Journal of Research and Innovation in Technology, Commerce and Management
Vol. 2 Issue 11, November 2025, pp. 21180-21192
**ISSN: 3049-3129(Online)**

# Learning Image Restoration Without Clean Data

**Nazia Parveen**
Student, Lovely Professional University, Punjab
Email: nazia.12113186@lpu.in

**Cite as:** Nazia Parveen. (2025). Learning Image Restoration Without Clean Data. Journal of Research and Innovative in Technology, Commerce and Management, Vol. 2(Issue 11), 21180–21192. https://doi.org/10.5281/zenodo.17597482

**DOI:** https://doi.org/10.5281/zenodo.17597482

## Abstract

We employ machine learning techniques, specifically neural networks, to perform signal reconstruction by learning to map corrupted or degraded observations to their corresponding clean signals. The key idea is to train a neural network model to estimate the underlying clean signal from its corrupted version, leveraging the ability of deep networks to learn complex mapping functions from data. Our approach leads to a simple yet powerful conclusion: it is possible to learn how to restore images by only looking at corrupted examples, achieving equal performance and sometimes surpassing training on clean data, without requiring specific image prior or probability models of corruption.

In practice, we demonstrate that a single model can learn photographic noise removal, denoising of synthetic Monte Carlo images, and reconstruction of under sampled MRI scans - all corrupted by different processes - based solely on noisy data. The model learns these diverse image restoration tasks from corrupted observations alone, without needing clean training examples or explicit corruption models.

## Keywords

Machine Learning, Convolutional Neural Network(CNN), Data Analysis

## Introduction

Reconstructing signals from distorted or incomplete observations is an important research are a in statistical data analysis.

Recently, there has been significant interest in learning to map corrupted observations to their corresponding clean versions, as an alternative approach to the traditional explicit a priori statistical modelling of signal corruption. This interest is driven by the advancements made in deep neural networks.

The learning process is accomplished by minimizing the empirical risk through training a regression model, such as a convolutional neural network (CNN), on many pairs (xˆi, yi) of clean targets yi and corrupted inputs xˆi. The goal is to find the parameters θ of the mapping function fθ that minimizes the expected loss L over the training data, where the corrupted input xˆ is treated as a random variable distributed according to the conditional distribution p(xˆ|yi) given the clean target yi.

To emphasize that the corrupted input xˆ is a random variable whose distribution depends on the clean target, the notation xˆ ∼ p(xˆ|yi) is used, which indicates that xˆ is sampled from the conditional distribution p(xˆ|yi).

In this work, we observe that it is possible to learn how to convert corrupt images into clean ones simply by training on corrupted images alone, without requiring clean target images. Remarkably, this approach can perform equally well, and sometimes even better, than using explicit models trained on clean data. Furthermore, we do not need explicit numerical probability models of the corruption process or prior images, as these are implicitly learned from the training data itself. (Indeed, in one of our examples, the non-stationary noise cannot be characterized analytically.)

In addition to image denoising, our findings extend to inverse problems such as reconstructing MRI scans from under sampled data, even in an unsupervised setting without access to clean ground truth targets. Our results demonstrate that purely data-driven approaches, without strong statistical assumptions, can achieve impressive signal reconstruction performance.

Moreover, the increasing availability of large training datasets makes it much easier to learn highly effective signal mappings in a data- driven manner, alleviating the need for manually engineered prior or corruption models.

## Theoretical Background

Consider a scenario where we have a collection of faulty room temperature readings (y1, y2, ...). To estimate an unknown true value from a set of observations, a typical approach is to find a value z that minimizes the average discrepancy between z and the observed data, according to a chosen loss function. Mathematically, this involves finding z that minimizes the expected value of the loss function over the data distribution:

z = argmin E_y [L(z, y)]

Where L(z, y) is the loss function that quantifies the deviation or error between the estimate z and each observation y. The choice of the loss function determines how the discrepancies between z and the observations are measured and penalized.

The goal is to find the value of z that yields the smallest expected loss or average deviation from the entire set of observations, based on the specified loss criterion L(z, y). This formulation allows different loss functions to be used, capturing different notions of deviation or error, and the optimal z is the value that minimizes the average loss over the data distribution.

