DenseNet

Densely Connected Convolutional Networks

2021.03.30

Overview

2017 CVPR 컨퍼런스
>> Densely Connected Network

[Advantage]

1. Vanishing Gradient 문제 완화
2. 더 강력한 feature propagation
3. Feature 재사용 촉진
4. 파라미터 수 감소
5. Regularlizing 효과와
   Overfitting 감소

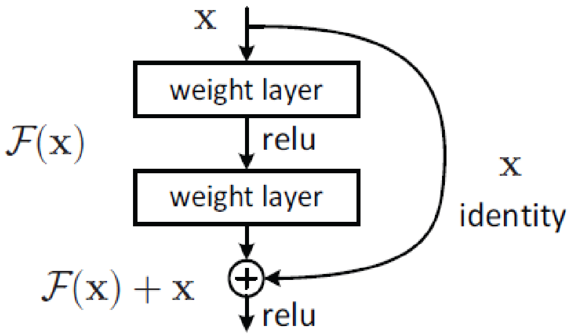
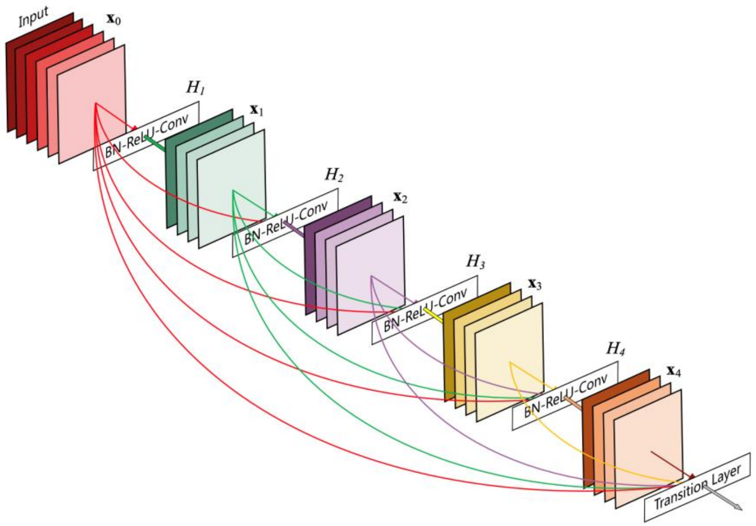
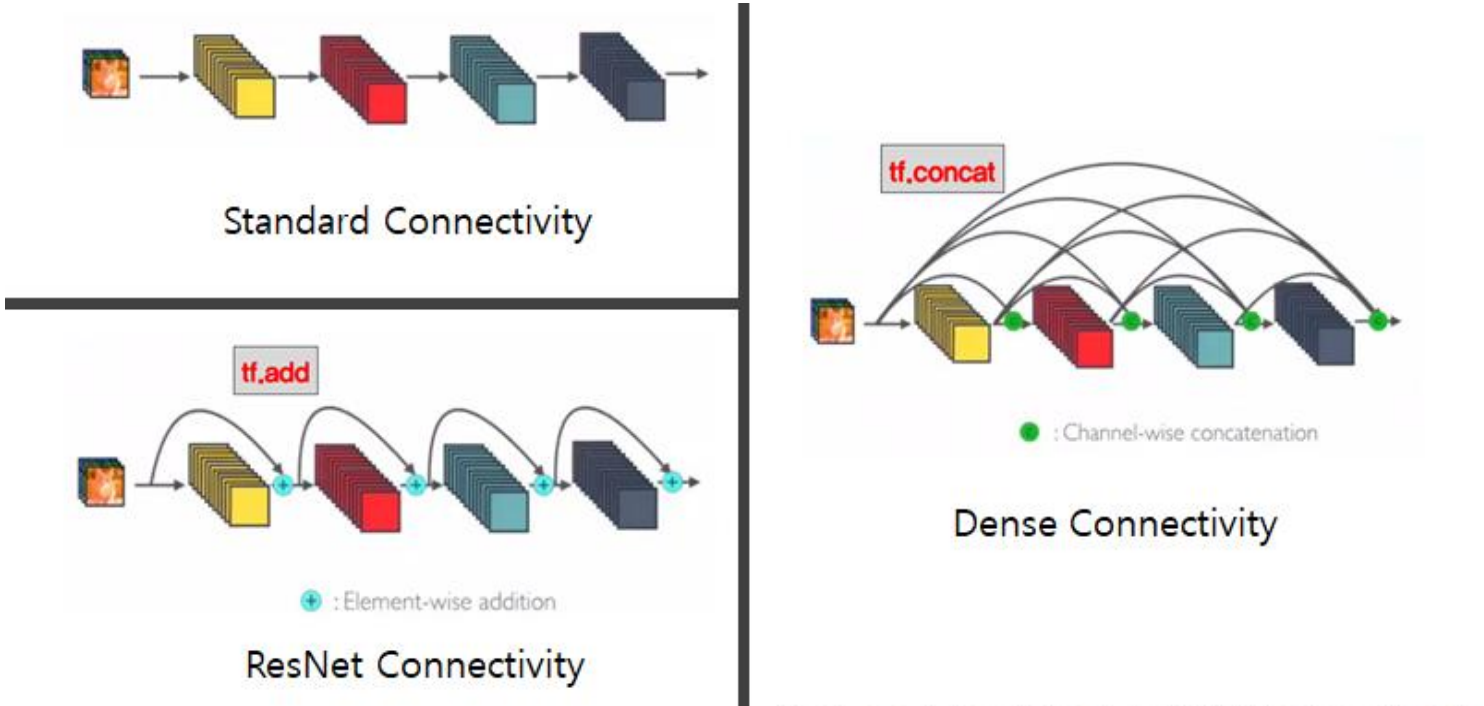


