Densely Connected Convolutional Networks

Gao Huang*
Cornell University
gh349@cornell.edu

Zhuang Liu*
Tsinghua University
liuzhuang13@mails.tsinghua.edu.cn

Laurens van der Maaten
Facebook AI Research
lvdmaaten@fb.com

Kilian Q. Weinberger
Cornell University
kqw4@cornell.edu

24/01/07
배성은

Abstract & Introduction

CNN이 깊은 구조를 가지게 되면서 여러 문제들이 대두되기 시작
- input이 많은 Layer를 통과하면서, 네트워크 끝단에 다다를수록 정보가 소실된다는 점

이런 문제를 해결하기 위해 여러 연구를 진행하였음
공통점
- 모두 앞 Layer를 뒷 Layer로 short path를 사용한다.

이러한 아이디어를 가지고 connectivity pattern을 제안
네트워크 사이의 maximum information flow를 보장하기 위해, 모든 Layer를 서로 연결한다.

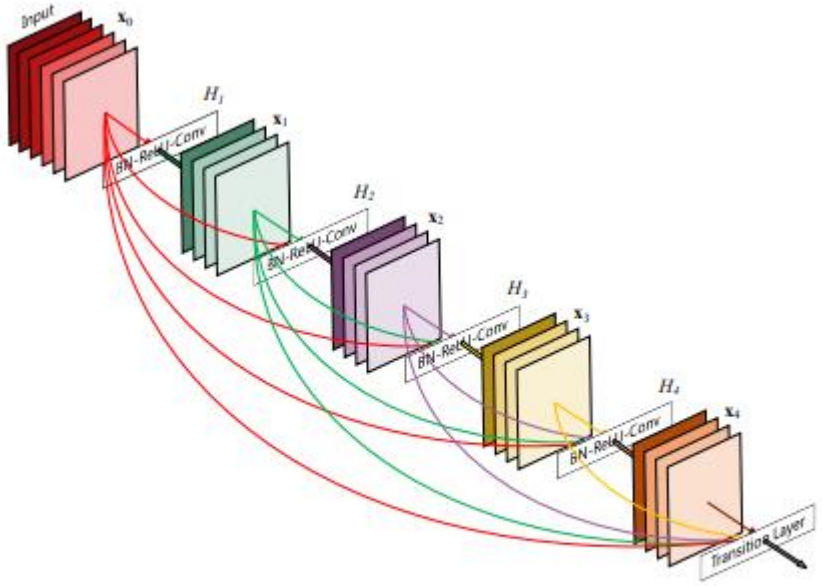


Figure 1: A 5-layer dense block with a growth rate of $k = 4$.

ResNet과의 차이점

ResNet은 Feature들이 Layer로 전달되기 전, summation을 통해 결합
(L개의 connection)

DenseNet은 Featrue들을 concatenation을 하여 결합
(L(L+1)/2개 connection)

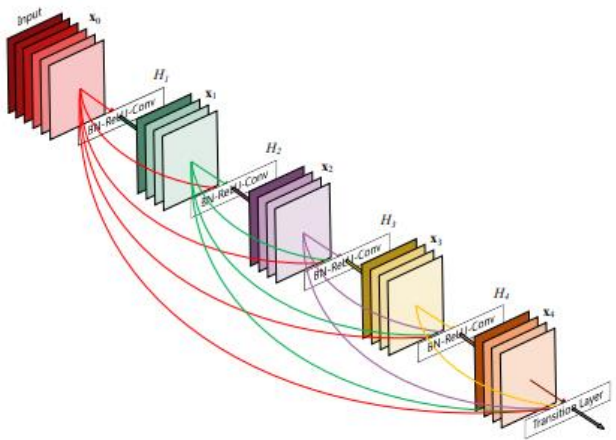
Abstract & Introduction

DenseNet의 장점

1) Fewer Parameter
   불필요한 feature map을 재학습할 필요가 없어 더 적은 parameter를 가짐

2) Information Preserved
   네트워크에 추가되는 정보와 보존되는 정보를 명확히 구분하여 정보를 보존
   그리고 좁은 형태 (layer당 12개 필터) 각 층이 적은 양의 새로운 특징만을 학습.

