State of the art CNN architectures

Dr. Tushar Sandhan

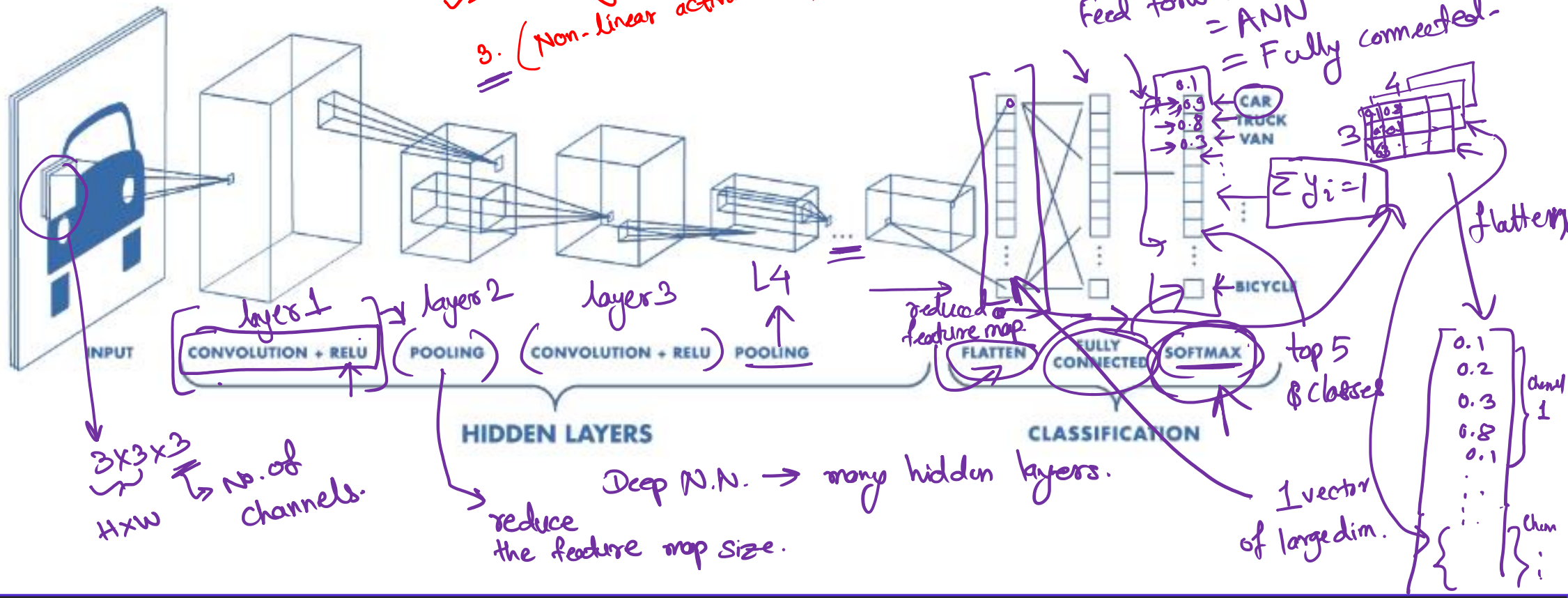
sandhan@iitk.ac.in

Contents

- AlexNet (2012)
- VGG-16
- Inception-v1
- Resnet-50
- ResNeXt-50
- DenseNet
- EfficientNet

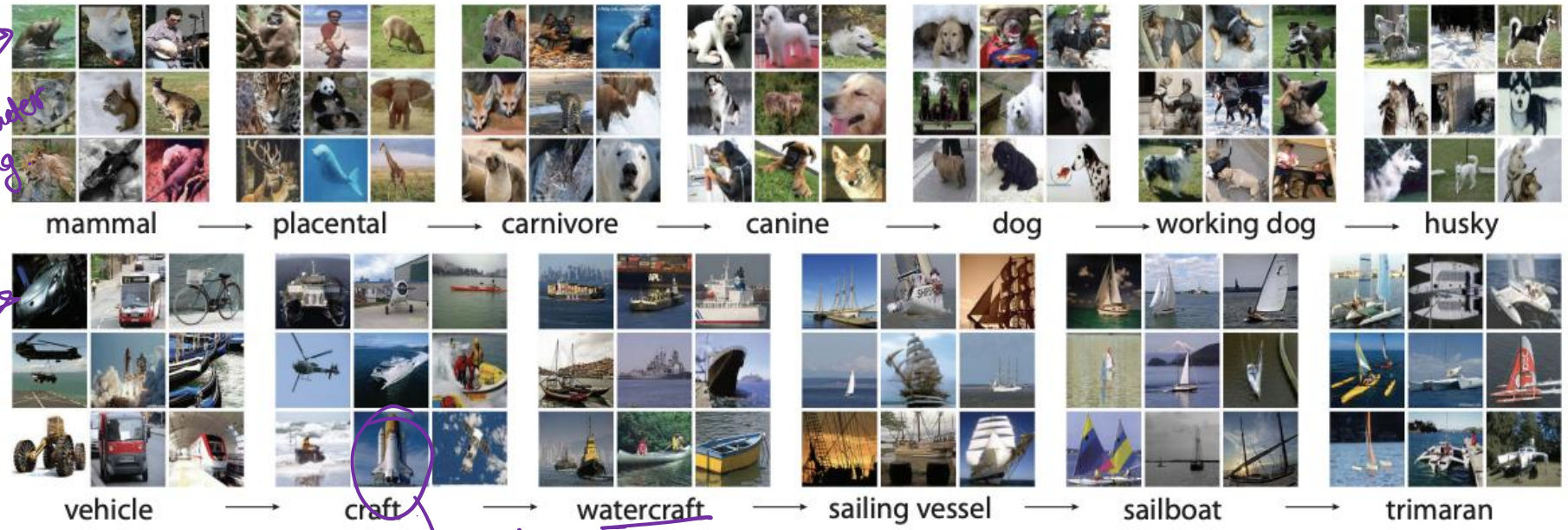
CNN Structure

- These operations are repeated over tens or hundreds of layers, with each layer learning to identify different features.

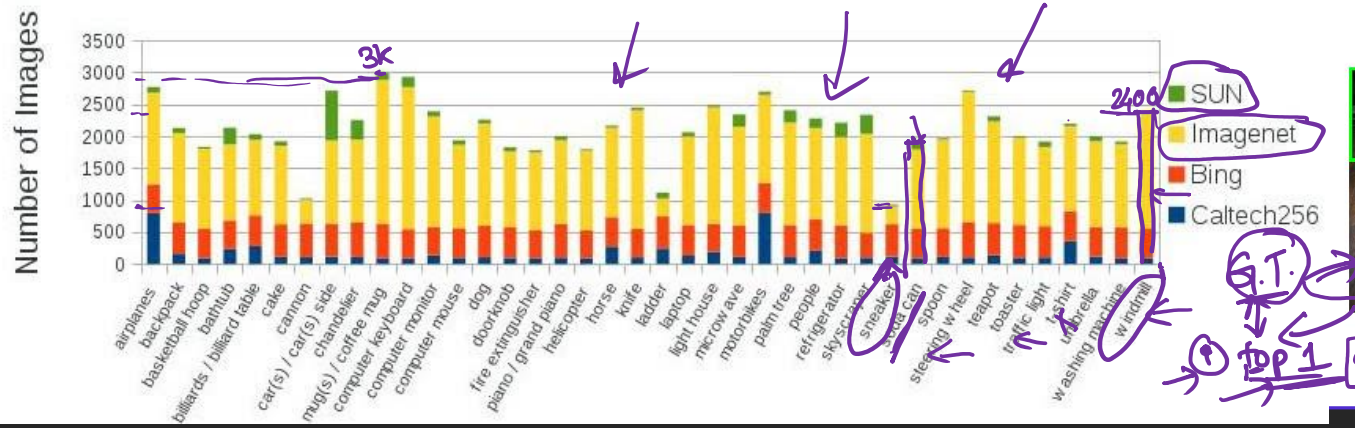


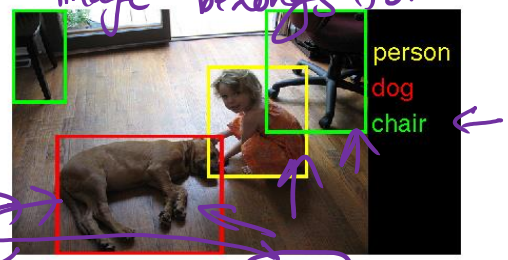
AI large scale problems

- ImageNet-1k
  - Train: 1,281,167
  - Val.: 50,000
  - Test: 100,000
  - Classes: 1000



