UNDERSTANDING NEURAL-NETWORK DENOISERS THROUGH AN ACTIVATION FUNCTION PERSPECTIVE

Yuxiang Li

Ecole polytechnique, Palaiseau, France

Bo Zhang, Raoul Florent

Philips Research - Medisys, Suresnes, France

ABSTRACT

Images are usually corrupted by noise during acquisition or transmission. Smoothing and thresholding within an adapted domain is a common practice to recover the clean image. Recently, many researchers try to train a denoiser using neural network on a large image training set. Some of them show competitive performance compared to state-of-the-art filter such as BM3D [1][2]. At the same time, neural network's performance has been improved with new activation functions, new solvers, etc., which makes machine learning techniques more reliable. In this work, we leverage cutting-edge techniques in the neural network world to reconsider the problem of image denoising with plain neural network. Instead of seeking blindly a better result, we focus on analysing and understanding the denoising strategies conceived by the neural network. We will provide a perspective through activation functions, and explain how they play a decisive role in the denoising mechanisms.

Index Terms— Activation function, Neural network denoising, Basis pursuit

1. INTRODUCTION

Image denoiser takes a noisy image and outputs a clean-image estimate. Reformulated this estimation problem as a minimization of the Mean Squared Error (MSE) or a maximization of the Signal-to-Noise Ratio (SNR) paves way for applying machine learning techniques, especially neural networks [2][3][4]. Despite their simplicity, neural networks are able to produce competitive results compared to the state-of-the-art filter BM3D [1]. In addition to the outstanding performance, Burger et al. give a first insight about how to apply plain neural network and how different settings can affect the output, e.g. depth of network, size of training data, etc. [5][6].

Several years after all these attempts, many new components of neural networks have been proposed: new activation functions such as ReLU [7], PReLU [8] and piecewise linear activation [9], as well as new solvers like RMSProp [10]. Record for image classification has been broken every year [11] with all these inventions. Equipped with novel techniques, can plain neural network break previous result on image denoising task? And more importantly, is the underlying denoising strategy changed when applying these techniques?

Contributions: Our work is based on a plain neural network-based image denoiser [2]. We concentrate on the activation function, which we believe to be crucial in deciding the network denoising strategy. We show first that different activation functions can lead to quite different performances. Secondly, we analyze how network denoises an image patch and in particular how this denoising mechanism changes in accordance to the activation function. Finally, we will study a Trainable activation layer, which leads to a link between network's denoising mechanism and the well-known dictionary thresholding method.

2. DENOISING PIPELINE

In this section, we describe how to train and test a neural-network based denoiser. Since our purpose is to understand activation functions, our network structure is taken directly from Burger et al.'s work [2]. All our models are implemented with Keras, a neural network framework.

2.1. Dataset

Like deep-learning based methods, the dataset needs to be general and large in order to cover enough possibilities. Thanks to the randomness of noise, we can generate an infinite number of noisy images from a rather small dataset. In our experiments, all images are represented by grayscale intensity from 0 to 255. The noise we assume is an additive white Gaussian noise with a standard deviation of 25. The following datasets are used for training and testing:

- *Training set:* 50,000 images from ImageNet validation set 2010 [12] cropped to 256×256
- *Test set:* Barbara, Cameraman, Shepp-Logan phantom [13] and 200 images from ImageNet test set 2010 [12]

2.2. Neural network structure

Our plain neural network has 4 hidden layers of 2047 neurons each just as in Burger's paper [2]. The input and output are an image patch of size 17×17 flattened to a vector of dimension

