Fast CNN enhancement using channel attention and residual networks for image super-resolution

Haoning Qu[1], **Yifeng Ruan**[2], **Zerui Wan**[3] **and Ming Zhu**[4,5]

[1]College of Arts and Science, New York University, New York, NY, 10012, USA
[2]Alfred Lerner College of Business & Economics, University of Delaware, Newark, DE, 19711, USA
[3]Faculty of Applied Science & Engineering, University of Toronto, Toronto, ON, M5S 1A4，Canada
[4]New College, University of Toronto, Toronto, ON, M5S 1A1, Canada

[5]ming.zhu@mail.utoronto.ca

Abstract. Single image super-resolution (SISR) refers to the process of reconstructing a high-resolution (HR) image from a low-resolution (LR) input image. Deep learning super-resolution algorithms have widely been used to solve SISR tasks. However, the demanding computation cost and memory occupation incurred through training the deep learning models has been hindering its real-world application. In this paper, we rebuild FSRCNN and apply it to solve SISR tasks. Firstly, we change the original training dataset to RealSR, a larger dataset consisting of real-world images. Secondly, channel attention and residual blocks have been applied to the mapping layers and important parameters including learning rate and optimizer have been reset. Thirdly, we change the cost function from L_2 loss to L_1 loss and replace the activation function from parametric rectified linear unit (PReLU) to exponential linear unit (ELU), to verify the discrepancies between different loss functions and activation functions. Finally, we compare the rebuilt models with the official FSRCNN based on the Peak signal-to-noise ratio (PSNR) and the structural similarity index measure (SSIM) on three common test datasets. The original model achieves better performance on all the test datasets across different scale factors while the rebuilt models show better generalization capability. Our analyses illustrate that residual blocks can slightly promote model performance while different loss functions and activation functions do not generate an evident impact on the rebuilt model.

Keywords: channel attention, fast convolutional neural networks, residual block, super-resolution

1. Introduction

The SISR task is an archetypical topic of computer vision research field, which attempts to reconstruct a single HR image from the given input LR image. Since the inception of digital imagery, the demand for high-resolution (HR) images has become increasingly higher within application fields including intelligent surveillance, medical imaging, and remote sensing. This is because higher image resolution contains rich details of the image and has a better visual quality. Since the same LR image could be originated from down-sampling infinitive HR images, SISR is still an ill-posed problem [1].

To address the above issue, Convolutional Neural Network (CNN) based models have become the preferred method among the considerable deep learning algorithms which have been introduced to solve SISR tasks. Dong et al. [2] introduced the first deep CNN, a three-layer Super-Resolution Convolutional Neural Network (SRCNN) to acquire the SR image in an end-to-end mapping manner. Though the simple three-layer network achieved great performance, deeper network structures have also been proven to enhance the performance of SISR tasks [3].

The desire of pursuing superior SR performance has motivated researchers to design deeper and more complex networks such as DRCN [4], SRGAN [5], and SwinIR [6] to solve SR tasks. DRCN is a deep CNN-based model that contains a very deep recursive layer which is consist of 16 recursions. Ledig et al. proposed SRGAN, which is the first model to solve the SR task through the utilization of a Generative Adversarial Network. SwinIR is designed mainly based on Swin Transformer. Although these deep and sophisticated models have pursued great performance on SISR tasks, the high computation cost for memory storage due to its architecture complexity has become an obstacle to its practical application. To solve this problem, multiple lightweight networks have been proposed, such as FSRCNN [7] and ESRT [8]. FSRCNN is a re-designed SRCNN that can solve SR tasks faster and better. ESRT is a combination of a lightweight CNN Backbone and a lightweight Transformer backbone. Though ESRT reduces the network parameters, its computation cost and training difficulty are still very high, while FSRCNN could be trained and achieve good performance on a generic CPU.

