Review: Accelerating the Super-Resolution Convolutional Neural Network

Jiho Park
School of Electrical and Electronic Engineering, Yonsei University

Abstract

SRCNN(Super-Resolution Convolutional Neural Network) showed superior performance, but still suffered from expensive computational cost, which made it not capable of real-time(24fps) restoration. This paper proposes FSRCNN(Fast SRCNN), advanced version of SRCNN. First, for upsampling to high-resolution, deconvolutional layer is introduced instead of bicubic interpolation. Second, the non-linear mapping is learned in lower dimension. By shrinking the feature dimension, and by placing upsampling(deconv) layer at the end of the network, the non-linear mapping is learned in low-dimension, and low-resolution. Finally, the mapping layer is replaced by small sized, but more layered filters. These three changes, or developments, led to a speed up of more than 40 times with better SR performance. Also the setting for real-time restoration on generic CPU, is presented.

1 Super-Resolution CNN

As in Fig1, SRCNN consists of Bicubic upsampling , Patch extraction and reprsentation, Non-linear Mapping, and Reconstruction part. Except bicubic interpolation for upsampling, other parts are composed of conv layers. Since upsampling is not learned, SRCNN aims to learn the mapping between bicubic interpolated LR(Low-Resolution) image and HR (High-Resolution) image, which have same output size with input. The patch extraction and representation extracts patches and represents it into high-dimensional feature. The non-linear mapping part maps the feature non-linearly with less feature dimension. The last reconstruction part actually restore the HR image with mapped features. Following time complexity is as in (1).

$$O\{(f_1^2 n_1 + n_1 f_2^2 n_2 + n_2 f_3^2) S_{HR}\} \tag{1}$$

SRCNN is faster than previous models, with the state-of-the-art performance. However it's not fast enough for real-time(24fps) restoration. For example, upsampling 240×240 image by factor of 3, it's speed is 1.32fps.

2 Fast SRCNN

As in Fig1, FSRCNN consists of Feature extraction, Shrinking, Mapping, Expanding, and Deconvolution part. Every part is only composed of conv layers. Comparing with SRCNN, feature extraction matches to patch extraction and representation part, and deconvolution part matches to reconstruction part, and obviously, mapping part matches to non-linear mapping. The calculated time complexity is as (2).

$$O\{(9ms^2 + 2sd + 106d) S_{LR}\} \tag{2}$$

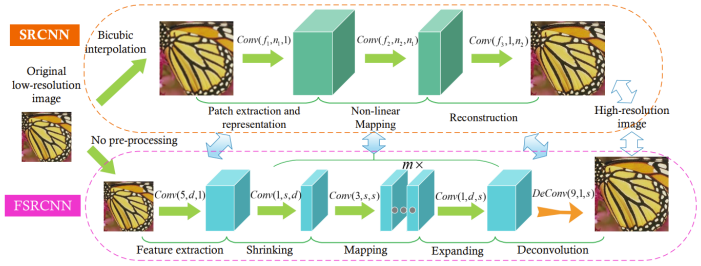

Figure 1: The architecture comparison between FSRCNN(bottom) and SRCNN(top).

2.1 How Acceleration Works?

The author picked up the two limits of SRCNN. The first is the bicubic interpolation. It might be efficient for training since it's not learned, but the bicubic interpolation makes the time complexity grow quadratically with spatial size of the HR image. The second is the costly non-linear mapping step. The non-linear mapping is done with high-dimensional HR feature map. This leads to high mapping accuracy[1] but costs the huge running time.

The previous acceleration attempts in CNN, focused on the redundancy and focused on focused on approximating the well-trained models. However this paper, reformulates the previous model which leads better performance and speed, in different manner.

Specifically, four ideas are proposed to overcome the limits of SRCNN. First is replacing the bicubic upsampling into deconvolutional layer. This might cost more training, but still makes faster forward process.

There are three solutions for second limitation. One is placing the upsampling layer at the end of the network. Then, the mapping happens on low-resolution, whether the upsampling factor is big or small. This also leads to another advantage. Regardless of the upsampling factor, the layers before deconv(deconvolution) layer, will work as same. This theory is proved through the experiment, training the deconv layer with other layers freezed. The second method is 'Shrinking'. The feature dimension is actually shrunken before the mapping part, and expanded back after the mapping. The last is using small sized, but more layered filter. The effectiveness of this concept is shown in many previous CNN researches (e.g. VGGNet [2]). This makes better performance with less parameters.

2.2 FSRCNN Architecture Details

After applying those ideas(Sec 2.1) to SRCNN, we can see an hourglass-shape CNN. The details of each parts are elaborated in this section.

