Chaining Identity Mapping Modules for Image Denoising

Saeed Anwar\textsuperscript{1,2}, Cong Phuoc Huynh\textsuperscript{1}, Fatih Porikli\textsuperscript{1}
\textsuperscript{1}The Australian National University, Australia \quad \textsuperscript{2}DATA61, CSIRO, Australia
firstname.lastname@anu.edu.au

Abstract

We propose to learn a fully-convolutional network model that consists of a Chain of Identity Mapping Modules (CIMM) for image denoising. The CIMM structure possesses two distinctive features that are important for the noise removal task. Firstly, each residual unit employs identity mappings as the skip connections and receives pre-activated input in order to preserve the gradient magnitude propagated in both the forward and backward directions. Secondly, by utilizing dilated kernels for the convolution layers in the residual branch, in other words within an identity mapping module, each neuron in the last convolution layer can observe the full receptive field of the first layer. After being trained on the BSD400 dataset, the proposed network produces remarkably higher numerical accuracy and better visual image quality than the state-of-the-art when being evaluated on conventional benchmark images and the BSD68 dataset.

1. Introduction

Image denoising is an essential building module for various computer vision and image processing algorithms. In the past few years, the research focus in this area has been shifted to how to make the best use of image priors. To this end, several approaches attempted to exploit non-local self-similar (NSS) patterns \cite{3,10,11}, sparse models \cite{17,34}, gradient models \cite{33,39,38}, Markov random field (MRF) models \cite{35}, external denoising \cite{42,1,30} and convolutional neural networks \cite{45,26,45}.

The nonlocal self-similar patch matching (NLM) and module matching with collaborative filtering (BM3D) have been two prominent baselines for image denoising for almost a decade now. Due to popularity of NLM \cite{3} and BM3D \cite{10}, a number of their variants \cite{15,11,25,16} were also proposed to execute the search for similar patches in similar transform domains.

Complementing above, use of external priors for denoising has been motivated by the pioneering studies \cite{27,6}, which showed that selecting correct reference patches from a large external image dataset of clean samples can theoretically suppress additive noise and attain infinitesimal reconstruction error. However, directly incorporating patches from an external database is computationally prohibitive even for a single image. To overcome this problem, Chan \textit{et al.} \cite{5} proposed efficient sampling techniques for large databases but still the denoising is impractical as it takes hours to search patches for one single image if not days. An alternative to these methods can be considered as the dictionary learning based approaches \cite{13,31,12} which learn an over-complete dictionary from a set of external natural clean images and then enforce patch self-similarity through

![image](a) Input (14.16 dB) (b) Our Denoised (32.64 dB)

![image](c) IRCNN \cite{45} (32.07 dB) (d) DnCNN \cite{44} (32.05 dB)

Figure 1. Denoising results for an image corrupted by the Gaussian noise with $\sigma = 50$. Our result has the best PSNR score, and unlike other methods, it does not have over-smoothing or over-contrast artifacts. Best viewed in color on high-res display.
Towards an efficient fusion of external datasets, many recent works [47, 44, 40] investigated the use of maximum likelihood frameworks to learn Gaussian Mixture Model (GMM) of natural image patches or group patches for clean patch estimation. Several studies, including [41, 7], modified Zoran et al. [47]'s statistical prior for reconstruction of class-specific noisy images by capturing the statistics of noise-free patches from a large database of same category images through the Expectation-Maximization algorithm. Other similar methods on external denoising include TID [30], CSID [1] and CID [43]; however, all of these have limited applicability in denoising of generic (from an unspecified class) images.

The advent of convolutional neural networks (CNN) provides a significant performance boost for image denoising methods [44, 45, 26, 4, 56] have also been proposed very recently. CSF [36] learns a single framework based on unification of random-field based model and half-quadratic optimization. Similarly, TNRD [8] adapts field-of-experts [35] prior into CNN framework by incorporating a preset number of inference steps. Undoubtedly, CSF and TNRD have shown improved results over more classical methods; however, the imposed image priors inherently impede their performances, which highly rely on the choice of hyperparameter settings, extensive fine-tuning and stage-wise training.