This paper proposes a rebuild of FSRCNN to solve SISR tasks. Firstly, we train the model on real-world super-resolution (RealSR) [9], a large dataset consisting of realistic images to obtain a well-trained model. Secondly, we rebuild the model by adding Squeeze and Excitation (SE) [10] blocks, residual blocks [11] in the mapping layers, and resetting parameters in convolutional layers and activation functions of the rebuilt model. Thirdly, we changed the loss function from Mean Squared Error (MSE) [12] to Mean Absolute Error (MAE) [13]. To verify the effectiveness of the proposed method, we compare our model to the official FSRCNN with two common objective methods: PSNR [14] and SSIM [15]. Finally, we set up several controlling experiments to further investigate the effectiveness of different model structures, different loss functions, and different activation functions. The original model gets a PSNR of 29.43 (dB) on Set14 and 28.60 (dB) with scale factor 3 on Set14 and BSD200 test datasets respectively. The results clearly show that the performance of the original decreased as the complexity of test datasets increased. However, our proposed model gets higher PSNR or remains similar performance when the complexity of the test dataset increases. Our rebuilt models achieve better performance on more complicated datasets illustrating stronger generalization capability compared to the original model.

2. The Proposed Research Method

This part introduces the proposed research method with the re-implemented FSRCNN model. Firstly, we use a larger dataset RealSR instead of using the original training dataset General 100 and 91-image dataset [16] to get a self-trained model for RSISR tasks. Then we rebuild FSRCNN by adding channel attention through SE blocks and set up residual blocks in the convolutional layers. Finally, we reset the parameters of the rebuilt model to increase the SR performance and make comparisons with the official model, including optimizer, learning rate, activation function, and loss function.

2.1. Data Pre-processing

Since we attempt to increase the generalization capability of the proposed model, the RealSR version 3 train dataset is utilized to get a well-trained model. Compared to the original training datasets, the RealSR dataset has a larger volume and contains more information. For the sake of reducing training time, we only apply images captured through Canon 5D3 which is only half of the RealSR training dataset.

2.2. Model Structure

We add SE blocks shown in figure.1 and residual blocks in the mapping layers of the rebuild FSRCNN model. The official FSRCNN is redesigned based on SRCNN. There are three main structural differences between them. Firstly, the input LR image is directly fed into the model. Secondly, the up-sampling task is accomplished by a deconvolution layer. Thirdly, the Non-linear mapping is substituted with a three-step process including shrinking, mapping, and expanding. Besides SE blocks and Residual blocks, we rebuild the FSRCNN by replacing the activation function of the rebuilt model from the PReLU to the ELU [17] and changing the loss function from MSE to MAE.

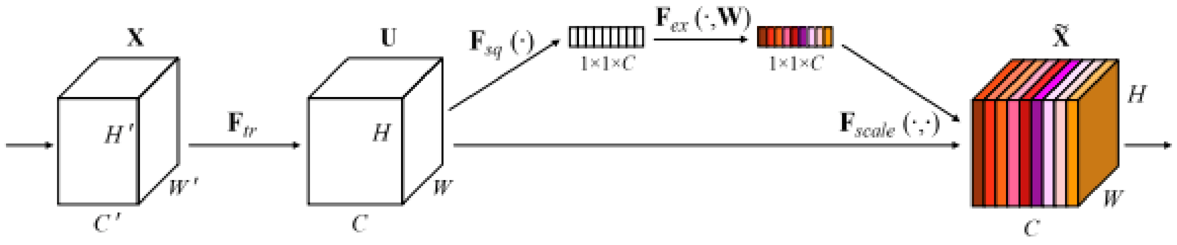

Figure 1. Structure of a SE block

2.2.1. SE blocks: The operating process of a SE block can be divided into two parts: Squeeze and Excitation. Squeeze operation aims to obtain the feature of the input through Global Average Pooling. Excitation operation is implemented by two fully connected layers, allowing the convolution layers to better utilize the output generated through the squeeze operation. SE blocks introduce channel attention to the network which enables the model to recalibrate channel-wise feature information. The recalibration can improve the model performance by emphasizing important features and mitigating unnecessary features. Moreover, SE blocks are computationally efficient which enhances the complexity of the model with a slight computational burden.

2.2.2. Residual blocks: By setting up residual blocks, we aim to ameliorate the degradation and vanishing gradients problem caused by model complexity. The residual blocks enable the output from the previous convolution layer to feed forward to layers from more than 1 hop which make it easier for the model to learn the expected feature mapping.

