Can a single image denoising neural network handle all levels of Gaussian noise?

Yi-Qing Wang, and Jean-Michel Morel[1]

EDICS: SAS-MALN, IMD-ANAL

Abstract

A recently introduced set of deep neural networks designed for the image denoising task achieves state-of-the-art performance. However, they are specialized networks in that each of them can handle just one noise level fixed in their respective training process. In this note, by investigating the distribution invariance of the natural image patches with respect to linear transforms, we show how to make a single existing deep neural network work well across all levels of Gaussian noise, thereby allowing to significantly reduce the training time for a general-purpose neural network powered denoising algorithm.

1 Introduction

Recently, a set of deep neural networks, or multi-layer perceptrons (MLPs) designed for the image denoising task [5] has been shown to outperform BM3D [8], widely accepted as the state-of-the-art. Although these enormous deep networks only work at the noise levels which they were trained for, this turn of events clearly demonstrates the potential of a pure learning strategy. A philosophical difference sets the two patch-based methods apart: BM3D, a major spin-off of the original non-local means [4], seeks information exclusively inside the noisy image while the neural network derives all its power by looking at noisy and clean patch pairs gathered from other images. Other algorithms exist [6, 9, 12–14] which fall between these two ends of the spectrum.

It is well known [1, 10] that neural networks form a class of universal approximators. Recent studies [2] further suggest that multi-layer neural networks tend to be more efficient in signal representation than their traditional one-hidden-layer counterpart. Whatever their architectures, neural networks in a regression framework seek to approximate the conditional expectation under some input distribution. In our setting, let x, $\tilde{x}$, and y denote the clean, noisy and denoised patch. Note that y does not necessarily have the same dimension as $\tilde{x}$. All the neural networks [5] under this

[1]This work was supported in part by DxO Labs, ERC(AG Twelve Labours) and ONR N00014-97-1-0839. The authors are with CMLA, Ecole Normale Supérieure de Cachan, Cachan, 94230, France. Email: yqwang9@gmail.com; morel@cmla.ens-cachan.fr. Telephone: (+33) 0147402987. Fax: (+33) 0147405901.

study, for instance, produce a $y \in \mathbb{R}^{17\times17}$ based on a noisy observation $\tilde{x} \in \mathbb{R}^{39\times39}$ which includes not only y's corresponding noisy pixels but also its surrounding ones. Also note that with the *true* natural patch distribution beyond reach, a huge number of patches are drawn stochastically [3] from a large natural image dataset to provide the underlying distribution for computing

$$\begin{aligned}\theta^* &= \operatorname*{argmin}_\theta \mathbb{E}\|f(\tilde{x},\theta) - x\|_2^2 \\ &= \operatorname*{argmin}_\theta \mathbb{E}\|f(\tilde{x},\theta) - \mathbb{E}[x|\tilde{x}]\|_2^2\end{aligned} \tag{1}$$

where $f(\cdot,\theta) : \tilde{x} \mapsto y$ is a neural network parametrized by its connection weights θ. Due to this problem's non-convex nature in general, instead of the potentially intractable θ^*, a good θ, judged on the basis of the resulting network's generalization error, is usually accepted as a solution.

In spite of their impressive performance, the proposed deep networks are impractical. As stated in the paper itself [5]: *our most competitive MLP is tailored to a single level of noise and does not generalize well to other noise levels. This is a serious limitation which we already tried to overcome with an MLP trained on several noise levels. However, the latter does not yet achieve the same performance for* $\sigma = 25$ *as the specialized MLP.* Since a standard general-purpose algorithm for Gaussian noise removal ought to be able to handle all levels of noise, this limitation seems to require a series of such networks, one for each noise level, which is impractical and even unrealistic especially in view of their prohibitive training time [7].

In this note, through an analysis of the interplay between the neural networks and the underlying patch distribution they seek to learn, we show how to construct a linear transform that moves natural image patches within the support of their distribution, which is then used to make a single existing deep neural network work well across all levels of Gaussian noise. In the concluding section, using the natural patch distribution invariance argument, we hint that further patch normalization may help scale down these deep neural networks by reducing their domain of definition without compromising their power.

2 A single neural network for all noise levels

To make a single neural network work for all noise levels, first we need to investigate the statistical regularity of the *natural patch space*, or the support of the natural patch distribution, with respect to some linear transforms: we drew 10^6 39-by-39 random patches from the *Berkeley Segmentation Dataset* (BSD500) rendered to grayscale with *Matlab*'s `rgb2gray` function. The patches were then normalized using the formula

$$\bar{p} = (p/255 - 0.5) \cdot 5 \tag{2}$$

provided in [5] in order to conform them to these deep neural networks' training patch distribution. For each normalized patch, we computed the mean and standard deviation of its 1521 pixels and then plotted their population distribution along these two dimensions. For comparison, we did the same with the *PASCAL VOC 2012* dataset.

