Fractal Residual Network and Solutions for Real Super-Resolution

Junhyung Kwak
NALBI Inc.
Seoul, Republic of Korea
snu11ee@gmail.com

Donghee Son
CASTIS Multimedia R&D Center
Seoul, Republic of Korea
son1113@snu.ac.kr

Abstract

The degradation function in single image super-resolution (SISR) is usually bicubic with an integer scale factor. However, bicubic is not realistic and a scale factor is not always an integer number in the real world. We introduce some solutions that are appropriate for realistic SR. First, we propose down-upsampling module which allows general SR network to use GPU memory efficiently. With the module, we can stack more convolutional layers, resulting in higher performance. We also adopt a new regularization loss, auto-encoder loss. That loss generalizes down-upsampling module. Furthermore, we propose fractal residual network (FRN) for SISR. We extend residual in residual structure by adding new residual shells and name that structure FRN because of the self-similarity like the fractal. We show that our proposed model outperforms state-of-the-art methods and demonstrate the effectiveness of our solutions by several experiments on NTIRE 2019 dataset [1].

1. Introduction

Single image super-resolution (SISR) is an image restoration problem to get a high-resolution (HR) image from its low-resolution (LR) image downsampled with some degradation function. SISR is one of the important low-level computer vision tasks because it can be applied in various fields such as medical image [15, 22], satellite image [24], and surveillance [28]. Recently, deep learning based methods boost performance in the computer vision fields [3, 11, 17]. Similarly, deep neural networks has provided significant improvements in SISR [2, 4, 8, 9, 12, 13, 18, 19, 26, 27].

To handle the resolution difference between the LR image and the HR image, there are two approaches in SISR literature. The early methods of adopting deep learning for super-resolution, such as SRCNN [2] and VDSR [8], aim to learn the mapping function between interpolated LR images and their HR images. In other words, these models take interpolated LR images as inputs. Later on, an efficient up-

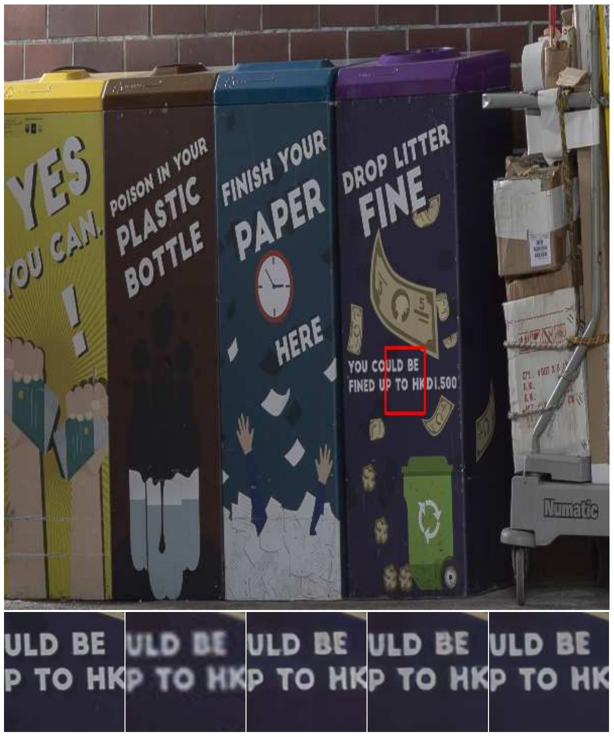

Figure 1: The result of our model on cam2_02 from NTIRE 2019 validation set compared with baseline models.

sampling layer is proposed in [16]. The proposed layer upsamples a LR feature map to a HR feature map at the end of the network.

However, most of these studies assume that the degradation function, which is used to generate LR images, is bicubic downsampling. Bicubic downsampling is non-realistic degradation function and hard to be applied in the real world. In addition, we could not use the upsampling layer because the scale factor is not always an integer in a realistic situation.

