Computational imaging through deep learning

by

Shuai Li

B.E., University of Electronic Science and Technology of China (2011)
M.E., Zhejiang University (2014)

Submitted to the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2019

Author .....
Department of Mechanical Engineering
May 3, 2019

Certified by .....
George Barbastathis
Professor
Thesis Supervisor

Accepted by .....
Nicolas G. Hadjiconstantinou
Chairman, Department Committee on Graduate Theses

Computational imaging through deep learning

by

Shuai Li

Submitted to the Department of Mechanical Engineering
on May 3, 2019, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

Computational imaging (CI) is a class of imaging systems that uses inverse algorithms to recover an unknown object from the physical measurement. Traditional inverse algorithms in CI obtain an estimate of the object by minimizing the Tikhonov functional, which requires explicit formulations of the forward operator of the physical system, as well as the prior knowledge about the class of objects being imaged.

In recent years, machine learning architectures, and deep learning (DL) in particular, have attracted increasing attentions from CI researchers. Unlike traditional inverse algorithms in CI, DL approach learns both the forward operator and the objects' prior implicitly from training examples. Therefore, it is especially attractive when the forward imaging model is uncertain (e.g. imaging through random scattering media), or the prior about the class of objects is difficult to be expressed analytically (e.g. natural images).

In this thesis, the application of DL approaches in two different CI scenarios are investigated: imaging through a glass diffuser and quantitative phase retrieval (QPR), where an Imaging through Diffuser Network (IDiffNet) and a Phase Extraction Neural Network (PhENN) are experimentally demonstrated, respectively.

This thesis also studies the influences of the two main factors that determine the performance of a trained neural network: network architecture (connectivity, network depth, etc) and training example quality (spatial frequency content in particular). Motivated by the analysis of the latter factor, two novel approaches, spectral pre-modulation approach and Learning Synthesis by DNN (LS-DNN) method, are successively proposed to improve the visual qualities of the network outputs.

Finally, the LS-DNN enhanced PhENN is applied to a phase microscope to recover the phase of a red blood cell (RBC) sample. Furthermore, through simulation of the learned weak object transfer function (WOTF) and experiment on a star-like phase target, we demonstrate that our network has indeed learned the correct physical model rather than doing something trivial as pattern matching.

Thesis Supervisor: George Barbastathis
Title: Professor

Acknowledgments

First of all, I would like to express my sincere gratitude to my PhD supervisor, Prof. George Barbastathis. Joining his group is one of the most correct decisions that I have ever made. To me, his is not only a research advisor, but also a life mentor. His guidance, support and encouragement in all these five years are really important to me. I learned from him how to become a good scientist. It is my fortune to have such a great advisor.

I would like to thank my thesis committee members: Prof. Nicholas Fang, Prof. Themistoklis Sapsis, Prof. Petros Koumoutsakos and Prof. Berthold Horn. Thank you for all the valuable comments and sincere assistance, which helped a lot in improving the quality of this thesis.

I would also like to thank all my colleagues at the 3D Optical Systems group: Ayan Sinha, Justin Lee, Yi Liu, Nilu Zhao, Yunhui Zhu, Chih-Hung Max Hsieh, Jeong-Gil Kim, Adam Pan, Kelli Xu, Zhengyun Zhang, Mo Deng, Alexandre Goy, Kwabena Arthur and so on. I would not have made it without their help.

Also, I would like to thank all my friends at MIT, especially Yichen Shen, Yiming Mo, Ziyu Wang, Zhong Yi Wan, Huifeng Du, Han Yin, Lingbo Zhang, Tianyi Chen, Yue Guan, Qiong Zhang and Shuo Han. Thank you very much for accompanying me. I will never forget the time that we spend together.

Lastly and specially, I would like to thank my parents, Hui Dong and Gang Li, for their love and support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Computational imaging

Computational imaging (CI) is a class of imaging systems that delivers estimate of an unknown object, based on the physical measurement and prior knowledge about the class of objects being imaged [1, 2], as shown in Fig.1-1.

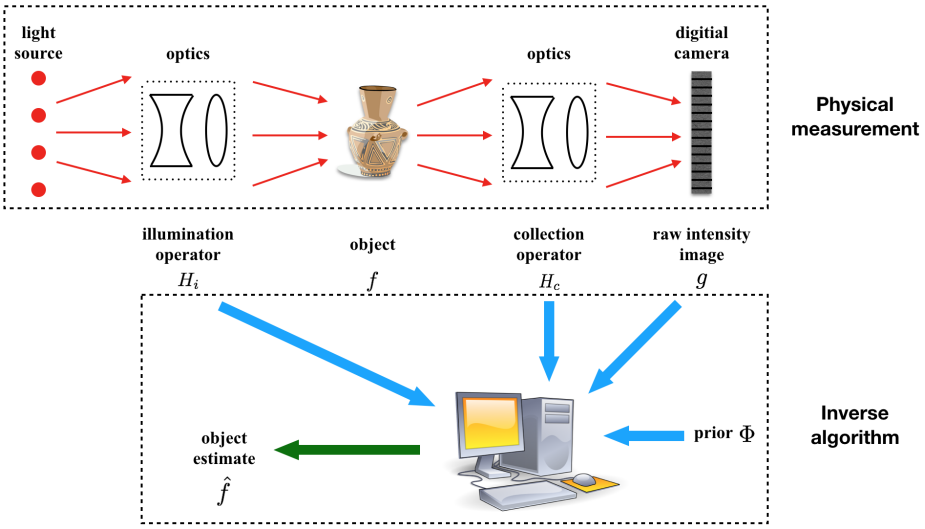

Figure 1-1: General computational imaging (CI) system.

Generally, It consists two parts: a physical measurement and the corresponding inverse algorithm. In the physical measurement part, the light emitted from the illumination source is shaped by the illumination optics according to the operator H_i

before reaching the object f. The radiation from the object is then shaped by the collection optics according to the operator H_c and forms a raw intensity image g on the digital camera. Here, we refer g as the measurement. After that, g is feed into the inverse algorithm, which takes into account the operators H_i, H_c together with the prior knowledge Φ about the class of objects being imaged, to obtain the object estimate $\hat{f}$. More detailed descriptions about the physical measurements and the inverse algorithms in CI can be found below in Section 1.1.1 and 1.1.2, respectively.

The motivation of performing CI rather than using the raw intensity measurement directly is that information may be hidden in the latter case due to the limitation of hardware, making the raw measurement uninterpretable. Therefore, post-computation is required. Actually, in CI, we no longer have the requirement that the raw image itself should be readily interpretable (e.g. satisfy some metrics of conformity and fidelity with respect to the object). As shown in Chapter 2, in the case of imaging through a strong glass diffuser, the raw measurement is a speckle pattern, in which no salient feature of the object can be observed. This aspect of CI largely reduces the complexity of the hardware, making the physical setup more compact and cost-effective.

In practice, due to the ill-posedness of the forward operator, the reconstruction from a raw image will become non-unique and very sensitive to noise. In these conditions, CI would have to rely on the prior to do the inversion. The analytical formulation of the prior is called the regularizer, which mathematically represents the patterns that commonly exist in the class of objects. These priors will help to restrict the space of possible solutions to the inverse problem by excluding the solutions that do not match object attributes known beforehand, thus ameliorating non-uniqueness.

1.1.1 Physical measurements

Referring to Fig. 1-1, f denotes the true representation of the object that the imaging system's user want to recover. For example, in the case of imaging through scattering media (see Chapter 2), f is the light reflectance of the object; while in the case of quantitative phase retrieval (see Chapter 3), f is the distribution of the phase

retardation caused by the object.

The optical field immediately after the object can be represented as $H_i f$, where H_i is the illumination operator. A number of ingenious strategies have been devised to design H_i that improve the imaging problem's condition, most famously by using nonlinear optics [3, 4] or stimulated emission [5]. Restricting oneself to linear optics, structured and coded illumination [6, 7, 8, 9, 10, 11, 12, 13] are effective strategies which modulate object information onto better-behaved spatial frequencies. Another interesting instance is ghost imaging [14, 15] , where the object is illuminated rapidly by a sequence of speckle patterns.

The quantity $H_i f$ passes through the collection optics and forms a raw intensity image g on the digital camera, which can be expressed as: $g = H_c H_i f$, with H_c denotes the collection operator. Usually, H_c is also specifically designed to achieve the desirable performance. Notable examples includes confocal microscopy [16, 17], transport of intensity equation (TIE) method [18, 19], lensless imaging [20, 21] and etc.

For brevity, we denote $H = H_i H_c$ as the forward operator of the entire physical measurement, then the raw intensity image can be expressed as:

$$g = Hf. \tag{1.1}$$

Nevertheless, Eq.1.1 is noiseless, which is impractical. In realistic scenarios, different sources of noise are, more or less, exist due to multiple causes: thermal agitation of the electrons inside the sensor, environmental disturbances, the discrete nature of electric charge, etc. Two typical types of noise are additive white Gaussian noise (AWGN) and Poisson noise, whose statistical models are shown as follows:

$$\text{AWGN:} \quad g = Hf + n; \tag{1.2}$$

$$\text{Poisson:} \quad g = \mathcal{P}\left\{Hf\right\}. \tag{1.3}$$

Here, AWGN is denoted as n; and $\mathcal{P}$ is a Poisson random variable generator with

the expected value equal to its argument.

1.1.2 Inverse algorithms

Given the explicit formulations of the forward operator H and the regularizer Φ, the reconstruction in CI may be obtained by minimizing the Tikhonov functional [22, 23]:

$$\hat{f} = \underset{f}{\operatorname{argmin}} \left\{ \|Hf - g\|_2^2 + \alpha\Phi(f) \right\}. \tag{1.4}$$

The first term is known as the fitness term and it is minimized when the result of applying the forward operator on the estimate matches the physical measurement. Nevertheless, minimizing the fitness term only is not enough. Due to the ill-posedness of the forward operator H, the noise in the measurement will be amplified, generating artifacts in the reconstruction. The situation becomes even worse when H is singular (which is always the case in practice), since then the estimate that will match the fitness term become non-unique, i.e. an infinite number of solutions $\hat{f}$ would satisfy $Hf = g$. Therefore, the regularization term $\Phi(f)$ becomes really important here. By minimizing the weighted sum of the fitness term and the regularizer, we are actually introducing a competition between the two terms, which effectively forces the estimate landing on the position that satisfies both the physical measurement and the prior knowledge of the object. The regularizing coefficient α controls the relative contribution of the two terms in the competition and is always chosen empirically, based on our relative belief in the measurement *vs.* prior knowledge.

The regularizer Φ has to be chosen in the way such that $\Phi(f)$ has a small value when f matches the prior knowledge of the class of objects. The very first regularizer, L^2 norm, was proposed by Tikhonov [22, 23]:

$$\Phi(f) = \|f\|_2^2. \tag{1.5}$$

The motivation of using L^2 norm as the regularizer is to reduce the energy of the reconstructed signal so as to mitigate the noise amplification effect caused by the

ill-posedness of H.

When H is a linear operator, the solution to Eq.1.4 with the L^2 regularizer can be readily obtained as:

$$\hat{f} = (H^t H + \alpha I)^{-1} H^t g, \tag{1.6}$$

where H^t is the transpose of H and I denotes the identity matrix.

Notably, Wiener filter [24, 25], which is derived from the matched filtering principle under the assumptions of additive white Gaussian noise and white Gaussian and uncorrelated to noise statistics for the signal, is actually a special case of Eq.1.6 with $\alpha = 1/\text{SNR}$, where SNR denotes the signal to noise ratio.

In the last two decades, compressive sensing [26, 27, 28, 29, 30], which utilizes regularizers that prompting sparsity, has becoming an increasingly promising technique to solve the inverse problems in CI. The main idea of compressive sensing is that most of the signals are compressible, i.e. there exists a non-singular transformation S that can transform the signal to the domain where the representation of the signal is sparse. Mathematically, we have,

$$s = Sf, \tag{1.7}$$

and $\|s\|_0$ is small.

Then, Eq.1.4 can be rewritten as,

$$\hat{s} = \underset{s}{\text{argmin}} \left\{ \left\| HS^{-1}s - g \right\|_2^2 + \alpha \|s\|_0 \right\}. \tag{1.8}$$

Here, S^{-1} is the inverse of the sparse transformation S.

However, the above optimization problem (Eq.1.8) is not easy to be solved directly since the L^0 norm is not amenable to optimizers. Fortunately, it has been discovered later that using L^1 norm has the equivalent effect [31, 32]. In this case, the solution to the inverse problem becomes,

$$\hat{s} = \underset{s}{\text{argmin}} \left\{ \left\| HS^{-1}s - g \right\|_2^2 + \alpha \|s\|_1 \right\}, \tag{1.9}$$

which can be solved by several existing numeral algorithms such as Lasso [33], ISTA [34, 35, 36], TwIST [37], FISTA [38] and Adam [39] and etc. Then, the object estimate can be readily obtained as,

$$\hat{f} = S^{-1}\hat{s}. \tag{1.10}$$

The choice of S is obviously crucial to the performance of compressive sensing. Popular candidates includes wavelet transforms [40, 28, 41], non-linear diffusion operators [42, 43, 44, 45] and sparse dictionaries [46, 47, 48, 49].

1.2 Deep learning

Machine learning is a technique that enables computational systems to improve by learning the mapping between the desired output and the features (representation) of the input from experience and data. Generally, machine learning approaches can be divided into three groups [50]: classic machine learning (CML), representation learning (RL) and deep learning (DL), depending on the way how the features of the input is obtained. In CML, the features are hand-designed [51]; in RL, features are learned from data through a shallow autoencoder [52]; and in DL, features are learned from data through a deep, multi-layered architecture [53]. Compared with CML and RL, the features learned in DL are simpler and more high-level (abstract), which makes the mapping from the features to the output easier to learn. Apart from providing the right representation for the data, another strength of DL is that the multi-layered architecture also offer greater power in learning the mapping since the later layers can refer back to the outputs of earlier layers.

The specific architecture used in DL is the neural network, which is a multi-layered computational geometry. Generally speaking, a neural network contains at least three layers: one input layer, one output layer and at least one hidden layer. Except for the input layer, the units (neurons) in every other layers are connected to the units in the previous layers according to some weights. Then, the input value to a specific neuron i is actually the weighted sum of the output values of all the neurons in the previous layers that i is connecting to. These architectures are called neural networks because

they are loosely inspired by neuroscience and they can be represented by composing together several different functions, e.g. each layer may be a function.

Now, I want to introduce two typical DL architectures: fully-connected neural network (FCNN) and convolutional neural network (CNN).

1.2.1 Fully-connected neural networks

Fully-connected neural network (FCNN), also known as multi-layer perceptron (MLP) [54, 55], is the most common architecture used in DL. Fig. 1-2 shows the structure of a three-layer FCNN (one hidden layer). Here, "fully-connected" means that every neuron in the hidden layers is connected to all the neurons in the preceding layer and the succeeding layer. In FCNN, the input is mapped to the output in a feed-forward fashion. As shown in Fig. 1-2, for an individual neuron, its input (activation) a is the weighted sum of the output values of all the in the previous layer and can expressed as:

$$a = \sum_{i=1}^{n} w_i x_i + b. \tag{1.11}$$

Here, n is the number of neurons in the previous layer, x_i is the output of the ith neuron, w_i is the weight and b denotes the bias term.

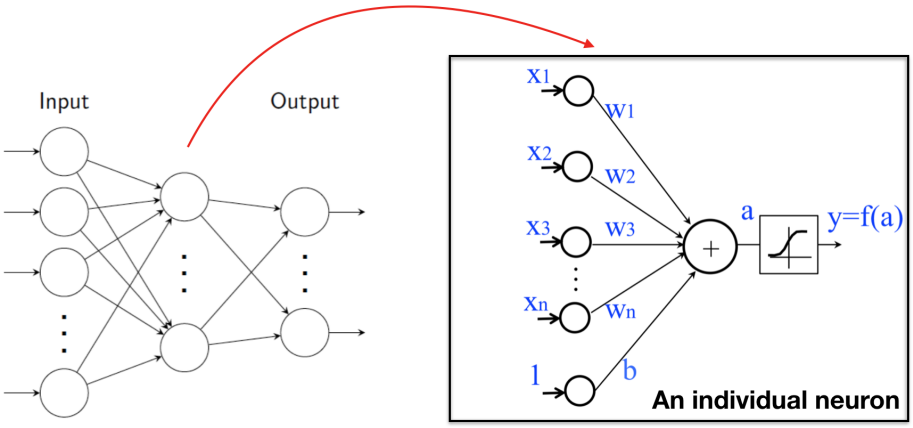

Figure 1-2: Fully-connected neural networks. [This figure is obtained from the slides of the Advances in Computer Vision (6.869) course at MIT taught by Bill Freeman and Antonio Torralba.]

After receiving the activation a, the output of the neuron y is given by,

$$y = f(a) = f\left(\sum_{i=1}^{n} w_i x_i + b\right), \tag{1.12}$$

where $f(\cdot)$ is a point-wise non-linear activation function.

Possible choices of the non-linear activation function $f(\cdot)$ include sigmoid [56], tanh [57], rectified linear unit (ReLU) [58] and leaky ReLU [59]. These functions are plotted in Fig. 1-3 and their respective formulations are:

$$\text{Sigmoid:} \quad f(a) = \frac{1}{1+e^{-a}}; \tag{1.13}$$

$$\text{Tanh:} \quad f(a) = \frac{e^{a}-e^{-a}}{e^{a}+e^{-a}}; \tag{1.14}$$

$$\text{ReLU:} \quad f(a) = \max(0, a); \tag{1.15}$$

$$\text{Leaky ReLU:} \quad f(a) = \begin{cases} a, & a > 0 \\ \alpha a, & a \leq 0 \end{cases}. \tag{1.16}$$

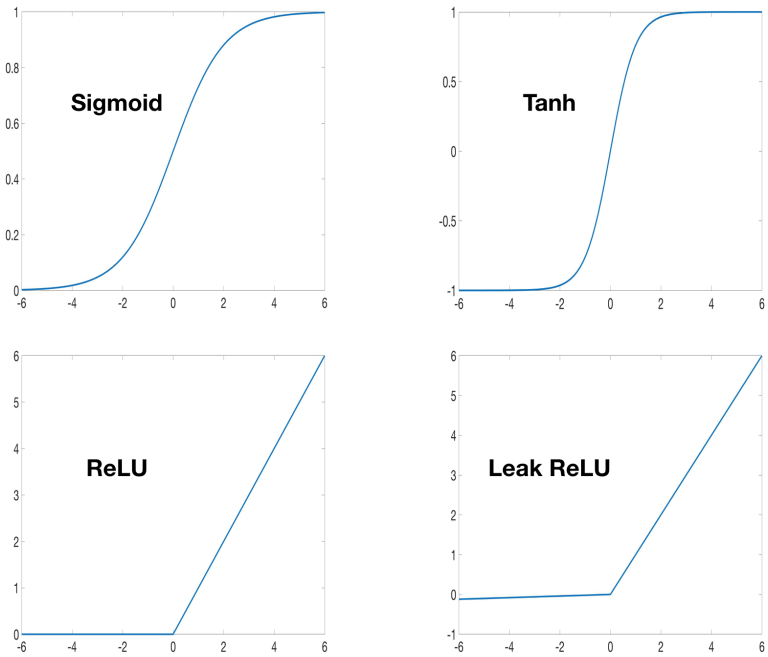

Figure 1-3: Different non-linear activation functions.

In Eq.1.16, the coefficient α is usually very small (e.g. 0.02 in Fig. 1-3). In some cases, its value can also be learned during the training [60]. Among these functions, ReLU and leaky ReLU are the most widely used ones nowadays since they do not have the problems of stagnating derivatives as sigmoid and they lead to faster convergence as compared with tanh [53].

1.2.2 Convolutional neural networks

Convolutional neural networks [61], evolved from the idea of Neocognitron [62], are a specialized kind of neural network that have been tremendously successful in practical applications dealing with time-series data and image data. As shown in Fig. 1-4, in CNN, each neuron is only connected to several nearby neurons in the previous layer and each neuron shares the same set of connecting weights. Mathematically, the input to an individual neuron i, which is connected to $2N+1$ nearby neurons in the previous layer, can be expressed as,

$$a_i = \sum_{n=-N}^{N} w_n x_{i-n} + b, \tag{1.17}$$

where x_i is the output of the ith neuron in the previous layer and w_i is the set of trainable weights. Here, for simplicity, we assume the input to be 1-D. If the input is 2-D, then the input will be the weighted sum of the outputs of $(2M+1) \times (2N+1)$ nearby neurons in the previous layer.

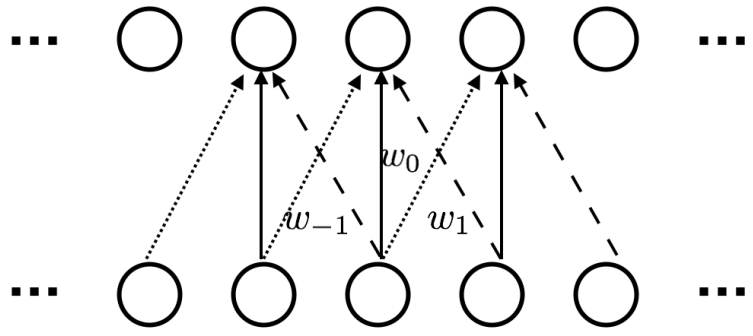

Figure 1-4: Convolutional neural networks.

Eq. 1.17 is actually a convolution operation (with a bias), and that is why this architecture is named as a "convolutional" neural network. Since performing convolution in the spatial domain is equivalent to filtering in the spatial frequency domain, the set of weights w_i is also called a filter. In Fig. 1-4, we plot the simplest case where only one filter is applied. In practice, however, multiple filters are used at each convolutional layer and the final output of a convolutional layer is obtained by stacking the output of each filter along the "channel" dimension. At the next layer, each channel of the input is summed up by some learnable weights while doing the convolution defined as Eq. 1.17.

The motivation of doing convolution in neural networks is three folds. First, dependencies are usually local, i.e. only several nearby neurons will contribute; second, convolution preserves translational equivariance (known as shift invariance in the optics community), which always exists in practice; third, implementing sparse connections and using the same set of weights reduce the complexity of the neural network, which is an effective way to prevent overfitting.

Apart from the convolution operation and the non-linear activation functions (usually ReLU), there are two other specialized operations in CNN: pooling and batch normalization. Pooling is the operation that reduces the lateral dimension of the input. The idea is to increase the receptive fields and also introduce invariance to small transformations [63]. Two common choices for pooling function are max pooling and average pooling, which are defined as (assume the downsampling rate to be 2),

$$\text{max pooling:} \quad y_k = \max\left(x_{2k-1}, x_{2k}\right); \tag{1.18}$$

$$\text{average pooling:} \quad y_k = \frac{x_{2k-1} + x_{2k}}{2}. \tag{1.19}$$

where x_k and y_k denotes the value of the kth neuron in the input and output of the pooling layer, respectively.

Batch normalization is a layer in the neural network that does normalization across data, which can accelerate the training process by reducing internal covariate shift [64]. The procedure of batch normalization is shown in Fig. 1-5.

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Algorithm 1: Batch Normalizing Transform, applied to activation x over a mini-batch.

Figure 1-5: Batch normalization. [This figure is adopted from [64].]

1.2.3 Training neural networks for computational imaging

The optimal values of the weights in the neural network is *learned* from data through an optimization routine. This process is denoted as the "training" of a neural network. Specifically, given a set of data, also known as training examples, the functional need to be optimized is a measure of the distance between the desired output and the actual output of the neural network. We refer this measure as the training loss functions. Some popular training loss functions that are being used nowadays include mean squared error (MSE), mean absolute error (MAE), cross entropy, etc. Hence, during the training, the weights in the neural network are tuned so as to make the network output match the desired output in the sense defined by the loss function. Similar to the Tikhonov functional shown in Eq. 1.4, regularizers such as L^2 norm of the weights [65] are usually added to the optimization functional during the training to prevent overfitting. The optimization is numerically solved by gradient descent based algorithms [66], where the gradients are computed via a procedure called back-propagation [67].

Since the neural network training process in DL shares a lot in common with

solving the optimization problem in CI, it becomes natural to apply DL techniques to CI. The principle is shown in Fig. 1-6, where the raw intensity measurement g is fed into a trained neural network to generate the object estimate $\hat{f}$. Given a specific CI scenario, the procedures of training such a neural network can be described in the following three steps: (1) Training data collection. We obtain a database of known objects and their corresponding raw intensity measurements through numerical simulations or physical experiments. (2) Network architecture selection. We decide the appropriate neural network architecture to be used (e.g. connectivity, pooling strategy, depth, etc). These hyper-parameters can be determined empirically or with the help of the validation data. While using the validation dataset, we divide the entire training dataset into two subsets: one set is used for training the network and the other set (named as the validation data) is used to evaluate the performance of the trained network, thus determining the optimal architecture. (3) Training the network using the collected training examples and the selected network architecture.

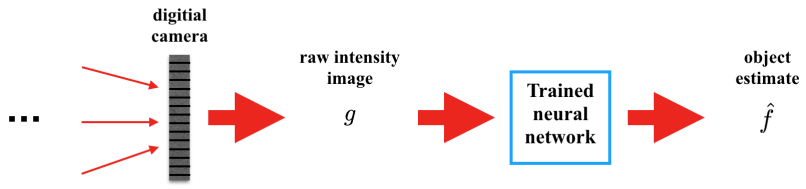

Figure 1-6: Principle of applying deep learning (DL) to computation imaging (CI).

Compared with those traditional CI inverse algorithms described in Section 1.1.2, the advantage of using DL techniques is three folds: (1) No explicit formulation of the forward operator is required. There are several scenarios where the image formation process is not entirely known, e.g. imaging through a glass diffuser (See Chapter 2) or some proprietary elements such as microscope objectives are used (See Chapter 7). In addition, the misalignment and random noise in practice will also make the theoretical forward model inaccurate. (2) Prior knowledge about the class of the objects is not required. It is not always possible to explicitly express the prior as a regularizer. (3) Fast inference speed. Traditional inverse algorithms solve each measurement iteratively. In other words, every new measurement has to go through

the entire iterative process, which is very time consuming in practice. In contrast, once a neural network is trained, all the weights are fixed. Each new measurement only have to go through the network once, which is much faster. This aspect can be very beneficial in real-time applications. Nevertheless, DL techniques also have their own drawback. They usually require a large training dataset, which can sometimes be very difficult to obtain. The comparisons between traditional CI algorithms and DL techniques are also shown in Table. 1.1.

Table 1.1: Comparisons between traditional CI algorithms and DL techniques.

	Traditional CI algorithms	**DL techniques**
Knowledge about the forward operator H	Required	Not required
Knowledge about the prior Φ	Required	Not required
Inference speed	Slow	Fast
Big data	Not required	Required

1.3 Outline of the thesis

The structure of the thesis is as follows: in Chapter 2, DL techniques are applied to solve one specific CI problem: imaging through scattering media. An imaging through diffuser network (IDiffNet) is proposed and demonstrated to be effective in reconstructing the unknown object hidden behind a glass diffuser. The effects of scattering strength, object complexity (i.e., the object priors that the neural networks must learn), and choice of the loss function for training are analyzed. The spatial resolution as well as the degree of shift invariance for the IDiffNets trained in different conditions are also tested.

In Chapter 3, DL techniques are applied to solve another typical CI problem: quantitative phase retrieval. A Phase Extraction Neural Network (PhENN) is proposed. We experimentally build and test a lensless imaging system where a phase extraction neural network (PhENN) is trained to recover unknown phase objects given their propagated intensity diffraction patterns. The robustness of PhENN to axial,

lateral and rotational shifts are also analyzed.

The performance of a trained neural network is influenced by two factors: network architecture and training example quality. In Chapter 4, we first investigate the influence of the network architecture, including the network depth, waist size, presence of skip connections and choice of the training loss function.

The influence of training example quality is studied in Chapter 5. Specifically, we investigate how the spatial frequency content of the training examples influence the spatial resolution of the neural network. Based on this study, we propose a spectral pre-modulation approach which is demonstrated to be effective in improving the spatial resolution of PhENN.

Motivated by the spectral pre-modulation approach, Chapter 6 propose the Learning Synthesis by DNN (LS-DNN) method, which effectively manage and synthesize different spectral bands so as to improve the visual qualities of recovered images in CI problems.

Different from all the previous chapters, where the objects are generated using a spatial light modulator (SLM), Chapter 7 shows that PhENN is able to image those actual phase objects. Specifically, a red blood cell (RBC) sample, is reconstructed through a microscope system, by using the optimal neural network architecture found in Chapter 4 and the LS-DNN method described in Chapter 6. In addition, we will also show a demonstration that PhENN has indeed learned the correct physical model, rather than doing something trivial such as pattern matching.

Finally, Chapter 8 states conclusions and future work.

Chapter 2

Imaging through scattering media using IDiffNet

2.1 Introduction

Imaging through random media [68, 69] remains one of the most useful as well as challenging topics in computational optics. This is because scattering impedes information extraction from the wavefront in two distinct, albeit related ways. First, light scattered at angles outside the system's Numerical Aperture is lost; second, the relative phases among spatial frequencies that pass are scrambled—convolved with the diffuser's own response. In most cases, the random medium is not known or it is unaffordable to characterize it completely. Even if the random medium and, hence, the convolution kernel are known entirely, deconvolution is highly ill-posed and prone to noise-induced artifacts.

Therefore, the strategy to recover the information, to the degree possible, must be two-pronged: first, to characterize the medium as well as possible so that at least errors in the deconvolution due to incomplete knowledge of the medium's response may be mitigated; and, second, to exploit additional *a priori* knowledge about the class of objects being imaged so that the inverse problem's solution space is reduced and spurious solutions are excluded. These two strategies are summarized by the Tikhonov optimization functional [Eq.1.4].

Several approaches characterize the random medium efficiently. One method is to measure the transmission matrix of the medium by interferometry or wavefront sensing [70, 71, 72]. Alternatively, one may utilize the angular memory effect in speckle correlation [73, 74, 75, 76, 77]. The angular memory principle states that rotating the incident beam over small angles does not change the resulting speckle pattern but only translates it over a small distance [78, 79]. In this case, computing the autocorrelation of the output intensity and deconvolving it by the autocorrelation function of the speckles, which is a sharply peaked function [80], results in the autocorrelation of the input field. Then, the object is recovered from its own autocorrelation using the Gerchberg-Saxton-Fienup (GSF) algorithm [81, 82] with additional prior constraints.

The regularizer Φ expresses prior knowledge by penalizing unacceptable objects so the optimization is prohibited from landing onto them; alternatively, the priors expressed by the regularizer can be thought of as helping to resolve non-uniqueness due to the ill-posed nature of the forward operator. In the case of strong scattering, it is common to say that information "is lost" because it is convolved into the high spatial frequencies escaping the system aperture. (The opposite may also be possible: a cleverly designed scattering medium may bring high-spatial frequency information back into the aperture, by convolving it to low spatial frequencies [7, 83, 84]) However, the prior may help to recover the missing information by enforcing properties such as edge sharpness or, more generally, sparsity, positivity, etc. During the past two decades, thanks to efforts by Grenander [85], Candés [27], and Brady [86], the use of sparsity priors was popularized and proved to be effective in a number of contexts including random media. For example, Liu *et al* successfully recovered the 3D positions of multiple LEDs embedded in turbid scattering media by taking phase-space measurements and imposing the $\mathcal{L}1$ sparsity prior [87].

Instead of establishing H and Φ independently and explicitly from measurements and prior knowledge, an alternative approach is to *learn* both operators simultaneously through examples of objects imaged through the random medium. To our knowledge, the first instance when this strategy was put forth was by Horisaki [88]. In that paper, a Support Vector Regression (SVR) learning architecture was used to

learn the scatterer and the prior of faces being imaged through. The approach was effective in that the SVR learned correctly to reconstruct face objects; it also elucidated the generalization limitations of SVRs, which are shallow fully-connected two-layer architectures: for example, when presented with non-face objects the SVR would still respond with one of its learned faces as a reconstruction. A deeper fully-connected architecture in the same learning scheme has been proposed recently [89].

In this chapter, we propose for the first time, to our knowledge, two innovations in the use of machine learning for imaging through scatter: the first is the use of the convolutional neural network (CNN) architecture [90] and the second the use of Negative Pearson Correlation Coefficient (NPCC) as loss function. Different from fully-connected network architectures, in the CNN each neuron is only connected to a few nearby neurons in the previous layer, and the same set of weights is used for every neuron. The fewer number of connections and weights reduces the complexity of the CNN architecture and makes convolutional layers relatively cheap in terms of memory needed. Moreover, overfitting is less of a problem, resulting in better generalization.

These two observations have further implications: first, due to the reduced memory requirement, we can tackle original objects of space-bandwidth product (SBP) 128×128, higher than previously reported [88, 89]. Second, the use of the convolutional architecture is counter-intuitive because the scatterer may not be shift invariant. Indeed, in Figure 2-2 we show that it is not. It may seem justified, therefore, to worry whether the reduced memory and anti-overfitting benefits of CNN may be outweighed. However, we found that the inverse estimate obtained by the CNN does in fact learn to compensate for the scatterer's shift variance, as shown in Figure 2-14.

To characterize IDiffNet response, we conducted training and testing with well-calibrated diffusers of known grit size and well-calibrated intensity objects produced by a spatial light modulator. We also examined a large set of databases, including classes of objects with naturally embedded sparsity (e.g. handwritten characters or digits). These experiments enabled us to precisely quantify when IDiffNet requires strong sparsity constraints to become effective, as function of diffuser severity (the smaller the grit size, the more ill-posed the inverse problem becomes.)

The adoption of NPCC instead of the more commonly used Mean Absolute Error (MAE) as loss function for training IDiffNet was an additional enabling factor in obtaining high-SBP image reconstructions through strong scatter. We compared the performance of these two loss functions under different imaging conditions and with different training databases determining the object priors that the networks learn and showed that NPCC is preferable for cases of relatively sparse objects (e.g. characters) and strong scatter. Lastly, we probed the interior of our trained IDiffNets through the well-established test of Maximally-Activated Patterns (MAPs) [91] and compared with standard denoising networks to eliminate the possibility that IDiffNet might be acting trivially instead of having learnt anything about the diffuser and the objects' priors.

