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

April 29, 2021

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 and show that it significantly outperforms other un-trained methods, in particular sparsity-based classical compressed sensing methods and naive applications of un-trained neural networks. We also compare performance (both in terms of reconstruction accuracy and computational cost) in an ideal setup for trained methods, specifically on the fastMRI dataset, where the training and test data come from the same distribution. We find that our un-trained algorithm achieves similar performance to a baseline trained neural network, but a state-of-the-art trained network outperforms the un-trained one. Finally, we perform a comparison on a non-ideal setup where the train and test distributions are slightly different, and find that our un-trained method achieves similar performance to a state-of-the-art accelerated MRI reconstruction method.

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 [1–5]. 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) [6], 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. Un-trained networks perform well for denoising [6, 7], compressive sensing [8, 9], phase retrieval [10–12], and even for reconstructing videos [13, 14]. Moreover, they provably succeed in denoising and reconstructing smooth signals from few measurements [15, 16]. 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 address the problem of accelerating Magnetic Resonance Imaging (MRI) with un-trained neural networks, to understand if un-trained networks can give state-of-the-art performance in practice, and to study if they can compete with trained neural networks for image reconstructions problems.

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 [17] by reconstructing an image from a few measurements, which in turn results in a faster scan time.

Traditional compressed sensing methods are based on ℓ_1-norm minimization or Total-Variation (TV) norm minimization [18] 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 [19, 20], a competition for accelerated MRI reconstruction.

Off-the-shelf un-trained neural networks have been applied to accelerated MRI reconstruction before [9, 13, 21], and have shown promising improvements over traditional un-trained methods for example images. At the same time, the two works [9, 21] 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 method specifically tailored to multi-coil MRI and understand its imaging performance relative to competing un-trained and trained approaches. Our contributions are:

- We improve image recovery performance with un-trained neural networks through i) architectural improvements, ii) introducing a data consistency step, iii) including sensitivity maps in the reconstruction process, and iv) combining reconstructions of different runs. Taken together, those steps give a significant accuracy improvement over naive application of un-trained networks.

- As an architectural improvement, we propose a variation of the Deep Image Prior (DIP) [6] and Deep Decoder [7] 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. ConvDecoder is a simple convolutional generator comprised of up-sampling, convolution, batch normalization, and ReLU blocks in each layer.

- We show that un-trained neural networks significantly outperform traditional un-trained methods based on sparse and low-rank models. We consider TV-minimization as a baseline [18] and ENLIVE [22] as a representative un-trained method based on low-rank solutions [23–25]. Traditional sparse and low-rank approaches are a natural comparison, because they perform well, are used in practice, and do not rely on training data, just like our un-trained method.

- Un-trained neural networks are relatively slow because they require a neural network to be fitted to observations via an iterative procedure. We propose an initialization technique that accelerates the reconstruction by a factor of 10.

- We compare un-trained methods to trained methods on a dataset where trained methods shine: the fastMRI dataset [26], a dataset for deep learning based accelerated MRI, where the training and test distribution are the same. We find that, perhaps surprisingly, even in this setup, un-trained methods have on-par performance with the U-net [27], a baseline trained method, and are only 1.78 dB worse (see Table 5) than the state-of-the art trained method, the VarNet [28]. This suggests that there is less benefit in learning-based approaches for imaging, at least in the context of MRI, than currently thought.

- Finally, we compare un-trained methods to trained methods in a setup where train and test distributions are slightly different: we train U-net and VarNet and tune un-trained methods on the fastMRI knee dataset, and test on the fastMRI brain dataset. All methods perform significantly worse under this distribution shift. In this regard, we propose two approaches to mitigate performance degradation due to distribution shifts for un-trained methods: (i) performing meta learning via a few examples from the test domain, and (ii) we propose an auto-tuning technique for un-trained networks with which our un-trained method achieves performance on par to the state-of-the-art VarNet under the distribution shift. Distribution shifts are common in practice, and a key advantage of un-trained neural networks (with auto-tuning) over trained ones is robustness to out-of-distribution examples.

As a baseline for trained methods, we consider U-net based reconstruction because of its popularity and ease of use. As the current state-of-the art method tested on the fastMRI dataset, we consider the end-to-end Variational Network (VarNet) introduced in [28]. We emphasize that *our method operates without any training data* and only leverages few data points for hyper-parameter tuning, while the U-net and VarNet are trained. The U-net is a baseline in the fastMRI challenge [19, 20]. Other interesting approaches that perform close to VarNet are the Pyramid Convolutional Recurrent Neural Network (PCRNN) [29], Σ-net [30], XPDNet [31], Cascade net [32], and invertible Recurrent Inference Machines (i-RIM) [33].

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 [26]. 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. Since 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

We recover images from measurements by using un-trained networks as image priors as follows. 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 image.

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 trained on a distribution of images. 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 were first proposed for image restoration problems by Ulyanov et al. [6]. Ulyanov et al. [6] 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 [7]. We discuss architectures in Section 3.3.

3.1 Single-coil reconstruction

To explain 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 is critical for performance: 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. [9]. Here, the first two output channels of the un-trained network generate the real and imaginary parts of the first image, the second two channels 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 $G(\mathbf{C})$ is a stack of reconstructed images for all coils, in that we use one generator to generate multiple images at once. Therefore, $G_i(\mathbf{C})$ (the $i-$th element of the output stack) 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

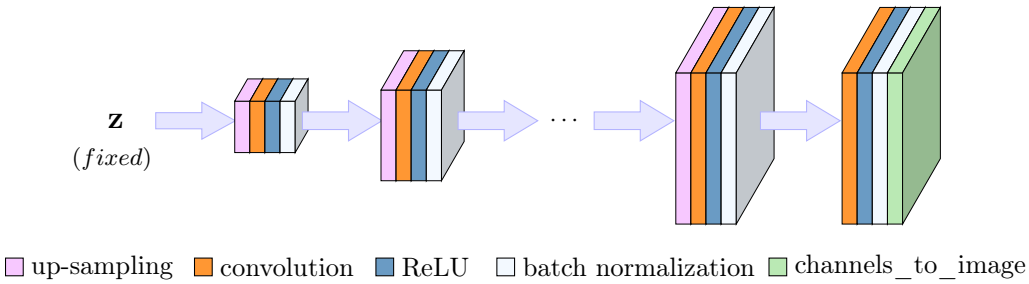

Figure 1: ConvDecoder architecture. It is comprised of up-sampling, convolutional, ReLU, batch normalization, and linear combination layers.

from the under-sampled data by applying ESPIRiT [34] into account. Specifically, the loss function is

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

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 located at x axis coordinates $\mathcal{S}_x = \{i_1, \ldots, i_m\}$. After reconstructing the image, in its corresponding Fourier domain representation, we replace frequency components located at $\mathcal{S}_x$ with the ones from the under-sampled measurement.

3.3 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 over the weights of the network. The DIP is the first and most popular architecture [6], and consists of an encoder, decoder and skip connections.

The Deep Decoder [7] is a simple decoder architecture, even simpler than the decoder part of the DIP, only consisting 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; and similar to the original Deep Decoder in other instances. We also tried a variety of other architectures, including combinations of ConvDecoders that reconstruct an image at different resolutions, but found the simple ConvDecoder to perform best (more details on the architectures we analyzed can be found in the supplementary material).