There are some trials to emulate the realistic situation

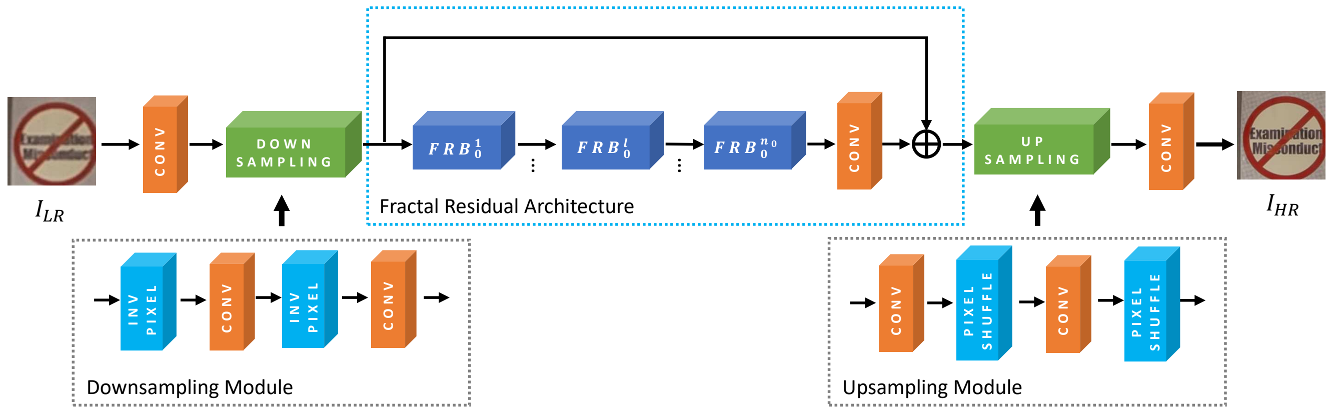

Figure 2: Overall architecture of our proposed fractal residual network.

for SR, such as [20]. In [20], new datasets for the realistic SR are provided. However, these datasets do not assume a completely realistic situation. The datasets are generated by using some operators can be represented as theoretic functions. Also, the scale factor of the datasets is an integer. That is to say, these datasets [20] cannot emulate the realistic situation perfectly. To approximate the realistic degradation function more accurately, new dataset [1] is released. The new dataset is generated by using DSLR cameras for reflecting a realistic situation. The scale factor of the new dataset is not an integer. With the real-data, we should use LR images upsampled to the same size as their HR images, as earlier deep learning based methods did.

In this paper, we propose two solutions for real SR. The first solution is down-upsampling module for exploiting GPU memory efficiently. The module consists of downsampling module and upsampling module applied at the front part and the back part of the model respectively. Downsampling module reduces the size of input feature map. Upsampling module expands the size of the feature map so that it makes the size of the feature map same as that of the original feature map. Excluding the first and the last layer of the model, almost of convolutional operations in our model take the size-reduced feature as input. Thereby, we can stack convolutional layers deeper than the model without down-upsampling module under the same GPU memory condition. The second solution is auto-encoder regularization loss, which is added to L1 loss in the training step. Auto-encoder loss helps the down-upsampling module to extract useful feature for restoring HR images. The improvement of performance by using proposed solutions loss is described in Section 4.4.

Furthermore, we propose fractal residual network where the residual in residual (RIR) patterns are fractally repeated. Residual architecture can recover the high-frequency components. Since a skip connection bypasses the low-frequency components in the input image, the architecture can focus on reconstructing the high-frequency components. RIR structure proposed in RCAN [26] to reconstruct the high-frequency components of the high-frequency components by stacking the residual architectures in a larger residual architecture. Inspired by RIR, we envelop the residual architectures in a larger residual architecture iteratively to get residual in residual in residual… architecture. The proposed model can learn more and more high-frequency information because residual architectures are hierarchically repeated.

To show the superiority of our method, we evaluate our proposed model on newly provided NTIRE 2019 dataset [1] which is realistic SR data. In addition, we show that our proposed solutions are effective for realistic SR via model analysis.

2. Related Works

After the sensational success of the deep learning based methods in computer vision [3, 11, 17], a lot of deep learning-based methods for SISR have been studied [2, 8, 9, 13, 26, 27]. The first SISR model using deep learning was proposed by Dong *et al.* [2]. They used three convolutional layers to learn the mapping between bicubic interpolated LR images and their HR images. Later on, Kim *et al.* [8] stacked very deep convolutional layers and introduced residual learning to train very deep network. Kim *et al.* [9] also used recursive learning for SISR. Tai *et al.* proposed DRRN [18] and MEMNET [19] with recursive block and memory block for super-resolution respectively. Since these models take bicubic interpolated LR images as inputs, the computational complexity of the models increases.