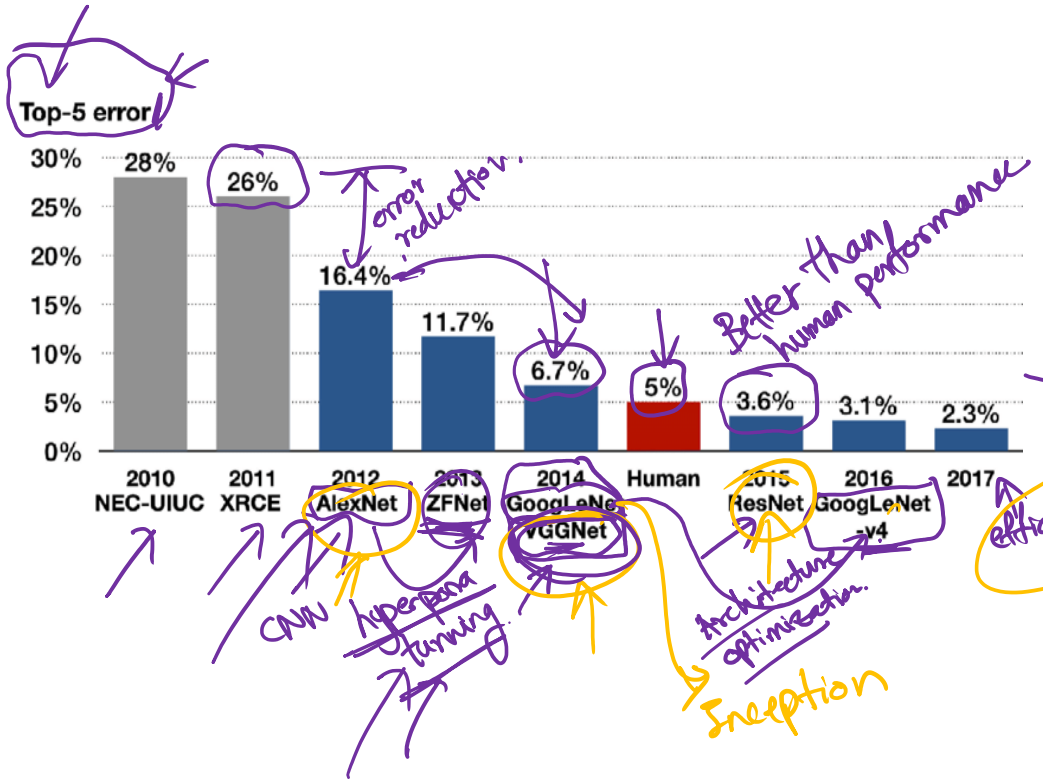
- ImageNet
  - Train: 14,197,122
  - Classes: 21,841

