Experimental optical retrieval of the Thermal Boundary Resistance of carbon nanotubes in water

IMAGE DENOISING WITH GRAPH-CONVOLUTIONAL NEURAL NETWORKS

Diego Valsesia, Giulia Fracastoro, Enrico Magli

Department of Electronics and Telecommunications – Politecnico di Torino, Italy

ABSTRACT

Recovering an image from a noisy observation is a key problem in signal processing. Recently, it has been shown that data-driven approaches employing convolutional neural networks can outperform classical model-based techniques, because they can capture more powerful and discriminative features. However, since these methods are based on convolutional operations, they are only capable of exploiting local similarities without taking into account non-local self-similarities. In this paper we propose a convolutional neural network that employs graph-convolutional layers in order to exploit both local and non-local similarities. The graph-convolutional layers dynamically construct neighborhoods in the feature space to detect latent correlations in the feature maps produced by the hidden layers. The experimental results show that the proposed architecture outperforms classical convolutional neural networks for the denoising task.

Index Terms— Image denoising, Convolutional Neural Networks, Graph convolution

1. INTRODUCTION

Image denoising is a staple of the research on inverse problems, aiming to restore a clean image from a noisy observation, usually corrupted by additive white Gaussian noise. The key to the solution of this problem is to exploit prior knowledge about the structure of a natural image. The extensive literature on the topic has traditionally focused on developing increasingly refined hand-crafted models for natural images. Popular model-based algorithms include sparse representations [1], total variation methods [2, 3], and methods based on non-local self-similarities such as BM3D [4], or WNNM [5], which are among the most successful ones. However, recent work [6, 7, 8, 9] has shown that a data-driven approach, whereby a convolutional neural network (CNN) is trained for the denoising task, can capture highly complex image priors without the need to handcraft them outperforming the best model-based algorithms. CNN-based approaches stack multiple convolutional layers interleaved by nonlinearities to create hierarchies of feature extractors. However, the main limitation is the local nature of the convolution operation, which is unable to increase the receptive field of a neuron-pixel to model non-local image features. This means that CNNs are unable to exploit the self-similar patterns that were proven to be highly successful in model-based methods.

This paper presents a novel convolutional modeling by defining a new layer exploiting graph convolution, a generalization of the traditional convolution operation to deal with data that may not lie on regular grids, drawing from ideas in graph signal processing [10, 11]. The proposed graph-convolutional layer allows to extract features that depend both on spatially-adjacent pixels, but also on spatially-distant pixels which nevertheless exhibit latent correlations by being close in the feature space. Notice that this approach exploits non-locality in a different manner with respect to algorithms based on global block matching (e.g., BM3D) because it builds a nonlinear hierarchy of filters with a non-local receptive field. While block-matching has a handcrafted notion of similarity (e.g., Euclidean distance between blocks in the noisy image), the proposed technique evaluates the receptive field from similarities in the latent space of the features constructed by the hidden layers of the network, thus greatly improving its expressive power. As a final remark, our method can also be used in conjunction with a preprocessing stage based on wavelets [12] or inside an iterative optimization scheme [13], which have been shown to improve denoising performance. In this paper, however, the core of a denoiser based on a graph-convolutional network is designed, while pre- or post-processing methods are left for future work.

2. BACKGROUND

2.1. Graph convolution

In the proposed model, we employ the Edge Conditioned Convolution (ECC) presented in [14]. This operation is defined using a spatial approach by performing weighted aggregations over a neighborhood. Let us consider a layer l with N^l feature vectors of dimensionality d^l and the corresponding graph $\mathcal{G}^l(\mathcal{V}^l, \mathcal{E}^l)$ where $\mathcal{V}^l$ is the set of vertices with $|\mathcal{V}^l| = N^l$ and $\mathcal{E} \subseteq \mathcal{V}^l \times \mathcal{V}^l$ is the set of edges. We assume that the edges of the graph are labeled, i.e. there exists a function $\mathcal{L} : \mathcal{E} \rightarrow \mathbb{R}^s$ that assigns a label to each edge. In the following, we define the edge labeling function as the difference between the features of the two nodes of the edge, i.e. $\mathcal{L}(i,j) = \mathbf{H}^l_j - \mathbf{H}^l_i$. For each node i of the graph $\mathcal{G}^l$, the convolution performs a weighted local aggre-

This research has been partially funded by the Smart-Data@PoliTO center for Big Data and Machine Learning technologies. We thank Nvidia for donating a Quadro P6000 GPU.