To handle the issue, shi *et al.* [16] designed an efficient upsampling layer often referred to as pixel shuffling layer. Their work used a LR image itself as input, not interpolated LR image. Most operations of the model are computed for LR feature maps. Next, the pixel shuffling layer is applied at the end of the network to upsample the last LR feature map to a HR feature map for an integer scale factor. Therefore,

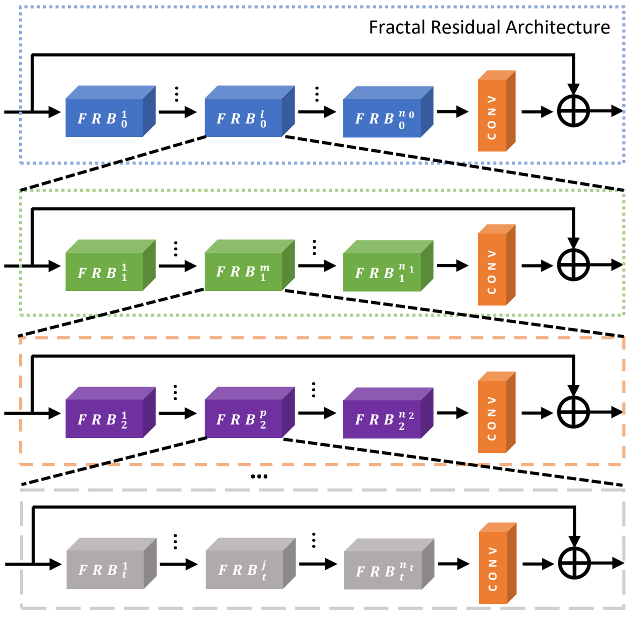

Figure 3: The structure of fractal residual architecture.

the computational complexity of the model decreases. Since the proposal of the pixel shuffling layer, it is used to reduce the computational complexity in many SISR models [12, 13, 26, 27].

Inspired by Resnet [5], Ledig *et al.* [12] proposed the SISR network composed of residual blocks. For efficient memory usage, the residual block without batch normalization [7] was proposed by Lim *et al.* [13]. They could make a deeper model than previous works. Moreover, they reported that L1 loss is better than L2 loss in terms of PSNR. After that, most SR models [26, 27] used L1 loss as loss function. To exploit hierarchical features, Zhang *et al.* [27] proposed residual dense network (RDN). They used residual dense block as a basic block, which is a combination of residual block and dense block. Hu *et al.* [6] introduced Squeeze-and-Excitation network which adopted channel attention mechanism to recalibrate the feature responses. Motivated by squeeze-and-excitation block [6], Zhang *et al.* [26] proposed RCAN which incorporated channel attention into SISR method. Another contribution of RCAN is residual in residual (RIR) architecture. RIR architecture can reconstruct the high-frequency components in a LR image more than a conventional residual architecture.

3. Proposed Model

3.1. Overall Structure of Fractal Residual Network

In this section, we describe our proposed model, fractal residual network (FRN) for SISR. FRN consists of five parts as shown in Figure 2. 1) LR feature extraction layer, 2) Downsampling module, 3) Fractal residual architecture, 4) Upsampling module, 5) Reconstruction layer.

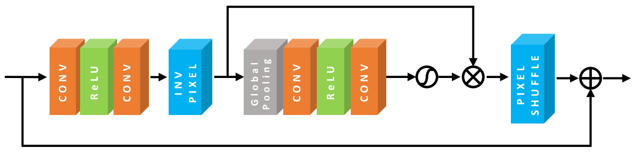

Figure 4: The architecture of RCAB_PS. Pixel shuffling layer and inverse pixel shuffling layer are added to RCAB.

Let I_{LR} and I_{SR} be the interpolated LR image and its corresponding output image of FRN respectively. We used just one convolutional layer for extracting feature X_{FE} from I_{LR} as RCAN [26] did.

$$X_{FE} = f_{FE}(I_{LR}) \tag{1}$$

where $f_{FE}(\cdot)$ is a convolutional operation. Then, X_{FE} is fed to downsampling module for memory efficiency. Downsampling module consists of two convolutional layers and two inverse pixel shuffling layers as shown in Figure 2.

$$X_{DOWN} = f_{DOWN}(X_{FE}) \tag{2}$$

where $f_{DOWN}(\cdot)$ is the operation of downsampling module. The size of X_{DOWN} is smaller than that of X_{FE} due to two inverse pixel shuffling layers in downsampling module. Because of that, most of the layers in FRN take the reduced feature as an input. We can build up a deeper network by using downsampling module. The loss of spatial information can occur due to the reduction of feature size. However, since we use interpolated LR images which have spatially low information density, there is not much loss of spatial information. Furthermore, since our downsampling module is trainable and reduces the size of a feature map in the feature domain, it is more efficient than bicubic downsampling which is not trainable and reduces the size of an image in the image domain.