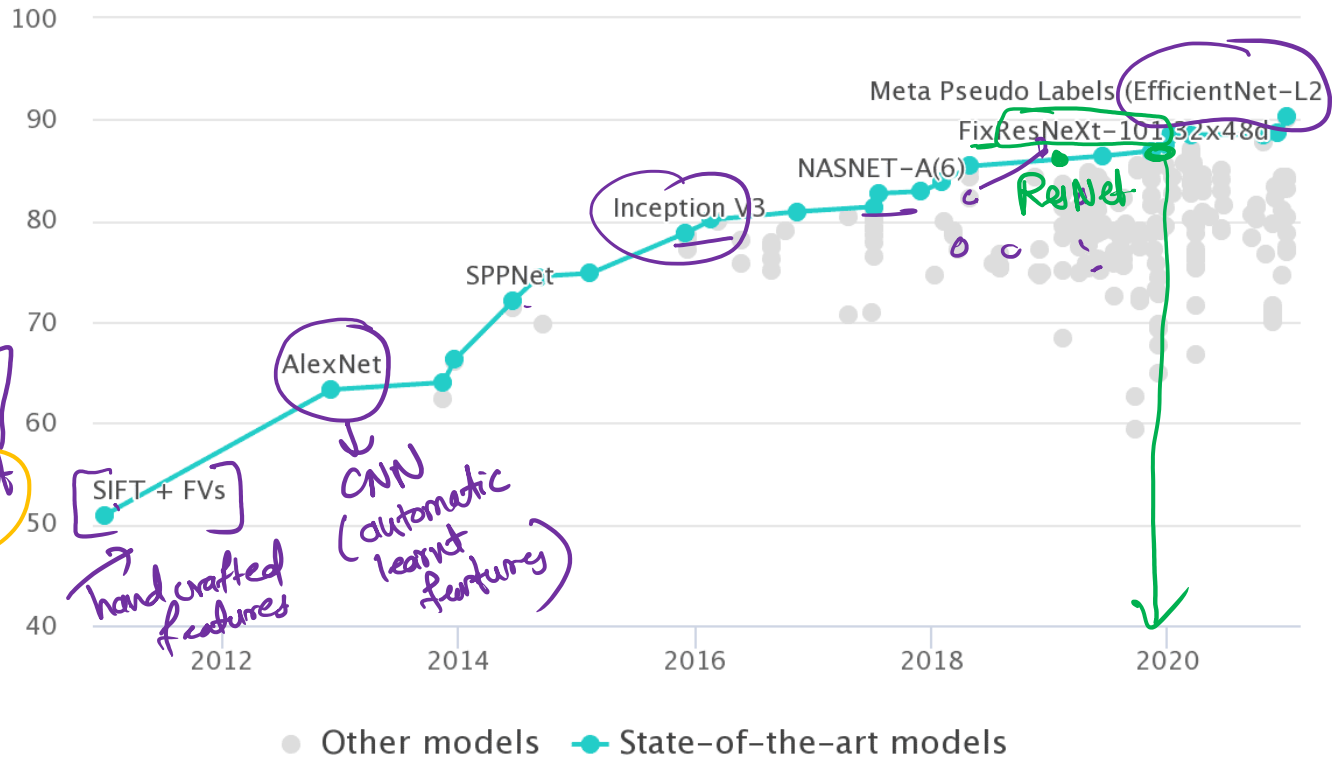




sandhan@iitk.ac.in

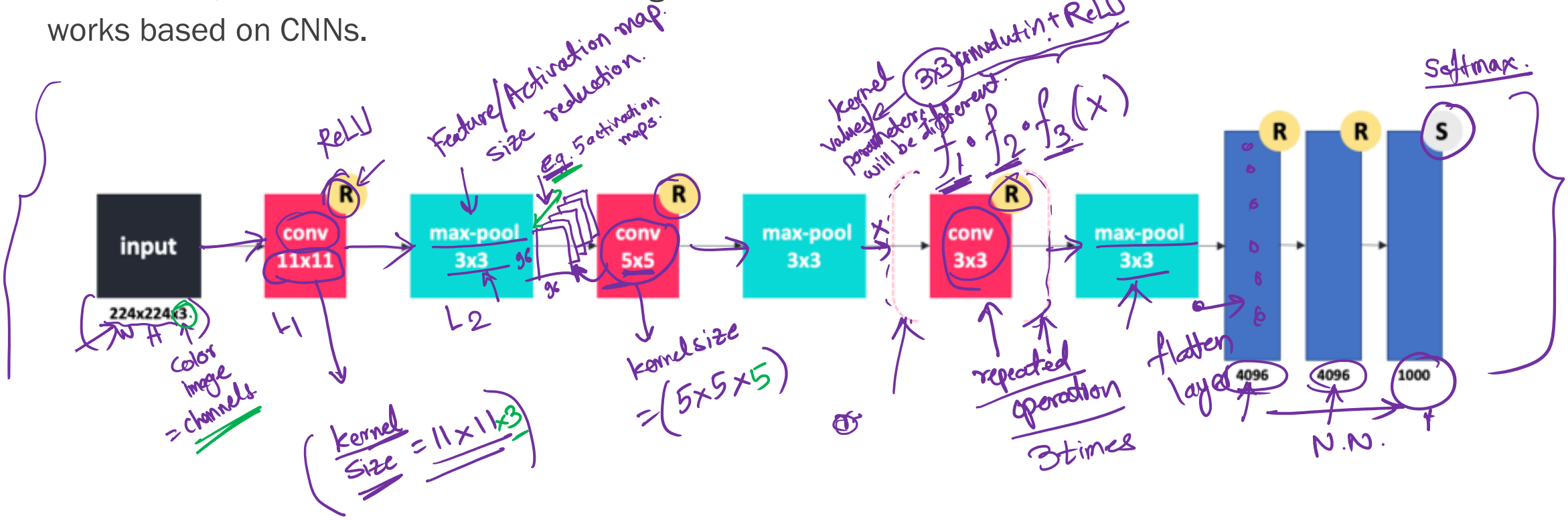
Different CNNs





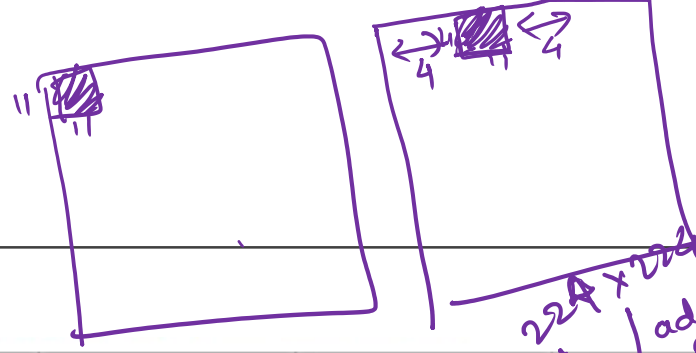
AlexNet (2012)

- In 2012, Alex Krizhevsky, Ilya Sutskever, and Geoff Hinton won the ImageNet Large Scale Visual Recognition Challenge with a test accuracy of 84.6%. The model significantly outperformed the second runner-up (top-5 error of 16% compared to runner-up with 26% error). Krizhevsky used GPUs to train the AlexNet, which enabled faster training of CNNs models and started a wave of interest and new works based on CNNs.



sandhan@iitk.ac.in

AlexNet (2012)

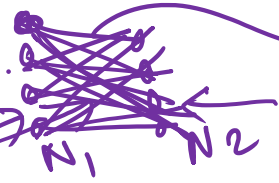


	Layer	# filters / neurons	Filter size	Stride	Padding	Size of feature map o/p.	Activation function
	Input	-	-	-	-	227 x 227 x 3	-
1	Conv 1	96	11 x 11	4	-	55 x 55 x 96	ReLU
2	Max Pool 1	-	3 x 3	2	-	27 x 27 x 96	-
3	Conv 2	256	5 x 5	1	2	27 x 27 x 256	ReLU
4	Max Pool 2	-	3 x 3	2	-	13 x 13 x 256	-
5	Conv 3	384	3 x 3	1	1	13 x 13 x 384	ReLU
6	Conv 4	384	3 x 3	1	1	13 x 13 x 384	ReLU
7	Conv 5	256	3 x 3	1	1	13 x 13 x 256	ReLU
8	Max Pool 3	-	3 x 3	2	-	6 x 6 x 256	-
9	Dropout 1	rate = 0.5	-	-	-	6 x 6 x 256	-

i/p. o/p.

Dropout 1 — useful for regularizing the neural network.

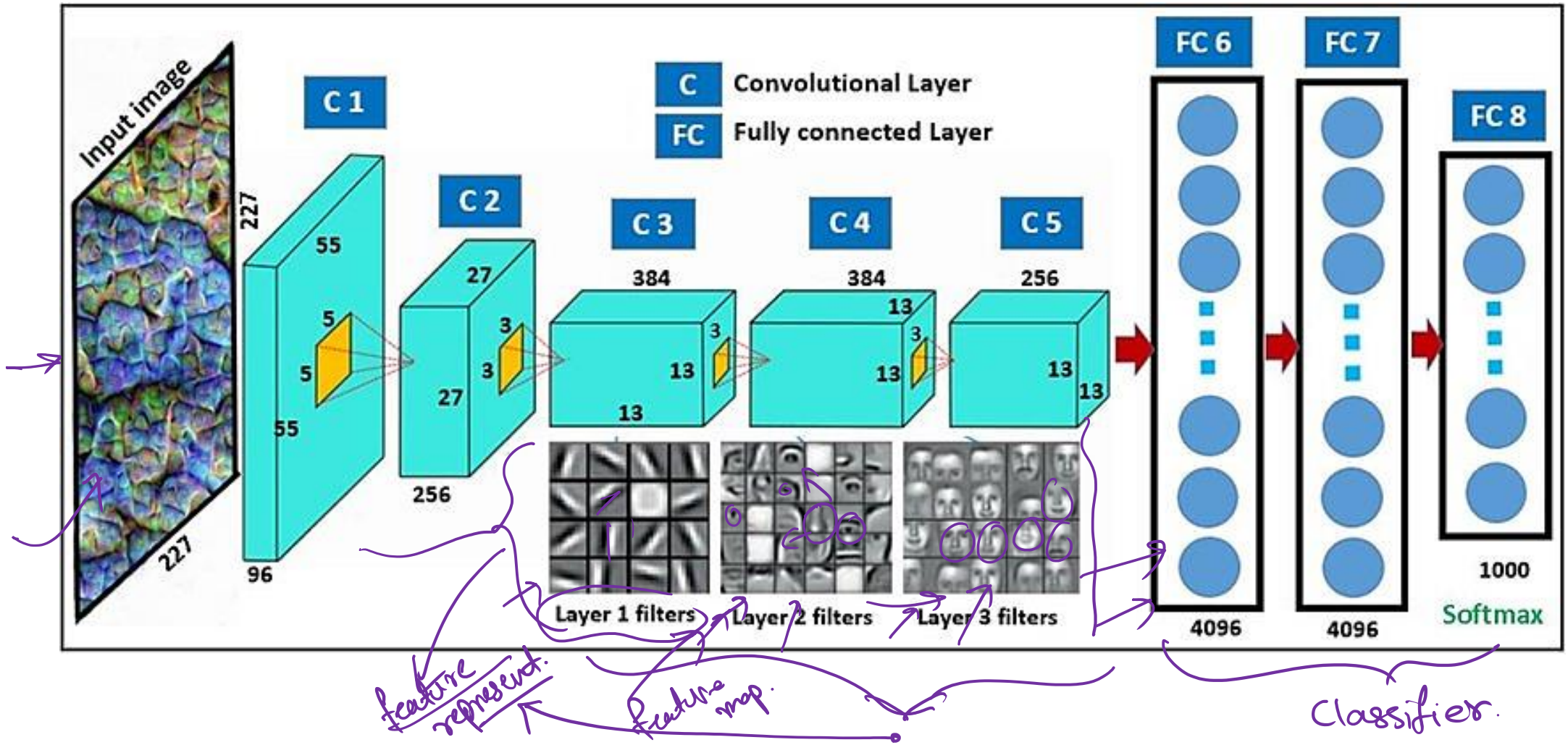
(fully connected layers) — To avoid it — may cause overfitting.



# of connections = $N_1 \times N_2$

sandhan@iitk.ac.in

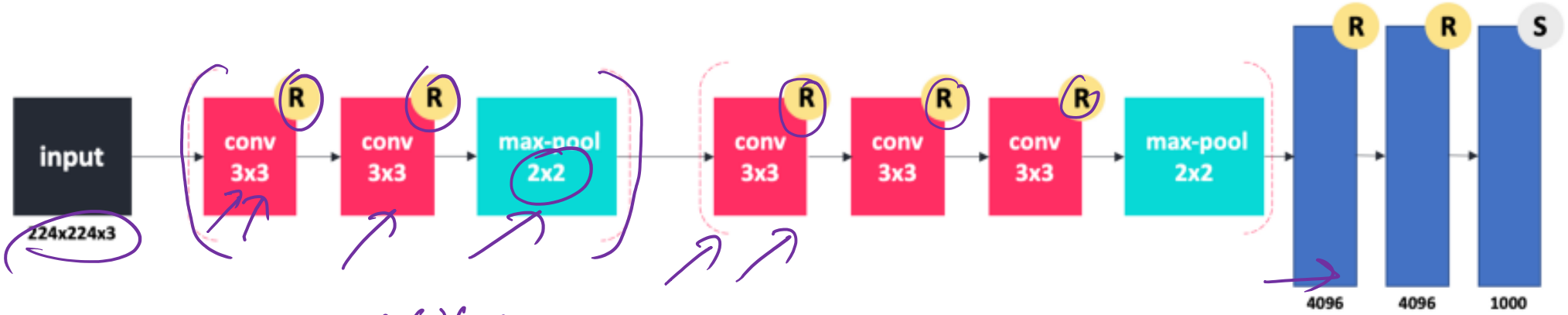
Modified AlexNet



sandhan@iitk.ac.in

VGG-16

- The model achieves 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes. It was proposed by Karen Simonyan and Andrew Zisserman of the Visual Geometry Group Lab of Oxford University in 2014.



