<u>DenseNet-Summary</u>

1. <u>ABSTRACT:-</u>

- for L layers,it has L(L+1)/2 connections
- Alleviate vanishing gradients
- Strengthen feature propagation
- Encourage feature reuse
- Decrease the number of parameters

2. <u>INTRODUCTION:-</u>

- As CNNs become increasingly deep,they introduce a new research problem of vanishing gradients
- This problem is addressed by ResNet, highway networks using identity mapping
- Many such papers like stochastic depth,fractalnet share a common characteristic “**they create short paths from early layers to later layers**”
- **In this architecture,they distill this insight into a simple connectivity pattern-> maximize information flow between layers**
- l th layer has l inputs, and its feature maps are passed onto **L-l layers**.

- **Requires less parameters:-**
    - Resnet can be viewed as an algo which takes a previous state and writes to subsequent states but also storing information which needs to be passed down.
    - Studies say that many layers are redundant and can be randomly dropped while training
    - **DenseNet** clearly differntiates information that is added and that is passed down,and doesnt need extra layers
- DenseNet layers are very narrow(around 12 filters per layer)
- **Improved gradient flow and flow of information:-**
    - Each layer has a direct access to the gradients from the loss function and original input signal
- Regularizing effect

3. DENSENET:-

- Input image- $\mathbf{X_0}$,network of **L layers,** each implementing a non-linear transformation $\mathbf{H_l(.)}$ where l indexes the layer
- $\mathbf{H_l(.)}$ can be a composite function of operations such as batch normalization,ReLU,Pooling,Conv
- **ResNet:-**
  - $X_l = H_l(X_{l-1})+X_{l-1}$
  - An advantage of resnet is **gradient flow can go directly** through identity mapping,from later layers to previous layers.
  - But, Identity function and output $\mathbf{H_l}$ are combined by summation,which **impedes with the information flow**
- **DenseNet:-**
- $X_l = H_l([X_0,X_1,X_{2,}\dots,X_{l-1}])$
- Composite function:- $H_l(.)$ as a composite function of BN -> ReLU -> 3x3 conv2d
- Pooling layers:- concatenation of layers is only feasible for same size feature maps,so the network is divided into densely-connected blocks,and pooling is done between them.
- Transition layer:- BN-> 1x1 conv ->2x2 avgpool2d

- Growth rate:-
  - if k_0 input no.of channels, and each function produces k feature maps, then l^{th} layer has $k_0+k*(l-1)$ feature maps.
  - DenseNet can have very narrow networks(i.e eg k=12) while other layers have wider networks.
  - Because the information flow obtained by wide networks can be achieved using narrow networks as we already concatenate previous layers
- Bottleneck layers:-
  - 1x1 conv is introduced before every 3x3 conv to reduce input feature maps,improving computational efficiency
  - BN->ReLU->Conv(1x1)->BN->ReLU->Conv(3x3)
- Compression:-
  - If a block generates m feature-maps,we let transition layer generate $[\theta m]$ feature maps,where $0 \leq \theta \leq 1$
  - DenseNet-C :- $\theta < 1$
  - DenseNet-BC:- both bottlenet and transition layers $\theta < 1$ are used

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			

- Implementation details:-
  - DenseNet used in our experiments has three dense blocks that each has an equal number of layers(except **ImageNet**)
  - Before entering the first dense block, a convolution with 16 (or twice the growth rate for DenseNet-BC) output channels is performed on the input images
  - For 3x3 conv-> zeropadded one pixel
  - 1x1 conv-> 2x2 avg pool as transition layer(between two contiguous denseblocks)
  - At the end ,global avg pool,followed by softmax classifier

4. DISCUSSION:-

- Model Compactness:-
    - As a result of input concatenation, the feature-maps learned by any of the DenseNet layers can be accessed by all subsequent layers. This encourages feature reuse making compact models.
- Implicit deep supervision:-
    - Each layer gets extra supervision from the loss function through the shorter connections.