# Accelerated MRI with Un-trained Neural Networks

Mohammad Zalbagi Darestani* and Reinhard Heckel†,*

*Dept. of Electrical and Computer Engineering, Rice University
†Dept. of Electrical and Computer Engineering, Technical University of Munich

November 24, 2020

**Abstract**

Convolutional Neural Networks (CNNs) are highly effective for image reconstruction problems. Typically, CNNs are trained on large amounts of training images. Recently, however, un-trained CNNs such as the Deep Image Prior and Deep Decoder have achieved excellent performance for image reconstruction problems such as denoising and inpainting, *without using any training data*. Motivated by this development, we address the reconstruction problem arising in accelerated MRI with un-trained neural networks. We propose a highly-optimized un-trained recovery approach based on a variation of the Deep Decoder. We show that the resulting method significantly outperforms conventional un-trained methods such as total-variation norm minimization, as well as naive applications of un-trained networks. Most importantly, we achieve on-par performance with a standard trained baseline, the U-net, on the FastMRI dataset, a dataset for benchmarking deep learning based reconstruction methods. While state-of-the-art trained methods still outperform our un-trained method, our work demonstrates that current trained methods only achieve a minor performance gain over un-trained methods, at the cost of a loss in robustness to out-of-distribution examples. Therefore, un-trained neural networks are a serious competitor to trained ones for medical imaging.

## 1 Introduction

CNNs are highly successful tools for image reconstruction tasks. In recent years, a large number of works have shown that they can outperform traditional image processing methods for tasks such as image denoising, compressive sensing, and image compression [Bor+17; The+17; Tod+15; Agu+17; BSH12]. Almost exclusively, CNNs are trained on large sets of images, and their success is typically attributed to their ability to learn from those training images.

However, starting with the Deep Image Prior (DIP) [UVL18], a number of works have demonstrated that the architecture of a CNN can act as a sufficiently strong prior to enable image reconstruction, even without any training data. Specifically, un-trained networks perform well for denoising [UVL18; HH19], compressive sensing [Vee+18; ARL20], phase retrieval [JH19; Bos+20; Wan+20], and even for reconstructing videos [Jin+19; HA20]. Moreover, they provably succeed in denoising and reconstructing smooth signals from few measurements [HS20a; HS20b]. They have, however, mainly been studied in relatively controlled setups (such as denoising with Gaussian noise), and their performance in practical medical imaging problems is relatively unexplored.

In this work, we consider the problem of accelerating Magnetic Resonance Imaging (MRI) with un-trained neural networks. MRI is an important medical imaging technique and is extremely popular because it is non-invasive. However, performing an MRI scan is slow due to physical limitations of the scanning process. These limitations have led to a line of research known as compressed sensing which focuses on accelerating MRI by reconstructing an image from a few measurements, which in turn results in a faster scan time.

Traditional Compressed Sensing (CS) methods are based on $\ell_1$-norm minimization and Total-Variation (TV) norm minimization [BUF07] and are un-trained, i.e., they do not rely on any training data. Those methods are implemented in modern MRI scanners, but are outperformed by an emerging class of deep-learning-based reconstruction techniques, as demonstrated for example by the FastMRI challenge [Zbo+18], a competition for accelerated MRI reconstruction.

Off-the-shelf un-trained neural networks have been applied to accelerated MRI reconstruction by Arora et al. [ARL20] and by our group [Hec19] before, and have shown promising improvements over traditional

un-trained methods. At the same time, those two works reported visibly worse images than those produced by the state-of-the-art trained neural networks. The goal of this paper is to develop an un-trained neural network based reconstruction specifically tailored to MRI with state-of-the-art imaging performance. We find that, perhaps surprisingly, un-trained networks perform on par with a baseline trained method, namely U-net-based reconstruction. As a consequence, they achieve performance close to state-of-the-art trained methods that outperform the U-net. We achieve this performance through a number of small improvements in architecture and how we apply the network for image reconstruction that collectively give a significant improvement over naive application of un-trained networks. In more detail, our contributions are:

- We show that un-trained neural networks significantly outperform traditional un-trained methods (i.e., TV-minimization) and, perhaps surprisingly, have on-par performance with the U-net [RFB15] trained on the FastMRI dataset. This suggests that there is less benefit in learning-based approaches for imaging, at least in the context of MRI, than currently thought.
- We propose a variation of the Deep Image Prior (DIP) [UVL18] and Deep Decoder [HH19] architectures, called ConvDecoder. We show that this architecture performs best for knee MRI images and similar to Deep Decoder and DIP for brain MRI images, which have less detail. The ConvDecoder is a simple convolutional generator comprised only of up-sampling, convolution, batch normalization, and ReLU blocks in each layer.
- We propose a number of small optimizations on how to perform image recovery with un-trained neural networks such as i) introducing a data consistency step, ii) including or not including sensitivity maps in the reconstruction process, and iii) combining reconstructions of different runs that taken together give a significant improvement over naive application of the un-trained networks. We show that taken together, these optimizations result in a significant performance improvement.
- Finally, we demonstrate that a key advantage of un-trained neural networks over trained ones is robustness to out-of-distribution examples: At inference time, trained methods–unlike un-trained methods–are often sensitive to a shift in the distribution, as demonstrated in Section 7.

As a baseline for trained methods, we take U-net based reconstruction, because of its popularity and ease of use. We emphasize that *our method operates without any training data*, while the U-net is trained. The U-net is a baseline in the FastMRI challenge [Zbo+18]. We note, however, that the U-net is (slightly) outperformed by the newer approaches including end-to-end variational neural network [Sri+20a], Pyramid Convolutional Recurrent Neural Network (PCRNN) [Wan+19], Cascade net [Sch+17], and invertible Recurrent Inference Machines (i-RIM) [PW19].

## 2 Problem statement: Accelerated multi-coil MRI

We consider accelerated multi-coil MRI. Our goal is to recover an image $\mathbf{x} \in \mathbb{C}^N$ from a set of measurements. The measurements are obtained as

$$\mathbf{y}_i = \mathbf{MFS}_i\mathbf{x} + \text{noise}, \quad i = 1, \dots, n_c,$$

where $\mathbf{S}_i$ is a complex-valued position-dependent sensitivity map, that is applied through entry-wise multiplication to the image $\mathbf{x}$, $\mathbf{F}$ implements the 2D discrete Fourier transform, $n_c$ is the number of magnetic coils, and $\mathbf{M}$ is a mask that implements under-sampling. The measurements $\mathbf{y}_i$ are called $k$-space measurements.

First, suppose that we are only given one measurement, this is called single-coil imaging. Also, suppose that the sensitivity map is equal to identity, and that the mask also corresponds to the identity, i.e., we are given a single measurement $\mathbf{y} = \mathbf{Fx} + \text{noise}$. In this case, we can estimate the image up to the uncertainty of the additive noise as $\hat{\mathbf{x}} = \mathbf{F}^{-1}\mathbf{y}$.

In accelerated imaging, the $k$-space is under-sampled which is modeled through multiplication with the mask $\mathbf{M}$ which simply sets some of the frequencies of the $k$-space measurement to zero. Under-sampling

by a factor of $K$ results in a scan speed-up by the same factor. Reconstruction from the under-sampled measurements amounts to estimating an image from few measurements which is known as compressed sensing.

The practically most relevant problem is multi-coil reconstruction. In multi-coil imaging, each of the coils results in a different measurement. The sensitivity maps which determine those measurements are typically not given, but can be estimated from the under-sampled measurements. In this paper we consider the problem of reconstructing an image from under-sampled multi-coil measurements.

We work with the recently-released FastMRI dataset [Zbo+18]. The FastMRI dataset consists of a training and validation set each consisting of full $k$-space measurements of knees taken with $n_c = 15$ coils, and of brains taken with a varying number of coils. The dataset also contains "reference" images which are obtained by reconstructing the coil images from each coil measurement as $\hat{\mathbf{x}}_i = \mathbf{F}^{-1}\mathbf{y}_i$ and then combining the coil images via the root-sum-of-squares (RSS) algorithm to a final single image:

$$\hat{\mathbf{x}} = \sqrt{\sum_{i=1}^{n_c} |\hat{\mathbf{x}}_i|^2}. \tag{1}$$

Here, $|\cdot|$ and $\sqrt{\cdot}$ denote element-wise absolute value and squared root operations. Because different coil sensitivities overlap only little, the RSS algorithm works well for combining the images.

