Abstract

기계의 시각에 대해 연구하는 분야인 컴퓨터 비전은 인간의 시각을 능가하곤 한다. 컴퓨터 비전의 작업은 Low, Mid, High -level로 나뉘며, 이번에 다루는 것은 입력과 출력이 이미지인 low-level의 computer vision 중 super-resoltion이다. super-resolution은 저해상도 이미지를 고해상도 이미지로 만드는 것으로, 딥러닝 분야에서 Neural Network에 interpolation과 up-sampling의 방법을 같이 사용하여 연구가 이루어지고 있다. FSRCNN [3] 모델 이전에는 주로 network의 layer를 통과 하기 이전 interpolation이 이루어 졌으며, flexible하지 못하고 연산량이 많아지는 결과를 가져왔다. 하지만 FSRCNN에서는 Deconvolution layer를 도입하여 비교적으로 큰 성과를 이루었다. 앞으로 FSRCNN과 이전 모델들의 차이와 FSRCNN의 구조에 대해 자세히 알아본다.

1 Transposed convolution

transposed convolution에서 주의깊게 봐야할 것은 input matrix와 output matrix 사이의 위치 연결성(positional connectivity)이다. 기존의 convolution은 1*1filter를 이용하거나 padding이 없는 경우 down-sampling으로 그 관계가 many-to-one이다. 반면 transposed convolution은 Up-sampling으로 그 관계가 one-to-many이다.

1.1 Operation of transposed convolution

Transposed convolution은 input에 대한 convolution의 gradient라고 볼 수 있다. 하나의 예시를 통해 transposed convolution의 연산을 수행본다. 4×4 input과 3×3 filter(stride:1, padding:0)를 convolution을 수행하는 경우, 2×2의 output이 나오며 결과는 아래와 같다.

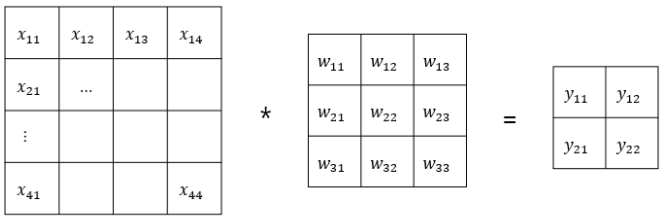

Figure 1: Convolution : 4*4 input, 3*3 filter, 2*2 output

transposed convolution은 convolution에서 input에 대한 output의 gradient로 representaion이 가능하며, Figure1의 예시를 들어 matrix로 표현하면 아래의 (1)과 같다.

$$\begin{bmatrix} \frac{\partial y_{11}}{\partial x_{11}} & \frac{\partial y_{12}}{\partial x_{11}} & \frac{\partial y_{13}}{\partial x_{11}} & \frac{\partial y_{14}}{\partial x_{11}} \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \frac{\partial y_{11}}{\partial x_{44}} & \frac{\partial y_{12}}{\partial x_{44}} & \frac{\partial y_{13}}{\partial x_{44}} & \frac{\partial y_{14}}{\partial x_{44}}, \end{bmatrix} \tag{1}$$

이 행렬을 다시 표현하면 filter의 가중치로 표현되며 (2)와 같다. (16×4 matrix가 맞지만 공간 상 4×16 matrix로 나타냄)

$$= \begin{bmatrix} w_{11} & w_{12} & w_{13} & 0 & w_{21} & w_{22} & w_{23} & 0 & w_{31} & w_{32} & w_{33} & 0 & 0 & 0 & 0 & 0 \\ 0 & w_{11} & w_{12} & w_{13} & 0 & w_{21} & w_{22} & w_{23} & 0 & w_{31} & w_{32} & w_{33} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & w_{11} & w_{12} & w_{13} & 0 & w_{21} & w_{22} & w_{23} & 0 & w_{31} & w_{32} & w_{33} & 0 \\ 0 & 0 & 0 & 0 & 0 & w_{11} & w_{12} & w_{13} & 0 & w_{21} & w_{22} & w_{23} & 0 & w_{31} & w_{32} & w_{33} \end{bmatrix} \tag{2}$$

이 행렬은 output을 reshape한 4×1 matrix와 곱해져 16×1 matrix가 만들어진다. 이 matrix를 다시 4×4 행렬로 reshape하여 Up-sampling을 마무리한다.

