Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Convolutional neural network with median layers for denoising salt-and-pepper contaminations

Luming Liang [a], Seng Deng [b], Lionel Gueguen [c], Mingqiang Wei [b,*], Xinming Wu [d], Jing Qin [e]

[a] Microsoft Applied Sciences Group, Redmond, WA 98052, United States
[b] School of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China
[c] Amazon, Denver, United States
[d] School of Earth and Space Sciences, University of Science and Technology of China, Hefei, China
[e] Center for Smart Health, School of Nursing, The Hong Kong Polytechnic University, Hong Kong

ARTICLE INFO

Article history:
Received 18 August 2019
Revised 8 August 2020
Accepted 6 February 2021
Available online 2 March 2021
Communicated by Zidong Wang

Keywords:
Median layer
Deep neural network
Salt-and-pepper noise
Image processing

ABSTRACT

We propose a deep fully convolutional neural network with a new type of layer, named median layer, to restore images contaminated by salt-and-pepper (s&p) noise. A median layer simply performs median filtering on all feature channels. By adding this kind of layer into some widely used fully convolutional deep neural networks, we develop an end-to-end network that removes extremely high-level s&p noise without performing any non-trivial preprocessing tasks. Experiments show that inserting median layers into a simple fully-convolutional network with the L_2 loss significantly boosts signal-to-noise ratio. Quantitative comparisons testify that our network outperforms the state-of-the-art methods with a limited amount of training data.

1. Introduction

Image denoising is well-studied [1–6], where the goal is to recover the underlying signal from its contaminated version. The contaminations can be categorized into many different types according to their distributions and behaviors, e.g. additive (Gaussian) noise, shot (Poisson) noise, JPEG noise and so on. We focus on *salt-and-pepper* (s&p) noise, which is an impulse contamination to the image. In an image with s&p noise, pixels become maximal or minimal values with a predefined probability, which is called the noise level, i.e. the higher this value is, the more pixels will be contaminated. The s&p noise is a special case of random-value impulse noise defined in [5,4]. For a given noise level $p \in (0, 1)$, a salt-and-pepper contaminated image could be defined as

$$I(i,j) = \begin{cases} 0, & r_1 < p \text{ and } r_2 < 0.5 \\ 255, & r_1 < p \text{ and } (r_2 \geqslant 0.5), \\ I(i,j), & r_1 \geqslant p \end{cases} \quad (1)$$

where both r_1 and r_2 are 2 random values generated on each pixel, with the former one determining if a pixel will be contaminated or not and later one controlling if that pixel will turn to be the maximal (salt) value or the minimal (pepper) value. From Eq. 1, one may observe that s&p noise is neither like additive (Gaussian) noise, which can be fully separated from the signal [1]; nor like shot (Poisson) noise, which is signal dependent [5]. It appears as pure noise at the contaminated locations (called missing pixel in [5]) and therefore erases all signals there. This fact prevents us to use any optimization method in a continuous space to recover the signal, since the gradients estimated from missing pixels are totally not reliable and they will further scatter to other unpolluted locations. Traditional ways to recover images from s&p pollution all require nonlinear searches and mappings. The search step [7–9] generally determines the locations of the contaminated pixels and the mapping step tries to give a feasible estimate at each contaminated pixel by weighted averaging the similar pixel values around it. This set of filters are named switching filters. However, when the noise level is increasing, the search step becomes more and more unreliable. On the other hand, the signal estimation step will also be degraded by the high-level noise since the similarity estimation becomes intractable.

To alleviate this limitation, [4] use switching templates to avoid noise disturbance in the process of measuring similarity. Based on the similarities, they extract repairable information in non-local regions instead of local patches. This filter is named as non-local switching filter (NLSF). The method uses a trained Convolutional Neural Network to finally refine the signal recovered by NLSF.

* Corresponding author.
E-mail address: mqwei@nuaa.edu.cn (M. Wei).

https://doi.org/10.1016/j.neucom.2021.02.010
0925-2312/© 2021 Elsevier B.V. All rights reserved.

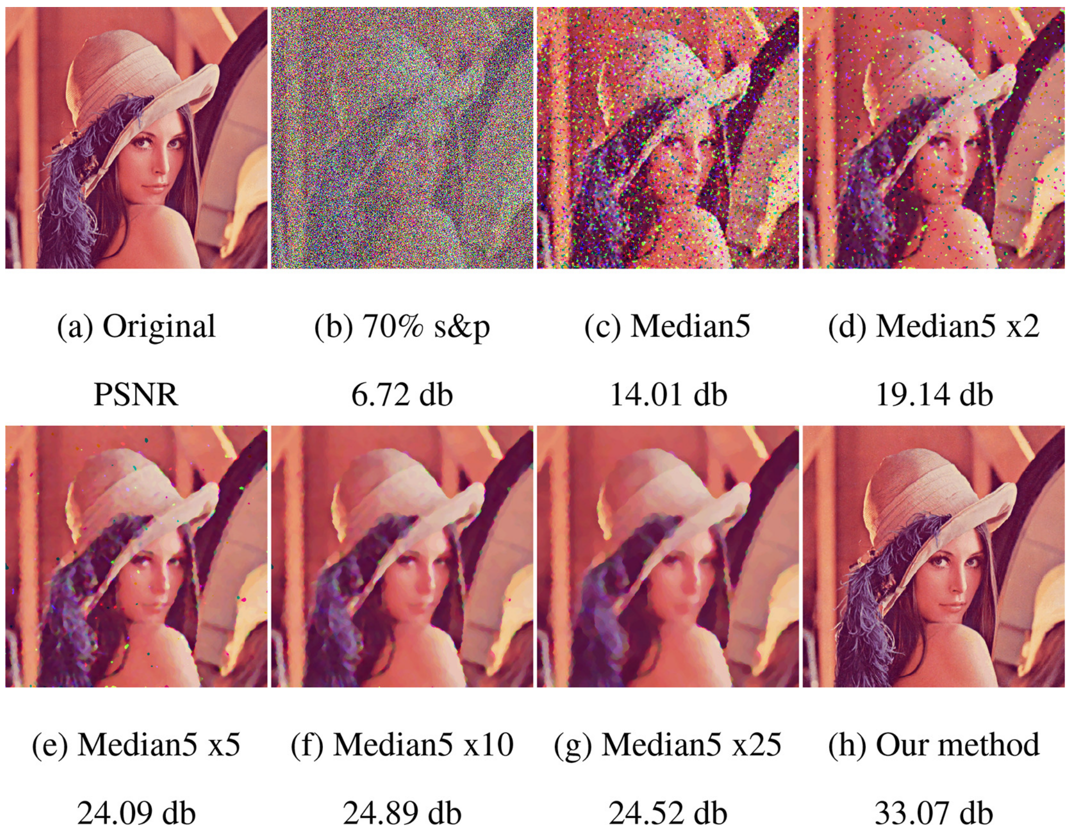


Fig. 1. Classic Lenna image with high-level s&p noise contamination and filtering results with repeated median filter.