By framing it as an expected loss minimization problem, this approach provides a principled way to estimate an unknown quantity from noisy observations, with the loss function acting as the objective function that trades off different types of errors or discrepancies between the estimate and data.

For the L2 loss function, L(z, y) = (z - y)^2, the minimizer z is the arithmetic mean of the observations, z = E_{y}{y}.

On the other hand, for the L1 loss function, L(z, y) = |z - y|, which measures the total absolute deviations, the minimizer z is the median of the observations.

M-estimators, introduced by Huber (1964), provide a generalized class of estimators that minimize deviations based on different loss functions. By viewing the loss function choice, widely used summary statistics like the mean and median can be interpreted as machine learning estimators that aim to learn the true signal from noisy data.

This perspective allows us to unify classical statistical estimation methods and modern machine learning approaches under a common framework of empirical risk minimization using different loss functions.

Training neural network regressors can be viewed as a generalization of the point estimation process discussed earlier. Consider the standard training objective for a set of input- target pairs (xi, yi), where the network function fθ(x) is parameterized by θ:

θ* = argmin E_{(x,y)} L(fθ(x), y)

This optimization problem aims to find the parameters θ* that minimize the expected loss L(fθ(x), y) over the data distribution of (x, y) pairs.

Interestingly, if we remove the dependency on the input data x and use a trivial network fθ that simply outputs a learned scalar value, the problem reduces to the point estimation task in

(2). Conversely, the general training objective in (4) can be decomposed into the same minimization problem at each training sample; through simple manipulations, it can be shown that (4) is equivalent to:

θ* = argmin E_{x} E_{y} L(fθ(x), y)

This formulation highlights that the training process aims to minimize the expected loss over both the input x and target y distributions simultaneously.

Therefore, the neural network training process can be interpreted as a generalization of the classical point estimation problem, where instead of estimating a single scalar value, we are learning a flexible function fθ(x) that maps inputs to targets while minimizing the expected loss over the data distribution.

The neural network has the capability, in principle, to minimize the loss by solving the point estimation problem separately for each input sample. Consequently, the properties of the underlying loss function are inherited by the neural network training process.

A subtle point is often overlooked in the standard procedure of training regressors using Equation 1 over a finite set of input-target pairs (xi, yi): in reality, the mapping between inputs and targets is multi-valued, not the one-to-one mapping that is (falsely) assumed by that procedure. For instance, in a super-resolution task (Ledig et al., 2017) involving natural images, a

low-resolution input image x can correspond to multiple high-resolution images y, as the decimation process loses information about the precise locations and orientations of edges and textures. In other words, the highly complex distribution of natural images consistent with the low-resolution x is represented by the conditional distribution p(y|x).

When training a neural network regressor on pairs of low- and high-resolution images using the L2 loss, the network learns to output the average of the conditional distribution p(y|x) for a given input x. This is because the L2 loss encourages the network to predict the conditional mean of the target distribution, minimizing the expected squared error.

Interestingly, we have found that for some problems, an initially perceived disadvantage of the L2 loss can lead to unexpected advantages. A seemingly trivial and useless property of the L2 loss is that the estimates of the conditional expectation remain unchanged if we replace the target values with random numbers whose expectations match the original targets. This can be easily seen from Equation (3), which holds true even if the ys are drawn from a different distribution, as long as the expectations are preserved.

Consequently, the optimal network parameters θ in Equation (5) also remain unaffected if the input-conditional target distribution p(y|x) is replaced by an arbitrary distribution with the same conditional expected values. Without changing the learned mapping, we can decompose the training objectives by adding zero-mean noise to the targets.

By combining this property with the degenerate case of Equation (1), where the input is ignored, we are left with the task of empirical risk minimization over the input-conditional distribution of corrupted targets drawn from an arbitrary distribution with the same conditional expectations as the original targets.

In other words, we can train the neural network to map inputs to the conditional expectations of the true targets by minimizing the empirical risk over corrupted target samples, as long as the corruptions preserve the conditional expectations. This opens up the possibility of learning the desired mapping without requiring access to the clean target data, relying solely on the corrupted observations.

In this setting, the targets and inputs are drawn from a corrupted distribution, which need not be the same, conditioned on the underlying, unobserved clean target yi, such that the conditional expectation E{yˆi|xˆi} = yi.