3) Improve Flow of Information and Gradient
   training에 용이하도록 정보와 gradient의 flow를 개선
   각 Layer들은 Loss function과 original input signal의 gradient에 직접적으로 접근 가능
   overfitting방지하는 regularization효과



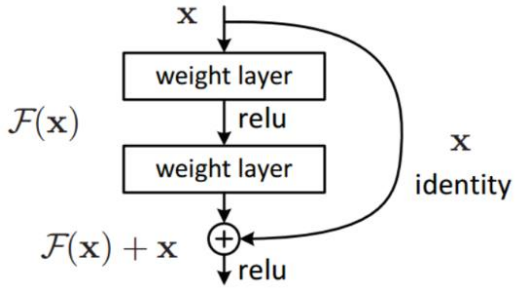
DenseNets

<ResNets>

Identity function을 통해 non-linear transformation을 건너뛰는 skip connection을 추가한다.
(skip connection = Residual Networks)

$$\mathbf{x}_\ell = H_\ell(\mathbf{x}_{\ell-1}) + \mathbf{x}_{\ell-1}.$$

Skip Connection은 이전 Layer의 정보를 직접적으로 Direct하게 이용하기 위해 이전 층의 입력(정보)를 연결한다는 개념이다.



장점 : 나중층에서 더 이전층으로 gradient가 직접 흐를 수 있다는 것
그러나 H의 output이 summation을 통해 합쳐지면서 네트워크의 information flow를 방해할 수도 있다.

DenseNets

<Dense connectivity>

Layer들간의 information flow를 향상시키기 위해 connectivity pattern을 제안
어떤 Layer던지 subsequent layer로 직접 연결하는 모든 Layer를 연결하는 방식

결과적으로 n번째 Layer는 이전의 모든 Layer들의 feature map을 input으로 받게 된다.

$$\mathbf{x}_\ell = H_\ell([\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{\ell-1}]),$$

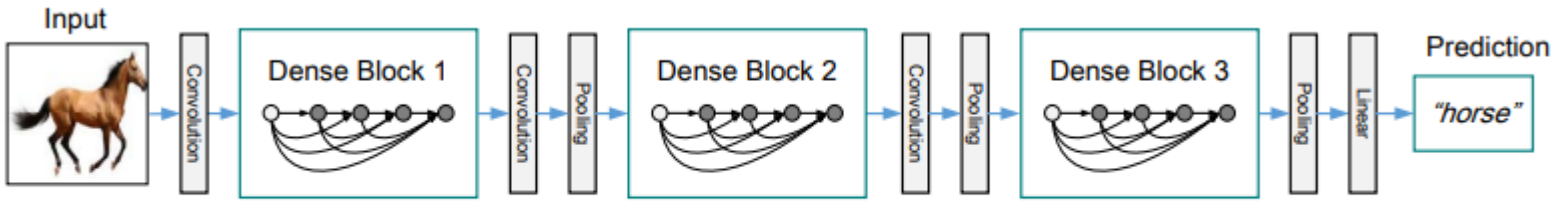


Figure 2: A deep DenseNet with three dense blocks. The layers between two adjacent blocks are referred to as transition layers and change feature-map sizes via convolution and pooling.

