Image Super Resolution

Dinesh Kumar Gnanasekaran, Xinan Ye

Repo Link: https://github.com/dinesh-GDK/image-super-resolution

1. Introduction

Super-resolution refers to the process of using optics and related knowledge to restore image details and other data information based on known image information. Simply put, it is to increase the resolution of an image and improve the image quality. Super-resolution methods can be divided into three categories: interpolation-based methods, reconstruction-based methods, and learning-based methods (i.e., deep learning methods). The method based on interpolation is simple and fast, but the processing effect is poor at pixel mutations such as edges and textures, and it is prone to aliasing and block effects; methods based on reconstruction include frequency and spatial domain methods, but they cannot be simulated well in a real scene; deep learning-based methods use neural networks, which can provide far better performance than other two methods mentioned.

The mainstream deep learning-based methods can be divided into methods based on convolutional neural networks (CNN), methods based on generative adversarial networks (GAN), and methods based on Transformer. In 2014, SRCNN[1] was proposed as the first convolutional neural network in the field of super-resolution. It uses the interpolation method to convert the resolution of low-resolution to high-resolution space and then uses neural networks to reconstruct the enlarged image. In 2015, VDSR[2] proposed to use of the residual deep network to solve the SR problem for the first time. In 2016, ESPCN[3] proposed a sub-pixel convolutional layer, which is based on pixel rearrangement, does not need to upsample LR and uses convolution to gradually restore the target resolution. In 2017, EDSR[4] deleted unnecessary modules of the traditional network (removed redundant BN operations). In 2017, SRGAN[5] introduced GAN into the field of super-resolution, and innovatively proposed content loss. During the learning process, the network pays more attention to the semantic feature difference between the reconstructed image and the original image, rather than the color and brightness difference between pixels. In 2018, MSRN[6] was proposed to improve the residual block and add a multi-scale convolution kernel to realize adaptive detection of image features of different scales. In 2020, Transformer is introduced into the field of image super-resolution. TTSR[7] proposes a super-resolution algorithm based on the Transformer network structure, which makes full use of the texture information of the reference image and proposes a feature fusion mechanism.

This project is based on the idea of SRCNN, uses MSE as the loss function, and updates parameters via gradient descent. We use two different feature extraction networks(U-Net[8] and RRDB-Net[9]) as the backbone for experiments.

2. Problem Statement

The super-resolution task requires us to create a model to upscale an image from low resolution to high resolution. Considering the consumption of calculation time and computing resources, we decided to upgrade the picture to a 2K resolution. We will use DIV2K dataset as the training set, validation set, and test set. We need to choose an appropriate loss function to calculate the loss value, and use this value to observe the training process of the model and decide the final model to be retained.

Picture Signal-to-Noise Ratio (PSNR), Root means square error (RMSE), and Structural Similarity Index (SSIM) are usually used as metrics to evaluate the quality of the super-resolution model. We will improve these indicators as much as possible by modifying the model's network, optimizer, and various parameters such as learning rate to provide a model that can outstandingly complete the task under this data set.

3. Dataset

DIV2K[11] is a single-image super-resolution dataset that was created for NTIRE2017 and NTIRE2018 Super-Resolution Challenge. The dataset contains many high-resolution(HR) images with different dimensions and the corresponding low-resolution(LR) images which are downsampled by many techniques. There are many techniques used for downsampling like bi-cubic, also there are many scales of downsampling like x2, x4, and x8.

3.1 Dataset Handling

For this project, we used the dataset that was degraded using bi-cubic interpolation and scaled down to x2 resolution. The size of the dataset used for training, validation, and testing is given in Table 3.1.

Table 3.1. Dataset split

Dataset	Number of Samples
Train	600
Validation	200
Test	100

3.2 Data Preprocessing

3.2.1 Patch Training

Since image sizes are relatively big we need a lot of GPU memory for training and testing, so instead of using the entire LR and HR images for training, we extract patches from the images. We resize the low-resolution image to the same dimension as the HR image, then we extract patches of size 64x64 from the resized low and high-resolution images with a stride of 64. The

patches of lower-resolution images are resized to 32x32. This reduces the amount of GPU memory used to train the model and the training time of the model can also be reduced. For testing, we use the whole image.