2.2 Computational imaging system architecture

The optical configuration that we consider in this chapter is shown in Fig. 2-1. Light from a He-Ne laser source (Thorlabs, HNL210L, 632.8nm) is transmitted through a spatial filter, which consists of a microscope objective (Newport, M-60X, 0.85NA) and a pinhole aperture ($D = 5\mu$m). After being collimated by the lens ($f = 150$mm), the light is reflected by a mirror and then passes through a linear polarizer, followed by a beam splitter. A spatial light modulator (Holoeye, LC-R 720, reflective) is placed normally incident to the transmitted light and acts as a pixel-wise intensity object. The SLM pixel size is $20 \times 20\mu\text{m}^2$ and number of pixels is 1280×768, out of which the central 512×512 portion only is used in the experiments. The SLM-modulated light is then reflected by the beam splitter and passes through a linear polarization analyzer before being scattered by a glass diffuser. A telescopic imaging system is built after the glass diffuser to image the SLM onto a CMOS camera (Basler, A504k), which has a pixel size of $12 \times 12\mu\text{m}^2$. In order to match the pixel size of the CMOS with that of the SLM, we built the telescope using two lenses L_1 and L_2 of focal lengths: $f_1 = 250mm$ and $f_2 = 150mm$. As a result, the telescope magnifies the object by a factor of 0.6, which is consistent with the ratio between the pixel sizes of

the CMOS and SLM. The total number of pixels on the CMOS is 1280×1024, but we only crop the central 512×512 square for processing; thus, the number of pixels measured by the CMOS camera, as well as their size, match 1:1 the object pixels at the SLM. Images recorded by the CMOS camera are then processed on an Intel i7 CPU. The neural network computations are performed on a GTX1080 graphics card (NVIDIA).

The modulation performance of the SLM depends on the orientations of the polarizer and analyzer. Here, we implement the cross polarization arrangement to achieve a high intensity modulation contrast. Specifically, we set the incident beam to be linearly polarized along the horizontal direction and also set the linear polarization analyzer to be oriented along the vertical direction. We experimentally calibrate the correspondence between the 8-bit grayscale input images projected onto the SLM and intensity modulation values of SLM (see Appendix Section A.1). We find that in this arrangement, the intensity modulation of the SLM follows a monotonic relationship with respect to assigned pixel value and a maximum intensity modulation ratio of ~ 17 can be achieved. At the same time, the SLM also introduces phase modulation which is correlated with the intensity modulation due to the optical anisotropy of the liquid crystal molecules. The phase depth is $\sim 0.6\pi$. Fortunately, the influence of this phase modulation is negligible in the formation of the speckle images that we captured in this system; detailed demonstration can be found in Appendix Section A.1. Therefore, we are justified in treating this SLM as a pure-intensity object.

As shown in Fig. 2-1(b), the glass diffuser is inserted at a distance z_d in front of the lens L1. Here, we approximate the glass diffuser as a thin mask whose amplitude transmittance is $t(x_1, y_1)$. In this case, a forward model can be derived to relate the optical field at the detector plane $g'(x', y')$ to the optical field at the object plane $f'(x, y)$ (constant terms have been neglected) [92]:

$$g'(x',y') = \left\{ e^{\frac{-i\pi f_1^2}{\lambda(f_1-z_d)f_2^2}(x'^2+y'^2)} \cdot \int\int dxdy \left[f'(x,y) e^{\frac{i\pi}{\lambda(f_1-z_d)}(x^2+y^2)} \cdot T\left(\frac{x+f_1x'/f_2}{\lambda(f_1-z_d)}, \frac{y+f_1y'/f_2}{\lambda(f_1-z_d)} \right) \right] \right\} * \left[\frac{J_1(\frac{2\pi R}{\lambda f_2}\sqrt{x'^2+y'^2})}{\sqrt{x'^2+y'^2}} \right] \tag{2.1}$$

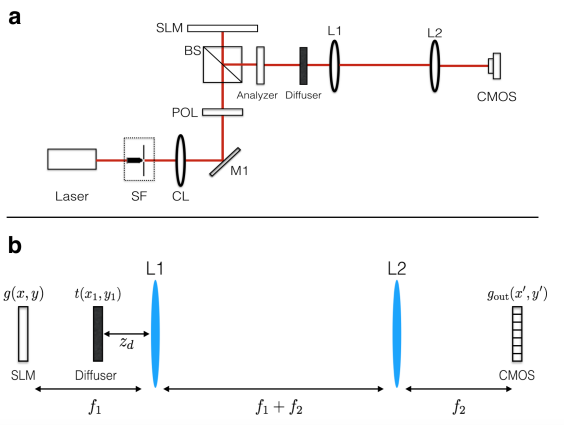

Figure 2-1: Optical configuration. (a) Experimental arrangement. SF: spatial filter; CL: collimating lens; M: mirror; POL: linear polarizer; BS: beam splitter; SLM: spatial light modulator. (b) Detail of the telescopic imaging system.

where λ is the light wavelength, R the radius of the lens L_2 and $J_1(\cdot)$ denotes the first-order Bessel function of the first kind. T is the Fourier spectrum of the diffuser: $T(u,v) = \int\int dx_1 dy_1 \left[t(x_1,y_1)e^{-i2\pi(x_1u+y_1v)}\right]$. Here, $*$ denotes the convolution product and the last term in the convolution accounts for the influence of the finite aperture size of the lenses.

We model the diffuser transmittance $t(x_1,y_1)$ as a pure-phase random mask, *i.e.* $t(x_1,y_1) = \exp\left[\frac{i2\pi\Delta n}{\lambda}D(x_1,y_1)\right]$, where $D(x_1,y_1)$ is the random height of the diffuser surface and Δn is the difference between the refractive indices of the diffuser and the surrounding air ($\Delta n \approx 0.52$ for glass diffusers). The random surface height $D(x_1,y_1)$ can be modeled as [93]:

$$D(x,y) = W(x,y) * K(\sigma). \tag{2.2}$$

Here, $W(x,y)$ is a set of random height values chosen according to the normal distribution at each discrete sample location (x,y), i.e. $W \sim N(\mu,\sigma_0)$; and $K(\sigma)$ is a zero-mean Gaussian smoothing kernel having full-width half-maximum (FWHM) value of σ.

The values of μ, σ_0 and σ are determined by the grit size of the chosen glass diffuser [94]. In this chapter, we use two glass diffusers of different grit size: 600-grit (Thorlabs, DG10-600-MD) and 220-grit (Edmund, 45-653). Using these values in Eqs. (2.1) and (2.2), we simulate the point spread function (PSF) of our imaging system

as shown in Fig. 2-2, with a point source at the center of the object plane as input. We can see that the PSF for the 600-grit diffuser has a sharp peak at the center, while the PSF for the 220-grit diffuser spreads more widely. This indicates that the 220-grit diffuser scatters the light much more strongly than the 600-grit diffuser.

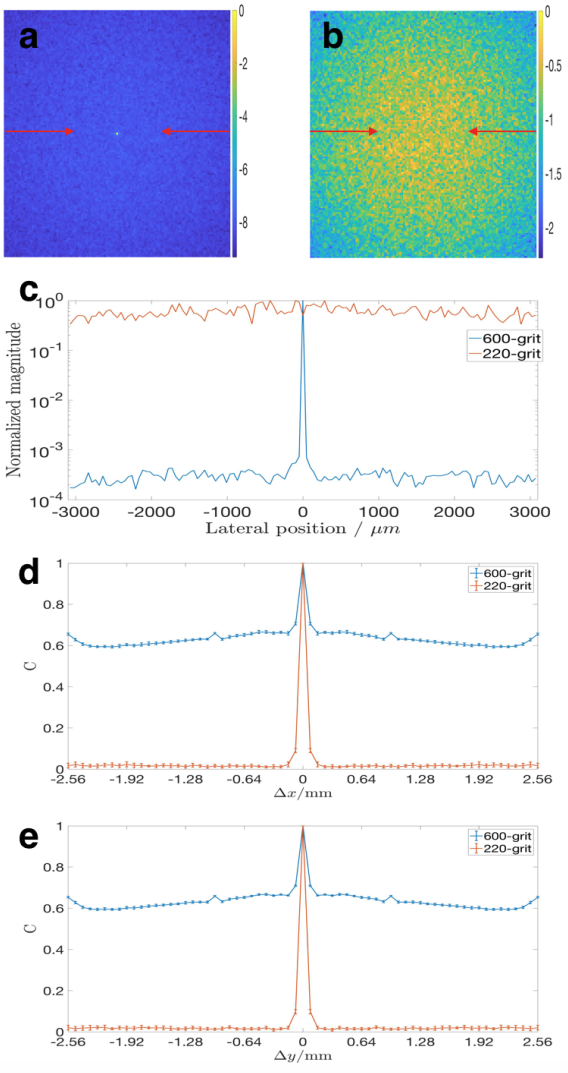

Figure 2-2: Point spread functions (PSFs) and degree of shift variance of the imaging system. (a) PSF for the 600-grit diffuser: $\mu = 16\mu$m, $\sigma_0 = 5\mu$m, $\sigma = 4\mu$m. (b) PSF for the 220-grit diffuser: $\mu = 63\mu$m, $\sigma_0 = 14\mu$m, $\sigma = 15.75\mu$m. (c) Comparison of the profiles of the two PSFs alone the lines indicated by the red arrows in (a) and (b). (d) Degree of shift variance along the x direction ($\Delta y = 0$). (e) Degree of shift variance along the y direction ($\Delta x = 0$). Other simulation parameters are set to be the same as the actual experiment: $z_d = 15$mm, $R = 12.7$mm and $\lambda = 632.8$nm. All the PSF plots are in logarithmic scale.

It is important to emphasize that, due to the existence of the diffuser, the imaging system is no longer shift-invariant. As can be seen in Eq. (2.1), the optical field at the detector plane g_{out} can not be expressed as a convolution of the object g and a shift-invariant PSF term. The degree of shift variance may be compared using the PSF correlation function

$$\begin{aligned} &C(\Delta x, \Delta y) \\ &= \iiiint \big\langle h(x', y'; x, y) h(x', y'; x + \Delta x, y + \Delta y) \big\rangle dx dy dx' dy'. \end{aligned} \tag{2.3}$$

Here, $h(x', y'; x, y)$ denotes the PSF on the detector plane (x', y') due to a point source in the object plane at location (x, y). Δx and Δy are the shifts in the object plane along x and y direction, respectively, and $\langle \cdot \rangle$ denotes the ensemble average over many simulated realizations of the diffuser. To make the comparison between different values of Δx, Δy possible, we normalized $h(.,.)$ to have zero mean and standard deviation equal to one.

Two slices of the PSF correlation function along the Δx and Δy directions, each for 10 random realizations of the simulated diffuser, are shown in Fig. 2-2d and Fig. 2-2e, respectively, for the two grit sizes. As expected, in the 600-grit case, where scattering is weak, the shifted PSFs are more correlated than those in the 220-grit case. In both cases, the degree of correlation between the shifted PSFs decreases as the shift becomes larger. In addition, the degree of shift variance along the x direction is almost identical to that along the y direction.

We may also represent equation (2.1) in terms of a forward operator H': $g'(x', y') = H' f'(x, y)$. When the object is pure-intensity, *i.e.* $f'(x, y) = \sqrt{f(x, y)}$, the relationship between the raw intensity captured at the detector plane $g(x', y')$ and the object intensity $f(x, y)$ can also be represented in terms of another forward operator H: $g(x', y') = Hf(x, y) = [SH_g Sr] f(x, y)$. Here, S denotes the modulus square operator and Sr denotes the square root operator. Then, in order to reconstruct the intensity

distribution of the object, we have to formulate an inverse operator H^{inv} such that

$$\hat{f}(x, y) = H^{\text{inv}} g(x', y') \tag{2.4}$$

where $\hat{f}(x, y)$ is an acceptable estimate of the intensity object.

Due to the randomness of H, it is difficult to obtain its explicit form and do the inversion accordingly; prior works referenced in Section 2.1 employed measurements of the scattering matrix to obtain H approximately. Here, we instead use IDiffNet, a deep neural network (DNN) trained to the underlying inverse mapping given a set of training data. IDiffNet uses the densely connected convolutional network (DenseNet) architecture [95], where each layer connects to every other layer within the same block in a feed-forward fashion. Compared with conventional convolutional networks, DenseNets have more direct connections between the layers, which strengthens feature propagation, encourages feature reuse and substantially reduces the number of parameters. Therefore, DenseNets have better generalization capability.

A diagram of IDiffNet is shown in Fig. 2-3. The input to IDiffNet is the speckle pattern captured by the CMOS. It first passes through a dilated convolutional layer with filter size 5×5 and dilation rate 2 and is then successively decimated by 6 dense and downsampling transition blocks. After transmitting through another dense block, it successively passes through 6 dense and upsampling transition blocks and an additional upsampling transition layer. Finally, the signals pass through a standard convolutional layer with filter size 1×1 and the estimate of the object is produced. This is the "encoder-decoder network" architecture [96, 97], where the dense and downsampling transition blocks serve as encoder to extract the feature maps from the input patterns, and the dense and upsampling transition blocks are served as decoder to perform pixel-wise regression. Due to the scattering by the glass diffusers, the intensity at one pixel of the image plane is influenced by several nearby pixels at the object plane. Therefore, we use dilated convolutions with dilation rate 2 and a filter size of 5×5, in all our dense blocks so as to increase the receptive field of the convolution filters. In addition, we also use skip connections [98] to pass high

frequency information learnt in the initial layers down the network towards the output reconstruction. Additional details about the architecture and training of IDiffNet are provided in Appendix Section B.1.

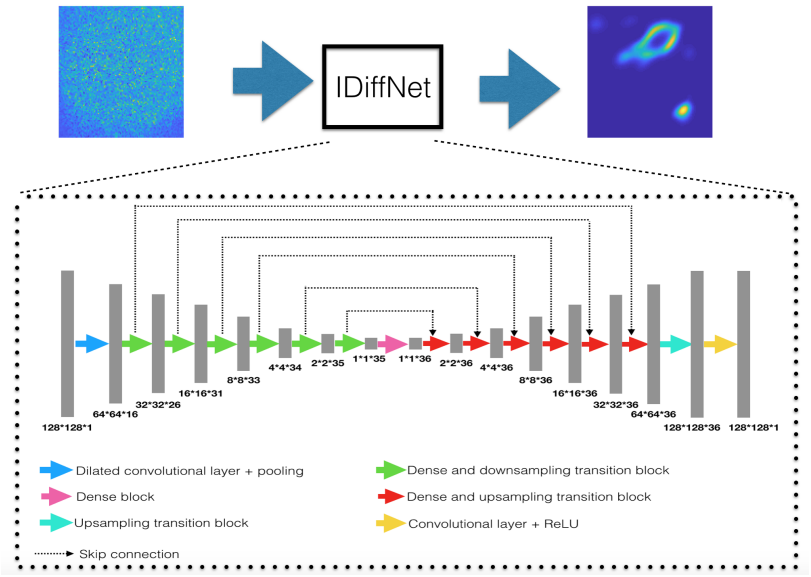

Figure 2-3: IDiffNet, our densely connected neural network that images through diffuse media.

2.3 Results and network analysis

Our experiment consists two phases: training and testing. During the training process, we randomly choose image samples from a training database. The space bandwidth product of the original images are all 128×128 and we magnify each image by a factor of 4 before uploading to the SLM. The corresponding speckle patterns are captured by the CMOS. As mentioned in Section 2.2, we only crop the central 512×512 square of the CMOS. We further downsample the captured speckle patterns by a factor of 4 and subtract from them a reference speckle pattern, which is obtained by uploading to the SLM a uniform image with all pixels equal to zero. The purpose of this subtraction operation is to eliminate the background noise on the CMOS and also to better extract differences between speckle patterns resulting from different objects.

After the subtraction operation, we feed the resulting speckle patterns into IDiffNet for training. In this way, the input and output signal dimensions are both 128×128. We collected data from six separate experiment runs: each time we used training inputs from one of the three different databases: Faces-LFW [99], ImageNet [100] or MNIST [101] and inserted one of the two glass diffusers that we have into the imaging system. Each of our training dataset consists of 10,000 object-speckle pattern pairs. These data were used to train six separate IDiffNets for evaluation. In the testing process, we sample disjoint examples from the same database (Faces-LFW, ImageNet or MNIST) and other databases such as Characters [102], CIFAR [103] and Faces-ATT [104, 105]. Altogether, 450 examples are used in the test dataset, including 50 Characters, 40 Faces-ATT, 60 CIFAR, 100 MNIST, 100 Faces-LFW and 100 ImageNet. We upload these test examples to the SLM and capture their corresponding speckle patterns using the same glass diffuser as the training phase. We then input these speckle patterns to our trained IDiffNet and compare the output to the ground truth.

In training the IDiffNets, we use two different loss functions and compare their performances. The first loss function that we consider is the mean absolute error (MAE), is defined as:

$$\text{MAE} = \frac{1}{bwh} \sum_{k=1}^{b} \sum_{i=1}^{w} \sum_{j=1}^{h} |\hat{f}_k(i,j) - f_k(i,j)|, \tag{2.5}$$

Here, f and $\hat{f}$ are the true object and the object estimated by the neural network, respectively; (i, j) denotes the pixels and k is the training example label; w, h are the width and height of the object and b is the batch size.

The qualitative and quantitative reconstruction results when using MAE as the loss function are shown in Fig. 2-4 and 2-5, respectively. From Fig. 2-4, we find that, generally speaking, IDiffNet's reconstruction performance for the 600-grit diffuser is better than that for the 220-grit diffuser. High quality reconstructions are achieved for the 600-grit diffuser when IDiffNets are trained on Faces-LFW (column iv) and ImageNet (column v). For the 220-grit diffuser, the best reconstruction is obtained

when IDiffNet is trained on the ImageNet database (column ix). The recovered images are close to the low-pass filtered version of the original image, where we can visualize the general shape (salient features) but the high frequency features are missing. This result is expected since the scattering caused by the 220-grit diffuser is much stronger than that of the 600-grit diffuser, as we had already deduced from Fig. 2-2. As a result, we can still visualize some features of the object in the raw intensity image captured in the 600-grit diffuser case. By contrast, what we capture in the 220-grit diffuser case looks indistinguishable from pure speckle, without any object details visible. This means we should expect it to be much more difficult for IDiffNet to do the inversion.

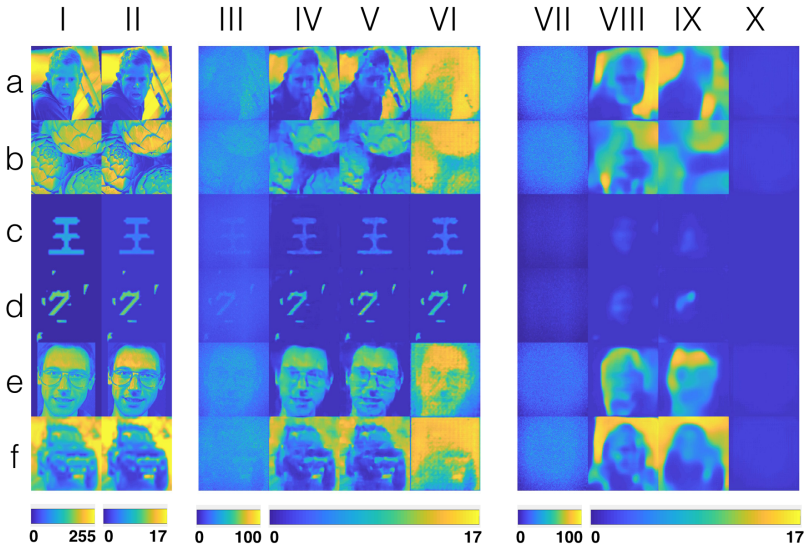

Figure 2-4: Qualitative analysis of IDiffNet trained using MAE as the loss function. (i) Ground truth pixel value inputs to the SLM. (ii) Corresponding intensity images calibrated by SLM response curve. (iii) Raw intensity images captured by CMOS detector for 600-grit glass diffuser. (iv) IDiffNet reconstruction from raw images when trained using Faces-LFW dataset [99]. (v) IDiffNet reconstruction when trained using ImageNet dataset [100]. (vi) IDiffNet reconstruction when trained using MNIST dataset [101]. Columns (vii-x) follow the same sequence as (iii-vi) but in these sets the diffuser used is 220-grit. Rows (a-f) correspond to the dataset from which the test image is drawn: (a) Faces-LFW, (b) ImageNet, (c) Characters [102], (d) MNIST, (e) Faces-ATT [104, 105], (f) CIFAR [103], respectively.

Noticeable from Fig. 2-4 is that when IDiffNet is trained on MNIST for the 220-grit diffuser (column x), all the reconstructions seem to be uniform. This is due to

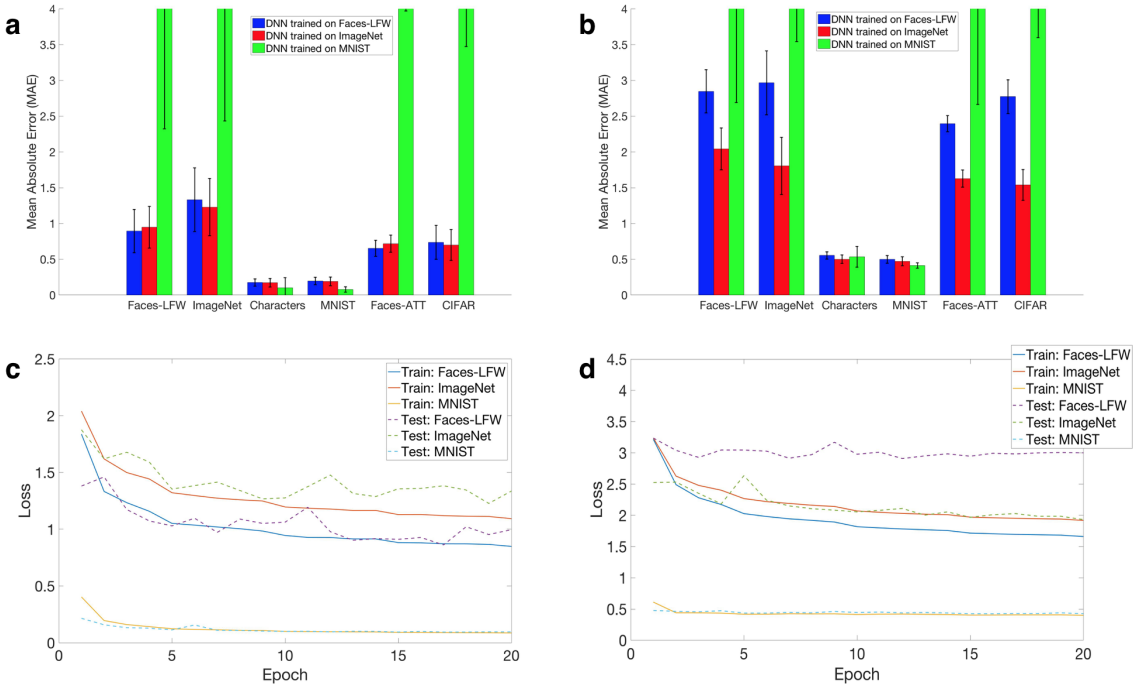

Figure 2-5: Quantitative analysis of IDiffNet trained using MAE as the loss function. Test errors for IDiffNet trained on Faces-LFW (blue), ImageNet (red) and MNIST (green) on six datasets when the diffuser used is (a) 600-grit and (b) 220-grit. The training and testing error curves when the diffuser used is (c) 600-grit and (d) 220-grit.

the fact that the objects that this IDiffNet was trained on were sparse; and, hence, it also tends to make sparse estimates. Unfortunately, in this case the sparse local minima where IDiffNet is trapped are featureless. Tackling this problem motivated us to examine the Negative Pearson Correlation Coefficient (NPCC) as alternative loss function.

The NPCC is defined as [106]:

$$\text{NPCC} = \frac{-1}{bwh} \times \sum_{k=1}^{b} \sum_{i=1}^{w} \sum_{j=1}^{h} \frac{\left(\hat{f}_k(i,j) - \mu_{\hat{f}_k}\right)\left(f_k(i,j) - \mu_{f_k}\right)}{\sigma_{f_k}\sigma_{\hat{f}_k}}. \tag{2.6}$$

Here, μ_{f_k} and $\mu_{\hat{f}_k}$ are the spatial averages of f_k and $\hat{f}_k$, respectively; σ_{f_k} and $\sigma_{\hat{f}_k}$ are the standard deviations of f_k and $\hat{f}_k$, respectively.

The qualitative and quantitative reconstruction results using NPCC as the loss function are shown in Fig. 2-6 and 2-7, respectively. The reconstructed images are

normalized since the NPCC value will be the same if we multiply the reconstruction by any positive constants. Similar to the case where MAE is used as the loss function, the reconstruction is better in the 600-grit diffuser case than the 220-grit diffuser case. However, when IDiffNet is trained on MNIST for the 220-grit diffuser (column x), high quality reconstruction is achieved for the test images coming from Characters and MNIST database (row c and d). This is in contrast to the MAE-trained case, thus indicating that NPCC is a more appropriate loss function to use in this case. It helps IDiffNet to learn the sparsity in the ground truth and in turn use the sparsity as a strong prior for estimating the inverse. In addition, when trained on ImageNet for the 220-grit diffuser (column ix), IDiffNet is still able to reconstruct the general shape (salient features) of the object. But the NPCC-trained reconstructions are visually slightly worse compared with the MAE-trained cases.

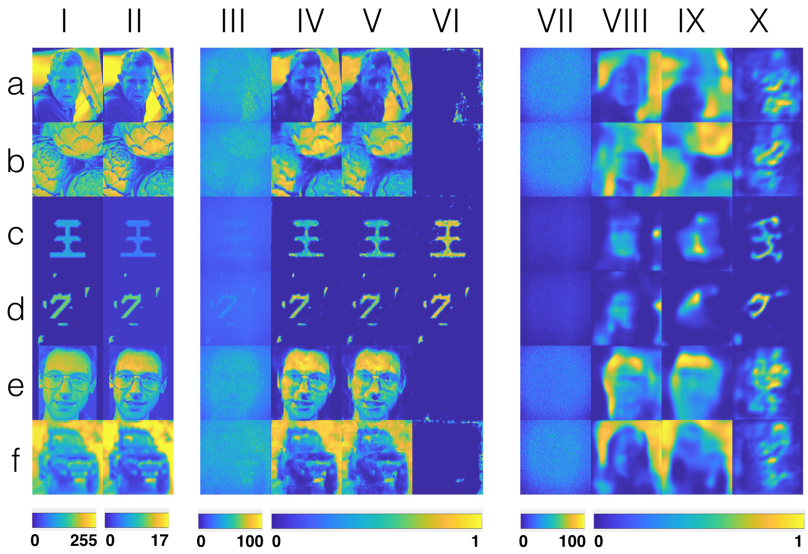

Figure 2-6: Qualitative analysis of IDiffNets trained using NPCC as the loss function. (i) Ground truth pixel value inputs to the SLM. (ii) Corresponding intensity images calibrated by SLM response curve. (iii) Raw intensity images captured by CMOS detector for 600-grit glass diffuser. (iv) IDiffNet reconstruction from raw images when trained using Faces-LFW dataset [99]. (v) IDiffNet reconstruction when trained using ImageNet dataset [100]. (vi) IDiffNet reconstruction when trained using MNIST dataset [101]. Columns (vii-x) follow the same sequence as (iii-vi) but in these sets the diffuser used is 220-grit. Rows (a-f) correspond to the dataset from which the test image is drawn: (a) Faces-LFW, (b) ImageNet, (c) Characters [102], (d) MNIST, (e) Faces-ATT [104, 105], (f) CIFAR [103], respectively.

In both MAE and NPCC training cases, IDiffNet performance also depends on the dataset that it is trained on. From Fig. 2-4 and 2-6, we observe that IDiffNet generalizes best when being trained on ImageNet and has the most severe overfitting problem when being trained on MNIST. Specifically, when IDiffNet is trained on MNIST, even for the 600-grit diffuser (column vi), it works well if the test image comes from the same database or a database that shares the same sparse characteristics as MNIST (e.g. characters). It gives much worse reconstruction when the test image comes from a much different database. When IDiffNet is trained on Faces-LFW, it generalizes well for the 600-grit diffuser, but for the 220-grit diffuser it exhibits overfitting: it tends to reconstruct a face at the central region, as in Horisaki's case. When IDiffNet is trained on ImageNet, it generalizes well even for the 220-grit diffuser. As we can see in column ix, for all the test images, IDiffNet is able to at least reconstruct the general shapes (salient features) of the objects. This indicates that IDiffNet has learned at the very least a generalizable mapping of low-level textures between the captured speckle patterns and the input images. Similar observation may also be made from Fig. 2-5 and 2-7. From subplots (a) and (b) in both figures, we notice that the IDiffNets trained on MNIST have much higher MAEs/lower PCCs when tested on other databases. As shown in subplot (d), the IDiffNets trained on Faces-LFW have a large discrepancy between training and test error, while for IDiffNets trained on ImageNet, the training and testing curves converge to almost the same level. An explanation for this phenomenon is that all the images in MNIST or Faces-LFW databases share the same characteristics (eg. sparse, circular shape), imposing a strong prior on IDiffNet. On the other hand, the ImageNet database consists of a mixture of generic images that not have too much in common. As a result, IDiffNet trained on ImageNet generalizes better. It is worth noting that overfitting in our case evidences itself as face-looking "ghosts" occurring when IDiffNet trained on Faces-LFW tries to reconstruct other kinds of images, for example (see Fig. 2-6, column viii). This is again similar to Horisaki's observations [88].

From comparing the four possible combinations of weak *vs* strong scattering and constrained dataset (e.g. MNIST) *vs* generic dataset (e.g. ImageNet), we conclude

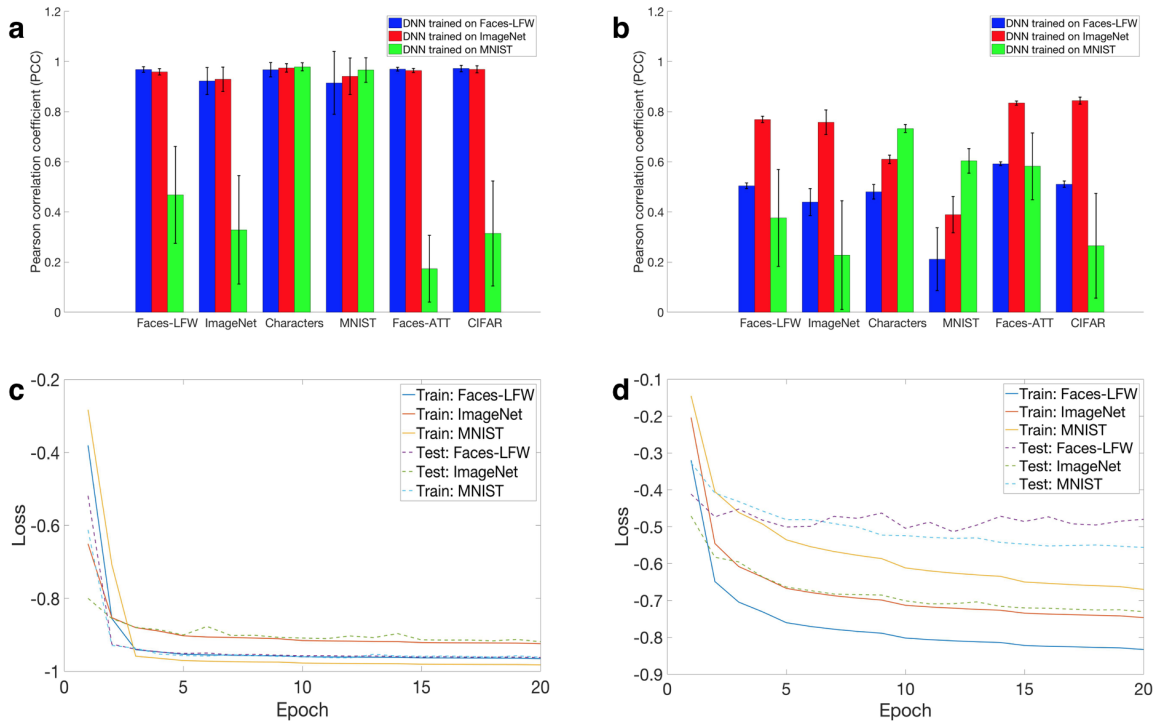

Figure 2-7: Quantitative analysis of our trained deep neural networks for using NPCC as the loss function. Test errors for the IDiffNets trained on Faces-LFW (blue), ImageNet (red) and MNIST (green) on six datasets when the diffuser used is (a) 600-grit and (b) 220-grit. The training and testing error curves when the diffuser used is (c) 600-grit and (d) 220-grit.

the following: when scattering is weak, it is to our benefit to train the IDiffNets on a generic dataset because the resulting neural networks generalize better and can cope with the scattering also for general test images. On the other hand, when scattering is strong, it is beneficial to use a relatively constrained dataset with strong sparsity present in the typical objects: the resulting neural networks are then more prone to overfitting, but now this works to our benefit for overcoming strong scattering (at the cost, of course, of working only for test objects coming from the more restricted database.) The choice of optimization functional makes this tradeoff even starker: MAE apparently does not succeed in learning the strong sparsity even for MNIST datasets, whereas the NPCC does much better, even being capable of reconstructing test objects under the most severe scattering conditions (220-grit diffuser, column x in Fig. 2-6) as long as the objects are drawn from the sparse dataset. These observations

are summarized in Table. 2.1.

Table 2.1: Summary of reconstruction results in different cases. [√: Visually recognizable; •: Salient feature recognizable; ×: Visually unrecognizable.]

		600-grit		220-grit	
	Training dataset	Loss: MAE	Loss: NPCC	Loss: MAE	Loss: NPCC
Test: Faces-LFW	Faces-LFW	√	√	•	×
	ImageNet	√	√	•	•
	MNIST	×	×	×	×
Test: ImageNet	Faces-LFW	√	√	×	×
	ImageNet	√	√	•	•
	MNIST	×	×	×	×
Test: Characters	Faces-LFW	√	√	×	×
	ImageNet	√	√	•	•
	MNIST	√	√	×	√
Test: MNIST	Faces-LFW	√	√	×	×
	ImageNet	√	√	•	•
	MNIST	√	√	×	√
Test: Faces-ATT	Faces-LFW	√	√	×	×
	ImageNet	√	√	•	•
	MNIST	×	×	×	×
Test: CIFAR	Faces-LFW	√	√	×	×
	ImageNet	√	√	•	•
	MNIST	×	×	×	×

2.4 Resolution and shift invariance tests for IDiffNet

In this section, we investigate the spatial resolution of our trained IDiffNet. Without the diffuser, our system is a telescope of numerical aperture NA = 12.7/250 = 0.0508. The diffraction-limited Abbé resolution is $d_0 = \lambda/(2\text{NA}) = 6.23\mu$m. With the 600-grit diffuser in the system, we analyze experimentally four IDiffNets that we trained on either the ImageNet or MNIST database and using either MAE or NPCC as the loss function. In order to evaluate the spatial resolution, we designed two different sets of test patterns, dots and fringes, as shown in Fig. 2-8. The dots are constructed as super-pixels from 4×4 pixels ($80\times80\mu$m), and the fringes are constructed as bands of width equal to 4 pixels (80μm). These choices make the dot and fringe spacings consistent with the sampling scheme chosen in the experiments of Section 6.3. It

should also be mentioned that the super-pixel size places a limit on IDiffNet resolution, since IDiffNet is trained with examples whose sampling distance is equal to one super-pixel. In the dot pattern set, each pattern contains 8 dot pairs and the spacings D between the two dots within the same pair are set to be the same. The entire set consists of 10 such dot patterns with D gradually varies from 1 super-pixel to 10 super-pixels. In the fringe pattern set, each pattern contains equally spaced fringes. Similarly, the entire set consists of 10 such fringe patterns with the spacing D gradually varying from 1 super-pixel to 10 super-pixels.

