The Devil is in the Upsampling: Architectural Decisions Made Simpler for Denoising with Deep Image Prior

Yilin Liu[1*], Jiang Li[1*], Yunkui Pang [1*], Dong Nie[1], and Pew-Thian Yap [1]
[1]University of North Carolina at Chapel Hill
{yilinliu, yunkuipa, dongnie}@cs.unc.edu, jiang.li@unc.edu, ptyap@med.unc.edu

Abstract

Deep Image Prior (DIP) shows that some network architectures inherently tend towards generating smooth images while resisting noise, a phenomenon known as spectral bias. Image denoising is a natural application of this property. Although denoising with DIP mitigates the need for large training sets, two often intertwined practical challenges need to be overcome: architectural design and noise fitting. Existing methods either handcraft or search for suitable architectures from a vast design space, due to the limited understanding of how architectural choices affect the denoising outcome. In this study, we demonstrate from a frequency perspective that unlearnt upsampling is the main driving force behind the denoising phenomenon with DIP. This finding leads to straightforward strategies for identifying a suitable architecture for every image without laborious search. Extensive experiments show that the estimated architectures achieve superior denoising results than existing methods with up to 95% fewer parameters. Thanks to this under-parameterization, the resulting architectures are less prone to noise-fitting[1].

1. Introduction

Image denoising is useful on its own and can be a plug-in module for many other image restoration tasks [37, 38, 4]. Deep neural networks have become the tool of choice for image denoising owing to their ability to learn natural image priors from large-scale datasets. Yet, Deep Image Prior (DIP) [33] requires only a single degraded image for image resotration. Remarkably, DIP shows that a randomly initialized convolutional neural network (CNN) can regularize image restoration through its architecture and early-stopping optimization. This is inspired by the phenomenon that *some* network architectures act inherently as image priors, favoring generating smooth, natural images and resisting noises or degradations. However, image denoising with DIP is greatly influenced by architectural design [3, 1, 33, 15, 17], and the associated tendency to fit the original noisy image, i.e., over-fitting [15, 16].

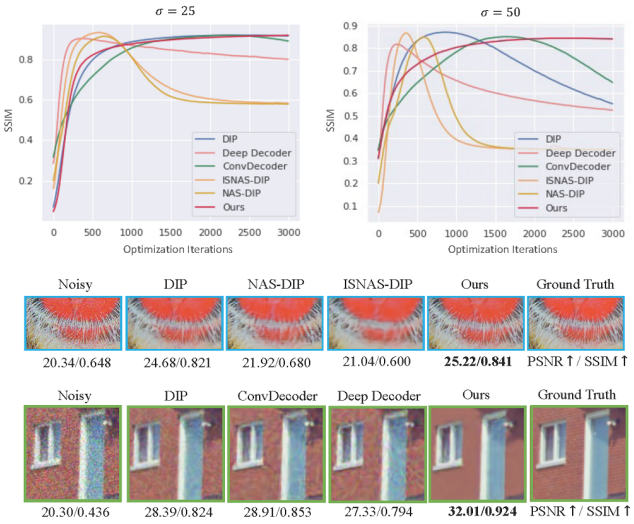


Figure 1: Top: Performance (SSIM↑) under different levels of Gaussian noises. **Bottom**: Denoising of a fine-grained (1^{st}row) and a coarse-grained image (2^{nd}row). Most existing methods, including the recent image-specific ISNAS-DIP[1], struggle to perform well simultaneously in both cases. Our simple strategies are flexible in image-specific architectural adaptation without requiring a search. Moreover, the results of the lightweight ConvDecoder[10] and Deep Decoder[15] suggest that without proper model setups, under-parameterization itself can neither ensure good denoising performance nor remove the need for early-stopping.

Architectural design for DIP remains an open problem. One prevailing view is that model under-parameterization limits noise over-fitting and thus mitigates the need for early stopping [15]. However, our experiments reveal that multiple model architectures can exist under a similar parameter budget, where inappropriate model setups can still lead to

* These authors contribute equally.
[1]https://github.com/YilinLiu97/FasterDIP-devil-in-upsampling.git

noise fitting and over-smoothing (Fig.1, Fig.3). Another line of work automates the architecture identification using Neural Architecture Search (NAS)[6, 17, 1]. Without prior knowledge on suitable architectures, extensive search incurs substantial computational costs, prohibiting image-wise NAS for optimal restoration [1]. Arican et al. [1] narrow the search space using training-free metrics, but the need for candidate comparison dramatically prolongs the restoration time ($\sim$ 7 hours/image). Moreover, NAS-based models, along with many DIP models, are typically heavily parameterized and prone to over-fitting, as corroborated in our experiments. Thus, their performance largely depends on the timing of early stopping, which is typically image-specific and hard to pinpoint without access to ground truth.

Directly identifying an effective under-parameterized architecture for each image is challenging in light of the vast number of different architectures and the absence of ground truth for explicit supervision. To simplify the architectural decisions for DIP, we rethink the architectural influences on its performance in the context of image denoising.

We start by noting that denoising performance is attributable to *only a few componenets*, primarily the unlearnt upsampling operations. Our frequency analysis reveals that the fixed upsampling operations tend to bias the architecture towards low-frequency contents more strongly than linear or convolutional layers, critically influencing both the peak PSNR and the point of early stopping to avoid noise-fitting.

Importantly, this finding leads to empirical discovery on the roles of typical architectural components in DIP: assuming a standard hourglass network, **i)** simply scaling the depth and width can balance smoothing and preservation of details, due to the low-pass filtering effects of the upsampling operations inserted in-between the layers. As we observed, a wider and shallower network is better at preserving details and therefore suitable for fine-grained images. This suggests that the "optimal" architecture can vary across images, and image texture should be considered for more effective denoising. **ii)** Skip connections make a deep network perform similarly as a shallower one likely by reducing the "effective upsampling rate". This implies the possibility of discarding skip connections to simplify DIP architectural design.

