Self-Validation: Early Stopping for Single-Instance Deep Generative Priors

Taihui Li[1]
lixx5027@umn.edu
Zhong Zhuang[2]
zhuan143@umn.edu
Hengyue Liang[2]
liang656@umn.edu
Le Peng[1]
peng0347@umn.edu
Hengkang Wang[1]
wang9881@umn.edu
Ju Sun[1]
jusun@umn.edu

[1] Department of Computer Science and Engineering
University of Minnesota
Minnesota, USA

[2] Department of Electrical and Computer Engineering
University of Minnesota
Minnesota, USA

Abstract

Recent works have shown the surprising effectiveness of deep generative models in solving numerous image reconstruction (IR) tasks, *even without training data*. We call these models, such as deep image prior and deep decoder, collectively as *single-instance deep generative priors* (SIDGPs). The successes, however, often hinge on appropriate early stopping (ES), which by far has largely been handled in an ad-hoc manner. In this paper, we propose the first principled method for ES when applying SIDGPs to IR, taking advantage of the typical bell trend of the reconstruction quality. In particular, our method is based on collaborative training and *self-validation*: the primal reconstruction process is monitored by a deep autoencoder, which is trained online with the historic reconstructed images and used to validate the reconstruction quality constantly. Experimentally, on several IR problems and different SIDGPs, our self-validation method is able to reliably detect near-peak performance and signal good ES points. Our code is available at https://sun-umn.github.io/Self-Validation/.

1 Introduction

Validation-based ES is one of the most reliable strategies for controlling generalization errors in supervised learning, especially with potentially overspecified models such as in gradient boosting and modern DNNs [10, 28, 47]. Beyond supervised learning, ES often remains critical to learning success, but there are no principled ways—universal as validation for supervised learning—to decide when to stop. In this paper, we make the first step toward filling in the gap, and focus on solving IR, a central family of inverse problems, using *training-free deep generative models*.

List of Acronyms (alphabetic order)	
AE	Autoencoder
CNN	convolutional neural network
DD	deep decoder
DGP	deep generative prior
DIP	deep image prior
DL	deep learning
DNN	deep neural network
ES	early stopping
IR	image reconstruction
MIDGP	multi-instance DGP
OPT	optimization
PG	PSNR gap
SG	SSIM gap
SIDGP	single-instance DGP
SIREN	sinusoidal representation network

IR entails estimating an image of interest, denoted as x, from a measurement $y = f(x)$, where f models the measurement process. This model covers classical image processing tasks such as image denoising, super-resolution, inpainting, deblurring, and modern computational imaging problems such as MRI/CT reconstruction [8] and phase retrieval [33, 40]. In this paper, *we assume that f is known*. Classical approaches normally formulate IR as a regularized data fitting problem (Ref. problem (1)) and solve it via iterative optimization algorithms [27]. Recently, DL-based methods have been developed to either directly approximate the inverse mapping f^{-1}, or enhance classical optimization algorithms for solving problem (1) by integrating pretrained or trainable DNN modules (e.g., plug-and-play or network unrolling; see the recent survey [25]). But, they invariably require extensive and representative training data. In this paper, we focus on an emerging lightweight approach *that requires no extra training data*: the target x is directly modeled via DNNs.

$$\min_x \ \ell(y, f(x)) + \lambda R(x) \quad \ell\text{: fitting loss, } R\text{: regularization, } \lambda\text{: regularization parameter} \tag{1}$$

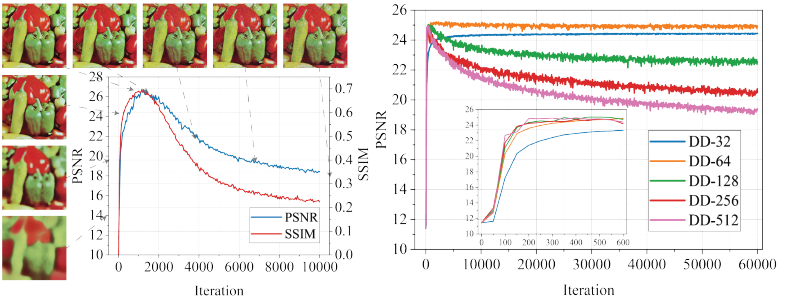

Figure 1: Illustration of the overfitting issue of DIP and DD on image denoising with Gaussian noise (noise level: $\sigma = 0.18$; pixel values normalized to $[0,1]$). The reconstruction quality (as measured by both PSNR and SSIM) typically follows a skewed bell curve: before the peak only the clean image content is recovered, but after the peak noise starts to kick in. Note that DD is not free from overfitting when the network is increasingly over-parametrized (indicated by the number following "DD-" in the legend).

Single-instance deep generative priors Many deep generative models can be used for modeling image collections [10, Part III]. For modeling single images, two families of models stand out: 1) *structural models*[1]: the image is modeled as $x \approx G_\theta(z)$. Here, z is a fixed or trainable seed/code, and G is a trainable DNN—often taken as CNN to bias toward structures in natural images—parameterized by θ. DIP [41] and DD [12] are two exemplars in this category, and they only differ in the choice of architecture for G. A variant [26] allows initializing G with a pretrained model. These models should be contrasted with *multi-instance DGPs* (MIDGPs, of which GAN inversion [2, 48] is a special case), where G is pretrained and frozen but z is trainable. Although MIDGPs have also been used to tackle IR

[1]Some authors call this *untrained DNN models/priors*; see [29]. But this may be confused with SIDGPs with prefixed but frozen G (e.g., random) as in [8, 10].

problems [8, 11], pretraining G requires massive training sets that can cover the target domains. Thus, MIDGPs are not suitable for single-image settings[2]. 2) *functional models*: the (discrete) image x is modeled as the collection of uniform-grid samples, i.e., discretization, from an underlying *continuous* function $\bar{x}$ supported on a bounded region in $\mathbb{R}^2$ (sometimes higher dimensions)[3]. Now $\bar{x}$ is parameterized as a DNN $\bar{x}_\theta$, i.e., $x = \mathcal{S}(\bar{x}_\theta)$ where $\mathcal{S}$ denotes the sampling process. Note that if $\bar{x}$ can be learned, in principle one can obtain arbitrary resolutions for the discrete version x—attractive for computer vision/graphics applications. These models are called implicit neural representation or coordinate-based multilayer perceptron (MLP) in the literature, of which sinusoidal representation networks (SIRENs, [35]) and MLP with Fourier features [39] are two representative models.

