Fast Image Restoration with Multi-bin Trainable Linear Units

Shuhang Gu[1], Wen Li[1], Luc Van Gool[1,2], Radu Timofte[1]
[1] ETH, Zurich, [2] KU Leuven
{shuhang.gu, liwen, vangool, radu.timofte@vision.ee.ethz.ch}

Abstract

Tremendous advances in image restoration tasks such as denoising and super-resolution have been achieved using neural networks. Such approaches generally employ very deep architectures, large number of parameters, large receptive fields and high nonlinear modeling capacity. In order to obtain efficient and fast image restoration networks one should improve upon the above mentioned requirements. In this paper we propose a novel activation function, the multi-bin trainable linear unit (MTLU), for increasing the nonlinear modeling capacity together with lighter and shallower networks. We validate the proposed fast image restoration networks for image denoising (FDnet) and super-resolution (FSRnet) on standard benchmarks. We achieve large improvements in both memory and runtime over current state-of-the-art for comparable or better PSNR accuracies.

1. Introduction

Image restoration refers to the task of estimating the latent clean image from its degraded observation, it is a classical and fundamental problem in the area of signal processing and computer vision. Recently, deep neural networks (DNNs) have been shown to deliver standout performance on a wide variety of image restoration tasks. With massive training data, very deep models have been trained for pushing the state-of-the-art of different restoration tasks.

The idea of DNN-based image restoration method is straight forward: training a neural network to capture the mapping function between the degraded images and their corresponding high quality images. By stacking the standard convolution layers and Rectified Linear Unit (ReLU) activation functions, the VDSR approach [22] and the DnCNN approach [43] have achieved state-of-the-art performance on the image super-resolution (SR) and image denoising tasks, resp. Furthermore, it is perhaps unsurprising that we are able to further improve the results by these models by further increasing their number of layers; since deeper network structures not only have stronger nonlinear modeling capacity but also incorporate input pixels from a larger area. However, in practice often only limited computational resources are available and how to design an appropriate network structure for such conditions is an important research direction.

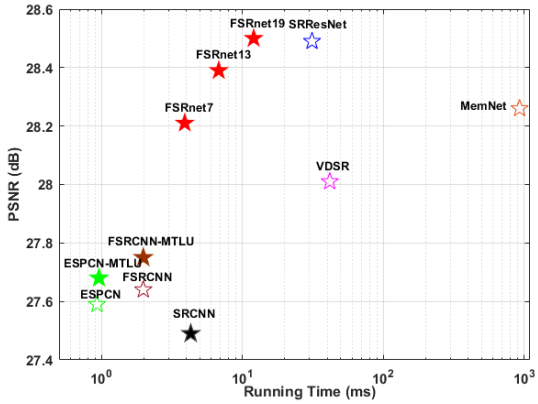


Figure 1. Average PSNR on Set 14 ($\times 4$) ***vs.*** runtime (on Titan X Pascal GPU with Caffe [20]) for processing a 512×512 HR image by different approaches. Our MTLU improves fast SR methods without increasing their runtime. Our FSRnet has a better trade-off between speed and accuracy than state-of-the-art SR methods.

Generally, the key challenge in learning fast image restoration networks is twofold: (i) to incorporate sufficient receptive fields, and (ii) to economically increase the nonlinear modeling capacity of networks. In order to increase the receptive field of the proposed network, we utilize a similar strategy with recent works [36, 45] and conduct all the computations in a lower spatial resolution. While for the purpose of stronger nonlinear capacity, we propose a multi-bin trainable linear unit (MTLU) as the activation function used in our networks. The proposed MTLU parameterizes each activation function with a group of parameters, which can be trained jointly with other network parameters in an end-to-end manner. Concretely, MTLU divides the activation space into equidistant bins and approximates the activation function in each bin with a linear function independently. Such a simple strategy enables us to greatly enhance the nonlinear capacity of each activation function without significantly increase its computational burden.

We evaluated the proposed MTLU on the classical SR

and image denoising tasks. Fig. 1 presents the trade-off between runtime and accuracy for different SR approaches on *Set 14* dataset with scaling factor 4. By replacing the activation functions used in fast SR approaches [36, 10] with MTLU, we are able to improve their accuracy without increasing the runtime. Furthermore, the proposed FSRnet is 0.2 up to 0.5dB better in PSNR terms than the benchmark method VDSR [22] while being 10 to $4\times$ faster, and achieves comparable PSNR with the state-of-the-art SRResNet [25] but $3\times$ faster. For the image denoising task, the proposed method achieves comparable performance with the state-of-the-art approach DnCNN [43] but $15\times$ faster.

Our main contributions are as follows: **1)** This paper is the first work to adopt a highly complex trainable activation function to improve general CNN-based image restoration algorithms. **2)** We proposed the parameterized activation function MTLU and evaluated it on a wide range of network structures. MTLU fits the CNN framework well and consistently outperforms commonly used activation functions (PReLU, ReLU and MaxOut) for solving the image denoising and SR tasks. **3)** Based on MTLU, we proposed the FSRnet and FDnet for the image SR and denoising tasks. Our nets achieve comparable performance with state-of-the-art approaches, at much lower computational demands.

1.1. Related Works

DNN for Image Restoration Due to its unparalleled nonlinear modeling capacity, deep neural networks (DNNs) have been widely applied to different image restoration/enhancement tasks [9, 22, 25, 36, 43, 45, 44]. In this part, we only provide a very brief review of DNN-based image denoising and SR approaches, which are the two tasks investigated in this paper.

