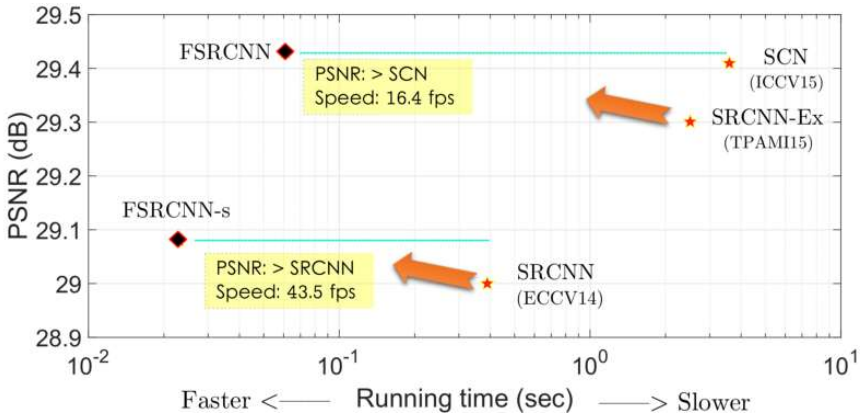
Review: FSRCNN (Super Resolution)

SH Tsang Follow
Oct 27, 2018 · 6 min read

This time, **FSRCNN**, by CUHK, is reviewed. In this paper, a **real-time super resolution approach** is proposed. **Fast Super-Resolution Convolutional Neural Network (FSRCNN)** has been published in **2016 ECCV** with nearly **300 citations** when I was writing this story. (SH Tsang @ Medium)

FSRCNN has a relatively shallow network which makes us easier to learn about the effect of each component. It is even faster with better reconstructed image quality than the previous SRCNN as the figure below.



From SRCNN to FSRCNN

By comparing SRCNN and FSRCNN-s, FSRCNN-s (a small model size version of FSRCNN) has a better PSNR (image quality) and much shorter running time, in which 43.5 fps is obtained.

By comparing SRCNN-Ex (A better SRCNN) and FSRCNN, FSRCNN has a better PSNR (image quality) and much shorter running time, in which 16.4 fps is obtained.

So, let's see how it can achieve this.

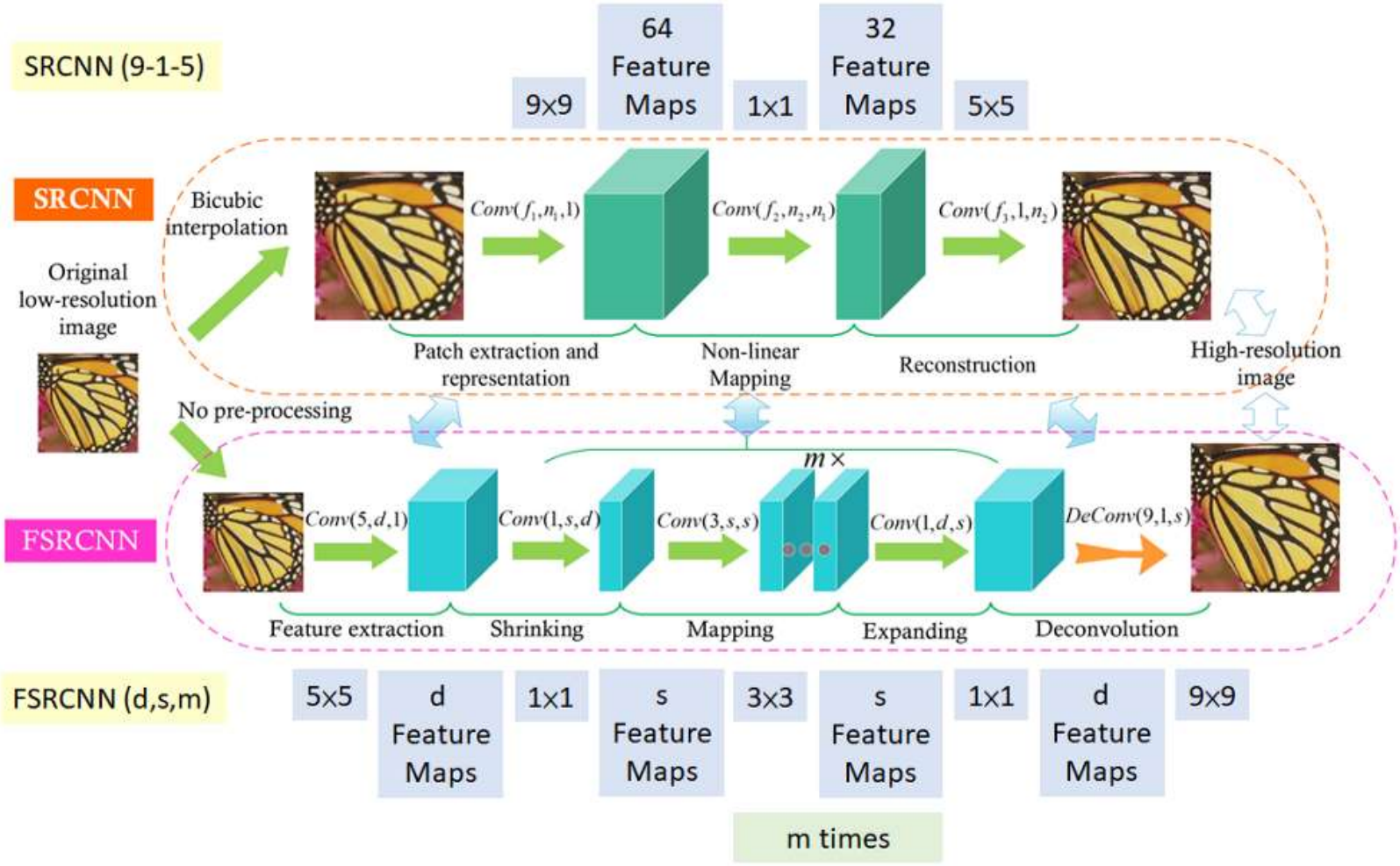
. . .

