Paper Review

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

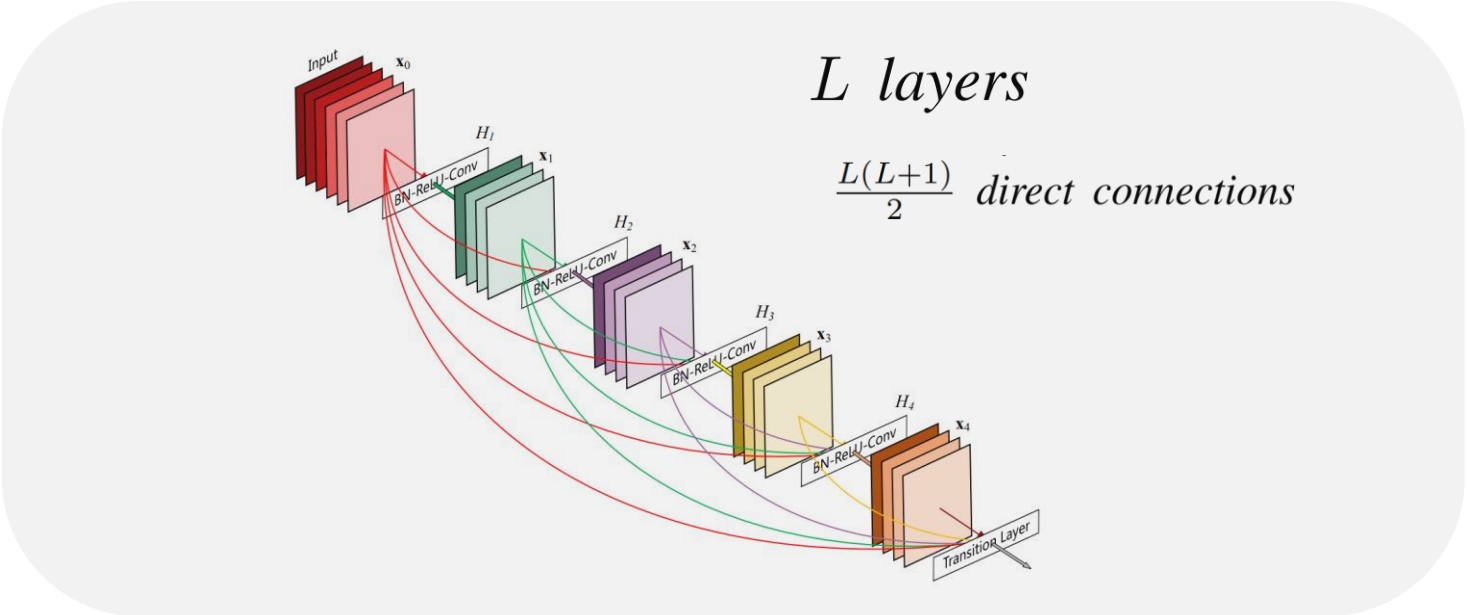
이준영
UST-KRISS 측정과학전공

2021. 05. 07

1 Introduction

많은 연구를 통해 CNN은 입력과 출력 사이의 연결 거리가 짧아질 수록 효율적인 훈련과 높은 정확도를 높일 수 있다는 것을 발견했음.

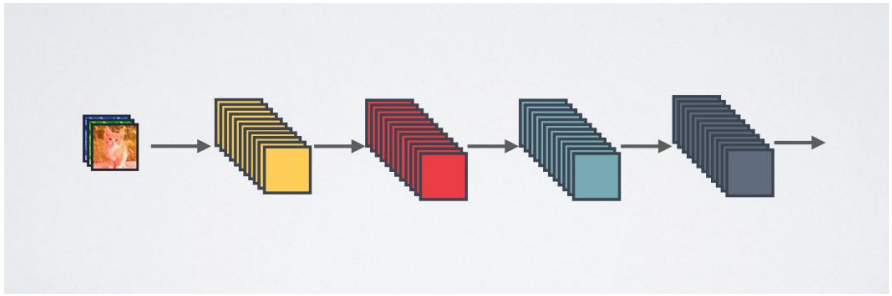
본 논문에서는 이러한 발견에 기반하여 **모든 레이어 사이를 연결해줌으로서** 더 강화된 신경망을 소개함.