We categorize all these single-image models into the family of SIDGPs. Substituting any of the SIDGPs into problem (1) and removing the explicit regularization term R, we can now solve IR problems by DL:

$$\textbf{structural OPT}: \min_\theta \ell(y, f(G_\theta(z))) \quad \text{or} \quad \textbf{functional OPT}: \min_\theta \ell(y, f(\mathcal{S}(\bar{x}_\theta))). \tag{2}$$

This simple approach is surprisingly effective in solving numerous IR problems, ranging from classical image restoration [12, 41], to advanced computational imaging problems [8, 7, 9, 11, 31, 35, 39, 45], and even beyond [22, 30]; see the recent survey [29].

The overfitting issue There is a caveat to all the claimed successes: overfitting. In practice, we have $y \approx f(x)$ instead of $y = f(x)$ due to various kinds of measurement noise and model misspecification, and the DNN used in (2) is also often (substantially) overparametrized. So when the objective in Eq. (2) is globally optimized, the final reconstruction $G_\theta(z)$ or $\mathcal{S}(\bar{x}_\theta)$ may account for the noise and model errors, alongside the desired image content. This overfitting phenomenon has indeed been observed empirically on all SIDGPs when noise is present. Fig. 1 shows the typical reconstruction quality of DIP and DD for image denoising (Gaussian noise) over iterations. Note the iconic skewed bell curves here: the reconstruction quality initially monotonically climbs to a peak level—noise effect is almost invisible during this period, and then monotonically degrades where the noise effect gradually kicks in until everything is fit by the overparametrized models. Also, when the level of overparametrization grows, overfitting becomes more serious (Fig. 1 (b) for DD). The overfitting phenomenon has also been partly justified theoretically: despite the typical nonconvexity of the objectives in Eq. (2), global optimization is feasible and happens with high probability [13, 16]. But it is a curse here.

Prior work on mitigating overfitting Several lines of methods have been developed to remove the overfitting. DD [12] proposes to control the network size for structural models, and shows that overfitting is largely suppressed when the network is suitably parameterized. However, choosing the right level of parameterization for an unknown image with unknown complexity is no easy task, and underparameterization can apparently hurt the performance, as acknowledged by [12]. So in practice people still tend to use overparametrization and counter overfitting with ES [14]; see Fig. 1 (b). The second line adds regularization. [36, 37] plug in total-variation or other denoising regularizers, and [8] appends weight-decay and runs SGD with Langevin dynamics (i.e., with additional noise) [42]. But their evaluations are limited to denoising with low Gaussian noise; our experiments in Section 3 show that overfitting still exists when the noise level is higher. The third line explicitly models the

[2] See [21] that tries to bridge the two regimes though.

[3] In a sense, this is inverting the discretization process in typical image formation. Also, the idea of modeling continuous functions directly is popular in DL for solving PDEs [43].

noise in the objective, so that hopefully both the image x and the noise are exactly recovered when the objective is globally optimized. [46] implements such an idea for additive sparse corruption. The overfitting is gone, but it is unclear how to generalize the algorithm, which is delicately designed for sparse corruption only, to other types of noise.

Our contribution In this paper, we take a different route, and capitalize on overfitting instead of suppressing or even killing it. Specifically, we leave the problem formulation (2) as is, and propose a reliable method for detecting near-peak performance, which is *first of its kind* and also 1) **effective**: empirically, our method detects near-peak performance measured in both PSNR and SSIM; 2) **fast**: our method performs ES once a near-peak performance is detected—which is often early in the optimization process (see Fig. 1), whereas the above mitigating strategies push the peak performance to the final iterations; 3) **versatile**: the only assumption that our method relies on is the performance curve (either PSNR or SSIM) is (skewed) bell-shaped. This seems to hold for the various structural and functional models, tasks, noise types that we experiment with; see Section 3.

After the initial submission, we became aware of two new works [18, 34] that also perform ES. [34] proposes an ES criterion based on a blurriness-to-sharpness ratio for natural images, but it only works for the modified DIP and DD they propose, not the original and other SIDGPs. [18] focuses on Gaussian noise, and designs an elegant Gaussian-specific regularized objective to monitor progress and decide ES. Compared to our approach, both methods are limited in versatility. We defer a detailed comparison with them to future work.