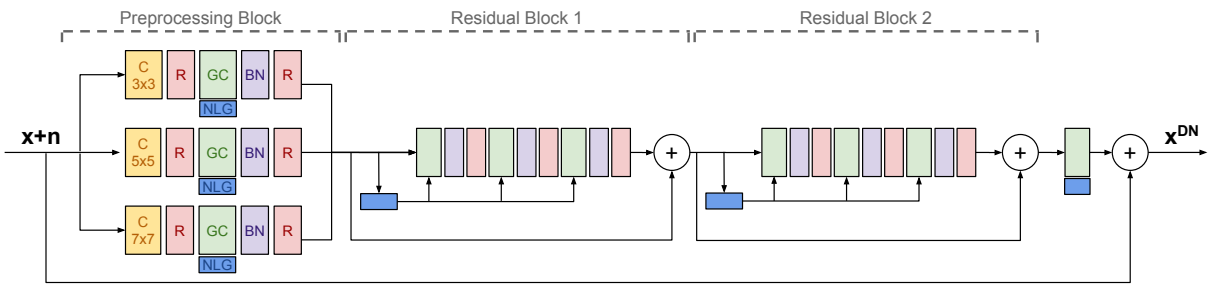

Fig. 1: GraphCNN denoiser. Colored blocks are defined in the leftmost part. C: 2D convolution, R: leaky ReLU, GC: graph convolution (as in Fig.2), NLG: non-local graph construction (nearest neighbors in feature space), BN: batch normalization.

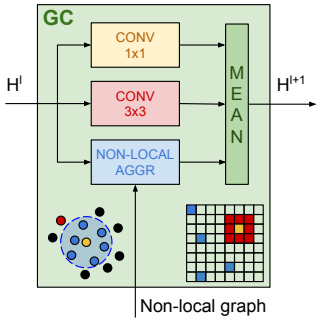

Fig. 2: Graph-convolutional layer. Non-local aggregation block implements Eq. (1).

gation of the feature vectors $\mathbf{H}_j^l \in \mathbb{R}^{d^l}$ on the neighboring nodes $j \in \mathcal{N}_i^l$, where $\mathcal{N}_i^l$ is the neighborhood of node i. The weights of the local aggregation are defined by a fully-connected network $F^l : \mathbb{R}^{d^l} \to \mathbb{R}^{d^{l+1} \times d^l}$, which takes as input the edge labels and outputs the corresponding weight matrix $\mathbf{\Theta}^{l,ji} = F_{\mathbf{w}^l}^l(\mathcal{L}(i,j)) \in \mathbb{R}^{d^{l+1} \times d^l}$. Hence, the convolution operation is defined as:

$$\mathbf{H}_i^{l+1} = \sigma\left(\sum_{j \in \mathcal{N}_i^l} \frac{F_{\mathbf{w}^l}^l\left(\mathbf{H}_j^l - \mathbf{H}_i^l\right)\mathbf{H}_j^l}{|\mathcal{N}_i^l|} + \mathbf{W}^l\mathbf{H}_i^l + \mathbf{b}^l\right)$$

$$= \sigma\left(\underbrace{\sum_{j \in \mathcal{N}_i^l} \frac{\mathbf{\Theta}^{l,ji}\mathbf{H}_j^l}{|\mathcal{N}_i^l|}}_{\text{neighborhood}} + \underbrace{\mathbf{W}^l\mathbf{H}_i^l}_{\text{node}} + \mathbf{b}^l\right), \quad (1)$$

where $\mathbf{w}^l$ are the weights parameterizing network F^l, $\mathbf{W}^l \in \mathbb{R}^{d^{l+1} \times d^l}$ is a linear transformation of the node itself, $\mathbf{b}^l$ a bias, and σ a non-linearity. It is important to note that the filter weights depend only on the edge labels. Therefore, two pairs of nodes with the same labels will have the same weigths. This behaviour is similar to weight sharing in classical CNNs.

