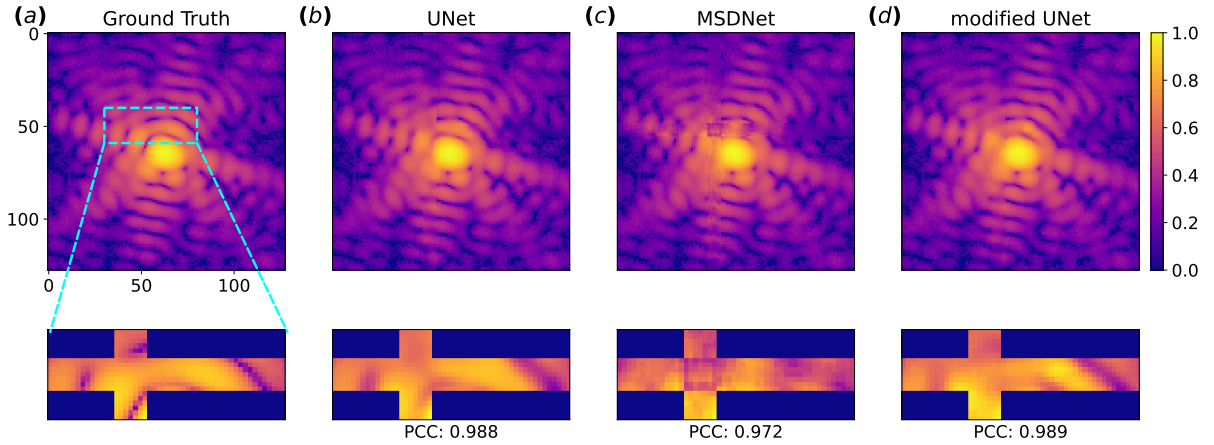


Figure 3.5: Comparison of different models Results on a test simulated diffraction pattern for the inpainting of a 9 pixel-wide cross-shaped gap using three different models trained with the same loss function. **(a)** Shows the ground truth. **(b)** The prediction of the U-Net, **(c)** of the MSD-Net, **(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.

3.3.3 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. 3.6 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 3.1: 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

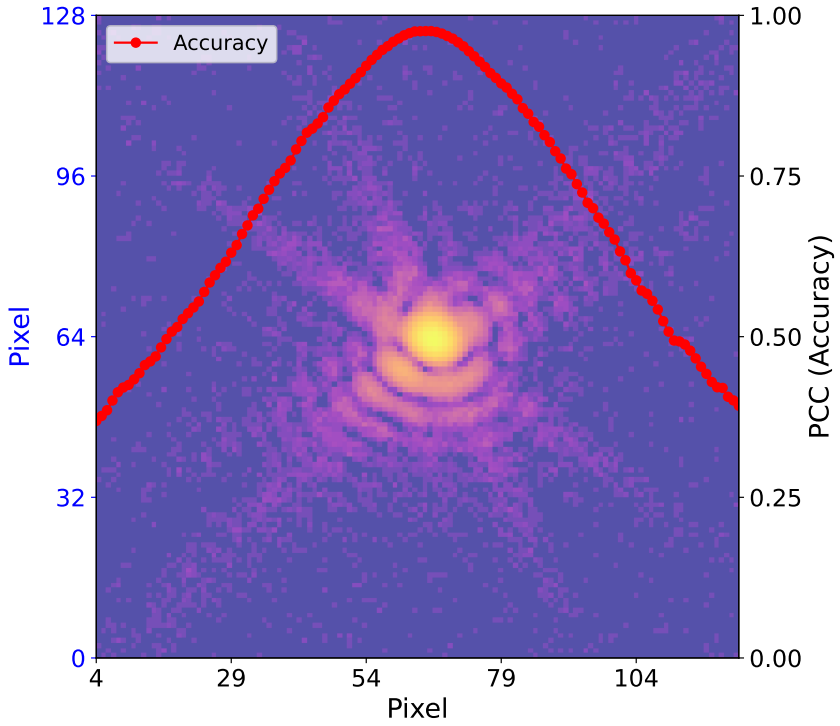


Figure 3.6: (Accuracy VS Gap position) Average Pearson Correlation Coefficient calculated over 150 9 pixel-wide vertical predicted gaps for each position of the gap inside the diffraction patterns. The model shows higher accuracies for high intensity regions.

the oversampling (Fig. 3.7). 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.

3.4 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

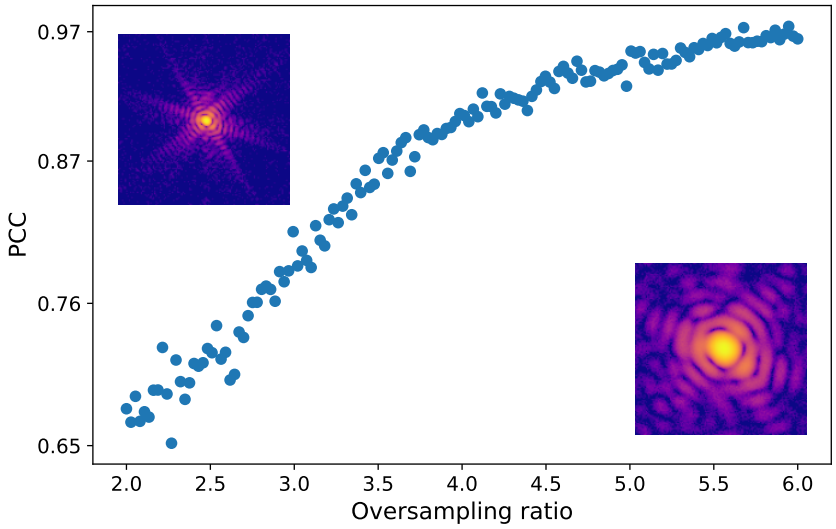


Figure 3.7: (Accuracy VS Oversampling ratio) Average Pearson Correlation Coefficient calculated over 280 9 pixel-wide vertical predicted gaps for each position of the gap inside the diffraction patterns. The model shows higher accuracies for high intensity regions.

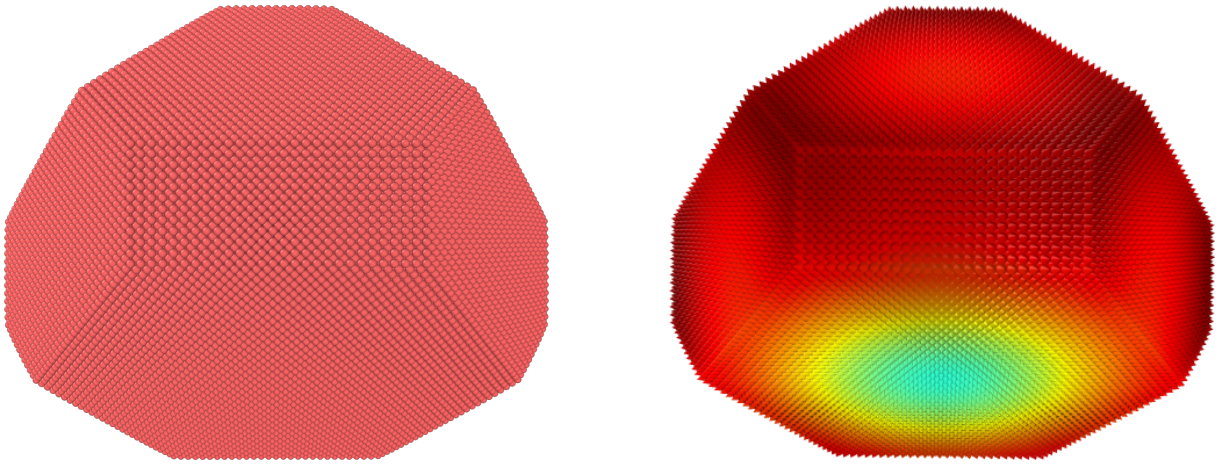
often affected by detector's gaps are necessarily large datasets (e.g. $512 \times 512 \times N_{\text{rocking\_steps}}$). Training a U-Net like model for 3D images of that size is overly expensive in terms of computing memory and time. Moreover, a common problem with this type of architectures is that 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 [31] we opted for 32 as good trade-off between amount of contained information and computing power required for training and inference.

3.4.1 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 [32]. 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 [33]. 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 3.8: (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.3.6) 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.

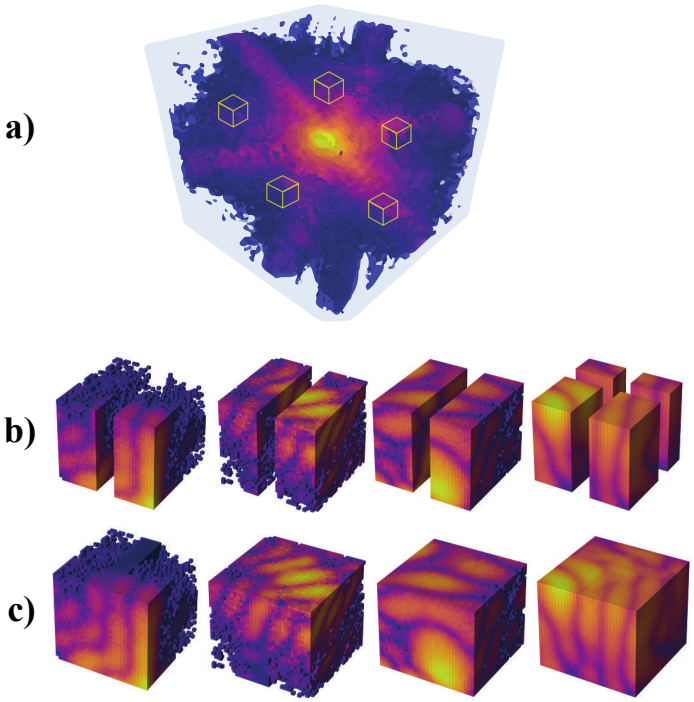


Figure 3.9: Schematic of the sub-volumes extraction. a) The 3D BCDI diffraction pattern and the sub-volumes. **b)** $32 \times 32 \times 32$ pixel-size sub-volumes with 9 pixel-wide vertical and cross-shaped gaps. **c)** Same sub-volumes with the DL inpainted gaps.

3.5 3D model architecture

The DL architecture used for the 3D patching inpainting is illustrated in Fig. 3.10. 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.3.3, 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 [34]. 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.

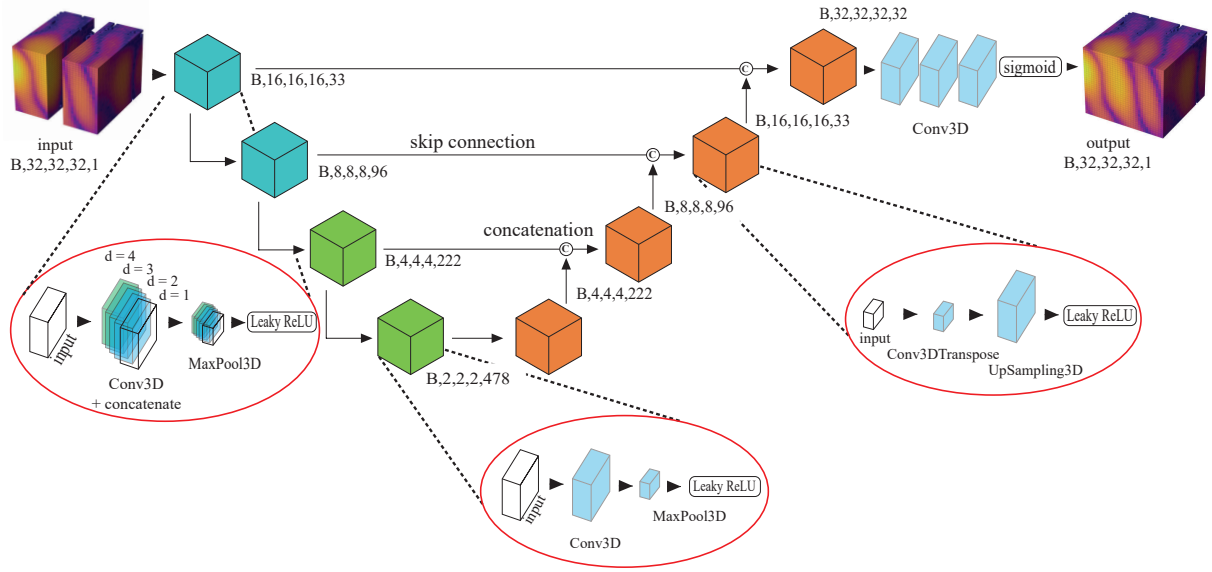


Figure 3.10: Schematic of the 3D model architecture The model uses a modified U-Net structure. In the first two encoder blocks (highlighted by the left red circle), dilated convolutions are applied where the original input is concatenated with its convolutions at various dilation rates (*d = 4,3,2,1*) 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.

3.6 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. 3.11 - 3.12). 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. 3.7. Another curious effect that we can observe, is the "smoothening" of features around noisy areas (see first column in Fig. 3.11 and last column in Fig. - 3.12). 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 [35].

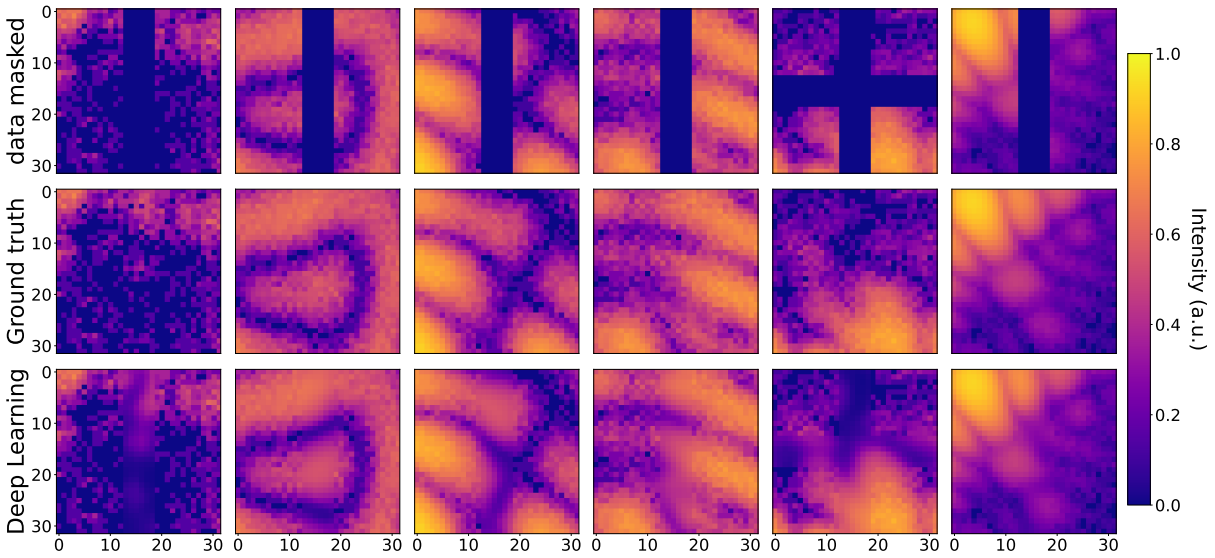


Figure 3.11: Results on portions of test simulated data. Central slices of portions taken from the simulated test dataset. Masked input with 6 pixel-wide gap in the first row, corresponding ground truth and DL inpainted in second and third row respectively.

3.6.1 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.3.14). We have proven that the amount of time for the full inpainting follows a power law (see Fig. 3.13a) and the accuracy starts dropping significantly for more than 5 pixels skipped at the time (see Fig. 3.13b).

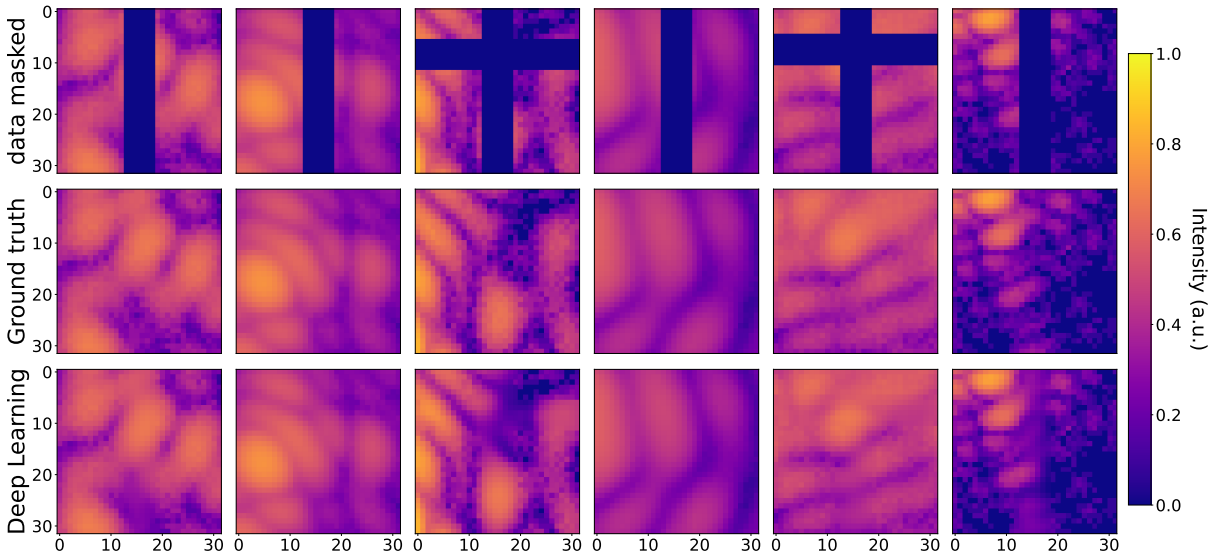
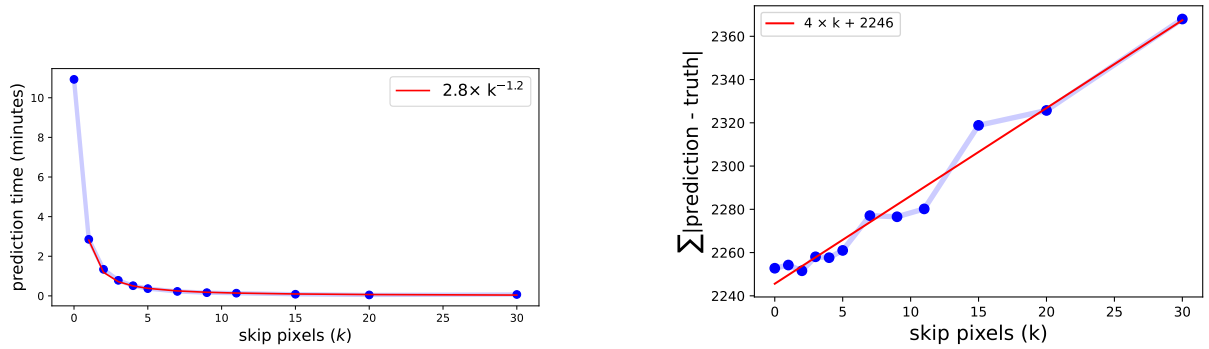


Figure 3.12: Results on portions of test experimental data. Central slices of portions taken from the experimental test dataset. Masked input with 6 pixel-wide gap in the first row, corresponding ground truth and DL inpainted in second and third row respectively.



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

Figure 3.13: (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.

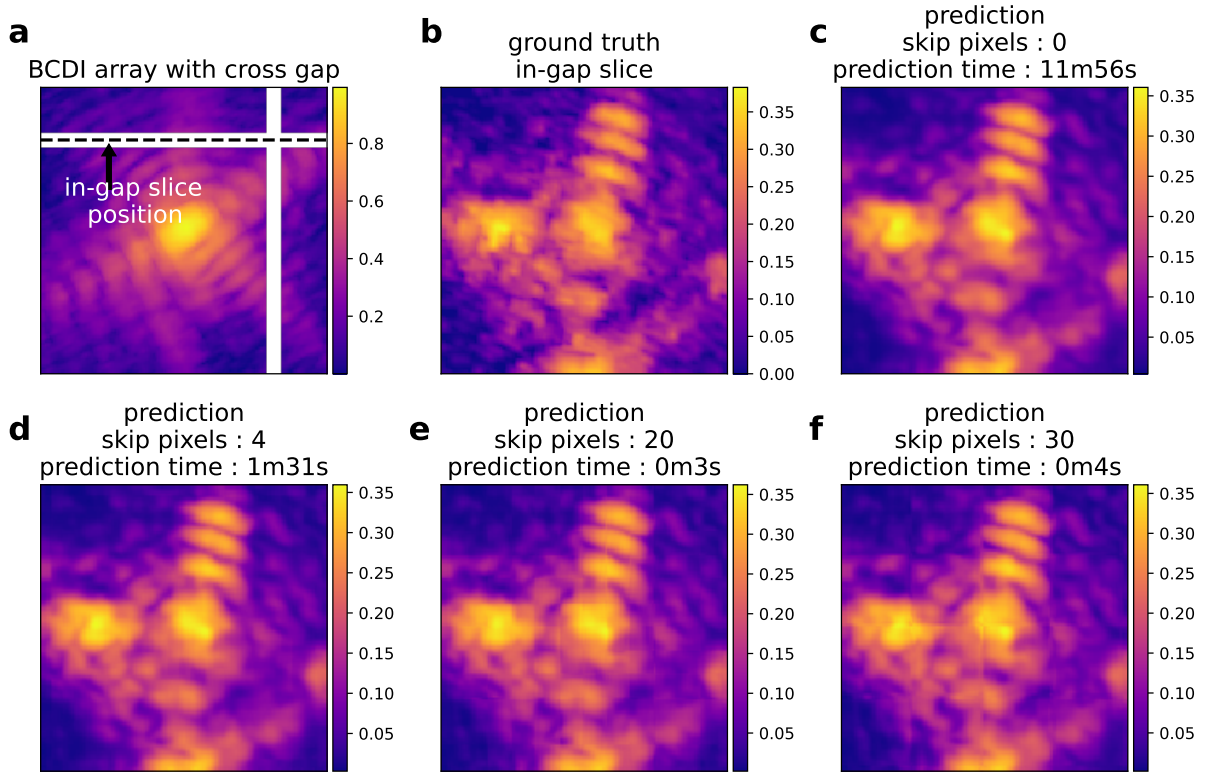
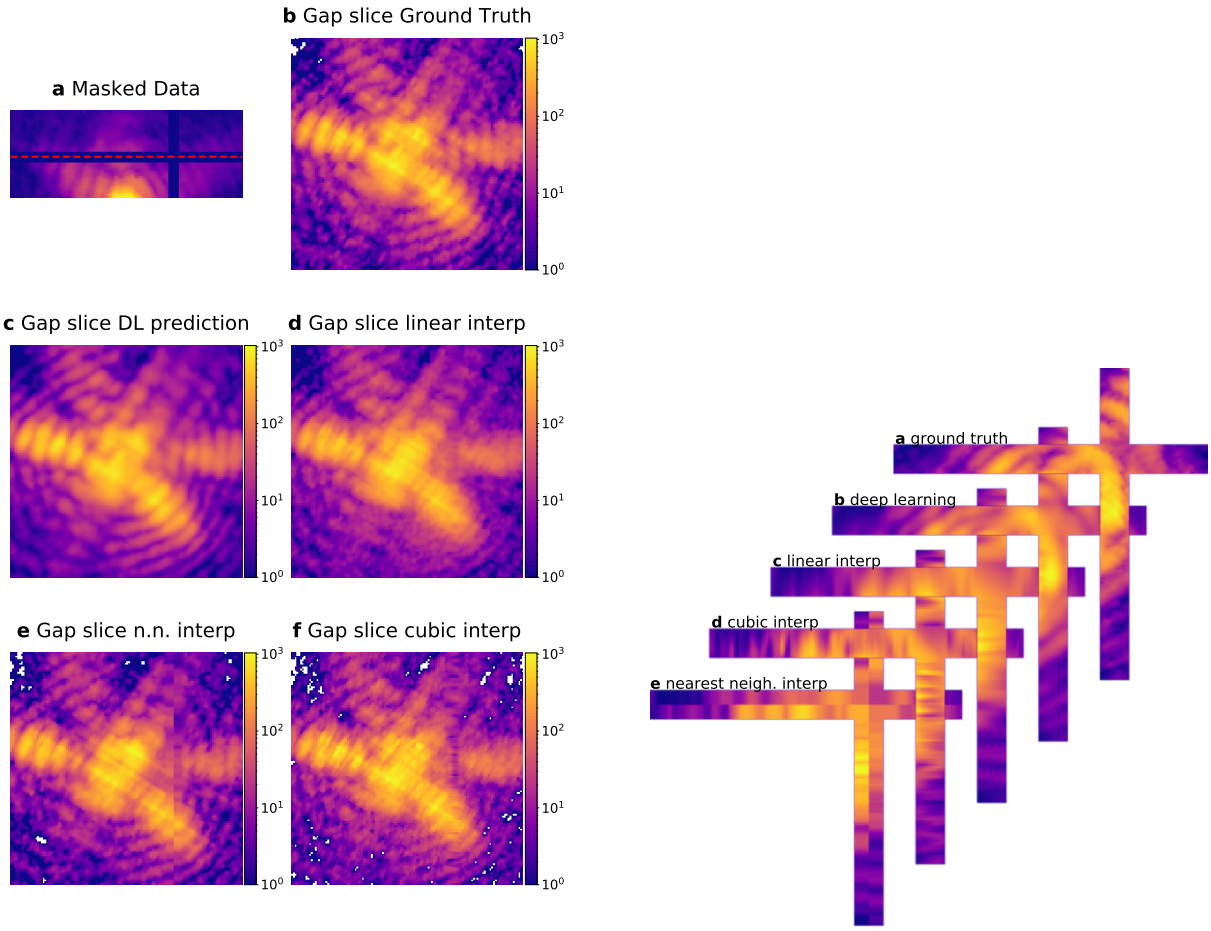


Figure 3.14: Full inpainting of an experimental BCDI pattern for different amounts of skipped pixels. **a** slice of the diffraction pattern perpendicular to the gap plane.**b** Ground truth intensity inside the gap. **c-d-e-f** In-gap prediction with 0, 4, 20 and 30 skipped pixels respectively, with corresponding execution time. Skipping 4 pixels is a good trade off between time and accuracy.

3.7 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. 3.15). In particular, we can notice in the *in-gap slice* (Fig. 3.15a) 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. 3.15b).



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

Figure 3.15

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. 3.16. 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.

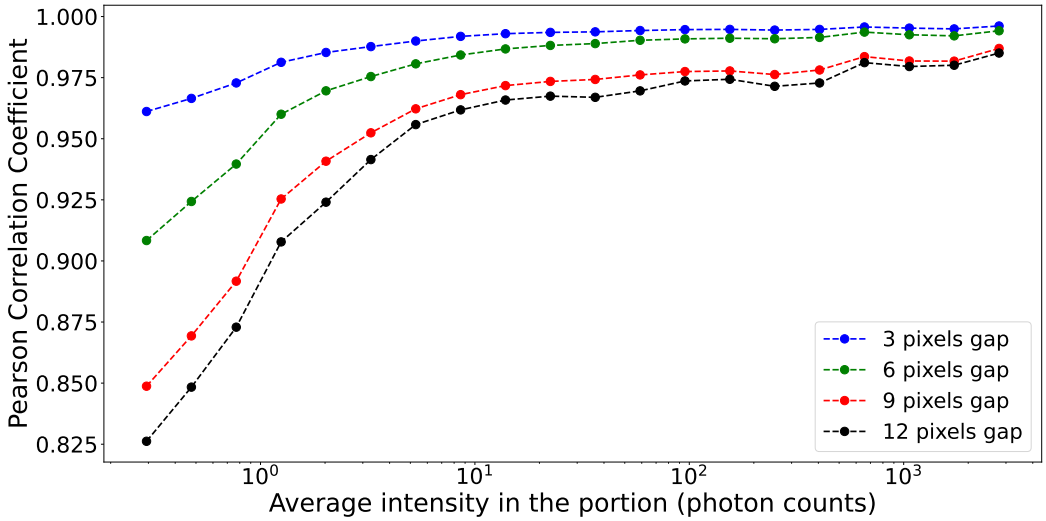


Figure 3.16: Accuracy scores (PCC) of the DL patching model

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. 3.6.1). 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. 3.17. 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).

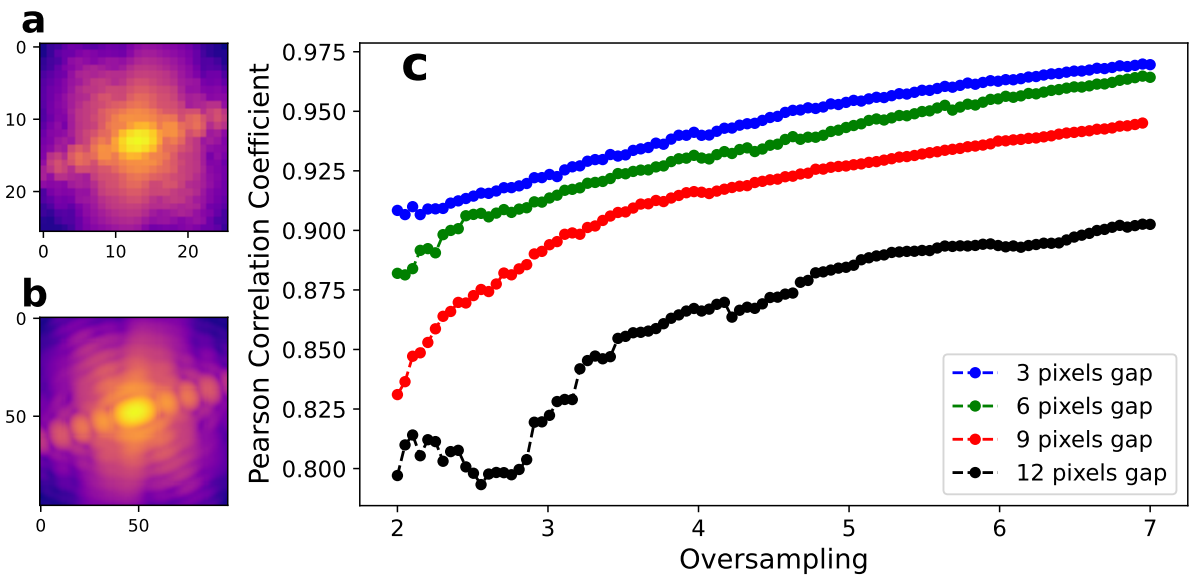


Figure 3.17: Accuracy scores (PCC) of the DL patching model against the oversampling ratio.

3.8 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 [12]. 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.3.18 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.

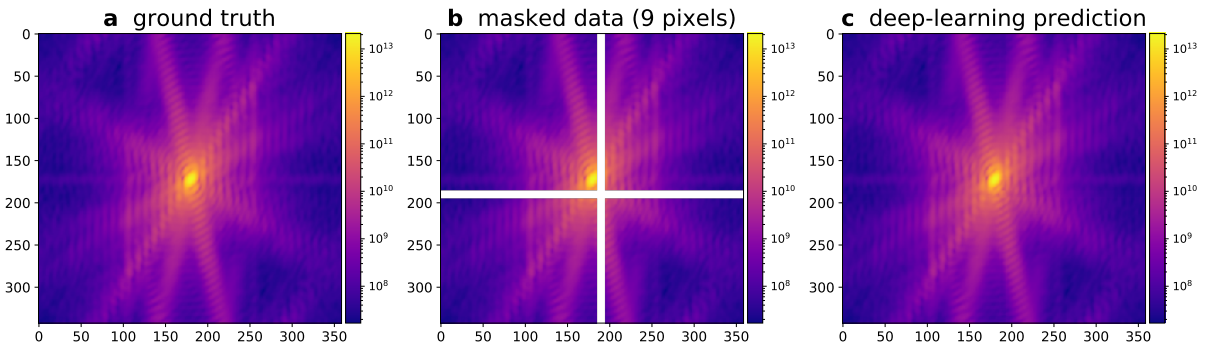


Figure 3.18: Projections along the rocking curve axis of the studied diffraction pattern in log scale. a Ground truth pattern obtained from the $|FFT(\mathbf{O})|^2$. **b** Pattern with a 9 pixel-wide cross shaped gap. The position close to the center of the peak is experimentally unlikely but here it 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.

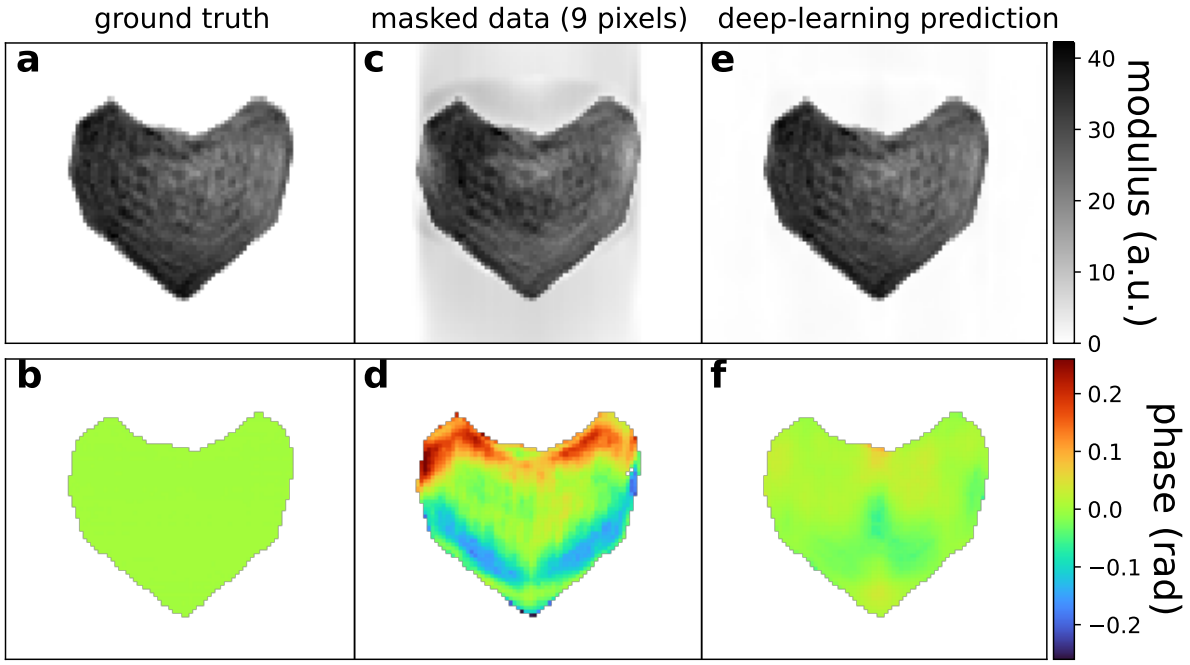


Figure 3.19: Reconstructed objects.a-b Ground truth modulus and phase. **c-d** Modulus and phase of $\mathbf{O}_{gap}$. **e-f** Modulus and phase of $\mathbf{O}_{DL}$

Figure Fig.3.19 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 [6] or in situ gas experiments [36]; 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.3.20 depicts the phases of $\mathbf{O}_{gap}$ and $\mathbf{O}_{DL}$ for the different gap sizes considered in this study while Fig.3.21 the strain distribution in the XY plane.

