Available online at **www.joac.info**

ISSN: 2278-1862

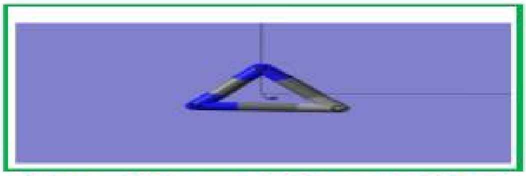
Journal of Applicable Chemistry

2023, 12 (3): 265-309

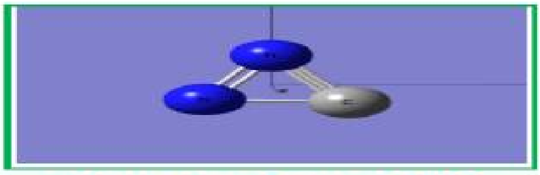
(International Peer Reviewed Journal)

New Chemistry News

$^-N=C=N^-$



New News of Chem (NNC)



ChemNewsNew (CNN)

CNN – 50
Convolution Neural Nets(ConvNN)
Part 1.Pretrained Nets

Information Source	**ACS.org ; sciencedirect.com**
K. Somasekhara Rao, Dept. of Chemistry, Acharya Nagarjuna Univ., Dr. M.R.Appa Rao Campus, Nuzvid-521 201, India	R. Sambasiva Rao, Dept. of Chemistry, Andhra University, Visakhapatnam 530 003, India

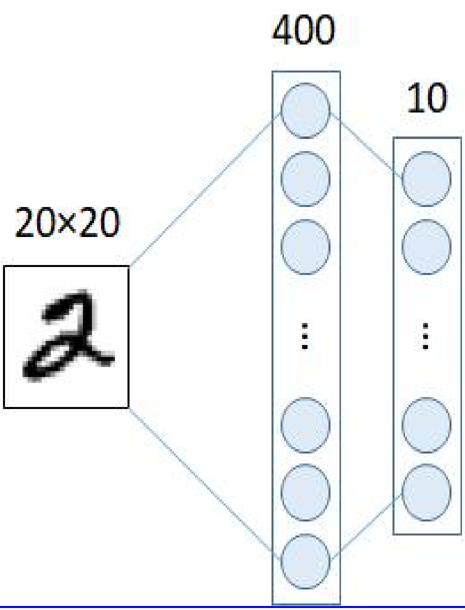
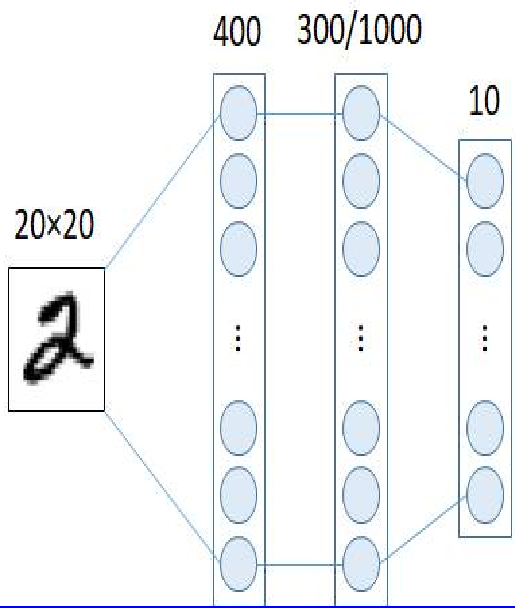
Conspectus: Neuron is processing unit accepting a scalar input and outputting a scalar. The transformation of input is affected by a transfer function (ranging from identity to tanh, fuzzy and so on). A number of neurons are structured in layers called one input layer, one output layer and single/many hidden layers. The neurons in any layer are not interconnected. The data moves in the forward direction through sequentially connected input-hidden-output layers. These of architectures are popular as feed-forward (FF)-sequential (seq)-single (multiple) (S/M) layer perceptron (SLP/MLP) neural network (NN). When layers are connected in reverse direction also (I←[H1←H2]← O), the models are called recurrent NNs. In the classical era (1943-1985, 1986 to 1995), the number of hidden layers were restricted to two or a maximum of three. Keeping aside the cognitron (1975) and neo-cognitron (1980) by Fukushima, a new era started with convolution neural nets (CNNs).