2.2. Image denoising with CNN

In the last years, several data-driven image denoising methods have been developed [15, 7, 9, 16, 6]. Many of these methods employ convolutional neural networks (CNN) in order to learn more powerful and descriptive features [6, 16, 7]. In this case, it has been shown that, rather than defining a CNN that outputs directly the denoised image, it is more effective to build a CNN that predicts the residual between the noisy observation and the clean image [6]. Using this approach, the CNN learns to progressively remove the clean image from the noisy observation. It is a common practice to train a CNN model for a fixed noise variance, even though works exist addressing the blind problem [6, 17, 9].

3. PROPOSED METHOD

In this section, we present the proposed architecture, called GraphCNN, to perform image denoising. An overview of the network is shown in Fig. 1. At a high-level, the network is composed of a sequence of graph-convolutional layers followed by batch normalization and leaky ReLU nonlinearities. The first part of the network is composed of a preprocessing block with parallel branches, reminiscent of similar constructions in [18] and [16], whose goal is to extract features at multiple scales, by performing a classic convolution with three different filter sizes (3×3, 5×5, 7×7) followed by a graph convolution operation and finally concatenating the resulting features. The network also has several residual connections: notably, the input-output residual has been shown to be very effective for the denoising task [6] because it makes the network estimate noise features by progressively removing the clean image. The connection between the input and output of each residual block also improves gradient backpropagation.

3.1. Graph-convolutional layer

The graph-convolutional layer is at the core of the proposed model. A schematic representation is shown in Fig. 2. This layer extends the classical convolutional layer by aggregating the hidden-layer feature vectors of spatially-adjacent pixels as well as the hidden-layer feature vectors of spatially-distant pixels that are similar (nearest neighbors) in the feature space. The local features are aggregated using a classic 3×3 convolution while the non-local features are aggregated using the edge-conditioned graph convolution as defined in (1). The local and non-local contributions are then averaged to produce the output features. The non-local pixels are chosen as the k-nearest-neighbor feature vectors in terms of Euclidean distance with respect to the feature vector of the current pixel

Table 1: Set12 PSNR (dB)

	Cman	House	Peppers	Starfish	Monarch	Airplane	Parrot	Lena	Barbara	Boat	Man	Couple	Avg
$\sigma = 15$													
BM3D [4]	31.91	34.93	32.69	31.14	31.85	31.07	31.37	34.26	33.10	32.13	31.92	32.10	32.372
WNNM [5]	32.17	**35.13**	32.99	31.82	32.71	31.39	31.62	34.27	**33.60**	32.27	32.11	32.17	32.696
OGLR [3]	31.36	34.88	32.31	30.70	31.26	30.46	30.87	33.97	32.54	31.58	31.59	31.71	31.936
DnCNN-S [6]	**32.61**	34.97	**33.30**	32.20	33.09	31.70	31.83	**34.62**	32.64	**32.42**	**32.46**	**32.47**	32.859
GraphCNN	32.58	**35.13**	33.27	**32.42**	**33.25**	**31.84**	**31.89**	34.57	32.84	32.41	32.42	32.40	**32.917**
$\sigma = 25$													
BM3D [4]	29.45	32.85	30.16	28.56	29.25	28.42	28.93	32.07	30.71	29.90	29.61	29.71	29.969
WNNM [5]	29.64	**33.22**	30.42	29.03	29.84	28.69	29.15	32.24	**31.24**	30.03	29.76	29.82	30.257
OGLR [3]	29.11	32.65	30.02	28.29	29.16	28.10	28.76	31.95	30.35	29.59	29.47	29.49	29.744
DnCNN-S [6]	**30.18**	33.06	**30.87**	29.41	30.28	29.13	29.43	**32.44**	30.00	**30.21**	**30.10**	**30.12**	30.436
GraphCNN	30.12	**33.22**	**30.87**	**29.76**	**30.51**	**29.30**	**29.48**	32.42	30.28	30.17	**30.10**	29.99	**30.516**
$\sigma = 50$													
BM3D [4]	26.13	29.69	26.68	25.04	25.82	25.10	25.90	29.05	27.22	26.78	26.81	26.46	26.722
WNNM [5]	26.45	**30.33**	26.95	25.44	26.32	25.42	26.14	29.25	**27.79**	26.97	26.94	26.64	27.052
OGLR [3]	25.98	29.19	26.26	24.75	25.80	25.05	25.80	28.80	27.04	26.53	26.69	26.34	26.520
DnCNN-S [6]	**27.03**	30.00	27.32	25.70	26.78	25.87	**26.48**	**29.39**	26.22	**27.20**	**27.24**	**26.90**	27.178
GraphCNN	27.00	30.16	**27.40**	**25.92**	**26.89**	**25.93**	26.43	29.32	26.56	27.05	27.19	26.75	**27.217**

within a search window of predefined size. Notice that the non-local selection is performed only at some hidden layers (as shown by the NLG block in Fig. 1), with the two 3-layer residual blocks sharing the same non-local graph. This helps reducing complexity without compromising performance.

