Unsupervised Phase Retrieval Using Deep Approximate MMSE Estimation

Mingqin Chen, Peikang Lin, Yuhui Quan*, Tongyao Pang, and Hui Ji

***Abstract*—Phase retrieval (PR) is about reconstructing a signal from the magnitude of a number of its complex-valued linear measurements. Recent rapid progress has been made on the development of neural network (NN) based methods for PR. Most of these methods employ pre-trained NNs for modeling target signals, and they require collecting large-scale datasets with ground-truth signals for pre-training, which can be very challenging in many scenarios. There are a few unsupervised learning methods employing untrained NN priors for PR which avoid using external datasets; however, their performance is unsatisfactory compared to pre-trained-NN-based methods. This paper proposes an unsupervised learning method for PR which does not rely on pre-trained NNs while providing state-of-the-art performance. The proposed method trains a randomly-initialized generative NN for signal reconstruction directly on the magnitude measurements of a target signal, which approximates the minimum mean squared error estimator via dropout-based model averaging. Such a model-averaging-based approach provides a better internal prior for the target signal than existing untrained-NN-based methods. The experiments on image reconstruction demonstrate both the advantage of our method over existing unsupervised methods and its competitive performance to pre-trained-NN-based methods.**

***Index Terms*—Phase retrieval, Inverse problems, Unsupervised learning, Untrained neural networks**

I. Introduction

In many physical measurement systems, one can only obtain the magnitude of certain linear measurements of a signal, *e.g.*, the power spectral density which is the magnitude of the Fourier transform of a signal. Since the phase encodes a lot of the structural content of a signal, the lack of phase will cause the loss of important information. The problem of reconstructing a signal from the magnitude of its linear measurements is known as phase retrieval (PR). It can be expressed as solving the following equation:

$$\boldsymbol{y}_0 = |\boldsymbol{A}\boldsymbol{x}_0| + \boldsymbol{n}, \tag{1}$$

where $\boldsymbol{x}_0 \in \mathbb{R}^N$ (or $\mathbb{C}^N$) denotes the signal for reconstruction in vector form, $\boldsymbol{n}$ denotes the measurement noise, $\boldsymbol{A} \in \mathbb{C}^{M\times N}$ denotes some linear transform, *e.g.* discrete Fourier transform (DFT), and $\boldsymbol{y}_0 \in \mathbb{R}^M$ denotes the collected measurements in vector form. This problem has a rich history and arises in many areas of engineering and applied physics, such as diffraction imaging [1], astronomical imaging [2], microscopy imaging [3], holographic imaging [4], ptychography [5], and crystallography [6]. For instance, diffraction imaging often uses the following system for measuring a signal:

$$\boldsymbol{A} = \left[(\boldsymbol{F}\boldsymbol{D}_1)^\top, \cdots, (\boldsymbol{F}\boldsymbol{D}_R)^\top\right]^\top, \tag{2}$$

where $\boldsymbol{F}$ is the DFT matrix, and $\boldsymbol{D}_1, ..., \boldsymbol{D}_R$ are defined as

$$\boldsymbol{D}_r\boldsymbol{x} \to \boldsymbol{d}_r \odot \boldsymbol{x},\ r = 1, \cdots, R. \tag{3}$$

The notation $\odot$ represents the Hadamard product, and $\boldsymbol{d}_r (r = 1, \cdots, R) \in \mathbb{C}^N$ are illumination masks for coded diffraction patterns (CDPs).

As a non-linear inverse problem, PR is usually ill-posed due to the lack of phase information and the insufficiency of measurements. There is no unique solution in general since any choice of the Fourier phase will generate a valid solution which can be far from the original signal [7]. Also, the inherent ambiguities about shift and flip in the Fourier measurements make PR very difficult [8]. In certain applications, *e.g.* compressive PR [9], [10], under-sampled PR [11], and super-resolution PR [12], there are additional solution ambiguities caused by the down-sampling process involved in the collection of measurements. How to resolve the ambiguity of solutions and the sensitivity to measurement noise, is a key issue to address when developing an effective method for PR.

There have been extensive studies on regularization methods for PR. In a regularization method, certain prior is imposed on the signal during reconstruction to resolve the solution ambiguity and possible noise amplification. A typical regularization method for PR can be expressed as

$$\min_{\boldsymbol{x}} \mathcal{L}\big(|\boldsymbol{A}\boldsymbol{x}|, \boldsymbol{y}_0\big) + \phi(\boldsymbol{x}), \tag{4}$$

where $\phi(\cdot)$ is some function derived from the prior on the signal, and $\mathcal{L}(\cdot, \cdot)$ is the fidelity term which depends on the statistical characteristics of measurement noise. For instance, in the presence of Gaussian white noise, we have

$$\mathcal{L}(|\boldsymbol{A}\boldsymbol{x}|, \boldsymbol{y}_0) = \frac{1}{2}\||\boldsymbol{A}\boldsymbol{x}| - \boldsymbol{y}_0\|_2^2. \tag{5}$$

In the past, one often-seen prior for imaging-related PR is the sparsity prior on image gradients, which assumes the underlying signal is piece-wise smooth with sparse gradients. Such a prior leads to the ℓ_1-norm-based regularization methods [11], [13], [14]. Another often-seen prior is the self-recurrence prior of local image patches, which assumes that each image patch is

Mingqin Chen, Peikang Lin and Yuhui Quan are with School of Computer Science and Engineering at South China University of Technology, Guangzhou 510006, China. (email: csmingqinchen@mail.scut.edu.cn; csalanlin@mail.scut.edu.cn; csyhquan@scut.edu.cn)

Tongyao Pang and Hui Ji are with Department of Mathematics at National University of Singapore, Singapore 119076. (email: matpt@nus.edu.sg, matjh@nus.edu.sg)

Corresponding author: Yuhui Quan

This work was supported in part by National Natural Science Foundation of China under Grant 61872151, and in part by Singapore MOE Academic Research Fund under Grant R-146-000-315-114.

likely to repeat itself many times over the image. Such a global recurrence prior leads to the so-called non-local methods [10], [15]–[17].

In recent years, deep learning has emerged as one promising approach for solving the PR problems in imaging, with an increasing interest in the community. Many works have been proposed to replace manually-crafted priors by the NN-based priors learned from training data. For instance, one approach is to unroll some iterative scheme for solving (4) and replaces the steps which involve the regularization term by denoising NNs. One or more denoising NNs are pre-trained over many pairs of noisy/ground-truth signals, and then plugged into the iterative scheme; see *e.g.* [18], [19]. Given sufficient amount of training data, such a plug-and-play approach showed its advantage over traditional regularization methods. Instead of using denoising NNs, another approach is modeling the signal by a generative NN, which considers

$$\boldsymbol{x}_0 = \mathcal{G}_{\boldsymbol{\mu}}(\boldsymbol{\epsilon}), \quad \boldsymbol{y}_0 = |\boldsymbol{A}\mathcal{G}_{\boldsymbol{\mu}}(\boldsymbol{\epsilon})| + \boldsymbol{n}, \tag{6}$$

where $\mathcal{G}_{\boldsymbol{\mu}}(\cdot)$ is an NN parameterized by $\boldsymbol{\mu}$ with some seed $\boldsymbol{\epsilon}$ as input. In [20], [21], a generative adversarial network (GAN), denoted by $\mathcal{G}^{\mathrm{GAN}}_{\boldsymbol{\mu}_0}$, is pre-trained over training samples and used as $\mathcal{G}_{\boldsymbol{\mu}}$. Then, the PR problem is solved by finding the seed through

$$\boldsymbol{\epsilon}^\star := \operatorname*{argmin}_{\boldsymbol{\epsilon}} \mathcal{L}\big(|\boldsymbol{A}\mathcal{G}^{\mathrm{GAN}}_{\boldsymbol{\mu}_0}(\boldsymbol{\epsilon})|, \boldsymbol{y}_0\big), \tag{7}$$

and the signal of interest is finally reconstructed as $\mathcal{G}^{\mathrm{GAN}}_{\boldsymbol{\mu}_0}(\boldsymbol{\epsilon}^\star)$.

All aforementioned deep-learning-based methods require an external dataset with ground-truth signals for the pre-training of NNs. There are several concerns on such a requirement. One is that in many scenarios, *e.g.* scientific imaging and medical imaging, it is very challenging and sometimes even impossible to collect many ground-truth signals with a very high signal-to-noise ratio (SNR). Another issue is the possible bias introduced by the training dataset. As the NN-based prior is learned over the training dataset, the reconstructed signal may be biased to the characteristics of the training samples, which can be either undesired in scientific imaging whose goal is for discovery, or very risky for medical imaging whose goal is for disease diagnosis and examination.

There is certainly a lot of interest in many scenarios to have a generative-NN-based method for PR that uses an untrained NN (*i.e.* not pre-trained but randomly initialized) for modeling the signal to be reconstructed, while still providing competitive performance to the methods that call pre-trained models. Owing to the great challenges raised by the absence of ground-truth training signals, there have been very few works on the development of such untrained-NN-based methods. An available one is Jagatap *et al.* [22] which introduced an under-parameterized two-layer decoder as $\mathcal{G}_{\boldsymbol{\mu}}$ for modeling the image prior and solved compressive PR as follows:

$$\min_{\boldsymbol{x}} \||\boldsymbol{A}\boldsymbol{x}| - \boldsymbol{y}_0\|_2^2, \quad \text{s.t.} \quad \boldsymbol{x} = \mathcal{G}_{\boldsymbol{\mu}}(\boldsymbol{z}_0), \tag{8}$$

where the seed $\boldsymbol{z}_0$ is fixed and the weights encoded by $\boldsymbol{\mu}$ need to be estimated. They also provided theoretical guarantees that under certain conditions, the solution with such a two-layer decoder-based prior can approximate the ground-truth sufficiently. While a heavily under-parameterized decoder can avoid overfitting such that the output image contains few artifacts, its modeling capacity is also severely limited. As a result, it will cause under-fitting such that complex image patterns (*e.g.* textures) contained in the ground-truth image will be erased in the output. Compared to the pre-trained-NN based methods, the performance of existing untrained-NN-based methods is not satisfactory. Their performance is even not as good as traditional handcrafted-prior-based methods.

A. Motivation and Basic Idea

This paper aims at developing an unsupervised deep learning method for PR, which leverages the image prior induced by the structure of an untrained NN to provide state-of-the-art (SOTA) performance, while avoiding the need for pre-training models or collecting ground-truth data. We consider training a randomly-initialized generative NN parameterized by $\boldsymbol{\mu}$, denoted by $\mathcal{G}_{\boldsymbol{\mu}}$, to reconstruct the image by solving

$$\min_{\boldsymbol{\mu}} \mathcal{L}\big(|\boldsymbol{A}\mathcal{G}_{\boldsymbol{\mu}}(\boldsymbol{\epsilon}_0)|, \boldsymbol{y}_0\big), \tag{9}$$

for some random seed $\boldsymbol{\epsilon}_0$. At a quick glance, there is no regularization on the prediction of image when solving (9). As a result, the solution ambiguity is not addressed and thus the overfitting is likely to occur.

Recently, the so-called *deep image prior* (DIP) [23] showed that the architecture of a convolutional neural network (CNN) itself imposed certain regularization property on the prediction. It is observed that when training a CNN to reconstruct a noisy image, regular structures appear before random noise in the output. Thus, the DIP uses early stopping during the training to output a noise-free prediction. The DIP showed fine denoising performance in the absence of external training samples. Nevertheless, while it is effective on removing noise from images, the DIP may not handle the severe solution ambiguity in PR effectively. Avoiding random noise in the estimation is simply not enough to provide an accurate image estimate in PR.

The DIP can be viewed as a maximum a posterior (MAP) estimator with implicit image priors induced by the CNN architecture. It is known that the minimum mean squared error (MMSE) estimator is another favorite Bayesian estimator different from the MAP estimator, and one main issue for the MMSE estimator lies in its computational feasibility. In this paper, we propose to train a generative NN which approximates the MMSE estimate of the image. To tackle the intractability of the posterior distribution of network parameters, we approximate the posterior distribution with a surrogate distribution parameterized by Bernoulli variables. The main reason for adopting such a surrogate distribution comes from the fact that it can be implemented with dropout [24], which is very computationally efficient and effective on regularizing the NN. The approximation of the surrogate distribution to the posterior distribution is done via minimizing the Kullback-Leibler (KL) divergence between them, which is equivalent to training the NN with dropout to approximate the given measurement data under a reconstruction loss.

Once the surrogate distribution is estimated via the dropout training, the integral involved in the MMSE estimator is then approximated by the Monte Carlo method, which is equivalent to sampling the trained NN with dropout to have multiple inferences of the image. Then, the final prediction is defined as the average of these inferences. In the context of Bayesian inference [25], [26], such an averaging scheme for prediction is effective on reducing the variance of the estimate, which is very helpful on resolving the solution ambiguity in PR.

B. Contributions

1) An effective untrained-NN-based method for PR built upon MMSE approximation: Compared to existing untrained-NN-based methods for PR, the proposed one exploits untrained NN priors based on an MMSE approximation framework. This leads to image (signal) priors with higher accuracy.

2) Competitive performance: Experiments show that our method bridges the performance gap between untrained-NN-based methods and pre-trained-NN based methods. It not only significantly outperforms traditional handcrafted-prior-based methods and existing untrained-NN-based methods, but also outperforms recent supervised-NN-based methods.