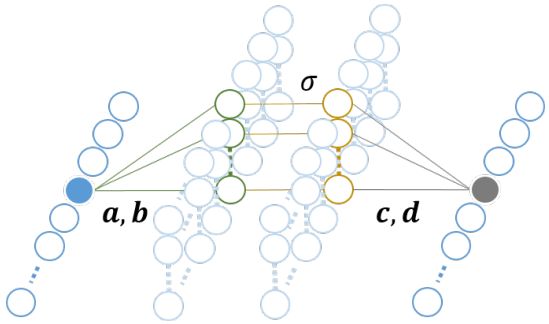

Fig. 1: Trainable activation layer

289. After each hidden layer, we append an activation layer that can be Tanh, ReLU or a Trainable activation layer.

Our Trainable activation layer is inspired by the Mlpconv layer in [14]. It is trained along with the network and it acts as a classic activation function. Our new layer is composed by 2 convolutional layers and 1 activation layer. We use ReLU in our experiments, but other functions may also work. The first convolution works on a support of 1×1, the number of channels at its output is 127. The second one works on a support of $1 \times 1 \times 127$ then outputs just one single channel which is the activation value. The number of channels, e.g. 127, is fixed arbitrarily. We believe it to be flexible enough to simulate any function we want. We note $\mathbf{a}$ and $\mathbf{c}$ the weight vector (one column matrix) of the convolutional layers, $\mathbf{b}$ and $\mathbf{d}$ their associate bias and σ the inner activation function. For a scalar x, the overall activation function can be expressed as:

$$\text{Trained}(x) = \mathbf{c} \cdot \sigma(x\mathbf{a} + \mathbf{b}) + \mathbf{d} \quad (1)$$

Our new activation is not adaptive to each neuron, contrary to other adaptive activation functions [8][9].

2.3. Training phase

We use MSE as our loss function, RMSProp [10] as our solver. The initial training rate is set to 0.0005 which is determined by experience. The batch size is 100, e.g. 100 image patches of size 17×17. Under these settings, the training process works as follow:

1. Pick randomly an image from the *Training set*;
2. Choose a random position and generate a clean patch;
3. Add noise to the patch according to the parameters;
4. Clip image range into [0, 255];
5. Rescale image dynamics within [-2.5, 2.5] [2];
6. Add the couple of noisy and clean patches to the batch.

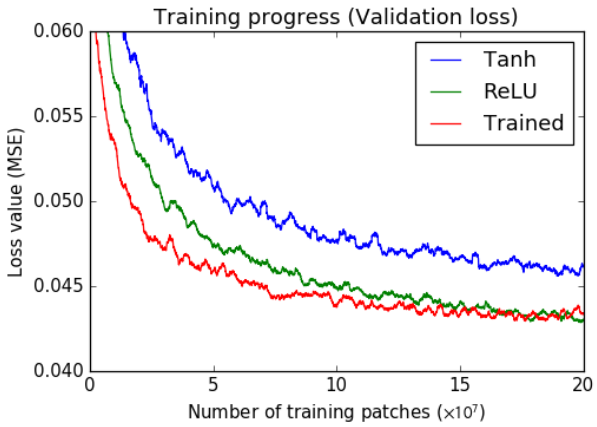

Fig. 2: Training progress of different models. The loss is evaluated on image patches from ImageNet test set 2010.

2.4. Testing phase

Once the neural network trained, it acts on the dimension of the patch. We need to denoise all possible patches on the image in order to get a full-sized output. For this, we use the framing window technique that sums up overlapping patches with a Gaussian kernel. In practice, we set the stride between windows to 3 and the standard deviation of Gaussian kernel to 3. Do notice that the shape of the kernel can have impact on the final outcome and it depends on the size of the patch we use.

3. RESULTS

Models with Tanh, ReLU and Trainable activation layer have been trained on 200 million image patches. Although they can be further improved with more training data, it is enough for us to analyze the relation between activation function and denoising mechanism. We notice on the training curve (Figure 2) that:

1. ReLU and trained activation show a faster convergence than Tanh with a significant margin.
2. Trained activation converges most rapidly, but the result falls eventually in the same range with ReLU.

The loss function gives us a first indication on how models perform with different activation function. We then test them on our *Test set*. We add, in addition, the BM3D filter [1] to illustrate the comparable performance from neural network-based denoisers (cf. Table 1).

Within network-based denoisers, we notice that ReLU and trained function achieve better result on almost every images compared to Tanh, especially for images with textures and black & white regions. "Barbara" and "Phantom" are 2 typical images where ReLU performs better than Tanh (cf. Figure 3 and 4).

