

Beyond Deep Residual Learning for Image Restoration: Persistent Homology-Guided Manifold Simplification

Woong Bae*, Jaewon Yoo*, and Jong Chul Ye

Bio Imaging Signal Processing Lab

Korea Ad. Inst. of Science & Technology (KAIST)
291 Daehak-ro, Yuseong-gu, Daejeon 34141, Korea

{iorism, jaewon2004, jong.ye}@kaist.ac.kr

* denotes co-first authors

Abstract

The latest deep learning approaches perform better than the state-of-the-art signal processing approaches in various image restoration tasks. However, if an image contains many patterns and structures, the performance of these CNNs is still inferior. To address this issue, here we propose a novel feature space deep residual learning algorithm that outperforms the existing residual learning. The main idea is originated from the observation that the performance of a learning algorithm can be improved if the input and/or label manifolds can be made topologically simpler by an analytic mapping to a feature space. Our extensive numerical studies using denoising experiments and NTIRE single-image super-resolution (SISR) competition demonstrate that the proposed feature space residual learning outperforms the existing state-of-the-art approaches. Moreover, our algorithm was ranked third in NTIRE competition with 5-10 times faster computational time compared to the top ranked teams. The source code is available on page : <https://github.com/iorism/CNN.git>

1. Introduction

Image restoration tasks such as denoising and super-resolution are essential steps in many practical image processing applications. Over the last few decades, various algorithms have been developed, which include non-local self-similarity (NSS) models [2], total variation (TV) approaches [22], and sparse dictionary learning models [9]. Among them, the block matching 3D filter (BM3D) [7] is considered as the state-of-the art algorithm. In general, these methods are dependent on the noise model. Moreover, these algorithms are usually implemented in an iterative manner, so they require significant computational resources.

Recently, deep learning approaches have achieved tremendous success in classification [19] as well as low-level computer vision problems [24]. In image denoising and super-resolution tasks, many state-of-the-art CNN algorithms [4, 5, 21, 32] have been proposed. Although the performance of these algorithms usually outperforms the non-local and collaboration filtering approaches such as BM3D, in case of certain images that have many patterns (such as Barbara image), CNN approaches are still inferior to BM3D.

Therefore, one of the main motivations of this work is to develop a new CNN architecture that overcomes the limitation of the state-of-the-art CNN approaches. The proposed network architectures are motivated from a novel *persistent homology* analysis [11] on residual learning for image processing tasks. Specifically, we show that the residual manifold is topologically simpler than the original image manifold, which may have attributed the success of residual learning. Moreover, this observation leads us to a new network design principle using *manifold simplification*. Specifically, our design goal is to find mappings for input and/or label datasets to feature spaces, respectively, such that the new datasets become topologically simpler and easier to learn. In particular, we show that the wavelet transform provides topologically simpler manifold structures while preserving the directional edge information.

Contribution: In summary, our contributions are as following. First, a novel network design principle using manifold simplification is proposed. Second, using a recent computational topology tool called the persistent homology, we show that the existing residual learning is a special case of manifold simplification and then propose a wavelet transform to simplify topological structures of input and/or label manifolds.

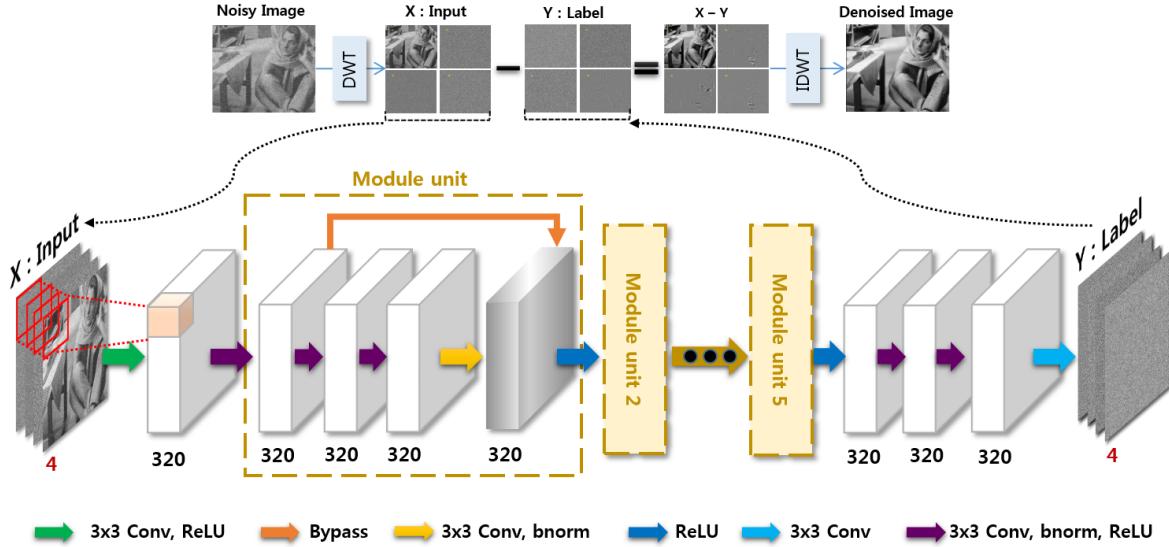


Figure 1. Proposed wavelet domain deep residual learning network for the Gaussian denoising task.