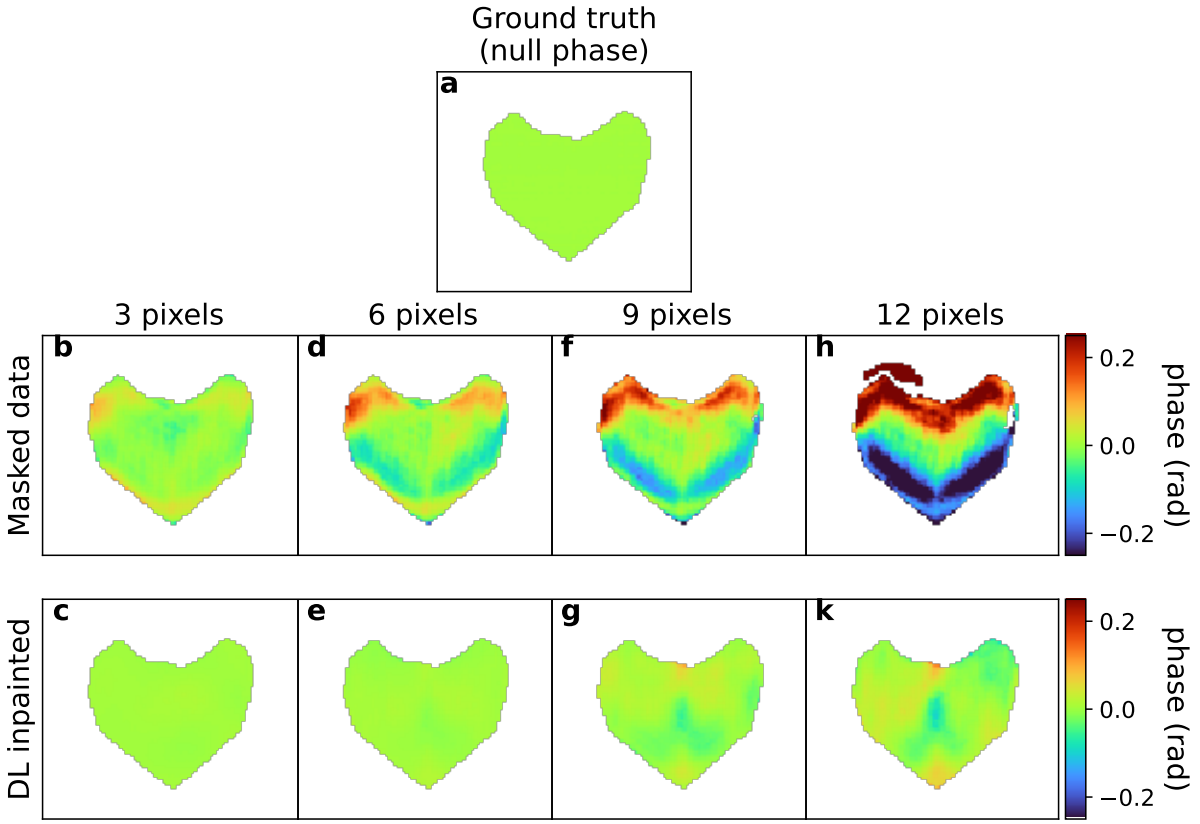


Figure 3.20: Artifacts on the phase of $\mathbf{O}_{gap}$ for different gap sizes, and phases of the corresponding $\mathbf{O}_{DL}$

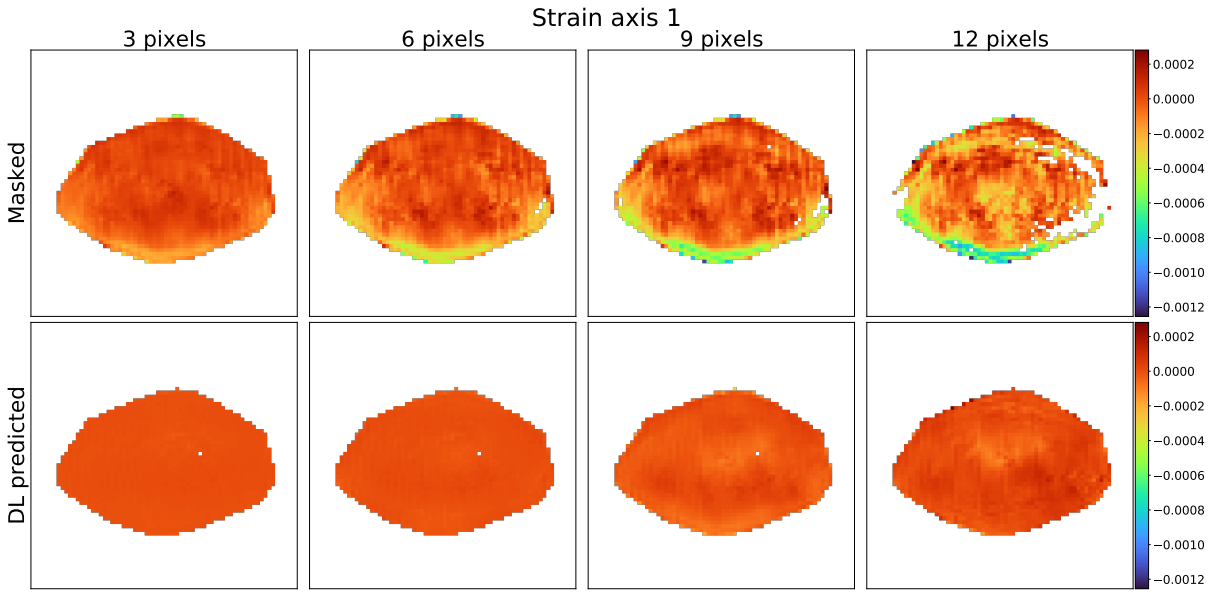


Figure 3.21: Strain distribution in the central XZ slice of $\mathbf{O}_{gap}$ for different gap sizes and corresponding results for $\mathbf{O}_{DL}$

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.3.22). The trend of the strain RMSE resembles indeed the curve showed in [12], 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.3.23 whereas our DL model maintains the average strain around zero.

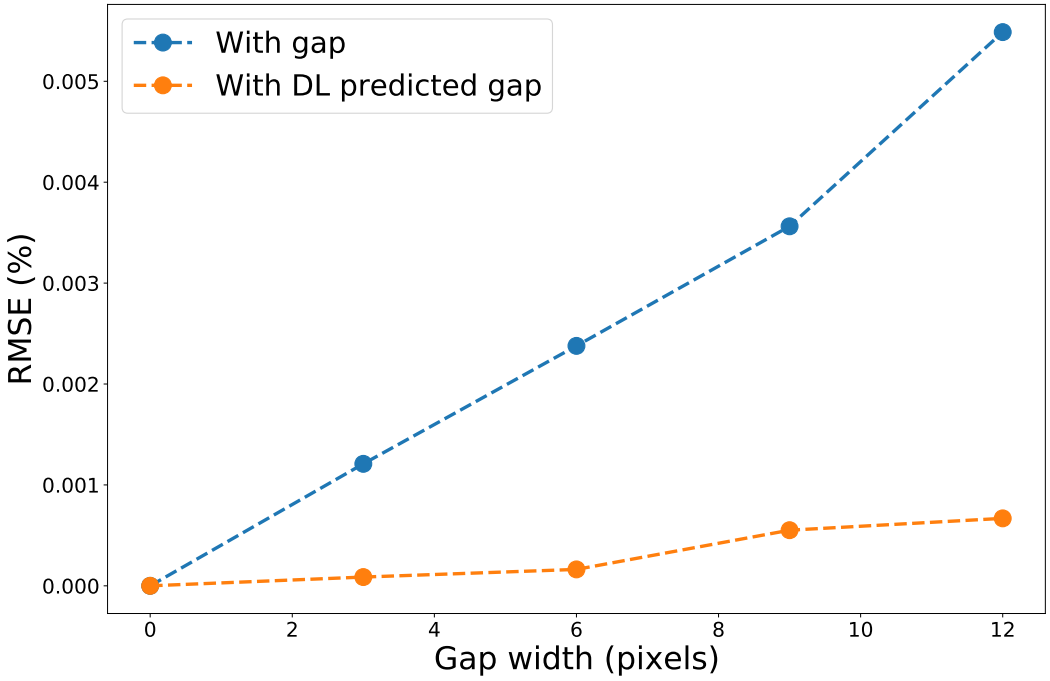


Figure 3.22: RMSE of the strain field versus gap size. For both cases of masked and DL inpainted diffraction patterns. For all gap sizes, the DL inpainted diffraction patterns yield a smaller error.

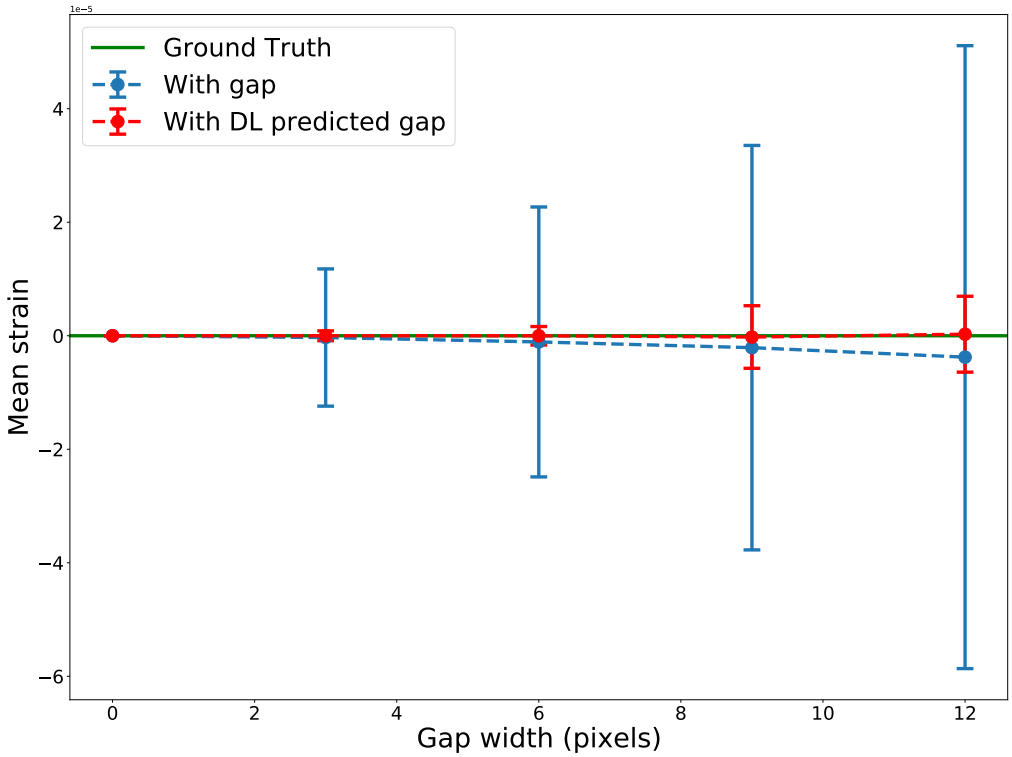


Figure 3.23: RMSE of the strain field versus gap size.

3.8.1 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. 3.24 - 3.25, 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.

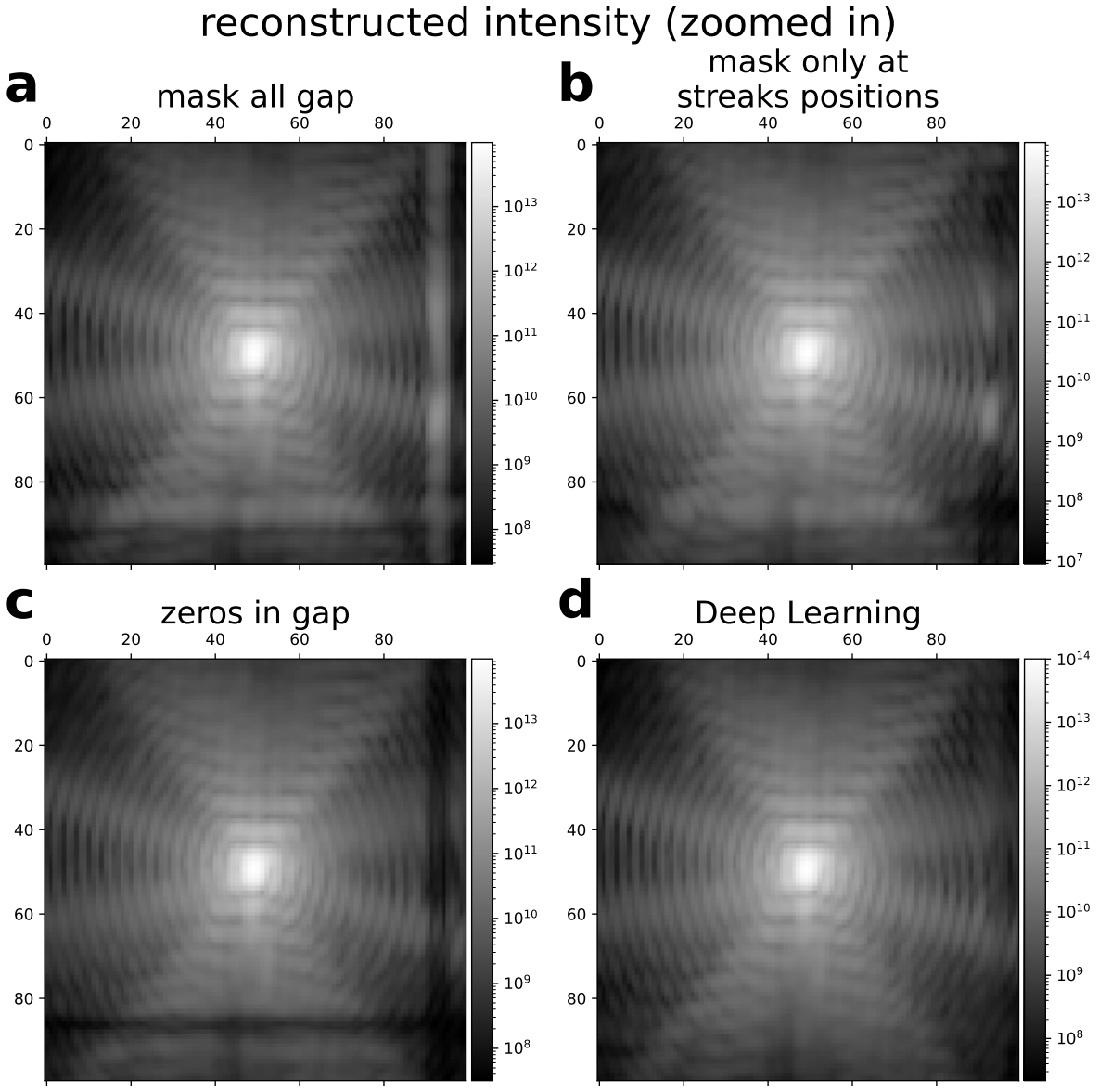
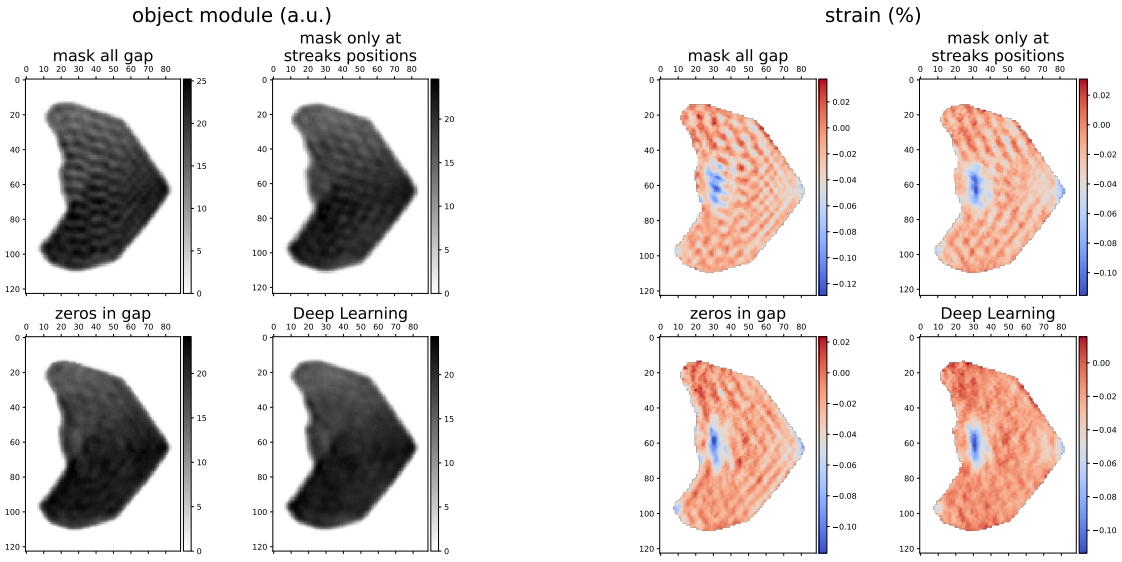


Figure 3.24: Zoom on the projection along the rocking curve axis of the Pt THH BCDI pattern calculated from the reconstructed object obtained with PyNX software using **a** a mask on the gaps, **b** a mask on the streaks only, leaving zeros inside the gaps **c** and inpainting the gaps with our DL model.



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

Figure 3.25: 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.

3.9 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.3.9. 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.

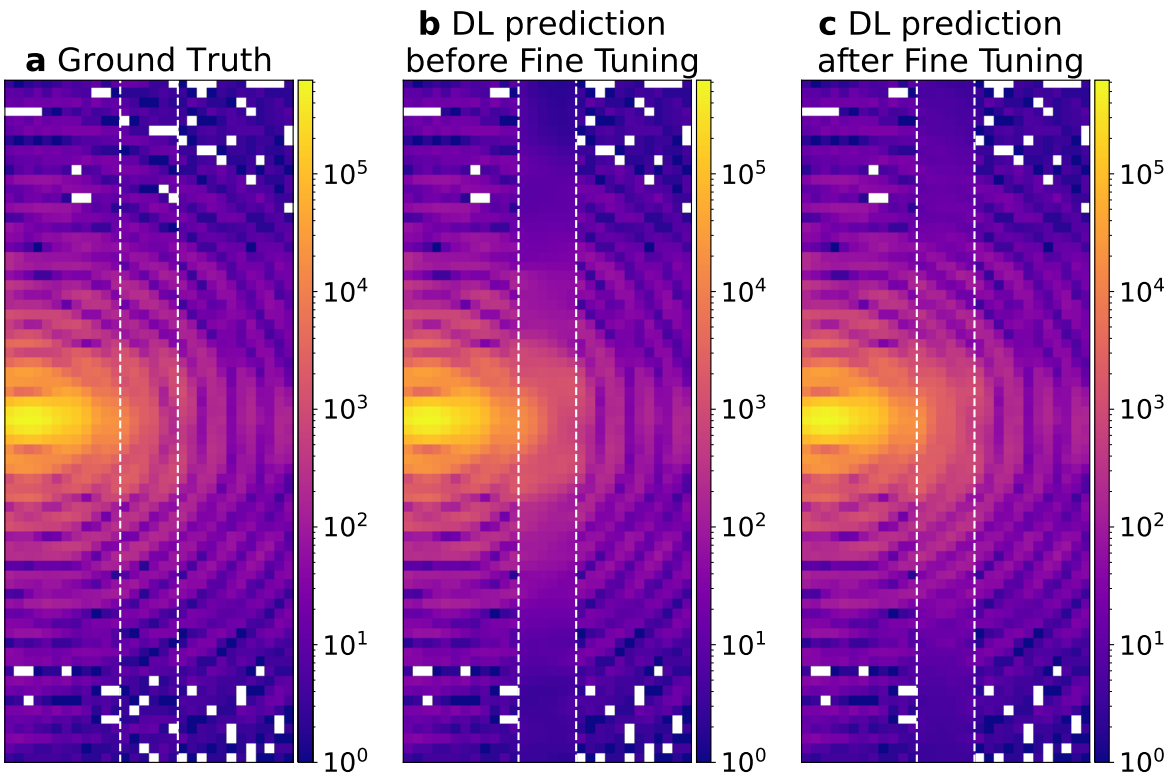


Figure 3.26: Example of improved accuracy after fine-tuning of the DL model. The fringes are better recovered after 5 epochs of fine-tuning.

Chapter 4

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 DL based approaches. Here I will discuss the main steps of this journey, starting off from the analysis of the most relevant works in literature and concluding with some comments on the final version of a DL model for highly strained particles. The latter has become the subject of an article, currently submitted, 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.

4.1 State of the art

In this paragraph I 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, I 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 [37]. 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 showed 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 [38] that employs a CNN model for 3D diffraction patterns. The fundamental difference is that the object's support was defined by its surface only, as it is assumed to be *compact* and *homogeneous* inside. Moreover, the surface was parametrized by

spherical harmonics and the DL model was trained to predict 28 of the first even coefficients of the spherical harmonics. The model architecture was 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 showed good performance 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, [39], 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 [40] 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 [32] and described above in Section 3.4.1. Training the CNN on a restricted set of such created BCDI patterns biases the predictions towards physically meaningful particles. Moreover, it is interesting to notice that the training of the model was conducted in a sort of unsupervised fashion as the loss function calculates the differences between the target diffracted intensity and the intensity obtained by the kinematic sum over the lattice sites of the predicted complex object. 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.* [41] published in the same year, which lifted the size to 64 pixel-sided cubes, that the model can be tested on several experimental cases. Their CNN model maintained 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 claimed 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 [42], 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 was trained to predict an object the diffracted intensity of which matches the observed one according to a normalized Mean Absolute Error metric. The model showed 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 [43] in which two CNNs are used in the "deep image prior" (DIP) framework. DIP [44] 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.* [45]. In this paper the authors proposed 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. Moreover, the authors made use of *skip connections* between encoder and decoder to enhance the training. This is a rather peculiar as this kind of residual links are typically used, in convolutional encoder-decoder networks, for tasks in which the input and output images are visually similar (i.e. segmentation, denoising, inpainting), thus, where it is more evident the information flow from the two blocks of the network. The model was used for the phase retrieval of experimental 2D diffraction patterns, for which an unsupervised refinement was used as well.
Before proceeding with our study, Table 4.1 summarizes the key features of the works from the two leading BCDI research groups at Brookhaven and Argonne National Laboratories, highlighting similarities and differences to guide the development of our model.

	Architecture	**Last Activation Layer**	**Loss Function**	**Refinement**
Cherukara - 2018 [37]	Two different UNets	Sigmoids	Cross Entropy	-
Wu - 2020 [39]	Encoder / 2 Decoders	ReLU	MSE on mod and phase + PCC on magnitudes	Iterative
Chan - 2021 [40]	Encoder / 2 Decoders	ReLU	MAE on normalized magnitudes	Automatic Differentiation
Wu - 2021 [41]	Encoder / 2 Decoders	LeakyReLU	MSE on mod and phase + PCC on magnitudes	Transfer learning + unsupervised training
Yao - 2022 [42]	Encoder / 2 Decoders	Sigmoid and Tanh	MAE on normalized magnitudes	Iterative (50 ER)
Yu - 2024 [45]	Complex encoder-decoder + skip connections	ReLU	MAE on real + MAE on imaginary	Transfer learning + unsupervised training

Table 4.1: Comparison of deep learning-based phase retrieval approaches.

First, it is interesting to notice that the architecture's choice, from treating the object's modulus and phase separately with two different detached networks, moved over the years to a single "standard" U-Net that accounts for the complex nature of the data. Second, I noticed that the choice of the last activation layers, which are the ones producing the modulus and phase outputs, in their final value range, is not uniform throughout the articles. While ReLU and sigmoid ensure real positive outputs, thus normally appropriate for real positive quantities like the modulus, LeakyReLU and Tanh allow for negative values as well, making them valid options for the phase array. Nevertheless, it seems that their impact is marginal since in some cases the model is able to predict correct moduli from LeakyReLUs and correct phases from ReLUs and sigmoids. Regarding this point, it is worth mentioning that a global offset of the phase that shifts the whole range to the real positive axis does not physically alter the solution. This would mean that a ReLU can still correctly yield a phase array, just shifted by a positive constant. The same holds for the sigmoid, as long as the phase span fits in the range of the activation function.
The most important component of the model is the loss function. Except the first work that employs a cross entropy loss, normally used for classification tasks, other works opt for MAE and MSE, of standard use for regression and PCC as well. Typically, when the loss is calculated between intensities the MAE and the PCC are used as they are more suitable for the high

dynamic range of the diffraction patterns. MSE in fact, "would overly de-emphasize errors in mid-intensity regions of the images" [40]. Lastly, I have listed the different ways used to refine the DL predictions. Here we can notice that very soon GPU accelerated gradient descent methods have been used in replacement of conventional iterative algorithms. The unsupervised training allows to easily switch from inference to refinement using the same model in the same GPU optimized computing environment guaranteed by machine learning libraries like PyTorch and Tensorflow.

4.2 Reciprocal space phasing

From the study of the literature I have started to delineate our approach, taking inspiration from these works but significantly changing the perspective. In particular, it was decided to predict the "reciprocal space" phase (RSP) that is lost during the measurement of the BCDI pattern rather than the complex object in real space. The main, intuitive, reason behind this choice is the visual similarity between the morphology of the diffraction pattern and its corresponding RSP. Furthermore, it is common that many samples studied with BCDI have facets that happen to be, to some degree, parallel with each other, thus interfering like a double-slit with the typical fringes of intensity that correspond to constructive interferences, interspersed with dark regions arising from destructive interferences. In these specific cases, the RSP shows a regular pattern in which there is always a π shift between two crests of the fringes. (add something in the introduction) Once retrieved the RSP one can then recompose the full complex diffracted wave-function and obtain the complex object via inverse Fourier transform.

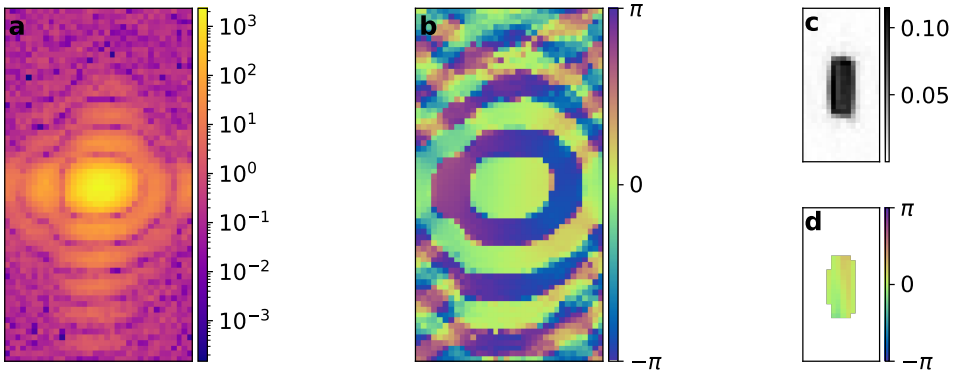


Figure 4.1: Central slice of a typical BCDI pattern (**a**) with the corresponding RSP (**b**) obtained after a successful reconstruction of the object (modulus and phase in **c - d** respectively). It is clear the structural similarity between the diffracted intensity in logarithmic scale and the RSP. Moreover, one can notice that in this case of low strain faceted particle, the RSP varies regularly between 0 and π (or $-\pi$) in correspondence of the intensity fringes.

Moreover, given this "simple" law of constructive-destructive interferences, we hypothesized the possibility to predict patches of this RSP given a portion of diffraction pattern and then, similarly to the inpainting case, stitch together them together and obtain the full RSP. This entails a number of complications related to the so-called phase symmetries that I have encountered during the development of the algorithms and that will be discussed in the next

sections.
Ultimately, the goal of this DL model for phasing is to facilitate the reconstruction of highly strained particles. While other works in literature have mostly leveraged the gain in computing time, here the model aims at tackling those reconstructions for which conventional algorithms struggle to find convergence because of the high strain in the particle. However, in this case, the aforementioned RSP π-shifts in between two fringes is much more complicated since the strong and extended displacement fields inside the crystal alter the Bragg peak, merging and spreading the fringes into an irregularly distributed intensity pattern.

4.3 Dataset creation

I have trained our model in a supervised manner, meaning, in this case, that the training was always conducted on simulated data only, as the RSP is never experimentally detectable. For this reason, I have simulated the training dataset following the same procedure described in Sections 3.3.1 and 3.4.1 for the 2D and 3D cases, respectively. However, in this case, the dataset size was reduced to $64 \times 64 \times 64$ pixels, and no gap was applied. Additionally, I have used the calculated RSP as the ground truth label for training instead of the masked diffraction pattern.

I will anticipate here that for the high strain case I created a dedicated training set simulating the strain by applying an artificial "strong" phase to the particles. In order to have a diverse population of strain distributions I have simulated each object's phase using different functions and parameters, namely: with the sum of two Gaussian functions, with the sum of two cosine functions and using a correlated Gaussian random field (see Appendix). In each case, amplitudes, variances, frequencies, and correlation lengths were randomly chosen to ensure a phase variation within the particle ranging between 2π and 5π. By doing this, I could obtain strongly distorted BCDI patterns, similar to experimental high-strain ones. In particular, the two Gaussian functions phase can closely emulate the effect of the substrate induced strain inside Winterbottom particles.

4.4 2D case low strain

Alike the inpainting case, I have first conducted some preliminary studies in 2D, on noise-less low strain data. Here I will briefly show the model's architecture, the loss function and the results.

4.4.1 Model structure

The architecture that I used has a U-Net like structure with an encoder and a decoder. The encoder is composed of six convolutional blocks through which the input diffracted intensity is progressively reduced from the 64 pixel-side squares to a 1D flattened vector. Each convolutional block is composed of a convolutional layer, a LeakyReLU activation function and a MaxPooling layer that halves the feature's map dimensions. (illustrate the parameters later).
At the end of the encoder the so-called bottleneck composed of a convolutional layer followed

by a LeakyReLU activation processes the feature map before passing it to the decoder which, by means of transposed convolutions, LeakyReLU activations and UpSampling layers, brings back the feature map to the input's size. Skip connections between encoder and decoder blocks are employed as well. The output tensor is the result of a last single-channeled convolutional layer with no activation function. In this way we let the model predict unbounded tensors to account for the phase symmetries (see Intro).

4.4.2 Input preprocessing

Similarly to the inpainting case, the BCDI patterns have been transformed into logarithmic scale and normalized between 0 and 1. Batches of 32 images at the time were used.

4.4.3 Loss function

The choice of the loss function was firstly based on what was used in literature. A sum of the MSE computed on the objects' amplitudes and one on the phases has thus been used (Eq. 4.2). The ground truth objects were indeed available from the simulated data while the predicted objects have been first calculated with a 2D inverse Fourier transform from the diffracted amplitude and the predicted RSP (Eq. 4.1).

$$\hat{o}(\mathbf{r}) \;=\; \mathcal{F}^{-1}\Big\{\sqrt{I(\mathbf{q})}\, e^{\,i\,\varphi_{\mathrm{pred}}(\mathbf{q})}\Big\}(\mathbf{r}) \quad , \tag{4.1}$$

$$\mathcal{L} \;=\; \frac{1}{N}\sum_{\mathbf{r}}\Big(\big|\hat{o}(\mathbf{r})\big| \;-\; \big|o(\mathbf{r})\big|\Big)^2 \;+\; \frac{1}{N}\sum_{\mathbf{r}}\Big(\phi(\mathbf{r}) \;-\; \phi_{\mathrm{gt}}(\mathbf{r})\Big)^2 \quad , \tag{4.2}$$

4.4.4 Results

The training of the model was conducted on 8500 simulated BCDI patterns over 30 epochs with a learning rate of 0.0003 and monitored both training and validation loss. Here, Fig.4.2 shows the model's loss during the 30 epoch long training. However, despite the good decaying trend, typical of proper training, the model does not perform optimally when tested on new data.

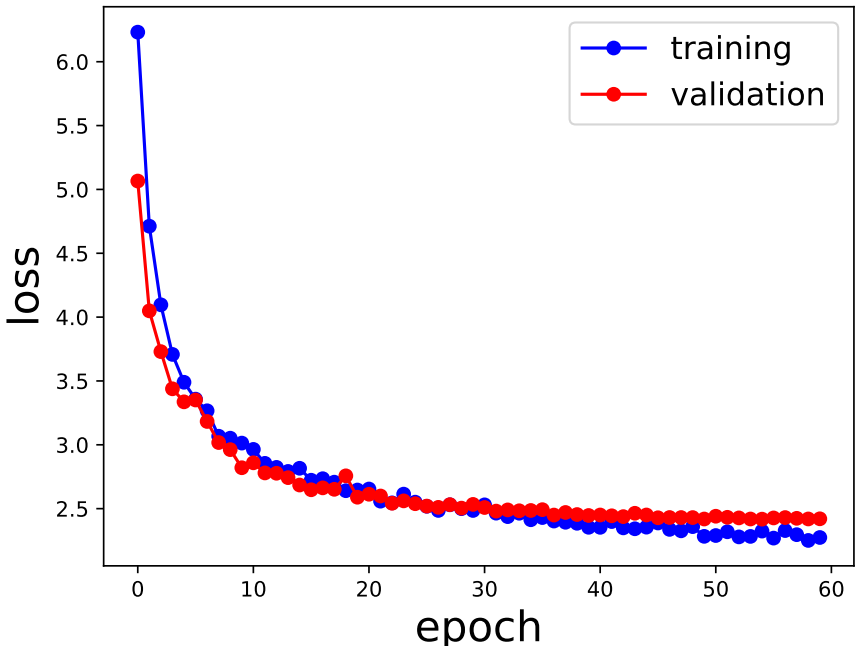


Figure 4.2: Training and validation loss over 30 epochs. The curve suggests a proper learning with no overfitting as both losses are decreasing reaching a plateau and the validation loss follows the same trend of the training loss.

Fig.4.3 illustrates the results of the predicted RSP of some test simulated BCDI patterns. Note that the displayed predicted RSP has been wrapped between 0 and 2π for better comparison with the ground truth but the raw output of the model is in fact an "unwrapped" array. This is expected since no activation layer was applied to the last convolutional layer, meaning that the last operation is the multiplication of the last feature map with the real values inside the convolutional kernel, hence linear.
When comparing the reconstructed objects obtained from the predicted RSP with the ground truth ones (Fig. 4.4) one can draw some interesting conclusions about the model's learning performances. First it can be observed that the model learns the approximate shape and size of the particle, it produces indeed images that resemble reasonable particles, sometimes similar to the ground truth ones. The amplitude is concentrated inside the support with little noise outside and the phase is overall correct around zero. However, when looking more carefully, it is clear that the shape is not quite correct, especially for highly non-centrosymmetric objects. For instance, if we consider the object in Fig. 4.4 **c**, we see that the predicted shape seems to be deriving from the incorrect superposition of the correct shape and its twin, as well correct. More in general it seems that the model tends to predict centrosymmetric objects. According to Sicairos *et al.* [46], if we name $\varphi(\vec{q})$ the correct RSP, this phenomenon is originated by a predicted RSP phase ϕ composed of $\varphi(\vec{q})$ in some regions of the q-space and $-\varphi(\vec{q})$ elsewhere. In other words, the model is not fully able to break the sign symmetry. This subject was recently studied by Zhang and coauthors in [47]. In their study, the authors show that if not broken in the dataset, meaning that during the training the model is exposed to both cases ($\varphi(\vec{q})$ and $-\varphi(\vec{q})$) indistinctly, the model is deceived to a mix of the two, since the sign information cannot be recovered from the input intensity. The authors conclude that in order prevent this detrimental effect, one should break the symmetry in the dataset to bias the model towards

one preferred sign.