Fig. 3: Comparison between clean "barbara" image, noisy image (20.28 dB), result of Tanh (28.18 dB) and ReLU (29.33 dB).

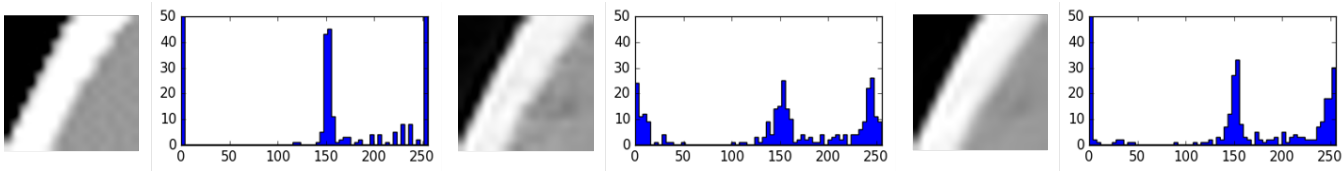

Fig. 4: Results on a 17 $\times$ 17 image patch from "Phantom": clean patch, result of Tanh (26.05 dB), result of ReLU (27.02 dB) and their historgram of intensity.

Image	Barbara	C.man	Phantom	Average
Noisy	20.28	20.55	21.53	20.63
BM3D	**30.67**	29.11	29.55	28.61
Tanh	28.18	28.97	31.87	29.11
ReLU	29.33	**29.28**	32.80	**29.38**
Trained	29.16	29.22	**32.91**	29.28

Table 1: PSNRs on 3 specific images and the average PSNR on 200 other images from the *Test set* (dB)

4. DISCUSSION

In this section, we try to provide insights into the denoising mechanism, its relation with activation functions and the implication of our Trainable activation layer.

4.1. How a network manages to denoise a patch

Let us visualize our neural network in details (cf. Figure 5).

The power of network lies on its nonlinearity and adaptiveness. It acts differently in accordance to the input patch. In order to visualize this adaptiveness, we feed our network with different patches and compute for each of them the **equivalent transform matrix**. The last equivalent matrix is plotted with 17 $\times$ 17 kernels of size 17 $\times$ 17 associated with 289 pixels on the output patch, e.g. the product of the first kernel with the input gives the value of the top-left pixel. Each kernel reflects the dependency between one output pixel and all input pixels. As we see on Figure 5, in case of ReLU, these kernels are related to structures in the input [15].

We know that neural network adapts its kernels to input, but why kernels from ReLU are significantly different than those of Tanh? One should have noticed that Tanh is a saturating function, whereas ReLU is not. This saturating property has a key impact on learning input and output patterns, e.g. the first and the last weight matrices. These patterns tell us which information interests the network at input and how the network outputs an image.

In case of Tanh, input patterns are full of Dirac patterns. Burger et al. state that Tanh reduces the noise by limiting the value of each neuron [6]. From pixel to edge, Tanh limits noise presence on different scales at different layers. However, by considering large value as a noise, the filter will degrade extreme value like black and white (see intensity histograms of Figure 4).

Unlike Tanh, ReLU is not saturating: it keeps extreme values directly to the output. Instead of working on pixels with Dirac patterns, ReLU works directly on edge / ridge detectors which are larger structures and thus more immune to noise. In terms of denoising mechanism, a neural network with ReLU disintegrates the image patch into multiple patterns and reconstruct it with noise-free ones. The filter is constructed in the same manner as other transform-domain (e.g. wavelets) denoisers do.