3) Dropout-based Bayesian approximation technique for resolving the ill-posedness of nonlinear inverse problems: This paper presents a technique to resolve the arisen ambiguity in PR, which implements an approximate MMSE estimation efficiently based on dropout. Such a technique has possible applications on solving other non-linear inverse problems.

II. Related Work

The studies on PR started several decades ago; see *e.g.* [1], [27]. However, the performance of these methods is unsatisfactory and sensitive to the initialization [28]. Since the publication of the seminal work [29] which uses convex programming for solving the problem of PR with theoretical guarantees, PR has drawn increasing attention from the optimization society and many subsequent works have emerged; see *e.g.* [30]–[39]. Most of these methods focus on the regularization for solving the problem. With its great success in many image-related applications, deep learning has been introduced to tackle the PR problem with very promising performance; see *e.g.* [20]–[22], [40]. Owing to the difficulty in training data collection, the end-to-end NN training is not considered in existing deep-learning-based approaches for PR in general, except for very specific scenarios and data. There are a few deep-learning-based approaches proposed for PR in specific imaging configurations. For instance, Hyder *et al.* [8] assumed that a known reference is added to the signal before capturing the measurements. In the next, we give a detailed discussion on recent regularization-based methods and deep-NN-based methods that are most related to our configurations for PR.

A. Hand-crafted Priors for PR

The most often-seen prior in the regularization methods for PR is the sparsity prior, which assumes signals are sparsified in the gradient domain or under some transform or dictionary. Chang *et al.* [13] used total variation (TV) regularization on the latent image for PR in the presence of Poisson noise. Shi *et al.* [41] proposed an accelerated algorithm for solving PR with TV regularization. Instead of using the sparsity prior on image gradients, Tillmann *et al.* [42] proposed a dictionary learning method to regularize the image via ℓ_1-regularization under the learned dictionary. For acceleration, Liu *et al.* [43] proposed a parallel algorithm for dictionary-based PR. Qiu *et al.* [11] tackled the problem of under-sampled PR using dictionary learning, and proposed a low-complexity algorithm for solving the related optimization problem via majorization minimization. Based on the self-recurrence of image patches, some methods [16], [17], [44] use the nonlocal sparsifying frames [45] to regularize the estimation. In [14], the sparsity of image gradients exploited by TV regularization is combined with the self-recurrence of image patches exploited by the spectral sparsity regularization on matched patches.

B. Plug-and-Play Denoising Priors for PR

If an image denoiser can effectively remove random noise from a noisy image, then it encodes accurate prior knowledge on noise-free images. Thus, many methods have been proposed by plugging some existing denoisers into the pipeline of a PR approach. Most of these methods unroll the iterative scheme of some regularization approach for PR. By viewing a certain step as a denoising process, an effective off-the-shelf image denoiser is then used to replace such a step. For instance, Metzler *et al.* [10] incorporated the well-established image denoiser, BM3D (block matching and 3D filtering) [46], in an iterative scheme unrolled by the generalized approximate message passing.

With recent rapid progress on deep-learning-based image denoisers, some approaches were proposed by plugging the NN denoiser pre-trained on some denoising dataset into the unrolled PR process. As long as the dataset for pre-training is closely related to the data for reconstruction, these methods benefit from the performance advantage of deep-learning-based denoisers over the traditional ones such as BM3D and sparsity-based regularization methods. Metzler *et al.* [18] proposed a method named prDeep for PR that leverages the regularization-by-denoising framework and a pre-trained off-the-shelf NN denoiser $\mathcal{D}(\cdot)$ (*e.g.* DnCNN [47]) by employing the explicit regularization term $\phi(\boldsymbol{x}) = \langle \boldsymbol{x}, \boldsymbol{x} - \mathcal{D}(\boldsymbol{x}) \rangle$. The FASTA (fast adaptive shrinkage/thresholding algorithm) is used to solve the regularized optimization problem. Shi *et al.* [19] proposed to combine the sparsity-based prior under some frames and an off-the-shelf NN-based denoiser by penalizing the difference between the frame coefficients of the image itself and that of its denoised version.

It is noted that the performance of above plug-and-play approaches highly depends on how well the off-the-shelf denoiser performs on target images. In the case of deep learning, the denoiser should be trained on a dataset where the contained images are highly related to the data for reconstruction.

C. Deep Generative Priors for PR

Instead of using pre-trained denoising NNs, a few methods (*e.g.* [20], [21]) exploit pre-trained generative NNs for PR.

The basic idea is to find the optimal seed for a pre-trained generative NN such that the NN's output fit the constraints of the magnitude measurements. Hand *et al.* [20] adopted the convolutional GAN from [48] as well as a variational auto-encoder as the generators, while Shamshad *et al.* [21] used the convolutional GAN from [40]. These two methods showed the advantages of deep generative priors over denoising priors and sparsity-based priors when dealing with images in certain domains (*e.g.* face images).

D. Untrained NN Priors for PR

The performance of above generative-prior-based methods relies on how well a pre-trained model can generalize to target images. When target images contain structures and patterns unseen in the samples used for pre-training, such a pre-trained model would not perform well on target images. There are several studies exploiting the prior induced by the structure of an untrained NN for PR; see *e.g.* [22], [49]–[51]. Instead of optimizing the input seed for a pre-trained generative NN, these methods employ an untrained NN with random input seed and adjust the NN weights to fit the measurements. Bostan *et al.* [49] and Lawrence *et al.* [50] exploited the untrained deep decoder [52] to solve specific PR problems. Also based on the untrained deep decoder, the theoretical work by Jagatap *et al.* [22] presented a projected gradient descent scheme with convergence guarantees. Briefly, the deep decoder is a heavily under-parameterized NN with only 1×1 convolutions. While such an under-parameterized model induces a strong prior which prefers regular structure patterns over random noise, it lacks the capability to accurately represent complex patterns such as textures. In Sun *et al.* [53], the untrained normalizing flows rather than the deep decoder are used for solving non-linear inverse imaging problems, which allows uncertainty quantization on the result. See also Liu *et al.* [51] for an untrained-NN-based framework for general inverse problems.

III. Proposed Method

This section presents the proposed method in details. Following the notations in Section I, we use $\boldsymbol{y}_0, \boldsymbol{x}_0$ to denote the phase-less measurement data and the latent image respectively.

A. Model Training for Approximating MMSE Estimation

Given $\boldsymbol{y}_0$, we train a generative NN so that it approximates the MMSE estimate defined by

$$\begin{aligned} \widehat{\boldsymbol{x}} &= \operatorname*{argmin}_{\boldsymbol{u}(\boldsymbol{y}_0)} \mathbb{E}_{(\boldsymbol{x}_0|\boldsymbol{y}_0)} \|\boldsymbol{u}(\boldsymbol{y}_0) - \boldsymbol{x}_0\|_2^2 \\ &= \mathbb{E}_{(\boldsymbol{x}_0|\boldsymbol{y}_0)}(\boldsymbol{x}_0|\boldsymbol{y}_0) = \textstyle\int \boldsymbol{x}_0 p(\boldsymbol{x}_0|\boldsymbol{y}_0) d\boldsymbol{x}_0, \end{aligned} \tag{10}$$

where $p(\boldsymbol{x}_0|\boldsymbol{y}_0)$ is the posterior probability density function (PDF) of the truth image $\boldsymbol{x}_0$. That is, the MMSE estimator is obtained as the posterior mean of $\boldsymbol{x}_0$. In this paper, we assume that the image random variable $\boldsymbol{x}_0$ can be re-parametrized by a CNN $\mathcal{G}_{\boldsymbol{\mu}}(\boldsymbol{\epsilon}_0)$ with the weight variable $\boldsymbol{\mu}$ and a fixed seed $\boldsymbol{\epsilon}_0$, *i.e.*, $\boldsymbol{x}_0 = \mathcal{G}_{\boldsymbol{\mu}}(\boldsymbol{\epsilon}_0)$. So the MMSE estimate $\widehat{\boldsymbol{x}}$ can be re-parametrized as follows:

$$\widehat{\boldsymbol{x}} = \int \boldsymbol{x}_0 p(\boldsymbol{x}_0|\boldsymbol{y}_0) d\boldsymbol{x}_0 = \int \mathcal{G}_{\boldsymbol{\mu}}(\boldsymbol{\epsilon}_0) p(\boldsymbol{\mu}|\boldsymbol{y}_0) d\boldsymbol{\mu}. \tag{11}$$

Then, $\widehat{\boldsymbol{x}}$ can be calculated via the Monte-Carlo integration if we have the posterior distribution $p(\boldsymbol{\mu}|\boldsymbol{y}_0)$ in (11). However, $p(\boldsymbol{\mu}|\boldsymbol{y}_0)$ is in general computationally intractable due to the high dimensionality of $\boldsymbol{\mu}$.

In this paper, we take a variational inference approach [54] to approximate $p(\boldsymbol{\mu}|\boldsymbol{y}_0)$ by a surrogate distribution $q(\boldsymbol{\mu}|\boldsymbol{\alpha})$ parameterized by $\boldsymbol{\alpha}$. Considering computational efficiency, we propose the following model of $q(\boldsymbol{\mu}|\boldsymbol{\alpha})$ to approximate $p(\boldsymbol{\mu}|\boldsymbol{y}_0)$:

$$q(\boldsymbol{\mu}|\boldsymbol{\alpha}) : \boldsymbol{\mu} = \boldsymbol{\alpha} \odot \boldsymbol{d}, \ \boldsymbol{d}(j) \sim \mathcal{B}(q_j), \forall j, \tag{12}$$

where $\mathcal{B}(s)$ denotes the Bernoulli distribution with probability s, and $\odot$ denotes element-wise multiplication. The Bernoulli sampling in (12) equals to running dropout on $\boldsymbol{\mu}$, which enjoys both high computational efficiency and certain regularization property. Approximating $p(\boldsymbol{\mu}|\boldsymbol{y}_0)$ with $q(\boldsymbol{\mu}|\boldsymbol{\alpha})$ is about estimating $\boldsymbol{\alpha}$, which is done by minimizing their KL-divergence:

$$\min_{\boldsymbol{\alpha}} \ \mathrm{KL}(q(\boldsymbol{\mu}|\boldsymbol{\alpha}) || p(\boldsymbol{\mu}|\boldsymbol{y}_0)). \tag{13}$$

It is equivalent to

$$\min_{\boldsymbol{\alpha}} \ \mathrm{KL}(q(\boldsymbol{\mu}|\boldsymbol{\alpha}) || p(\boldsymbol{\mu})) - \mathbb{E}_{\boldsymbol{\mu} \sim q(\boldsymbol{\mu}|\boldsymbol{\alpha})} \log p(\boldsymbol{y}_0|\boldsymbol{\mu}), \tag{14}$$

where $p(\boldsymbol{\mu})$ is the prior distribution of the network parameters $\boldsymbol{\mu}$. Suppose $p(\boldsymbol{\mu})$ is a uniform distribution on a sufficiently large bounded set $\mathbb{U}$. Let $\delta_{\mathbb{U}}(\boldsymbol{\alpha}) = +\infty$ if $\boldsymbol{\alpha} \notin \mathbb{U}$ and 0 otherwise. Then $\mathrm{KL}(q(\boldsymbol{\mu}|\boldsymbol{\alpha}) || p(\boldsymbol{\mu}))$ is equivalent to $\delta_{\mathbb{U}}(\boldsymbol{\alpha})$ up to a constant. Therefore, we can rewrite (13) as

$$\max_{\boldsymbol{\alpha} \in \mathbb{U}} \ \mathbb{E}_{\boldsymbol{\mu} \sim q(\boldsymbol{\mu}|\boldsymbol{\alpha})} \log p(\boldsymbol{y}_0|\boldsymbol{\mu}). \tag{15}$$

The constraint $\boldsymbol{\alpha} \in \mathbb{U}$ can be omitted in practice, due to the feasible set $\mathbb{U}$ is sufficiently large. Considering the measurement noise $\boldsymbol{n}$ to be *i.i.d.* Gaussian white noise, we have $\log p(\boldsymbol{y}_0|\boldsymbol{\mu}) \propto -\|\boldsymbol{y}_0 - |\boldsymbol{A}\mathcal{G}_{\boldsymbol{\mu}}(\boldsymbol{\epsilon}_0)|\|_2^2$. Finally, we obtain the trained model by solving

$$\min_{\boldsymbol{\alpha}} \ \mathbb{E}_{\boldsymbol{d} \sim \mathcal{B}(\boldsymbol{q})} \big\|\boldsymbol{y}_0 - |\boldsymbol{A}\mathcal{G}_{\boldsymbol{\alpha} \odot \boldsymbol{d}}(\boldsymbol{\epsilon}_0)|\big\|_2^2. \tag{16}$$

Owing to space limitation, more details are referred to our supplementary materials.

The training loss in (16) is equivalent to introducing dropout to the convolutional layers of the NN. Then the model training can be implemented by training the NN with dropout. Recall that dropout refers to randomly dropping out some nodes of an NN during training. It enables a single NN to approximate numerous NNs of different structures in parallel during training. In other words, instead of solving (16) multiple times, we can learn many NN instances in parallel with shared weights but different dropouts. The training procedure is stopped if the maximum iteration number is reached or the residual $\|\boldsymbol{y}_0 - |\boldsymbol{A}\mathcal{G}_{\boldsymbol{\alpha} \odot \boldsymbol{d}}(\boldsymbol{\epsilon}_0)|\|_2^2 / R$ is less than a pre-defined tolerance defined based on the noise level.