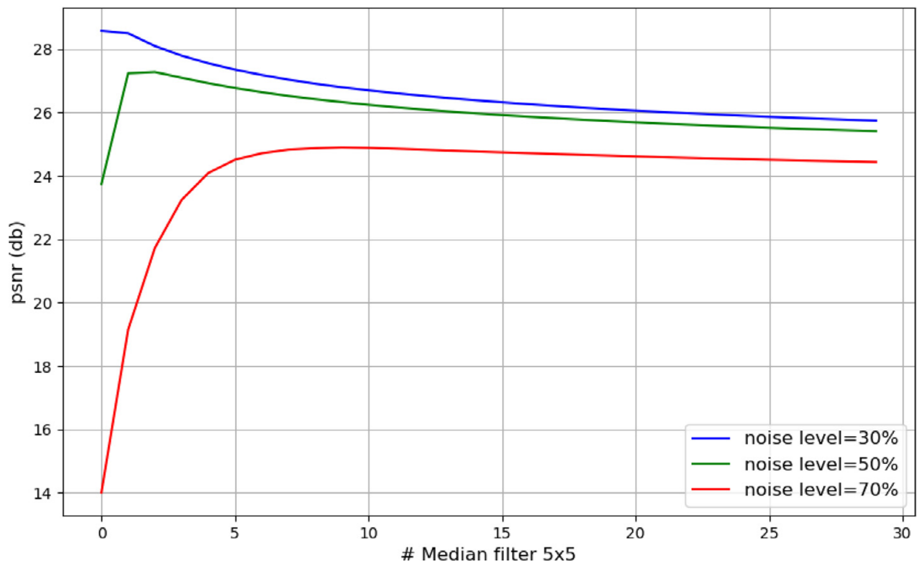
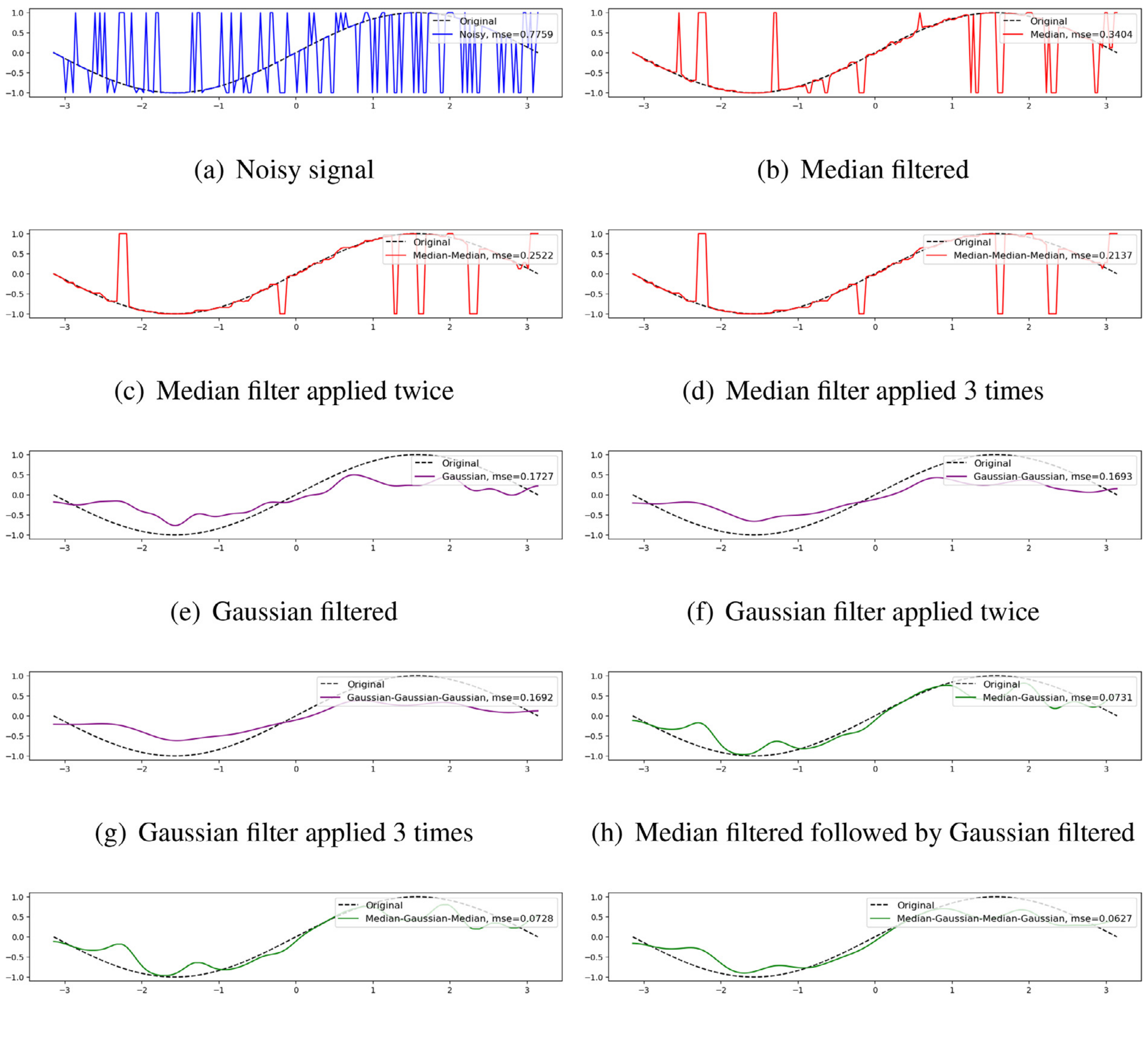


Fig. 2. Peak Signal to Noise Ratio trends with respect to the number of iterations of repeated 5×5 median filters.

Therefore, NLSF is considered as a prepocessing step to the neural network. This method is a combination of traditional methods and the learning-based method.