Based on these insights, we find it sufficient to restrict the design choices to only the network depth and width, reducing the problem to searching through only a handful of sub-networks and adapting them according to the level of image details. This can be done as a pre-processing step without costly searching or evaluation. We show that this simple strategy works with both the hourglass and decoder structures, and with proper setups, the estimated networks can denoise while preserving the details better than the larger networks with only 5%$\sim$40% number of parameters. The resulting under-parameterization alleviates the need for early stopping. Our contributions are as follows:

- We pinpoint that unlearnt upsampling is the main driving force behind the spectral bias of DIP.
- Leveraging this insight, we empirically identify the influences of depth, width and skip connections, along with their correlations with image texture, allowing for **quick**, **effective** and **more interpretable** architectural design for *every image* without the laborious search.
- We are the first to associate DIP architectural design with image texture. To promote future research, we build a *Texture-DIP Dataset* consisting of images from three popular datasets, reclassified into several predefined width choices – validated through experiments, according to textural complexity.
- We show that with proper setups, a highly under-parameterized subnetwork could match and even outperform larger counterparts, especially at a higher noise level. We conducted extensive synthetic and real-world noise removal experiments to validate our findings and approach.

2. Related Work

DIP Variants. Deep Decoder [15] is an under-parameterized network proposed to avoid early stopping. However, our investigations show that under-parameterization alone is *not* sufficient for denoising. For instance, ConvDecoder [10], a convolutional variant of Deep Decoder, contains more parameters but demonstrates less tendency to over-fit (Fig 1). In contrast to NAS-based methods[6, 17, 1], our strategy leverages the observed relationship between the architecture and the image to prevent the exhaustive search. Other variants such as DIP-RED [24] and DIP-TV [20] augment DIP with additional priors.

Early-stopping criterion. Cheng et al. [7] perform posterior inference in DIP so as to prevent the need for early stopping by adding Gaussian noise to the gradients. Similarly, Shi et al. [29] regularize the matrix norm of the network weights to alleviate performance decay. However, tuning the regularization granularity can be challenging (Suppl. B). Jo et al. [18] combine DIP with Stein's unbiased risk estimator (SURE) [32] for training without clean images, but SURE is limited to only a few known noise types [31, 23]. Wang et al. [35] propose to track the running variance of the outputs, but this also introduces new hyper-parameters that require non-trivial tuning.

3. Investigation and Method

3.1. Preliminaries

Deep Image Prior. A noisy image $\mathbf{y} \in \mathbb{R}^N$ can be modeled as: $\mathbf{y} = \mathbf{x}+\mathbf{n}$, where $\mathbf{x} \in \mathbb{R}^N$ is the clean counterpart to

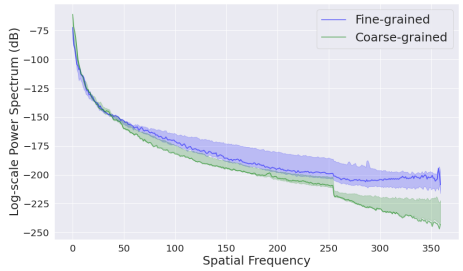


Figure 2: Spectral density of images of different levels of textural complexity, obtained by azimuthally integrateing over the power spectrum of the image.

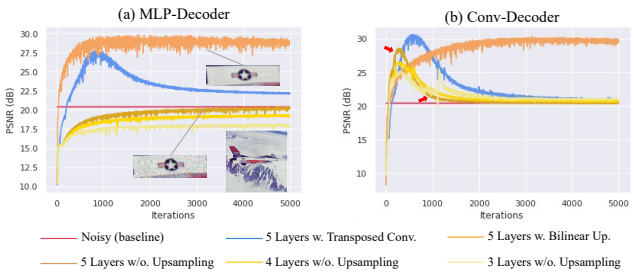


Figure 3: Influences of architecture components on image denoising. (**a**) Without convolutional layers, upsampling still enables denoising. Transposed convolutions lead to faster performance drop due to earlier noise-fitting than bilinear upsampling. Removing the upsampling decreases denoising capability, which cannot be compensated by simply reducing the number of layers, i.e., under-parameterization. (**b**) Convolutional layers *alone* exhibit certain denoising effects but necessitate early stopping. Increasing the number of convolutional layers achieves higher peak PSNR at the expense of earlier noise fitting (see red arrows). Better results achieved when combined with upsampling.

be recovered and $\mathbf{n}$ is assumed to be $i.i.d.$ Gaussian Noise drawn from $\mathcal{N}(0, \sigma^2\mathbf{I})$ with $\mathbf{I}$ being the identity matrix. DIP parameterizes the clean image $\mathbf{x}$ via a network $\mathbf{G}_\theta$ and is optimized to fit the noisy image $\mathbf{y}$, formulated as:

$$\theta^* = \arg\min_\theta \mathcal{L}(\mathbf{y}; \mathbf{G}_\theta(\mathbf{z})), \quad \mathbf{x}^* = \mathbf{G}_{\theta^*}(\mathbf{z}), \qquad (1)$$

where $\mathbf{z}$ denotes the fixed white noise input tensor. Such parameterization allows lower-frequency contents to be fitted before the higher-frequency ones [3, 29], exhibiting high impedance to image noises or degradations. Early-stopping is often required to prevent excessive high frequencies from being fitted, such as noise.

Image complexity. Ideally, DIP should denoise an image while preserving textural details. We define image complexity by its texture, which can be characterized by its power spectral density (PSD). The spectral power of a natural image typically follows an exponential decay from low frequencies to high frequencies [30]. High-frequency components correspond to fine features such as details, while low-frequency components correspond to coarse structures. Hence, fine-grained images contain more high frequencies than coarse-grained images, giving a flatter PSD curve in (Fig.2). Each image is scored based on its texture features, as described in Sec.3.5.