Figure 2. Residual learning: a building block





Densely connected convolution networks CVPR 2017 oral presentation slide

Dense Connectivity

\# 전통 CNN

$$x_l = H_l(x_{l-1})$$

\# ResNet

$$x_l = H_l(x_{l-1}) + x_{l-1}$$

\# DenseNet

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

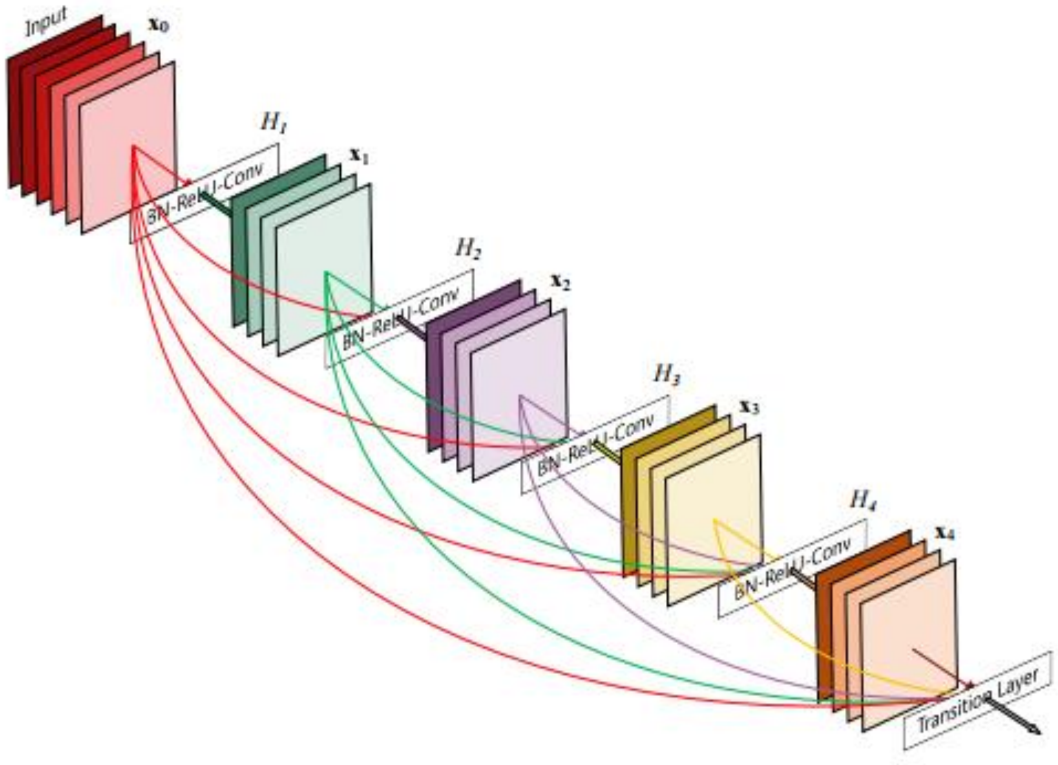
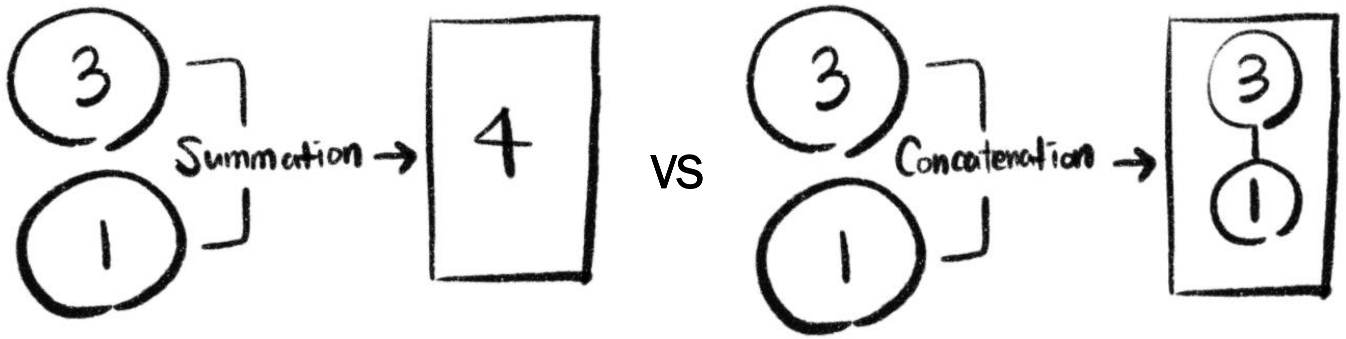
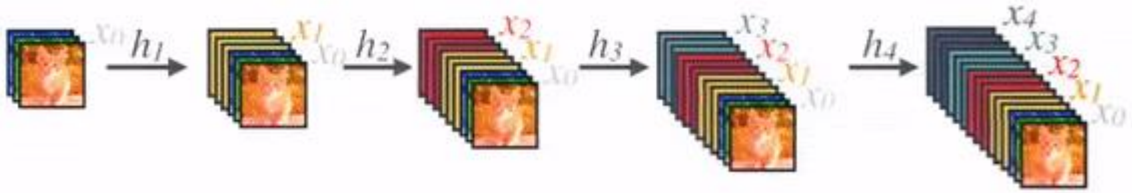


Figure 1: A 5-layer dense block with a growth rate of $k = 4$. Each layer takes all preceding feature-maps as input.