2.2.3. Convolution layers: The convolution layer is the cardinal part of CNN-based models, which incur the most computation tasks in the model. The computation volume of output O is determined by the size of input data W, Stride S, filter size F, and padding P, which is defined in (1):

$$O = \frac{W-F+2*P}{S} + 1 \tag{1}$$

2.2.4. Activation function: The original FSRCNN model adopts PReLU instead of Rectified Linear Unit (ReLU) as the default activation function, which is designed to mitigate the zero gradients caused by ReLU. Since ELU enables faster learning, we also applied ELU as the activation function for our rebuilt models.

2.3. Model parameter settings

We reset several parameters in the rebuilt model, including optimizer and learning rate. A detailed description is as below.

2.3.1. Optimizer: Different from official FSRCNN, we use Adam optimizer [18] instead of using stochastic gradient descent (SGD). This is because the Adam optimizer has minimal memory usage. Besides, the convergence speed of the Adam optimizer is relatively fast and it can dynamically adjust the hyper-parameters. By applying a higher learning rate for low-frequency parameters the network can learn more information.

2.3.2. Learning rate: The learning rate determines the extent of weight assigned to newly acquired information and the convergence time. Different from the official FSRCNN , we reset the learning rate of convolution layers to 0.0002 and decay the learning rate to half of the previous rate every 200 epochs. Moreover, we reserve the original learning rate of 0.0001 in the deconvolution layer.

2.4. Loss function
One type of learning strategy in machine learning is the loss function, which is used to measure prediction error or reconstruction error. Furthermore, the loss function is a crucial parameter for training process optimization. In our proposed model, we investigated two frequently used loss functions.

2.4.1. MSE loss function: The MSE loss function, also known as L_2 loss, is the quadratic difference between a prediction and the actual value for each sample in the given dataset. The aggregation of all these loss values is calculated as below in (2):

$$L_2 = \frac{1}{hwc}\sum_{i,j,k}\left(\hat{I}_{i,j,k} - I_{i,j,k}\right)^2 \tag{2}$$

In this L_2 equation, h is the height of the image, w is the width of the image, and c is the number of channels of the image. $\hat{I}_{i,j,k}$ is the constructed individual pixel value at row i, column j, and channel k, $I_{i,j,k}$ is the original individual pixel value. Using MSE as the cost function enables the model to get a high PSNR, which is one of the assessment metrics to evaluate model performance. Many CNN-based networks such as SRCNN and FSRCNN use MSE as the cost function.

2.4.2. MAE loss function: The L_1 loss, which is the sum of all the absolute differences between the actual value and the predicted value, is also known as Mean Absolute Error. Compared to L_2 loss, L_1 loss provides commendable accuracy and convergence ability to the model, although it may not promote the model to achieve a better PSNR. The equation of L_1 loss is defined as in (3):

$$L_1 = \frac{1}{hwc}\sum_{i,j,k}\left|\hat{I}_{i,j,k} - I_{i,j,k}\right| \tag{3}$$

3. Experiment Settings
This section illustrates the experimental setting in this paper. Firstly, we introduce the dataset used for training, validation, and test (Sec. 3.1). Secondly, we describe the performance measurement indicators including PSNR and SSIM (Sec. 3.2). Then we illustrate our test on the different loss functions. Finally, we elaborate on our baseline model and controlling experiments with different settings (Sec. 3.3).

3.1. Dataset Description
This part gives a detailed description of our training dataset, validation, and test dataset. In addition, we also introduce our training strategy in the training dataset section.

3.1.1. Training dataset: We use RealSR version 3 as the training dataset instead of using The 91-image and the General-100 dataset. RealSR dataset contains realistic HR-LR image pairs captured by Canon 5D3 and Nikon D810 through focal length adjusting with the scales of 2,3,4 respectively. In terms of assuring the convergence of our model, we trained our model with a dynamic learning rate for 1000 epochs. We trained our model on a generic device, an RTX 2060, the whole training process takes about 12 hours.

3.1.2. Validation and Test dataset: To compare the performance of our rebuild model with the original FSRCNN model, we also adopt Set5[19], Set14 [20], and BSD200 [21] as test datasets. Set5 and Set 14 are commonly used testing dataset which contains 5 and 14 different images respectively. The BSD200 dataset contains 200 different images with a high noise level. The dataset contains images of animals, natural scenes, and artificial landscapes. Among the pictures, some subjects have similar shapes or colors to the background of the images.