What Are Covered

1. **Brief Review of SRCNN**
2. **FSRCNN Network Architecture**

3. **Explanation of 1×1 Convolution Used in Shrinking and Expanding**
4. **Explanation of Multiple 3×3 Convolutions in Non-Linear Mapping**
5. **Ablation Study**
6. **Results**

. . .



Network Architecture: SRCNN (Top) and FSRCNN (Bottom)

The above figure shows the network architectures of SRCNN and FSRCNN. In the figure, **Conv(f,n,c)** means the convolution with **f×f filter size** of **n number of filters** and **c number of input channels**.

1. Brief Review of SRCNN

In SRCNN, the steps are as follows:

1. Bicubic interpolation is done first to upsample to the desired resolution.
2. Then 9×9, 1×1, 5×5 convolutions are performed to improve the image quality. For the 1×1 conv, it was claimed to be used for non-linear mapping of the low-resolution (LR) image vector and the high-resolution (HR) image vector.

The computation complexity is:

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

where it is linearly proportional to the size of HR image, SHR. **The larger the HR image, the higher the complexity.**

. . .

2. FSRCNN Network Architecture

In FSRCNN, 5 main steps as in the figure with more convolutions are involved:

1. **Feature Extraction**: Bicubic interpolation in previous SRCNN is replaced by 5×5 conv.
2. **Shrinking**: 1×1 conv is done to reduce the number of feature maps from d to s where s<<d.
3. **Non-Linear Mapping**: Multiple 3×3 layers are to replace a single wide one
4. **Expanding**: 1×1 conv is done to increase the number of feature maps from s to d.
5. **Deconvolution**: 9×9 filters are used to reconstruct the HR image.

The overall structure above is called FSRCNN(d,s,m). And the computational complexity is:

$$O\{(25d + sd + 9ms^2 + ds + 81d) S_{LR}\} = O\{(9ms^2 + 2sd + 106d) S_{LR}\}$$

where it is linearly proportional to the size of LR image, SLR, which is much lower than that of SRCNN.

PReLU is used as activation function. PReLU is the one with parametric leaky ReLU, which is claimed as better than ReLU. (If

interested, please also read my PReLU review.)

