CSHO23 – DEEP LEARNING

ASSIGNMENT 1

NETWORK ARCHITECTURE: DENSENET-121

TEAM MEMBERS:

106118001 - BALA ADITYA ANAND

106118019 – BALAJI JAYASHRRI

106118053 – MADHAV AGGARWAL

MOTIVATION FOR DENSENET ARCHITECTURE

A DenseNet is a type of CNN that utilizes dense connections between layers, where all layers are directly connected. In a standard CNN, connections exist between each layer and its subsequent layer to obtain high-level features. However, this leads to declining accuracy in high-level networks due to a vanishing gradient caused by longer paths between the input and output layers.

A DenseNet architecture solves this problem by connecting every layer to every other layer in a feed-forward fashion, leading to shorter connections between layers close to the input and layers close to the output. This approach also shows other advantages such as strengthening feature propagation, encouraging feature reuse, and reducing the number of parameters required.

DENSENET STRUCTURE

1. DENSE BLOCK

A dense block is a module used in the DenseNet architecture that connects all the contained convolutional layers directly to each other in a feed-forward manner. For a particular layer, the feature-maps of all the preceding layers are used as inputs, and its feature-map is used as input to all subsequent layers. In contrast with the ResNet architecture, the features are not combined through summation but instead through concatenation. A single layer is a composite function of multiple operations such as batch normalization, ReLU, pooling or convolution.

The growth rate is defined as the number of output feature-maps of a layer. It is a hyperparameter that regulates how much new information each layer contributes to the global state. All layers within the same dense block share a collective knowledge, which leads to lower redundancy in the learned features. Therefore, narrow layers and a relatively small growth rate is sufficient to obtain state-of-the-art results.

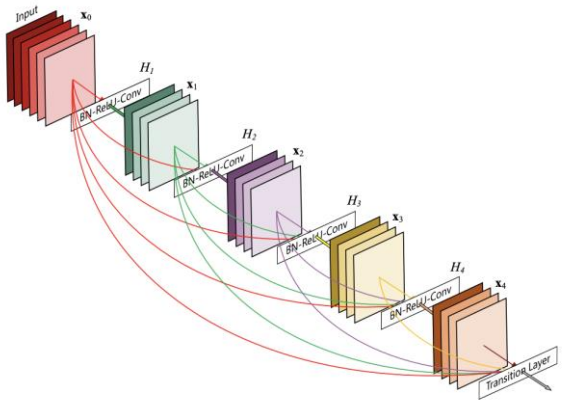


Figure 1: A 5-layer dense block with a growth rate of $k = 4$. Each layer takes all preceding feature-maps as input.

2. TRANSITION LAYER

When the size of the feature-maps propagated between layers change, the viability of the concatenation operation comes into question. An essential part of the CNN architecture is the down-sampling of feature-maps. Introducing concatenated dense connections to serve as input to the current layer fails to incorporate this. Hence, to facilitate down-sampling, the architecture is divided into dense blocks with transition layers between them. These transition layers perform convolution and pooling and typically consist of a batch normalization layer, a 1×1 convolutional layer and a 2×2 average pooling layer.

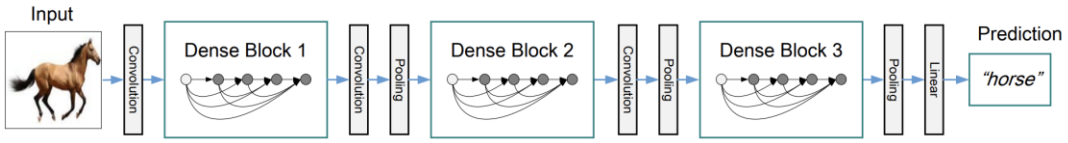


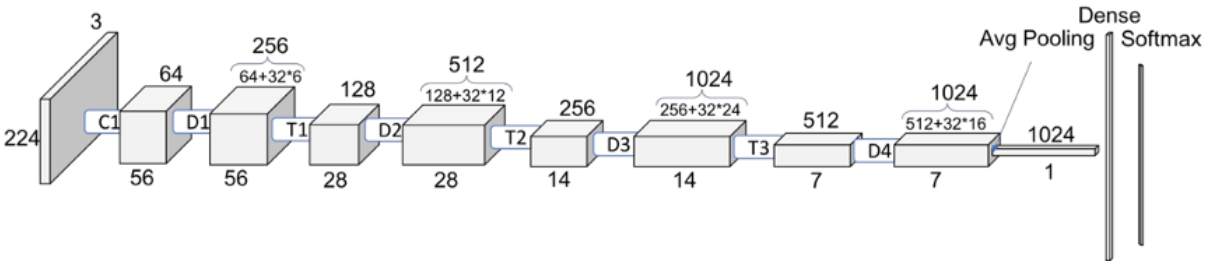
Figure 2: A deep DenseNet with three dense blocks. The layers between two adjacent blocks are referred to as transition layers and change feature-map sizes via convolution and pooling.

