

# DenseNet

## 0. Abstract

1. 딥러닝 가까운 층과 출력에 가까운 층의 short conn (resnet)

↓ 핵심

feed forward 방식으로 각 layer를 다른 모든 층과 연결

2. 정의

→ Resnet 이런:  $1 \times L = L$  (각 층마다 들어오는 edge가 모두 있음)

Resnet:  $1 \times L + \text{short conn}$

DenseNet:  $\eta_L \rightarrow \eta_1 : 1 \eta_h$

$\eta_0, \eta_1 \rightarrow \eta_2 : 2 \eta_h$

:

$\eta_0, \dots, \eta_{L-1} \rightarrow \eta_L : L \eta_h$

$$\left. \begin{array}{l} \frac{1+L}{2} \times L \\ = \frac{L(L+1)}{2} \end{array} \right\} \text{"Dense Connectivity"}$$

3. 각 층은  $f \cdot m$ 을 거친다

4. gradient vanishing 문제

feature propagation↑

feature reuse

parameter ↓ (전체수 ↓)

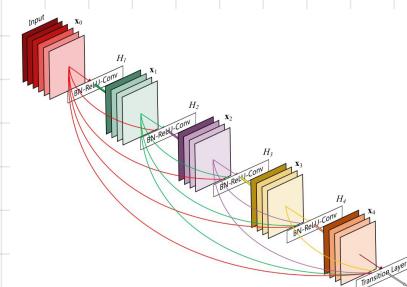


Figure 1: A 5-layer dense block with a growth rate of  $k = 4$ . Each layer takes all preceding feature-maps as input.

## 1. Introduction

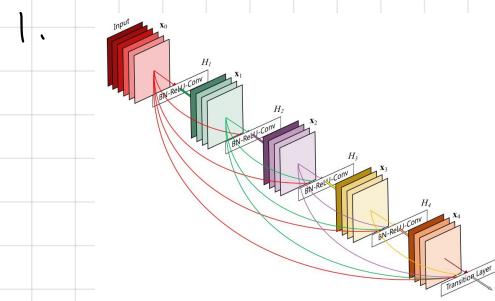


Figure 1: A 5-layer dense block with a growth rate of  $k = 4$ . Each layer takes all preceding feature-maps as input.

→ ResNet와 유사한 특징: "concatenate"

↳ 같은 층은 같은 경로를 따른다 (유선)

→ 같은 층은 같은 경로를 따른다 (유선)

## 2. feed-forward

→ traditional: state 전달 시  $H(x) \approx x$  를 직접 치적화하므로 보존/수정 성능 저하 ↴

→ ResNet: identity mapping,  $\text{FID} \uparrow$

→ DenseNet:  $\text{ID} / \text{FCN}$  성능↑,  $\text{FID} \downarrow$

↳ 각 층  $\text{FID} \downarrow \rightarrow \text{FCN} \text{ performance} \uparrow$

↳ "implicit deep supervision": gradient 시 각 층 출력 gradient 사용 가능 (은은한 辅助 classifier 역할)

"Regularization": dense-conn

## 2. Related work

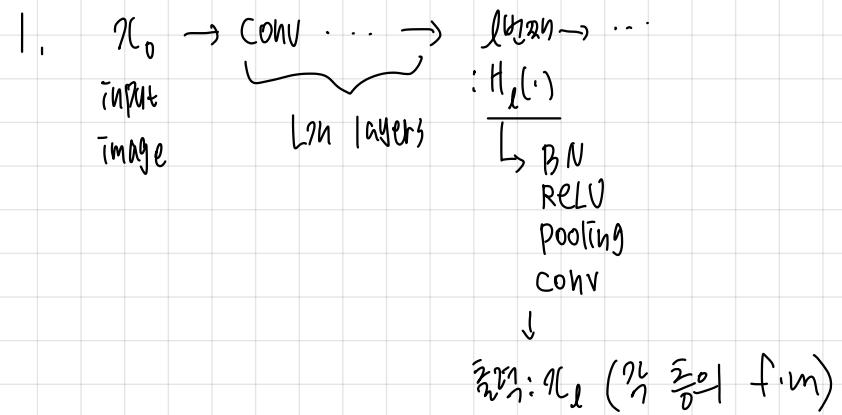
1. ≈ Cascade: layer-by-layer 학습, but 소규모 데이터만