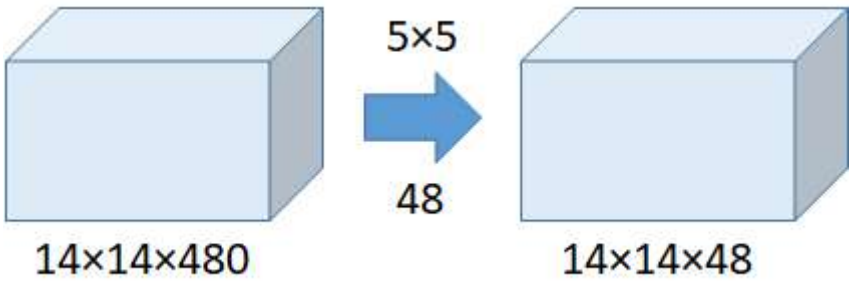
Cost function is just the standard mean square error (MSE):

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

. . .

3. Explanation of 1×1 Convolution Used in Shrinking and Expanding

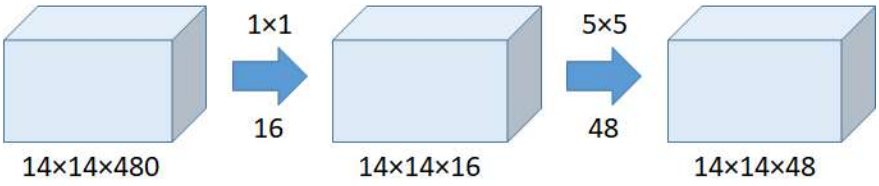
Suppose we need to perform 5×5 convolution **without the use of 1×1 convolution** as below:



Without the Use of 1×1 Convolution

Number of operations = (14×14×48)×(5×5×480) = 112.9M

With the use of 1×1 convolution:



With the Use of 1×1 Convolution

Number of operations for 1×1 = (14×14×16)×(1×1×480) = 1.5M
Number of operations for 5×5 = (14×14×48)×(5×5×16) = 3.8M
Total number of operations = 1.5M + 3.8M = 5.3M
which is much much smaller than 112.9M !!!!!!!!!!!!!!!

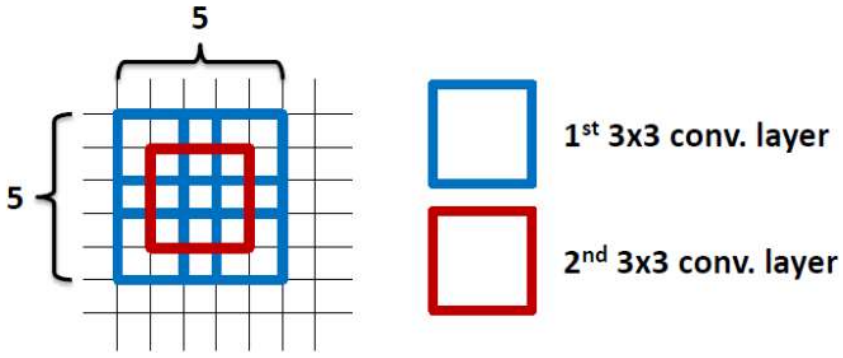
Network-In-Network (NIN) suggested 1×1 conv introduces more non-linearity and improves the performance while GoogLeNet

suggested 1×1 conv helps to reduce the model size while maintaining the performance. (If interested, please read my GoogLeNet review.)

Thus, 1×1 is used in between two convolutions to reduce the number of connections (parameters). By reducing the parameters, we only need fewer multiplication and addition operations, and finally speed up the network. That's why FSRCNN is faster than SRCNN.

. . .

4. Explanation of Multiple 3×3 Convolutions in Non-Linear Mapping



2 layers of 3×3 filters already covered the 5×5 area

By using 2 layers of 3×3 filters, it actually have already covered 5×5 area with fewer number of parameters as in the above figure.

By using **1 layer of 5×5 filter**, number of parameters = **5×5=25**
By using **2 layers of 3×3 filters**, number of parameters = **3×3+3×3=18**
Number of parameters is reduced by 28%

With **fewer parameters** to be learnt, it is better for **faster convergence**, and **reduced overfitting problem**.

This problem has been addressed in VGGNet. (If interested, please read my VGGNet review.)