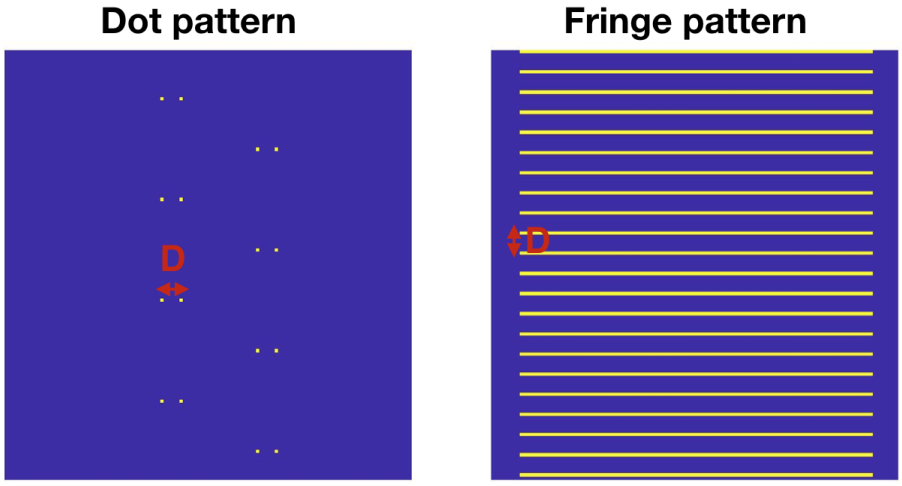

Figure 2-8: Resolution test patterns. Left: Dot pattern; Right: Fringe pattern.

Those resolution test patterns are displayed on the SLM and the corresponding speckle patterns are captured and fed into our trained IDiffNet for reconstruction. Here, we first show the resolution test results of the IDiffNets trained using MAE as loss function.

As shown in Fig. 2-9, the IDiffNet trained on MNIST database is able to resolve two dots with spacing $D = 4$ super-pixels, but fails to distinguish two dots with spacing $D = 3$ super-pixels. Same spatial resolution is demonstrated using fringe patterns as well, where nearby fringes with spacing $D = 4$ super-pixels are resolved while fringes with spacing $D = 3$ super-pixels are unable to be distinguished. In addition, we find that the reconstruction qualities of dot patterns are better than those of the fringe patterns. This result is as expected since the MNIST training

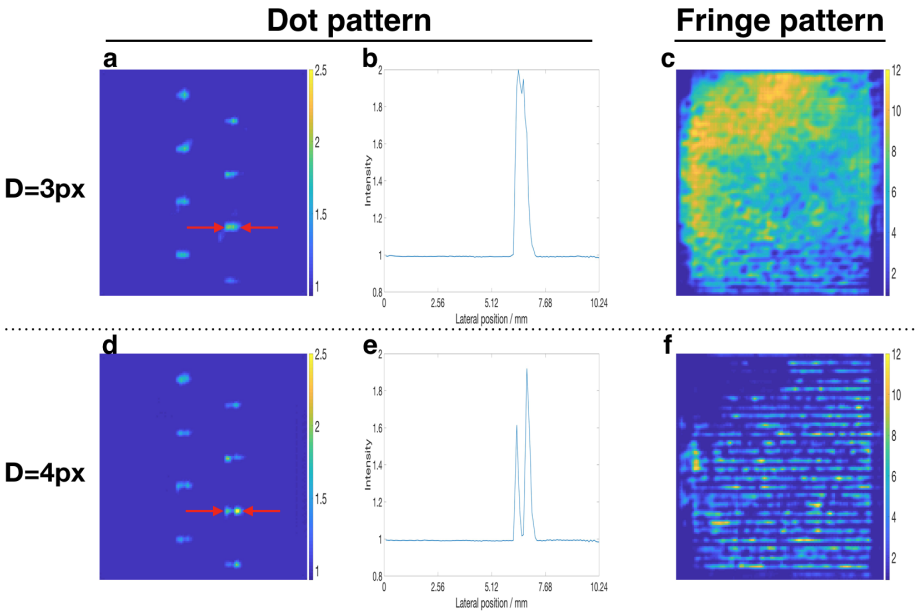

Figure 2-9: Experimental resolution test result for IDiffNet trained on MNIST using MAE as loss function. The diffuser used is 600-grit. (a) Reconstructed dot pattern when $D = 3$ super-pixels. (b) 1D cross-section plot along the line indicated by red arrows in (a). (c) Reconstructed fringe pattern when $D = 3$ super-pixels. (d) Reconstructed dot pattern when $D = 4$ super-pixels. (e) 1D cross-section plot along the line indicated by red arrows in (d). (f) Reconstructed fringe pattern when $D = 4$ super-pixels.

database imposes a strong sparsity prior in a set of basis functions that themselves look relatively spatially sparse [107]. This property makes IDiffNet perform better on spatially sparse test samples (dot patterns) than other less sparse test samples (fringe patterns). Therefore, dot patterns are more appropriate to be used to test the resolution of IDiffNet trained on MNIST.

For the IDiffNet trained on ImageNet, its spatial resolution is the same as the MNIST training case, as demonstrated in Fig. 2-10. However, the reconstruction qualities of fringe patterns are better than those of the dot patterns since the ImageNet training database contains more general images which are sparse in a set of basis functions that is spatially richer than the MNIST dictionary [108]. Because of this observation, fringe patterns are more appropriate be used to test the resolution of IDiffNet trained on ImageNet.

Now, for the 600-grit diffuser, we show the resolution test results of the IDiffNets

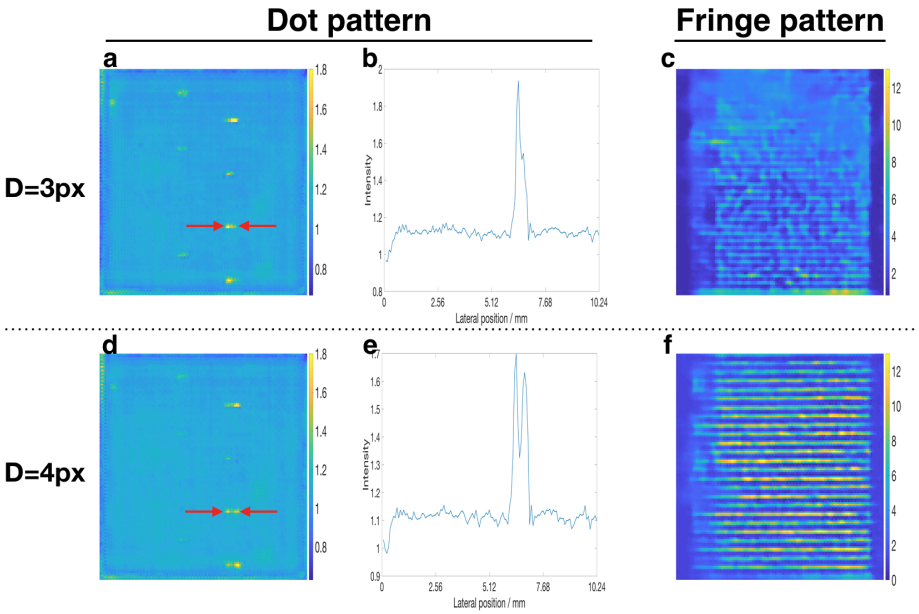

Figure 2-10: Experimental resolution test result for IDiffNet trained on ImageNet using MAE as loss function. The diffuser used is 600-grit. (a) Reconstructed dot pattern when $D = 3$ super-pixels. (b) 1D cross-section plot along the line indicated by red arrows in (a). (c) Reconstructed fringe pattern when $D = 3$ super-pixels. (d) Reconstructed dot pattern when $D = 4$ super-pixels. (e) 1D cross-section plot along the line indicated by red arrows in (d). (f) Reconstructed fringe pattern when $D = 4$ super-pixels.

trained using NPCC as loss function.

As shown in Fig. 2-11, the IDiffNet trained on MNIST database is able to resolve two dots with spacing $D = 4$ super-pixels, but fails to distinguish two dots with spacing $D = 3$ super-pixels. Same spatial resolution is demonstrated using fringe patterns as well, where nearby fringes with spacing $D = 4$ super-pixels are resolved while fringes with spacing $D = 3$ super-pixels are unable to be distinguished. In addition, we find that the reconstruction qualities of dot patterns are better than those of the fringe patterns. This result is as expected since the MNIST training database imposes a strong sparsity prior, making the IDiffNet perform better on sparse test samples (dot patterns) than other less sparse test samples (fringe patterns). Therefore, dot patterns are more appropriate to be used to test the resolution of IDiffNet trained on MNIST. For the IDiffNet trained on ImageNet, its spatial resolution is the same as the MNIST training case, which is demonstrated in Fig. 2-12. However, the

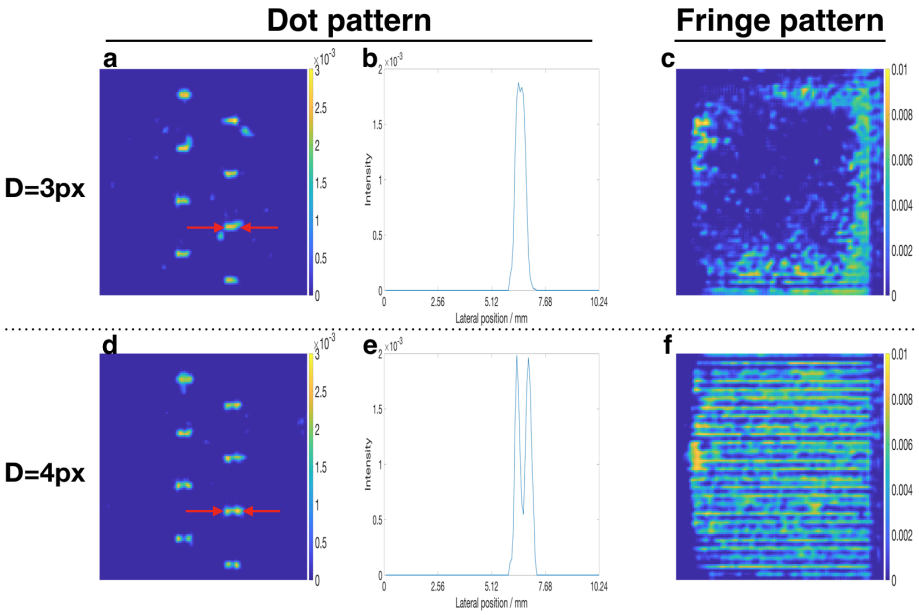

Figure 2-11: Experimental resolution test result for IDiffNet trained on MNIST using NPCC as loss function. The diffuser used is 600-grit. (a) Reconstructed dot pattern when $D = 3$ super-pixels. (b) 1D cross-section plot along the line indicated by red arrows in (a). (c) Reconstructed fringe pattern when $D = 3$ super-pixels. (d) Reconstructed dot pattern when $D = 4$ super-pixels. (e) 1D cross-section plot along the line indicated by red arrows in (d). (f) Reconstructed fringe pattern when $D = 4$ super-pixels.

reconstruction qualities of fringe patterns are better than those of the dot patterns since the ImageNet training database contains more general images and no longer impose the sparsity prior. Hence, fringe patterns should be used to test the resolution of IDiffNet trained on ImageNet. All the above observations are the same as those in the MAE training case. Therefore, the choice of loss function does not affect the spatial resolution of the trained IDiffNet in the 600-grit diffuser case.

Now, let us test the spatial resolution of IDiffNet trained using the 220-grit diffuser. In this strong scattering case, as described in Section 6.3, we have to use MNIST as the training database and use NPCC as the loss function. In order to match the strong prior imposed by MNIST database, we design the resolution test pattern as shown in Fig. 2-13a, where those dots are placed in a layout resembles the digit '7' and the spacing between nearby dots is D. The entire set consists of 10 such patterns with the spacing D gradually varies from 10 super-pixel to 19 super-pixels. As shown

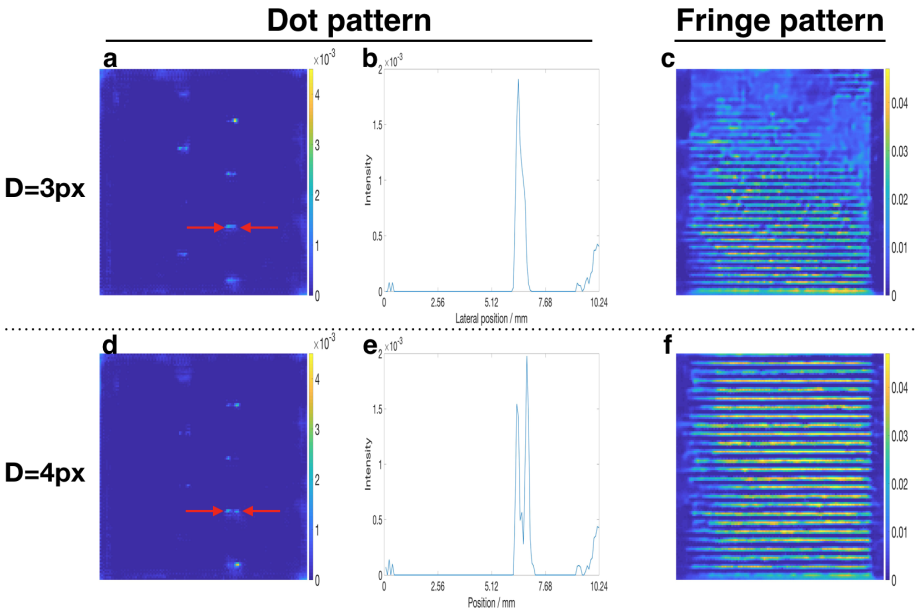

Figure 2-12: Experimental resolution test result for IDiffNet trained on ImageNet using NPCC as loss function. The diffuser used is 600-grit. (a) Reconstructed dot pattern when $D = 3$ super-pixels. (b) 1D cross-section plot along the line indicated by red arrows in (a). (c) Reconstructed fringe pattern when $D = 3$ super-pixels. (d) Reconstructed dot pattern when $D = 4$ super-pixels. (e) 1D cross-section plot along the line indicated by red arrows in (d). (f) Reconstructed fringe pattern when $D = 4$ super-pixels.

in Fig. 2-13, we can find that the trained IDiffNet is able to resolve nearby dots with spacing $D = 17$ super-pixels, but fails to distinguish two dots with spacing $D = 16$ super-pixels. As expected, the spatial resolution in this case is worse than that in the 600-grit diffuser case.

In light of our earlier observation about limited shift invariance in the forward operator (see discussion after Fig. 2-2), we also analyzed IDiffNet's shift invariance. In simulation, as in Section 5.2, we chose 100 test images in the MNIST database. For each image $I(x, y)$, we first obtain its corresponding speckle pattern $I_{\text{out}}(x', y'; x, y)$. Then, we shift $I(x, y)$ along the x direction for some distance Δx and obtain the corresponding speckle pattern $I_{\text{out}}(x', y'; x + \Delta x, y)$. After that, we shift the speckle pattern $I_{\text{out}}(x', y'; x + \Delta x, y)$ back by a distance $\Delta x' = m\Delta x$ to obtain $I_{\text{out}}(x' - \Delta x', y'; x + \Delta x, y)$, where $m = 0.6$ is the ratio between the camera and SLM pixel

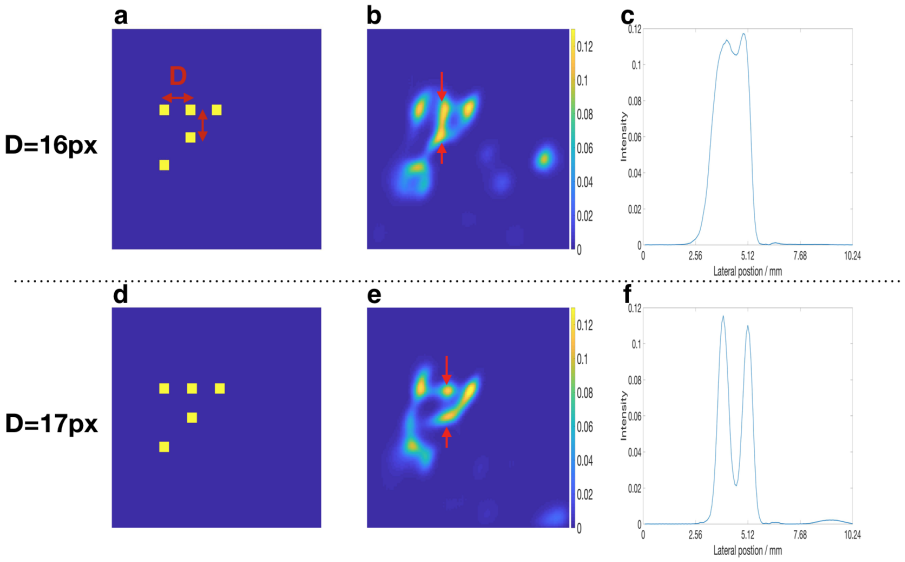

Figure 2-13: Experimental resolution test result for IDiffNet trained on MNIST using NPCC as loss function. The diffuser used is 220-grit. (a) Resolution test pattern when $D = 16$ super-pixels. (b) Reconstructed test pattern when $D = 16$ super-pixels. (c) 1D cross-section plot along the line indicated by red arrows in (b). (d) Resolution test pattern when $D = 17$ super-pixels. (e) Reconstructed test pattern when $D = 17$ super-pixels. (f) 1D cross-section plot along the line indicated by red arrows in (e).

sizes. Finally, we compute the correlations in the speckle patterns as

$$C_s(\Delta x) = \mathrm{PCC}\left[I_{\mathrm{out}}(x', y'; x, y), I_{\mathrm{out}}(x' - \Delta x', y'; x + \Delta x, y)\right]. \tag{2.7}$$

Here, PCC is defined as in equation (2.6) and without the negative sign.

Similarly, we can compute the correlations in the reconstructions $\hat{I}(x, y; x, y)$ as

$$C_r(\Delta x) = \mathrm{PCC}\left[\hat{I}(x, y; x, y), \hat{I}(x - \Delta x, y; x + \Delta x, y)\right]. \tag{2.8}$$

As shown in Fig. 2-14, shift invariance after IDiffNet increases ($C_r > C_s$), demonstrating that IDiffNet has learnt to compensate for shift variance in the forward operator. In fact, the domain of shift invariance obtained with IDiffNet is bigger than generally obtained by approaches based on the memory effect [74, 77]. In the latter, the Field of View (FOV) is limited by shift invariance in the forward operator, *i.e.* the domain of high correlation between shifted PSFs, and is typically small, e.g.

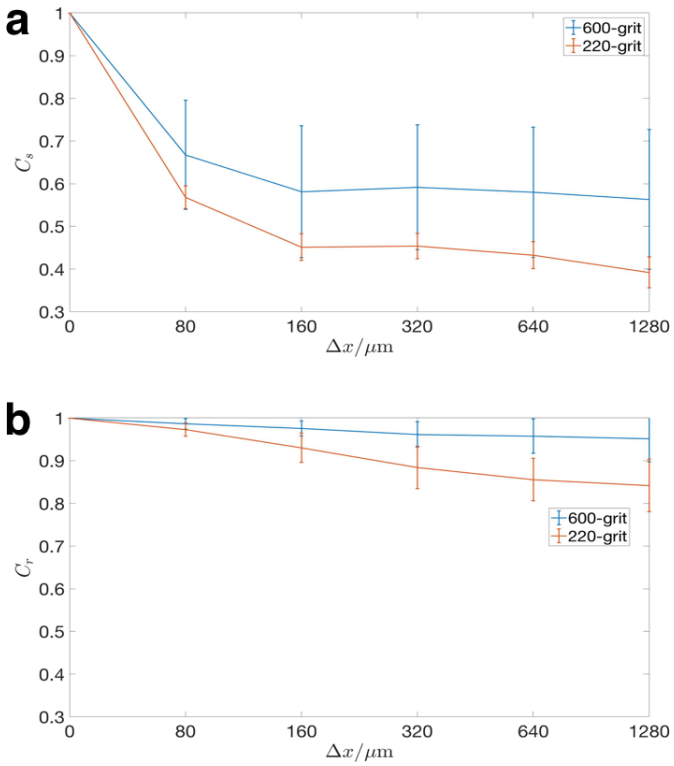

Figure 2-14: Simulated shift invariance test. (a) Correlations in the speckle patterns C_s calculated on MNIST database. (b) Correlations in the reconstructions C_r calculated on MNIST database. In the 600-grit case, the IDiffNet is trained on ImageNet using MAE loss function; in the 220-grit case, the IDiffNet is trained on MNIST using NPCC loss function.

FOV$\sim 2.5 \times 10^{-2}$ in [74]. On the other hand, our experimental results with IDiffNet demonstrate FOV$\gtrsim 4 \times 10^{-2}$ for the same diffuser of 220-grit size.

2.5 Comparison with denoising neural networks

To get a sense of what IDiffNets learn, we first compare their reconstruction results with that of a denoising neural network. Specifically, we use ImageNet as our training database. To each image in the training dataset, we simulate a noisy image using Poisson noise and make the peak signal-to-noise ratio (PSNR) of the resulting noisy image visually comparable to that of the corresponding speckle image captured using the 600-grit diffuser. We use Poisson noise other than different kinds of noise such as Gaussian because Poisson noise is signal-dependent, similar to the diffuser case. We

then train a denoising neural network using those noisy images. For the denoising neural network, we implement the residual network architecture [109]. Finally, we feed the test speckle images captured using the 600-grit diffuser into this denoising neural network and compare the outputs with those reconstructed by IDiffNet (using MAE as the loss function).

The comparison results are shown in Fig. 2-15. From column iv, we find that the denoising neural network works well when the test images are indeed noisy according to the Poisson model. However, as shown in column v, if we input the diffuse image into the denoising network, then it can only output a highly blurred image, much worse than IDiffNet given the same diffuse input, as can be seen in column vi. This result demonstrates that IDiffNet is not doing denoising, although the speckle image obtained using the 600-grit diffuser visually looks similar to a noisy image. We posit the reason for this as follows: Poisson noise operates pixel-wise. Consequently, denoising for Poisson noise is effectively another pixel-wise operation that does not depend much on spatial neighborhood, except to the degree that applying priors originating from the structure of the object helps to denoise severely affected signals. A denoising neural network, then, learns spatial structure only as a prior on the class of objects it is trained on. However, this is not the case when imaging through a diffuser: then every pixel value in the speckle image is influenced by a set of nearby pixels in the original image. This may also be seen from the PSF plots shown in Fig. 2-2. The denoising neural network fails because it has not learnt the spatial correlations between the nearby pixels and the correct kernel of our imaging system, as our IDiffNet has.

We also examined the maximally-activated patterns (MAPs) of the IDiffNets and the denoising neural network; *i.e.* what types of inputs would maximize network filter response (gradient descent on the input with average filter response as loss function) [91]. Fig. 2-16 shows the MAP analysis of two convolutional layers at different depths for all the three neural networks. For both the shallow and deep layers, we find the MAPs of our IDiffNets are qualitatively different from those of the denoising network. This further corroborates that IDiffNet is not merely doing denoising. In addition,

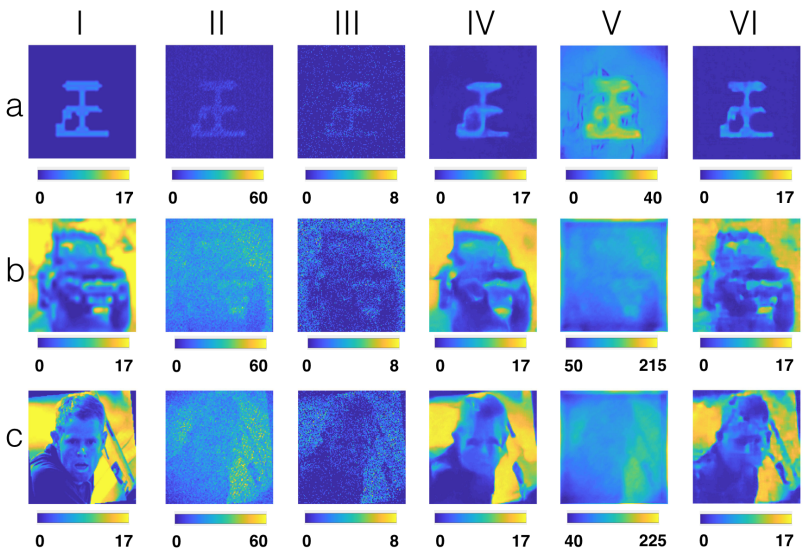

Figure 2-15: Comparison between IDiffNets and a denoising neural network. (i) Ground truth intensity images calibrated by SLM response curve. (ii) Speckle images that we captured using the 600-grit diffuser (after subtracting the reference pattern). (iii) Noisy images generated by adding Poisson noise to the ground truth. (iv) Reconstructions of the denoising neural network when inputing the noisy image in (iii). (v) Reconstructions of the denoising neural network when inputing the speckle image in (ii). (vi) IDiffNet reconstructions when inputing the noisy image the speckle image in (ii). [The images shown in column vi are the same as those in the column v of Fig .2-4, duplicated here for the readers' convenience]. Rows (a-c) correspond to the dataset from which the test image is drawn: (a) Characters[102], (b) CIFAR [103], (c) Faces-LFW [99], respectively.

the MAPs of the 600-grit IDiffNet show finer textures compared with that of the 220-grit IDiffNet, indicating that the IDiffNet learns better in the 600-grit diffuser case.

2.6 Conclusions

We have demonstrated that IDiffNets, built according to the densely connected convolutional neural network architecture, can be used as an end to end approach for imaging through scattering media. The reconstruction performance depends on the scattering strength of the diffusers, the type of the training dataset (in particular, its sparsity), as well as the loss function used for optimization. The IDiffNets seem to learn automatically the properties of the scattering media, including the degree

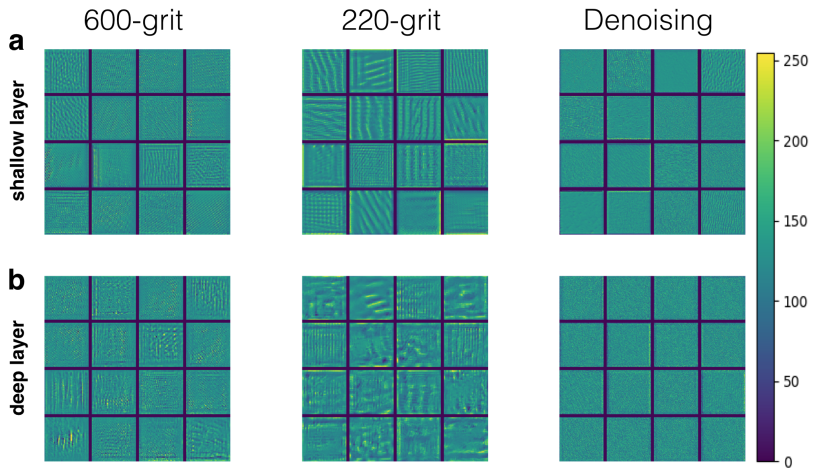

Figure 2-16: Maximally-activated patterns (MAPs) for different DNNs. (a) 128×128 inputs that maximally activate the filters in the convolutional layer at depth 5. (b) 128×128 inputs that maximally activate the filters in the convolutional layer at depth 13. [There are actually more than 16 filters at each convolutional layer, but we only show the 16 filters have the highest activations here.]

of shift invariance and how to at least partially compensate it; as well as the priors restricting the objects where the network is supposed to perform well, depending on the database the network was trained with.

Chapter 3

Quantitative phase retrieval using PhENN

3.1 Introduction

Quantitative phase retrieval (QPR) is a classical problem in optical imaging and has important applications in the fields of biomedical diagnosis and nanostructure inspection. Conventional cameras are only sensitive to the intensity of the light field. However, the phase carries crucial information about the object [110]: e.g. refractive index, 3D shape, etc. The task of QPR is to quantitatively retrieve the phase of the light field from intensity measurements. Traditional approaches include digital holography (DH) [111, 112, 113, 86, 114] and the related phase shifting interferometry method [115], propagation based methods such as the Transport of Intensity Equation (TIE) [18, 116, 117, 118, 119, 120, 121, 122, 12], iterative projection methods such as the Gerchberg-Saxton-Fienup (GSF) algorithm [81, 82, 123, 124, 125] and optimization based methods [126, 127, 128, 129].

The hypothesis that we set out to test in this chapter is whether a deep neural network (DNN) can be trained to realize QPR. In addition to the well-established adaptability of DNNs as effective approximators to general-purpose nonlinear operators, our approach is attractive for several reasons: the DNN, once trained, should recover every possible object within a hopefully large enough class, unlike GSF where

the estimate is obtained separately and iteratively for each new object presented. Alternatively, since during DNN training iteration takes place over several objects simultaneously, the learning approach is expected to be a more robust generalization of GSF. Moreover, the DNN approach is not subject to any of the sparsity requirements of either DH (sparsity in space) or TIE (sparsity in spatial gradients). Indeed, our experiment, described in detail in Section 3.2, was explicitly designed to violate the DH and TIE assumptions. Lastly, compared to optimization-based approaches, the user using DNN is not required to specify the forward operator H and the priors Φ or decide about the magnitude of the regularization parameter α; instead, to the degree that our results show successful inversion, they suggest that the DNN learns H, Φ, and α implicitly through training. The downside of the DNN approach is that a sufficiently large database of known pairs of phase objects and their corresponding raw intensity images must be available for training and testing.

Neural network approaches often come under criticism because the quality of training depends on the quality of the examples given to the network during the training phase. For instance, if the inputs used to train a network are not diverse enough, then the DNN will learn priors of the input images instead of generalized rules for phase retrieval from intensity. This was the case in [88], where an SVM trained using images of faces could adequately reconstruct faces, but when given the task of reconstructing images of natural objects such as a pair of scissors, the trained SVM still returned an output that resembled a human face.

For our specific problem, an ideal training set would encompass all possible phase objects. Unfortunately, phase objects, generally speaking, constitute a rather large class and it would be unrealistic to attempt to train a network sampling from across all possible objects from this large class. Instead, we synthesize phase objects in the form of natural images derived from the ImageNet [100] database because it is readily available and widely used in the study of various machine learning problems. For comparison, we also trained a separate network using a narrower class of (facial) images from the Faces-LFW [99] database.

As expected, our network, Phase Extraction Neural Network (PhENN), did well

when presented with unknown phase objects in the form of faces or natural images that it had been trained to. Notably, the network also performed well when presented with objects outside of its training class – the DNN trained using images of faces was able to reconstruct images of natural objects, and the DNN trained using images of natural objects was able to reconstruct images of faces. Additionally, both DNNs were able to reconstruct completely distinct images including: handwritten digits, characters from different languages (Arabic, Mandarin, English), and images from a disjoint natural image dataset. Both trained networks yielded accurate results even when the object-to-sensor distance(s) in the training set slightly differed from that of the testing set, suggesting that the network is not merely pattern-matching but instead has actually learned a generalizable model approximating the inverse operator.

3.2 Experiment

We consider a lensless phase imaging system this chapter. The phase object is illuminated by the collimated monochromatic light. The light transmitted/reflected through the phase object propagates in free space and forms an intensity image on the detector placed at a distance z away. Assuming that the illumination plane wave has unit amplitude, the image formation model in this system can be described as:

$$g(x,y) = \left| \exp\left\{ if(x,y) \right\} * \exp\left\{ i\frac{\pi}{\lambda z}\left(x^2+y^2\right) \right\} \right|^2 . \tag{3.1}$$

Here, $f(x,y)$ is the phase distribution of the object, $g(x,y)$ is the intensity image captured by the detector, λ is the wavelength of the illumination light, i is the imaginary unit and $*$ denotes the convolution operation.

Our experimental arrangement is as shown in Fig. 3-1. Light from a He-Ne laser source (Thorlabs, HNL210L, 632.8nm) first transmits through a spatial filter, which consists of a microscope objective (Newport, M-60X, 0.85NA) and a pinhole aperture ($D = 5\mu$m), to remove spatial noise. After being collimated by the lens ($f = 150$mm), the light is reflected by a mirror and then passes through a linear polarizer, followed

by a beam splitter. A spatial light modulator (Holoeye, LC-R 720, reflective) is placed normally incident to the transmitted light and acts as a pixel-wise phase object. The SLM pixel size is $20 \times 20\mu\text{m}^2$ and number of pixels is 1280×768, out of which the central 512×512 portion is only used in the experiments. The SLM-modulated light is then reflected by the beam splitter and passes through a linear polarization analyzer, before being collected by a CMOS camera (Basler, A504k) placed at a distance z. The CMOS camera pixel size is $12 \times 12\mu\text{m}^2$ and number of pixels is 1280×1024, which is cropped to a 1024×1024 square used for processing. The total used linear camera size of ≈ 12.3mm is slightly larger than the active SLM size of ≈ 10.2mm to accommodate expansion of the propagating beam due to diffraction. Images recorded by the CMOS camera are then processed on an Intel i7 CPU, with neural network computations performed on a GTX1080 graphics card (NVIDIA).

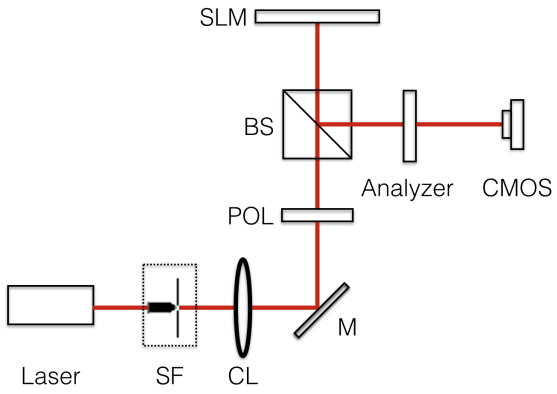

Figure 3-1: Experimental arrangement. SF: spatial filter; CL: collimating lens; M: mirror; POL: linear polarizer; BS: beam splitter; SLM: spatial light modulator.