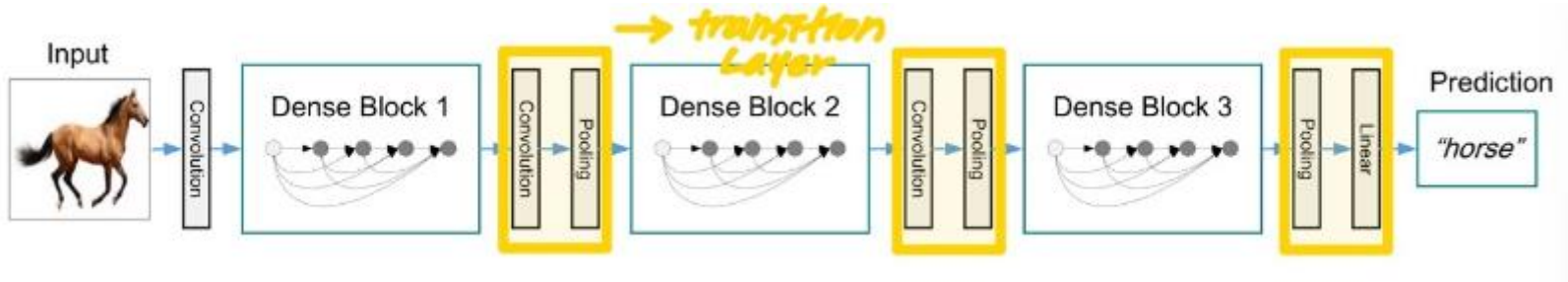
DenseNets

<Composite Function>

다른 연구에서 영감을 받아, H(n)을 다음 3가지 operation의 composite function으로 정의
Batch Normalization, RELU, 3X3 Convolution

<Pooling Layers>

앞의 식에서 사용된 concatenation operation은 feature map의 사이즈가 바뀌면 사용할 수 없다.
하지만, convolutional network의 down-sampling Layer를 통해 feature map의 사이즈를 바꿀 수 있다.
각 Block사이에 Convolution과 pooling을 수행하는 Layer를 Transition Layer라고 한다.
Batch normalization layer, 1X1 convolutional Layer, 2x2 pooling Layer로 구성



DenseNets

<Bottleneck Layers>

각 Layer들은 K개의 output feature map을 생성하지만, 일반적으로 그에 비해 많은 input이 필요하다.
Bottleneck구조는 1x1 convolution을 통해 3x3 convolution에서 input을 줄이면서 computational effciency를 높일 수 있다.

<Compression>

모델의 compactness를 향상시키기 위해, transition layer의 feature map의 개수를 줄일 수 있다.
만약 dense block이 m개의 feature map을 가지고 있다면,
그 뒤의 transition Layer는 θm개의 output feature map을 반환 (θ:compression factor) (0< θ<=1)

DenseNets

<Implementation Details>

Layers	Output Size	DenseNet-121	DenseNet-169	DenseNet-201	DenseNet-264
Convolution	112 × 112	7 × 7 conv, stride 2			
Pooling	56 × 56	3 × 3 max pool, stride 2			
Dense Block (1)	56 × 56	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6
Transition Layer (1)	56 × 56	1 × 1 conv			
	28 × 28	2 × 2 average pool, stride 2			
Dense Block (2)	28 × 28	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12
Transition Layer (2)	28 × 28	1 × 1 conv			
	14 × 14	2 × 2 average pool, stride 2			
Dense Block (3)	14 × 14	[1 × 1 conv; 3 × 3 conv] × 24	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 48	[1 × 1 conv; 3 × 3 conv] × 64
Transition Layer (3)	14 × 14	1 × 1 conv			
	7 × 7	2 × 2 average pool, stride 2			
Dense Block (4)	7 × 7	[1 × 1 conv; 3 × 3 conv] × 16	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 48
Classification Layer	1 × 1	7 × 7 global average pool			
		1000D fully-connected, softmax			

Table 1: DenseNet architectures for ImageNet. The growth rate for all the networks is $k = 32$. Note that each "conv" layer shown in th table corresponds the sequence BN-ReLU-Conv.

Experiments

1. Datasests
   <CIFAR>
   두개의 CIFAR dataset(32x32 pixel의 color이미지) (CIFAR-19, CIFAR-100)
   Training, test set 각각 50,000, 10,000개, training set중 5,000개는 validation에 사용
   <SVHN>
   SVHN (32x32 pixel color 이미지)
   Training, test set 각각 73,257, 26,032개, training set중 6,000개는 validation에 사용

2. Training
   Optimization : SGD
   Batch size : 64
   Epoch : 300
   Learning rate : 0.1
   Wight Decay : 0.0001
   Dropout : 0.2