3.2. The importance of upsampling

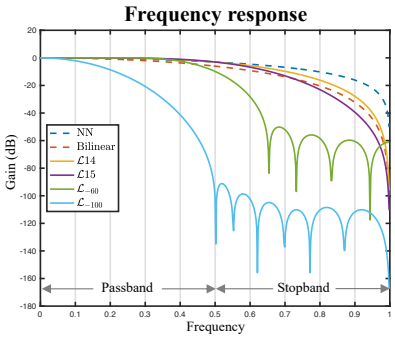
We first identified the core architecture components that affect the denoising performance of DIP. To this end, we analyzed a decoder-only architecture by removing the encoder from the typical encoder-decoder architecture [33], since a decoder is the minimum requirement for reconstructing the final image. Our base model for analysis is a 6-layered convolutional decoder (Conv-Decoder[10]), with 128 3×3 filters per layer except for the last regression layer, followed by batch normalization, ReLU activation function and upsampling. We further simplify the setup by replacing the spatial filters with pixel-wise 1×1 filters, constructing a non-convolutional variant dubbed MLP-Decoder.

Results for MLP-Decoder, shown in Fig.3 (a), suggest that upsampling plays a vital role. Removal of upsampling results in significant performance degradation of denoising, which cannot be compensated by simply reducing the size of the network for under-parameterization. This holds true for other tasks such as super-resolution (Suppl.A). Fig.3 (b) shows that learnable spatial filters alone also enable denoising, in contrast to the pixel-wise filters, re-affirming the frequency bias of the convolutional layers [3]. However, the effects vanish as the network size increases and noise-fitting occurs sonner. Convolutional layers together with upsampling achieve better results as manifested in the higher peak PSNR and longer denoising effects.

Discussion. These results suggest that an appropriate upsampling operation is crucial for effective network image priors, and that under-parameterization alone is insufficient. Different upsampling operations induce varying extents of denoising effects: transposed convolutions [25] tend to fit noise faster than bilinear upsampling, necessitating early stopping. In the next section, we investigate the behaviors of these upsampling operations from a signal-processing perspective to gain insights into their influences on denoising performance.

3.3. Spectral effects of upsampling

The design choice for upsampling is typically standard: bilinear/nearest neighbor (NN) interpolation or transposed convolutions. These upsampling operations can all be decomposed into two steps: (i) zero-insertion and (ii) filtering. Given a target upsampling factor R, the low-resolution feature map is first interleaved with (R-1) rows/columns of zeros to increase its sampling rate, and then convolved with



LPFs	Coarse	Fine
NN	31.1	24.5
Bilinear	31.3	22.9
$\mathcal{L}14$	31.4	24.7
$\mathcal{L}15$	31.2	24.7
$\mathcal{L}_{-60}$	31.2	22.5
$\mathcal{L}_{-100}$	27.5	20.0
NN	27.8↓	24.2
Bilinear	31.1	22.9
$\mathcal{L}14$	25.2↓	24.0
$\mathcal{L}15$	27.1↓	24.5
$\mathcal{L}_{-60}$	31.2	22.5
$\mathcal{L}_{-100}$	27.5	20.0

Figure 4: (**Left**) Frequency responses of the tested LPFs. Different LPFs result in upsamplers with different extents of smoothing. NN interpolation preserves most signals in the passband but also the high-frequency replica in the stopband; $\mathcal{L}_{-100}$ attenuates the signals most significantly ($\sim 100dB$) and suppresses the high-frequency replica most. (**Right**) Denoising results on coarse- and fine-textured images from Set9. **Top rows**: Peak PSNR values. **Bottom rows**: PSNR values at the last training iteration.

a low-pass filter (LPF) to remove the alias high frequencies introduced by zero-insertion. The key difference between the upsampling operations lies in the nature of the filters. The filters for transposed convolutions are learnable, but are fixed for bilinear and nearest nighbor upsampling.

To better illustrate this, we consider the case of a 1D signal $x(n), n = 0, ..., N-1$, and its discrete Fourier representation $X(k) = \sum_{n=0}^{N-1} x(n) e^{\frac{-i2\pi}{N} kn}, k = 0, ..., N-1$.

For an upsampling factor of 2, we have

$$X^{\text{up}}(\hat{k}) = \sum_{n=0}^{2N-1} x^{\text{up}}(n) e^{\frac{-i2\pi}{2N}\hat{k}n} \tag{2}$$

$$= \sum_{n=0}^{N-1} x^{\text{up}}(2n) e^{\frac{-i2\pi}{2N}(2n)\hat{k}} + \sum_{n=0}^{N-1} x^{\text{up}}(2n+1) e^{\frac{-i2\pi}{2N}(2n+1)\hat{k}}, \tag{3}$$

where $\hat{k} = 0, \ldots, 2N-1$, $x^{\text{up}}(2n) = x(n)$, $x^{\text{up}}(2n+1) = 0$ due to interleaved zero insertion. Hence, for $0 \leq \hat{k} < N$, $X^{\text{up}}(\hat{k}) = X(k)$.

For $\hat{k} \geq N$, let $k' = \hat{k} - N$, $k' = 0, ..., N-1$, we have

$$X^{\text{up}}(\hat{k}) = \sum_{n=0}^{N-1} x(n) e^{\frac{-i2\pi}{2N}(k'+N)2n} \tag{4}$$

$$= \sum_{n=0}^{N-1} x(n) e^{\frac{-i2\pi}{N}nk' - i2n\pi} = X(k'), \tag{5}$$

where Eq.5 exploits the periodicity of the complex exponential function. Thus, zero-insertion will preserve the original spectrum at $[0, N]$ (passband) and additionally create a mirrored copy at $[N, 2N-1]$ (stopband). The high-frequency replica beyond N should be suppressed by the subsequent LPF to avoid image artifacts. According to duality and convolution theorem, convolving with NN or bilinear interpolation filter is equivalent to multiplication of $X^{up}(\hat{k})$ with a `Sinc` or `Sinc`^{2} function corresponding to low-pass filtering, while transposed convolutional filter may not necessarily be low-pass as it depends on the optimization objective.

Table 1: The iteration number [iter./5000] at the peak PSNR for different upsamplers. The upsamplers are designed to mainly differ in the stopband. The peak PSNR is reached more slowly when the attenuation is stronger (from left to right).