The role of the non-local graph in such residual architecture, whose goal is to successively remove the clean image from the noise features, is to identify the latent correlations in the feature space which are due to the residual image content rather than the uncorrelated noise.

We now analyse more in detail the role of the graph-convolution operation, implemented using the ECC of Eq. (1). Such definition provides two main contributions to the effectiveness of the algorithm but it is also a source of issues. The key to understanding it is the function F, which takes as input the difference between the feature vector of the current pixel and the feature vector of a non-local neighboring pixel and outputs the weight matrix used to transform the non-local feature vector before the aggregation. First, ECC can be called "convolution" because this function provides meaningful weight sharing under suitable stability assumptions: for a similar input difference, the output weight matrix should be similar. Second, differently from classical convolution this function enables a data-dependent aggregation because the weights depend directly on the relationships among feature vectors. The function F was originally proposed [14] to be implemented as fully-connected neural network with 2 layers. However, this leads to over-parameterization problems which negatively impact training. In order to understand this, let us focus on the last layer of the F-network: this layer is a dense layer with a d^l-dimensional input and a $d^{l+1} \times d^l$-dimensional output, meaning that the number of weights depends cubically on the number of features. This quickly leads to an excessively large number of parameters, causing vanishing gradients or overfitting. In order to reduce it, in this paper we propose to use a partially-structured matrix for the last layer of the F-network: instead of an unstructured matrix, we stack multiple row-subsampled circulant matrices. The effective number of parameters depends on i) how many subsampled matrices are stacked; ii) the row-subsampling factor of each of them. As an example, in our experiments we use $d^l = d^{l+1} = 66$ and an unstructured matrix would require $66^3 = 287496$ parameters, while we stack $\frac{66^2}{3}$ partial circulant matrices with 3 rows each for a total of $66\frac{66^2}{3} = 95,832$. Tradeoffs are possible by controlling how much structure one wants to enforce. Similar approaches to approximate fully connected layers have been studied in the literature [19, 20].

4. RESULTS

In this section we perform an experimental evaluation of the proposed method, comparing the proposed method with several state-of-the-art denoising methods. We consider two model-based methods that exploit non-local priors (i.e., BM3D [4] and WNNM [5]), a graph-based variational method (i.e., OGLR [3]) and a state-of-the-art CNN model for image denoising (i.e., DnCNN [6]).

4.1. Experimental settings

In the following experiments, we consider grayscale images. For training, we use the 432 training images of the BSD500 dataset [21]. Instead, for testing we use a set of 12 widely used images. We train the network using a fixed noise standard deviation, considering three different noise levels $\sigma = 15, 25, 50$. We subdivide the images into patches of size 32×32 and train the network on 200k patches for 30 epochs. The non-local graph selects the 8 nearest neighboring pixels

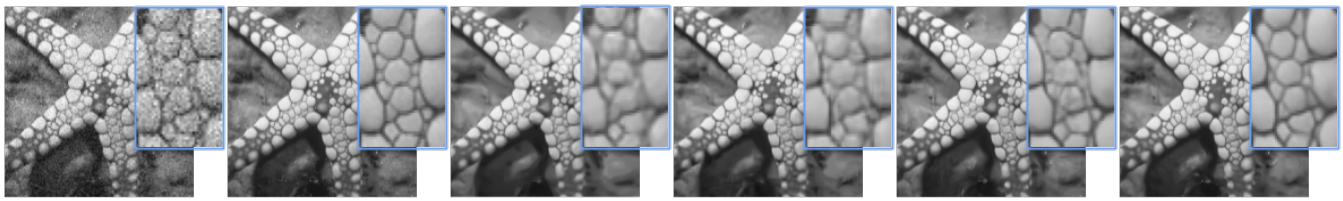

Fig. 3: Denoising results for *starfish*. Left to right: noisy ($\sigma = 25$), original, OGLR (28.29 dB), BM3D (28.56 dB), DnCNN-S (29.41 dB), GraphCNN (**29.76 dB**).