Summation vs Concatenation



Dense Connectivity



Composite Function

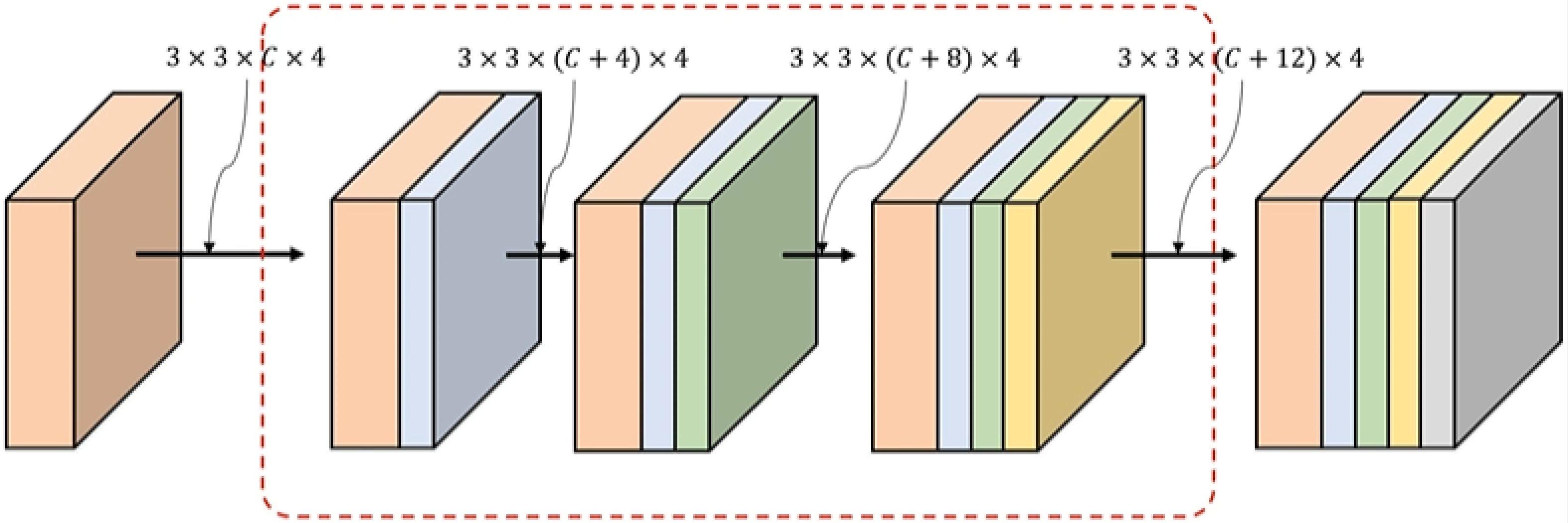
Composite function. Motivated by [12], we define $H_\ell(\cdot)$ as a composite function of three consecutive operations: batch normalization (BN) [14], followed by a rectified linear unit (ReLU) [6] and a 3×3 convolution (Conv).

Three consecutive operations

$$\boxed{H_\ell(\cdot)}$$

1. **Batch normalization (BN)**
2. **Rectified linear unit (ReLU)**
3. **3*3 convolution**