DenseNet은 기울기소실을 완화하고 파라미터 수를 효과적으로 줄인다.

2 DenseNet

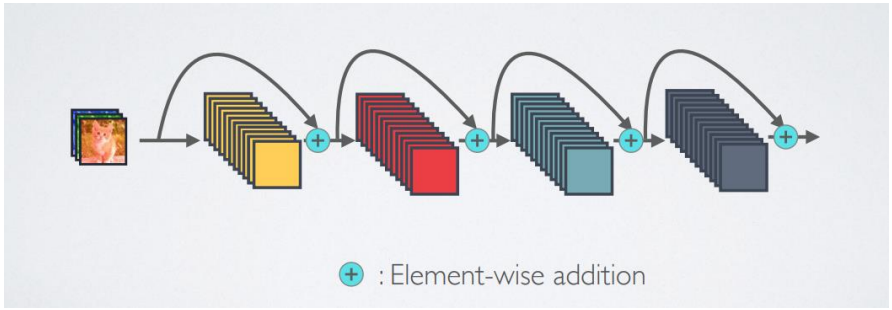
STANDARD



기존의 feed-forward 신경망은 l 번째 layer 의 output 을 다음층의 input과 연결

→ $X_l = H_l(X_{l-1})$

RESNET

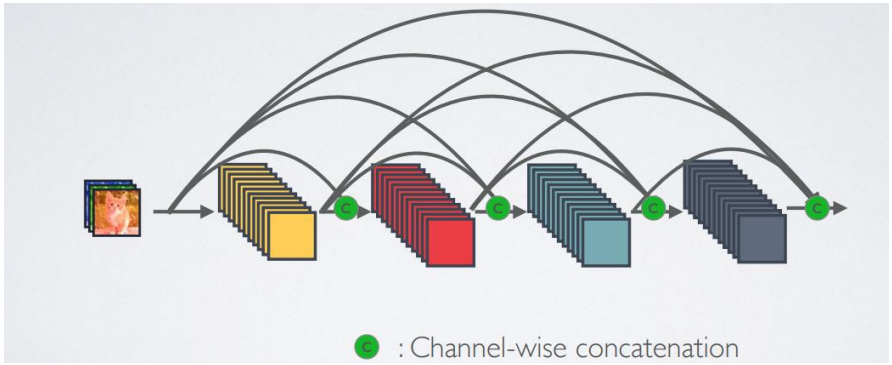


ResNet 은 skip-connection 을 추가

→ $X_l = H_l(X_{l-1}) + X_{l-1}$

기울기 소실 문제를 일부 해결할 수 있으나, 합으로 결합되기 때문에 **정보전달이 방해를 받을 수 있음.**

DENSENET



정보전달을 향상시키기 위해 모든 레이어 사이를 연결함.

→ $X_l = H_l(X_0, X_1, \cdots, X_{l-1})$

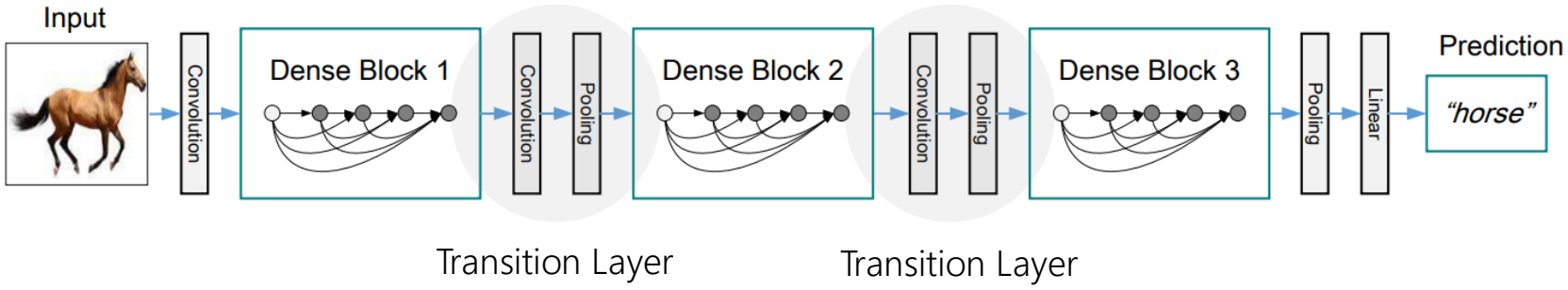
Composition function : Composite function은 모든 층의 feature map을 입력으로 받아 **연결**(concatenation)하여 다음의 세가지 연산을 수행한다.