	Bicubic x2 (256ch)	Bicubic x3,x4 (320ch)	Unknown x2,x3,x4 (320ch)
Input	WT(BU(LR))	COPY.ch(LR)	
Label	Input - WT(HR)	Input - PS(HR)	
1st layer	Conv → ReLU	Conv → ReLU	
2nd layer	Conv → BN → ReLU	Conv → BN → ReLU	
Long bypass layer	LongBypass(1)	-	-
1st module	BypassM1 → Conv → BN → ReLU → Conv → BN → ReLU → Conv → BN → SumF(BypassM) → ReLU	BypassM1 → Conv → BN → ReLU → Conv → BN → ReLU → Conv → BN → SumF(BypassM) → ReLU	BypassM1 → Conv → BN → ReLU → Conv → BN → ReLU → Conv → BN → SumF(BypassM) → ReLU
Repeat 1st module	5 times (2th~6th module)	11 times (2th~12th module)	12 times (2th~12th module)
Long bypass & catch layer	Sum of “LongBypass(1)” and “Output of 6th module” → BN → ReLU	-	-
Long bypass layer	LongBypass(2)	-	-
Repeat 1st module	6 times (7th~12th modules)	-	-
Long bypass & catch layer	Sum of “LongBypass(2)” and “Output of 12th module” → BN → ReLU	-	-
Last layer	Conv → BN → ReLU → Conv → BN → ReLU → Conv	Conv → BN → ReLU → Conv → BN → ReLU → Conv	Conv → BN → ReLU → Conv → BN → ReLU → Conv
Restoration	IWT(Input-Output)		IPS(Input-Output)

* **WT**: Haar Wavelet Transform, **BU**: Bicubic Upsampling, **LR**: Low Resolution image, **HR**: High Resolution image, **Conv**: 3×3 Convolution, **BN**: Batch Normalization, **BypassM**: Sending output of previous layer to last layer of module(M is module number), **SumF**: Sum of output of previous layer and BypassM output, **COPY.ch**: Copy input image (scale \times scale) times on channel direction, **PS**: sub-Pixel Shuffling, **IPS**: Inverse sub-Pixel Shuffling, **IWT**: Inverse Wavelet Transform

Table 1. Proposed network architectures for NTIRE SISR competition from bicubic and unknown downsampling schemes.

2. Related Work

One of the classical approaches for image denoising is a wavelet shrinkage approach [10], which decomposes an image into low and high frequency subbands and applies thresholding in the high frequency coefficients [23]. Advanced algorithms in this field are to exploit the intra- and inter- correlations of the wavelet coefficients [6].

In neural network literature, the work by Berger et al [3] was the first which demonstrated similar denoising performance to BM3D using multi-layer perceptron (MLP). Chen *et al.* [4, 5] proposed a deep learning approach called trainable nonlinear reaction diffusion (TNRD) that can train filters and influence functions by unfolding a variational optimization approach. Recently, based on skipped connec-

tion and encoder-decoder architecture, a very deep residual encoder-decoder networks (RED-Net) was proposed for image restoration problems [21].

Residual learning has multiple realizations. The first approach is using a skipped connection that bypasses input data of a certain layer to another layer during forward and backward propagations. This type of residual learning concept was first introduced by He *et al.* [13] for image recognition. In low-level computer vision problems, Kim *et al.* [18] employed a residual learning for a super-resolution method. In these approaches, the residual learning was implemented by a skipped connection corresponding to an identity mapping. In another implementation, the label data is transformed into the difference between the input data and clean data. For example, Zhang *et al.* [32] proposed a denoising convolutional neural networks (DnCNNs) [32], which has inspired our method.

3. Theory

3.1. Generalization bound

Let $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$ denote the input and label data and $f : X \rightarrow Y$ denotes a function living in a functional space \mathcal{F} . Then, one is interested in the minimization problem: $\min_{f \in \mathcal{F}} L(f)$, where $L(f) = E_D \|Y - f(X)\|^2$ denotes the risk. A major technical issue is, however, that the associated probability distribution D is unknown. Thus, an upper bound is used to characterize the generalization performance. Specifically, with probability $\geq 1 - \delta$ with a small $\delta > 0$, for every function $f \in \mathcal{F}$,

$$L(f) \leq \underbrace{\hat{L}_n(f)}_{\text{empirical risk}} + \underbrace{2\hat{R}_n(\mathcal{F})}_{\text{complexity penalty}} + 3\sqrt{\frac{\ln(2/\delta)}{n}} \quad (1)$$

where $\hat{R}_n(\mathcal{F})$ denotes the Rademacher complexity [1].

In neural network, empirical risk is determined by the representation power or capacity of a network [27], and the complexity penalty is determined by the structure of a network. It was shown that the capacity of representation power grows exponentially with respect to the number of layers [27], which justifies the use of a *deep* network compared to shallow ones. However, the complexity penalty in (1) also increases with a complicated network structure. The main remedy for this trade-off is to use large number of training dataset such that the contribution of the complexity penalty reduces much more quickly so that the empirical risk minimization (ERM) converges consistently to the risk minimization [28].

However, for the intermediate size of the training data, there still exist gaps between the ERM and the risk minimization. One of the most important contributions of this paper is to reduce the gap by using a relatively simpler network, by reducing the complexity of the data manifold.

Specifically, for a given deep network $f : X \rightarrow Y$, our design goal is to find mappings Φ and Ψ to the *feature spaces* for the input and label datasets, respectively. Then, the resulting datasets composed of $X' = \Phi(X)$ and $Y' = \Psi(Y)$ may have simpler manifold structures. This can be shown in the following diagram:

$$\begin{array}{ccc} X & \xrightarrow{f} & Y \\ \Phi \downarrow & \Phi^{-1} & \downarrow \Psi \\ X' & \xrightarrow{g} & Y' \end{array}$$

from which our goal is to find an equivalent neural network $g : X' \rightarrow Y'$ that has a better performance than the original network $f : X \rightarrow Y$.

For example, in recent deep residual learning [32], the input transform T is an identity mapping and the label transform is given by $Y' = \Psi(Y) = Y - X$. Using persistent homology analysis, Section 1 in the supplementary material shows that the label manifold of the residual is topologically simpler than that of Y . Accordingly, the upper bound of the risk of $g : X' \rightarrow Y'$ can be reduced compared to that of $f : X \rightarrow Y$.

Inspired by this finding, this paper proposes a wavelet transform as a good transform to reduce the topological complexity of resulting input and label manifolds. More specifically, thanks to the vanishing moments of wavelets, the wavelet transform can annihilate the smoothly varying signals while retaining the image edges [8, 20], which results in the dimensional reduction and manifold simplification. Indeed, this property of the wavelet transform has been extensively exploited in wavelet-based image compression tools such as JPEG2000 [26], and this paper shows that this property also improves the performance of deep network for image restoration tasks.