Next, fractal residual architecture takes X_{DOWN} as an input.

$$X_{FRA} = f_{FRA}(X_{DOWN}) \tag{3}$$

where $f_{FRA}(\cdot)$ is the operation of fractal residual architecture. Fractal residual architecture extracts the useful features for reconstructing HR images. The details about fractal residual architecture will be described in the next section (see Section 3.2). Upsampling module increases the size of the extracted feature. The feature map becomes the same size as X_{FE}. The module is composed of two convolutional layers and two pixel shuffling layers [16] similar to downsampling module.

$$X_{UP} = f_{UP}(X_{FRA}) \tag{4}$$

where $f_{UP}(\cdot)$ is the operation of the upsampling module. Finally, I_{SR} is obtained as the output of reconstruction layer. The reconstruction layer transforms the feature map to an image.

$$I_{SR} = f_{REC}(X_{UP}) = f_{FRN}(I_{LR}) \tag{5}$$

where $f_{REC}(\cdot)$ is a convolutional layer and $f_{FRN}(\cdot)$ denotes all operations of our FRN.

3.2. Fractal Residual Architecture

In this section, the details about fractal residual architecture in FRN are described. Fractal residual architecture has a self-similarity like the fractal structure. The similar structure is repeated with different level as shown in Figure 3.

The operation of fractal residual architecture is denoted by $f_{FRA}(\cdot)$. This operation can be presented in the following formula.

$$f_{FRA}(x) = f_{conv} \circ f^{n_0}_{FRB_0} \circ \cdots \circ f^{1}_{FRB_0}(x) + x \tag{6}$$

where x is an input of fractal residual architecture, $f^{l}_{FRB_0}(\cdot)$ is the operation of the l-th outermost fractal residual block in fractal residual architecture and f_{conv} is a convolutional operation. We can represent the operations $f^{i}_{FRB_0}(\cdot)$ (for $1 \leq i \leq n_0$) as general form $f_{FRB_0}(\cdot)$ because all operations have the same form. We present $f_{FRB_0}(\cdot)$ with detail.

$$f_{FRB_0}(x) = f_{conv} \circ f^{n_1}_{FRB_1} \circ \cdots \circ f^{1}_{FRB_1}(x) + x \tag{7}$$

where x is an input of $f_{FRB_0}(\cdot)$ and $f^{i}_{FRB_1}(\cdot)$ (for $1 \leq i \leq n_1$) is the second outermost fractal residual blocks. We can generalize the operations of fractal residual architecture.

$$f_{FRB_{k-1}}(x) = f_{conv} \circ f^{n_k}_{FRB_k} \circ \cdots \circ f^{1}_{FRB_k}(x) + x \tag{8}$$

In short, because of self-similarity in the architecture, we name this architecture as "Fractal" residual architecture. Fractal residual architecture contains lots of complex skip connections. Because low-frequency components in the input image can be bypassed via lots of skip connections, the model can concentrate on reconstructing high-frequency components in the input image.

We use RCAB_PS (Residual Channel Attention Block with Pixel Shuffling) as a basic block in fractal residual architecture. The innermost blocks of fractal residual architecture are different from the above.

$$f_{FRB_t}(x) = f_{RCAB\_PS}(x) \tag{9}$$

where $f_{RCAB\_PS}(\cdot)$ is the operation of RCAB_PS. We extend RCAB proposed in [26] by adding a pixel shuffling layer and an inverse pixel shuffling layer (see Figure 4). It can speed up the inference time without the performance degradation.