To overcome the drawbacks of CSF and TNRD, IR-CNN [45] and DnCNN [44] learn the residual present in the contaminated image by using the noise in the loss function instead of the clean image as the ground-truth. Although both models were able to report favorable results, their performance depends heavily on the accuracy of noise estimation without knowing the underlying structures and textures present in the image. Besides, they are computationally expensive because of the batch normalization operations after every convolutional layer. Another notable work in denoising is NLNET [26] which exploits the nonlocal self-similarity using deep networks. This model improves on classical methods but lagging behind IRCNN and DnCNN, as it inherits the limitations associated with the NSS priors as not all patches recur in an image.

**Inspiration & Motivation:** Current convolutional neural network based image denoising methods [4, 44, 45, 26] connect weight layers consecutively and learn the mapping by brute force without putting any effort into the architecture. One problem with such an architecture is the addition of more weight layers to increase the depth of the network. Even if the new weight layers are added to the mentioned CNN based denoising methods, it will fall into gradients vanishing problem and impel it further [4]. This property of increasing the size of the network is important and helps in performance boost [28, 20]. Therefore, our goal is to propose a model that overcomes this deficiency.

Another reason is the lack of true color denoising. Most of the current denoising systems are either for grayscale image denoising or treat each color channel separately ignoring the relationship between the color channels. Only a handful of works [9, 1, 44, 26] approached color image denoising in its own context.

To provide a solution, our choice is the convolutional neural networks in a discriminative prior setting for image denoising. There are many advantages of using CNNs, including efficient inference, incorporation of robust priors, integration of local and global receptive fields, regressing on nonlinear models, and discriminative learning capability. Furthermore, we propose a modular network where we call each module as a mapping modules (MM). The mapping modules can be replicated and easily extended to any arbitrary depth for performance enhancement.

**Contributions:** The contributions of this work can be summarized as follows:

- An effective CNN architecture that consists of a Chain of Identity Mapping modules (CIMM) for image denoising. These modules share a common composition of layers, with residual connections between them to facilitate training stability.
- The use of dilated convolutions for learning suitable filters to denoise at different levels of spatial extent.
- A single denoising network that can handle various noise levels.

## 2. Chain of Identity Mapping Modules

This section presents our approach to image denoising by learning a Convolutional Neural Network consisting of a Chain of Identity Mapping Modules (CIMM). Each module is composed of a series of preactivation units followed by convolution functions, with residual connections between them. Section 2.1 formulates the learning objective. Subsequently, the meta-structure of the CIMM network in Section 2.1.

### 2.1. Network Design

Residual learning has recently delivered state of the art results for object classification [18, 21] and detection [29], while offers training stability. Inspired by the Residual Network variant with identity mapping [21], we adopt a modular design for our denoising network. The design consists of a Chain of Identity Mapping modules (CIMM).
Network elements: Figure 2 depicts the entire architecture, where identity mapping modules are shown as a blue block, which are in turn composed of basic ReLU (orange) and convolution (green) layers. The output of each module is a summation of the identity function and the residual function. In our experiments, we typically employ 64 filters of size $3 \times 3$ in each convolution layer.

The meta-level structure of the network is governed by three parameters: the number of identity modules ($i.e.$ M), the number of pre-activation-convolution pairs in each module ($i.e.$ L), and the number of output channels ($i.e.$ C), which we fixed across all the convolution layers.

The high-level structure of the network can be viewed as a chain of identity mapping modules, where the output of each module is fed directly into the subsequent one. Subsequently, the output of this chain is fed to a final convolution layer to produce a tensor with the same number of channels as the input image. At this point, the final convolution layer directly predicts the noise component from a noisy image. The noise-free image/patch is then subtracted from the input to recover the noise-free image.