3.2. Performance measurement metrics

We use two commonly used quality assessment methods to evaluate the results of the rebuilt model, PSNR and SSIM. A detailed description of each of them is below.

3.2.1. Peak signal-to-noise ratio: The PSNR is a widely used objective quality evaluation metrics for image reconstruction. With a given maximum, $\hat{\boldsymbol{X}}$ and X, the PSNR is calculated as in (4):

$$\mathrm{PSNR} = 10 \cdot \log_{10}\left(\frac{R^2}{\mathrm{MSE}}\right) \tag{4}$$

Where R stands for the maximum pixel value and MSE represents the Mean Squared Error between $\hat{\mathbf{X}}$ and X. Generally, higher PSNR indicates a higher restoration quality. However, in some cases, a higher PSNR may result in a poor correlation in visual quality based on human perception.

3.2.2. structural similarity index measure: The SSIM is commonly used objective quality evaluation metric that evaluates the structural similarity between the reconstructed image and the original image. The evaluation process is collectively performed in three aspects: luminance, contrast, and structure. The SSIM is defined as in (5):

$$\mathrm{SSIM} = \left[l(X,\hat{X})\right]^{\alpha}\left[c(X,\hat{X})\right]^{\beta}\left[s(X,\hat{X})\right]^{\gamma}, \tag{5}$$

3.3. Baseline model and different settings

We rebuild FSRCNN by adding SE blocks and residual blocks as our baseline, then we designed several controlling experiments to test the effectiveness of our rebuild model. We give a detailed description of them below.

3.3.1. Baseline model: FSRCNN is a relatively shallow network that consists of five parts: feature extraction layer, shrinking layer, Non-linear mapping layer, expanding layer, and deconvolution layer. Since the Non-linear mapping layer exerts a significant impact on the SR performance of the model, we add SE blocks and residual blocks through all the mapping layers to enhance the efficiency of the mapping layer.

3.3.2. Investigation of different settings: We implemented controlling experiments to verify the discrepancies of the baseline model with different settings, including model structure, loss functions, and activation functions. Firstly, we test the influence of residual blocks by removing all the residual blocks of the baseline model. Secondly, we examine the usefulness of different loss functions, we apply MAE loss as the loss function for the baseline model and models without residual blocks. Then we measure the effect of different activation functions via adopting ELU as the activation function for the baseline model and models without residual blocks. Finally, we analyze the overall results of the rebuilt models and the original model to verify the effectiveness of the proposed training strategy.

4. Result and Discussion

This section illustrates the detailed result from the rebuilt model with different settings, including PSNR, SSIM, and visual quality. Firstly, we test the effectiveness of different structure - residual blocks (Sec. 4.1). Secondly, we compare the model performance between L_1 loss and L_2 loss (Sec. 4.2). Then, we investigate the influence of different activation functions based on PReLU and ELU (Sec. 4.3). Finally, we compare the original model with the proposed methods (Sec. 4.4).

4.1. Comparison of Residual blocks

From table 1 we can see that all the models with residual blocks reach a slightly higher PSNR and SSIM compared to models without residual blocks except for the factor 4 test on BSD200. Though the residual block is originally designed to mitigate the vanishment of gradients and degradation problems for models with deeper structure and more complexity, the experiments result indicate that it can also slightly improve the performance of models with a relatively shallow structure. We apply all the residual blocks in our mapping layers. If more layers are added to the mapping layers, the performance of the model would be further improved by residual blocks.

Table 1. The PSNR results (dB) on test datasets for baseline model without residual blocks and apply MAE as loss function, baseline model and apply MAE as loss function.