2 Early stopping via self-validation

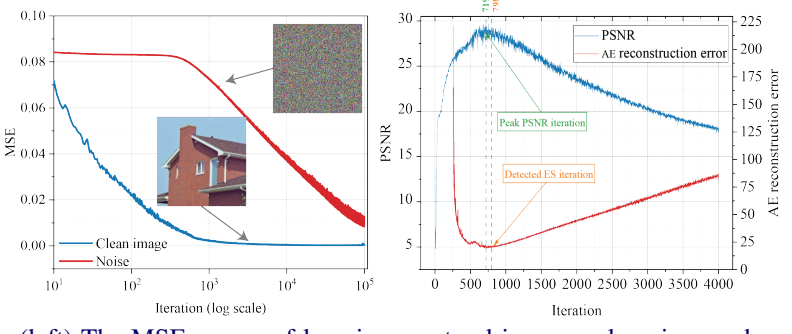

Figure 2: (left) The MSE curves of learning a natural image vs learning random noise by DIP. DIP fits the natural image much faster than noise. This induces the typical bell-shaped reconstruction quality curves (as shown in Fig. 1 and (right) here) when fitting noisy images using DIP; (right) The PSNR curve vs our online AE reconstruction error curve when fitting a noisy image with DIP. The peak of the PSNR curve is well aligned with the valley of the AE error curve. So by detecting the valley of the latter curve, we are able to detect near-peak points of the former—corresponding to reconstructed images of near-peak quality.

In this section, we describe our detection method taking structural SIDGPs as an example; this can be easily adapted for functional SIDGPs.

Bell curves: the why and the bad When $y = f(x)$, solving the structural OPT in (2) with an overparameterized CNN G_θ using gradient descent has disparate behaviors on natural images vs pure noise: the term $G_\theta(z)$ fits natural images much more rapidly than fitting pure noise. This is illustrated in Fig. 2 (left) and has been highlighted in numer-

ous places [12, 13]; it can be explained by the strong bias of convolutional operations toward natural image structures, which are low-frequency dominant, during the early learning stage [12, 13]. When $y \approx f(x)$, e.g., the measurement is noisy, intuitively $G_\theta(z)$ would quickly fit the image content with little noise initially, and then gradually fit more noise until reaching the full capacity. This induces the skewed bell curves as shown in Fig. 1 and Fig. 2 (right). Of course, this argument is handwavy and the fitting effect might not be decomposable into that of image content and noise separately for complicated f's and noise types (it has been rigorously established for simple cases, e.g., [13]). Nonetheless, the bell-shaped performance curves are also observed for nonlinear f's with complex noise types, e.g., [4, 20], as well as functional models as we show in Section 3. The downside is overfitting, if no suitable ES is performed. Recent methods that mitigate the overfitting issue defer the performance peak until the final convergence and work only in limited scenarios.

Bell curves: the good Here we do not try to alter the bell curves but turn them into our favor, as explained below. In practice, we do not directly observe these curves, as they depend on the groundtruth x. Let $\{\theta_k\}$ denote the iterate sequence and $\{x_k\}$ the corresponding reconstruction sequence, i.e., $x_k = G_{\theta_k}(z)$. We only observe $\{x_k\}$ and $\{\theta_k\}$ directly, and we know that there exists a $k^* \in \mathbb{N}$ so that the subsequence $\{x_k\}_{k\le k^*}$ has increasingly better quality and contains very little noise.

Our first technical idea is constructing a quality oracle by training an AE

$$\min_{w,v} \sum_{i=1}^{n} \ell_{\mathrm{AE}}(s_i, d_v \circ e_w(s_i)) \quad e_w\text{: encoder DNN} \quad d_v\text{: decoder DNN}, \tag{3}$$

where $\{s_i\}_{i\le n}$ is a given training set. AEs are now a seminal tool for manifold learning and nonlinear dimension reduction [10, Chap. 14]. When $\{s_i\}_{i\le n}$ is sampled from a low-dimensional manifold, after training we expect that $s \approx d_v \circ e_w(s)$ for any s from the same manifold and $s \not\approx d_v \circ e_w(s)$ otherwise. This implies that we can use the AE reconstruction loss $\ell_{\mathrm{AE}}(s, d_v \circ e_w(s))$ as a proxy to tell if a novel data point s comes from the training manifold. Thus, if the training set $\{s_i\}_{i\le n}$ consists of images very close to x, we can use $\ell_{\mathrm{AE}}(s, d_v \circ e_w(s))$ to score the reconstruction quality of any s: higher the loss, lower the quality, and vice versa. If we apply this ideal AE to $\{x_k\}$, the AE loss curve will have an inverted bell shape, where the valley corresponds to the quality peak. So we can detect the peak of the quality curve by detecting the valley of the AE loss curve.

But it is unclear how to ensure that $\{s_i\}_{i\le n}$ be uniformly close to x. Our second technical idea is exploiting the monotonicity of reconstruction quality in $\{x_k\}_{k\le k^*}$ and $\{x_k\}_{k\ge k^*}$ to train a sequence of AEs. We take a consecutive length-n subsequence of $\{x_k\}_{k\le k^*}$ to train an AE each time, and test the next reconstructed image against the resulting AE. Although the quality of any initial training set can be low, resulting in poor AEs, it improves over time and culminates when getting near the x_{k^*}. Similarly, afterward, the quality gradually degrades. Away from the peak x_{k^*}, the quality of x_k's changes rapidly over iterations, and so we expect high AE loss, regardless of the quality of the AE training set. In contrast, around x_{k^*} all x_k's are of high quality, leading to near-ideal AEs and low AE losses. Thus, even after the practical modification, we expect an inverted bell shape in the AE loss curve, and alignment of the quality peak with the loss valley again, as confirmed in Fig. 2 (right).