Both the Deep Decoder and the ConvDecoder are convolutional neural networks 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 [35]. BN normalizes each channel of its input volume independent of other channels. The Deep Decoder uses bi-linear upsampling and 1x1 convolutions, 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 supplementary material. 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 has 8 layers (including the last layer) and 256 channels per layer.

4 Reconstruction accuracy of un-trained neural networks for MRI

In this section, we study the performance of un-trained neural networks for multi-coil 4x accelerated knee MRI reconstruction. We also provide results for 8x accelerated knee measurements and 4x accelerated brain measurements in the supplementary material. We focus on multi-coil reconstruction, because multi-coil is clinically more relevant than single-coil reconstruction. Here, we focus on a setup that is ideal for trained methods: specifically, we train on the fastMRI training set, and evaluate on the fastMRI validation set, and both of those sets come from the same distribution. In practice, the train an test set often come from slightly different distributions, hence we study this aspect in Section 7.

After discussing how to evaluate performance (Section 4.1), we provide the evaluation results for 4x accelerated knee measurements (Section 4.2). Our main findings are: (i) perhaps surprisingly, our un-trained networks perform as well as a standard baseline *trained* method, the U-net, and only slightly worse than the state-of-the-art network VarNet—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 sparsity based baseline method, and ENLIVE, a calibration-less low-rank based 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 the supplementary material. We also include a discussion in the supplementary material on how the results depend on the initialization and the choice of hyper-parameters.

4.1 Considerations when 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. [36] 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) [37] is often a better metric for assessing diagnostic quality than widely-used metrics such as PSNR and Structural Similarity Index (SSIM) [38]. Interestingly, the study ranked PSNR and SSIM—the perhaps most widely-used metrics for assessing image quality–among the most inept metrics in this study. But even a higher VIF score sometimes corresponds to a visibly worse performance; to evaluate algorithms, we therefore measure performance in VIF, PSNR, SSIM, and Multi-Scale SSIM (MS-SSIM) [39], and also show example images for visual comparison.

Normalization It is often necessary to normalize images when comparing them to a ground truth image. Marcin et al. [40] 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.

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 a volume. All afore-mentioned metrics depend on the dynamic range of the volumes, and computing scores in a volume- or image-wise fashion gives different values. In the fastMRI challenge, scores are computed in a volume-based manner, i.e., the dynamic range of the volume is considered for computing the scores [28, 29, 41, 42]. Since images within each volume are analyzed independently, we consider the dynamic range of each image separately.

We refer the reader to the supplementary material for further discussion on the mentioned evaluation challenges.

4.2 Evaluation results

We evaluate the performance of ConvDecoder along with other methods on the 4x accelerated multi-coil knee measurements of the fastMRI dataset. We consider seven methods (five un-trained and two trained), specifically (i) ConvDecoder, (ii) Deep Decoder [7], (iii) DIP [6], (iv) a standard un-trained TV-norm minimization method [18], (v) the recently published calibration-less un-trained method ENLIVE [22], (vi) the U-net [27], a standard trained method, and finally (vii) the end-to-end variational network (VarNet) [28]. We did an extensive grid search to select the best parameters for each un-trained neural network. See the supplementary material for details and for the parameter setup for each method.

Method	VIF	MS-SSIM	SSIM	PSNR
ConvDecoder	**0.6717 ± 0.0411**	**0.9443 ± 0.0152**	**0.7775 ± 0.0258**	**31.81 ± 0.96**
DIP	0.6311 ± 0.0391	0.8981 ± 0.0122	0.5938 ± 0.0264	28.40 ± 1.02
DD	0.6359 ± 0.0421	0.8599 ± 0.0176	0.6991 ± 0.0287	29.16 ± 1.13

Table 1: 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. Marginal errors denote 95% confidence interval.

We start by comparing the ConvDecoder with Deep Decoder and DIP architectures, and find that ConvDecoder performs best for knee MRI images. Specifically, to compare the afore-mentioned un-trained networks, we computed the scores by averaging over 20 randomly-chosen mid-slice (i.e., the middle slice of the volume) images of different volumes in the validation set, and as mentioned earlier, the scores are computed in an image-based manner; see Table 1 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 present in other images as well, and hence are reflected in the scores in Table 1.

From Table 1, 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 un-trained baselines Next, we compare ConvDecoder with U-net, VarNet, TV, and ENLIVE. We trained a standard U-net with 8 layers and width factor equal to 32 on the whole training set (974 volumes). We also trained an end-to-end variational network with 12 cascades and width factor equal to 18 on the

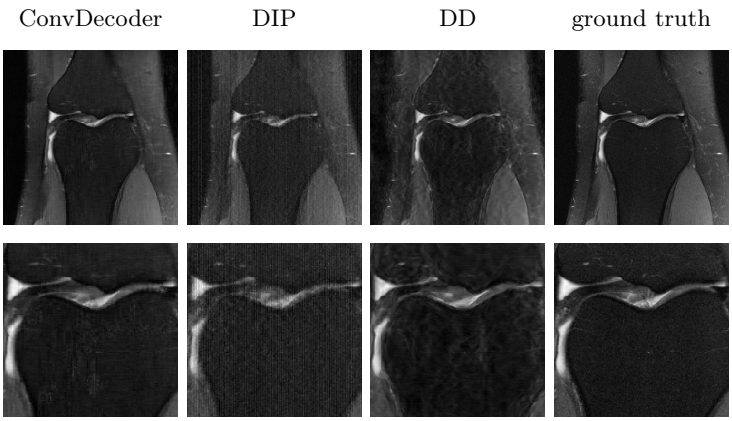

Figure 2: Sample reconstructions for ConvDecoder, DIP, and Deep Decoder (DD) for a validation image from multi-coil knee measurements (4x accelerated). The top row represents zoomed-in version of the bottom row. ConvDecoder gives the best reconstruction for this image.

training set[1]. To compute the results on the whole validation set, we ran the four mentioned methods on the mid-slice image of each volume in the validation set (a set of 200 images). We also randomly chose the mask for each run, but the marginal errors shown in Table 2 denote that even with the randomness associated with masks, confidence intervals are sufficiently tight.

Method	VIF	MS-SSIM	SSIM	PSNR
ConvDecoder	**0.6823 ± 0.0217**	0.9387 ± 0.0059	0.7753 ± 0.0145	31.67 ± 0.39
ConvDecoder-noDC	0.6323 ± 0.0236	0.9265 ± 0.0061	0.7107 ± 0.0153	30.46 ± 0.42
U-net	0.5955 ± 0.0147	0.9489 ± 0.0035	0.7883 ± 0.0097	32.04 ± 0.21
VarNet	0.6456 ± 0.0122	**0.9592 ± 0.0028**	**0.8342 ± 0.0084**	**34.20 ± 0.18**
TV	0.4412 ± 0.0272	0.9262 ± 0.0061	0.6977 ± 0.0141	30.20 ± 0.41
ENLIVE	0.3906 ± 0.0346	0.8516 ± 0.0098	0.5531 ± 0.0213	26.14 ± 0.29

Table 2: Performance for reconstructing 200 mid slice knee images 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. Marginal errors denote 95% confidence interval.