Experiments

3. Classification Result on CIFAR

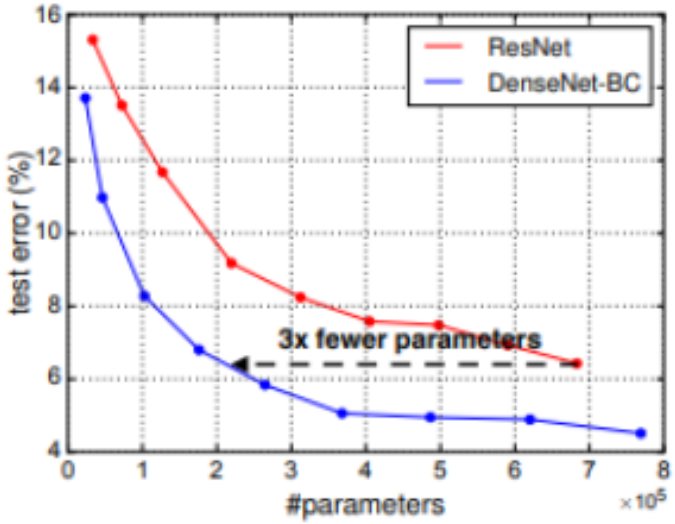
Method	Depth	Params	C10	C10+	C100	C100+	SVHN
Network in Network [22]	-	-	10.41	8.81	35.68	-	2.35
All-CNN [32]	-	-	9.08	7.25	-	33.71	-
Deeply Supervised Net [20]	-	-	9.69	7.97	-	34.57	1.92
Highway Network [34]	-	-	-	7.72	-	32.39	-
FractalNet [17]	21	38.6M	10.18	5.22	35.34	23.30	2.01
with Dropout/Drop-path	21	38.6M	7.33	4.60	28.20	23.73	1.87
ResNet [11]	110	1.7M	-	6.61	-	-	-
ResNet (reported by [13])	110	1.7M	13.63	6.41	44.74	27.22	2.01
ResNet with Stochastic Depth [13]	110	1.7M	11.66	5.23	37.80	24.58	1.75
	1202	10.2M	-	4.91	-	-	-
Wide ResNet [42]	16	11.0M	-	4.81	-	22.07	-
	28	36.5M	-	4.17	-	20.50	-
with Dropout	16	2.7M	-	-	-	-	1.64
ResNet (pre-activation) [12]	164	1.7M	11.26*	5.46	35.58*	24.33	-
	1001	10.2M	10.56*	4.62	33.47*	22.71	-
DenseNet ($k = 12$)	40	1.0M	**7.00**	5.24	**27.55**	24.42	1.79
DenseNet ($k = 12$)	100	7.0M	**5.77**	**4.10**	**23.79**	**20.20**	1.67
DenseNet ($k = 24$)	100	27.2M	**5.83**	**3.74**	**23.42**	**19.25**	**1.59**
DenseNet-BC ($k = 12$)	100	0.8M	**5.92**	4.51	**24.15**	22.27	1.76
DenseNet-BC ($k = 24$)	250	15.3M	**5.19**	**3.62**	**19.64**	**17.60**	1.74
DenseNet-BC ($k = 40$)	190	25.6M	-	**3.46**	-	**17.18**	-

Table 2: Error rates (%) on CIFAR and SVHN datasets. k denotes network's growth rate. Results that surpass all competing methods are **bold** and the overall best results are **blue**. "+" indicates standard data augmentation (translation and/or mirroring). $*$ indicates results run by ourselves. All the results of DenseNets without data augmentation (C10, C100, SVHN) are obtained using Dropout. DenseNets achieve lower error rates while using fewer parameters than ResNet. Without data augmentation, DenseNet performs better by a large margin.

Discussion

구조는 ResNet과 유사하지만, 작은 차이가 큰 차이를 만듬.

<Model Compactness>
DenseNet이 ResNet에 비해 Parameter대비 더 좋은 성능을 냈다.



Conclusion

성능 저하나 overfitting없이 parameter의 개수가 증가할수록 정확도가 향상되는 모습을 보였다.

다른 구조들에 비해 적은 parameter개수와 연산량으로 더 좋은 결과를 냈다.

DenseNet은 모든 Layer들을 연결한다는 간단한 connectivity rule을 따르면서도,
Identity mapping, deep supervision, diversified depth등의 특징을 모두 실현시켜
Feature reuse를 더 용이하고, 모델을 더 compact하게 만들었다.