According to its user manual, the LC-R 720 SLM can realize (approximate) pure-phase modulation if we modulate the light polarization properly. Specifically, for He-Ne laser light, if we set the incident beam to be linearly polarized at 45° with respect to the vertical direction and also set the linear polarization analyzer to be oriented at 340° with respect to the vertical direction, then the amplitude modulation of the SLM will become almost independent of the assigned (8-bit gray-level) input. In this arrangement, the phase modulation of the SLM follows a monotonic, almost-linear relationship with the assigned pixel value (with maximum phase depth: $\sim$

1π). We experimentally evaluated the correspondence between 8-bit grayscale input images projected onto the SLM and phase values in the range $[0, -\pi]$ (see Appendix Section A.2). In this chapter, we approximate our SLM as a pure-phase object and computationally recover the phase using a neural network.

The CMOS detector was placed after a free-space propagation distance z. Distinct DNNs were trained from recorded diffraction patterns at three distances $z_1 =$ 37.5cm$\pm$2mm (NA=0.0164$\pm 9\times 10^{-4}$, Fresnel number $\mathcal{F}$=159$\pm$1), $z_2 = 67.5$cm$\pm$2mm (NA=$0.0091 \pm 1 \times 10^{-4}$, Fresnel number $\mathcal{F}$=88.3 ± 0.3) and $z_3 = 97.5\text{cm} \pm 2\text{mm}$ (NA=$0.0063 \pm 1 \times 10^{-4}$, Fresnel number $\mathcal{F}$=61.2 ± 0.1). Our experiment consisted of two phases: training and testing. During the training phase, we modulated the phase SLM according to samples randomly selected from the Faces-LFW or ImageNet database. We resized and padded selected images before displaying them on our SLM. Two examples of inputs, as they are sent to the SLM, and their corresponding raw intensity images (diffraction patterns) as captured on the CMOS are shown in Fig. 3-2. Our training set consisted of 10,000 such faces/images - diffraction pattern pairs. The raw intensity images from all these training examples were used to train the weights in our neural network. We used a Zaber A-LST1000D stage with repeatability $2.5\mu m$ to translate the camera in order to analyze the robustness of the learnt network to axial perturbations. The positional accuracy of $\pm$2mm reported earlier on the training distances z_1, z_2, z_3 is derived from our error in manually establishing the absolute distance between SLM and camera; whereas the stage repeatability determines the error in camera displacement relative to these initial training positions, respectively, for each axial robustness test in Section 3.3.

Our Phase Extraction Neural Network (PhENN) uses a convolutional residual neural network (ResNet) architecture. In a convolutional neural network (CNN), inputs are passed from nodes of each layer to the next, with adjacent layers connected by convolution. Convolutional ResNets extend CNNs by adding short term memory to each layer of the network. The intuition behind ResNets is that one should only add a new layer if one stands to gain by adding that layer. ResNets ensure that the $(N+1)$th layer learns something new about the network by also providing the

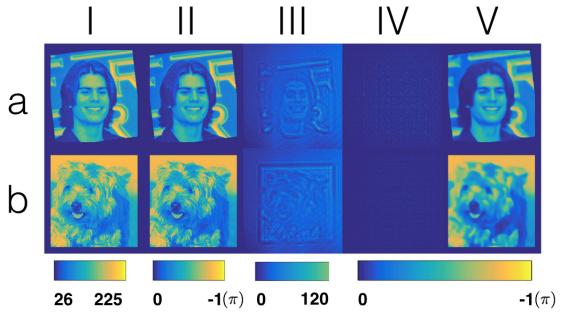

Figure 3-2: Neural network training. Rows (a) and (b) denote the networks trained on Faces-LFW and ImageNet dataset, respectively. (i) randomly selected example drawn from the database; (ii) calibrated phase image of the drawn sample; (iii) diffraction pattern generated on the CMOS by the same sample; (iv) DNN output before training (*i.e.* with randomly initialized weights); (v) DNN output after training.

original input to the output of the $(N+1)$th layer and performing calculations on the residual of the two. This forces the new layer to learn something different from what the input has already encoded/learned [130].

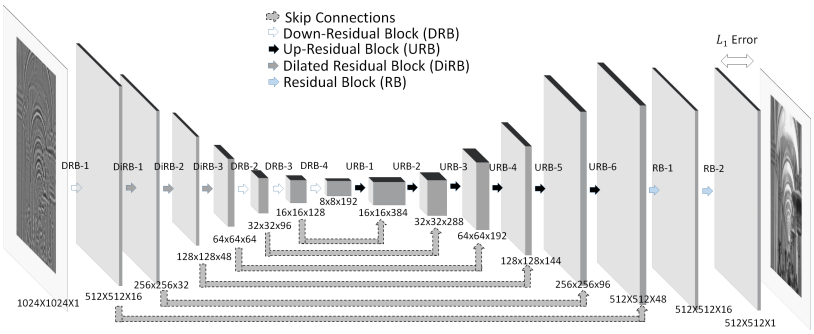

Figure 3-3: Detailed schematic of PhENN architecture, indicating the number of layers, nodes in each layer, etc.

A diagram of our PhENN architecture is shown in Fig. 3-3. The input layer is the image captured by the CMOS camera. It is then successively decimated by 7 residual blocks of convolution + downsampling followed by 6 residual blocks of deconvolution + upsampling, and finally 2 standard residual blocks. Some of the residual blocks are comprised of dilated convolutions so as to increase the receptive field of the convolution filters, and hence, aggregate diffraction effects over multiple scales [131]. We use skip connections to pass high frequency information learnt in the initial layers down the network towards the output reconstruction, and have two

standard residual blocks at the end of the network to finetune the reconstruction. At the very last layer of our neural network, the values represent an estimate of our input signal. The connection weights are trained using backpropagation (not to be confused with optical backpropagation) on the MAE loss function [Eq.2.5]. Additional details about the architecture of each blocks and the training of PhENN are provided in the Appendix Section B.2.

We collected data from six separate experiment runs using training inputs from Faces-LFW or ImageNet and object-to-sensor distances of z_1, z_2 , or z_3. These data were used to train six separate PhENNs for evaluation.

Fig. 3-2(iv) shows a sample PhENN's output at the beginning of its training phase (*i.e.* with randomly initialized weights), and Fig. 3-2(v) shows the network output after training, for the same example object-raw image pairs. Training each network took $\approx$ 20 hours using Tensorflow on our GPU. We provide analysis of the trained PhENN in Section 3.3.

Our testing phase consisted of: (1) sampling disjoint examples from the same database (either Faces-LFW or ImageNet) and other databases such as MNIST, CIFAR, Faces-ATT etc., (2) using these test examples to modulate the SLM and produce raw intensity images on the camera, (3) passing these intensity images as inputs to out trained PhENN, and (4) comparing the output to ground truth.

3.3 Results and network analysis

The standard method of characterizing neural network training is by plotting the progression of training and test error across training epochs (iterations in the back-propagation algorithm over all examples). These curves are shown in Fig. 3-4 for our network trained using the ImageNet database and tested using images from: (a) Faces-LFW (b) a disjoint ImageNet set, (c) images from an English/Chinese/Arabic characters database, (d) the MNIST handwritten digit database, (e) Faces-ATT, (f) CIFAR, (g) a constant-value "Null" image. Our ImageNet learning curves in Fig. 3-4(d) show convergence to low value after $\sim$10 epochs, indicating that our network

has not overfit to our training dataset. We plot bar graphs for the mean absolute error (MAE) over test examples in the 7 different datasets for each of the 3 object-to-sensor distances in Fig. 3-4. Lower MAE was reported for test images with large patches of constant value (characters, digits, Null) as their sparse diffraction patterns were easier for our PhENN to invert. Notably, both our bar graphs and learning curves show low test error for the non-trained images, suggesting that our network generalizes well across different domains.

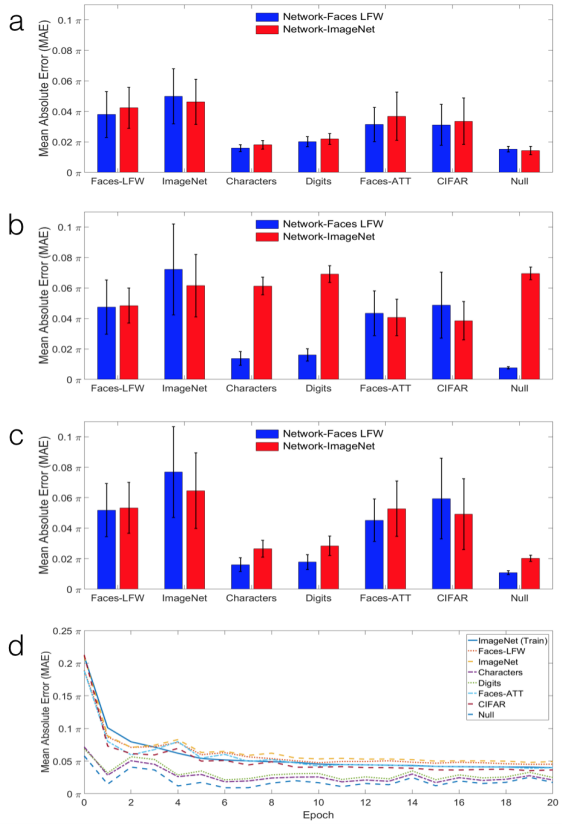

Figure 3-4: Quantitative analysis of our trained PhENNs for three object-to-sensor distances (a) z_1, (b) z_2, and (c) z_3 for the PhENNs trained on Faces-LFW (blue) and ImageNet (red) on 7 datasets. (d) The training and testing error curves for network trained on ImageNet at distance z_3 over 20 epochs.

This is an important point and worth emphasizing: despite the fact that our network was trained exclusively on images from the ImageNet database – i.e., images of planes, trains, cars, frogs, artichokes, etc., it is still able to accurately reconstruct images of a completely different class (e.g., faces, handwritten digits, and characters

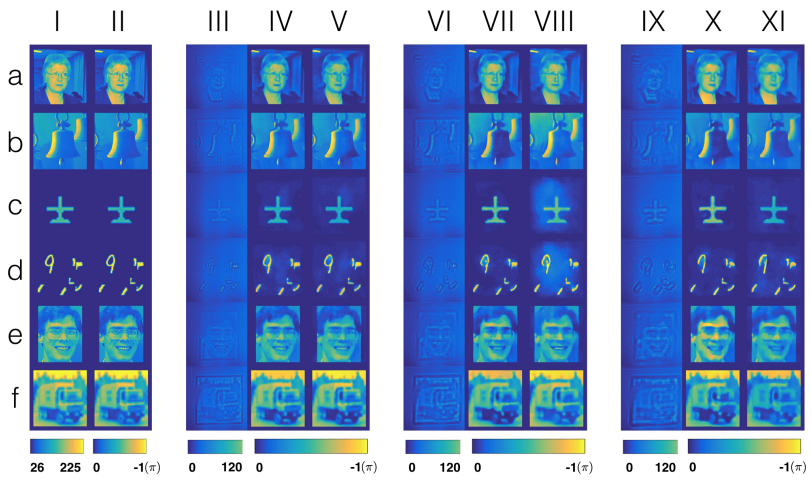

Figure 3-5: Qualitative analysis of our trained PhENNs for combinations of object-to-sensor distances z and training datasets. (i) Ground truth pixel value inputs to the SLM. (ii) Corresponding phase imaged calibrated by SLM response curve. (iii) Raw intensity images captured by CMOS detector at distance z_1. (iv) PhENN reconstruction from raw images when trained using Faces-LFW dataset. (v) PhENN reconstruction when trained used ImageNet dataset. Columns (vi-viii) and (ix-xi) follow the same sequence as (iii-v) but in these sets the CMOS is placed at a distance of z_2 and z_3, respectively. Rows (a-f) correspond to the dataset from which the test image is drawn: (a) Faces-LFW, (b) ImageNet, (c) Characters, (d) MNIST Digits, (e) Faces-ATT, or (f) CIFAR, respectively.

from different languages). This strongly suggests that our network has learned a model of the underlying physics of the imaging system or at the very least a generalizable mapping of low-level textures between our output diffraction patterns and input images.

A more pronounced qualitative example demonstrating this is shown in the columns (iv) (vii) and (x) of Fig. 3-5. Here, we trained our network using images exclusively from the Faces-LFW database. Despite this limited training set, the learned network was able to accurately reconstruct images from the ImageNet, handwritten digits, and characters datasets. This is in contrast to results shown in [88], where an SVM trained on images of faces was able to accurately reconstruct images of faces but not other classes of objects.

We tested the robustness of our network to rotation and lateral displacement in the presented test phase objects, as well as axial displacement of the CMOS camera for

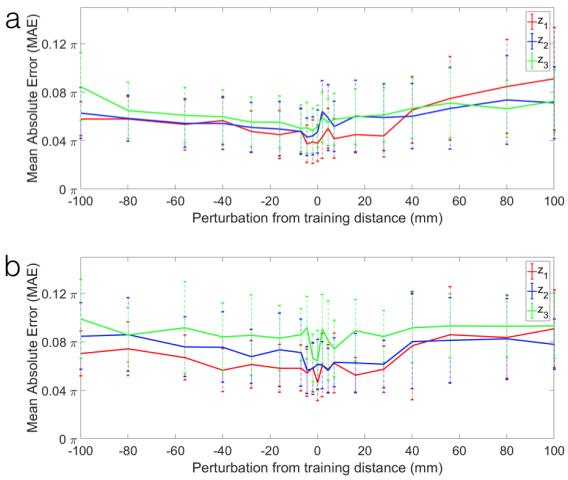

Figure 3-6: Quantitative analysis of the sensitivity of the trained PhENN to the object-to-sensor distance. The network was trained on (a) Faces-LFW database and (b) ImageNet and tested on disjoint Faces-LFW and ImageNet sets, respectively. The nominal depths of field for the three corresponding training distances z_1, z_2, z_3, respectively, are: $(\text{DOF})_1 = 1.18 \pm 0.1\text{mm}$, $(\text{DOF})_2 = 3.82 \pm 0.2\text{mm}$, and $(\text{DOF})_3 = 7.97 \pm 0.3\text{mm}$.

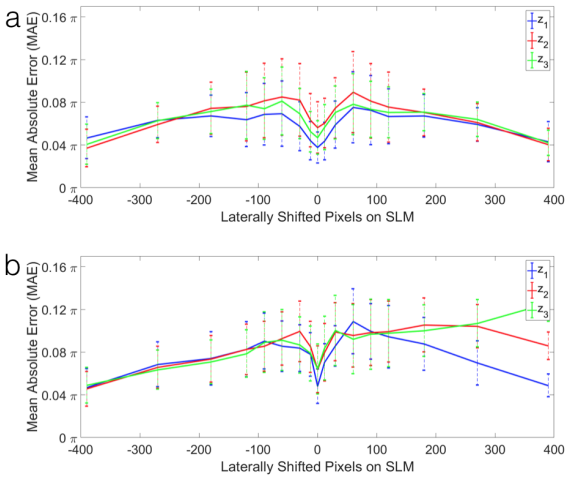

Figure 3-7: Quantitative analysis of the sensitivity of the trained PhENN to laterally shifted images on the SLM. The network was trained on (a) Faces-LFW database, (b) ImageNet and tested on disjoint Faces-LFW and ImageNet sets, respectively.

each PhENN relative to the PhENN's trained axial locations z_1, z_2, z_3, respectively. Quantitative results of these perturbations are shown in Figs. 3-6, 3-7, 3-8, and qualitative results for the networks trained at distance z_1 are shown in Figs. 3-9, 3-10 and 3-11. The results show that our trained network is robust to moderate

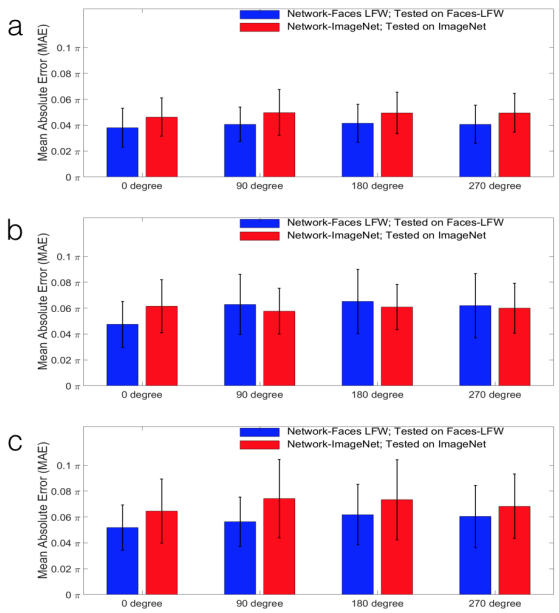

Figure 3-8: Quantitative analysis of the sensitivity of the trained PhENN to rotation of images on the SLM. The baseline distance on which the network was trained is (a) z_1, (b) z_2 and (c) z_3, respectively.

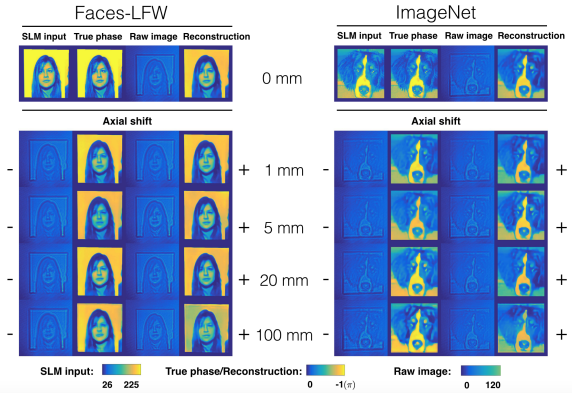

Figure 3-9: Qualitative analysis of the sensitivity of the trained PhENN to the object-to-sensor distance. The baseline distance on which the network was trained is z_1.

perturbations in sensor displacement and is somewhat shift and rotation invariant. As expected, the system fails when the displacement is significantly greater (Fig. 3-12).

To get a sense of what the network has learned, we examined its maximally-activated patterns (MAPs), i.e., what types of inputs would maximize network filter response (gradient descent on the input with average filter response as loss function

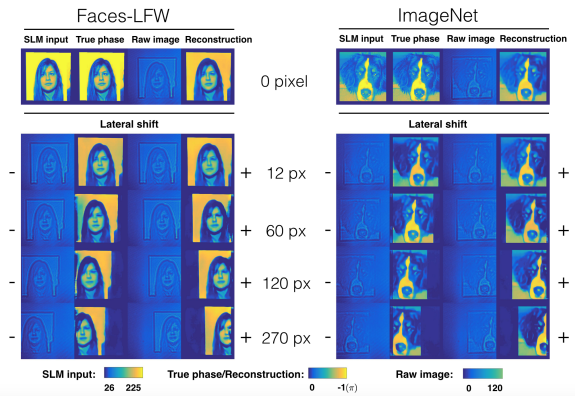

Figure 3-10: Qualitative analysis of the sensitivity of the trained PhENN to lateral shifts of images on the SLM. The baseline distance on which the network was trained is z_1.

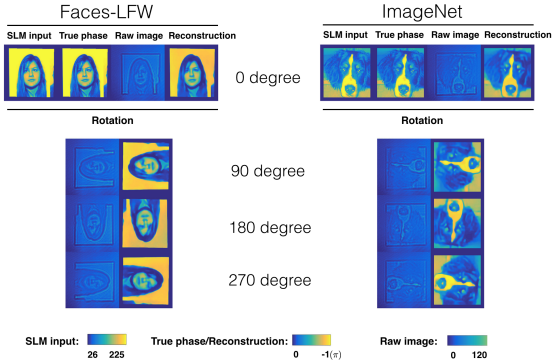

Figure 3-11: Qualitative analysis of the sensitivity of the trained PhENN to rotation of images in steps of 90. The baseline distance on which the network was trained is z_1.

[91]). Our results are shown in Fig. 3-13 together with the results of analogous analysis of a deblurring network of similar architecture as well as an ImageNet classification PhENN. Compared with MAPs of ImageNet and a deblurring network, the MAPs of our phase-retrieval network show two primary differences: First, compared to ImageNet, we observe much finer textures; this is because ImageNet is meant to do classification, a task that requires high-level features to emerge out of learning; whereas our phase network is performing a form of regression. Secondly, compared with the deblurring network, we observe somewhat finer textures, especially at the shallower layers (although in both cases low-level textures are present). Our interpretation is that both phase retrieval and deblurring require local operations but

of different nature: deblurring converts intensity gradients into sharp edges, whereas phase retrieval converts diffraction rings into phase gradients. The difference between the two cases is not easily discernible by simple visual inspection of the MAPs. The point is that phase retrieval and deblurring share this common locality feature while acting differently for the sake of their respective functions; and both are radically different than classification networks, such as ImageNet.

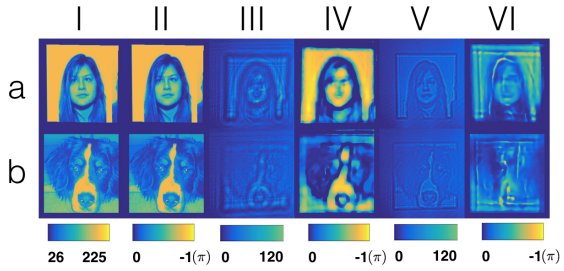

Figure 3-12: Failure cases on PhENNs trained on Faces-LFW (row a) and ImageNet (row b) datasets. (i) Ground truth input, (ii) calibrated phase input to SLM, (iii) raw image on camera (iv) reconstruction by PhENN trained on images at distance z_1 between SLM and camera and tested on images at distance 107.5 cm, (v) raw image on camera and (vi) reconstruction by network trained on images at distance z_3 between SLM and camera and tested on images at distance 27.5 cm.

3.4 Conclusions and discussion

The architecture presented here was deliberately well controlled, with an SLM creating the phase object inputs to the neural network for both training and testing. This allowed us to quantitatively and precisely analyze the behavior of the learning process. Application-specific training, e.g. replacing the SLM with physical phase objects for more practical applications, we judged beyond the scope of the present work. Other obvious and useful extensions would be to include optics, e.g. a microscope objective for microscopic imaging in the same mode; and to attempt to reconstruct complex objects, *i.e.* imparting both attenuation and phase delay to the incident light. The significant anticipated benefit in the latter case is that it would be unnecessary to characterize the optics for the formulation of the forward operator—the neural network should "learn" this automatically as well. We intend to undertake such studies

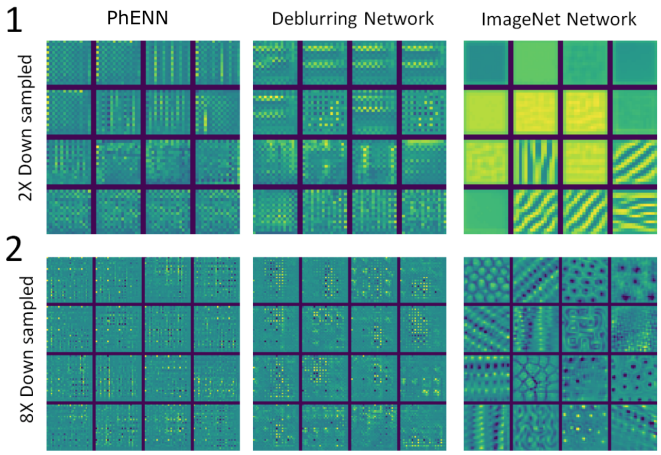

Figure 3-13: (1) 16×16 inputs that maximally activate the last set of 16 convolutional filters in layer 1 of our PhENN trained on ImageNet at distance of z_1, a deblurring network, and an ImageNet classification network. The deblurring network was trained on images undergoing motion blur in a random angle within the range [0,180] degrees and a random blur length in the range [10,100] pixels. The image is downsampled by a factor of 2 in this layer. (2) 32×32 inputs that maximally activate the last set of 16 randomly chosen convolutional filters in layer 3 of: our PhENN, the same deblurring network, and the ImageNet classification network. The raw image is downsampled by a factor of 8 in this layer.

in future work.

3.5 My contributions

The work in this chapter was led by Ayan Sinha. My contribution was in building up the experimental setup, performing the SLM calibration and analyzing the data.

Chapter 4

Analysis of the dependence of PhENN's performance on its architecture

4.1 Introduction

In the previous two chapters, we introduced two deep neural networks (DNNs): IDiffNet and PhENN, and demonstrated their performances in two different computational imaging scenarios: imaging through scattering media and quantitative phase retrieval, respectively. When the neural networks are trained, instead of minimizing the Tikhonov functional (Eq.1.4), the object estimate is then readily obtained as

$$\hat{f} = \mathrm{DNN}(g), \tag{4.1}$$

where DNN(.) denotes the output of the trained deep neural network.

Just as the performance of minimization principle Eq. (1.4) depends upon knowledge of the operators H and Φ, performance of the DNN principle Eq. (4.1) depends on the specific DNN architecture chosen (number of layers, connectivity, etc.) and the quality of the training examples. Therefore, start from this chapter, we set out to investigate these two dependences. We chose to study this question in the specific

context of quantitative phase retrieval, but our findings might have merit for several other challenging imaging problems.

In this chapter, we first analyze the dependence of PhENN's phase recovery performance on its architecture. Specifically, we analyze the influences of the training loss functions, the skip connections, the depth of the neural network, and the waist size. Through this study, we wish to provide some insights about how to design an optimal neural network architecture for quantitative phase retrieval. The influence of the training example quality to the performance of PhENN will be studied in the next chapter.

4.2 Methods

4.2.1 Default PhENN architecture

The architecture of PhENN that we use here is shown in Fig. 4-1. Similar to Chapter 3, it follows the U-net architecture [98] and uses residual units [130] to facilitate learning. PhENN is composed of three different kinds of blocks: down-residual blocks (DRBs), up-residual blocks (URBs) and residual blocks (RBs). The default downsampling rate (DSR) of each DRB and the default upsampling rate (USR) of each URB are 2. In our PhENN, the numbers of DRBs and URBs are made to be equal (denoted as N) and different values of N are used for comparison. The number of RBs is fixed to 2. Skip connections can be used in PhENN to pass the information extracted at shallow layers to deep layers.

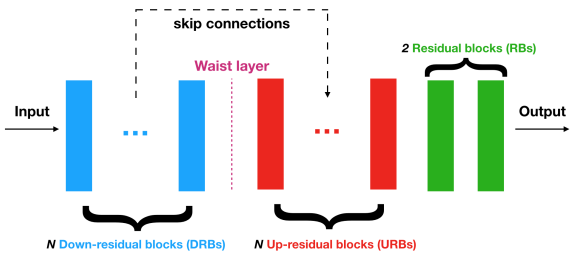

Figure 4-1: Phase Extraction Neural Network (PhENN) architecture.

4.2.2 Data preparation

In this paper, all the neural networks are trained and tested using synthetic data. In generating the training dataset, we select $10,000$ images from the Faces-LFW database [99]. For each image, we make it to be $f(x, y)$ and insert it into Eq. (3.1) to compute the corresponding intensity measurement $g(x, y)$. In this simulation, we set $\lambda = 0.633\mu$m. Both the pixel sizes of the object and the detector are set as $\Delta x = 20\mu$m. Different values of z are used for comparison. The testing dataset is generated in the same way (at the same distance z as the training data) and consists of 450 images chosen from 6 different databases: 100 Faces-LFW, 100 ImageNet [100], 50 Characters [102], 40 Faces-ATT [104, 105], 60 CIFAR [103] and 100 MNIST [101].

4.3 Results

4.3.1 Choice of training loss function

In this section, we compare the performances of PhENN under different training loss functions. The architecture of PhENN is kept to be the same: $N = 4$ and skip connections are used. In generating the training and testing dataset, we set $z = 200$mm and the space-bandwidth product (SBP) of the object to be 256×256.

Altogether three different training loss functions are considered here: the mean absolute error (MAE), the structural similarity (SSIM) and the negative Pearson correlation coefficient (NPCC). MAE is defined as Eq.2.5, NPCC is defined as Eq.2.6 and SSIM is defined as following [132]:

$$\text{SSIM} = \frac{-1}{b} \times \sum_{k=1}^{b} \frac{\left(2\mu_{f_k}\mu_{\hat{f}_k} + c_1\right)\left(2\sigma_{f_k\hat{f}_k} + c_2\right)}{\left(\mu_{f_k}^2 + \mu_{\hat{f}_k}^2 + c_1\right)\left(\sigma_{f_k}^2 + \sigma_{\hat{f}_k}^2 + c_2\right)}, \tag{4.2}$$

Here, f_k and $\hat{f}_k$ are the kth true object and the object estimate, respectively; b is the batch size; μ_{f_k} and $\mu_{\hat{f}_k}$ are the spatial averages of f_k and $\hat{f}_k$, respectively; σ_{f_k} and $\sigma_{\hat{f}_k}$ are the standard deviations of f_k and $\hat{f}_k$, respectively; $\sigma_{f_k\hat{f}_k}$ denotes the covariance of f_k and $\hat{f}_k$; $c_1 = 0.01^2$ and $c_2 = 0.03^2$.

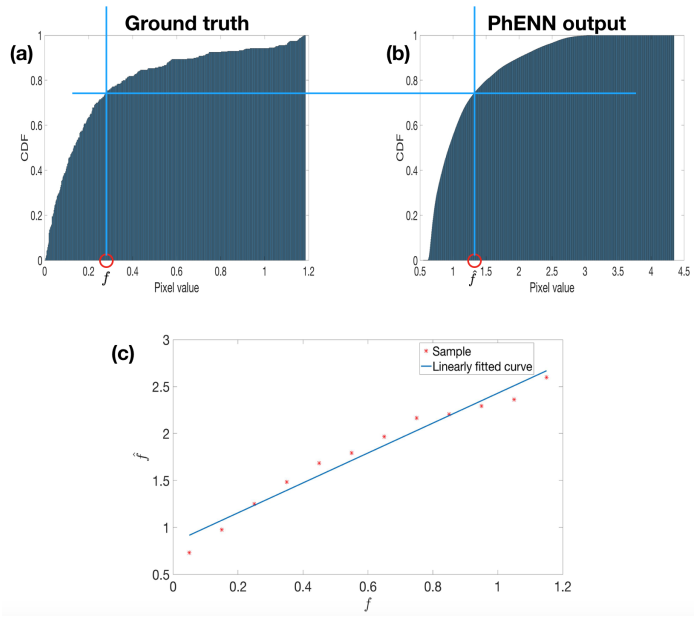

Figure 4-2: Calibration process. (a) Cumulative distribution function (CDF) of the ground truth. (b) Cumulative distribution function (CDF) of the PhENN output. (c) Linear curve fitting.

From the definition of NPCC [Eq. (2.6)], it follows that for any function ψ and arbitrary real constants a and b representing linear amplification and bias, respectively,

$$\mathcal{E}_{\text{NPCC}}(\psi, a\psi + b) = -1. \tag{4.3}$$

In other words, a DNN trained with NPCC as loss function can only produce affine transformed estimates; there is no way to enforce the requirement $a = 1$, $b = 0$ which would guarantee linear amplification- and bias-free reconstruction and is especially important for quantitative phase retrieval. Neither does there exist a way that we know of to predetermine the values of a and b through specific choices in DNN training.

Therefore, after DNN training a calibration step is required to determine the values of a and b that have resulted so that they can be compensated. This is realized by histogram matching according to the process shown in Fig. 4-2. Given a set of calibration data, we compute the cumulative distribution functions (CDFs) for the ground truth values as well as the PhENN output values, as shown in Figs. 4-2(a) and 4-2(b). For an arbitrary value f in the ground truth, we find its corresponding

PhENN output value $\hat{f}$ that is at the same CDF level; and repeat the process for several $(f, \hat{f})$ samples. Subsequently, the values of a and b are determined by linear fitting of the form $\hat{f} = af + b$, as shown in Fig. 4-2(c).

The phase retrieval results of PhENNs trained with different loss functions are shown in Fig. 4-3. There are two important observations:

- The trained PhENN peforms well on test samples drawn from the same dataset as the training data (Faces-LFW), as well as test samples drawn from other categories. This held true for all training loss functions used. This result indicates that PhENN has indeed learned a model of the underlying physics of the imaging system or at the very least a generalizable mapping of low-level textures between the phase objects and their respective intensity images.

- Among the three training loss functions that were compared, NPCC worked better in retrieving finer features, as highlighted by the color boxes in Fig. 4-3.

In the subsequent sections, the training loss function used is NPCC.

4.3.2 Presence of skip connections

In this section, we investigate the function of the skip connections in the PhENN architecture and demonstrate the necessity of their presence. Here, we set $N = 6, z = 200\text{mm}, \text{SBP} = 256 \times 256$.

Intuitively, the function of a skip connection is to pass the high frequency information (local features) extracted at a shallow layer to a deeper layer. Without those skip connections, the reconstruction will entirely depend on the feature map being extracted at the waist, which mostly contains low frequency information due to the downsampling. This is verified by the test results as shown in Fig. 4-4. With skip connections, the phase reconstructions show much more details than their counterparts in the no skip connection case. Using the Pearson correlation coefficient (PCC) as the evaluation metric, Fig. 4-5 quantitatively compares the performances of PhENNs with and without skip connections. As expected, all the test results obtained with skip connections have a higher PCC.

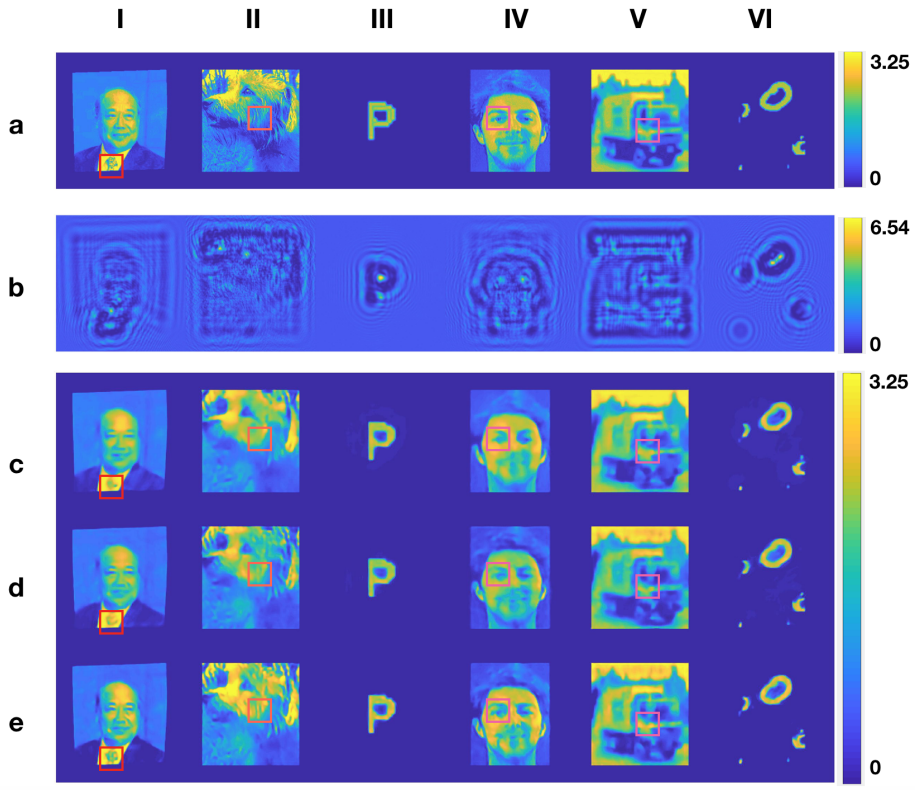

Figure 4-3: Qualitative comparison of reconstructions using different training loss functions. (a) Ground truth phase objects. (b) Raw intensity measurements. (c) Reconstructions when PhENN is trained with MAE. (d) Reconstructions when PhENN is trained with SSIM. (e) Reconstructions when PhENN is trained with NPCC. Columns (i-vi) correspond to the dataset from which the test image is drawn: (i) Faces-LFW, (ii) ImageNet, (iii) Characters, (iv) Faces-ATT, (v) CIFAR, or (vi) MNIST Digits, respectively.

