Low Shot Learning with Untrained Neural Networks for Imaging Inverse Problems

Oscar Leong[*]
Rice University
oscar.f.leong@rice.edu

Wesam Sakla
Lawrence Livermore National Laboratory
sakla1@llnl.gov

Abstract

Employing deep neural networks as natural image priors to solve inverse problems either requires large amounts of data to sufficiently train expressive generative models or can succeed with no data via untrained neural networks. However, very few works have considered how to interpolate between these no- to high-data regimes. In particular, how can one use the availability of a small amount of data (even $5 - 25$ examples) to one's advantage in solving these inverse problems and can a system's performance increase as the amount of data increases as well? In this work, we consider solving linear inverse problems when given a small number of examples of images that are drawn from the same distribution as the image of interest. Comparing to untrained neural networks that use no data, we show how one can pre-train a neural network with a few given examples to improve reconstruction results in compressed sensing and semantic image recovery problems such as colorization. Our approach leads to improved reconstruction as the amount of available data increases and is on par with fully trained generative models, while requiring less than 1% of the data needed to train a generative model.

1 Introduction

We study the problem of recovering an image $\boldsymbol{x}_0 \in \mathbb{R}^n$ from m linear measurements of the form

$$\boldsymbol{y}_0 = \boldsymbol{A}\boldsymbol{x}_0 + \boldsymbol{\eta} \in \mathbb{R}^m$$

where $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ is a known measurement operator and $\boldsymbol{\eta} \in \mathbb{R}^m$ denotes the noise in our system. Problems of this form are ubiquitous in various domains ranging from image processing, machine learning, and computer vision. Typically, the problem's difficulty is a result of its ill-posedness due to the underdetermined nature of the system. To resolve this ambiguity, many approaches enforce that the image must obey a natural image model. While traditional approaches typically use hand-crafted priors such as sparsity in the wavelet basis [5], recent approaches inspired by deep learning to create such natural image model surrogates have shown to outperform these methods.

Deep Generative Priors: Advancements in generative modelling have allowed for deep neural networks to create highly realistic samples from a number of complex natural image classes. Popular generative models to use as natural image priors are latent variable models such as Generative Adversarial Networks (GANs) [6] and Variational Autoencoders (VAEs) [18]. This is in large part due to the fact that they provide a low-dimensional parameterization of the natural image manifold that can be directly exploited in inverse imaging tasks. When enforced as a natural image prior, these models have shown to outperform traditional methods and provide theoretical guarantees in problems such as compressed sensing [4, 24, 11, 14, 20, 15], phase retrieval [10, 21, 16], and blind

[*]This work was performed at the Data Science Summer Institute (DSSI) under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

deconvolution/demodulation [2, 9]. However, there are two main drawbacks of using deep generative models as natural image priors. The first is that they require a large amount of data to train, e.g., hundreds of thousands of images to generate novel celebrity faces. Additionally, they suffer from a non-trivial representation error due to the fact that they model the natural image manifold through a low-dimensional parameterization.

Untrained Neural Network Priors: On the opposite end of the data spectrum, recent works have shown that randomly initialized neural networks can act as natural image priors without any learning. [22] first showed this to be the case by solving tasks such as denoising, inpainting, and super-resolution via optimizing over the parameters of a convolutional neural network to fit to a single image. The results showed that the neural network exhibited a bias towards natural images, but due to the high overparameterization in the network, required early stopping to succeed. A simpler model was later introduced in [13] which was, in fact, underparameterized and was able to both compress images while solving various linear inverse problems. Both methods require no training data and do not suffer from the same representation error as generative models do. Similar to generative models, they have shown to be successful image priors in a variety of inverse problems [13, 12, 23, 17].

Based on these two approaches, we would like to investigate how can one interpolate between these data regimes in a way that improves upon work with untrained neural network priors and ultimately reaches or exceeds the success of generative priors. More specifically, we would like to develop an algorithm that 1) performs just as well as untrained neural networks with no data and 2) improves performance as the amount of provided data increases.