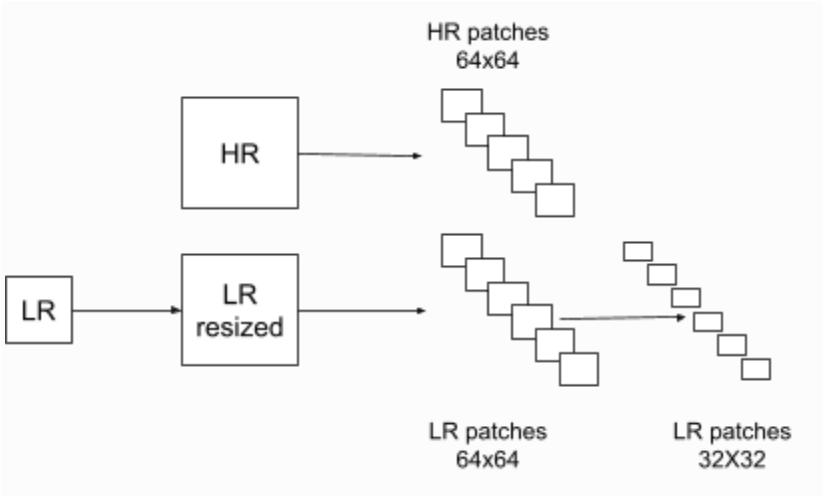


Figure 3.1. Patch creation for training and validation.

3.2.2 Color Channels

Also instead of training the images in RGB color scheme, we can use YUV(chrominance) color scheme. Since the Y channel(luminance) has most of the information on the edges and content of the patches, we are going to use only the Y channel. We develop the model to improve the Y channel of the low-resolution patches using the Y channel of the high-resolution patches as the labels. Later we combine the UV channels with the enhanced Y channel predicted by the model to get back the enhanced image. Finally, we normalize the value of the Y channel.

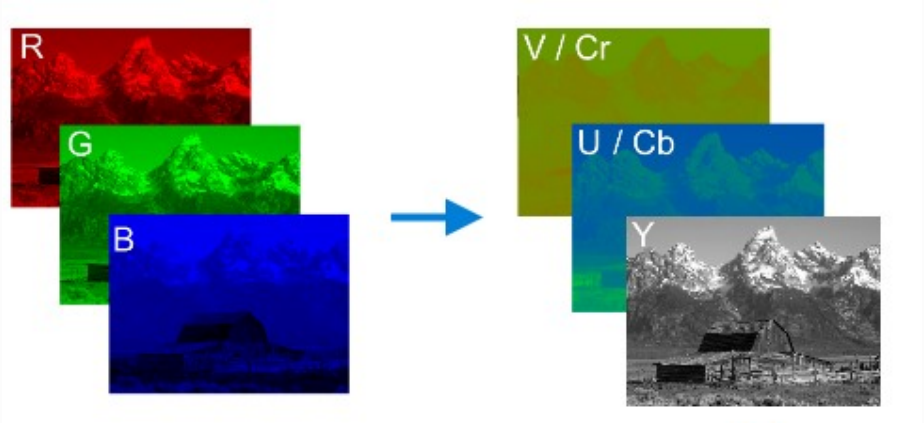


Figure 3.2. RGB and YUV color channels

4. Evaluation metrics

We are going to use Root Mean Squared Error(RMSE), Peak-Signal-to-Noise_Ratio(PSNR), and Structural Similarity Index Measure(SSIM) to evaluate the performance of the models. These are standard metrics used to evaluate image processing algorithms.

$$MSE = \frac{\sum_{M,N} [I_1(m,n) - I_2(m,n)]^2}{M * N}$$

Where I_1 and I_2 are the images with dimensions M and N.

$$RMSE = \sqrt{MSE}$$

$$PSNR = 10log_{10}(\frac{R^2}{MSE})$$

Where R is the maximum fluctuation of the input. In this case, the pixels are using 8 bits so the value is 255.