We consider accelerated imaging by obtaining measurements with a mask. We utilize a standard 1D variable-density mask (i.e., random or equi-spaced vertical lines across the $k$-space), because those masks are challenging but practically most relevant, and are the default in the FastMRI challenge. For evaluation, we compare to the "reference" images reconstructed from the full $k$-space.

# 3 Image recovery with un-trained neural networks

In this section, we discuss image recovery with un-trained neural networks. We view un-trained neural networks as convolutional image priors mapping a parameter space to an image space, i.e., $G\colon \mathbb{R}^p \to \mathbb{R}^{c\times w\times h}$, where $c$ is the number of channels of the output image (for example $c = 1$ for single grayscale image), and $w$ and $h$ are the width and height of the output image, respectively.

Deep-learning-based image models are typically trained; specifically, they are parameterized functions mapping an input to an output, with trainable (weight) parameters. For such trained image priors, the input parameterizes the image, the output is an image, and the weights are the trainable parameters. In contrast, an un-trained neural network is an image model where the input to the network is relatively irrelevant and fixed, the weights are the parameters of the model, and the output of the model is again an image.

As mentioned before, un-trained neural networks have been proposed first for image restoration problems by Ulyanov et al. [UVL18]. Ulyanov et al. [UVL18] proposed to use a simple U-net architecture consisting of an encoder, decoder, and skip connections for image reconstruction by fitting this architecture to a measurement. It has relatively quickly been realized that the encoder and the skip connections are irrelevant for performance and the decoder can even be further simplified [HH19]. We discuss architectures in Section 4.

## 3.1 Single-coil reconstruction

To illustrate how images are recovered with an un-trained neural network, we start with single-coil reconstruction. Let $\mathbf{y} \in \mathbb{C}^M$ be the under-sampled $k$-space measurement and let $\mathbf{M}$ and $\mathbf{F}$ be the mask and Fourier transform defined in Section 2 that maps an image to a measurement. Given an un-trained neural network $G\colon \mathbb{R}^p \to \mathbb{R}^{c\times w\times h}$ with parameter vector $\mathbf{C} \in \mathbb{R}^p$, we estimate an image based on the measurements $\mathbf{y}$ by first minimizing the mean-squared loss function

$$\mathcal{L}(\mathbf{C}) = \frac{1}{2}\|\mathbf{y} - \mathbf{M}\mathbf{F}G(\mathbf{C})\|_2^2, \tag{2}$$

with an iterative first order method to obtain the estimate $\hat{\mathbf{C}}$, and second, estimating the image as $\hat{\mathbf{x}} = G(\hat{\mathbf{C}})$. The image consists of a real and imaginary part, each described by one channel, therefore $c = 2$. As will become clear later, the choice of optimization algorithm matters: The network together with the optimization acts as a prior.

### 3.2 Multi-coil accelerated MRI reconstruction

In multi-coil imaging, multiple magnetic coils take different $k$-space measurements of the same image, and thus there are a variety of ways to use un-trained methods for image reconstruction. We found the following two to work best in different problem setups (i.e., one works best for brain images, the other for knee images); the first reconstructs an image with estimated sensitivity maps, and the second works without sensitivity map estimates.

**Step 1-A: Reconstruction without coil sensitivity maps** The perhaps most straightforward way to recover the image is to treat each measurement/coil independently, reconstruct as described in the single-coil reconstruction section above, and then combine the images using the RSS algorithm in 1.

A better performing and computationally more efficient approach is to impose the same prior to all images pertaining to the coils. This approach was first used by Arora et al. [ARL20]. Here, the first two output channels of the un-trained network generate the real and imaginary parts of the first image, the second two channel generate the the real and imaginary parts of the second image and so on. The final single image is then obtained with the RSS algorithm in 1. The loss function pertaining to this method is

$$\mathcal{L}(\mathbf{C}) = \frac{1}{2}\sum_{i=1}^{n_c} \|\mathbf{y}_i - \mathbf{M}\mathbf{F}G_i(\mathbf{C})\|_2^2, \tag{3}$$

where $G_i(\mathbf{C})$ is the reconstructed image associated with the measurements from the $i$-th coil $\mathbf{y}_i$. We found that using a single image prior for all images as proposed here gives slightly better reconstruction quality relative to reconstructing image by image and is significantly more efficient (approximately $n_c$-times faster).

**Step 1-B: Reconstruction with sensitivity map estimates** Instead of reconstructing all coil images separately, this method reconstructs the final output image directly, and takes the sensitivity maps $\hat{\mathbf{S}}$ estimated from the under-sampled data by applying ESPIRiT [Uec+14] into account. Specifically, the loss function is

$$\mathcal{L}(\mathbf{C}) = \frac{1}{2}\sum_{i=1}^{n_c} \left\|\mathbf{y}_i - \mathbf{M}\mathbf{F}\hat{\mathbf{S}}_i G(\mathbf{C})\right\|_2^2. \tag{4}$$

Note that the convolutional generator $G(\mathbf{C})$ has only two output channels, corresponding to the final image (since we reconstruct real and imaginary parts of the image in two separate channels).

**Step 2: Enforcing data consistency** After fitting the network by minimizing the respective loss function, we enforce data constancy. Specifically, recall that the measurement consists of under-sampled frequencies of the original image. We enforce the frequency components that we are given to conincide with the original measurements.

## 4 Network architectures

In this section, we discuss the network architectures we consider in this paper. All of the architectures we considered for MRI reconstruction are convolutional image-generating networks that map an input volume to an output. We choose a fixed input (specifically, we choose it randomly at initialization) and optimize

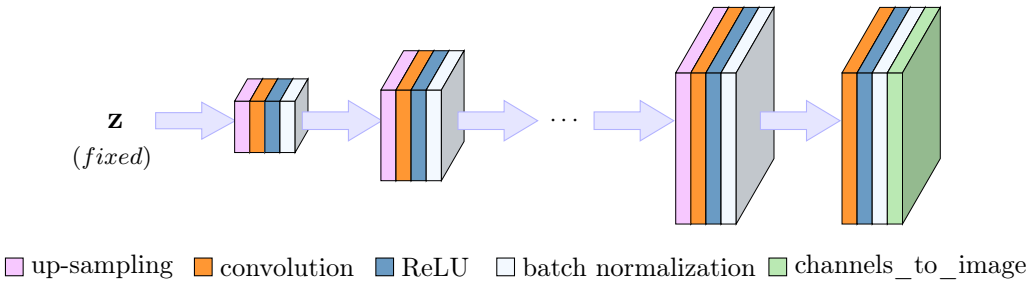


Figure 1: ConvDecoder architecture. It is comprised of up-sampling, convolutional, ReLU, batch normalization, and linear combination layers.

over the weights of the network. The DIP is the first and most popular architecture [UVL18], and consists of an encoder, decoder and skip connections.

The Deep Decoder [HH19] is a simple decoder architecture, even simpler than the decoder part of the DIP. Specifically, it only consists of convolutional operations with fixed convolutional filters followed by linear combinations (i.e., 1x1 convolutions).

In this paper, we found a variation of the Deep Decoder architecture, that we call ConvDecoder to perform best in most instances, in some similar to the original Deep Decoder, as discussed later. We also tried a variety of other architectures, including combinations of ConvDecoders that reconstruct an image at different resolutions, but again found the simple ConvDecoder to perform best (more details on the architectures we analyzed can be found in Appendix B).

Both the Deep Decoder and the ConvDecoder are convolutional neural network mapping a parameter space to images, i.e., $G\colon \mathbb{R}^p \to \mathbb{R}^{c\times w\times h}$, where $c$ is the number of output channels, and $w$ and $h$ are the width and height of the image in each channel. In each layer, except the last one, the network is composed of the following components: up-sampling, a convolutional operation, ReLU activation function, and finally a Batch Normalization (BN) block [IS15]. BN normalizes each channel of its input volume independent of other channels. The Deep Decoder uses bi-linear upsampling and 1x1 covolutions, while the ConvDecoder uses Nearest-Neighbor up-sampling and a 3x3 convolutional layer. Figure 1 depicts the network.