Table 2 shows the results for each method. The scores show that the ConvDecoder has higher VIF performance than U-net and VarNet (recall that VIF is deemed most relevant by clinicians), and achieves on-par performance with U-net on all other metrics. Moreover, it significantly outperforms TV and ENLIVE. Figure 3 shows a sample reconstruction for an image from the validation set for the mentioned methods. As we also noted before, the state-of-the-art VarNet outperforms both ConvDecoder and the U-net on the other metrics.

5 Computational cost and making un-trained networks 10x faster

We next show that un-trained neural networks are relatively slow, but in order to mitigate that, we propose a domain-specific initialization that accelerates the method by a factor of 10. Table 3 shows runtimes for testing each method on a single validation image, averaged over 10 runs. All runtime values denote CPU clock times and the experiments were run on a single TITAN V GPU. As those numbers demonstrate, and as is well known, un-trained methods, and in particular un-trained networks, are significantly slower than trained neural networks. That is because un-trained methods solve an optimization problem via an iterative algorithm, whereas neural networks only require a forward pass through the network.

[1]Both U-net and VarNet are trained by following the exact instructions outlined in the fastMRI repository.

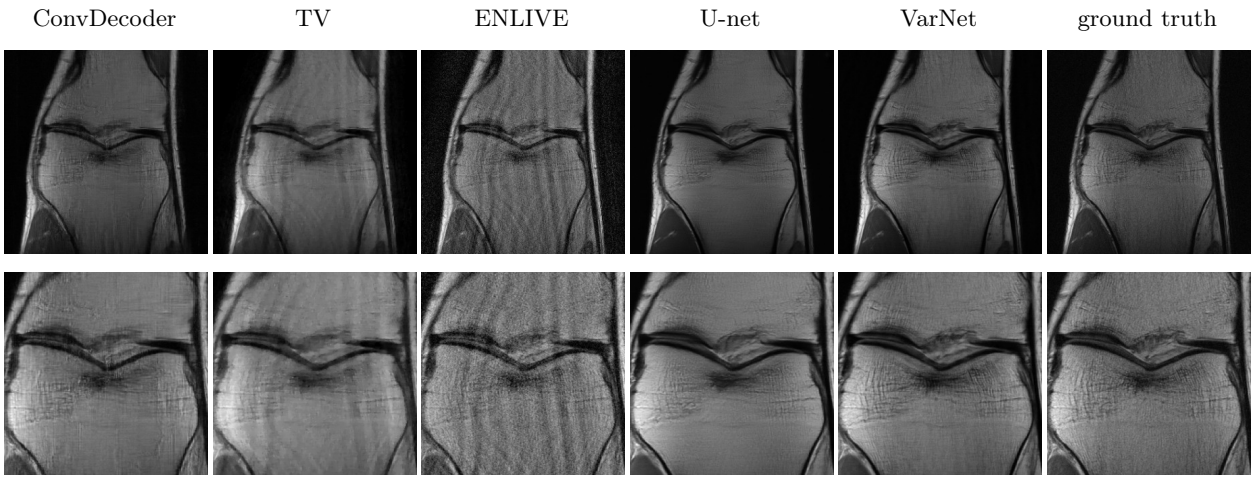

Figure 3: Sample reconstructions for ConvDecoder, TV, ENLIVE, U-net, and the end-to-end variational network (VarNet) for a validation image from multi-coil knee measurements (4x accelerated). The bottom row represents zoomed-in version of the top row. ConvDecoder and the end-to-end variational network (VarNet) find the best reconstructions for this image (slightly better than U-net and significantly better than TV and ENLIVE).

Method	test	train	tune
ConvDecoder	63.5 ± 0.3 minutes	-	4 days
DIP	119.7 ± 0.2 minutes	-	8 days
DD	56 ± 0.2 minutes	-	4 days
U-net	0.2 ± 0.1 seconds	8 days	-
VarNet	0.7 ± 0.3 seconds	2 weeks	-
TV	20 ± 3 seconds	-	5 minutes
ENLIVE	230 ± 10 seconds	-	60 minutes

Table 3: Runtimes when applying each method to a validation knee image (a single slice). The numbers are averaged over 10 runs and marginal errors are for the 95% confidence interval.

We next propose an approach to accelerate un-trained networks by a factor of 10. It is well known that for iterative optimization methods (e.g., gradient descent), the initialization affects the time it takes to get a solution of desired accuracy. Our acceleration approach is based on starting from a good initialization.

Specifically, we exploit that MRI knee images fall into two categories that are statistically similar: fat-suppressed and non-fat-suppressed images. Images in each class share statistical properties such as contrast, pixel-wise histogram, etc. It turns out that a decoder optimized to represent (or reconstruct) a non-fat-suppressed image provides a good initialization for other non-fat-suppressed images.

Accordingly, we first select one non-fat-suppressed knee slice randomly and fit ConvDecoder (based on loss function (3)) until full convergence to reconstruct that image from its 4x under-sampled measurements. As a result, we obtain a set of parameters $\mathbf{C}_1$ for the decoder. We then use $\mathbf{C}_1$ as an initialization to reconstruct other non-fat-suppressed images using ConvDecoder. Table 4 and Figure 4 show that this approach enables achieving the same reconstruction accuracy in 10 times less iterations. Thus, 60 minutes in Table 3 reduces to 6 minutes at the cost of no performance loss. This result shows that interestingly, only one sample gives sufficient information to represent a specific category of knee images.

We hasten to add that the idea of initializing generative models through training has been used before with the goal of enhancing reconstruction quality, see [43, 44] for examples where a generative model is initialized by training on some training set.

Method	VIF	MS-SSIM	SSIM	PSNR
ConvDecoder	**0.8449 ± 0.0108**	**0.9612 ± 0.0032**	0.8244 ± 0.0097	32.94 ± 0.28
ConvDecoder-Fast	0.8317 ± 0.0115	0.9598 ± 0.0036	**0.8297 ± 0.0092**	**33.05 ± 0.26**

Table 4: Guided domain-specific initialization results in achieving the same reconstruction accuracy, yet 10x faster. Scores are averaged over all 100 non-fat-suppressed mid-slice images from the fastMRI validation set for 4x acceleration. Marginal errors denote 95% confidence interval.