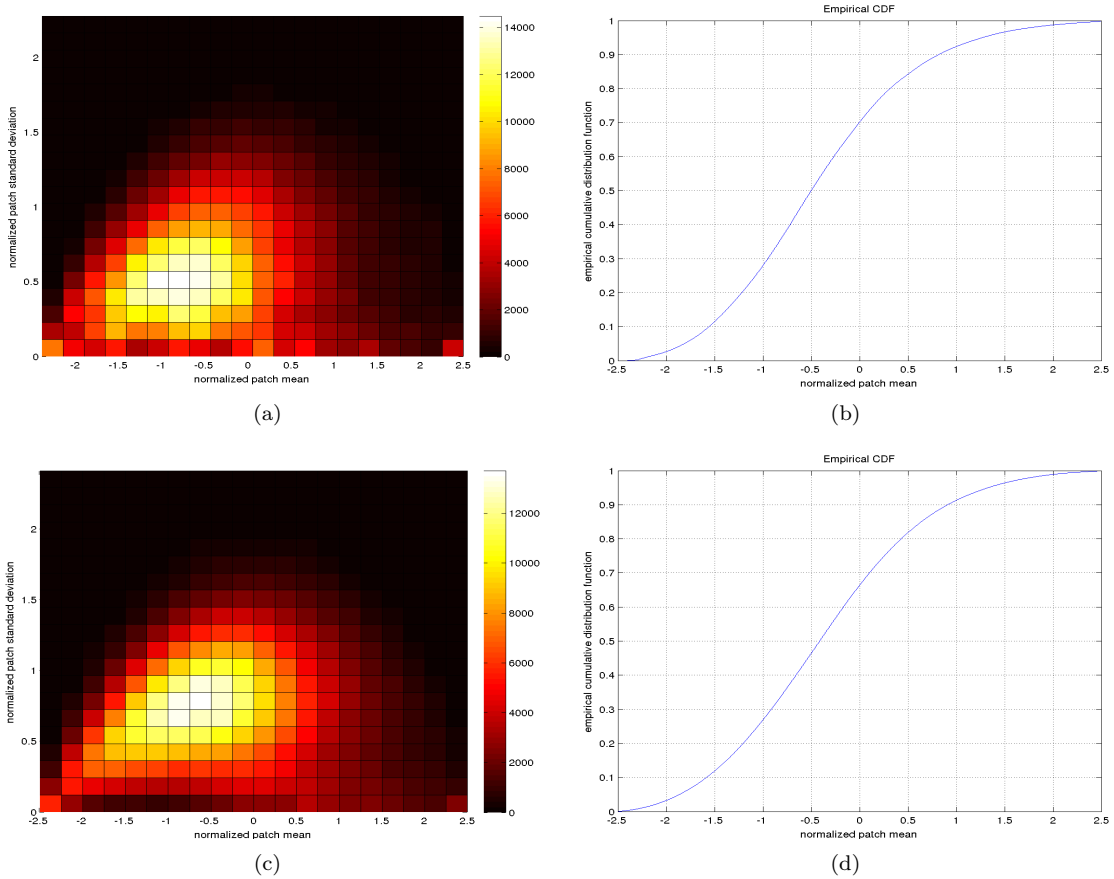


Figure 1: 2D histogram of random patches from 1(a) the grayscale BSD500 and 1(c) the grayscale *PASCAL VOC 2012*. Their horizontal (resp. vertical) axis represents the normalized patch's mean (resp. standard variation). 1(d) and 1(b) plot their respective marginal cumulative distribution function along the patch mean. In both cases, the normalized patches have their mean concentrated around -0.5. Also note that at the two ends of the horizontal axis, there are two important flat patch clusters because of saturation.

The results (Fig.1) show that the means of the normalized natural patches concentrate around -0.5. Even without access to the supervised pairs used in [5], we can therefore expect their neural networks trained according to the criterion (1) to do well with patches from this neighborhood because of the sheer number of examples available. Moreover, the patch-wide variance peaking at the patch-wide means around -0.5 strongly indicates a higher tolerance for linear transforms there, that is, it is more probable for a natural patch p transformed by

$$q = a \cdot (p - s) \quad \text{with } s \text{ some constant patch and } a \geq 0$$

to remain *natural* if q's mean is close to -0.5.