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_x^2 + c_2)}$$

Where μ_x is average of x, μ_y is average of y, σ_x^2 is variance of x, σ_y^2 is variance of y, σ_{xy} is covariance of x and y. $c_1 = (k_1 R)^2$, $c_2 = (k_2 R)^2$ are used to stabilize the value when there is a weak denominator, usually $k_1 = 0.01$ and $k_2 = 0.03$.

4. Models

The original network in SRCNN [1] has only 3 convolutional layers, including 2 hidden layers and 1 output layer. Due to the limitation of its network depth, its learning ability is very limited, and it basically has no generalization ability. So we chose two networks with greater depth, and both networks use residual connections to reduce the possible problems of gradient explosion and disappearance.

4.2 U-Net

U-Net was first introduced in [8] for biomedical image segmentation. Due to its robustness, it is being used in many applications in various fields. The name U-Net comes from the layers arranged like the alphabet U. It has an encoder-decoder structure, with skip connections. These skip connections allow U-Net to get information from previous layers to decode the encoded parts well. The architecture of the U-Net is given in Figure 4.2.1. Since U-Net requires the input and output data to have the same dimensions we resize the LR patches to the dimension of the HR patches using bi-cubic interpolation.

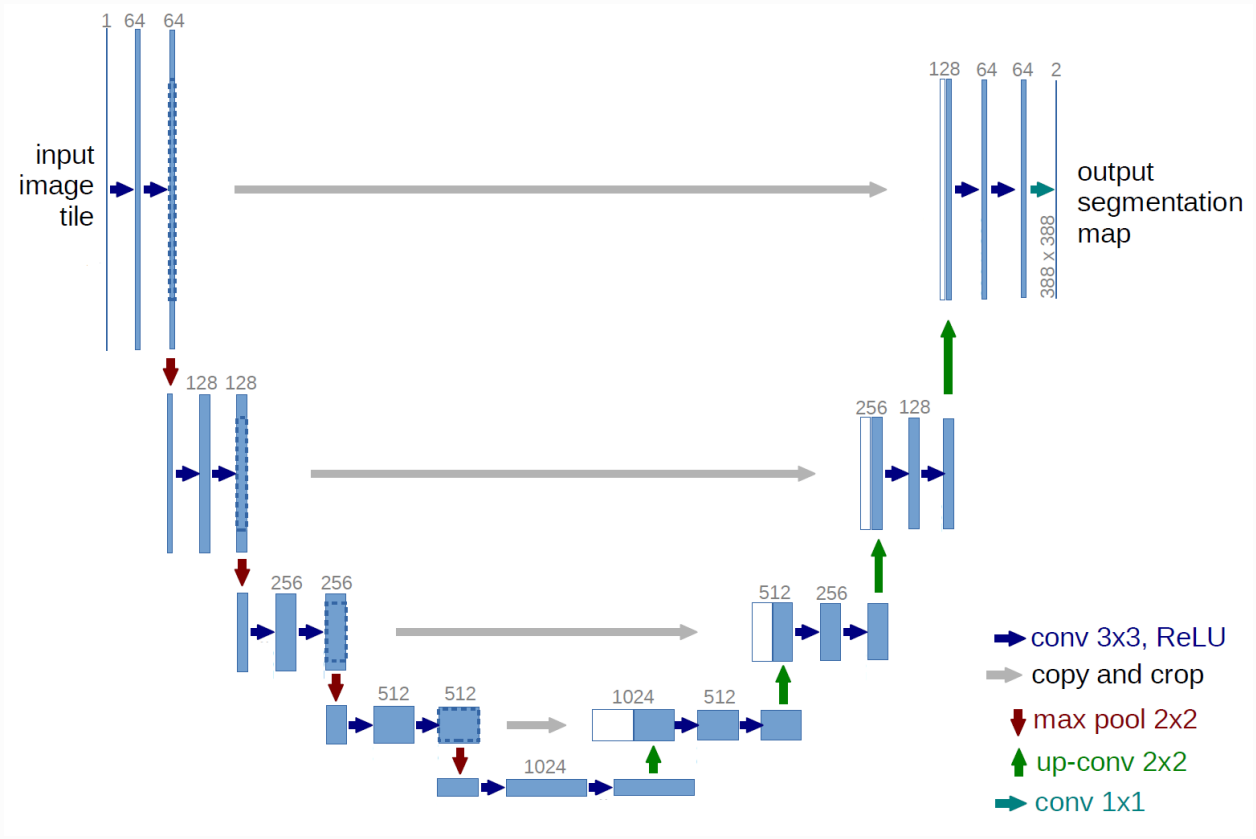


Figure 4.1 The architecture of U-Net

4.3 RRDB-Net

RRDB-Net first passes the low-resolution image through a convolutional layer, sends the data to several layers of basic blocks and the layers after the basic blocks, and then outputs high-resolution images through the upsampling layer and convolutional layers.