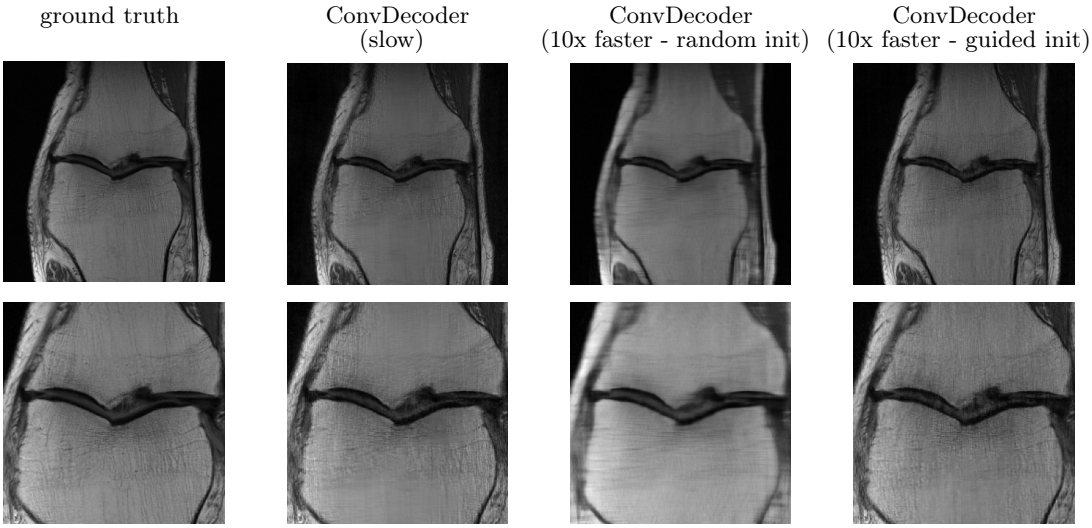

Figure 4: Guided initialization results in the same reconstruction accuracy as full convergence from random initialization. "10x faster - random init" denotes fitting ConvDecoder from random initialization, but for 10 times less number of full convergence iterations.

6 Better performance at the cost of more computations

According to Section 4.2, ConvDecoder significantly outperforms TV and achieves performance close to U-net. In this section, we study 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. This, shown to be effective for denoising in [6], also results in a higher reconstruction accuracy in our setup, in terms of SSIM and PSNR scores, but makes very little visible difference.

Table 5 shows PSNR scores when using this averaging technique. The ConvDecoder's output is obtained through 20 runs based on the mentioned averaging technique. Also, the numbers are averaged over 18 random images (9 proton-density and 9 fat suppressed) from the 4x accelerated knee measurements from the fastMRI validation set. According to Table 5, ConvDecoder marginally outperforms U-net and achieves performance close to the end-to-end variational network.

How many runs are enough to achieve this higher performance? For a given image, Figure 5 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 18 images as in Table 5). 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. A sample reconstruction of our ensemble trick is included in the supplementary material. The sample reconstruction shows that the visible performance increase over the ConvDecoder without the averaging technique is small.

Method	SSIM	PSNR
ConvDecoder-A	0.8127	32.72
ConvDecoder	0.7946	32.13
U-net	0.8094	32.71
VarNet	**0.8365**	**34.50**
TV	0.7012	30.35
ENLIVE	0.5763	26.72

Table 5: Using our averaging technique, ConvDecoder's performance (ConvDecoder-A) exceeds U-net's and becomes close to the performance of the state-of-the-art reconstruction method, the end-to-end variational network (VarNet).

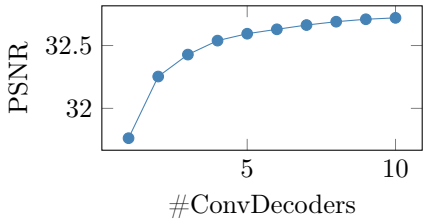

Figure 5: Averaging the output of $k \in \{1, 2, ...10\}$ ConvDecoders monotonically enhances the PSNR score for the resulting image.

7 Robustness to out-of-distribution samples

In Section 4, we found that ConvDecoder achieves on-par performance with U-net when evaluated on in-distribution data (i.e., fastMRI validation set). This is a setup where trained methods shine because the test and train distribution match perfectly. However, this might not reflect performance in practice, where it is difficult to impossible to train on a distribution that matches the test distribution perfectly. Un-trained networks only mildly rely on the training distribution through hyper-parameter tuning and we therefore expect them to potentially be less sensitive to a distribution shift.

In this section, we first show that, perhaps surprisingly, even un-trained methods suffer from distribution shifts through parameter tuning. However, we demonstrate that for un-trained methods, this can be mitigated (i) either through tuning on a few images on the test domain, if available, or (ii) through an auto parameter tuning approach introduced here. With either technique, un-trained networks become robust to distribution, and even perform on par with the best trained method, VarNet, when evaluated under distribution shifts. In practice we suggest to choose option (i) if it is possible to obtain a very small tuning dataset from the test domain, and the auto-tuning technique (ii) otherwise.

7.1 Performance loss under distribution shifts

We start by studying the effect of a distribution shift on reconstruction performance. Even un-trained methods are sensitive to such distribution shifts, because its hyper-parameters are tuned on the training distribution.

We consider a distribution shift from knee to brain images, i.e., we train/tune on knees and test on brains. Specifically, we train the U-net and VarNet models on both knee and brain images, one model for each set of images. Moreover, we tune the un-trained methods on 10 randomly-chosen knee images, which results in an 8-layer ConvDecoder with 256 channels per layer. We also tune the un-trained methods on 10 randomly-chosen brain images, which results in a 5-layer ConvDecoder with 64 channels per layer.

Next, we evaluate those methods on 100 randomly-chosen mid-slice brain images from the fastMRI validation set. Table 6 demonstrates that both trained and un-trained methods suffer a similar performance loss under this distribution shift.

7.2 Mitigating performance loss under distribution shifts with meta learning

A straightforward approach to eliminate the performance loss of un-trained methods under distribution shifts is to record a few ground-truth samples from the test domain for hyper-parameter tuning. To illustrate this approach for the distribution shift from knee to brain images, we tune un-trained methods on 10 randomly-chosen brain images.

The results in Table 7 show that un-trained methods are robust against a distribution shift provided that a few samples from the test domain are available for meta learning (i.e., tuning their hyper-parameters). In this setup, our un-trained network achieves on-par performance with the state-of-the art VarNet and it outperforms the baseline U-net.

Method	train/tune on knee test on brain PSNR	train/tune on knee test on brain SSIM	train/tune on brain test on brain PSNR	train/tune on brain test on brain SSIM
ConvDecoder	31.75 ± 0.22	0.8589 ± 0.0056	34.46 ± 0.13	0.9097 ± 0.0049
U-net	31.21 ± 0.14	0.8842 ± 0.0043	34.29 ± 0.10	0.9173 ± 0.0036
VarNet	**33.96 ± 0.11**	**0.9067 ± 0.0039**	**37.46 ± 0.07**	**0.9421 ± 0.0031**
TV	26.02 ± 0.48	0.7659 ± 0.0078	27.10 ± 0.57	0.7846 ± 0.0082
ENLIVE	24.86 ± 0.42	0.7534 ± 0.0081	26.51 ± 0.55	0.7780 ± 0.0087

Table 6: Validation results for a set of 100 mid-slice brain images from the fastMRI validation set. All methods suffer a significant performance loss under a distribution shift. **Left.** PSNR and SSIM scores when evaluating on the brain validation set (training U-net and VarNet as well as tuning ConvDecoder, TV, and ENLIVE is done on the knee training set). **Right.** PSNR and SSIM scores when evaluating on the brain validation set (training U-net and VarNet as well as tuning ConvDecoder, TV, and ENLIVE is done on the brain training set). Marginal errors denote 95% confidence intervals.