≈ ResNet, Highway Net: Gating Unit, bypass path

### 2. ResNet vs DenseNet

: feature reuse

## 3. DenseNets



↳ traditional:  $x_L = H_L(x_{L-1})$

ResNets :  $x_L = H_L(x_{L-1}) + x_{L-1}$   
↳ 중복 학습 가능

DenseNets :  $x_L = H_L([x_0, x_1, \dots, x_{L-1}])$ ,

↳ Concatenate

## 2. DenseNets

### 1) Dense connectivity

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

Concatenate

### 2) Composite function

$$\rightarrow H_\ell(\cdot) = BN + RELU + 3 \times 3 \text{ Conv}$$

### 3) Pooling layers

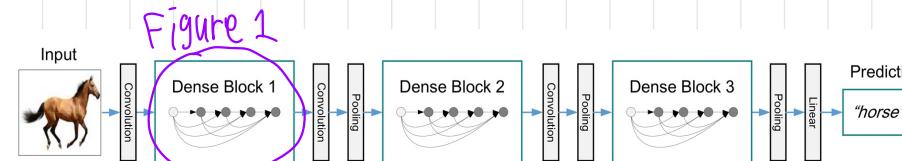


Figure 2: A deep DenseNet with three dense blocks. The layers between two adjacent blocks are referred to as transition layers and change feature-map sizes via convolution and pooling.

↳ feature 3가지 이유로 Concatenate: downsampling

↳ Dense Block + transition layer (전이층)

Conv

Conv + pooling

(= BN + 1x1 conv + 2x2 avg pooling)

### 4) Growth rate

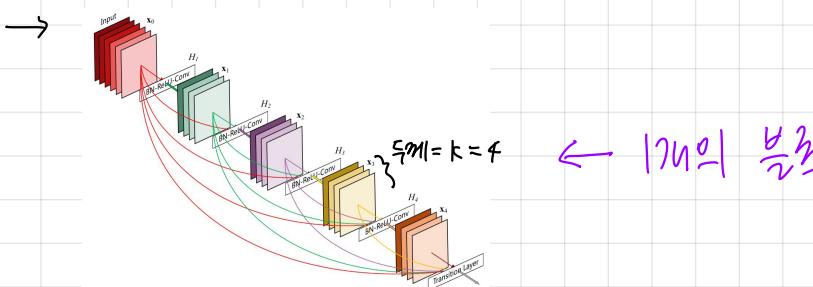


Figure 1: A 5-layer dense block with a growth rate of  $k = 4$ . Each layer takes all preceding feature-maps as input.

$H_\ell$ 의 총 채널 수

$$\rightarrow \text{전체 채널 수} = (l-1) \cdot k + k_0$$

전체 채널 수

$\rightarrow "k"$  (growth rate): 같은 이미지를 (같은 품질을 갖도록) 더 잘 찾기 (collective knowledge, global state)의 수

= 각 층 추가 채널 수

Transition:  $k_0^{\text{next}} = l \cdot C_{\text{out}}$

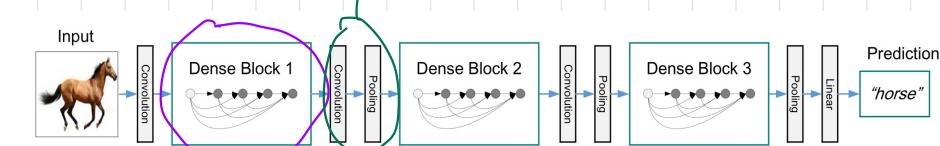


Figure 2: A deep DenseNet with three dense blocks. The layers between two adjacent blocks are referred to as transition layers and change feature-map sizes via convolution and pooling.

$C_{\text{in}}$ :  $k_0 + (l-1)k$

Concatenate

$l = 21001$  index  
(0~4)

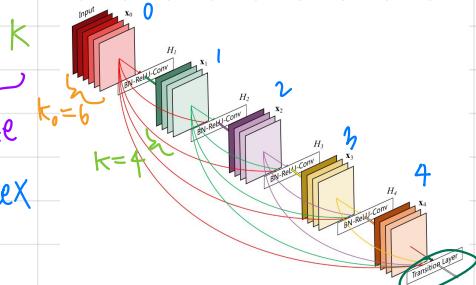


Figure 1: A 5-layer dense block with a growth rate of  $k = 4$ . Each layer takes all preceding feature-maps as input.