More precisely, ReLU makes the network behave as a sparse-representation based denoiser. In fact, when we look at the activations of the last layer, we notice that activated neurons in case of ReLU are rare. The activation values follow a spiky distribution, whereas for Tanh, the activation of noisy patterns is not marginal enough. As a consequence, Tanh's re-

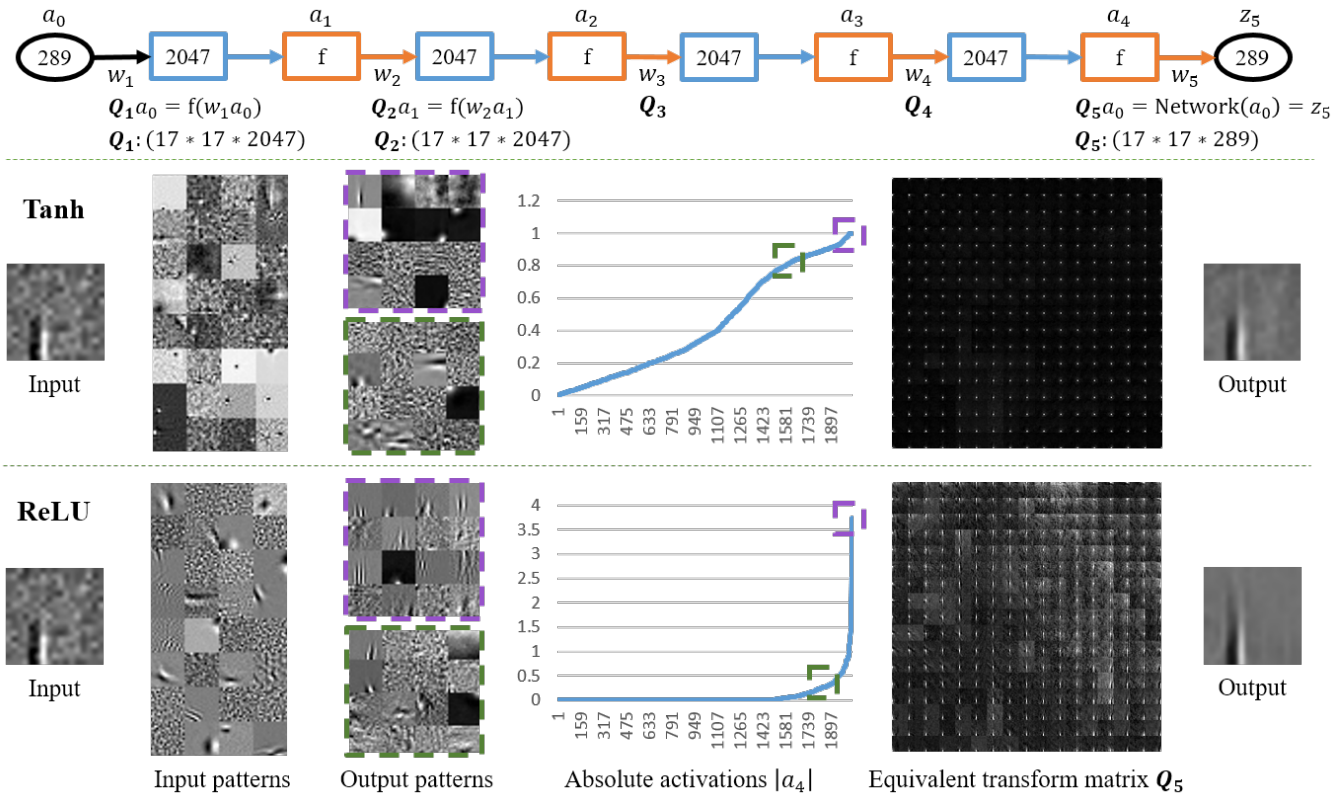

Fig. 5: Visualization of network-based denoising mechanism in case of Tanh and ReLU. We use z and a for before and after activation value, w for weight matrix and Q for equivalent transform matrix. For output patterns, we show those with high activation (purple, first group) and with low activation (green, second group).

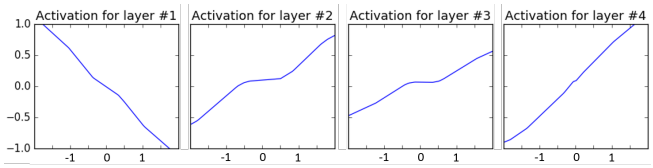

Fig. 6: Learned activation functions for a neural network with 4 hidden layers after 200 million training patches

sult includes clearly more noisy values. In other words, ReLU promotes a compact and sparse image representation, while Tanh seems not.