Our contributions: We introduce a framework to solve inverse problems given a few examples (e.g., $5 - 25$) drawn from the same data distribution as the image of interest (e.g., if the true image is of a human face, the examples are also human faces). Our main contributions are the following:

- We show how one can pre-train a neural network using a few examples drawn from the data distribution of the image of interest. Inspired by [3], we propose to jointly learn a latent space and parameters of the network to fit to the examples that are given and compare the use of an ℓ_2 reconstruction loss and a Maximum Mean Discrepancy (MMD) loss.
- We then propose to solve the inverse problem via a two-step process. We first optimize over the pre-trained network's latent space. Once a solution is found, we then refine our estimate by optimizing over the latent space and parameters of the network jointly to improve our solution. [15] found this method to work well in the case when the network is a fully trained generative model, and we show here that even a pre-trained neural network from a small number of examples can benefit from such an approach.
- We show that our approach improves upon untrained neural networks in compressed sensing even with as few as 5 examples from the data distribution and exhibits improvements as the number of examples increases. We also show that semantics can be learned from these few examples in problems such as colorization where untrained neural networks fail. With only 100 examples, our model's performance is competitive with fully trained generative models.

Related work: We mention that there has been previous work [23] in investigating how to use a small amount of data to help solve the compressed sensing problem. The authors use an untrained neural network as a natural image prior and, when given a small amount of data, adopt a learned regularization term when solving the inverse problem. This term is derived by posing the recovery problem as a Maximum a Posteriori (MAP) estimation problem and by placing a Gaussian prior on the weights of the untrained network. While we have not compared our method to this learned regularization approach here, we aim to do so in a subsequent manuscript.

2 Low Shot Learning For Imaging Inverse Problems

We consider the problem of recovering an image $\boldsymbol{x}_0 \in \mathbb{R}^n$ from noisy linear measurements of the form $\boldsymbol{y}_0 = \boldsymbol{A}\boldsymbol{x}_0 + \boldsymbol{\eta} \in \mathbb{R}^m$ where $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ and $m \leqslant n$. We also assume that $\boldsymbol{x}_0$ is drawn from a particular data distribution $\mathcal{D}$ and that we are given a low number of examples drawn from the same distribution, i.e., $\boldsymbol{x}_0 \sim \mathcal{D}$ and we are given $\boldsymbol{x}_i \sim \mathcal{D}$ where $i \in [S]$. Here and throughout this work, we refer to these examples drawn from $\mathcal{D}$ as *low shots*.

We propose using the range of a deep neural network as a natural image model. In particular, we model the image $\boldsymbol{x}_0$ as the output of a neural network $\mathcal{G}(\boldsymbol{z};\boldsymbol{\theta})$, where $\boldsymbol{z} \in \mathbb{R}^k$ is a latent code and $\boldsymbol{\theta} \in \mathbb{R}^P$ are the parameters of the network.

Pre-training: Prior to solving the inverse problem, we propose to first pre-train the network using the low shots that are given. More specifically, we fit the weights and input of the neural network to the low shots to provide a crude approximation to the data distribution underlying the image of interest. Given low shots $\{\boldsymbol{x}_i\}_{i=1}^S$, we aim to find latent codes $\{\boldsymbol{z}_i\}_{i=1}^S$ and parameters $\boldsymbol{\theta}$ that solve

$$\min_{\boldsymbol{\theta},\boldsymbol{z}_1,\dots,\boldsymbol{z}_S} \frac{1}{S}\sum_{i=1}^{S} \mathcal{L}(\mathcal{G}(\boldsymbol{z}_i;\boldsymbol{\theta}),\boldsymbol{x}_i). \tag{1}$$

where $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is a loss function. We investigate the use of different loss functions in a later section. The resulting optimal parameters found are denoted by $\hat{\boldsymbol{\theta}}$, $\hat{\boldsymbol{z}}_1,\dots,\hat{\boldsymbol{z}}_S$.