B. Reconstruction via Model Averaging

Once the model training is done with the optimal parameter $\boldsymbol{\alpha}^\star$, we obtain an approximate distribution $q(\boldsymbol{\mu}|\boldsymbol{\alpha}^\star)$ to the desired posterior distribution $p(\boldsymbol{\mu}|\boldsymbol{y}_0)$. The approximate MMSE estimate to $\boldsymbol{x}^\star$ is then given by

$$\boldsymbol{x}^\star = \int \mathcal{G}_{\boldsymbol{\mu}}(\boldsymbol{\epsilon}_0) q(\boldsymbol{\mu}|\boldsymbol{\alpha}^\star) d\boldsymbol{\mu}. \tag{17}$$

The above integration is still intractable, whereas it can be calculated using Monte Carlo integration:

$$\boldsymbol{x}^\star = \frac{1}{K}\sum_{k=1}^{K} \mathcal{G}_{\boldsymbol{\mu}^{(k)}}(\boldsymbol{\epsilon}_0),\ \boldsymbol{\mu}^{(k)} \sim q(\boldsymbol{\mu}|\boldsymbol{\alpha}^\star), \tag{18}$$

with a sufficiently-large sampling number K. In other words, the reconstruction is done by averaging the prediction results from K instances of the dropout-trained NN. Concretely, we use the trained NN to infer K times, with dropout enabled. Then, at each inference an instance of the NN with dropout is generated for prediction. Since the model has been sufficiently trained with dropout, those instances are very likely to provide different effective estimators for averaging.

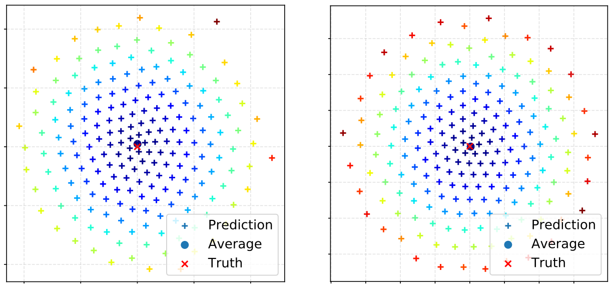


Fig. 1: Effectiveness of the model averaging prior used in the proposed method for PR. A dropout model is trained with the proposed scheme on the noisy amplitude measurements of two images "Boat" (left) and "Barbara" (right) respectively. Then 200 images are reconstructed by 200 tests with dropout from the dropout-trained model. We use t-SNE to visualize the 200 reconstructed images and their average as well as the ground-truth. As can be seen, the reconstructed images are randomly scattered around the ground-truth, and their average is closer to the ground-truth than each of them.

Model averaging prior for PR. As mentioned before, the model-averaging-based reconstruction process defined by (17) is an approximation to MMSE estimate. The concern is then whether such an approximation can effectively handle the solution ambiguity and the sensitivity to noise. To answer this, we trained the models by the proposed method given the magnitude measurements of two natural images respectively. For each image, 200 reconstructions are generated via realizing the trained dropout model for prediction by 200 times. These 200 reconstructions and their average, as well as the ground-truth image, are projected into a 2D space via t-SNE, as shown in Fig. 1. We can find that the reconstructions are randomly distributed around the ground-truth, and their average is closer to the truth image than any of them. Such results imply that $p(\boldsymbol{\mu}|\boldsymbol{y}_0)$ is well approximated by our dropout training and the model averaging can increase the prediction accuracy.

C. Network Implementation

We adopt the popular U-shape convolutional architecture to construct our NN. See Fig. 2 for an illustration. To reconstruct an image of size $H \times W \times C$, we use a random seed of the same size (*i.e.* $\boldsymbol{\epsilon}_0 \in \mathbb{R}^{H\times W\times C}$) as input, that is

$$\mathcal{G}:\ \mathbb{R}^{H\times W\times C} \to \mathbb{R}^{H\times W\times C}. \tag{19}$$

The input seed is generated with its entries independently drawn from the normal distribution $\mathcal{N}(0, 0.1)$, and then is passed to the network as the input. The CNN used in our method contains six encoder blocks and five decoder blocks. For the encoder blocks, the first block consists of two C_1-channel convolutional layers. Each of the rest of blocks sequentially connects a max pooling layer with a window of size 2×2 and stride of 2, and a C_1-channel convolutional layer. For the decoder blocks, each of the first four blocks sequentially connects an upsampling layer with factor of 2 and two C_2-channel convolutional layers. The last block contains an upsampling layer with factor of 2, as well as three convolutional layers whose numbers of output channels are C_3, C_4, C respectively. All the convolutional layers are equipped with the leaky ReLU activation, except that the last one which is equipped with the sigmoid activation. The bi-linear interpolation is used on all upsampling layers.

There are in total 18 convolutional layers in our CNN. We configure dropout on each of these layers. The dropout probabilities $\boldsymbol{q}$ on the layers are set with an increasing/decreasing order on the encoder/decoder part. Let $q_j (j = 1, \cdots, 18)$ denote the dropout probability on the j-th convolutional layer and q_0 denote the initial point. We set them using the following symmetric rule: $q_2 = q_1 = q_0$, $q_j = q_{j-1} + 0.075$ for $j = 3, \cdots, 8$, $q_9 = q_8$, $q_j = q_{j-2} - 0.075$ for $j = 10, \cdots, 16$, $q_{18} = q_{17} = q_{16}$. The computational flow for a convolutional layer with dropout is as follows: $\boldsymbol{d} \sim \mathcal{B}(q_j)$, $\boldsymbol{o} = \boldsymbol{w} \otimes (\boldsymbol{d} \odot \boldsymbol{z}) + \boldsymbol{b}$ where $\boldsymbol{o}, \boldsymbol{z}, \boldsymbol{w}, \boldsymbol{b}$ denote the output, input, convolution kernel and bias respectively, and $\otimes$ denotes the convolution operation.

D. Augmentation and Algorithm

Viewing model averaging as an ensemble method, one of its key is increasing the diversity among the model's predictions. Thus, in addition to the utilization of dropout layers, we introduce more randomness via adding a random soft mask in the training loss function as follows:

$$\mathcal{L}_{\text{aug}}(\boldsymbol{\alpha}) := \mathbb{E}_{\boldsymbol{\omega}}\mathbb{E}_{\boldsymbol{d}}\left\|\frac{\boldsymbol{\omega}}{\|\boldsymbol{\omega}\|_1} \odot (\boldsymbol{y}_0 - |\boldsymbol{A}\mathcal{G}_{\boldsymbol{\alpha}\odot\boldsymbol{d}}(\boldsymbol{\epsilon}_0)|)\right\|_2^2, \tag{20}$$

where each entry of $\boldsymbol{\omega}$ is drawn from the normal distribution $\mathcal{N}(0.5, 0.1)$. During training, the soft random masks $\boldsymbol{\omega}$ enforce the model to focus on different spatial locations for reconstruction with varying weights at each iteration. Note that the squared ℓ_2 loss is not optimal for MAP and MMSE in the presence of Poisson noise. As the randomized loss $\mathcal{L}_{\text{aug}}$ can be viewed as a re-weighted squared ℓ_2 loss, it might perform better in the presence of signal-dependent noise, *e.g.* Poisson noise, than the standard squared ℓ_2 loss. The experiments show that the introduction of the soft random masks brings slight performance gain indeed.

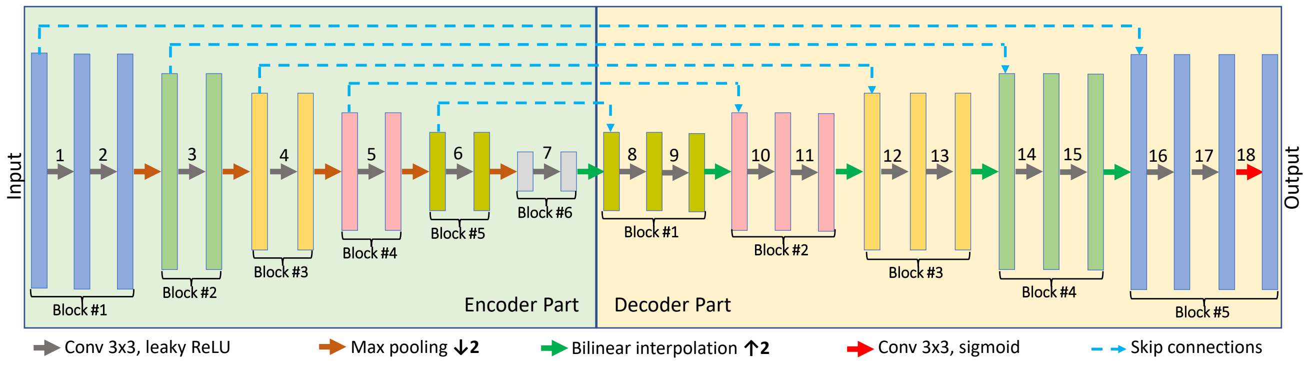


Fig. 2: Diagram of structure of the CNN used in the proposed method. There are in total 18 convolutional layers in the model, all of which are equipped with dropout.

To conclude, we present an untrained-NN-based method for PR which can be viewed as a deep approximate MMSE estimator, and we call it DeepMMSE through the remaining of the paper. See Algorithm 1 for its complete description.

Algorithm 1: DeepMMSE for PR

```
   Input: $\boldsymbol{y}, \boldsymbol{A}, \mathcal{G}, \boldsymbol{\alpha}^0, \boldsymbol{q}, K$
   Output: $\boldsymbol{\alpha}^\star, \boldsymbol{x}^\star$
1  Draw $\boldsymbol{\epsilon}_0 \sim \mathcal{N}(0, 0.1)$;
2  $\boldsymbol{\alpha} = \boldsymbol{\alpha}^0$;
3  repeat
4      Draw $\boldsymbol{\omega} \sim \mathcal{N}(0.5, 0.1)$;
5      Draw $\boldsymbol{d} \sim \mathcal{B}(\boldsymbol{q})$;
6      $\mathcal{L}_{\text{aug}} = \|\frac{\boldsymbol{\omega}}{\|\boldsymbol{\omega}\|_1} \odot (\boldsymbol{y}_0 - |\boldsymbol{A}\mathcal{G}_{\boldsymbol{\alpha}\odot\boldsymbol{d}}(\boldsymbol{\epsilon}_0)|)\|_2^2$;
7      update $\boldsymbol{\alpha}$ via some optimizer on $\mathcal{L}_{\text{aug}}$;
8  until iternaion number or tolerance reaches threshold;
9  $\boldsymbol{\alpha}^\star = \boldsymbol{\alpha}, \boldsymbol{x} = \boldsymbol{0}$;
10 for $k = 1, 2, \cdots, K$ do
11     Draw $\boldsymbol{d} \sim \mathcal{B}(\boldsymbol{q})$;
12     $\boldsymbol{x} = \boldsymbol{x} + \mathcal{G}_{\boldsymbol{\alpha}^\star\odot\boldsymbol{d}}(\boldsymbol{\epsilon}_0)$;
13 end
14 $\boldsymbol{x}^\star = \boldsymbol{x}/K$;
```

IV. Experimental Evaluation

The proposed DeepMMSE is evaluated with several PR settings, which includes PR from CDPs, compressive PR from Gaussian measurements and Fourier PR. Specifically, bipolar masks and uniform masks are used as the CDP masks respectively, and additive white Gaussian noise (AWGN) and Poisson noise are considered as the measurement noise respectively. An ablation study is also conducted.

A. Implementation Details of Proposed Method

We implemented the proposed method using PyTorch with parallel computation. Throughout all the experiments, unless specified, the setting of the proposed method is as follows. For all convolution layers, the kernel size is set to 3×3, and both the stride and padding number are set to 1. The negative slope is fixed to 0.01 for all leaky ReLUs. All convolution weights are initialized by Xavier initialization scheme [55], and all biases are initialized to zeros. The model is trained by the Adam optimizer [56] with learning rate fixed at 10^{-4}. The maximum iteration number is set to 10^5 and the residual tolerance is set to the maximum between noise variance and 10^{-5}. The sampling number K used in the model averaging of (18) is set to 40. Our method requires no external training data but the input measurements themselves. Recall that the number of CDP masks (*i.e.* R) varies in different scenarios. To better deal with the varying value of R, we allow the CNN to be scalable with the number of measurements, which is done by setting the number of CNN's channels to be proportional to the number of CDP masks using the following rule:

$$C_1 = \lceil 24\sqrt{R}\rceil, C_2 = \lceil 48\sqrt{R}\rceil, C_3 = \lceil 32\sqrt{R}\rceil, C_4 = \lceil 16\sqrt{R}\rceil.$$

The implementation using PyTorch v1.7 will be available to the public at https://github.com/AlanLin1995.