The identity mapping modules are the building blocks of the network, which share the following structure. Each module consists of two branches: a residual branch and an identity mapping branch. The residual branch of each module contains a series of layers pairs, i.e. a nonlinear pre-activation (typically ReLU) layer, followed by a convolution layer. Its main responsibility is to learn a set of convolution filters to predict image noise. In addition, the identity mapping branch in each module allows the propagation of the loss gradients in both directions without any bottleneck.

Justification of design For image denoising, several previous works have adopted a fully convolutional network design, without any pooling mechanism [44, 23, 45]. This is necessary in order to preserve the spatial resolution of the input tensor across different layers. We follow this design by using only non-linear activations and convolution layers across our network.

Furthermore, as inspired by non-local denoising method, we design the convolution layers in such a way that neurons in the last layer of each identity mapping (IM) module observe the full spatial receptive field in the first convolution layer. This design helps learning to connect input neurons at all spatial locations to the output neurons, in much the same way as well-known non-local mean methods such as [10, 3]. Instead of using a unit stride within each layer, we also experimented with dilated convolutions to increase the receptive fields of the convolution layers. By this design, we can reduce the depth of each IM module while the final layer’s neurons can still observe the full input spatial extent.

Pre-activation has been shown to offer the highest performance for classification when used together with identity mapping [21]. In a similar fashion, our design employs ReLU before each convolution layer. This design differs from existing neural network architectures for denoising [23, 44, 26]. The pre-activation helps training to converge more easily, by while the identity function preserves the range of gradient magnitudes. Also, the resulting network generalizes better as compared to the post-activation alternative. This property enhances the denoising ability of our network.

Formulation Now we formulate the prediction output of this network structure for a given input patch $x$. Let $W$ denote the set of all the network parameters, which consists of the weights and biases of all constituting convolution layers. Specifically, we let $w_{m,l}$ denote both the kernel and bias parameters of the $l$-th convolution layer in the $m$-th residual branch. Within such a branch, the intermediate output $x_{m,l}$ of the $l$-th ReLU-convolution pair is a composition of two functions

$$ x_{m,l} = f(g(x_{m,l-1}; w_{m,l}), $$

where $f$ and $g$ are the notation for the convolution and the ReLU functions, $x_{m,l-1}$ is the output of the $i.e.$ $m-1$-th ReLU-convolution pair. By convention, we let $x_{m,0} = r_{m-1}$.

By composing the series of ReLU-convolution pairs, we obtain the output of the $m$-th residual branch as

$$ r_m \triangleq x_{m,L} = f(g(\ldots f(g(r_{m-1}; w_{m,0})\ldots); w_{m,l}) $$

Chaining all the identity mapping modules, we obtain the intermediate output as $x + \sum_{m=1}^{M} r_m$. Finally, the output of this chain is convolved with a final convolution layer with learnable parameters $w_{M+1}$ to predict the noise component as $h(x, W) = f(x + \sum_{m=1}^{M} r_m, w_{M+1})$.

2.2. Learning to Denoise

Our convolutional neural network (CNN) is trained on image patches or regions rather than at the image-level.
This decision is driven by a number of reasons. Firstly, it offers random sampling of a large number of training samples at different locations from various images. Random shuffling of training samples is well-known to be a useful technique to stabilize the training of deep neural networks. Therefore, it is preferable to batch training patches with a random, diverse mixture of local structures, patterns, shapes and colors. Secondly, there has been success in approaches that learn image patch priors from external data for image denoising [47].

From a set of noise-free training images, we randomly crop a number of training patches \( y_i, i = 1, \ldots , N \) as the groundtruth. The noisy version of these patches is obtained by adding (Gaussian) noise to the ground truth training images. Let us denote the set of noisy patches corresponding to the former as \( y_i, i = 1, \ldots , N \). With this setup, our image denoising network (described in Section 2.1) is aimed to reconstruct a patch \( x^*_i = h(x_i, W) \) from the input patch \( y_i \).