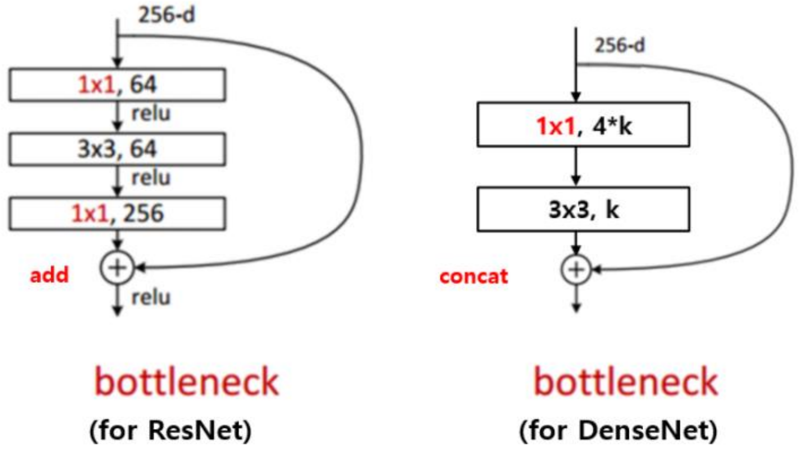
Growth Rate

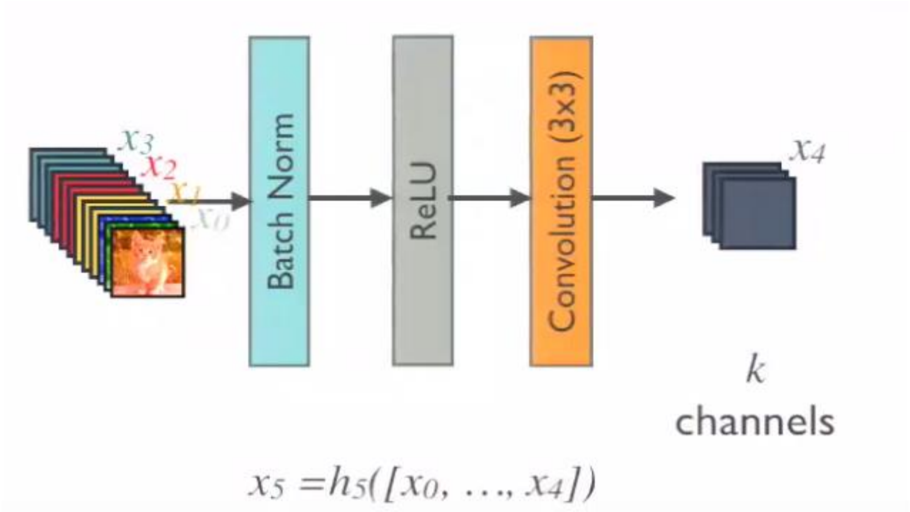


Dense Block with Growth Rate $k = 4$

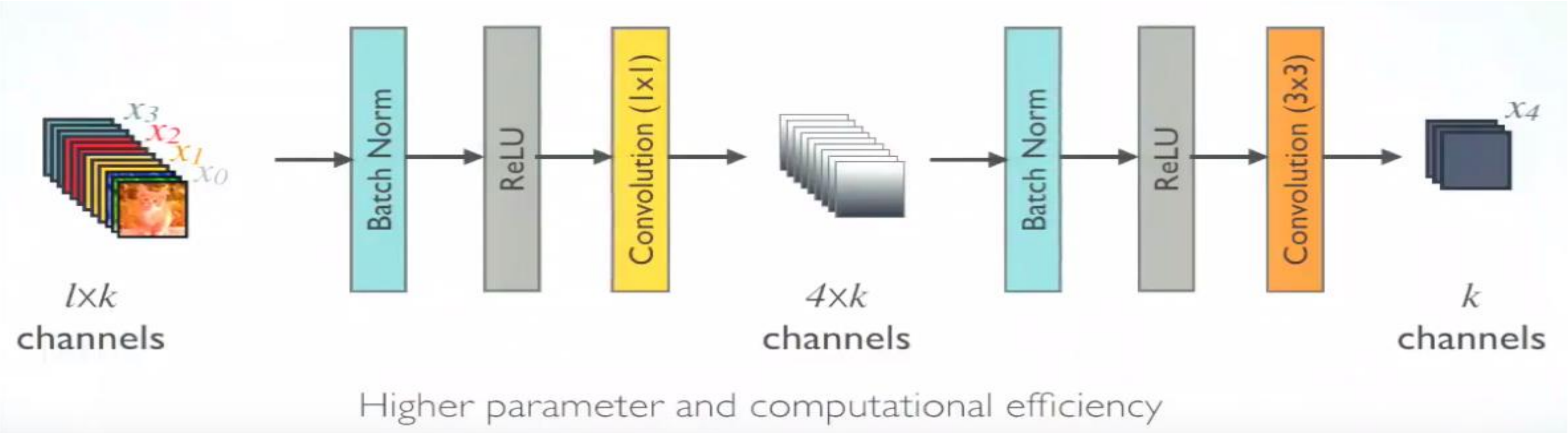
One explanation for this is that each layer has access to all the preceding feature-maps in its block and, therefore, to the network's "collective knowledge".
One can view the **feature-maps as the global state of the network.**
Each layer adds k feature-maps of its own to this state.
The growth rate regulates how much new information each layer contributes to the global state.