B. Bipolar Masks + AWGN

Bipolar masks are one often-seen type of CDP masks in realistic experiments due to its easy implementation. A bipolar mask $\boldsymbol{D}$ has its entries (1 or -1) drawn from a Bernoulli distribution $\mathcal{B}(1/2)$. In this experiment, the AWGN is added to the clean measurements generated by (2) with a series of bipolar masks $\boldsymbol{D}_r (r = 1, ..., R)$ to simulate the practical noisy observations. We follow the simulation scheme from [19]. The strength (amount) of the AWGN is measured by the signal-to-noise ratio (SNR) in dB of the input measurements, which is defined by $\text{SNR} = 10\log_{10}(\|\boldsymbol{y}_0 - \boldsymbol{n}\|_2^2/\|\boldsymbol{n}\|_2^2)$. Concretely, the clean measurements are corrupted by the AWGN such that the SNRs of input noisy measurements are $10\text{dB}, 15\text{dB}, 20\text{dB}$ respectively. Accordingly, the initial dropout probability q_0 in our CNN is set to $10\%, 20\%, 30\%$ respectively. The number of bipolar masks is set to $R = 1, 2, 3, 4$ respectively.

Three datasets are used for the test, including (a) Natural-6 that contains six "natural" images used in [18]; (b) Unnatural-6 that contains six "unnatural" images (*e.g.* fractal images) used in [18]; and (c) Set-20 that contains twenty classic "natural" images shown in supplementary materials. Such three datasets cover a wide range of image content for evaluation. Following the scheme of [18], we resize all images to 128×128. The approaches selected for the performance comparison include

WF [32], DOLPHIn [42], BM3D-prGAMP [10], ConPR [17], prDeep [18] and DPSR [19]. Specially, WF is a flow-based method which has become a common baseline for PR. DOLPHIn is a sparsity-regularization-based method. BM3D-prGAMP and ConPR are plug-and-play regularization methods which exploit the patch recurrence prior of a natural image via BM3D. DPSR and prDeep are plug-and-play methods.

Recall that DeepMMSE leverages untrained NN priors. To demonstrate its advantages, we also adopt two deep untrained-NN-based methods for comparison, including DIP [23] and Net-PGD [22]. To avoid overfitting, DIP uses early stopping to regularize the CNN training while Net-PGD employs an under-parameterized NN. We implement a DIP-based PR method by using the same cost function as (8), *i.e.* $\min_{\boldsymbol{\mu}} \||\boldsymbol{A}\mathcal{G}_{\boldsymbol{\mu}}(\boldsymbol{\epsilon}_0)| - \boldsymbol{y}_0\|_2^2/R$. Specially, the CNN used in the DIP-based PR method is also the same as the original one for super-resolution, *i.e.* an encoder-decoder CNN with skip connections. The maximum iteration number is set to 2×10^4 while it is early stopped if the residual $\||\boldsymbol{A}\mathcal{G}_{\boldsymbol{\mu}}(\boldsymbol{\epsilon}_0)| - \boldsymbol{y}_0\|_2^2/R$ reaches the same tolerance as ours. The input of DIP is jitterred at each iteration, as its original work did.

The quantitative results in terms of PSNR (peak signal-to-noise ratio) and SSIM (structural similarity index measure) are listed in Table I for comparison. For the compared methods, we quote their results directly from the literature if possible. Otherwise, we follow the instructions to run their methods using the codes provided by the authors and make our effort to tune the parameters to obtain the optimal results. It can be seen from Table I that DeepMMSE consistently outperforms all others by a large margin on all Gaussian noise levels across all datasets, in terms of both criteria. The overall average PSNR gain of DeepMMSE over the second best performers is about 1.93dB, 1.45dB and 1.56dB for SNR $= 10$dB, 15dB, 20dB respectively. Such significant quantitative improvement is also consistent with the improvement on visual quality. In Fig. 3, we compare the visual reconstruction results of image "Butterfly" from different approaches. See supplementary materials for more visual comparison. We can observe that the results of DeepMMSE contain more image details than other approaches and are the most consistent with the original images. The superior performance of DeepMMSE is mainly attributed to that the model averaging prior shown in Fig. 1 is effective to handle overfitting which leads to accurate estimate. It can also be seen that DeepMMSE outperforms DIP and Net-PGD by a significant margin in all configurations. This demonstrates the advantages of the model averaging prior over early stopping and under-parameterization for avoiding overfitting.

C. Bipolar Masks + Poisson Noise

Poisson shot noise is one of the dominant noise sources in many PR-related applications. In this experiment, Poisson noise instead of AWGN is added to the clean measurements generated by bipolar masks to simulate the practical observations. Following the scheme of [18], the Poisson noise is simulated as follows:

$$|\boldsymbol{y}_0|^2/\gamma^2 \sim \text{Poisson}(|\boldsymbol{A}\boldsymbol{x}_0|^2/\gamma^2), \tag{21}$$

where the parameter γ controls the noise strength and is set to $9, 27, 81$ in the experiment. Smaller (larger) γ indicates lower (higher) noise strength. The initial probability q_0 of dropout is set to $10\%, 30\%, 50\%$ accordingly.

The compared methods in the previous experiment are also selected for comparison in this experiment. The quantitative results are listed in Table II for comparison. Our DeepMMSE consistently outperforms all others by a large margin on almost all noise strengths across all datasets, in terms of both PSNR and SSIM. The overall average PSNR gain of DeepMMSE over the second best ones is about 0.78dB, 0.45dB, and 0.27dB for $\gamma = 9, 27, 81$ respectively. Such significant quantitative improvement is also consistent with the improvement on visual quality. In Fig. 4, we compare the reconstruction results of image "Pollen" from different approaches. Visually, it can be found that the result of DeepMMSE contains more realistic details than that of other methods. Furthermore, ours is the most consistent with the original image among the nine compared results. See also supplementary materials for more visual comparison. The superior performance of DeepMMSE clearly indicates the capability of the proposed method for handling non-Gaussian noise in PR-based image reconstruction.

D. Uniform CDPs + Poisson Noise

To see how well DeepMMSE performs on other types of CDPs instead of that from bipolar masks, in this experiment, we employ (2) with uniform CDP masks to generate the clean measurements, followed by contamination of the Poisson noise. The uniform CDP masks $\boldsymbol{D}_i\,(i = 1, ..., R)$ are defined as the diagonal matrices with non-zero elements drawn uniformly from the unit circle in the complex plane. We follow the same experimental protocol presented in [57]. The experiments are conducted on 12 gray-scale images with Poisson noise of three different strengths and the number of CDP masks is set to $R = 4$. We use the same model setting as that used in Section IV-C.

Seven representative methods are included for comparison, including HIO [27], WF [32], DOLPHIn [42], SPAR [15] BM3D-prGAMP [10], prDeep [18] and TFPnP [57]. The quantitative results are listed in Table V. It can be seen DeepMMSE outperforms others at the low and moderate noise levels. This has indicated the effectiveness of DeepMMSE in dealing with other types of CDPs. At the high noise level, DeepMMSE is the second-best performer and its result is only slightly worse than the best one, TFPnP. In comparison to TFPnP which needs a large dataset that covers considerable image patterns relative to the task for training an effective denoising CNN, DeepMMSE only uses the measurement themselves to recover the images. In addition, TFPnP needs to train different models for different types of CDPs. In comparison, DeepMMSE can handle different types of CDPs directly.

Computational time. For a deep-NN-based method, it is difficult to have a theoretical computational complexity. For the empirical computational complexity, the running time of DeepMMSE in the setting of bipolar/uniform masks with Poisson noise are listed in Table III. The test environment includes a Intel Core i7-8700 CPU, DDR4-2666MHz 16GB

TABLE I: Average PSNR(dB)/SSIM values of PR-based reconstruction using different numbers of bipolar masks in the presence of different levels of AWGN. The best and second results are marked in **blue** and green respectively.

	Dataset	Natural-6			Unnatural-6			Set-20		
	SNR(dB)	10	15	20	10	15	20	10	15	20
R=1	WF	11.8/0.100	12.1/0.106	12.2/0.110	15.0/0.119	15.3/0.121	15.4/0.131	11.5/0.120	11.9/0.133	12.0/0.135
	DOLPHIn	19.3/0.394	22.3/0.561	24.8/0.683	22.3/0.512	24.3/0.634	24.5/0.652	18.0/0.367	20.3/0.494	22.6/0.613
	BM3D-prGAMP	23.1/0.680	25.1/0.740	27.6/0.809	27.4/0.806	29.6/0.851	31.9/0.888	22.3/0.655	24.6/0.733	26.6/0.797
	ConPR	23.1/0.664	25.6/0.762	27.2/0.822	25.5/0.710	27.3/0.772	29.4/0.834	22.3/0.660	24.6/0.751	26.3/0.817
	prDeep	21.0/0.563	25.6/0.763	27.7/0.835	27.6/0.770	29.8/0.820	32.3/0.902	19.0/0.500	24.5/0.723	26.4/0.815
	DPSR	23.6/0.670	25.8/0.760	27.9/0.822	27.7/0.801	30.1/0.866	32.1/0.899	22.3/0.624	24.5/0.731	26.4/0.801
	DIP	23.7/0.653	26.1/0.754	28.6/0.842	28.3/0.816	30.8/0.816	33.2/0.916	23.0/0.664	25.4/0.759	27.9/0.844
	Net-PGD	23.0/0.668	25.4/0.753	27.4/0.819	27.5/0.804	29.6/0.857	31.5/0.896	22.4/0.655	24.5/0.756	26.6/0.816
	DeepMMSE	**24.9/0.710**	**27.3/0.797**	**29.5/0.860**	**29.5/0.846**	**31.9/0.897**	**34.4/0.933**	**24.2/0.706**	**26.7/0.801**	**29.0/0.865**
R=2	WF	12.6/0.169	14.3/0.221	15.5/0.274	16.5/0.198	17.7/0.234	17.9/0.247	12.5/0.208	13.9/0.263	14.6/0.288
	DOLPHIn	21.5/0.506	25.0/0.692	27.5/0.797	24.0/0.526	26.4/0.668	29.5/0.818	19.9/0.485	23.2/0.654	25.5/0.742
	BM3D-prGAMP	24.5/0.737	27.1/0.802	29.6/0.861	29.0/0.851	31.7/0.898	34.4/0.932	23.8/0.720	26.4/0.798	28.9/0.862
	ConPR	25.6/0.768	28.5/0.846	31.2/0.903	28.6/0.828	30.7/0.873	32.9/0.904	24.7/0.766	27.6/0.851	30.2/0.906
	prDeep	24.9/0.736	27.7/0.827	30.2/0.889	29.5/0.823	31.2/0.850	34.2/0.929	23.8/0.723	26.7/0.824	29.3/0.888
	DPSR	25.4/0.749	28.4/0.843	30.9/0.896	30.2/0.874	32.6/0.918	34.9/0.940	24.5/0.744	27.2/0.831	29.5/0.887
	DIP	25.5/0.741	28.0/0.828	30.7/0.894	30.2/0.869	32.7/0.926	35.4/0.946	24.3/0.763	27.4/0.832	30.2/0.898
	Net-PGD	24.8/0.769	27.5/0.828	29.7/0.878	29.2/0.848	31.5/0.899	33.4/0.929	24.2/0.740	26.6/0.827	29.3/0.887
	DeepMMSE	**26.7/0.776**	**29.3/0.855**	**32.1/0.911**	**31.4/0.882**	**34.3/0.932**	**37.2/0.957**	**26.1/0.787**	**28.9/0.863**	**31.8/0.919**
R=3	WF	15.4/0.275	19.6/0.449	22.2/0.557	20.7/0.369	24.2/0.495	28.5/0.644	15.3/0.329	19.0/0.475	22.5/0.617
	DOLPHIn	23.7/0.636	26.8/0.785	30.1/0.868	24.7/0.578	26.7/0.702	29.8/0.834	22.2/0.605	25.7/0.769	29.0/0.858
	BM3D-prGAMP	25.3/0.767	28.2/0.833	30.9/0.890	30.0/0.875	32.8/0.917	35.8/0.944	24.5/0.756	27.4/0.831	30.2/0.890
	ConPR	26.8/0.808	29.7/0.883	32.9/0.931	29.4/0.854	31.8/0.893	33.9/0.910	25.7/0.802	28.8/0.880	31.8/0.930
	prDeep	26.1/0.787	28.7/0.853	31.3/0.909	30.2/0.837	32.1/0.867	35.4/0.941	25.0/0.777	27.8/0.851	30.5/0.910
	DPSR	26.6/0.808	29.7/0.877	32.5/0.925	30.6/0.895	34.2/0.936	36.9/0.959	25.7/0.802	28.6/0.873	31.2/0.918
	DIP	26.5/0.781	29.3/0.861	32.1/0.919	31.3/0.890	34.0/0.931	32.3/0.801	25.9/0.789	28.6/0.867	31.6/0.924
	Net-PGD	25.9/0.783	28.5/0.856	31.6/0.916	30.3/0.875	32.1/0.913	34.5/0.942	25.3/0.782	27.9/0.861	30.7/0.912
	DeepMMSE	**27.9/0.816**	**30.7/0.886**	**33.6/0.934**	**32.6/0.900**	**35.7/0.949**	**38.3/0.970**	**27.3/0.822**	**30.2/0.891**	**33.2/0.939**
R=4	WF	18.1/0.391	23.0/0.595	28.0/0.770	23.7/0.480	28.6/0.654	33.5/0.801	17.7/0.426	22.5/0.616	27.2/0.773
	DOLPHIn	24.9/0.689	28.1/0.811	31.3/0.899	25.4/0.624	27.4/0.742	30.0/0.843	23.4/0.659	26.9/0.795	30.4/0.892
	BM3D-prGAMP	26.0/0.789	28.9/0.854	31.7/0.904	30.6/0.889	33.6/0.928	36.7/0.953	25.2/0.783	28.2/0.852	31.1/0.905
	ConPR	27.4/0.829	30.7/0.901	33.8/0.944	30.0/0.866	32.1/0.899	34.3/0.915	26.3/0.824	29.8/0.898	32.8/0.942
	prDeep	26.8/0.808	29.3/0.866	32.2/0.922	30.9/0.858	32.7/0.876	36.2/0.948	26.0/0.809	28.5/0.865	31.4/0.922
	DPSR	26.9/0.822	30.2/0.898	33.3/0.939	31.0/0.909	34.8/0.948	37.7/0.966	25.9/0.820	29.3/0.895	32.2/0.935
	DIP	27.4/0.813	30.1/0.885	33.2/0.935	32.2/0.907	35.2/0.947	38.4/0.970	26.7/0.817	29.6/0.888	32.7/0.938
	Net-PGD	26.6/0.803	29.1/0.869	32.4/0.928	31.1/0.897	34.0/0.935	36.4/0.958	26.0/0.806	28.8/0.880	31.7/0.929
	DeepMMSE	**28.7/0.841**	**31.7/0.904**	**34.7/0.947**	**33.6/0.925**	**36.7/0.957**	**39.5/0.971**	**28.1/0.846**	**31.2/0.910**	**34.4/0.951**