3.2. Persistent homology

The complexity of a manifold is a topological concept. Thus, it should be analyzed using topological tools. In algebraic topology, Betti numbers (β_m) represent the number of m -dimensional holes of a manifold. For example, β_0 and β_1 are the number of connected components and cycles, respectively. They are frequently used to investigate the characteristics of underlying manifolds [11]. Specifically, we can infer the topology of our data manifold by varying the similarity measure between the data points and tracking the changes of Betti numbers. As allowable distance ϵ increases, data point clouds merge together and finally become a single cluster (Fig. 2(a)). Therefore, the point clouds with high diversity will merge slowly and this will be represented as a slow decrease in Betti numbers. For example, in Fig. 2(a), the dataset $Y1$ is a doughnut with a hole (i.e. $\beta_0 = 1$ and $\beta_1 = 1$) whereas $Y2$ is a sphere-like cluster

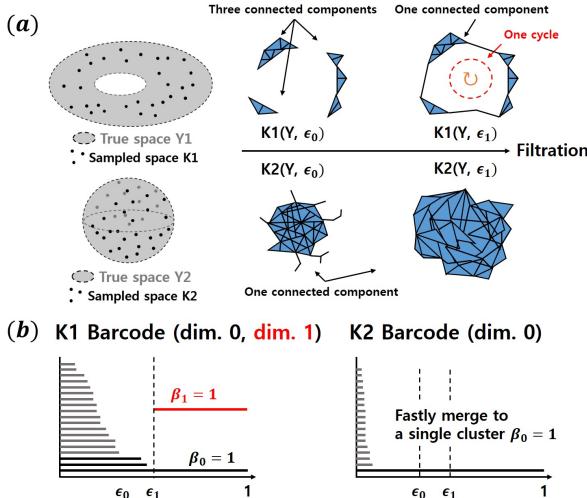


Figure 2. (a) Point cloud data K of true space Y and its configuration over ϵ distance filtration. Y_1 is a doughnut and Y_2 is a sphere shaped space each of which represents a complicated space and a simpler space, respectively. (b) Zero and one dimensional barcodes of K_1 and K_2 . Betti number can be easily calculated by counting the number of barcodes at each filtration value ϵ .

(i.e. $\beta_0 = 1$ and $\beta_1 = 0$). Accordingly, Y_1 has longer zero dimensional *barcodes* persisting over ϵ in Fig. 2(b). This persistence of Betti number is an important topological feature and the recent *persistent homology* analysis utilizes this to investigate the topology of the data manifold[11].

4. Proposed architecture

This section describes two network structures based on the manifold simplification. One is the primary architecture used for Gaussian denoising and the other is our NTIRE 2017 competition architecture used for RGB based SISR problems, which has been developed based on the primary architecture. For the wavelet transform, we used one level discrete wavelet transform using Haar wavelet filter.

Denoising architecture: The input and the clean label images are first decomposed into four subbands (i.e. LL, LH, HL, and HH) using the wavelet transform. The wavelet residual images, which are now used as our new labels, are obtained by the difference between the input and the clean label images in the wavelet domain. Then, the network is trained to learn multi-input and multi-output functional relationship between these newly processed input and label. Here, four patches at the same locations in each wavelet subband are extracted and used for training. For Gaussian denoising, 40×40 image patches are used, resulting in $40 \times 40 \times 4$ patches.

The proposed denoising network architecture is shown in Fig. 1. It consists of five modules between the first and the last stages. Each module has one bypass connection,

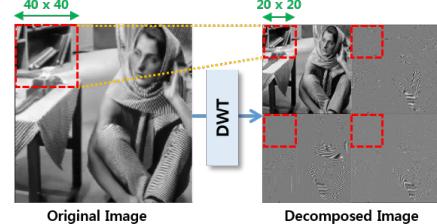


Figure 3. Wavelet decomposition reduces the patch size to a quarter.

three convolution layers, three batch normalizations [17], and three Rectified Linear Unit (ReLU) [12] layers. The bypass connection was used for an efficient network training because it is helpful for training a deep network by alleviating the gradient vanishing problem [15, 21]. The first stage contains two layers: one with a convolution layer with ReLU which is followed by the other convolution layer with batch normalization and ReLU. The last stage is composed of three layers: two layers with a convolution, batch normalization, and ReLU and the last layer with a convolution layer. Accordingly, the total number of convolution layers is 20. The convolution filter size is $3 \times 3 \times 320 \times 320$. During the convolution, we used zero padding to maintain the image size and reduce the boundary effect [18].

In addition to the aforementioned advantage of the wavelet transform for feature space mapping, there are two more advantages to perform the wavelet transform. As shown in Fig.3, the first advantage is that the patch size can be reduced by half. It can reduce the runtime of the network due to the size of the output images of layers being halved. The second one is that the minimum required size of receptive field can be reduced to obtain a good performance. Since the number of convolutions is related to the runtime and learning time, the smaller the required receptive field size, the more effective it is to reduce the computation time.

NTIRE SISR competition architecture: The proposed networks for NTIRE 2017 SISR competition are shown in Table 1. These architectures are extended from the primary denoising architecture. Depending on the decimation schemes (bicubic x2, x3, x4, and unknown x2, x3, and x4) for low resolution dataset, we implemented three different architectures.

Specifically, for the bicubic cases, we first generated the upsampled image using bicubic interpolation and the extended denoising network structures with the wavelet transform were used for manifold simplification. For the unknown decimation scheme, however, we employed the sub-pixel shuffling scheme [25] as the input and label transform to save the memory and augment the input data to a bigger image size. As will be shown later in persistent homology analysis in the supplementary material, this sub-pixel shuffling transform does not reduce the manifold complexity by itself. Still, we could exploit the manifold simplification

from the residual learning in sub-pixel shuffling domain as shown in Fig. 4 and Table 2.

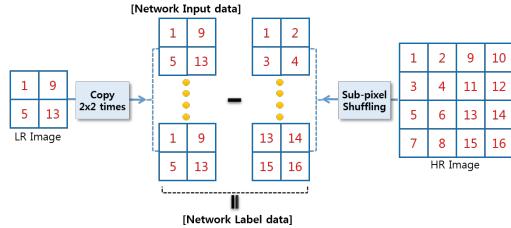


Figure 4. Residual based sub-pixel shuffling.