The global state, once written, can be accessed from everywhere within the network and,
unlike in traditional network architectures, there is no need to replicate it from layer to layer.

Bottleneck Layer





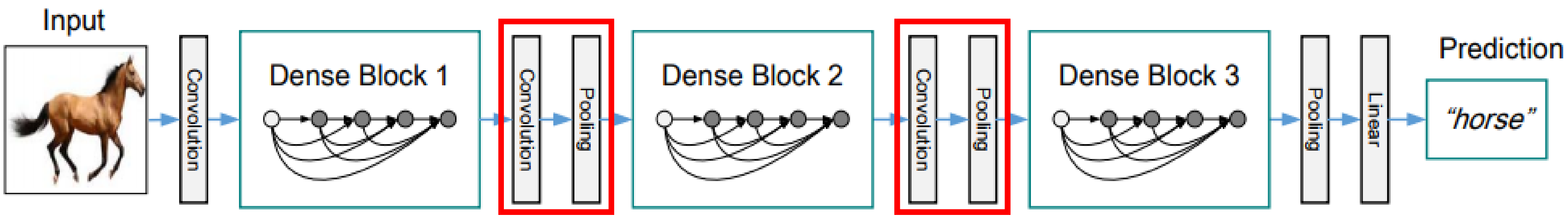
Densely connected convolution networks CVPR 2017 oral presentation slide



Densely connected convolution networks CVPR 2017 oral presentation slide

Compression (Transition Layer)

>> Transition Layer = Convolution Layer + Pooling Layer



Huang, Gao, et al. "Densely connected convolutional networks." *arXiv preprint arXiv:1608.06993* (2016).