3. BOTTLENECKS AND COMPRESSION

Although each layer only produces a small number of feature-maps, it typically has many more inputs instead. In order to improve computational efficiency, the number of input feature-maps can be reduced by incorporating a 1×1 convolution as a bottleneck before each 3×3 convolution.

In order to further improve efficiency, the number of feature-maps across transition layers can be reduced as well. The hyperparameter compression factor determines the number of feature-maps that are allowed to pass across the transition layer.

DENSENET-121

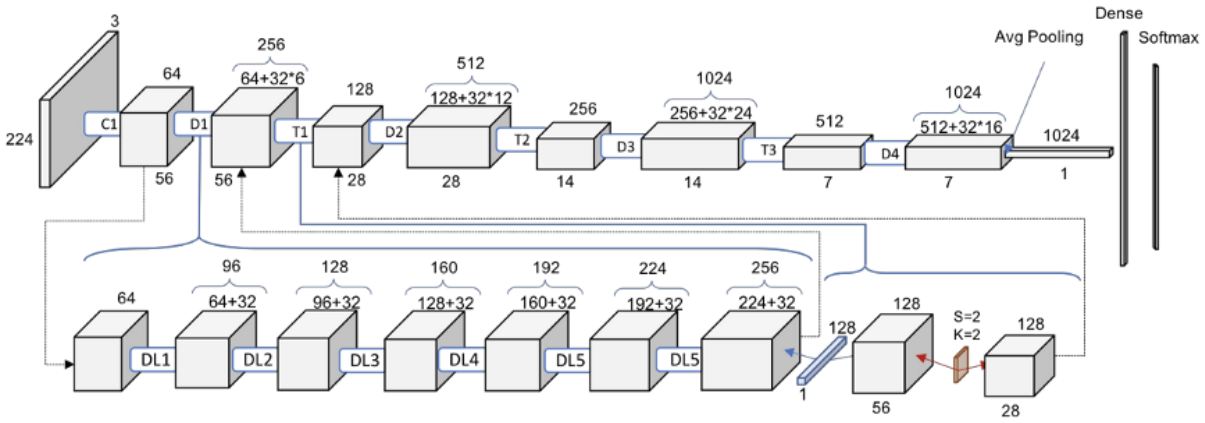


The above diagram shows the overall architecture of the DenseNet-121 on a block level. The values at the bottom of each volume indicates the size, and those at the top indicate the dimensions of the corresponding feature-map. This architecture uses a growth rate of 32 and consists of the following arrangement of blocks and transition layers.

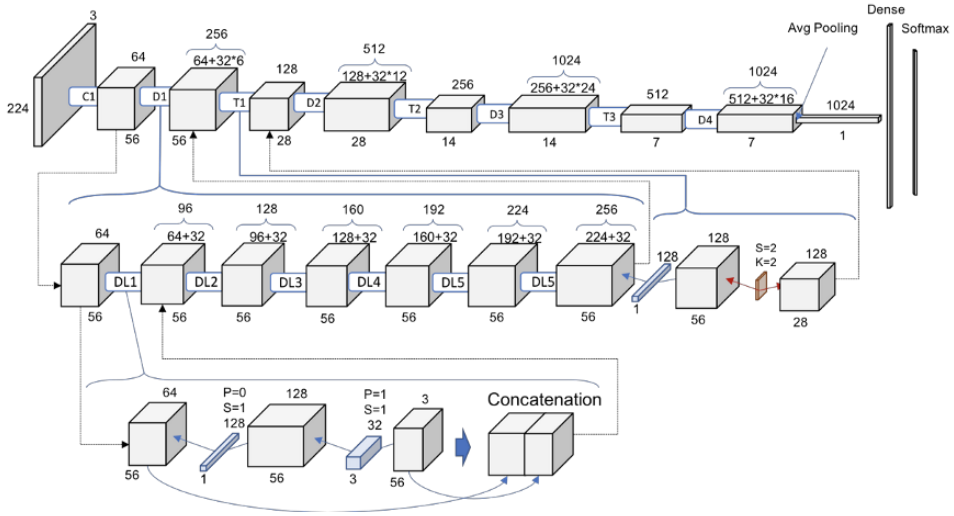
Layers	**Operation**
Convolution	7×7 Convolution, Stride 2
Pooling	3 ×3 Max Pooling, Stride 2
Dense Block (1)	[1×1 and 3×3 Convolution] * 6
Transition Layer (1)	[1×1 Convolution and 2×2 Average Pooling, Stride 2]
Dense Block (2)	[1×1 and 3×3 Convolution] * 12
Transition Layer (2)	[1×1 Convolution and 2×2 Average Pooling, Stride 2]
Dense Block (3)	[1×1 and 3×3 Convolution] * 24
Transition Layer (3)	[1×1 Convolution and 2×2 Average Pooling, Stride 2]
Dense Block (4)	[1×1 and 3×3 Convolution] * 16
Classification Layer	7×7 Average Pooling and Softmax