Label image set	Original	Residual
Unknown x3	29.1242 / 0.8327	30.3025 / 0.8572

Table 2. The effectiveness of the residual based sub-pixel shuffling in terms of PSNR/SSIM for ‘‘Unknown x3’’ dataset in the super-resolution task. The training step was stopped at epoch 50 and the results are calculated from 100 validation data of DIV2K dataset.

All three SISR architectures have 41 convolution layers to deal with the dataset composed of 800 high resolution images. In every case, we used 20×20 patch size. After the first two layers, a basic module is repeated twelve times, which is followed by three convolution layers. To reconstruct the bicubic x2 downsampled dataset, we included two long bypass connections between six basic modules in the network and the number of channels were 256. For the other datasets, we did not use the long bypass connection and the number of channels were 320. The long bypass connection allows faster computation and less parameter size than using the concatenation layer. Although concatenation layer is good for reducing the depth of convolution layer, it is very slow because of inefficient GPU memory usage. Thus, the long bypass connection is more efficient for SISR problem. Table 3 shows the effectiveness of the long bypass connection.

We used RGB data for different channels rather than luminance channel, because RGB based learning has the effect of data augmentation.

Problem	Without LongBypass	LongBypass
Bicubic x2	35.3436 / 0.9426	35.3595 / 0.9427

Table 3. The effectiveness of the long bypass layer in terms of PSNR/SSIM. This result was calculated from 50 validation data of DIV2K dataset.

5. Methods

5.1. Dataset

Dataset for denoising network: We used publicly available Berkeley segmentation (BSD500) [4] and Urban100

[16] datasets. Specifically, we used 400 images of BSD500 and Urban100 datasets for training in the Gaussian denoising task. In addition, we generated 4000 training images by using data augmentation via image flipping, rotation, and cropping. To get various noise patterns and avoid overfitting, we re-generated the Gaussian noise in every other epoch during training. For the test dataset, BSD68 and Set12 datasets were used. All the images were encoded with eight bits, so the pixel values are within $[0, 255]$. For training and validation, Gaussian noises with $\sigma = 15, 30$, and 50 were added.

Dataset for NTIRE competition: We used only 800 training dataset of DIV2K for each SISR problem. Instead of using cropped images, we cropped (20×20) patches randomly from a full image and performed data augmentation using image flipping, rotation, and downsampling with the corresponding scale factors of $x2, x3$, and $x4$ for each epoch. It helps to create more diverse patterns of images. For training dataset with bicubic x3 and x4 decimation, we used all bicubic x2,x3 and x4 images of DIV2K datasets together as a kind of data augmentation.

5.2. Network training

Denoising network: The network parameters were initialized using the Xavier method [14]. We used the regression loss across four wavelet subbands under l_2 penalty and the proposed network was trained by using the stochastic gradient descent (SGD). The regularization parameter (λ) was 0.0001 and the momentum was 0.9. The learning rate was set from 10^{-1} to 10^{-4} which was reduced in log scale at each epoch. The mini-batch size for batch normalization was 32 where the images were selected randomly at every epoch [17]. To use a high learning rate and guarantee a stable learning, we employed the gradient clipping technique [18] so that the maximum and minimum values of the update parameter are bounded by the predefined range. These parameter settings were equally applied to all experiments of the image denoising. We used 40×40 patch size and the network was trained using 53 epochs.

The network was implemented using MatConvNet toolbox (beta.20) [29] in MATLAB 2015a environment (Math-Works, Natick). We used a GTX 1080 graphic processor and i7-4770 CPU (3.40GHz). The Gaussian denoising network took about two days for training.

Training for NTIRE competition: We used 20×20 patch size and trained the network for 150 epochs. We used 64 mini-batch size and learning rate of $(0.1, 0.00001)$ in log scale for 150 epochs with 0.05 gradient clipping factor. For each epoch, to train more various patterns, we used sub-epoch system that repeats forward and back propagation 512 times by randomly cropping patches from a full size image. For the bicubic x3 and x4 cases, we trained the network with the bicubic x2, x3, and x4 datasets together to in-

Images	C.man	House	Peppers	Starfish	Monar.	Airpl.	Parrot	Lena	Barbara	Boat	Man	Couple	Average
Algorithm	Noise Level: $\sigma = 30$												
BM3D	28.6376	32.1417	29.2140	27.6354	28.3458	27.4857	28.0707	31.2388	29.7894	29.0465	28.8016	28.8417	29.1041
DnCNN-S	29.2748	32.3199	29.8497	28.3970	29.3165	28.1570	28.5375	31.6104	28.8925	29.3117	29.2492	29.2091	29.5105
Proposed(Primary)	29.6219	32.9357	30.1054	29.0584	29.5597	28.3288	28.6770	32.0163	29.8941	29.6107	29.4065	29.5563	29.8976

Table 4. Performance comparison in terms of PSNR for “Set12” dataset in the Gaussian denoising task. The primary architecture was used.

Images	C.man	House	Peppers	Starfish	Monar.	Airpl.	Parrot	Lena	Barbara	Boat	Man	Couple	Average
Algorithm	Noise Level: $\sigma = 30$												
BM3D	0.8373	0.8480	0.8502	0.8282	0.8866	0.8361	0.8320	0.8456	0.8673	0.7777	0.7783	0.7937	0.8318
DnCNN-S	0.8580	0.8515	0.8670	0.8478	0.9032	0.8544	0.8425	0.8559	0.8514	0.7841	0.7949	0.8029	0.8428
Proposed(Primary)	0.8662	0.8589	0.8729	0.8604	0.9116	0.8584	0.8459	0.8669	0.8795	0.7978	0.8022	0.8189	0.8533

Table 5. Performance comparison in terms of SSIM for “Set12” dataset in the Gaussian denoising task. The primary architecture was used.