H_l : composite function

Batch Normalization
+
ReLU
+
Convolution (3×3)

Pooling layer (Transition Layer)

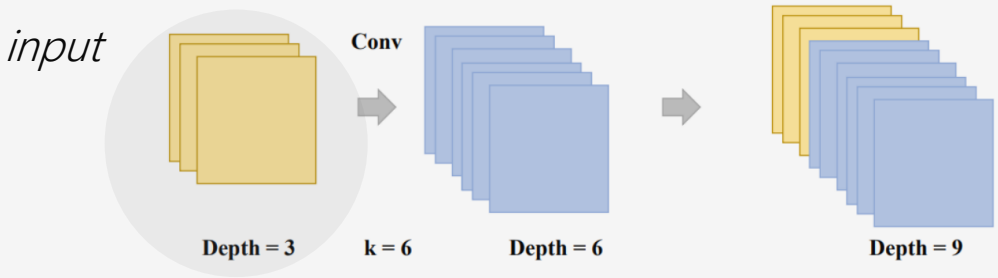
Concatenation 특성상 레이서 사이의 down sampling이 불가능하기 때문에 DenseNet 순전파를 하나의 Dense Block으로 구분하여 block 사이에 1×1 convolution과 2×2 average pooling을 수행한다.

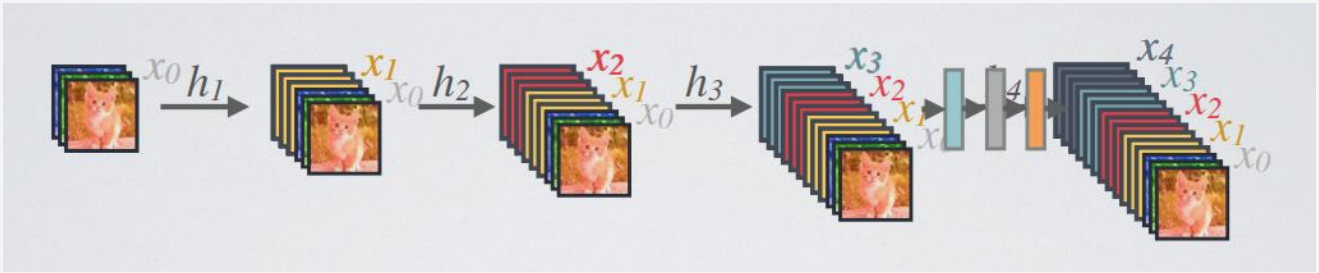


Growth rate (k)

- H_l이 k개의 feature-map을 생성하면, 다음 layer는 $k_0 + k \times (l-1)$개의 input feature-map을 가짐
  (k_0 는 dense block 첫번째의 input layer 의 feature map 개수)

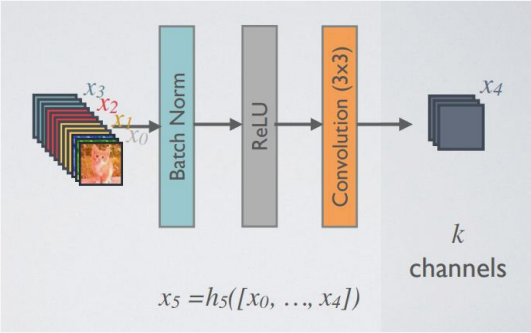
- 한번 기록되면 network 내의 어디서든 접근 가능, 계층별로 layer 복제할 필요 없음