DenseNet 실험 – 모델 구조

	CIFAR	SVHN	ImageNet
Optimization Method	SGD	SGD	SGD
Batch Size	64	64	256
Epoch	300	40	90
Initial Learning Rate	0.1	0.1	0.1
Initalization Method	He	He	He

>> DenseNet ImageNet 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			

Table 1: DenseNet architectures for ImageNet. The growth rate for the first 3 networks is $k = 32$, and $k = 48$ for DenseNet-161. Note that each "conv" layer shown in the table corresponds the sequence BN-ReLU-Conv.

Huang, Gao, et al. "Densely connected convolutional networks." *arXiv preprint arXiv:1608.06993* (2016).

DenseNet 실험 – 모델 구조

	CIFAR	SVHN	ImageNet
Optimization Method	SGD	SGD	SGD
Batch Size	64	64	256
Epoch	300	40	90
Initial Learning Rate	0.1	0.1	0.1
Initalization Method	He	He	He

>> DenseNet CIFAR, SVHN architecture

Layers	Output Size	DenseNet (k=12, L=40)		DenseNet (k=12, L=100)		DenseNet (k=24, L=100)		DenseNet-BC (k=12, L=100)		DenseNet-BC (k=24, L=250)		DenseNet-BC (k=40, L=190)	
Convolution	32x32	3x3 conv											
Dense Block (1)	32x32	3x3 conv	x12	3x3 conv	x32	3x3 conv	x32	1x1 conv 3x3 conv	x 16	1x1 conv 3x3 conv	x 41	1x1 conv 3x3 conv	x 31
Transition Layer (1)	32x32	1x1 conv											
	16x16	2x2 average pool, stride=2											
Dense Block (2)	16x16	3x3 conv	x12	3x3 conv	x32	3x3 conv	x32	1x1 conv 3x3 conv	x 16	1x1 conv 3x3 conv	x 41	1x1 conv 3x3 conv	x 31
Transition Layer (2)	16x16	1x1 conv											
	8x8	2x2 average pool, stride=2											
Dense Block (3)	8x8	3x3 conv	x12	3x3 conv	x32	3x3 conv	x32	1x1 conv 3x3 conv	x 16	1x1 conv 3x3 conv	x 41	1x1 conv 3x3 conv	x 31
Classification Layer	1x1	8x8 global average pool											
		10D fully-connected, softmax											