Our method: ES by self-validation We now only need to detect the valley of the AE loss curve due to the blessed alignment discussed above. To sum up our algorithm: we train the structural OPT and an AE side-by-side, feed the most recent n reconstructed images to train the AE, and then test the next image for AE loss. We make two additional modifications

Algorithm 1 Structural OPT with ES for IR

Input: $y, f, \ell, \ell_{\text{AE}}, G, \theta^{(0)}$ (of Eq. (2)), $w^{(0)}, v^{(0)}$ (of Eq. (3)), $k = 0$, window size n, patience number p

while not stopped **do**
  update θ via one iterative step on Eq. (2) to obtain $\theta^{(k+1)}$ and $x^{(k+1)}$
  update w, v via one iterative step on Eq. (3) using $\{x_i\}_{i=k-n+1}^{k}$ to obtain $w^{(k+1)}, v^{(k+1)}$
  calculate the reconstruction loss $\ell_{\text{AE}}(x^{(k+1)}, d_{v^{(k+1)}} \circ e_{w^{(k+1)}}(x^{(k+1)}))$
  if no improvement in the reconstruction loss in p consecutive iterations **then**
    early stop and exit
  end if
  $k = k + 1$
end while

Output: the reconstructed image $x^{(k-p+1)}$

for better efficiency: 1) the AE is updated only once per new training set instead of being greedily trained, i.e., an online learning setting. This helps to substantially cut down the learning cost and save hyperparameters while remaining effective; and 2) to ensure the AEs learn the most compact representations, we borrow the idea of IRMAE [17] and add several trainable linear layers in the AEs before feeding the hidden code to the decoder. Our whole algorithmic pipeline is summarized in Algorithm 1.

3 Experiments

Setup We experiment with 3 SIDGPs (DIP [4], DD[5], SIREN[6]) and 4 IR problems (image denoising, inpainting, MRI reconstruction, and image regression). Unless stated otherwise, we work with their default DNN architectures, optimizers, and hyperparameters. We fix the window size $n = 256$, and the patience number $p = 500$ for denoising and inpainting and $p = 200$ for MRI reconstruction and image regression. For training the collaborative AEs, ADAM is used with a learning rate 10^{-3}. To assess the IR quality, we use both PSNR [5] and SSIM [42]. To evaluate our detection performance, we use ES-PG which is the absolute difference between the detected and the true peak PSNRs; similarly for ES-SG. Sometimes we also report BASELINE-PG which measures the difference between the peak and the final overfitting PSNRs; similarly for BASELINE-SG. For most small-scale experiments, we repeat all experiments 3 times and report means and standard deviations. **All omitted details and experimental results can be found in the Appendix**.

Image denoising The power of DIP and DD was initially only demonstrated on Gaussian denoising. Here, to make the evaluation more thorough, we also experiment with denoising impulse, shot, and speckle noise, on a standard image denoising dataset[7] (9 images). For each of the 4 noise types, we test a low and a high noise level, respectively. The mean squares error (MSE) is used for Gaussian, shot, and speckle noise, while the ℓ_1 loss is used for impulse noise. To obtain the final degraded results, we run DIP for 150K iterations.

The denoising results are measured in terms of the gap metrics are summarized in Fig. 3. Note that our typical detection gap is ≤ 1 measured in ES-PG, and ≤ 0.1 measured in ES-SG. If DIP just runs without ES, the degradation of quality is severe, as indicated by both BASELINE-PG and BASELINE-SG. Evidently, our DIP+AE can save the computation and

[4] https://github.com/DmitryUlyanov/deep-image-prior
[5] https://github.com/reinhardh/supplement_deep_decoder
[6] https://vsitzmann.github.io/siren/
[7] http://www.cs.tut.fi/~foi/GCF-BM3D/index.html#ref_results

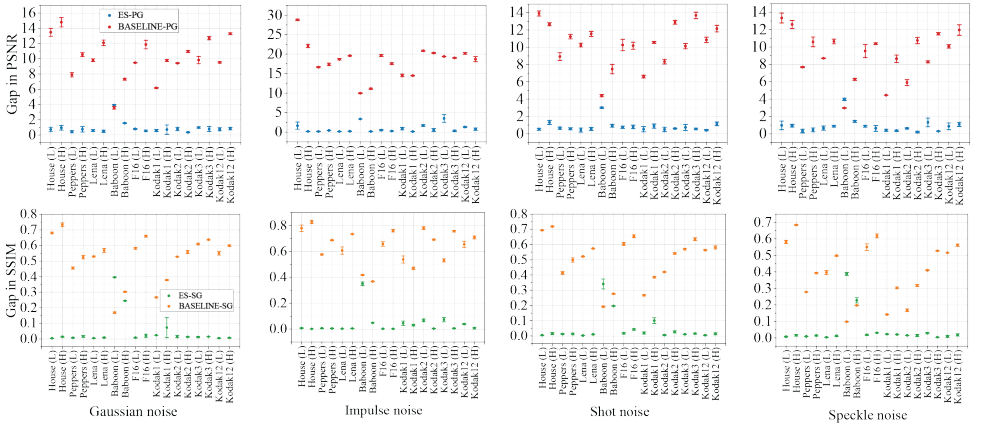

Figure 3: DIP+AE for image denoising. Each columns represents a distinct noise type, and the 1st row contains the PGs, and 2nd in SGs. The L and H in the horizontal annotations indicate low and high noise levels, respectively.

the reconstruction quality, and return an estimate with near-peak performance for almost all images, noise types, and noise levels that we test. The Baboon image is an outlier. It contains substantial high-frequency textures, and actually the peak PSNR/SSIM itself is also much worse compared to other images. We suspect that dealing with similar kinds of images can be challenging for DIP and DD and our detection method, which warrants future study.