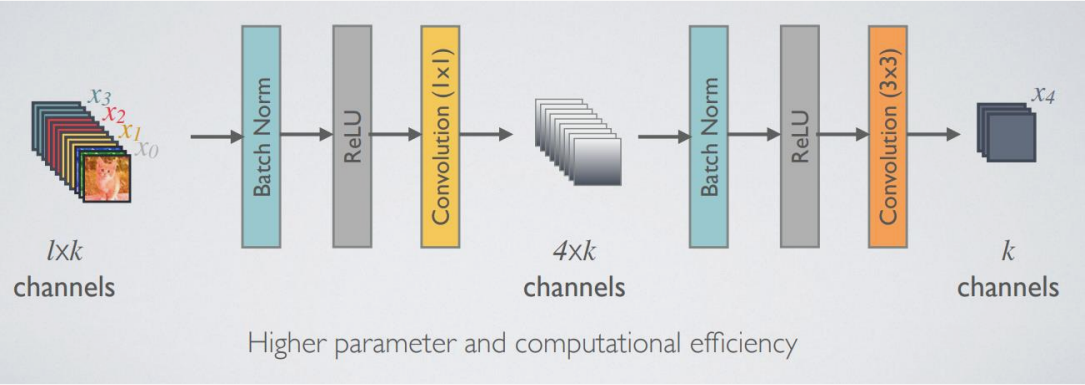


Bottleneck layer

- 층이 진행 될수록 input의 채널이 쌓이기 때문에 feature-map을 생성하기 위한 연산양 또한 증가함.



- 증가한 연산양을 효율적으로 처리하기 위해 bottleneck layer를 이용하여 $4\times k$ 데이터로 변환 후 composition function을 수행함.



Compression

- 모델의 압축률을 더 향상 시키기 위해 transition layer에서 feature-map의 수를 줄임.
- 하나의 Dense block의 output이 m개의 feature-map을 포함한다면, transition layer에서 θ×m개의 feature-map을 갖는 output 생성
- DenseNet-C : θ < 1 인 DenseNet (θ = 0.5)
- DenseNet-BC : θ < 1 인 bottleneck과 transition layer를 모두 사용