θ* = argmin Σ L(fθ(xˆi), yˆi) θ i

Given infinite training data, the solution is the same as the original objective in Equation (1). For finite data, the variance of the estimator is proportional to the average variance of the target corruptions divided by the total number of training samples (see appendix).

Interestingly, none of the above depends on either a prior density model for the

underlying clean image manifold or a likelihood model of the corruption process. In other words, as long as we have access to data distributed according to p(noisy|clean) and p(clean), we do not require explicit formulations for either of these distributions.

The key requirement is that the corrupted inputs and targets are drawn from distributions that preserve the conditional expectations given the clean targets. With this condition satisfied, the neural network can learn to map the corrupted inputs to the conditional expectations of the clean targets by minimizing the empirical risk over the corrupted data, without needing explicit models for the clean data distribution or the corruption process.

In many image restoration tasks, obtaining corrupted input data is a natural by product of the problem we are trying to solve. Low-light photography is a prime example: a long exposure without noise is, on average, equivalent to a relatively short, noisy exposure. With this perspective in mind, the preceding analysis suggests that it is possible to learn how to remove photon noise simply by providing pairs of noisy images, without the need for complex imaging setups or expensive long exposures that would otherwise be required to obtain clean ground truth images.

Similar opportunities arise in other domains as well. For instance, with the L1 loss, which recovers the median of the values, a neural network can be trained to reconstruct an image from highly corrupted inputs where a significant portion (e.g., 50%) of the data is compromised by outliers, requiring only pairs of such corrupted images during training.

In the following sections, we present a diverse set of examples demonstrating that these theoretical capabilities of learning from corrupted data can be efficiently realized in practice across various image restoration tasks.

## Practical Experiments

The targets and inputs are drawn from a corrupted distribution (which can be different for targets and inputs), conditioned on the underlying, unobserved clean target yi, such that the conditional expectation E{yˆi|xˆi} = yi. The training objective can be formulated as:

$$\theta^* = \text{argmin} \sum L(f\theta(x\hat{}i), y\hat{}i)\ \theta\ i$$

Given an infinite amount of training data, the solution θ* is the same as the solution obtained from the original objective in Equation (1) using clean data. For a finite training dataset, the variance of the estimator θ* is proportional to the average variance of the target corruptions divided by the total number of training samples (see appendix for details).

Notably, the above formulation does not require an explicit density model for the underlying clean image distribution or a likelihood model for the corruption process. As long as the available data is distributed according to p (noisy|clean) and p(clean), we do not need to know or specify these distributions explicitly.

The key requirement is that the corrupted inputs and targets are drawn from distributions that preserve the conditional expectation relationship E{yˆi|xˆi} = yi with respect to the clean targets yi. With this condition satisfied, the neural network can learn to map the corrupted inputs to the conditional expectations of the clean targets by minimizing the empirical risk over the corrupted data, without needing explicit models for the clean data distribution or the corruption process.

## Additive Gaussian Noise

We begin by examining the effect of corrupted targets using synthetic additive Gaussian noise. Since the noise has zero mean, we employ the L2 loss function for training to estimate the mean.

We use three well-known datasets: BSD300 (Martin et al., 2001), SET14 (Zeyde et al., 2010), and KODAK. As summarized in Table 1, the behavior is qualitatively similar across all three datasets, and consequently, we report the averages. When trained using the standard approach with clean targets (Equation 1), RED30 achieves an average peak signal-to- noise ratio (PSNR) of 31.63 ± 0.02dB with noise standard deviation σ = 25. The confidence interval was computed by sampling 5 random initializations. The widely-used benchmark denoiser BM3D (Dabov et al., 2007) performs approximately 0.7dB worse. Interestingly, when we modify the training to use noisy targets (Equation 6) instead, the denoising performance remains equally high.

Moreover, the training converges just as quickly when using noisy targets, as shown in Figure 1a. This leads us to conclude that clean targets are unnecessary for this application of image denoising. This potentially surprising observation holds true even when using different network architectures and capacities. Figure 2a shows an example denoising result.