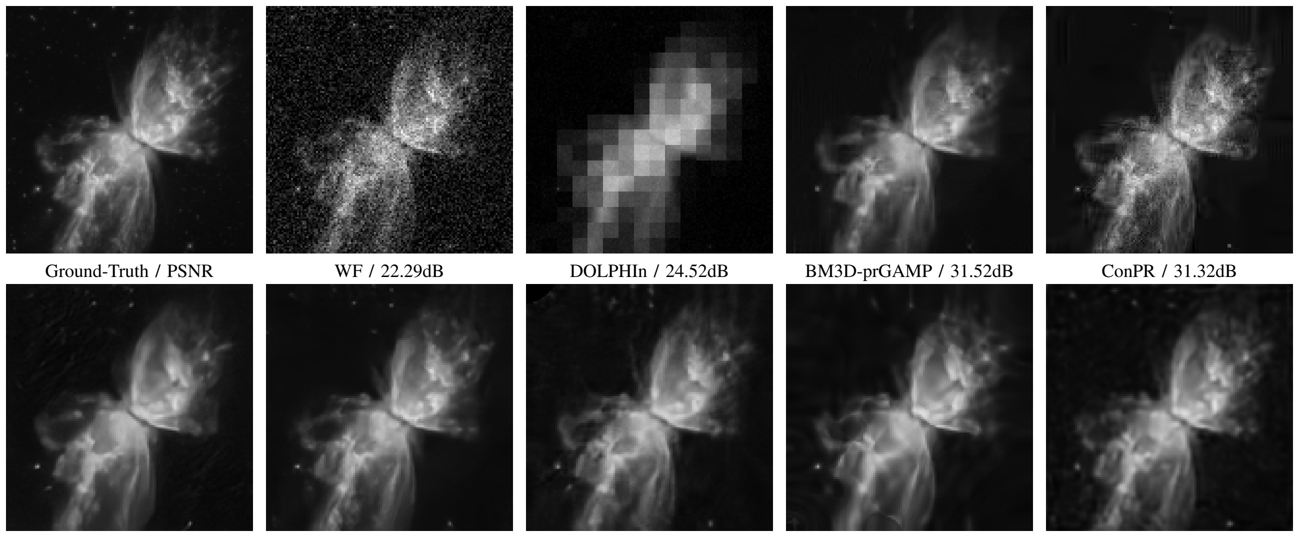


Fig. 3: Reconstructions on image “Butterfly” in the presence of AWGN (SNR=10dB) with $R = 3$ bipolar masks.

TABLE II: Average PSNR(dB)/SSIM values of PR-based reconstruction using different numbers of bipolar masks in the presence of Poisson noise with different strengths. The best and second results are marked in **blue** and green respectively.

	Dataset	Natural-6			Unnatural-6			Set-20		
	γ	9	27	81	9	27	81	9	27	81
R=1	WF	12.7/0.126	12.6/0.123	12.4/0.119	15.9/0.147	15.5/0.130	14.4/0.100	12.3/0.142	12.2/0.137	12.0/0.135
	DOLPHIn	24.9/0.697	22.8/0.585	17.6/0.316	24.4/0.653	23.8/0.597	19.3/0.346	24.0/0.677	21.5/0.555	17.1/0.326
	BM3D-prGAMP	32.7/0.913	28.0/0.822	22.1/**0.663**	34.6/0.930	29.5/0.857	23.7/**0.740**	31.8/0.910	27.3/**0.813**	21.7/**0.649**
	ConPR	27.6/0.839	26.6/0.798	21.2/0.571	30.7/0.846	27.2/0.758	21.8/0.525	26.6/0.834	25.4/0.789	20.5/0.573
	prDeep	31.6/0.902	24.3/0.666	19.0/0.433	34.3/0.920	24.9/0.602	22.7/0.564	31.3/0.915	24.1/0.700	19.1/0.512
	DPSR	30.1/0.871	26.8/0.786	22.4/0.606	32.7/0.905	29.1/0.842	24.4/0.702	28.4/0.850	25.7/0.775	21.6/0.585
	DIP	32.3/0.920	27.1/0.796	22.0/0.571	34.4/0.932	28.8/0.821	24.3/0.660	32.1/0.927	26.8/**0.813**	21.5/0.590
	Net-PGD	31.1/0.899	26.3/0.798	21.4/0.603	33.5/0.924	28.4/0.835	23.9/0.666	30.0/0.892	25.9/0.801	21.2/0.607
	DeepMMSE	**33.6/0.934**	**28.6/0.840**	**23.0**/0.649	**35.7/0.951**	**30.3/0.865**	**24.9**/0.724	**33.4/0.940**	**27.4/0.813**	**22.0**/0.632
R=2	WF	15.3/0.270	15.0/0.252	12.5/0.163	18.5/0.271	17.7/0.229	14.5/0.129	15.1/0.304	14.9/0.295	12.5/0.209
	DOLPHIn	30.3/0.877	26.0/0.754	17.8/0.344	30.0/0.833	25.9/0.636	21.3/0.409	28.8/0.843	25.1/0.752	17.2/0.355
	BM3D-prGAMP	36.2/0.955	30.1/0.872	23.2/0.717	37.7/0.960	31.4/0.897	24.8/0.779	35.7/0.958	29.8/0.879	23.0/**0.712**
	ConPR	34.2/0.947	28.9/0.865	23.2/0.666	33.9/0.907	29.6/0.835	24.9/0.684	32.8/0.941	28.3/0.866	22.7/0.676
	prDeep	36.2/0.960	30.1/0.883	24.0/0.714	37.6/0.953	31.6/0.898	26.3/0.772	35.8/0.961	29.6/0.883	23.4/0.701
	DPSR	33.5/0.930	29.1/0.861	24.0/0.690	35.5/0.942	31.0/0.884	26.1/0.771	31.7/0.919	28.3/0.857	23.5/0.697
	DIP	34.5/0.947	28.8/0.848	23.8/0.669	33.4/0.903	28.3/0.794	24.4/0.660	34.7/0.927	28.9/0.865	**23.8**/0.695
	Net-PGD	33.5/0.937	28.3/0.849	23.4/0.681	32.2/0.896	28.2/0.815	24.1/0.688	32.6/0.934	28.0/0.853	23.3/0.688
	DeepMMSE	**36.9/0.966**	**30.8/0.892**	**24.5/0.721**	**38.5/0.968**	**32.2/0.908**	**26.5/0.779**	**36.8/0.970**	**30.4/0.894**	23.7/0.704
R=3	WF	25.3/0.679	21.0/0.504	14.2/0.235	27.8/0.613	23.2/0.438	15.8/0.185	25.7/0.720	21.2/0.557	14.2/0.289
	DOLPHIn	31.2/0.905	27.1/0.779	19.3/0.412	30.1/0.844	26.8/0.695	22.1/0.455	31.3/0.915	26.8/0.791	19.0/0.449
	BM3D-prGAMP	38.1/0.969	31.3/0.898	24.0/0.752	39.4/0.970	32.5/0.915	25.3/0.804	37.7/0.972	31.1/0.903	23.7/0.748
	ConPR	36.2/0.963	30.1/0.892	24.3/0.722	34.7/0.915	30.6/0.862	25.7/0.729	34.8/0.959	29.5/0.892	23.8/0.727
	prDeep	37.8/0.967	31.7/0.911	25.2/0.763	39.0/0.965	33.0/0.916	27.1/0.808	37.5/0.969	31.4/0.917	24.6/0.756
	DPSR	35.3/0.951	30.5/0.894	25.0/0.735	37.1/0.957	32.1/0.907	26.8/0.803	33.7/0.946	29.8/0.893	24.3/0.744
	DIP	36.2/0.964	30.1/0.883	24.7/0.716	34.7/0.925	29.3/0.828	25.0/0.697	36.5/0.969	30.2/0.897	24.7/0.743
	Net-PGD	34.1/0.946	29.3/0.875	24.3/0.721	33.4/0.916	29.1/0.845	24.9/0.702	34.2/0.952	29.4/0.886	24.1/0.733
	DeepMMSE	**38.5/0.976**	**32.1/0.917**	**25.4/0.767**	**40.0/0.975**	**33.3/0.921**	**27.3/0.815**	**38.5/0.979**	**31.8/0.921**	**24.8/0.760**
R=4	WF	34.0/0.917	24.8/0.666	15.7/0.295	34.5/0.847	25.6/0.531	17.5/0.240	34.0/0.926	24.9/0.701	15.8/0.354
	DOLPHIn	32.9/0.930	28.3/0.820	20.1/0.450	30.2/0.849	27.2/0.726	23.0/0.499	32.3/0.929	28.0/0.829	19.7/0.486
	BM3D-prGAMP	39.4/0.976	32.3/0.912	24.4/0.773	40.6/0.976	33.2/0.926	25.7/0.822	39.0/0.978	32.0/0.918	24.3/0.776
	ConPR	37.1/0.969	31.0/0.907	24.9/0.748	34.8/0.917	30.9/0.870	26.0/0.751	35.7/0.965	30.4/0.908	24.3/0.752
	prDeep	38.9/0.972	32.7/0.920	25.9/**0.796**	39.8/0.961	33.9/0.933	27.7/0.812	38.7/0.973	32.4/0.931	25.4/**0.806**
	DPSR	36.6/0.963	30.9/0.909	25.6/0.768	37.9/0.966	32.7/0.920	27.2/0.825	34.9/0.957	30.4/0.907	25.0/0.774
	DIP	37.9/0.975	31.3/0.907	24.9/0.741	39.6/0.975	32.9/0.913	27.0/0.807	37.9/0.978	31.1/0.917	24.6/0.760
	Net-PGD	36.9/0.972	30.5/0.903	24.1/0.746	38.0/0.966	32.2/0.915	26.6/0.800	36.5/0.971	30.2/0.907	24.0/0.753
	DeepMMSE	**39.7/0.982**	**32.9/0.926**	**26.0**/0.785	**41.0/0.980**	**34.2/0.935**	**27.8/0.831**	**39.7/0.983**	**32.8/0.936**	**25.5**/0.797

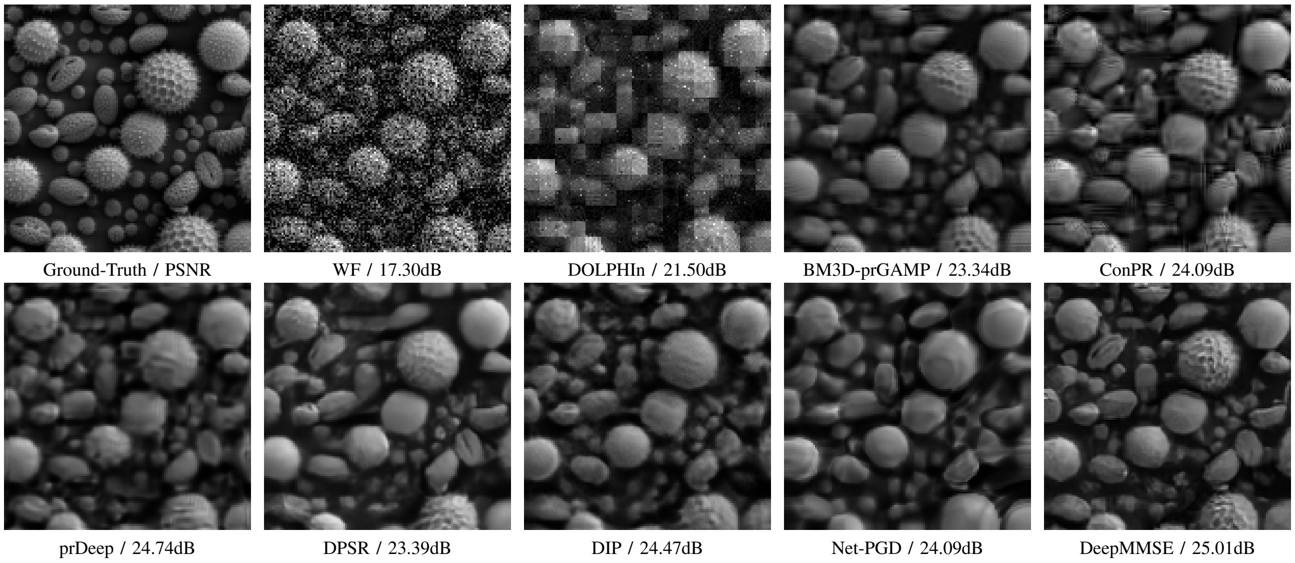


Fig. 4: Reconstructions on image "Pollen" in the presence of Poisson noise ($\gamma = 81$) with $R = 4$ uniform masks.