Skip connections are maintained in the PhENN architecture in subsequent analyses.

4.3.3 Influence of depth

Now, we study the influence of the depth N on the performance of PhENN. We set $z = 200$mm and consider two different SBPs for the objects: 256×256 and 128×128.

From the results depicted in Fig. 4-6, we can make the following three observations:

- The phase recovery performance for objects of SBP$= 128 \times 128$ is better than

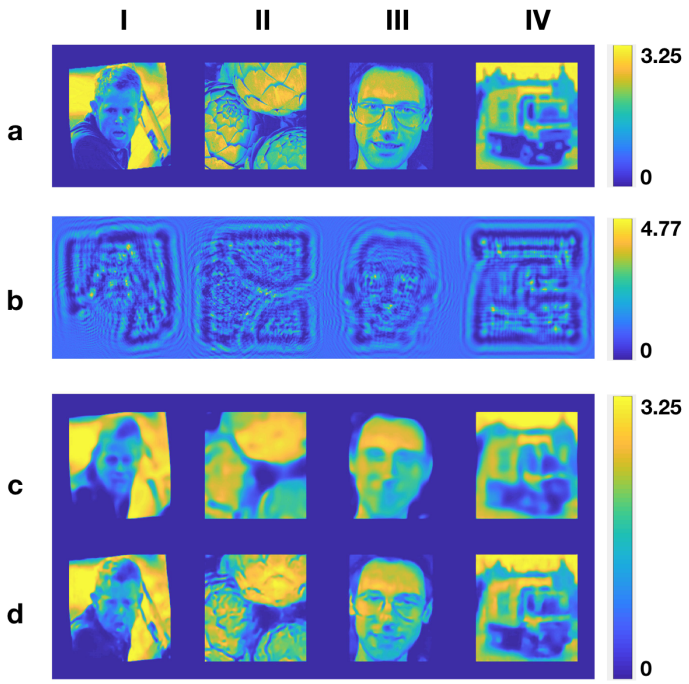

Figure 4-4: Qualitative comparison of reconstructions with and without skip connections. (a) Ground truth phase objects. (b) Raw intensity measurements. (c) Reconstructions with no skip connections in PhENN. (d) Reconstructions with skip connections present in PhENN. Columns (i-iv) correspond to the dataset from which the test image is drawn: (i) Faces-LFW, (ii) ImageNet, (iii) Faces-ATT, or (iv) CIFAR, respectively.

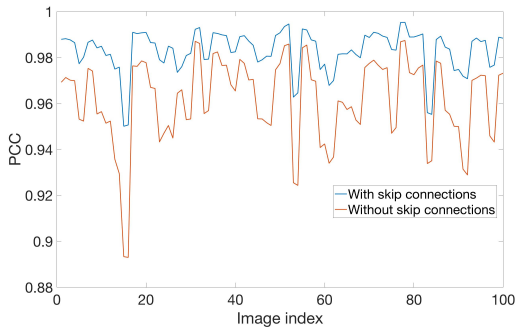

Figure 4-5: Quantitative comparison of reconstructions with and without skip connections using 100 Faces-LFW test images.

that for objects of SBP$= 256 \times 256$, in terms of PCC. This result is expected since the objects with smaller SBP have fewer entries to be estimated.

- As the depth increases, the phase recovery performance will saturate at some depth. We denote this depth as 'saturating depth.'

- The saturating depth depends on the SBP of the input. Specifically, the saturating depth for SBP$= 256 \times 256$ is one block deeper than that for SBP$= 128 \times 128$.

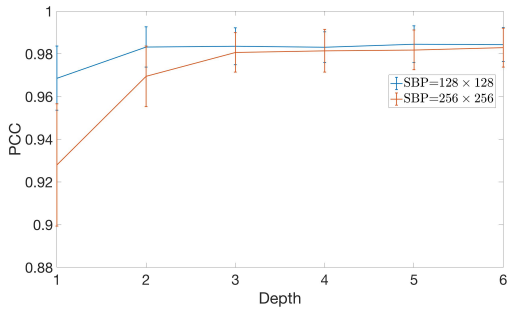

Figure 4-6: Quantitative analysis of the influence of the depth using 100 Faces-LFW test images.

4.3.4 Influence of waist size

As shown in Fig. 4-1, we refer the feature map extracted by the last DRB as the waist layer. The size of the waist layer (waist size) can be controlled by tuning the DSR of the last DRB. Accordingly, the USR of the first URB needs to be tuned to the same value. For example, when $N = 4$ and SBP $= 256 \times 256$, to achieve a waist size of 4×4, both the DSR of the last DRB and the USR of the first URB should be set to 8. Here, we consider four different values for the waist size to investigate its influence: 1×1, 2×2, 4×4 and 8×8. Other hyper-parameters are set as: $N = 4$, $z = 200$mm and SBP $= 256 \times 256$.

From the results shown in Fig. 4-7, we find that the waist size has a minor impact on the phase recovery performance. One possible explanation for this phenomenon is that most of the information extracted by the second to last DRB transmits to deeper layers through skip connections rather than through the last DRB.

4.4 Conclusions

In this paper, we investigate the dependence of PhENN's performance on its architecture as well as the layout of the lensless imaging system. Our observations can be

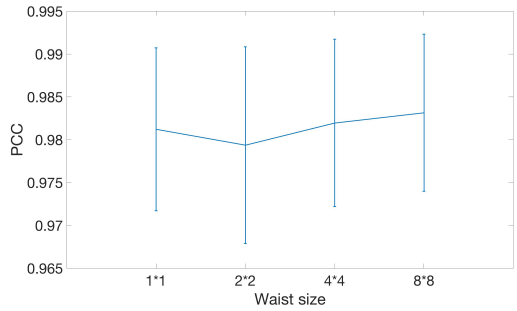

Figure 4-7: Quantitative analysis of the influence of the waist size using 100 Faces-LFW test images.

summarized as follows:

- The NPCC loss function is better at retrieving the fine features of the phase object compared with the MAE and SSIM loss functions.
- Skip connections are important as they pass high frequency information to deeper layers.
- As the depth of PhENN increases, the phase recovery performance saturates at some saturating depth and this saturating depth depends on the SBP of the object.
- In an architecture with skip connections, the waist size does not influence PhENN's performance significantly.

Chapter 5

Resolution enhancement of PhENN by spectral pre-modulation

5.1 Introduction

In this chapter, we begin to study the dependence of PhENN's phase recovery performance on the quality of the training examples. More specifically, we are concerned with the spatial resolution that PhENN can achieve, depending on the spatial frequency content of the examples PhENN is trained with.

From the point of view of the original inverse problem formulation Eq. (1.4), PhENN in effect has to learn both the forward operator H and the prior Φ at the entire range of spatial frequencies of interest. The examples presented to PhENN during training establish the spatial frequency content that is stored in the network weights contributing to the retrieval operation Eq. (4.1). In principle, this should be sufficient because, if the training examples are representative enough of the object class, then retrieval of each spatial frequency should be learnt proportionally to that spatial frequency's presence in the database. In practice, however, we found that spatial frequencies with relatively low representation in the database tend to be overshadowed by the more popular spatial frequencies, perhaps due to the nonlinearities in the network training process and operation.

Invariably, high spatial frequencies tend to be less popular in most available

databases. ImageNet, in particular, exhibits the well-known inverse-square power spectral density of natural images [133], as we verify in Fig. 5-5. This means that high spatial frequencies are inherently under-represented in PhENN training. Compounded by the nonlinear suppression of the less popular spatial frequencies due to PhENN nonlinearities, as mentioned above, this results in low-pass filtering of the estimates and loss of fine detail. Detailed analysis of this effect is presented in Section 5.3.

To better recover high spatial frequencies in natural objects then, one should emphasize high spatial frequencies more during training; this may be achieved, for example, by flattening the power spectral density of the training examples *before* they are presented to the neural network. It would appear that this spectral intervention violates the object class priors: PhENN does not learn the priors of ImageNet itself, it rather learns an edge-enhanced version of the priors. Yet, in practice, again probably because of nonlinear PhENN behavior, we found this spectral pre-modulation strategy to work quite well. The detailed approach and results are found in Section 5.4.

5.2 Imaging system architecture

5.2.1 Optical configuration

Our optical configuration is shown in Fig. 5-1. Unlike Chapter 3, a transmissive spatial light modulator (SLM) (Holoeye, LC2012, pixel size 36μm) is used in this system as a programmable phase object f representing the ground truth. The transmissive SLM is coherently illuminated by a He-Ne laser light source (Research Electro-Optics, Model 30995, 633nm). The light is transmitted through a spatial filter consisting of a microscope objective (Newport, M-60X, 0.85NA) and a pinhole aperture ($D = 5\mu$m) and then collimated by a lens (focal length 200mm) before illuminating the SLM. A telescope consisting of two plano-convex lenses L_1 and L_2 is placed between the SLM and a CMOS camera (Basler, A504k, pixel size 12μm). The CMOS camera captures the intensity g of the diffraction pattern produced by the SLM at a defocus

distance $\Delta z = 50$mm. The focal lengths of L_1 and L_2 are set to $f_1 = 150$mm and $f_2 = 50$mm, respectively. As a result, this telescope demagnifies the object by a factor of 3, consistent with the ratio between SLM and CMOS camera pixel sizes. An iris with diameter 5mm is placed at the pupil plane of the telescope to keep the 0^{th} diffracted order of the SLM and filter out all the other orders.

The modulation performance of the SLM depends on the input and output polarizations, which are controlled by the polarizer P and the analyzer A, respectively. In order to realize phase-mostly modulation, we set the incident beam to be linearly polarized at 310° with respect to the vertical direction and also set the analyzer to be oriented at 5° with respect to the vertical direction. The specific calibration curves for the SLM's modulation performance are shown in Appendix Section A.3.

In this chapter, all the training and testing objects are of size 256×256. They are zero-padded to the size 1024×768, before being uploaded to the SLM. For the diffraction patterns captured by the CMOS camera, we crop the central 256×256 region for processing.

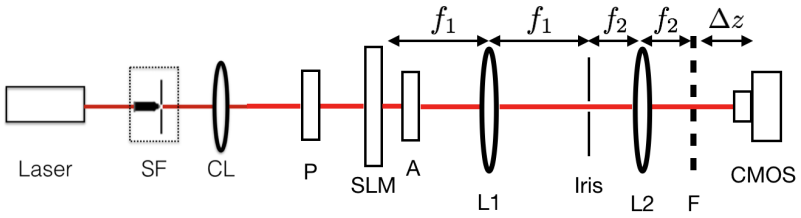

Figure 5-1: Optical configuration. SF: spatial filter; CL: collimating lens; P: linear polarizer; A: analyzer; SLM: spatial light modulator; L1 and L2: plano-convex lenses; F: focal plane of L2.

5.2.2 Neural network architecture and training

According to the analysis in the last chapter, the PhENN architecture that we implement in this chapter is shown in Fig. 5-2, which consists 4 down-residual blocks (DRBs), 4 up-residual blocks (URBs) and 2 residual blocks (RBs). Skip connections are used in the architecture to pass downstream local spatial information learnt in the initial layers. The training loss function used is NPCC.

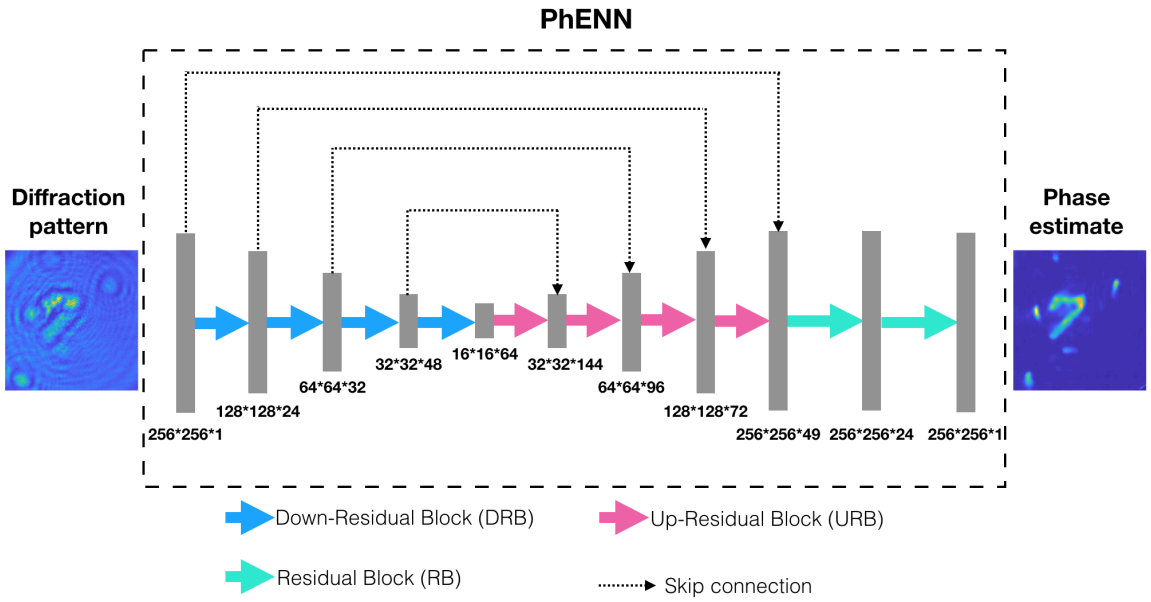

Figure 5-2: Phase extraction neural network (PhENN) architecture.

5.3 Resolution analysis of ImageNet-trained PhENN

In Chapter 3, we trained separate PhENNs using the databases Faces-LFW [99] and ImageNet [100] and found that both PhENNs generalize to test objects within and outside these two databases. In this chapter, we restrict our analysis to the ImageNet database only.

In the PhENN training phase, a total of $10,000$ images selected from the ImageNet database are uploaded to the SLM and the respective diffraction patterns are captured by the CMOS. For testing, we use a total of 471 images selected from several different databases: 50 Characters, 40 Faces-ATT [104, 105], 60 CIFAR [103], 100 MNIST [101], 100 Faces-LFW, 100 ImageNet, 20 resolution test patterns [dot patterns as described in Section 6.3], and 1 all-zero (dark) image. The diffraction pattern corresponding to the all-zero image is used as the background. For every test diffraction pattern that we capture, we first subtract the background and then normalize, before feeding into the neural network.

5.3.1 Reconstruction results

The phase reconstruction results are shown in Fig. 5-3. Here, we use 100 ImageNet test images as calibration data to compensate for the unknown affine transform effected by the NPCC-trained PhENN (Section 4.3.1). As expected, PhENN is not only able to quantitatively reconstruct the phase objects within the same category as its training database (ImageNet), but also to retrieve the phase for those test objects from other databases. This indicates that PhENN has indeed learned a model of the underlying physics of the imaging system or at the very least a generalizable mapping of low-level textures between the phase objects and their respective diffraction patterns.

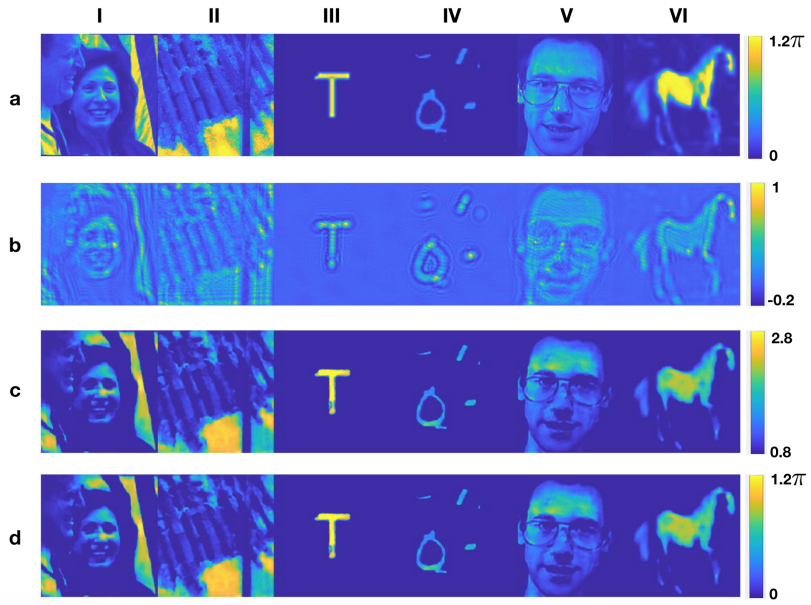

Figure 5-3: Reconstruction results of PhENN trained with ImageNet. (a) Ground truth for the phase objects. (b) Diffraction patterns captured by the CMOS (after background subtraction and normalization). (c) PhENN output. (d) PhENN reconstruction after the calibration shown in Section 4.3.1. Columns (i-vi) correspond to the dataset from which the test image is drawn: (i) Faces-LFW [99], (ii) ImageNet [100], (iii) Characters, (iv) MNIST Digits [101], (v) Faces-ATT [104, 105], or (vi) CIFAR [103], respectively.

5.3.2 Resolution test

In order to test the spatial resolution our trained PhENN, we use dot patterns as test objects, shown in Fig. 5-4(a). Altogether 20 dot patterns are tested, with spacing D between dots gradually increasing from 2 pixels to 21 pixels. From the resolution test results shown in Fig. 5-4 it can be observed that the PhENN trained with ImageNet is able to resolve two dots down to $D = 6$ pixels but fails to distinguish two dots with spacing $D \leq 5$ pixels. Thus, $D \approx 6$ pixels can be considered as the Rayleigh resolution limit of this PhENN for point-like phase objects.

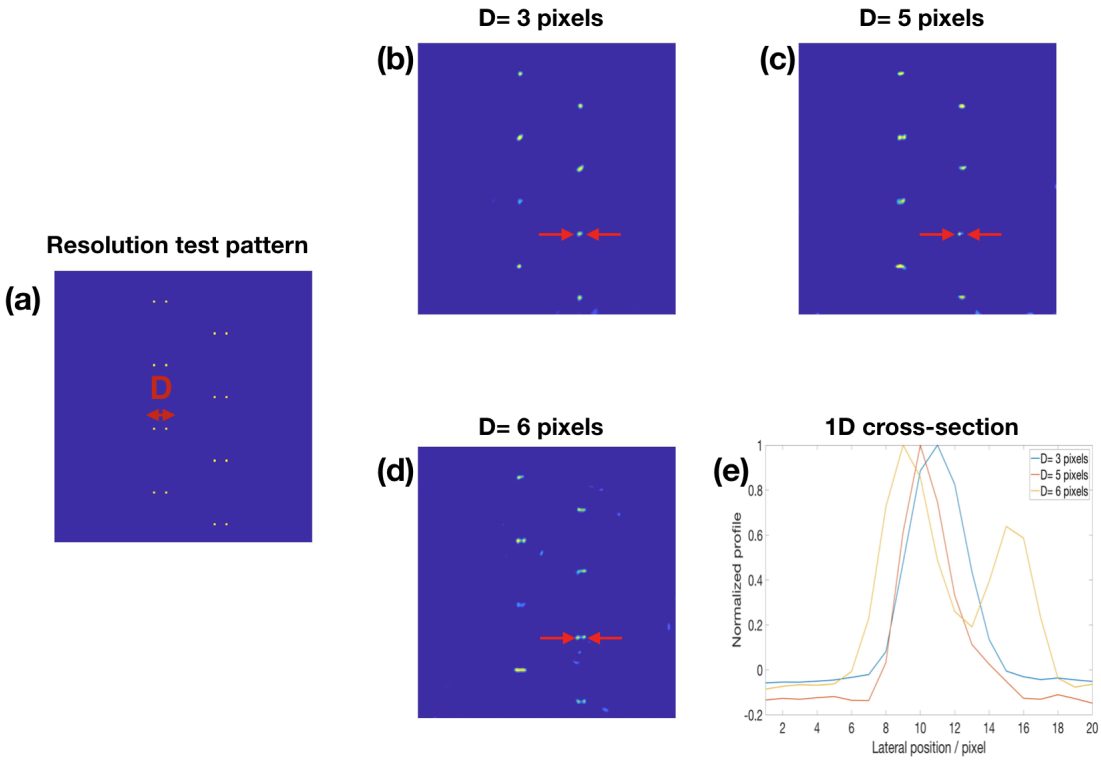

Figure 5-4: Resolution test for PhENN trained with ImageNet. (a) Dot pattern for resolution test. (b) PhENN reconstructions for dot pattern with $D = 3$ pixels. (c) PhENN reconstructions for dot pattern with $D = 5$ pixels. (d) PhENN reconstructions for dot pattern with $D = 6$ pixels. (e) 1D cross-sections along the lines indicated by red arrows in (b)-(d).

5.4 Resolution enhancement by spectral pre-modulation

In our imaging system, the SLM pixel size limits the spatial resolution of the trained PhENN since the minimum sampling distance in all the training and testing objects displayed on the SLM equals one pixel $d_p = 36\mu$m, or maximum spatial frequency[134] 13.9mm^{-1}. However, as we saw in Section 5.3.2, the resolution achieved by our PhENN trained with ImageNet database is merely 6 pixels (216μm), much worse than the theoretical value.

The additional factor limiting the spatial resolution of the trained PhENN is the spatial frequency content of the training database. Generally, databases of natural objects, such as natural images, faces, hand-written characters, etc. do not cover the entire spectrum up to $1/(2d_p)$. For example, below we analyze the ImageNet database and show that it is dominated by low spatial frequency components, with the prevalence of higher spatial frequencies decreasing quadratically.

During training, the neural network learns the particular prevalence of spatial frequencies in the training examples as prior Φ, in addition to learning the physical forward operator H. What this implies is that the less prevalent spatial frequencies are actually learnt *against*, meaning that by presenting them less frequently we may be teaching PhENN to suppress or ignore them. In the rest of this section, we present evidence to corroborate this fact, and suggest as solution a pre-processing step that edge enhances the training examples as a way to impress their importance better upon PhENN.

5.4.1 Spectral pre-modulation

The 2D power spectral density (PSD) $S(u, v)$ as function of spatial frequencies u and v for the $10,000$ images in the ImageNet is shown in Figs. 5-5(a) and 5-5(b) in linear and logarithmic scales, respectively; and in cross-section along the spatial frequency u in Figs. 5-5(c) and 5-5(d). Not surprisingly [133], the cross-sectional power spectral

density follows a power law of the form $|u|^p$ with $p \approx -2$.

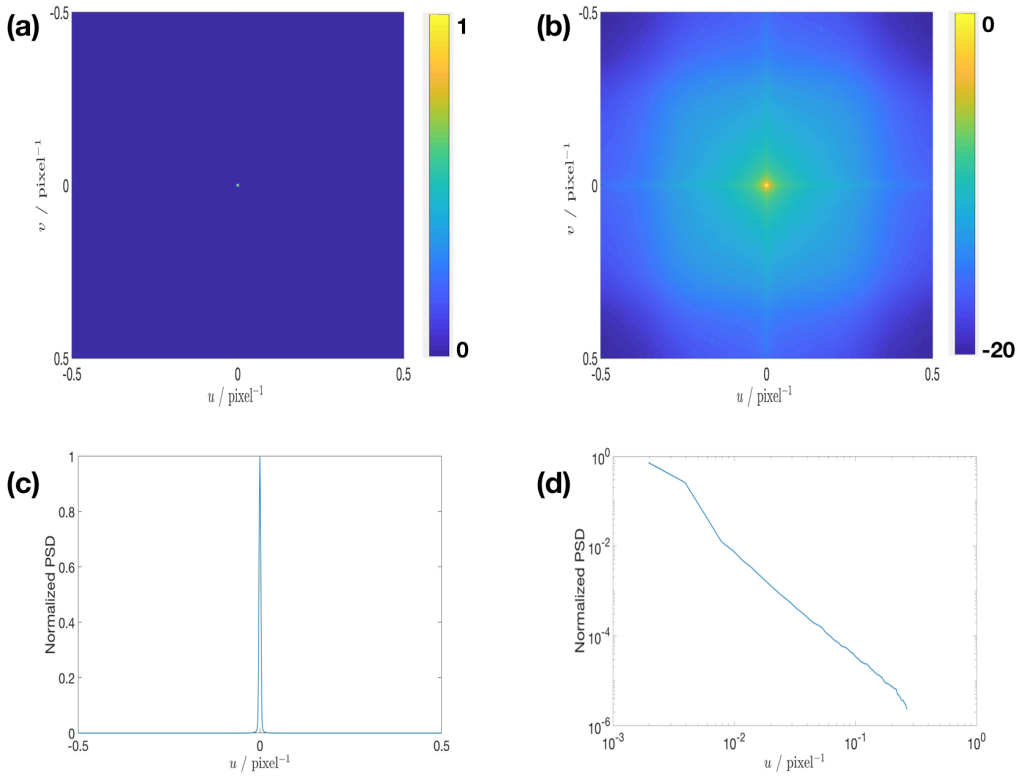

Figure 5-5: Spectral analysis of the ImageNet database. (a& b) 2D normalized power spectral density (PSD) of the ImageNet database in linear and logarithmic scale. (c& d) 1D cross-sections along the spatial frequency u of (a& b), respectively.

Therefore, we may approximately represent the 2D PSD of ImageNet database as

$$S(u,v) \propto \left(\sqrt{u^2+v^2}\right)^{-2} = \frac{1}{u^2+v^2}. \tag{5.1}$$

This is flattened by the inverse filter

$$G(u,v) = \sqrt{u^2+v^2}. \tag{5.2}$$

As expected, the high spatial frequency components in the image are amplified after the modulation, as can be seen, for example, in Fig. 5-6.

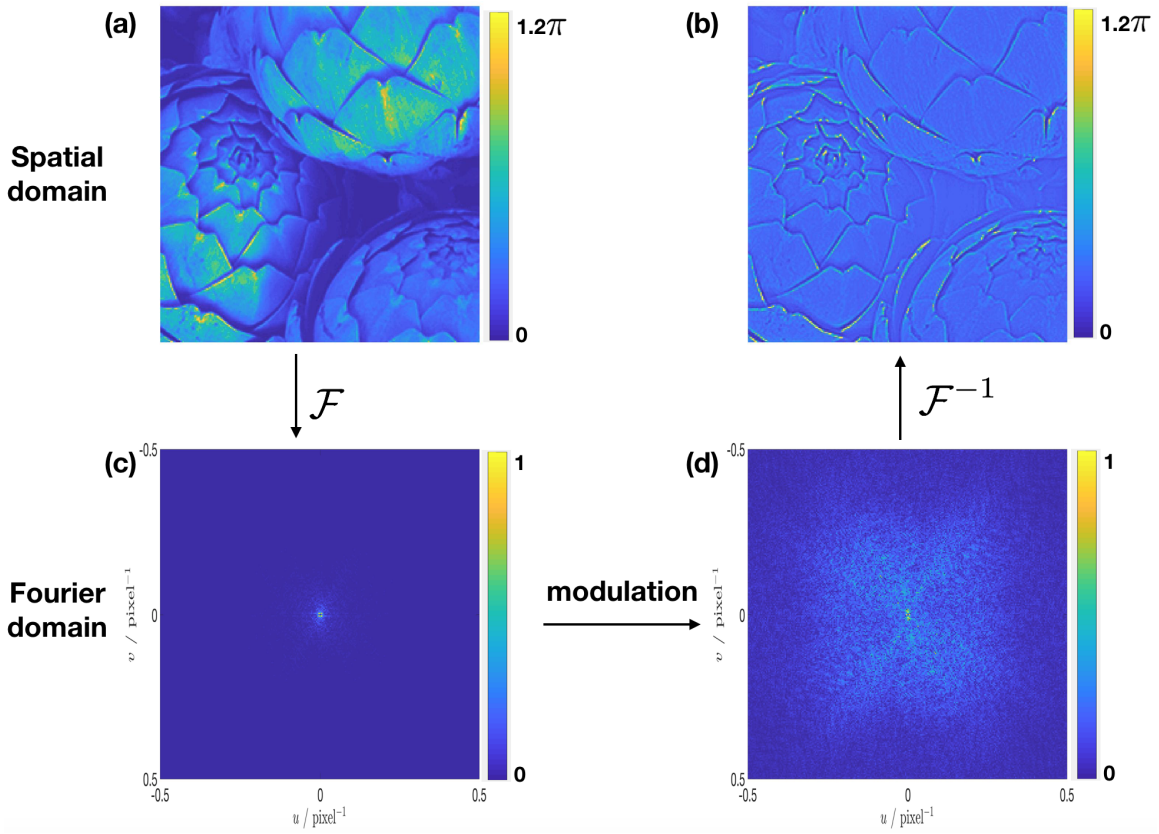

Figure 5-6: Spectral pre-modulation. (a) Original image [100]. (b) Modulated image. (c) Fourier spectrum of the original image. (d) Fourier spectrum of the modulated image.

5.4.2 Resolution enhancement

We trained a new PhENN using training examples that were spectrally pre-modulated according to Eq. (5.2). That is, we replaced every training example $f(i,j)$ with $f_{\mathrm{e}}(i,j)$, where

$$F_{\mathrm{e}}(u,v) = G(u,v)F(u,v) \tag{5.3}$$

and F, F_{e} are the Fourier transforms of f, f_{e}, respectively. We also collected the corresponding diffraction patterns $g_e(i,j)$. The test examples were left without modulation, *i.e.* the same as in the original use of PhENN described in Section 5.3. All the training parameters were also kept the same. Both dot pattern and ImageNet test images were used to demonstrate the resolution enhancement, shown in Figs. 5-7 and 5-8, respectively.

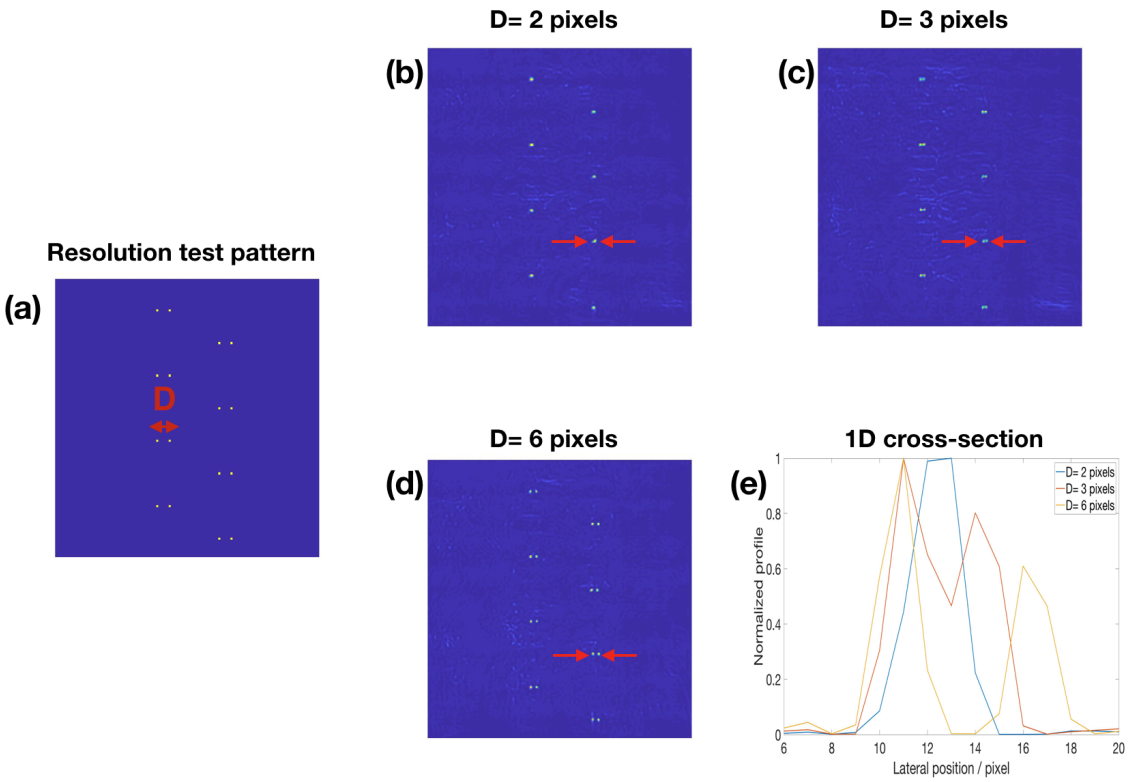

Figure 5-7: Resolution test for PhENN trained with examples from the ImageNet database with spectral pre-modulation according to Eq. (5.3). (a) Dot pattern for resolution test. (b) PhENN reconstructions for dot pattern with $D = 2$ pixels. (c) PhENN reconstructions for dot pattern with $D = 3$ pixels. (d) PhENN reconstructions for dot pattern with $D = 6$ pixels. (e) 1D cross-sections along the lines indicated by red arrows in (b)-(d).

From Fig. 5-7, we find that with spectral pre-modulation of the training examples according to Eq. (5.3), PhENN is able to resolve two dots with spacing $D = 3$ pixels. Compared with the resolution test results shown in Fig. 5-4, it can be said that the spatial resolution of PhENN has been enhanced by a factor of 2 with the spectral pre-modulation technique. In Fig. 5-8, for the same test image selected from ImageNet database, more details are recovered by the PhENN that was trained with spectrally pre-modulated ImageNet, albeit at the cost of amplifying some noisy features of the object, near edges most notably.