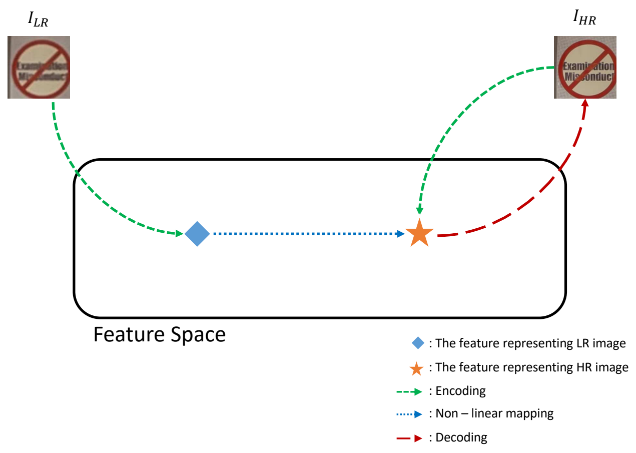
I_{LR}
I_{HR}
Feature Space
◆ : The feature representing LR image
★ : The feature representing HR image
: Encoding
: Non – linear mapping
: Decoding

Figure 5: Illustration of relation between SR and auto-encoder loss.

3.3. Loss Function

Conventionally, super-resolution networks have been interpreted three parts 1) encoder, 2) non-linear mapper, 3) decoder as shown in Figure 5. The encoder maps an image to the feature representing the image. The non-linear mapper transforms the feature representing the LR image to the feature representing the HR image corresponding to its LR image. The decoder reconstructs an image from the feature map representing the image.

In our proposed model, LR feature extraction layer and downsampling module correspond to the encoder (i.e. $f_{encoder} = f_{DOWN} \circ f_{FE}$). The fractal residual architecture is a non-linear mapper. Upsampling module and reconstruction layer correspond to the decoder (i.e. $f_{decoder} = f_{REC} \circ f_{UP}$). As Zeng *et al.* mentioned in [25], the proper encoder and decoder help the SR model to improve the performance.

As shown in Figure 5, without using a non-linear mapper, the input image should be restored as the output of the decoder. Therefore, we propose auto-encoder loss to generalize the encoder and decoder for useful feature extraction. Only HR images are used for auto-encoder loss because it helps to restore high-frequency components in the HR image.

$$I_{AE} = f_{decoder} \circ f_{encoder}(I_{HR}) = f_{AE}(I_{HR}) \tag{10}$$

where I_{HR} is the high-resolution image and I_{AE} is the restored image without the non-linear mapper. By minimizing the difference between I_{AE} and I_{HR}, we can get the generalized encoder and decoder.

Let $\{I^{i}_{LR}, I^{i}_{HR}\}^{N}_{i=1}$ be the training data where I^{i}_{LR} is the LR image and I^{i}_{HR} is the HR image corresponding to its LR image. First, we use L1 loss which is generally utilized in the SR network [13, 27, 26]. Additionally, we use auto-