Besides the usual learning-based ideas that train models to denoise using pairs of clean images and their noisy versions, noise-to-noise method [5] trains models only on noisy images. They discover that training without using clean images can achieve, sometimes even exceeds, the result obtained by training using ground truths. Following this paradigm, [5] shows the ability to remove Random-valued impulse noise, which can be considered as a superset of s&p noise. To deal with the gradient loss problem introduced by pure noisy pixels, they adopt an annealed version of the "L_0 loss" to replace the traditional L_2 loss. The loss function is gradually changing from L_2 to L_0 as the training progress. However, the speed of this annealing procedure must be carefully chosen (usually reducing the power of the norm on the loss function according to the number of iterations). When the prediction is far from the truth, the loss function must be closer to L_2; when



(a) Noisy signal (b) Median filtered

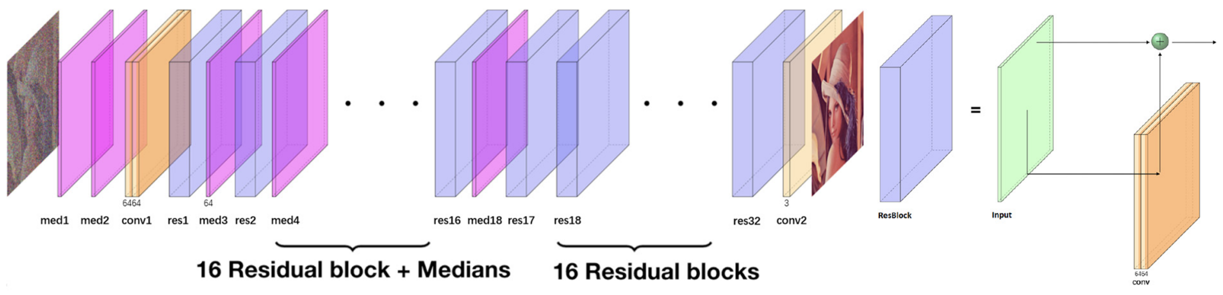
(c) Median filter applied twice (d) Median filter applied 3 times

(e) Gaussian filtered (f) Gaussian filter applied twice

(g) Gaussian filter applied 3 times (h) Median filtered followed by Gaussian filtered

(i) filters applied: Median, Gaussian and Median (j) Median, Gaussian, Median and Gaussian

Fig. 3. 1D signal denoising example using median filters and gaussian filters.



(a) Fully convolutional with median layers in between residual blocks. (b) Our residual blocks.

Fig. 4. Our network structure.

the prediction is getting closer enough, L_0 becomes more favorable, since L_0 loss emphasis the number of different pixels, which leads the learning process to a detail amendment stage. Some more recent neural network based salt and pepper denoising methods can be found in [10–12].

We introduce the use of local nonlinear search into the neural network without performing any pre-processing and also avoid changing the loss function from L_2 loss to some other losses that are not easy to optimize. We resort to median filter [13], which is the first efficient method to denoise salt and pepper noise. By

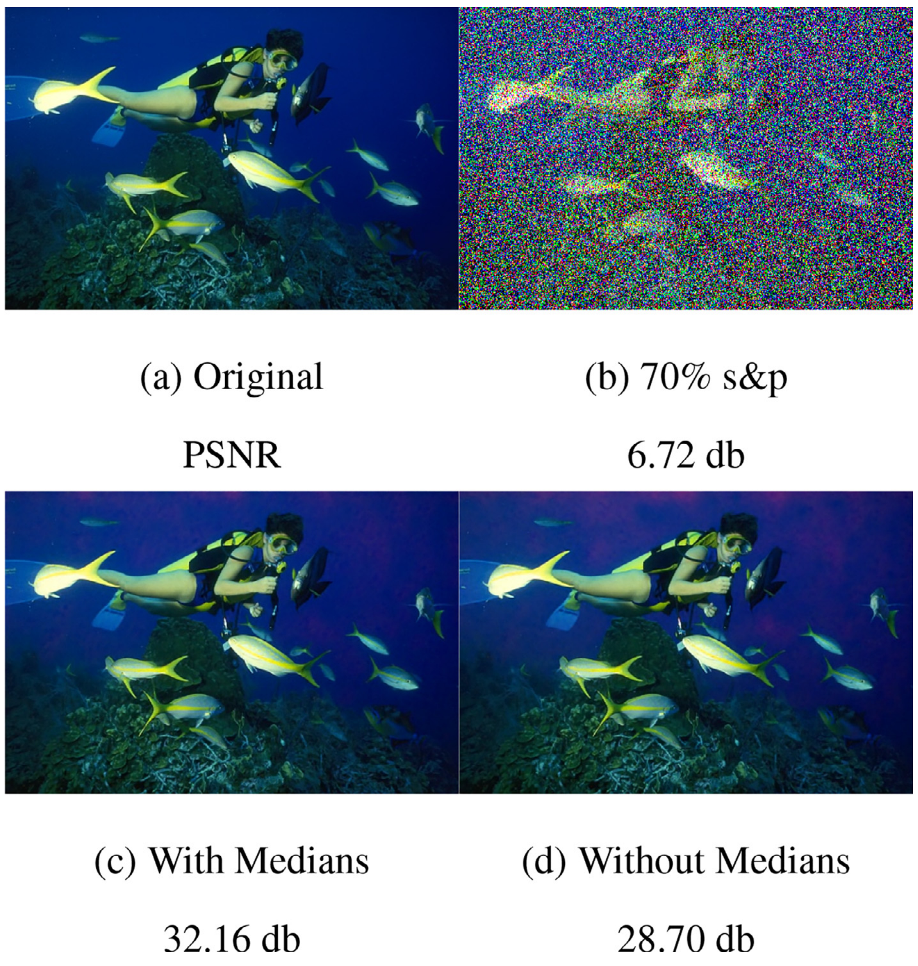


Fig. 5. Denoise results with and without median layers on an image of BSD300.

incorporating median-filter like operations into deep neural networks, our method outperforms state-of-the-art methods. Details of our methodology as well as the model design can be found in Section 2, evaluations are presented in Section 3. Section 4 is for conclusion of our work. We release our source code, training dataset and pretrained models at https://github.com/llmpass/median Denoise for reproducibility.