We also investigated the effect of spectral post-modulation in the original PhENN; that is, if we use a PhENN trained without spectral pre-modulation, and modulate

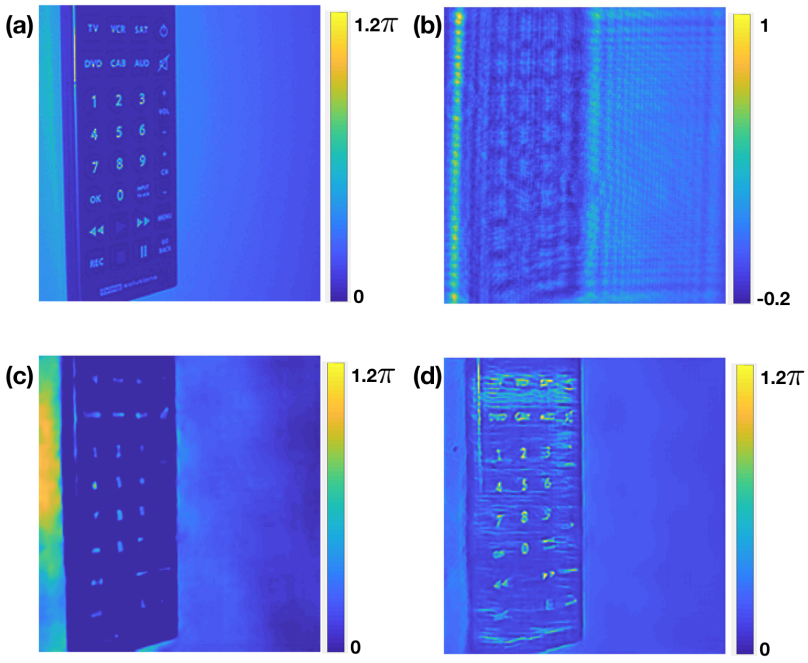

Figure 5-8: Resolution enhancement demonstration. (a) Ground truth for a phase object [100]. (b) Diffraction pattern captured by the CMOS (after background subtraction and normalization). (c) Phase reconstruction by PhENN trained with ImageNet examples. (d) Phase reconstruction by PhENN trained with ImageNet examples that were spectrally pre-modulated according to Eq. (5.3).

the PhENN output $\hat{f}(i,j)$ according to

$$\hat{F}_{\mathrm{e}}(u,v) = G(u,v)\hat{F}(u,v) \tag{5.4}$$

and $\hat{F}$, $\hat{F}_{\mathrm{e}}$ are the Fourier transforms of $\hat{f}$, $\hat{f}_{\mathrm{e}}$, respectively, do we obtain a similar resolution enhancement? The answer is no, as can be clearly verified from the results of Fig. 5-9.

This negative result illustrates that in the original training scheme (without spectral pre-modulation) the fine details are indeed lost and not recoverable by simple means, e.g. linear post-processing. It also highlights the effect of the nonlinearity in PhENN's operation and bolsters our claim that spectral pre-modulation does something non-trivial: it teaches PhENN a prior, namely how to recover high spatial frequency content.

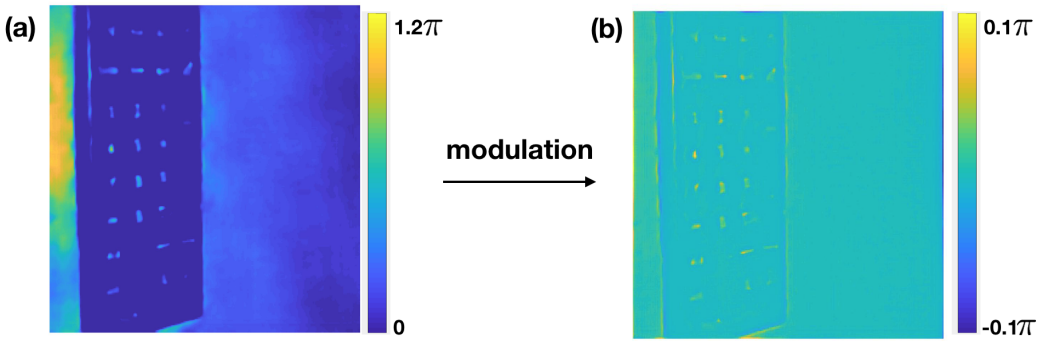

Figure 5-9: Spectral post-modulation. (a) Output of PhENN trained with ImageNet. The same as Fig. 5-8 (c). (b) Modulated output.

5.5 Conclusions

The spectral flattening approach Eq. (5.3) as pre-modulation is a simple approach that we found to be effective in enhancing PhENN's resolution by a factor of 2 when trained and tested on ImageNet examples. We have not investigated the performance of other (non-flattening) filters; indeed, it would be an interesting theoretical question to ask: given a particular form of the PSD in the training examples, what is the optimal spectral pre-modulation for improving spatial resolution?

It is also worth repeating the concern about the priors that PhENN is learning from the spatially pre-modulated examples that we pointed out in Section 5.1. The amplification of certain noise artifacts, clearly seen in the result of Fig. 5-8(d), shows that, in addition to learning how to resolve fine details in the artifact, PhENN has learnt, somewhat undesirably, to edge enhance (since all the examples it was trained with were also edge enhanced.) These observations should present fertile ground for further improvements upon the work presented here.

Chapter 6

Learning to synthesize: splitting and recombining low and high spatial frequencies for image recovery

6.1 Introduction

In the last chapter, one feature of the examples that we have found to strongly influence training is the examples' spatial frequency content. In the case of quantitative phase retrieval (QPR), the forward operator is ill-posed due to the frequency scrambling, together with the loss of high spatial frequencies caused by the finite numerical aperture. Therefore, the neural network would have to rely on the prior in the training examples to do the inversion. However, if we train the neural network using a database such as ImageNet, which is well-known to have an inverse-square law (spatial) power spectral density (PSD) [133], the DNN fails to learn the inverse square law. Because high frequencies are sparser in the database, and the training process is highly nonlinear, low frequencies may end up dominating the prior by more than their fair share; low-pass filtering of the inverse $\hat{f}$ and loss of fine detail ensues. We have proposed a spectral pre-modulation approach to compensate for the scarcity of high spatial frequencies in the database and showed that indeed fine details are recovered;

however, at the same time, high-frequency artifacts appeared in the reconstructions, evidently because the spectral pre-modulation also taught the DNN to edge enhance.

In this chapter, we propose a novel Learning Synthesis by DNN (**LS-DNN**) method to effectively manage and synthesize different spectral bands according to their relative behavior in the forward operator and prevalence in the training database. Our approach, similar to [135], processes the signal separately into two DNNs, assigned to low- and high-frequency bands, respectively. However, unlike [135] we *learn* how to synthesize the two bands adaptively; we have also modified the training procedure. More specifically, we train the two DNNs as follows: the low-band DNN (DNN-L) is trained so that its output $\hat{f}^{\mathrm{LF}}$ match the unfiltered examples; whereas the high-band DNN (DNN-H) is trained such that its output $\hat{f}^{\mathrm{HF}}$ match a filtered version of the examples where high spatial frequencies have been amplified. This choice for the DNN-L is for databases such as ImageNet, where high frequencies are naturally under-represented, as pointed out earlier. We then use a third DNN (DNN-S) to synthesize the reconstructions $\hat{f}^{\mathrm{LF}}$ and $\hat{f}^{\mathrm{HF}}$ from DNN-L and DNN-H, respectively, into a final estimate $\hat{f}$. This ensures that the low and high spatial frequencies are rebalanced correctly in the final reconstruction, according to the relative behavior of the two bands in the forward operator and the prior expressed by the examples. In this chapter, we demonstrate the effectiveness of the **LS-DNN** method in the scenario of QPR, but this concept may be generalized to a wider range of image recovery problems.

6.2 Methods

6.2.1 Learning Synthesis by Deep Neural Networks (LS-DNN)

We propose a two-step approach (Fig.6-1) to tackle the difficulty in restoring high frequency components. First, we use two separate deep neural networks (DNNs) parallel,DNN-L and DNN-H. DNN-L is what typically exists in previous works – it learns to map from the measurements g to the un-filtered ground truth images f;

while DNN-H takes in measurements g and maps them to a frequency-modulated ground truth $\tilde{f}$. The modulation in the frequency domain (see Section 5.4), amplifies the relative weights of the high spatial frequencies, thereby facilitating the extraction of high frequencies from the priors.

We expect the reconstruction by DNN-L to be reliable in the low frequency range but not in the high frequency range; whereas the one by DNN-H should be better in the high-frequency range but distorting the low frequency range and may be creating hallucination artifacts. The training of DNN-L and DNN-H can be done in parallel. We denote the reconstructions by DNN-L and DNN-H, during training, as $\hat{f}^{LF}$ and $\hat{f}^{HF}$, respectively.

Moreover, we train a third DNN, denoted as DNN-S, which learns to synthesize the low frequency and high frequency reconstructions from the previous networks $\hat{f}^{LF}$ and $\hat{f}^{HF}$, to generate the final reconstruction $\hat{f}$ matching the un-modulated ground truth.

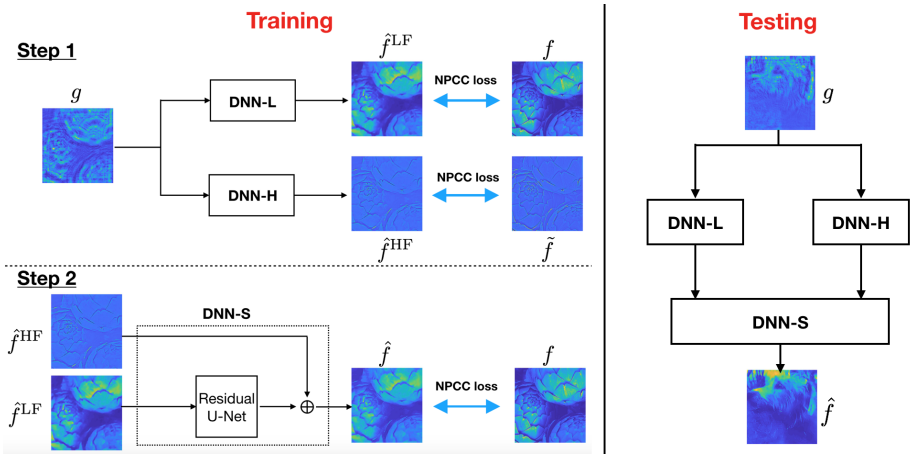

Figure 6-1: Proposed LS-DNN.

6.2.2 Architectures of Deep Neural Networks

DNN-L and DNN-H follow the PhENN architecture that we implement in Chapter 5. DNN-S takes in $\hat{f}^{LF}$ and $\hat{f}^{HF}$ as inputs. To best preserve the high frequency information contained in $\hat{f}^{HF}$, we do **not** have it pass through most layers of DNN-S. Instead, it only adds to the final outcome of $\hat{f}^{LF}$ passing through the Residual U-Net (has

the same architecture as PhENN) to form the final reconstruction $\hat{f}$. In other words, the DNN-S trains to map $\hat{f}^{LF}$ to a variant of itself that removes artifacts brought up by $\hat{f}^{HF}$ and (conceptually) synthesizes the low-frequency components contained in $\hat{f}^{LF}$ and the high-frequency components contained in $\hat{f}^{HF}$. A potential variation of DNN-S is to make it structurally the same as DNN-L (and DNN-H), except for the input to be instead the concatenation of $\hat{f}^{LF}$ and $\hat{f}^{HF}$(256x256x2) so that they both pass through all successive layers in the DNN. However, due to the low-frequency dominant nature of the DNN-S ground truth examples, exposure of $\hat{f}^{\mathrm{HF}}$ to trainable convolutional filters may cause loss of high frequency information contained in $\hat{f}^{\mathrm{HF}}$, as in DNN-L. This is why we opted for the first variant.

6.3 Results

6.3.1 Implementation details

The training and testing data used in this chapter is the same as those being used in Chapter 5. All the neural networks are trained with NPCC loss function. The training is conducted on a Nvidia GTX1080 GPU using TensorFlow. Adam optimizer is used in the optimization of each deep neural network. Each neural network is trained for 20 epochs and the batch size is 10.

6.3.2 Reconstruction Results - Spatial Domain

Fig.6-2 shows test results of our LS-DNN approach. DNN-L, DNN-H and DNN-S label the outputs of the corresponding DNNs. We can observe that, DNN-H succeeds in preserving high frequency components better than DNN-L, as expected, and the DNN-S removes the artifacts introduced by DNN-H through combining with the the low-frequency components captured well by DNN-L. Quantitative comparisons for the DNN-L and DNN-S outputs on the entire 100 test images are shown in Table.6.1. DNN-S also achieves better quantitative performance in the problem of QPR.

Dot patterns are also used to test the resolution achieved by LS-DNN, which is

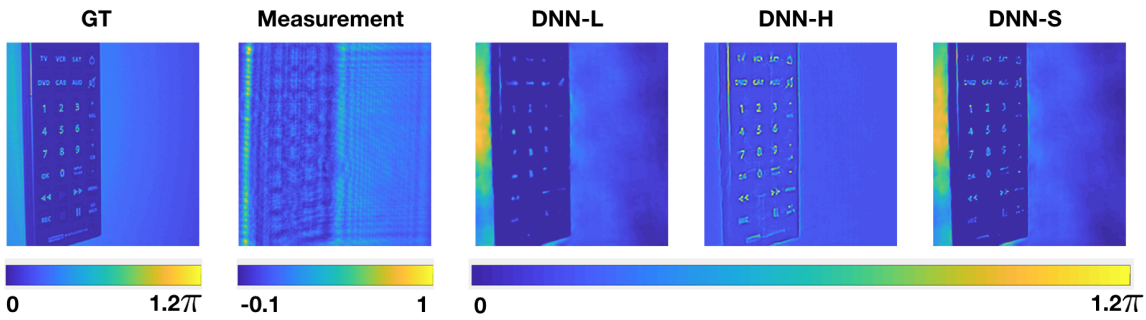

Figure 6-2: Reconstruction results for QPR.

Table 6.1: Quantitative evaluations of DNN-L and DNN-S performance in QPR.

	DNN-L PSNR/SSIM	DNN-S PSNR/SSIM
QPR	18.7531/0.3684	**18.8525/0.3884**

shown in Fig.6-3. For the test dot pattern with a spacing of $D = 4$ pixels, while the DNN-L reconstruction failed to distinguish the nearby dots, the DNN-S output successfully resolve the two dots (according to the Rayleigh criterion). This demonstrates the DNN-S's superiority over DNN-L in reconstructing fine details.

To demonstrate that performance improvement of DNN-S over DNN-L is a not a sole consequence of computational capacity increase, we design another ResNet, DNN-L-3, which has the same architecture but twice as many feature maps (except in the last residual block) on each convolutional layer as its counterpart in DNN-L. The resulting DNN-L-3 has more than three times as many trainable parameters as DNN-L(or DNN-H, DNN-S). Fig.6-4 shows the reconstructions by DNN-L-3 as compared to those by DNN-S. The superiority of DNN-S results proves our claim.

6.3.3 Reconstruction Results - Frequency Domain

Next we analyze LS-DNN performance in the frequency domain. Fig.6-5 show the 2D Fourier spectra of the images in Fig.6-2 [in logarithmic scale]. We find that, as expected, while DNN-L works well in the low frequency range but fails to recover the high frequency components, DNN-H is better at retrieving the high frequency components but loses some low frequency information. DNN-S serves as a learned

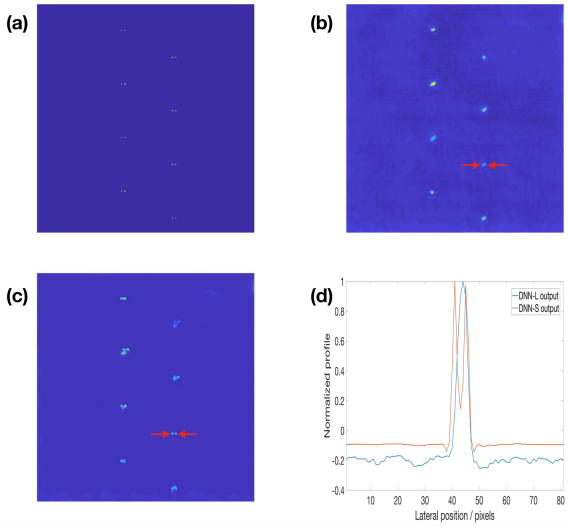

Figure 6-3: Resolution test results. (a) Dot pattern with spacing $D = 4$ pixels, (b) DNN-L reconstruction, (c) DNN-S reconstruction, (d) 1D cross-sections along the line indicated by red arrows in (b) and (c).

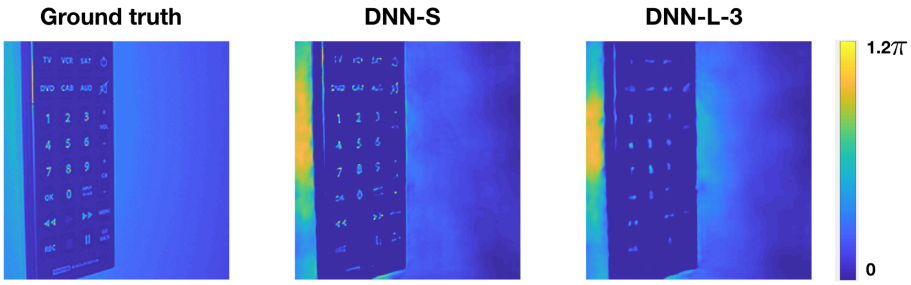

Figure 6-4: Comparison with DNN-L-3 for QPR.

synthesizer of the DNN-L and DNN-H outputs at all frequency bands, low and high.

To investigate the performance of our approach on the entire test dataset, we compute the 2D power spectral density (PSD) for the reconstructions of the entire ensemble of the 100 test images and show the 1D cross-sections (along the diagonal direction) in Fig.6-6. As expected, the PSD for the DNN-L output matches well with the ground truth in the low frequency range but is much lower than the ground truth in the high frequency range. Notably, in the case of DLI, the point that the performance of DNN-L starts to drop almost coincides with the cutoff frequency in the measurement. The PSD for DNN-H output is getting closer to the ground truth

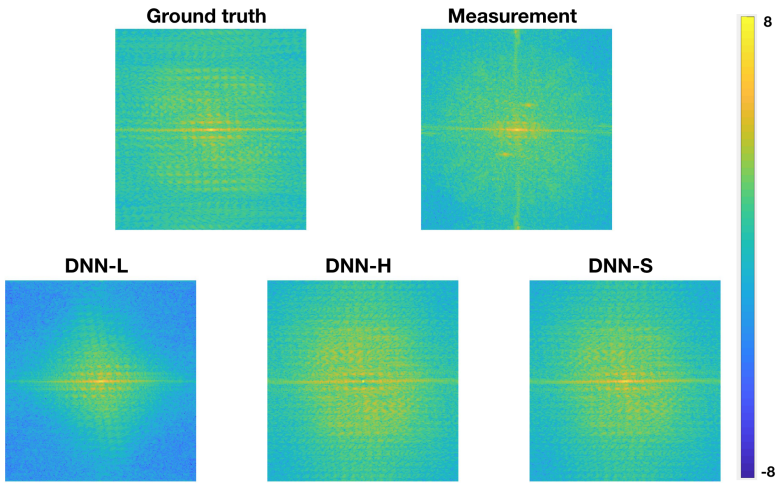

Figure 6-5: Fourier spectra of the reconstructions in QPR (logarithmic scale).

in the high frequency range than the DNN-L output, benefiting from training data pre-modulation. However, it is much worse in the low frequency range. The DNN-S output, which takes the advantage of both DNN-L and DNN-H, has a PSD that is close to the ground truth within the entire frequency range.

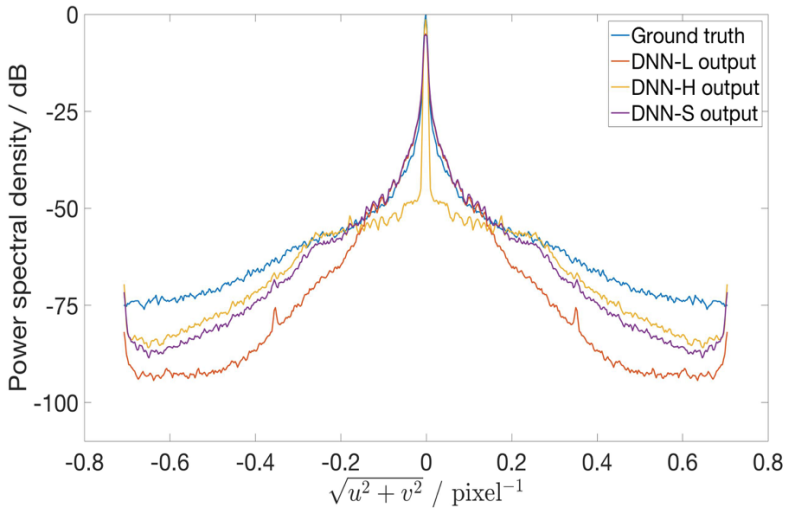

Figure 6-6: 1D cross-sections of the reconstructions' power spectral density (PSD) on 100 ImageNet test images.

6.4 Conclusions

We have proposed a novel LS-DNN architecture that leverages the learning capability of a neural network to optimally synthesize parallel reconstructions that are reliable on low and high frequencies, respectively.

The LS-DNN methodology is applicable to many other ill-posed image recovery problems, when there is imbalance between the spatial frequency region where the forward operator suppresses or loses content and the availability of content in the same frequency range in the training database to compensate. Generalizations may also be possible, for example to more than two frequency bands, similarly trained separately with pre-filtering and recombined in learned fashion; however, splitting the frequency bands too finely may be counterproductive if it results in too many parameters to learn. This is an interesting topic for future work.

6.5 My contributions

The work in this chapter was led by Mo Deng. I contributed in developing the concept, performing the experiment, analyzing the data and writing the manuscript.

Chapter 7

Quantitative phase microscopy by LS-DNN enhanced PhENN

7.1 Introduction

Up to now, we have demonstrated the capability of deep learning techniques in solving computational imaging problems and also showed how to enhance reconstructed image quality by selecting the appropriate network structure as well as modulating the training examples. Nevertheless, in all the experiments, the training data and the testing data were emulated using the same SLM. Then, it becomes natural to ask: *how are the performances of those SLM-trained neural networks when tested by actual objects such as biological cells.*

In this chapter, we set out to test the performances of a SLM-trained PhENN in recovering actual phase objects. To perform these tests, the first decision that we have to make is the selection of an appropriate training database. Existing works that applying deep learning to reconstruct actual phase objects all use a specific training database that contains the same type of images as the phase objects that they want to recover [137, 138, 139], e.g. ref.[137] trained and tested their neural network on Pap smear samples; in ref. [138], the training and tested images were all taken from Hela cell samples. This training strategy generally leads to high reconstruction qualities, but also has two main drawbacks at the same time. Firstly, this kind of training

dataset is not easy to obtain in practice. For an arbitrarily given phase sample, we usually do not have access to a large number of training images of the the same type. Secondly, this kind of training dataset usually contains a strong prior, which limits the generalizability of the trained neural network. In other words, the neural network trained using this strategy only performs well on a specific kind of samples. If a new type of sample is present, the neural network need to be re-trained. Therefore, training the neural network on a more general database such as ImageNet may be a better option. ImageNet is a large open database, which can be readily obtained. More importantly, as we have shown in Chapter 3 and 5, the PhENN trained on ImageNet has superior generalizability: it performs well on objects created from other databases (e.g. Faces-LFW, MNIST, etc). Because of that, we should expect an ImageNet-trained PhENN to perform well on different types of actual phase objects. Hence, we choose ImageNet as the training database in this chapter.

Actual phase objects usually contains fine features, e.g. the diameter of a typical red blood cell (RBC) is $\sim 8\mu$m. Nevertheless, when a neural network is trained with examples emulated by a SLM, its spatial resolving ability is limited by the pixel size of the SLM. Although the spatial resolution can be enhanced by using the spectral pre-modulation technique (Chapter 5) and the subsequent LS-DNN approach (Chapter 6), the optimal resolution that a trained neural network can achieve still can not be smaller than 1 pixel (36μm for the SLM that we use here), which is not sufficient to resolve the fine features of many samples. In order to solve this problem, we insert a wide-field microscope module into the imaging system. In this way, as we will show in detail in Section 7.2, the equivalent pixel size at the sample plane can be down to sub-micros, making those fine features resolvable.

The structure of this chapter is as follows: the details about the imaging system will be described in Section 7.2; the reconstruction result of a RBC sample (Carolina, unstained), as well as the demonstrations that PhENN has indeed learned the physical model, will be shown in Section 7.3; and concluding thoughts will be provided in Section 7.4.

7.2 Imaging system

The optical configuration of the imaging system is shown in Fig. 7-1. The setup is quite similar to the one that we use in Chapter 5 [Fig.5-1]. The difference is that we insert a wide-field microscope module (including a sample stage, an objective lens and a tube lens) between the collimating lens (CL) and the polarizer (P). The system is aligned in a way such that the SLM is placed at the focal plane of the tube lens. Then, we may approximate the optical field immediately before the SLM to be a magnified version of the field at the sample plane. In this system, another objective lens with long working distance and small exit pupil (Mitutoyo, Plan Apo NUV, 50X/0.42) is used as CL, so as to increase the illumination intensity. The same as Chapter 5, the polarizer and the analyzer are set to the orientations under which the SLM is operated in the phase-mostly modulation mode. The focal lengths for the two plano-convex lenses in the telescope are $f_1 = 300$mm and $f_2 = 100$mm, which result in a de-magnification ratio of 3, matching the ratio between the pixel sizes of the SLM and the CMOS.

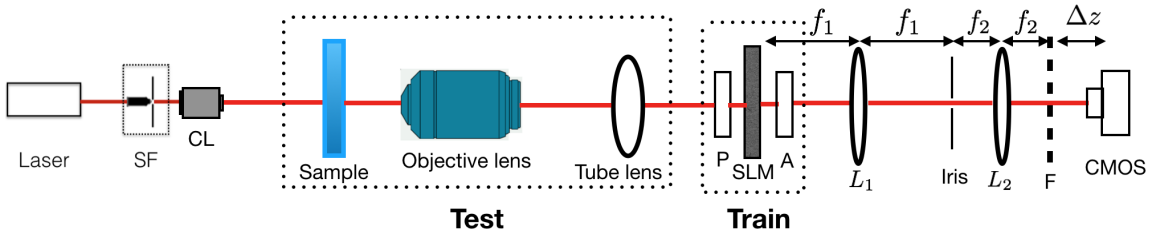

Figure 7-1: Optical configuration. SF: spatial filter; CL: collimating lens; P: polarizer; A: analyzer; L_1,L_2: plano-convex lenses; F: focal plane of L_2.

The system is operated in three modes: training, validation and testing. In the training mode, nothing is placed on the sample stage. 10000 images from the ImageNet database are sequentially displayed on the SLM and their respective diffraction patterns are captured by the CMOS. When the neural network is trained using this set of training data, it learns the mapping between the 2-D phase distribution at the SLM plane and the 2-D intensity pattern at the CMOS plane. In the validation mode, nothing is placed on the sample stage. 450 images from the ImageNet (disjoint

with the training data), Faces-LFW, Characters, MNIST, faces-ATT and CIFAR databases are sequentially displayed on the SLM and their respective diffraction patterns are captured by the CMOS. This dataset is used to validate the performance of the trained neural network. In the testing mode, we place the phase object that we want to image on the sample stage. In the meantime, an uniform image where each pixel has the same value is displayed on the SLM. In this case, the SLM itself does not introduce any additional phase contrast. Then, the field immediately after the SLM is just a magnified version of the the phase object and it can be readily obtained by feeding the intensity pattern captured by the CMOS into the trained neural network.

7.3 Results

In this chapter, we implement the LS-DNN enhanced PhENN (see Chapter 6) so as to obtain a good reconstruction quality. The neural network architecture is described in Section 6.2.2. The same as Chapter 5, while collecting the training and validation data, each image is resized to 256×256 and then zero-padded to 1024×768 before being displayed on the SLM. Also, we only crop the central 256×256 region of the diffraction patterns captured by the CMOS for processing. The output of the neural network is calibrated following the procedure described in Section 4.3.1 to compensate for the unknown affine transform effected by the NPCC training loss function.

7.3.1 Red blood cell imaging

The actual phase object that we image is a red blood cell (RBC) sample (Carolina, unstained). Here, we use a high magnification objective lens (Olympus, UPlanFL N, 100X/1.30) so as to increase the spatial resolving ability of the imaging system. In addition, a preprocessing step [140] based on the Gerchberg-Saxton (GS) algorithm is applied to the raw measurement before feeding it into the neural network.

The validation results of our LS-DNN enhanced PhENN in this case are shown in Fig.7-2. As expected, our neural network generalizes well: it is able to reconstruct the phase profiles of images from different classes, despite the fact that it was trained

exclusively on images from the ImageNet database.

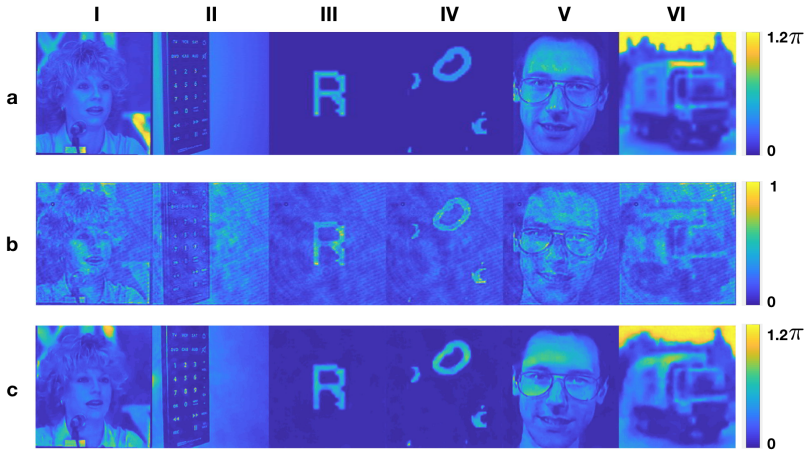

Figure 7-2: Validation results by LS-DNN enhanced PhENN. (a) Ground truth for the phase objects. (b) Diffraction patterns captured by the CMOS (after normalization). (c) Reconstructions. Columns (i-vi) correspond to the dataset from which the validation image is drawn: (i) Faces-LFW [99], (ii) ImageNet [100], (iii) Characters, (iv) MNIST Digits [101], (v) Faces-ATT [104, 105], or (vi) CIFAR [103], respectively.

The reconstruction result of the RBC sample is shown in Fig.7-3. The LS-DNN enhanced PhENN is able to qualitatively reconstruct the phase profiles of the RBC sample. To evaluate the reconstruction results quantitatively, we convert the phase value to the thickness of the sample, given the refractive index of the RBC (~ 1.39) [141] and the PBS buffer (~ 1.335). We find that for some of the RBCs, our recovered thickness matches well with the thickness measurement reported in [142] ($1.7-2.2\mu$m), as shown in the 1D cross-section plot (Fig. 7-4). However, at some other regions, quantitative error still exists. Possible sources for quantitative error will be discussed in Section 7.3.1.

7.3.2 Demonstration that PhENN indeed learned the physics

Although the reconstruction by the neural network contains quantitative error, our LS-DNN enhanced PhENN actually is not doing something trivial such as pattern matching, as some readers may argue by looking at the raw measurements by the CMOS (e.g. Row b in Fig.7-2), where some salient features of the objects are already visible.

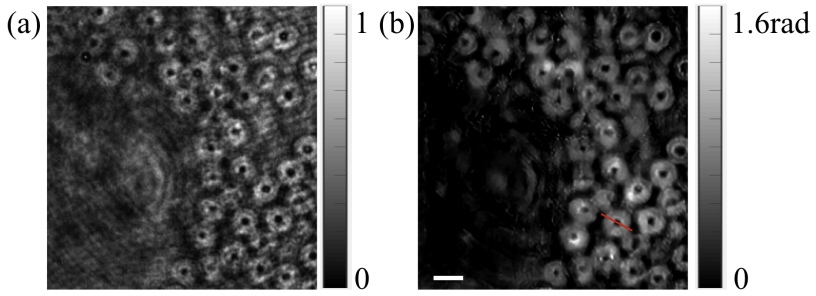

Figure 7-3: Red blood cell (RBC) imaging results. Scale bar: 10μm. (a) Intensity measurement of the diffraction pattern. (b)Phase reconstruction by LS-DNN enhanced PhENN.

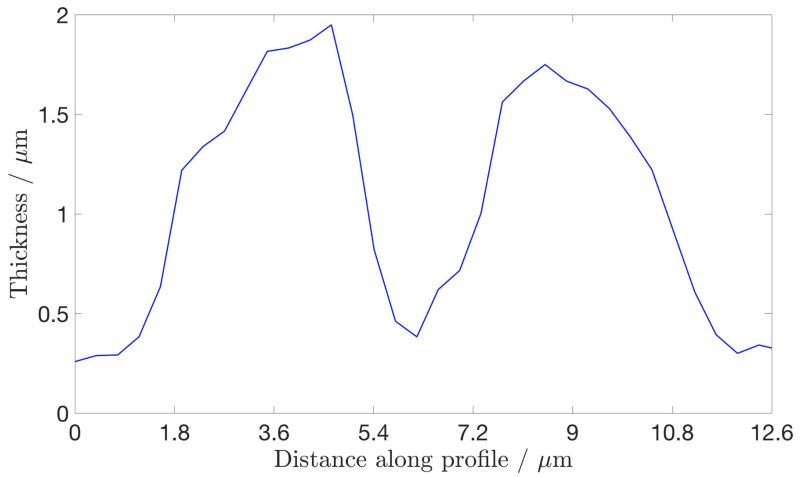

Figure 7-4: 1D cross-section of the reconstructed RBC sample. This profile is along the line indicated by the red line in Fig. 7-3(b).

The forward imaging model of lensless QPR, as shown in Eq. 3.1, is non-linear. However, when the weak object approximation holds ($\exp\{if(x,y)\} \approx 1 + if(x,y)$) [143], the forward imaging model may be linearized as,

$$G(u,v) \approx \delta(u,v) + 2\sin\left(\pi\lambda z(u^2+v^2)\right) F(u,v). \tag{7.1}$$

Here, $G(u,v)$ and $F(u,v)$ are the Fourier transform of the intensity measurement $g(x,y)$ and the phase distribution of the object $f(x,y)$, respectively. $\sin\left(\pi\lambda z(u^2+v^2)\right)$ is the weak object transfer function (WOTF) for lensless QPR, which is plotted in Fig.7-5.

