

#3. Densely Connected convolutional Networks

1. Abstract

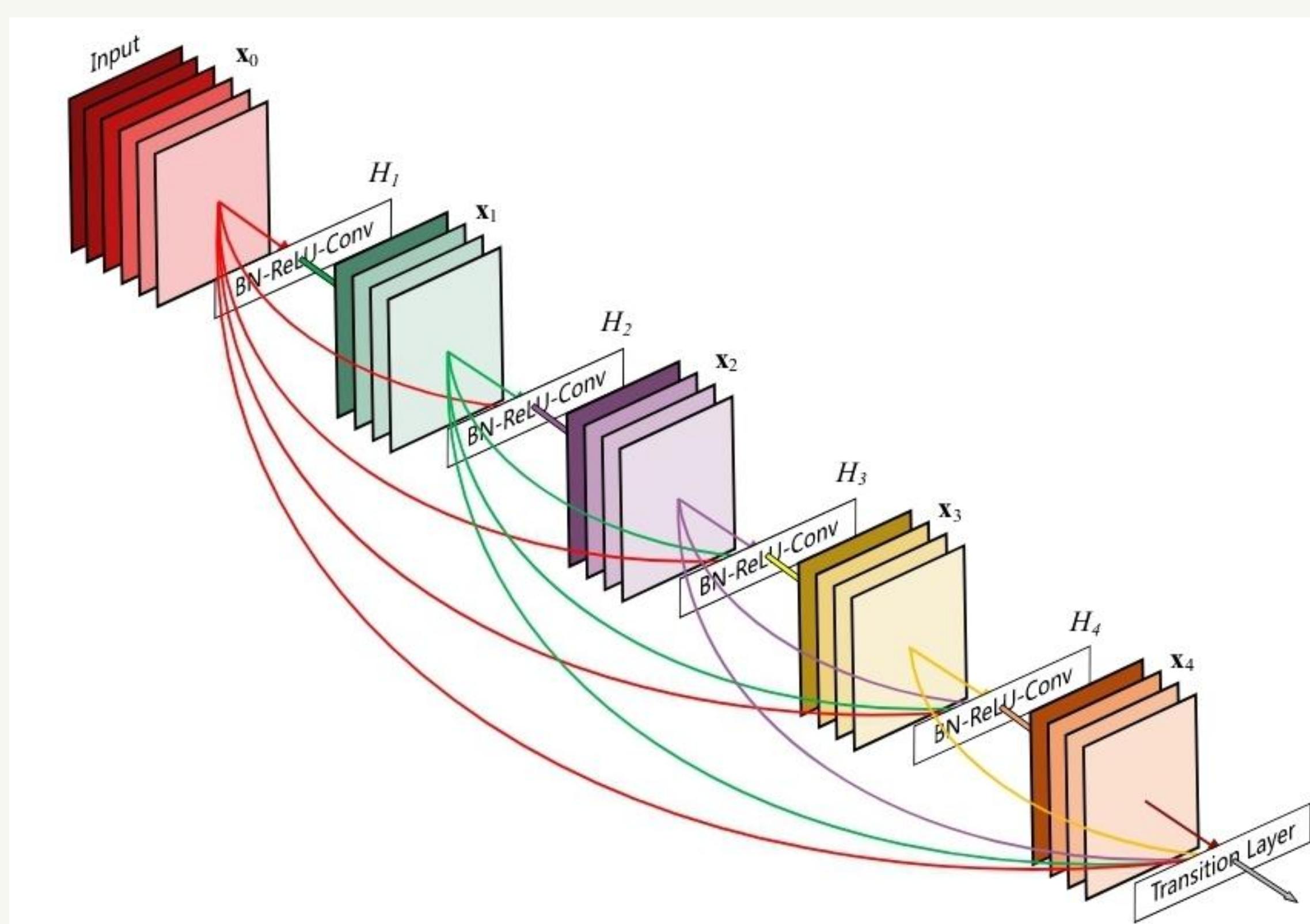
- feed forward 방식으로 레이어를 연결한 DenseNet 제안
- 이전 레이어의 모든 feature map이 다음 레이어로 입력됨
- DenseNet 장점
 - ① gradient vanishing ↓
 - ② strengthen feature propagation
 - ③ encourage feature reuse
 - ④ Reduce # of parameters

2. Introduction

- ① CNN이 깊어지면서 feature info & gradient가 vanishing
→ ResNet, Highway Networks ...