Method	train on knee test on brain PSNR	train on knee test on brain SSIM	train on brain test on brain PSNR	train on brain test on brain SSIM
ConvDecoder-SM	**34.46 ± 0.13**	**0.9097 ± 0.0049**	34.46 ± 0.13	0.9097 ± 0.0049
U-net	31.21 ± 0.14	0.8842 ± 0.0043	34.29 ± 0.10	0.9173 ± 0.0036
VarNet	33.96 ± 0.11	0.9067 ± 0.0039	**37.46 ± 0.07**	**0.9421 ± 0.0031**
TV	27.10 ± 0.57	0.7846 ± 0.0082	27.10 ± 0.57	0.7846 ± 0.0082
ENLIVE	26.51 ± 0.55	0.7780 ± 0.0087	26.51 ± 0.55	0.7780 ± 0.0087

Table 7: Validation results for a set of 100 mid-slice brain images from the fastMRI validation set. Trained neural networks suffer a significant performance loss under a distribution shift. **Left.** PSNR and SSIM scores for evaluating on the brain validation set (training is done on the knee training set only for VarNet and U-net). **Right.** PSNR and SSIM scores when evaluating on the brain validation set (training is done on the brain training set only for VarNet and U-net). Marginal errors denote 95% confidence interval.

7.3 Mitigating performance loss under distribution shifts with auto-tuning

Next, we consider again the setup in Section 7.1 where no data points from the test distribution are available for parameter tuning. Here, we show that it is possible to auto-tune the parameters of the un-trained neural networks to adjust their hyper-parameters for a given reconstruction problem.

The idea of our new auto-tuning approach is to measure how well ConvDecoder with a set of hyper-parameters interpolates missing information in the k-space. Specifically, given an under-sampled measurement $\mathbf{y}$, we first randomly remove (i.e., set to zero) a fraction q of the given k-space measurements, denoted by the hold-out set $\mathcal{S}$. The measurement vector without those frequencies is denoted by $\mathbf{y}_{-\mathcal{S}}$. Then, for a specific hyper-parameter configuration h, we reconstruct an image with ConvDecoder based on the measurement $\mathbf{y}_{-\mathcal{S}}$. The Fourier representation of the reconstructed image is denoted by $\hat{\mathbf{y}}$. We then compute the MSE between $\mathbf{y}$ and $\hat{\mathbf{y}}$ over $\mathcal{S}$, the set of hold-out values in the k-space. The resulting number is a good proxy on how well the given ConvDecoder configuration performs on all the missing frequency elements, even those not in the hold-out set $\mathcal{S}$.

At the cost of computation, this procedure might be performed k times for each parameter setup, similar to performing k-fold cross validation. The described auto-tuning framework selects a set of hyper-parameters whenever a new sample appears, therefore it does not rely on any training or tuning data except the given sample itself. As mentioned before, this approach is computationally expensive as it increases the computational cost of reconstructing an image by a factor of $|\mathcal{H}| \cdot k$, where k is the number of folds and $|\mathcal{H}|$ is the number of hyper-parameter configurations. We set $k = 2$ in our experiment and also use any combination from #layers $= \{5, 8\}$, #channels $= \{64, 256\}$, and sens $= \{0, 1\}$ for the hyper-parameters (sens $= 1$ denotes incorporating coil sensitivity estimates in the optimization as shown in loss function (4)).

We next evaluate the performance of ConvDecoder with the auto-tuning approach under the distribution shift from knee to brain. We compare: (i) tuning ConvDecoder on a few brain images, which is the best-

performing option but relies on training data (cf. Section 7.2), (ii) simply applying knee-tuned ConvDecoder to brain images (cf. Section 7.1), and (iii) using our auto-tuning approach. Table 8 shows that our auto-tuning approach significantly mitigates the performance loss due to distribution shifts, yet still incurs a slight performance loss through this distribution shift when relative to tuning on the test distribution.

Setup	Score	
	PSNR	SSIM
Tune on brain - test on brain	**34.46 ± 0.13**	**0.9097 ± 0.0049**
Tune on knee - test on brain	31.75 ± 0.22	0.8589 ± 0.0056
Auto-tune - test on brain	33.54 ± 0.16	0.8944 ± 0.0052

Table 8: Validation results for a group of 100 mid-slice brain images from the fastMRI validation set. Un-trained methods lose performance through a restricted distribution shift. Our auto-tuning framework significantly improves over naive usage of hyper-parameters obtained from the original domain. Marginal errors denote 95% confidence interval.

8 Discussion and Conclusion

MRI scans 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 ℓ_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. For example, if trained on knees, a neural network might perform poorly on brains as demonstrated here.

In this paper we studied un-trained neural networks (from both reconstruction accuracy and reconstruction speed aspects) 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, but without using any training data. We further demonstrated that under a distribution shift, un-trained networks suffer a similar performance than trained methods, but this loss can be mitigated through (i) meta learning with a small tuning dataset, or (ii) auto-tuning without any access to the test distribution.

While our results show that the best trained neural networks still slightly outperform our un-trained methods if sufficient training data is available, the (i) robustness to distribution shift and (ii) not needing any training data (except a few images for meta learning) make un-trained networks an important tool in practice, especially in regimes with 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 and an online demonstration that uses ConvDecoder for reconstructing a sample is available at colab-ConvDecoder.

Acknowledgment

M. Zalbagi Darestani and R. Heckel are (partially) supported by NSF award IIS-1816986, and R. Heckel acknowledges support of the IAS at TUM and the DFG.

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 in the supplementary material).

References

[1] 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.

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

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

[4] 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.

[5] 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.

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

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

[8] 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.

[9] 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.

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

[11] 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.

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

[13] J. Yoo, 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.

[14] 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.

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

[16] 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.

[17] M. Lustig, D. L. Donoho, J. M. Santos, and J. M. Pauly. "Compressed sensing MRI". In: *IEEE signal processing magazine* (2008), pp. 72–82.

[18] 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.

[19] F. Knoll, T. Murrell, A. Sriram, N. Yakubova, J. Zbontar, M. Rabbat, A. Defazio, M. J. Muckley, D. K. Sodickson, C. L. Zitnick, et al. "Advancing machine learning for MR image reconstruction with an open competition: overview of the 2019 fastMRI challenge". In: *Magnetic Resonance in Medicine* (2020).

[20] M. J. Muckley, B. Riemenschneider, A. Radmanesh, S. Kim, G. Jeong, J. Ko, Y. Jun, H. Shin, D. Hwang, M. Mostapha, et al. "State-of-the-art Machine Learning MRI reconstruction in 2020: results of the second fastMRI challenge". In: *arXiv: 2012.06318 [eess.IV]* (2020).

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

[22] H. C. M. Holme, S. Rosenzweig, F. Ong, R. N. Wilke, M. Lustig, and M. Uecker. "ENLIVE: an efficient nonlinear method for calibrationless and robust parallel imaging". In: *Scientific reports* (2019), pp. 1–13.

[23] J. P. Haldar. "Low-rank modeling of local k-space neighborhoods (LORAKS) for constrained MRI". In: *IEEE Transactions on Medical Imaging* (2013), pp. 668–681.

[24] P. J. Shin, P. E. Larson, M. A. Ohliger, M. Elad, J. M. Pauly, D. B. Vigneron, and M. Lustig. "Calibrationless parallel imaging reconstruction based on structured low-rank matrix completion". In: *Magnetic Resonance in Medicine* (2014), pp. 959–970.