Has our LS-DNN enhanced PhENN really learned the correct WOTF?, we first in-

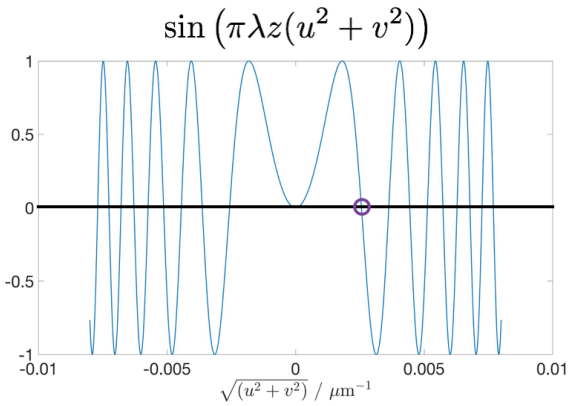

Figure 7-5: Weak object transfer function (WOTF) for lensless QPR (i.e. Fresnel propagation is the free space). An example of nulls is indicated by the purple circle. For this plot, the propagation distance $z = 240$mm, the wavelength $\lambda = 633$nm.

vestigate this point by simulation. The training and testing database contains 10000 and 100 images from ImageNet, respectively. To satisfy the weak object approximation, the maximum phase depth is set to be 0.1π. In generating the corresponding intensity measurements, we set the propagation distance to be $z = 20$mm and the pixel size of the object to be $\Delta x = 12\mu$m. Once the network is trained, the learned WOTF is computed as,

$$\hat{\text{WOTF}} = \frac{1}{100}\sum_{k=1}^{100}\frac{G_k(u,v) - \delta(u,v)}{\hat{F}_k(u,v)}. \tag{7.2}$$

Here, $G_k(u,v)$ and $\hat{F}_k(u,v)$ are the Fourier transform of the intensity measurement $g_k(x,y)$ and the network's estimated phase $\hat{f}_k(x,y)$ for the kth testing object, respectively.

The 1D cross-section along the diagonal direction of the learned WOTF is shown in Fig. 7-6. We find that the learned WOTF matches well with the actual WOTF in low and middle frequency range ($\sqrt{u^2+v^2} < 0.4\text{pixel}^{-1}$). The mismatch becomes larger at higher spatial frequencies, which is due to the inverse-square power law in the power spectral density (PSD) of ImageNet images, as we have discussed in Chapter 5. This result indicates that our network has indeed learned the physics.

As you may have already observed, there exists several nulls (the locations where

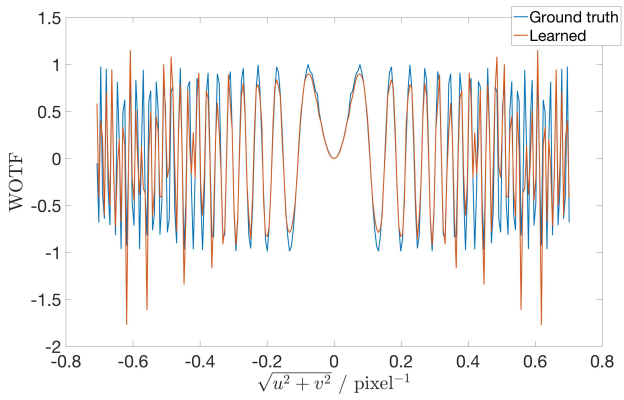

Figure 7-6: Learned WOTF by LS-DNN enhanced PhENN. This is the 1D cross-section along the diagonal direction.

the values are equal to zero) in the transfer function shown in Fig.7-5. At those nulls, the sign of the transfer function switches, thus introducing a phase delay of π in the spatial frequency domain. As a result, the measured pattern at the detector plane will shift by half the period in the spatial domain at those frequencies. We refer this phenomenon as the 'phase shift effect'. Because of that, when we image a star-like phase object, the fringes in the measurement will become discontinuous. As we highlight by the circles in Fig. 7-7, the fringes shift by about half the period at some particular radii.

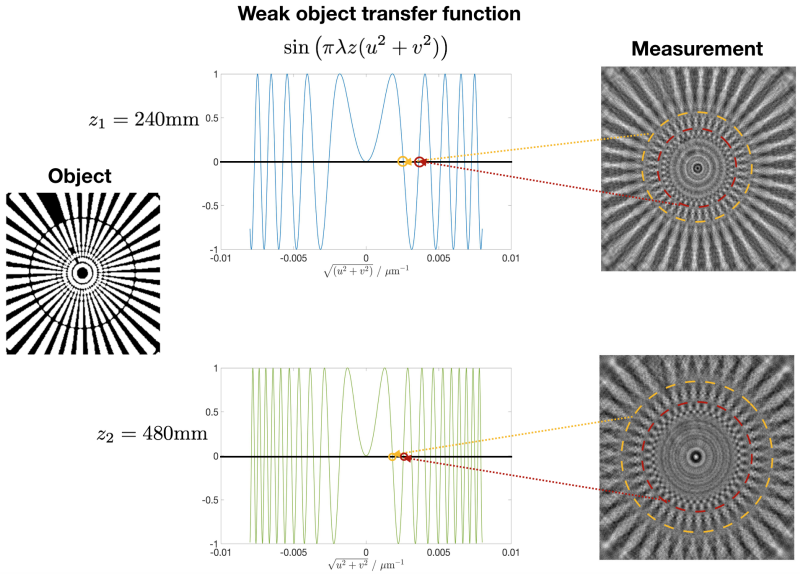

Figure 7-7: Simulated phase shift effect on a star-like phase target.

Is our LS-DNN enhanced PhENN able to undo this phase shift effect? We perform experiments on a star-like phase target (Benchmark Technologies)[136] using the system shown in Fig. 7-1. In this test, an objective lens with 20X magnification (Nikon, Plan, 20X/0.4) is used so as to achieve a field of view that is large enough to capture the entire sample. As expected, the fringes in the measurement become discontinuous, as we highlight by the yellow and red circles in Fig.7-8(a). We note that although the phase depth of the star target (0.55π) does not quite satisfy the weak object approximation, the 'phase shift effect' still exist. In the meantime, for the reconstruction result shown in Fig.7-8(b), all the fringes in the phase target become continuous again. This result indicates that our PhENN is not merely doing pattern matching; it has indeed learned the Fresnel propagation operator and is attempting to undo the 'phase shift effect' in the correct way.

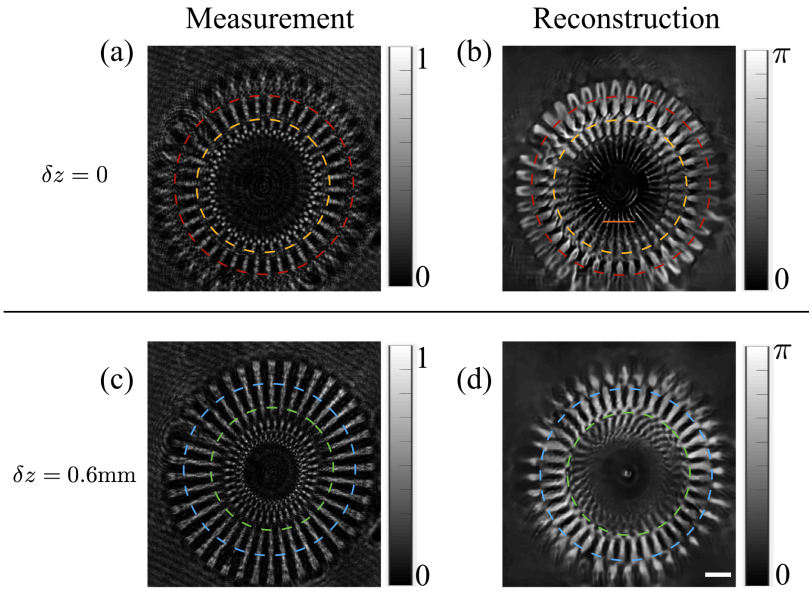

Figure 7-8: Fringe continuity analysis. Scale bar: 50μm. (a) Intensity measurement of the diffraction pattern when $\delta z = 0$. (b) Phase reconstruction by LS-DNN enhanced PhENN when $\delta z = 0$. (c) Intensity measurement of the diffraction pattern when $\delta z =$ 0.6mm. (d) Phase reconstruction by LS-DNN enhanced PhENN when $\delta z = 0.6$mm.

To further demonstrate this point, we place the star-like phase target at an out-of-focused sample plane, where the distance between the sample plane and the focal plane of the objective lens is $\delta z = 0.6$mm. In this case, the optical field at the SLM plane,

which is aligned at the focal plane of the tube lens, will become a de-focused version of the star pattern, i.e. shifts in the fringes should occur due to Fresnel propagation in the free space. As expected, our neural network, which is trained to recover the phase field at the SLM plane from the CMOS measurement, outputs a reconstruction where the fringes indeed become discontinuous at some spatial frequencies (as indicated by the green and blue circles in Fig. 7-8(d)), although the raw measurement shown in 7-8(c) does not contain these fringe discontinuities. Furthermore, the spatial frequencies (radii) where the fringe shifts happen in the network output match with the theoretical predictions. Therefore, our PHENN has indeed learned the actual light propagation model, rather than doing something trivial like pattern matching.

The reconstruction results of the star-like phase target is also evaluated quantitatively. This phase target is made of an acrylate polymer ($n = 1.52$) on Corning Eagle XG Glass. The nominal height of the target is 335.4nm, which results in a phase delay of $\phi = 0.55\pi$ for the light with wavelength $\lambda = 633$nm. The reconstructed phase values are close to 0.55π in some regions, as shown in the 1D cross-section plot (Fig. 7-9), but are very different from the ground truth in some other regions.

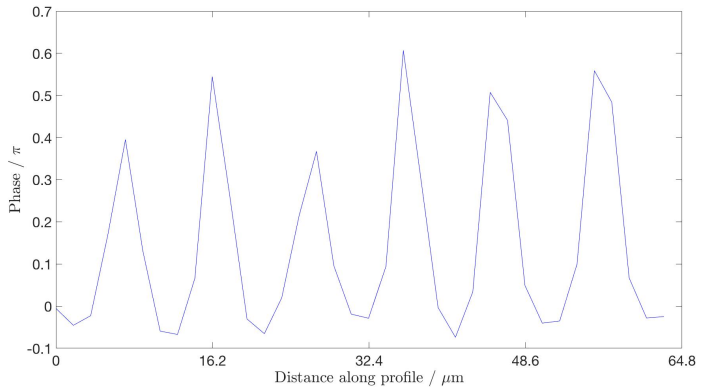

Figure 7-9: 1D cross-section of the reconstructed phase target. This profile is along the line indicated by the orange line in Fig. 7-8(b).

The quantitative error of our PhENN can be explained in the following aspects. First, due to the wide-field microscopy module, the signal to noise ratio (SNR) of the captured diffraction pattern is small, making it difficult for the neural network to do the reconstruction. Second, what the neural network tends to reconstruct is actually

the phase image at the SLM plane. However, due to the limited numerical aperture of the microscopy module, the phase image at FP is not simply a magnified version of the phase target, but a low-pass filtered version. As a result, a binary phase target at the sample plane will no longer generate a binary image at SLM plane. In addition, the aberrations caused by the microscopy module and the inevitable misalignment of the system will also make the SLM plane phase image distorted. Third, the training data are generated by the transmissive SLM, which is not a perfect phase object. It has a coupled intensity modulation with maximum ratio ~ 2.7(see Section A.3) and a small fill factor of 58%. These will also influence the performance of the neural network.

7.4 Conclusions

In this chapter, we build up a phase microscopy system to test the performance of our LS-DNN enhanced PhENN in imaging actual phase objects. We demonstrate that when the neural network is trained on a general database such as the ImageNet, it generalizes well and is able to qualitatively reconstruct two different actual phase objects: one star-like phase target and a RBC sample. In addition, through the learned WOTF simulation and the star-like phase target experiment, we also demonstrate that our trained neural network has indeed learned the physics rather than doing something trivial like pattern matching.

Nevertheless, relatively large quantitative errors still exist in the reconstructions by PhENN due to several factors including the low SNR, the aberrations and misalignment in the microscopy module, as well as the residual intensity modulation of the SLM. Further improvement is required to reduce all these influences to achieve a better phase retrieval quality.

Chapter 8

Conclusions and future works

This thesis has studied the application of deep learning (DL) techniques in solving the inverse problems in computational imaging (CI). Compared with traditional inverse algorithms in CI, DL approaches do not need explicit knowledge about the forward model and the prior of the class of the objects being imaged. As a result, accurate characterization of the imaging system is no longer required and those objects whose priors are difficult to be analytically formulated (e.g. flowers, cars, etc) can be imaged. In addition, the use of DL also increases the inference speed of CI, which is very beneficial in real-time applications.

The main contributions of this thesis are summarized as follows:

- Proposed and experimentally demonstrated the first convolutional neural network (CNN) architecture, IDiffNet, for imaging through scattering media.

- Proposed and experimentally demonstrated the first end-to-end DL architecture, PhENN, for quantitative phase retrieval. (Together with leading author Ayan Sinha)

- Proposed a novel training loss function, NPCC, as well as the related calibration procedure. We demonstrated that NPCC is better at recovering spatially sparse objects and fine features.

- Investigated the influences of the network architecture (e.g. connectivity, net-

work depth, waist size) and the training example quality (spatial frequency content in particular) to the performance of trained network.

- Proposed and experimentally demonstrated a spectral pre-modulation approach to enhance the spatial resolving ability of PhENN.

- Proposed and experimentally demonstrated a Learning Synthesis by DNN (LS-DNN) method, which is a general DL approach that effectively manage and synthesize different spectral bands so as to improve the visual qualities of recovered images in CI problems. (Together with leading author Mo Deng)

- Applied the LS-DNN enhanced PhENN to a phase imaging microscope and experimentally demonstrated that the network was able to recover actual phase objects when trained on ImageNet images emulated by a SLM.

- Based on the simulation of the learned WOTF and the experiment of the phase shift effect in the star-like target measurement, we demonstrated that our network has indeed learned the physical model.

Beyond this work, there are many more problems in the application of DL techniques to CI that are worth investigating in the future:

- In this thesis, each neural network is treated as a "black box" and is trained in the end-to-end fashion, i.e. the raw measurement from the detector is feed directly into the neural network. In other words, no physics is included in the training process. The neural network has to learn everything from scratch. However, in some imaging scenarios, the knowledge about the forward model is available (e.g. quantitative phase retrieval). In these cases, if we are able to explicitly include the known physical model into the network training process, it will serve as a good initialization. As a result, we shall expect a boost in the neural network performance.

- In this thesis, DL technique is only used as an alternative method to solve the inverse problem in CI. However, DL can also contribute in the physical

measurement part. It will be interesting to study if we can use DL to co-design the hardware and software for a CI system, i.e. optimize the optical setup (e.g. illumination pattern, coded aperture pattern, etc.) and the inverse algorithm (regression network) together in the end-to-end fashion.

- One major drawback of DL techniques is the requirement of big data. For some tasks, however, large training dataset is not readily available. Therefore, it will be beneficial to conduct research on data augmentation approaches. Moreover, to the best of our knowledge, all the neural networks in CI nowadays are trained in the supervised fashion, which requires paired data. If semi-supervised or unsupervised learning can be applied to CI, the data collection process will become much easier.

Appendix A

Calibration of the SLMs and the related analysis

A.1 Calibration of the reflective SLM in the intensity modulation mode

A.1.1 Calibration

In order to calibrate the modulation performance of the SLM, we built a Michelson interferometer shown in Fig. A-1. Compared with our experimental setup shown in Fig. 2-1, we remove the glass diffuser in this calibration setup and place a reflecting mirror M2 to the left of the beam splitter, creating the reference beam.

While calibrating the intensity modulation, we use a photon diode sensor (Newport, 818-SL) as the detector. The mirror M2 is blocked so that the light being detected comes from the SLM only. All the pixels of the SLM is driven by the same value V (uniform) and the corresponding intensity values measured by the photon diode is recorded. By changing V from 0 to 255 and repeating the measurement, we obtain the intensity modulation curve as shown in Fig. A-2, which is normalized to the intensity values at $V = 0$. The intensity modulation of the SLM follows a monotonic relationship with respect to the assigned pixel value and a maximum intensity

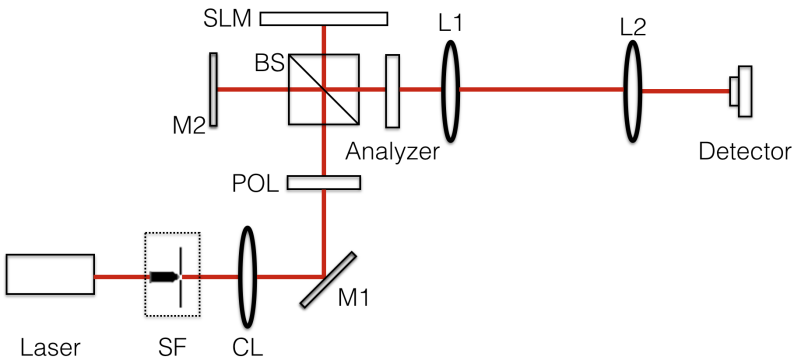

Figure A-1: The optical setup for calibrating the phase and intensity modulation of SLM. SF: spatial filter; CL: collimating lens; M1, M2: mirror; L1,L2: lens; POL: linear polarizer; BS: beam splitter; SLM: spatial light modulator.

modulation ratio of ~ 17 can be achieved.

While calibrating the phase modulation, we use the CMOS camera as the detector and the interferometer setup is used. All the pixels of the SLM is driven by the same value V (uniform) and the corresponding interference patterns are captured by the CMOS camera. Let $\phi(V)$ denote the relative phase between the light reflected by the uniform SLM when its pixels are all driven with gray scale value V, relative to M2; and I_0 denote the intensity reflected by M2. In the uniform illumination case, given $M(V)$, which is the calibrated intensity modulation, the intensity recorded by the CMOS should be of the form

$$\frac{I(V)}{I_0} = 1 + M(V) + 2\sqrt{M(V)}\cos\phi(V) \tag{A.1}$$

We arbitrarily assigned $\phi(0) = 0$ radians. The phase modulation curve is shown in Fig. A-3. The phase depth is $\sim 0.6\pi$.

A.1.2 Analysis of the influence of phase modulation

In Chapter 2, we want the SLM to performance as a pure intensity object. However, as we showed in the last section, due to the optical anisotropy of the liquid crystal molecules, the SLM will always perform a correlated phase modulation and the phase depth is $\sim 0.6\pi$ for our experimental arrangement. In order to analyze the influence

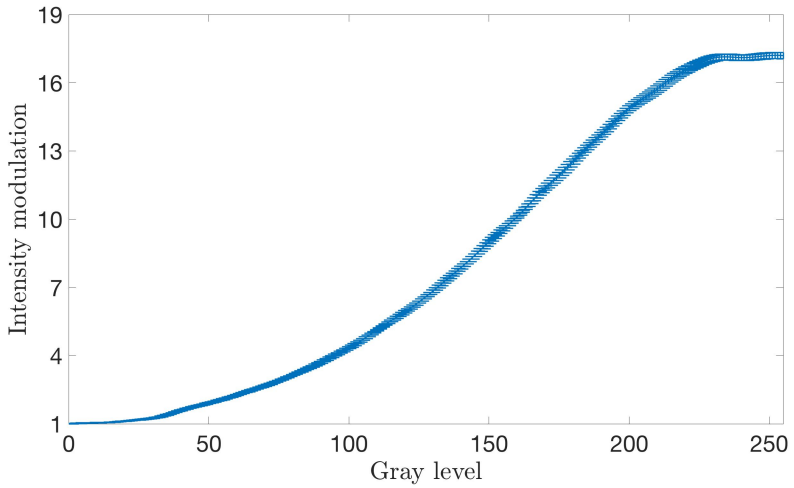

Figure A-2: Experimentally calibrated intensity modulation curve with error bounds in the grayscale range of [0,255] for the SLM.

of this phase modulation in the formation of the speckle patterns, we carry out the following simulations.

For a randomly given image uploaded to the SLM, we simulate the corresponding speckle patterns on the CMOS camera using Eqs.2.1 and 2.2 in the main manuscript for two cases:

(a): Assuming the SLM to perform both intensity and phase modulation, which is the actual modulation in practice, i.e. $g(x,y) = \sqrt{M\left[V(x,y)\right]}\exp\{iP\left[V(x,y)\right]\}$. Here, $V(x,y)$ is the pixel size of the uploaded image, $M(V)$ and $P(V)$ are the intensity modulation and phase modulation curves as shown in Fig. A-2 and Fig. A-3, respectively.

(b): Assuming the SLM to perform intensity modulation only, i.e. $g(x,y) = \sqrt{M\left[V(x,y)\right]}$.

In our simulation, we set those parameters as: $\mu = 16\mu$m, $\sigma_0 = 5\mu$m, $\sigma = 4\mu$m for the 600-grit diffuser and $\mu = 63\mu$m, $\sigma_0 = 14\mu$m, $\sigma = 15.75\mu$m for the 220-grit diffuser. Other simulation parameters are set to be the same as the actual experiment: $z_d = 15$mm, $R = 12.7$mm and $\lambda = 632.8$nm.

For the speckle patterns obtained in case (a) and (b), we also compute their respective autocorrelation functions and take the element-wise ratios between them.

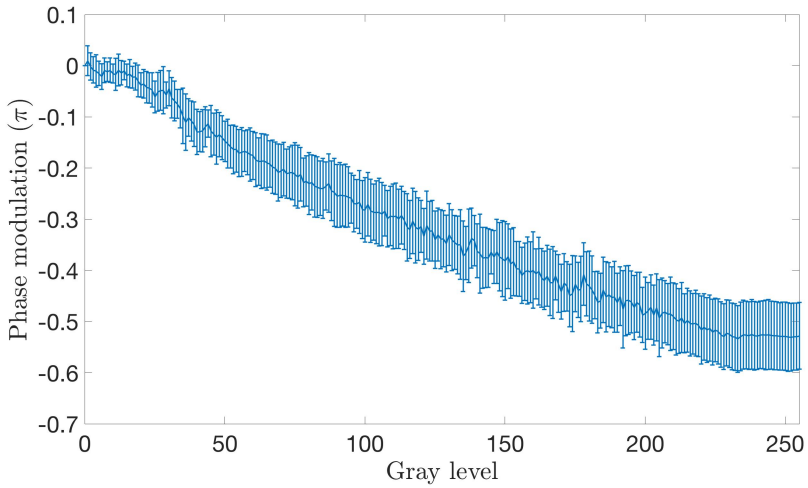

Figure A-3: Experimentally calibrated phase modulation curve with error bounds in the grayscale range of [0,255] for the SLM.

This process is repeated for both diffusers and the corresponding results are shown in Fig. A-4 and A-5.

We find that for both diffusers, the speckle patterns obtained in case (a) and (b) are qualitatively similar. Most importantly, the autocorrelation ratio values are all ~ 1 with a small standard deviation, which indicates that the correlated phase modulation will not change the statistics of the resulted speckle patterns. To further demonstrate this point, we repeat this simulation process for 10 times for both diffusers and plot the probability histograms for the resulting autocorrelation ratios. The results are shown in Fig. A-6. As expected, the autocorrelation ratio values are all ~ 1. We also compute the mean values μ_r and standard deviations σ_r for the two histograms: $\mu_r = 1.0009, \sigma_r = 0.0020$ for the 600-grit diffuser and $\mu_r = 1.0005, \sigma_r = 0.0013$ for the 220-grit diffuser. Therefore, we conclude that the influence of the correlated phase modulation can be neglected and we can reasonably approximate the SLM as a pure intensity object.

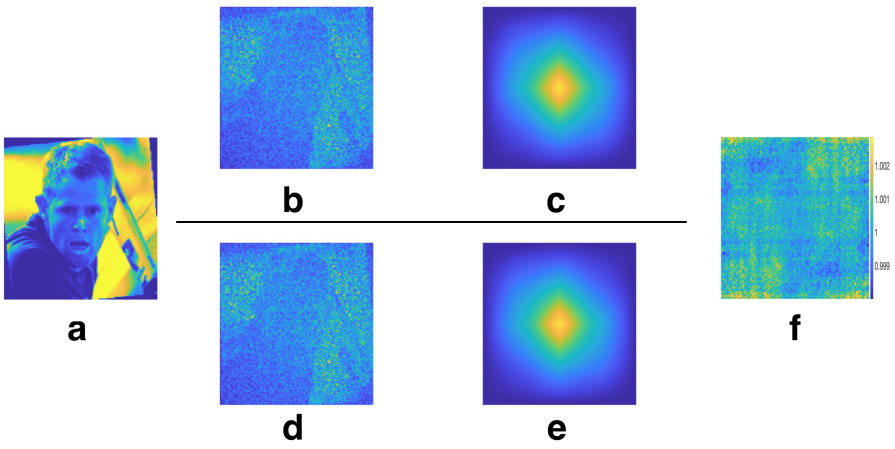

Figure A-4: Analysis of the influence of phase modulation in the formation of speckle patterns for 600-grit diffuser. (a) Input image; (b) Simulated speckle pattern for the complex object; (c) Autocorrelation of the speckle in (b); (d) Simulated speckle pattern for the pure-intensity object; (e) Autocorrelation of the speckle in (d); (f) Element-wise ratios between the autocorrelations in (c) and (e).

A.2 Calibration of the reflective SLM in the phase modulation mode

We use the same system as shown in Fig. A-1 to calibrate the reflective SLM in the phase modulation mode. While calibrating the intensity modulation curve, we block the mirror M2 so that the beam on the CMOS is coming from the SLM only and set the left-hand half pixels of the SLM driven by a constant value zero and the right-hand half driven by a different value (half-half), to determine $M(V)$. After that, we obtain the phase modulation curve following the same procedure as described in section A.1 [Drive the SLM pixels uniformly from 0 to 255 and use Eq. **??**].

The two curves are shown, respectively, in Figs A-7 and A-8. The maximum intensity modulation ratio measured was $\sim$ 1.3 for our configuration of polarizer and analyzer. In contrast, per manual specification, the SLM's maximum intensity modulation ratio is $\sim$ 12000 for other polarizer/analyzer configurations. Since our intensity modulation is small, we neglected it while designing our neural network architecture.

Figure A-9 shows the variation of phase modulation with 8-bit gray scale values

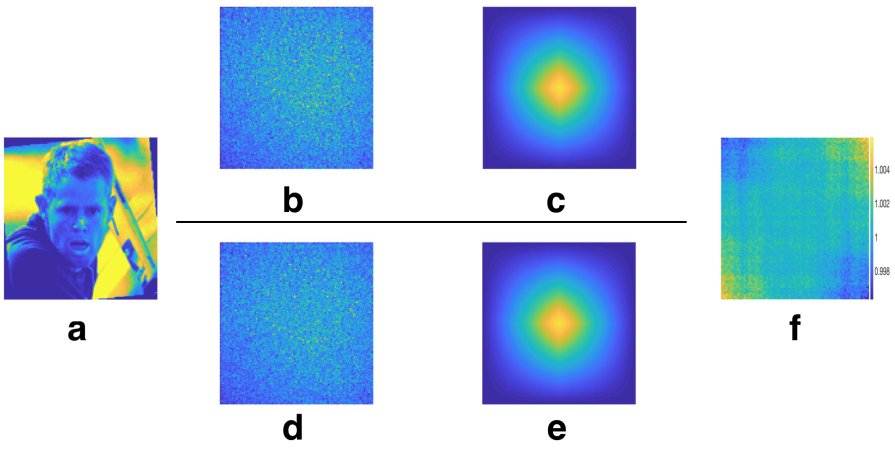

Figure A-5: Analysis of the influence of phase modulation in the formation of speckle patterns for 220-grit diffuser. (a) Input image [99]; (b) Simulated speckle pattern for the complex object; (c) Autocorrelation of the speckle in (b); (d) Simulated speckle pattern for the pure-intensity object; (e) Autocorrelation of the speckle in (d); (f) Element-wise ratios between the autocorrelations in (c) and (e).

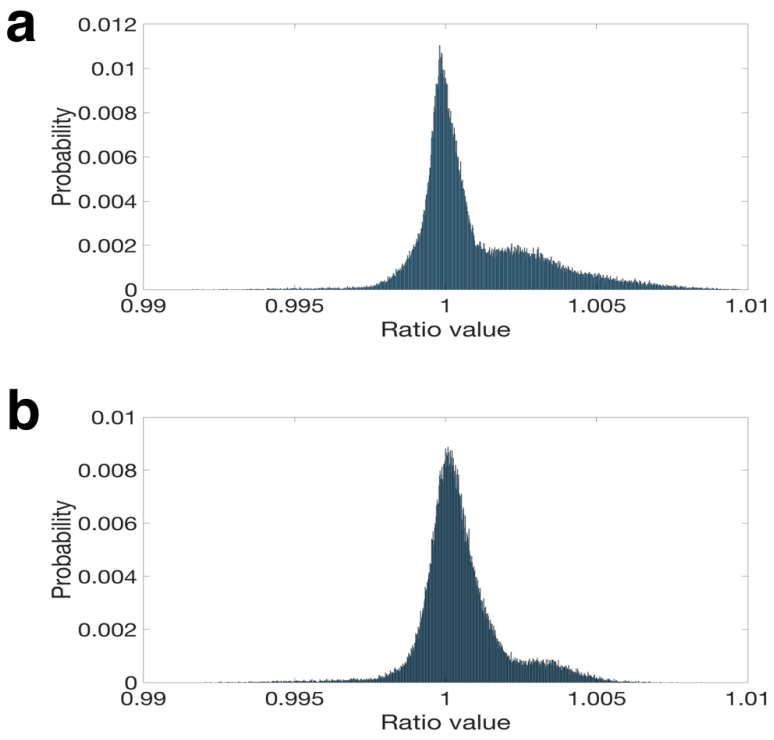

Figure A-6: Quantitative analysis of the influence of phase modulation in the formation of speckle patterns. (a) 600-grit diffuser; (b) 220-grit diffuser.

and three piecewise linear segments fitted to the phase modulation curve to obtain phase values varying linearly with gray scale values. The mean square error (MSE) of

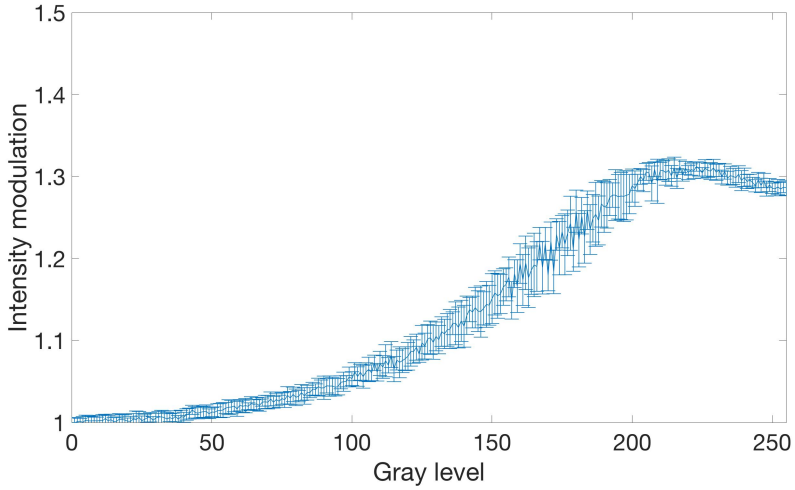

Figure A-7: Experimentally calibrated intensity modulation curve with error bounds in the grayscale range of [0,255] for the SLM.

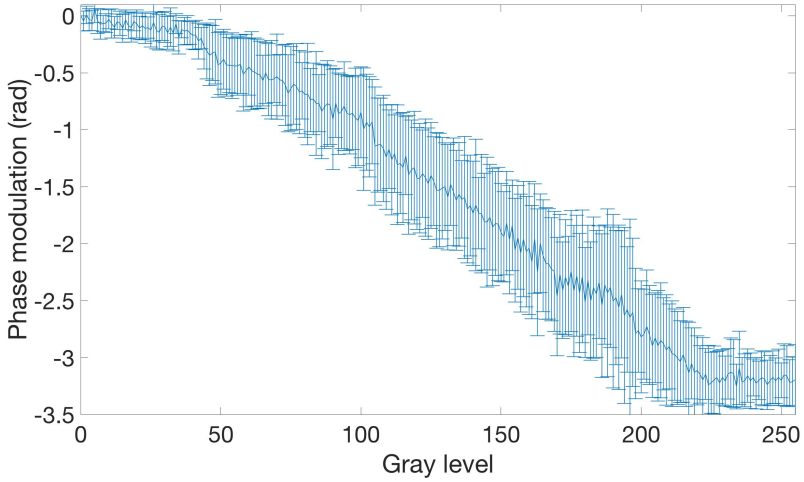

Figure A-8: Experimentally calibrated phase modulation curve with error bounds in the grayscale range of [0,255] for the SLM.

the fit is 0.18. In Figure A-10 we fit a single linear segment to the phase modulation curve with MSE of 0.19. All experiments in the main manuscript were done by training neural network with phase values obtained by fitting 3 segments to the curve.

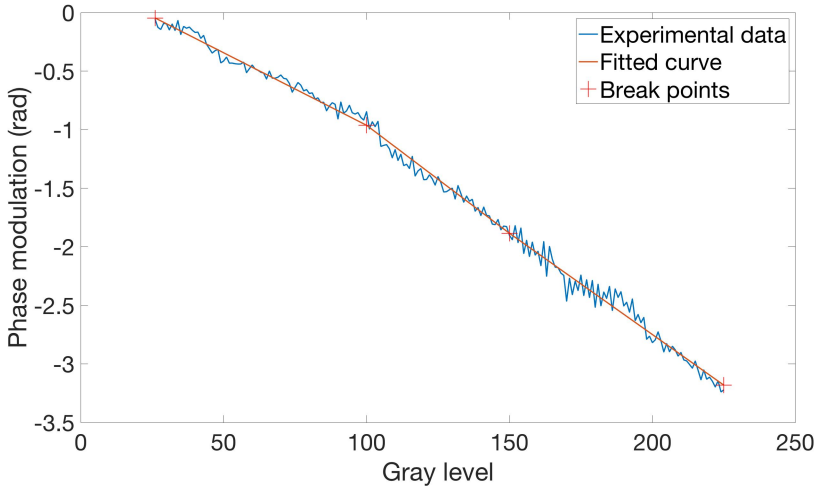

Figure A-9: Phase modulation curve along with three linear segments fitted to the curve.

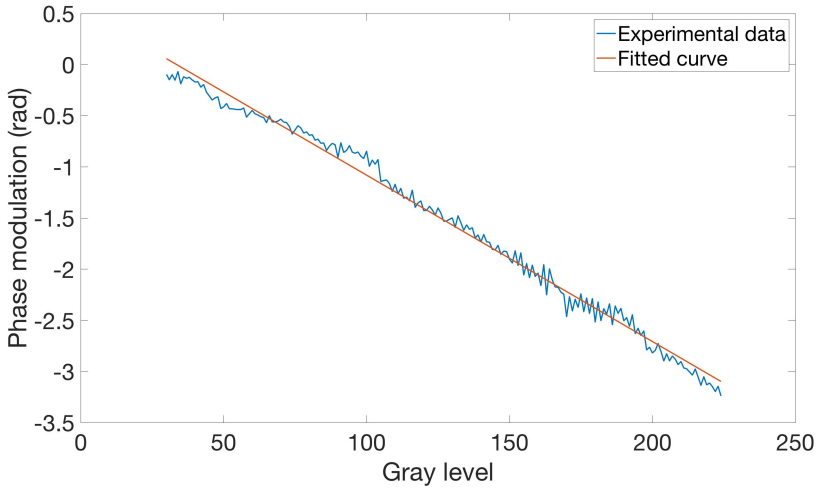

Figure A-10: Phase modulation curve along one linear segment fitted to the curve.