The learning objective is to minimize the following sum of squares of \( \ell^2 \)-norms

\[
\mathcal{L} \triangleq \sum_{i=1}^{N} \| h(x_i, W) - y_i \|^2
\]  

(3)

To train the proposed network on a large dataset, we minimization of the objective function in Equation 3 on minibatches of training examples. Training details for our experiments are described in Section 3.2.

3. Experiments

3.1. Datasets and Baselines

We performed experimental validation on the widely used classical images and BSD68 datasets. To generate noisy test images, we corrupt the images by additive white Gaussian noise with standard deviations (std) of \( \sigma_n = 15, 25, 50, 70 \), as employed by [45] [44] [26]. For evaluation purposes, we use the Peak Signal-to-Noise Ratio (PSNR) index as the error metric. We compare our proposed method with numerous state-of-the-art methods, including BM3D [10], WNNM [17], MLP [4], EPLL [47], TNRD [8], IRCNN [45], DnCNN [44] and NLNET [26]. To ensure a fair comparison, we use the default setting provided by the authors.

3.2. Training Details

The training input to our network is noisy and noise-free patch pairs of size \( 40 \times 40 \) cropped randomly from the BSD400 dataset. Note that there is no overlap between the training, i.e. BSD400 and evaluation, i.e. BSD68 dataset. We also augment the training data with horizontally and vertically flipped versions of the original patches and those rotated at an angle of \( \frac{\pi n}{n} \), where \( n = 1, 2, 3 \). The number of patches are randomly cropped on the fly from the 400 images during training.

We offer two strategies for handling different noise levels. The first one is to train a network for each specific noise level. Alternatively, we train a single blind model for the noise range \([1, 50]\) (similar to [44]) and we refer to this model as Ours-Blind. At each update of training, we construct a batch by randomly selecting noisy patches with noise levels between 1 and 50.

We implement the denoising method in the Caffe framework on Tesla P100 GPUs, and employ the Adam optimization algorithm [24] for training. The initial learning rate was set to \( 10^{-4} \) and the momentum parameter was 0.9. We scheduled the learning rate such that it is halved at every \( 1.5 \times 10^5 \) mini-batches of size 64. We train our network from scratch by a random initialization of the convolution weights according to the method in [19] and a regularization strength, i.e. weight decay, of 0.0001.

3.3. Boosting Denoising Performance

To boost the performance of the trained model, we use the a late fusion strategy as adopted by [37]. During the evaluation, we perform eight types of augmentation (including identity) of the input noisy images \( y \) as \( y_i = \Gamma_i(y) \). From these geometrically transformed images, we estimate corresponding denoised images \( \{ \hat{x}_1, \hat{x}_2, \ldots , \hat{x}_8 \} \) using our model. To generate the final denoised image \( \hat{x} \), we perform the corresponding inverse geometric transform \( \hat{x}_i^{-1} = \Gamma_i^{-1}(\hat{x}_i^t) \) and then take the average of the outputs as \( \hat{x} = 0.125 \sum_{i=1}^{8} \hat{x}_i^t \). Self ensemble is beneficial as it saves training time and have small number of parameters as compared to individually trained eight models. We also found empirically that self ensemble gives approximately the same performance as the models trained individually with geometric transform.

3.4. Ablation Studies

3.4.1 Influence of the patch size

In our network, patch size plays an important role and here we show the influence of patch size on our network. Table 1 shows the average PSNR on BSD68 [22] for \( \sigma_n = 25 \) with respect to the increase in size of the training patch. It is obvious that performance improves as the patch size increases. The main reason for this phenomenon is the size of the receptive field, with a larger patch size network learns more
3.5. Comparisons

3.5.1 Number of modules

We show the effect of the number of modules on denoising results. As mentioned earlier, each module $M$ consists of five convolution layers, by increasing the number of modules, we are making our network deeper. In this settings, all parameters are constant, except the number of modules as shown in Table 2. It is clear from the results that making the network deeper increase the average PSNR. However, since fast restoration is desired, we prefer a small network of five modules i.e. $M = 5$, which achieves better performance than other methods.