To deal with SR problem, DNN-based approaches proposed to train a neural network for buliding the mapping function between low-resolution (LR) and high-resolution (HR) images. Dong *et al*. [9] firstly proposed a deep learning based model for SR – SRCNN. SRCNN [9] achieved comparable performance with then state-of-the-art conventional SR approaches based on sparse representations [42, 15] and anchored neighborhoods [40], triggering the tremendous investigation of DNN-based SR approaches. Kim *et al*. [22] proposed VDSR which utilizes a deeper neural network to estimate the residual map between HR and LR image, and achieved superior SR performance. Recently, Ledig *et al*. [25] built SRResNet, a very deep neural network of residual blocks, obtaining state-of-the-art SR performance. Besides designing deeper networks for pursuing better SR performance, other interesting research directions include investigating of better losses [21, 25] for generating perceptual plausible SR results, extending the generalization capacity of SR network for different kernels settings and faster SR networks [36].

The application of discriminatively learned networks [4, 19, 41, 35] for image denoising task is earlier than its application on SR tasks. Different types of networks, including auto-encoder [41], multi-layer perceptron [35] and unfolded inference process of optimization models [34, 4] have been suggested for dealing with the image denoising problem. Recently, Zhang *et al*. [43] combined recent advances in DNN design and proposed a DnCNN model reaching state-of-the-art performance on denoising tasks with different noise levels. After DnCNN [43], some more complex networks have been suggested for further improving the denoising performance. Mao *et al*. [31] proposed to introduce skip connections for training very deep network, and proposed a residual encoding-decoding(RED) framework to solve image restoration problems. Tai *et al*. [38] adopt recursive unit and gate unit to learn multi-level representations, and proposed a very deep persistent memory network (MemNet) to deal with image restoration tasks. Although RED [31] and MemNet [38] achieved better denoising performance than the DnCNN method, they also requires much higher computational resources in both the training and testing phases.

Activation Functions of DNN The nonlinear capacity of deep neural networks come from the non-linear activation functions (AFs). In the study of early years, the designing of AFs often with strong biological or probability theory motivations, the sigmoid and tanh functions have been suggested for introducing nonlinearity in networks. While, the recent study of AFs take more consideration on practical training performance, the ReLU [32] function became the most popular AF since it enables better training of deeper networks [11]. Different AFs, including the Leaky ReLu (LReLU) [30], Exponential Linear Units (ELU) [7] and the Max-Out Unit (MU) [12] have been designed for improving the performance of DNN.

Theoretically, a sufficiently large networks with any of the above hand-crafted AFs can approximate arbitrarily complex functions [5]. However, for many practical applications where the computational resources are limited, the choice of AF affects greatly the network capacity. To improve the model fitting ability of network, He *et al*. [16] extend the original ReLU and propose Parametric ReLU (PReLU) by learning parameters to control the slopes in the negative part for each channel of feature maps. Besides PReLU [16], there are still some more complex parameterization of nonlinear functions [1, 26, 4]. However, these approaches [1, 26, 4] share a similar idea of adaptively summing several simple functions (kernel function) to achieve a more complex model and, therefore, their computational burden largely increase with the demand on parameterization accuracy. Furthermore, some of these functions [1, 26] were designed for classification tasks with fixed input size, the AFs were learned to be spatial vari-

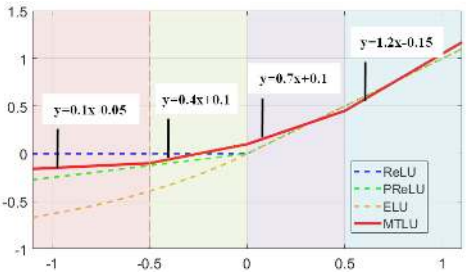


Figure 2. Examples of 4-bin MTLU, ReLU, PReLU, and ELU. The proposed MTLU divided the activation space into equidistant bins and approximates the activation function each bin with a linear function. Each AF in the network can be learned from the training data. See Sec. 2 for more details.

ant, making them not suitable for spatial invariant image restoration tasks. Yan *et al*. [37] unfold the optimization process of ADMM algorithm to stage-wise networks for solving the MR image reconstruction problem. They adopt the piece-wise linear function (PLF) to approximate nonlinear penalty functions for filter responses in each stage of network. Yan *et al*. [37] train the network parameters with an elaborate optimization algorithm, LBFGS [29], which requires to calculate gradients on all the training samples and utilize line search method to determine the step length in each update. Whether such a piece-wise linear function can be plugged into general CNN framework and be optimized with stochastic algorithms is still an open problem.

In this paper, we propose MTLU and directly plug it into the general CNN framework as the AF. We evaluate it on a wide range of network structures and train it with the commonly used stochastic optimization scheme. Our experimental results show that MTLU is not only faster than PLF, but also achieves better restoration performance on both the SR and denoising tasks.

2. Multi-bin Trainable Linear Unit

In this section, we firstly introduce the proposed MTLU activation function. Then, we present the gradient used for training MTLU and further discuss some important characters of MTLU.

2.1. MTLU: formulation

The nonlinearity of neural networks comes from the nonlinear activation functions, by stacking some simple operations, *e.g*. convolution and ReLU, the networks are able to model any nonlinear function. However, in many real applications the computational resources are limited and, thus, we are not able to deploy very deep models to fully capture the nonlinearity. This motivated us to improve the capacity of activation functions for better nonlinearity modeling.

Instead of designing fixed activation functions, we proposed to parameterize the activation functions and learn optimal functions for different stage of networks. We present the following MTLU activation function, which simply divides the activation space into multiple equidistant bins, and uses different linear functions to generate activations in different bins:

$$f(x) = \begin{cases} a_0 x + b_0, & if\ x \le c_0; \\ a_k x + b_k, & if\ c_{k-1} < x \le c_k; \\ \quad ... & \\ a_K x + b_K, & if\ c_{K-1} < x. \end{cases} \quad (1)$$

where $\{c_k\}_{k=0,...,K-1}$ are K hyper-parameters for MTLU, and $\{a_k, b_k\}_{k=0,...,K}$ are parameters to be learned in the training process. Since the anchor points c_k in our model are uniformly assigned, they are defined by the number of bins (K) and the bin width. Furthermore, given the input value x, a simple dividing and flooring function can be utilized to find its corresponding bin-index. Having the bin-index, the activation output can be achieved by an extra multiplication and addition function.