DenseNet 실험 – 실험 결과

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**	-

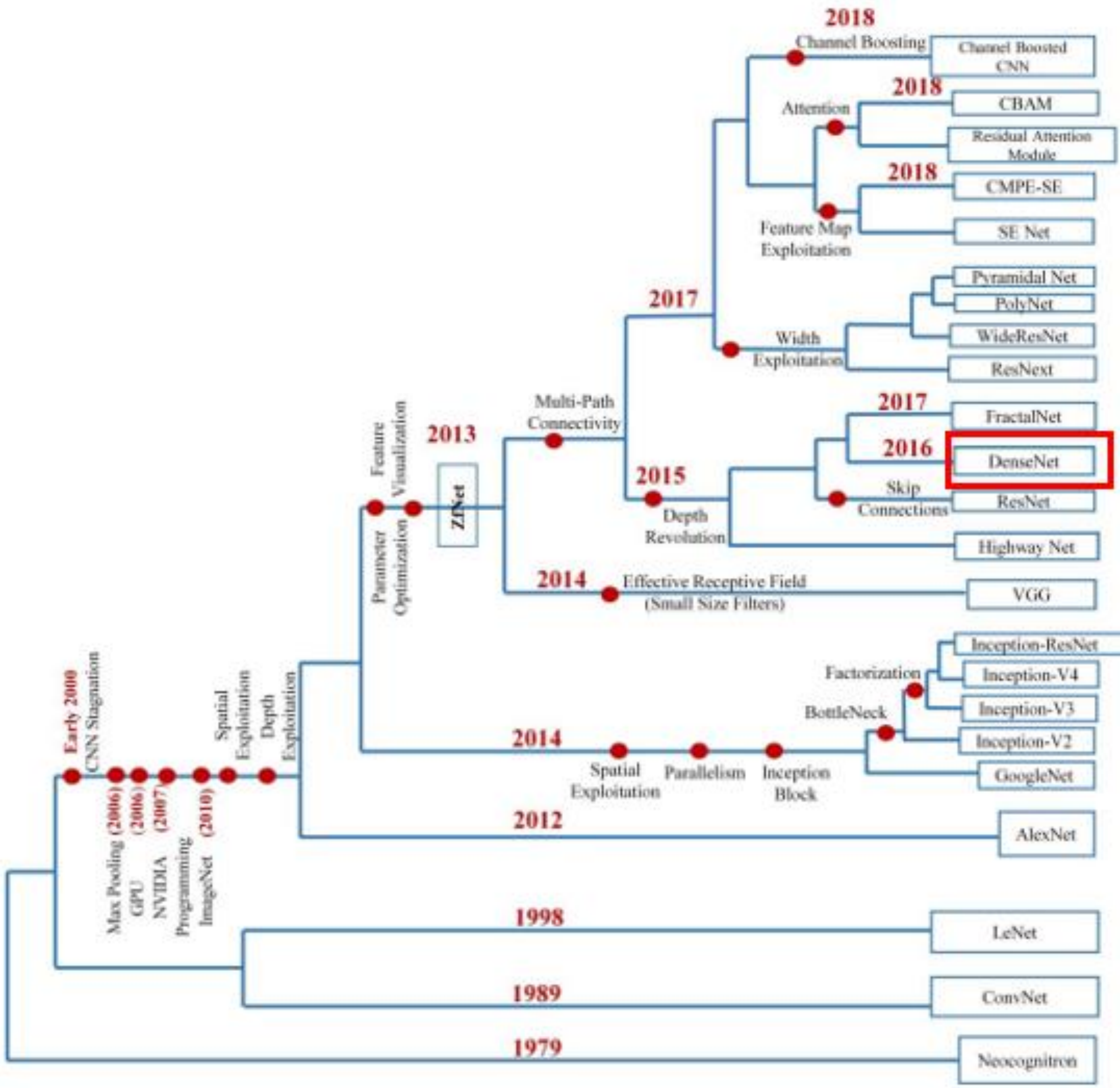
Table 2: Error rates (%) on CIFAR and SVHN datasets. k denotes network's growth rate. Results that surpass all competing methods are **bold** and the overall best results are **blue**. "+" indicates standard data augmentation (translation and/or mirroring). $*$ indicates results run by ourselves. All 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.

DenseNet 실험 – 실험 결과

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**	-

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.

Thank You
☺☺☺



2018
Channel Boosting
Channel Boosted CNN
2018
Attention
CBAM
Residual Attention Module
2018
CMPE-SE
Feature Map Exploitation
SE Net
2017
Pyramidal Net
PolyNet
Width Exploitation
WideResNet
ResNext
2017
FractalNet
2016
DenseNet
Multi-Path Connectivity
2015
Skip Connections
ResNet
Depth Revolution
Highway Net
2013
Feature Visualization
ZfNet
Parameter Optimization
2014
Effective Receptive Field (Small Size Filters)
VGG
Inception-ResNet
Factorization
Inception-V4
Inception-V3
BottleNeck
Inception-V2
2014
Spatial Exploitation
Parallelism
Inception Block
GoogleNet
Early 2000
CNN Stagnation
Max Pooling (2006)
GPU (2006)
NVIDIA Programming (2007)
ImageNet (2010)
Spatial Exploitation
Depth Exploitation
2012
AlexNet
1998
LeNet
1989
ConvNet
1979
Neocognitron