VRDL Homework 4: Image Super Resolution

Zhi-Yi Chin
joycenerd.cs09@nycu.edu.tw

January 13, 2022

1 Introduction

In this homework, we participate in a super-resolution challenge on CodaLab. In this challenge, we perform image super-resolution on the dataset provided by the TA. There are 291 high-resolution images in the training set. There are 14 low-resolution images, which are generated by performing bicubic and shrinking the original image by 1/3 in the testing set. Also, in this challenge, a pretrained model is not allowed. Code is available at https://github.com/joycenerd/image-super-resolution.

2 Methodology

2.1 Data pre-processing

2.1.1 Train validation split

There are 291 images in the original training data, and no validation data was provided. In order to evaluate our super-resolution method, we split the data to 80% for training and 20% for validation.

2.1.2 Offline data augmentation

The first step is to generate low-resolution images since we only have high-resolution images in the training and the validation set. We apply bicubic while resizing the image to 1/3 of the original size to create a low-resolution image aligned with the testing set. The second step is to resize the original high-resolution to [0.8,0.7,0.6,0.5] times and generate the corresponding low-resolution image. We do this augmentation because we hope our trained model can perform well in all kinds of image sizes. The third step is to rotate all the images 180 degrees. We save all the augmentation results locally. Initially, we only had 291 total images for training and validation. After our offline data augmentation, we have 2,330 images for training and 580 images for validation. We think more data is better for training the model.

2.2 Model Architecture

We apply image super-resolution feedback network (SRFBN) [2] as our method. This work was published in CVPR 2019. SRFBN is a single-image super-resolution method that mainly uses the return mechanism to improve the super-resolution effect without introducing too many parameters. The primary purpose is to design a feedback module to return multiple times, as shown in Figure

1. The output of the last feedback and the input of the entire network is re-entered together to the feedback module and continuously returned to a certain level. The advantage of the return is that no additional parameters are added, and multiple returns are equivalent to deepening the network and continuously refining the generated SR image. Although DRRN [3] and DRCN [1] also use the recurrent structure, these networks cannot achieve the front layer to obtain helpful information from the back layer, so there is still a certain difference between the recurrent structure and the feedback structure. Among them, the training strategy based on curriculum learning discussed in the paper is to sort the pictures of the training set according to the degree of difficulty and then train.

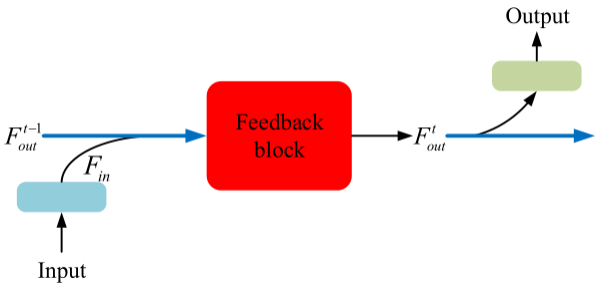

Figure 1: The feedback module of SRFBN.

2.2.1 Why SRFBN?

There are mainly three reasons we apply SRFBN as a super-resolution model. First, we only have limited data, 291 total images for training and validation. This method's official code has a script to generate more data from the existing data (scale and rotate the image). Second, the idea of this method interests us. This method uses an RNN like structure called a feedback network, so previous layers of the network can access helpful information from the following layers. Lastly, some methods cannot use high-level information to refine low-level representations. This method solves this problem by proposing a feedback network with a curriculum-based training strategy to use high-level information back to previous layers and refine low-level encoded information.

2.2.2 How SRFBN?

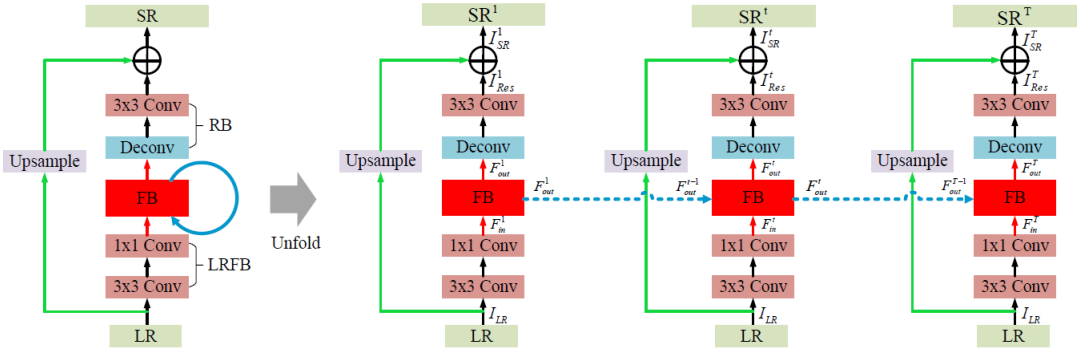

Figure 2: SRFBN network architecture.