[25] K. H. Jin, D. Lee, and J. C. Ye. "A general framework for compressed sensing and parallel MRI using annihilating filter based low-rank Hankel matrix". In: *IEEE Transactions on Computational Imaging* (2016), pp. 480–495.

[26] J. Zbontar et al. "fastMRI: an open dataset and benchmarks for accelerated MRI". In: *arXiv: 1811.08839 [cs.CV]*. 2018.

[27] 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.

[28] A. Sriram et al. "End-to-end variational networks for accelerated MRI reconstruction". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. 2020.

[29] 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.

[30] J. Schlemper, C. Qin, J. Duan, R. M. Summers, and K. Hammernik. "Sigma-net: ensembled iterative deep neural networks for accelerated parallel MR image reconstruction". In: *arXiv:1912.05480 [eess.IV]* (2019).

[31] Z. Ramzi, P. Ciuciu, and J. Starck. "XPDNet for MRI reconstruction: an application to the fastMRI 2020 brain challenge". In: *arXiv:2010.07290 [eess.IV]* (2020).

[32] 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.

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

[34] M. Uecker et al. "ESPIRiT—an eigenvalue approach to autocalibrating parallel MRI: where SENSE meets GRAPPA". In: *Magnetic resonance in medicine*. 2014, pp. 990–1001.

[35] 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.

[36] 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.

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

[38] 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.

[39] 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.

[40] 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.

[41] 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.

[42] 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.

[43] A. P. S. Kohli, V. Sitzmann, and G. Wetzstein. "Semantic implicit neural scene representations with semi-supervised training". In: *2020 International Conference on 3D Vision (3DV)*. 2020, pp. 423–433.

[44] S. A. Hussein, T. Tirer, and R. Giryes. "Image-adaptive GAN based reconstruction". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 2020, pp. 3121–3129.

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

APPENDIX

A Details on evaluating the reconstruction performance

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.

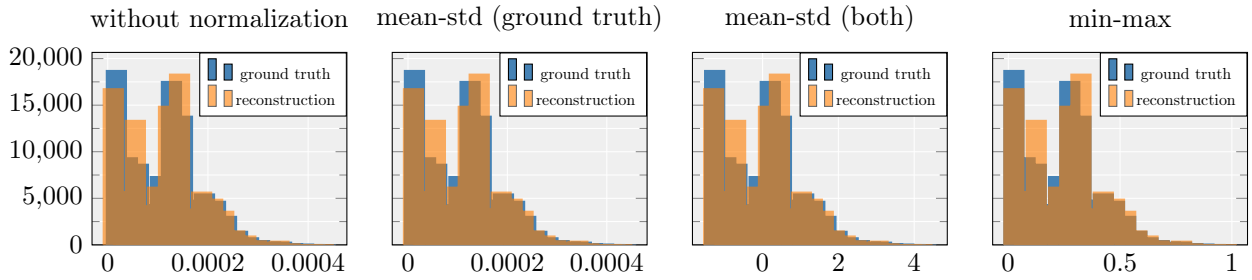

Figure 6: The effect of different image normalization techniques on the distribution of ground truth and reconstructed images.

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 9: 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 6 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 9 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 7 where the ground truth image is very noisy. In

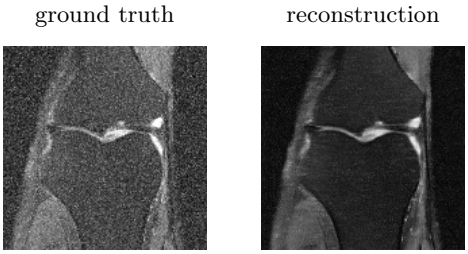

Figure 7: A sample image from the multi-coil knee dataset where the ground-truth image is very noisy. In the evaluation, we compare a reconstructed image to such a noisy image, and even if the reconstruction is a subjectively sharp and clear image, the error metric is large.

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.

Volume-based vs. image-based evaluation To illustrate the point that volume- and image-based evaluations give very different numbers, Table 10 provides average volume-based as well as image-based SSIM and PSNR scores for ConvDecoder, U-net, the end-to-end variational network, 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.

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
VarNet	**0.9062**	**38.21**	**0.8432**	**34.10**
TV	0.7214	33.21	0.6832	30.12

Table 10: Volume-based vs. image-based score compution leads to different numbers: Average scores for ConvDecoder, U-net, the end-to-end variational network (VarNet), and TV. For comparing volume- vs. image-based evaluation, SSIM and PSNR scores are averaged over 20 randomly-chosen volumes (640 slices).

B Parameter setup and optimization

We tuned the parameters of ConvDecoder, DIP, and Deep Decoder by performing a grid search over a group of 10 randomly-chosen training 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 11 shows the grid parameters for each network. For the U-net and the end-to-end variational network (VarNet), we chose the set of parameters used in the fastMRI challenge[2] [26] and trained them accordingly. The parameter setup is provided in Table 11.

In order to fit the un-trained methods to the under-sampled measurements, we used the Adam optimizer [45] with constant stepsize 0.01 (without early stopping) for optimizing the loss function (which is MSE loss function in our experiments).

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

Method	#layers	#channels (or width)	convolutional kernel size	#pools	#sens-pools	#sens-channels	#cascades
ConvDecoder	{4, 5**, 6, 7, 8*, 9}	{32,64**,128, 160, 256*, 480}	3	0	-	-	-
U-net	8	32***	3	4	-	-	-
Varnet	-	18	3	4	4	8	12
DIP	{10, 12**, 14, 16*, 18}	{64**,160, 256*, 360}	3	2	-	-	-
DD	{6, 7, 8, 9, 10*,**,11}	{64**,128,256, 368*, 512}	1	0	-	-	-

* Knee chosen parameters.
** Brain chosen parameters.
*** 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 11: Model parameters for ConvDecoder, U-net, Varnet, DIP, and Deep Decoder (DD).

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).

B.1 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 8 (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, we ran the ConvDecoder 10 times on an under-sampled measurement and averaged the scores. Figure 8 (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).

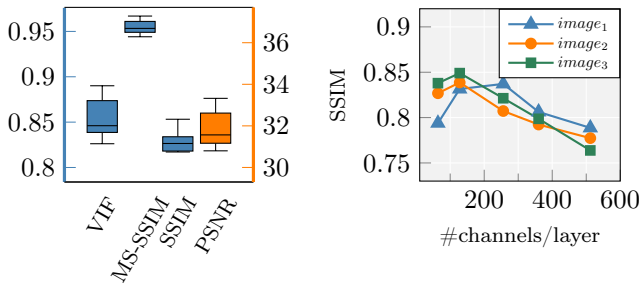

Figure 8: Effect of hyper-parameters on the ConvDecoder's performance. **Left**: fluctuations of different scores during 10 runs on a single data point. **Right**: effect of network width on the SSIM score for three data points from the validation set.