5. Ablation Study

	SRCNN-Ex	Transition State 1	Transition State 2	FSRCNN (56,12,4)
First part	Conv(9,64,1)	Conv(9,64,1)	Conv(9,64,1)	**Conv(5,56,1)**
Mid part	Conv(5,32,64)	Conv(5,32,64)	**Conv(1,12,64)-4Conv(3,12,12)-Conv(1,64,12)**	**Conv(1,12,56)-**4Conv(3,12,12)**-Conv(1,56,12)**
Last part	Conv(5,1,32)	**DeConv(9,1,32)**	**DeConv(9,1,64)**	**DeConv(9,1,56)**
Input size	S_{HR}	S_{LR}	S_{LR}	S_{LR}
Parameters	57184	58976	17088	12464
Speedup	1×	8.7×	30.1×	41.3×
PSNR (Set5)	32.83 dB	32.95 dB	33.01 dB	33.06 dB

Ablation Study of Each Step

- **SRCNN-Ex**: A better version of SRCNN, with **57184 parameters**.
- **Transition State 1**: Deconv is used, with **58976 parameters,** higher PSNR is obtained.
- **Transition State 2**: More convs are used at the middle, with **17088 parameters,** even higher PSNR is obtained.
- **FSRCNN (56,12, 4)**: Smaller filter sizes and fewer filter number, with **12464 parameters,** even higher PSNR is obtained. The improvement is due to fewer parameters to be trained, easier to converge.

This shows there is contribution for each component.

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)

Study of m, s, d

With higher m (m=4), higher PSNR.

With m=4, d=56, s=12, it has the better tradeoff between the HR image quality (33.16dB) and the model complexity (12464 parameters).

Finally, we got **FSRCNN: FSRCNN (56,12,4).**
And a smaller version, **FSRCNN-s: FSRCNN (32,5,1)**.

6. Results

- Train the network from scratch using 91-image dataset under the upscaling factor 3, then fine-tune the deconvolutional layer only by adding the General-100 dataset under the upscaling factor 2 and 4.

- Data augmentation with scaling: 0.9, 0.8, 0.7, 0.6, and rotation of 90, 180, 270-degree.

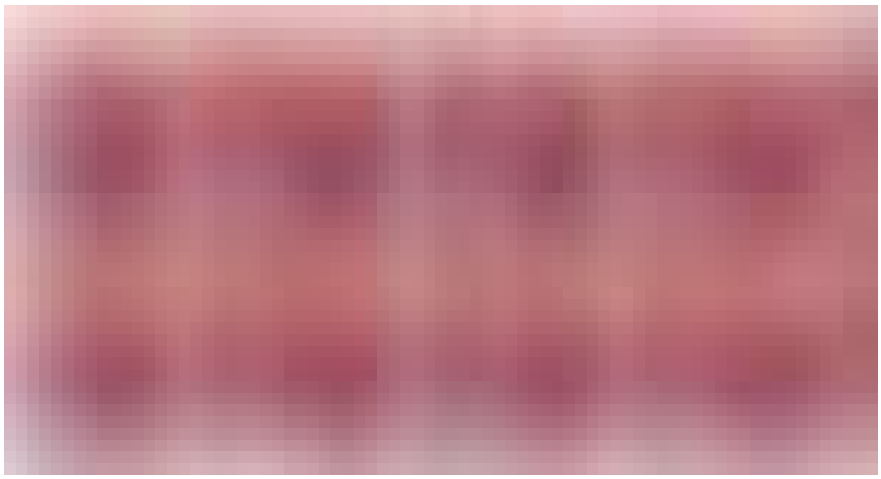
test dataset	upscaling factor	Bicubic		SRF [7]		SRCNN [1]		SRCNN-Ex [2]		SCN [8]		FSRCNN-s		FSRCNN	
		PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time
Set5	2	33.66	-	36.84	2.1	36.33	0.18	36.67	1.3	36.76	0.94	36.53	**0.024**	**36.94**	0.068
Set14	2	30.23	-	32.46	3.9	32.15	0.39	32.35	2.8	32.48	1.7	32.22	**0.061**	**32.54**	0.16
BSD200	2	29.70	-	31.57	3.1	31.34	0.23	31.53	1.7	31.63	1.1	31.44	**0.033**	**31.73**	0.098
Set5	3	30.39	-	32.73	1.7	32.45	0.18	32.83	1.3	33.04	1.8	32.55	**0.010**	**33.06**	0.027
Set14	3	27.54	-	29.21	2.5	29.01	0.39	29.26	2.8	29.37	3.6	29.08	**0.023**	**29.37**	0.061
BSD200	3	27.26	-	28.40	2.0	28.27	0.23	28.47	1.7	28.54	2.4	28.32	**0.013**	**28.55**	0.035
Set5	4	28.42	-	30.35	1.5	30.15	0.18	30.45	1.3	**30.82**	1.2	30.04	**0.0052**	30.55	0.015
Set14	4	26.00	-	27.41	2.1	27.21	0.39	27.44	2.8	**27.62**	2.3	27.12	**0.0099**	27.50	0.029
BSD200	4	25.97	-	26.85	1.7	26.72	0.23	26.88	1.7	**27.02**	1.4	26.73	**0.0072**	26.92	0.019