The parameters of the convolutional layers (and BN) are optimized when fitting the network to the given under-sampled measurement. The final layer excludes the up-sampling layer and simply combines the images via a 1x1 convolutional layer that performs linear combinations of the channels.

Convolutional and up-sampling blocks are essential. The former is responsible for capturing local information among the pixels and refines that information from one layer to another. The up-sampling block induces a notion of resolution into each layer as elaborated in Appendix C. Note that an $(n+1)$-layer ConvDecoder outputs an image $\hat{\mathbf{x}} \in \mathbb{R}^{c_{n+1}\times w_{n+1}\times h_{n+1}}$ from a fixed input $\mathbf{z} \in \mathbb{R}^{c_0\times w_0\times h_0}$, which we drawn from a Gaussian distribution and then fix. The default architecture we consider (upon which we slightly tune for each experiment) has 7 layers (including the last layer) and 256 channels per layer.

# 5 Performance of un-trained neural networks for MRI

In this section, we study the performance of un-trained neural networks for multi-coil 4x and 8x accelerated MRI reconstruction. We focus on multi-coil reconstruction, because multi-coil is clinically more relevant than single-coil reconstruction. After discussing the reconstruction evaluation challenges (Section 5.1), we provide the evaluation results for 4x accelerated knee measurements (Section 5.2.1), 8x accelerated knee measurements (Section 5.2.2), and finally 4x accelerated brain measurements (Section 5.2.3).

Our main findings are: (i) perhaps surprisingly, our un-trained networks perform as well as a standard baseline *trained* method, the U-net, for accelerated MRI—but without any training data, and (ii) our un-trained network significantly outperforms other un-trained methods, in particular total-variation minimization (TV), a standard baseline method.

We performed a grid search to find the best parameters for each method; for the ConvDecoder, that resulted in an 8-layer network with 256 channels in each layer. More details on the architecture setup and optimization parameters can be found in Appendix B. We also include a discussion in Appendix B.4 on how the results depend on the initialization and the choice of hyper-parameters.

## 5.1 Evaluating reconstruction performance

It is surprisingly non-trivial to compare different reconstruction methods for MRI. We have faced the following challenges and we have made the following choices for measuring the reconstruction performance:

**Image comparison metrics** Popular image metrics between a ground-truth image and a reconstructed image, like peak-signal-to-noise ratio (PSNR), are often unsuitable to capture reconstruction performance: Mason et al. [Mas+19] investigated a number of metrics based on the diagnostic quality of MR images according to the feedback given by a group of five radiologists. Their study shows that Visual Information Fidelity (VIF) [SB06] is generally a better metric for assessing diagnostic quality than widely-used metrics such as PSNR and Structural Similarity Index (SSIM) [Wan+04]. Interestingly, PSNR and SSIM—the perhaps most widely-used metrics for assessing image quality–were ranked among the most inept metrics in this study. To conform with common practice, we nevertheless include PSNR and SSIM metrics along with VIF and Multi-Scale SSIM (MS-SSIM) [WSB03].

**Normalization** It is often necessary to normalize images when comparing them to a ground truth image. Marcin et al. [MMR20] have shown that image normalization techniques have a significant impact on various texture features being extracted from a medical image. We chose mean-std normalization (applied to the ground-truth image to match the statistical properties of the reconstructed image) because the resulting scores were more consistent with the literature.

**Comparison to noisy ground truth** In some cases, the ground-truth image itself is corrupted with measurement artifacts. Hence, even if the reconstructed image is of high quality, the score might not reflect that and this might result in difficulties of comparing different reconstructions. We therefore show a few example reconstruction along with the reconstruction scores.

**Volume- vs. image-based comparison** Finally, all the afore-mentioned metrics are sensitive to whether the comparison is done on an image-based level, or is volume-based. Specifically, each scan of a knee or brain consists of a number of slices. Each slice is an image and together those images form the volume. All aforementioned metrics depend on the dynamic range of the values, therefore computing scores in a volume or image wise fashion gives different values. It is common in the literature [Wan+19; RCS20; Sri+20b; Sri+20a] to compute scores in a volume-based manner, that is, the dynamic range of the volume is considered for computing the scores. However, since images within each volume are analyzed independently, a more reasonable approach is to consider the dynamic range of each image separately.

To illustrate the point that volume- and image-based evaluations give very different numbers, Table 1 provides average volume-based as well as image-based SSIM and PSNR scores for ConvDecoder, U-net, and TV. Note that the results are averaged over 20 randomly-chosen volumes (640 slice images in total) from the validation set. The numbers show that the image-based score computation results in a lower range of numbers compared with the volume-based one, yet we employ the former because we think it better reflects the performance. In any case, the ranking of algorithms is typically roughly preserved when transitioning from image- to volume-based evaluation. We refer the reader to Appendix A for further discussion on the mentioned evaluation challenges.

| Method | Volume-based | | Image-based | |
|---|---|---|---|---|
| | SSIM | PSNR | SSIM | PSNR |
| ConvDecoder | 0.8713 | 35.61 | 0.7868 | 31.81 |
| U-net | **0.8753** | **35.68** | **0.7992** | **32.14** |
| TV | 0.7214 | 33.21 | 0.6832 | 30.12 |

Table 1: Volume-based vs. image-based score compution leads to different numbers: Average scores for ConvDecoder, U-net, and TV. For comparing volume- vs. image-based evaluation, SSIM and PSNR scores are averaged over 20 randomly-chosen volumes (640 slices).

## 5.2 Evaluation results

### 5.2.1 4x accelerated multi-coil knee measurements

We evaluate the performance of ConvDecoder along with other methods on the 4x accelerated multi-coil knee measurements of the FastMRI dataset. We consider five methods (four un-trained and one trained), specifically (i) ConvDecoder, (ii) Deep Decoder, (iii) DIP, (iv) a standard un-trained TV-norm minimization method, and finally (v) the U-net, a standard trained method. We did an extensive grid search to select the best parameters for each un-trained neural network. See Appendix B for details and for the parameter setup for each method.

We start by comparing ConvDecoder with Deep Decoder and DIP. To compare the aforementioned un-trained networks, we computed the scores by averaging over 20 randomly-chosen mid-slice images of different volumes in the validation set, and as mentioned earlier, the scores are computed in an image-based manner; see Table 2 for those scores and Figure 2 for sample reconstructions. From Figure 2, it can be seen that DIP induces a noticeable amount of vertical artifacts. Moreover, Deep Decoder–in addition to having reconstruction artifacts–tends to generate rather smooth images and therefore misses texture details. These differences are reflected in other images as well, and hence by averaging the scores over 20 images, in Table 2, we observe a gap between ConvDecoder scores and the scores we obtain from DIP and Deep Decoder.

| Method | VIF | MS-SSIM | SSIM | PSNR |
|---|---|---|---|---|
| ConvDecoder | **0.6717** | **0.9443** | **0.7775** | **31.81** |
| DIP | 0.6311 | 0.8981 | 0.5938 | 28.40 |
| DD | 0.6359 | 0.8599 | 0.6991 | 29.16 |

Table 2: Average image-based scores for the ConvDecoder, DIP, and Deep Decoder (DD) on the mid-slice images of 20 randomly-chosen volumes in the multi-coil knee measurements from the FastMRI validation set (4x accelerated). ConvDecoder significantly outperforms DIP and DD according to all metrics.

Based on Table 2, we conclude that ConvDecoder performs best for knee MRI images. Therefore, in the rest of the knee experiments, we consider ConvDecoder and compare it to the baselines.

**ConvDecoder without training performs as well as a trained U-net and significantly better than an un-trained baseline** Next, we compare ConvDecoder with U-net and TV. We trained a standard U-net with 8 layers and width factor equal to 32 on the whole training set (974 volumes). To compute the results on the whole validation set, we ran the three mentioned methods on the mid-slice image of each volume in the validation set (a set of 200 images over all).

Table 3 shows the results for each method. The scores show that the ConvDecoder has higher VIF performance than U-net (recall that VIF is deemed most relevant by clinicians), and achieves on-par performance with the U-net according to all metrics. Moreover, it significantly outperforms TV. Figure 3 shows a sample reconstruction for an image from the validation set for the mentioned methods.