	NN	$\mathcal{L}14$	$\mathcal{L}15$	$\mathcal{L}16$	$\mathcal{L}17$
Coarse-grained	1783	1589	1681	2205	2214
Fine-grained	2424	2942	3898	4361	4957

We experimentally demonstrate that different upsampling operations bias the architecture towards different spectral properties. Specifically, we constructed four upsamplers by first interleaving the input with zeros and then convolving it with handcrafted LPFs (shortened as $\mathcal{L}$): $\mathcal{L}14$, $\mathcal{L}15$, $\mathcal{L}_{-60}$ and $\mathcal{L}_{-100}$, with the subscript denoting the decayed dB. By construction, $\mathcal{L}14$ and $\mathcal{L}15$ are very close to NN in the passband ($< 0.03dB$) and only differ in the stopband. The frequency responses of the compared LPFs are detailed in Fig. 4. We applied the customized upsamplers on ConvDecoder and tested them on the fine- and coarse-textured images from Set9 respectively. Similar findings hold for the encoder-decoder architecture (Suppl. C).

From Fig. 4, upsampling critically influences both the peak PSNR value and the timing for early stopping with respect to images of different textural complexities. Upsamplers (NN, $\mathcal{L}14$, $\mathcal{L}15$) with less attenuation on the high-frequency replica are beneficial for generating fine-grained images, but they tend to cause noise-fitting especially for coarser-grained images. Also, they are generally the fastest to reach the peak PSNR (Table 1). This explains why the transposed convolution requires early-stopping, as the filters may not learn to attenuate the introduced high frequencies effectively. Similar issues with the learned upsampling are also prevalent in generative models [28, 5, 11, 12].

On the other hand, bilinear and $\mathcal{L}_{-60}$ exhibit a greater amount of attenuation, more strongly biasing the network against high frequencies and leading to longer-lasting denoising effects. They turn out to work sufficiently well for both kinds of images, especially on the coarser-grained ones which are typically the majority in the dataset. LPF$_{-100}$ tends to over-smooth the output and performs the worst as it attenuates both the passband and stopband signals substantially, though not requiring early stopping.

Discussion. These results lead us to conclude that the upsamplers with fixed LPFs are key to the denoising effects of DIP for their tendency towards smooth images (i.e., re-

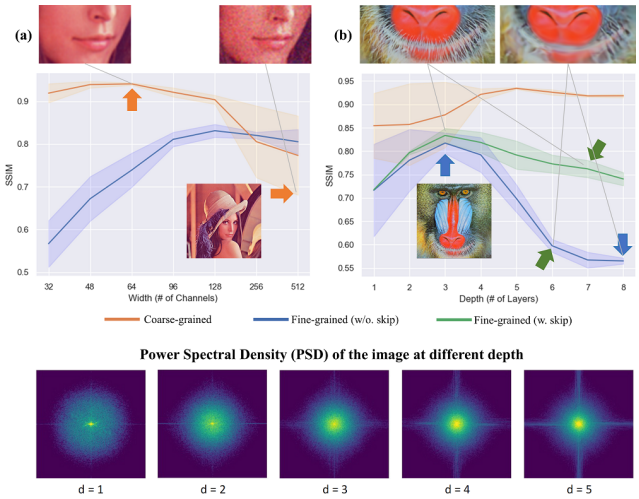


Figure 5: Influences of width, depth, and skip connections, assuming upsampling inserted in-between the layers. **(a)** Tendency to over-fit coarse-grained images increases with width. SSIM scores averaged across the depths. **(b)** Tendency to over-smooth fine-grained images increases with depth, but alleviated by skip connections. SSIM scores are averaged across the widths.

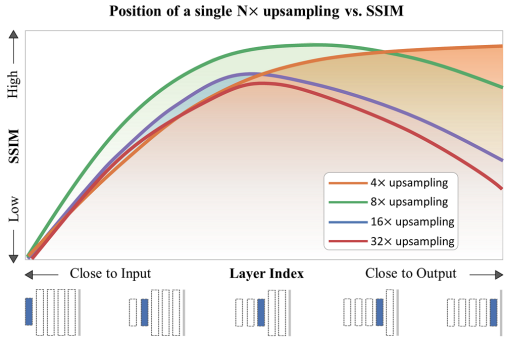


Figure 6: Position of upsampling. Placing the upsampling close to the input (or encoder) can easily cause over-fitting, regardless of the scaling factor. When close to the output (end of the decoder), upsampling with large scaling factors (e.g., 32×) causes over-smoothing.

duced high-frequency contents), which aligns well with the spectral statistics of natural images (Fig.2). Probably due to a good balance between the denoising performance and persistency, bilinear upsampling has been widely adopted in DIP models for various applications [33, 15, 13, 16].

3.4. Interactions with other architecture elements

After establishing the significance of upsampling, we will now consider how it may interact with other common architectural elements in affecting the ultimate output.

Convolutions + non-linearity. Ideal upsampling does not modify the signal representations but only expands the spectrum for the subsequent layers to add new frequency contents. Convolution followed by nonlinearity such as ReLU, is the only operation capable of introducing arbitrarily high frequencies [19]. Increasing the number of layers (depth) or channels (width) enhances the capability of generating new high frequencies, as theoretically and empirically proved in [26].

Intuitively, using an excessive number of layers or channels can accelerate the learning of both details and noise. However, the effects can be attenuated by the fixed upsampling operations between the layers. As shown in Fig.5, when using fewer upsampling operations (i.e., a shallower network), increasing the width of the network increases the tendency to cause overfitting on simpler images while benefiting the more complex images. Increasing the number of upsampling operations (i.e., a deeper network) can alleviate over-fitting with stronger attenuation but results in blurry outputs for fine-grained images. Increasing only the upsampling factors without adding more layers can make the output even more blurry (Fig.6). In other words, the final output is determined by the balance between the generation of high frequencies by the layers and signal attenuation caused by the upsampling operations.