We present an example of a 4-bin MTLU in Fig. 2 and, for reference, some other commonly used activation functions are also included. One can see that in this simple case, MTLU divides the activation space into 4 parts, $(-\infty, -0.5]$, $(-0.5, 0]$, $(0, 0.5]$ and $(0.5, \infty)$, and adopts different linear functions in different parts to form the nonlinear activation function. PReLU [16] can be seen as a special case of the proposed parameterization in which the input space is divided into two bins $(-\infty, 0]$ and $(0, \infty)$ and only the parameter a_0 is learned, the other parameters b_0, a_1 and b_1 being fixed to 0, 1 and 0, resp. The proposed MTLU adopts a more flexible formula and is expected to have a stronger nonlinear capacity than PReLU. The detailed settings for the bin number as well as bin width will be discussed in the experimental section 5.1. In the next subsection, we introduce the gradients used for training MTLU, we show that MTLU can be directly plugged into the CNN framework and jointly trained with convolution kernels in an end-to-end manner.

2.2. MTLU: gradients

MTLU can be trained with the back-propagation algorithm. The parameters for the $k-$th bin will be affected by all the signals drop into $(c_{k-1}, c_k]$, thus, the gradients with respect to a_k and b_k can be written as follows

$$\frac{\partial loss}{\partial a_k} = \frac{1}{N_k} \sum_{x_i \in S_k} x_i \partial f_i, \quad \frac{\partial loss}{\partial b_k} = \frac{1}{N_k} \sum_{x_i \in S_k} \partial f_i \quad (2)$$

where N_k is the normalization factor that counts the number of signals in each bin, S_k indicate the range corresponding to a_k and b_k, ∂f_i denotes the gradient coming from the next layer in position i. And the gradient flow toward the bottom layer can be achieved by

$$\frac{\partial loss}{\partial x_i}_{x_i \in S_k} = a_k. \quad (3)$$

In our implementation, we do not count the number of signals laying in each bin, and just use the accumulated gradients. The gradient of MTLU parameters is relatively small and we found weight decay will affect the training of MTLU. For all the experiments in this paper, we do not conduct weight decay on MTLU. While, the other parameters are regularized by a weight decay factor of 10^{-4} which is the same as [22, 43]. Our implementation of MTLU is available at our project webpage[1].

2.3. Discussion

We emphasize that we choose the formulation of MTLU in (1) based on efficiency. Although several other non-linear approximation approaches have been suggested in different areas of computer vision, we will show in the experimental section that the proposed MTLU is able to deliver good results with lower computational burden. Concretely, since most of previous approaches [1, 26, 4] adopted a summation strategy, the increasing of parameterization accuracy will greatly increase the computational burden in the inference phase. Compared with recent work (PLF) [37], MTLU decoupled the linear functions in each bins and learn AF in each area independently. While PLF uses anchors to parameterize the linear functions in the intervals, each anchor affects the linear functions in two adjacent bins. Consequently, the parameterizations in different intervals affect each other and limit the flexibility of PLF. Furthermore, anchor-based formula requires to pre-determine the range of parameterization, for the inputs lies outside the parameterization range, PLF can only adopt a fixed activation function. While, the proposed MTLU is trainable for the entire activation space.

Another thing which is worth mention is the discontinuity issue. Some readers may raise concerns on the training stability of MTLU due to its discontinuity. Actually, some recent advances in the field of network compression and DNN-based image compression [18, 17, 27, 2, 39] have shown that the networks work well even with non-continuous functions. Furthermore, we experimentally found that the training of MTLU is very stable, even without a BN layer to normalize the range of inputs, it still able to deliver good restoration performance.

3. Fast Super Resolution Network with MTLU

To tackle SR a CNN is trained to extract local structure from LR images for estimating the lost high frequency details of HR images. By stacking the simple convolution (CONV) + ReLU operations, VDSR [22] achieved very good SR performance. To obtain a fast SR network, we modify the network structure to (i) process the image in a lower spatial resolution and (ii) use MTLU for improved nonlinear modeling capacity. Conducting the major SR operations in the LR space was originally adopted in the CSC-SR approach [15], where the convolution sparse coding decomposes the LR input image to get feature maps for SR. For DNN-based SR approaches, Shi *et al*. [36] firstly suggested to use the LR image (instead of interpolated image) as input and conduct SR operation in the LR space, and such a strategy has been widely applied in recent DNN-based SR methods [25, 28]. Different from recent approach [25], which utilizes larger channel numbers and filter sizes to gradually generate the final HR reconstruction from LR feature maps, we utilize the 64 LR feature maps to generate the shuffled HR image with only one 3×3 convolution layer for the purpose of efficiency. As we will show in section 5.3, MTLU greatly improves the network nonlinear capacity, enabling good SR results with a lower depth (number of layers). Fig. 3(a) illustrates the proposed FSRNet. Since the purpose of this paper is to find a good trade-off between restoration performance and processing speed, and FSRNet is capable to achieve top SR results with a few layers, we did not employ residual blocks in the middle of our networks, and only set one residual connection between the feature maps of first layer and last layer.

