Convolution Padding in Recurrent Neural Networks for Image Denoising with Limited Data

Alex Ho
Department of Applied Mathematics
University of California, Merced
Merced, CA 95343 USA
aho38@ucmerced.edu

Jacqueline Alvarez
Department of Applied Mathematics
University of California, Merced
Merced, CA 95343 USA
jalvarez94@ucmerced.edu

Roummel F. Marcia
Department of Applied Mathematics
University of California, Merced
Merced, CA 95343 USA
rmarcia@ucmerced.edu

2021 55th Asilomar Conference on Signals, Systems, and Computers | 978-1-6654-5828-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/IEEECONF53345.2021.9723313

***Abstract*—Recurrent neural networks are widely used in applications for time sequence prediction, such as speech recognition, text prediction, and target tracking. These networks are not popular for image restoration tasks due to the fact that there is no time dependency on images. In this paper, we repurpose a recurrent neural network to recover images from noisy observations and investigate convolutional padding to improve the results. Our proposed method artificially creates a time dependency between the image reconstructions at different iterations of the algorithm, allowing us to use a recurrent neural network. In addition, we do not train the network over the true images. Rather, we only utilize the noisy image and the structure of the network to perform image denoising tasks. We test our method using images from the CIFAR-10 dataset and present our results using the structural similarity index.**

***Index Terms*—Image denoising, recurrent neural networks, convolutional padding, limited data**

I. Introduction

Recurrent neural networks (RNNs) have shown to be very effective in many supervised learning applications. In many sequence prediction problems, such as text or speech, an RNN is able to create promising predictions by using information from all time-steps. Usually the network is used to discover patterns that exist within the data between different time-steps. The structure of the network allows correlations between elements within a sequence to be explored during training. However, RNNs are not widely used in image proccessing applications because images do not possess a sequential structure. Here, we use RNNs for image denoising in contrast to other machine learning approaches (see e.g., [1]–[5]). In particular, by adding a dependency of the previous time-step on an input image for all the time-steps, a sequence of data is created for images that will allow an RNN to do its intended task [6], [7]. In addition, we study various types of convolutional padding to improve the results.

II. Problem Formulation

Recurrent neural networks have been shown to effectively process sequential data for tasks such as speech recognition, target tracking, and a variety of others [8]–[10]. Unlike feed-forward networks, RNNs contain feedback loops which correspond to an internal memory. Therefore these networks can draw correlations between different elements in the sequence, making them useful for time-dependent problems [11], [12]. A typical RNN structure can be described by the following:

$$\mathbf{h}_t = \tanh(\mathbf{W}\mathbf{h}_{t-1} + \mathbf{U}\mathbf{x}_t + \mathbf{b})$$
$$\hat{\mathbf{y}}_t = \text{softmax}(\mathbf{V}\mathbf{h}_t + \mathbf{c})$$

where $\mathbf{W}, \mathbf{U}$ and $\mathbf{V}$ are weight matrices, $\mathbf{b}$ and $\mathbf{c}$ are bias vectors, $\mathbf{x}_t$ is the input vector and $\mathbf{h}_t$ is the current state used to find the predicted output vector $\hat{\mathbf{y}}_t$ where time goes from $t = 0$ to $t = T$.

Related work. Recurrent neural networks are not traditionally used in image processing applications, however there has been recent work in this field. For example, RNNs have been utilized for hyperspectral imaging denoising [16]. In addition, Long Short-Term Memory (LSTM) networks have gained popularity and have been used in image denoising methods [14], [15]. However, these methods all require a traditional training dataset and procedure.

In [13], the authors showed that the success of neural networks for image restoration tasks, such as denoising, super-resolution, and inpainting, is not solely based on the ability of these networks to learn from the training data. Their experiments demonstrated that the structure of a convolutional neural network captures some of the image statistics prior to training. The model presented in [13] is untrained, in the sense that we do not require a training dataset with the true images. Instead this formulation allows for the model to be applied in a "plug and play" manner. The input to the network is a random noise "image", and so the only prior information comes from the network itself. The model is optimized over a single noisy image by solving

$$\min_{\theta} ||f_\theta(z) - x_0||^2$$

where z is image-sized Gaussian noise, θ is the randomly initialized parameters of our model, f_θ is the mapping from z to an image x, and x_0 is the noisy image. In this paper, we perform image denoising by reformulating the problem in [13] using a recurrent neural network. This formulation performs the task of denoising while allowing for limited information on the ground truth.