The basic block is composed of a dense block. For the specific connection method, see Figure 4.2.2. The Dense block is composed of several convolutional layers and a RELU activation function. The output of each layer must be sent to all layers after this layer at the same time. The author of the original article removed the BN layer because BN layers tend to introduce unpleasant artifacts and limit the generalization ability. Removing BN layers has also proven to increase performance and reduce the computational complexity in different PSNR-oriented tasks including SR. In this project, we use RRDB-Net with one basic block.

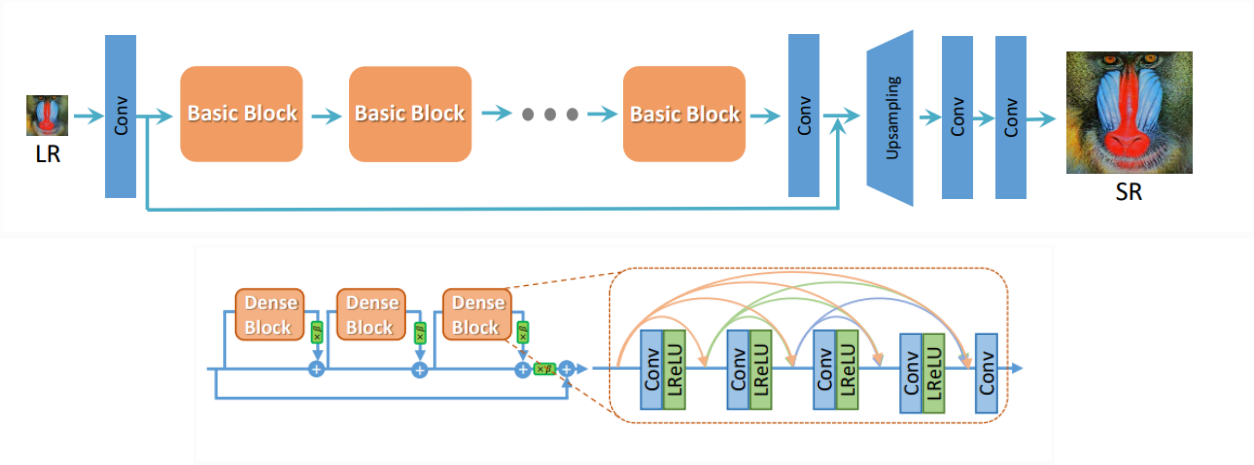


Figure 4.2 The architecture of RRDB-Net

4.3 Challenges faced using other models

Initially we chose the HAT [12] network, a transformer-based architecture, but the GPU memory requirements for training the model were too large, so we could not move forward with the model. And the whole HAT project is encapsulated in a library, so if we want to read the detailed code or modify the lines, it will become very difficult. So we chose a framework that is more transparent and allows us to dig deeper.

5. Results

We used the Adam optimizer with a learning rate equal to 0.0001 and used Mean Squared Error as the loss function with a batch size of 128. We used two NVIDIA Tesla A100 40GB GPUs provided on CARC as computation resources for this task. We turned the training set to .npz format to reduce memory occupation.