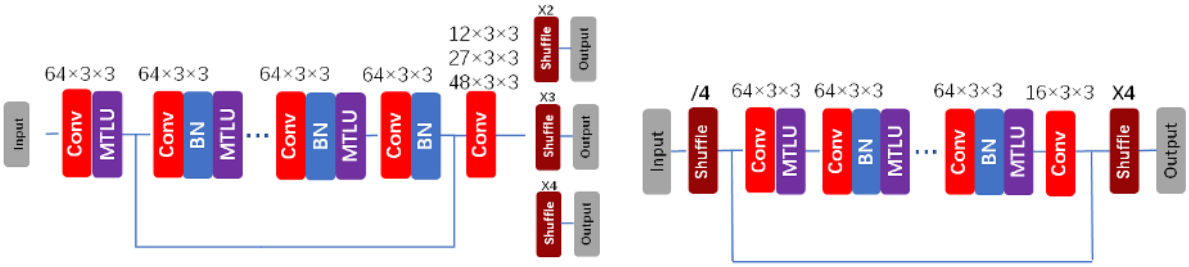
4. Fast Image Denoising Network with MTLU

Different from the SR task, the input and the target noise-free image in the denoising task are of same size. For the purpose of efficiency, we shuffle the input image and conduct the denoising operations at a lower spatial resolution. The shuffling operation has been adopted in several previous works [45, 36, 33] to change the spatial resolution of image/feature maps. Although processing the shuffled LR multi-channel image helps to reduce the computational burden in the training and testing phases as well as greatly improves the perception field of networks, It also has a higher demand on the nonlinear power of each layer since the shuffling operation narrows the network width. To balance speed and performance, we shuffle the input noisy image with factor 4, *e.g*. a noisy image with size $H\times W\times C$ is shuffled to $H/4\times W/4\times 16C$ as the input to the network. An illustration of the proposed FDnet structure can be found in Figure 3(b).

5. Experimental Results

In this section, we provide experimental results to show the advantage of the proposed models. First, we discuss some training aspects of the proposed MTLU and conduct experiments to compare MTLU with other activation functions which have been widely used in other image restoration networks. Then, we compare the proposed networks with representative state-of-the-art SR and denoising networks. All the experiments are conducted on a computer

[1] https://github.com/ShuhangGu/MTLU_ICCV2019.



(a) FSRnet (factor 2, 3 and 4) (b) FDnet

Figure 3. The network structures of the proposed FSRnet and FDnet. The hyper parameters for the shuffle layer and convolution layer are shown in the figures. Convolution block with $64 \times 3 \times 3$ represents convolution layer with kernel size 3 and 64 output feature maps. Shuffle blocks with parameter $/4$, $\times 4$ utilize shuffle operations to enlarge or reduce spatial resolution of input with factor 4.

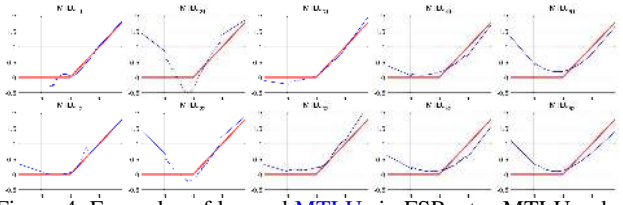


Figure 4. Examples of learned MTLUs in FSRnet$_7$. MTLU$_{ij}$ denotes the activation function for the j-th channel of layer i. ReLU function (in red) is included for reference.

with Intel Xeon CPU e5-2620, 64 GB of RAM and Nvidia Titan X Pascal GPU. We evaluate the running time of different networks with Caffe toolbox [20].

5.1. Training with MTLU

The proposed MTLU has several hyper parameters, in this section, we discuss some training details as well as parameter settings for the MTLU layers used in this paper.

MTLU: initialization In our experiments, we initialize MTLU as a ReLU function. With other initializations, such as random initialization of $\{a_k, b_k\}_{k=0,\ldots,K}$ and initialization MTLU as a identity mapping function $f(x) = x$, MTLU is still trainable. However, we experimentally found that the ReLU initialization often delivers a better convergence (about 0.05dB for SR *Set 14* with factor 4) than the models trained with other initialization methods.

Table 1. SR Results by FSRnet$_7$ with different bin-width on *Set 5* ($\times 4$). Number of bins are fixed to 2/bin-width.

Bin-width	0.025	0.05	0.1	0.2	0.5
PSNR [dB]	31.49	31.52	31.48	31.49	31.43

MTLU: number of bins and bin width The number of bins as well as bin-width determines the parameterization accuracy and range of the proposed MTLU layer. In both the FDnet and FSRnet structures, we have a Batch Normalization (BN) layer to help us adjust the range of inputs. We, thus, only carefully parameterize the activations between the range of [-1, 1], since most of the inputs of MTLU lies in this range. Note that for input signals out of the range [-1, 1], MTLU is still trainable and can generates valid activations, but for all the values $x \leq c_0$ or $x > c_{N-1}$, they share the same two groups of parameters $\{a_0, b_0\}$ or $\{a_N, b_N\}$.

After fixing the parameterization ranges, we only need to choose the bin-width and the number of bins is obtained as 2/bin-width. To choose the bin-width values, we train a group of 7 layers FSRnet (FSRnet$_7$) with different bin-width values, and evaluate the SR results by different models. The average PSNR values by different settings for SR the *Set 5* data set with zooming factor 4 are shown in Table 1. Intuitively, a smaller bin-width enables a higher parameterization accuracy of MTLU, and is expected to improve the nonlinearity modeling capacity of the network. However, this comes at the price of introducing more parameters (larger number of bins) to cover the range between (-1,1). Furthermore, as shown in Table 1, the result obtained with a bin-width smaller than 0.025 is worse than the ones with larger bin-widths. A possible reason is that the number of signals in certain bins may be very small rendering the training process unstable. In our experiments, we divided the space between -1 and 1 into 40 bins (bin-width 0.05) and learn different AFs for different channels. Thus, for a network with 64 channels of feature map, the number of parameters of each activation function is $64 \times 80 = 5120$, which is less than $1/7$ of the parameter number of one 3×3 convolution layer. Fig. 4 depicts AFs of the 1^{st} and 2^{nd} feature maps in each MTLU layers of learned FSRnet$_7$.

5.2. Comparison with other non-linear parameterization approaches

We compare MTLU with two representative non-linear parameterization approaches: APL [1] and PLF [37]. Please note that both APL and PLF functions have only been used in highly specific network structures, and we are the first to adapt them into the standard CNN framework for solving image restoration tasks. Here we compare MTLU with the two parameterization approaches to show the advantages of MTLU, *i.e.* efficiency and robustness.