This research is partially supported by the National Science Foundation grants DMS 1840265 and IIS 1741490. A. Ho and J. Alvarez contributed equally to this paper.

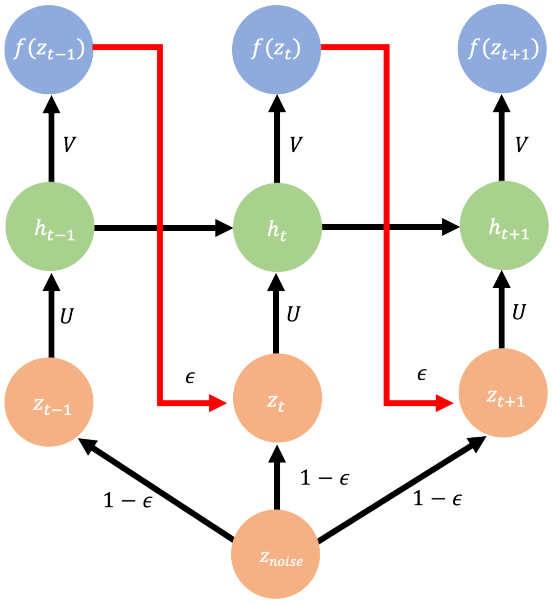

Fig. 1. The architecture of our recurrent neural network at time t takes in a weighted sum of a random noise image z_t as inputs and the output from the previous time. The input is mapped onto a hidden state by U, a convolutional encoder, and outputted as an image using V, a convolutional decoder.

III. Proposed Method

We have incorporated an autoencoder within a recurrent neural network to help construct our hidden states and outputs. The first part of an autoencoder is the encoder and its objective is to reduce the dimension of the input and store it inside a latent space representation. We will be using it as our function to produce the hidden state. The latent space representation is then passed on to the next time step as an input to construct the hidden state of the next time step (see [17] for details). The second part is the decoder, where it will take the information stored within the latent space representation and use it to reconstruct an image. In our case, it is what we use to reconstruct images using the information stored within the hidden state of the current time step. Finally, our result is the output after the final time step. For the structure of the autoencoder, we have chosen convolutional layers as the basis of our encoder and decoder since fully-connected layers, or linear functions, do not have the same denoising properties as convolutional layers [13], [18]–[23].

Architecture. The architecture of the RNN takes the following form:

$$\begin{aligned} z_t &= z_{noise} \times (1-\epsilon) + f(z_{t-1}) \times \epsilon \\ h_t &= U_\phi(z_t) + h_{t-1} \\ f(z_t) &= \tanh(V_\theta(h_t)) \end{aligned} \tag{1}$$

where U is the encoder with parameters ϕ, V is the decoder with parameters θ, z_{noise} is Gaussian noise, and z_t is the current input that is formulated as an auto-regressive Gaussian model using ϵ [18], [24].

Encoder		
Input	**Output**	**Kernel**
3	16	6×6
16	32	6×6
32	64	6×6
64	128	6×6
128	256	5×5

Decoder		
Input	**Output**	**Kernel**
256	128	5×5
128	64	6×6
64	32	6×6
32	16	6×6
16	3	6×6

TABLE I
ARCHITECTURE OF CONVOLUTIONAL AUTOENCODER. THE ENCODER IS COMPOSED OF FIVE 2D CONVOLUTIONAL LAYERS AND THE DECODER IS COMPOSED OF FIVE 2D CONVOLUTIONAL TRANSPOSE LAYERS. ALL LAYERS USE A KERNEL WITH STRIDE = 1.