5.1 Learning Curves

We first train on U-Net for 280 epochs. The loss first showed a fluctuating downward trend, but the model is overfitting on the validation set after 50 epochs, which may be caused by too small a dataset, too large a network model, and too much number of training epochs when the learning rate is fixed. Then we trained 60 epochs on RRDB-Net, and the loss decreased smoothly on both the training set and the validation set.

In this project, the model with the smallest loss on the validation set is used as the final model for metrics testing and high-resolution image generation.

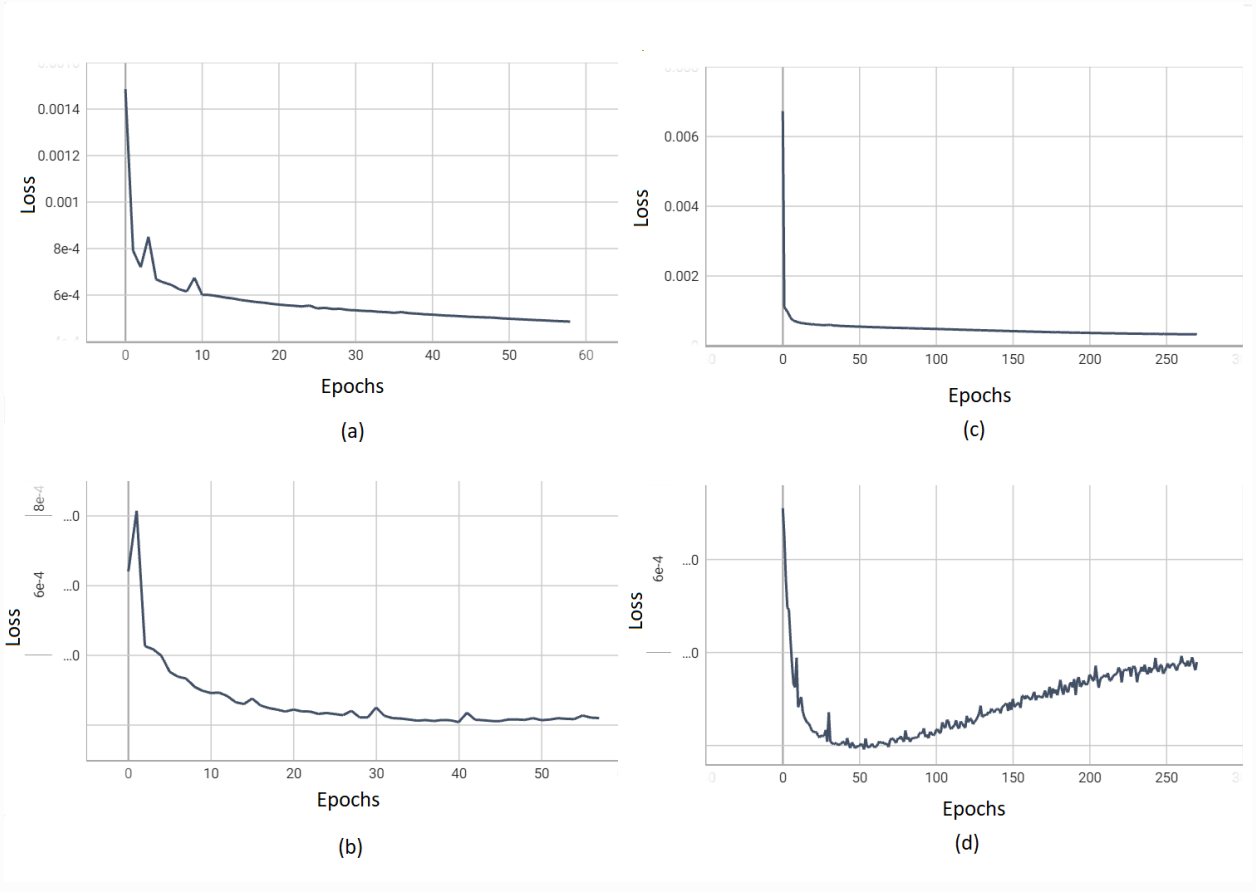


Figure 5.1.1. (a) Training loss of RRDB-Net, (b) Validation loss for RRDB-Net, (c) Training loss of U-Net, (d) Validation loss for U-Net