3.5.2 Kernel dilation and number of layers

It has been shown that the performance of some networks can be improved either by increasing the depth of the net-
Figure 5. Denoising performance for state-of-the-art versus the proposed method on sample color images from the dataset in [32], where the noise standard deviation $\sigma_n$ is 50. The image we recover is more natural, contains less contrast artifacts and is closest to the ground-truth.

Figure 6. A sample color image with rich textures, selected from the BSD68 dataset [32] for $\sigma_n = 25$. On a magnified view, the image our network recovers is sharper than those generated by most of the methods.

Figure 7. Two real images from [44] denoised by our color blind and grayscale blind models, respectively.

work or by using large convolution filter size to capture the context information [45, 44]. This helps the restoration of noisy structures in the image. The usage of traditional $3 \times 3$ filters is popular in deeper networks. However, using dilated filters there is a tradeoff between the number of layers and the size of the dilated filters. In Table 3, we present the relation between the dilated filter size and the number of layers using three experimental setting. In the first experiment as shown in the first column of Table 3 we use a traditional filter of size $3 \times 3$ and depth of 18 to cover the receptive field of training patch of size 40. In the next experiment, we keep the size of the filter same but enlarge the filter using a dilation factor of two. This increases the size of the filter to $5 \times 5$ but having nine non-zero entries it can be interpreted as a sparse filter. Therefore, the receptive field of the training patch can now be covered by nine non-linear mapping.
Table 4. Performance comparison between image denoising algorithms on widely used classical images, in terms of PSNR (in dB). The best results are highlighted with red color while the blue color represents the second best denoising results.

| Algorithm | Man | House | Peppers | Starfish | Monar | Airpl | Parrot | Lena | Barbara | Boat | Man | Couple | Average |
|-----------|-----|-------|---------|----------|-------|-------|--------|------|---------|------|-----|--------|---------|
| BM3D [10] | 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 [17] | 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 |
| EPLL [47] | 31.85 | 34.17 | 32.64 | 31.13 | 32.10 | 31.19 | 31.42 | 33.92 | 31.38 | 31.93 | 32.00 | 31.93 | 32.138 |
| CSF [56] | 31.95 | 34.39 | 32.85 | 31.55 | 32.33 | 31.33 | 31.37 | 34.06 | 31.92 | 32.01 | 32.08 | 31.98 | 32.318 |
| TNRD [8] | 32.19 | 34.53 | 33.04 | 31.75 | 32.56 | 31.46 | 31.63 | 34.24 | 32.13 | 32.14 | 32.23 | 32.11 | 32.502 |
| DnCNNs [44] | 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 |
| DnCNNb [44] | 32.10 | 34.93 | 33.15 | 32.02 | 32.94 | 31.56 | 31.63 | 34.56 | 32.09 | 32.35 | 32.41 | 32.41 | 32.680 |
| IrCNN [45] | 32.55 | 34.89 | 33.31 | 32.07 | 33.27 | 31.70 | 31.83 | 34.53 | 32.43 | 32.34 | 32.40 | 32.40 | 32.769 |
| Ours-blind | 32.11 | 35.10 | 33.28 | 32.31 | 33.07 | 31.58 | 31.80 | 34.67 | 32.48 | 32.42 | 32.40 | 32.50 | 32.812 |
| Ours | 32.61 | 35.21 | 33.21 | 32.35 | 33.33 | 31.77 | 32.01 | 34.69 | 32.74 | 32.44 | 32.50 | 32.52 | 32.950 |