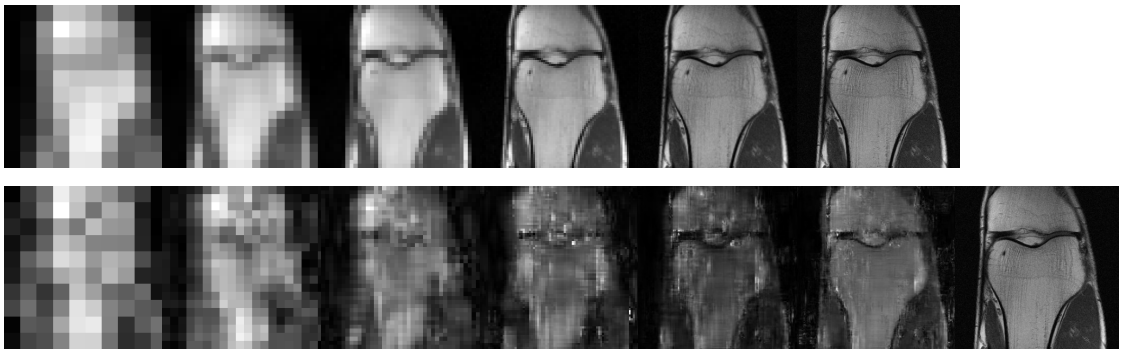

Figure 9: ConvDecoder finds an image representation by constructing fine details per each layer. The network is fitted for representing an image from the 4x under-sampled k-space of the ground-truth image in the mid row. **Top row:** different resolutions of the ground-truth image in the bottom row to each of which we fit the layer outputs. **Bottom row:** output visualization for each of the six layers.

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 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 un-trained 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 9 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 9, 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 10 (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 10 (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.

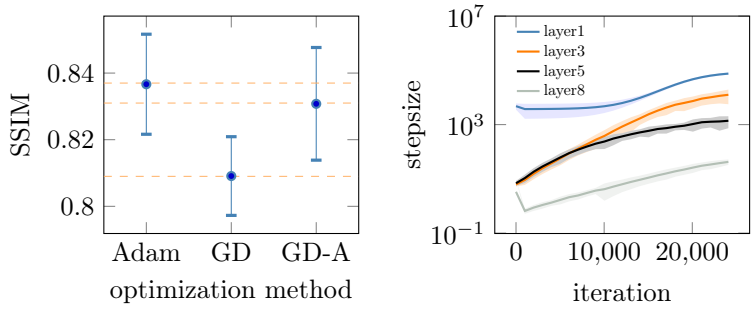

Figure 10: **Left.** Assigning larger stepsizes to shallower layers and utilizing an increasing schedule enhances the performance of GD relative to Adam. SSIM scores are shown for different optimizers over 10 runs on a sample image from the 4x accelerated fastMRI validation set. Adam, GD, and GD-A (GD 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.

D 8x accelerated multi-coil knee measurements

In this section, we report the results for 8x accelerated 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 Section 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 ± 0.0214	0.8827 ± 0.0070	0.6815 ± 0.0142	28.49 ± 0.31
U-net	0.5233 ± 0.0146	0.9148 ± 0.0029	0.7115 ± 0.0088	29.25 ± 0.15
VarNet	**0.5821 ± 0.0115**	**0.9432 ± 0.0025**	**0.7812** ± 0.0076	**31.65 ± 0.13**
TV	0.3119 ± 0.0232	0.8340 ± 0.0075	0.5986 ± 0.0139	26.55 ± 0.35
ENLIVE	0.3636 ± 0.0283	0.7889 ± 0.0086	0.4986 ± 0.0183	23.11 ± 0.26

Table 12: Average image-based scores for reconstructing 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 as well as ENLIVE, yet the end-to-end variational network (VarNet) performs significantly better for 8x acceleration. Marginal errors denote 95% confidence interval.

Table 12 shows that ConvDecoder achieves similar performance to U-net (according to all metrics except SSIM and MS-SSIM which U-net slightly outperforms ConvDecoder) and significantly outperforms TV as well as ENLIVE. Figure 11 shows a sample reconstruction for all considered methods.

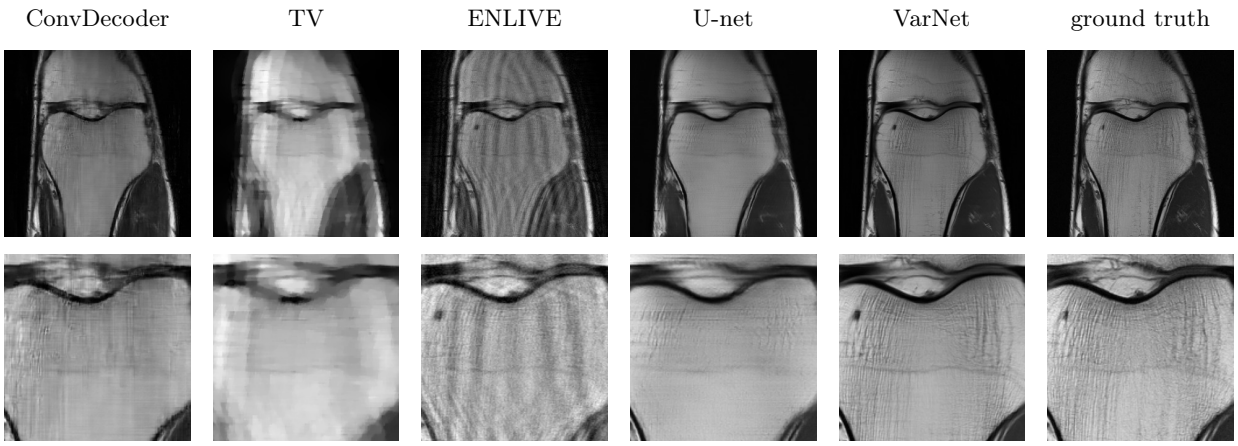

Figure 11: Sample reconstructions for ConvDecoder, TV, ENLIVE, U-net, VarNet for a validation image from multi-coil knee measurements (8x accelerated). The bottom row represents zoomed-in version of the top row. ConvDecoder finds a similar reconstruction as U-net and outperforms TV. The state-of-the-art VarNet yields the best reconstruction for this image.

E 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) as well as a modern un-trained method (ENLIVE). In this section, we consider un-trained networks for the reconstruction of brain images to fortify our claims.

For the sake of consistency with the rest of the paper, we first compare different un-trained neural networks and then compare the best-performing one to competing methods. We again performed an extensive grid search and the resulting hyper-parameters are shown in Section B.

Table 13 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 new data consistency step which resulted in approximately 8% SSIM score improvement for these un-trained networks.

Method	VIF	MS-SSIM	SSIM	PSNR
ConvDecoder	0.8002 ± 0.0168	**0.9743 ± 0.0055**	0.9018 ± 0.0121	34.58 ± 0.41
DIP	**0.8223 ± 0.0144**	0.9736 ± 0.0051	0.8918 ± 0.0131	**34.96 ± 0.35**
DD	0.7952 ± 0.0184	0.9713 ± 0.0063	**0.9023 ± 0.0138**	34.52 ± 0.51

Table 13: 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. Marginal errors denote 95% confidence interval.