APL [1] uses summation of ReLU-like units to increase the capacity of the activation function. As APL $h_i(x) = \max(0, x) + \sum_s a_i^s \max(0, -x + b_i^s)$ was designed for high-

level vision tasks and the AF varies with the position i, it is not directly applicable to restoration tasks with different input sizes. Thus, we remove the spatially variant part and adopt the same activation in different channels of feature map (the same setting as adopted for MTLU). We adopt APL with different number of kernels in FSRnet$_7$; the average PSNR on Set 5 ($\times$4) and runtime of FSRnet$_7$ with APL and with MTLU are reported in Table 2. FSRnet$_7$-MTLU is much faster and $\sim$0.2dB better than FSRnet$_7$-APL.

Table 2. SR results (Set 5, $\times$4) and run time for processing a 512$\times$ 512 image by FSRnet$_7$ with MTLU and variations of APL [1].

AFs	MTLU$_{40}$	APL_{10}	APL_{20}	APL_{40}
[dB]/[ms]	31.52/4.0	31.35/16.0	31.29/34.1	31.32/87.6

PLF [37] is another recently proposed non-linear parameterization approach, it uses a group of anchor points to determine the function values in the intervals. To compare MTLU with PLF, we incorporate them in both the FSRnet$_7$ and FDnet$_{10}$. In [37], the PLF was adopted in a highly specific network structure, and the authors only parameterized the interval between -1 and 1. While, in order to thoroughly compare MTLU and PLF, we generalize PLF approach. Specifically, we keep both the bin-width of MTLU and anchor interval of PLF as 0.05, and use different bin-numbers (anchor numbers). Changing the number of bins for both the MTLU and PLF approach will not affect their running time. Processing a 512×512 HR image by FSRnet$_7$-MTLU and FSRnet$_7$-PLF takes 4.0 and 4.4 ms, respectively. The SR (Set 5/$\times$4) and denoising (BSD68/$\sigma = 50$) results by different AFs are reported in Table 3.

On different settings, MTLU consistently outperforms the PLF. For the same setting as we adopted in this paper, *i.e*. 40 bin with bin-width 0.05, MTLU outperforms PLF 0.13 and 0.18 dB on the SR and denoising tasks, resp. Furthermore, when the two methods use a small number of anchors/bins, MTLU outperforms PLF by a large margin. One possible reason is that PLF uses anchors to parameterize the linear functions in the intervals, each anchor will affect the linear functions in two adjacent bins; as a result, when the intervals do not cover all the possible values range of input, *e.g*., PLF20 and PLF40 only covers input range of [-0.5,0.5] and [-1,1], the values outside the range will greatly affect the parameterization and PLF can not achieve a performance as good as MTLU, which decouples the functions in each bin. The relative larger performance gap between MTLU and PLF on the denoising task may also due to the correlated parameterization scheme adopted in PLF. Such a drawback of PLF may limit its application on feature maps which we do not have prior knowledge on the value ranges (*e.g*. feature maps without BN).

5.3. Comparison with other activation functions

In this part, we compare MTLU with other AFs recently adopted in image restoration networks [43, 25, 45]: ReLU [32], the most utilized AF, PReLU [16], adopted in the state-of-the-art SR algorithm SRResNet [25], and the Max-out Unit (MaxOut) [12] which has been adopted in a recently proposed SR approach [6]. Specifically, the fast SR approaches ESPCN [36], FSRCNN [10], the state-of-the-art SR approach SRResNet [25] as well as the proposed FSRnet (Fast SR Net, see Section 3) are utilized to compare different AFs on the SR task. For the denoising task, we compare different AFs on the proposed FDNet (Fast Denoising Net, see Section 4).

For existing network structures ESPCN [36], FSRCNN [10] and SRResNet [25], we directly replace the AFs in these networks to compare different AFs. In order to thoroughly compare the performance of different AFs with the proposed network structures, FSRnet and FDnet, we vary the basic building blocks (CONV+BN+AF) in the 2 network structures and compare the performance/speed curves of networks with different AFs. The details for the employed settings of the different network structures are described in the next.

5.3.1 Existing network structures with different AFs

In this part, we compare different AFs on representative existing restoration network structures: ESPCN [36], FSRCNN [10] and SRResNet [25]. ESPCN adopts three convolution+ReLU layers, the kernel sizes of different layers are $\{5, 3, 3\}$ and the feature map numbers of the first and second layers are $\{64, 32\}$. We follow the same setting and only conduct SR operation for the illumination channel. FSRCNN adopts more layers than ESPCN but less feature maps to trade off between SR performance and inference speed. The original FSRCNN [10] utilizes PReLU [16], and we derive FSRCNN-ReLU and FSRCNN-MTLU for comparison. SRResNet utilizes a large number (16) of residual blocks, conducts most of the computation in the LR space, to then use large kernels and 2 shuffle steps to gradually reconstruct the HR estimation with the LR feature maps.

For all the variations of FSRCNN, ESPCN and SRResNet, we adopt the DIV2K [3] as training set. We initialize the learning rate as 1^{-3}, for ESPCN [36] and FSRCNN [10], we divide the learning rate every 50K iterations until the learning rate is less than 1^{-5}; while, for the complex SRResNet [25], we divide its learning rate every 150K iterations. We train all the networks with Adam [23] solver using default parameters. Table 4 summarizes the PSNR results ($\times$4) on Set 5 and Set 14 as well as the running time of the different methods for processing a 512×512 image. MTLU greatly improves the fast SR approaches without a noticeable extra computational burden.

5.3.2 FSRnet with different AFs

We further evaluate the effectiveness of MTLU on the proposed FSRnet structure. In order to thoroughly compare dif-

Table 3. SR and denoising results for FSRnet_7 and FDnet_{10} variants using our MTLU, PLF [37], ReLU [32] and PReLU [16] as AF, resp.