For the subsequent experiments, we replace the RED30 network with a shallower U-Net architecture (Ronneberger et al., 2015), which provides comparable denoising performance (around 0.2dB lower PSNR for Gaussian noise) but trains approximately 10 times faster. Details of the U-Net architecture and training parameters are provided in the appendix.

The rapid convergence achieved when training with noisy targets is understandable, as each training example effectively requires the network to learn to map one instance of noise to another instance of noise, which is a relatively easier task compared to mapping noisy inputs to clean targets.

Consequently, the training loss does not actually decrease as training progresses, and the loss gradients remain very high. However, the convergence speed does not seem to depend on these larger, noisier gradients. Although the activation gradients exhibit noise, the weight gradients exhibit a relatively clean appearance due to the independent and identically distributed (i.i.d.) Gaussian noise across all pixels.

Figure 1b makes the scenario more challenging by introducing inter-pixel correlation to the noise. This brown additive noise is obtained by blurring white Gaussian noise using a spatial Gaussian filter with varying bandwidths and scaling to maintain a noise standard deviation of σ = 25. An example is shown in Figure 1b. As the correlation increases, the effective averaging of weight gradients decreases, and the weight updates become noisier. This causes the convergence to become slower, but despite high levels of blur, the eventual peak performance remains similar (within 0.1dB of the uncorrelated case).

The key insight is that while inter-pixel noise correlations make the training process noisier and slower, the network can still effectively learn to map the correlated noisy inputs to their conditional expectations, given enough training data and iterations. The final denoising performance is only marginally affected by the presence of noise correlations.

In previous research, an abundance of noisy instances generated by introducing artificial noise into clear photos was relied upon. However, in practical scenarios with limited data and fixed funding for acquisition, we now contrast training data corrupted with clean data. Here's how our experiment is designed: A single "capture unit" (CU) comprises an Image Net image with white additive Gaussian noise at σ = 25. Assuming 19 CUs are needed for a clean capture, 20 CUs are used for one noisy realization plus the clean version (the mean of the 19 noisy realizations). Our overall goal is to stay within a total capture budget of 2000 CUs.

In our experiment, we use 100 training pairs (N = 100, M = 20), each consisting of one detected noise and its corresponding clean image (the average of 19 noise images). Initially, we observe that employing the same captured data as 100 * 20 * 19 = 38000 training pairs with degenerate targets, where each hidden value generates 19 * 20 possibilities as both noise/clean pairs, surprisingly yields slightly better results (0.1 dB fewer) than the traditional approach of fixed noisy+ clean pairs. However, only N = 100 pairs are retained in our case.

Furthermore, by setting N = 1000 and M = 2, i.e., increasing the number of pure hidden spots but having only two noisy noise detections for each (resulting from 2000 training pairs), we achieve even better results (again, by an improvement of 0.1 dB). Thus, we conclude, as shown in Figure 1c, that corrupted targets offer benefits—beyond just matching the performance of clean targets, they actually provide superior performance—due to two main factors: 1) exposure to more latent clean images, even if there are only two corrupted realizations of each, is advantageous; and 2) encountering more corruption realizations for the same latent clean image is beneficial.

## Other Synthetic Noises

We will now explore another type of artificial noise while following the training protocol outlined previously. In images, one prevalent source of noise is Poisson

noise. Once zero- mean is achieved, eliminating it becomes challenging as it depends on the sign. We adopt an L2 loss and vary the noise level $\lambda \in [0,50]$ during training. Training with a clean target yields a performance of 30.59 ± 0.02dB, while using a noisy target results in a similarly good performance of 30.57 ± 0.02dB. We determine this using the same convergence speed comparison method (Makitalo & Foi¨ , 2011) as with Gaussian noise (Anscombe transform), which turns on, denoises using BM3D, and then reverses the change, resulting in a 2dB loss. Various effects, such as quantization and dark current, predominantly caused by Poisson noise, can be rendered zero-mean (Hasinoff et al., 2016), and therefore do not pose issues during training on noisy targets.

We conclude that this application does not necessitate noise-free training data. However, it's important to note that saturation (or gamut clipping) removes a portion of the distribution, rendering the assumption inaccurate. Despite this limitation, saturation is generally undesirable for other reasons as well.