Test Dataset	Upscaling Factor	w/o res	w/o res+L_1	baseline	base+L_1
		PSNR	PSNR	PSNR	PSNR
Set5	× 2	31.51	31.58	31.68	31.52
Set14	× 2	27.99	28.05	28.06	27.97
BSD200	× 2	28.62	28.72	28.71	28.71
Set5	× 3	27.21	27.13	27.27	27.20
Set14	× 3	25.06	25.00	25.08	25.06
BSD200	× 3	25.10	24.91	25.12	24.99
Set5	× 4	26.19	26.16	26.31	26.19
Set14	× 4	24.28	24.28	24.33	24.29
BSD200	× 4	23.38	23.07	23.37	23.09

4.2. Comparison of different Loss Functions

The quantitative results across all the upscaling factors are shown in table 1. MSE and MAE loss functions do not result in a salient difference. Specifically, for the models without residue blocks, the results of using MAE as the loss function are in analogy to models using the MSE loss function. The PSNR values for the scale 2 test sets using MAE are slightly higher than those using MSE. For the scale 3 test sets, the PSNR values for two different methods are almost the same. However, the performance for the model using the MAE loss function goes down by 0.3 dB specifically for the scale 4 BSD200 testing set compared to the baseline model. Moreover, for the experiments utilizing residue block, the overall test results for the models adopting MAE as the loss function appear to be lower than the results using MSE. The differences for scale 2 and scale 3 testing sets are not prominent, at a maximum of 0.13 dB. Nevertheless, the scale 4 BSDS200 testing result for the model using MAE is 0.33 dB lower than the one using MSE. Generally, the MSE loss function enables the model to reach a higher PSNR, and the experiment results are consistent.

4.3. Comparison of different Activation Functions

In our experiments, we investigate the effect of the ELU activation function and compared the results with the original activation function.

Table 2 shows the averaged PSNR values on all the test datasets of models with different residual block settings and different activation functions. In the experiments with no residue block, the PSNR values corresponding to the models using ELU are similar to the baseline PReLU methods. For scale 2 and scale 4, the results are almost equal. The result of scale 3 testing sets for the model using ELU is slightly better. However, the differences are trivial, which are at most 0.02 dB. In addition, for the experiments using residue block, the result is consistent. The PSNR values decrease trivially while choosing the ELU activation function instead of PReLU. Thus, overall, the performance of PReLU and

ELU activation functions is analogical. In addition, while using the ELU as activation function the parameters in the model dropped by 172 which slightly release the computational cost of the model.

Table 2. The PSNR results (dB) on test datasets for baseline model without residual blocks and apply ELU as activation function, baseline model and apply ELU as activation function.

Test Dataset	Upscaling Factor	w/o res	w/o res+ELU	base	base+ELU
		PSNR	PSNR	PSNR	PSNR
Set5	× 2	31.51	31.50	31.68	31.51
Set14	× 2	27.99	27.97	28.06	27.99
BSD200	× 2	28.62	28.63	28.71	28.64
Set5	× 3	27.21	27.22	27.27	27.32
Set14	× 3	25.06	25.07	25.08	25.13
BSD200	× 3	25.10	25.11	25.12	25.16
Set5	× 4	26.19	26.16	26.31	26.23
Set14	× 4	24.28	24.26	24.33	24.32
BSD200	× 4	23.38	23.39	23.37	23.40

4.4. Comparison using different Training sets

The results in table 3 show that the official model reached a higher PSNR on Set14 and BSD200 on both upscaling factors 2 and 3. Original FSRCNN utilized 91-image and General-100 datasets with data augmentation techniques, by employing the augmentation strategy, the training dataset increased to 19 times larger compared to the original dataset. While we only trained our model on images captured by canon 5D3 in the RealSR dataset, which contains 200 HR-LR image pairs.

Table 3. The PSNR results (dB) on Set14 and BSD200 from the official model and our proposed models.

Test Dataset	Upscaling Factor	FSRCNN	w/o res	base	base+L_1	base+elu
		PSNR	PSNR	PSNR	PSNR	PSNR
Set14	× 2	32.63	27.99	28.06	27.97	27.99
BSD200	× 2	31.80	28.62	28.71	28.71	28.64
Set14	× 3	29.43	25.06	25.08	25.06	25.13
BSD200	× 3	28.60	25.10	25.12	24.99	25.16

Since most SR models would be benefited from large training datasets, the training dataset volume difference would be a primary result of the performance gap. Another reason could be the original training datasets and test datasets all consist of synthetic images while the training dataset we picked consists of realistic images. The official FSRCNN model gets higher PSNR on Set14 than BSD200, which indicates that the performance dropped as the complexity of test datasets increased. However, our models reach a higher PSNR on BSD200 compared to Set14, which indicates our rebuild models have better generalization capabilities than the original model. The reconstructed images of "lenna" from the Set14 dataset generated by our proposed models are shown in figure.2.