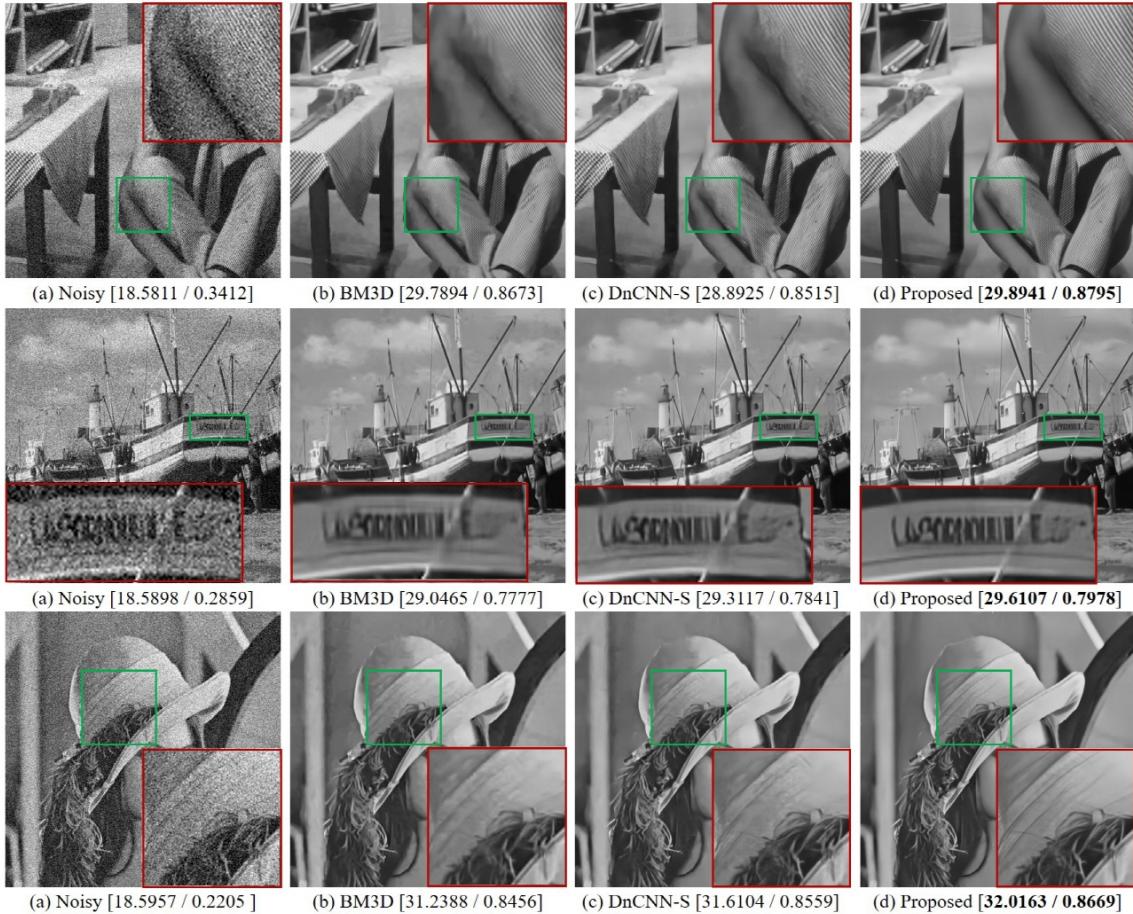


Figure 5. Denoising results of Barbara, Boats, and Lena images using various methods. [PSNR/SSIM] values are displayed.

crease the performance of x3 and x4 cases [18]. Other hyper parameters are remained same with the denoising network. Using GTX 1080ti, the training of networks for the bicubic x2 and bicubic x3, x4 and unknown x2, x3, x4 datasets took almost six days, 21 days and seven days, respectively.

6. Results

6.1. Persistent homology results

To show the correlation between the network performance and manifold simplification, we compared the topology of the input and the label manifolds in both image and wavelet domains. The results in the supplementary material clearly showed that feature space mappings provide simpler

data manifolds. Specifically, the proposed denoising and super-resolution algorithms can be benefited from simpler *input* manifold from a feature space mapping using wavelet transform as well as additional simpler *label* manifold from residual learning.

6.2. Experimental Results

Denoising: For the quantitative comparison of the denoising performance, we used the objective measures such as the peak signal to noise ratio (PSNR) and the structural similarity index measure (SSIM) [31]. Table 4, 5 show that the proposed network outperforms the state-of-the-art denoising methods in terms of PSNR and SSIM for all Set12 images. Especially, in the patterned images such as Barbara and House, we attained better performance than using BM3D in terms of PSNR (0.1dB and 0.8dB, respectively). Fig. 5 shows the denoising examples in various images. The proposed methods showed the best visual quality especially in the edge regions. Moreover, as shown in Table 6, the proposed method showed superior results to the state-of-the-art approaches in the experiments with BSD68 dataset which contains diverse patterned images. For 512x512 image size, the proposed network took only 0.157 seconds even with the current MATLAB implementation. This is comparable or even better than the existing approaches.

To further demonstrate the importance of the wavelet decomposition, additional comparative studies with the baseline network were performed. Here, an input image is decomposed to four channels using so-called *polyphase quadrature filter (PQF)* bank [30]. Specifically, the PQF just splits an input image into four equidistant sub-bands with distinct horizontal and vertical offset without wavelet filtering. Therefore, it is equivalent to the sub-pixel shuffling scheme in [25] except that input image is first interpolated. Accordingly, the networks using PQF have the exactly same architecture except the input and label images so that we can investigate the effect of the wavelet transform. Fig. 6 clearly shows that the wavelet transform can improve the performance compared to the baseline network.

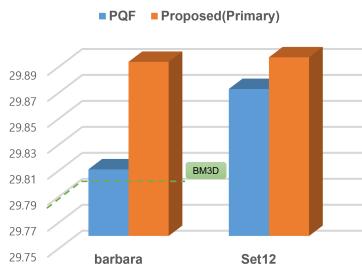


Figure 6. Importance of the wavelet transform for manifold simplification. Here, the Gaussian denoising algorithms with $\sigma = 30$. Barbara and Set12 images were used. In the case of BM3D, Barbara image was used for comparison.

Noise (σ)	BM3D	DnCNN-S	Proposed
15	31.0761/0.8722	31.7202/0.8901	31.8607/0.8941
30	27.7492/0.7735	28.3324/0.8003	28.5599/0.8092
50	25.6103/0.6868	26.2275/0.7163	26.3577/0.7270

Table 6. Performance comparison in terms of PSNR/SSIM for “BSD68” dataset in the Gaussian denoising task.