4.2. Thresholding function: an optimal activation

To find an optimal activation function and its associated denoising mechanism, we adopt the Trainable activation layer. In the simplest case of a single-hidden-layer network where we have a transform - inverse transform pipeline, the network works in the similar way as basis-pursuit denoising methods. The learned activation should look like a Wiener thresholding [16].

In case of more hidden layers (here we used 4), this intuition seems still valid: the resulting activation functions look surprisingly similar to thresholding functions (cf. Figure 6). This shows an interesting link to the widely used dictionary thresholding denoisers [16][17][18]. We believe that the denoising mechanism behind this old function is different from what we have presented for Tanh or ReLU, however, the kernels we obtain with thresholding functions are not easily recognizable which prevents us from giving a more concrete explanation.

5. CONCLUSION

Plain neural network has shown to achieve state-of-the-art denoising performance. Moreover, its denoising mechanism fundamentally changes as we select different activation function. One could imagine applying specific activation functions on some specific tasks to get a better performance. For this purpose, we introduce the Trainable activation layer which reveals the optimal function expected by the network. For the denoising task, it produces a thresholding function which seems to share a similar strategy of classical sparsity-promoting basis-pursuit approaches.

6. REFERENCES

[1] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian, "Image denoising by sparse 3-d transform-domain collaborative filtering," *Image Processing, IEEE Transactions on*, vol. 16, no. 8, pp. 2080–2095, 2007.

[2] Harold C Burger, Christian J Schuler, and Stefan Harmeling, "Image denoising: Can plain neural networks compete with bm3d?," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 2392–2399.

[3] Forest Agostinelli, Michael R Anderson, and Honglak Lee, "Adaptive multi-column deep neural networks with application to robust image denoising," in *Advances in Neural Information Processing Systems*, 2013, pp. 1493–1501.

[4] Junyuan Xie, Linli Xu, and Enhong Chen, "Image denoising and inpainting with deep neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 341–349.

[5] Harold Christopher Burger, Christian J Schuler, and Stefan Harmeling, "Image denoising with multi-layer perceptrons, part 1: comparison with existing algorithms and with bounds," *arXiv preprint arXiv:1211.1544*, 2012.

[6] Harold Christopher Burger, Christian J Schuler, and Stefan Harmeling, "Image denoising with multi-layer perceptrons, part 2: training trade-offs and analysis of their mechanisms," *arXiv preprint arXiv:1211.1552*, 2012.

[7] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[9] Forest Agostinelli, Matthew Hoffman, Peter Sadowski, and Pierre Baldi, "Learning activation functions to improve deep neural networks," *arXiv preprint arXiv:1412.6830*, 2014.

[10] Tijmen Tieleman and Geoffrey Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural Networks for Machine Learning*, vol. 4, pp. 2, 2012.

[11] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al., "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 248–255.

[13] Lawrence A Shepp and Benjamin F Logan, "The fourier reconstruction of a head section," *Nuclear Science, IEEE Transactions on*, vol. 21, no. 3, pp. 21–43, 1974.

[14] Min Lin, Qiang Chen, and Shuicheng Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013.

[15] Aapo Hyvärinen and Erkki Oja, "Independent component analysis: algorithms and applications," *Neural networks*, vol. 13, no. 4, pp. 411–430, 2000.

[16] Robert D Nowak and Richard G Baraniuk, "Wavelet-domain filtering for photon imaging systems," *IEEE Transactions on Image Processing*, vol. 8, no. 5, pp. 666–678, 1999.

[17] S Grace Chang, Bin Yu, and Martin Vetterli, "Adaptive wavelet thresholding for image denoising and compression," *IEEE transactions on image processing*, vol. 9, no. 9, pp. 1532–1546, 2000.

[18] Xiao-Ping Zhang, "Space-scale adaptive noise reduction in images based on thresholding neural network," in *Acoustics, Speech, and Signal Processing, 2001. Proceedings.(ICASSP'01). 2001 IEEE International Conference on*. IEEE, 2001, vol. 3, pp. 1889–1892.