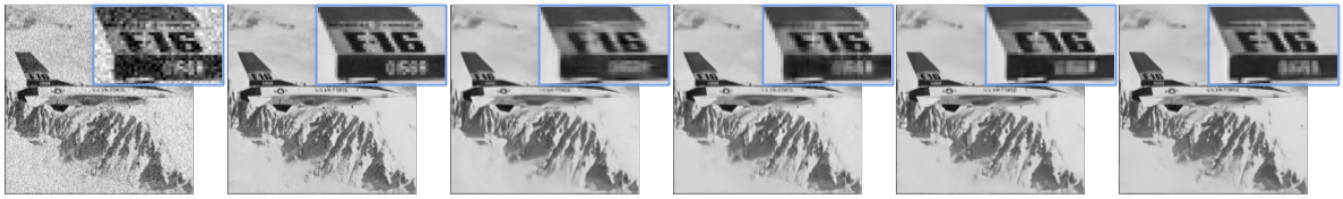

Fig. 4: Denoising results for *airplane*. Left to right: noisy ($\sigma = 25$), original, OGLR (28.10 dB), BM3D (28.42 dB), DnCNN-S (29.13 dB), GraphCNN (**29.30 dB**).

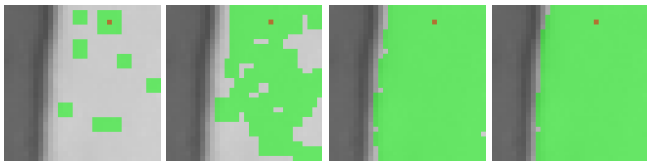

Fig. 5: Receptive field (green) of a single pixel (red) for graph-convolutional layers 1-4 (ignoring the 5×5 and 7×7 multiscale branches for ease of representation).

in terms of Euclidean distance between the hidden feature vectors, excluding the spatially adjacent pixels. The number of hidden features is 66 for all layers, except for the ones in the branches of the preprocessing block, for which is 22.

4.2. Quantitative and qualitative results

Table 1 shows the PSNR results on the 12 images of the test set. We can see that the proposed architecture outperforms the competing methods on most of the images and it provides the best average scores. In order to highlight the importance of the non-local filters, Table 2 compares the proposed method with a network having the same architecture of the proposed model but employing only local neighbors instead of local and 8 non-local. The results show that the nonlocal model is indeed key to achieving the best possible performance. Notice that the 0-NN GraphCNN still falls short of the DnCNN-S model, which suggests that there is still room for improvement (e.g., adding more layers), leading to further gains also for the 8-NN GraphCNN. In addition to these quantitative results, we also show a qualitative comparison of the denoising methods in Figs. 3 and 4. We can clearly see that the proposed method provides the best visual quality, recovering finer details and producing fewer artifacts. Lastly, we show in Fig. 5 the receptive field of a pixel for the first four graph-convolutional layers. It is interesting to note that the receptive field is adapted to the image characteristics, covering only a homogeneous region of the image.

Table 2: Set12 average PSNR (dB) without non-local NN

	$\sigma = 15$	$\sigma = 25$	$\sigma = 50$
DnCNN-S	32.859	30.436	27.178
GraphCNN (0-NN)	32.757	30.348	27.048
GraphCNN (8-NN)	32.917	30.516	27.217

Table 3: Set12 average PSNR (dB) with NN3D

σ	DnCNN-S	DnCNN-S+NN3D	GraphCNN	GraphCNN+NN3D
25	30.436	30.448	30.516	**30.517**
50	27.178	27.215	27.217	**27.258**

4.3. Comparison with other non-local approaches

An alternative approach, called NN3D, to include non-local information in CNNs for the denoising task has recently been proposed by Cruz et al. [22] by using a global post-processing stage based on a non-local filter after the output of a denoising CNN. This stage performs block matching and filtering over the whole image denoised by the CNN. The graph-convolutional method we presented in this paper exploits non-locality in a different way by constructing hierarchical non-local filters. Due to this, the method by Cruz et al. could also be used as a post-processor to our network to further increase the performance, as shown in Table 3 (a single iteration of the NN3D method is used).