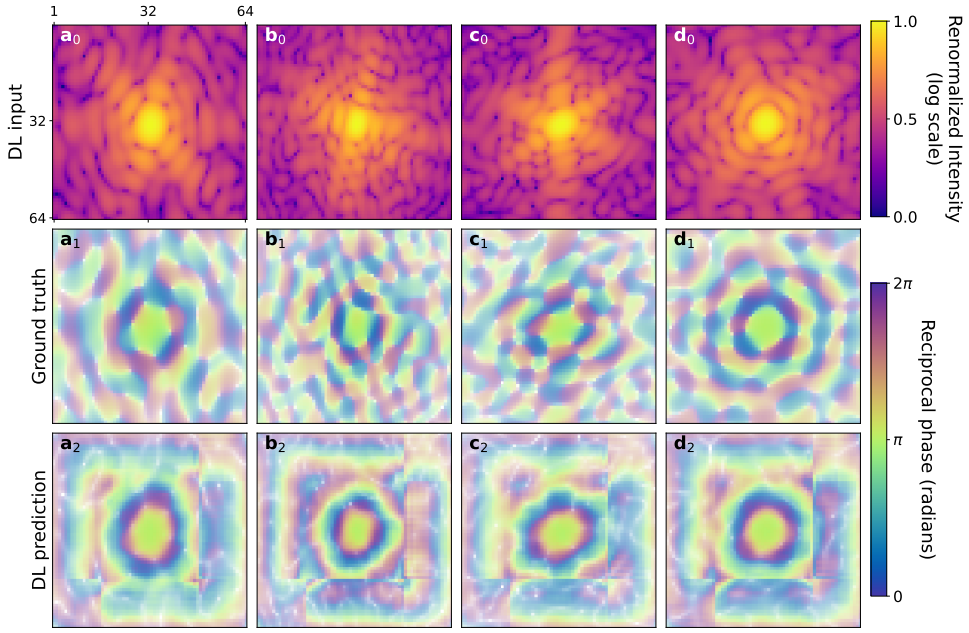


Figure 4.3: Model testing on new 2D data using MSE loss function. First row shows four simulated BCDI patterns, second row the ground truth RSP corresponding to the pattern and last row the DL prediction

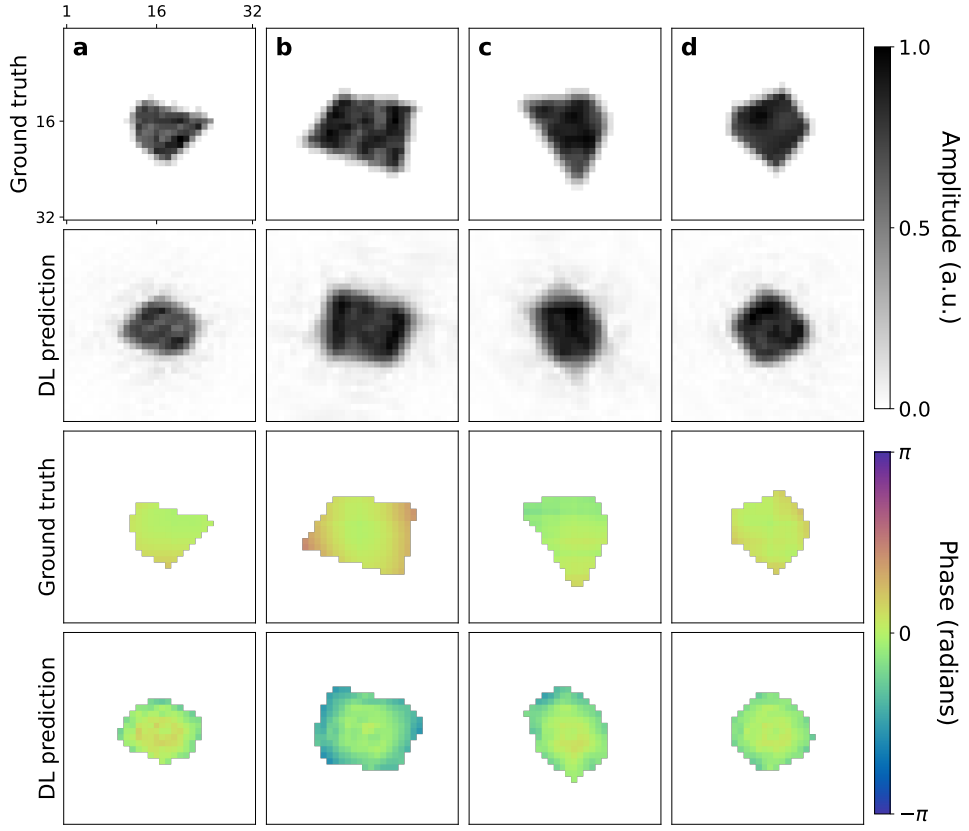


Figure 4.4: Corresponding reconstructed objects. Ground truth and predicted objects' amplitudes (first two rows respectively) and ground truth and predicted objects' phases (first two rows respectively)

The procedure presented in the article for the removal of the phase symmetries consists in: (i) the centering of all the objects in real space (phase ramp removal), (ii) the shift of the RSP such that the zero value in the same array position across the dataset (phase offset removal), and lastly, (iii) they flip the sign of the RSP when its value in corresponding to a fixed position across the dataset is negative. In our case the phase ramp symmetry was already broken by simulating particles with the center of mass in the center of the array. In this way the model is already biased towards the prediction of RSPs that yield centered objects. For the offset and the sign, the method proposed by Zhang *et al.* has been implemented in the model and the results are shown in Fig. 4.5 for the RSP and Fig. 4.6 for the reconstructed objects.

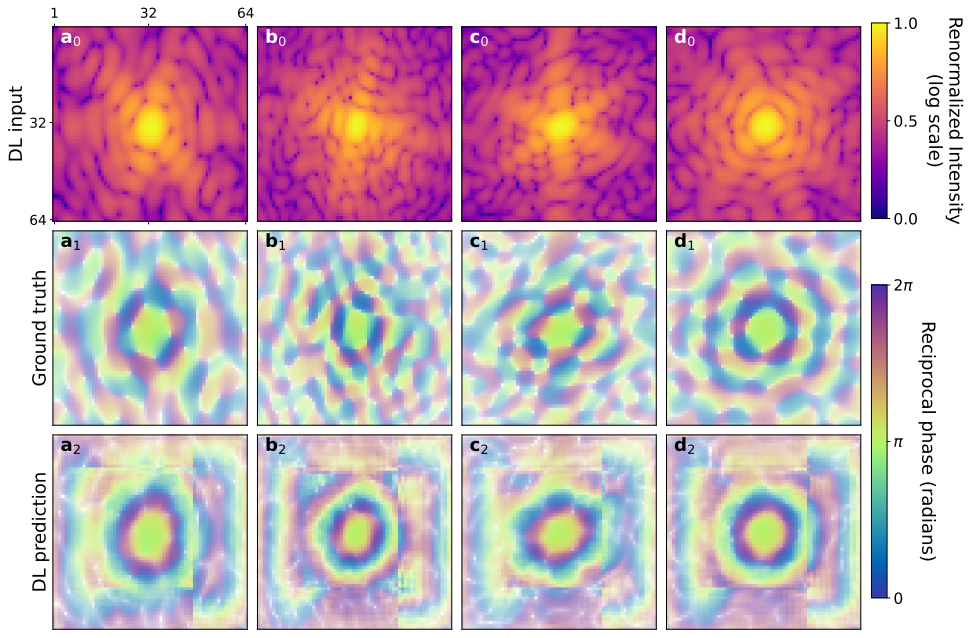


Figure 4.5: Model testing using MSE loss function and biased dataset. First row shows four simulated BCDI patterns, second row the ground truth RSP corresponding to the pattern and last row the DL prediction

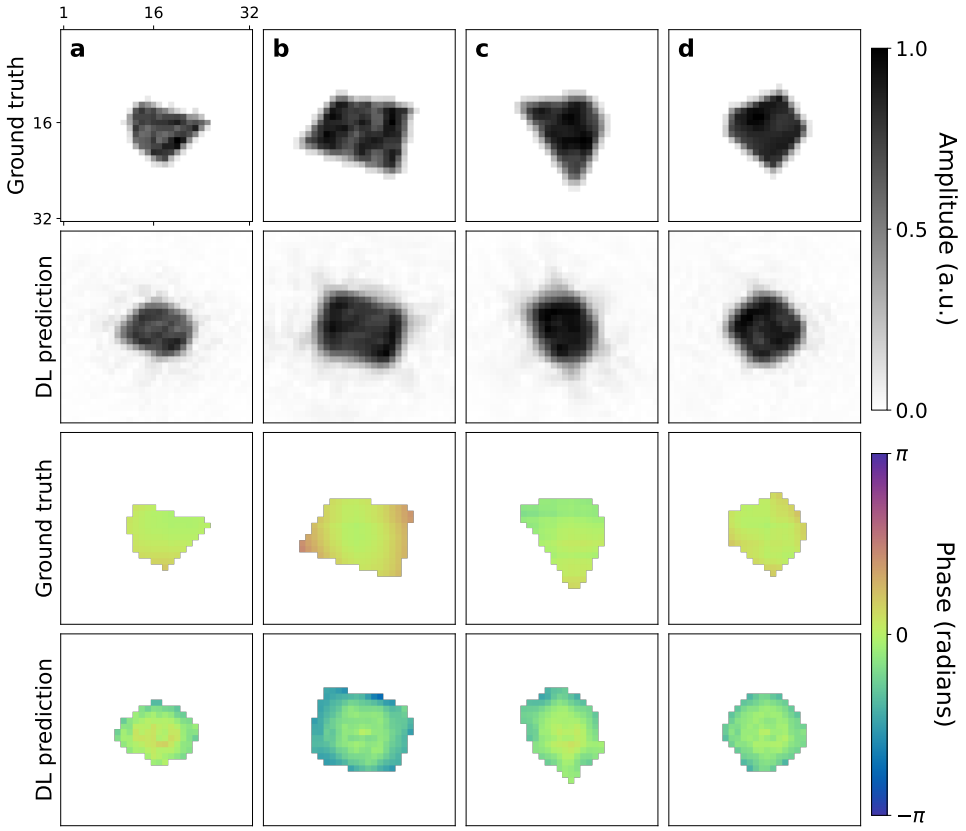


Figure 4.6: Corresponding reconstructed objects. Ground truth and predicted objects' amplitudes (first two rows respectively) and ground truth and predicted objects' phases (first two rows respectively). No significant improvement can be observed after the adopted sing-symmetry breaking procedure.

Unfortunately, the proposed method did not seem to solve the sign ambiguity of the RSP. The model is still unable to discriminate between the plus/minus sign of the RSP and the result is the incorrect overlap of the object with its twin obtained by the inversion symmetry. The phase, though small for this case, is also showing a kind of centro-symmetry as its variations tend to spread radially from the center of the array.

4.4.5 The Weighted Coherent Average loss function

At this point in the study, and in anticipation of applying the model to portions of the RSP, it became necessary to consider a loss function that would operate directly on the phase, without requiring transformations into real space. However, the main challenges were posed by the symmetries inherent to the phase. Upon further reflection, it was concluded that a Mean Squared Error (MSE) is not an appropriate metric for comparing the phases of complex functions. Indeed, MSE fails to account for the 2π periodicity and the possibility of a global phase offset. One could argue that 2π wraps can be fixed with a modulo 2π operation and the offset can be removed by shifting the tensor by a constant. However, the modulo wrapping function jumps abruptly by 2π every time phase crosses an integer multiple of 2π, meaning that the gradients are infinite thus not advised for gradient-based optimizations. Moreover, the MSE (or MAE and other *divergent* metrics) will have problems at the 0-2π boundary. In fact, when considering the phase mapped in the 0-2π range, if we suppose a $\varphi^0_{pred} = -0.1$ where

$\varphi^0_{G.T.} = 0$, the wrap will move the φ^0_{pred} to the value $2\pi - 0.1 = 6.183$ amplifying the error (Δ) from 0.1^2 to 6.183^2 improperly.
In order to bypass these shortcomings a new loss function was designed. Here it follows the reasoning process that leads to the mathematical expression of the loss.
The best way to account for the periodicity and the wrap without discontinuities and error unbalances, is to evaluate the ground truth - predicted phase differences (Δ_k) on the unit circle. To do such, it's necessary to express Δ_k as angles of a complex exponential. This means that if φ_{pred} is an array of random values, each complex number $z_k = e^{(i\Delta_k)}$, when represented on the Argand plane, can be seen as a vector pointing at a random coordinate on the unit circle. Now, the goal of the optimization is not to minimize Δ_k for all k but to have the same Δ_k throughout k. In fact, for $\varphi_{pred} \Leftrightarrow \varphi_{G.T.}$ each vector z_k points in the same direction, but it does not necessarily lie on the x-axis ($\Delta_k = 0$ condition). Therefore, the loss function should ultimately drive all the z_k from randomly distributed to coherently aligned along a common direction. A helpful quantity in this case can be the complex average vector $\langle z \rangle = \sum_{k=1}^{N} z_k = \sum_{k=1}^{N} e^{(i\Delta_k)}$ where k runs over all the N pixels. In particular the length of $\langle z \rangle$, represented by the modulus $|\langle z \rangle|$, is an efficient metric for the measurement of the degree of "coherence" among all the complex phase differences. In fact, $|\langle z \rangle|$ scores 0 for randomly oriented z_k, as opposite contributions cancel out each other because incoherent, while it scores 1 for perfectly aligned ones. It follows that one wants to maximize $|\langle z \rangle|$ during the optimization. Moreover, given the natural normalization between 0 and 1 of this metric, it follows naturally that the loss function can be expressed as $L = 1 - |\langle z \rangle|$.
Additionally, an importance mask can be applied during the averaging process. In particular, we know that the brightest pixels of the BCDI pattern are the ones contributing the most to the object's reconstruction. For this reason one could weigh the complex average multiplying by the input magnitudes. The effect of this operation is to "give a direction" to the optimization, meaning that the $\langle \Delta \rangle$ the model will tend to converge to, will be mostly steered close to the Δ_k of the brightest k pixels. The loss can now be expressed as:

$$L = 1 - \left| \frac{1}{N} \sum_{k=1}^{N} \sqrt{I_k} \exp\left(i(\varphi_{\text{GT},k} - \varphi_{\text{pred},k})\right) \right| \tag{4.3}$$

Where N is the total number of pixels in each RSP array and k is the pixel index. $\sqrt{I}$ is the magnitude of the BCDI pattern normalized between 0 and 1 with respect to the sum, and φ_{GT} and φ_{pred} the ground truth and predicted RSP.
The last missing piece is the removal of sign symmetry. Rather than biasing the dataset preferring one sign over the opposite, the function L is computed for both φ_{GT} and $-\varphi_{\text{GT}}$ and in a second passage, the minimum of the two along the batch dimension is kept for backpropagation. The final form of the Weighted Coherent Average (WCA) loss is then given by:

$$L_{\text{WCA}} = \min\left(L_+, L_-\right) \tag{4.4}$$

To better visualize the functioning of the WCA loss function, a simple model has been trained to fit the ground truth phase of a single 2D BCDI pattern using the WCA. The complex phase differences vectors were extracted at each step of the optimization together with the updates obtained from the gradients of the WCA with respect to the trainable parameters. Fig. 4.7 shows the evolution of the predicted RSP as well as the progressive alignment of the z_k.

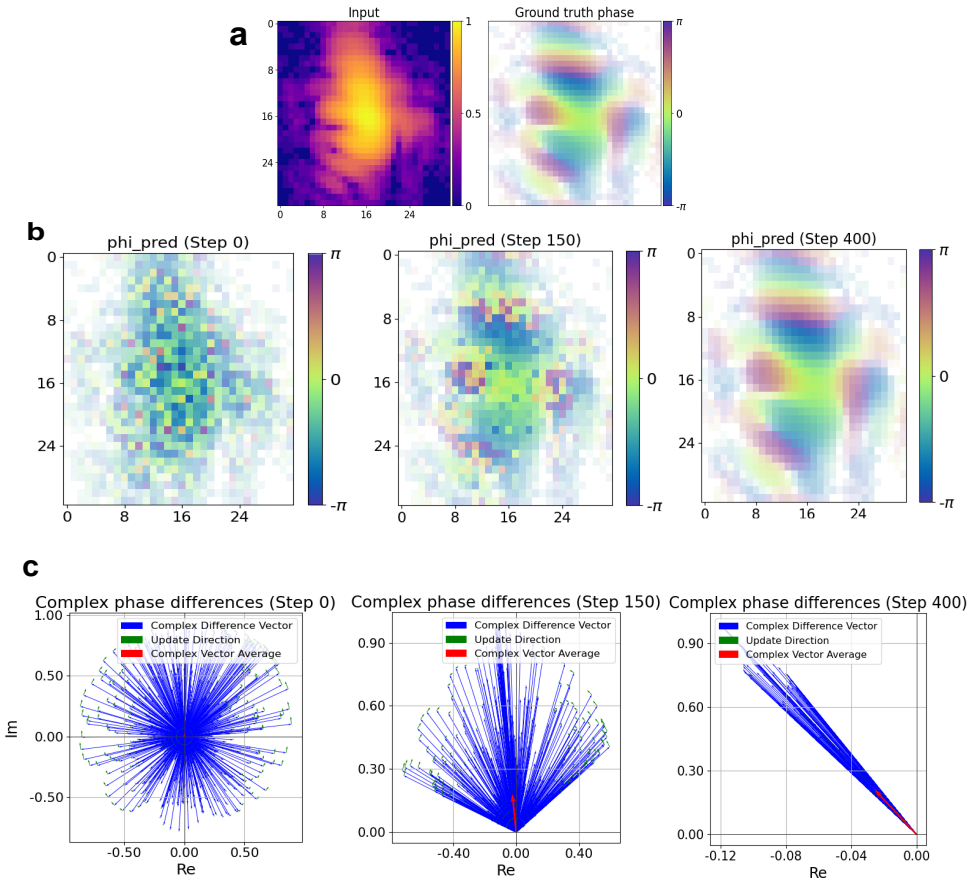


Figure 4.7: Illustration of the WCA loss function. **a** Input intensity (log-scale normalized) and ground truth RSP. **b** Predicted RSP in steps 0 - 150 - 400 of the optimization. **c** Corresponding complex phase-differences vectors z_k on the Argand plane (blue arrows), together with the updates (green arrows) obtained from the gradients of the WCA, and the resultant complex average $\langle z \rangle$ (red arrow). It is visible that during the fit, as the z_k align around a common one, the amplitude of $\langle z \rangle$ grows bigger and the predicted RSP converges to the ground truth one.

The same model has been trained using the WCA for the same number of epochs on the same dataset and here the results are shown. First, it can be noticed in Fig.4.8 that the training and validation loss values throughout the training are following different trends with respect to the model trained with the MSE loss (Fig. 4.2)

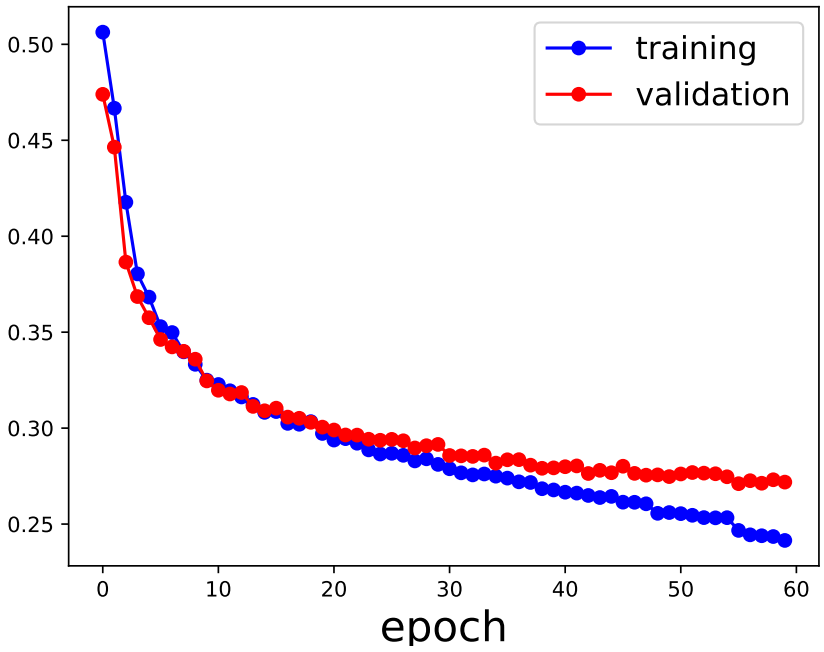


Figure 4.8: Training and validation loss curves over 60 epochs using the WCA loss function.

In this case the correct learning curve does not reach a plateau within the first 25 epochs but maintains a negative slope for longer, indicating a better learning. This suggests indeed better results when used on test data. In particular, for the same input diffraction patterns tested above in Figs.4.5 - 4.6 the model trained with the WCA yields the prediction shown in Fig.4.9 for the RSP and Fig.4.10 for the corresponding reconstructed objects.

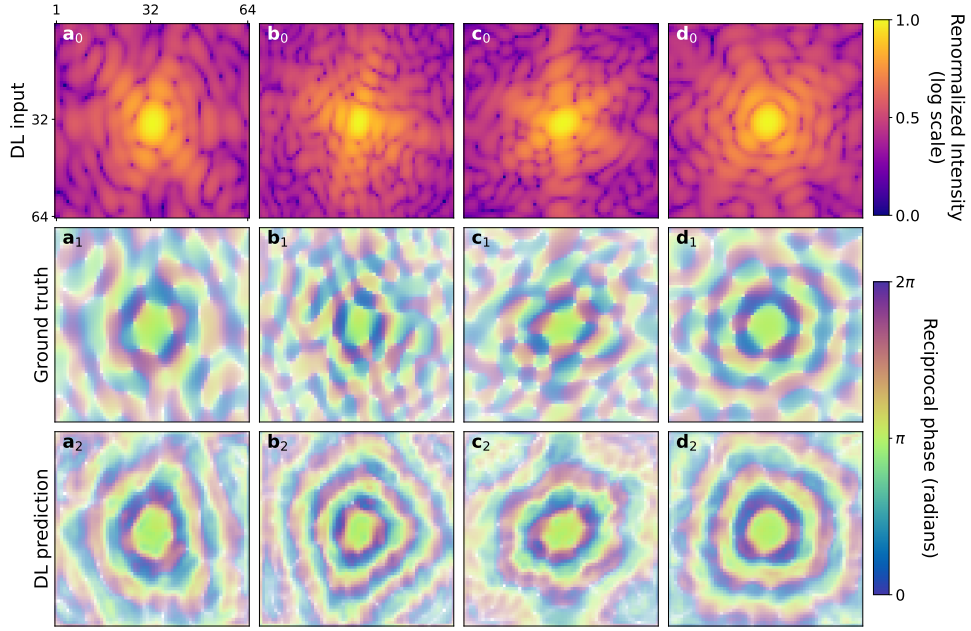


Figure 4.9: Model testing using WCA loss function. First row shows four simulated BCDI patterns, second row the ground truth RSP corresponding to the pattern and last row the DL prediction

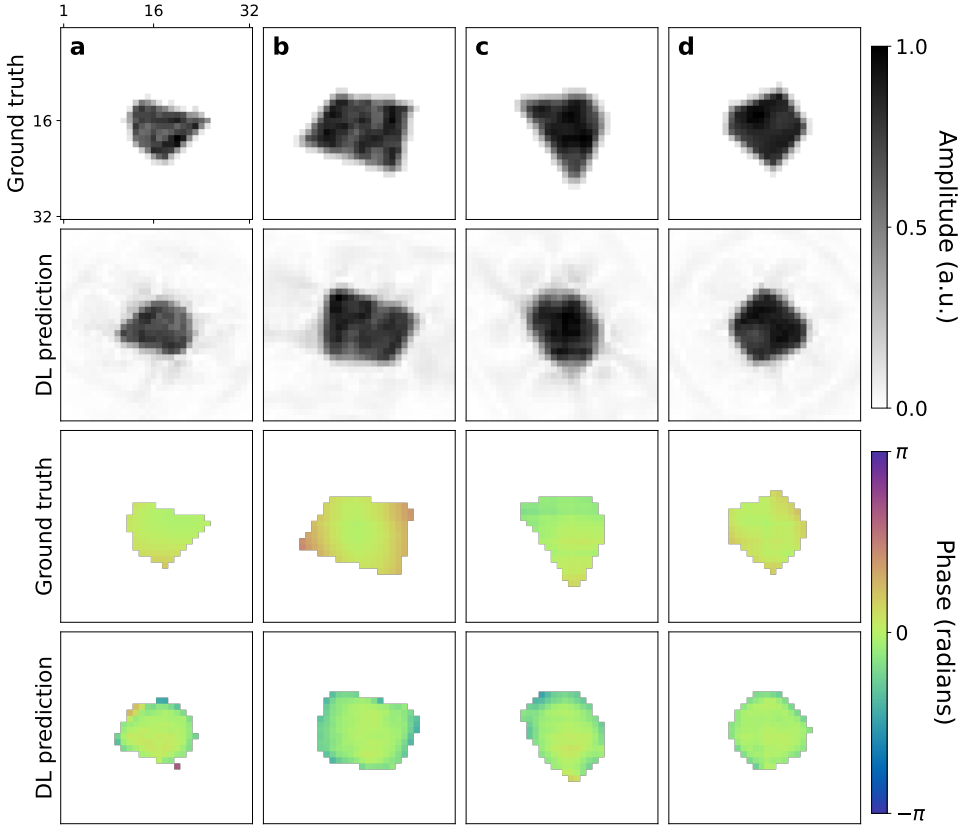


Figure 4.10: Corresponding reconstructed objects. Ground truth and predicted objects' amplitudes (first two rows respectively) and ground truth and predicted objects' phases (first two rows respectively).

The results obtained from the model trained with the WCA loss function are visually better than the MSE ones. Although not completely removed, the sign symmetry that gives rise to the superposition of the object with its twin, is less pronounced. For example, particles in Fig. 4.10(a-b-d) have a clear orientation and a shape that matches the ground truth. In all those cases though, the model has opted for the conjugate solution as the predicted object are flipped with respect to the ground truth ones. In Fig. 4.10(c) instead the symmetry is not broken and the result is still a superposition of the particle with its twin. This suggests that the symmetry breaking method implemented in the WCA, and the one proposed by Zhang and coauthors, is only partially playing a role in the actual model learning. It is interesting to notice indeed that when the training dataset or the model trainable parameters are increased, the sign symmetry is completely removed in the most difficult cases as well. Fig. 4.11 shows the effect of the dataset and models sizes for both MSE and WCA loss functions on the same simulated test data. The first important piece of information this figure shows is that the model trained with the WCA reaches higher accuracy. Moreover, it is much faster to compute since no FFT or IFFT is involved (see Fig. 4.11), thus the training time is drastically reduced. For what concerns the accuracy metric, in order to properly account for both modulus and phase, it has been calculated using

$$\left(\frac{PCC(m_{pred}, m_{GT}) + WCA(m_{pred}, \varphi_{pred}, \varphi_{GT})}{2}\right) \times 100 \tag{4.5}$$

where $PCC(m_{pred}, m_{GT})$ is the Pearson Correlation Coefficient on the object's moduli and $WCA(m_{pred}, \varphi_{pred}, \varphi_{GT})$ is the WCA function applied to the object's phase weighted by the

normalized predicted modulus. For what concerns the sign symmetry problem it is evident that while for the MSE trained model it is resolved only for a larger number of trainable parameters, for the WCA trained one it is already sufficiently overcome. As last observation, it is interesting to notice that when the model size is kept fixed and the training dataset augmented, the WCA improves the performances while for the MSE it is not the case.

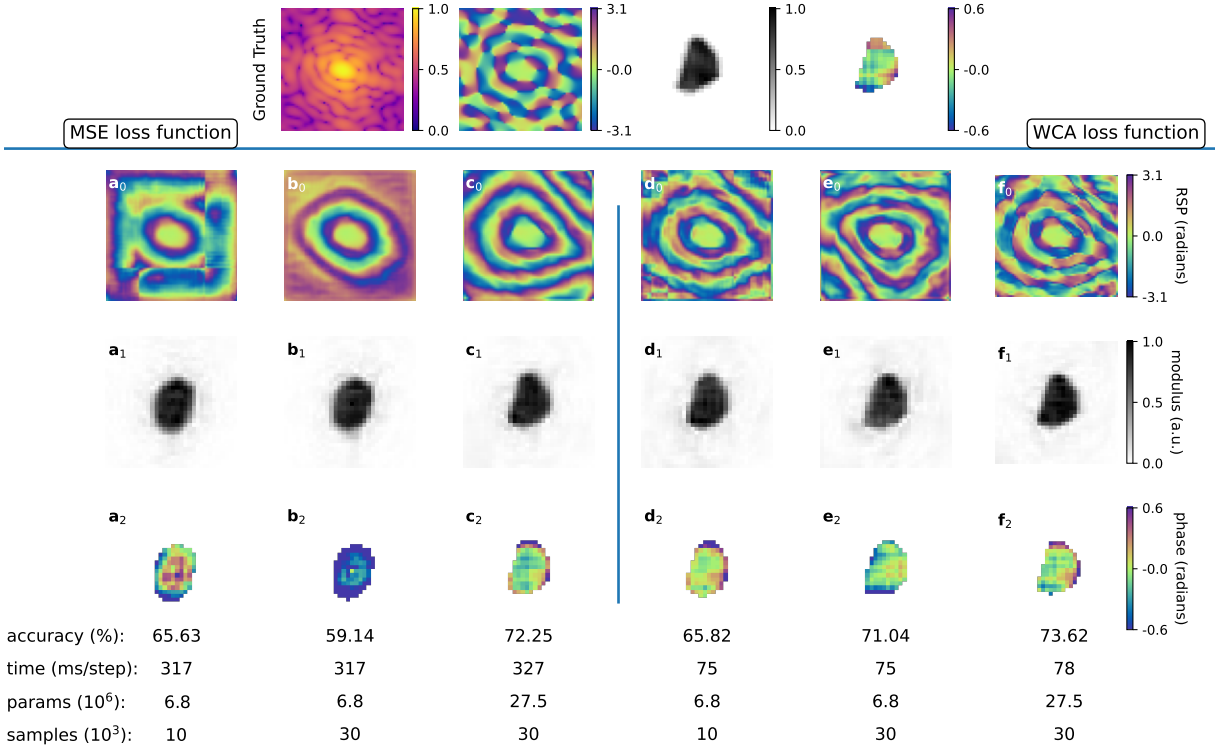


Figure 4.11: Comparison of MSE and WCA loss function for different model and training dataset sizes In the first row from left to right the input intensity, the ground truth RSP and the corresponding object (modulus and phase) are represented. a_0, b_0, c_0 are the results of the predicted RSP obtained from the model trained with the MSE loss function with the initial number of parameters and training set (**a**), with the augmented dataset (**b**) and with both model and dataset size increased (**c**) In third and fourth rows the corresponding reconstructed objects are displayed. d, e, f columns symmetrically shows the results obtained with the model trained using the WCA loss.

4.5 2D high strain case

In this paragraph the model training was performed on a dataset of highly strained 2D BCDI pattern simulated as described above in section 3.3.1. In this case Poisson statistic was applied to each dataset to better simulate the experimental condition. A set containing 30'000 samples was created and the "bigger" model of 27.5M parameters mentioned in Fig 4.11 was trained over 50 epochs with a learning rate of 0.001. Similarly to the low-strain case described above the same model has been trained with the MSE and WCA losses separately for comparison. The goal of this study is in fact to test the relevance of the loss function compared to the size of the model when the complexity of the task increases. The results on 4 test BCDI pattern are shown in Figs. 4.12-4.13 for the MSE one and 4.15-4.15 for the WCA.

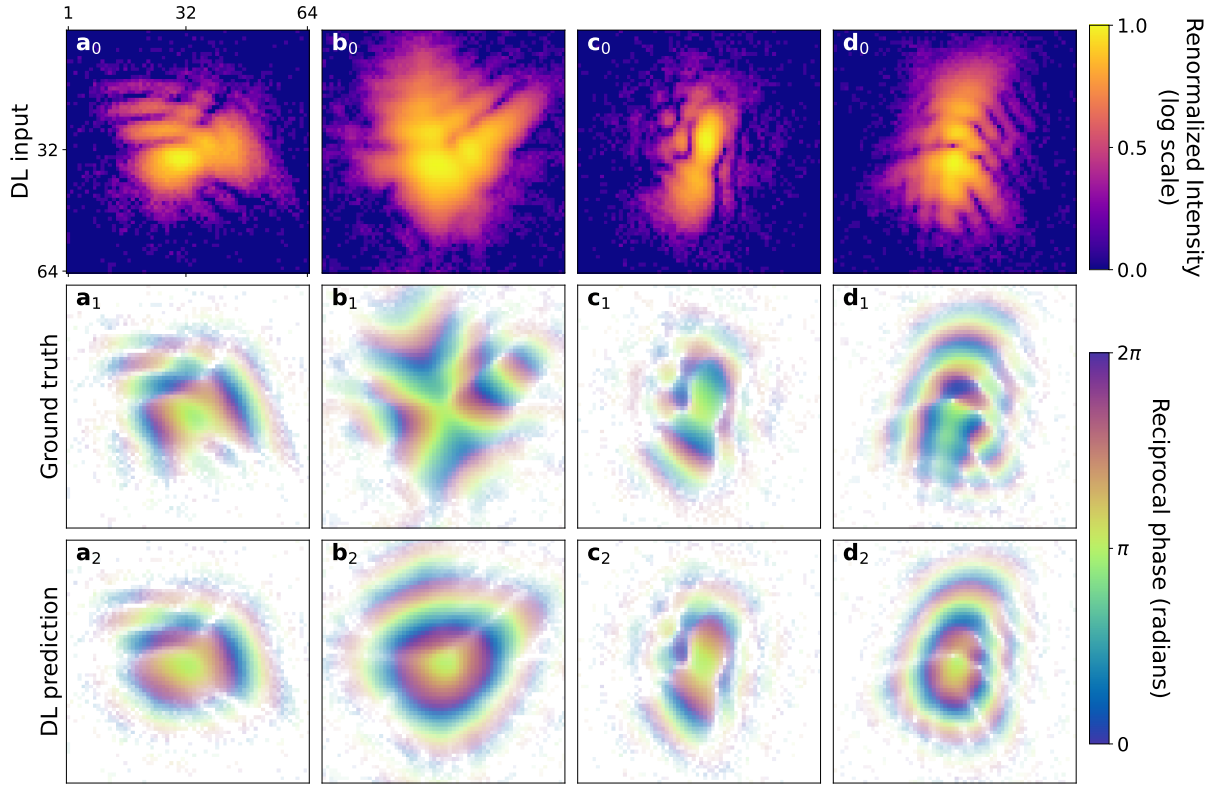


Figure 4.12: RSP predicted by the model trained with the MSE loss function calculated on both modulus and phase of the reconstructed object. First row: simulated 2D strained BCDI patterns (test dataset). Second row: corresponding ground truth RSP. Third row: predicted RSP. Once can notice that the model cannot predict correctly the RSP where the "iso-phases" do not have a circular symmetry (see **b-d**).

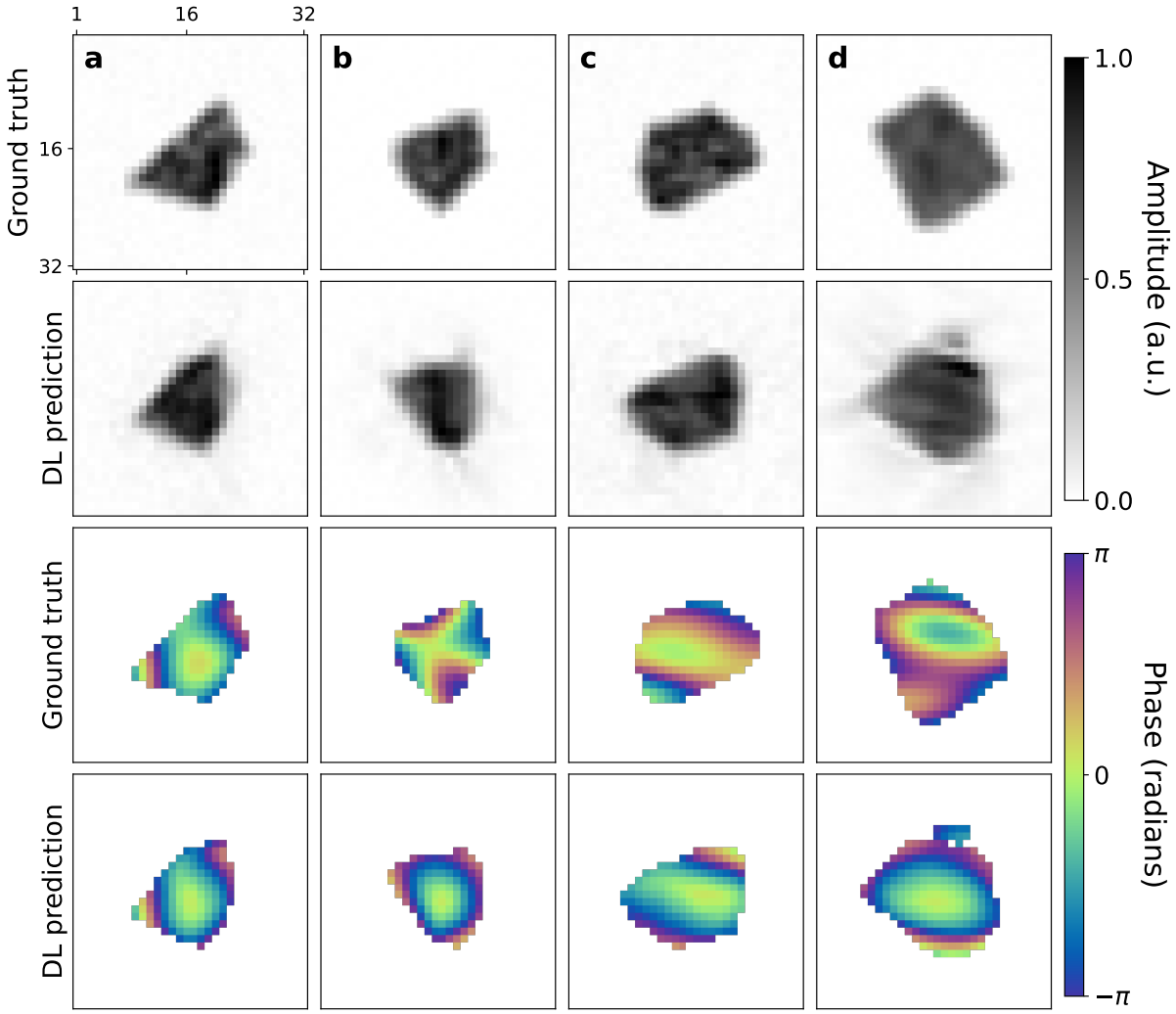


Figure 4.13: Corresponding reconstructed objects. First and third row: ground truth modulus and phase. Second and fourth row: model's results of objects' modulus and phase. Although the shape might at first sight look like the ground truth one (or the twin) the phase is often incorrect. It is curious to notice that better results are obtained when the object's phase possesses a certain degree of symmetry with respect to the center, analogously to the corresponding RSP.