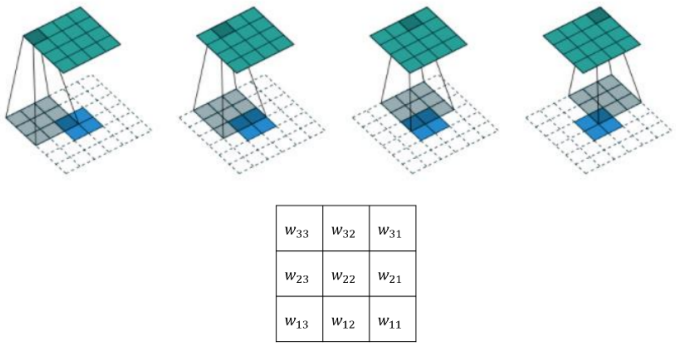

Figure 2: Transposed convolution filter of transposed convolution

transposed convolution의 filter와 output을 이 filter로 convolution하는 과정은 Figure2에서 확인할 수 있다. 위에서 연결성을 강조했는데, convolution 과정을 살펴보자. input의 x_{12}와 관련된 output은 y_{11}, y_{12}이며, 각각 가중치 w_{12}, w_{11}를 통해 연결성을 가지고 있다.Figure2의 transposed convolution이 진행되는 2번 째 그림에서 x_{12}의 자리가 y_{11}, y_{12}이 각각 w_{12}, w_{11}와 곱해지는 것을 확인할 수 있다.

1.2 Transposed convolution을 사용하는 이유

딥러닝에서 Up-sampling을 위해 보간법을 사용하는 경우, 네트워크의 architecture를 결정할 때 어떤 Interpolation을 사용해야 최적의 결과를 끌어낼 수 있는 지 알 수 없다. 즉, 보간법은 모델에서 학습을 할 수 없으며 pre-processing으로 처리된다는 것이다.
보간법을 대신하여 사용되는 방식으로, Transposed convolution은 Up-sampling을 최적으로 하기 위해 사용되며, 학습가능한 parameter가 존재하며 전처리를 하지 않아도 된다.

2 FSRCNN

2.1 SRCNN의 한계

최초로 딥러닝을 이용한 새로운 super-resolution을 제안한 SRCNN [1] [2]은 간단한 구조로 당시 대부분의 SR알고리즘에서 주목할만한 성능을 보였다. 하지만 SRCNN의 속도를 제한하는 2가지 한계가 존재한다. Figure3에서 SRCNN의 구조를 보면 확인할 수 있는 Bicubic interpolation과 non-linear mapping layer이다.

2.1.1 pre-processing

SRCNN 혹은 SR을 deep learning을 통해 하는 이전의 model은 네트워크에 입력으로 들어가기 이전에, 원본의 LR 이미지에 bicubic interpolation을 진행하였다. 이에 따라 up-sampling되어 HR이미지를 input으로 넣기 때문에 convolution의 계산 복잡도가 n에서 n^2으로 매우 크게 증가하였다.

2.1.2 mapping layer

mapping layer는 네트워크 parameters의 대부분을 차지한다. SRCNN의 input image patched는 high-dimensional LR feature space에 투영된 후에 HR feature space에 매핑된다. mapping layer가 dimension이 클수록 정확도가 향상되기는 하지만 시간이 많이 소요되는 단점이 있다.

2.2 main concept

FSRCNN은 SRCNN과 달리 pre-processing을 하지 않고 원래의 저해상도 이미지로 학습하며, 네트워크 마지막에 deconvolution layer를 사용한다. 미리 interpolation을 하여 해상도를 높이는 작업이 없어졌기 때문에 계산 복잡도가 원래의 LR image의 공간 사이즈 n에만 비례한다.
또한, 기존의 성능을 유지하며 네트워크의 크기를 줄이는 방법으로 shrinking layer와 expanding layer를 각각 mapping layer의 시작과 끝에 추가한다. 또한 기존의 wide한 mapping layer를 3*3 filter로 대체하여, depth를 증가시켰다. 더 작은 filter를 사용하고, 더 많은 layer를 쌓음으로써 빠르고 성능이 더 좋은 모델을 구현할 수 있다.

2.3 Architecture of FSRCNN

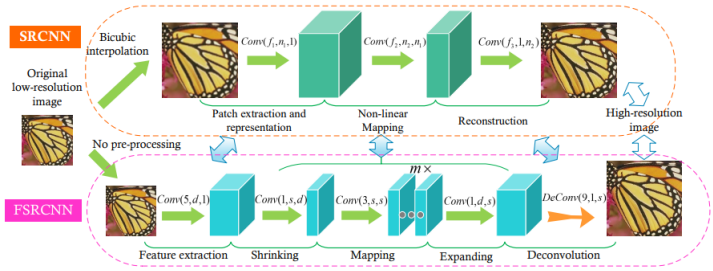

Figure 3: SRCNN과 FSRCNN의 구조

FSRCNN은 3개의 part로 나누어진 SRCNN과 달리 5가지의 부분으로 나눌 수 있다. 앞의 4개는 convolution layer이며 마지막 하나는 deconvolution layer이다. FSRCNN network는 $FSRCNN(d,s,m)$으로 representation 할 수 있다. d는 LR feature dimension, s는 shrinking filter의 개수, m은 mapping layer의 depth이다. $FSRCNN(d,s,m)$를 더 자세히 나타낸 전체 네트워크는 다음과 같이 구성된다. 각 layer의 자세한 설명은 아래에서 다룬다. $Conv(5,d,1)-PReLU-Conv(1,s,d)-PRuLU-m\times Conv(3,s,s)-PReLU-Conv(1,d,s)-PReLU-Deconv(9,1,d)$.