Inception

- Inception net achieved a milestone in CNN classifiers when previous models were just going deeper to improve the performance and accuracy but compromising the computational cost. The Inception network, on the other hand, is heavily engineered. It uses a lot of tricks to push performance, both in terms of speed and accuracy.

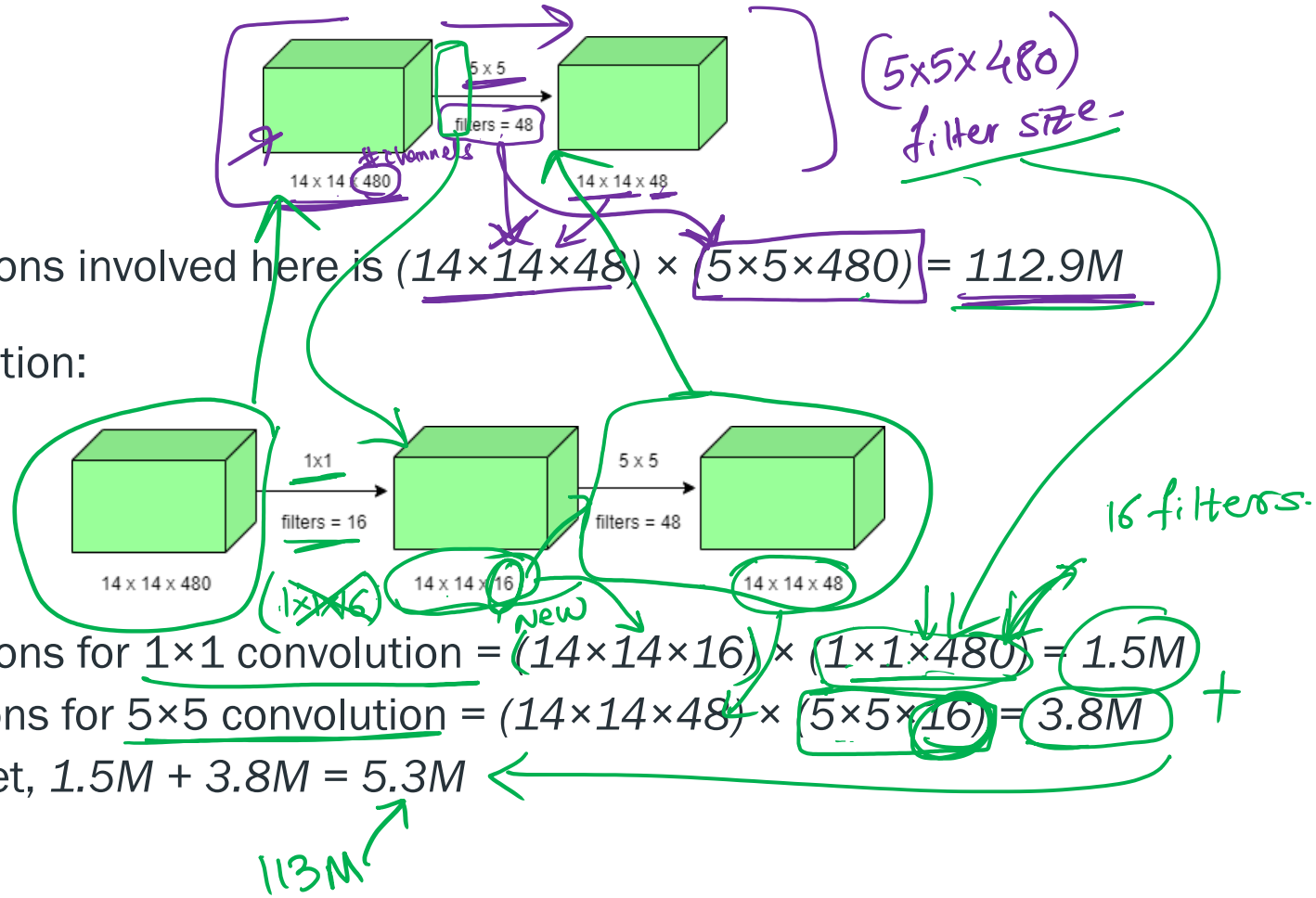
- It is the winner of the ImageNet Large Scale Visual Recognition Competition in 2014, an image classification competition, which has a significant improvement over ZFNet (The winner in 2013), AlexNet (The winner in 2012) and has relatively lower error rate compared with the VGGNet (1st runner-up in 2014).

Before digging into Inception Net model, it's essential to know an important concept that is used in Inception network:

- **1 X 1 convolution:** A *1×1* convolution simply maps an input pixel with all its respective channels to an output pixel. *1×1* convolution is used as a dimensionality reduction module to reduce computation to an extent.

Inception

- For instance, we need to perform 5×5 convolution without using 1×1 convolution as below:

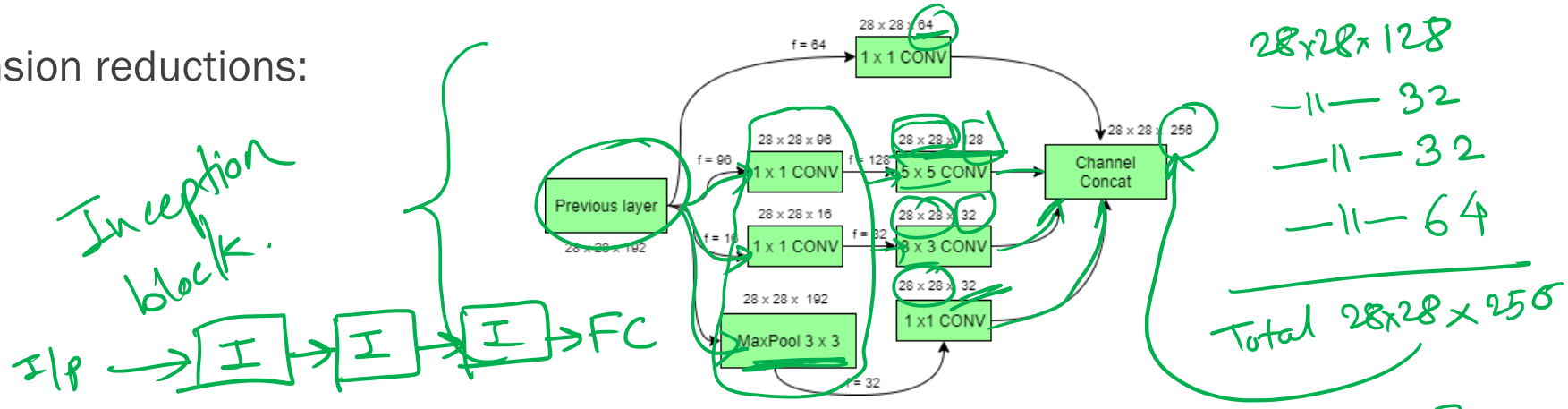


- Number of operations involved here is $(14 \times 14 \times 48) \times (5 \times 5 \times 480) = 112.9M$
- Using 1×1 convolution:
- Number of operations for 1×1 convolution = $(14 \times 14 \times 16) \times (1 \times 1 \times 480) = 1.5M$
  Number of operations for 5×5 convolution = $(14 \times 14 \times 48) \times (5 \times 5 \times 16) = 3.8M$
  After addition we get, $1.5M + 3.8M = 5.3M$

Inception

- Inception is immensely smaller than 112.9M ! Thus, 1×1 convolution can help to reduce model size which can also somehow help to reduce the overfitting problem.