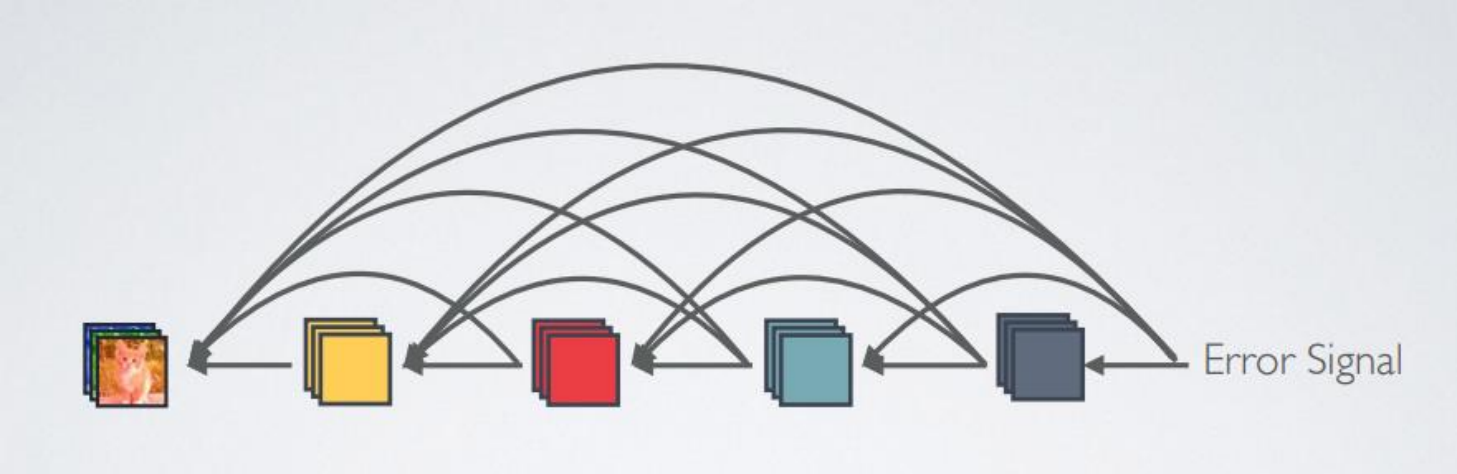
Architecture

Layers	Output Size	DenseNet-121($k = 32$)	DenseNet-169($k = 32$)	DenseNet-201($k = 32$)	DenseNet-161($k = 48$)
Convolution	112×112	7×7 conv, stride 2			
Pooling	56×56	3×3 max pool, stride 2			
Dense Block (1)	56×56	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$
Transition Layer (1)	56×56	1×1 conv			
	28×28	2×2 average pool, stride 2			
Dense Block (2)	28×28	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$
Transition Layer (2)	28×28	1×1 conv			
	14×14	2×2 average pool, stride 2			
Dense Block (3)	14×14	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 24$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 48$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 36$
Transition Layer (3)	14×14	1×1 conv			
	7×7	2×2 average pool, stride 2			
Dense Block (4)	7×7	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 16$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 24$
Classification Layer	1×1	7×7 global average pool			
		1000D fully-connected, softmax			