The dimension of the feature-map of each dense block is calculated from the dimension of the previous volume's feature-map and the number of layers in the block as given below.

$$Dimension(V_x) = Dimension(V_{x-1}) + (Growth\ Rate\ x\ Number\ of\ Layers)$$



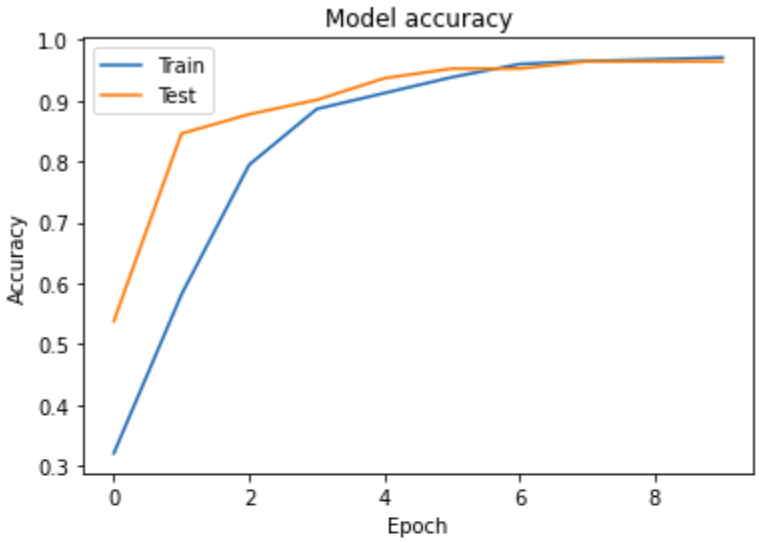
Inside the first dense block, there are 6 dense layers, each of which adds dimensions equal to the growth rate to the dimension of the input feature-map. Therefore, in 6 layers, an input of dimension 64 results in a feature-map of dimension 256. The transition block following the dense block performs a 1×1 convolution with 128 filters, followed by a 2×2 pooling which halves the size to 28 from 56 and dimension to 128. This is passed as input to the next block.

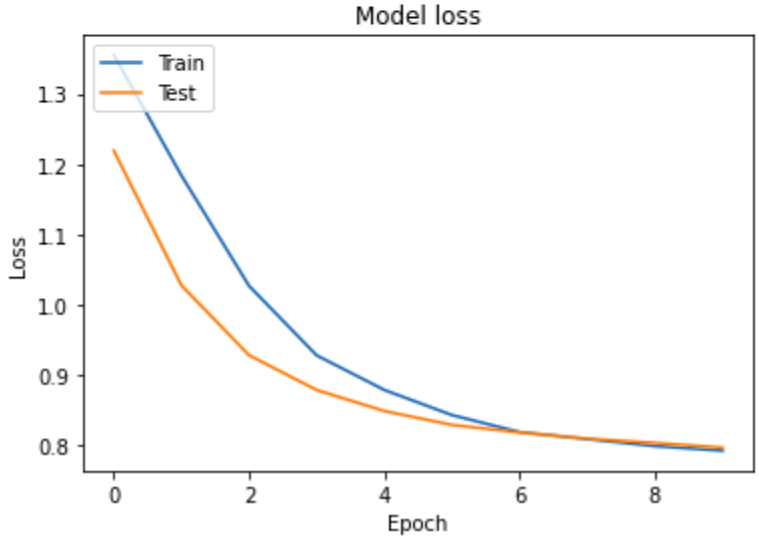


Inside a dense layer, a 1×1 convolution with 128 filters is performed to reduce the size of the feature-map, following which a 3×3 convolution is performed to get a volume of dimension 32. This is then concatenated with the input volume and the result is passed as input to the next layer.

TRANSFER LEARNING WITH DENSENET-121

For our version of the DenseNet Architecture using Transfer Learning we have added a Convolution Layer – Global Average Pooling followed by a Dropout layer with dropout rate 0.2. The results are finally compiled with a Softmax Layer at the end. The results obtained using the transfer-learning approach are close to 97% in accuracy with very minimal loss indicating very good fitting over the model, and a good testing accuracy.





For the approach with the conventional Densenet-121 architecture we obtain an accuracy of 94% while training and 65% while testing. Still indicating satisfactory results but accuracy being much better with the transfer-learning approach. This can be due to slight overfitting on the training data.