Both encoder and decoder are convolutional neural networks, each with five convolutional layers followed by a $\tanh$ activation (for more details see Table II) [25]. The encoder takes the input image of size 32×32 and reduces it to a latent space representation which becomes our hidden state, h_t. The decoder, on the other hand, uses transposed convolutional layers that do exactly opposite of our encoder. The input channels and output channels go in reverse order as the encoder, and therefore, will output an image of 3 channels.

Loss. The RNN is trained to minimize the following loss function $\mathcal{L}(\theta;\phi) = \mathbb{E}\left[\ell(\theta;\phi)\right]$, where ℓ is the standard mean-squared error, $\ell(\theta;\phi) = \|y - V_\theta(h_{tf})\|_F^2$, with h_{tf} as the hidden state of the final time and y as the initial noisy image. Back-propagation will be used to update all the weights in our RNN at each time step. This formulation does not require training data, so no information from the true image is needed, instead we utilize the structure of our RNN [19].

IV. Numerical Experiments

Dataset. To evaluate our proposed method we selected 15 images from the CIFAR-10 dataset [26]. The dataset contains 50,000 training images and 10,000 testing images. Each image is RGB and contains 32×32 pixels. The noisy/degraded images, i.e., x_0, used throughout the experiments are created using additive Gaussian noise using a standard deviation of $\sigma = 0.1$. The inputs to our model at each time step is a auto-regressive model shown in Eq. (1).

Architecture Parameters. During training the method is evaluated using the Mean Square Error (MSE) (for details see [17]). We trained using stochastic gradient descent with $T = 2000$ iterations while using a learning rate of 0.001 to update the weights. From preliminary experiments we acheived optimal accuracy by using $\epsilon = 0.1$. After recovering the image we evaluate the denoising approach using the structural similarity index (SSIM) between the recovered image and the true image, which is given by

$$\text{SSIM}(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$

where μ_x and μ_y are the averages of each input respectively, σ_x and σ_y are the associated variances, and c_1 and c_2 are variables used to stabilize the denominator [27].

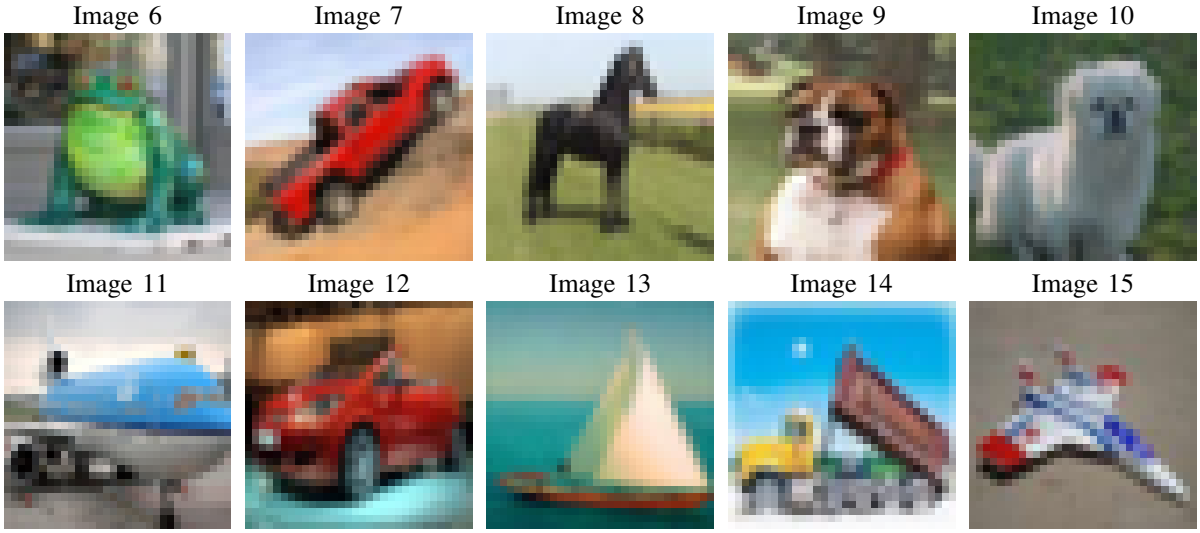

Fig. 2. CIFAR-10 images used in our numerical experiments. Images 1-5 and the corresponding results are presented in detail in Fig. 3.