2. Methodology

Median filter is a traditional nonlinear filter which is especially efficient for removing impulse noise. It replaces the pixel centered in a given window with the median of this window. As shown in Fig. 1, applying median filter on a highly contaminated image (b) removes spikes and therefore greatly improves the signal to noise ratio. Applying a 5×5 median filter once (Fig. 1c) and twice (Fig. 1d) respectively, removes about 50% and 90% noise. A natural idea is to repeatedly apply the median filter upon the image until all spikes are replaced by the median in a fixed-size local window. It does remove the noise, however, it fails to recover the signal. The Peak Signal to Noise Ratio (PSNR) increases in the first several iterations but drops finally as the image becomes blocky and blurry, see Fig. 1g. This phenomena indicates that the median filter deviates the signal too much from its original shape, which is also the main reason why modern researchers abandon median filter in denoising s&p noise.

In addition, the best PSNR value appears at different iterations of repeated median filtering when denoising different levels of noise. Fig. 2 shows the higher density the noise is, the more iterations of median filters are required.

Our basic idea is to keep the ability of spike removal from the traditional median filter but try to recover the degradations intro-

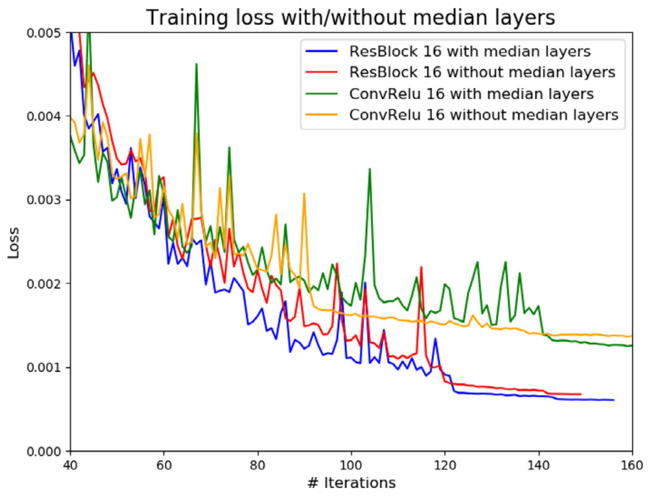


Fig. 6. Training losses with and without median layers.

duced by it. Fig. 3 illustrates a simple 1D synthetic example. We contaminate an evenly-sampled 1D sine function (dotted curves in Fig. 3a) by 50%-level s&p noise. After that, we tried to use different ways to recover the clean signal:

1. Repeated Median filters, see Fig. 3b-d;
2. Repeated Gaussian filters, see Fig. 3e-f;
3. Alternating Median and Gaussian filters, see Fig. 3h-j.

Here, all Median and Gaussian filters have the same window size that equals to 5 pixels. One may observe the third schema yields the best approximations (green curves) to the original sine function, no matter in the aspect of the signal shape or mean square errors (mse) between the smoothed curves (solid) and the true signal (dotted) curve. Using only Median filters creates plateau-like artifacts; using only Gaussian filters over smooth the noisy curves. By alternating Median and Gaussian filters, apparent plateau-like artifacts are washed out while the resulting curve still stays close to the true signal. The quick-dropping mse values between smoothed curves generated by the third schema and the truth quantitatively support our observation.

We leverage these observations to design our deep neural network model for 2d image denoising. We replace the Gaussian filter, which is a fix-parameter smoothing filter, by a set of learnable convolution operations and thus design an end-to-end fully convolutional network with Median and other convolutions alternatingly appearing.

Instead of directly applying median filters on the images, we implement median filtering as a neural network operation and perform it on different feature channels. In this way, we essentially remove spikes in different feature spaces and then combine the de-spiked features to predict a better noise removed image. On one hand, the median filtering in the feature space acts just like the switch filters in the traditional methods [7,4]; on the other hand, the de-spike ability introduced by median operations allow the gradients to pass through the non-noisy pixels.

2.1. Median layer definition

Median filter is applied to each element of a feature channel in a moving window fashion. For example, an input image that consists of RGB channels, corresponds to 3 feature channels; a set of features generated after the convolution generally contains many number of channels. For each feature channel, we first extract a set of given size $(3 \times 3, 5 \times 5, \ldots)$ size patches centered at each pixel. Then, we find the median of the sequence formed by all elements in that patch. We show a pseudo-code implementation of this median layer in Algorithm 1. Here, parameter x is a channel of the input tensor and k denotes an integer kernel size.

Algorithm 1: Median Layer

```
patches := extract $k \times k$ size patches around each pixel
$m_i$ := int($k \times k/2 + 1$)
median := fetch $m_i$ elements from patches
return median
```

Table 2
PSNR (db) comparisons of inserting Median layers into ConvRelu 16 model at different locations on BSD300. First column indicates the locations of the median layers, the second, third and forth column show PSNRs on the same image set over different levels of contaminations.

Location	30%	50%	70%
None	31.15	30.15	28.88
2 median layers inserted before 1st convolution	31.02	30.22	29.27
2 median layers inserted before +1 median layer inserted after first every 8 convolutions	33.82	32.01	29.37
2 median layers inserted before +1 median layer inserted after every convolutions	35.65	33.51	30.54

In practice, this median layer is applied on each feature channel and then we concatenate them to form a new set of features, e.g. median layer will be applied 64 times given a set of 64 feature channels generated by Convolutions.

2.2. Network architecture

As shown in Fig. 4a, our network is a fully convolutional network, so that no restrictions are posed on the size of the input. It starts with 2 consecutive median layers, which are then followed by a sequence of residual blocks and median layers. The last part of the network is just residual blocks without inserting median layers in between them. In practice, we only insert median layers into the first half of the sequence of residual blocks. The first part of the network is dedicated to remove noise from the image, the second half of the network is designed for recovering the signal (see Fig. 5).

We choose to generate 64 features per convolution layer and our residual block is designed as a skip connection over 2 64-convolutions, followed by batch normalization layers and nonlinear activations (relu in practice), as shown in Fig. 4b.

As mentioned beforehand, we stick to use the simplest L_2 loss as our objective function. This loss is simply defined as the mean square error of the estimated image and the ground truth image, as minimizing mse directly relates to increase of denoise metrics psnr. Details can be found in Eq. 3.

3. Evaluation

We design several experiments to evaluate the properties of median layers (Section 3.2) and performances of the proposed network (Section 3.4).