RAM, a single RTX 2080Ti GPU, and PyTorch v1.7. It can be seen from the table that our code takes about 4.45 minutes on average for reconstructing an image of size 128×128 using measurements from one mask. In addition, Table IV gives the comparison between DeepMMSE and several closely-related ones in terms of the following metrics: the number of parameters, the number of floating-point operations (FLOPs) within one forward pass, and the average running time on reconstructing an 128×128 image with one bipolar mask and Poisson noise ($\gamma = 81$). It can be seen that DeepMMSE is comparable to prDeep and Net-PGD in terms of the number of parameters and FLOPs. The running time of DeepMMSE is higher than other methods. The reason is that the usage of dropout in the training requires more iterations to converge, in comparison to the one without calling dropout. Since the initialization is very important to the result when solving a non-convex problem like PR, for more fairness, we restart other methods using different random initializations until the running time is comparable, then select the best results. This compensates the margin of computational cost between DeepMMSE and other methods. Note that the multiple inferences with dropout does not bring much time cost in our implementation.

TABLE III: Running time (minutes) of DeepMMSE for reconstructing an image of size 128×128 in the setting of bipolar/uniform masks with Poisson noise ($\gamma = 81$). The number of masks (*i.e.* R) varies from 1 to 4 for test.

Mask	$R=1$	$R=2$	$R=3$	$R=4$
Bipolar	4.452	5.801	7.515	9.127
Uniform	8.192	8.529	10.434	11.017

TABLE IV: Complexity comparison for reconstructing an image of size 128×128 with one bipolar mask and Poisson noise of $\gamma = 81$.

Metric	prDeep	DPSR	DIP	Net-PGD	DeepMMSE
#Parameters	667.7K	485.4K	2.2M	164.7K	247.8K
#FLOPs	10.9G	2.0G	4.8G	0.3G	0.7G
Time cost	0.51min	0.23min	1.42min	0.71min	4.45min

TABLE V: Average PSNR(dB) values of PR-based reconstructions using four uniform masks in the presence of Poisson noise with different γ. The best and second results are marked in **blue** and green respectively.

Method	$\gamma = 9$	$\gamma = 27$	$\gamma = 81$
HIO	35.96	25.76	14.82
WF	34.46	24.96	15.76
DOLPHIn	29.93	27.45	19.35
SPAR	35.20	31.82	22.44
BM3D-prGAMP	40.25	32.84	25.43
prDeep	39.70	33.54	26.82
TFPnP	40.33	33.90	**27.23**
DeepMMSE	**40.58**	**33.97**	27.12

E. Compressive PR

The DeepMMSE can be straightforwardly extended for handling compressive PR. Following the experiment on compressive PR conducted in [22], we generate the measurements using the Gaussian matrix $\boldsymbol{A} \in \mathbb{R}^{M\times N}$ whose entries are *i.i.d.* random variables drawn from the Gaussian distribution $\mathcal{N}(0, 1/M)$. The test is conducted on five RGB images from the CelebA dataset with six compression rates, *i.e.*, $M/N \in \{0.1, 0.2, 0.3, 0.5, 1, 3\}$. The noise is assumed to be negligible and the clean measurements are used as input. Since the measurements are noiseless, the learning rate of DeepMMSE is changed to 10^{-5} and the iteration number is set to 1.5×10^5. The initial probability q_0 of dropout is set to 10%.

TABLE VI: Average PSNR(dB)/SSIM values of compressive PR on CelebA dataset. The best and second results are marked in **blue** and green respectively.

M/N	DPR	Net-GD	Net-PGD	DeepMMSE
0.1	23.59/0.837	22.59/0.784	26.50/0.897	**28.65/0.938**
0.2	23.83/0.837	27.92/0.927	28.37/0.935	**31.13/0.962**
0.3	24.21/0.853	28.53/0.937	28.64/0.937	**31.96/0.967**
0.5	24.21/0.849	29.24/0.946	28.39/0.936	**31.95/0.968**
1	24.08/0.851	29.25/0.944	27.64/0.927	**32.73/0.972**
3	24.16/0.850	29.60/0.950	25.85/0.890	**32.97/0.974**

Three recent methods especially designed for compressive PR are used for comparison, including DPR [20], Net-GD [22] and Net-PGD [22]. See Table VI for the quantitative comparison in terms of PSNR and SSIM. Surprisingly, DeepMMSE outperforms other methods with a large margin across all compression rates. The results of DeepMMSE at the compression rate of 0.2 are even comparable to others at the compression rate of 1. Such results show that DeepMMSE can also be capable of dealing with compressive PR. See also Fig. 5 for some visual comparison, where the images recovered by DeepMMSE is visually better than others.

Since there is no noise in this experimental setting, the results in this part have demonstrated that the model averaging prior used in DeepMMSE can well address the influence from the compressive measurement process. In other words, the advantages of DeepMMSE are not only about handling the ambiguity arising from noise, but also about overcoming the ambiguity of the measurement system in the PR problem.

F. Fourier PR

We follow [18] to conduct an experiment on Fourier PR in the presence of Poisson noise with three different strengths, using the Natural-6 and Unnatural-6 datasets. The magnitude measurements are generated based on Fourier transform without any masking and pre-processing, and 4X sampling is done by zero-padding the latent image to four times of its original size. See [18] for the detailed description on the experimental setting. Eight representative methods are included for comparison, including WF [32], DOLPHIn [42], BM3D-prGAMP [10], ConPR [17], prDeep [18] and DPSR [19], DIP [23] and Net-PGD [22]. Following [18], the results of HIO are used for the initialization in each compared method. Specifically, we use the ℓ_1 loss between the NN's output and the HIO result for training our NN at the first 500 epochs. The same scheme is employed for DIP and Net-PGD.

The quantitative results are listed in Table VII. DeepMMSE consistently outperforms all others by a large margin on almost

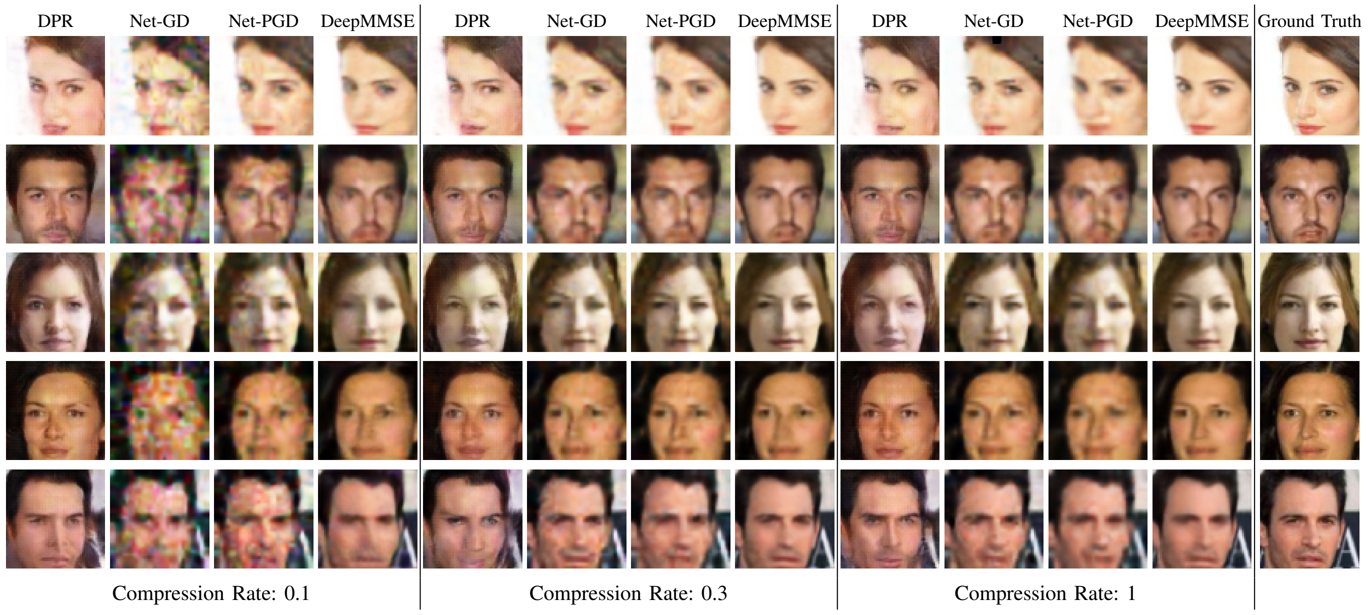


Fig. 5: Visual reconstruction results of compressive PR on CelebA dataset by different methods at different compression rates.

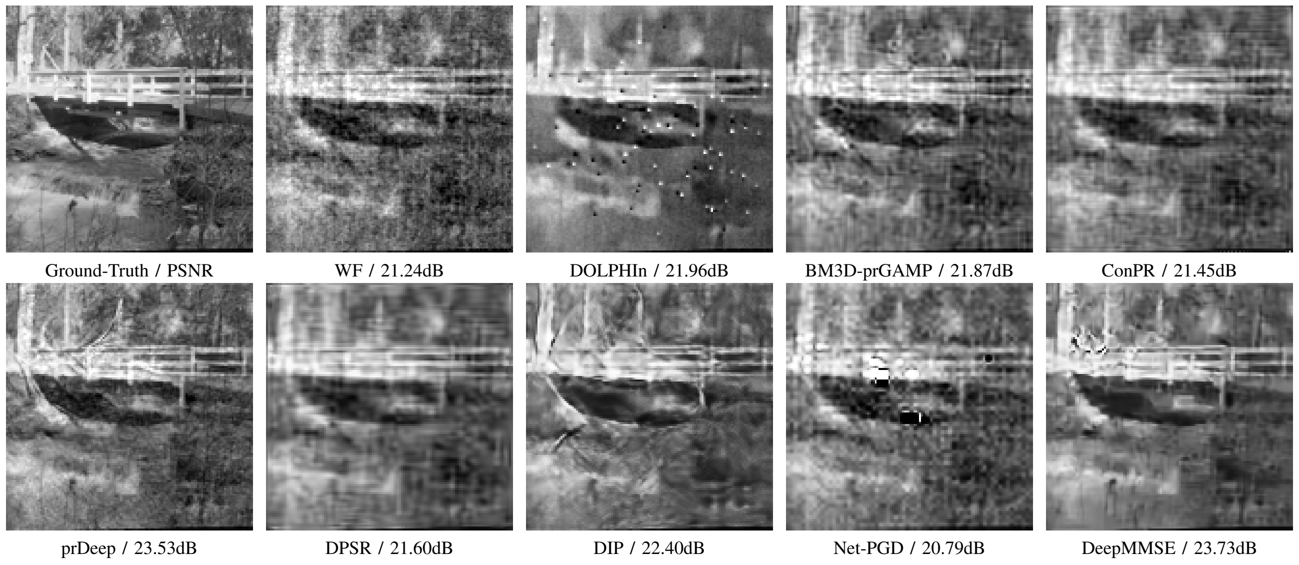


Fig. 6: Reconstructions on image “Boat” from $4\times$ oversampled Fourier magnitude measurements with Poisson noise $(\gamma = 3)$.

all noise strengths across all datasets, in terms of PSNR. The overall average PSNR gain of DeepMMSE over the second best ones is about 0.3dB, 0.2dB, and 0.2dB for $\gamma = 2, 3, 4$ respectively. Such significant quantitative improvement is also consistent with the improvement on visual quality shown in Fig. 6. As can be seen, our result contains fewer artifacts than that of other methods, and it is the most consistent with the original image. See supplementary materials for more results.

G. PR on Measurements from a Physical System

The proposed method is also evaluated on the test set from Metzler *et al.* [58]. It provides noisy magnitude measurement samples captured based on a calibrated amplitude-only spatial light modulator. The associated measuring (transmission) matrix estimated by the prVAMP method proposed in [58] is used for PR. In this experiment, the prVAMP method [58] is also included for comparison. See Fig. 7 and supplementary materials for the visual inspection on some results. Overall, the performance of all PR methods decreases in comparison to their performance on the previous synthetic datasets. One reason is that the estimated transmission matrix is not error-free, and the noise distribution is unknown. Among all compared methods, DeepMMSE remains one of the good performers in terms of recovering structure and suppressing noise.