V. Results

In this section, we investigate the effectiveness of convolutional padding within recurrent neural networks for imaging denoising.

Experiment 1: In the first experiment, we use the method described in Section III with an unpadded image for x_0. Figure 3 includes the results for five different sample images. In Row 2 of Figure 3, we see that our method produces a denoised image of x_0. However, we notice that the corners and edges of the image are of lesser quality. This is likely attributed to the filters of each convolutional layer since there is not enough information along the edges of each image.

Experiment 2: Next, we tested our method using padded images for x_0. Since the restorations using the unpadded images resulted with blurred corners and edges, the padded images should allow the filter of each convolution to pick up more information along the edges of the image. In particular, we use edge padding where the added pixels have the same value as the pixels along the edge of the image, and reflection padding where the pixels are reflected along the edge of the image. Both paddings were used with length of 4. We also investigated using zero padding; however the edge and reflection padding resulted in more accurate denoised images [20]. The results are presented in Rows 3-4 of Figure 3. We find that after incorporating the padding, the corners and edges are restored with better quality. However, we notice that as the quality of the image increases along the edges, the opposite is true for the center.

We looked at two sets of SSIMs for each image: the first compares the entire image and the second compares the center of the image, specifically the center 16×16 pixels. For the padded images, the SSIM is higher for the entire image since the recovery from the corners is of higher quality. In contrast, the unpadded images have a higher SSIM in the center.

VI. Conclusions

In this paper, we have shown that we are able to produce promising results for an image denoising task using a recurrent neural network without any information from the true image. By adding the output from the decoder of the previous time-step to the input Gaussian noise, we created a dependency between current input and previous output which allowed the recurrent neural network to produce the desired result of denoising the image. Furthermore, we found the overall restoration of the image can be improved by adding padding to the input image; however, this slightly diminishes the quality in the center of the image.

	SSIM (full image)			**SSIM** (center of image)		
Image	**Unpad**	**Edge**	**Refl**	**Unpad**	**Edge**	**Refl**
1	0.880	0.907	0.901	0.881	0.827	0.799
2	0.816	0.821	0.830	0.777	0.743	0.788
3	0.907	0.914	0.905	0.874	0.844	0.884
4	0.860	0.887	0.884	0.868	0.855	0.883
5	0.904	0.940	0.944	0.954	0.955	0.954
6	0.873	0.913	0.901	0.919	0.916	0.902
7	0.875	0.886	0.878	0.857	0.851	0.745
8	0.881	0.908	0.903	0.949	0.930	0.929
9	0.863	0.884	0.888	0.959	0.922	0.934
10	0.864	0.888	0.858	0.887	0.894	0.888
11	0.821	0.844	0.870	0.928	0.932	0.935
12	0.868	0.864	0.868	0.873	0.771	0.747
13	0.844	0.870	0.869	0.864	0.874	0.846
14	0.851	0.857	0.865	0.966	0.935	0.945
15	0.832	0.770	0.806	0.947	0.886	0.881

TABLE II
Structural similarity index (SSIM) between denoised image and ground truth of 15 images. Column 2-4: SSIM value of the full image for input images with various padding. Column 4-5: SSIM of the center region of the image for input images with various padding. Results are shown for input images with no padding, edge padding, and reflection padding.