공통점 : early layer에서 later layer로 가는 short path 사용

- ② 본 논문에서는 기존의 short path pattern을 distill하고 네트워크의 레이어 사이에서 information flow가 원활히 이루어 어지도록 각 레이어를 모두 연결한 **connectivity pattern**이용
- ③ 각 레이어는 이전의 모든 레이어에서 나온 feature map을 입력으로 받으며, 자신의 output feature map을 이후에 있는 모든 레이어에 전달하는 구조



- ④ ResNet은 feature map끼리 summation
DenseNet은 feature concat
- ⑤ L개의 레이어 $\rightarrow \frac{L(L+1)}{2}$ 개의 connection \exists
- ⑥ Dense connectivity pattern은 traditional CNN보다
파라미터의 수가 ↓, 중복된 feature map 학습할 필요 X
- ⑦ DenseNet의 구조는 information flow를 개선하여 네트워크에서 gradient가 사라지는 문제를 방지하여 학습이 쉬워짐

3. Related Work

- ① Highway Network는 bypassing path를 통해 100개 이상의 레이어를 사용하는 deep model에서도 학습이 험고적
- ② ResNet은 1202개까지 layer를 늘렸을 때 성능 ↓
: Stochastic depth에서는 ResNet-1202를 학습하면서 랜덤하게 특정 layer를 dropping 시켜 성공적으로 학습함.
 \therefore 1202개 레이어에서 모든 레이어가 반드시 필요한 것이 아님
이렇게 많은 레이어를 사용하는 것은 redundancy가 커지는 문제야!
- ③ DesNet은 네트워크 내에서 feature reuse를 통해 condensed 된 모델을 만들어 파라미터의 수를 줄이고 쉽게 학습

4. DenseNet

→ 0~ $\ell-1$ 번째 블록의 feature map을

① $\underline{x}_\ell = H_\ell (\underline{[x_0, x_1, \dots, x_{\ell-1}]})$ concat한 feature map

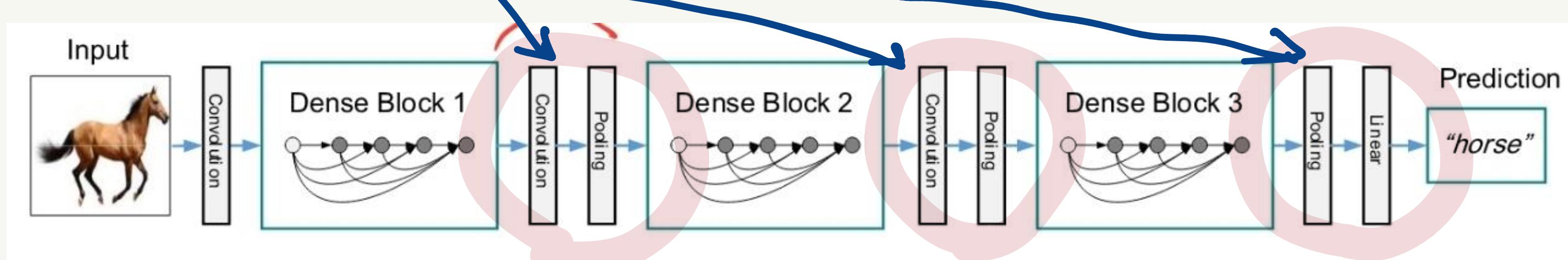
output of ℓ^{th} layer

② composite function (복합 함수) : H_ℓ

H_ℓ 은 BN, ReLU, 3×3 conv 순으로 사용

③ Pooling layers

- concatenation operation은 feature map의 width, height dimension이 일치해야 함
- DenseNet의 dense connectivity는 dense block 사이에서 이루어지며 spatial size를 down-sampling 하기 위해 dense block 사이에 transition layer를 추가함
- transition layer: BN, 1×1 conv, 2×2 avg pooling