Solving the inverse problem: Using the weights found via pre-training, we begin solving the inverse problem by first optimizing over the latent code space to find an approximate solution:

$$\min_{\boldsymbol{z}} \frac{1}{2}\left\|\boldsymbol{A}\mathcal{G}(\boldsymbol{z};\hat{\boldsymbol{\theta}}) - \boldsymbol{y}_0\right\|_2^2. \tag{2}$$

We investigated different ways to initialize the latent code and found that sampling from a multivariate Gaussian distribution fit using $\{\hat{\boldsymbol{z}}_i\}_{i=1}^S$ was sufficient. Note here that we keep the parameters of the network fixed after training. The intuition is that we want to use the semantics regarding the data distribution learned via pre-training the network's parameters and find the optimal latent code that corresponds to the image of interest. Once the optimal latent code $\hat{\boldsymbol{z}}$ is found, we then refine our solution by solving

$$\min_{\boldsymbol{\theta},\boldsymbol{z}} \frac{1}{2}\left\|\boldsymbol{A}\mathcal{G}(\boldsymbol{z};\boldsymbol{\theta}) - \boldsymbol{y}_0\right\|_2^2 \tag{3}$$

with $\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{z}}$ as our initial iterates. The resulting parameters $\boldsymbol{\theta}_0$ and $\boldsymbol{z}_0$ give our final estimate: $\boldsymbol{x}_0 \approx \mathcal{G}(\boldsymbol{z}_0;\boldsymbol{\theta}_0)$.

Losses to learn the data distribution: We discuss two loss functions that we considered in our experiments to learn semantics regarding the underlying data distribution. The first is a simple ℓ_2 reconstruction loss to promote data fidelity, i.e., we pre-train our network by solving

$$\min_{\boldsymbol{\theta},\boldsymbol{z}_1,\dots,\boldsymbol{z}_S} \frac{1}{S}\sum_{i=1}^{S} \left\|\mathcal{G}(\boldsymbol{z}_i;\boldsymbol{\theta}) - \boldsymbol{x}_i\right\|_2^2. \tag{4}$$

While [3] used a combination of the Laplacian-L1 loss and ℓ_2 loss, we found the ℓ_2 loss to work well.

The second loss is an estimate of the kernel MMD for comparing two probability distributions using only finitely many samples [7]. In our case, given a kernel $k(\cdot,\cdot)$[2] and low shots $\boldsymbol{x}_j \sim \mathcal{D}$ for $j \in [S]$, we want to find parameters and S inputs that solve the following:

$$\min_{\boldsymbol{\theta},\boldsymbol{z}_1,\dots,\boldsymbol{z}_S} \frac{1}{\binom{S}{2}}\sum_{i\neq i'} k(\mathcal{G}(\boldsymbol{z}_i;\boldsymbol{\theta}),\mathcal{G}(\boldsymbol{z}_{i'};\boldsymbol{\theta})) + \frac{1}{\binom{S}{2}}\sum_{j\neq j'} k(\boldsymbol{x}_j,\boldsymbol{x}_{j'}) - \frac{2}{\binom{S}{2}}\sum_{i\neq j} k(\mathcal{G}(\boldsymbol{z}_i;\boldsymbol{\theta}),\boldsymbol{x}_j). \tag{5}$$

We compare the success of these two loss functions in the following section.

3 Experiments

We now consider solving inverse problems with our approach and compare to three different baselines: an untrained neural network, optimizing the latent space of a trained Wasserstein GAN [1] with gradient penalty [8], and the image-adaptivity approach of [15] (IAGAN). Each method uses the same DCGAN architecture with a latent code dimension of 128. In each problem, the image of interest is from a hold-out test set from the CelebA dataset [19]. The GAN was trained on a corpus of over $200,000$ 64×64 celebrity images and our low-shot models were trained on small ($5 - 100$ images) subsets of this.

[2]We only consider the Gaussian kernel $k_\alpha(\boldsymbol{x}_1,\boldsymbol{x}_2) := \exp(-\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2^2/\alpha)$ in our experiments.