Skip connections between the encoder and the decoder often complicate the design space [6]. While they are not directly responsible for denoising, they may lower the effective upsampling rate, making deep networks perform similarly to shallower ones, i.e., resulting in lesser suppression of the high-frequency replica. This finding is validated on a large-scale experiment comprising 7424 architectures (Suppl. D). As examplified in Fig. 5 (b), skip connections notably ameliorate the over-smoothing issue induced by the same deep network. Overall, skip connections exert a more pronounced influence on the deeper networks compared to the shallower ones, as evident in the smaller deviation observed when depth $\leq$3 in Fig.5 (b).

3.5. Practical application to architectural design

Based on the above findings and analysis, we argue that it is possible to estimate an effective architecture for each image without extensive search. Assuming every decoder layer is followed by a 2× bilinear upsampling layer, our strategies are as follows:

Depth estimation. We have shown that increased depth tends to over-smooth the output, affecting fine-grained images much more than on coarse-grained images. For fine-grained images, we have three options: **a**) add more skip connections to a deep network; **b**) simply use a shallower one; **c**) keep all the layers but reduce down-/up-sampling layers. We recommend **c** for decoder-only architectures since they are already under-parameterized; for hourglass network this can easily lead to over-fitting (Fig.6). We find

Table 2: Quantitative results on Gaussian noise. σ denotes the noise level. All methods were trained with **a fixed iteration number (3000)** throughout the experiments. The highest score is in **bold**, and the second highest is underlined.

Datasets		DIP [33]	Deep Decoder [15]	ConvDecoder [10]	NAS-DIP [6]	ISNAS-DIP [1]	Ours
		$\sigma = 25$					
Set9 [9]	PSNR	30.10	28.45	28.51	26.37	29.11	**30.26**
	SSIM	0.893	0.848	0.854	0.753	0.862	**0.900**
Set12 [37]	PSNR	26.97	25.98	25.78	20.86	24.10	**28.14**
	SSIM	0.812	0.789	0.786	0.534	0.745	**0.884**
CBSD68 [27]	PSNR	**28.93**	25.50	25.19	23.80	24.51	28.57
	SSIM	**0.892**	0.809	0.793	0.693	0.745	0.888
		$\sigma = 50$					
Set9 [9]	PSNR	25.04	25.22	25.01	21.07	23.91	**26.13**
	SSIM	0.761	0.764	0.769	0.593	0.698	**0.833**
Set12 [37]	PSNR	22.15	20.44	22.72	18.92	19.20	**24.59**
	SSIM	0.623	0.687	0.706	0.476	0.537	**0.805**
CBSD68 [27]	PSNR	23.74	23.52	24.06	17.92	19.93	**24.17**
	SSIM	0.746	0.725	0.767	0.323	0.573	**0.774**
# Params (Millions)		2.3M	**0.1M**	0.89M	4.4M	*Varied*	**0.05M∼0.92M**

Table 3: Comparisons of desired properties. Restoration time is computed on an image of size 512×512 with 3000 iterations.

	NAS [6]	ISNAS [1]	Ours
Image-Specific	✗	✓	✓
Architecture Search	3 days	5 mins	–
Per-image Restoration	∼23 mins	∼7 hrs	∼6 mins
Early Stopping Required?	Yes	Yes	No

b is generally better than **a** in trading off between good performance and the need for early stopping, especially under higher-level noise, as shown in our results section. This also holds for coarse-grained images as they are not sensitive to depth. More specifically, we find a 2-level hourglass network sufficient for both types of images (Suppl. D).

Width estimation. Width is crucial for learning sufficient details while avoiding over-fitting especially to a shallow network (less signal attenuation). We find that the width needs to be set according to textural complexity of the image: a finer-grained image requires more channels per layer and vice versa. To further validate this, we treated width estimation as a classification problem, and trained three SVMs [8] with texture features as the inputs to classify the images into three width choices $\{32, 64, 128\}$. Note that these widths were empirically chosen for the datasets used in this work and are by no means optimal for all cases, but tuning for other images should be straightforward. The classification results and analysis are in Sec.4.4.

Image scoring. To more robustly classify image texture, we extracted both spatial and frequency features and trained a Decision Tree [22] for feature selection. We used the classic Gray Level Co-occurrence Matrix (GLCM) [14] for deriving the spatial texture features from each image. The most useful features turn out to be the following: correlations measured at $0°$, homogeneity at $45°$, and contrast at $0°$. Frequency features are captured using a 1D PSD vector obtained by first converting the 2D PSD from Cartesian coordinates $S(k, l)$ to polar coordinates, and then azimuthally averaging over θ, defined as $\hat{S}(r) = \frac{1}{2\pi}\int_0^{2\pi} S(r, \theta)d\theta$ with $r = \sqrt{k^2 + l^2}$ and $\theta = \text{atan2}(k, l)$, which represents the mean magnitude of the frequencies with respect to the radial distance r. Refer to Suppl. E for more details.

Table 4: Quantitative evaluation on Poisson noise. Deep Decoder shortened as DD and ConvDecoder as CD.

Noise scale		DIP	DD	CD	NAS	ISNAS	Ours
$\zeta = 0.01$	PSNR	**31.58**	29.81	29.51	29.70	30.12	30.95
	SSIM	**0.915**	0.880	0.875	0.864	0.875	0.910
$\zeta = 0.1$	PSNR	22.66	23.91	24.94	15.72	17.68	**24.99**
	SSIM	0.640	0.718	0.776	0.417	0.496	**0.789**
$\zeta = 0.2$	PSNR	20.36	21.13	21.87	12.99	14.64	**22.55**
	SSIM	0.563	0.609	0.672	0.311	0.383	**0.774**

4. Experiments

4.1. Implementation Details

We conducted the experiments on three popular datasets and a real-world noisy dataset: **Set9** [9] consisting of 9 colored images, **Set12** [37] consisting of 12 grey-scaled images, **CBSD68** [27] consisting of 68 colored images, and **PolyU** [36] consisting of 100 real noisy and clean image pairs. We first report our results with **2-level** hourglass architecture with the same components as in DIP [33] when comparing with the existing methods, and then extend our strategy to ConvDecoder [10], a decoder-only architecture. All models were trained for 3000 iterations.