H. Uncertainty Quantization

Uncertainty quantification on the reconstruction is a property welcomed in many applications, *e.g.* scientific imaging. The nature of DeepMMSE, *i.e.* the estimate over multiple inferences, enables us to quantify the uncertainty of the recon-

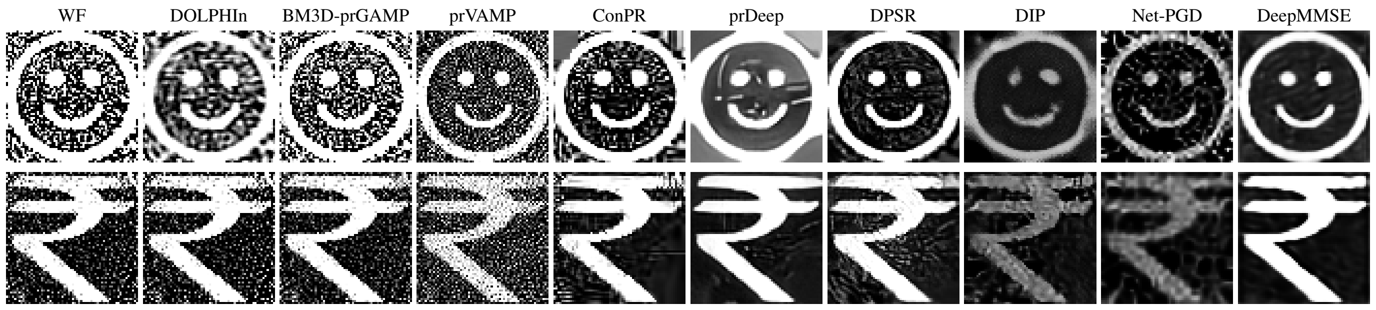


Fig. 7: Visual comparison of the results of different PR methods from the public dataset from [58]

TABLE VII: Quantitative results in Fourier PR in terms of average PSNR(dB) (odd rows) and SSIM (even rows), using $4\times$ over-sampled Fourier magnitude measurements with Poisson noise of different strengths. The best and second results are marked in **blue** and green respectively.

Method	Natural-6			Unnatural-6		
	$\gamma=2$	$\gamma=3$	$\gamma=4$	$\gamma=2$	$\gamma=3$	$\gamma=4$
WF	22.57	20.91	19.13	24.07	22.30	21.56
	0.596	0.492	0.433	0.580	0.500	0.455
DOLPHIn	25.00	23.45	20.69	25.72	23.82	23.62
	0.733	0.671	0.559	0.707	0.631	0.619
BM3D-prGAMP	24.41	21.51	19.23	25.09	22.87	21.77
	0.718	0.559	0.467	0.693	0.606	0.550
ConPR	22.64	20.99	18.93	23.30	21.38	20.86
	0.627	0.515	0.443	0.589	0.507	0.482
prDeep	30.54	29.23	26.27	30.83	27.57	26.29
	0.861	0.823	0.732	0.811	0.692	0.652
DPSR	22.40	20.94	18.83	22.72	20.66	20.31
	0.629	0.519	0.442	0.554	0.470	0.450
DIP	25.53	23.70	20.39	25.03	23.83	22.83
	0.770	0.685	0.553	0.759	0.694	0.640
Net-PGD	21.50	20.86	18.89	19.80	18.81	18.81
	0.587	0.512	0.440	0.468	0.422	0.418
DeepMMSE	**30.85**	**29.48**	**26.51**	**31.24**	**27.97**	**26.53**
	0.884	**0.845**	**0.744**	**0.854**	**0.737**	**0.679**

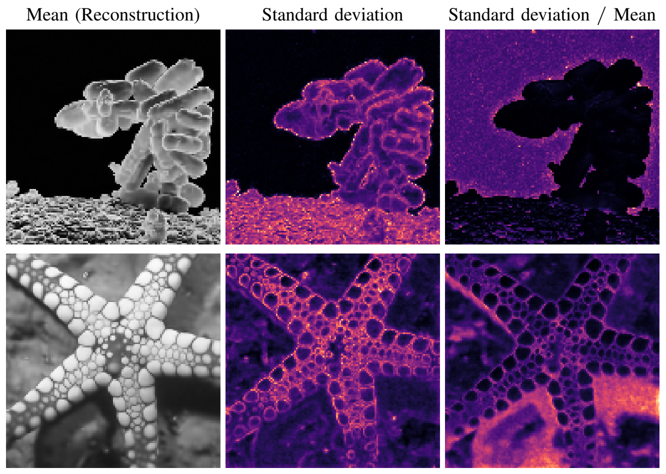


Fig. 8: Pixel-wise statistics over 100 predictions on image "E.Coli" and "Starfish" with four bipolar masks. See more results in supplementary materials.

structed image. See Fig. 8 for an illustration. It shows pixel-wise statistics over 100 inferences in terms of mean, standard deviation, and fractional standard deviation (*i.e.* standard deviation over mean) [53]. It can be seen that, one can estimate the uncertainty on the reconstructed image in the framework of DeepMMSE, where flat regions have lower uncertainty while structure regions (*e.g.* edges) have higher uncertainty.

I. *Ablation Study*

For further analyzing DeepMMSE and verifying the effectiveness of its main components, the following ablation studies and experiments are conducted.

1) Effectiveness of Model Averaging: Recall that our DeepMMSE method uses the dropout-based model averaging to have an accurate estimate. To verify the benefit of such a mechanism in DeepMMSE, we construct a baseline ("w/o MA-1") of DeepMMSE by removing the model averaging step in test and only using the output of the dropout-trained model for reconstruction. We also construct a baseline ("w/o MA-2") by further removing the dropout during model training from "w/o MA-1". These two baselines share the same architecture and stopping criteria as DeepMMSE. The baseline methods are tested in the PR-based reconstruction using four bipolar masks in the presence of AWGN and Poisson noise respectively.

The quantitative comparison of DeepMMSE versus the baselines is given in Table VIII. It can be seen that DeepMMSE with model averaging performs better than the two baselines which do not use model averaging. Such a performance gain has clearly justified the necessity of the introduction of the model averaging mechanism on handling the ill-posedness in PR-based reconstruction. We can also see that the "w/o MA-1" using dropout training performs sometimes better and sometimes worse than "w/o MA-2" which uses no dropout in both training and reconstruction. This is probably because dropout training helps to reduce the overfitting during learning (for better performance) but meanwhile it reduces the capacity of a CNN.

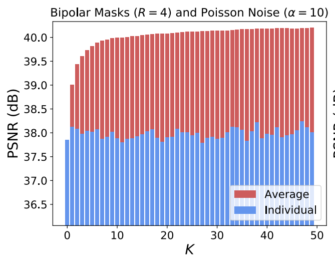
We also test how the value K (*i.e.* the number of models for averaging) affects the reconstruction accuracy. The test is conducted using four bipolar masks with AWGN and Poisson noise respectively. The quantitative results are given in Table IX. See also Fig. 9 for the PSNR of the individual/average reconstructed image *vs* K. As K increases from 1 to 8, the average PSNR of the reconstructed images increases fast

TABLE VIII: Average PSNR(dB) values of PR-based reconstruction using the proposed method and two baselines (w/o MA-1 and w/o MA-2) respectively. Four bipolar masks are used. The noise is set to AWGN with different amount (measured by SNR(dB) of input measurements) and Poisson noise with different amount (measured by γ). The best results are marked in **blue**.

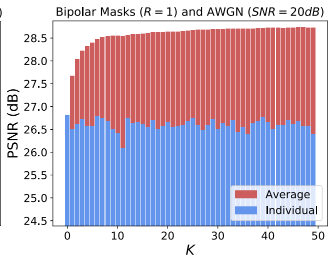
	Natural-6			Unnatural-6			Set-20		
SNR(dB)	10	15	20	10	15	20	10	15	20
w/o MA-2	26.8	29.6	32.6	32.0	34.7	37.5	25.9	28.8	31.7
w/o MA-1	26.5	29.6	33.3	31.0	34.7	38.1	26.6	29.7	33.0
Original	**28.7**	**31.7**	**34.7**	**33.6**	**36.7**	**39.5**	**28.1**	**31.2**	**34.4**
γ	9	27	81	9	27	81	9	27	81
w/o MA-2	36.3	30.4	24.7	39.1	32.9	27.0	36.4	30.3	24.0
w/o MA-1	38.3	31.0	23.4	39.1	32.3	25.7	37.8	31.1	23.3
Original	**39.7**	**32.9**	**26.0**	**41.0**	**34.2**	**27.8**	**39.7**	**32.8**	**25.5**

TABLE IX: Average PSNR(dB) results of DeepMMSE and "Simple" with varied K (*i.e.* the number of models for averaging) on Set-20. Four bipolar masks are used. The noise is set to AWGN with SNR=10dB or Poisson noise with $\gamma = 81$. The total running time (minutes) for $K = 50$ models is reported in the rightmost column. The best results are marked in **blue**.

K		1	2	4	8	16	32	50	Time
Deep MMSE	AWGN	26.6	27.1	27.6	27.9	28.0	28.0	**28.1**	10.4
	Poisson	23.3	24.7	25.0	25.2	25.4	25.4	**25.5**	9.1
Simple	AWGN	26.5	27.3	27.6	27.7	27.7	27.7	27.8	241.7
	Poisson	23.8	24.7	25.0	25.1	25.2	25.2	25.2	203.3



(a) Average result on "Boat"



(b) Average result on "Barbara"

Fig. 9: PSNR of individual/average reconstructed images *vs* K. The average reconstruction is improved as K increases.

and noticeably, which has verified the effectiveness of model averaging. The increase becomes saturated after $K = 32$. In addition, DeepMMSE is compared to a baseline ("Simple"), which runs model averaging by using the models trained with random initializations. That is, we use the NN in such a simple method by removing all dropouts in our NN model. Then in total K models are separately trained by using different random initializations. Finally, the prediction is the average of the predictions from these K trained models. The results from the baseline are included in Table IX for comparison, in terms of both PSNR and time cost. Take $\gamma = 81$ for instance. The running time of the baseline is about 20 times as that of DeepMMSE, while its PSNR result has 0.3dB gap towards ours. Such results have demonstrated the advantages of the dropout model averaging in DeepMMSE over the model averaging with random initializations, in terms of computational efficiency and estimation accuracy.

2) *Effectiveness of loss augmentation:* To verify the benefit of the loss augmentation strategy employed by DeepMMSE, we construct a baseline named "w/o LA" which does not use the loss augmentation in DeepMMSE. The baseline uses the same architecture and stopping criteria as DeepMMSE. The quantitative comparison is given in Table X, which is conducted using four bipolar masks with AWGN and Poisson noise respectively. It can be seen that the loss augmentation brings slight performance improvement.

TABLE X: Average PSNR(dB) values of PR-based reconstruction using DeepMMSE and the baseline "w/o LA" respectively. Four bipolar masks are used. The noise is set to AWGN with strength measured by SNR(dB) of input measurements or Poisson noise with strength measured by γ. The best results are marked in **blue**.

Dataset	Natural-6			Unnatural-6			Set-20		
SNR(dB)	10	15	20	10	15	20	10	15	20
w/o LA	28.4	31.4	34.4	33.2	36.4	39.1	27.8	31.0	34.1
w/ LA	**28.7**	**31.7**	**34.7**	**33.6**	**36.7**	**39.5**	**28.1**	**31.2**	**34.4**
γ	9	27	81	9	27	81	9	27	81
w/o LA	39.2	32.6	25.7	40.6	34.0	27.4	39.3	32.4	25.1
w/ LA	**39.7**	**32.9**	**26.0**	**41.0**	**34.2**	**27.8**	**39.7**	**32.8**	**25.5**

J. *Impact of Certain Parameters to Performance*

The impact of the setting of various parameters to the performance is analyzed, which includes the maximum iteration number, tolerance threshold, and dropout pattern/ratios. The experiments are conducted on Set-20 dataset using four bipolar masks in the presence of AWGN with SNR=20dB. Regarding the maximum iteration number and tolerance threshold, we run DeepMMSE without using the stopping criterion and plot the PSNR values *w.r.t.* iteration number in Fig. 10. In Fig. 10, five points with the corresponding tolerances are marked out. It can be seen that the performance of DeepMMSE increases very fast at the beginning and saturates before the stopping criterion (*i.e.* iteration number or tolerance) is satisfied. Afterwards, the performance with further iterations starts to decrease. This is because the model becomes overfitting with the loss being close to 0. Overall, the performance of DeepMMSE is not sensitive to the maximal number and tolerance threshold within a reasonable range.

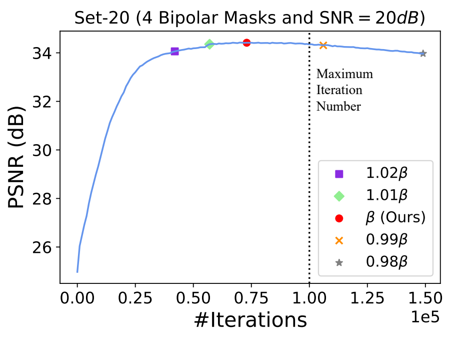


Fig. 10: Influence of iteration number and tolerance threshold.

TABLE XI: Average PSNR(dB) results of DeepMMSE using two dropout probability setting schemes on Set-20 using four bipolar masks in the presence of AWGN with SRN=20dB.

q_0	10%	20%	30%
Proposed Dropout	34.2	34.3	34.4
Constant Dropout	33.9	33.7	33.2

For the setting of dropout pattern/probabilities, two configurations are used for the experiments. The first setting uses a varying dropout probability defined by the proposed symmetric rule with initial point q_0. The second setting uses a constant dropout probability across all dropout layers, *i.e.* $q_j = q_0, \forall j$. On both settings, we evaluate the performance of the models trained with $q_0 = 10\%, 20\%, 30\%$ respectively. See Table XI for the comparison of the results using different settings. The performance from the proposed symmetric rule for setting the varying dropout probability is significantly better than that from the constant rule. The main reason is that different layers in the NN have different capacities due to their differences in feature size and channel number. As different dropout probabilities will reduce the model capacity by different degrees. Thus, different dropout probabilities need to be imposed on different layers to fit the model capacity of the layers. In addition, it can be seen that the performance of DeepMMSE is not sensitive to the initial point q_0 within a reasonable range.