| Algorithm | $\sigma_n = 25$ | $\sigma_n = 50$ | $\sigma_n = 70$ |
|-----------|----------------|----------------|----------------|
| BM3D [10] | 32.17 | 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 [17] | 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 |
| EPLL [47] | 31.85 | 34.17 | 32.64 | 31.13 | 32.10 | 31.19 | 31.42 | 33.92 | 31.38 | 31.93 | 32.00 | 31.93 | 32.138 |
| CSF [56] | 31.95 | 34.39 | 32.85 | 31.55 | 32.33 | 31.33 | 31.37 | 34.06 | 31.92 | 32.01 | 32.08 | 31.98 | 32.318 |
| TNRD [8] | 32.19 | 34.53 | 33.04 | 31.75 | 32.56 | 31.46 | 31.63 | 34.24 | 32.13 | 32.14 | 32.23 | 32.11 | 32.502 |
| DnCNNs [44] | 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 |
| DnCNNb [44] | 32.10 | 34.93 | 33.15 | 32.02 | 32.94 | 31.56 | 31.63 | 34.56 | 32.09 | 32.35 | 32.41 | 32.41 | 32.680 |
| IrCNN [45] | 32.55 | 34.89 | 33.31 | 32.07 | 33.27 | 31.70 | 31.83 | 34.53 | 32.43 | 32.34 | 32.40 | 32.40 | 32.769 |
| Ours-blind | 32.11 | 35.10 | 33.28 | 32.31 | 33.07 | 31.58 | 31.80 | 34.67 | 32.48 | 32.42 | 32.40 | 32.50 | 32.812 |
| Ours | 32.61 | 35.21 | 33.21 | 32.35 | 33.33 | 31.77 | 32.01 | 34.69 | 32.74 | 32.44 | 32.50 | 32.52 | 32.950 |

7 layers, contrary to the 18-layers depth per module. Similarly, by expanding the filter by a dilation of three would result in the depth of each module to be six. As in Table 3, all three trained models result in similar denoising performance, with the obvious advantage of the shallow network being the most efficient.

In this section, first we demonstrate how our method performs on classical images and then report results on the BSD68 dataset.

### 3.5.3 Classical Images

For completeness, we compare our algorithm to several state-of-the-art denoising methods using grayscale classical images shown in Figure 3 and reported in Table 4.

![Classical Images](image_url)

In Table 4, we present the average PSNR scores for the denoised images. Our network is the best performer for almost all classical images except ‘Barbara’. The reason for this may be the repetitive structures in the mentioned image, which makes it easy for BM3D [10] and WNNM [17] to find and employ patches with great similarity to the noisy input, hence providing better results.

Subsequently, we depict an example from the classical images. The visual quality of our recovered image, as shown in Figure 3, is better than all others. This also illustrates that our network restores aesthetically pleasing text.

In Table 4, we present the average PSNR scores for the denoised images. Our network is the best performer for almost all classical images except ‘Barbara’. The reason for this may be the repetitive structures in the mentioned image, which makes it easy for BM3D [10] and WNNM [17] to find and employ patches with great similarity to the noisy input, hence providing better results.

Subsequently, we depict an example from the classical images. The visual quality of our recovered image, as shown in Figure 3, is better than all others. This also illustrates that our network restores aesthetically pleasing text.
We present the average PSNR scores for the estimated denoised images in Table 5. The IRCNN [45] and DnCNN [44] network structures are similar, hence produce nearly similar results. On the other hand, our method reconstructs the images accurately, achieving higher PSNR than completing methods on all four levels of noise. Furthermore, the difference in PSNR increases between our method and the-state-of-the-art as the noise level increases.

For a comprehensive evaluation, we demonstrate the visual results on a selected grayscale image from BSD68 [32] dataset in Figure 4. In our results, the image details are more similar to the ground-truth details, and our quantitative results are numerically higher than the others. Our method leads the second best method by several orders of magnitude (PSNR is computed in the logarithmic scale). Also, note that the denoising results of other CNN based algorithms are comparable to each other. This experiment indicates that our network is well-suited for real-world applications.

3.6. Color Image Denoising

For noisy color images, we train our network with the noisy RGB input patches of size 40×40 with the corresponding clean ground-truth patches. We only modify the last convolution layer of the grayscale network to output three channels instead of one channel, keeping all other parameters same as the grayscale network. This is very convenient for hardware implementations in real applications. Small and noticeable features restored by our network include the sharpness and the clarity of the subtle textures around the fore and hind wings, mouth, and antennas of the butterfly. Furthermore, a magnified view of the results in Figures 5 for methods [10, 45, 26] shows artifacts and failures in the smooth areas. Our CNN network also outperforms [45, 26, 44], which are trained using deep neural networks.