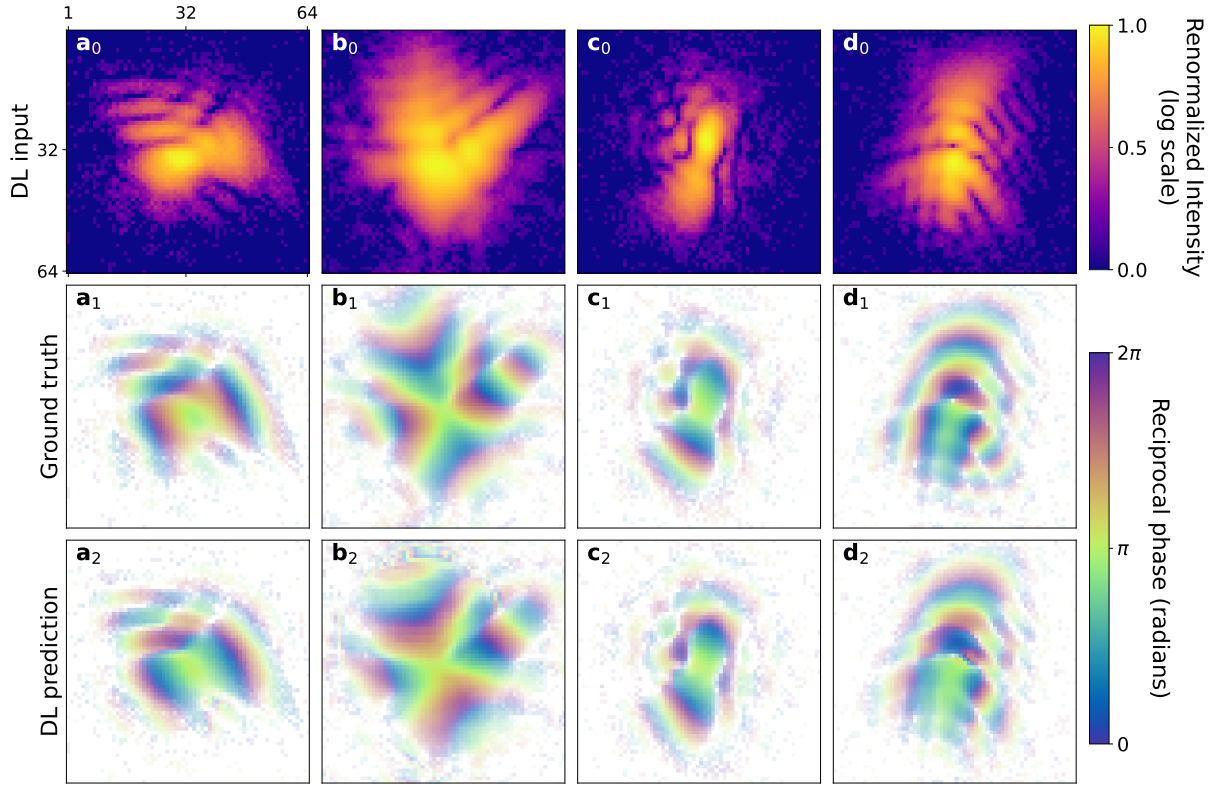


Figure 4.14: RSP predicted by the model trained with the WCA loss function. Here the model correctly retrieves the RSP for non-circularly symmetric structures as well (**b-d**)

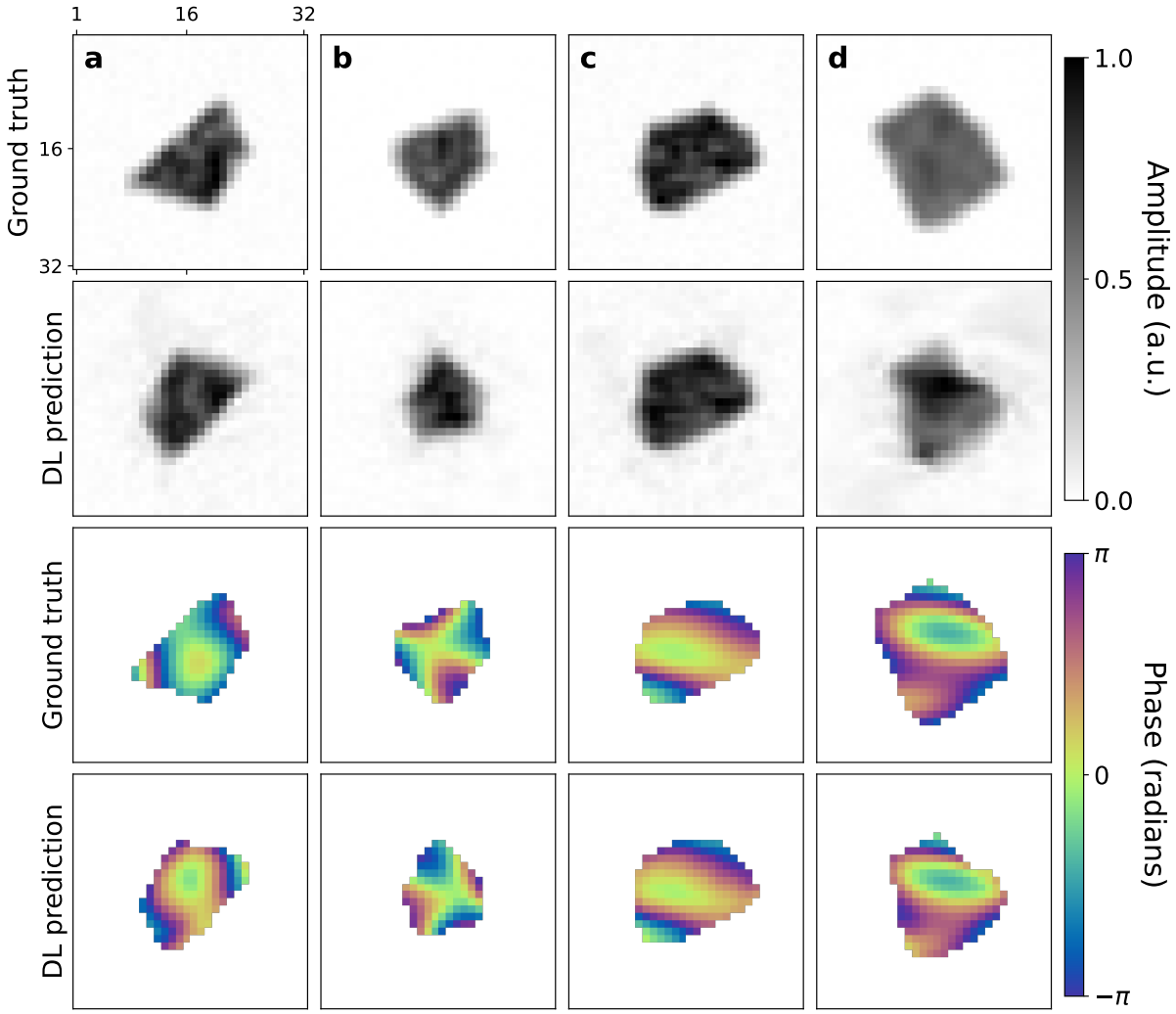


Figure 4.15: Corresponding reconstructed objects. Compared to Fig.4.13 the phase of the object is correctly recovered.

The preliminary studies on the 2D case for low-strain particles have demonstrated the possibility to recover the RSP from the diffracted intensity pattern with a U-Net like architecture without ever calculating the object in real space. Moreover, the studies on the high strain particles has shown that this model configuration is well suited for this case as well. From these promising results, it was decided to investigate the mapping intensity-RSP for portions of the reciprocal space.

4.6 Phasing patches: 3D case low strain

In this section of the manuscript the DL prediction of "patches" of the RSP will be explored and discussed. Three-dimensional BCDI pattern of low strained particles were used to conduct this study. Although this patching approach has not given satisfactory results for the PR, it is nevertheless reported in the manuscript as study on the *local* rather than *global* relationship between the diffracted intensity and the RSP. It is indeed known that there exist a unique mapping, barring some trivial RSP symmetries, between the diffracted intensity and the RSP in

3D [48]. What is interesting to investigate is whether this relationship exists also for subsets of the reciprocal space, and in particular if it can be retrieved by a DL model. (From now on the term "patches" will be used to refer to cubic subsets of the reciprocal space).

When deciding to work with patches, there is a number of questions that arise and the answer to which is not straightforward nor unique in many cases. Namely: What size is best? Can the patches be extracted at random positions or should there be an order? What about the normalization of the intensity range inside the patch? How are the patches stitched together into the full RSP eventually? How are the phase symmetries taken into account during the stitching? Here I will present the approach that allowed me to address these questions.

4.6.1 The choice of the size

Similarly to the inpainting case, 32 pixel-side cubic patches, cropped out of 128 pixel-side simulated BCDI patterns were considered. The choice is supported by the following reasons:

- The good results obtained for the inpainting case suggested that the amount of information contained inside a 32 pixel-side patch of reciprocal space is enough for the model to grasp spatial correlations.
- The average oversampling ratio of BCDI experimental data is such that in a 32 pixel-side volume a sufficient amount of fringes is contained, meaning intuitively that the model can predict the corresponding RSP.
- An even number multiple of 2 is usually considered GPU-friendly since it facilitates the shared calculations across different threads.

4.6.2 Patches division and stitching

At first, the patches were thought to be extracted randomly from the full BCDI pattern as for the inpainting case. However, by doing so the RSP of each patch would in principle have different offsets and different wraps than the neighbors and this would complicate the stitching of the patches back into the full RSP. For this reason, and considering the approximate spherical symmetry of the average BCDI pattern it was decided to crop patches radially, starting from the region around the center of the Bragg peak and the progressively moving outwards to higher q-values. In this configuration, an integer step (10 pixels in our case) was chosen beforehand and the first patch around the center of the Bragg peak was selected together with all the patches centered in distances of integer multiples of the chosen step. Fig. 4.16 shows a simplified schematic of the patches extraction. For the DL model training the patches of the intensity pattern need to be selected as well as the for the corresponding RSP for ground truth comparison.

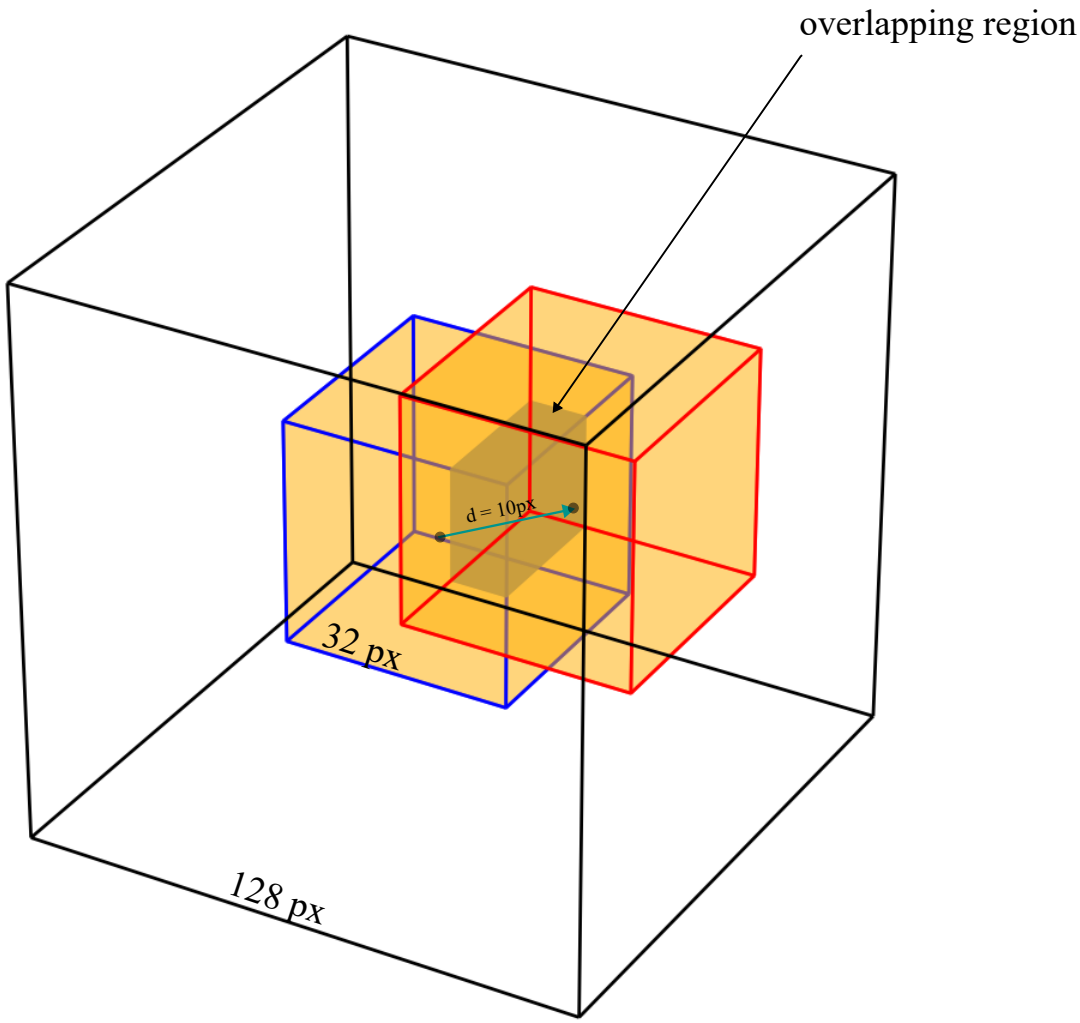


Figure 4.16: Schematic of the cropping of patches. From the full BCDI pattern (white 128 px-sided cube) the first patch is cropped out in the center and (orange 32 px-side cube with blue outline). Other patches are extracted radially from concentric shells separated by a 10 pixels step. Here only one patch from the first shell at distance 10 pixels from the center of the peak is displayed for simplicity. The gray shaded area highlights the overlapping volume between the two patches.

Being the step size smaller than the semi-diagonal of the 32 pixel-sided patches, it follows that the patches of adjacent cells have overlapping volumes. These common regions can have a twofold purpose. Firstly, they reduce the complexity of the stitching procedure since when this is executed progressively starting from the central patch, the sign and the offsets of the RSP are unambiguously fixed for all the following ones. Secondly, during the DL model training, for patches belonging to the outer shells, the overlapping volume of RSP belonging to the innermost adjacent shell can be provided as initial guess along with the input intensity patches. This of course cannot be exploited for the central patch that necessarily has to be predicted without initial RSP guesses.
The last question to be answered concerns the normalization. Since each patch is processed independently of the others by the DL model, it was decided to normalize each patch between

0 and 1 (always in log scale).

To summarize, the final design implied the use of two distinct training datasets and two different CNNs. The first dataset was dedicated to the central portion, therefore the first CNN was provided with 3D intensity patches in input (normalized log scale) and corresponding RSP patches as ground truth labels. A second training dataset containing patches from outer shells (5 concentric ones for a 128 pixel-sided full BCDI pattern) was created. Here each file was made of the pair intensity-RSP initial guess - from the closest neighbor patch belonging to the innermost shell - as input, and the full RSP ground truth patch corresponding to the input intensity. This second dataset was used to train a second CNN identical to the first one. One observation regarding the datasets is that there is an intrinsic imbalance between the number of central patches and the outer ones. In fact, for a single full BCDI pattern, the number of patches in the first shell is 1, while the number of outer portions can go up to several hundreds. Moreover, the central patch is the most important one as it contains a low resolution representation of the particle in real space. In order to balance the training, the first dataset was augmented with more simulated data.

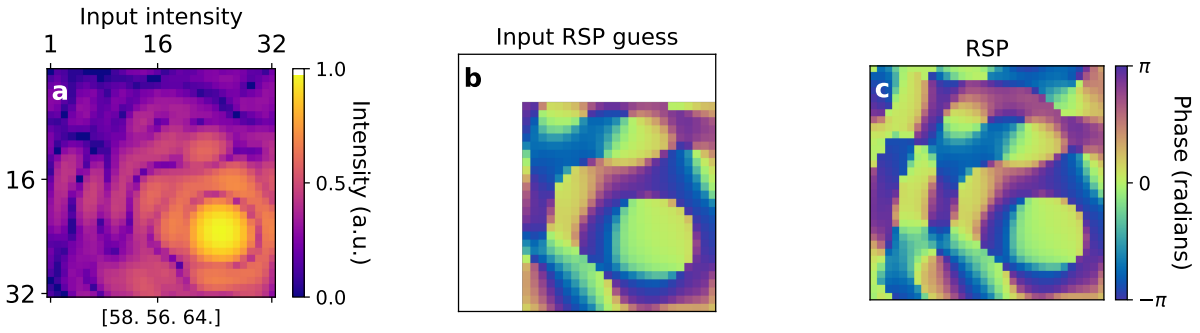


Figure 4.17: Example of input-ground truth pairs for the outer patches model. **a** Central slice of the input intensity for the patch cropped from the first shell at position [58,56,64] (the central patch is at [64,64,64]). **b** Initial RSP guess deriving from the overlapping region of the intensity patch with the central one. The blank area represents the part that needs to be predicted. **c** Ground truth RSP corresponding to the intensity patch in **a**.

4.6.3 Model architecture

The model architecture is similar to the one used for the inpainting case, with a U-Net like structure. 5 encoder blocks reduce the feature map to 512 one-dimensional vectors in the bottleneck and 5 decoder blocks upsample the feature map back to the original size. Skip connections are used as well and similarly to the 2D case for Phase Retreival, the last layer has been left with no activation function. The total number of trainable parameters is of the order of 3.5 millions.
For the second CNN the layers and features are identical to the first one but three modifications were made. In particular, (i) the input tensor was composed of the intensity patch concatenated with the initial RSP guess and a binary mask marking the RSP guess voxels from the others. (ii) the mask was used at the exit of the decoder as well to select the new predicted voxels only for the backpropagation. (iii) The WCA loss function has been restricted only to the positive sign of the RSP since no sign ambiguity is left when fixing an initial guess.

4.6.4 Results on patches

The first model has been trained over 50 epochs on 10'000 samples of central $32 \times 32 \times 32$ pixel-size patches cropped out of $128 \times 128 \times 128$ pixel-size full BCDI simulated pattern using the WCA loss function. Fig.4.18 shows good learning curve which is supported by the results on test data, illustrated in Figs. 4.19 - 4.20

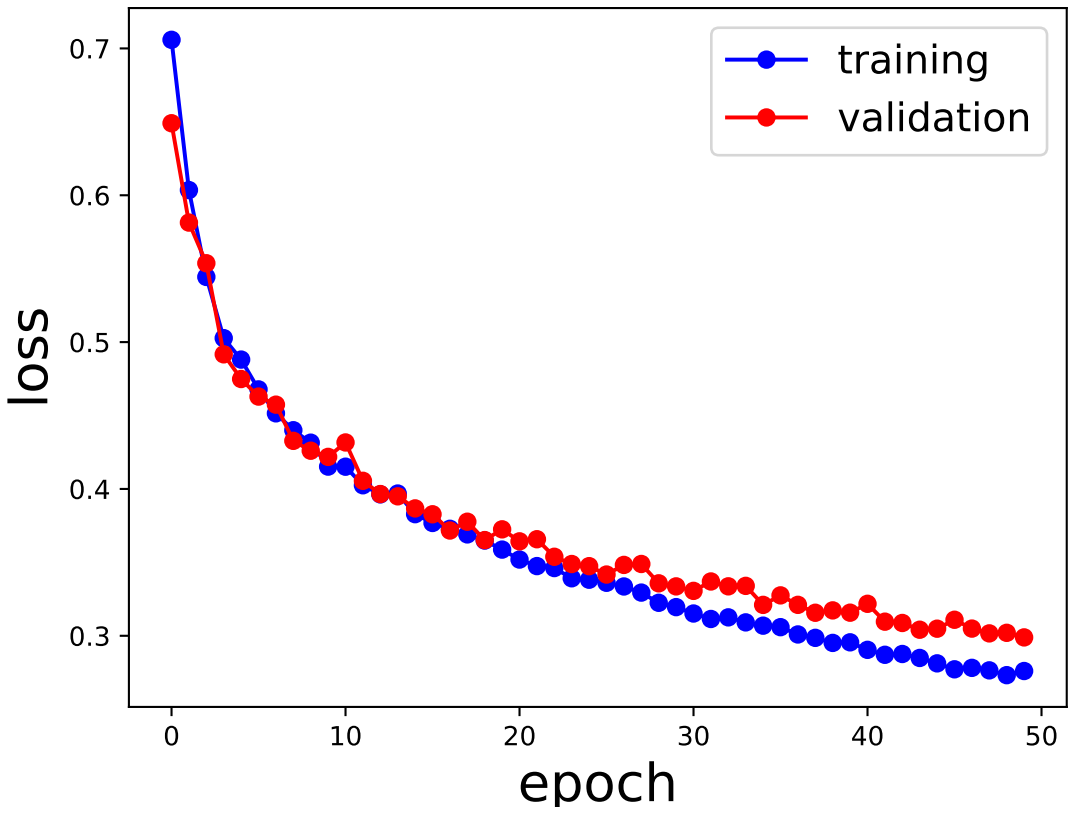


Figure 4.18: Training and validation loss curves over the 50 epochs long training. The plots indicate a good learning of the model because both training and validation losses are decreasing monotonically with the same pace.

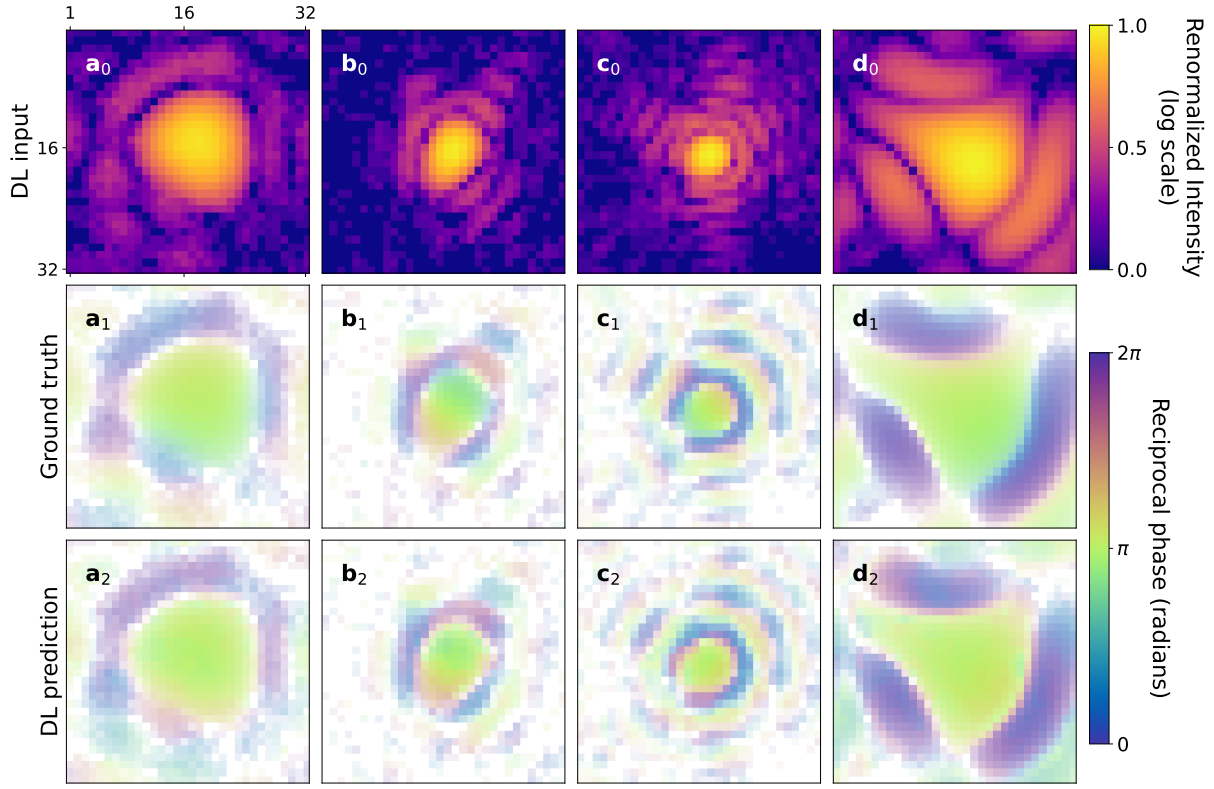


Figure 4.19: Slices of the central patches. First row shows four different examples of BCDI patterns cropped around the center of the peak. The small oversampling ratio of $\mathbf{b}_0$ and $\mathbf{b}_1$ makes such that the central portion already contains the full useful signal and shows the diversity of the dataset. Row from $\mathbf{a}_1$ to $\mathbf{d}_1$ shows the corresponding ground truth RSP while the last row shows the RSP predicted by the DL model.

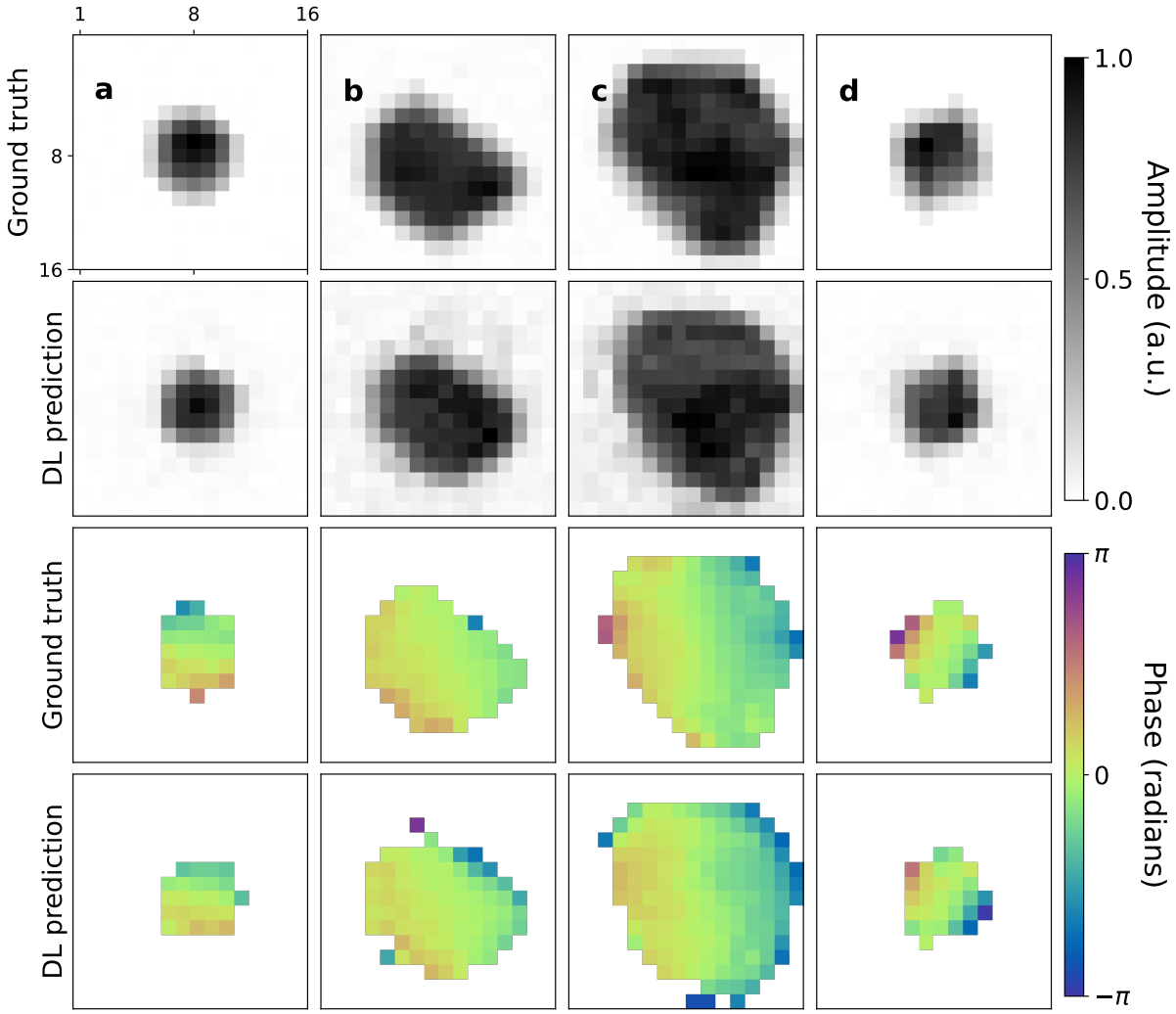


Figure 4.20: Corresponding objects. First and third rows show the modulus and the phase of the ground truth objects while second and fourth rows show the predicted ones. Although the low resolution due to the limited reciprocal space window, the model proves to correctly find the shape and phase distribution of the particle.

From the results obtained after the training of the model dedicated to the central portion of the simulated BCDI patterns, one can conclude that the model is capable of retrieving the correct RSP for the low strain case, meaning that the leap from the 2D case to the 3D case does not imply unforeseen complications. Moreover, given the diversity of the training dataset the model manages successfully for full peaks contained in a small patch.

The second CNN was trained on patches extracted from outer shells over 50 epochs on a dataset containing 50'000 samples. Fig. 4.21 illustrates some relevant results. In particular one can observe that the model can predict the RSP in the missing regions providing a relatively smooth transition between the "known" and "unknown" parts. This result is particularly interesting as it proves that a CNN trained with the WCA loss function learn the map that links a portion of diffracted signal with the corresponding RSP with no information on the particle in real space nor the position of this portion with respect to the center of the diffraction peak. It also shows that the size of the patches is contains a sufficient amount of information for this

map to be learned. However, one can also notice that some "noise" is present in the predicted regions. These discrepancies from the ground truth become detrimental during the stitching procedure.

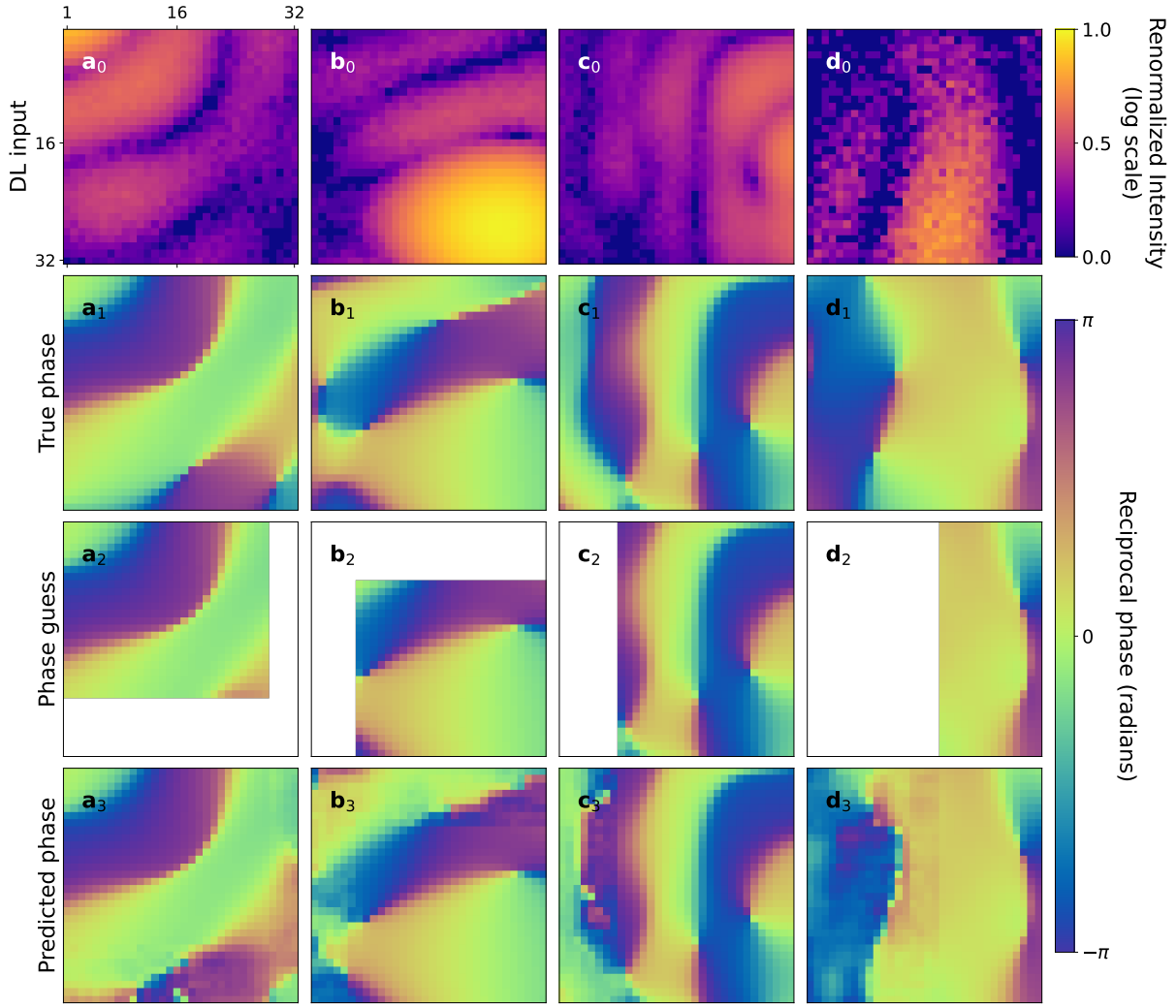


Figure 4.21: Examples of RSP prediction for outer patches. The first row shows the central slice of some patches extracted far from the central part of the BCDI peak. The second row shows the ground truth RSP, the third row the RSP initial guess obtained from the overlap with the nearest neighbor of the innermost shell, and the last row shows the DL output. The DL prediction is limited to the missing region.

4.6.5 Results on the full RSP

Here the results of the combination of the two CNNs are presented for the low-strain case. Once completed the training the model was tested on full simulated and experimental BCDI pattern. In order to properly retrieve the full RSP corresponding to the diffracted intensity a stitching algorithm for the predicted patches was designed. The stitching takes place progressively starting from the central patch and updating the full RSP array shell by shell. Once the prediction of the central patch RSP is obtained from the first CNN, the patches of intensity belonging to the first shell at distance 10 pixels are extracted and the overlapping regions between each

patch with the central one are calculated. The predicted central patch RSP for each overlapping region is therefore located in the initial guess RSP array that is given to the model as input paired with the intensity patch. Subsequently, the full batch of pairs $(I, \varphi_{guess})_{shell=1}$ is sent through the second CNN and the o corresponding RSP output is obtained. At this stage two main issues have to be considered, namely: (i) during the training of the second CNN the initial guess RSP was taken from simulated *ground truth* RSPs while here it is taken from the model's previous prediction itself. It is for this reason that any small unavoidable discrepancies between the predicted and ground truth RSP can lead the model to further errors. (ii) When a round of RSP is predicted there are overlapping regions between patches of the same shell and patches of the previous batch. The most straightforward way to perform the stitching of the patches into the full RSP is to overwrite each time the results. However, although a better approach based on the average of the overlapping prediction was implemented, the issue did not seem to be fully resolved. A first test was conducted with this simpler solutions for both issues, and a more robust approach is briefly discussed at the end of the section.

The crop-predict-stitch method is hence repeated until the last shell. The number of shells and predictions scales with the size of the dataset, and it is however always restricted to a spherical region around the Bragg peak. It is therefore less accurate for non-cubic data. Here, Fig.4.22 shows an example of full RSP stitching for a cubic 128 pixel-sided simulated BCDI pattern, while Fig.4.23 reports the analogous result for an experimental data phased with PyNX.