3 Advantage

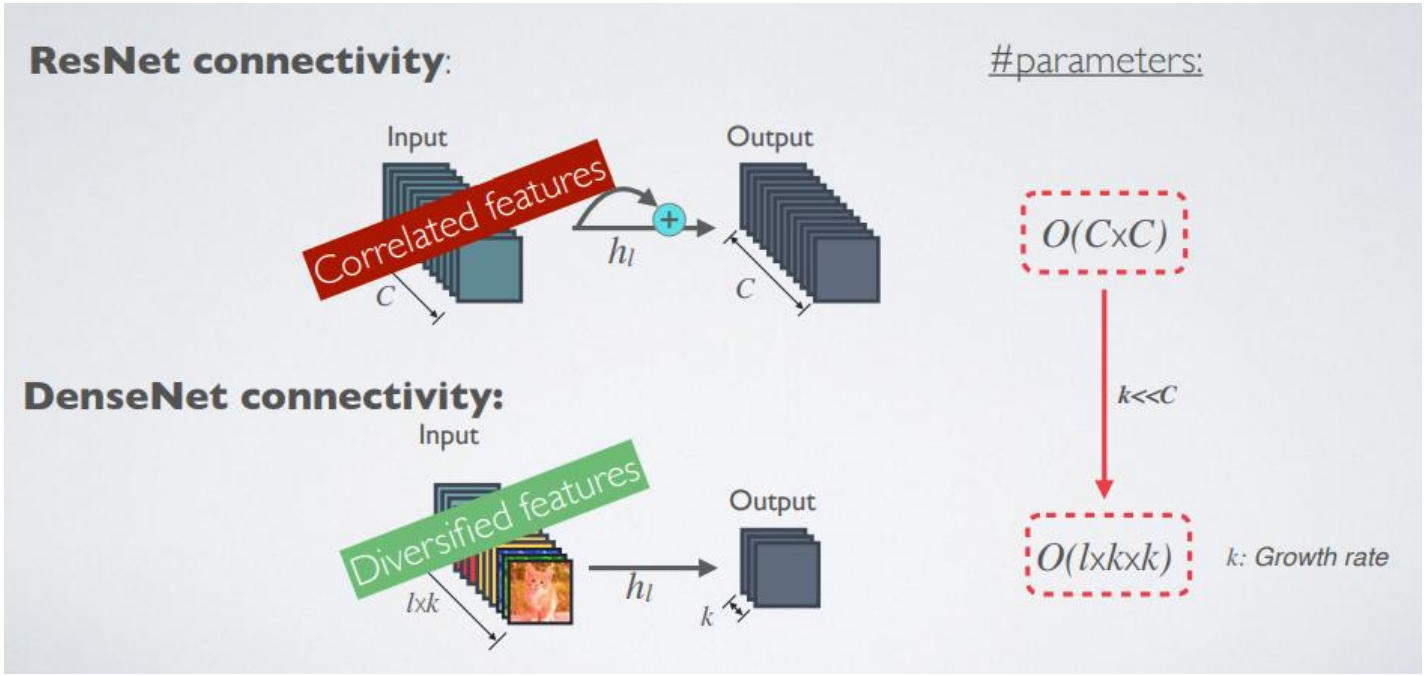
3.1 Strong gradient flow

- 모든 층이 연결되어 있기 때문에 역전파시에도 기울기 소실 문제가 줄어들음.



3.2 Maintains low complexity features

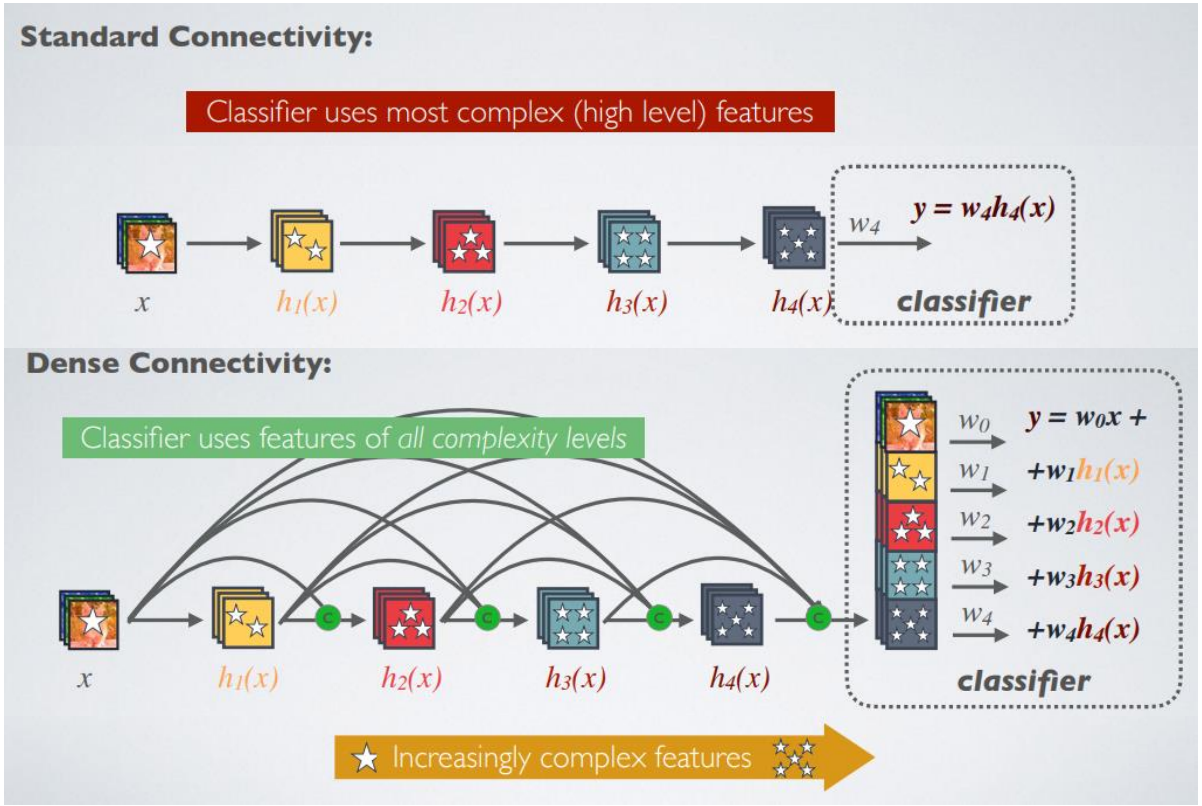
- 기존 모델은 채널 수가 많아야 학습이 잘되기 때문에 좋은 성능을 내기 위해서 복잡도가 증가함.
- 그러나 DenseNet은 기존의 채널(C)보다 굉장히 작은 k를 파라미터로 하기 때문에 복잡도가 작음



3.3 Parameter & Computational Efficiency

- 기존의 경우 초반 layer의 정보가 깊은 망에서 뒤로 갈수록 희미 해지는데,
  Dense connectivity pattern을 사용하면 초반 layer 의 정보를 쌓아가며 마지막까지 전달됨.

