DLLAB WEEK 4 Pre-Report : FSRCNN Network Introduction to Troubleshoot SR Problems

Hyunsoo Cha[1], Kwanghoon Sohn[2]
[1]Yonsei University. [2]Yonsei University.

1 Brief Summary

Although SRCNN performed well in Image Super Resolution (SR) testing, the problem is that the computational cost is too high to serve real-time. In this paper, the researchers focused on lightening and accelerating existing SRCNNs by utilizing a shrinking-expanding layer CNN structure that looks like a sand clock. For this purpose, three methods were taken:

First, the researchers insert the deconvolution[4] [5] network into the end of the SRCNN. This allows mapping between high-resolution images with output and low-resolution images with input to form an end-to-end learning network. After shrinking the input feature map, expand the feature map using the deconvolution network. This will allow better extraction of the last feature. This structure is called a sand clock-shaped structure and can be seen as an encoder-decoder form which is not mentioned on paper. This enabled the researchers to implement a model that was 40 times faster[2]. In addition, it was able to have practical computational time for good performance on a typical CPU. In other words, the computation was reduced.

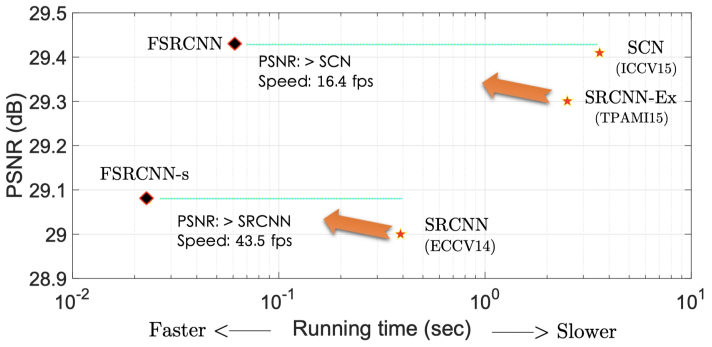


Figure 1: It is the Fast Super-Resolution Convolutional Neural Network (FSRCNN), introduced in the paper by a model with solutions for problem solving. FSRCNN[2] has 40 times faster speed and higher performance than SRCNN-Ex with 6 times more parameters than conventional SRCNN. FSRCNN-s is a lightweight version of FSRCNN.

2 Learn more about FSRCNN

Image Super-Resolution Single image super-resolution (SR) means a task that restores a high-resolution image using a given low-resolution image. SRCNN, the first model to introduce deep learning in SR tasks, already had significant results early on, and based on this, Fast R-CNN and Fast R-CNN could emerge. However, what was pointed out as a disadvantage was that it was too slow to do real-time. On the other hand, the advantage is that it performs superior compared to models without traditional deep learning to solve SR problems as previously mentioned. For example, when upsampling 240×240 images to 720×720, the original SR-CNN has a speed of 1.32 fps while reaching 24 fps, which is 17 times faster than previous one, to achieve real-time performance.

Computation The computational complexity of SRCNN increases by the spatial size of high-resolution images, not low-resolution images. For the upscaling factor n, upscaling of the original low-resolution image increases the computation by n^2[1]. SRCNN computationally goes through non-linear mapping steps. SRCNN first passes patch extraction. Non-linear mapping is followed by reconstruction. In the non-linear phase, a complex mapping is followed by a different high-dimensional high-resolution feature space. According to Dong et al.[1], mapping accuracy can be increased by using the wide width of the mapping layer, but has not benefited from the execution time.

Address SR problem In this paper, the researchers show the following solutions for the two aforementioned problems. First troubleshooting: the deconvolution layer was adopted to replace the bicubic interpolation. To alleviate the computational burden. The deconvolution layer consists of various upsampling kernels that perform the process of producing the last high resolution output. Second troubleshooting: use shrinking layer and expansion layer. The shrinking layer enters the beginning of the mapping layer and the expansion layer is located at the last of the mapping layer.

3 Roles and Characteristics of Multiple Layers

FSRCNN has a total of five stages of network structure as I will mention. Only the last step is the deconvolution layer, and the rest step is the convolution layer. Notation is as follows. f_i is filter size for the ith layer. n_i is the number of filters on the ith layer which means a number of output channels. c_i is a number of input channels on the ith layer. $Conv(f_i, n_i, c_i)$ is a convolution layer. $DeConv(f_i, n_i, c_i)$ is a deconvolution layer Because parameter values are relatively sensitive, the influential factors are shown in SR.

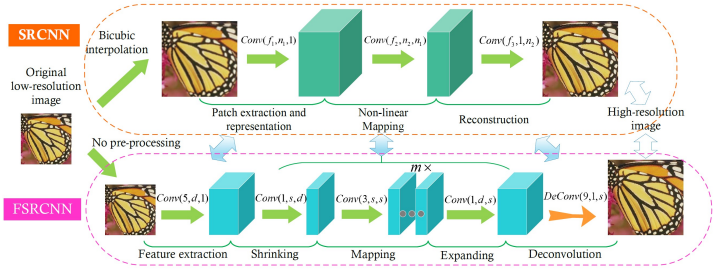


Figure 2: A network structure plot of SRCNN and FSRCNN. FSRCNN differs from SRCNN in three parts.