- Inception model with dimension reductions:

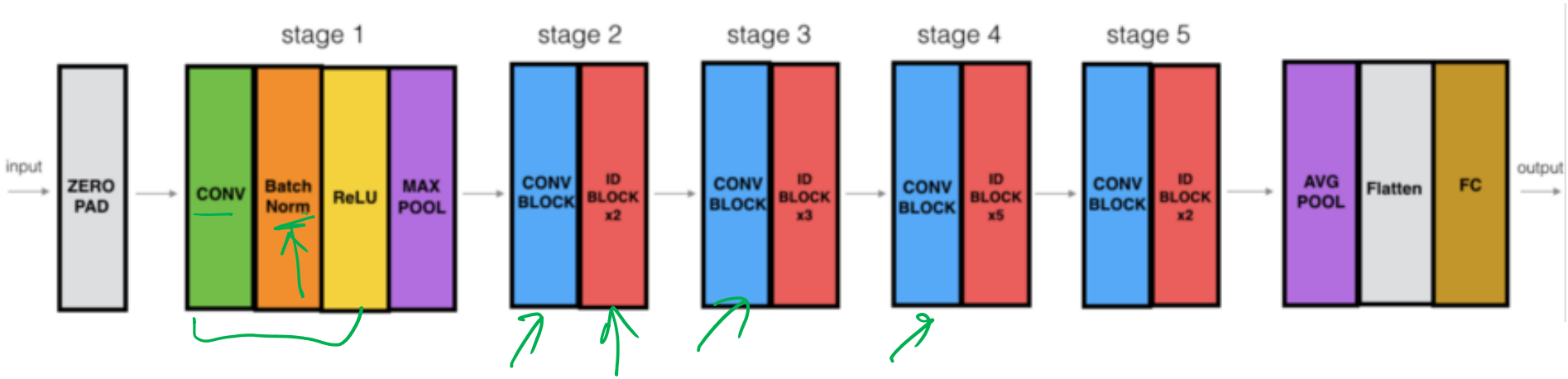


- Deep Convolutional Networks are computationally expensive. However, computational costs can be reduced drastically by introducing a **1 x 1 convolution**. Here, the number of input channels is limited by adding an extra *1×1* convolution before the *3×3* and *5×5* convolutions. Though adding an extra operation may seem counter-intuitive but *1×1* convolutions are far cheaper than *5×5* convolutions. Do note that the *1×1* convolution is introduced after the max-pooling layer, rather than before. At last, all the channels in the network are concatenated together i.e. *(28 x 28 x (64 + 128 + 32 + 32)) = 28 x 28 x 256.*

ResNet50

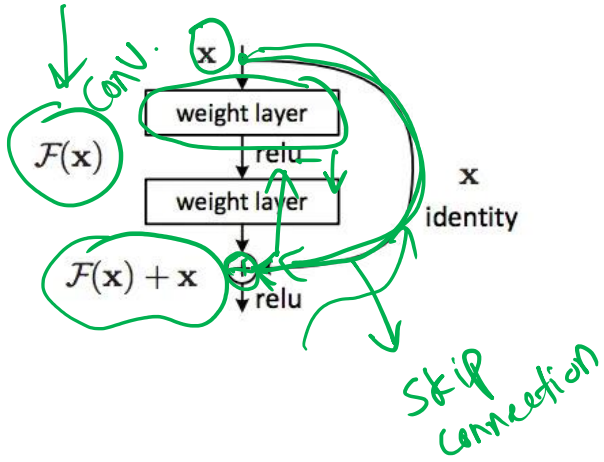
- The network has been created and acquainted by Microsoft with 96.4% accuracy this model won the 2016 ImageNet competition. It is well-known due to its depth (to 152 layers) and the introduction of residual blocks.

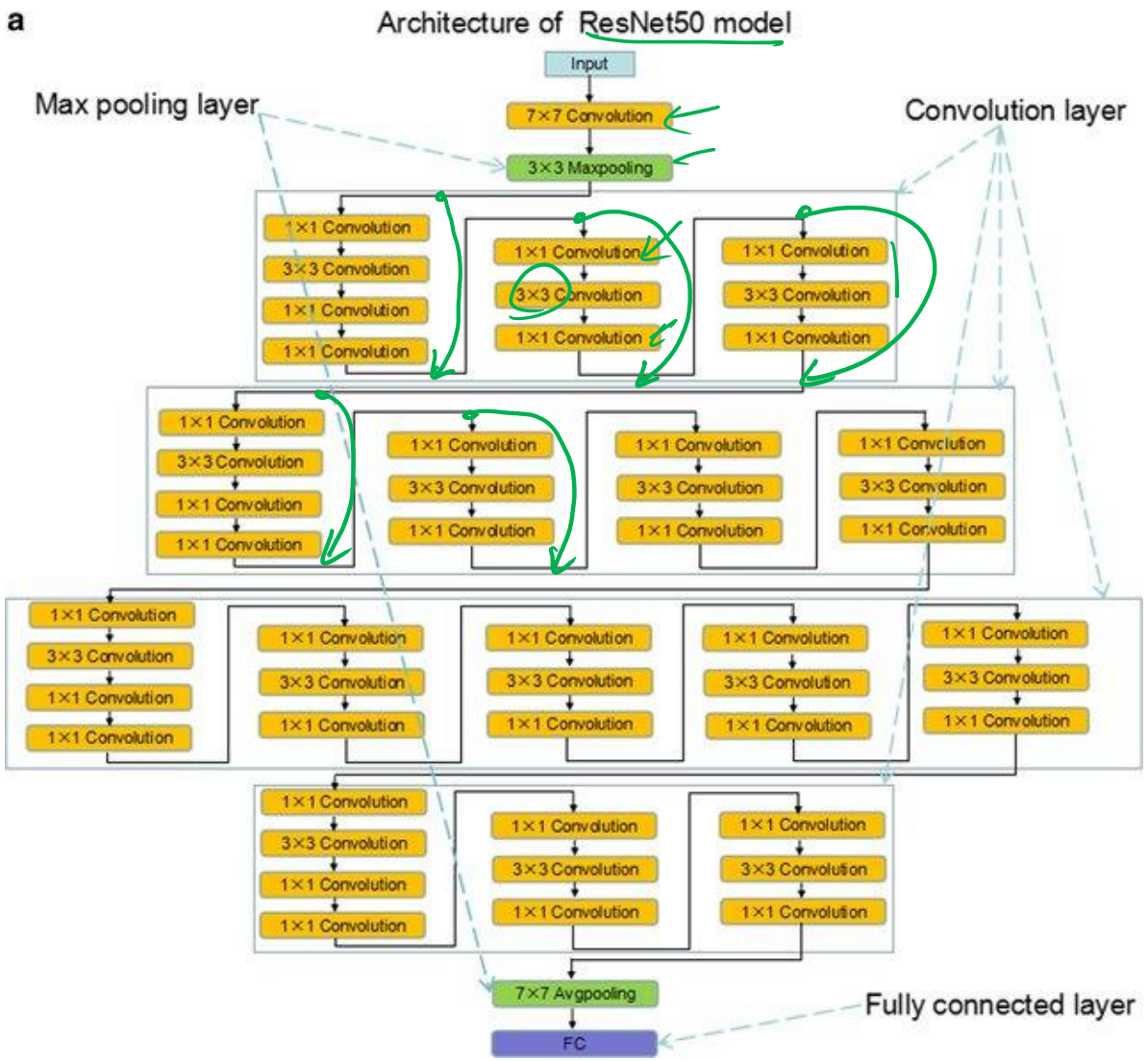
- The identity and convolution blocks coded in the notebook are then combined to create a ResNet-50 model with the architecture shown below:



- The ResNet-50 model consists of 5 stages each with a convolution and Identity block. Each convolution block has 3 convolution layers and each identity block also has 3 convolution layers. The ResNet-50 has over 23 million trainable parameters.