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; although 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 for brain images. To quantify the gains of employing the estimated coil sensitivity maps, we include ConvDecoder scores with/without sensitivity maps along with U-net, the end-to-end variational network, TV, and ENLIVE

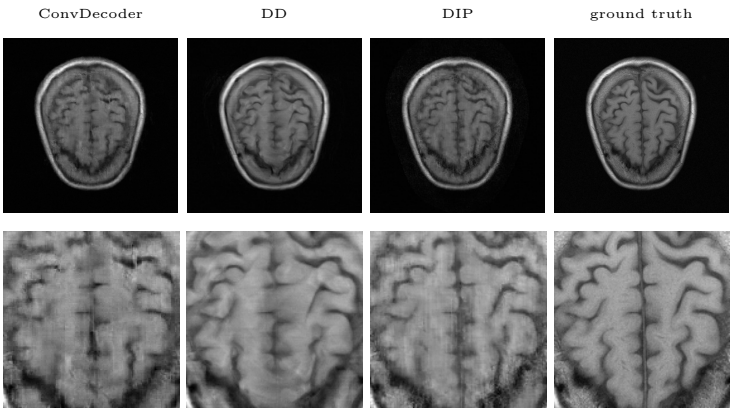

Figure 12: ConvDecoder, DD, and DIP yield similar reconstructions for brain images. The artifacts can be mitigated by incorporating coil sensitivity maps as shown in Fugure 13. Images in the bottom row are zoomed-in versions of the top row.

in Table 14. The numbers are averaged over 100 validation brain images.

Method	VIF	MS-SSIM	SSIM	PSNR
ConvDecoder	0.8178 ± 0.0091	0.9722 ± 0.0024	0.8892 ± 0.0054	33.06 ± 0.11
ConvDecoder-SM	0.8707 ± 0.0087	0.9804 ± 0.0021	0.9097 ± 0.0049	34.46 ± 0.13
U-net	0.8714 ± 0.0062	0.9792 ± 0.0014	0.9173 ± 0.0036	34.29 ± 0.10
VarNet	**0.9173 ± 0.0051**	**0.9912 ± 0.0009**	**0.9421 ± 0.0031**	**37.46 ± 0.07**
TV	0.6734 ± 0.0379	0.8748 ± 0.0113	0.7846 ± 0.0082	27.10 ± 0.57
ENLIVE	0.6431 ± 0.0371	0.8660 ± 0.0108	0.7780 ± 0.0087	26.51 ± 0.55

Table 14: Average image-based scores for the ConvDecoder, ConvDecoder-SM (ConvDecoder + sensitivity maps) vs U-net, VarNet, TV, and ENLIVE on 100 mid-slice images of 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 as well as ENLIVE. However, both U-net and ConvDecoder are slightly outperformed by the state of the art (VarNet). Marginal errors denote 95% confidence interval.

Figure 13 depicts sample reconstructions for ConvDecoder-SM (Convdecoder + sensitivity maps), ConvDecoder, U-net, the end-to-end variational network, TV, and ENLIVE for a validation image. Note the significant improvement in the reconstruction quality as a result of using estimated sensitivity maps. Since ConvDecoder-SM, U-net, and the end-to-end variational network (VarNet) 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 and VarNet are 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. We observe that for the region denoted by the red circle, VarNet also merges the mentioned black points to some extent, yet this effect is less severe compared to the Unet.

F Better performance at the cost of more computations

In Section 6, we showed a better reconstruction accuracy can be achieved via ConvDecoder at the cost of more computation. Specifically, we introduced an ensemble trick to enhance the performance by averaging the outputs of multiple decoders. We further demonstrated monotonic improvement of PSNR as a function of the number of decoders. The outcome of such analysis was getting closer (within 2 dB) to the performance of the state-of-the-art VarNet.

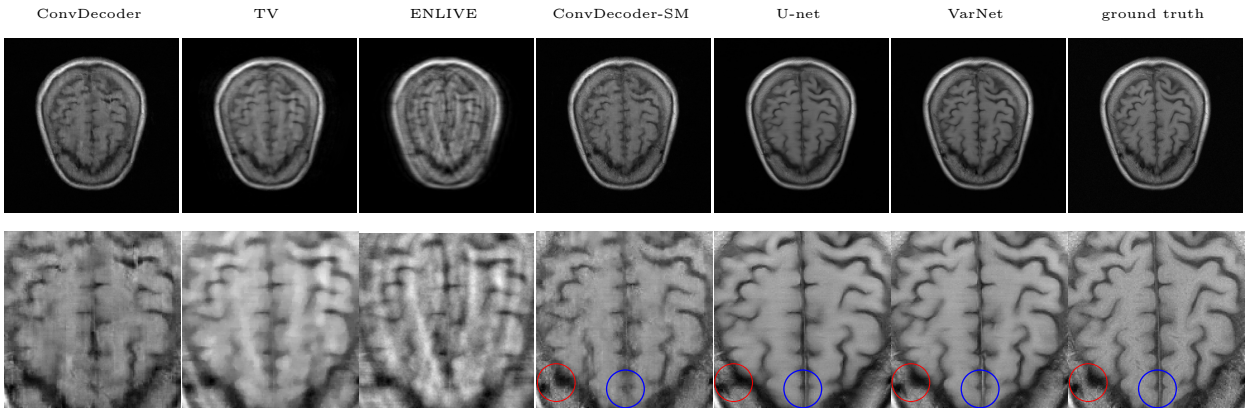

Figure 13: ConvDecoder-SM, without any training data, yields noticeable improvement over ConvDecoder for brain MRI images, significantly outperforms TV (a traditional compressed sensing method) as well as ENLIVE (a recently-introduced compressed sensing method), and performs slightly worse than U-net (a baseline trained neural network) as well as VarNet (a state-of-the-art trained neural network). Images in the bottom row are zoomed-in versions of the top row.

Figure 14 shows sample reconstructions for ConvDecoder with and without our averaging technique (the average image is obtained using ten decoders). As shown, the averaging technique results in slightly smoother regions in the middle parts of the image, yet removes small reconstruction artifacts, and overall yields a higher score (e.g., 0.59 dB higher PSNR as shown in Section 6).

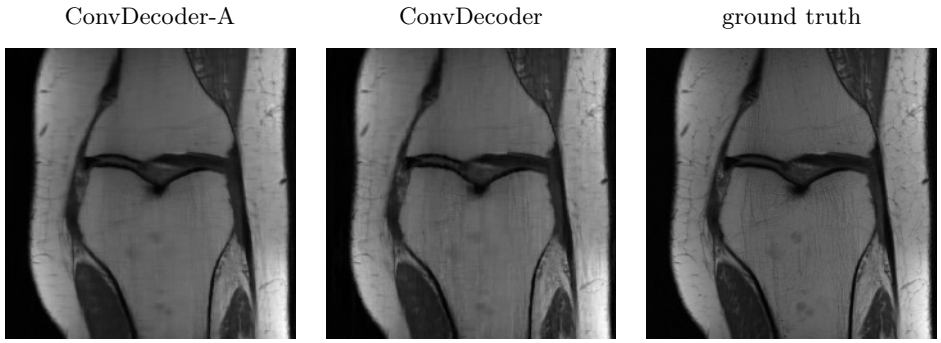

Figure 14: Sample reconstructions for ConvDecoder and ConvDecoder-A (with the averaging technique) for a validation image from multi-coil knee measurements (4x accelerated).