Binomial noise, also known as multiplicative Bernoulli noise, generates a random mask, where 0 represents zeroed or missing pixels, and 1 represents valid pixels. To prevent back propagating gradients from missing pixels, we exclude them from the loss function, as described by Ulyanov et al. (2017) in the context of their deep image prior (DIP).

Gaussian (σ = 25)

Poisson (λ = 30)

Bernoulli (p = 0.5)

Ground Truth Input Our Comparison

Figure 2 illustrates example results for Gaussian, Poisson, and Bernoulli noise. Our results were computed using noisy values, while the corresponding results with pure values are omitted since different comparison methods are employed for each noise type, although they are nearly identical in all three cases, as discussed throughout the text.

The probability of a corrupted pixel is represented by p; during training, we vary $p \in [0.0, 0.95]$, while during testing, p is set to 0.5. Training with clean targets yields an average performance of 31.85 ± 0.03dB, while using noisy targets

(separating m for input and target) results in a slightly higher performance of 32.02 ± 0.03dB, possibly because noisy targets facilitate skipping a method that performs well (Srivastava et al., 2014) on the grid output. DIP, however, performs about 2dB worse—DIP is not a learning-based solution and is quite different from our approach, but it shares the property that neither a clean model nor an explicit model of corruption is necessary. We utilized the "Image reconstruction" procedure as described in the DIP supplementary material. Figure 3 demonstrates blind text removal.

The corruption consists of a significant, fluctuating amount of random strings arranged in irregular locations, sometimes overlapping. Additionally, the font's color and size are randomly selected. Both the font and the string orientation remain consistent. In the realm of Monte Carlo Rendering, physically accurate renderings of virtual environments are typically achieved through a process called Monte Carlo path tracing. This involves generating random sequences of scattering events ("light paths") in the scene, connecting light sources and virtual sensors, and integrating the radiance carried by these paths over all possible trajectories (Veach & Guibas, 1995). The Monte Carlo integrator is designed such that the intensity of each pixel represents the expectation of the random path sampling process, meaning the sampling noise is zero-mean. However, despite extensive research into importance sampling techniques, the distribution remains difficult to characterize. It varies from pixel to pixel, heavily relies on the scene configuration and rendering parameters, and can exhibit arbitrarily complex multimodal patterns. Additionally, certain lighting effects, like focused caustics, produce highly skewed distributions with rare, bright outliers.

These complexities make removing Monte Carlo noise significantly more challenging

compared to, for instance, Gaussian noise removal. Nevertheless, this problem is somewhat mitigated by the potential to generate auxiliary information that empirically correlates with the clean result during data generation. In our experiments, the denoiser input comprises not only per-pixel luminance values but also the average albedo (texture color) and normal vector of the surfaces visible at each pixel. In the context of High Dynamic Range (HDR) images, even with sufficient sampling, the luminance values of floating- point pixels can vary by several orders of magnitude. To render these images suitable for display on standard 8-bit devices, the high dynamic range needs to be compressed to a fixed range using a tone mapping operator, such as a variant of Reinhard's global operator (Reinhard et al., 2002): $T(v) = (v/(1 + v))^{1/2.2}$, where v is a scalar luminance value, possibly pre-scaled with an image-wide exposure constant. This operator maps any $v \geq 0$ into the range $0 \leq T(v) < 1$.

However, the combination of the virtually unlimited range of luminances and the nonlinearity of the tone mapping operator poses a challenge. If we attempt to train a

denoiser using the mean squared error (MSE) loss L2 = (fθ(xˆ) − yˆ)^2, where fθ(xˆ) represents the denoiser's output and yˆ denotes the target, the presence of long-tail effects (outliers) in the targets can overshadow the normal MSE loss, leading to convergence issues. Moreover, the nonlinearity of the tone mapping operator can cause the expected value of noisy target images, E{T(v)}, to deviate from the clean training target, T(E{v}), resulting in inaccurate predictions if the denoiser produces tonemapped values T(v).

A commonly used metric for evaluating HDR image quality is the relative mean squared error (Rousselle et al., 2011), where the squared difference is divided by the square of the calculated brightness of the pixel. However, this metric also suffers from the nonlinearity issue associated with tone mapped output. To address this, we propose using the grid output, which tends to a valid value within the limits of the denominator:

LHDR = (fθ(xˆ) − yˆ)^2 / (fθ(xˆ) + 0.01)^2

We can demonstrate that LHDR converges exactly to the expected value as long as we consider the gradient of the denominator to be zero.