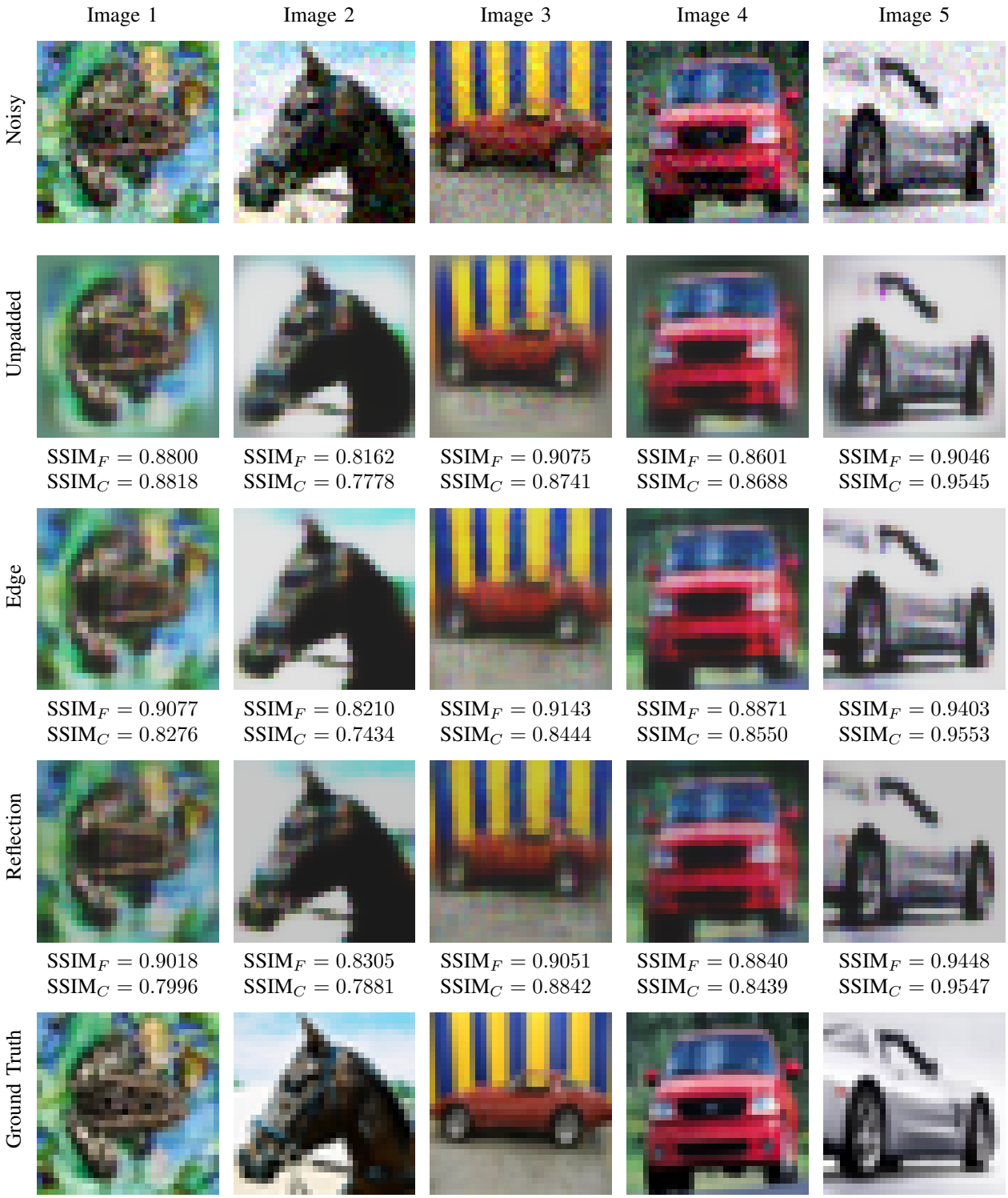

Fig. 3. Image restoration of noisy CIFAR images. Row 1: Noisy image. Row 2: Restoration using an unpadded reference image with corresponding structural similarity index (SSIM) values, where SSIM$_F$ indicates the measurements of the full images and SSIM$_C$ indicates the measurements taken on the center of the images. Row 3-4: Restorations using padded reference images with edge and reflection padding, respectively, are presented with corresponding SSIM$_F$ and SSIM$_C$ values. Row 5: Ground truth image.

REFERENCES

[1] J. Xie, L. Xu, and E. Chen, "Image denoising and inpainting with deep neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 341–349.