5. CONCLUSIONS AND FUTURE WORK

In this paper we proposed a novel architecture of a convolutional neural network for image denoising that leverages graph-convolutional layers in order to create a hierarchy of non-local filters. Experimental results showed improved performance in terms of PSNR and visual quality. We can clearly attribute the gains to the non-local filters. Future work will focus on further improving the performance of the architecture. In particular, we will address the over-parameterization issues of graph convolution, which may unlock further gains.

6. REFERENCES

[1] Michael Elad and Michal Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Transactions on Image Processing*, vol. 15, no. 12, pp. 3736–3745, Dec 2006.

[2] Leonid I. Rudin, Stanley Osher, and Emad Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: nonlinear phenomena*, vol. 60, no. 1-4, pp. 259–268, 1992.

[3] Jiahao Pang and Gene Cheung, "Graph Laplacian regularization for image denoising: analysis in the continuous domain," *IEEE Transactions on Image Processing*, vol. 26, no. 4, pp. 1770–1785, 2017.

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

[5] Shuhang Gu, Lei Zhang, Wangmeng Zuo, and Xiangchu Feng, "Weighted nuclear norm minimization with application to image denoising," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 2862–2869.

[6] 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*, vol. 26, no. 7, pp. 3142–3155, 2017.

[7] Stamatios Lefkimmiatis, "Universal denoising networks: a novel CNN architecture for image denoising," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3204–3213.

[8] Xiaojiao Mao, Chunhua Shen, and Yu-Bin Yang, "Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections," in *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds., pp. 2802–2810. Curran Associates, Inc., 2016.

[9] Jaakko Lehtinen, Jacob Munkberg, Jon Hasselgren, Samuli Laine, Tero Karras, Miika Aittala, and Timo Aila, "Noise2Noise: learning image restoration without clean data," in *International Conference on Machine Learning (ICML)*, 2018.

[10] David Shuman, Sunil Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Processing Magazine*, vol. 3, no. 30, pp. 83–98, 2013.

[11] Diego Valsesia, Giulia Fracastoro, and Enrico Magli, "Sampling of graph signals via randomized local aggregations," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 5, no. 2, pp. 348–359, June 2019.

[12] Woong Bae, Jae Jun Yoo, and Jong Chul Ye, "Beyond deep residual learning for image restoration: Persistent homology-guided manifold simplification," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, July 2017, pp. 1141–1149.

[13] Weisheng Dong, Peiyao Wang, Wotao Yin, Guangming Shi, Fangfang Wu, and Xiaotong Lu, "Denoising prior driven deep neural network for image restoration," *arXiv preprint arXiv:1801.06756*, 2018.

[14] Martin Simonovsky and Nikos Komodakis, "Dynamic edge-conditioned filters in convolutional neural networks on graphs," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017, pp. 29–38.

[15] Harold C. Burger, Christian J. Schuler, and Stefan Harmeling, "Image denoising: Can plain neural networks compete with BM3D?," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2012, pp. 2392–2399.

[16] Nithish Divakar and R. Venkatesh Babu, "Image denoising via CNNs: an adversarial approach," in *New Trends in Image Restoration and Enhancement, CVPR*, 2017.

[17] Jingwen Chen, Jiawei Chen, Hongyang Chao, and Ming Yang, "Image blind denoising with generative adversarial network based noise modeling," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3155–3164.

[18] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich, "Going deeper with convolutions," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 1–9.

[19] J. Wu, "Compression of fully-connected layer in neural network by kronecker product," in *2016 Eighth International Conference on Advanced Computational Intelligence (ICACI)*, Feb 2016, pp. 173–179.

[20] Yu Cheng, Felix X. Yu, Rogerio S. Feris, Sanjiv Kumar, Alok Choudhary, and Shi-Fu Chang, "An exploration of parameter redundancy in deep networks with circulant projections," in *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[21] Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik, "Contour detection and hierarchical image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 5, pp. 898–916, 2011.

[22] Cristovao Cruz, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian, "Nonlocality-reinforced convolutional neural networks for image denoising," *IEEE Signal Processing Letters*, vol. 25, no. 8, pp. 1216–1220, Aug 2018.