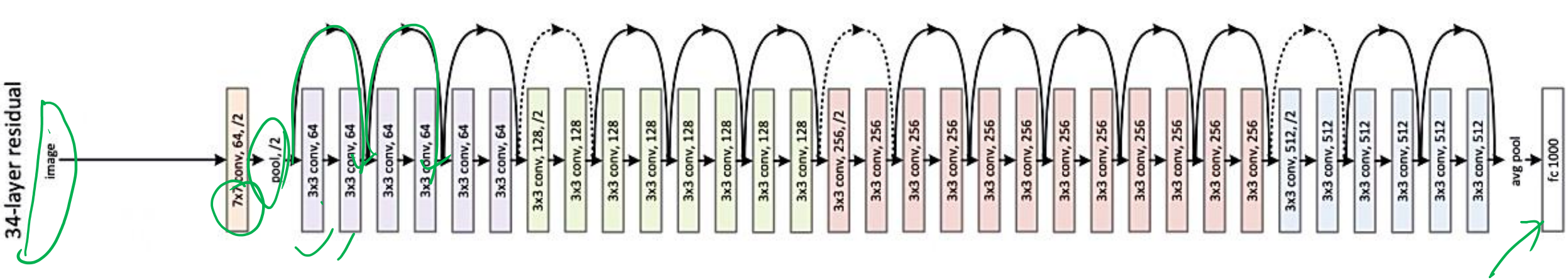
ResNet50





sandhan@iitk.ac.in

ResNet34


34-layer residual
image
7x7 conv, 64, /2
pool, /2
3x3 conv, 64
3x3 conv, 128, /2
3x3 conv, 128
3x3 conv, 256, /2
3x3 conv, 256
3x3 conv, 512, /2
3x3 conv, 512
avg pool
fc 1000

ResNeXt-50

- The model name, ResNeXt, contains Next. It means the next dimension, on top of the ResNet. This next dimension is called the “cardinality” dimension. And ResNeXt becomes the 1st Runner Up of ILSVRC classification task.
- Inherited from ResNet, VGG, and Inception, ResNeXt includes shortcuts from the previous block to next block, stacking layers and adapting split-transform-merge strategy.
- ResNet: Introducing a shortcut from the previous layer to the next layer.
- VGG: Leveraging repeating layers to build a deep architecture model.
- Inception: Following split-transform-merge practise to split the input to multiple blocks and merging blocks later on.
- ResNeXt: The principle is stacking the same topology blocks. Within the residual block, hyper-parameters (width and filter sizes) are shared.

ResNeXt-50

- *Cardinality* is introduced by authors. It means the size of the set of transformations. Refer to the following figure, the architecture includes 32 same topology blocks so the value of cardinality is 32. Because of using the same topology, fewer parameters are required while more layers are added into this architecture.

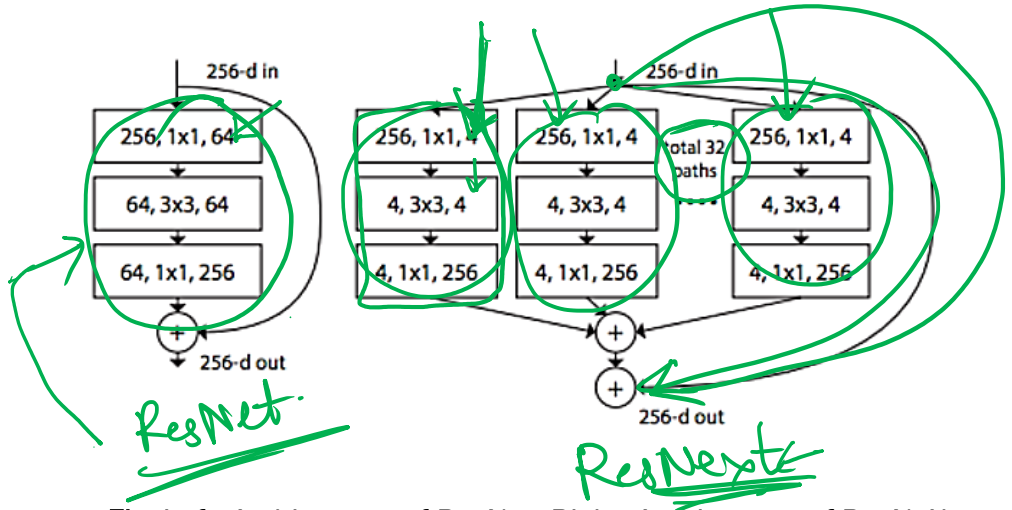


Fig: Left: Architecture of ResNet, Right: Architecture of ResNeXt.

stage	output	ResNet-50	ResNeXt-50 (32×4d)
conv1	112×112	7×7, 64, stride 2	7×7, 64, stride 2
		3×3 max pool, stride 2	3×3 max pool, stride 2
conv2	56×56	[1×1, 64 3×3, 64 1×1, 256] ×3	[1×1, 128 3×3, 128, C=32 1×1, 256] ×3
conv3	28×28	[1×1, 128 3×3, 128 1×1, 512] ×4	[1×1, 256 3×3, 256, C=32 1×1, 512] ×4
conv4	14×14	[1×1, 256 3×3, 256 1×1, 1024] ×6	[1×1, 512 3×3, 512, C=32 1×1, 1024] ×6
conv5	7×7	[1×1, 512 3×3, 512 1×1, 2048] ×3	[1×1, 1024 3×3, 1024, C=32 1×1, 2048] ×3
	1×1	global average pool 1000-d fc, softmax	global average pool 1000-d fc, softmax
# params.		25.5×10^6	25.0×10^6
FLOPs		4.1×10^9	4.2×10^9

Table: A number of parameters are similar. The residual block is showed in the brackets while outside numbers represent a number of stacked blocks.