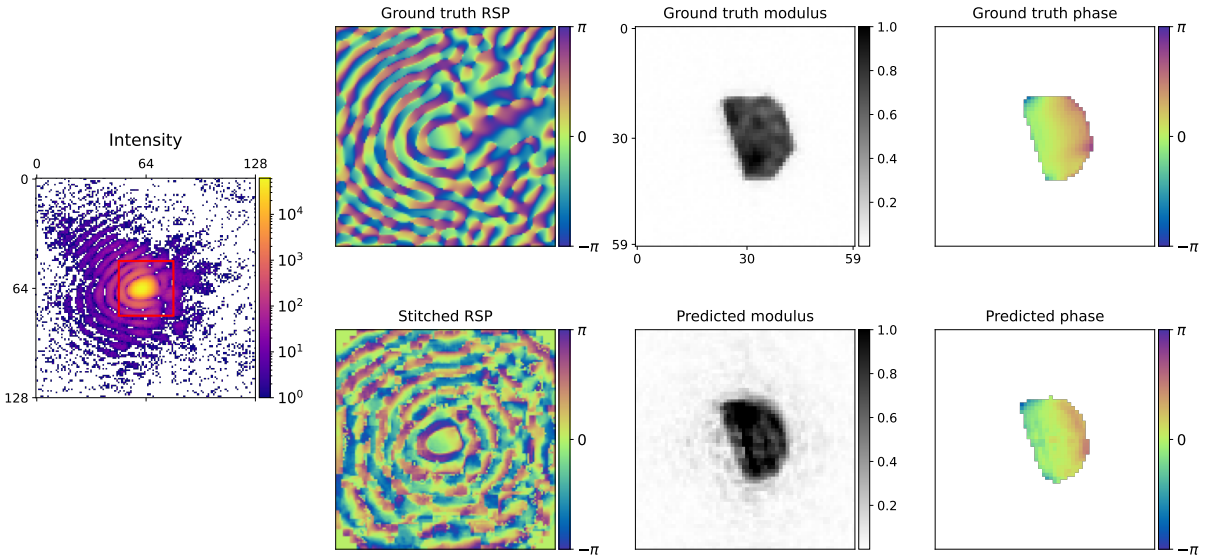


Figure 4.22: Results of the stitching of RSP predicted patches for 3D simulated data (central slice displayed). The red square on the intensity figure represents the central patch.

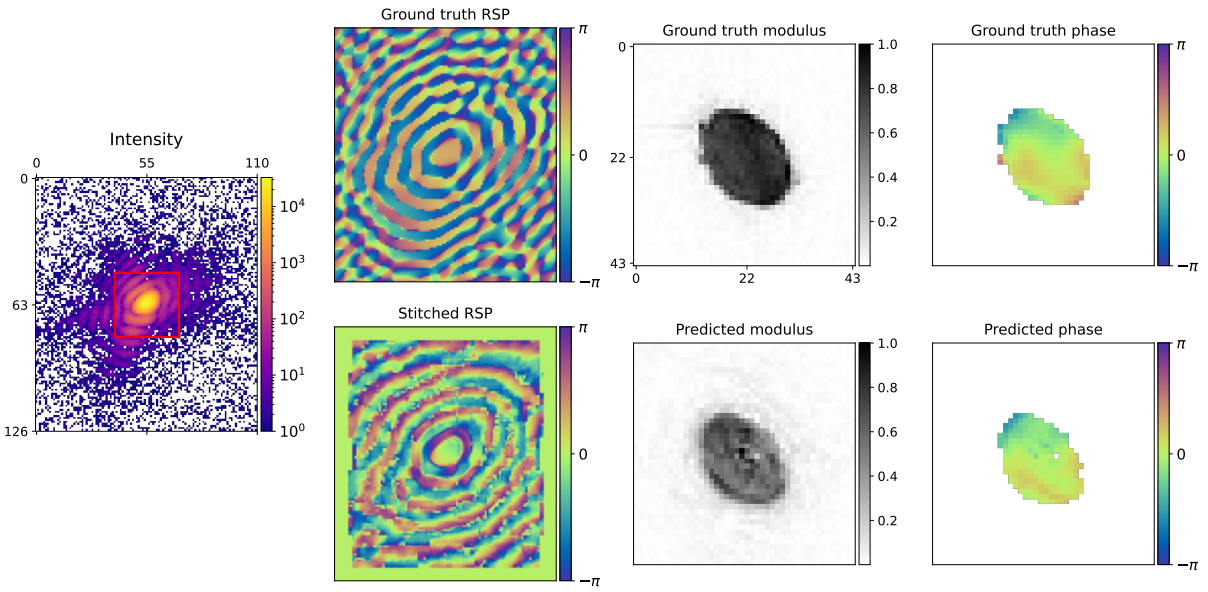


Figure 4.23: Results of the stitching of RSP predicted patches for 3D experimental data (central slice displayed).

What emerges from the above results can be summarized as follows:
(i) the accurate prediction of the central RSP patch is fundamental for retrieving the low resolution estimate of the object
(ii) the stitching is problematic already from the first shell, most likely because of the two issues pointed out above (initial guess from model prediction and RSP averaging of overlapping regions)
(iii) the outer RSP patches correctly infer the oversampling ratio as the "thickness" of the RSP oscillations matches the one of the diffracted intensity, nonetheless, the overall orientation seems to prefer a circular..
(iv) from the reconstructed object's modulus a non-physical more intense spot in the center of the array had to be filtered out. The occurrence of this spike indicates the presence of wrong zero frequency components in reciprocal space, hence wrong zero RSP values. Fig. 4.24 shows the object's modulus before the removal of the high intensity spike located in the center.

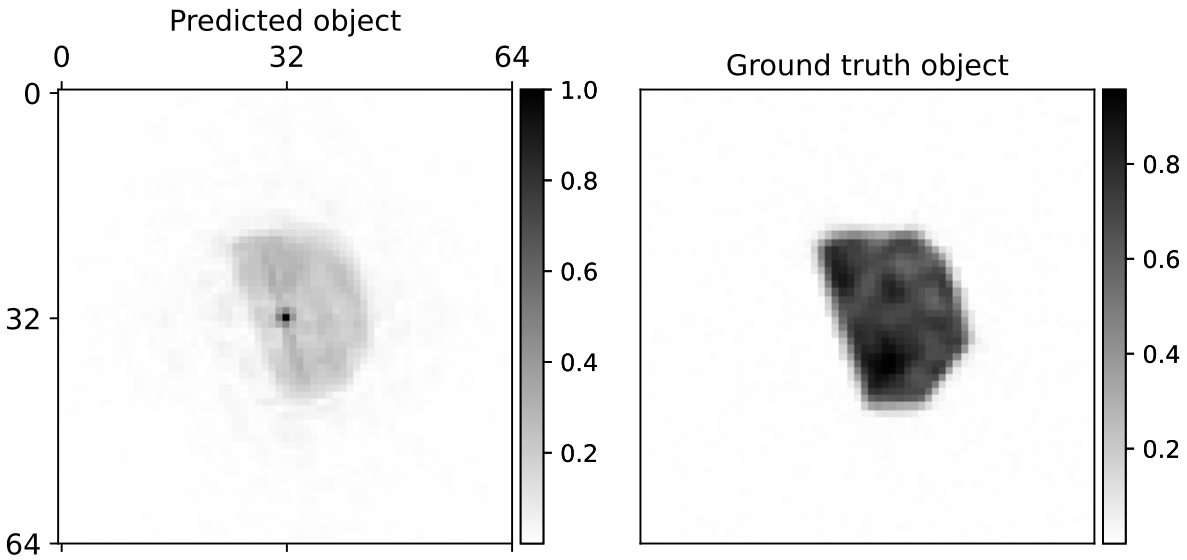


Figure 4.24: Predicted object's modulus before the filtering of the zero-frequency component.

4.7 Patches: 3D case high strain

For completeness the same model has been trained on a dataset containing highly strained simulated diffraction patterns. Each BCDI pattern has been simulated following the procedure explained above in Sec. 3.4.1 and 4.6.2 for a total amount of 50'000 for both central and outer patches. The two CNN have been trained for 50 epochs each and then tested on new simulated data.

For what concerns the firs CNN trained on central patches it was possible to deduce a more difficult learning from the loss curves. While in Fig.4.18 the validation loss reaches 0.3 at the end of the training, for Fig.4.25 it only drops to 0.6 with a significant divergence between training and validation starting from the 20th epoch. A higher loss value is nevertheless expected because of the more complex intensity-RSP mapping for the high strain cases, while the increasing gap between training and validation loss is a signature of poor generalization, early symptom of overfitting.

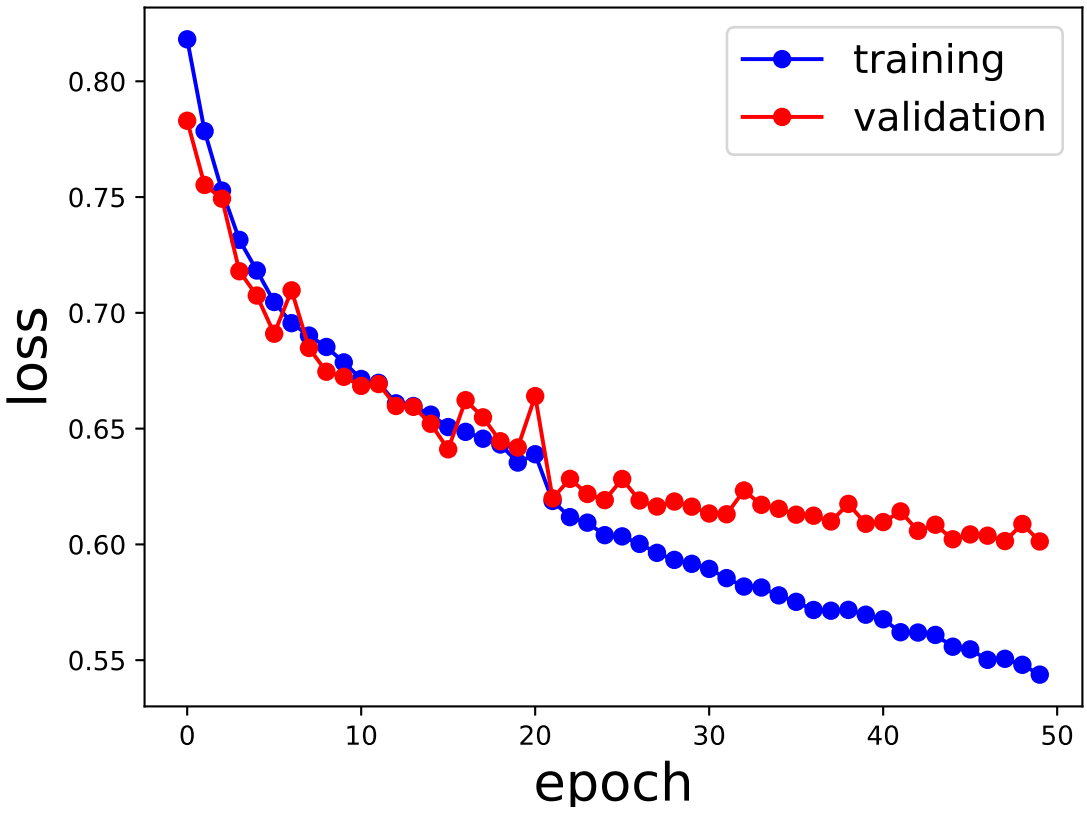


Figure 4.25: Training and validation loss trends for the training of the first CNN on central patches of highly strained BCDI patterns. Higher loss values can be observed when comparing with the low strain case (Fig. 4.18) and a beginning of overfitting from the 20th epoch is also visible.

The results on test data show in fact that the model not always manages to correctly predict the RSP, especially when the iso-phase regions are not spherically symmetrical (Fig. 4.26c). It however succeeds to retrieve the correct RSP oscillations (Fig. 4.26a) inside the central fringes elongated by the high strain. The reconstructions in real space, shown in Fig. 4.27, confirm satisfactory result for the first column and poor for the third column.

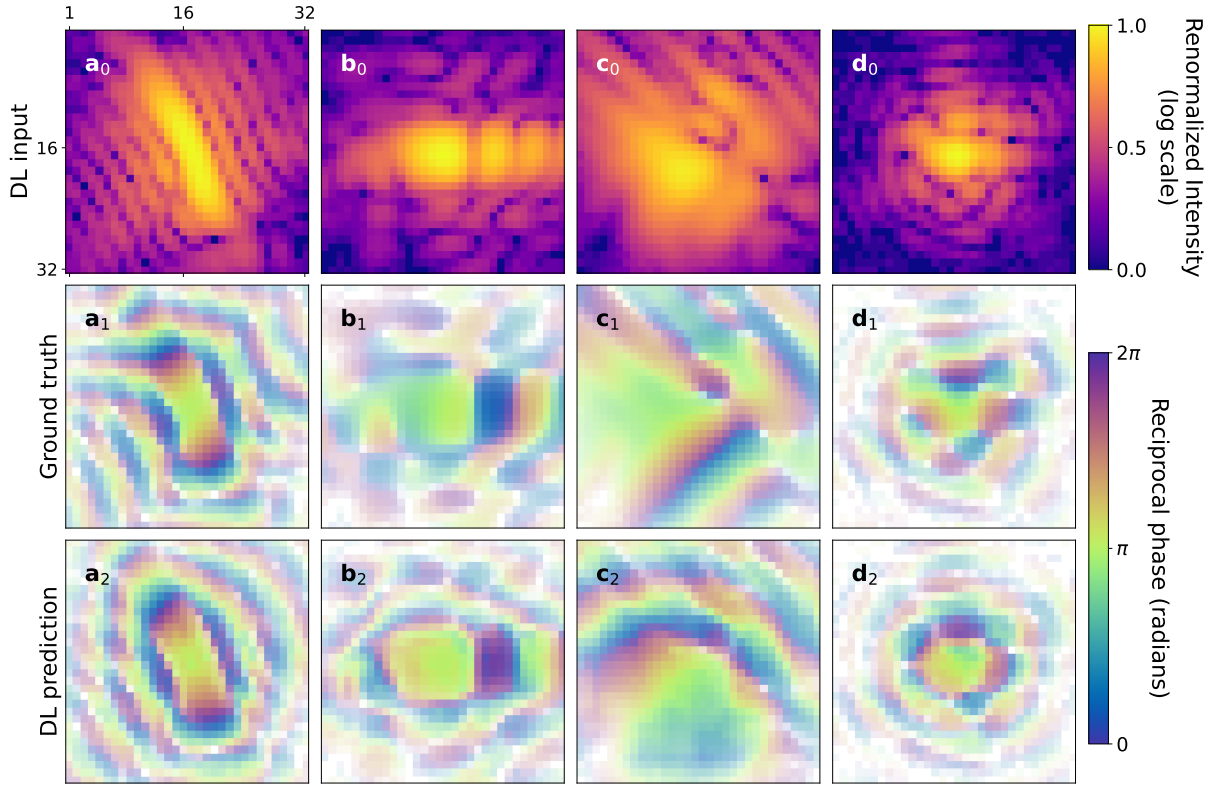


Figure 4.26: Slices of the central patches. First row shows four different examples of simulated high-strain BCDI patterns cropped around the center of the peak. Ground truth and predicted RSP are shown in second and third row respectively. The model manages to estimate the correct RSP when this shows spherically symmetric iso-phase regions while struggles more for different symmetries.

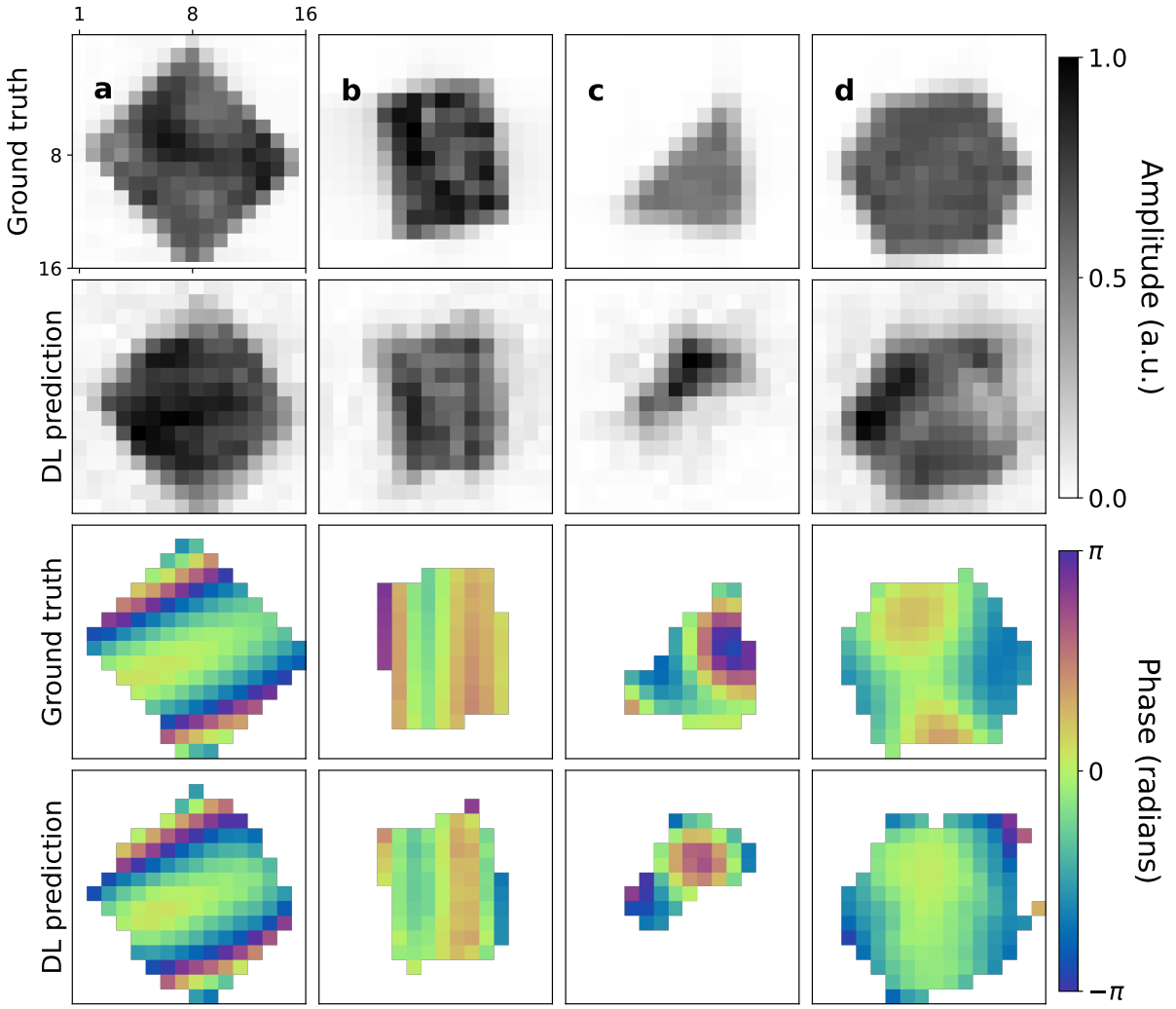


Figure 4.27: Corresponding reconstructed objects. Except for case **c** the predicted RSP is good enough to well estimate the size, shape and phase of the real space object.

Regarding the second CNN trained on patches instead, one can observe that the performance is not severely affected by the high-strain. Strong discrepancies between predicted and ground truth RSP are mostly present where there is no intensity signal, thus not relevant (see Fig.4.28). The good accuracy of on the outer patches predictions suggests that the crucial and more challenging mapping to retrieve involves the central patch mostly.

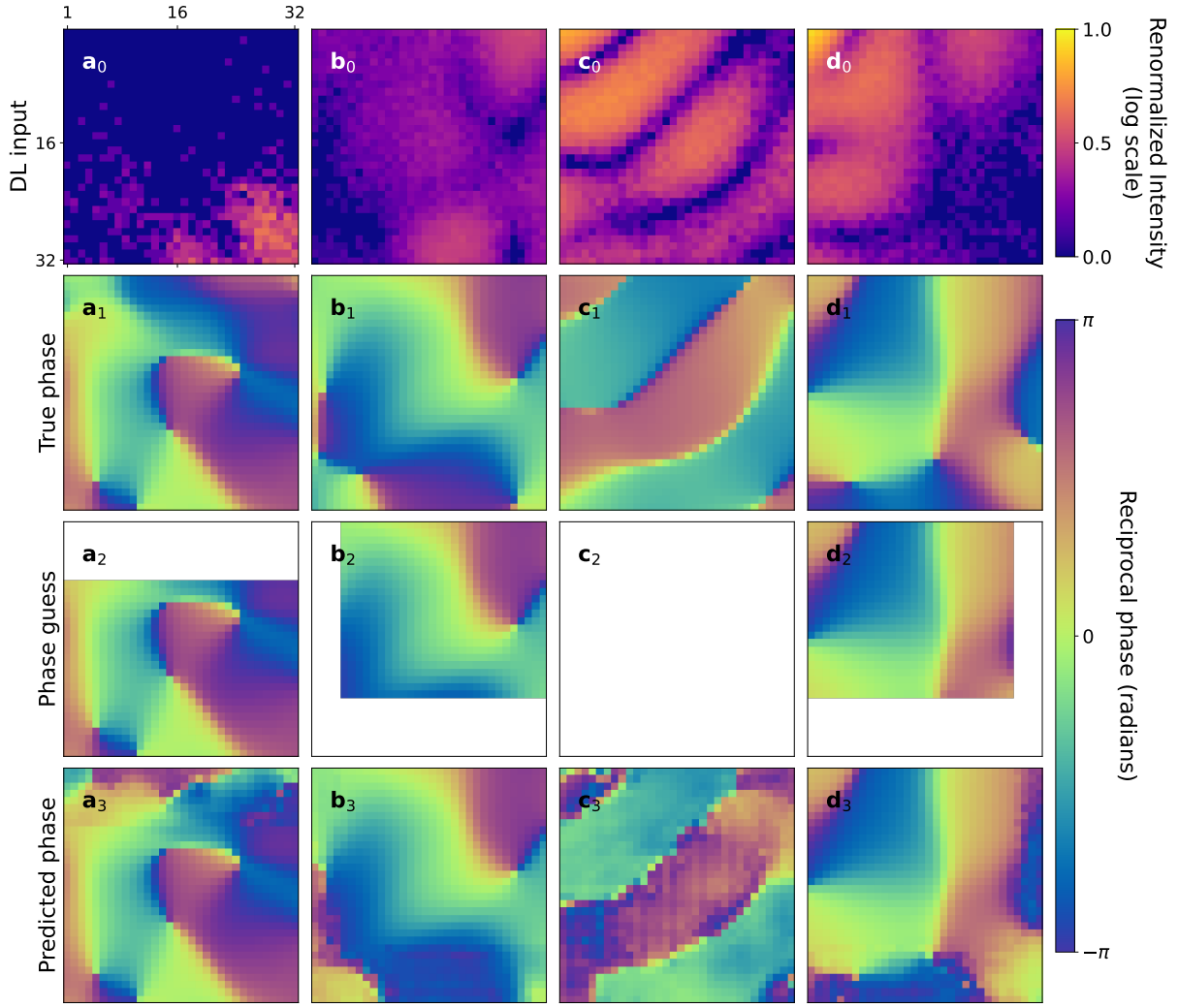


Figure 4.28: Examples of RSP prediction for outer patches cropped from simulated high-strain data. Similarly to the examples shown in Fig. 4.21 for the low-strain case, the model yields relatively correct outputs. Worse predictions are observed where low intensity signal is recorded, therefore less important during the reconstruction.

For completeness, it is shown here (Fig.4.29) the result of the full RSP stitching and the corresponding reconstructed object for the high-strain case as well. A simulated test data has been used for the ground truth comparison. It is clear that the stitching algorithm is performing poorly as observed for the low-strain case. However, the central RSP patch is fairly similar to the ground truth and therefore the low resolution reconstructed object shows roughly the correct shape and phase.

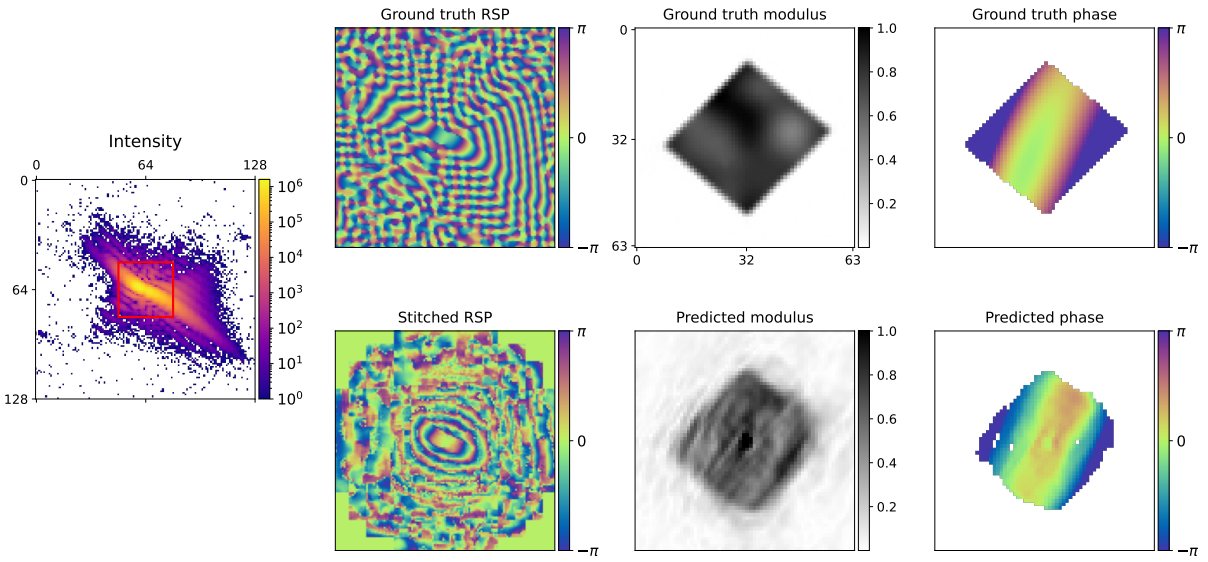


Figure 4.29: Results of the stitching of RSP predicted patches for 3D simulated "high-strain" data (central slice displayed).

4.7.1 Discussion

Although failed, this study on the prediction of the RSP in smaller patches has led to better understanding of the problem and nevertheless unveiled some interesting insights. For instance, it showed that the retrieval of the mapping between patches is possible with a CNN trained with the WCA loss function, untying even more the relationship with the real space object. Moreover, it emerged that the main difficulty of this approach is given by the stitching of the RSP patches into the full array. As mentioned above, the hypothesized reasons for this problem are (i) the fact that the model "sees" simulated ground truth RSP guesses during the training and predicted ones during inference, and (ii) the averaging of RSP predictions for overlapping voxels. In order to overcome these limitations other approaches were contemplated but never realized for lack of time. In particular, it was imagined a way to extract, analyze and stitch the patches inside the model into a sort of Recurrent Convolutional Neural Network (RCNN) that would keep track of the previous innermost shell thanks to a dedicated convolutional Long-Short Term Memory (LSTM) [49]. By doing this the model would always be exposed to its own RSP predictions as initial guess for outer patches and the RSP average over overlapping voxels could be replaced with a convolutional layer with non-linear activation function. While the attempts of setting up such model are not reported here, it is mentioned the idea as possible inspiration for future works.

To conclude, main finding of this study on patches is that it is crucial for a good object estimate to accurately predict the RSP in the vicinity of the center of the Bragg peak, and that the CNN model trained with the WCA can accomplish this task for highly strained patterns as well. It was therefore decided to invest the efforts into a regular CNN for the prediction of the RSP of 3D highly strained BCDI patterns with the intermediate size of $64 \times 64 \times 64$ pixels.

4.8 Model design: 3D case high strain

The model here described and the results obtained on simulated and experimental 3D BCDI patterns can be found entirely in the paper "Phase Retrieval of Highly Strained Bragg Coherent Diffraction Patterns using Supervised Convolutional Neural Network" ...

The architecture that was employed is an adaptation of the 3D U-Net employed for smaller patches Fig. 4.30 As learned from the preliminary study on the 2D case, to better interpolate the diverse distribution of distorted BCDI pattern, the number of trainable parameters and samples in the training dataset was increased significantly. Additional encoder and corresponding decoder blocks were added for a total of 145 million parameters. Moreover, similarly to the model used for the gap inpainting, dilated convolutions were adopted in the first two encoder blocks in order to improve the receptive field of the model.

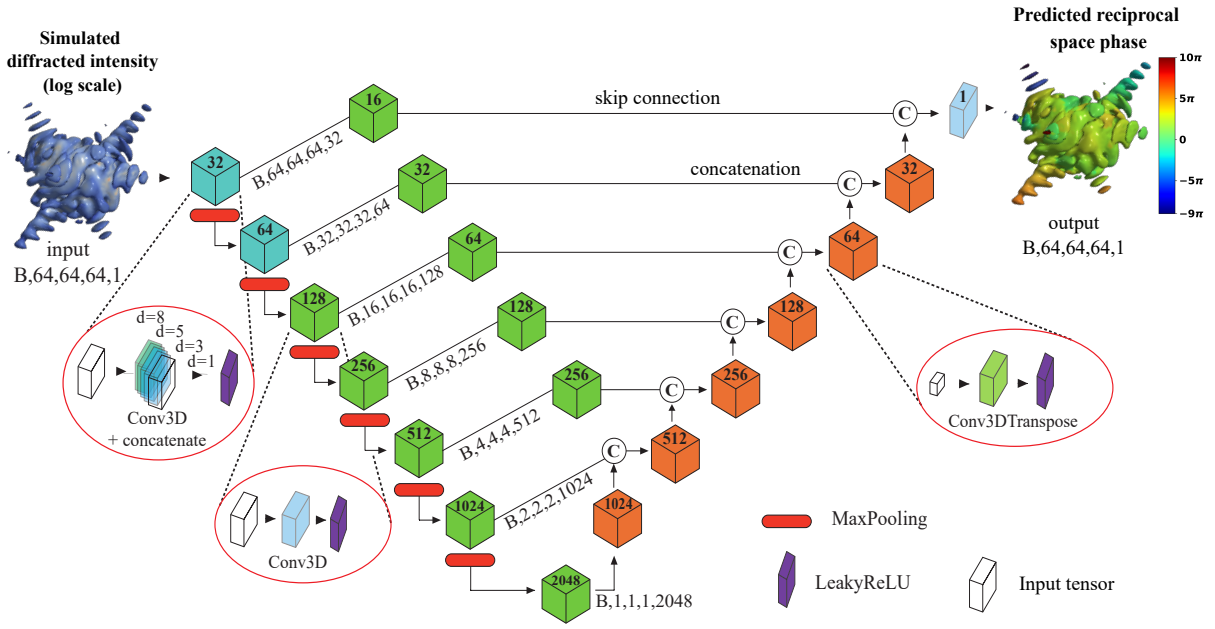


Figure 4.30: Schematic of the 3D U-Net model employed for the RSP prediction of highly strained patterns.

95'000 simulated BCDI patterns have been created following the procedure described in Sec.3.4.1 on a cubic 64 pixel-sided grid for different particle's shape, strain distribution, oversampling conditions and noise levels. Another smaller dataset containing 4'000 samples was created instead for testing the model.

The model was trained with the WCA loss function for 60 epochs with a learning rate of $10^{-}4$. To speed up the process, the training has been conducted using two NVIDIA TeslaV100-SXM2-32GB GPUs using the MirroredStrategy feature for synchronous training across multiple devices provided by Tensorflow library. This measure allowed to reduce the training time from 2 hours per epoch to 30 mins.

4.8.1 Results: simulated data

Once completed the training, the model has been tested on simulated data. Here, Figs.4.31 - 4.32 illustrate some examples of comparison between ground truth and predictions for both the RSP and the reconstructed objects respectively.

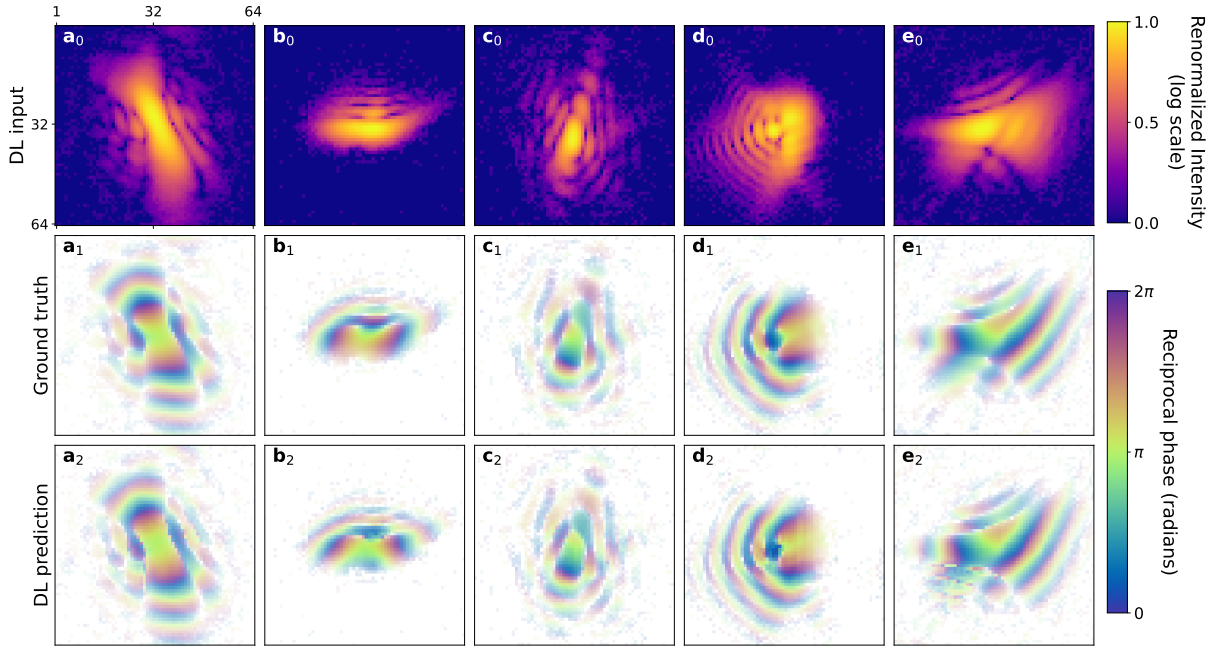


Figure 4.31: a_0 - e_0**)** Central slices of simulated input intensities from the test dataset. a_1 - e_1**)** Corresponding ground truth RSP. a_2 - e_2**)** Corresponding slices taken from the Deep Learning (DL) model prediction of the RSP.

The model correctly predicts the RSP oscillations inside the fringes also when these have been distorted or merged into a single continuous intensity stripe (Fig. 4.31 a-b) because of the high strain along the corresponding axis in real space.

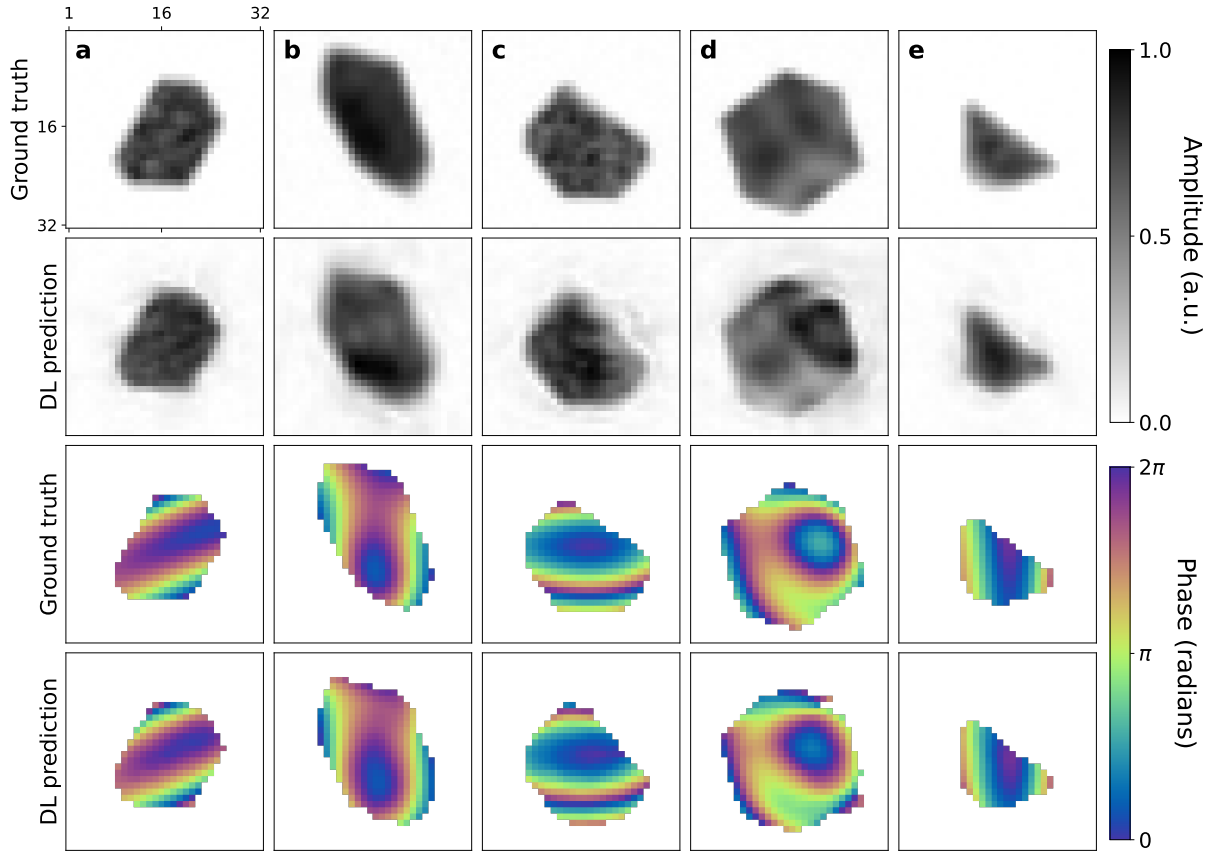


Figure 4.32: Reconstructed objects from the diffraction patterns in Fig. 4.31. **a-e)** Central slices of both the modulus and phase for the ground truths and the DL reconstructions.