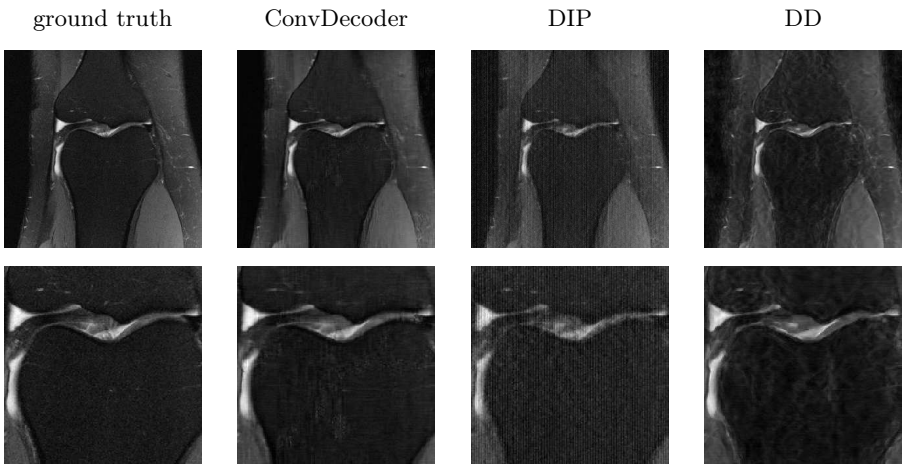


Figure 2: Sample reconstructions for ConvDecoder, DIP, and Deep Decoder (DD) for a validation image from multi-coil knee measurements (4x accelerated). The second row represents zoomed-in version of the first row. ConvDecoder gives the best reconstruction for this image.

| Method | VIF | MS-SSIM | SSIM | PSNR |
|---|---|---|---|---|
| ConvDecoder | **0.6823** | 0.9387 | 0.7753 | 31.67 |
| U-net | 0.5955 | **0.9489** | **0.7883** | **32.04** |
| TV | 0.4412 | 0.9262 | 0.6977 | 30.20 |

Table 3: For comparing different methods on the whole dataset (in an image-based manner), 200 mid-slice images from 200 volumes have been considered. All data points are chosen from multi-coil knee measurements of the FastMRI validation set (4x accelerated). ConvDecoder performs best in the VIF metric deemed most relevant by Clinicians. In the other metrics, ConvDecoder and U-net are extremely similar.

#### 5.2.2 8x accelerated multi-coil knee measurements

For 8x acceleration, we found different hyper-parameters of the ConvDecoder (relative to 4x acceleration) to work best. To find good hyper-parameters for the ConvDecoder, we again performed a grid search—the details of which is provided in Appendix B—which resulted in a ConvDecoder with 6 layers, 64 channels, and $(4, 4)$ as the input size.

| Method | VIF | MS-SSIM | SSIM | PSNR |
|---|---|---|---|---|
| ConvDecoder | **0.5234** | 0.8827 | 0.6815 | 28.49 |
| U-net | 0.5233 | **0.9148** | **0.7115** | **29.25** |
| TV | 0.3119 | 0.8340 | 0.5986 | 26.55 |

Table 4: Average image-based scores for the ConvDecoder, U-net, and TV on the mid-slice images of 200 volumes in the multi-coil knee measurements from the FastMRI validation set (8x accelerated). ConvDecoder achieves on-par performance with U-net and outperforms TV.

Table 4 shows that ConvDecoder achieves similar performance than U-net (according to all metrics except SSIM and MS-SSIM which U-net slightly outperforms ConvDecoder) and significantly outperforms TV. Figure 4 shows a sample reconstruction for the three mentioned methods. Those experiments for 8x acceleration second our finding for 4x acceleration that the structure of the network is a sufficient prior for the MRI compressed sensing task, at least when considering a baseline trained method (U-net) for comparison.

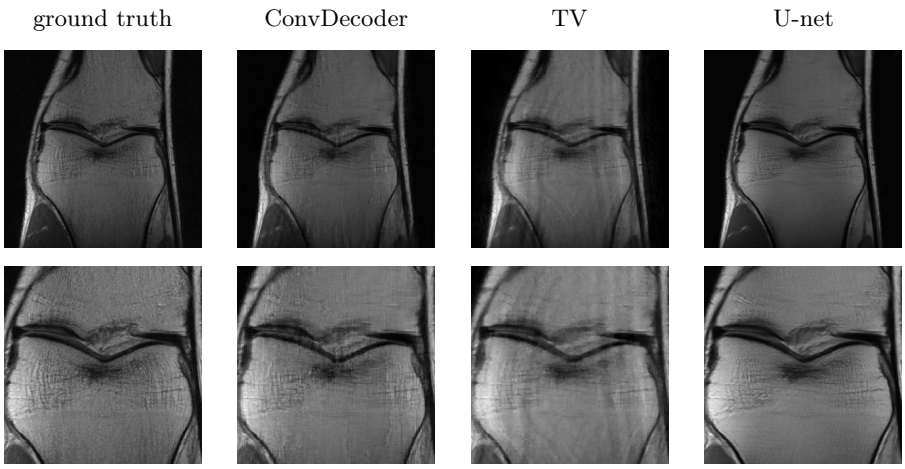


Figure 3: Sample reconstructions for ConvDecoder, TV, and U-net for a validation image from multi-coil knee measurements (4x accelerated). The second row represents zoomed-in version of the first row. ConvDecoder finds the best reconstruction for this image (slightly better than U-net and significantly better than TV).

#### 5.2.3 4x accelerated multi-coil brain measurements

So far, we have shown that un-trained neural networks perform surprisingly well for knee MRI reconstruction. Specifically, they perform on-par with a baseline trained neural network and significantly better than a baseline un-trained method (TV). In this section, we perform a similar comparison for brain images to fortify the claim that un-trained neural networks can be practically used for accelerated MRI reconstruction.

Similar to Section 5.2.1, we first compare different un-trained neural networks and then compare the best-performing one to the baselines. We again performed an extensive grid search and the resulting hyper-parameters are in Appendix B.

Table 5 shows average scores for the ConvDecoder, DIP, and DeepDecoder. Interestingly, all three un-trained networks perform similar on the brain images and unlike knee, there is not a significant difference among them. However, we emphasize the role of our data consistency step which resulted in approximately %10 SSIM score improvement for these un-trained networks.

| Method | VIF | MS-SSIM | SSIM | PSNR |
|---|---|---|---|---|
| ConvDecoder | 0.8002 | **0.9743** | 0.9018 | 34.58 |
| DIP | **0.8223** | 0.9736 | 0.8918 | **34.96** |
| DD | 0.7952 | 0.9713 | **0.9023** | 34.52 |

Table 5: Average image-based scores for the ConvDecoder, DIP, and Deep Decoder (DD) on the mid-slice images of 20 randomly-chosen volumes in the multi-coil brain measurements from the FastMRI validation set (4x accelerated). All three networks perform similar on the brain images.

Since there is not a noticeable difference among ConvDecoder, DIP, and Deep Decoder on the brain images, we proceed with the ConvDecoder for comparison to baselines, to be consistent with the previous sections. However, ConvDecoder can be replaced with the Deep Decoder or DIP architectures in the following comparison with similar results; albeit the DIP has a larger computational overhead.

During tuning, we noticed that Step 1-B—described in Section 3.2, which takes the estimates of the coil sensitivity maps into account instead of using Step 1-A which does not, works significantly better. To quantify the gains of employing the estimated coil sensitivity maps, we include ConvDecoder scores with/without sensitivity maps along with U-net and TV in Table 6. The numbers are averaged over 100 validation brain images.