Feature extraction: $Conv(5,d,1)$ FSRCNN performs original LR images without interpolation and feature extraction. To distinguish it from SRCNN, let's give a small low-resolution input of Y_s. By convolutionizing the first set of filters, 1-pixel overlap becomes a high-dimensional feature vector. In SRCNN, the filter size for the first layer was set to 9. Most Y is interpolated as Y_s, but 5×5 of Y_s contains all the information in the 9×9 patch of Y. Therefore, $f_1 = 5$. The researchers estimate that setting these parameters will result in a small loss of information. The researchers estimate $c_1 = 1$ as SRCNN. Since n_1 can be considered as a number of low-resolution feature dimensions, it can be finally expressed as $Conv(5,d,1)$.

Shrinking: $Conv(1,s,d)$ In SRCNN, mapping steps follow feature extraction steps, so high-dimensional, low-resolution features are mapped directly to high-resolution feature spaces. The phenomenon is observed in high-dimensional vision tasks. In order to reduce the amount of computation, it is sometimes applied as a 1×1 layer.

The researchers fixed the filter size with $f_2 = 1$ so that the filter operates in linear combinations within low-resolution features. The researchers let the low-resolution feature dimension go from d to s. Thus, the shrinking layer can be expressed modular in $Conv(1,s,d)$.

Non-linear mapping: $Conv(3,s,s) \times m$ The researchers had no choice but to trade-off between performance and network size, which they set to $f_3 = 3$, and used a number of 3×3 to replace a single wide layer. This is called a diluted convolutional layer[4]. The number of mapping layers is m. This number determines mapping accuracy and complexity. All mapping layers contain $n_3 = s$. Therefore, non-linear mapping can be defined as $Conv(3,s,s) \times m$.

Expanding: $Conv(1,d,s)$ The expansion layer has the process of reversing the shrinking layer. Shrinking operations reduce the number of low-

resolution feature dimensions for the efficiency of computation. To maintain the constant of the shrinking layer, the researchers added 1×1 layers to equal numbers for low-resolution feature extraction layers. The expansion layer is $Conv(1,d,s)$, which is the same form of exchanging $dands$ in the shrinking layer.

Deconvolution: $DeConv(9,1,d)$ The last one is the deconvolution layer. This layer serves to add upsamples and deconvolution filters to the set of features. The deconvolution performs the opposite operation of the convolution. The researchers saw the benefit by setting it to stride $k = n$. The output is a reconstructed HR image.

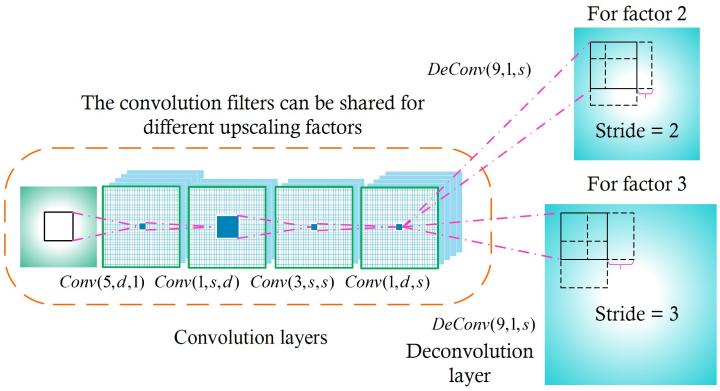


Figure 3: The conv layer shares other upscaling factors. Certain deconvolution layers train different upscaling factors.

PReLU After passing each convolution layer, use the PReLU as the activation function. It differs from ReLU in the negative. ReLU and PReLU can be integrated and organized into the following formulas:

$$f(x_i) = \max(x_i, 0) + a_i \min(0, x_i) \tag{1}$$

x_i is a signal that receives input from the ith channel in the activation function. a_i is a constant of negative parts. In PReLU, this constant is set as a trainable constant. To avoid dead feature, PReLU was used because it causes zero gradient in ReLU. the researchers show that the PReLU activation network is more stable.

Overall Structure The FSRCNN is constructed as Conv(5,d,1)-PReLU-Conv(1, s, d)-PReLU-m×Conv(3, s, s)-PReLU-Conv(1, d, s)-PReLU-DeConv(9, 1, d)[2]. Low resolution feature dimensions d, shrinking filters s, mapping depth In short, FSRCNN can be represented as follows: $FSRCNN(d,s,m)$

Cost Function Similar to SRCNN, mean square error was adopted and used as cost function.

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} ||F(Y_s^i; \theta) - X^i||_2^2 \tag{2}$$

Y_s^i, X^i means the ith low and high-resolution sub-image pair in training data. In addition, $F(Y_s^i; \theta)$ is the parameter θ All parameters are optimized using a stochastic gradient descent. Also, use the usual backpropagation.