3.1. Training and testing setup

For fair comparisons, we train all models with the same data set described in [14] that contains 91 different images, which is also employed in other works [4]. Since our network is a fully convolutional network, the input size can be arbitrary. We first resize these 91 images to 200×200 and then we generate 70×70 patches from them as clean images. We degrade each patch by the s&p noise with levels from 10% to 90% with a step equals to 10% as a sequence of noisy images. The models are trained to learn a series of weights in layers that can transfer the input noisy image to the clean image.

Table 1
PSNR (db) comparisons w/o Median layers on BSD300.

Noise level	ConvRelu 16 **o** median	ConvRelu 16 **w** median	ResBlock 16 **o** median	ResBlock 16 **w** median	ResBlock 32 **o** median	ResBlock 32 **w** median
30%	31.15	**33.82**	40.38	**40.89**	36.86	**40.90**
50%	30.15	**32.01**	36.55	**36.93**	36.86	**37.28**
70%	28.88	**29.37**	31.98	**32.23**	32.22	**32.40**

Table 3
PSNR (db) Comparisons with state-of-the-arts on a set of classic images and BSD300 image database.

Image	Noise Level	DBA [16]	NAS NLM [8]	PARIGI [9]	NLSF [4]	NLSF -MLP [17]	NLSF -CNN [4]	Noise2 Noise [5]	Ours
	30%	34.42	28.09	33.90	34.20	30.80	35.38	36.39	**37.04**
	50%	30.11	26.15	29.91	30.12	29.28	32.55	34.68	**35.00**
	70%	25.84	25.97	25.22	25.79	27.63	30.18	32.83	**33.07**
	30%	28.07	23.68	25.19	28.21	25.19	28.71	30.89	**40.46**
	50%	24.24	22.91	22.61	24.45	23.86	26.01	27.96	**34.83**
	70%	21.12	22.63	20.06	21.02	22.61	24.11	25.09	**29.96**
	30%	29.41	20.61	29.74	32.88	29.64	33.47	39.98	**40.65**
	50%	27.47	16.69	27.25	29.66	28.28	30.92	36.13	**38.84**
	70%	24.99	16.32	24.29	26.33	26.90	29.06	30.55	**33.29**
	30%	26.85	22.38	28.88	32.27	30.01	**32.99**	30.70	30.83
	50%	25.27	21.82	25.44	27.99	28.57	**30.23**	29.86	30.07
	70%	22.11	21.58	21.46	23.04	27.04	27.70	28.79	**29.05**
BSD300 Average	30%	29.92	25.74	12.04	30.01	29.77	30.87	39.83	**40.90**
	50%	26.32	24.50	6.01	26.25	26.19	27.84	35.92	**37.28**
	70%	22.81	24.65	5.42	22.85	26.19	25.35	31.42	**32.40**

Table 4
Computational time evaluation of median layers.

Model	Running Time (ms)
ResBlock 16	132
ResBlock 16 Median 3×3	145
ResBlock 16 Median 5×5	146

To quantitatively compare the performance of different methods, we perform denoising on two sets of the images. The first set of image consist of some classical images in the image processing field (also used in [4]), the second set is BSD300 [15].

The metric considered in the comparison is Peak Signal to Noise Ratio (PSNR). It is defined by

$$PSNR = 10\log_{10}\left(\frac{255^2}{MSE}\right), \tag{2}$$

where MSE is the mean-squared error between two $M \times N$ 8-bit images I_1 and I_2, defined by

$$MSE = \frac{\sum_{M,N}[I_1(m,n) - I_2(m,n)]^2}{M \times N}. \tag{3}$$

3.2. Effects of the median layer

The first experiment is designed to show the effectiveness of the median layer. We trained several pairs of fully convolutional networks mainly consisting of residual or convolution-batchNorm-relu blocks, but one with median layers and one without them.

We train two sets of networks: the first set of networks are traditional ones that contain no median layers, the second set of networks are the counterparts of the first set with median layers inserted into them with the same strategy shown in Fig. 4a, i.e. the first half of the network contains median layers, the second half does not.

Losses in Fig. 6 shows how median layers boost the PSNR value of the network. Training losses of two networks with median layers inserted converge to a better minima comparing to the losses without the median layers. The "ConvRelu 16" network in Fig. 6 is a **DnCnn** [1] style network, which consists of 16 stacked Convolution-BatchNormalization-Activation units. The "ResBlock 16" network in Fig. 6 is formed by simply replacing the Convolution-BatchNormalization-Activation units to residual blocks shown in Fig. 4b. All convolution layers here generate 64 features.

PSNR comparisons of models with and without median layers inserted (Table 1) show the improvements of PSNR. The PSNR values of models with median layers are usually 0.5db higher than the ones that do not have them. Note that, shallower networks tend to perform better in lower noise contaminations.

Additional ablation study to reveal the best schema of inserting median layers is shown in Table 2. We insert several median layers with kernel size equals to 5×5 into "ConvRelu 16" network. One can tell all 3 schemas perform better than the model without having the median layers.

3.3. Time consumption

We evaluate the running time of median layers with different sizes in Table 4. The baseline model we choose is "ResBlock 16", which contains 16 residual blocks, with each convolution layer containing 64 features in both input and output. We insert 10 median layers into this model. From Table 4, we can tell this 10 median layers only slow the model by 10%, about 1 ms per layer consumes. In addition, the kernel size does not affect the speed too much. All timing experiments are performed on GPU 1080ti.

3.4. Comparisons to the state-of-the-arts

Both quantitative and qualitative comparisons to the state-of-the-arts are performed in this section.

3.4.1. Quantitative comparisons

We quantitatively compare our network in Fig. 4a to several state-of-the-art methods. Baselines include 5 traditional methods: Decision-Based Algorithm (DBA) [16], Adaptive Switching Non-local Filter (NASNLM) [8], PARIGI [9], NLSF [4] (prepocessing part of NLSF-CNN), NLSF-MLP (NLSF with multi-layer perception proposed in [17]) and 2 most recent neural network based methods: NLSF-CNN [4] and Noise2Noise [5], as shown in Table 3. Many methods here are designed for denoising s&p noise with moderate levels, therefore, we choose to evaluate the methods under noise levels equal to 30%, 50% and 70%.

Our method outperforms most of the state-of-the-arts besides the pepper image. Comparing to current best baseline method Noise2Noise [5], PSNR values achieved by our model is about 1-2db higher in average and the severer the noise contamination, the comparably better our method performs.