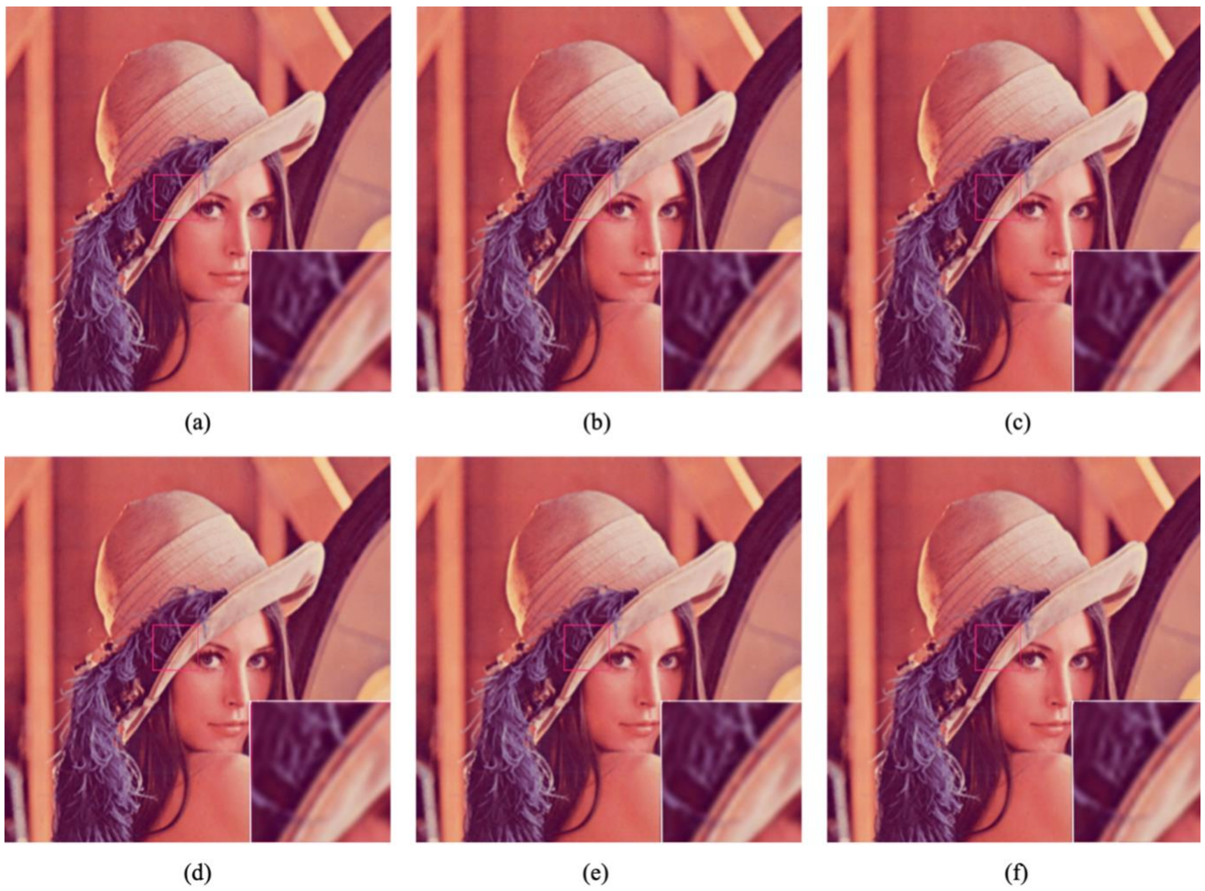

Figure 2. (a)w/o res (b)w/o res + ELU (c)w/o res + L_1 (d)base (e)base + ELU (f)base + L_1.

5. Conclusion

In this paper, we rebuild the FSRCNN by adding SE blocks and residual blocks in the mapping layers to solve SISR tasks. Meanwhile, we also substitute the original synthetic training dataset with a training dataset consisting of realistic images. Then, we implement several controlling experiments to explore the impact of residual blocks, MAE and MSE loss function, PReLU and ELU activation function. The results of experiments illustrate that the official FSRCNN achieves better performance on the proposed test datasets while our rebuild models illustrate greater generalization capability. Our analysis illustrates that the residual blocks can slightly improve the performance of shallow networks. In addition, MAE and MSE loss functions, PReLU, and ELU activation functions do not exert a significant impact on our rebuild model.