Dataset (scale)	VDSR	DnCNN-3	Proposed-P	Proposed
Set5 (2)	37.53/0.9586	37.53/0.9582	37.57/0.9586	38.06/0.9602
Set14 (2)	33.03/0.9124	33.08/0.9126	33.09/0.9129	34.04/0.9205
BSD100 (2)	31.90/0.8960	31.90/0.8956	31.92/0.8965	32.26/0.9006
Urban100 (2)	30.76/0.9140	30.75/0.9134	30.96/0.9169	32.63/0.9330
Average (2)	33.30/0.9202	33.31/0.9199	33.39/0.9212	34.25/0.9286
Set5 (3)	33.66/0.9213	33.73/0.9212	33.86/0.9228	34.45/0.9272
Set14 (3)	29.77/0.8314	29.83/0.8321	29.88/0.8331	30.56/0.8450
BSD100 (3)	28.82/0.7976	28.84/0.7976	28.86/0.7987	29.18/0.8071
Urban100 (3)	27.14/0.8279	27.15/0.8272	27.28/0.8334	28.50/0.8587
Average (3)	29.85/0.8445	29.89/0.8445	29.97/0.8470	30.67/0.8595
Set5 (4)	31.35/0.8838	31.40/0.8837	31.52/0.8864	32.23/0.8952
Set14 (4)	28.01/0.7674	28.07/0.7681	28.11/0.7699	28.80/0.7856
BSD100 (4)	27.29/0.7251	27.29/0.7247	27.32/0.7266	27.66/0.7380
Urban100 (4)	25.18/0.7524	25.21/0.7518	25.36/0.7614	26.42/0.7940
Average (4)	27.95/0.7821	27.99/0.7820	28.08/0.7861	28.78/0.8032

Table 7. Performance comparison in terms of luminance PSNR/SSIM for various datasets in SISR tasks. The VDSR, DnCNN-3, Proposed-P(primary) networks are 291 dataset[18] luminance-trained network whereas the proposed network is trained using RGB of DIV2K dataset. For fair comparison, restored RGB was used to calculate the luminance PSNR/SSIM values.

NTIRE SISR competition: Our proposed networks were ranked third in NTIRE SISR competition. In particular, our networks were competitive in both performance and speed. More specifically, compared to the 14-67 seconds computational time by the top ranked groups’ results, our computational time was only 4-5 seconds for each frame. Since the wavelet transform is effective in reducing the manifold complexity compared to the sub-pixel shuffling scheme, in terms of dataset specific ranking, our ranking with the bicubic SISR dataset was better than with the unknown decimation dataset where we just exploited the manifold simplification from the residual learning.

Figs. 7, 8 clearly show the performance of our network in various SISR problems. We also confirmed that our network outperforms the existing state-of-the-art CNN approaches for various dataset in Table 7. In particular, the proposed methods exhibited outstanding performance in the edge areas. We provide more comparative examples in the supplementary material.

7. Conclusion

In this paper, we proposed a feature space deep residual learning algorithm that outperforms the existing residual learning approaches. In particular, using persistent homology analysis, we showed that the wavelet transform and/or



Figure 7. Performance comparison of SISR at scale factor of 4 of bicubic downsampling. The proposed network is RGB based competition network. Left : input, Center : restoration result, Right : label.



Figure 8. Performance comparison of SISR at scale factor of 4 of *unknown* downsampling. The proposed network is RGB based competition network. Left : input, Center : restoration result, Right : label.

residual learning results in simpler data manifold. This finding as well as the experimental results of Gaussian denoising and NTIRE SISR competition results confirmed that the proposed approach is quite competitive in terms of performance and speed. Moreover, we believe that the persistent homology-guided manifold simplification provides a novel design tool for general deep learning networks.

8. Acknowledgement

This work is supported by Korea Science and Engineering Foundation, Grant number NRF-2013M3A9B2076548.

References

- [1] P. L. Bartlett and S. Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002. 3
- [2] A. Buades, B. Coll, and J.-M. Morel. Nonlocal image and movie denoising. *International Journal of Computer Vision*, 76(2):123–139, 2008. 1
- [3] H. C. Burger, C. J. Schuler, and S. Harmeling. Image denoising: Can plain neural networks compete with BM3D? In *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2392–2399. IEEE, 2012. 2

- [4] Y. Chen and T. Pock. Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration. *arXiv preprint arXiv:1508.02848*, 2015. 1, 2, 5
- [5] Y. Chen, W. Yu, and T. Pock. On learning optimized reaction diffusion processes for effective image restoration. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5261–5269, 2015. 1, 2
- [6] M. S. Crouse, R. D. Nowak, and R. G. Baraniuk. Wavelet-based statistical signal processing using hidden Markov models. *IEEE Transactions on Signal Processing*, 46(4):886–902, 1998. 2
- [7] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3-D transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8):2080–2095, 2007. 1
- [8] I. Daubechies et al. *Ten lectures on wavelets*, volume 61. SIAM, 1992. 3
- [9] W. Dong, L. Zhang, G. Shi, and X. Li. Nonlocally centralized sparse representation for image restoration. *IEEE Transactions on Image Processing*, 22(4):1620–1630, 2013. 1
- [10] D. L. Donoho. De-noising by soft-thresholding. *IEEE Transactions on Information Theory*, 41(3):613–627, 1995. 2
- [11] H. Edelsbrunner and J. Harer. Persistent homology—a survey. *Contemporary Mathematics*, 453:257–282, 2008. 1, 3, 4
- [12] X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier neural networks. In *Aistats*, volume 15, page 275, 2011. 4
- [13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015. 3
- [14] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *2015 IEEE International Conference on Computer Vision (CVPR)*, pages 1026–1034, 2015. 5
- [15] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. *arXiv preprint arXiv:1603.05027*, 2016. 4
- [16] J.-B. Huang, A. Singh, and N. Ahuja. Single image super-resolution from transformed self-exemplars. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5197–5206. IEEE, 2015. 5
- [17] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 4, 5
- [18] J. Kim, J. K. Lee, and K. M. Lee. Accurate image super-resolution using very deep convolutional networks. *arXiv preprint arXiv:1511.04587*, 2015. 3, 4, 5, 6, 7
- [19] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012. 1
- [20] S. Mallat. *A wavelet tour of signal processing*. Academic press, 1999. 3
- [21] X.-J. Mao, C. Shen, and Y.-B. Yang. Image denoising using very deep fully convolutional encoder-decoder networks with symmetric skip connections. *arXiv preprint arXiv:1603.09056*, 2016. 1, 3, 4
- [22] S. Osher, M. Burger, D. Goldfarb, J. Xu, and W. Yin. An iterative regularization method for total variation-based image restoration. *Multiscale Modeling & Simulation*, 4(2):460–489, 2005. 1
- [23] J. Portilla, V. Strela, M. J. Wainwright, and E. P. Simoncelli. Image denoising using scale mixtures of Gaussians in the wavelet domain. *IEEE Transactions on Image Processing*, 12(11):1338–1351, 2003. 2
- [24] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241. Springer, 2015. 1
- [25] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1874–1883, 2016. 4, 7
- [26] A. Skodras, C. Christopoulos, and T. Ebrahimi. The JPEG 2000 still image compression standard. *IEEE Signal processing Magazine*, 18(5):36–58, 2001. 3
- [27] M. Telgarsky. Benefits of depth in neural networks. *arXiv preprint arXiv:1602.04485*, 2016. 3
- [28] V. N. Vapnik and V. Vapnik. *Statistical learning theory*, volume 1. Wiley New York, 1998. 3
- [29] A. Vedaldi and K. Lenc. Matconvnet: Convolutional neural networks for matlab. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 689–692. ACM, 2015. 5
- [30] M. Vetterli and J. Kovacevic. *Wavelets and subband coding*. Number LCAV-BOOK-1995-001. Prentice-Hall, 1995. 7