3.4.2. Qualitative comparisons on extremely high-level noise

We further qualitatively compare our method to noise2noise method [5] on denoising extremely high-level s&p noise (noise level equals to 90%). In Fig. 7, we choose three images from BSD300 dataset, where different challenges can be found there:

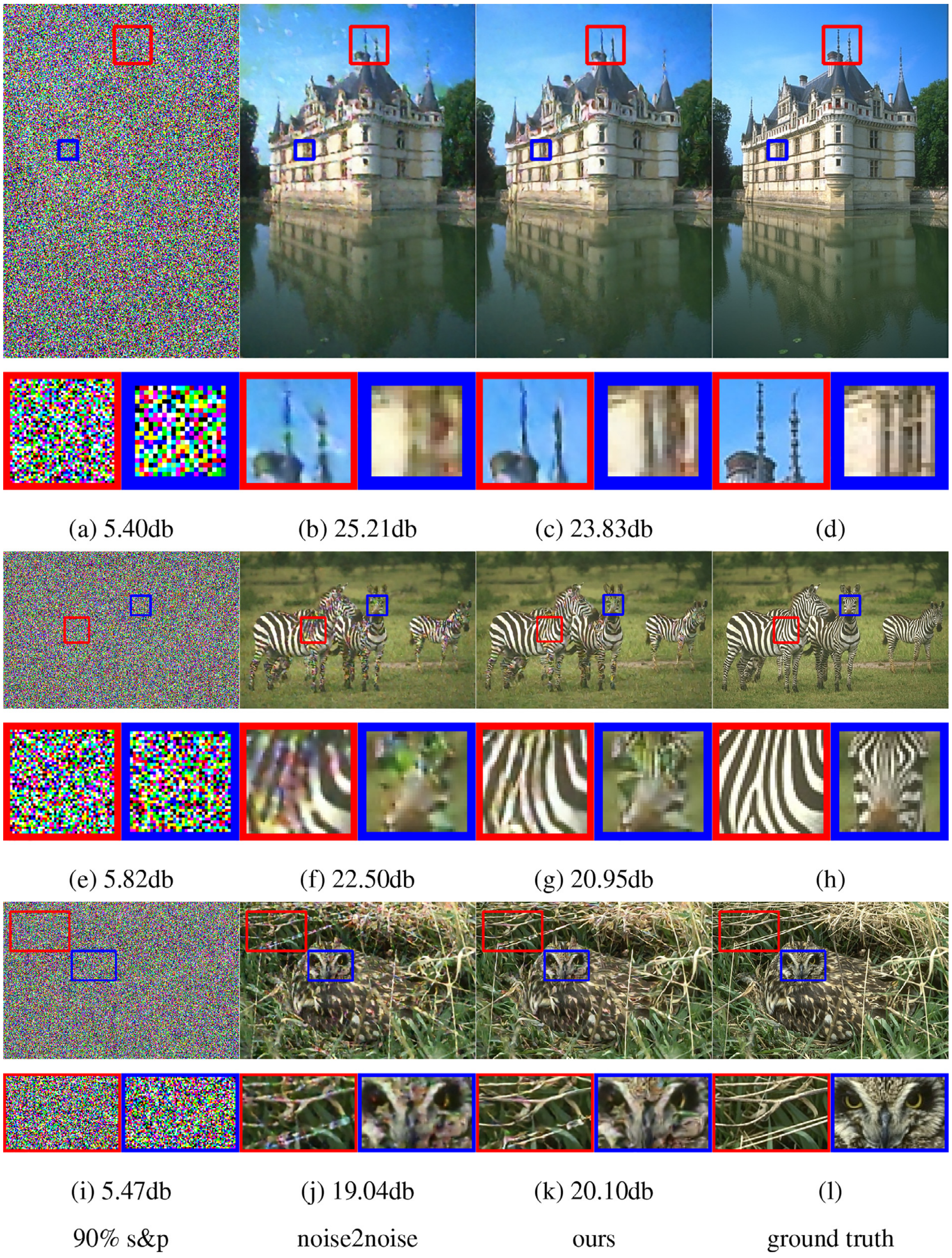


Fig. 7. Detailed comparisons between noise2noise and our model. Our model outperforms noise2noise consistently on different challenges: 1) smoothly changing background (the first row); 2) white and black strips (the second row) and 3) noise-like natural scene.

- both sharp feature and smooth background exist in the first image (Fig. 7d);
- pure black and white interphase pattern in the second image (Fig. 7h);
- noise-like nature scene background (Fig. 7l).

The left-most column in Fig. 7 shows the contaminated images, which are the noisy version of their counterparts in the right-most column. One may hardly see the contours of the original salient objects there, since 90% of pixels become either maximal or minimal values.

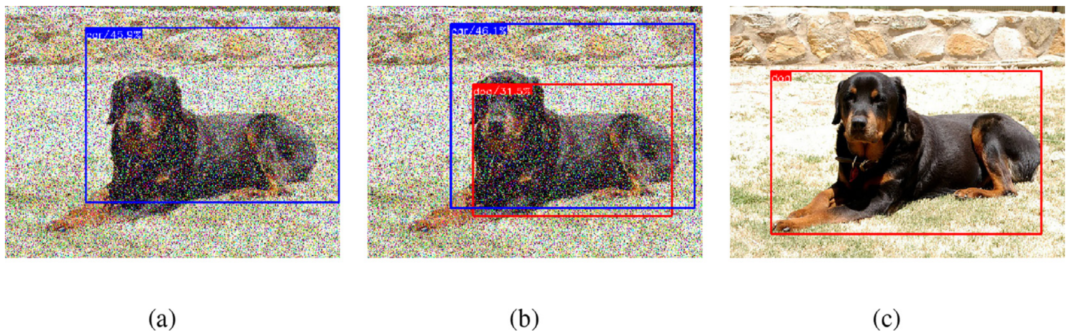


Fig. 8. Detection results of contaminated pascal voc test image: (a) original darknet53; (b) median layers inserted; (c) ground truth label.