In the future, our study will focus on further improvement of the rebuild models. Meanwhile, we will try to apply our models to solve real-world image super-resolution tasks and strengthen the possibility of its real-world application.

References

[1] Chunwei T Ruibin Z Zhihao W Yong X Wangmeng Z Chen C and Chia-Wen L 2020 Lightweight image super-resolution with enhanced CNN *arXiv* eess.IV(2020)2007.04344v3
[2] Chao D, Chen C L, Kaiming H, and Xiaoou T Image super-resolution using deep convolutional networks *arXiv* cs.CV (2015)1501.00092v3
[3] Jin Y Shigesumi K and Takio K Fast and accurate image super resolution by deep CNN with skip connection and network in network *arXiv* cs.CV (2017) 1707.05425v7
[4] Jiwon K Jung K L and Kyoung M L Deeply-recursive convolutional network for image super-

resolution *arXiv* cs.CV (2016)1501.04491v2

[5] Christian L Lucas T Ferenc H Jose C Andrew C Alejandro A Andrew A Alykhan T Johannes T Zehan W and Wenzhe S Photo-realistic single image super-resolution using a generative adversarial network *arXiv* cs.CV (2017)1609.04802v5

[6] Zhisheng L Juncheng L Hong L Chaoyan H Linlin Z and Tieyong Z Transformer for single image super-resolution *arXiv* cs.CV (2022)2108.11084v3

[7] Chao D Chen C L and Xiaoou T Accelerating the super-resolution convolutional neural network *arXiv* cs.CV (2016)1608.00367v1

[8] Zhisheng L Juncheng L Hong L Chaoyan H Linlin Z and Tieyong Z Transformer for single image super-resolution *arXiv* cs.CV (2022) 2108.11084v3

[9] Jianrui C Hui Z Hongwei Y Zisheng C and Lei Z Toward real-world single image super-resolution: a new benchmark and a new model *arXiv* cs.CV (2019) 1904.00523

[10] Jie H Li S Samuel Al Gang S and Enhua W Squeeze-and-excitation networks *arXiv* cs.CV (2019) 1709.01507v4

[11] Kaiming H Xiangyu Z Shaoqing R and Jian S Deep residual learning for image recognition *arXiv* cs.CV (2015)1512.03385v1

[12] Jiali W Zhengchun L Ian F Won C Rajkumar K and V. Rao K Fast and accurate learned multiresolution dynamical downscaling for precipitation *arXiv* cs.LG(2021) 2101.06813v1

[13] Hang Z Orazio G Iuri F and Jan K Loss functions for image restoration with neural networks *arXiv* cs.LG(2021) 1511.08861v3

[14] Alain H and Djemel Z Image quality metrics: PSNR vs. SSIM *2010 International Conference on Pattern Recognition*

[15] Honggang C Xiaohai H Linbo Q Yuanyuan W Chao R and Ce Z Real-world single image super-resolution: a brief review *arXiv* eess.IV(2021 2103.02368v1

[16] Yang J Wright J Huang T S and Ma Y Image super resolution via sparse representation *arXiv* IEEE Transactions on Image Processing, vol. 19, no. 11, pp. 2861-2873 (2010)

[17] Djork-Arne´ C Thomas U and Sepp H Fast and accurate deep network learning by exponential linear units (ELUS) *arXiv* cs.LG(2016) 1511.07289v5

[18] Xuezhe M Apollo: an adaptive parameter-wise diagonal quasi-newton method for nonconvex stochastic optimization *arXiv* cs.LG (2016)2009.13586v6

[19] Bevilacqua M Roumy A Guillemot C and Morel M.L.A. Low-complexity single-image super-resolution based on nonnegative neighbor embedding *arXiv* BMVC. (2012)

[20] Zeyde R Elad M and Protter M On single image scale-up using sparse-representations *arXiv Curves and Surfaces* (2012) 711–730

[21] Martin D Fowlkes C Tal D and Malik J A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics *arXiv* ICCV Volume 2 (2001) 416–423