		AFs:	MTLU_{20}	MTLU_{40}	MTLU_{80}	PLF_{20}	PLF_{40}	PLF_{80}	ReLU	PReLU
Super-resolution:	FSRnet_7	Set5, $\times 4$	31.50	31.52	31.54	31.34	31.39	31.49	31.44	31.43
Denoising:	FDnet_{10}	BSD68, $\sigma = 50$	26.20	26.24	26.23	26.01	26.06	26.13	26.06	26.14

Table 4. SR PSNR results [dB] on Set 5 and Set 14 ($\times 4$) and runtime [ms] (on a 512×512 image) for ESPCN [36], FSRCNN [10] and SRResNet [25] variants. The subscript letters *R*, *P* and *M* represents the ReLU [32], PReLU [16] and MTLU, resp.

Networks	ESPCN_R	ESPCN_P	ESPCN_M	FSRCNN_R	FSRCNN_P	FSRCNN_M	SRResNet_R	SRResNet_P	SRResNet_M
Set 5	30.66	30.67	30.82	30.73	30.76	30.96	32.07	32.06	32.13
Set 14	27.60	27.60	27.68	27.61	27.64	27.75	28.49	28.49	28.54
runtime	0.94	0.94	0.95	1.97	1.98	1.98	30.28	30.97	31.29

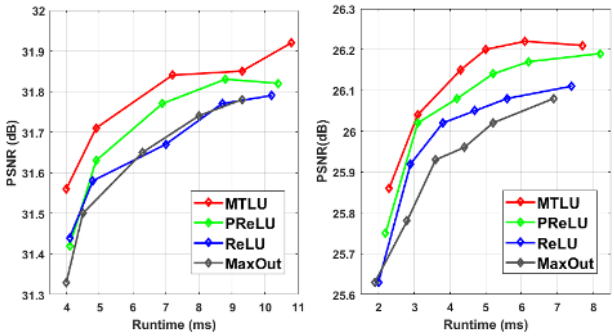


Figure 5. SR (Set5, $\times 4$) and Denoising ($\sigma = 50$) results by different network structures with different AFs. (*Left*) The Markers represent FSRnet with 7, 9, 11, 13 and 17 convolution layers, resp. (*Right*) The Markers represent FDnet with 4, 6, 8, 10, 12 and 16 layers, resp.

ferent AFs on the proposed FSRnet, we train FSRnet with 7, 9, 11, 13, and 17 layers, resp. We use the same training data DIV2K to train all the methods. The initial learning rate is 1^{-3} and is divided by 2 every 80K, 80K, 120K, 120K and 160K iterations for different layer numbers. All the models are trained with 96×96 sub-images with batch size 32.

The SR results for FSRnet are shown in Figure 5. Compared with other AFs, the MTLU-based network trades off better between the processing speed and SR performance.

5.3.3 FDnet with different AFs

In this part, we compare different AFs on the proposed FDnet. FDnet shuffles the input noisy images with factor $1/4$, the receptive field of network increases quickly with the increase of number of layers. Furthermore, the same number of 64 feature maps in FDnet need to process the input shuffled image cubic with 16 channels. Due to both mentioned reasons make FDnet has a higher demand on nonlinear capacity of each layers.

We collected 80,000 72×72 image crops from the 400 training images used by the DnCNN paper [43], and white Gaussian noise with $\sigma = 50$ was added into the clean images to generate the noisy input images. We train a group FDnet with 4, 6, 8, 12, and 16 convolution layers. For all the models, we set the batch size as 32, and trained them with Adam solver. The learning rate for different models was initialized with 1×10^{-3}, and divided the learning rate by 2 every 60K, 80K, 100K, 120K and 120K iterations for models with layer numbers of 4, 6, 8, 12 and 16, respectively. The training process stopped when the learning rate dropped below 1×10^{-5}.

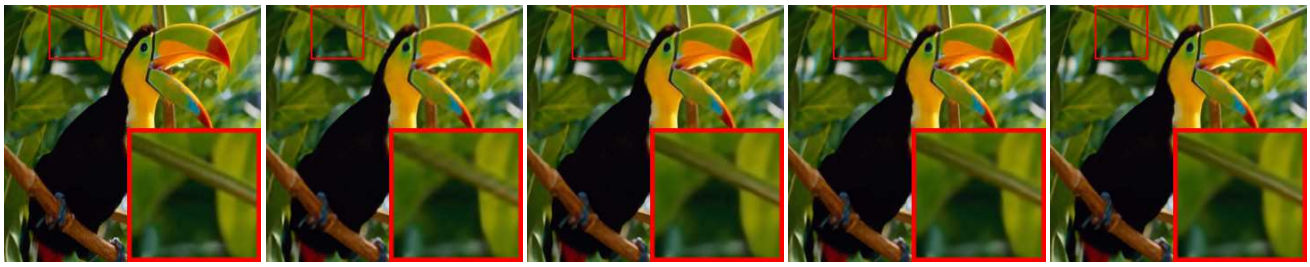
The PSNR indexes by different variations on the BSD68 dataset are shown in Figure 5. From the figure, one can easily see that the proposed MTLU achieves much better results than the competing AFs when combined with the proposed FDnet structure. For equal number of layers, the MTLU-based network outperforms the MaxOut, ReLU and PReLU based networks by about 0.16, 0.12 and 0.08 dB. Furthermore, since we conduct all the operations at a LR space, the adoption of MTLU introduces only a negligible computation burden in comparison with the compared AFs. The processing speed by FDnet-PReLu and FDnet-MTLU are almost the same.

5.4. Comparison with s-o-t-a SR algorithms

We compare the proposed FSRnet with five typical CNN-based SR approaches: the seminal CNN-based SR approach SRCNN [9], the benchmark VDSR method [22] and current state-of-the-arts LapSRN [24], MemNet [38] and SRResNet [25]. We compare the SR results on 3 commonly used datasets, *i.e*. Set 5, Set 14 and BSD100, following the settings from [40, 9, 22, 25] with zooming factors 2, 3, 4.