[2] Y. Tang, R. Salakhutdinov, and G. Hinton, "Robust boltzmann machines for recognition and denoising," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 2264–2271.

[3] J. Xu, L. Zhang, W. Zuo, D. Zhang, and X. Feng, "Patch group based nonlocal self-similarity prior learning for image denoising," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 244–252.

[4] L. Gondara, "Medical image denoising using convolutional denoising autoencoders," in *2016 IEEE 16th International Conference on Data Mining Workshops*. IEEE, 2016, pp. 241–246.

[5] T. Remez, O. Litany, R. Giryes, and A. M. Bronstein, "Deep class-aware image denoising," in *2017 International Conference on Sampling Theory and Applications*. IEEE, 2017, pp. 138–142.

[6] A. Buades, B. Coll, and J. Morel, "A non-local algorithm for image denoising," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, 2005, pp. 60–65.

[7] P. Pinheiro and R. Collobert, "Recurrent convolutional neural networks for scene labeling," in *International Conference on Machine Learning*, 2014, pp. 82–90.

[8] A. Graves, A. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 6645–6649.

[9] N. Agarwala, Y. Inoue, and A. Sly, "Music composition using recurrent neural networks," *CS 224n: Natural Language Processing with Deep Learning, Spring*, 2017.

[10] A. Graves and J. Schmidhuber, "Offline handwriting recognition with multidimensional recurrent neural networks," in *Advances in Neural Information Processing Systems*, 2009, pp. 545–552.

[11] I. Sutskever, *Training recurrent neural networks*. University of Toronto Toronto, Ontario, Canada, 2013.

[12] R. Jozefowicz, W. Zaremba, and I. Sutskever, "An empirical exploration of recurrent network architectures," *Journal of Machine Learning Research*, 2015.

[13] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9446–9454.

[14] K. N. Haque, M. A. Yousuf, and R. Rana, "Image denoising and restoration with cnn-lstm encoder decoder with direct attention," *arXiv preprint arXiv:1801.05141*, 2018.

[15] R. Rajeev, J. A. Samath, and N. Karthikeyan, "An intelligent recurrent neural network with long short-term memory (lstm) based batch normalization for medical image denoising," *Journal of medical systems*, vol. 43, no. 8, pp. 1–10, 2019.

[16] K. Wei, Y. Fu, and H. Huang, "3-d quasi-recurrent neural network for hyperspectral image denoising," *IEEE transactions on neural networks and learning systems*, 2020.

[17] I. Goodfellow, Y. Bengio, and A.Courville, *Deep Learning*. MIT Press, 2016.

[18] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," *IEEE Transactions on Image Processing*, vol. 26, no. 7, pp. 3142–3155, 2017.

[19] Y. Gandelsman, A. Shocher, and M. Irani, "Double-DIP: Unsupervised image decomposition via coupled deep-image-priors," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 018–11 027.

[20] X. Mao, C. Shen, and Y. Yang, "Image restoration using convolutional auto-encoders with symmetric skip connections," *arXiv preprint arXiv:1606.08921*, 2016.

[21] V. Jain and S. Seung, "Natural image denoising with convolutional networks," in *Advances in Neural Information Processing Systems 21*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, Eds. Curran Associates, Inc., 2009, pp. 769–776.

[22] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-d transform-domain collaborative filtering," *IEEE Transactions on Image Processing*, vol. 16, no. 8, pp. 2080–2095, 2007.

[23] Q. Yang, P. Yan, M. Kalra, and G. Wang, "Ct image denoising with perceptive deep neural networks," *ArXiv*, vol. abs/1702.07019, 2017.

[24] J. Yoon, J. Jordon, and M. V. D. Schaar, "ASAC: Active sensing using actor-critic models," in *Machine Learning for Healthcare Conference*, 2019.

[25] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, "Activation functions: Comparison of trends in practice and research for deep learning," *ArXiv*, vol. abs/1811.03378, 2018.

[26] A. Krizhevsky, "Learning multiple layers of features from tiny images," University of Toronto, Tech. Rep., 2009.

[27] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.