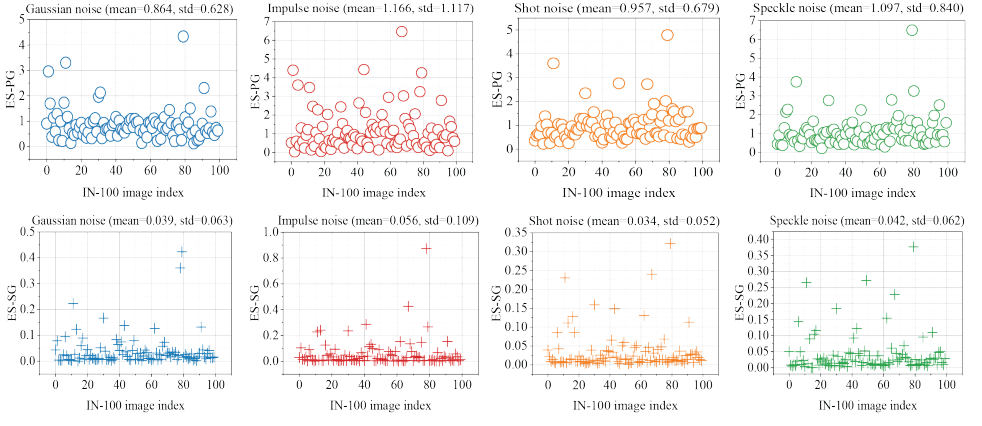

Figure 4: DIP+AE on IN-100. 1st row: ES-PGs; 2nd row: ES-SGs.

We further test our method on 100 randomly selected images from ImageNet [37], denoted as IN-100. We follow the same evaluation protocol as above, except that we only experiment a medium noise level and we do not estimate the means and standard deviations; the results are reported in Fig. 4. It is easy to see that the ES-PGs are concentrated around 1 and the ES-GSs are concentrated around 0.1, consistent with our observation on the small-scale dataset above.

We compare DIP+AE with three other competing methods that try to eliminate overfitting, i.e., DIP+TV [36, 37] and SGLD [44] on Gaussian denoising, and DOP [46] on removing sparse corruptions. We take a medium noise level. All these methods claim to eliminate the overfitting altogether, and so we directly compare the detected PSNRs (SSIMs) by our method and the final PSNRs returned by their methods at the final convergence. The results

Table 1: DIP+AE, DIP+TV, and SGLD on Gaussian denoising. The best PSNRs are colored as red; the best SSIMs are colored as blue.

	PSNR ↑			SSIM ↑		
	DIP+AE	DIP+TV	SGLD	DIP+AE	DIP+TV	SGLD
House	30.824(0.236)	18.866(0.093)	21.800(2.430)	0.834 (0.008)	0.174 (0.006)	0.495 (0.152)
Peppers	26.471 (0.249)	19.000 (0.004)	20.516 (0.263)	0.688 (0.008)	0.238 (0.001)	0.453 (0.017)
Lena	27.462 (0.602)	18.961 (0.028)	20.217 (0.808)	0.744 (0.003)	0.253 (0.001)	0.447 (0.051)
Baboon	19.453 (0.251)	18.416 (0.024)	19.111 (0.091)	0.335 (0.008)	0.467 (0.0)	0.572 (0.003)
F16	27.644 (0.114)	19.183 (0.122)	20.431 (0.091)	0.818 (0.005)	0.248 (0.005)	0.457 (0.008)
Kodak1	24.455 (0.111)	18.688 (0.088)	19.555 (0.049)	0.649 (0.006)	0.409 (0.006)	0.528 (0.003)
Kodak2	26.886 (0.218)	19.535 (0.160)	20.569 (0.629)	0.677 (0.006)	0.243 (0.014)	0.430 (0.046)
Kodak3	27.894 (0.311)	18.904 (0.181)	19.921 (0.194)	0.761 (0.002)	0.187 (0.008)	0.379 (0.012)
Kodak12	28.269 (0.239)	18.998 (0.183)	19.974 (0.240)	0.727 (0.003)	0.193 (0.011)	0.391 (0.015)

are tabulated in Table 1 and Table 2, respectively. It seems that 1) DIP+TV and SGLD do not quite eliminate overfitting when different images and noise levels (noise levels used in the respective papers are relatively low) than they experimented with are used; and 2) our detection method, i.e., DIP+AE, returns far better reconstruction than DIP+TV and SGLD (except on Baboon). For the comparisons with DOP (presented in Table 2), our method (DIP+AE) can detect reasonable good ES points, which is consistent with the finding in Fig. 3, and requires far fewer optimization iterations compared with that of DOP. On the other hand, the detected performance in terms of PSNR and SSIM by our method slightly lag behind that of DOP, which however is expected as there is an inherent performance gap between DIP and DOP, which has been confirmed in [46]. Nevertheless, our purpose in this work is finding an appropriate ES points for DIP instead of improving its performance, we thus leave how to boost DIP performance as a future research direction. Furthermore, Fig. 9 in the Appendix visualizes the reconstruction results of both DIP+AE and DOP. Although there is a slightly disparity in PSNR values between DIP+AE and DOP, visually they lead to similar reconstruction qualities and there is almost no perceivable differences.

Table 2: DOP vs. DIP+AE on removing sparse corruptions

Metrics	Models	House	Peppers	Lena	Baboon	F16	Kodak1	Kodak2	Kodak3	Kodak12
PSNR ↑	DOP	41.688	31.227	32.547	21.802	31.300	26.626	31.427	32.206	32.223
	DIP+AE	38.947	28.260	30.683	19.392	28.684	24.933	30.358	30.592	30.790
SSIM ↑	DOP	0.957	0.823	0.869	0.614	0.925	0.777	0.821	0.887	0.855
	DIP+AE	0.946	0.777	0.836	0.357	0.887	0.698	0.770	0.855	0.819
Iteration ↓	DOP	149041	84583	90427	36535	87824	63533	92821	76279	84812
	DIP+AE	3038	1419	1665	381	1711	1184	1026	1183	1397

We further compare our method with a group of baseline methods based on no-reference image quality metrics: the classical BRISQUE [23] and NIQE [24], and a state-of-the-art, NIMA (both technical quality and aesthetic assessment) [38] which is based on pretrained DNNs. We experiment with medium noise level across all four noise types with DIP, run DIP until the final convergence (i.e., overfitting), and then select the highest quality one from among all the intermediate reconstructions according to each of the three metrics, respectively. Performance gaps on medium Gaussian and impluse noise are presented in Table 3, on shot noise and speckle noise presented in Table 9 in the Appendix. Told from Table 3 and Table 9, our method is a clear winner: in most cases, our method has the smallest gaps.