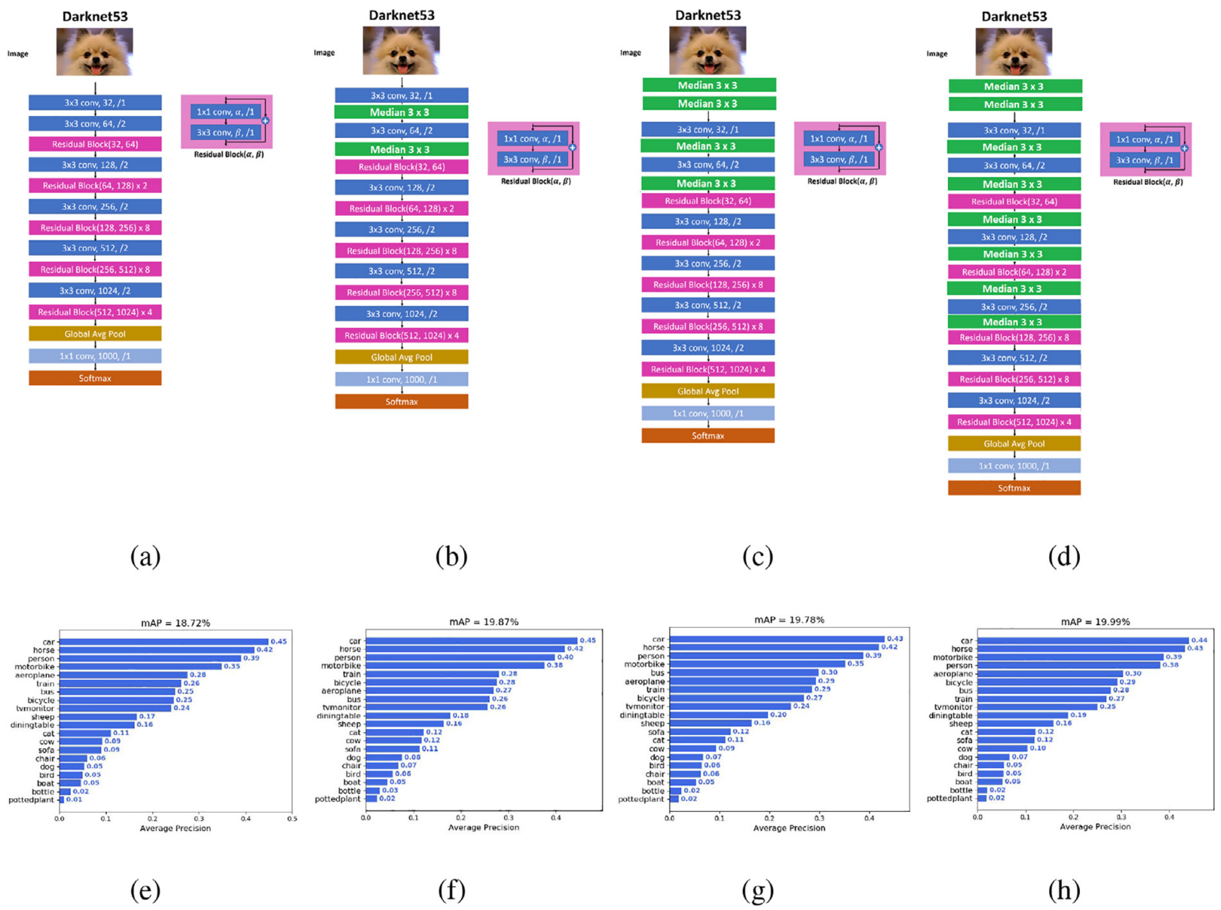


Fig. 9. Median layers inserted to different positions of Darknet53 (a)-(d), yield 1% improvements on mean Average Precision (mAP) (e)-(h).

Our method performs consistently better than noise2noise on all of these challenges. In Fig. 7b, noise2noise generates many small white blocky artifacts on the sky (red rectangle) and also blurs the sharp edges (blue rectangle) of the windows. Both of these 2 degradations are alleviated in our result shown in Fig. 7c.

Recovering underlying signal with pure black and white interphase pattern from high-level s&p noise contamination is a very difficult problem because both signal and noise are almost binary in each channel. The method may have a hard time to distinguish which pixel is contaminated. By comparing the results shown in Fig. 7f (noise2noise) and Fig. 7g (ours), one may observe that our method produces higher quality images.

Noise-like patterns are common to many nature scene images, for example, the grass and the feathers of the owl in Fig. 7l and the leaves in the bridge image in Table 3. We observe that the image still looks noisy after being processed by noise2noise method, where apparent small blue and red dots stand out on the grass (Fig. 7j). However, such dots are invisible in our result, as shown in Fig. 7k.

3.5. Median layer inserted into common detector

The median layer is designed as a plugin to networks that can be used to deal with images contaminated by salt and pepper noise. To better illustrate this theme, we insert median layers into different locations of the backbone network of YoloV3 [18], i.e. darknet53, to deal with the images with s&p noise. We first train our baseline model (original darknet53) using only Pascal Voc 2007 [19] training set and test it on Pascal Voc 2007 test set, as shown in Fig. 8. Then, we randomly contaminate both the train and test image sets with 10% to 70% of s&p noise. Fig. 9 illustrates

3 different schema of median layer placements, inserting in first half of the darknet53 (8 locations) yields the best result.

4. Conclusion

In this paper, we show that incorporating median filtering technique in deep neural network helps achieving compelling results in denoising s&p noise, especially when the noise level is high. The ability of the median layer to denoise is also experimentally testified with increasing PSNR. Our work opens the door in adopting traditional low-level nonlinear signal processing techniques in deep neural networks. The methodology of inserting non-linear spatial layers may boost the performances of some well-known deep networks.

The median is the optimum point of a set of values under L_1 norm, which minimizes the sum of absolute deviations. This fact makes median layers act as a regularizer to the feature channels. Unlike the annealing procedure on the loss function adopted in [5], where the speed of evolving the loss from L_2 to L_0 must be carefully chosen to achieve the best result (with respect to the amount of noises), median layers is a more feasible way to control the quality of the extracted features. A single model can be trained to recover latent images with different levels of noise contaminations only using L_2 loss.

Spatial filtering have been invented and could be leveraged into convolutional neural networks to deal with images affected by non-linear noise. More study on the median placements could result in understanding its impact in the process.