3.7. Real-world noisy images

As a last experiment, we demonstrate the performance of our network on real-world noisy images. Figure 7 shows such examples denoised by our blind (requiring no noise prior) denoising models. As visible, the details are preserved properly and the noise is removed effectively. When the noise is AWGN or adequately satisfies the criteria for additive Gaussian-like noise criteria, our model works accurately. This experiment indicates that our network is well-suited for real-world applications.

4. Conclusions

To sum up, we employ residual learning and identity mapping to predict the denoised image using a five-module and five-layer deep network of 26 weight layers with dilated convolutional filters without batch normalization. Our choice of network is based on the ablation studies performed in the experimental section of this paper.

This is the first modular framework to predict the denoised output without any dependency on the pre- or post-processing. Our proposed network removes the potentially authentic image structures while allowing the noisy observations to go through its layers, and learns the noise patterns to estimate the clean image. In future, our aim is to generalize our denoising network to other image restoration tasks.
References

[1] S. Anwar, F. Porikli, and C. P. Huynh. Category-specific object image denoising. *IEEE Transactions on Image Processing*, 26(11):5506–5518, 2017.
[2] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 1994.
[3] A. Buades, B. Coll, and J.-M. Morel. A non-local algorithm for image denoising. In *CVPR*, pages 60–65, 2005.
[4] H. C. Burger, C. J. Schuler, and S. Harmeling. Image denoising: Can plain neural networks compete with bm3d? In *CVPR*, pages 2392–2399, 2012.
[5] S. H. Chan, T. Zickler, and Y. M. Lu. Monte carlo non-local means: Random sampling for large-scale image filtering. *TIP*, pages 3711–3725.
[6] P. Chatterjee and P. Milanfar. Is denoising dead? *Image Processing, IEEE Transactions on*, pages 895–911, 2010.
[7] F. Chen, L. Zhang, and H. Yu. External patch prior guided internal clustering for image denoising. 2015.
[8] Y. Chen and T. Pock. Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1256–1272, 2017.
[9] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Color image denoising via sparse 3-D collaborative filtering with grouping constraint in luminance-chrominance space. In *ICIP*, 2007.
[10] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3-D transform-domain collaborative filtering. *Image Processing, IEEE Transactions on*, pages 2080–2095, 2007.
[11] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. BM3D image denoising with shape-adaptive principal component analysis. In *Signal Processing with Adaptive Sparse Structured Representations*, 2009.
[12] W. Dong, X. Li, D. Zhang, and G. Shi. Sparsity-based image denoising via dictionary learning and structural clustering. In *CVPR*, pages 457–464, June 2011.
[13] M. Elad and D. Datsenko. Example-based regularization deployed to super-resolution reconstruction of a single image. *Comput. J.*, pages 15–30, 2009.
[14] L. Z. F. Chen and H. Yu. External Patch Prior Guided Internal Clustering for Image Denoising. In *ICCV*, pages 1211–1218, 2015.
[15] A. Foi, V. Katkovnik, and K. Egiazarian. Pointwise shape-adaptive DCT for high-quality denoising and deblocking of grayscale and color images. *IEEE transactions on image processing*, pages 1395–1411, 2007.
[16] B. Goossens, H. Luong, A. Pizurica, and W. Philips. An improved non-local denoising algorithm. In *Local and Non-Local Approximation in Image Processing, International Workshop, Proceedings*, page 143, 2008.
[17] S. Gu, L. Zhang, W. Zuo, and X. Feng. Weighted nuclear norm minimization with application to image denoising. In *CVPR*, pages 2862–2869, 2014.
[18] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.
[19] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *CoRR*, abs/1502.01852, 2015.
[20] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
[21] K. He, X. Zhang, S. Ren, and J. Sun. Identity Mappings in Deep Residual Networks, pages 630–645. 2016.
[22] H. Hirschmüller and D. Scharstein. Evaluation of cost functions for stereo matching. In *CVPR*, pages 1–8, 2007.
[23] J. Kim, J. Kwon Lee, and K. Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *CVPR*, 2016.
[24] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.
[25] M. Lebrun, A. Buades, and J.-M. Morel. A nonlocal bayesian image denoising algorithm. *SIAM Journal on Imaging Sciences*, pages 1665–1688, 2013.
[26] S. Lefkimmiatis. Non-local color image denoising with convolutional neural networks. *CVPR*, 2016.
[27] A. Levin and B. Nadler. Natural image denoising: Optimality and inherent bounds. In *CVPR*, pages 2833–2840, 2011.
[28] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee. Enhanced deep residual networks for single image super-resolution. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2017.
[29] T. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie. Feature pyramid networks for object detection. *CoRR*, abs/1612.03144, 2016.
[30] E. Luo, S. H. Chan, and T. Q. Nguyen. Adaptive image denoising by targeted databases. *Image Processing, IEEE Transactions on*, pages 2167–2181, 2015.
[31] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Non-local sparse models for image restoration. In *ICCV*, pages 2272–2279, 2009.
[32] D. Martin, C. Fowlkes, D. Tal, and 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*, pages 416–423, 2001.
[33] S. Osher, M. Burger, D. Goldfarb, J. Xu, and W. Yin. An iterative regularization method for total variation-based image restoration. *Multiscale Modeling & Simulation*, pages 460–489, 2005.
[34] Y. Peng, A. Ganesh, J. Wright, W. Xu, and Y. Ma. RASL: Robust alignment by sparse and low-rank decomposition for linearly correlated images. *TPAMI*, pages 2233–2246, 2012.
[35] S. Roth and M. J. Black. Fields of experts. *International Journal of Computer Vision*, 82(2):205–229, 2009.
[36] U. Schmidt and S. Roth. Shrinkage fields for effective image restoration. In *CVPR*, pages 2774–2781, 2014.
[37] R. Timofte, R. Rothe, and L. Van Gool. Seven ways to improve example-based single image super resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1865–1873, 2016.
[38] Y. Weiss and W. T. Freeman. What makes a good model of natural images? In *Computer Vision and Pattern Recognition, 2007. CVPR’07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[39] J. Xu and S. Osher. Iterative regularization and nonlinear inverse scale space applied to wavelet-based denoising. *Image Processing, IEEE Transactions on*, pages 534–544, 2007.