Top of Form

In our approach, we found it beneficial to tone map the image input T(xˆ) instead of using the HDR input directly. This allows the network to continuously generate non-tone mapped (linearly scaled) light values while maintaining the accuracy of the expected value. Figure 6 illustrates the loss functions applied to photos produced by Monte Carlo noise reduction.

For our experiments, we utilized 64 samples per pixel (spp) in Monte Carlo path-traced images to train a denoiser. Our training set consisted of 860 architectural photos, while an additional 34 photographs from a different set of scenarios were reserved for validation. We rendered three alternative versions of the training images: one with 131k spp (clean target), and two with 64 spp (noisy input, noisy target) using various random seeds. Both the input version (64 spp) and the reference version (131 spp) of the validation images were generated. Each image had a resolution of 960 by 540 pixels, and as mentioned earlier, we retained the normal and albedo buffers for every input image. Rendering the crisp images with 131k spp posed a challenge given the limited collection size.

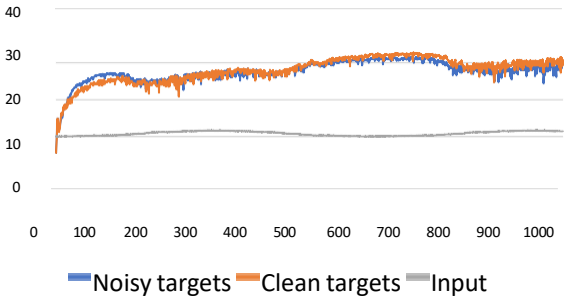
The average Peak Signal-to-Noise Ratio (PSNR) of the 64 spp validation inputs relative to the corresponding reference photos was measured to be 22.31dB.

(Refer to Figure 7a for an example.) The network, trained for 2000 epochs using clean target images, achieved a mean PSNR of 31.83dB on the validation set, while the additionally trained network using noisy target images yielded 0.5dB less. Examples of results are depicted in Figure 7b and 7c. The training process took 12 hours utilizing a single NVIDIA Tesla P100 GPU.

After 4000 epochs, the PSNR for the noisy targets matched 31.83dB, indicating that noisy targets took approximately twice as

long to converge. However, the gap between the two approaches had not significantly narrowed, leading us to believe that some quality difference may persist even within the limit.

Top of Form




## Magnetic Resonance Imaging (MRI)

Volumetric images of biological tissues are generated using magnetic resonance imaging (MRI), which essentially samples the signal's Fourier transform, known as "k-space." Compressed sensing (CS) techniques have long been employed in modern MRI to overcome the Nyquist-Shannon limit by under sampling k- space and performing non-linear reconstruction that exploits the image's sparsity in a suitable transform domain (Lusting et al., 2008).

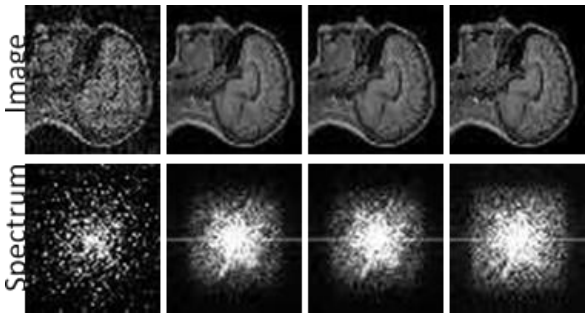
Our approach remains applicable if we conceptualize the k-space sampling as a random process with a specified probability density p(k) over the frequencies k. Specifically, we model the k-space sampling process as a Bernoulli process, where the probability of selecting a frequency for acquisition is determined by p(k) = e^(-λ|k|) for each unique frequency. Frequencies that are not selected are set to zero, and the selected frequencies are weighted by the inverse of the selection probability. This "Russian roulette" approach ensures the desired under sampling. The parameter λ dictates the total fraction of k-space maintained; in our tests, we set it to retain 10% of the samples compared to a complete Nyquist-Shannon sampling.