Figure 5 depicts sample reconstructions for ConvDecoder-SM (Convdecoder + sensitivity maps), Con-

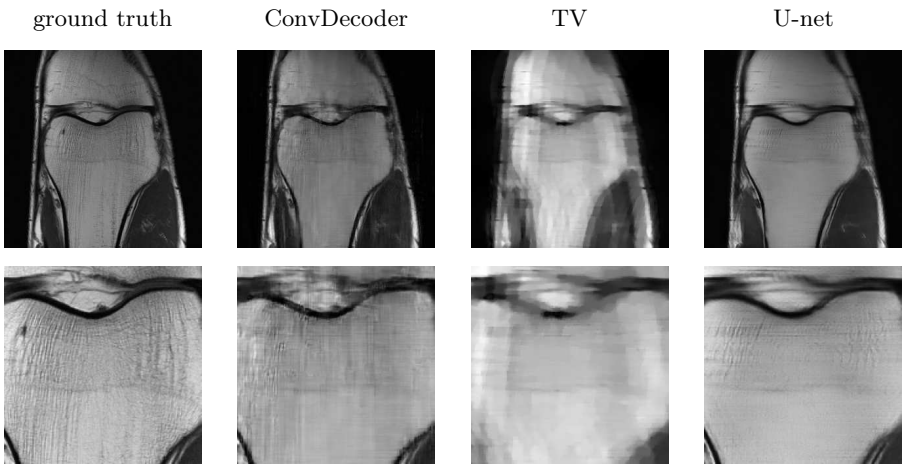


Figure 4: Sample reconstructions for ConvDecoder, TV, and U-net for a validation image from multi-coil knee measurements (8x accelerated). The second row represents zoomed-in version of the first row. ConvDecoder finds a similar reconstruction as U-net and outperforms TV.

| Method | VIF | MS-SSIM | SSIM | PSNR |
|---|---|---|---|---|
| ConvDecoder | 0.8178 | 0.9722 | 0.8892 | 33.06 |
| ConvDecoder-SM | 0.8707 | **0.9804** | 0.9054 | **34.32** |
| U-net | **0.8714** | 0.9792 | **0.9173** | 34.29 |
| TV | 0.5932 | 0.9364 | 0.6998 | 29.52 |

Table 6: Average image-based scores for the ConvDecoder, ConvDecoder-SM (ConvDecoder + sensitivity maps), U-net, and TV on the mid-slice images of 100 volumes in the multi-coil brain measurements from the FastMRI validation set (4x accelerated). ConvDecoder with sensitivity maps achieves on-par performance with U-net and outperforms TV.

vDecoder, U-net, and TV for a validation image. Note the significant improvement in the reconstruction quality as a result of using estimated sensitivity maps. Since ConvDecoder-SM and U-net are performing best in reconstructing the shown sample, we also annotated two specific parts on their reconstructions to point out their differences. The blue circle denotes a region where U-net is giving a sharp reconstruction, while ConvDecoder-SM yields a smooth reconstruction. The red circle on the other hand, denotes a region fully recovered by ConvDecoder-SM, whereas U-net has merged the two black points in the region.

# 6 Better performance at the cost of more computations

According to Sections 5.2.1–5.2.3, ConvDecoder significantly outperforms TV and achieves performance close to U-net. In this section, we propose an approach to obtain an even better performance with ConvDecoder. Consider an under-sampled measurement $\mathbf{y}$ and $k$-many ConvDecoders with the same hyper-parameters, but initialized independently. After fitting each decoder to $\mathbf{y}$, we average the resulting $k$ reconstructed images which results in a reconstruction of higher quality. This averaging technique tends to remove small reconstruction artifacts in each run and hence gives a better output image.

Table 7 shows PSNR scores when using this averaging technique. The numbers are averaged over 10 random proton-density images from the 4x accelerated knee measurements from the FastMRI validation set. Note that if we run ConvDecoder once, the estimated PSNR score for an image according to Table 7 would be the average run (32.46 dB), that is the averge score of all 10 runs. However, the image averaged over 10 runs results in a reconstruction with 1 dB higher PSNR than the best image of all 10 runs.

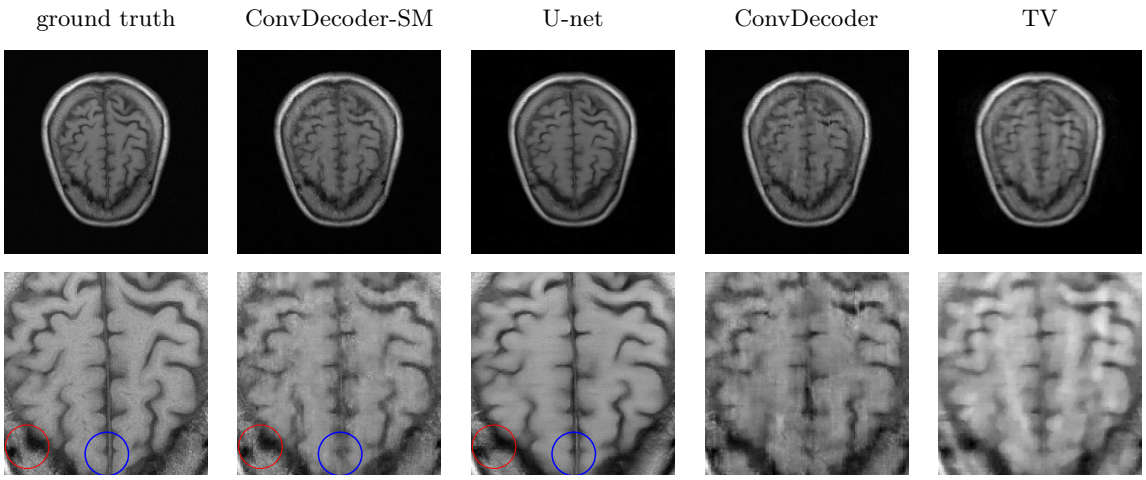


Figure 5: ConvDecoder-SM, without any training data, yields noticeable improvement over ConvDecoder for brain MRI images, significantly outperforms TV (a traditional compressed sensing method), and performs on-par with U-net (a trained neural network). Images in the second row are zoomed-in versions of the first row.

How many runs are enough to achieve this higher performance? For a given image, Figure 6 shows how PSNR changes based on the number of ConvDecoders whose outputs are averaged together to form the final reconstruction (the numbers are averaged over the same 10 images as in Table 7). It is interesting that even averaging over the output of two decoders has a noticeable impact (more than 0.5 dB) on the reconstruction quality. Note that the improvement rate decreases as we increase the number of decoders.

| Method | PSNR |
|---|---|
| average image | **33.70** |
| average run | 32.46 |
| best run | 32.71 |

Table 7: PSNR scores for a sample image reconstructed by ConvDecoder. *average image* denotes the image resulted by averaging the output of 10 ConvDecoders, *average run* denotes the average PSNR score of 10 runs, and *best run* denotes the best reconstruction among all 10 runs according to PSNR.

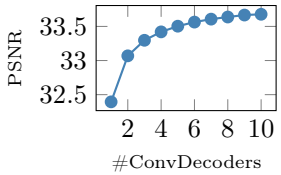


Figure 6: PSNR scores for an image resulted by averaging the output of $k \in \{1, 2, ...10\}$ number of ConvDecoders. This averaging technique results in a significant improvement over the PSNR score.

## 7 Robustness to out-of-distribution samples

In Section 5.2, we found that ConvDecoder achieves on-par performance with U-net when evaluating them on the in-distribution samples (i.e., FastMRI validation set). Another aspect to consider for comparing U-net and ConvDecoder is their robustness to out-of-distribution samples.

Un-trained neural networks are robust to a shift in the distribution, as they are *un-trained*, however their optimal hyper-parameters are typically tuned on a distribution of images. Trained methods, however, do not enjoy such robustness as they learn a prior from a data distribution. To illustrate this difference, we ran U-net (trained on 4x accelerated knee MRI measurements) and ConvDecoder on an out-of-distribution image (e.g., a non-MRI image). We used the same 4x accelerated mask (as in the MRI reconstruction

problem) for under-sampling the frequency-domain representation of this image. Figure 7 shows that while the ConvDecoder reconstructs the out-of-distribution image well, the U-net trained on knees introduces significant artifacts.

For a more principled comparison, we trained U-net on both knee and brain images, respectively to attain U-net$_{\text{knee}}$ and U-net$_{\text{brain}}$. Afterward, we tested both of them on the brain validation set. We observed that based on this evaluation, U-net$_{\text{knee}}$ obtains 3dB less PSNR than U-net$_{\text{brain}}$. Contrary, our un-trained method does not suffer any loss in performance. This experiment demonstrates how fragile trained neural networks are when it comes to a shift in the distribution, while un-trained networks are not.