Except for some noise and inhomogeneities in the objects' moduli, the reconstructions from the predicted RSP achieve good accuracy on simulated data, for different particle shapes and strain distributions. It is worth mentioning that the presence of noise affecting the objects' shapes is an effect of the loss function that is not computed in real space. The model is never directly shown that the real space object are compactly supported, as it comes as consequence of the correct RSP prediction. However, while small discrepancies of the RSP prediction result in noise on the object shape, an overall accurate RSP prediction ensures the retrieval of both the correct shape and phase of the object. This fact is of primary importance when considering the use of the DL prediction as starting point of iterative refinement with conventional algorithms. It is indeed easier to reach convergence from a low resolution noisy but accurate estimate of the object rather than a clean but inaccurate one.

4.8.2 Results: experimental data

Given the satisfactory results obtained on simulated data the DL model was tested on experimental data as well. Two relevant examples of BCDI patterns collected at the ID01 beamline of the ESRF-EBS are considered here. The first pattern (Fig. 4.33 a) is given by a platinum nanoparticle on Yttria-stabilized zirconia (YSZ) (Particle 1) while the second (Fig. 4.35 b) is a dewetted platinum/palladium bilayer on a sapphire substrate (Particle 2).

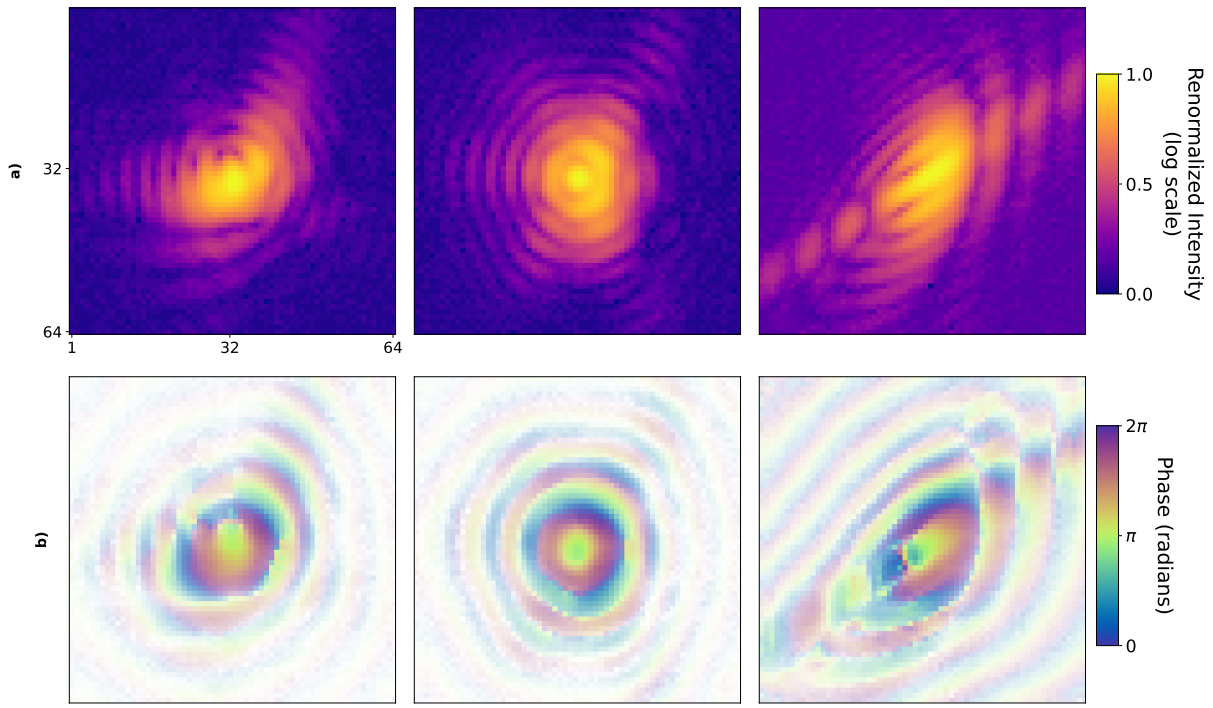


Figure 4.33: a) Central slices of the experimental diffraction pattern from Particle 1. **b)** Corresponding slices of the DL model RSP predictions.

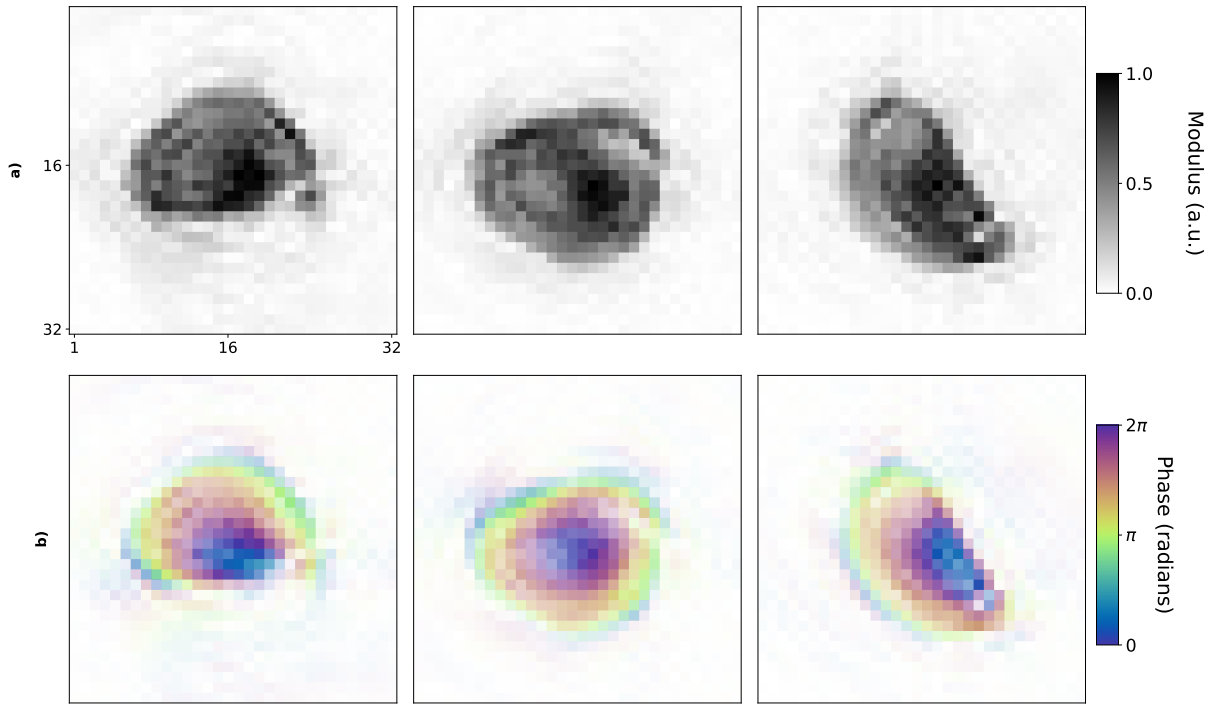


Figure 4.34: Central slices of the reconstructed Particle 1 modulus (**a**) and phase (**b**)

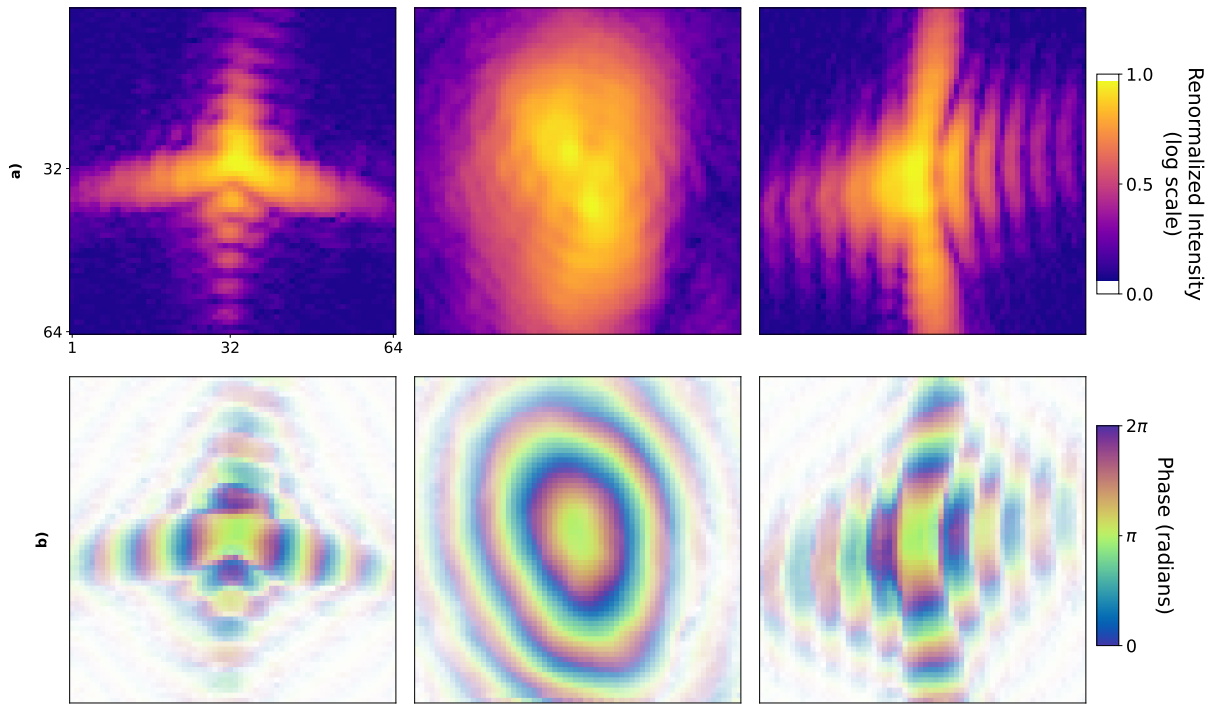


Figure 4.35: a) Central slices of the reconstructed Particle 2. **b)** Corresponding slices of the DL model RSP predictions.

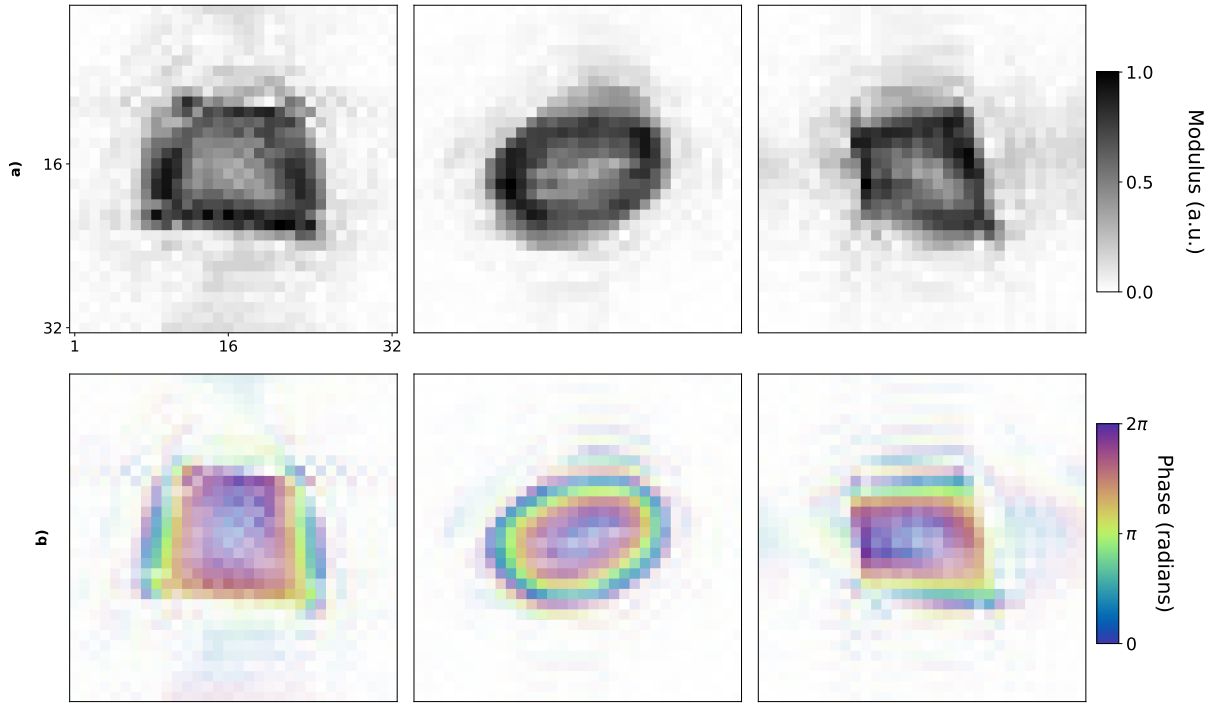


Figure 4.36: Central slices of the reconstructed Particle 2, modulus (**a**) and phase (**b**)

Fig. 4.34 and 4.36 show the reconstructed object obtained from the predicted RSP. Despite the low resolution and the presence of noise it is possible to recognize the shape of realistic particles and phases.

In these cases of PR of experimental data it is impossible to establish a comparison with a ground truth because not available. In fact, 60 independent runs of standard iterative algorithm have been launched in order to reconstruct these datasets but no satisfactory results were obtained. Precisely, the recipe of 400 HIO + 1000 RAAR + 300 ER and different thresholds for the support estimation were used for each run. Moreover, the 3 best results according to the Free LLK metric [50] were combined with mode decomposition to improve the quality of the reconstruction. However, in both cases the obtained reconstructions were not satisfactory as holes were present (Fig.4.37), or the support was excessively shrunk (Fig.4.38).

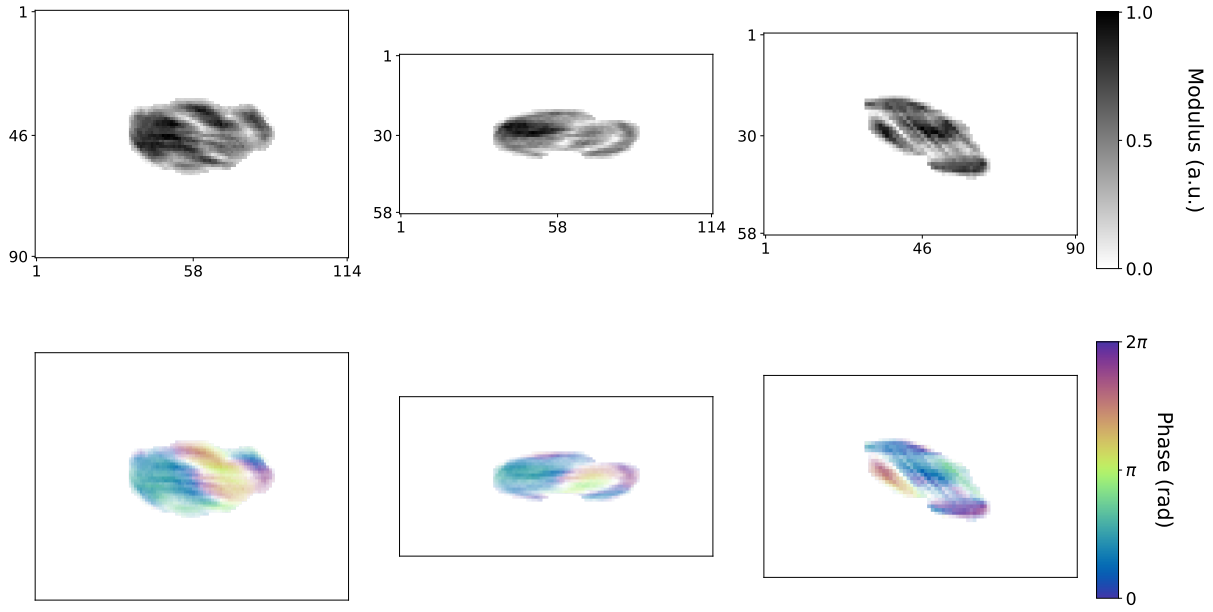


Figure 4.37: Combination of the 3 best reconstructions of particle 1 out of 60 independent runs with PyNX (central slices). Although the shape is guessed the presence of holes in the modulus denotes a poor quality of the result.

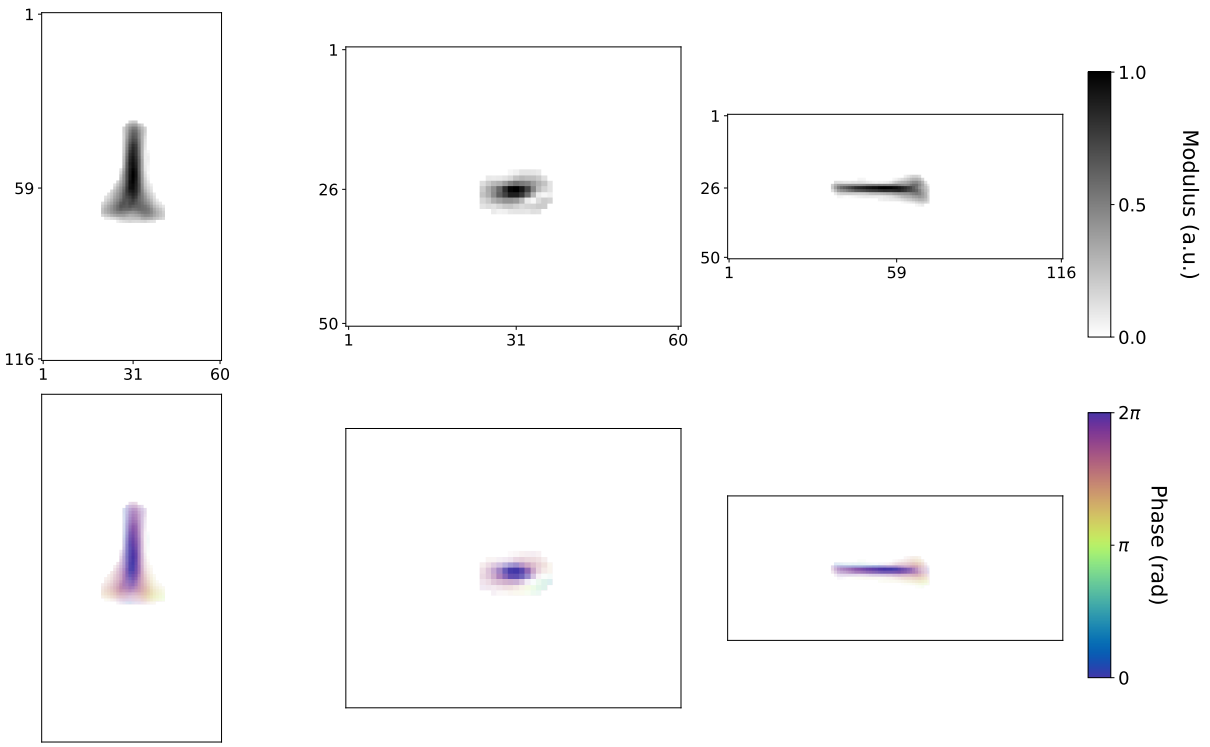


Figure 4.38: Combination of the 3 best reconstructions of particle 1 out of 60 independent runs with PyNX (central slices). The high strain induced by the substrate deceives the standard PR algorithm that excessively shrinks the object's support along the axes perpendicular to the highly strained ones.

4.9 Refinement with iterative algorithms

The use of the DL model for the phasing of experimental data has to be intended as a preprocessing step that can provide a starting point for further iterative refinement. Classical ER algorithms can be used to polish the DL estimate, simplifying the convergence. At this stage it is worth mentioning that, since the DL model was trained on 64 pixel-size cubic volumes, it was necessary to crop and resize the experimental diffraction patterns. This procedure requires special attention as for different cropping/resizing parameters the DL prediction varies. In particular, it is curious to notice that the cropping/padding operations in reciprocal space correspond to interpolations in real space and vice versa. For this reason the procedure illustrated in Fig.4.39 had to be adopted to adapt the dataset to the DL format and then bring back the predicted object to the size relative to the original dataset one. This step is fundamental when the DL object is then used as starting point for further refinement with classical algorithms. Because of the large diversity of the region of interest (ROI) from one experimental dataset to the other, the manual intervention is for the moment needed to adjust the cropping and binning parameters such that the data fed into the DL model resembles the simulated data used for training.

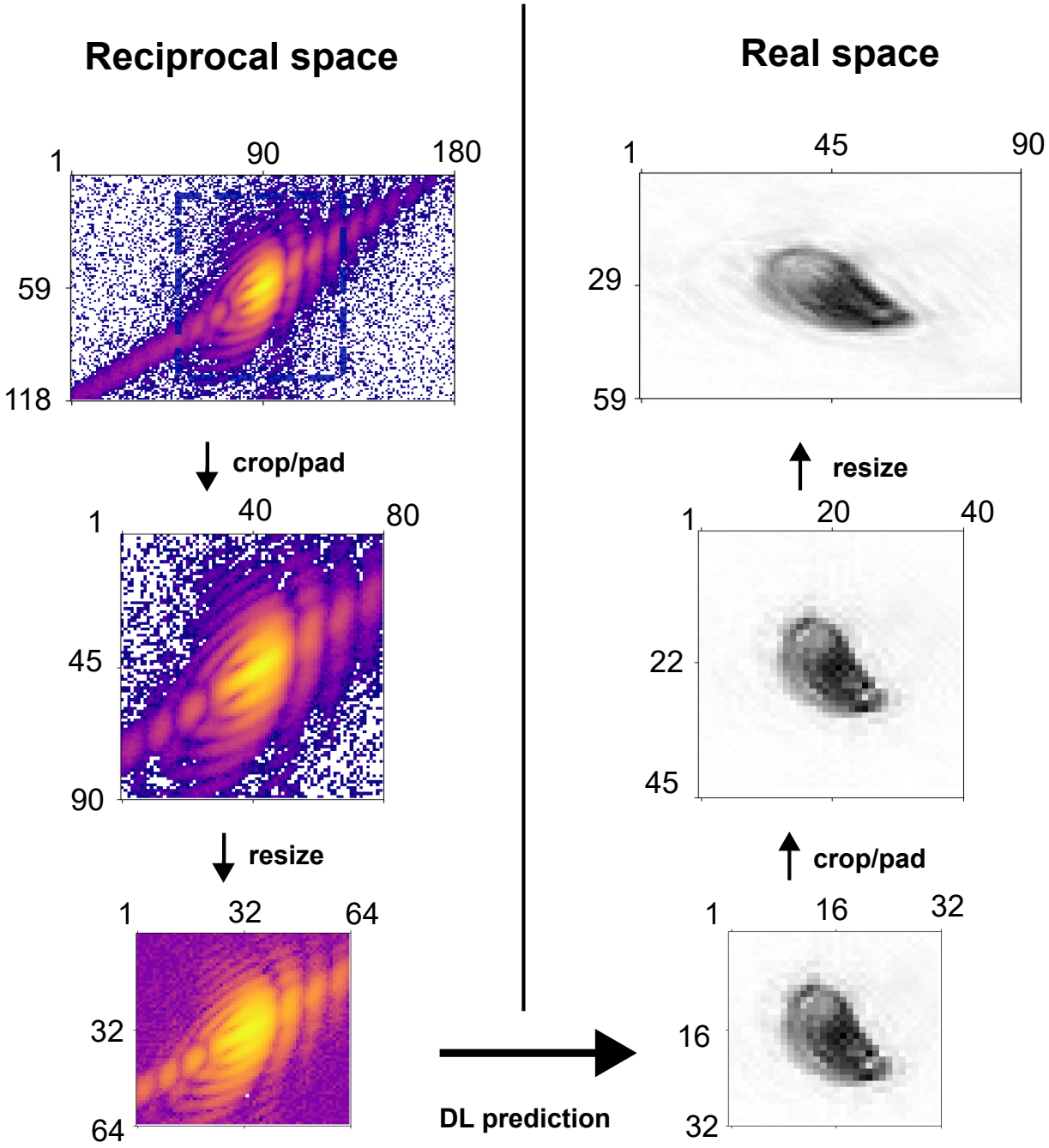


Figure 4.39: Manipulation of the datasets. Here the experimental BCDI pattern is firstly cropped around the COM of the Bragg peak, cutting out parts of the signal. The data is then interpolated into a 64 pixel sided cubic grid and transformed in normalized log-scale. The object obtained after the DL RSP prediction is then padded and resized back to the original shape such that it can directly be plugged into an iterative algorithm for refinement.

If the DL predicted object is a good low-resolution guess of the solution, few steps of ER are usually enough to reach convergence. ER is indeed the simplest alternating projection method and converges linearly to the local minimum [51] so it represents the optimal choice when the DL prediction places the starting point around the minimum. In this specific case 300 iterations of ER were performed and, since the estimated DL object is assumed to be close to the solution, only the pixels at the border the support were allowed to be updated. Figures 4.40 and 4.41

show the results obtained after refinement presented above.

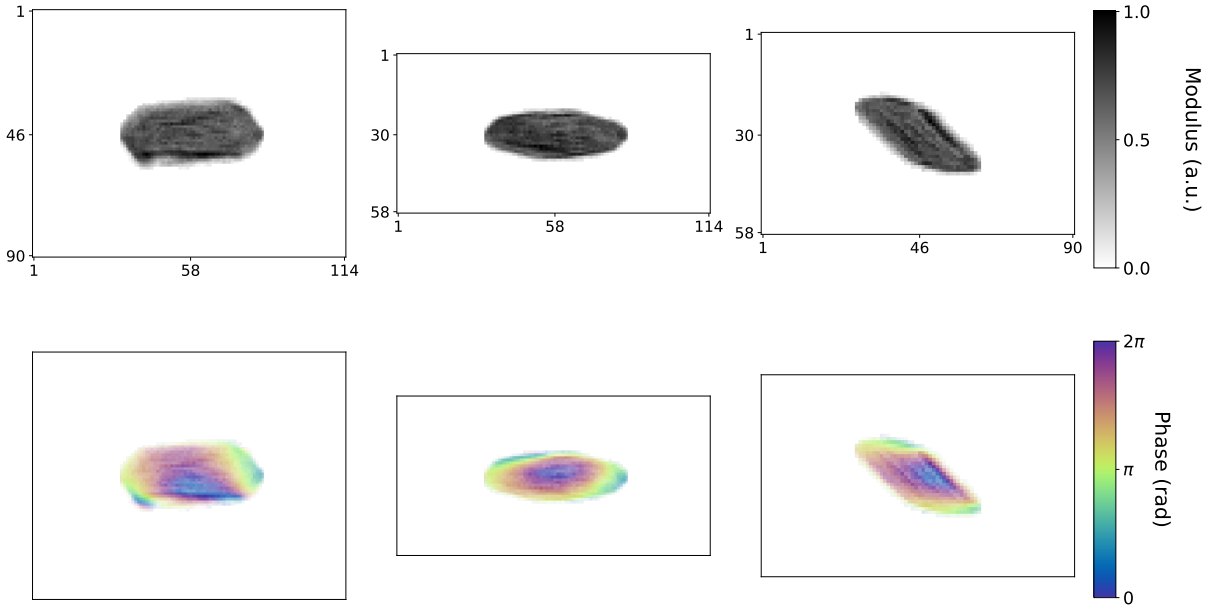


Figure 4.40: Central slices of Particle 1 after 300 cycles of ER for refinement of the DL initial estimate.

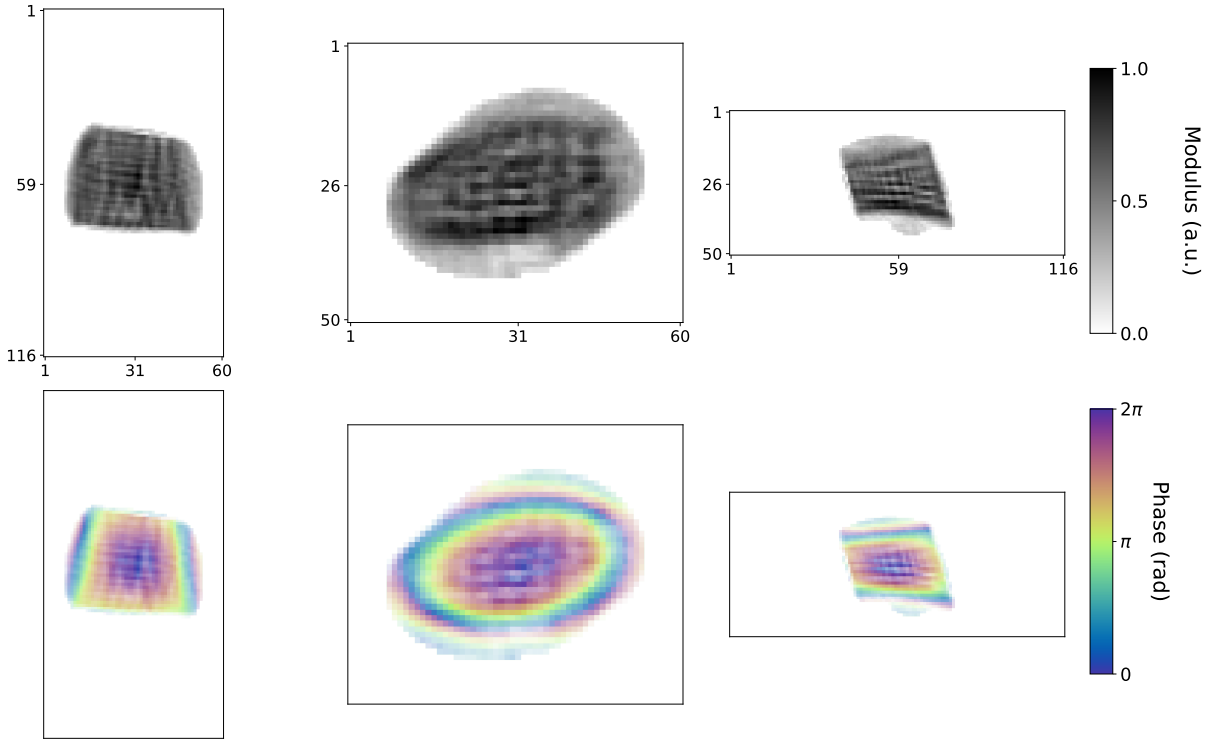


Figure 4.41: Central slices of Particle 2 after 300 cycles of ER for refinement of the DL initial estimate.

The improved homogeneity of the modulus and the more physical shape of a Winterbottom particle are a clear sign of better reconstructions for both cases. One can observe that the shape and phase of the final reconstruction are not remarkably different from the DL estimates. This proves that the DL model is able to generalize correctly to experimental data as well and that

the following ER steps are a good choice for refinement.

While for Particle 1 it is obvious that the DL + PyNX solution is better than the PyNX one because of the absence of "holes" in the modulus, for Particle 2 it can be harder to believe that the DL + PyNX approach actually yields an improved solution. For this reason it is worth calculating the diffracted intensity from the reconstructed object and its fidelity with respect to the observed one. Here, Fig. 4.42 shows that the calculated intensity obtained from the DL + PyNX is more accurate than the one obtained form the PyNX - only solution.

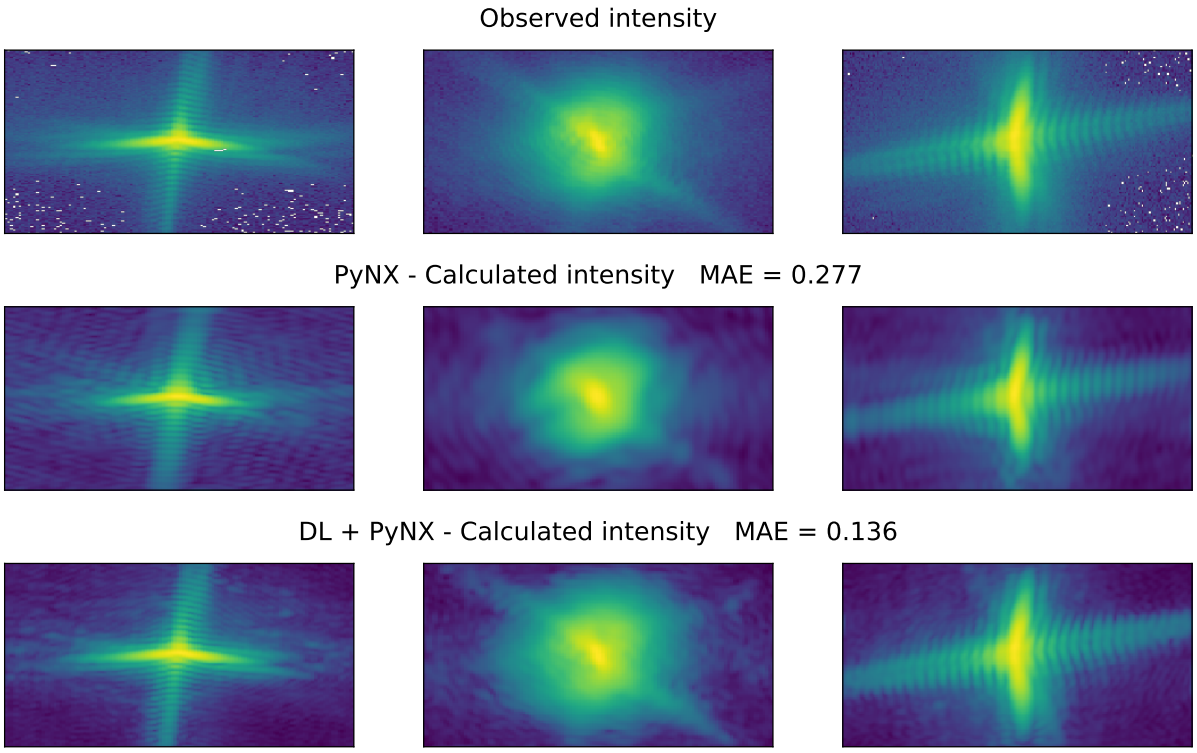


Figure 4.42: Projections along the three axis of the observed and calculated diffraction patterns relative to Particle 2. The mismatch (MSE) of the one calculated from the particle reconstructed using PyNX only, is higher than the one calculated from the particle obtained with the DL + ER method. This proves that the latter is the correct solution.

Beside the higher quality of the results it is worth mentioning the significant reduction of the wall-clock time. Table 4.2 summarizes the time taken for the 3 different methods. A 40X to 50X speed-up is recorded for these dataset sizes and the time saving increases for larger ones.

4.10 Performance assessment

In this paragraph the results of the DL model presented above, tested against different strain configurations and magnitudes are discussed. The scope of the study is to assess the model's performances for these different cases, to evaluate when it works best and ultimately to estimate appraise the gain in accuracy when coupled with ER refinement. The first test was prepared simulating a Winterbottom shaped particle, similar to the ones used for the training set, with an

	Particle 1	Particle 2
Data size	(118,180,230)	(110,160,200)
DL inference	3.04 s	2.21 s
PyNX: 60 runs	227.41 s	123.55 s
DL + PyNX: 1 run	4.73 s	3.08 s

Table 4.2: Computation times for the three different methods. DL RSP prediction and inverse FT are considered for DL time. PyNX - only time includes 60 runs of 400 HIO + 1000 RAAR + 300 ER performed in parallel by PyNX. The best reconstructions selection and following mode decomposition is not considered. DL + PyNX time includes the full DL time plus the reshaping of the initial guess to the original size and a single run of 300 ER.

applied phase built with two Gaussian functions with two different and increasing amplitude ranges. The corresponding BCDI pattern has been simulated for each case, keeping the same oversampling ratio and noise level across the simulations. For each calculated diffraction pattern the RSP has been predicted using the DL model and corresponding objects have been obtained with inverse FT. At this point the accuracy of the prediction was calculated with using the formula in 4.5 and the results are shown in Fig.4.43.