- 모든 feature map 들을 쌓기 때문에 layer 사이 사이 최대한 가치 있는 정보가 전달됨.



4 Experiment

CIFAR-10, CIFAR-100 (32*32*3, train 5만, test 1만) SVHN (32*32*3 train 73257 test 26032)

ImageNet (224*224, train 120만, test 5만)

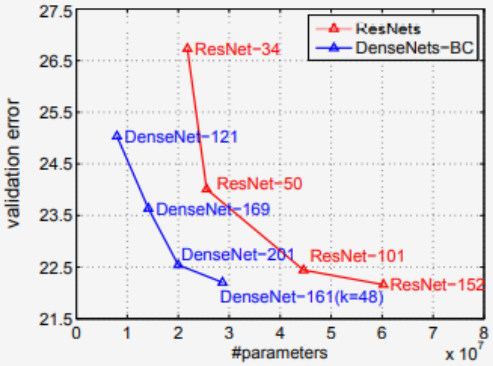
Method	Depth	Params	C10	C10+	C100	C100+	SVHN
Network in Network [22]	-	-	10.41	8.81	35.68	-	2.35
All-CNN [31]	-	-	9.08	7.25	-	33.71	-
Deeply Supervised Net [20]	-	-	9.69	7.97	-	34.57	1.92
Highway Network [33]	-	-	-	7.72	-	32.39	-
FractalNet [17]	21	38.6M	10.18	5.22	35.34	23.30	2.01
with Dropout/Drop-path	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 [41]	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**	-

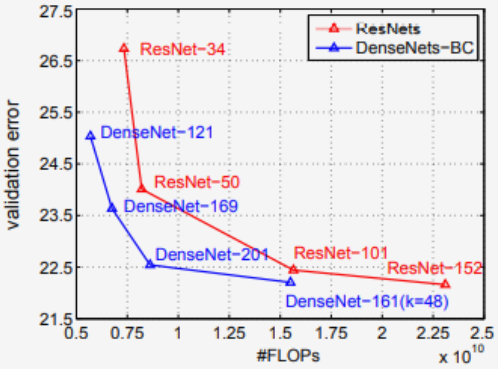
1. K 가 증가하면 parameter 의 수가 많아지고
Bottleneck Compression 을 사용하면 parameter 가 감소

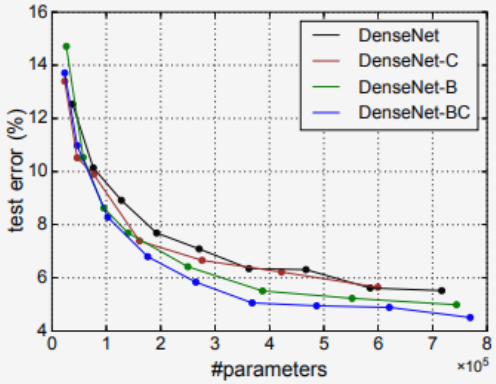
2. SVHN 이 상대적으로 쉬운 작업이기 때문에
더 deep 한 모델이 학습데이터에 overfitting 되어서
BC를 안쓰는 network에서 결과가 좋음