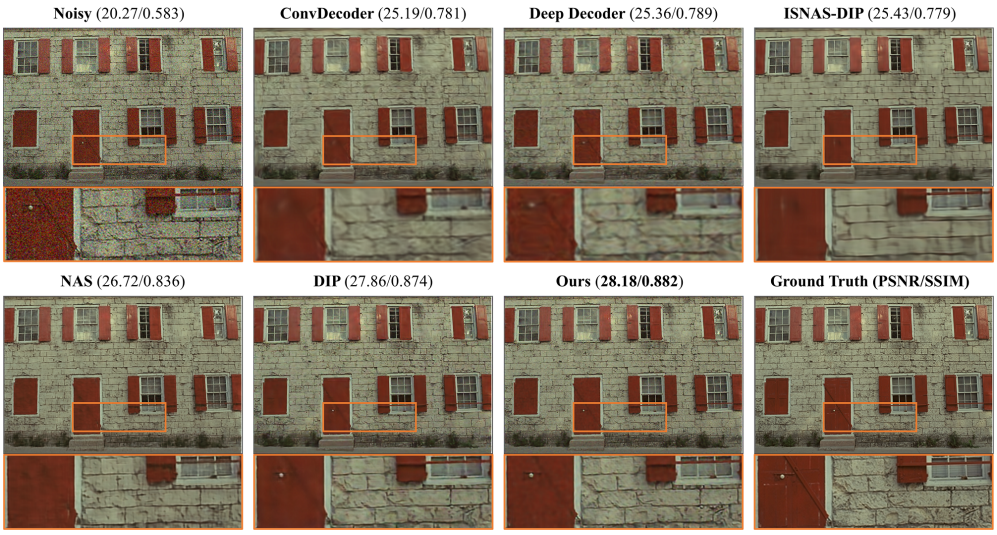


Figure 7: Denoising results on a **fine-grained** image ("kodim01" from Set9) with **Gaussian noise** ($\sigma = 25$). Our estimated architecture for this image is a two-level hourglass network with one skip connection and 128 channels, which is much smaller than others.

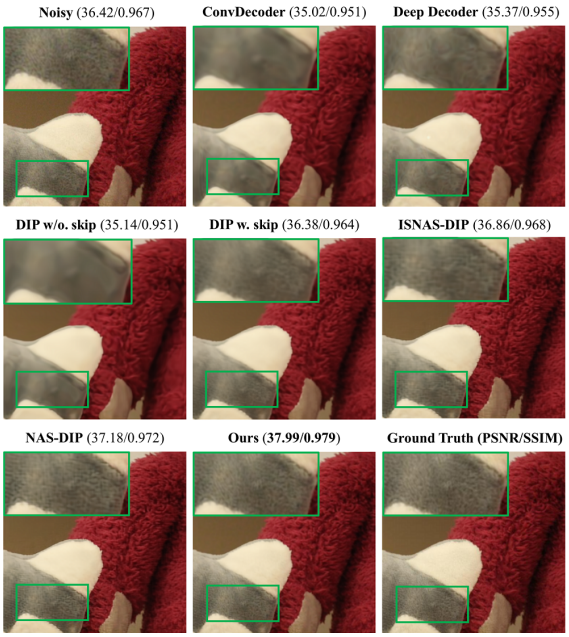


Figure 8: Denoising results on a **real-world noisy** image.

4.2. DIP variants

Gaussian Noise. Compared with the base DIP network, the properly-designed under-parameterized networks estimated with our strategy perform on par at a mild noise level while excel at a higher noise level (Table 2). This cannot be solely explained by under-parameterization since Deep Decoder and ConvDecoder contain similar or even fewer parameters while are unable to achieve similar results. Fig. 7 shows that a shallow and wide network can preserve the details better than many deeper ones. Note that DIP is a 5-level hourglass network with full skip connections, which by our standard can also well preserve the details. In fact, it is a very strong baseline under mild noise. NAS-DIP and ISNAS-DIP suffer from various degrees of over-fitting on different datasets. This also suggests that the point of optimal stopping varies across images. Besides, they are time-intensive in either searching or evaluation (Table 3). Similar conclusions hold for **Poisson noise**, which was tested on Set9 [9] as shown in Table 4.

Table 5: Quantitative evaluation on PolyU, a real-world noisy dataset. Deep Decoder and ConvDecoder shortened as DD, CD.

	DIP	DD	CD	NAS	ISNAS	Ours
PSNR	**38.15**	37.22	37.00	37.83	37.78	38.05
SSIM	0.982	0.978	0.976	0.982	0.977	**0.984**

Real-World Noise. We additionally evaluated all methods on the PolyU dataset [36]. We applied again the two-level hourglass architectures with variable width estimated for the images. Table 5 summarizes the numerical results. Fig.9 shows that our method tends to preserve more details, though this may not be reflected by the metrics. Simply removing the skip connections from DIP causing blurring, similar to the decoder-only architectures.

4.3. Decoder-only Architectures

We applied our strategy to ConvDecoder and tested it on all three datasets. Here we keep all 5 layers but remove

Table 6: Application to ConvDecoder.

Datasets		$\sigma = 25$		$\sigma = 50$	
		Before	After	Before	After
Set9	PSNR	28.51	28.74	25.01	25.11
	SSIM	0.854	**0.873**	0.769	**0.784**
Set12	PSNR	25.79	**26.98**	22.72	23.01
	SSIM	0.786	**0.854**	0.706	**0.742**
CBSD68	PSNR	25.19	**28.29**	24.36	24.12
	SSIM	0.793	**0.877**	0.767	0.768
# Params (Millions)		0.89M	0.06M∼0.89M	0.89M	0.06M∼0.89M

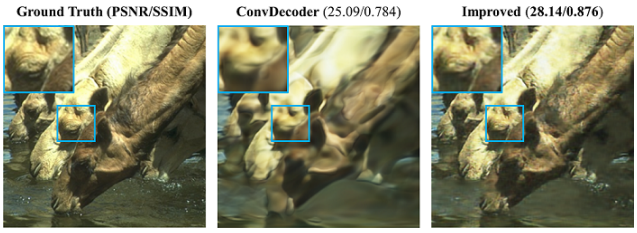


Figure 9: Qualitative improvement on ConvDecoder simply by removing three upsampling layers in this case while preserving all the convolutional layers.

a certain number of upsampling layers and scale the width accordingly for the images. These simple changes effectively alleviate the over-smoothing issue that often brought by ConvDecoder as shown in Fig.8 and improve the quantitative results as shown in Table 6.