Compressed Sensing: We first consider the compressed sensing problem where we want to recover an image $\boldsymbol{x}_0 \in \mathbb{R}^n$ from random Gaussian measurements of the form $\boldsymbol{y}_0 = \boldsymbol{A}\boldsymbol{x}_0 \in \mathbb{R}^m$ where $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ has i.i.d. $\mathcal{N}(0,1)$ entries with $m \ll n$. We refer to amount of undersampling $\frac{m}{n}$ as the *compression ratio*. We trained our models using the two different loss functions proposed in the previous section for various numbers of shots $S \in [5, 10, 15, 25, 50, 100]$.

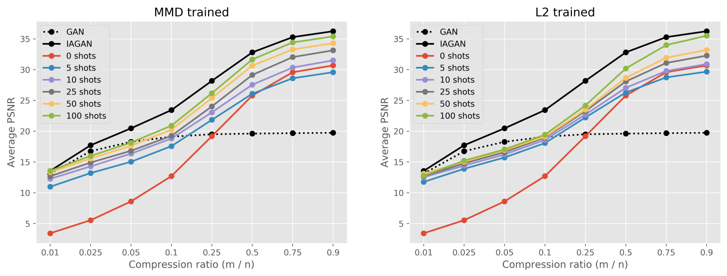

Figure 1: Average PSNR over 50 different test images for models trained with the MMD loss (left) and the ℓ_2 loss (right).

Figure 1 compares the average PSNR for each method at various compression ratios over 50 different test images and different loss functions. We note that as the number of shots increases, our method continues to improve and we see comparable performance between our method with only 100 shots and optimizing over the latent code space of a fully trained GAN. While the ℓ_2 trained nets perform slightly better than the MMD trained nets for low numbers of shots, the MMD trained nets improve more steadily and consistently as the number of shots increases. While we expect IAGAN to be superior due to being trained with over 200,000 images, the MMD trained model's performance with 100 images is not far behind. We note that for higher numbers of measurements, the untrained neural network's performance surpasses that of our 5 shot models. This is mainly due to the fact that we optimized the untrained neural network's parameters for a longer period of time and with a higher learning rate than our low shot models in this easier setting.

Colorization: We now consider an inverse problem that requires an understanding of the underlying data's semantics: the colorization task. Here we want to recover an RGB image $\boldsymbol{x}_0 \in \mathbb{R}^{64\times64\times3}$ from its grayscale version $\boldsymbol{y}_0 = \boldsymbol{A}\boldsymbol{x}_0 \in \mathbb{R}^{64\times64}$. The operator $\boldsymbol{A}$ mixes the color channels of the image via the ITU-R 601-2 luma transform (the same transform used by the Python Imaging Library (PIL)). Untrained neural networks clearly fail in solving this type of problem since they have no prior information regarding the data distribution. We compare our model trained with 10 shots using the MMD loss to the various baselines in Figure 2. Note that even with only 10 previous examples, our model does not fall prey to the usual issues with using untrained neural networks for colorization. Our algorithm provides faithful image reconstructions that are even on par with a trained GAN.

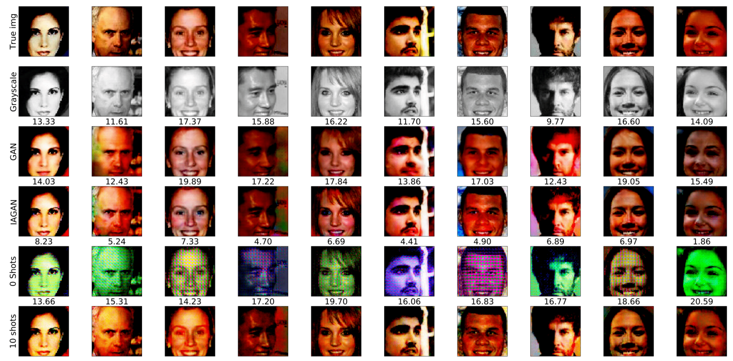

Figure 2: Qualitative results for the colorization task. Rows 1 and 2 are the true and grayscale images while rows 3-6 are the GAN, IAGAN, untrained neural network, and our 10 shot model's reconstructions, respectively. The PSNR of each result is shown above the reconstruction.