To verify this conjecture that a linear transform exists which only moves patches within the support of the natural patch distribution, we designed a test that shifts the means of the patches to the same value before denoising them using a neural network, after which the shifted differences were added back individually to obtain their final, denoised versions. It is important to note that thanks to a high ratio between the patch size and σ, estimating the clean patch mean from its noisy version (Line 7 in Test) is very reliable.

Test Patch Mean Normalization Test

1: **Input:** n clean patches p_j of dimension 39×39
2: **Output:** n denoised (resp. clean) 17×17 patches $\widehat{\mathfrak{p}}_j$ (resp. $\mathfrak{p}_j$)
3: **Parameter:** noise variance σ^2 and desired patch mean value $\mathfrak{m}$
4: **for** $j = 1$ to n **do**
5: retrieve the 17×17 clean patch $\mathfrak{p}_j$ in the center of p_j
6: generate the normalized noisy patch

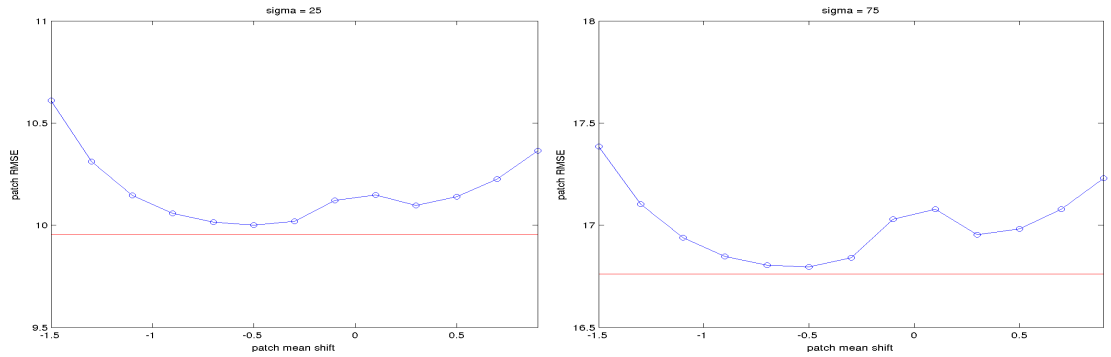
$$\tilde{x}_j \leftarrow ((p_j + n_j)/255 - 0.5) \cdot 5$$

with n_j having 1521 i.i.d. Gaussian random variables $\mathcal{N}(0, \sigma^2)$ and n_j independent of $n_{j'}$ for $j \neq j'$
7: compute the patch mean shift $s_j = \frac{1}{1521} \sum_{k=1}^{1521} \tilde{x}_{jk} - \mathfrak{m}$ where $\tilde{x}_{jk}$ is the k-th pixel in $\tilde{x}_j$
8: shift the network's input $\forall k,\ \tilde{x}_{jk} \leftarrow \tilde{x}_{jk} - s_j$
9: denoise $y_j = f_\sigma(\tilde{x}_j, \theta)$ where $f_\sigma(\cdot, \theta)$ is the deep neural network [5] trained with Gaussian noise standard deviation σ
10: shift the network's output by the same amount $\forall k,\ y_{jk} \leftarrow y_{jk} + s_j$ in the opposite direction
11: reverse the normalization $\widehat{\mathfrak{p}}_j \leftarrow (y_j/5 + 0.5) \cdot 255$
12: **end for**

Running the test on 10^5 39-by-39 random patches drawn from the grayscale BSD500, we computed the resultant root mean square error (RMSE). Fig.2 shows that regardless of the applied noise strength σ, the best empirical performance is attained with the patch mean shift set to -0.5, thereby fully confirming our supposition. In addition, observe that the optimal patch mean shift incurs surprisingly little RMSE loss, which is less than 0.1 for a noise level as high as 75.

The previous analysis paves the way for a generic network able to handle all levels of noise. Let us use the neural network trained with the noise level σ^* for instance. To restore a noisy patch $\tilde{p}$ with noise standard deviation at σ, one multiplies $\tilde{p}$ by $\sigma^*\sigma^{-1}$, apply the patch mean shift and let the neural network operate on the normalized patch (see Algorithm). Henceforth σ^* itself becomes a parameter for further optimization, too.