[40] J. Xu, L. Zhang, W. Zuo, D. Zhang, and X. Feng. Patch Group Based Nonlocal Self-Similarity Prior Learning for Image Denoising. In *ICCV*, pages 1211–1218, 2015.

[41] L. Xu, L. Zhang, W. Zuo, D. Zhang, and X. Feng. Patch group based nonlocal self-similarity prior learning for image denoising. 2015.

[42] H. Yue, X. Sun, J. Yang, and F. Wu. Cid: Combined image denoising in spatial and frequency domains using web images. In *CVPR*, pages 2933–2940, June 2014.

[43] H. Yue, X. Sun, J. Yang, and F. Wu. Image denoising by exploring external and internal correlations. *TIP*, pages 1967–1982, 2015.

[44] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Transactions on Image Processing*, 2017.

[45] K. Zhang, W. Zuo, S. Gu, and L. Zhang. Learning deep cnn denoiser prior for image restoration. *CVPR*, 2017.

[46] Q. W. Y. B. Zhiyuan Zha, Xinggan Zhang and L. Tang. Group sparsity residual constraint for image denoising. In *arXiv preprint arXiv:1703.00297*, 2017.

[47] D. Zoran and Y. Weiss. From learning models of natural image patches to whole image restoration. In *ICCV*, pages 479–486, 2011.