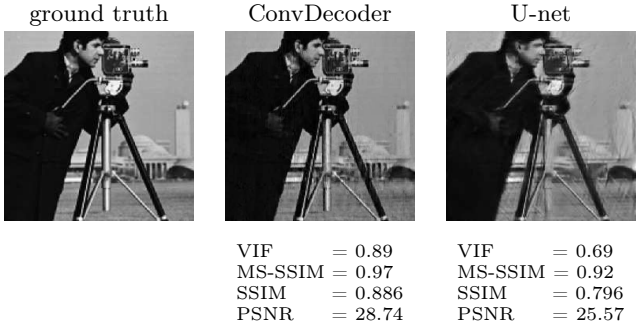


Figure 7: Un-trained methods are robust to a shift in the distribution. Both methods are run for 4x under-sampled (in the frequency domain) version of the cameraman test image. U-net is trained on 4x accelerated multi-coil knee measurements from the FastMRI dataset.

## 8 Discussion and Conclusion

MRI scanners are typically accelerated by under-sampling the measurements, and classically the images are reconstructed with an un-trained method by solving an optimization problem such as $\ell_1$-minimization or total-variation minimization. Those classical algorithms are significantly outperformed by deep learning approaches that learn to reconstruct an image from measurements. Thus, those learning based approaches enable higher accelerations. However, concerns have been raised over the stability of those methods and the ability to generalize to different scanners and distributions of images (e.g., if trained on knees, a neural network might perform poorly on brains as we have demonstrated).

In this work, we promote un-trained neural networks as a method that combines the best of those two worlds: it is un-trained and as such does not raise robustness concerns, and as shown in this paper, achieves performance that matches baseline trained neural networks. Specifically, in this paper we studied un-trained neural networks for accelerated MRI reconstruction and proposed an architecture that is most suitable for MRI. We find that this architecture significantly outperforms other un-trained methods–including traditional CS methods–and has image reconstruction performance close to that of trained neural networks. We further demonstrated that a key advantage of un-trained over trained neural networks is its robustness to distribution shifts. While the best trained neural networks still slightly outperform our un-trained methods in terms of reconstruction accuracy, the (i) robustness to distribution shift and (ii) not needing any training data make un-trained networks an important tool in practice, especially in regimes where there is a lack of training data, and a need for robustness.

## Reproducibility

The code to reproduce all results in this paper is available at https://github.com/MLI-lab/ConvDecoder.

## Acknowledgment

R. Heckel and M. Zalbagi Darestani are partially supported by NSF award IIS-1816986, and R. Heckel acknowledges support of the NVIDIA Corporation in form of a GPU, and is partially supported by the IAS at TUM.

The authors would like to thank Zalan Fabian for sharing his finding that averaging the outputs of multiple runs of deep decoders improves denoising performance, which led us to study the performance of averaging multiple ConvDecoder outputs (i.e., Section 6). The authors would also like to thank Lena Heidemann for helpful discussions and for running a grid search on the brain images for the Deep Decoder (i.e., Table 11).

## References

[Agu+17] E. Agustsson et al. "Soft-to-hard vector quantization for end-to-end learning compressible representations". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2017, pp. 1141–1151.

[ARL20] S. Arora, V. Roeloffs, and M. Lustig. "Untrained modified deep decoder for joint denoising parallel imaging reconstruction". In: *International Society for Magnetic Resonance in Medicine Annual Meeting*. 2020.

[BUF07] K. T. Block, M. Uecker, and J. Frahm. "Undersampled radial MRI with multiple coils. Iterative image reconstruction using a total variation constraint". In: *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*. 2007, pp. 1086–1098.

[Bor+17] A. Bora, A. Jalal, E. Price, and A. G. Dimakis. "Compressed sensing using generative models". In: *International Conference on Machine Learning (ICML)*. 2017, pp. 537–546.

[Bos+20] E. Bostan, R. Heckel, M. Chen, M. Kellman, and L. Waller. "Deep phase decoder: self-calibrating phase microscopy with an untrained deep neural network". In: *Optica*. 2020, pp. 559–562.

[BSH12] H. C. Burger, C. J. Schuler, and S. Harmeling. "Image denoising: can plain neural networks compete with BM3D?" In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2012, pp. 2392–2399.

[HH19] R. Heckel and P. Hand. "Deep Decoder: concise image representations from untrained non-convolutional networks". In: *International Conference on Learning Representations (ICLR)*. 2019.

[HS20a] R. Heckel and M. Soltanolkotabi. "Denoising and regularization via exploiting the structural bias of convolutional generators". In: *International Conference on Learning Representations (ICLR)*. 2020.

[Hec19] R. Heckel. "Regularizing linear inverse problems with convolutional neural networks". In: *arXiv preprint arXiv:1907.03100* (2019).

[HS20b] R. Heckel and M. Soltanolkotabi. "Compressive sensing with un-trained neural networks: Gradient descent finds the smoothest approximation". In: *International Conference on Machine Learning (ICML)*. 2020.

[HA20] R. Hyder and M. S. Asif. "Generative models for low-dimensional video representation and reconstruction". In: *IEEE Transactions on Signal Processing*. 2020, pp. 1688–1701.

[IS15] S. Ioffe and C. Szegedy. "Batch Normalization: accelerating deep network training by reducing internal covariate shift". In: *International Conference on Machine Learning (ICML)*. 2015, pp. 448–456.

[JH19] G. Jagatap and C. Hegde. "Algorithmic guarantees for inverse imaging with untrained network priors". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2019.

[Jin+19] K. H. Jin, H. Gupta, J. Yerly, M. Stuber, and M. Unser. "Time-dependent deep image prior for dynamic MRI". In: *arXiv:1910.01684 [eess.IV]*. 2019.

[KB15] D. P. Kingma and J. Ba. "Adam: a method for stochastic optimization". In: *International Conference on Learning Representations (ICLR)*. 2015.

[MMR20] K. Marcin, S. Michał, and O. Rafał. "Does image normalization and intensity resolution impact texture classification?" In: *Computerized Medical Imaging and Graphics*. 2020, p. 101716.

[Mas+19] A. Mason et al. "Comparison of objective image quality metrics to expert radiologists' scoring of diagnostic quality of MR images". In: *IEEE Transactions on Medical Imaging*. 2019, pp. 1064–1072.

[PW19] P. Putzky and M. Welling. "Invert to learn to invert". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2019, pp. 444–454.

[RCS20] Z. Ramzi, P. Ciuciu, and J. L. Starck. "Benchmarking deep nets MRI reconstruction models on the FastMRI publicly available dataset". In: *International Symposium on Biomedical Imaging*. 2020.

[RFB15] O. Ronneberger, P. Fischer, and T. Brox. "U-net: convolutional networks for biomedical image segmentation". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. 2015, pp. 234–241.

[Sch+17] J. Schlemper, J. Caballero, J. V. Hajnal, A. Price, and D. Rueckert. "A deep cascade of convolutional neural networks for MR image reconstruction". In: *International Conference on Information Processing in Medical Imaging*. 2017, pp. 647–658.

[SB06] H. R. Sheikh and A. C. Bovik. "Image information and visual quality". In: *IEEE Transactions on Image Processing*. 2006, pp. 430–444.

[Sri+20a] A. Sriram et al. "End-to-end variational networks for accelerated MRI reconstruction". In: *arXiv:2004.06688 [eess.IV]*. 2020.

[Sri+20b] S. Sriram, J. Zbontar, T. Murrell, C. L. Zitnick, A. Defazio, and D. K. Sodickson. "GrappaNet: combining parallel imaging with deep learning for multi-coil MRI reconstruction". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 14315–14322.

[The+17] L. Theis, W. Shi, A. Cunningham, and F. Huszár. "Lossy image compression with compressive autoencoders". In: *International Conference on Learning Representations (ICLR)*. 2017.

[Tod+15] G. Toderici et al. "Variable rate image compression with recurrent neural networks". In: *International Conference on Learning Representations (ICLR)*. 2015.

[Uec+14] M. Uecker et al. "ESPIRiT—an eigenvalue approach to autocalibrating parallel MRI: where SENSE meets GRAPPA". In: 2014, pp. 990–1001.

[UVL18] D. Ulyanov, A. Vedaldi, and V. Lempitsky. "Deep image prior". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 9446–9454.

[Vee+18] D. V. Veen, A. Jalal, M. Soltanolkotabi, E. Price, S. Vishwanath, and A. G. Dimakis. "Compressed sensing with deep image prior and learned regularization". In: *arXiv:1806.06438 [stat.ML]*. 2018.

[Wan+20] F. Wang et al. "Phase imaging with an untrained neural network". In: *Light: Science & Applications*. 2020, pp. 1–7.