ResNeXt-50

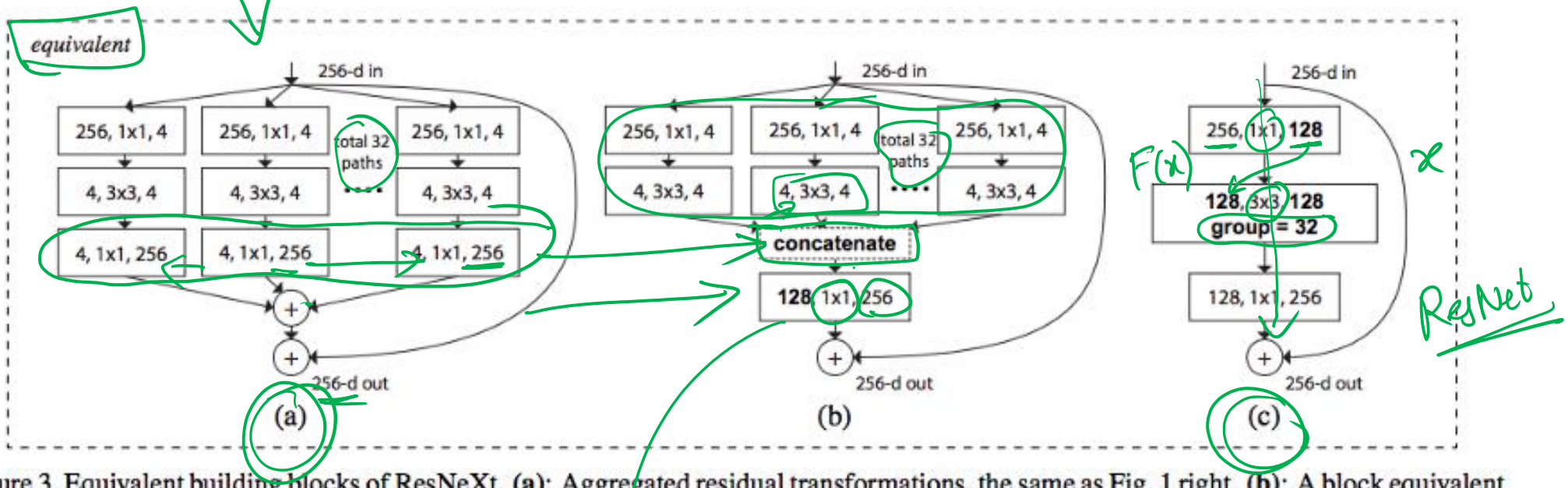


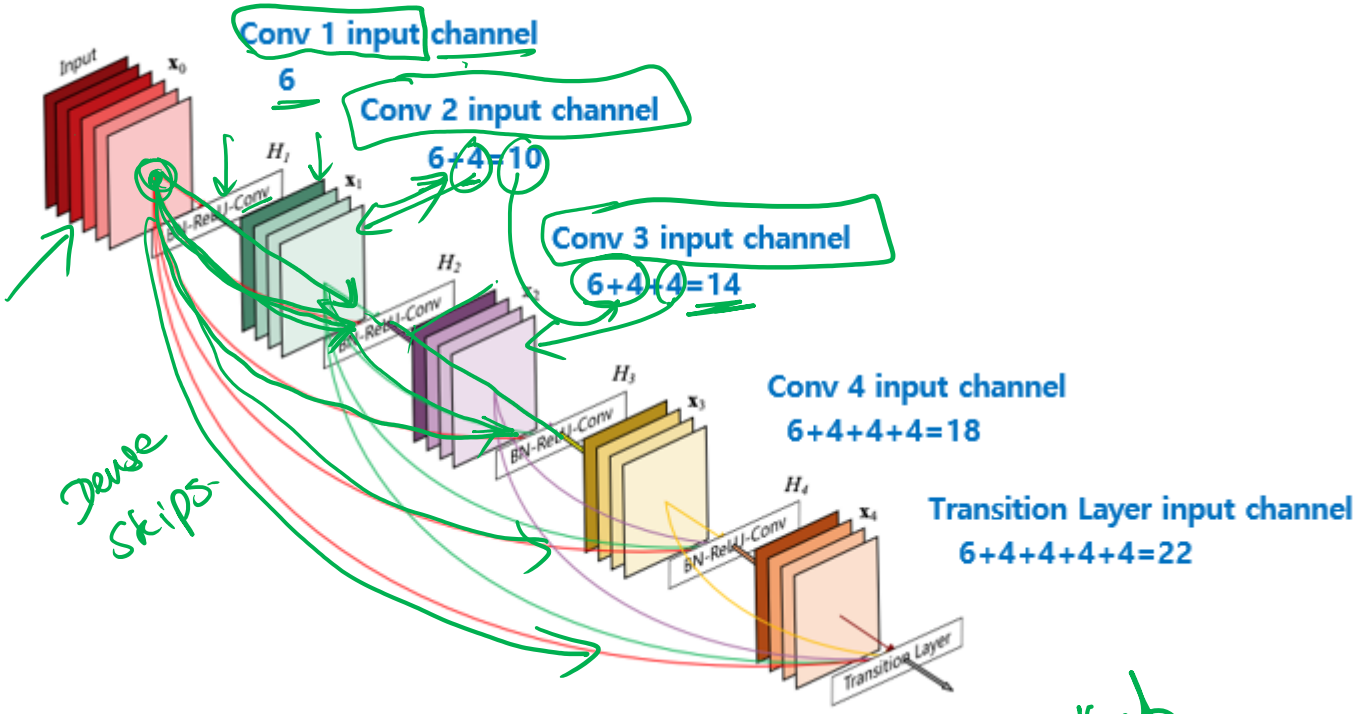
Figure 3. Equivalent building blocks of ResNeXt. **(a)**: Aggregated residual transformations, the same as Fig. 1 right. **(b)**: A block equivalent to (a), implemented as early concatenation. **(c)**: A block equivalent to (a,b), implemented as grouped convolutions [24]. Notations in **bold** text highlight the reformulation changes. A layer is denoted as (# input channels, filter size, # output channels).

DenseNet

- Dense net is densely connected-convolutional networks. It is very similar to a ResNet with some-fundamental differences. ResNet is using an additive method that means they take a previous output as an input for a future layer, & in DenseNet takes all previous output as an input for a future layer

- Skip-connection was specially developed to improve accuracy caused by the vanishing gradient in deep neural networks due to the long distance between input and output layers & the information vanishes before reaching its destination.

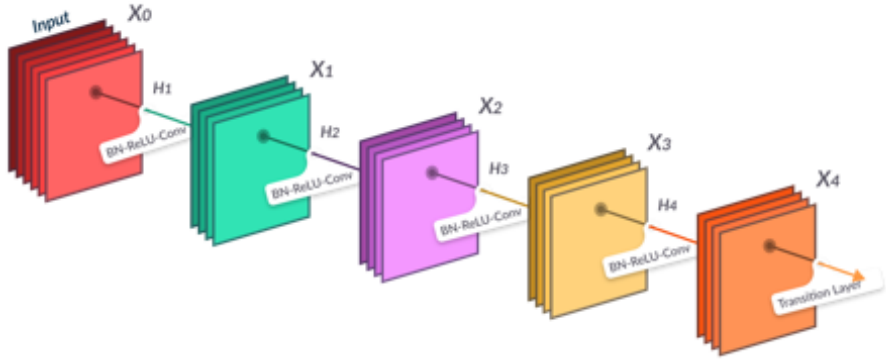
DenseNet



DenseNet Structure

"concatenation" of "all" previous features.

$$a^{[l]} = g\left([a^{[0]}, a^{[1]}, a^{[2]}, \ldots\ldots\ldots, a^{[l-1]}\right)$$