Here, architectural details, performance, depth of net, number of refinable parameters of typical large-size CNNs for example LeNet-x, Alex Net, YOLO, VGG, ResNet, Inception, Xception, EfficientNet, MobileNet, DenseNet, ConvNeXt etc. are summarized. The genes i.e. padding, pooling, flattening and normalization express different outcomes.

Keywords: Single (hidden) layer perceptron (SLP) neural net(NN); M(ulti)LPNN-Convolution neural net(ConvNN)-Images of W*H-pixels; grey/RBG; AlexNet; VGG;ResNet

	Layout	
I	Sequential Layered (seqL) Feed forward (FF) Fully Connected (FulCon) NeuronNets (NNs)	
II	Math-DNA- Components of $$$-CNN	K(nowledge)Lab rsr.chem1979
III	Architectures and Performance measures Fortypical $$$-ConvNNs	
IV	Anatomy and characteristics of typical $$$-CNN	
V	xAiProbes for ConvNN, CapsNN	

I. Sequential Layered (seqL) Feed forward (FF) Fully Connected (FulCon) NeuronNets (NNs)

MLP	
No (Zero hidden) Layer Image Classifier IC (zhLIC)	**Single (One hidden) Layer Image Classifier (SoHLIC)**
	
Multiple (Two hidden) Layer Image Classifier (MLIC)	