- [31] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. [7](#)
- [32] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang. Beyond a Gaussian denoiser: Residual learning of deep cnn for image denoising. *arXiv preprint arXiv:1608.03981*, 2016. [1](#), [3](#)

9. Appendix

9.1. Persistent homology results

For the denoising task, total 4500 numbers of 40×40 patches with four components of the wavelet transformed data and 80×80 patches of the image domain data were set to a set of point clouds in \mathbb{R}^{6400} vector space of both image and wavelet domain manifolds. Note that the patch size of the image domain is doubled to match the size of the receptive field. For the super-resolution experiments, a set of 20×20 patches was cropped to generate a point cloud in both bicubic and unknown datasets. Similarly, 40×40 image patch was used for image domain approaches.

To investigate the topology of the dataset, a metric should be defined. Since we used the l_2 regression error as the loss, the natural choice should be l_2 distance, i.e. $d_2(X_i, X_j) = \|X_i - X_j\|_2$. However, care need to be taken for the input dataset, because the data should pass through the batch-normalization which changes their mean and variance. Note the distance metric that is invariant under batch normalization is the correlation-based metric: $d_{corr}(X_i, X_j) = \sqrt{1 - corr(X_i, X_j)}$, where $corr(X_i, X_j)$ denotes the normalized Pearson's correlation between X_i and X_j . Therefore, we used the d_{corr} as the metric for the input space, whereas d_2 was used as a metric for the label space.

To show that the advantages of residual learning comes from the simpler topology of residual labels, we first calculated the barcodes of the original images and residuals. We calculated Betti numbers and barcodes using a toolbox called JAVAPLEX (<http://appliedtopology.github.io/javaplex/>). The barcodes in the left column of the Fig. 9 clearly show that the topology of the label manifold composed of residual image patches is much simpler. The point clouds of the residual image patches merged earlier than that of the original ones. Note that the residual manifolds of both image and wavelet domains have an identical topological complexity in the image domain residual due to the orthogonal Haar wavelet transform. The barcodes of residual manifolds in Gaussian denoising case showed rapid drops at $\epsilon = 0.17$, which is related to the noise standard deviation. These results imply that the label manifold for residual has much simpler topology than that of original one. We believe that this is the main reason for the success of deep residual learning.

Next, we investigated the advantages of wavelet transform which further reduces the topological complexity of the input spaces in both Gaussian denoising and super-resolution datasets (the right column of Fig. 9). While the residual manifolds of both image and wavelet domains had a very similar topological complexity represented by barcodes, the input manifold of the wavelet transformed images had simpler topology than the original images. How-

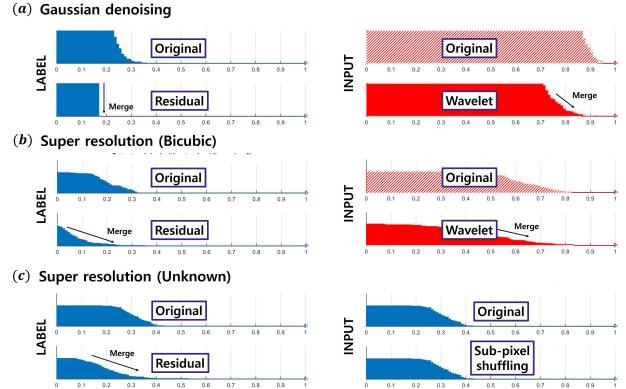


Figure 9. (Left) Persistent homology analysis of *label* manifold. Zero dimensional barcodes of the label manifolds between the original and residual images are compared for (a) Gaussian denoising, (b) super-resolution (bicubic decimation), and (c) super-resolution (unknown decimation) tasks. (Right) Persistent homology analysis of *input* manifold. Zero dimensional barcodes of the input manifolds between the original and feature space mappings are compared for (a) Gaussian denoising, and (b) super-resolution (bicubic decimation), and (c) super-resolution (unknown decimation) tasks.

ever, the sub-pixel shuffling in the unknown decimation case does not change the manifold structure because it merely shuffles the order of the pixels. Accordingly, the topology of the input manifold does not change, but the residual learning in sub-pixel shuffling domain reduces the complexity of the label manifold.

Accordingly, we expect that the proposed denoising and super-resolution algorithms are benefited from the simpler *input* manifolds from feature space mapping as well as additional simpler *label* manifold from residual learning.

9.2. Additional Results from Gaussian denoising and NTIRE SISR competition

In this section, we provide more results for Gaussian denoising and SISR reconstruction from NTIRE competition dataset.