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. *International Conference on Machine Learning (ICML)*, 2017.

[2] Muhammad Asim, Fahad Shamshad, and Ali Ahmed. Blind image deconvolution using deep generative priors. *arXiv preprint*, arXiv:1802.04073, 2018.

[3] Piotr Bojanowski, Armand Joulin, David Lopez-Paz, and Arthur Szlam. Optimizing the latent space of generative networks. *International Conference on Machine Learning (ICML)*, 2018.

[4] Ashish Bora, Alexandros G. Dimakis, Ajil Jalal, and Eric Price. Compressed sensing using generative models. *International Conference on Machine Learning (ICML)*, 2017.

[5] David Donoho, Michael Lustig, and John M. Pauly. Sparse mri: The application of compressed sensing for rapid mr imaging. *Magnetic Resonance in Medicine*, 58(6):1182–1195, 2007.

[6] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *arXiv preprint*, arXiv:1406.2661, 2014.

[7] Arthur Gretton, Karsten M. Borgwardt, Bernhard Schölkopf Malte J. Rasch, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13:723–773, 2012.

[8] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. *Neural Information Processing Systems (NIPS)*, 2017.

[9] Paul Hand and Babhru Joshi. Global guarantees for blind demodulation with generative priors. *arXiv preprint*, arXiv:1905.12576, 2019.

[10] Paul Hand, Oscar Leong, and Vladislav Voroninski. Phase retrieval under a generative prior. *Neural Information Processing Systems (NeurIPS)*, 2018.

[11] Paul Hand and Vladislav Voroninski. Global guarantees for enforcing generative priors by empirical risk. *Conference on Learning Theory (COLT)*, 2018.

[12] Reinhard Heckel. Regularizing linear inverse problems with convolutional neural networks. *arXiv preprint*, arXiv:1907.0310, 2019.

[13] Reinhard Heckel and Paul Hand. Deep decoder: Concise image representations from untrained non-convolutional networks. *International Conference on Learning Representations (ICLR)*, 2019.

[14] Wen Huang, Paul Hand, Reinhard Heckel, and Vladislav Voroninski. A provably convergent scheme for compressive sensing under random generative priors. *arXiv preprint*, arXiv:1812.04176, 2018.

[15] Shady Abu Hussein, Tom Tirer, and Raja Giryes. Image-adaptive gan based reconstruction. *arXiv preprint*, arXiv:1906.05284, 2019.

[16] Rakib Hyder, Viraj Shah, Chinmay Hegde, and M. Salman Asif. Alternating phase projected gradient descent with generative priors for solving compressive phase retrieval. *arXiv preprint*, arXiv:1903.02707, 2019.

[17] Gauri Jagatap and Chinmay Hegde. Algorithmic guarantees for inverse imaging with untrained network priors. *arXiv preprint*, arXiv:1906.08763, 2019.

[18] Diederik P. Kingma and Max Welling. Auo-encoding variational bayes. *ICLR*, 2014.

[19] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. *Proceedings of the IEEE International Conference on Computer Vision*, pages 3730–3738, 2015.

[20] Viraj Shah and Chinmay Hegde. Solving linear inverse problems using gan priors: An algorithm with provable guarantees. *arXiv preprint*, arXiv:1802.08406, 2018.

[21] Fahad Shamshad and Ali Ahmed. Robust compressive phase retrieval via deep generative priors. *arXiv preprint*, arXiv:1808.05854:0, 2018.

[22] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. *arXiv preprint*, arXiv:1711.10925, 2017.

[23] Dave Van Veen, Ajil Jalal, Mahdi Soltanolkotabi, Eric Price, Sriram Vishwanath, and Alexandros G. Dimakis. Compressed sensing with deep image prior and learned regularization. *arXiv preprint*, arXiv:1806.06438, 2019.

[24] Yan Wu, Mihaela Rosca, and Timothy Lillicrap. Deep compressed sensing. *International Conference on Machine Learning (ICML)*, 2019.