Figure 2 shows the whole network architecture of SRFBN. The sub-network placed in each iteration t contains three parts: an LR feature extraction block (LRFB), a feedback block (FB), and a reconstruction block (RB). The weights of each block are shared across time. The LRFB block is used to extract features in the low-resolution image. The FB block receives the hidden state from the previous iteration through a feedback connection. The RB block upscales the low-resolution features to high-resolution to generate a residual image. This method chooses a bilinear upsample kernel. After t iterations, in total T super-resolution images are obtained.

The main contribution of SRFBN is the feedback block, which is shown in Figure 3. The FB contains G projection groups sequentially with dense skip connections among them. Each projection group, which can project HR features to LR, mainly includes an upsample operation and a downsampling operation. In order to reduce the number of parameters and computation efficiency, 1×1 convolution is added to three places in this module. The first place is at the beginning of this method, which is used to refine input low-resolution features. The second place and third place are before the upsampling and downsampling procedure, except for the first projection group.

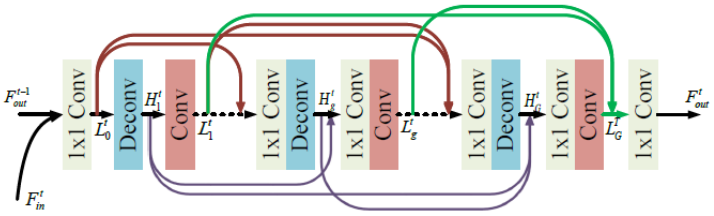

Figure 3: The detail architecture of feedback block.

Another contribution of SRFBN is the curriculum learning strategy. For the single degradation model, all T target SR images are the same. For the complex degradation model, the target SR images are ordered based on the task's difficulty from easy to hard to enforce a curriculum, where curriculum strategy is used. Under this strategy, each output has an equal contribution.

We highly reference Paper99/SRFBN_CVPR2019 [4] for our implementation.

2.3 Hyperparameters

We set the training batch size to 16 same as the paper. We set the training input LR patch size to 13. Although the paper state that the input patch size set to 50 when the scaling factor is 3 yields the best result, but some of our data, especially after the offline data augmentation in 2.1.2 is smaller than 50. We use the smallest image as the LR patch size, which is 13. We set the base feature map's feature size to 64, the number of time steps T to 4, and the number of projection groups G to 6. We use Adam as the optimizer, the initial learning rate to 0.0001, and weight decay to 0. Also, we use MultiStepLR as the learning rate scheduler, which will decrease the learning rate to half after 200, 400, 600, and 800 epochs. We train for 1000 epochs on NVIDIA RTX 1080Ti.

3 Experimental Results

model	PSNR (dB)
SRFBN-S ($G = 3, m = 32$)	28.1438
SRFBN-L ($G = 6, m = 64$)	28.3645

Table 1: This table shows the two best results. m is the base feature map feature size, and G is the number of projection goups.

From the results, we can see that SRFBN-L yields higher PSNR than SRFBN-S in the testing set. We think there are two reasons. First, SRFBN-L is a larger model than SRFBN-S; this can be told from feature size. A larger feature size means more model parameters. Second, larger G leads to higher accuracy due to stronger representative ability of deeper networks.

4 Summary

4.1 Discussion

4.1.1 Adding noise to LR

We have tried to boost the accuracy more using the same model by adding additional noise to the low-resolution image. We have added Gaussian noise with noise level 30 to train the same model. The results are shown in Table 2. The results do not come out so well. After looking at the predicted SR images, we have found the problem. After adding the noise to the LR images, our output SR images do not look sharp and clear; instead, it is a little bit smooth, which is not what we want. We think if we change to add noise in the offline data augmentation can solve the problem. Because in this way, we have both images with or without noise, our model can learn from more.

model	noise	PSNR (dB)
SRFBN-L	-	28.1438
SRFBN-L	Gaussian 30	27.3363
SRFBN-L	Gaussian 10	27.3837

Table 2: The results or adding noise to the input LR image.

4.1.2 GAN-based method

We have looked into the GAN-based method. For many GAN-based methods, the input HR size must be fixed. However, we doubt that if we fix the HR size, resize is needed for most of our input data. If we apply resize, then degradation of the HR image occurs. Our knowledge of these methods is limited, so we did not try this method.

4.2 Conclusion

In this homework, we have applied SRFBN as our super-resolution method. The results come out good. We think the feedback module contributed the most compared to other feedforward networks

at that time. Also, we have tried to degrade the image more by adding Gaussian noise to the input LR image, but this did not work out. We think we can boost the performance more by using the Gaussian noise as the offline data augmentation.

References

[1] J. Kim, J. K. Lee, and K. M. Lee. Deeply-recursive convolutional network for image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 1637–1645, 2016.

[2] Z. Li, J. Yang, Z. Liu, X. Yang, G. Jeon, and W. Wu. Feedback network for image super-resolution. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[3] Y. Tai, J. Yang, and X. Liu. Image super-resolution via deep recursive residual network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[4] F. K. Zhen Li. Feedback network for image super-resolution, 2019. URL https://github.com/Paper99/SRFBN_CVPR19.