[Wan+19] P. Wang, E. Z. Chen, T. Chen, V. M. Patel, and S. Sun. "Pyramid convolutional RNN for MRI reconstruction". In: *arXiv: 1912.00543 [eess.IV]*. 2019.

[Wan+04] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. "Image quality assessment: from error visibility to structural similarity". In: *IEEE Transactions on Image Processing*. 2004, pp. 600–612.

[WSB03] Z. Wang, E. P. Simoncelli, and A. C. Bovik. "Multiscale structural similarity for image quality assessment". In: *Asilomar Conference on Signals, Systems & Computers*. 2003, pp. 1398–1402.

[Zbo+18] J. Zbontar et al. "FastMRI: An open dataset and benchmarks for accelerated MRI". In: *arXiv: 1811.08839 [cs.CV]*. 2018.

# APPENDIX

## A Details on evaluating the reconstruction performance

As mentioned in the main body of the paper (Section V), evaluating the performance of different reconstruction methods is challenging for reasons related to the (i) choice of image comparison metrics, (ii) impact of the normalization of images, (iii) the fact that we often compare to a noisy ground truth image, and (iv) because we can compare image wise or volume wise. In this section, we further iterate points ii and iii.

**Normalization** As for the image normalization method, which is typically required to fairly compare two images, we investigated three image normalization methods: min-max normalization, which transforms image $I$ to $\frac{I-\min(I)}{\max(I)-\min(I)}$; mean-std normalization on both ground-truth and reconstructed images, which transforms image $I$ to $\frac{I-\text{mean}(I)}{\text{std}(I)}$; and mean-std normalization which is only applied to the ground-truth image to match its histogram to that of the reconstructed image.

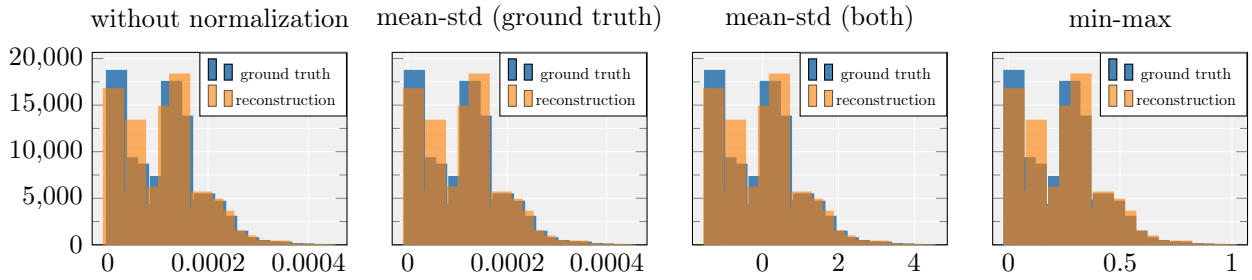


Figure 8: The effect of different image normalization techniques on the distribution of ground truth and reconstructed images.

| Method | no norm | | min-max | | mean-std (both) | | mean-std (gt) | |
|---|---|---|---|---|---|---|---|---|
| | SSIM | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM | PSNR |
| ConvDecoder | 0.7563 | 29.93 | **0.7464** | **30.09** | 0.6957 | **29.57** | 0.7753 | 31.67 |
| U-net | **0.7867** | **32.01** | 0.7354 | 28.04 | **0.7091** | 29.37 | **0.7883** | **32.04** |
| TV | 0.6592 | 27.21 | 0.6875 | 26.67 | 0.6565 | 28.43 | 0.6977 | 30.20 |

Table 8: Average image-based scores for the ConvDecoder, U-net, and TV on the mid-slice images of the volumes in the multi-coil knee measurements from the fastMRI validation set (4x accelerated). Scores are computed based on different image normalization methods. Surprisingly, different image normalization types result in a very different ranking of the reconstruction methods.

Figure 8 illustrates how each of the mentioned normalization methods affects the distribution of ground-truth and reconstructed images for a sample file from the multi-coil knee dataset. In addition, Table 8 shows the average SSIM and PSNR scores for ConvDecoder, Unet, and TV after running them on 200 mid-slice images from the multi-coil knee dataset (4x accelerated). The remarkable outcome of these results is that different image normalization methods can result in a totally different winning reconstruction method.

**Comparison to noisy ground truth** As mentioned, in some cased the ground-truth image itself is corrupted with measurements images, as illustrated in Figure 9 where the ground truth image is very noisy. In such cases, the scores may not reflect the true quality of the reconstructed image. Note that the reconstruction is almost free from noise, demonstrating the image prior also denoises the image.

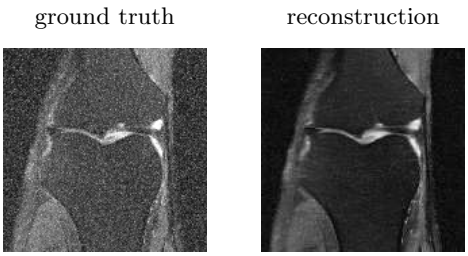


Figure 9: A sample image from the multi-coil knee dataset where the ground-truth image is very noisy. In the evaluation, we compare a reconstructed image to such a noisy image, and even if the reconstruction is a subjectively sharp and clear image, the error metric is large.

# B Parameter setup and optimization

## B.1 4x knee

We tuned the parameters of ConvDecoder, DIP, and Deep Decoder by performing a grid search over group of 10 randomly-chosen images. We also tried different architectures (e.g., adding skip connections, or stacking multiple decoders to form a multi-resolution network) but found the plain ConvDecoder to perform best.

Table 9 shows the grid parameters for each network. For the U-net (and also TV), we chose the set of parameters used in the FastMRI challenge[1] [Zbo+18]. The parameter setup is provided in Table 10.

| Method | layers | channels | residuals |
|---|---|---|---|
| ConvDecoder | {6, 7, 8, 9} | {160, 256, 480} | - |
| DIP | {12, 14, 16, 18} | {160, 256, 360} | {2, 4} |
| DD | {6, 7, 8, 9, 10} | {256, 368, 512} | - |

Table 9: Grid-search parameters for ConvDecoder, DIP, and Deep Decoder (DD).

| Method | #layers | #channels (or width) | convolutional kernel size | #residuals |
|---|---|---|---|---|
| ConvDecoder | 8 | 256 | 3 | 0 |
| U-net | 8 | 32* | 3 | 4 |
| DIP | 16 | 256 | 3 | 2 |
| DD | 10 | 368 | 1 | 0 |

* This is the number of channels for the first layer of U-net. For the 8-layer U-net that we used, the number of channels are [32, 64, 128, 256, 512, 256, 128, 64, 32] including a non-pooling layer in the middle.

Table 10: Model parameters for ConvDecoder, U-net, DIP, and Deep Decoder (DD).

In order to fit the un-trained methods to the under-sampled measurements, we used the Adam optimizer [KB15] with constant stepsize 0.01 (without early stopping) for optimizing the loss function (which is MSE loss function in our experiments).

Regarding the output dimension of un-trained methods, as an example, for a $15 \times 640 \times 368$ (15 is the number of coils) under-sampled measurement, ConvDecoder (also DIP or DD) generates an image of size $30 \times 640 \times 368$, because it recovers the real and complex pixel values of an image separately with two separate channels. Finally, we fixed the input of the un-trained methods which is sampled from $\mathcal{N}(0, I)$ and has dimension $256 \times 10 \times 5$ (256 being the number of channels).

[1] https://github.com/facebookresearch/fastMRI/tree/master/models

### B.2 8x knee

For 8x, our experiments reveal that finding a set of hyper-parameters that work well on multiple images is a more difficult task. Therefore, we did a more extensive grid search compared to 4x. We first performed a grid search over the same grid parameters as before (specifically as shown in Table 9 and we also considered 64 channels per layer and different input sizes).

We then sorted all sets of parameters based on all four metrics. Afterward, we picked 3 sets of parameters that appeared in the top-5 best-performing hyper-parameter configurations according to all metrics. Finally, we selected one out of the three best-performing hyper-parameter configurations by running the three hyper-parameter configurations on 6 more (randomly-chosen) images (the selected setup—which is 6 layers, 64 channels and $(4, 4)$ as the input size—outperformed the other two based on all metrics).