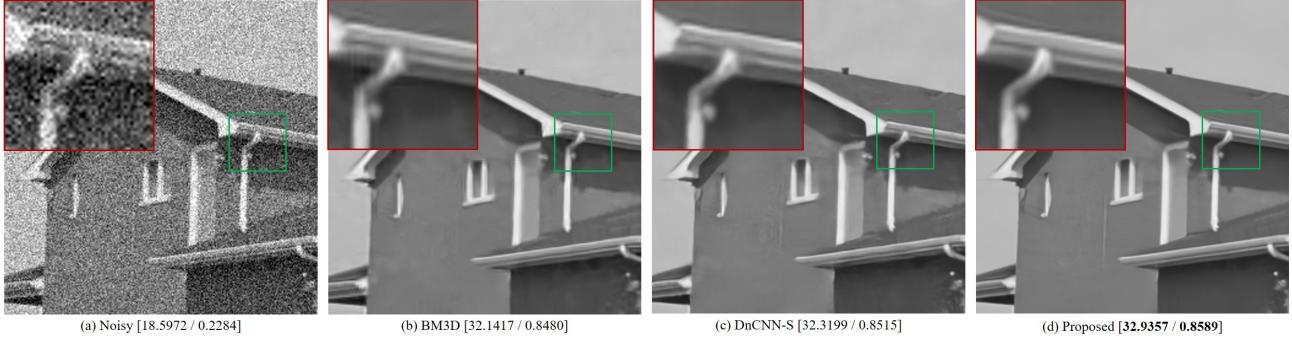


Figure 10. Denoising performance comparison. PSNR/SSIM values are displayed. Gaussian noise with $\sigma=30$ were added. . .

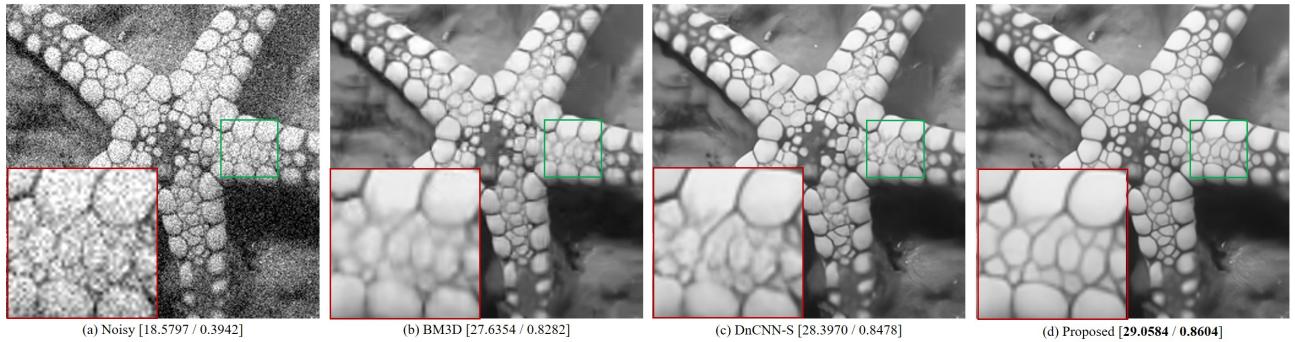


Figure 11. Denoising performance comparison. PSNR/SSIM values are displayed. Gaussian noise with $\sigma=30$ were added. . .

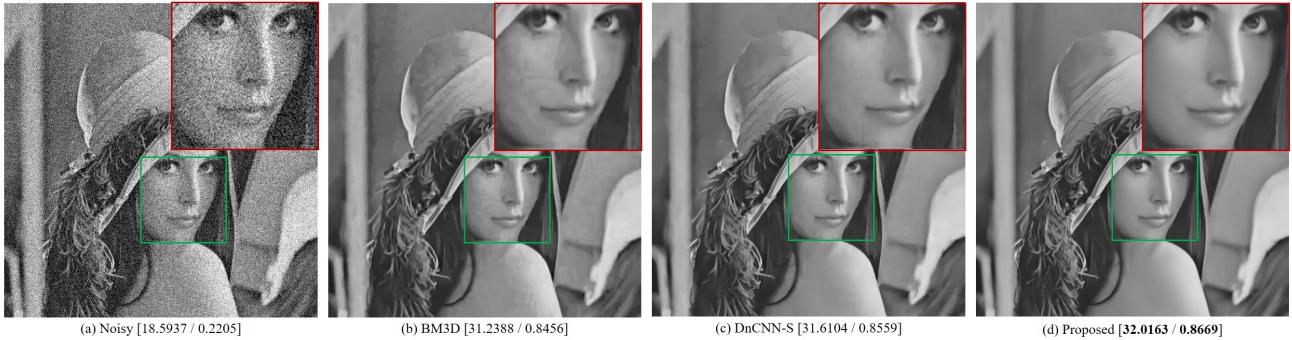


Figure 12. Denoising performance comparison. PSNR/SSIM values are displayed. Gaussian noise with $\sigma=30$ were added. . .



Figure 13. Performance comparison of SISR at scale factor of 4 of unknown downsampling. The proposed network is RGB based network. Left : input, Center : restoration result, Right : label.



Figure 14. Performance comparison of SISR at scale factor of 4 of unknown downsampling. The proposed network is RGB based network. Left : input, Center : restoration result, Right : label.



Figure 15. Performance comparison of SISR at scale factor of 4 of unknown downsampling. The proposed network is RGB based network. Left : input, Center : restoration result, Right : label.



Figure 16. Performance comparison of SISR at scale factor of 4 of bicubic downsampling. The proposed network is RGB based network. Left : input, Center : restoration result, Right : label.



Figure 17. Performance comparison of SISR at scale factor of 4 of bicubic downsampling. The proposed network is RGB based network. Left : input, Center : restoration result, Right : label.



Figure 18. Performance comparison of SISR at scale factor of 4 of bicubic downsampling. The proposed network is RGB based network. Left : input, Center : restoration result, Right : label.



Figure 19. Performance comparison of SISR at scale factor of 4 of bicubic downsampling. The proposed network is RGB based network. Left : input, Center : restoration result, Right : label.