A comparison among various σ^*-indexed generic networks constructed this way (see Fig.3) shows that the one built with $\sigma^* = 25$ is the best, which is hardly surprising because a neural network can learn the most about the underlying patch space when noise is weak. Moreover, except for the extremely noisy case, one sees no tangible difference in Fig.3(b) between a dedicated network and the best generic network. Also observe that the higher the σ^*, the worse the resulting RMSEs at low noise levels, an expected phenomenon since a high $\sigma^*\sigma^{-1}$ exaggerates the patch-wide variation



(a) (b)

Figure 2: 2(a) (resp. 2(b)) plots the RMSEs resulting from the test with $\sigma = 25$ (resp. 75). The blue curve records the dedicated network's performance with the patch mean values ranging from -1.5 to 0.9. And the red line is the RMSE achieved on the same test data without patch mean shift. Other available neural networks have also been tested with very similar results.

Algorithm Generic Neural Network Denoising

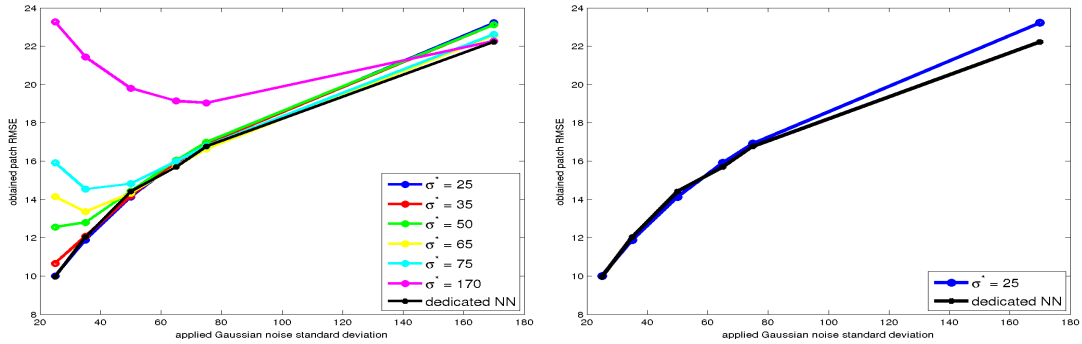
1: **Input:** noisy patch $\tilde{p}$ of dimension 39×39 and its noise level σ
2: **Output:** denoised 17×17 patch $\widehat{\mathfrak{p}}$
3: **Parameter:** noise level σ^* of the trained neural network $f_{\sigma^*}(\cdot, \theta)$ and optimal shift $\mathfrak{m} = -0.5$
4: Scale the noise $\tilde{p} \leftarrow \sigma^* \sigma^{-1} \tilde{p}$
5: Normalize the patch $\tilde{x} \leftarrow (\tilde{p}/255 - 0.5) \cdot 5$
6: Compute the patch mean shift $s = \frac{1}{1521} \sum_{k=1}^{1521} \tilde{x}_k - \mathfrak{m}$ where $\tilde{x}_k$ is the k-th pixel in $\tilde{x}$
7: Shift the network's input mean $\forall k,\ \tilde{x}_k \leftarrow \tilde{x}_k - s$
8: Run the generic neural network denoising $y = f_{\sigma^*}(\tilde{x}, \theta)$
9: Shift the network's output by the same amount $\forall k,\ y_k \leftarrow y_k + s$ in the opposite direction
10: Reverse the normalization $\widehat{\mathfrak{p}} \leftarrow (y/5 + 0.5) \cdot 255 \cdot (\sigma^* \sigma^{-1})^{-1}$

so much that resultant patches no longer remain in the natural patch space.

The implication of this analysis for future research is straightforward: training a network with the same architecture but at an even lower noise level may be rewarding. But it should also be said that too low a σ^* is not likely to work well in a strong noise environment, as already observed in Fig.3. Because the factor $\sigma^* \sigma^{-1}$ will then flatten all the meaningful patch-wide variations, leaving the network unable to tell one patch from another.

Put differently, what we have shown is that the trained deep network is not very different from its most informative section. Hence, one may want to train a network on mean normalized patches to improve performance, thanks to a denser data distribution. However, in so doing, one implicitly trains on a marginal distribution and thus risks losing information.

To conclude, we tested the Algorithm with $\sigma^* = 25$ on some real images. The test set (Fig.4)



(a) (b)

Figure 3: 3(a) Performance discrepancy between dedicated neural networks and σ^*-indexed generic neural networks at handling different noise levels. The horizontal axis marks various test Gaussian noise levels and the vertical axis the achieved RMSEs on 10^5 random patches from BSD500. 3(b) singles out the best generic network with $\sigma^* = 25$.

comprises six noiseless images proposed for benchmarking various denoising algorithms [11] and the results are compiled in Tab.1. Recall that even for $\sigma = 25$, our generic network is different from the dedicated one, in that ours involves one additional step of patch mean shift. The obtained results are thus consistent with Fig.2(a).