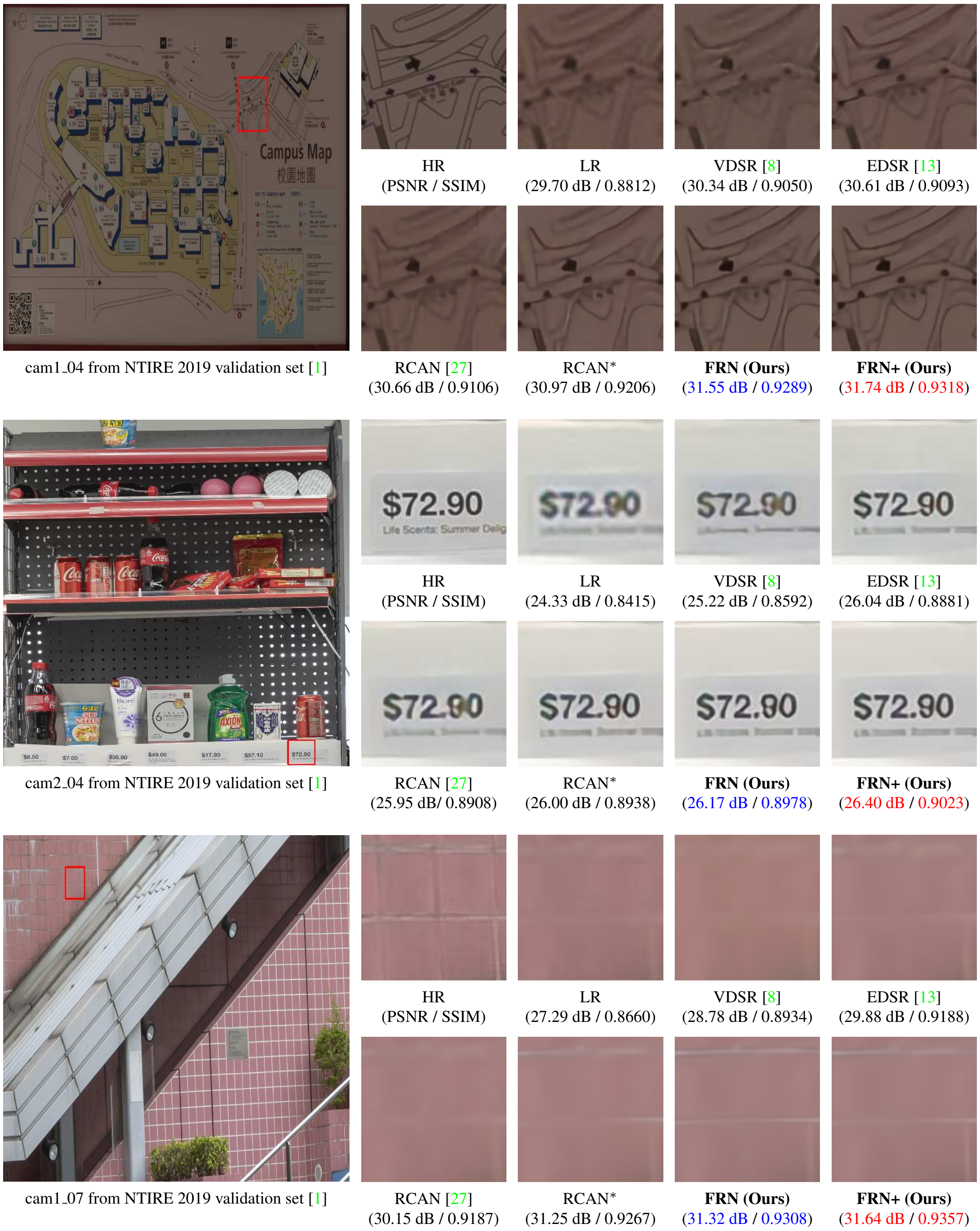

Figure 6: Qualitative comparison of our model with other works on NTIRE 2019 validation set [1].

method	LR	VDSR [8]	EDSR [13]	EDSR*	RCAN [27]	RCAN*	FRN (ours)	FRN+ (ours)
PSNR	27.78	28.38	29.10	29.36	29.36	29.60	29.85	30.06
SSIM	0.8712	0.8789	0.8989	0.9041	0.9013	0.9069	0.9108	0.9143

Table 1: Quantitative evaluation on NTIRE 2019 validation set. Red and blue indicate the best and the second best respectively. Baseline model with * means that down-upsampling module is added to the model and auto-encoder loss is also used to train.

encoder loss to get the generalized encoder and decoder.

$$Loss = \frac{1}{N}\sum_{i=1}^{N} ||I^i_{HR} - I^i_{SR}||_1 + ||I^i_{HR} - I^i_{AE}||_1 \quad (11)$$

where $I_{SR} = f_{FRN}(I_{LR})$ and $I_{AE} = f_{AE}(I_{HR})$. We minimize the loss function to train our proposed FRN. The more details of the training step will be described in Section 4.2

4. Experiments

4.1. Dataset and Metric

The NTIRE 2019 dataset [1] is the newly released image dataset for the realistic super-resolution. The dataset consists of 60 training images, 20 validation images, and 20 test images. The dataset is composed of interpolated LR images and their corresponding HR images. To get realistic super-resolution dataset, the dataset has been prepared using DSLR cameras. We compare the performance on the validation set because the ground truth images of the test set are not provided.

All experimental results are evaluated with PSNR and SSIM [23] and measured on RGB channels.

4.2. Implementation Details

In this section, we describe training and model hyper-parameter details. We randomly extract 192×192 patches and set the size of mini-batch as 8. We augment the patch pairs with horizontal and vertical flips and 90° rotations randomly. The optimizer used is Adam optimizer [10] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$. We set the learning rate as 10^{-4} and use early-stopping at 1.6×10^5 iterations because of overfitting resulting from not enough data.