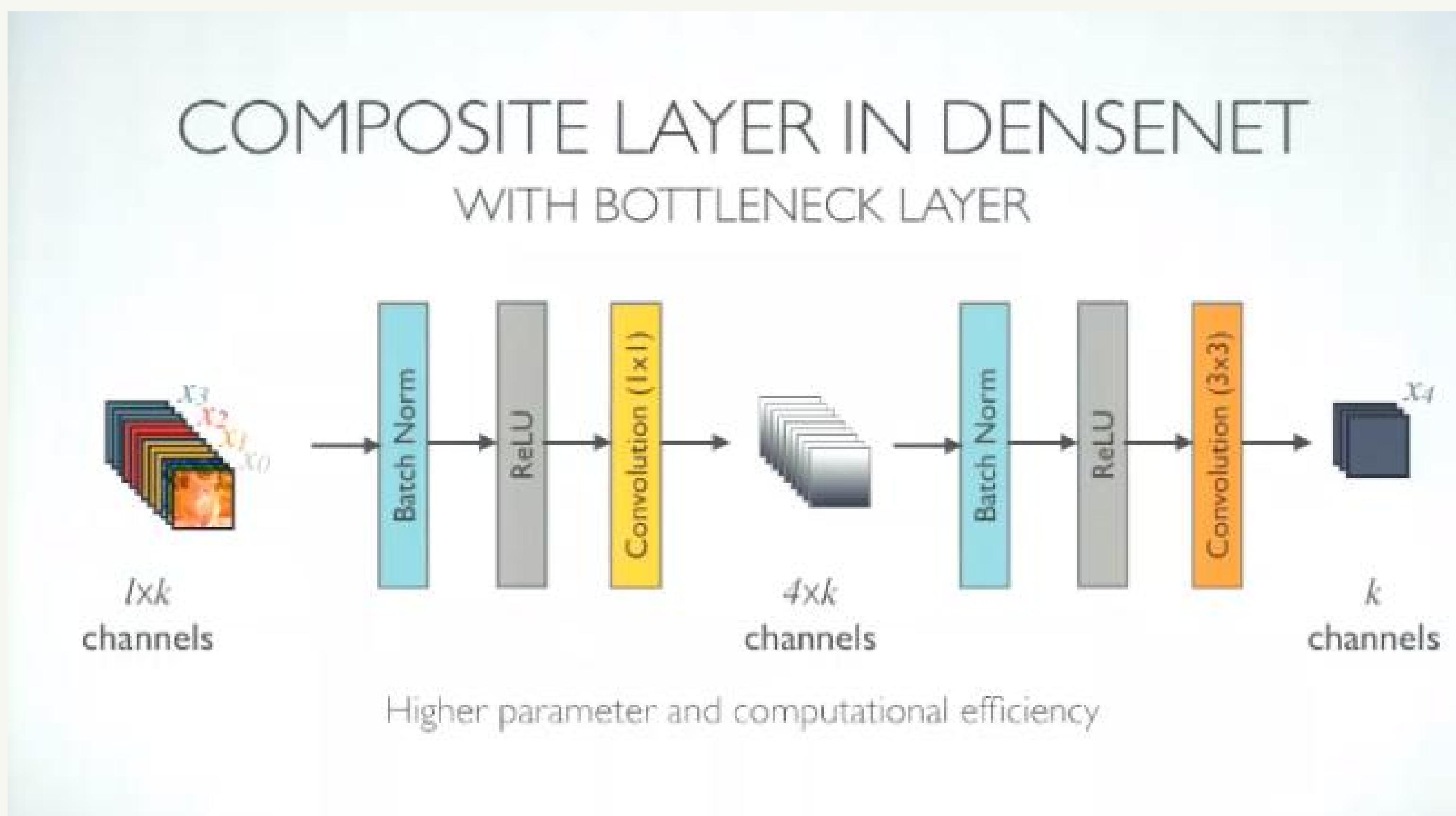
④ Growth rate

- Input layer의 feature map을 제외하고 convolution layer에서 만들어지는 feature map의 채널 수를 조절하기 위한 하이퍼파라미터

⑤ Bottleneck layer

- 각 레이어에서 채널수가 k개인 output feature map을 만들어냄
→ k가 작은 값이라도 레이어가 쌓일수록 채널수가 굉장히 많아짐.
∴ 채널수를 조절하기 위한 bottleneck layer 사용

- 1×1 conv에서 채널수가 growth rate * 4인 feature map을 만들고 3×3 conv에서는 growth rate 만큼 채널수를 가지는 feature들을 뽑음.



- bottleneck layer에서 유사한 구조를 DenseNet-B라고 부른. (DenseNet-Bottleneck)

⑥ Compression

- model의 compactness를 improve하기 위해 transition layer에서 feature map size를 조절함.
- Denseblock에서 m개의 feature map이 나오면 transition layer에서는 $\lceil \theta m \rceil$ 개의 output feature map generate.
- $0 < \theta \leq 1$
 $\hookrightarrow \theta = 1$, feature map 개수 변화 X
- $\theta < 1$ 인 버전을 DenseNet-C (compression)
- 실험에서는 $\theta = 0.5$
- DenseNet-B + DenseNet-C \rightarrow DenseNet-BC

⑦ 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	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$
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	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 24$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 48$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 64$
Transition Layer (3)	14×14			1×1 conv	
	7×7			2×2 average pool, stride 2	
Dense Block (4)	7×7	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 16$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 48$
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 the table corresponds the sequence BN-ReLU-Conv.

- Imagenet 제작한 데일리셋에서 dense block이 37%인 DenseNet 사용 & 모든 dense block 안에서 반복되는 레이어의 수는 같음.