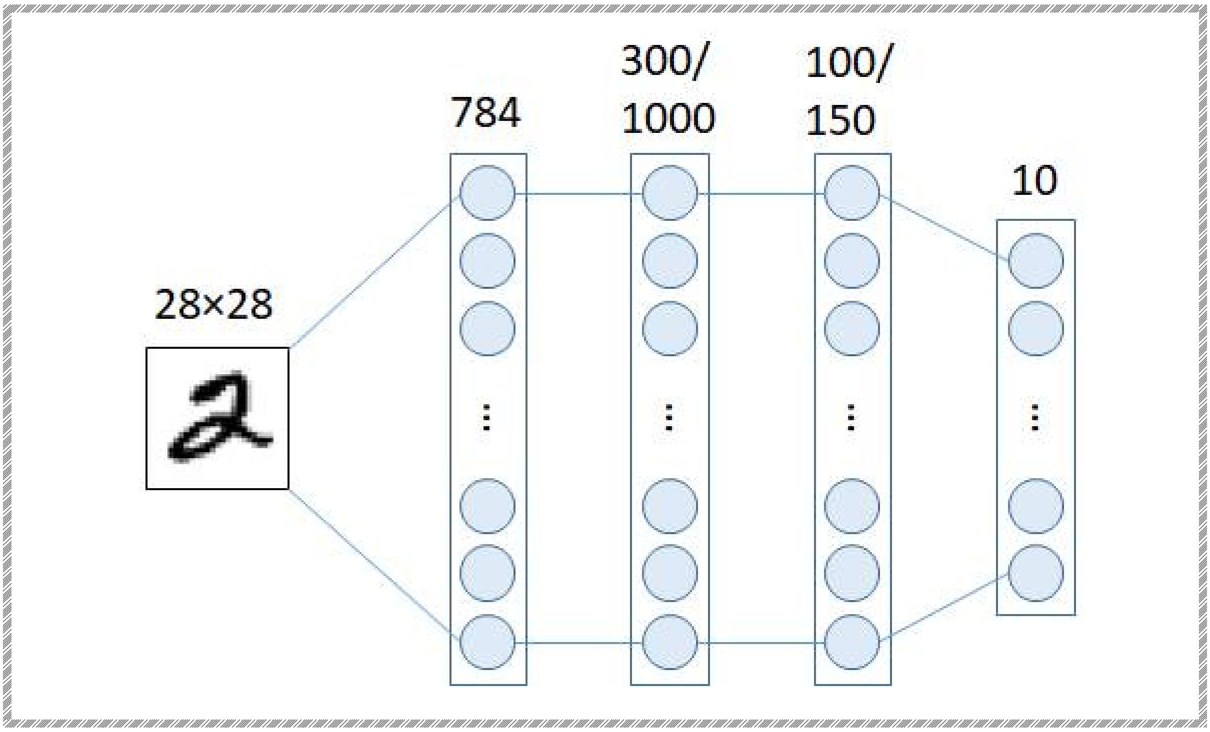
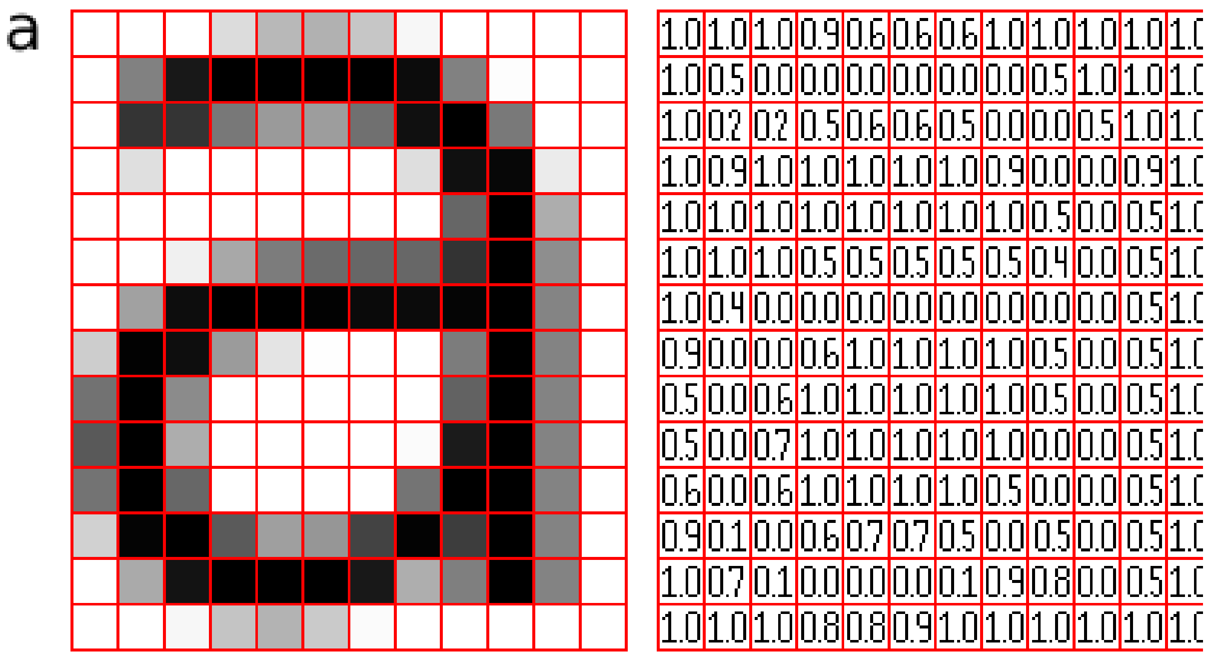
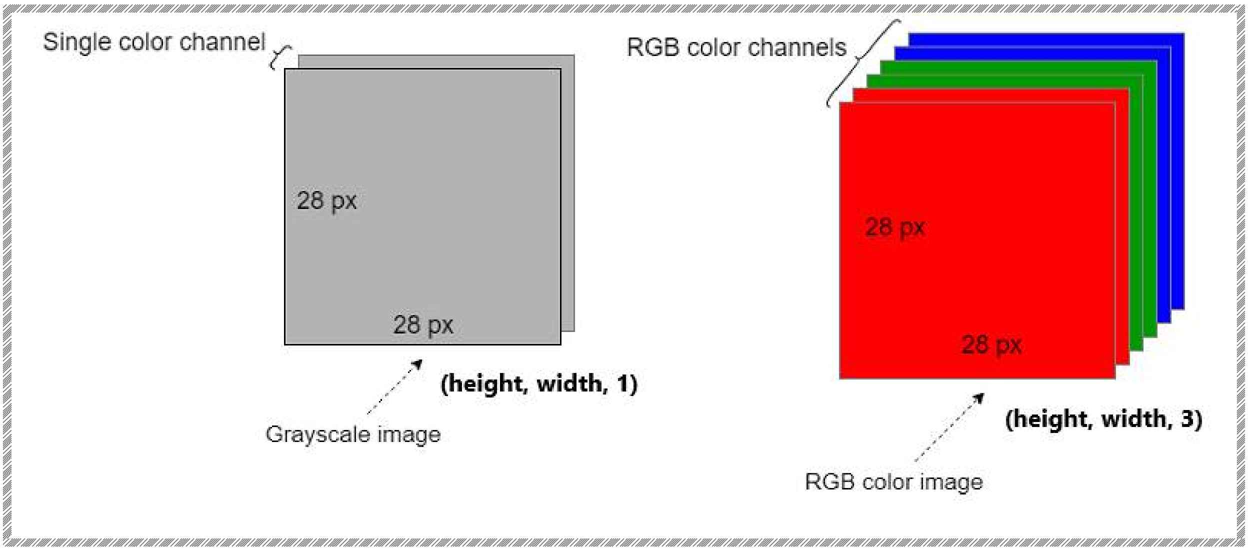