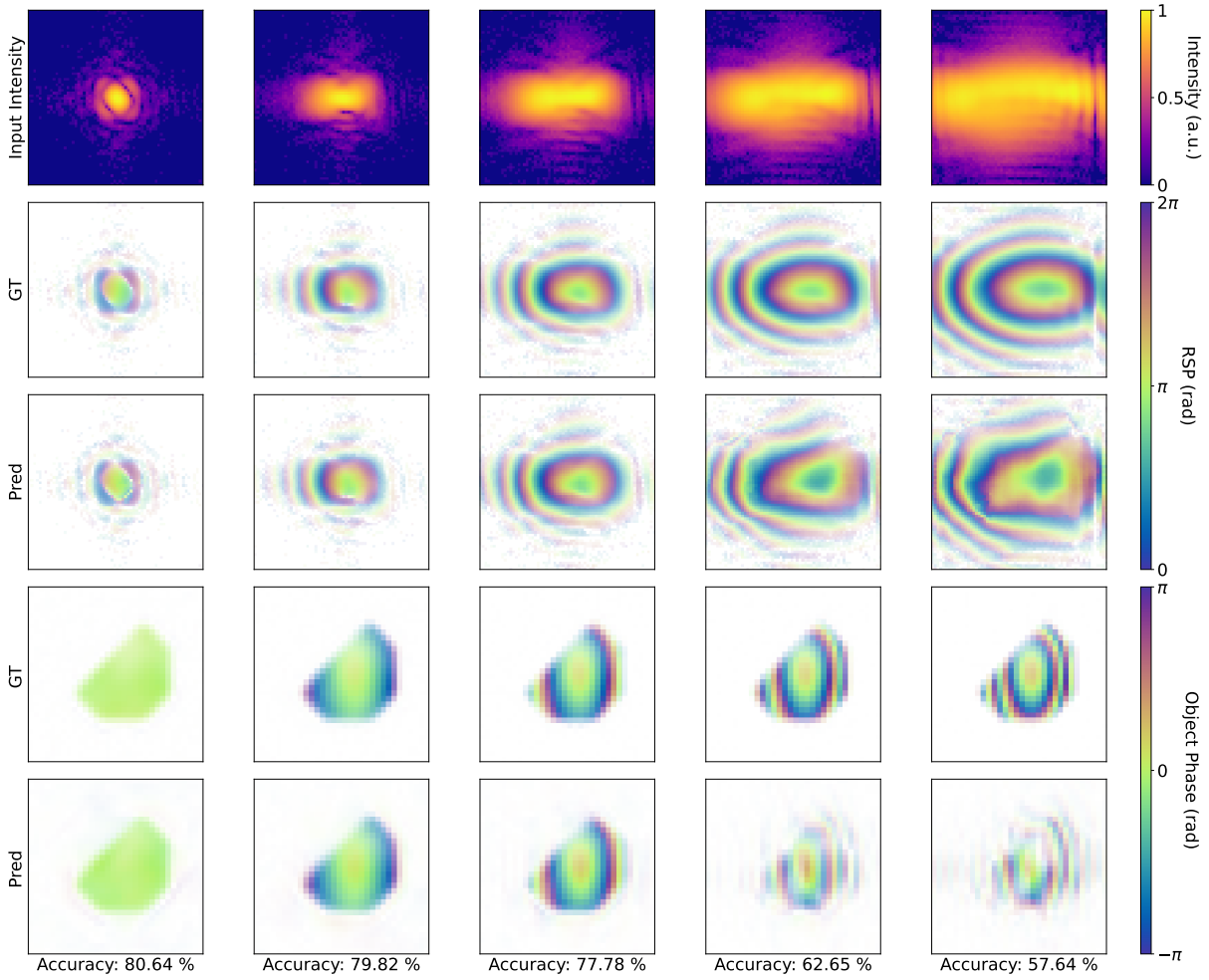


Figure 4.43: Evolution of the BCDI pattern and corresponding RSP and object for higher strain calculated with the sum of two Gaussian functions of increasing amplitude. Third and fifth row show the DL predicted RSP and the reconstructed object's phase respectively. As expected the prediction worsens for high phase ranges but it overall maintains a similarity with the ground truth.

As expected the accuracy drops as the phase range increases. It is interesting to notice that despite the worsening of the prediction with the increasing strain the object's phase structure inside the support recalls the ground truth phase. This detail is fundamental for the ER refinement as the initial guess, notwithstanding the inhomogeneous and "shattered" support, represents a good estimate of the solution. Objects with similar cleaner supports of incorrect shapes and phases are much worse starting point for iterative refinement since far from the solution.

The same procedure has been repeated for a different strain configuration, this time constructed with two cosine functions with increasing amplitude. The phase distribution induced by these functions is rather different, more in the spatial structure than the amplitude. Fig. 4.44 reports the results showing that the model accuracy is significantly poorer than the previous case. It is worth reminding that the training set was composed of equal amounts of particles simulated with Gaussian and cosine phase profiles, meaning that the lower accuracy scores are not due to some possible imbalance of the training. The phase range inside the particle, that can be estimated visually by the amount of wraps inside the support, is also in the same order

of magnitude for both cases meaning again that the reason for the poorer results is of other nature.

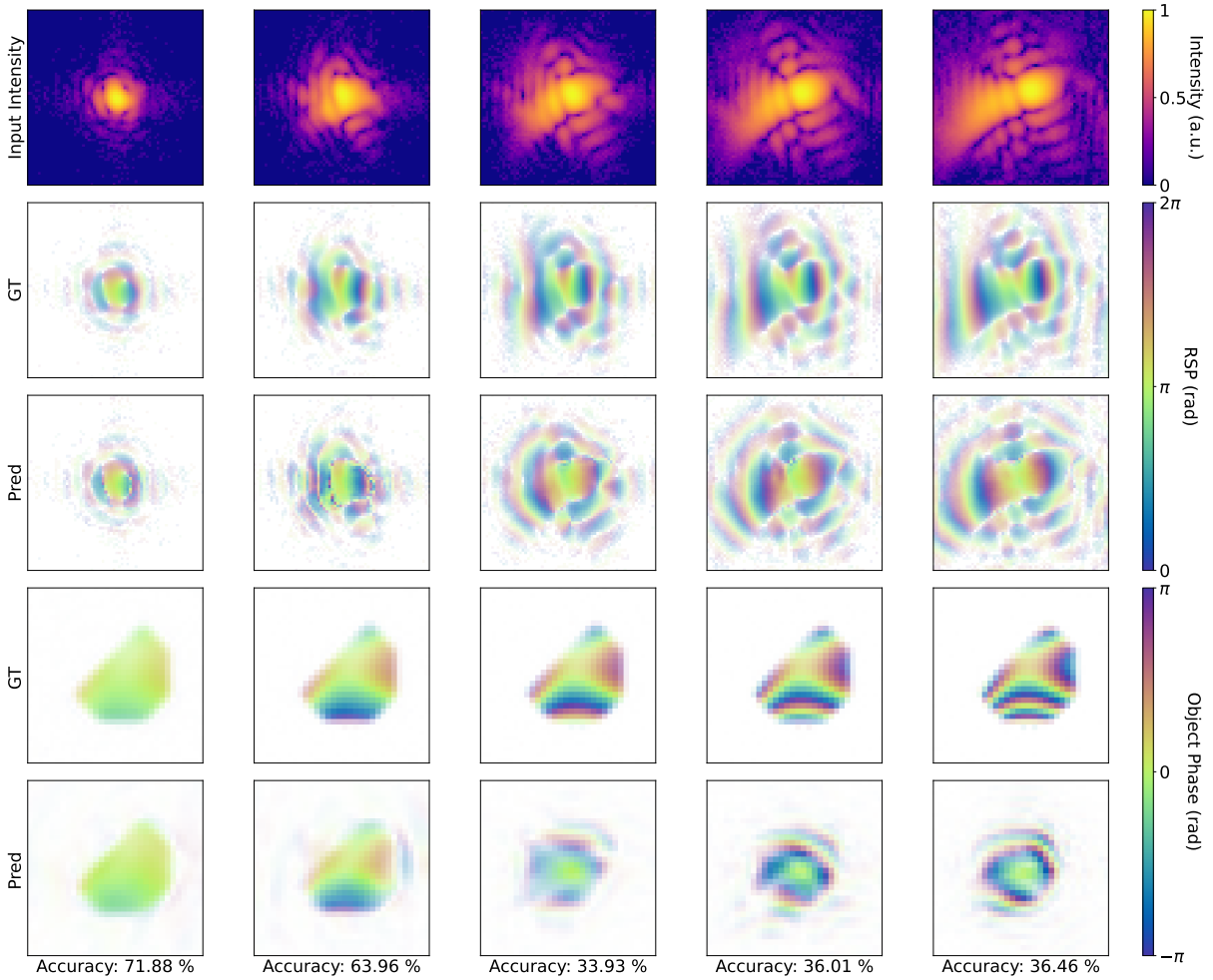


Figure 4.44: Evolution of the BCDI pattern and corresponding RSP and object for higher strain calculated with the sum of two cosine functions of increasing amplitude. Third and fifth row show the DL predicted RSP and the reconstructed object's phase respectively. The model rapidly struggles with this type of phase field.

Similarly, the model seems to struggle more for phase fields simulated with using Gaussian correlated random field (same as in 3.4.1) than for the first case. Again the reason doesn't seem to be related to the different phase range nor population imbalance in the training set (Fig. 4.45)

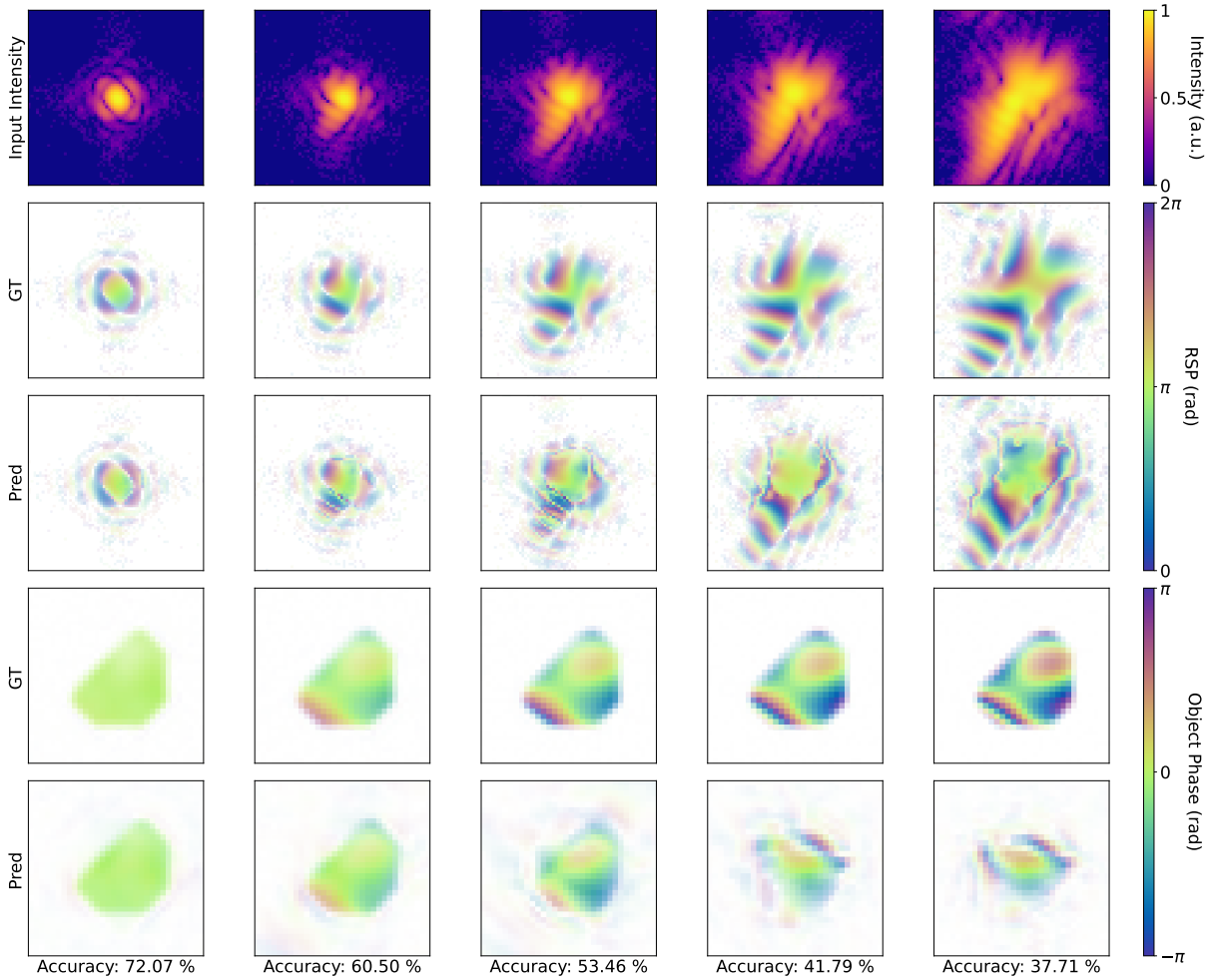


Figure 4.45: Evolution of the BCDI pattern and corresponding RSP and object for higher strain applied by using a Gaussian correlated random field of increasing amplitude. Third and fifth row show the DL predicted RSP and the reconstructed object's phase respectively.

To better assess the performances of the model on the three different cases, the same procedure has been repeated on a bigger and more diverse population. Namely, 20 randomly shaped particles have been simulated and for each of them an increasing strain field has been applied in 10 different steps, for the three type of distributions. The results of the model are shown in Fig. 4.46

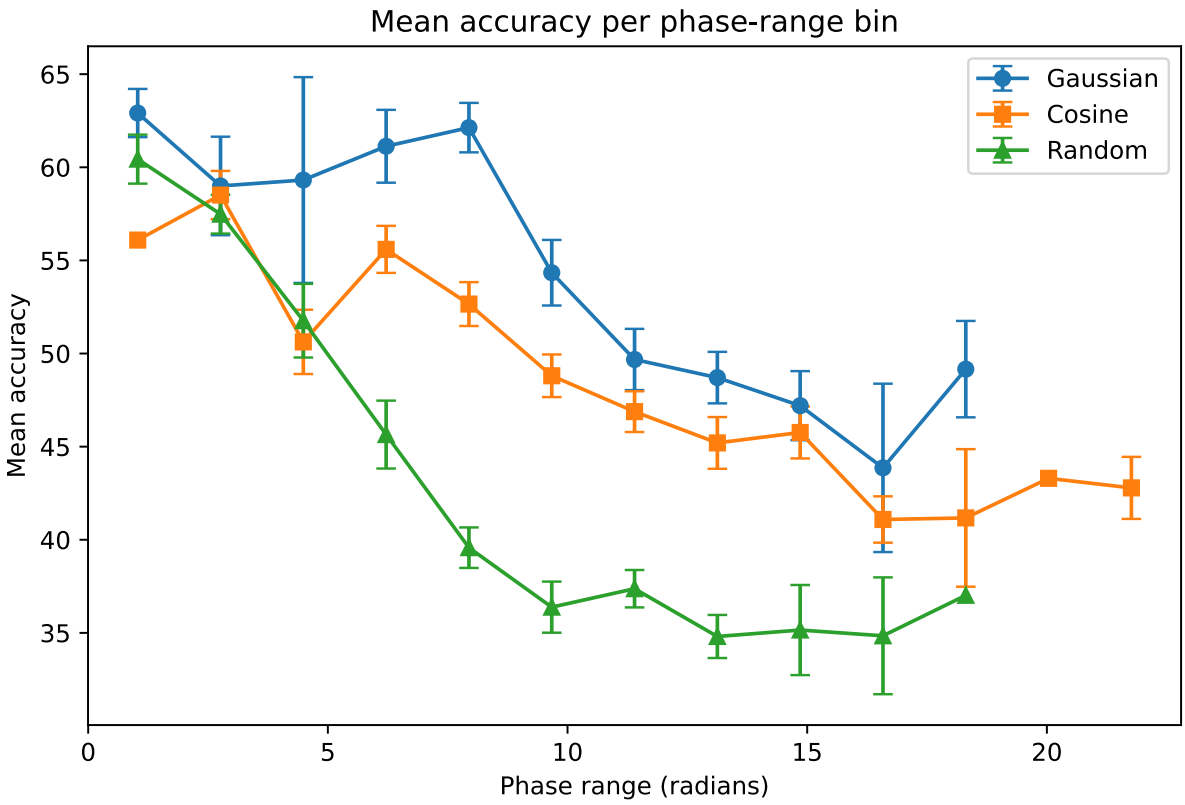


Figure 4.46: Mean accuracy of the DL model for different phase ranges and types. The phase field simulated with two Gaussian functions yield RSPs that are better predicted by the DL model than the ones obtained with two cosine functions or random Gaussian fields.

Another study that has been conducted aimed at estimating the accuracy gain of the DL model coupled with ER refinement compared to the sole iterative algorithm. In this case a single Winterbottom particle has been selected for simplicity, and an increasing phase field was applied in 50 steps for the three different types. At this point the accuracy scores have been calculated for the DL model only first. Separately, a single run of 400 HIO + 1000 RAAR + 300 ER has been performed using PyNX for each of the 50 diffraction patterns. At last, the DL model predicted objects have been used as starting point for 300 ER refinement, always with PyNX. In all cases, it was made sure that the solution found by either the DL or the iterative algorithm wasn't the twin of the ground truth. If so, the object was flipped before calculating the accuracy.

The analysis has been repeated for the three different phase fields used in the above cases and the results are shown in Figs. 4.47, 4.48, 4.49.

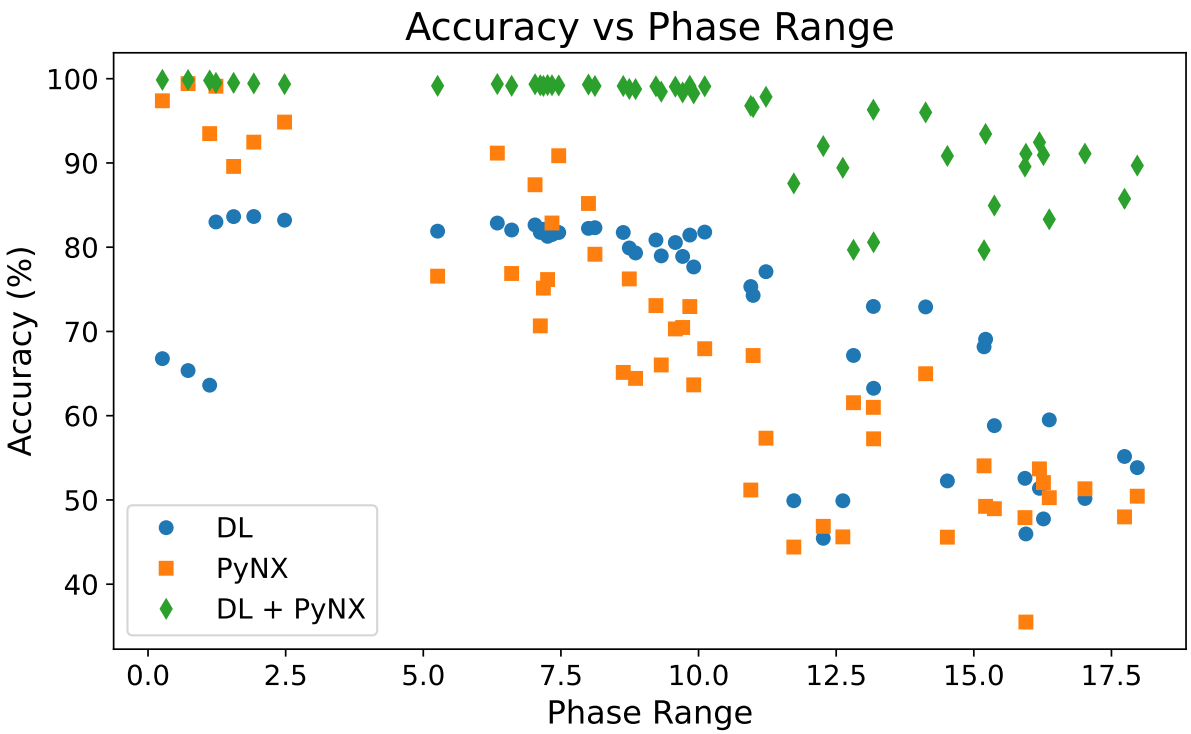


Figure 4.47: Comparison of the accuracy scores of the DL model prediction, iterative PR and DL + ER approaches for objects with phase fields simulated with two Gaussian functions with increasing amplitude. The DL + ER reconstructions always achieve higher accuracies than the other methods, also in those cases in which the DL ones are low.

While for lower phase ranges the DL prediction is worse than the two other methods, the main result of Fig.4.47 is that in all cases the accuracy of the DL + ER method is always superior, meaning that the DL model can significantly aid the PR process also when the first estimate is not very close to the solution (down 50% accuracy in the figure.).

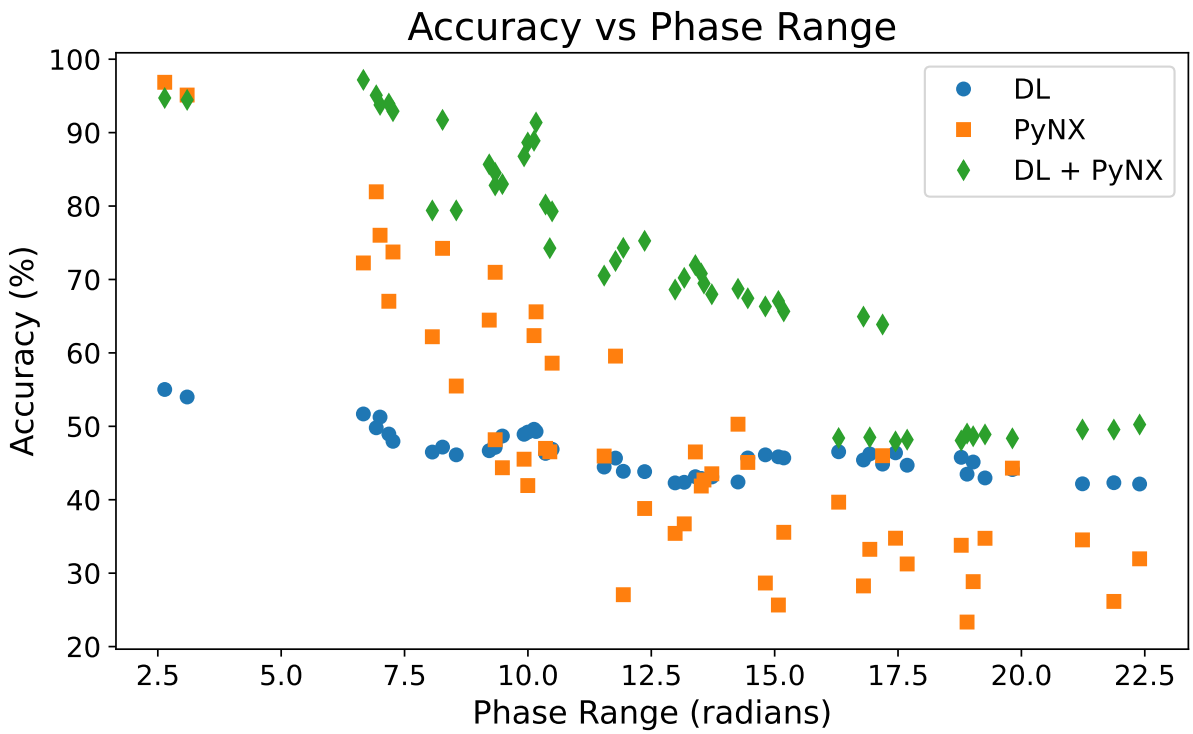


Figure 4.48: Comparison of the accuracy scores of the DL model prediction, iterative PR and DL + ER approaches for objects with phase fields simulated with two cosine functions with increasing amplitude.

Similarly, the DL model proves to be significantly helpful at providing a good initial guess for the ER refinement when the object's phase is simulated with cosine functions.

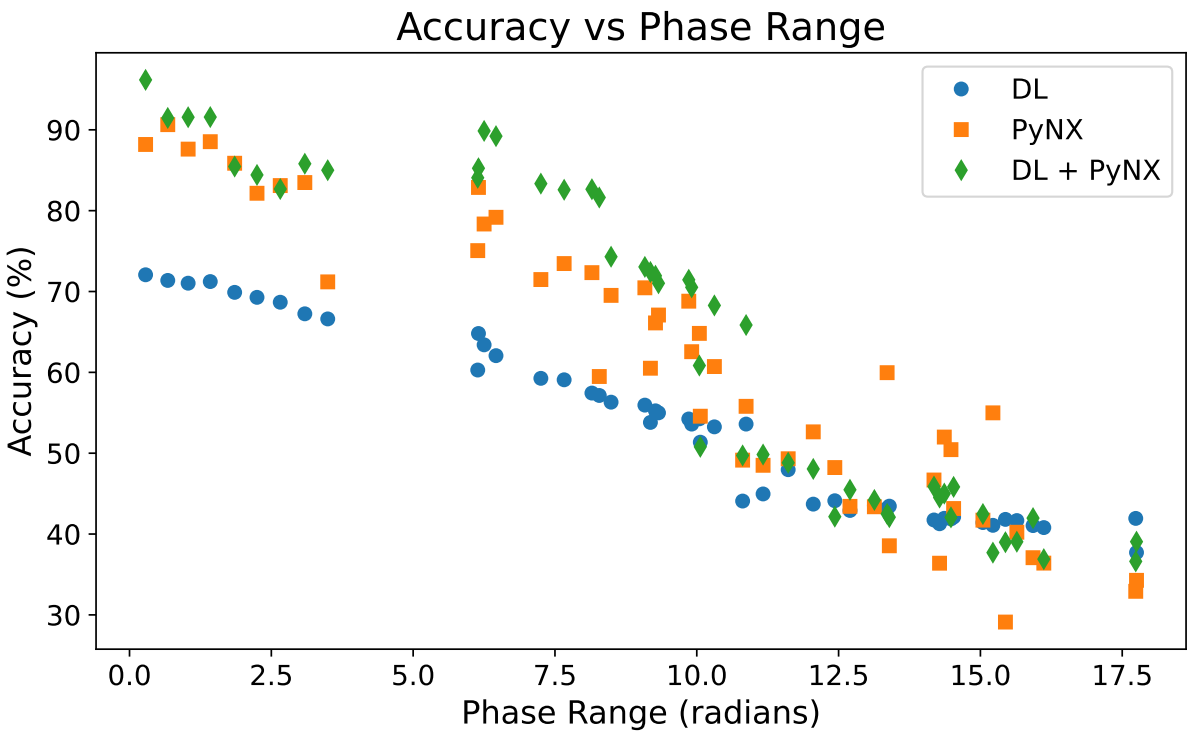


Figure 4.49: Comparison of the accuracy scores of the DL model prediction, iterative PR and DL + ER approaches for objects with phase simulated with a Gaussian correlated random field with increasing amplitude. The combination of DL + ER yields better results up to a phase range value of 10 radians. For higher values the poor DL prediction does not provide a good enough estimate of the solution thus it does not help the convergence of the ER method.

In this case as well, the DL model, although when used alone yields relatively low accuracy scores, improves the quality of the solution when coupled with some ER cycles for refinement. At the same time it is interesting to notice that when the accuracy of the DL prediction is too poor - most likely when, as in Fig.4.45, the object's phase differs too much from the ground truth one - the ER cycles do not improve the result as the initial estimate is wrong.

As last study on the performances of the DL model, the reasons behind the imbalance of accuracy scores for different phase fields are investigated on a qualitative level. From the visual assessment of the DL prediction one can observe that the model performs best when the RSP is characterized by "iso-phase" lines with spherical symmetry. This effect was already observed in the 2D case in which the WCA loss managed to overcome this limitation. In the 3D case this effect is more pronounced and the model still struggles to break this structural symmetry despite the large number of trainable parameters and training samples. A possible explanation is given by the symmetry of the diffraction pattern used in input. The averaged spatial intensity distribution of the ensemble of training samples possess a spherical symmetry, decaying with a power low radially. As a consequence, the model replicates a similar distribution in the predicted RSP. This result is clearer when the direct output of the model, in the form of an unwrapped RSP, is inspected.

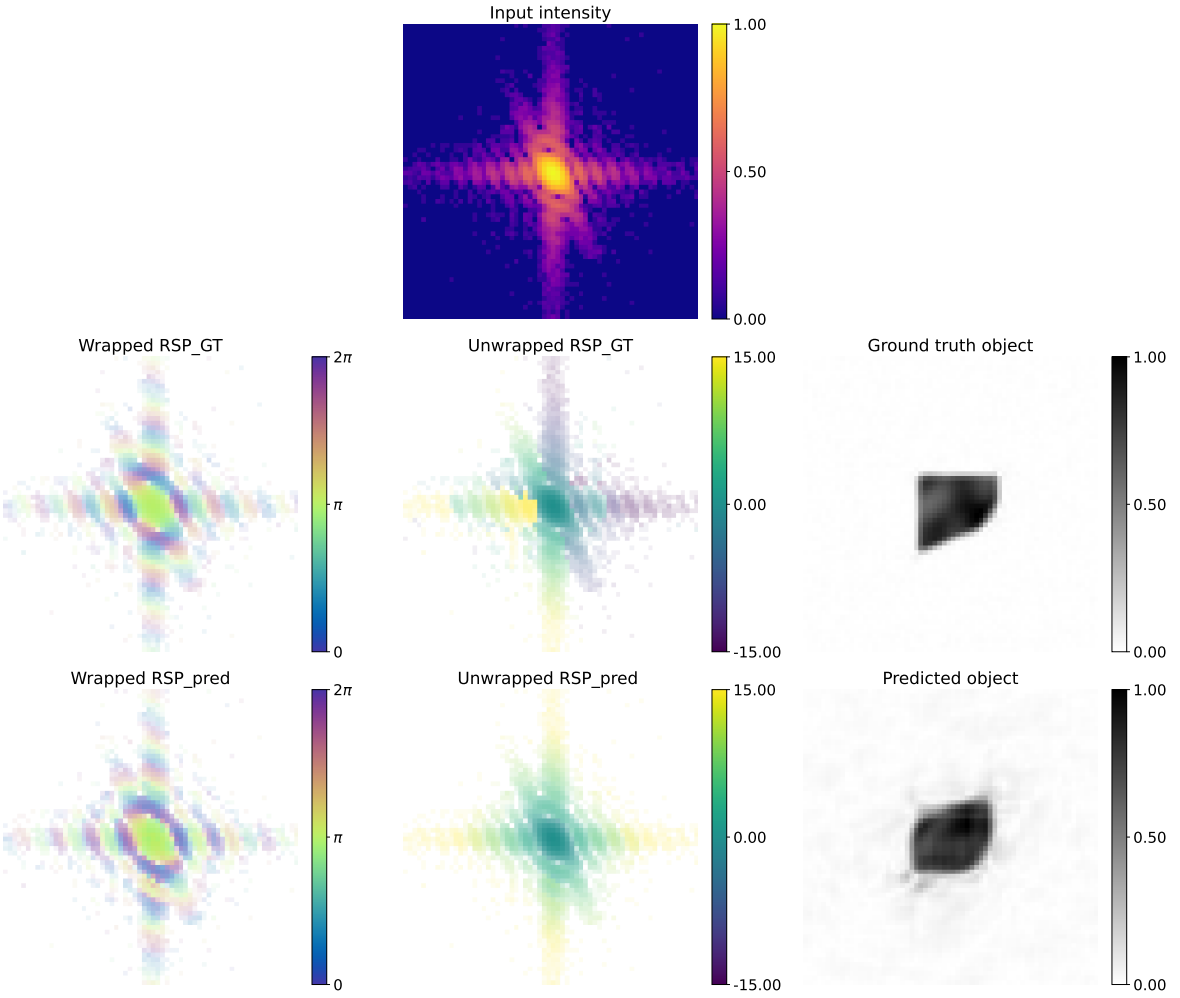


Figure 4.50: Example of DL prediction for a low strain randomly shaped particle. The non-centrosymmetriciy of the object's shape and the small strain produce a symmetric diffraction pattern with a RSP that when unwrapped possesses a non-spherical symmetry. This deceives the model that wrongly predicts a symmetric RSP, resulting in an incorrect reconstructed object.

When comparing the unwrapped phases one can observe that while the ground truth, along the main streaks, grows monotonically from one to the other end, the predicted one, which is the direct output of the model, grows isotropically along all radial directions. The consequence is a wrong result of the retrieved object as visible in Fig.4.50. It is worth noticing that the example reported in the figure is a low-strain case, meaning that the main obstacle for the model is given by the structural symmetry of the RSP rather than the strain. A support of this thesis, to the same object a stronger phase obtained with two Gaussian functions has been applied and the same test has been conducted. The results illustrated in Fig. 4.51 show that the RSP symmetry has changed to a more "spherical" one thanks to the strain. Therefore, the predicted RSP is more accurate as well as the reconstructed object's shape.

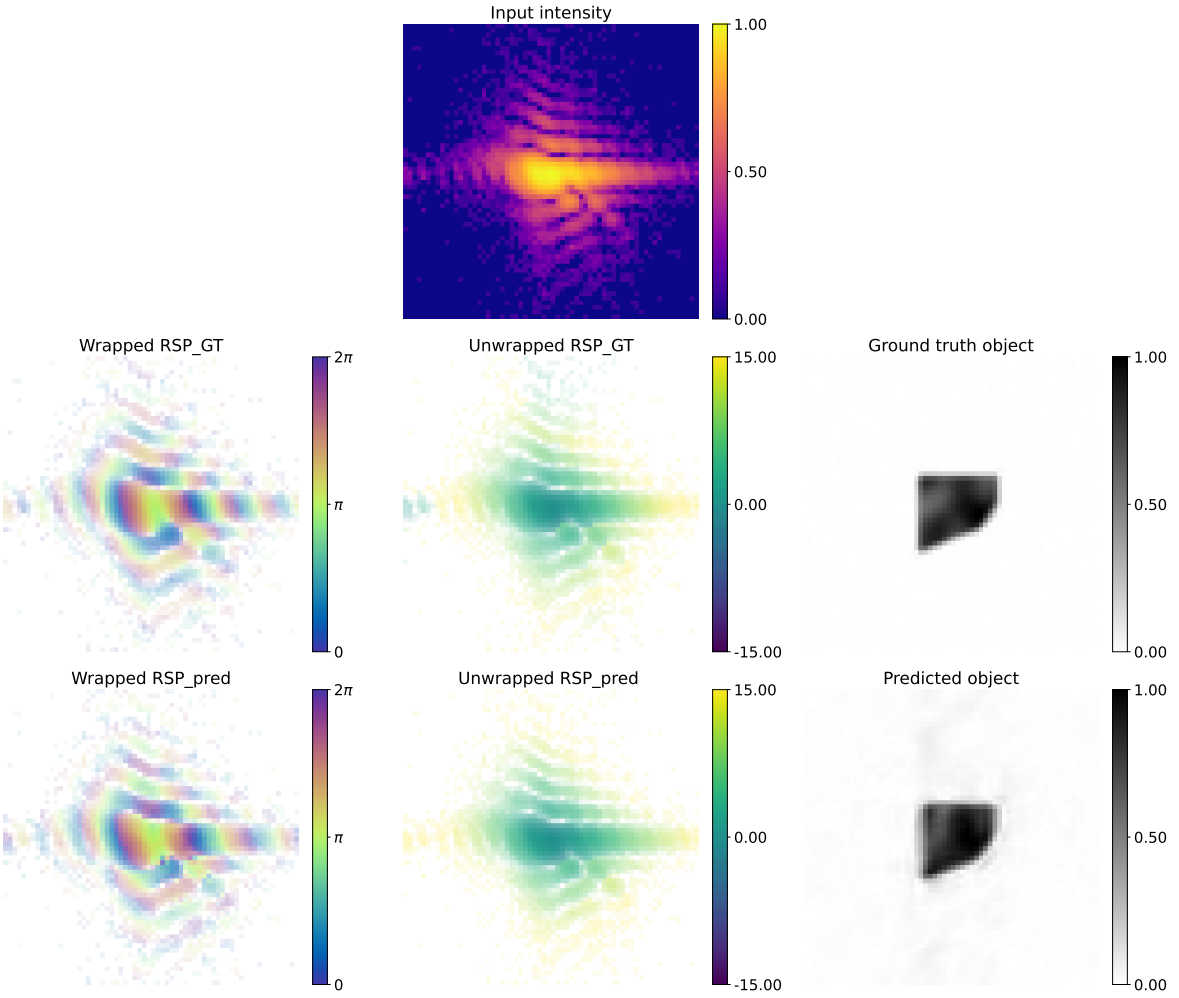


Figure 4.51: A stronger phase has been applied to the same object in Fig. 4.50. The strain field produces a more symmetric RSP that is correctly retrieved by the DL model, thus a better object is retrieved despite the higher strain.

4.11 Other model test

In this last section I would like to assess the real advantage of predicting the RSP rather than the complex real space object. In order to conduct this experiment, the most recent model in the literature of Deep Learning for BCDI Phase Retrieval was considered. In particular, the model presented by Yu and coauthors in [45], that makes use of complex convolutional layers seem to be a good candidate for the testing of our DL model. At this scope, the 2D model that was presented in the paper was transformed into a similar one for 3D data, for a total amount of 36 million trainable parameters.