5.2 Metrics

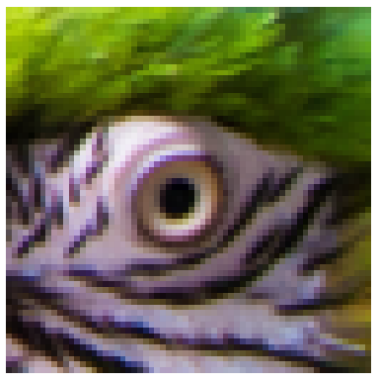
The performance of the models on the test set of 100 images is given in Table 5.2.1. All the metrics are calculated using an HR image as a reference. To compute metrics with the LR images we use bi-cubic interpolation to scale the image to the dimensions of the HR image. As we can see in Table 5.2.1 both deep learning improves the resolution of the images. U-Net outperforms RRDB-Net in terms of all the metrics. Also since U-Net is also smaller and less complex than RRDB-Net it is also faster to train and test.

Table 5.2.1 Summary results on the Test set of 100 images

Image	Mean RMSE	Std RMSE	Mean PSNR	Std PSNR	Mean SSIM	Std SSIM
LR	7.9605	4.1327	31.2770	4.6581	0.9057	0.0560
U-Net	6.0342	3.1438	33.6843	4.6421	0.9353	0.0436
RRDB-Net	7.7102	3.9075	31.5125	4.5656	0.9227	0.0455

5.3 Sample Images

We can see that the details of images after super-resolution have been significantly improved compared with the original low-resolution images, and there is no obvious distortion. To compare with human eyes, the super-resolution results produced by RRDB-Net are better than that by U-Net.



(a)

(b)

(c)

(d)

Figure 5.3.1. (a) LR image (b) HR image (c) Results from U-Net (d) Results from RRDB-Net