4 Experiments

Training dataset Training dataset consists of the following: 91 image datasets have been widely used for training sets based on SR techniques[7]. However, 91 images alone have sufficient difficulty in constructing deep learning models[3]. The researchers produced a General-100 dataset containing 100 bitmap images. In this dataset, the size of the larger image is 710×704, and the size of the smaller image is 131×112. This dataset contains mostly good quality images, with the exception of ambiguous areas such as the sky and the sea. The researchers made two variations[6] to make more efficient use of the dataset. Scaling: Each image was downscaled from 0.6 to 0.9. Rotation: Each image was rotated 90 degrees, 180 degrees, and 270 degrees. Through the above process, the dataset was constructed for training by increasing 19 times more images than the existing dataset. Researchers used Set5, Set14, and BSD200 datasets for testing. A set of 20 images from the BSD500 dataset were used.

Training Samples In training data, the researchers downsampled the original training image by a n scaling factor to form the original training image from the lower resolution image. Then, the researchers cut the $f_{sub} \times f_{sub}$-pixel sub-image into stride k with low resolution training images. The matching high-resolution sub-image was also truncated from the ground true image.

Issue Due to the Padding issue, the researchers found that the padding of the input map or output map did not affect the last performance. As a matter of fact, changes in sub-image size became unnecessary when applied to other networks. Another issue affected by the sub-image size is the deconvolution layer: deconvolution filters generated output using $(nf_{sub} - n + 1)^2$ instead of $(nf_{sub})^2$. Therefore, the researchers cut the boundaries of $n-1$ pixels into high-resolution sub-images.

Training strategy The researchers trained their network with 91-image datasets. When training converges, the researchers added a General-100 dataset for fine-tuning. The researchers initialized the weight of the convolution filters using the PReLU. No activation function was used for the termination.

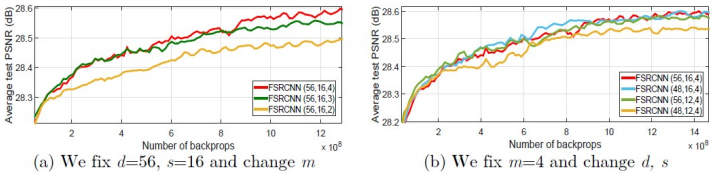
5 Result

Observing results according to parameter changes To test the properties of FSRCNN structures, parameters for low-resolution features are set using mapping depth m, shrinking filters s and dimension d. In addition, a total of 12 experiments were conducted using these numbers.

Settings	$m = 2$	$m = 3$	$m = 4$
$d = 48, s = 12$	32.87 (8832)	32.88 (10128)	33.08 (11424)
$d = 56, s = 12$	33.00 (9872)	32.97 (11168)	33.16 (12464)
$d = 48, s = 16$	32.95 (11232)	33.10 (13536)	33.18 (15840)
$d = 56, s = 16$	33.01 (12336)	33.12 (14640)	33.17 (16944)

Figure 4: The table above is a comparison table of PSNR using Set5 according to parameter settings.

Graph Analysis The table Fig. 4 is analyzed as follows. First, we can see that when d, s is fixed, m becomes 4, the best result compared to other results. This tendency can also be seen from graphs Fig 5. Secondly, the researchers fixed m and conducted experiments accordingly with changes in d, s. The bottom line is that parameters do not always guarantee good results. This tendency can also be seen from graphs. Three large networks can see the tendency to converge together. Based on these results, the researchers can see that the best trade off is FSRCNN (56, 12, 4).



(a) We fix d=56, s=16 and change m (b) We fix m=4 and change d, s

Figure 5: The Schematic of the graph with tendancy

6 Conclusion

I observed the limitations of deep learning based on SRCNN, SR, one of the main problems in the field of computer vision. The researchers implement FSRCNN with high running speed and low loss of recovery quality. The real computational speed and the limitations of computer resources are significant issues in the practical use of deep learning networks. The researchers were able to achieve various goals, including fast speed and similar performance to existing models, by re-designing the SRCNN architecture, and finally the computation of the network was reached with the last acceleration, which was 40 times faster.

[1] 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, 2015.

[2] Chao Dong, Chen Change Loy, and Xiaoou Tang. Accelerating the super-resolution convolutional neural network. In *European conference on computer vision*, pages 391–407. Springer, 2016.

[3] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5197–5206, 2015.

[4] Paul Louis. An introduction to different types of convolutions in deep learning, Jun 2018. URL https://towardsdatascience.com/types-of-convolutions-in-deep-learning-717013397f4d.

[5] Naoki. Up-sampling with transposed convolution, Nov 2017. URL https://naokishibuya.medium.com/up-sampling-with-transposed-convolution-9ae4f2df52d0.

[6] Zhaowen Wang, Ding Liu, Jianchao Yang, Wei Han, and Thomas Huang. Deep networks for image super-resolution with sparse prior. In *Proceedings of the IEEE international conference on computer vision*, pages 370–378, 2015.

[7] Jianchao Yang, John Wright, Thomas S Huang, and Yi Ma. Image super-resolution via sparse representation. *IEEE transactions on image processing*, 19(11):2861–2873, 2010.