The model has been trained for the same amount of epochs (60) on the same training dataset used for our DL model. Model structure and loss function were adopted like presented in the paper and the tests have been performed on the same data shown in Fig. 4.32.

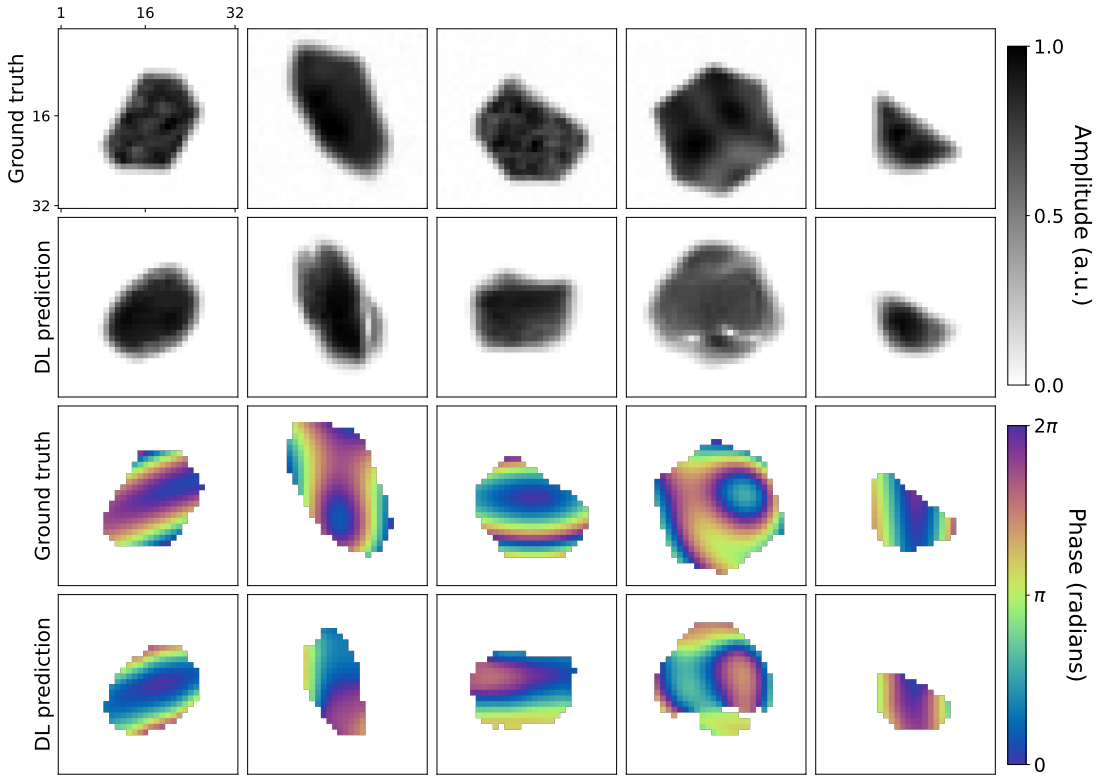


Figure 4.52: Predicted objects obtained by the 3D adaptation of the 2D Complex U-Net presented in [45]. The model returns a complex tensor that represents the real space solution. Although much less noisy than our predictions, the reconstructed objects possess incorrect shapes and phases.

The model returns objects that have a better resolution and less noise levels because of the loss function calculated in real space. However, the shape and phase of the results are not correct, thus unusable for ER refinement because misleading initial guesses. This simple example shows that the prediction of the RSP suites better the PR of BCDI patterns with convolutional neural network, especially for highly strained particles.

4.12 Conclusion

To conclude this journey through the developments, results and interpretations of the DL model for the BCDI RSP prediction the main points can be summarized as follows.

- A novel approach for the DL-based PR of BCDI patterns, focused on the prediction of the RSP has been investigated. The method proved to be advantageous since (i) it entails the inference of a single array rather than two coupled ones, (ii) it exploits the exact calculation of the IFT, directly using the diffraction pattern in the transform, rather than a learned map from the diffraction pattern and the solution in real space. (iii) it takes

advantage of the similarity between the input and the output variables, making the best use of the skip connections of the U-Net architecture. The superiority of this approach over the more conventional one found in literature has been demonstrated as discussed in 4.11.

- The training of the DL model for the prediction of the RSP can be optimized with the use of the custom WCA loss function, designed specifically for the handling of complex phases. Adopting the WCA loss not only improves the results but is also speeding up the training as it takes shorter time than a loss computed in real space. Moreover, the use of the WCA avoids any reference to the real space object during the training, opening the door for a patching approach, where the RSP is predicted from a sub-volume of diffracted intensity.

- Such designed model is trained in supervised fashion on simulated data only. This can be at first a limiting factor as it requires the simulation of big and diverse datasets. However, being the BCDI technique restricted to single crystals, the population of shapes, though being potentially infinite, is physically confined. It is thus easier for the DL model to generalize for new particle's shapes.

- The DL model trained as presented is able to perform on highly-strained particles as well, showing promising results in 2D and 3D. It has proven to be successful inverting experimental data that happened to be extremely challenging with conventional iterative methods only. This achievement marks a milestone for the BCDI community, upgrading the DL studies on BCDI PR from a *"proof of concept"* level to an actual *practical use* in the analysis of experimental data.

- The DL model proved to be compatible with iterative refinement using conventional algorithms, contributing to a faster and more robust pipeline for PR.

- The DL model exhibits certain shortcomings attributable to the intrinsic symmetry of RSP. It in fact "prefers" RSPs with a spherically symmetric distribution and the major hypothesis for this fact is that such symmetry is typical of the intensity distribution of the average diffraction pattern. Hence, the difficulty of the model to escape this bias for 3D data more than 2D ones. Additionally, it is curious to notice that this symmetry seems to be the strongest limitation for the DL model, lying above the sign symmetry and the high strain, which are both resolved when in many cases. Future investigations that elucidate the root causes of this phenomenon may yield improved performance on more heterogeneous data sets.

- At last, the substrate-induced strain imposed on Winterbottom particles often manifests a symmetry that is faithfully reproduced by the pair of Gaussian functions used to generate the phase of the training ensembles. This symmetry gives rise to a spherically symmetric RSP, which the DL model, by design, generally predicts with high accuracy.

Chapter 5

Automatic Differentiation for BCDI Phase Retrieval

In this chapter a different approach to the BCDI phase retrieval will be presented. It originated from the need to resolve those cases in which neither standard alternating algorithms, nor the DL assisted PR can succeed to converge to a satisfactory reconstruction. The developed approach differs from the alternating projections algorithms classically used for the Fourier PR, as it is formulated as minimization problem solved with gradient descent (GD). The gradients however are computed through the efficient automatic differentiation (AD) enabled by graph-based differentiable programming packages like Tensorflow and PyTorch, accelerated on GPU. For this reason one could see the AD approach as unsupervised machine learning on a single training dataset.
The GD - based optimization is fundamentally different from fixed point alternating projections. Here one could qualitatively say that if the latter switches between real and reciprocal space applying constraints in both, the former initializes a complex object and updates at each cycle its modulus and phase using the gradients, with respect to them, of the differences between the observed and calculated diffracted intensities. In this way, the knowledge on the particle can be implemented by initializing the object with some physical constraints or adding regularization terms that will drive the updates towards more reasonable solutions.

After mentioning the most relevant literature on AD, and more generally GD-based, phase retrieval for CDI, we will present our formulation of the problem and the results obtained on simulated and experimental BCDI patterns.

5.1 State of the Art

5.2 Model implementation

In an AD-driven optimization problem some trainable parameters are initialized. In the first basic formulation these trainable parameters can be the values of the voxels corresponding to the modulus m and the phase φ of the complex objects that represents the solution of the

PR problem. All of these voxels contribute to the creation of a simulated diffracted intensity pattern via the forward model $I_{calc} = |\mathcal{F}\{me^{i\varphi}\}|^2$. Subsequently, the gradients of a metric (loss function) that estimates the distance between the observed BCDI pattern I_{obs} and I_{calc} are calculated with respect to each of the trainable variables with automatic differentiation. At this point the value of each of these voxels is updated using a chosen optimizer (SGD, ADAM, etc.) and a given learning rate. The Tensorflow library allows for an easy implementation of the trainable variables and loss function and handles gradient operations with predefined methods. It is therefore straightforward to run the optimization as it follows the same structure of a deep learning model, with less trainable parameters and for a single data.

However, such simple formulation of the complex object as mere real-valued variables is not optimal for a non-linear and non-convex inverse problem such the Fourier phase retrieval. In fact, many non-physical modulus-phase configuration could yield a I_{calc} that is close to I_{obs}. The presence of these local minima is the reason why, in conventional PR, algorithms like hybrid input-output, capable of escaping them, are employed. Moreover, it was shown by Marchesini in [51] that steepest GD and even more sophisticated conjugate GD are more prone to get stuck in local minima, reason why they are not commonly utilized for Fourier PR. However, the active research field of machine learning has brought important advancements in the formulation of efficient and robust optimizers based on stochastic gradient descent with powerful features like Nesterov or adaptive momentum (ADAM [34]). These GD techniques are more robust to local minima, since the gradient is computed on mini-batches of trainable variables rather all of them (stochastic rather than classical steepest GD), and converge faster thanks to the "memory" of previous steps. Additionally, they are often wrapped into handy classes, ready to use, in Tensorflow and Pytorch libraries.

However, to facilitate the convergence the formulation of the complex object to be optimized has embedded some physical considerations that helped to restrict the solution space. First of all, both support and phase built on a 3D grid occupying half the volume of the input BCDI data to account for the oversampling ratio which has to be at least 2 in all directions to ensure invertibility. Additionally, other constraints specifically designed for the object shape and phase were considered.

5.2.1 Object's shape

The formulation of the object's shape has started considering the typical crystalline samples that are studied with the BCDI technique and the requirements the modulus of the reconstructed object need to fulfill to be considered a "good solution". Usually, successful reconstruction show a *homogeneous* modulus, sometimes quantitatively assessed through the mean-to-max metric [52] , [53], as in standard BCDI the form factor is approximated uniform across all the scattering sites. Enforcing a homogeneous modulus by construction limits the search space and helps the convergence to the solution. It follows that parametrizing the *surface* of the support, and setting to 1 the inside, is much more advantageous than optimizing the full 3D volume. This approach, already proposed by Scheinker and Pokharel in [38], also significantly reduces the number of variables to optimize.

An additional consideration is that the probed samples are crystalline, thus often *faceted* and *convex*. Therefore, one could simplify even more the construction of the object shape by building a certain amount of planes in the 3D space and obtain the support from the volume

that lies inside the intersections of all them. This would remove the possibility to have spikes or rough surfaces that might satisfy some local minimum but wouldn't represent a crystal. Moreover, with this representation the number of trainable variables would be further reduced.

According to this scheme the relevant parameters to be optimized are the angles θ and φ of the spherical coordinates and the length d of a given number N of the so-called *half-spaces*. More formally, the normals n_i for each of the N half-spaces are defined with a pair of (θ, φ) that its orientation in space (Eq. 5.1). Subsequently, only the intersection of those (x, y, z) coordinates for which the dot product with each n_i is smaller than the length d_i is considered as support (Eq. 5.2).

$$\mathbf{n}_i \;=\; \begin{pmatrix} \sin\varphi_i \cos\theta_i \\ \sin\varphi_i \sin\theta_i \\ \cos\varphi_i \end{pmatrix}, \tag{5.1}$$

$$\mathcal{S} \;=\; \bigcap_{i=1}^{N} \Big\{ \mathbf{x} = (x, y, z) \in \mathbb{R}^3 : \mathbf{n}_i \cdot \mathbf{x} \leq d_i \Big\}, \tag{5.2}$$

A schematic representation of this construction is provided by Fig. 5.1.

With this approach the user needs to provide a number of half-spaces as hyperpareter meaning that a sort of prior knowledge on the sample can be leveraged in these regards as well. However, this number doesn't have to be precisely the number of facets expected. In fact, a large N is often advised for unknown sample shape such that even roundish objects can be retrieved. In case of well faceted samples the large N is a minor problem as many n_i will be automatically aligned to the same $(\theta_i, \varphi_i, d_i)$ at the cost of some more trainable parameters.

The first drawback of this convex-hull parametrization is that concave objects can't be retrieved. However, these cases are much less frequent in typical BCDI experiment. The second limitation is that this formulation is incapable of modeling defects that would zero the contribution of the object's modulus to the diffraction pattern [54]. A correct BCDI reconstruction of particles affected by this type of defects presents "holes" inside the hull in correspondence of the defect. However, the current model cannot address this type of features as the support is by construction fully homogeneous inside the borders. Further developments of the algorithm could indeed aim at a more complete formulation of the construction of the object modulus.

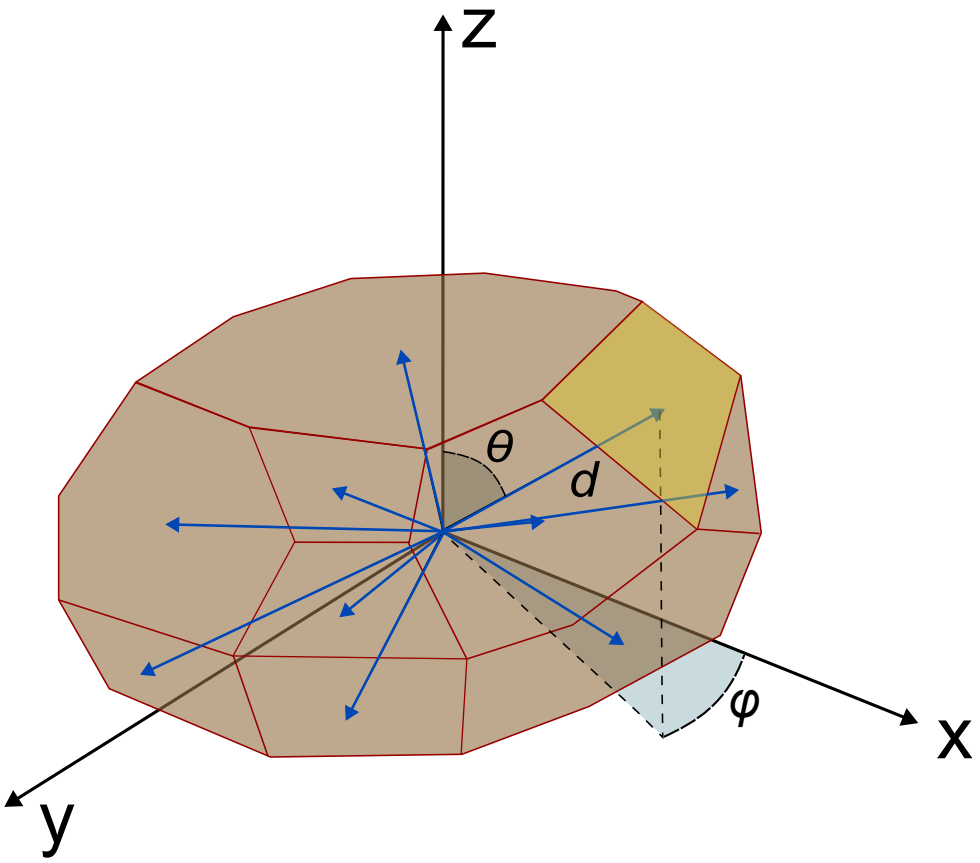


Figure 5.1: Construction of the convex hull with half-spaces expressed with spherical coordinates

The last important consideration of this parametrization is that the support $\mathcal{S}$ is sharply divided into a binary variable (1 inside and 0 outside) thus leading to differentiability problems. In fact, in such a way the gradients, essential for the support update, are not defined. For this reason $\mathcal{S}$ is first passed through a sigmoid function controlled by a hyperparameter ϵ responsible for the smoothening of the support borders. This measure can also be seen as a control of the resolution of the object. Additionally, a mildly steep sigmoid in the early stage of the optimization can function help retrieving a low resolution estimate of the support, that can be further refined adjusting the ϵ parameter.

5.2.2 Object's phase

The parametrization of the object's phase is more challenging. From a qualitative point of view, the prior knowledge that can be exploited for a tailored implementation, is limited to the awareness that a physically meaningful atomic displacement field cannot have "too many" sharp variations. This observation is translated into code by smooth functions parametrization or total variation (TV) regularization of the object's phase. While the former would enforce smoothness by construction the latter would operate adding a penalty to the data-fidelity term of the loss function for non-smooth solutions. Both approaches have been explored and are here reported.

Forcing a scalar field defined on an $L \times H \times W$ grid to exhibit smooth behavior is equivalent to seeking a sparse representation of that field—that is, to concentrating its essential information into far fewer degrees of freedom than the original $L \times H \times W$ samples. Concretely, one looks for a change of basis in which the field can be written as a linear combination of a hierarchy of modes or atoms, ordered by "importance." In a Fourier or wavelet expansion, for instance, the expansion coefficients are naturally sorted from largest (low-frequency or coarse-scale modes) to smallest (high-frequency or fine-scale modes). Retaining only the largest coefficients both compresses the data and removes rapid oscillations, yielding an inherently smoother reconstruction. Equivalently, in the matrix case a Singular Value Decomposition (SVD) identifies an orthonormal basis in which only a few singular values are nonzero; by truncating to the top singular values one obtains a low-rank — and thus smoother—approximation [55]. For higher dimensional data, this same principle underlies higher-order generalizations of the SVD—Tucker/HOSVD, CP, Tensor-Train, and T-SVD—each of which orders multilinear "modes" by their singular-value (or eigenvalue) strength, and truncating to a small subset produces both compression and smoothness [56].

In this case the Tucker decomposition was chosen, among the several possible methods, for its simplicity of implementation with the Tensorflow library and for the suitability for moderately low dimensions [57]. For a 3D tensor the Tucker decomposition is done as follows:

Considering $\varphi \in \mathbb{R}^{L \times H \times W}$ the 3D object's phase. The Tucker decomposition expresses φ as:

$$\varphi = \mathcal{G} \times_1 U^{(1)} \times_2 U^{(2)} \times_3 U^{(3)},$$

where:

- $\mathcal{G} \in \mathbb{R}^{R_1 \times R_2 \times R_3}$ is the **core tensor**,
- $U^{(1)} \in \mathbb{R}^{I \times R_1}$, $U^{(2)} \in \mathbb{R}^{J \times R_2}$, and $U^{(3)} \in \mathbb{R}^{K \times R_3}$ are the **factor matrices**,
- $\times_n$ denotes the mode-n tensor-matrix product.

In index notation, this becomes:

$$\varphi_{i,j,k} = \sum_{\alpha=1}^{R_1} \sum_{\beta=1}^{R_2} \sum_{\gamma=1}^{R_3} \mathcal{G}_{\alpha,\beta,\gamma} \cdot U^{(1)}_{i,\alpha} \cdot U^{(2)}_{j,\beta} \cdot U^{(3)}_{k,\gamma}.$$

With this formulation the parameters R_1, R_2, R_3 are set by the user and define the "storage space" in which the information required to represent φ has to be condensed. It is proven that for $R_i = L, H, W$ respectively, the tensor φ is exactly represented. However, being the goal a spare representation of the object's phase these numbers are chosen significantly smaller than any of the sizes of the array. The Tensorflow implementation of the Tucker decompostion is rather straightforward as the function `tf.einsum()` takes care of the tensor contraction.

A different approach that has been considered, leverages the TV regularization to push the algorithm towards a smooth object's phase. The full $L \times H \times W$ tensor is therefore optimized and a penalty on the sum of the absolute value of the gradients of the phase is added to the loss function. Precisely, the formula that has been used calculates the sum of the *squared*

gradients, since the square root operation, necessary to obtain the correct formula, creates problem around zero because of the infinite gradient. The final equation is therefore:

$$TV = \alpha \sum_{i=1}^{L} \sum_{j=1}^{H} \sum_{k=1}^{W} \mathcal{S}[(\varphi_i - \varphi_{i-1})^2 + \varphi_j - \varphi_{j-1})^2 + \varphi_k - \varphi_{k-1})^2] \tag{5.3}$$

where α is a hyperparameter that acts as a scaling factor, (i, j, k) are the indices running over the coordinates of the $L \times H \times W$ grid and $\mathcal{S}$ is the object support. The parameter α in this case was chosen to be assigned as a fraction, imposed by the user, of the value of the data fidelity loss. Further developments could aim at finding adaptive formulations for the magnitude of α.

5.2.3 Loss function

Another important aspect of the model is the loss function. Typically, for inverse problems there is *data fidelity* term that in this case measures the distance between I_{obs} and I_{calc} according to some metric, and other additional *regularization* terms that guide the optimization process with physical constraints.

Data fidelity: The most common and intuitive metrics are the Mean Squared Error (MSE) and the Mean Absolute Error (MAE) that evaluate the Euclidean distance between the observed and calculated intensities. Practically, because of the large dynamic range of typical BCDI data, the MAE performs better as it doesn't focus on bright pixels only, but manages to correct for lower intensity tails as well. A more faithful metric for BCDI experimental data is the Poisson Negative Log-Likelihood (P-NLLK). This metric assumes indeed the handling of count data, like the type obtained by photon counting detectors, and that the stochasticity of physical process is Poisson distributed. When summed over the full dataset, the discrepancies between calculated and observed intensities are not intended as Euclidean distances but like divergences between two probability distributions. In other words, the P-NLLK estimates the likelihood that I_{calc} belongs to the same Poisson distribution of I_{obs} [58]. Derived from the equation for the probability for Poissonian events, the formula of the averaged P-NLLK, in the form of a Kullback-Liebler divergence is:

$$\langle \mathrm{P-LLK} \rangle = \frac{2}{N_{\mathrm{obs}}} \left[\sum_{I_{\mathrm{obs}}>0} \left(I_{\mathrm{calc}} - I_{\mathrm{obs}} + I_{\mathrm{obs}} \ln \frac{I_{\mathrm{obs}}}{I_{\mathrm{calc}}} \right) + \sum_{I_{\mathrm{obs}}=0} I_{\mathrm{calc}} \right]. \tag{5.4}$$

Both the MAE and the P-NLLK have been tested on several simulated and experimental datasets and the MAE has always shown better convergence. An explanation for this unexpected result is yet to be found, but the main suspect is that the gradients calculated during the backpropagation can have instabilities because of the logarithm.

Regularizations: Beside the TV on the object's phase to ensure smoothness, another term that was considered concerns the size of the support. For a given data fidelity value, it is known that the object with the smallest support represents the optimal solution [50]. Intuitively this could be explained with the fact that there are many more object phase configuration that would combine constructive and destructive interferences to match the observed intensity in

reciprocal space. The analogous measure is the shrinkwrap algorithm [59] utilized in alternating projections algorithms. For this reason a penalty P on the size of the support can be added to the loss function with the formula:

$$P = \beta \sum_{i,j,k} \mathcal{S} \tag{5.5}$$

where β is a hyperparameter that similarly to α is chosen by the user with respect to the data fidelity loss. Both the hyperparameters can be tuned manually during the optimization, to adjust in case of need, the strength of the regularization terms. The final formula for the loss function can be ultimately expressed as:

$$L = \frac{1}{N_{\text{obs}}} \sum_{i=1}^{N_{\text{obs}}} \left| I_{\text{calc},i} \; - \; I_{\text{obs},i} \right| + \alpha TV(\varphi) + \beta P(\mathcal{S}) \tag{5.6}$$

For the optimization, a tolerance on the MAE value or a fixed number of steps can be set to stop the algorithm. Empirical observations have shown that a MAE value around 0.2 is sufficiently low for the result to be considered good. However, an additional refinement with a few iterations ($\sim$ 300) is recommended for cross-validation.

Before concluding the paragraph, it is worth highlighting that this AD implementation offers the possibility to simultaneously run multiple reconstructions in parallel, efficiently on the GPU. One can create a 4D tensor by stacking several 3D intensity data, creating therefore a batch. For each element in the batch a different initial support and phase configuration can be chosen, hence increasing the likelihood to converge to the solution.

5.3 Results

In this section two relevant results will be presented. The first examples is a highly strained Nickel particle on Sapphire substrate measured around the (111) Bragg reflection at 8.8 KeV at the ID01 beamline of the ESRF [60]. The large strain inside the particle distorts the BCDI pattern and makes the reconstruction with conventional iterative algorithms overly challenging. This dataset has been measured with the novel Bragg Coherent Modulation Imaging (BCMI) technique that yielded excellent result, presented as ground truth reference.

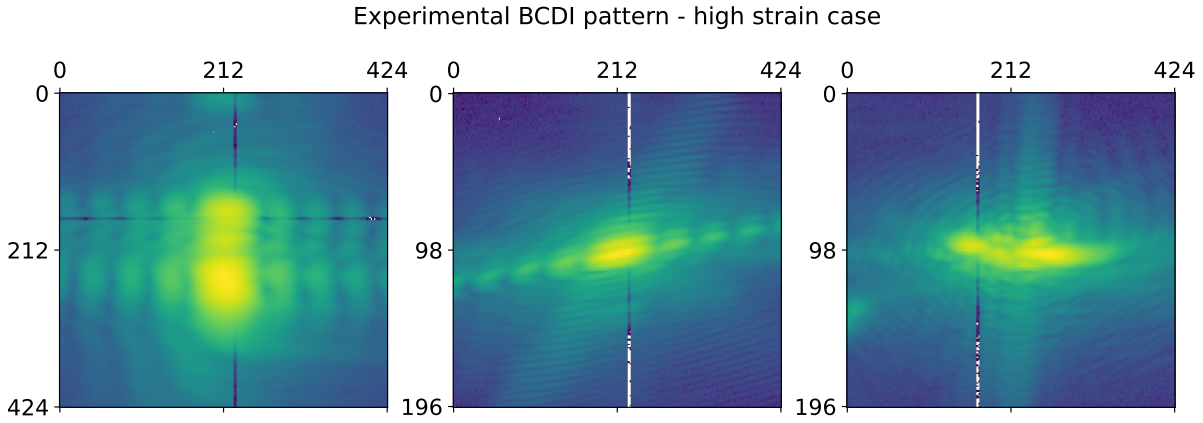


Figure 5.2: Projections along the three axes of a BCDI pattern of the highly strained Nickel particle.

In the following lines a comparison between different reconstruction methods will follow. Namely, (i) the results obtained from 40 independent runs of standard PR using PyNX software. Each run consisted of 400 HIO iterations followed by 1000 RAAR and 300 ER ones. The parameters of are displayed in Table 5.1. This method is referred in the text to as "PyNX".

Parameter	**Value**
recipe	400 HIO + 1000 RAAR + 300 ER
nb_runs	40
support_threshold	(0.15, 0.4)
smooth_width	(2, 0.5, 600)
post_expand	None
support_update_period	50
update_border_n	2
smooth_width_begin	2
smooth_width_end	0.5
support_autocorrelation_threshold	(0.09, 0.11)
update_psf	100
psf	'pseudo-voigt,0.5,0.1,10'
support_update_border_n	2
support_post_expand	'1,-2'

Table 5.1: PyNX parameter settings for standard PR.

The second method makes use of the DL model presented in the previous chapter. The large original data is firstly binned to a (196, 140, 140) shape and then cropped in a (80,90,110) shaped ROI. The DL predicted object is then interpolated back to the original size and refined with PyNX using a single run of 300 ER, with the parameters listed in Table 5.2.

Parameter	Value
recipe	300 ER
nb_runs	1
support_threshold	0.3
smooth_width	(2, 0.5, 600)
post_expand	None
obj	DL_obj
support_update_period	30
update_border_n	1
smooth_width_begin	2
smooth_width_end	0.5
update_psf	100
psf	'pseudo-voigt,0.5,0.1,10'
support_update_border_n	1
support_post_expand	'1,-1'

Table 5.2: PyNX parameter settings for the refinement after the DL prediction

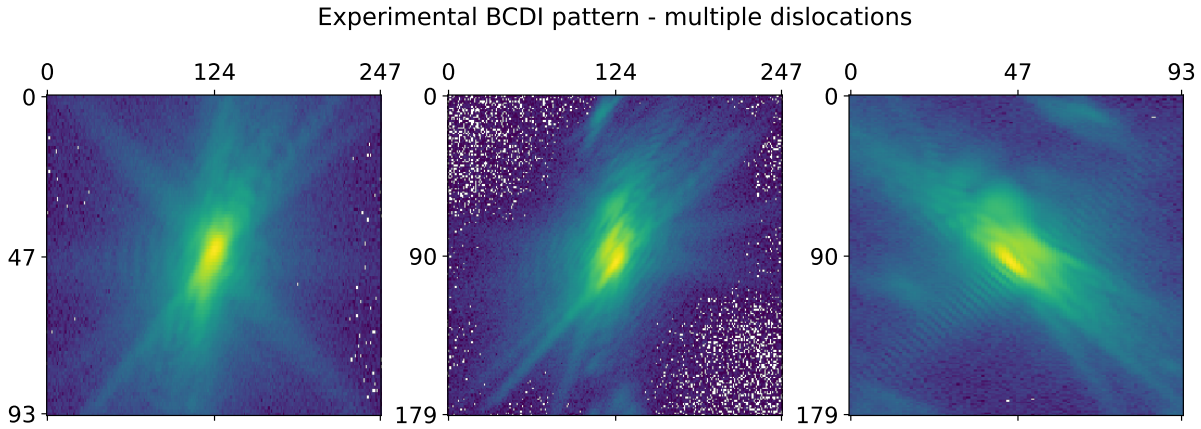


Figure 5.3: Projections along the three axes of a dataset with multiple dislocations .

5.3.1 Low-strain case

5.3.2 High-strain case

5.4 Conclusions

Chapter 6

Conclusions

Appendix A

Additional Data and Methods

Appendix B

Appendix

Bibliography

1. Sun, Y. & Singer, A. Bragg coherent diffractive imaging for defects analysis: Principles, applications, and challenges. *Chemical Physics Reviews* **5,** 031310. ISSN: 2688-4070. eprint: https://pubs.aip.org/aip/cpr/article-pdf/doi/10.1063/5.0219030/20154496/031310_1_5.0219030.pdf. https://doi.org/10.1063/5.0219030 (Sept. 2024).
2. Robinson, I. K., Vartanyants, I. A., Williams, G. J., Pfeifer, M. A. & Pitney, J. A. Reconstruction of the Shapes of Gold Nanocrystals Using Coherent X-Ray Diffraction. *Phys. Rev. Lett.* **87,** 195505. https://link.aps.org/doi/10.1103/PhysRevLett.87.195505 (19 Oct. 2001).
3. Favre-Nicolin, V. *et al.* Analysis of strain and stacking faults in single nanowires using Bragg coherent diffraction imaging. *New Journal of Physics* **12,** 035013. https://dx.doi.org/10.1088/1367-2630/12/3/035013 (Mar. 2010).
4. Singer, A. *et al.* Nucleation of dislocations and their dynamics in layered oxide cathode materials during battery charging. *Nature Energy* **3,** 641–647. ISSN: 2058-7546. https://doi.org/10.1038/s41560-018-0184-2 (Aug. 2018).
5. Serban, D. *et al.* Imaging in-operando LiCoO2 nanocrystallites with Bragg coherent X-ray diffraction. *Communications Chemistry* **7,** 243. ISSN: 2399-3669. https://doi.org/10.1038/s42004-024-01331-y (2024).
6. 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).
7. Grünewald, T. A., Liebi, M. & Birkedal, H. Crossing length scales: X-ray approaches to studying the structure of biological materials. *IUCrJ* **11,** 708–722. https://doi.org/10.1107/S2052252524007838 (Sept. 2024).
8. Li, P. *et al.* 4th generation synchrotron source boosts crystalline imaging at the nanoscale. *Light: Science & Applications* **11,** 73. ISSN: 2047-7538. https://doi.org/10.1038/s41377-022-00758-z (Mar. 2022).
9. Leake, S. J. *et al.* The Nanodiffraction beamline ID01/ESRF: a microscope for imaging strain and structure. *Journal of Synchrotron Radiation* **26,** 571–584. https://doi.org/10.1107/S160057751900078X (Mar. 2019).
10. 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).
11. 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).
12. 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).
13. 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).

14. 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).

15. 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.

16. 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.

17. Mairal, J., Elad, M. & Sapiro, G. Sparse Representation for Color Image Restoration. *IEEE Transactions on Image Processing* **17,** 53–69 (2008).

18. Allene, C. & Paragios, N. *Image Renaissance Using Discrete Optimization* in *18th International Conference on Pattern Recognition (ICPR'06)* **3** (2006), 631–634.

19. Goodfellow, I. J. *et al. Generative Adversarial Networks* 2014. arXiv: 1406.2661 [stat.ML]. https://arxiv.org/abs/1406.2661.

20. Jiang, Y., Xu, J., Yang, B., Xu, J. & Zhu, J. Image Inpainting Based on Generative Adversarial Networks. *IEEE Access* **8,** 22884–22892 (2020).

21. 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).

22. 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/.

23. 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.

24. 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).

25. 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).

26. 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).

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

28. 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).

29. 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).

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

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

32. 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).

33. 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).

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

35. 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.

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

37. 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).

38. 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).

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

40. 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).

41. 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).

42. 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).

43. 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).

44. 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).

45. 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).

46. Guizar-Sicairos, M. & Fienup, J. R. *Understanding the twin-image problem in phase retrieval* (2012).

47. Zhang, W., Wan, Y., Zhuang, Z. & Sun, J. What is Wrong with End-to-End Learning for Phase Retrieval? arXiv: 2403.15448. http://arxiv.org/abs/2403.15448 (Mar. 17, 2024).

48. Miao, J., Sayre, D. & Chapman, H. N. Phase retrieval from the magnitude of the Fourier transforms of nonperiodic objects. *J. Opt. Soc. Am. A* **15,** 1662–1669. https://opg.optica.org/josaa/abstract.cfm?URI=josaa-15-6-1662 (June 1998).

49. Shi, X. *et al. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting* 2015. arXiv: 1506.04214 [cs.CV]. https://arxiv.org/abs/1506.04214.

50. Favre-Nicolin, V., Leake, S. & Chushkin, Y. Free log-likelihood as an unbiased metric for coherent diffraction imaging. *Scientific Reports* **10.** Publisher: Nature Research. ISSN: 20452322. arXiv: 1904.07056 (Dec. 1, 2020).

51. Marchesini, S. A unified evaluation of iterative projection algorithms for phase retrieval. *Review of Scientific Instruments* **78.** ISSN: 00346748. arXiv: physics/0603201 (2007).

52. Frisch, M. L. *et al.* Unraveling the synergistic effects of Cu–Ag tandem catalysts during electrochemical CO_2 reduction using nanofocused X-ray probes. *Nature Communications* **14.** Published: 2023-11-29, 7833. ISSN: 2041-1723. https://doi.org/10.1038/s41467-023-43693-2 (2023).

53. Grimes, M. *et al.* Capturing Catalyst Strain Dynamics during Operando CO Oxidation. *ACS Nano* **18.** Published: 2024-07-30, 19608–19617. ISSN: 1936-0851. https://doi.org/10.1021/acsnano.4c04127 (2024).

54. Favre-Nicolin, V. *et al.* Analysis of strain and stacking faults in single nanowires using Bragg coherent diffraction imaging. *New Journal of Physics* **12.** Publisher: Institute of Physics Publishing. ISSN: 13672630 (Mar. 31, 2010).

55. Golub, G. H. & Van Loan, C. F. *Matrix Computations* 3rd. ISBN: 0-8018-5414-8 (Johns Hopkins University Press, Baltimore, MD, 1996).

56. Kolda, T. G. & Bader, B. W. Tensor Decompositions and Applications. *SIAM Review* **51,** 455–500. eprint: https://doi.org/10.1137/07070111X. https://doi.org/10.1137/07070111X (2009).

57. Oseledets, I. V. Tensor-Train Decomposition. *SIAM Journal on Scientific Computing* **33,** 2295–2317. eprint: https://doi.org/10.1137/090752286. https://doi.org/10.1137/090752286 (2011).

58. Thibault, P. & Guizar-Sicairos, M. Maximum-likelihood refinement for coherent diffractive imaging. *New Journal of Physics* **14,** 063004. https://dx.doi.org/10.1088/1367-2630/14/6/063004 (June 2012).

59. Marchesini, S. *et al.* X-ray image reconstruction from a diffraction pattern alone. *Phys. Rev. B* **68,** 140101. https://link.aps.org/doi/10.1103/PhysRevB.68.140101 (14 Oct. 2003).

60. Bellec, E., Leake, S., Richard, M.-I. & Zhao, J. *Bragg Coherent Modulation Imaging for Highly Strained Nanocrystal* Dataset (European Synchrotron Radiation Facility, 2027). https://doi.org/10.15151/ESRF-ES-1507879918.

Annexes

Appendix A

Appendix