We set the reduction ratio, the kernel size and the depth of convolutional layers to the values same as RCAN [26]. For fractal structure, $t = 4$ and n_0, n_1, n_2, n_3 are 2, 4, 8, 8 respectively. All pixel shuffling and inverse pixel shuffling scale is 2. All networks are implemented with the PyTorch [14]. We use NVIDIA GTX 1080 ti GPUs for experiments. Our source code and trained model are publicly available at https://github.com/Junshk/FRN.

4.3. Experimental Results on NTIRE 2019 dataset

We evaluate the performance of FRN on NTIRE 2019 dataset [1] which is newly provided for realistic SR. The proposed model and several state-of-the-art models, VDSR [8], EDSR [13], and RCAN [26], are compared via quantitative and qualitative analysis. We evaluate not only original EDSR and RCAN with scale factor 1 but also modified EDSR and RCAN which applied our solutions (both down-upsampling module and auto-encoder loss). Adding "*" superscript to the model name means the modified model that applied our contributions. We present the method using self-ensemble [21] by putting "+" postfix to the method name.

As shown in Table 1, FRN+ and FRN achieve the best and the second best performance respectively. Our proposed model outperforms the other models in terms of quantitative comparison. The qualitative results are shown in Figure 6. Compared the other models, FRN not only can reconstruct the high-frequency components, textures, and details but also remove the noise in the interpolated LR image. Furthermore, the modified baseline models achieve better performance than the original baseline models, respectively. As the proposed solutions are applied for baseline models, the performance is improved. Thus, we demonstrate the effectiveness of our proposed solutions for realistic SR. The more details about model analysis are described in the next section (Section 4.4)

We also participated in NTIRE 2019 real super-resolution challenge with a team name of LulluVision. The results are measured on NTIRE 2019 test set. We submitted our results to challenge. We ranked the 5th with 28.88 PSNR value and 0.84 SSIM value.

4.4. Model Analysis

We analyze the proposed model to demonstrate the effectiveness of our contributions, 1) down-upsampling module, 2) auto-encoder loss, and 3) fractal residual architecture, via experiments. For each contribution, we compare the model with the contribution to the model without it. Moreover, we do an experiment to show the effect of pixel shuffling and inverse pixel shuffling in a basic block (RCAB_PS) of FRN. All experiments are performed on NTIRE 2019 dataset [1].

Down-upsampling	✗	✓	✓	✓	✓
Auto-encoder loss	✗	✗	✓	✗	✓
FRN	✗	✗	✗	✓	✓
PSNR	29.36	29.57	29.56	29.75	29.85

Table 2: Ablation investigations of architecture, down-upsampling module and auto-encoder loss. For a fair comparison, RCAN in the third investigation is set deeper than the original version.

The results are evaluated with PSNR on RGB channels.

1) Down-upsampling module: FRN without down-upsampling module is not trainable because of OOM (out of memory) problem. We use RCAN [26] instead of FRN in this experiment. For a fair comparison, RCAN with down-upsampling module does not use auto-encoder loss because the model without down-upsampling module could not apply auto-encoder loss. Down-upsampling module provides performance improvement from 29.36 dB to 29.57 dB (see the first and the second columns in Table 2).

2) Auto-encoder loss: We compare our proposed loss function (L1 + auto-encoder loss) with the only L1 loss to validate the effect of auto-encoder loss. Except for the loss function, the other conditions are same. FRN is used for this experiment. Performance is improved from 29.75 dB to 29.85 dB when auto-encoder loss is added to L1 loss. In short, auto-encoder loss make down-upsampling module (i.e. encoder and decoder) extract the useful feature for SR. Thereby the performance is improved (see the fourth and the fifth columns in Table 2).

3) Fractal residual architecture: We compare FRN with the network which does not use fractal residual architecture. We choose RCAN [26], state-of-the-art SR network, as a baseline model. For a fair comparison, we make the number of parameters in both models roughly same. The number of parameters in FRN is about 45.5M. We increase the residual group number of RCAN to 30 and the number of parameters in the modified RCAN is 46.4M. Down-upsampling module and auto-encoder loss are applied to both models for the same condition. Fractal residual architecture makes improvement from 29.56 dB to 29.85 dB under the condition where the number of parameters in models is nearly same (see the third and the fifth columns in Table 2).