Image as a 2D-grid of pixels



Grayscale vs RGB image representation



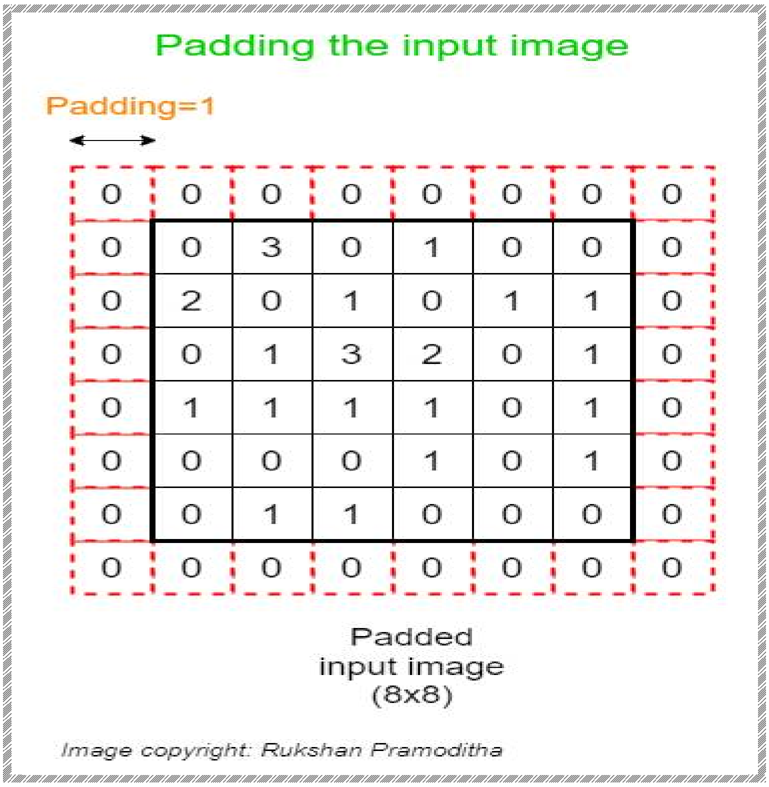
ImageNet

- Dataset of over 15 millions labeled High-resolution images
- Around 22,000 categories