$C_{\text{out}}$ :  $k_0 + N \cdot k$

$$= 6 + 5 \cdot 4$$

= 26 (Transition 01 번째는 Dense Block의 첫 번째 채널 수)

$N = \text{layer 개수}$

## 5) Bottleneck layers

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			
Dense Block (2)	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			
Dense Block (3)	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			
Dense Block (4)	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.

↳ Dense block : (x1 차원의 bottleneck 블록 (resnet 참고) 향상↑)

↳ DenseNet-B :  $H_o = \underbrace{\text{BN-ReLU-Conv}(1 \times 1)}_{4k(\text{4Unit}) \text{ f.m}} - \underbrace{\text{BN-ReLU-Conv}(3 \times 3)}_{\text{3Unit}} \rightarrow \text{output 4Unit f.m from input 3Unit}$

## 6) Compression

→ transition layer 은 3Unit f.m 3Unit

Conv + pooling

(= BN + 1x1 conv + 2x2 avg pooling)

→ Dense block ; min f.m + transition layer ;  $\lfloor \theta_m \rfloor$  f.m

$0 < \theta \leq 1$  : compression factor

(transition layer 차원의 1x1 conv 3Unit f.m의 차원은  $\frac{1}{\theta}$  차원)

$$\theta = 1 \text{ or } 2 \text{ or } \dots$$

↳ DenseNet-C :  $\theta < 1$  (0.5 차원)

↳ DenseNet-BC : Bottleneck +  $\theta (= 0.5)$

## 7. Implement 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			
Dense Block (2)	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			
Dense Block (3)	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			
Dense Block (4)	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.

↳ DenseNet-BC

from DenseNet

→ 2kUnit growth rate (ch)

$\left. \right\} kUnit$

# 4. Experiments

(실험 결과)

Method	Depth	Params	C10	C10+	C100	C100+	SVHN
Network in Network [22]	-	-	10.41	8.81	35.68	-	2.35
All-CNN [3]	-	-	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	-
FractionalNet [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	<b>7.00</b>	5.24	<b>27.55</b>	24.42	1.79
DenseNet ( $k = 12$ )	100	7.0M	<b>5.77</b>	<b>4.10</b>	<b>23.79</b>	<b>20.20</b>	1.67
DenseNet ( $k = 24$ )	100	27.2M	<b>5.83</b>	<b>3.74</b>	<b>23.42</b>	<b>19.25</b>	<b>1.59</b>
DenseNet-BC ( $k = 12$ )	100	0.8M	<b>5.92</b>	4.51	<b>24.15</b>	22.27	1.76
DenseNet-BC ( $k = 24$ )	250	15.3M	<b>5.19</b>	<b>3.62</b>	<b>19.64</b>	<b>17.60</b>	1.74
DenseNet-BC ( $k = 40$ )	190	25.6M	-	<b>3.46</b>	-	<b>17.18</b>	-

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.

Model	top-1	top-5
DenseNet-121	25.02 / 23.61	7.71 / 6.66
DenseNet-169	23.80 / 22.08	6.85 / 5.92
DenseNet-201	22.58 / 21.46	6.34 / 5.54
DenseNet-264	22.15 / 20.80	6.12 / 5.29

Table 3: The top-1 and top-5 error rates on the ImageNet validation set, with single-crop / 10-crop testing.

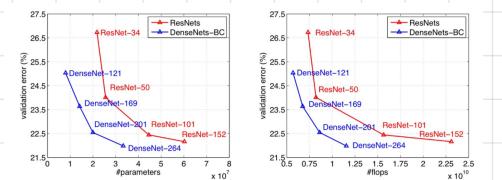


Figure 3: Comparison of the DenseNets and ResNets top-1 error rates (single-crop testing) on the ImageNet validation dataset as a function of learned parameters (left) and FLOPs during test-time (right).

↳ ResNet hyper 힘을 높임

→ ACC↑, capacity(parameters), parameter efficiency, overfitting↓)

# 5. Discussion

## 1. "Model Compactness"

→ from concatenate (feature reuse)

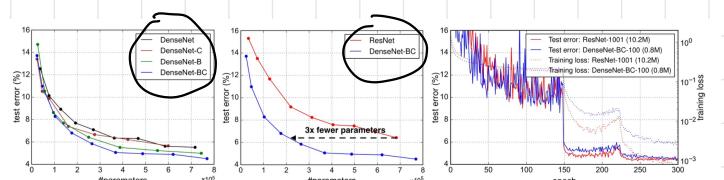


Figure 4: Left: Comparison of the parameter efficiency on C10+ between DenseNet variations. Middle: Comparison of the parameter efficiency between DenseNet-BC and (pre-activation) ResNets. DenseNet-BC requires about 1/3 of the parameters as ResNet to achieve comparable accuracy. Right: Training and testing curves of the 100-layer pre-activation ResNet [12] with more than 10M parameters and a 100-layer DenseNet with only 0.8M parameters.