4) RCAB_PS: RCAB was proposed in [26]. To show the effect of pixel shuffling and inverse pixel shuffling in RCAB_PS, we experiment with two models. Both models adopt our solutions, down-upsampling module, auto-encoder loss, and fractal residual architecture. The only difference between each model is the presence of pixel shuffling and inverse pixel shuffling in the basic block of fractal residual architecture. There is little difference in PSNR between the two models. However, the average inference speed of the model with RCAB_PS is faster than that of the model with RCAB on NTIRE 2019 validation set [1]. The average inference time is improved from 11.25 sec to 9.1 sec as shown in Table 3. Because the spatial size of the feature map decrease, the time required for global average pooling is reduced. It is a bottleneck of inference. Thus the overall inference time is reduced.

Basic block	PSNR	Avg. inference time (sec)
RCAB	29.82	11.3
RCAB_PS	29.85	9.1

Table 3: Comparison with RCAB_PS and RCAB in terms of PSNR and average inference time.

5. Conclusion

In this paper, we propose fractal residual network (FRN) and specific solutions for real super-resolution. FRN is a much deeper model than previous super-resolution networks and the structure that adds long skip connection to the lower level of residual blocks iteratively. As a result of the fractal residual architecture, our model could learn the high-frequency information more easily.

The first solution, down-upsampling module is an essential part for real super-resolution task because it reduces the resolution of the feature map and increases the size of receptive fields without spatial information loss. The second solution is auto-encoder loss. The loss is effective regularizer for the first solution despite its simplicity.

By adopting our strategies, the proposed method achieves the performance improvement on realistic SR compared with existing methods.

References

[1] Jianrui Cai, Hui Zeng, Hongwei Yong, Zisheng Cao, and Lei Zhang. Toward real-world single image super-resolution: A new benchmark and a new model. *arXiv preprint arXiv:1904.00523*, 2019.

[2] 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*, 38(2):295–307, 2016.

[3] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[4] Muhammad Haris, Gregory Shakhnarovich, and Norimichi Ukita. Deep back-projection networks for super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1664–1673, 2018.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceed-*

ings of the IEEE conference on computer vision and pattern recognition, pages 770–778, 2016.

[6] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[7] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.

[8] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1646–1654, 2016.

[9] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Deeply-recursive convolutional network for image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1637–1645, 2016.

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

[11] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[12] Christian Ledig, Lucas Theis, Ferenc Huszár, 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

[13] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 136–144, 2017.

[14] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[15] Andrea Rueda, Norberto Malpica, and Eduardo Romero. Single-image super-resolution of brain mr images using overcomplete dictionaries. *Medical image analysis*, 17(1):113–132, 2013.

[16] 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1874–1883, 2016.

[17] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[18] Ying Tai, Jian Yang, and Xiaoming Liu. Image super-resolution via deep recursive residual network. In *Proceedings of the IEEE Conference on Computer vision and Pattern Recognition*, pages 3147–3155, 2017.

[19] Ying Tai, Jian Yang, Xiaoming Liu, and Chunyan Xu. Memnet: A persistent memory network for image restoration. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[20] Radu Timofte, Shuhang Gu, Jiqing Wu, and Luc Van Gool. Ntire 2018 challenge on single image super-resolution: methods and results. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 852–863, 2018.

[21] Radu Timofte, Rasmus Rothe, and Luc 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.

[22] Dinh-Hoan Trinh, Marie Luong, Francoise Dibos, Jean-Marie Rocchisani, Canh-Duong Pham, and Truong Q Nguyen. Novel example-based method for super-resolution and denoising of medical images. *IEEE Transactions on Image processing*, 23(4):1882–1895, 2014.

[23] Zhou Wang, Alan C Bovik, Hamid R Sheikh, Eero P Simoncelli, et al. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[24] Deniz Yıldırım and Oğuz Güngör. A novel image fusion method using ikonos satellite images. *Journal of Geodesy and Geoinformation*, 1(1):75–83, 2012.

[25] Kun Zeng, Jun Yu, Ruxin Wang, Cuihua Li, and Dacheng Tao. Coupled deep autoencoder for single image super-resolution. *IEEE transactions on cybernetics*, 47(1):27–37, 2017.

[26] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 286–301, 2018.

[27] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2472–2481, 2018.

[28] Wilman WW Zou and Pong C Yuen. Very low resolution face recognition problem. *IEEE Transactions on image processing*, 21(1):327–340, 2012.