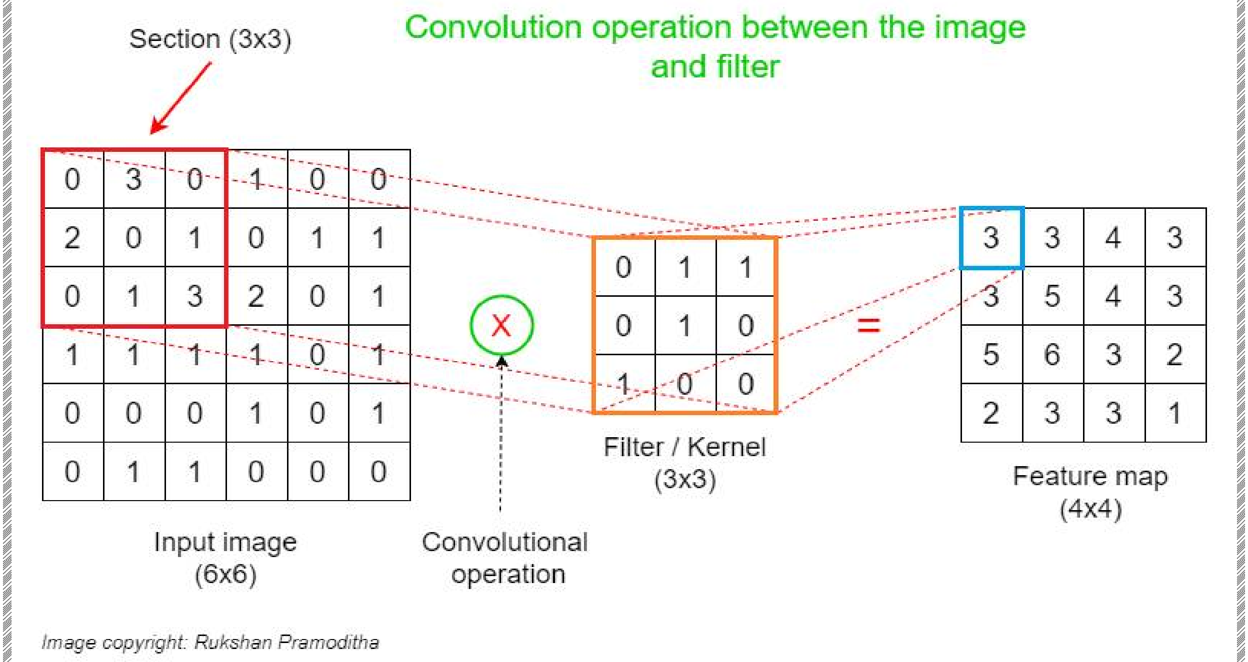
ILSVRC uses a subset of ImageNet

- Roughly 1.3 million training images
- 1000 images in each of 1000 categories
- 50,000 validation images
- 100,000 testing images