Moreover, while our detection gaps are uniformly small (with rare outliers), the three baseline methods can frequently lead to intolerably large gaps.

Table 3: The performance gaps of BRISQUE [23], NIQE [24], NIMA [38], and DIP+AE on Gaussian and impulse noises. For NIMA, we report both technical quality assessment (the number before "/") and aesthetic assessment (the number after "/"). The best PSNR gaps are colored as red; the best SSIM gaps are colored as blue.

	Gaussian noise								Impulse noise							
	Gap in PSNR ↓				Gap in SSIM ↓				Gap in PSNR ↓				Gap in SSIM ↓			
	BRISQUE	NIQE	NIMA	DIP+AE	BRISQUE	NIQE	NIMA	DIP+AE	BRISQUE	NIQE	NIMA	DIP+AE	BRISQUE	NIQE	NIMA	DIP+AE
House	10.961	10.961	12.906/3.361	1.057	0.408	0.408	0.659/0.122	0.015	25.227	9.886	25.227/21.403	5.832	0.835	0.052	0.835/0.344	0.005
Peppers	3.715	5.053	4.893/5.204	0.603	0.271	0.341	0.334/0.345	0.006	12.872	2.296	2.117/9.701	0.305	0.404	0.013	0.012/0.198	0.012
Lena	4.681	7.194	10.180/1.581	0.845	0.260	0.397	0.537/0.054	0.008	2.374	6.204	15.646/12.674	0.231	0.011	0.039	0.376/0.209	0.002
Baboon	0.247	0.694	1.375/3.450	2.583	0.008	0.014	0.174/0.416	0.304	7.092	1.312	1.157/12.707	1.426	0.224	0.013	0.153/0.462	0.231
F16	6.236	8.128	1.336/3.991	0.850	0.429	0.530	0.036/0.088	0.023	14.180	5.505	8.621/9.372	0.495	0.369	0.036	0.111/0.135	0.006
Kodak1	2.558	3.158	6.723/19.546	0.520	0.069	0.103	0.275/0.626	0.029	5.913	0.232	11.791/10.366	0.538	0.142	0.007	0.410/0.584	0.048
Kodak2	6.959	6.855	9.094/0.143	0.353	0.361	0.365	0.516/0.004	0.007	18.575	0.312	13.870/9.657	0.429	0.741	0.014	0.304/0.127	0.026
Kodak3	2.029	7.686	19.834/19.716	0.574	0.152	0.475	0.731/0.624	0.005	16.867	8.769	13.217/2.031	1.959	0.351	0.077	0.180/0.005	0.046
Kodak12	7.066	8.093	6.436/3.075	0.788	0.419	0.456	0.384/0.178	0.007	16.781	1.785	9.139/2.685	1.068	0.817	0.003	0.087/0.008	0.048

MRI reconstruction We now test our detection method on MRI reconstruction, a classical medical IR problem involving a nontrivial linear f. Specifically, the model is $y = f(x) + \xi = \mathcal{F}(x) + \xi$, where $\mathcal{F}$ is the subsampled Fourier operator and ξ models the noise encountered in practical MRI imaging. Here, we take 8-fold undersampling and choose to parametrize x using a DD, inspired by the recent work [14]. Due to the heavy overparametrization of DD, [14] needs to choose an appropriate ES to produce quality reconstruction. We report the performance in Fig. 5 (results for all randomly selected samples can be found in the Appendix). Our method is able to signal stopping points that are reasonably close to the peak points, which also yield reasonably faithful reconstruction.

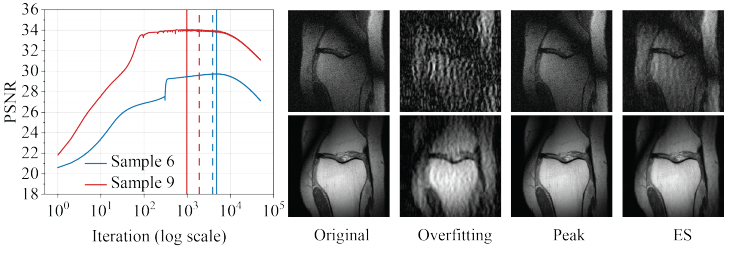

Figure 5: Results for MRI reconstruction. (left) The solid vertical lines indicate the peak performance iterate while the dash vertical lines are ES iterate detected by our method. (right) Visualizations for Sample 6 (1st row) and Sample 9 (2nd row).

Image regression Now we turn to SIREN [35], a recent functional SIDGP model that is designed to facilitate the learning of functions with significant high-frequency components. We consider a simple task from the original task, image regression, but add in some Gaussian noise. Mathematically, the $y = x + \varepsilon$, where $\varepsilon \sim_{iid} \mathcal{N}(0, 0.196)$, and we are to fit y by performing functional OPT listed in Eq. (2). Clearly, when the MLP used in SIREN is sufficiently overparamterized, the noise will also be learned. We test our detection method on this using the same 9-image dataset as in denoising. From Fig. 6, we can see again that our

method is capable of reliably detecting near-peak performance measured by either ES-PG or ES-SG, much better than without implementing any ES.