Feature Extraction: The different thing with previous SRCNN is that there is no pre-processing. The feature extraction is done lower resolution. So, the filter size is set as 5, which can cover information of previous 9-sized-filter.

Shrinking and Expanding: Before and after the mapping, the shrinking and expanding layers are placed each. This is realized with 1×1 conv layer, and makes the mapping layer remain the small number of channels.($s << d$).

Non-linear Mapping: Multiple 3×3 conv layers are stacked with shrunken channel number s. The exact number s, d, m are chosen by experiments.

Deconvolution: Single deconv layer with filter size 9, upsamples the image.

PRELU Activation: The Parametric ReLU function($f(x_i) = max(x_i, 0) + a_i min(0, x_i)$) is used. This avoids the "dead feature" since the negative value is reflected in learned manner.

3 Experiments

Identically with SRCNN, MSE loss is used for experiment. For training dataset, mainly 91-image dataset and General-100 dataset are used. The crop after downsampling is applied to make $f_{sub} \times f_{sub}$-pixel training image set. The train is sequentially done in 91-image, General-100 order. General-100 is trainned after the satruation in 91-image.

3.1 Optimal Hyperaprameter

To find the optimal setting of hyperparameter d, s, m, the experiment was designed. Specially for $d = 48, 56$, $s = 12, 16$, $m = 2, 3, 4$, thus 12 models are tested(Fig 2). They found out the best trad-off between performance and parameters $d = 56$, $s = 12$, $m = 4$. Suprisingly even smallest network

achieved PSNR of 23.87dB which is better than SRCNN-Ex, with about 58.3 times acceleration in parameters.

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 2: The performance with different settings.

3.2 Real-Time SR Setting

The number of parameter for real-time implementation is 3976. This is obtained by calculating the ratio between the number of parameters in SRCNN and upscaling factor to speed(fps). The author found moderate configuration $d = 32$, $s = 5$, $m = 1$. This FSRCNN-s(short) reached 24.7fps, while outperforming SRCNN(9-1-5).

3.3 With Different Upscaling Factor

To train different upscaling factor, the whole network of SRCNN must be re-trained. This is because of the change of spatial dimension. Whole conv layers must learn in new manner. However, since the dimension of layers before deconvolution doesn't changes in FSRCNN, this layer doesn't have to be trained again. To prove this theory, training the changed upscaling factor was done by only fine-tuning the deconv layer. As a result, we can check the superior speed of convergence and the performance in Fig 3. This shows that only fine-tunning deconv layer is enough for the change of the upscaling factor.

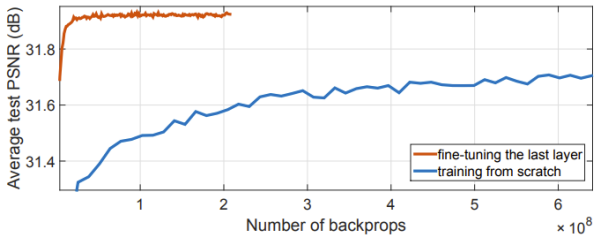

Figure 3: The performance comparison with the sota models.

3.4 Results

FSRCNN and FSRCNN-s were tested with other state-of-the-art models(SRF, SRCNN, SRCNN-Ex, SCN). For upscaling factor 3, FSRCNN was at least 40 times faster than SRCNN-Ex, SRF and SCN. Also, FSRCNN outperformed on PSNR specially in upscaling factor 2, 3 (Fig 4). For upscaling factor 4, FSRCNN achieved lower PSNR than SCN. But, by adopting two models of upscaling factor 2 like SCN, it got comparable performance. Theses models were also tested in different dataset and also in new metrics(e.g. SSIM). As expected, still FSRCNN performed sharper and clearer results. In some other respect, the restoration quality of FSRCNN-s was worse than larger models.

4 Conclusion

By observing the limitation of previous SR models, especially to SRCNN, author explored the method to accelerate the SR convolutional network which

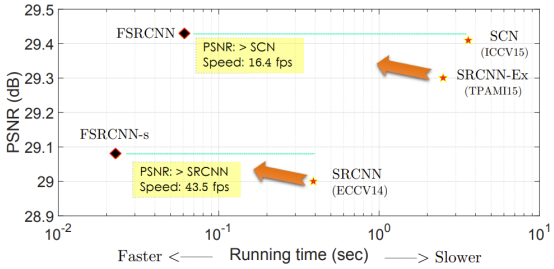

Figure 4: The performance comparison with the sota models.

also improves the performance. The proposed hourglass-shape FSRCNN achieved acceleration of over 40 times, and enable the real-time SR.

Reference

[1] Loy C.C. He K. Tang X Dong, C. Image super-resolution using deep convolutional. *TPAMI*.

[2] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*.