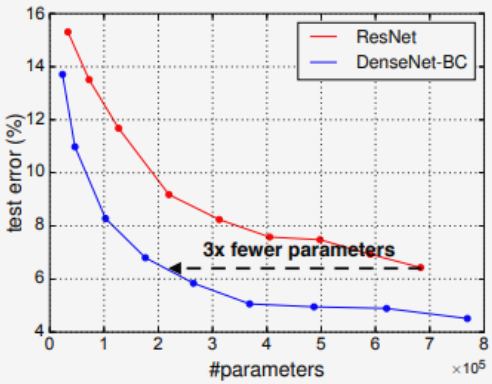
4 Experiment

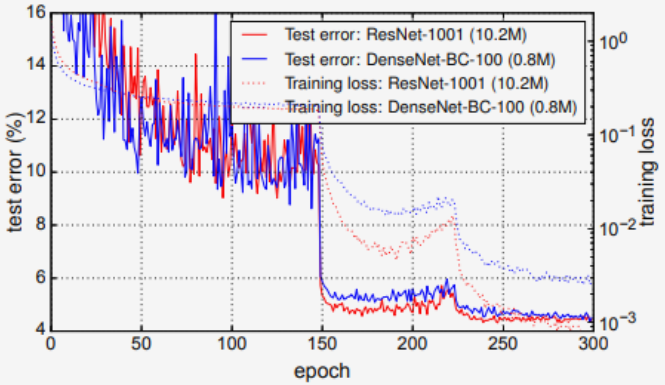
Model	top-1	top-5
DenseNet-121 ($k=32$)	25.02 (23.61)	7.71 (6.66)
DenseNet-169 ($k=32$)	23.80 (22.08)	6.85 (5.92)
DenseNet-201 ($k=32$)	22.58 (21.46)	6.34 (5.54)
DenseNet-161 ($k=48$)	22.33 (20.85)	6.15 (5.30)





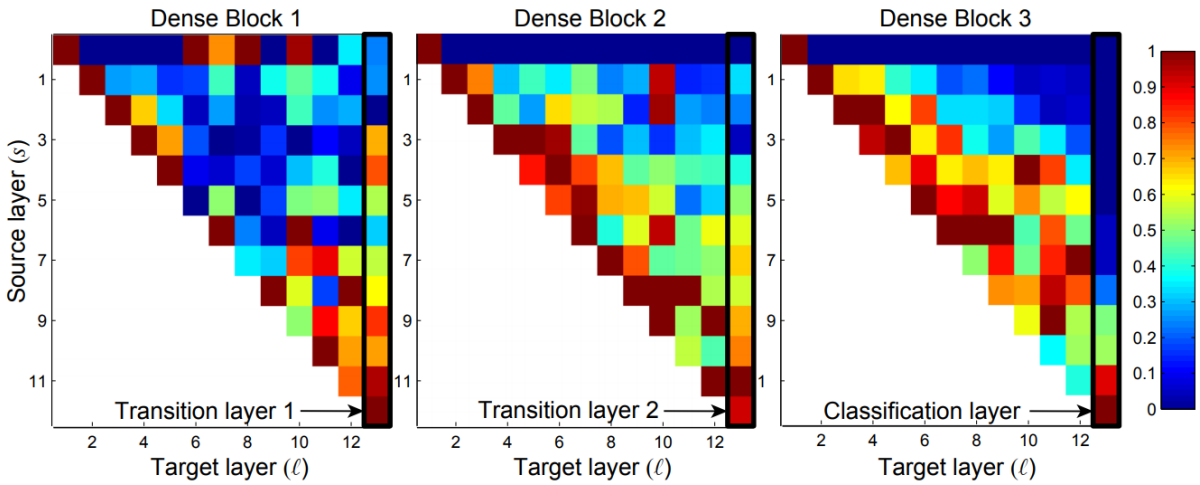






5 Discussion

- Dense block 내부에서 convolution layer 들의 필터 가중치의 평균
- 픽셀 색깔이 source layer(s) 에서 target layer(l) 로 연결되어 있는 가중치의 L1norm
- 색이 빨간색일수록 더 높은 가중치



- 같은 block 내에서는 가중치가 잘 흩어져 있음
- Transition layer 에서 나온 가중치도 잘 퍼져 있음
- Classification layer 가 전체 weight 를 가져가기는 하지만,
  네트워크 마지막 단에서 생긴 high-level feature 를 더 많이 가짐

Q&A

Reference

- https://jayhey.github.io/deep%20learning/2017/10/13/DenseNet_1/
- https://youtu.be/fe2Vn0mwALI
- https://www.youtube.com/watch?v=fe2Vn0mwALI&t=922s