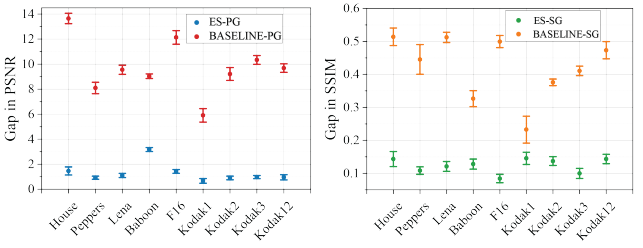

Figure 6: Results for image regression.

Ablation studies There are two crucial parameters—the window size n and the patience number p—dictating the performance of our detection method. Here, we evaluate the sensitivity of the performance to n and p on two settings: 1) fix $n = 256$, and vary p in $\{100, 300, 500, 700, 900\}$; 2) fix $p = 500$, and vary n in $\{32, 64, 128, 256, 512\}$. We again use the 9-image dataset, simulate medium-level Gaussian noise, and adopt DIP. For simplicity, we only report the gaps averaged over all images. Fig. 7 summarizes the results. Two observations: 1) our detection method is relatively insensitive to the two parameters; 2) large p and n seem to benefit the performance more. To be sure, we need p to be at least reasonably large so that our detection will not be trapped by local spikes. However, larger n requires better GPU resources to hold and compute with the data. In addition, we also explore the influence of the learing rates—for SIDGP and AE, respectively—on the performance. For this, we again consider medium Gaussian noise with DIP, and freeze $n = 256$ and $p = 500$. We fix one of the two learning rates while changing the other, and present the results in Table 4. We observe that despite different learning rates lead to slightly different levels of performance, the variation is not significant, in view that our typical ES-PG is less than 1, and typical ES-SG less than 0.1 in our previous evaluation.

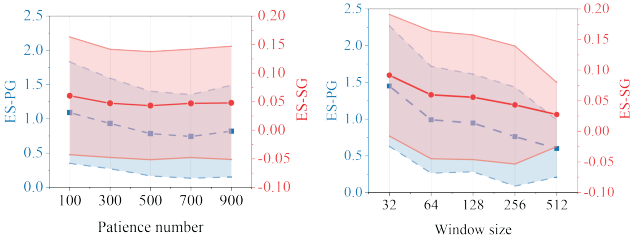

Figure 7: Ablation study for the sensitivity of the detection performance with respect to the patience number p and window size n.

Table 4: The impacts of different learning rates.

Learning Rate	(vary) DIP + (fixed .001) AE		(fixed .01) DIP + (vary) AE	
	ES-PG ↓	ES-SG ↓	ES-PG ↓	ES-SG ↓
0.01	0.756 (0.629)	0.044 (0.093)	0.814 (0.623)	0.043 (0.090)
0.001	0.593 (0.497)	0.043 (0.085)	0.713 (0.628)	0.045 (0.098)
0.0001	1.073 (0.984)	0.074 (0.118)	1.216 (1.036)	0.071 (0.098)

Acknowledgement

Zhong Zhuang, Hengkang Wang, and Ju Sun are partly supported by NSF CMMI 2038403. The authors acknowledge the Minnesota Supercomputing Institute (MSI) at the University of Minnesota for providing resources that contributed to the research results reported within this paper.

References

[1] A. Almansa, C. Ballester, V. Caselles, and G. Haro. A TV based restoration model with local constraints. *Journal of Scientific Computing*, 34(3):209–236, oct 2007. doi: 10.1007/s10915-007-9160-x.

[2] David Bau, Jun-Yan Zhu, Jonas Wulff, William Peebles, Hendrik Strobelt, Bolei Zhou, and Antonio Torralba. Seeing what a gan cannot generate. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4502–4511, 2019.

[3] Ashish Bora, Ajil Jalal, Eric Price, and Alexandros G Dimakis. Compressed sensing using generative models. In *International Conference on Machine Learning*, pages 537–546. PMLR, 2017.

[4] Emrah Bostan, Reinhard Heckel, Michael Chen, Michael Kellman, and Laura Waller. Deep phase decoder: self-calibrating phase microscopy with an untrained deep neural network. *Optica*, 7(6):559–562, 2020.

[5] Raymond H Chan, Chung-Wa Ho, and Mila Nikolova. Salt-and-pepper noise removal by median-type noise detectors and detail-preserving regularization. *IEEE Transactions on image processing*, 14(10):1479–1485, 2005.

[6] Zezhou Cheng, Matheus Gadelha, Subhransu Maji, and Daniel Sheldon. A bayesian perspective on the deep image prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5443–5451, 2019.

[7] MZ Darestani and R Heckel. Accelerated mri with un-trained neural networks. *arXiv preprint arXiv:2007.02471*, 2020.

[8] Charles L Epstein. *Introduction to the mathematics of medical imaging*. SIAM, 2007.

[9] Yosef Gandelsman, Assaf Shocher, and Michal Irani. " double-dip": Unsupervised image decomposition via coupled deep-image-priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11026–11035, 2019.

[10] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[11] Paul Hand, Oscar Leong, and Vladislav Voroninski. Phase retrieval under a generative prior. *arXiv preprint arXiv:1807.04261*, 2018.

[12] Reinhard Heckel and Paul Hand. Deep decoder: Concise image representations from untrained non-convolutional networks. In *International Conference on Learning Representations*, 2018.

[13] Reinhard Heckel and Mahdi Soltanolkotabi. Denoising and regularization via exploiting the structural bias of convolutional generators. *arXiv preprint arXiv:1910.14634*, 2019.