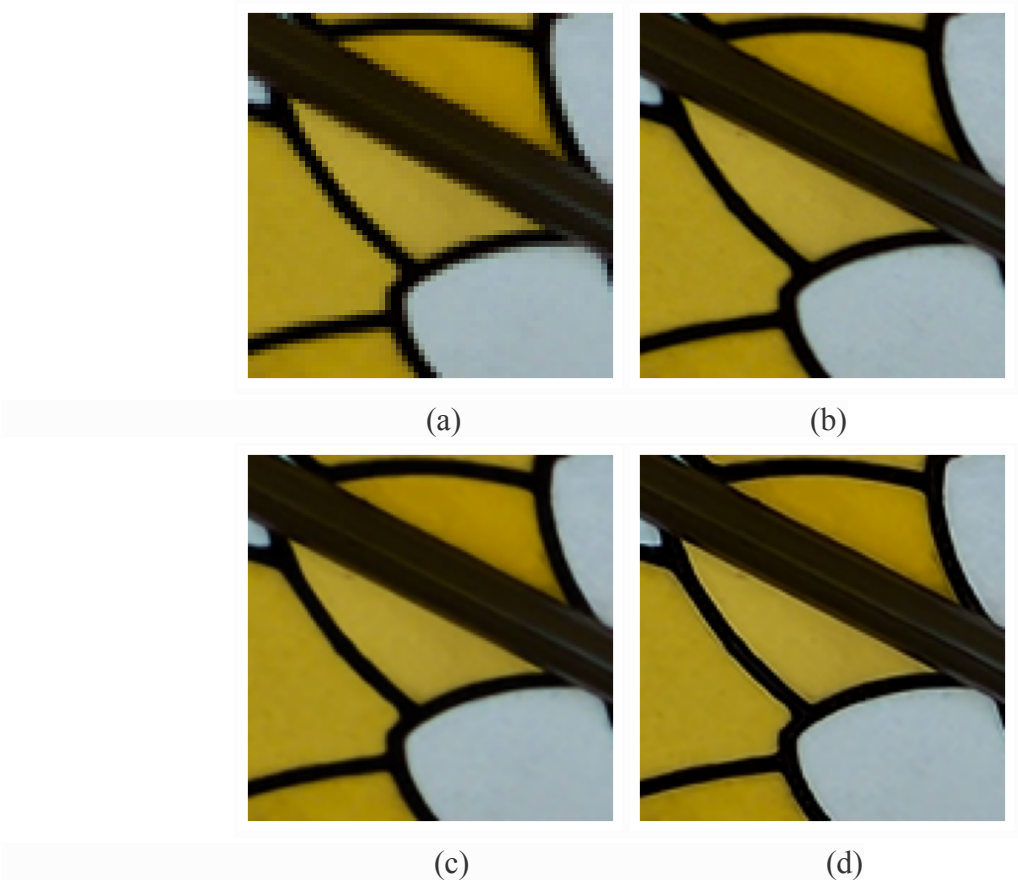


Figure 5.3.2. (a) LR image (b) HR image (c) Results from U-Net (d) Results from RRDB-Net

6. Conclusion

We built a super-resolution framework inspired by SRCNN and tried to use different networks (U-Net and RRDB-Net) as the backbone. Both were successfully trained to upscale images from low resolution to 2K resolution. U-Net is better than RRDB-Net in terms of these metrics. But seeing the enhanced images, RRDB-Net provides better results. Considering that RRDB-Net is not overfitted yet after 50 epochs, it may have the potential to improve performance.

7. Extension

As we saw in section 3.2 on data preprocessing we only enhanced the Y channel to improve resolution, since training a model with three channels requires more GPU memory. In theory, we can use all three channels to enhance the image, which will give better results than enhancing a single channel.

Another way to improve the performance is to compute the 2D FFT of the LR and HR images to train the model, rather than giving the pixel values. Since in FFT we can see the traces of edges of the images more clearly using this technique might improve the results significantly.

8. Reference

[1] Dong, C., Loy, C.C., He, K., Tang, X. (2014). Learning a Deep Convolutional Network for Image Super-Resolution. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds) Computer Vision – ECCV 2014. ECCV 2014. Lecture Notes in Computer Science, vol 8692.

[2] W. Shi, et al., "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), Las Vegas, NV, USA, 2016 pp. 1874-1883.

[3] Kim, Jiwon et al. “Accurate Image Super-Resolution Using Very Deep Convolutional Networks.” 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015): 1646-1654.

[4] Lim, Bee & Son, Sanghyun & Kim, Heewon & Nah, Seungjun & Lee, Kyoung Mu. (2017). Enhanced Deep Residual Networks for Single Image Super-Resolution. 1132-1140. 10.1109/CVPRW.2017.151.

[5] Li, J., Fang, F., Mei, K., Zhang, G. (2018). Multi-scale Residual Network for Image Super-Resolution. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds) Computer Vision – ECCV 2018. ECCV 2018. Lecture Notes in Computer Science(), vol 11212.

[6] Ledig, Christian et al. “Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network.” 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016): 105-114.

[7] Yang, Fuzhi et al. “Learning Texture Transformer Network for Image Super-Resolution.” 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2020): 5790-5799.

[8] Ronneberger, O., Fischer, P., Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In: Navab, N., Hornegger, J., Wells, W., Frangi, A. (eds) Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. MICCAI 2015. Lecture Notes in Computer Science(), vol 9351.

[9] Wang, X. et al. (2019). ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks. In: Leal-Taixé, L., Roth, S. (eds) Computer Vision – ECCV 2018 Workshops. ECCV 2018. Lecture Notes in Computer Science(), vol 11133.

[10] Cai, S., Wu, Y., & Chen, G. (2022). A Novel Elastomeric UNet for Medical Image Segmentation. Frontiers in Aging Neuroscience, 14.

[11] https://paperswithcode.com/dataset/div2k

[12] Xiangyu Chen, Xintao Wang, Jiantao Zhou, Chao Dong: “Activating More Pixels in Image Super-Resolution Transformer”, 2022; [http://arxiv.org/abs/2205.04437 arXiv:2205.04437].