CRediT authorship contribution statement

Luming Liang: Methodology. **Seng Deng:** Methodology. **Lionel Gueguen:** Methodology. **Mingqiang Wei:** Writing - original draft, Supervision. **Xinming Wu:** Visualization, Methodology. **Jing Qin:** Writing - review & editing, Validation.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] K. Zhang, W. Zuo, Y. Chen, D. Meng, L. Zhang, Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising, IEEE Trans. Image Process. 26 (7) (2017) 3142–3155.
[2] K. Zhang, W. Zuo, L. Zhang, Ffdnet: Toward a fast and flexible solution for CNN based image denoising, IEEE Transactions on Image Processing..
[3] D. Liu, B. Wen, X. Liu, Z. Wang, T. Huang, When image denoising meets high-level vision tasks: A deep learning approach, in: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18, International Joint Conferences on Artificial Intelligence Organization, 2018, pp. 842–848. doi:10.24963/ijcai.2018/117. https://doi.org/10.24963/ijcai.2018/117..
[4] B. Fu, X. Zhao, Y. Li, X. Wang, Y. Reng, A convolutional neural networks denoising approach for salt and pepper noise, Multimedia Tools Appl. (2018) 1–18.
[5] J. Lehtinen, J. Munkberg, J. Hasselgren, S. Laine, T. Karras, M. Aittala, T. Aila, Noise2noise: Learning image restoration without clean data, in: International Conference on Machine Learning (ICML) 2018, 2018, pp. 2971–2980..
[6] R. Furuta, N. Inoue, T. Yamasaki, Fully convolutional network with multi-step reinforcement learning for image processing, in, in: AAAI Conference on Artificial Intelligence (AAAI), 2019.
[7] W. Wang, P. Lu, An efficient switching median filter based on local outlier factor, IEEE Signal Process. Lett. 18 (2011) 551–554.
[8] J. Varghese, N. Tairan, S. Subash, Adaptive switching non-local filter for the restoration of salt and pepper impulse-corrupted digital images, Arabian J. Sci. Eng. 40 (2015) 3233–3246.
[9] J. Delon, A. Desolneux, T. Guillemot, Parigi: a patch-based approach to remove impulse-gaussian noise from images, Image Process On Line 5 (2016) 130–154.
[10] L. Jin, W. Zhang, G. Ma, E. Song, Learning deep cnns for impulse noise removal in images, J. Vis. Commun. Image Represent. (2019) 193–205.
[11] J. Chen, F. Li, Denoising convolutional neural network with mask for salt and pepper noise, IET Image Proc. 13 (13) (2019) 2604–2613.
[12] Y. Xing, J. Xu, J. Tan, D. Li, W. Zha, Deep cnn for removal of salt and pepper noise, IET Image Proc. 13 (9) (2019) 1550–1560.
[13] T.S. Huang, G.J. Yang, G.Y. Tang, A fast two-dimensional median filtering algorithm, IEEE Trans. Acoust. Speech Signal Process. 27 (1979) 13–18.
[14] C. Dong, C.C. Loy, K. He, X. Tang, Image super-resolution using deep convolutional networks, IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI) 38 (2015) 295–307.
[15] D. Martin, C. Fowlkes, D. Tal, J. Malik, A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics, in: Proc. 8th Int'l Conf. Computer Vision, Vol. 2, 2001, pp. 416–423..
[16] K.S. Srinivasan, D. Ebenezer, A new fast and efficient decision-based algorithm for removal of high-density impulse noises, IEEE Signal Process. Lett. 14 (2007) 189–192.
[17] H. C. Burger, C. J. Schuler, S. Harmeling, Image denoising: Can plain neural networks compete with bm3d?, in: Proc. 2012 IEEE Conference on Computer Vision and Pattern Recognition, Vol. 157, pp. 2392–2399..
[18] J. Redmon, A. Farhadi, Yolov3: An incremental improvement, CoRR abs/1804.02767. arXiv:1804.02767. URL http://arxiv.org/abs/1804.02767.
[19] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, A. Zisserman, The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results, http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

.

Luming Liang is a senior researcher at Microsoft. Before working at Microsoft, Luming was a software engineer and then a data & applied scientist at Uber and Microsoft, respectively. He received his BSc. and MSc. in the School of Information Science and Engineering from Central South University, China, in 2005 and 2008, respectively and his PhD in the Department of Electrical Engineering and Computer Science from Colorado School of Mines, USA in2014. His primary research interest is finding shape correspondences and image analysis.

Sen Deng is a master candidate at Nanjing University of Aeronautics and Astronautics (NUAA), China. He received his Bachelor's degree in the University of Electronic Science and Technology of China. His research interests include deep learning, image processing and computer vision.

Lionel Gueguen is a Senior Engineer at Uber, CO,USA since 2016, where he conducts research and engineering for information extraction from images. He had been an R&D Scientist with Image Mining Labs of Digital Globe Inc. Lionel received the engineering degree in telecommunications and the M.S. degree in signal and image processing from the Ecole Nationale Superieure des Telecommunications de Bretagne, Brest, France, in 2004 and the Ph.D. degree in signal and image processing from the Ecole Nationale Superieure des Telecommunications, Paris, France, in 2007.

Mingqiang Wei received his Ph.D degree (2014) in Computer Science and Engineering from the Chinese University of Hong Kong (CUHK). He is an associate professor at the School of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics (NUAA). Before joining NUAA, he served as an assistant professor at Hefei University of Technology, and a postdoctoral fellow at CUHK. His research interest is computer graphics with an emphasis on smart geometry processing.

Xinming Wu received the Ph.D. degree in geophysics from the Colorado School of Mines, Golden, CO, USA, in 2016. He was a member of the Center for Wave Phenomena, Colorado School of Mines. He is currently a professor at School of earth and space sciences of University of Science and Technology China (USTC). Post-Doctoral Fellow with The University of Texas at Austin, Austin, TX, USA. His research interests include image processing, 3D seismic interpretation, subsurface modeling, and geophysical inversion. Dr. Wu received the awards for the Best Paper in Geophysics in 2016 and the Best Student Poster Paper presented at the2017 SEG Annual Meeting.

Jing Qin is currently an assistant professor in the Center for Smart Health, School of Nursing, The Hong Kong Polytechnic University. He received his PhD degree in Computer Science and Engineering from the Chinese University of Hong Kong in 2009.Dr Qin received the Champion in the 3rd Hong Kong Innovation Day and Innovation Awards Competition, Medical Image Analysis-MICCAI'17 Best Paper Award, the Best Paper Award in Medical Image Computing in International Conference on Medical Imaging and Augmented Reality 2016, the HongKong Medical and Health Device Industries Association Student Research Award in 2009 and Global Scholarship Program for Research Excellence(CNOOC Grants) from CUHK in 2008. He and his collaborators were nominated for outstanding paper award in International Simulation and Gaming Association 40th Annual Conference in 2009. Dr Qin's research interests include medical image processing, virtual/augmented reality for healthcare and medicine training, deep learning, visualization and human-computer interaction and health informatics.