[14] Reinhard Heckel and Mahdi Soltanolkotabi. Compressive sensing with un-trained neural networks: Gradient descent finds a smooth approximation. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 4149–4158. PMLR, 13–18 Jul 2020. URL http://proceedings.mlr.press/v119/heckel20a.html.

[15] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2018.

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

[17] Li Jing, Jure Zbontar, et al. Implicit rank-minimizing autoencoder. *Advances in Neural Information Processing Systems*, 33, 2020.

[18] Yeonsik Jo, Se Young Chun, and Jonghyun Choi. Rethinking deep image prior for denoising. *arXiv:2108.12841*, August 2021.

[19] Alessandro Lanza, Serena Morigi, Federica Sciacchitano, and Fiorella Sgallari. Whiteness constraints in a unified variational framework for image restoration. *Journal of Mathematical Imaging and Vision*, 60(9):1503–1526, sep 2018. doi: 10.1007/s10851-018-0845-6.

[20] Hannah Lawrence, David Bramherzig, Henry Li, Michael Eickenberg, and Marylou Gabrié. Phase retrieval with holography and untrained priors: Tackling the challenges of low-photon nanoscale imaging. *arXiv preprint arXiv:2012.07386*, 2020.

[21] Oscar Leong and Wesam Sakla. Low shot learning with untrained neural networks for imaging inverse problems. *arXiv preprint arXiv:1910.10797*, 2019.

[22] Michael Michelashvili and Lior Wolf. Audio denoising with deep network priors. *arXiv preprint arXiv:1904.07612*, 2019.

[23] Anish Mittal, Anush Krishna Moorthy, and Alan Conrad Bovik. No-reference image quality assessment in the spatial domain. *IEEE Transactions on image processing*, 21 (12):4695–4708, 2012.

[24] Anish Mittal, Rajiv Soundararajan, and Alan C Bovik. Making a "completely blind" image quality analyzer. *IEEE Signal processing letters*, 20(3):209–212, 2012.

[25] Gregory Ongie, Ajil Jalal, Christopher A Metzler, Richard G Baraniuk, Alexandros G Dimakis, and Rebecca Willett. Deep learning techniques for inverse problems in imaging. *IEEE Journal on Selected Areas in Information Theory*, 1(1):39–56, 2020.

[26] Xingang Pan, Xiaohang Zhan, Bo Dai, Dahua Lin, Chen Change Loy, and Ping Luo. Exploiting deep generative prior for versatile image restoration and manipulation. In *European Conference on Computer Vision*, pages 262–277. Springer, 2020.

[27] Neal Parikh and Stephen Boyd. Proximal algorithms. *Foundations and Trends in optimization*, 1(3):127–239, 2014.

[28] Lutz Prechelt. Early stopping-but when? In *Neural Networks: Tricks of the trade*, pages 55–69. Springer, 1998.

[29] Adnan Qayyum, Inaam Ilahi, Fahad Shamshad, Farid Boussaid, Mohammed Bennamoun, and Junaid Qadir. Untrained neural network priors for inverse imaging problems: A survey. 2021.

[30] Sriram Ravula and Alexandros G Dimakis. One-dimensional deep image prior for time series inverse problems. *arXiv preprint arXiv:1904.08594*, 2019.

[31] Dongwei Ren, Kai Zhang, Qilong Wang, Qinghua Hu, and Wangmeng Zuo. Neural blind deconvolution using deep priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3341–3350, 2020.

[32] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

[33] Yoav Shechtman, Yonina C Eldar, Oren Cohen, Henry Nicholas Chapman, Jianwei Miao, and Mordechai Segev. Phase retrieval with application to optical imaging: a contemporary overview. *IEEE signal processing magazine*, 32(3):87–109, 2015.

[34] Zenglin Shi, Pascal Mettes, Subhransu Maji, and Cees G. M. Snoek. On measuring and controlling the spectral bias of the deep image prior. *arXiv:2107.01125*, July 2021.

[35] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33, 2020.

[36] Zhaodong Sun. Solving inverse problems with hybrid deep image priors: the challenge of preventing overfitting. *arXiv preprint arXiv:2011.01748*, 2020.

[37] Zhaodong Sun, Fabian Latorre, Thomas Sanchez, and Volkan Cevher. A plug-and-play deep image prior. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8103–8107. IEEE, 2021.

[38] Hossein Talebi and Peyman Milanfar. Nima: Neural image assessment. *IEEE Transactions on Image Processing*, 27(8):3998–4011, 2018.

[39] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *NeurIPS*, 2020.

[40] Kshitij Tayal, Chieh-Hsin Lai, Vipin Kumar, and Ju Sun. Inverse problems, deep learning, and symmetry breaking. *arXiv:2003.09077*, March 2020.

[41] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9446–9454, 2018.

[42] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[43] E Weinan. Machine learning and computational mathematics. *arXiv preprint arXiv:2009.14596*, 2020.

[44] Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688. Citeseer, 2011.

[45] Francis Williams, Teseo Schneider, Claudio Silva, Denis Zorin, Joan Bruna, and Daniele Panozzo. Deep geometric prior for surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10130–10139, 2019.

[46] Chong You, Zhihui Zhu, Qing Qu, and Yi Ma. Robust recovery via implicit bias of discrepant learning rates for double over-parameterization. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 17733–17744. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/cd42c963390a9cd025d007dacfa99351-Paper.pdf.

[47] Tong Zhang, Bin Yu, et al. Boosting with early stopping: Convergence and consistency. *The Annals of Statistics*, 33(4):1538–1579, 2005.

[48] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative visual manipulation on the natural image manifold. In *European conference on computer vision*, pages 597–613. Springer, 2016.