To reconstruct the primal image, we transform the under sampled spectra. Next, we define the regression task according to equation (6) and utilize a convolutional neural network (CNN) with two separate, un-under sampled images, denoted as xˆ and yˆ, both of identical dimensions. Additionally, we enhance the result by Fourier transforming the output fθ(xˆ), replacing it with frequencies from the input, converting it back to the primary field before calculation. This approach allows precise control over the frequencies xˆ of the input image.

The final loss function (F−1(Rxˆ(F(fθ(xˆ). )))) is formulated as (− yˆ)^2, where R represents the process of substituting a zero from the input. This entire process is trained end to end. We conducted our studies using 2D slices extracted from the IXI brain scan MRI dataset. To simulate spectrum sampling, we randomly sampled from the Fast Fourier Transform (FFT) of the already reconstructed images in the dataset. Consequently, our data is real-valued and takes into account the discrete FFT's periodicity, which differs from actual MRI samples.

The training set consisted of a total of 5000 256 x 256 resolution images from 50 individuals, while for validation, we randomly selected 1000 images from 10 distinct subjects. The baseline Peak Signal-

to-Noise Ratio (PSNR) was measured at 20.03dB when poorly sampled input images were immediately reconstructed using Inverse Fast Fourier Transform (IFFT). The network required 300 epochs to complete training.




(a) Input (b) Noisy trg. (c)Clean trg.

(d) Reference

18.93dB 29.77dB 29.81dB

Figure nine showcases an MRI reconstruction example. (a) Depicts the input image with the best 10% of spectrum samples retained and scaled by means of 1/p. (b) Please display the output generated by a neural network trained using noisy target images, which should resemble the input picture. (c) Demonstrates the same reconstruction process as in (b), but with training performed using clean target images similar to the reference photograph. (d) Displays the original, uncorrupted image. PSNR values refer to the images depicted here; for averages over the entire validation set, please refer to the text.

The network trained with clean targets achieved a PSNR of 31.77dB, while the network trained with noisy targets achieved an average PSNR of 31.74dB on the validation data. This training approach with clean targets aligns with previous research conducted by Wang et al. (2016) and Lee et al. (2017). Training on an NVIDIA Tesla P100 GPU required 13 hours. Figure 9(b, c) presents an example of reconstruction outcomes comparing convolutional networks trained with clean and noisy targets, respectively. Our results are consistent with recent work in terms of PSNR.

## References

1. Ashish Bora, Eric Price, Alexandros G. Dimakis. AmbientGAN: Generative models from lossy measurements. ICLR, 2018.
2. Cerda-Company, Xim, P´ arraga, C. Alejandro, and Otazu,´ Xavier. Which tone-mapping operator is the best? A comparative study of perceptual quality. CoRR, abs/1601.04450, 2016.
3. Chaitanya, Chakravarty R. Alla, Kaplanyan, Anton S., Schied, Christoph, Salvi, Marco, Lefohn, Aaron, Nowrouzezahrai, Derek, and Aila, Timo. Interactive reconstruction of Monte Carlo image sequences using a recurrent denoising auto encoder. ACM Trans. Graph., 36 (4):98:1–98:12, 2017.
4. Dabov, K., Foi, A., Katkovnik, V., and Egiazarian, K. Image denoising by sparse 3-D transform-domain collaborative filtering. IEEE Trans. Image Process., 16(8):2080–2095, 2007.
5. Good fellow, Ian, Pouget-Abadie, Jean, Mirza, Mehdi, Xu,
6. Bing, Warde-Farley, David, Ozair, Sherjil, Courville, Aaron, and

Bengio, Yoshua. Generative Adversarial Networks. In NIPS, 2014.

7. Hasinoff, Sam, Sharlet, Dillon, Geiss, Ryan, Adams, Andrew, Barron, Jonathan T., Kainz, Florian, Chen, Jiawen, and Levoy, Marc. Burst photography for high dynamic range and low-light imaging on mobile cameras. ACM Trans. Graph., 35(6):192:1–192:12, 2016.
8. He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, and Sun,
9. Jian. Delving deep into rectifiers: Surpassing humanlevel performance on imagenet classification. CoRR, abs/1502.01852, 2015.