All trained on 91-image dataset.

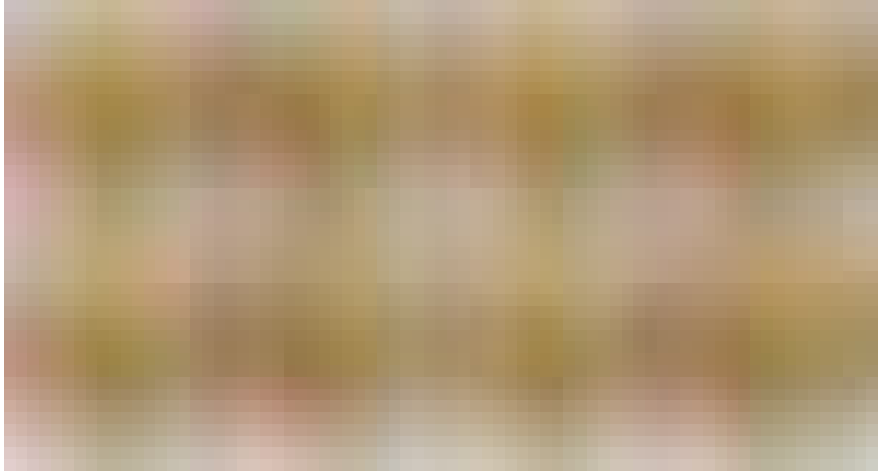
test dataset	upscaling factor	Bicubic	KK [28]	A+ [5]	SRF [7]	SRCNN [1]	SRCNN-Ex [2]	SCN [8]	FSRCNN-s	FSRCNN
		PSNR	PSNR	PSNR	PSNR	PSNR	PSNR	PSNR	PSNR	PSNR
Set5	2	33.66	36.20	36.55	36.89	36.34	36.66	36.93	36.58	**37.00**
Set14	2	30.23	32.11	32.28	32.52	32.18	32.45	32.56	32.28	**32.63**
BSD200	2	29.70	31.30	31.44	31.66	31.38	31.63	31.63	31.48	**31.80**
Set5	3	30.39	32.28	32.59	32.72	32.39	32.75	33.10	32.61	**33.16**
Set14	3	27.54	28.94	29.13	29.23	29.00	29.30	29.41	29.13	**29.43**
BSD200	3	27.26	28.19	28.36	28.45	28.28	28.48	28.54	28.32	**28.60**
Set5	4	28.42	30.03	30.28	30.35	30.09	30.49	**30.86**	30.11	30.71
Set14	4	26.00	27.14	27.32	27.41	27.20	27.50	**27.64**	27.19	27.59
BSD200	4	25.97	26.68	26.83	26.89	26.73	26.92	**27.02**	26.75	26.98

FSRCNN and FSRCNN-s are trained on 91-image and general-100 dataset.

From the results above, FSRCNN and FSRCNN-s work well for upscaling factors 2 and 3. But for upscaling factor 4, FSRCNN and FSRCNN-s are slightly worse than SCN.



Lenna image with upscaling factor 3



Butterfly image with upscaling factor 3

From above figures, we can see that FSRCNN has much clearer image.

• • •

In this paper, with such shallow network, we can know much about the effect of each component or technique, such as 1×1 convolution and multiple 3×3 convolutions.

References

1. [2016 ECCV] [FSRCNN]
   Accelerating the Super-Resolution Convolutional Neural Network

My Reviews

[SRCNN] [PReLU-Net] [GoogLeNet] [VGGNet]