4.4. More Analysis on Depth and Width

In practice, we find it more efficient to first determine the depth, and then the width. This is because when the network is deep enough (strong attenuation due to upsampling), width becomes less influential, as evident in Fig.10 (b) and the small deviation in Fig.5 (b). However, to relax the need for early stopping at a higher noise level, one may prefer a shallower one, where width matters more (Fig.10 (a)). In this regard, we use the texture features of the images to predict the desired width. This makes intuitive sense as noise-fitting is associated with the amount of high frequency contents in the image, which is manifested in its texture. This is corroborated by the 0.86 Micro-average AUC score shown in Fig.11 (a). The "optimal" width labels we used to train the classifiers were obtained by experimenting with all three width choices on a 2-level hourglass architecture. These width labels are also applicable to decoder architectures as demonstrated by our experiments.

Although the width choices seem limited, we found the images robust to the choice of width to some extent, and simple averaging also works well for ambiguous cases (Appendix D). In fact, some images have multiple width labels. We included these cases in the released Texture-DIP dataset.

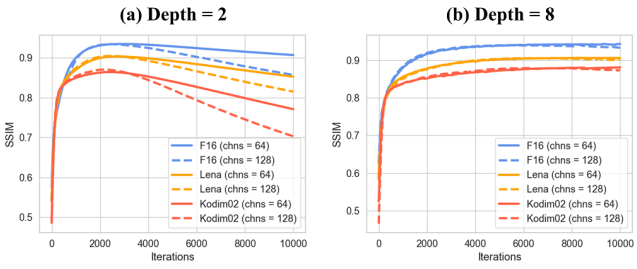


Figure 10: (a) Width critically influences a shallower network, while **(b)** it rarely has any impact on a sufficiently deep network.

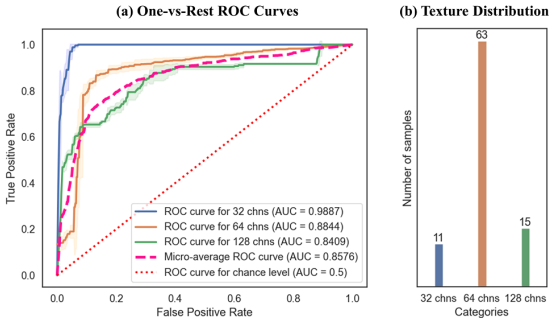


Figure 11: (a) ROC curves with AUC scores for width classification on images from Set9, Set12 and CBSD68. 5-fold cross-validation is performed and repeated 10 times. **(b)** Overview of **our Texture-DIP Dataset**.

4.5. What about Transformers?

Transformers [34] have become integral parts of modern deep neural network architectures. We experimented with Swin-UNet [2], an encoder-decoder pure transformer consisting of Swin Transformer blocks [21] and skip connections. Note that Swin-UNet relies on patch merging/expanding layers for $2\times$ down-/upsampling without involving unlearnt upsampling. We replaced all the patch expanding layers with $2\times$ bilinear upsampling operations and compared the performance of the modified network with the original. We did not replace the patch merging layers with the corresponding downsampling layers or strided convolutions because we did not observe any significant difference. The results presented in Fig. 12 are consistent with our findings on CNNs. For more implementation details, please refer to our code.

5. Conclusion and Future Work

We presented simple but effective solutions to the challenging DIP architectural design in the context of image denoising. Leveraging the spectral effects of upsampling and its relationships with other architectural compoenents, we show that simple architectural changes yield highly-

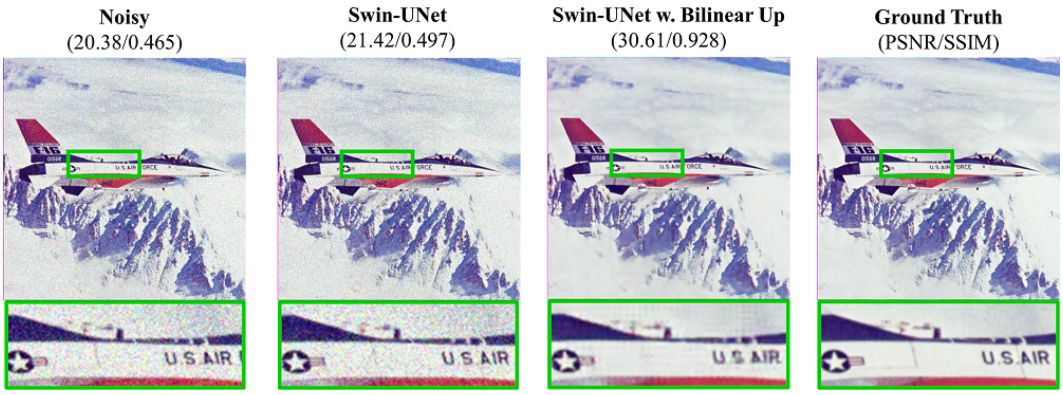


Figure 12: Visual comparisons on the transformer-based Swin-UNet [2] with and without bilinear upsampling.

effective under-parameterized networks that could surpass the larger counterparts and does not critically rely on early-stopping. Although this work focuses on denoising, insights learned about upsampling can be employed in the future to understand the relationship between architectural characteristics and other image restoration tasks. We hope our study could encourage efficient architectural design for DIP and image synthesis in general.

Acknowledgements

This work was supported in part by the United States National Institutes of Health (NIH) under Grant CA266702 and Grant EB008374.