V. Conclusion

In this paper, we proposed to solve the problem of PR using the deep prior provided by an untrained NN. The proposed approach used an over-parameterized deep generative CNN for reconstructing the unknown image from the input measurements, which is trained without any training samples, which may avoid possible bias introduced by external training data and cut the cost of data collection in the meanwhile. To seek for an approximation to the MMSE estimator, we used dropout model averaging and demonstrated it as a powerful prior to overcome the overfitting to noise and resolve the ambiguity caused by the lack of phase information. The effectiveness of proposed approach was justified on extensive experiments with various configurations. In future, we will study the extension to handle other nonlinear inverse problems in signal processing.

Appendix

A. Implementation Details of Compared Methods

1) Regarding all experiments except the one on the real dataset from [58]: Following the experimental protocol in DPSR [19], the HIO, WF, DOLPHIn, BM3D-prGAMP, ConPR, and DPSR run with the suggested parameters in their public official implementations. The results of DPR, Net-GD and Net-PGD are also obtained using its official code with suggested parameters. The results of SPAR and TFPnP are quoted from [57]. For prDeep, its official implementation with suggested parameters is used in the experiments on CDP measurements and Fourier magnitude measurements. See Section IV-B for the implementation details of DIP.

2) Regarding the experiment on the real dataset from [58]: The results of prVAMP are produced by the official implementation from [58]. For other methods, their implementations and parameter settings for bipolar masks and AWGN are used. Since the implementation of prVAMP provides the noise level parameter tuned by the dataset provider, we use it for the methods (*e.g.* prDeep) which require noise level as input.

References

[1] R. W. Gerchberg, "A practical algorithm for the determination of phase from image and diffraction plane pictures," *Optik*, 1972. 1, 3

[2] J. C. Dainty and J. R. Fienup, "Phase retrieval and image reconstruction for astronomy," *Image Recovery: Theory and Application*, pp. 231–275, 1987. 1

[3] F. Zhang, X. Liu, C. Guo, S. Lin, J. Jiang, and X. Ji, "Physics-based iterative projection complex neural network for phase retrieval in lensless microscopy imaging," in *Proc. CVPR*, 2021, pp. 10 523–10 531. 1

[4] Y. Rivenson, Y. Zhang, H. Gunaydin, D. Teng, and A. Ozcan, "Phase recovery and holographic image reconstruction using deep learning in neural networks," *Light, science & applications*, vol. 7, 2017. 1

[5] G. Jagatap, Z. Chen, S. Nayer, C. Hegde, and N. Vaswani, "Sample efficient fourier ptychography for structured data," *IEEE Trans. Comput. Imaging*, vol. 6, pp. 344–357, 2020. 1

[6] R. P. Millane, "Phase retrieval in crystallography and optics," *Journal of the Optical Society of America A*, vol. 7, no. 3, pp. 394–411, 1990. 1

[7] M. Hayes, "The reconstruction of a multidimensional sequence from the phase or magnitude of its fourier transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 30, no. 2, pp. 140–154, 1982. 1

[8] R. Hyder, Z. Cai, and M. S. Asif, "Solving phase retrieval with a learned reference," *Proc. ECCV*, pp. 425–441, 2020. 1, 3

[9] P. Schniter and S. Rangan, "Compressive phase retrieval via generalized approximate message passing," *IEEE Trans. Signal Process*, vol. 63, no. 4, pp. 1043–1055, 2014. 1

[10] C. A. Metzler, A. Maleki, and R. G. Baraniuk, "Bm3d-prgamp: Compressive phase retrieval based on bm3d denoising," in *Proc. ICIP*, 2016, pp. 2504–2508. 1, 2, 3, 7, 10

[11] T. Qiu and D. P. Palomar, "Undersampled sparse phase retrieval via majorization–minimization," *IEEE Trans. Signal Process.*, vol. 65, pp. 5957–5969, 2017. 1, 3

[12] G. Baechler, M. Kreković, J. Ranieri, A. Chebira, Y. M. Lu, and M. Vetterli, "Super resolution phase retrieval for sparse signals," *IEEE Trans. Signal Process.*, vol. 67, pp. 4839–4854, 2019. 1

[13] H. Chang, Y. Lou, Y. Duan, and S. Marchesini, "Total variation based phase retrieval for poisson noise removal," *SIAM J. Imaging Sciences*, vol. 11, pp. 24–55, 2018. 1, 3

[14] B. Shi, Q. Lian, S. Chen, Y. Tian, and F. Xiaoyu, "Coded diffraction imaging via double sparse regularization model," *Digit. Signal Process.*, vol. 79, pp. 23–33, 2018. 1, 3

[15] V. Katkovnik, "Phase retrieval from noisy data based on sparse approximation of object phase and amplitude," *arXiv: 1709.01071*, 2017. 2, 7

[16] V. Katkovnik and K. O. Egiazarian, "Sparse superresolution phase retrieval from phase–coded noisy intensity patterns," *Optical Engineering*, vol. 56, 2017. 2, 3

[17] B. Shi, Q. Lian, X. Huang, and N. An, "Constrained phase retrieval: when alternating projection meets regularization," *J. Opt. Soc. Amer. B*, vol. 35, pp. 1271–1281, 2018. 2, 3, 7, 10

[18] C. A. Metzler, P. Schniter, A. Veeraraghavan, and R. G. Baraniuk, "prdeep: robust phase retrieval with a flexible deep network," in *Proc. ICML*, 2018. 2, 3, 6, 7, 10

[19] B. Shi, Q. Lian, and H. Chang, "Deep prior-based sparse representation model for diffraction imaging: a plug-and-play method," *Signal Process.*, vol. 168, 2020. 2, 3, 6, 7, 10, 14

[20] P. Hand, O. Leong, and V. Voroninski, "Phase retrieval under a generative prior," in *Proc. NeurIPS*, 2018. 2, 3, 4, 10

[21] F. Shamshad and A. Ahmed, "Robust compressive phase retrieval via deep generative priors," *arXiv: 1808.05854*, 2018. 2, 3, 4

[22] G. Jagatap and C. Hegde, "Algorithmic guarantees for inverse imaging with untrained network priors," in *Proc. NeurIPS*, 2019. 2, 3, 4, 7, 10

[23] D. Ulyanov, A. Vedaldi, and V. S. Lempitsky, "Deep image prior," in *Proc. CVPR*, 2018, pp. 9446–9454. 2, 7, 10

[24] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *J. Mach. Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014. 2

[25] Y. Gal and Z. Ghahramani, "Bayesian convolutional neural networks with bernoulli approximate variational inference," in *Proc. ICLR*, 2015. 3

[26] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?" in *Proc. NeurIPS*, 2017, pp. 5574–5584. 3

[27] J. R. Fienup, "Phase retrieval algorithms: a comparison," *Applied Optics*, vol. 21, no. 15, pp. 2758–2769, 1982. 3, 7

[28] J. Ma, R. Dudeja, J. Xu, A. Maleki, and X. Wang, "Spectral method for phase retrieval: an expectation propagation perspective," *IEEE Trans. Information Theory*, vol. 67, no. 2, pp. 1332–1355, 2021. 3

[29] E. J. Candès, T. Strohmer, and V. Voroninski, "Phaselift: exact and stable signal recovery from magnitude measurements via convex programming," *Comm. Pure Applied Math*, vol. 66, no. 8, pp. 1241–1274, 2013. 3

[30] Y. Shechtman, A. Beck, and Y. C. Eldar, "Gespar: Efficient phase retrieval of sparse signals," *IEEE Trans. Signal Process*, vol. 62, no. 4, pp. 928–938, 2014. 3

[31] I. Waldspurger, A. d'Aspremont, and S. Mallat, "Phase recovery, maxcut and complex semidefinite programming," *Math. Program.*, vol. 149, pp. 47–81, 2015. 3

[32] E. J. Candès, X. Li, and M. Soltanolkotabi, "Phase retrieval via wirtinger flow: theory and algorithms," *IEEE Trans. Information Theory*, vol. 61, pp. 1985–2007, 2015. 3, 7, 10

[33] P. Netrapalli, P. Jain, and S. Sanghavi, "Phase retrieval using alternating minimization," *IEEE Trans. Signal Process*, vol. 63, no. 18, pp. 4814–4826, 2015. 3

[34] T. Cai, X. Li, and Z. Ma, "Optimal rates of convergence for noisy sparse phase retrieval via thresholded wirtinger flow," *The Annals of Statistics*, vol. 44, no. 5, pp. 2221–2251, 2016. 3

[35] N. Vaswani, S. Nayer, and Y. C. Eldar, "Low-rank phase retrieval," *IEEE Trans. Signal Process*, vol. 65, no. 15, pp. 4059–4074, 2017. 3

[36] T. Goldstein and C. Studer, "Phasemax: convex phase retrieval via basis pursuit," *IEEE Trans. Information Theory*, vol. 64, pp. 2675–2689, 2018. 3

[37] B. Wang, J. Fang, H. Duan, and H. Li, "Phaseequal: convex phase retrieval via alternating direction method of multipliers," *IEEE Trans. Signal Process.*, vol. 68, pp. 1274–1285, 2020. 3

[38] Y. Xia and Z. Xu, "Sparse phase retrieval via phaseliftoff," *IEEE Trans. Signal Process*, vol. 69, pp. 2129–2143, 2021. 3

[39] J. Kornprobst, A. Paulus, J. Knapp, and T. F. Eibert, "Phase retrieval for partially coherent observations," *IEEE Trans. Signal Process*, vol. 69, pp. 1394–1406, 2021. 3

[40] A. Bora, A. Jalal, E. Price, and A. Dimakis, "Compressed sensing using generative models," in *Proc. ICML*, 2017. 3, 4

[41] B. Shi, S. Chen, Y. Tian, F. Xiaoyu, and Q. Lian, "Faspr: A fast sparse phase retrieval algorithm via the epigraph concept," *Digit. Signal Process.*, vol. 80, pp. 12–26, 2018. 3

[42] A. M. Tillmann, Y. C. Eldar, and J. Mairal, "Dolphin-dictionary learning for phase retrieval," *IEEE Trans. Signal Process.*, vol. 64, pp. 6485–6500, 2016. 3, 7, 10

[43] T. Liu, A. M. Tillmann, Y. Yang, Y. C. Eldar, and M. Pesavento, "A parallel algorithm for phase retrieval with dictionary learning," in *Proc. ICASSP*. IEEE, 2021, pp. 5619–5623. 3

[44] V. Katkovnik and J. Astola, "Phase retrieval via spatial light modulator phase modulation in 4f optical setup: numerical inverse imaging with sparse regularization for phase and amplitude," *J. Opt. Soc. Amer. A*, vol. 29, no. 1, pp. 105–116, 2012. 3

[45] A. Danielyan, V. Katkovnik, and K. Egiazarian, "Bm3d frames and variational image deblurring," *IEEE Trans. Image Process.*, vol. 21, no. 4, pp. 1715–1728, 2012. 3

[46] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-d transform-domain collaborative filtering," *IEEE Trans. Image Process.*, vol. 16, no. 8, pp. 2080–2095, 2007. 3

[47] K. Zhang, W. Zuo, S. Gu, and L. Zhang, "Learning deep cnn denoiser prior for image restoration," in *Proc. CVPR*, 2017, pp. 3929–3938. 3

[48] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *CoRR, abs/1511.06434*, 2015. 4

[49] E. Bostan, R. Heckel, M. Chen, M. Kellman, and L. Waller, "Deep phase decoder: self-calibrating phase microscopy with an untrained deep neural network," *Optica*, vol. 7, no. 6, pp. 559–562, 2020. 4

[50] H. Lawrence, D. A. Barmherzig, H. Li, M. Eickenberg, and M. Gabrié, "Phase retrieval with holography and untrained priors: Tackling the challenges of low-photon nanoscale imaging," 2021. 4

[51] J. Liu, M. M. Balaji, C. A. Metzler, M. S. Asif, and P. Rangarajan, "Solving inverse problems using self-supervised deep neural nets," in *Computational Optical Sensing and Imaging*. Optical Society of America, 2021, pp. CTh5A–2. 4

[52] R. Heckel and P. Hand, "Deep decoder: concise image representations from untrained non-convolutional networks," in *Proc. ICLR*, 2018. 4

[53] H. Sun and K. L. Bouman, "Deep probabilistic imaging: Uncertainty quantification and multi-modal solution characterization for computational imaging," in *Proc. AAAI*, vol. 9, 2021. 4, 12

[54] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul, "An introduction to variational methods for graphical models," *Machine learning*, vol. 37, no. 2, pp. 183–233, 1999. 4

[55] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. ICAIS*. JMLR Workshop and Conference Proceedings, 2010, pp. 249–256. 6

[56] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014. 6

[57] K. Wei, A. Aviles-Rivero, J. Liang, Y. Fu, C.-B. Schnlieb, and H. Huang, "Tuning-free plug-and-play proximal algorithm for inverse imaging problems," in *Proc. ICML*, 2020. 7, 14

[58] C. A. Metzler, M. K. Sharma, S. Nagesh, R. G. Baraniuk, O. Cossairt, and A. Veeraraghavan, "Coherent inverse scattering via transmission matrices: Efficient phase retrieval algorithms and a public dataset," in *Proc. ICCP*. IEEE, 2017, pp. 1–16. 11, 12, 14