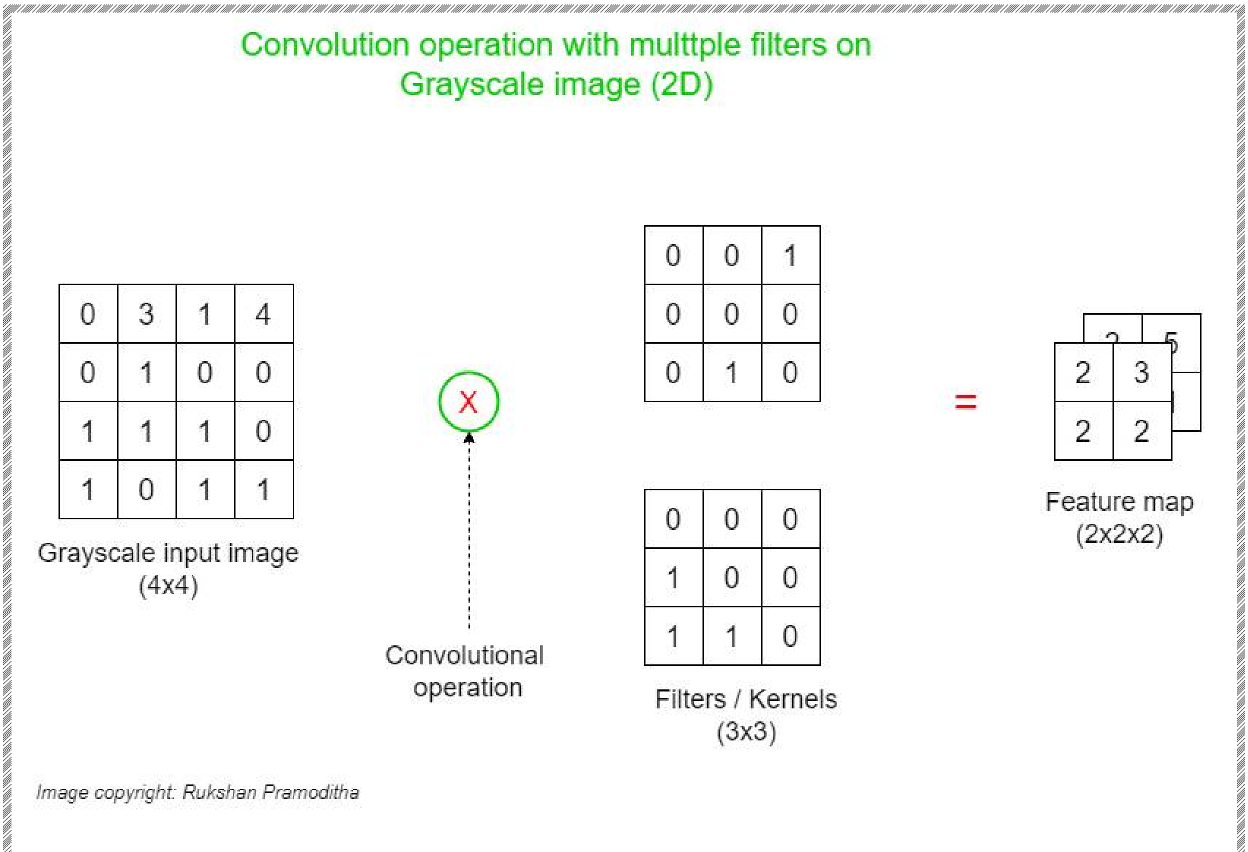
II. Math-DNA- Components of $$$-CNN



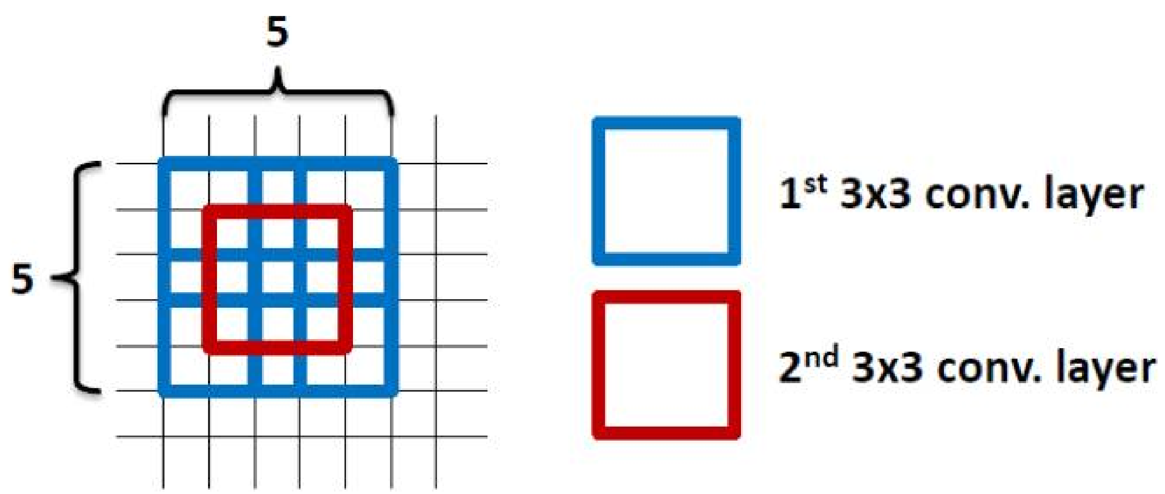
Operation (conv)