### B.3 4x brain

Similar to previous sections, here we also performed a grid search (Table 11) to find the hyper-parameters for each un-trained neural network. The chosen parameters are marked with an asterisk. The search as done on a group of 10 randomly-chosen images from brain validation set.

| Method | layers | channels | residuals |
|---|---|---|---|
| ConvDecoder | {4, 5*, 6, 7, 8} | {32, 64*, 128} | - |
| DIP | {10, 12*, 14} | {64*} | {4*} |
| DD | {6, 7, 8, 9, 10*, 11} | {64*, 128, 256} | - |

Table 11: Grid-search parameters for ConvDecoder, DIP, and Deep Decoder (DD).

### B.4 Sensitivity to initialization and choice of hyper-parameters

We next discuss (i) how the width of the network affects the reconstruction quality, and (ii) demonstrate that there is little variance in the scores given a specific setup when fitting the ConvDecoder starting from a random initialization to a given under-sampled measurement over multiple runs on the same problem.

A key hyper-parameter of the ConvDecoder is the width of the network (the number of channels per layer). In order to check how different wideness factors affect the performance, we ran a seven-layer ConvDecoder on three under-sampled measurements (again from the multi-coil accelerated knee measurements of the FastMRI dataset) and computed the SSIM score. We performed this experiment four times to average the results. Figure 10 (right) shows the SSIM score based on network width for the three mentioned data points. It can be seen that if the network width is either too small or too large, it does not perform well. A width parameter around 200 performs well across images.

Recall that to recover an image, we run gradient descent starting from a random initialization. It is natural to ask whether the reconstruction quality varies significantly as a function of the random initialization. We find that the reconstruction quality is relatively insensitive to the particular initialization. Specifically, for the general setup we used in Section B.1, we ran the ConvDecoder 10 times on an under-sampled measurement and averaged the scores. Figure 10 (left) depicts the variances of different scores over several runs of the algorithm, and illustrates that the scores vary relatively mildly (VIF as well as PSNR and MS-SSIM tend to have the highest and lowest variations, respectively).

## C How does ConvDecoder represent an image?

Despite the notable empirical success of un-trained neural networks for solving inverse problems, there is still little knowledge about why these methods work so well in practice. In this section, we illustrate *how* un-trained networks functions as image priors. Specifically we demonstrate (i) how different layers play a

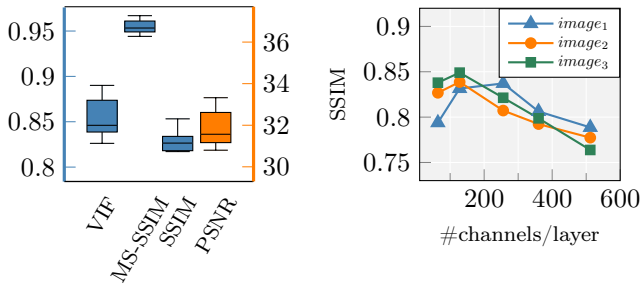


Figure 10: Effect of hyper-parameters on the ConvDecoder's performance. **Left**: fluctuations of different scores during 10 runs on a single data point. **Right**: effect of network width on the SSIM score for three data points from the validation set.

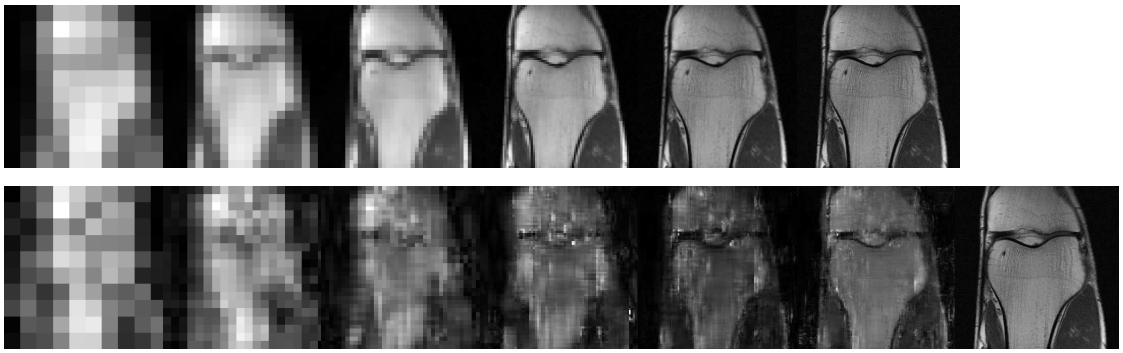


Figure 11: ConvDecoder finds an image representation by constructing fine details per each layer. The network is fitted for representing an image from the 4x under-sampled $k$-space of the ground-truth image in the mid row. **Top row:** different resolutions of the ground-truth image in the bottom row to each of which we fit the layer outputs. **Bottom row:** output visualization for each of the six layers.

role in forming successively higher resolution versions of an image and (ii) how different layers are fitted in the optimization, and why this matters.

## C.1 Successive approximation of an image

To understand the role of different layers in reconstructing the image, it is instructive to see how an untrained network generates an image by visualizing the outputs of each layer. Visualizing the layers' 256 channels, however, is not informative. Instead, we visualize the best representation that can be achieved by linearly combining the channels in each layer to the re-scaled ground-truth image. For example, if the image size (omitting the number of channels) in layer $i$'s output is $(w_i, h_i)$, then we down-sample the ground-truth image to match this size.

Figure 11 shows the results of our visualization method. The top row shows different resolutions of the ground-truth image $\mathbf{x}_1$. The bottom row shows the visualization results for a network fitted to reconstruct $\mathbf{x}_1$ from the 4x under-sampled measurements $\mathbf{y}_1$. We observe that: (i) ConvDecoder finds a fine representation of an image by adding more detail in each layer. (ii) As shown in Figure 11, we observe the role of up-sampling blocks in inducing the notion of resolution to the network, in that different layers represent reconstructions of different resolutions.

## C.2 It is critical to fit different layers at different speeds

The optimization method employed to minimize the loss function of an un-trained network has a significant impact on the quality of the reconstructed image because it determines which layer is fitted at which speed. The Adam optimizer yields a significantly better reconstruction than Gradient Descent (GD) both qualitatively and quantitatively (an approximately 3% higher SSIM score).

The reason behind this discrepancy in performance lies in the fact that Adam chooses different stepsizes (and hence different fitting speeds) for each parameter in the network, whereas GD treats all network parameters the same. It is important to (i) choose larger stepsizes for earlier layers and (ii) employ an increasing stepsize schedule, but it is not critical to choose the stepsize adaptively or differently within a layer.

To illustrate this point, we recorded layer-wise stepsizes assigned by Adam over the number of iterations which is shown in Figure 12 (right). We then ran GD with the same stepsize schedule for each layer as we found Adam to use (we refer to this setup as GD-A), and observed that it performs essentially the same as Adam (See Figure 12 (left)). This demonstrates that it is critical to learn the layers with different stepsizes, but the adaptive gradients chosen by Adam are not critical nor relevant for performance.

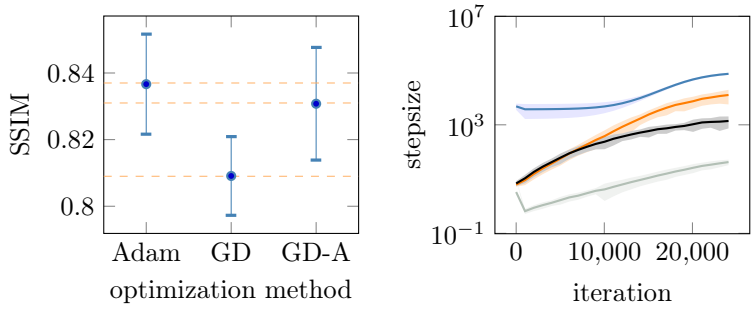


Figure 12: **Left.** Assigning larger stepsizes to shallower layers and utilizing an increasing schedule enhances the performance of GD relative to Adam. SSIM scores are shown for different optimizers over 10 runs on a sample image from the 4x accelerated FastMRI validation set. Adam, GD, and GD-A (GD with the same stepsize schedule as Adam) are considered. **Right.** Average layer-wise stepsizes for different layers of ConvDecoder based on the iteration number when using the Adam optimizer. All values are averaged over 6 randomly-chosen images from the 4x accelerated FastMRI validation set.