We train on DIV2K [3] dataset three versions of FSRnet, with 7, 13, and 19 layers namely FSRnet_7, FSRnet_{13}, and FSRnet_{19}, to trade-off between performance and speed. For different zooming factors, we collected training images to make the networks with a spatial resolution of 24×24 in the training phase, which means the corresponding HR training images for factors 2, 3, and 4 are with size of 96, 72 and 96, resp. We use a batch size of 32, Adam solver, the initial learning rate set to 0.001, and divide the learning rate by 2 every 80K, 120K and 200K iterations. The training process stops when the learning rate is less than 1×10^{-5}.

The SR results by different approaches are shown in Table 5, and the runtimes for processing a 512×512 image are reported in Table 6. The proposed FSRnet_7 achieved slightly better results than VDSR [22] but much faster ($\times 4$, $\times 7$ and $\times 10$ for zooming factors 2, 3 and 4, resp.). Also, FSRnet_{19} compares with the state-of-the-art SRResNet [25], while being $> 3\times$ faster. Figure 1 provide a visualization of the trade-off power of our FSRnet model equipped with MTLU. Fig. 6 shows some SR image results.



Ground Truth (PSNR/runtime) SRCNN [9] (35.63 dB/1.9 ms) VDSR [22] (36.76 dB/19.3 ms) FSRnet$_7$ (36.94 dB/2.4 ms) FSRnet$_{19}$ (37.88 dB/6.6 ms)

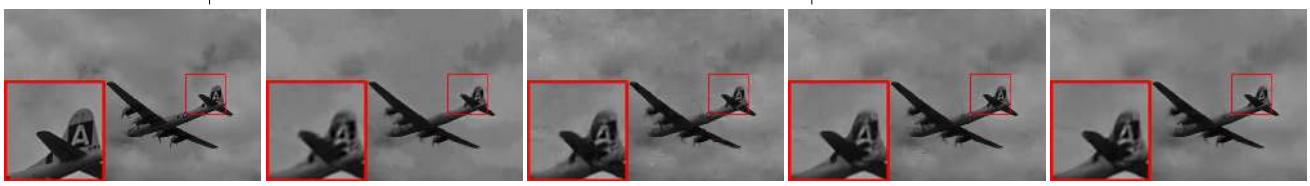
Figure 6. SR results of the *bird* image by different methods ($\times 3$). The running times are evaluated on a Titan X Pascal GPU.

Table 5. Super-resolution PSNR results [dB] by different methods.

Dataset	Factor	SRCNN [9]	VDSR [22]	LapSRN [24]	MemNet [38]	SRResNet [25]	FSRnet$_7$	FSRnet$_{13}$	FSRnet$_{19}$
Set 5	$\times 2$	36.66	37.53	37.52	37.78	-	37.58	37.73	37.82
	$\times 3$	32.75	33.66	-	34.09	-	33.71	34.07	34.13
	$\times 4$	30.48	31.35	31.54	31.74	32.05	31.52	31.82	31.89
Set 14	$\times 2$	32.42	33.03	33.08	33.28	-	33.19	33.31	33.47
	$\times 3$	29.28	29.77	-	30.00	-	29.93	30.16	30.22
	$\times 4$	27.49	28.01	28.19	28.26	28.49	28.22	28.42	28.51
BSD 100	$\times 2$	31.36	31.90	31.80	32.08	-	31.89	32.02	32.10
	$\times 3$	28.41	28.82	-	28.96	-	28.80	28.96	29.01
	$\times 4$	26.90	27.29	27.32	27.40	27.58	27.31	27.45	27.52

Table 6. Runtimes [ms] by SR methods processing a 512×512 pixels HR image.

Factor	SRCNN [9]	VDSR [22]	MemNet [38]	SRResNet [25]	FSRnet$_7$	FSRnet$_{13}$	FSRnet$_{19}$
$\times 2$	4.3	42.2	926.8	-	9.7	18.5	27.4
$\times 3$	4.2	42.0	930.1	-	6.0	11.1	14.2
$\times 4$	4.3	41.7	928.0	31.2	3.9	6.8	8.9



Ground Truth WNNM [13] TNRD [4] DnCNN [43] FDnet (**our**)

Figure 7. Examples of denoising results by different methods (noise level, $\sigma = 50$).

5.5. Comparison with s-o-t-a denoising algorithms

We compare in Tables 7 and 8 and Figure 7 the proposed FDnet with state-of-the-art CNN-based DnCNN [43] and other representative methods, including state-of-the-art unsupervised approaches BM3D [8] and WNNM [14], and one discriminative learning approach TNRD [4]. To train FDnet we used the same 80000 image crops, initialized the learning rate to 1×10^{-3} and divided it by 2 every 100K iterations. Our FDnet uses 10 convolution layers to balance between denoising performance and speed. FDnet$_{10}$ compares favorable to DnCNN and the other methods in PSNR terms and is $> 10\times$ faster than DnCNN.

Table 7. Image Denoising PSNR results [dB] on BSD68.

Noise Level	BM3D [8]	WNNM [14]	TNRD [4]	DnCNN [43]	FDnet
$\sigma = 25$	28.57	28.83	28.92	29.23	29.12
$\sigma = 50$	25.62	25.87	25.97	26.23	26.24
$\sigma = 75$	24.21	24.40	-	24.64	24.76

Table 8. Runtime [ms] for processing 512×512 noisy image.

	DnCNN [43]	**FDnet (our)**
runtime (TitanX Pascal GPU)	74.8	**5.0**

6. Conclusion

We introduced the multi-bin trainable linear unit (MTLU), a novel activation function for increasing the nonlinear capacity of neural networks. MTLU is a robust alternative to the current activation functions, it improves the results of a wide range of restoration networks. Based on MTLU, we proposed two efficient networks: a fast super-resolution network (FSRnet) and a fast denoising network (FDnet). They trade-off between speed and performance better than the prior art in image restoration. On standard super-resolution and denoising benchmarks the proposed networks achieve comparable results with the current state-of-the-art deep learning networks but significantly faster and with lower memory requirements.