References

[1] Metin Ersin Arican, Ozgur Kara, Gustav Bredell, and Ender Konukoglu. Isnas-dip: Image-specific neural architecture search for deep image prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1960–1968, 2022. 1, 2, 6

[2] Hu Cao, Yueyue Wang, Joy Chen, Dongsheng Jiang, Xiaopeng Zhang, Qi Tian, and Manning Wang. Swin-unet: Unet-like pure transformer for medical image segmentation. In *European conference on computer vision*, pages 205–218. Springer, 2022. 8, 9

[3] Prithvijit Chakrabarty and Subhransu Maji. The spectral bias of the deep image prior. *arXiv preprint arXiv:1912.08905*, 2019. 1, 3

[4] Stanley H Chan, Xiran Wang, and Omar A Elgendy. Plug-and-play admm for image restoration: Fixed-point convergence and applications. *IEEE Transactions on Computational Imaging*, 3(1):84–98, 2016. 1

[5] Keshigeyan Chandrasegaran, Ngoc-Trung Tran, and Ngai-Man Cheung. A closer look at fourier spectrum discrepancies for cnn-generated images detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7200–7209, 2021. 4

[6] Yun-Chun Chen, Chen Gao, Esther Robb, and Jia-Bin Huang. Nas-dip: Learning deep image prior with neural architecture search. In *European Conference on Computer Vision*, pages 442–459. Springer, 2020. 2, 5, 6

[7] 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. 2

[8] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20:273–297, 1995. 6

[9] Kostadin Dabov, Alessandro Foi, and Karen Egiazarian. Video denoising by sparse 3d transform-domain collaborative filtering. In *2007 15th European Signal Processing Conference*, pages 145–149. IEEE, 2007. 6, 7

[10] Mohammad Zalbagi Darestani and Reinhard Heckel. Accelerated mri with un-trained neural networks. *IEEE Transactions on Computational Imaging*, 7:724–733, 2021. 1, 2, 3, 6

[11] Ricard Durall, Margret Keuper, and Janis Keuper. Watch your up-convolution: Cnn based generative deep neural networks are failing to reproduce spectral distributions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7890–7899, 2020. 4

[12] Joel Frank, Thorsten Eisenhofer, Lea Schönherr, Asja Fischer, Dorothea Kolossa, and Thorsten Holz. Leveraging frequency analysis for deep fake image recognition. In *International conference on machine learning*, pages 3247–3258. PMLR, 2020. 4

[13] 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. 5

[14] Robert M Haralick, Karthikeyan Shanmugam, and Its' Hak Dinstein. Textural features for image classification. *IEEE Transactions on systems, man, and cybernetics*, (6):610–621, 1973. 6

[15] Reinhard Heckel and Paul Hand. Deep decoder: Concise image representations from untrained non-convolutional networks. *arXiv preprint arXiv:1810.03982*, 2018. 1, 2, 5, 6

[16] Reinhard Heckel and Mahdi Soltanolkotabi. Denoising and regularization via exploiting the structural bias of convolu-

tional generators. *arXiv preprint arXiv:1910.14634*, 2019. 1, 5

[17] Kary Ho, Andrew Gilbert, Hailin Jin, and John Collomosse. Neural architecture search for deep image prior. *Computers & graphics*, 98:188–196, 2021. 1, 2

[18] Yeonsik Jo, Se Young Chun, and Jonghyun Choi. Rethinking deep image prior for denoising. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5087–5096, 2021. 2

[19] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. *Advances in Neural Information Processing Systems*, 34:852–863, 2021. 5

[20] Jiaming Liu, Yu Sun, Xiaojian Xu, and Ulugbek S Kamilov. Image restoration using total variation regularized deep image prior. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7715–7719. Ieee, 2019. 2

[21] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021. 8

[22] Wei-Yin Loh. Classification and regression trees. *Wiley interdisciplinary reviews: data mining and knowledge discovery*, 1(1):14–23, 2011. 6

[23] Florian Luisier, Thierry Blu, and Michael Unser. Image denoising in mixed poisson–gaussian noise. *IEEE Transactions on image processing*, 20(3):696–708, 2010. 2

[24] Gary Mataev, Peyman Milanfar, and Michael Elad. Deepred: Deep image prior powered by red. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019. 2

[25] Augustus Odena, Vincent Dumoulin, and Chris Olah. Deconvolution and checkerboard artifacts. *Distill*, 1(10):e3, 2016. 3

[26] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *International Conference on Machine Learning*, pages 5301–5310. PMLR, 2019. 5

[27] Stefan Roth and Michael J Black. Fields of experts. *International Journal of Computer Vision*, 82(2):205–229, 2009. 6

[28] Katja Schwarz, Yiyi Liao, and Andreas Geiger. On the frequency bias of generative models. *Advances in Neural Information Processing Systems*, 34:18126–18136, 2021. 4

[29] Zenglin Shi, Pascal Mettes, Subhransu Maji, and Cees GM Snoek. On measuring and controlling the spectral bias of the deep image prior. *International Journal of Computer Vision*, 130(4):885–908, 2022. 2, 3

[30] Eero P Simoncelli and Bruno A Olshausen. Natural image statistics and neural representation. *Annual review of neuroscience*, 24(1):1193–1216, 2001. 3

[31] Shakarim Soltanayev and Se Young Chun. Training deep learning based denoisers without ground truth data. *Advances in neural information processing systems*, 31, 2018. 2

[32] Charles M Stein. Estimation of the mean of a multivariate normal distribution. *The annals of Statistics*, pages 1135–1151, 1981. 2

[33] 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. 1, 3, 5, 6

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 8

[35] Hengkang Wang, Taihui Li, Zhong Zhuang, Tiancong Chen, Hengyue Liang, and Ju Sun. Early stopping for deep image prior. *arXiv preprint arXiv:2112.06074*, 2021. 2

[36] Jun Xu, Hui Li, Zhetong Liang, David Zhang, and Lei Zhang. Real-world noisy image denoising: A new benchmark. *arXiv preprint arXiv:1804.02603*, 2018. 6, 7

[37] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE transactions on image processing*, 26(7):3142–3155, 2017. 1, 6

[38] Kai Zhang, Wangmeng Zuo, Shuhang Gu, and Lei Zhang. Learning deep cnn denoiser prior for image restoration. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3929–3938, 2017. 1