2.3.1 Feature extraction

input image가 interpolation되지 않은 원본 이미지이다. 따라서, 기존에는 9×9 patch가 필요한 반면 원본 이미지는 5×5 patch만으로 약간의 little information loss가 있지만, 모든 정보를 커버할 수 있다. Feature extraction layer는 $Conv(5,d,1)$로 표현되며 d는 sensitive variable이고, 본 논문에서는 주로 48, 56을 사용하고 있다.

2.3.2 Shrinking

기존 SRCNN은 high-dimensional features를 그대로 mapping한다. 이에 따른 문제로 feature의 dimension이 크다는 것이고, 이를 바로 mapping하기에는 계산 복잡도가 매우 높다. 따라서 FSRCNN은 1×1 layer를 적용하여 dimension을 줄인다. 따라서, 계산 복잡도를 줄여 학습 시간을 많이 단축시킬 수 있다. Shrinking layer는 $Conv(1,s,d)$이며, s는 이전의 dimension인 d의 값보다 작다($s<<d$).

2.3.3 Non-linear mapping

딥러닝을 사용한 Super-Resolution의 성능에 가장 중요한 부분으로 두 가지 factors인 width와 depth가 영향을 많이 미친다. SRCNN에서는 5×5 layer를 사용하였지만, FSRCNN에서는 3×3 layer를 사용하여 layer의 개수(depth)를 증가시켰다. 이때, 각 layer의 filter 수는 s로 동일하다. layer의 개수를 m이라고 할 때 representation은 $m\times Conv(3,s,s)$이다.

2.3.4 expanding

shrunk features의 low-dimensional feature를 사용하여 그대로 Hing resolution image를 추출한다면 restoration quality가 떨어지므로, mapping이후 expanding layer를 넣어준다. shrinking layer와의 연관성을 유지하여 1×1 filters를 사용하며, shrinking layer를 통과 하기 이전의 상태와 동일하게 하기 위해, 즉 대칭을 이루어 $Conv(1,d,s)$로 구성한다.

2.3.5 deconvolution

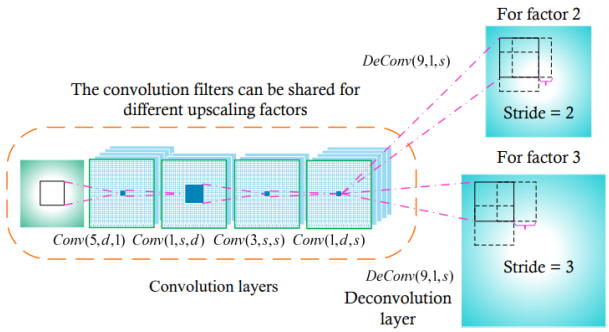

Figure 4: FSRCNN의 Performance

FSRCNN의 마지막 부분으로 이전의 features를 high resolution image로 재구성한다. SRCNN의 첫 번째 layer에서 HR image를 받아 9×9 filter를 적용시키는데, 이와 일관성을 유지하여 deconvolution layer의 kernel의 size를 동일하게 적용하며, factor로 stride를 적용하여 다른 크기로 이미지를 upsampling할 수 있다.

3 Conclusion

	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

Figure 5: FSRCNN의 Performance

Figure4는 SRCNN model에서 속도와 성능을 제한하는 요소를 바꾸며 최종적으로 FSRCNN을 구성하였을 때 보여주는 성능을 나타내는 사진이다. interpolation을 하지 않고 Deconvolution을 함으로써 속도를 8.7배 빠르게 하였고, mapping layer의 전후로 1×1 filter를 넣음으로써 파라미터 수를 크게 줄여 기존 SRCNN 모델 대비 30.1배 속도 향상이 있었다. 또한 불필요한 파라미터를 제거하고, 첫 번째 convolution layer의 filter size를 줄여 최종적으로 41.3배의 속도향상과 성능 증가까지 보여주었다.
interpolation을 전처리로 하는 이전의 모델과 달리 deconvolution layer를 사용함으로써 얻는 이점은 가장 큰 이점은 학습시간이라 생각한다. 최초의 학습 이외에도 deconvolution의 up-sacaling factor를 조정하는 것이 잘 학습된 convolution layer를 바꾸지 않고(재 학습하지 않고)도 가능하다. 이는 훈련과 테스트 모두 시간을 단축시켜 줄 수 있다.

[1] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In *European conference on computer vision*, pages 184–199. Springer, 2014.

[2] 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.

[3] 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.