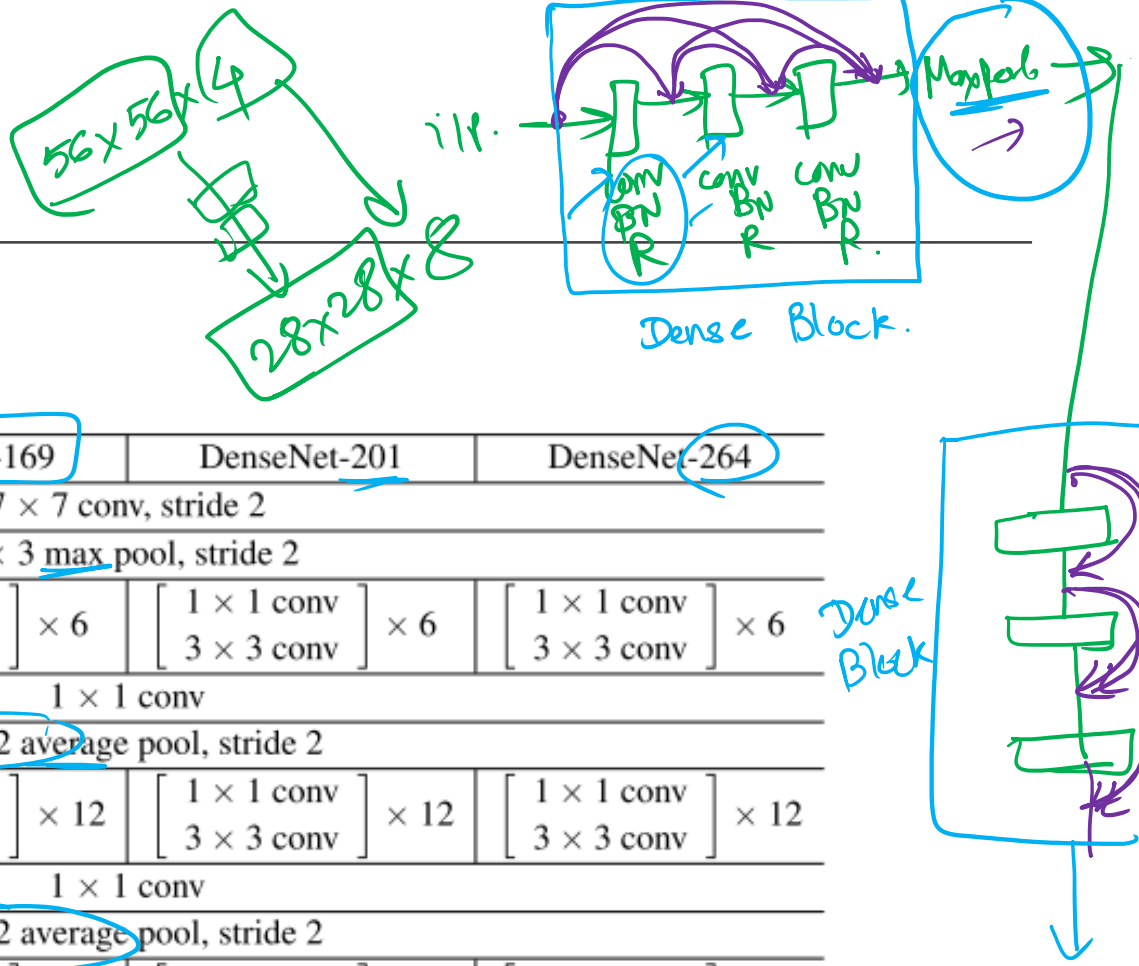
ResNet Structure

$$a^{[l]} = g\left(z^{[l+1]} + a^{[l]}\right)$$

DenseNet

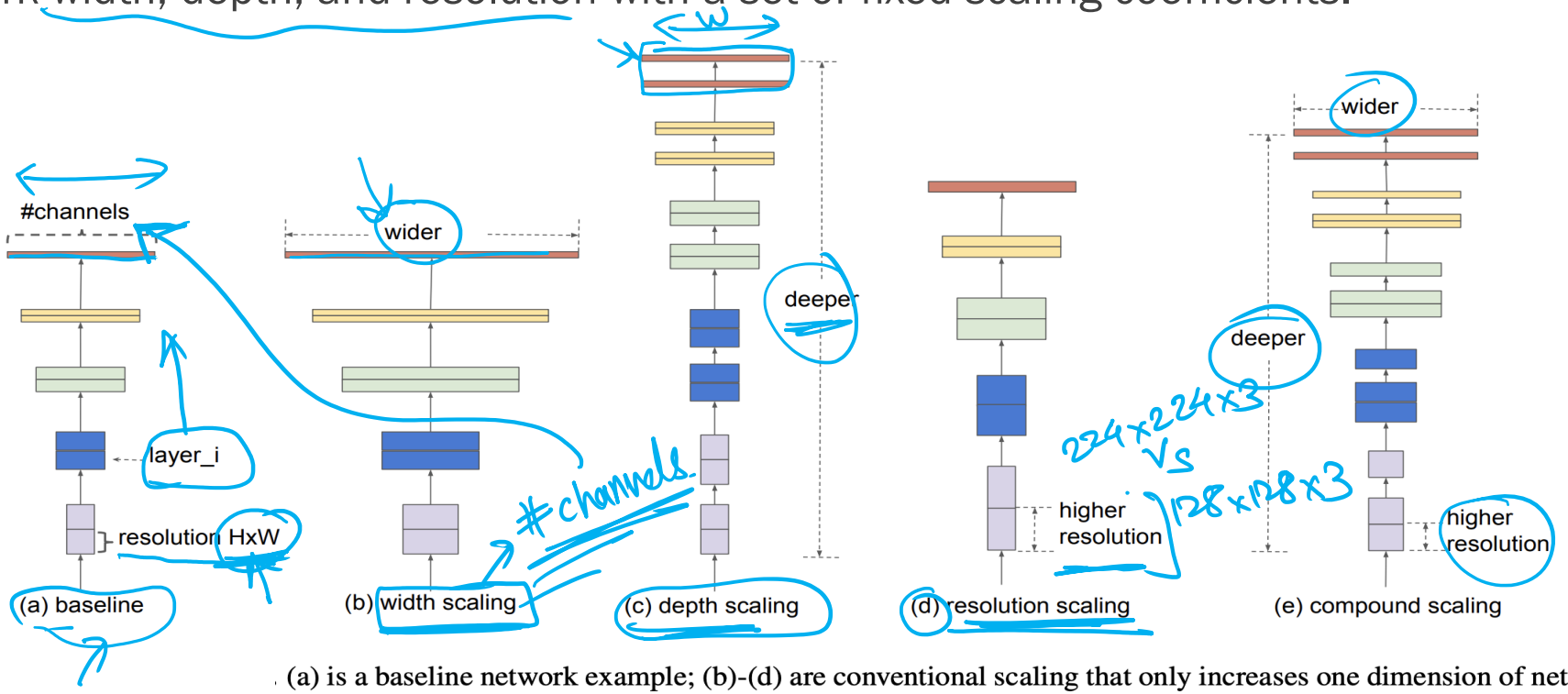
- DenseNet Architecture as a collection of DenseBlocks

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	[1 × 1 conv, 3 × 3 conv] × 6	[1 × 1 conv, 3 × 3 conv] × 6	[1 × 1 conv, 3 × 3 conv] × 6	[1 × 1 conv, 3 × 3 conv] × 6
Transition Layer (1)	56 × 56	1 × 1 conv			
	28 × 28	2 × 2 average pool, stride 2			
Dense Block (2)	28 × 28	[1 × 1 conv, 3 × 3 conv] × 12	[1 × 1 conv, 3 × 3 conv] × 12	[1 × 1 conv, 3 × 3 conv] × 12	[1 × 1 conv, 3 × 3 conv] × 12
Transition Layer (2)	28 × 28	1 × 1 conv			
	14 × 14	2 × 2 average pool, stride 2			
Dense Block (3)	14 × 14	[1 × 1 conv, 3 × 3 conv] × 24	[1 × 1 conv, 3 × 3 conv] × 32	[1 × 1 conv, 3 × 3 conv] × 48	[1 × 1 conv, 3 × 3 conv] × 64
Transition Layer (3)	14 × 14	1 × 1 conv			
	7 × 7	2 × 2 average pool, stride 2			
Dense Block (4)	7 × 7	[1 × 1 conv, 3 × 3 conv] × 16	[1 × 1 conv, 3 × 3 conv] × 32	[1 × 1 conv, 3 × 3 conv] × 32	[1 × 1 conv, 3 × 3 conv] × 48
Classification Layer	1 × 1	7 × 7 global average pool			
		1000D fully-connected, softmax			



EfficientNet

- Prediction EfficientNet is a convolutional neural network architecture and scaling method that uniformly scales all dimensions of depth/width/resolution using a compound coefficient. Unlike conventional practice that arbitrary scales these factors, the EfficientNet scaling method uniformly scales network width, depth, and resolution with a set of fixed scaling coefficients.

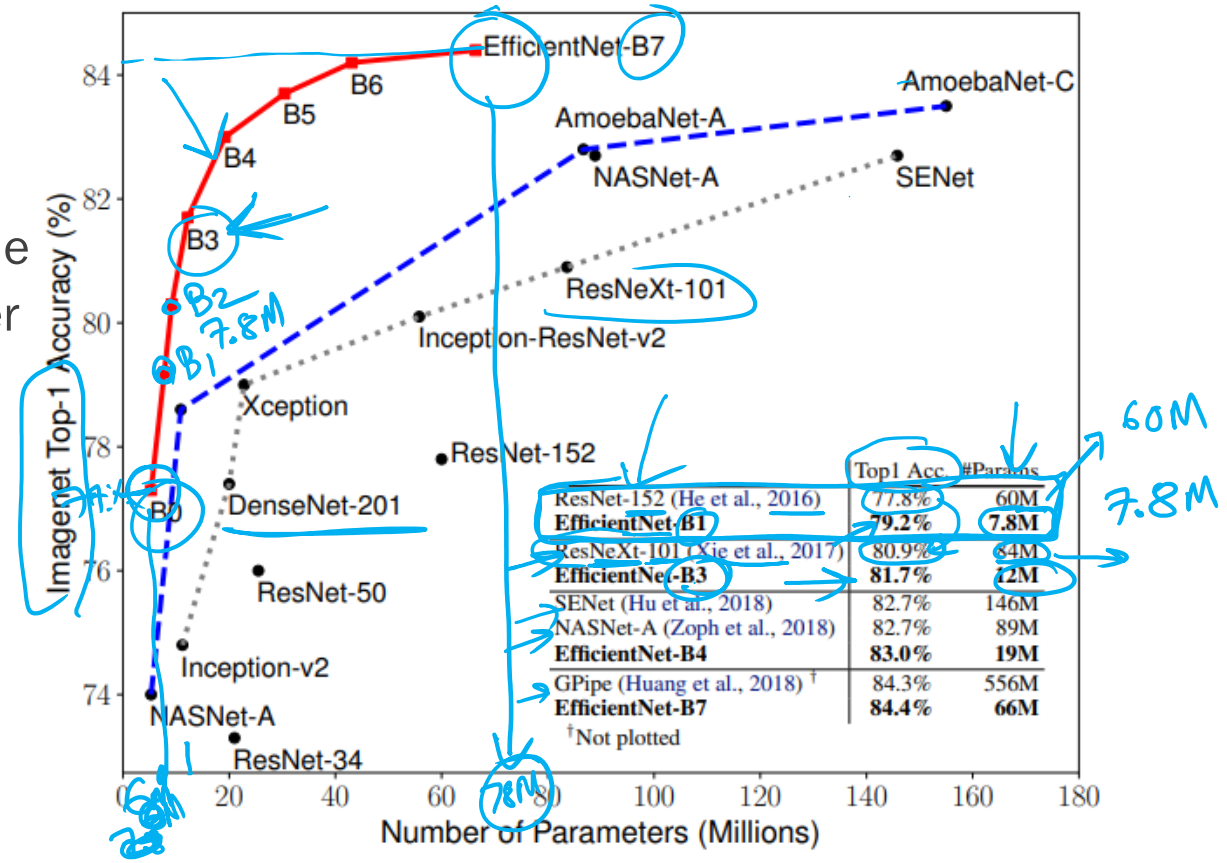


(a) is a baseline network example; (b)-(d) are conventional scaling that only increases one dimension of network width, depth, or resolution. (e) is our proposed compound scaling method that uniformly scales all three dimensions with a fixed ratio.

EfficientNet

- We have compared our EfficientNets with other existing CNNs on ImageNet. In general, the EfficientNet models achieve both higher accuracy and better efficiency over existing CNNs,

- With considerably fewer numbers of parameters, the family of models are efficient and also provide better results.



sandhan@iitk.ac.in

Thank you

sandhan@iitk.ac.in