## 2. "Implicit Deep Supervision"

→ 각 블록은 다른 블록의 loss를 통해 학습

→ gradient, loss 전파



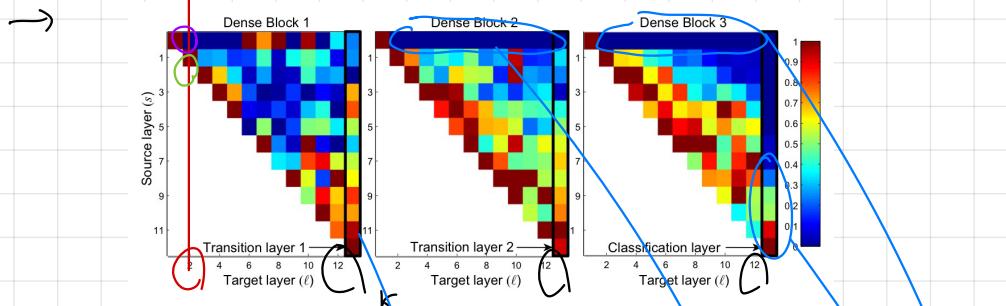
### 3. "Stochastic VS Deterministic Connection"

ResNet

→ ResNet dropout 시 유행 풀는 것과 concatenate

↳ pooling 풀는 drop x ≈ DenseNet

### 4. "Feature Reuse"



**Figure 5:** The average absolute filter weights of convolutional layers in a trained DenseNet. The color of pixel  $(s, l)$  encodes the average  $L_1$  norm (normalized by number of input feature-maps) of the weights connecting convolutional layer  $s$  to  $l$  within a dense block. Three columns highlighted by black rectangles correspond to two transition layers and the classification layer. The first row encodes weights connected to the input layer of the dense block.

↳  $(l, s)$  : 해당 뉴런은  $l$  번째 층까지 사용된 f.m 사용↑ (가중치↑)  
↳ 전이 계층  
conv

1. All layers spread their weights over many inputs within the same block.

→ 같은 뉴런이 대체로의 종이 여러 곳에 퍼져  
걸고 가중치를 분산한다.

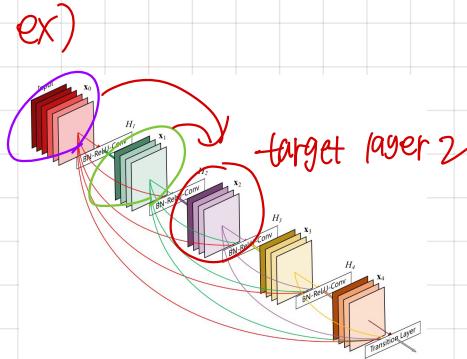
2. The weights of the transition layers also spread their weight across all layers within the preceding dense block,

→ Transition layer 역시 이런 특징의 예를 풀  
할 때 두 가지 가능성을 한다

3. The layers within the second and third dense block consistently assign the least weight to the outputs of the transition layer (the top row of the triangles),

→ 그 뉴런은 복잡 내부 풀는 것에  
transition layer 풀기 = heatmap 풀기  
이 가능해질 가능성이 있다

4. Although the final classification layer, shown on the very right, also uses weights across the entire dense block,



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

→ DenseNet의 경우 앞에서 몇 단계까지 풀기  
하는 것과 계층을 잘 풀

→ 이런 DenseNet-BC가 transition layer θ (전이 뉴런)을  
통해 가능성을 (증폭이거나)

→ 그런 오른쪽 카운트 블록 중도 블록 간의  
특성들에 대해 가중치

→ but, 마지막 카운트가 더 가중치↑ = "고수운 특징 사용"

## 6. Conclusion

친밀화된 모드 X

TURDIE↑ ↗ fine acc↑

인코딩 X

TURDIE↓ ↘

깊은 감독, Deep supervision, 차양한 갈이, Feature Reuse, Compact