Acknowledgments: This work was supported by the ETH Zurich General Fund and an Nvidia GPU hardware grant.

References

[1] Forest Agostinelli, Matthew Hoffman, Peter Sadowski, and Pierre Baldi. Learning activation functions to improve deep neural networks. *arXiv preprint arXiv:1412.6830*, 2014. 2, 4, 5, 6

[2] Eirikur Agustsson, Fabian Mentzer, Michael Tschannen, Lukas Cavigelli, Radu Timofte, Luca Benini, and Luc V Gool. Soft-to-hard vector quantization for end-to-end learning compressible representations. In *NIPS*, 2017. 4

[3] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *CVPR Workshops*, 2017. 6, 7

[4] Yunjin Chen and Thomas Pock. Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration. *IEEE transactions on pattern analysis and machine intelligence*, 2017. 2, 4, 8

[5] Youngmin Cho and Lawrence K Saul. Large-margin classification in infinite neural networks. *Neural computation*, 2010. 2

[6] Jae-Seok Choi and Munchurl Kim. Can maxout units downsize restoration networks?-single image super-resolution using lightweight cnn with maxout units. *arXiv preprint arXiv:1711.02321*, 2017. 6

[7] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015. 2

[8] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on image processing*, 2007. 8

[9] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 2016. 2, 7, 8

[10] Chao Dong, Chen Change Loy, and Xiaoou Tang. Accelerating the super-resolution convolutional neural network. In *ECCV*, 2016. 2, 6, 7

[11] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011. 2

[12] Ian J Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. Maxout networks. *arXiv preprint arXiv:1302.4389*, 2013. 2, 6

[13] Shuhang Gu, Qi Xie, Deyu Meng, Wangmeng Zuo, Xiangchu Feng, and Lei Zhang. Weighted nuclear norm minimization and its applications to low level vision. *International journal of computer vision*, 2017. 8

[14] Shuhang Gu, Lei Zhang, Wangmeng Zuo, and Xiangchu Feng. Weighted nuclear norm minimization with application to image denoising. In *CVPR*, 2014. 8

[15] Shuhang Gu, Wangmeng Zuo, Qi Xie, Deyu Meng, Xiangchu Feng, and Lei Zhang. Convolutional sparse coding for image super-resolution. In *ICCV*, 2015. 2, 4

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015. 2, 3, 6, 7

[17] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. In *NIPS*, 2016. 4

[18] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Quantized neural networks: Training neural networks with low precision weights and activations. *arXiv preprint arXiv:1609.07061*, 2016. 4

[19] Viren Jain and Sebastian Seung. Natural image denoising with convolutional networks. In *NIPS*, 2009. 2

[20] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014. 1, 5

[21] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 2016. 2

[22] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *CVPR*, 2016. 1, 2, 4, 7, 8

[23] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[24] Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Deep laplacian pyramid networks for fast and accurate superresolution. In *CVPR*, 2017. 7, 8

[25] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, 2017. 2, 4, 6, 7, 8

[26] Hongyang Li, Wanli Ouyang, and Xiaogang Wang. Multi-bias non-linear activation in deep neural networks. In *ICML*, 2016. 2, 4

[27] Mu Li, Wangmeng Zuo, Shuhang Gu, Debin Zhao, and David Zhang. Learning convolutional networks for content-weighted image compression. 4

[28] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *CVPR Workshops*, 2017. 4

[29] Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989. 3

[30] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *ICML*, 2013. 2

[31] 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*, pages 2802–2810, 2016. 2

[32] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010. 2, 6, 7

[33] Mehdi SM Sajjadi, Raviteja Vemulapalli, and Matthew Brown. Frame-recurrent video super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6626–6634, 2018. 4

[34] Uwe Schmidt and Stefan Roth. Shrinkage fields for effective image restoration. In *CVPR*, 2014. 2

[35] Christian J Schuler, Harold Christopher Burger, Stefan Harmeling, and Bernhard Scholkopf. A machine learning approach for non-blind image deconvolution. In *CVPR*, 2013. 2

[36] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *CVPR*, 2016. 1, 2, 4, 6, 7

[37] Jian Sun, Huibin Li, Zongben Xu, et al. Deep admm-net for compressive sensing mri. In *NIPS*, pages 10–18, 2016. 3, 4, 5, 6, 7

[38] Ying Tai, Jian Yang, Xiaoming Liu, and Chunyan Xu. Memnet: A persistent memory network for image restoration. In *CVPR*, 2017. 2, 7, 8

[39] Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár. Lossy image compression with compressive autoencoders. *arXiv preprint arXiv:1703.00395*, 2017. 4

[40] Radu Timofte, Vincent De Smet, and Luc Van Gool. A+: Adjusted anchored neighborhood regression for fast super-resolution. In *ACCV*, 2014. 2, 7

[41] Junyuan Xie, Linli Xu, and Enhong Chen. Image denoising and inpainting with deep neural networks. In *NIPS*, 2012. 2

[42] Jianchao Yang, John Wright, Thomas Huang, and Yi Ma. Image super-resolution as sparse representation of raw image patches. In *CVPR*, 2008. 2

[43] 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*, 2017. 1, 2, 4, 6, 7, 8

[44] Kai Zhang, Wangmeng Zuo, Shuhang Gu, and Lei Zhang. Learning deep cnn denoiser prior for image restoration. *arXiv preprint arXiv:1704.03264*, 2017. 2

[45] Kai Zhang, Wangmeng Zuo, and Lei Zhang. Ffdnet: Toward a fast and flexible solution for cnn based image denoising. *arXiv preprint arXiv:1710.04026*, 2017. 1, 2, 4, 6