A.3 Calibration of the transmissive SLM in the phase modulation mode

In this section, we describe the approaches that we use to calibrate the intensity modulation and phase modulation curves of the SLM used in the system.

The optical setup for calibrating the intensity modulation of the SLM is shown in Figure A-11(a). We use a photon diode sensor (Newport, 818-SL) as the detector. All the pixels of the SLM is driven by the same value V (uniform) and the corre-

sponding intensity values measured by the photon diode is recorded. By changing V from 0 to 255 and repeating the measurement, we obtain the intensity modulation curve as shown in Figure A-12(a), which is normalized to the intensity values at $V = 0$. The maximum intensity modulation ratio measured was ~ 2.7 for our configuration of polarizer and analyzer. In contrast, per manual specification, the SLM's maximum intensity modulation ratio is ~ 1300 for other polarizer/analyzer configurations. Therefore, we consider this intensity modulation is negligible and the SLM can be approximated as a pure phase object.

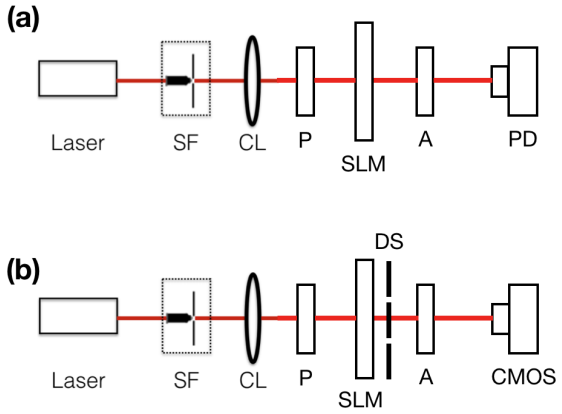

Figure A-11: The optical setup for calibrating the (a) intensity and (b) phase modulation of SLM. SF: spatial filter; CL: collimating lens; P: linear polarizer; A: linear polarization analyzer; SLM: spatial light modulator; DS: double slits; PD: photon diode sensor.

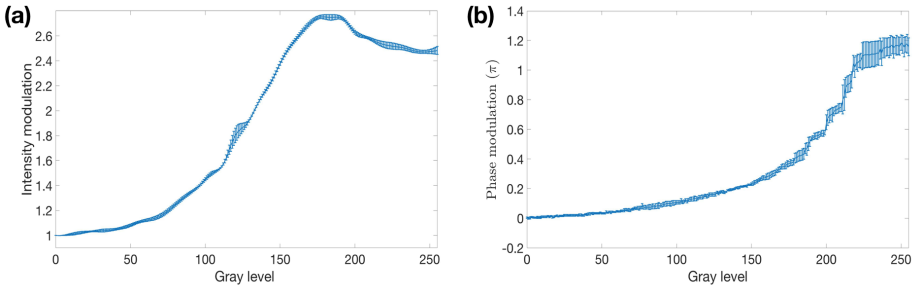

Figure A-12: Experimentally calibrated (a) intensity modulation and (b) phase modulation curve with error bounds in the grayscale range of [0,255] for the SLM.

To calibrate the phase modulation of the SLM, we build the optical setup as shown in Figure A-11(b). A double-slits mask is placed immediately after the SLM, making

the system a Young's interferometer. As a result, a set of interference fringes can be observed at the CMOS plane. The SLM is driven in the half-half mode: the left-hand half pixels is driven by a constant value zero and the right-hand half is driven by a different value V. Under paraxial approximation, the location of the central peak of the interference pattern (constructive interference) can be determined as:

$$x = \frac{\phi(V)\lambda L}{2\pi d} \tag{A.2}$$

where λ is the light wavelength, d is the spacing between the two slits, L is the distance between the mask and the CMOS, $\phi(V)$ is the phase modulation of SLM when it is driven by value V [Assuming $\phi(0) = 0$rad]. Therefore, by gradually increasing V from 0 to 255 and finding the central peak locations of the corresponding interference patterns, we can determine $\phi(V)$.

In our experiment, we use a commercial double-slits mask (3B Scientific) with spacing $d = 250\mu m$ and slit width $d = 150\mu m$. The distance between the mask and the CMOS is set to be $L = 1000$mm. The evolution of 1D profile of the fringes as V increases from 0 to 255 is shown in Figure A-13, and the calibrated phase modulation curve is shown in Figure A-12(b).

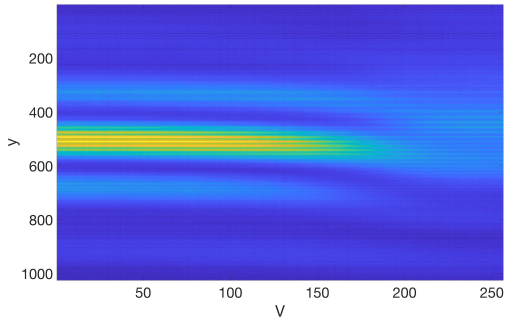

Figure A-13: Evolution of 1D profile of the fringes as V increases from 0 to 255.

Appendix B

Details about the neural network architecture and training

B.1 IDiffNet

As described in Chapter 2, we use densely connected convolutional networks (DenseNets) to construct our IDiffNet. Fig. B-1 shows the detailed architectures of the different blocks used in our IDiffNet. Each dense blocks consist of three composite convolutional layers and each layer connects to every other layer within the same block in a feed-forward fashion. The growth rate k is set to be 12 and the initial number of filters is set to be 16. Each composite convolutional layer is comprised of three consecutive operations: batch normalization (BN), rectified linear unit (ReLU) and dilated convolution (DiConv) with filter size 5×5 and dilation rate 2. We use dilated convolutions so as to increase the receptive field of the convolution filters. The downsampling transition block consists an average pooling operation with stride (2,2). As a result, the dimension of the input to this block is reduced by a factor of 2 at the output. The upsampling transition block increases the dimension of the input by a factor of 2. This is achieved by the subpixel upscaling operation [144]. The dense and downsampling transition block is built by placing a downsampling transition block after a dense block, while the dense and upsampling transition block is built by placing a dense block after a upsampling transition block.

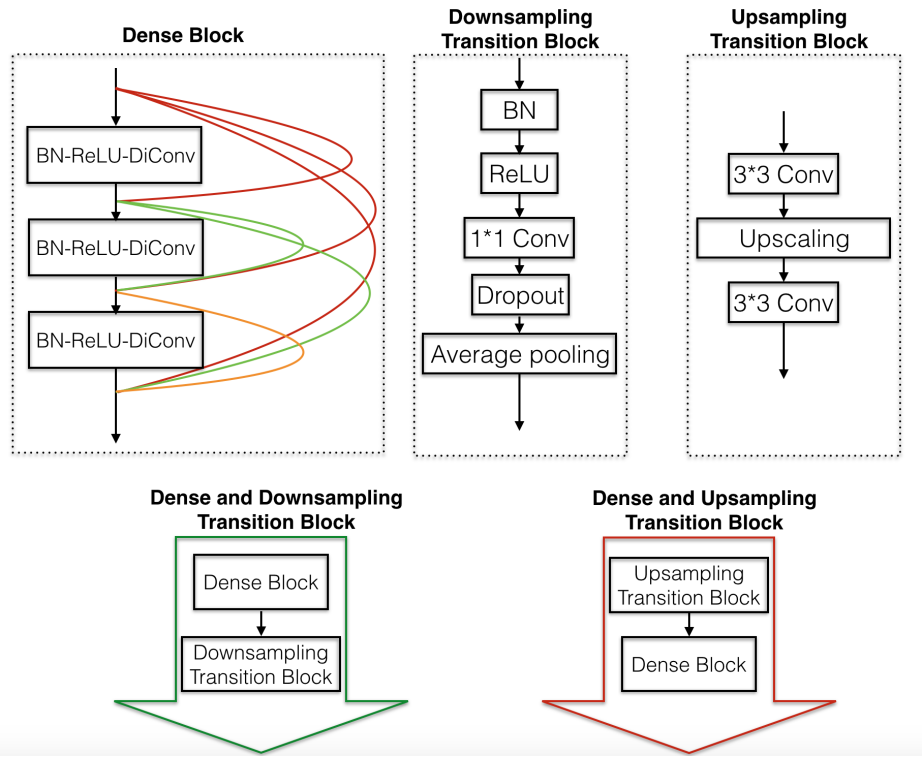

Figure B-1: Detailed architectures of the different blocks in our IDiffNet

We used $\mathcal{L}2$ regularization with weight decay of $1E-4$ in all convolutional filters initialized with random numbers from a Gaussian distribution. The same regularization was used in batch normalization as well. A small dropout rate of 0.05 was set to prevent overfitting. Because of GPU memory constraints, our IDiffNet was trained with a mini-batch size 8 using ADAM optimizer in Tensorflow. We started the training with a learning rate of 0.001 and dropped it by a factor of 2 after every 5 epochs. Additionally, we clipped the gradients at value 1 to stabilize the training. The neural network was trained for 20 epochs with the training samples being shuffled at every epoch.

B.2 PhENN

Fig. B-2 shows three different kinds of residual blocks (bottom row) composed of residual units (top row). All residual units are composed of 2 sets of batch normalization (BN), nonlinearity (ReLU) and a convolutional layer stacked one above the

other. The strides for each convolution or convolution transpose filters are shown in the figure. Short connections are either (a) direct connections that sum the input and output for residual units that do not change the size of the input, (b) 1×1 convolution filters for residual units that change the size of the input, (c) 1×1 convolution filters with stride $(2, 2)$ for residual downsampling units, or (d) 2×2 convolution transpose filters with stride $(2, 2)$ for residual upsampling units. A dilated residual block consists two dilated residual units, which replace the convolution filters in the residual units with dilated convolutional filters of dilation rate 2. All other convolution filters in the network are of size 3×3 and convolution transpose filters are of size 2×2.

We used weight decay of 1E-4 in all convolutional filters initialized with random numbers from a Gaussian distribution. All residual blocks are composed of two residual units. In our experiments, we observed that residual blocks composed of 3 residual units (instead of 2) resulted in slightly lower mean absolute error (MAE) but much longer training times. We set a small dropout rate of 0.02 between all layers to prevent overfitting. We used the ADAM optimizer in Tensorflow to minimize the MAE over the 10000 training samples with batch size 3. The batch size was constrained by the memory available on the GPU. We start the training with a learning rate of 0.001 and drop it by a factor of 2 after every 5 epochs. Additionally, we clip the gradients at value 1 to stabilize the training. We trained the neural network for 20 epochs shuffling the training samples at every epoch.

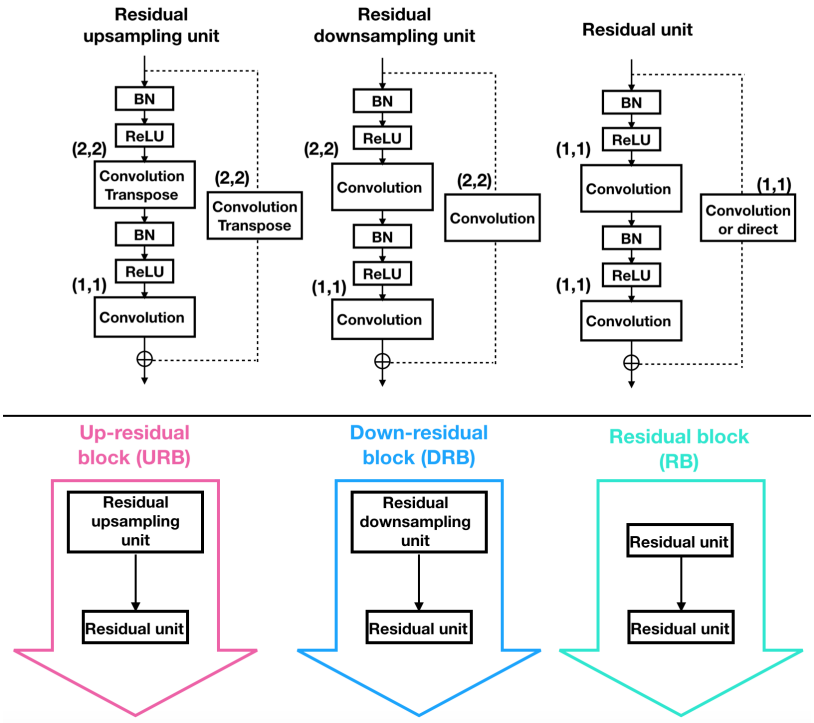

Figure B-2: Detailed architectures of DRBs, URBs and RBs

Bibliography

[1] Mario Bertero and Patrizia Boccacci. *Introduction to inverse problems in imaging*. CRC press, 1998.

[2] David J Brady. *Optical imaging and spectroscopy*. John Wiley & Sons, 2009.

[3] Winfried Denk, James H Strickler, Watt W Webb, et al. Two-photon laser scanning fluorescence microscopy. *Science*, 248(4951):73–76, 1990.

[4] L Moreaux, O Sandre, and J Mertz. Membrane imaging by second-harmonic generation microscopy. *JOSA B*, 17(10):1685–1694, 2000.

[5] Stefan W Hell and Jan Wichmann. Breaking the diffraction resolution limit by stimulated emission: stimulated-emission-depletion fluorescence microscopy. *Optics letters*, 19(11):780–782, 1994.

[6] Rainer Heintzmann and Christoph G Cremer. Laterally modulated excitation microscopy: improvement of resolution by using a diffraction grating. In *Optical Biopsies and Microscopic Techniques III*, volume 3568, pages 185–197. International Society for Optics and Photonics, 1999.

[7] Mats GL Gustafsson. Surpassing the lateral resolution limit by a factor of two using structured illumination microscopy. *Journal of microscopy*, 198(2):82–87, 2000.

[8] T Wilson. Optical sectioning in fluorescence microscopy. *Journal of microscopy*, 242(2):111–116, 2011.

[9] Daryl Lim, Kengyeh K Chu, and Jerome Mertz. Wide-field fluorescence sectioning with hybrid speckle and uniform-illumination microscopy. *Optics letters*, 33(16):1819–1821, 2008.

[10] Guoan Zheng, Roarke Horstmeyer, and Changhuei Yang. Wide-field, high-resolution fourier ptychographic microscopy. *Nature photonics*, 7(9):739, 2013.

[11] Lei Tian, Xiao Li, Kannan Ramchandran, and Laura Waller. Multiplexed coded illumination for fourier ptychography with an led array microscope. *Biomedical optics express*, 5(7):2376–2389, 2014.

[12] Yunhui Zhu, Aamod Shanker, Lei Tian, Laura Waller, and George Barbastathis. Low-noise phase imaging by hybrid uniform and structured illumination transport of intensity equation. *Optics Express*, 22(22):26696–26711, 2014.

[13] Rainer Heintzmann and Thomas Huser. Super-resolution structured illumination microscopy. *Chemical reviews*, 117(23):13890–13908, 2017.

[14] Jeffrey H Shapiro. Computational ghost imaging. *Physical Review A*, 78(6):061802, 2008.

[15] Ori Katz, Yaron Bromberg, and Yaron Silberberg. Compressive ghost imaging. *Applied Physics Letters*, 95(13):131110, 2009.

[16] Minsky Marvin. Microscopy apparatus, December 19 1961. US Patent 3,013,467.

[17] CJR Sheppard and CJ Cogswell. Three-dimensional image formation in confocal microscopy. *Journal of microscopy*, 159(2):179–194, 1990.

[18] Michael Reed Teague. Deterministic phase retrieval: a green's function solution. *Journal of the Optical Society of America*, 73(11):1434–1441, 1983.

[19] Zhong Jingshan, Rene A Claus, Justin Dauwels, Lei Tian, and Laura Waller. Transport of intensity phase imaging by intensity spectrum fitting of exponentially spaced defocus planes. *Optics express*, 22(9):10661–10674, 2014.

[20] Waheb Bishara, Ting-Wei Su, Ahmet F Coskun, and Aydogan Ozcan. Lensfree on-chip microscopy over a wide field-of-view using pixel super-resolution. *Optics express*, 18(11):11181–11191, 2010.

[21] Derek Tseng, Onur Mudanyali, Cetin Oztoprak, Serhan O Isikman, Ikbal Sencan, Oguzhan Yaglidere, and Aydogan Ozcan. Lensfree microscopy on a cellphone. *Lab on a Chip*, 10(14):1787–1792, 2010.

[22] Andrei Nikolaevich Tikhonov. On the solution of ill-posed problems and the method of regularization. In *Doklady Akademii Nauk*, volume 151, pages 501–504. Russian Academy of Sciences, 1963.

[23] Andrei Nikolaevich Tikhonov. On the regularization of ill-posed problems. In *Doklady Akademii Nauk*, volume 153, pages 49–52. Russian Academy of Sciences, 1963.

[24] N Wiener and E Hopf. Über eine klasse singulärer integralgleichungen sitzungsber. *Preuss. Akad. Wiss. Berlin Phys.-Math. Kl*, pages 696–706, 1931.

[25] Norbert Wiener. Extrapolation, interpolation and smoothing of stationary time series-with engineering applications' mit press. 1949.

[26] EJ Candes and T Tao. Decoding by linear programming. *IEEE Transactions on Information Theory*, 51(12):4203–4215, 2005.

[27] Emmanuel J Candès, Justin Romberg, and Terence Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on information theory*, 52(2):489–509, 2006.

[28] David L Donoho et al. Compressed sensing. *IEEE Transactions on information theory*, 52(4):1289–1306, 2006.

[29] Emmanuel J Candes, Justin K Romberg, and Terence Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 59(8):1207–1223, 2006.

[30] Yonina C Eldar and Gitta Kutyniok. *Compressed sensing: theory and applications*. Cambridge University Press, 2012.

[31] Richard Baraniuk, Mark Davenport, Ronald DeVore, and Michael Wakin. A simple proof of the restricted isometry property for random matrices. *Constructive Approximation*, 28(3):253–263, 2008.

[32] Emmanuel J Candes, Michael B Wakin, and Stephen P Boyd. Enhancing sparsity by reweighted ? 1 minimization. *Journal of Fourier analysis and applications*, 14(5-6):877–905, 2008.

[33] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[34] Antonin Chambolle, Ronald A De Vore, Nam-Yong Lee, and Bradley J Lucier. Nonlinear wavelet image processing: variational problems, compression, and noise removal through wavelet shrinkage. *IEEE Transactions on Image Processing*, 7(3):319–335, 1998.

[35] Ingrid Daubechies, Michel Defrise, and Christine De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 57(11):1413–1457, 2004.

[36] Elaine T Hale, Wotao Yin, and Yin Zhang. A fixed-point continuation method for l1-regularized minimization with applications to compressed sensing. *CAAM TR07-07, Rice University*, 43:44, 2007.

[37] José M Bioucas-Dias and Mário AT Figueiredo. A new twist: Two-step iterative shrinkage/thresholding algorithms for image restoration. *IEEE Transactions on Image processing*, 16(12):2992–3004, 2007.

[38] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

[39] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[40] Emmanuel J Candes, Yonina C Eldar, Deanna Needell, and Paige Randall. Compressed sensing with coherent and redundant dictionaries. *Applied and Computational Harmonic Analysis*, 31(1):59–73, 2011.

[41] Mohit Kalra and D Ghosh. Image compression using wavelet based compressed sensing and vector quantization. In *Signal Processing (ICSP), 2012 IEEE 11th International Conference on*, volume 1, pages 640–645. IEEE, 2012.

[42] Pietro Perona and Jitendra Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on pattern analysis and machine intelligence*, 12(7):629–639, 1990.

[43] Leonid I Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268, 1992.

[44] Joachim Weickert. A review of nonlinear diffusion filtering. In *International Conference on Scale-Space Theories in Computer Vision*, pages 1–28. Springer, 1997.

[45] Lei Tian, Jonathan C Petruccelli, and George Barbastathis. Nonlinear diffusion regularization for transport of intensity phase imaging. *Optics letters*, 37(19):4131–4133, 2012.

[46] Michael Elad and Michal Aharon. Image denoising via learned dictionaries and sparse representation. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 895–900. IEEE, 2006.

[47] Michal Aharon, Michael Elad, Alfred Bruckstein, et al. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on signal processing*, 54(11):4311, 2006.

[48] Bruno A Olshausen and David J Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607, 1996.

[49] Ron Rubinstein, Alfred M Bruckstein, and Michael Elad. Dictionaries for sparse representation modeling. *Proceedings of the IEEE*, 98(6):1045–1057, 2010.

[50] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[51] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. *Applied logistic regression*, volume 398. John Wiley & Sons, 2013.

[52] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[53] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[54] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[55] Robert Schalkoff. Pattern recognition: Statistical, structural and neural approaches, john wiley & sons. *Inc, New York*, 1992.

[56] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

[57] Bekir Karlik and A Vehbi Olgac. Performance analysis of various activation functions in generalized mlp architectures of neural networks. *International Journal of Artificial Intelligence and Expert Systems*, 1(4):111–122, 2011.

[58] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[59] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3, 2013.

[60] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[61] Yann LeCun et al. Generalization and network design strategies. In *Connectionism in perspective*, volume 19. Citeseer, 1989.

[62] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193–202, 1980.

[63] Y-Lan Boureau, Jean Ponce, and Yann LeCun. A theoretical analysis of feature pooling in visual recognition. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 111–118, 2010.

[64] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.

[65] Anders Krogh and John A Hertz. A simple weight decay can improve generalization. In *Advances in neural information processing systems*, pages 950–957, 1992.

[66] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[67] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 1986.

[68] Valerian Ilich Tatarski. *Wave propagation in a turbulent medium*. Courier Dover Publications, 2016.

[69] Akira Ishimaru. *Wave propagation and scattering in random media*, volume 2. Academic press New York, 1978.

[70] SM Popoff, G Lerosey, R Carminati, M Fink, AC Boccara, and S Gigan. Measuring the transmission matrix in optics: an approach to the study and control of light propagation in disordered media. *Physical review letters*, 104(10):100601, 2010.

[71] SM Popoff, G Lerosey, M Fink, AC Boccara, and S Gigan. Image transmission through an opaque material. *Nature Communications*, 1:81, 2010.

[72] Angélique Drémeau, Antoine Liutkus, David Martina, Ori Katz, Christophe Schülke, Florent Krzakala, Sylvain Gigan, and Laurent Daudet. Reference-less measurement of the transmission matrix of a highly scattering material using a dmd and phase retrieval techniques. *Optics express*, 23(9):11898–11911, 2015.

[73] Jacopo Bertolotti, Elbert G van Putten, Christian Blum, Ad Lagendijk, Willem L Vos, and Allard P Mosk. Non-invasive imaging through opaque scattering layers. *Nature*, 491(7423):232–234, 2012.

[74] Ori Katz, Pierre Heidmann, Mathias Fink, and Sylvain Gigan. Non-invasive single-shot imaging through scattering layers and around corners via speckle correlations. *Nature photonics*, 8(10):784–790, 2014.

[75] Nicolino Stasio, Christophe Moser, and Demetri Psaltis. Calibration-free imaging through a multicore fiber using speckle scanning microscopy. *Optics letters*, 41(13):3078–3081, 2016.

[76] Amir Porat, Esben Ravn Andresen, Hervé Rigneault, Dan Oron, Sylvain Gigan, and Ori Katz. Widefield lensless imaging through a fiber bundle via speckle correlations. *Optics express*, 24(15):16835–16855, 2016.

[77] Gerwin Osnabrugge, Roarke Horstmeyer, Ioannis N Papadopoulos, Benjamin Judkewitz, and Ivo M Vellekoop. Generalized optical memory effect. *Optica*, 4(8):886–892, 2017.

[78] Shechao Feng, Charles Kane, Patrick A Lee, and A Douglas Stone. Correlations and fluctuations of coherent wave transmission through disordered media. *Physical review letters*, 61(7):834, 1988.

[79] Isaac Freund, Michael Rosenbluh, and Shechao Feng. Memory effects in propagation of optical waves through disordered media. *Physical review letters*, 61(20):2328, 1988.

[80] Eric Akkermans and Gilles Montambaux. *Mesoscopic physics of electrons and photons*. Cambridge university press, 2007.

[81] Ralph W Gerchberg. A practical algorithm for the determination of phase from image and diffraction plane pictures. *Optik*, 35:237–246, 1972.

[82] James R Fienup. Reconstruction of an object from the modulus of its fourier transform. *Optics Letters*, 3(1):27–29, 1978.

[83] YongKeun Park, Wonshik Choi, Zahid Yaqoob, Ramachandra Dasari, Kamran Badizadegan, and Michael S Feld. Speckle-field digital holographic microscopy. *Optics express*, 17(15):12285–12292, 2009.

[84] Nick Antipa, Grace Kuo, Reinhard Heckel, Ben Mildenhall, Emrah Bostan, Ren Ng, and Laura Waller. Diffusercam: lensless single-exposure 3d imaging. *Optica*, 5(1):1–9, 2018.

[85] Ulf Grenander. *General pattern theory-A mathematical study of regular structures*. Clarendon Press, 1993.

[86] David J Brady, Kerkil Choi, Daniel L Marks, Ryoichi Horisaki, and Sehoon Lim. Compressive holography. *Optics Express*, 17(15):13040–13049, 2009.

[87] Hsiou-Yuan Liu, Eric Jonas, Lei Tian, Jingshan Zhong, Benjamin Recht, and Laura Waller. 3d imaging in volumetric scattering media using phase-space measurements. *Optics express*, 23(11):14461–14471, 2015.

[88] Ryoichi Horisaki, Ryosuke Takagi, and Jun Tanida. Learning-based imaging through scattering media. *Optics express*, 24(13):13738–13743, 2016.

[89] Meng Lyu, Hao Wang, Guowei Li, and Guohai Situ. Exploit imaging through opaque wall via deep learning. *arXiv preprint arXiv:1708.07881*, 2017.

[90] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

[91] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[92] Joseph W Goodman. *Introduction to Fourier optics*. Roberts and Company Publishers, 2005.

[93] Nicholas Antipa, Sylvia Necula, Ren Ng, and Laura Waller. Single-shot diffuser-encoded light field imaging. In *Computational Photography (ICCP), 2016 IEEE International Conference on*, pages 1–11. IEEE, 2016.

[94] https://www.unc.edu/~rowlett/units/scales/grit.html.

[95] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. Densely connected convolutional networks. *arXiv preprint arXiv:1608.06993*, 2016.

[96] Xiaojiao Mao, Chunhua Shen, and Yu-Bin Yang. Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. In *Advances in neural information processing systems*, pages 2802–2810, 2016.

[97] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.

[98] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[99] Gary B Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report, Technical Report 07-49, University of Massachusetts, Amherst, 2007.

[100] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[101] Yann LeCun, Corinna Cortes, and Christopher JC Burges. Mnist handwritten digit database. *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*, 2, 2010.

[102] Cheng-Lin Liu, Fei Yin, Da-Han Wang, and Qiu-Feng Wang. Casia online and offline chinese handwriting databases. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 37–41. IEEE, 2011.

[103] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[104] Ferdinando S Samaria and Andy C Harter. Parameterisation of a stochastic model for human face identification. In *Proceedings of the Second IEEE Workshop on Applications of Computer Vision*, pages 138–142. IEEE, 1994.

[105] AT&T Laboratories Cambridge. AT&T Database of Faces, 1994. data retrieved from https://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html.

[106] A Miranda Neto, A Correa Victorino, Isabelle Fantoni, Douglas Eduardo Zampieri, Janito Vaqueiro Ferreira, and Danilo Alves Lima. Image processing using pearson's correlation coefficient: Applications on autonomous robotics. In *Autonomous Robot Systems (Robotica), 2013 13th International Conference on*, pages 1–6. IEEE, 2013.

[107] Clement Gehring and Simon Lemay. Sparse coding. *sibi*, 1:1, 2012.

[108] Byung-soo Kim, Jae Young Park, Anna C Gilbert, and Silvio Savarese. Hierarchical classification of images by sparse approximation. *Image and Vision Computing*, 31(12):982–991, 2013.

[109] Tal Remez, Or Litany, Raja Giryes, and Alex M Bronstein. Deep convolutional denoising of low-light images. *arXiv preprint arXiv:1701.01687*, 2017.

[110] Frits Zernike. Phase contrast, a new method for the microscopic observation of transparent objects part ii. *Physica*, 9(10):974–986, 1942.

[111] Joseph W Goodman and RW Lawrence. Digital image formation from electronically detected holograms. *Applied Physics Letters*, 11(3):77–79, 1967.

[112] Yair Rivenson, Adrian Stern, and Bahram Javidi. Compressive fresnel holography. *Journal of Display Technology*, 6(10):506–509, 2010.

[113] Jerome H Milgram and Weichang Li. Computational reconstruction of images from holograms. *Applied Optics*, 41(5):853–864, 2002.

[114] Logan Williams, Georges Nehmetallah, and Partha P Banerjee. Digital tomographic compressive holographic reconstruction of three-dimensional objects in transmissive and reflective geometries. *Applied Optics*, 52(8):1702–1710, 2013.

[115] Katherine Creath. Phase-shifting speckle interferometry. *Applied Optics*, 24(18):3053–3058, 1985.

[116] Shan Shan Kou, Laura Waller, George Barbastathis, and Colin JR Sheppard. Transport-of-intensity approach to differential interference contrast (ti-dic) microscopy for quantitative phase imaging. *Optics Letters*, 35(3):447–449, 2010.

[117] David Paganin and Keith A Nugent. Noninterferometric phase imaging with partially coherent light. *Physical Review Letters*, 80(12):2586, 1998.

[118] Jelena A Schmalz, Timur E Gureyev, David M Paganin, and Konstantin M Pavlov. Phase retrieval using radiation and matter-wave fields: Validity of teague's method for solution of the transport-of-intensity equation. *Physical Review A*, 84(2):023808, 2011.

[119] Laura Waller, Shan Shan Kou, Colin JR Sheppard, and George Barbastathis. Phase from chromatic aberrations. *Optics Express*, 18(22):22817–22825, 2010.

[120] Laura Waller, Mankei Tsang, Sameera Ponda, Se Young Yang, and George Barbastathis. Phase and amplitude imaging from noisy images by kalman filtering. *Optics Express*, 19(3):2805–2815, 2011.

[121] Lei Tian, Jonathan C Petruccelli, Qin Miao, Haris Kudrolli, Vivek Nagarkar, and George Barbastathis. Compressive x-ray phase tomography based on the transport of intensity equation. *Optics Letters*, 38(17):3418–3421, 2013.

[122] Adam Pan, Ling Xu, Jon C Petruccelli, Rajiv Gupta, Bipin Singh, and George Barbastathis. Contrast enhancement in x-ray phase contrast tomography. *Optics Express*, 22(15):18020–18026, 2014.

[123] RA Gonsalves. Phase retrieval from modulus data. *Journal of the Optical Society of America*, 66(9):961–964, 1976.

[124] JR Fienup and CC Wackerman. Phase-retrieval stagnation problems and solutions. *Journal of the Optical Society of America A*, 3(11):1897–1907, 1986.

[125] Heinz H Bauschke, Patrick L Combettes, and D Russell Luke. Phase retrieval, error reduction algorithm, and fienup variants: a view from convex optimization. *Journal of the Optical Society of America A*, 19(7):1334–1345, 2002.

[126] Emmanuel J Candes, Thomas Strohmer, and Vladislav Voroninski. Phaselift: Exact and stable signal recovery from magnitude measurements via convex programming. *Communications on Pure and Applied Mathematics*, 66(8):1241–1274, 2013.

[127] Emmanuel J Candes, Xiaodong Li, and Mahdi Soltanolkotabi. Phase retrieval via wirtinger flow: Theory and algorithms. *IEEE Transactions on Information Theory*, 61(4):1985–2007, 2015.

[128] Philip Schniter and Sundeep Rangan. Compressive phase retrieval via generalized approximate message passing. *IEEE Transactions on Signal Processing*, 63(4):1043–1055, 2015.

[129] Michael R Kellman, Emrah Bostan, Nicole Repina, Michael Lustig, and Laura Waller. Physics-based learned design: Optimized coded-illumination for quantitative phase imaging. *arXiv preprint arXiv:1808.03571*, 2018.

[130] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778. IEEE, 2016.

[131] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *ArXiv:1511.07122*, 2015.

[132] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[133] van A Van der Schaaf and JH van van Hateren. Modelling the power spectra of natural images: statistics and information. *Vision Research*, 36(17):2759–2770, 1996.

[134] L1 determines the aperture stop with diameter 25.4mm, *i.e.* a numerical aperture NA= 12.7/150 = 0.0847. The nominal diffraction-limited resolution should be $d_0 = \lambda/(2\text{NA}) = 3.74\mu$m. That calculation is irrelevant to PhENN, since objets of that spatial frequency are never presented to it during training.

[135] Jinshan Pan, Sifei Liu, Deqing Sun, Jiawei Zhang, Yang Liu, Jimmy Ren, Zechao Li, Jinhui Tang, Huchuan Lu, Yu-Wing Tai, and Ming-Hsuan Yang. Learning dual convolutional neural networks for low-level vision. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[136] Zachary F Phillips and Michael Chen. Technical report: Benchmark technologies quantitative phase target.

[137] Yair Rivenson, Yibo Zhang, Harun Günaydın, Da Teng, and Aydogan Ozcan. Phase recovery and holographic image reconstruction using deep learning in neural networks. *Light: Science & Applications*, 7(2):17141, 2018.

[138] Thanh Nguyen, Yujia Xue, Yunzhe Li, Lei Tian, and George Nehmetallah. Deep learning approach for fourier ptychography microscopy. *Optics express*, 26(20):26470–26484, 2018.

[139] Yichen Wu, Yair Rivenson, Yibo Zhang, Zhensong Wei, Harun Günaydin, Xing Lin, and Aydogan Ozcan. Extended depth-of-field in holographic imaging using deep-learning-based autofocusing and phase recovery. *optica*, 5(6):704–710, 2018.

[140] Alexandre Goy, Kwabena Arthur, Shuai Li, and George Barbastathis. Low photon count phase retrieval using deep learning. *Physical review letters*, 121(24):243902, 2018.

[141] YongKeun Park, Monica Diez-Silva, Gabriel Popescu, George Lykotrafitis, Wonshik Choi, Michael S Feld, and Subra Suresh. Refractive index maps and

membrane dynamics of human red blood cells parasitized by plasmodium falciparum. *Proceedings of the National Academy of Sciences*, 105(37):13730–13735, 2008.

[142] Monica Diez-Silva, Ming Dao, Jongyoon Han, Chwee-Teck Lim, and Subra Suresh. Shape and biomechanical characteristics of human red blood cells in health and disease. *MRS bulletin*, 35(5):382–388, 2010.

[143] Lei Tian and Laura Waller. Quantitative differential phase contrast imaging in an led array microscope. *Optics express*, 23(9):11394–11403, 2015.

[144] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1874–1883, 2016.