(a) (b) (c) (d) (e) (f)

Figure 4: Test images (a) computer (b) dice (c) flower (d) girl (e) traffic (f) valldemossa. All images are of dimension 704×469 except for valldemossa (769×338).

3 Conclusion

In this note, we have shown how to make a single existing neural network work well across all levels of Gaussian noise, thereby allowing to reduce significantly the training time for a general-purpose neural network powered denoising algorithm.

To make deep neural network based algorithm more practical, the other major challenge is to reduce their sizes. This might be achieved through further patch normalization so as to reduce the neural network's input complexity. Rotation and scale invariances of natural image statistics might be used for that purpose.

Table 1: Comparison between dedicated neural networks and our algorithm (generic $\sigma^* = 25$)

$\sigma = 25$	**dedicated**	**generic**
computer	**7.99**	8.10
dice	2.77	2.77
flower	**4.59**	4.63
girl	**3.38**	3.41
traffic	**9.16**	9.22
valldemossa	**11.85**	11.99
avg.	**6.62**	6.68

$\sigma = 35$	**dedicated**	**generic**
computer	**9.63**	9.82
dice	**3.29**	3.45
flower	**5.53**	5.64
girl	**3.88**	4.07
traffic	**10.81**	10.94
valldemossa	**14.21**	14.28
avg.	**7.89**	8.03

$\sigma = 50$	**dedicated**	**generic**
computer	**11.59**	11.92
dice	**4.17**	4.50
flower	**6.85**	7.06
girl	**4.60**	4.92
traffic	**12.83**	13.07
valldemossa	**16.98**	17.06
avg.	**9.50**	9.75

$\sigma = 65$	**dedicated**	**generic**
computer	**13.16**	13.82
dice	**4.83**	5.39
flower	**7.89**	8.20
girl	**5.28**	5.74
traffic	**14.32**	14.78
valldemossa	**18.67**	18.99
avg.	**10.69**	11.15

$\sigma = 75$	**dedicated**	**generic**
computer	**14.10**	14.78
dice	**5.48**	6.14
flower	**8.57**	8.96
girl	**5.66**	6.20
traffic	**15.08**	15.56
valldemossa	**19.97**	20.34
avg.	**11.47**	11.99

$\sigma = 170$	**dedicated**	**generic**
computer	**20.51**	21.81
dice	**9.23**	10.92
flower	**12.84**	13.64
girl	**8.79**	10.54
traffic	**20.71**	21.72
valldemossa	**26.42**	27.26
avg.	**16.41**	17.64

References

[1] A. R. Barron. Approximation and estimation bounds for artificial neural networks. *Machine Learning*, 14(1):115–133, 1994.

[2] Y. Bengio. Learning deep architectures for ai. *Foundations and Trends® in Machine Learning*, 2(1):1–127, 2009.

[3] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *Proc. Int. Conf. Computational Statistics*, pages 177–186. Springer, 2010.

[4] A. Buades, B. Coll, and J. M. Morel. A review of image denoising algorithms, with a new one. *Multiscale Modeling & Simulation*, 4(2):490–530, 2005.

[5] H. Burger, C. Schuler, and S. Harmeling. Image denoising: Can plain neural networks compete with BM3D? In *IEEE Conf. Computer Vision and Pattern Recognition*, pages 2392–2399, 2012.

[6] H. Burger, C. Schuler, and S. Harmeling. Learning how to combine internal and external denoising methods. In *Pattern Recognition*, pages 121–130. Springer, 2013.

[7] H.C. Burger. *Modelling and Learning Approaches to Image Denoising*. PhD thesis, Eberhard Karls Universität Tübingen, Wilhelmstr. 32, 72074 Tübingen, 2013.

[8] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image restoration by sparse 3D transform-domain collaborative filtering. In *Electronic Imaging*, 2008.

[9] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Trans. Image Processing*, 15(12):3736–3745, 2006.

[10] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

[11] M. Lebrun. An Analysis and Implementation of the BM3D Image Denoising Method. *Image Processing On Line*, 2012:175–213, 2012.

[12] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Non-local sparse models for image restoration. In *IEEE Int. Conf. Computer Vision*, pages 2272–2279, 2009.

[13] Y. Q. Wang and J. M. Morel. SURE guided gaussian mixture image denoising. *SIAM Journal on Imaging Sciences*, 6(2):999–1034, 2013.

[14] D. Zoran and Y. Weiss. From learning models of natural image patches to whole image restoration. In *IEEE Int. Conf. Computer Vision*, pages 479–486, 2011.