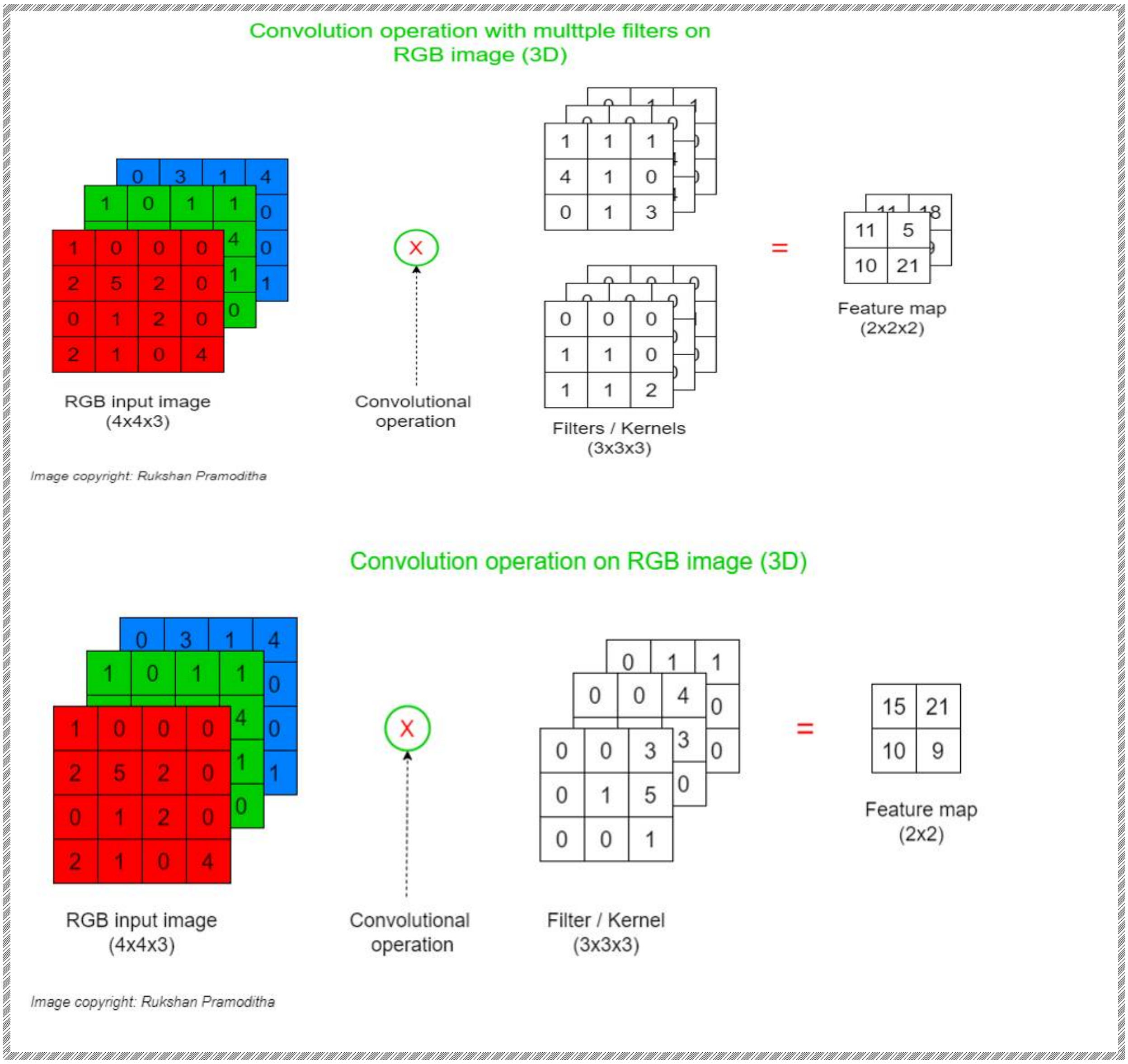


(Courtesy of author)

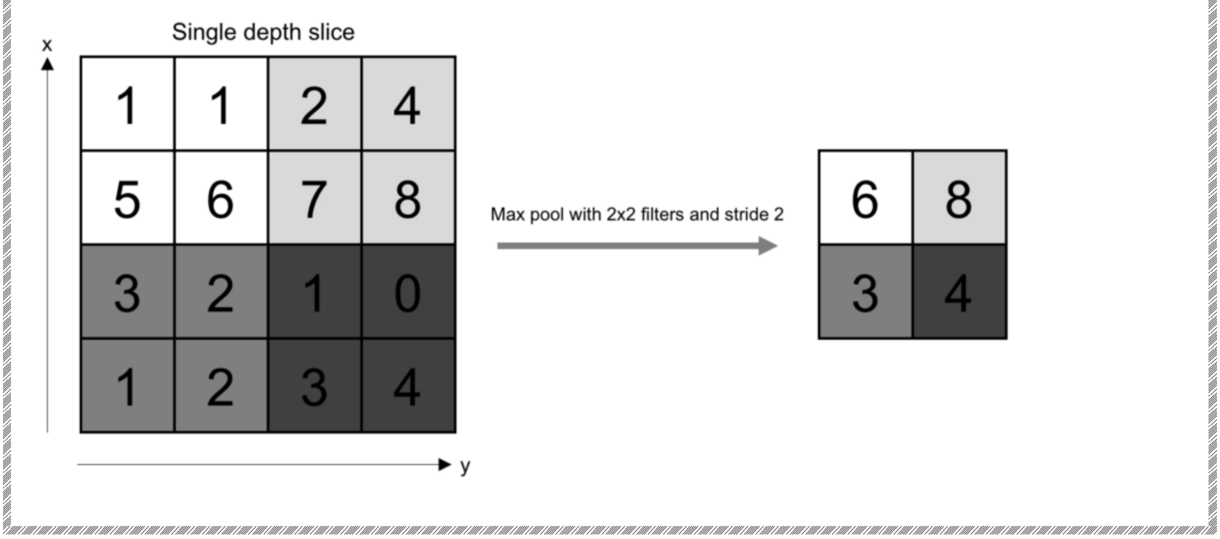


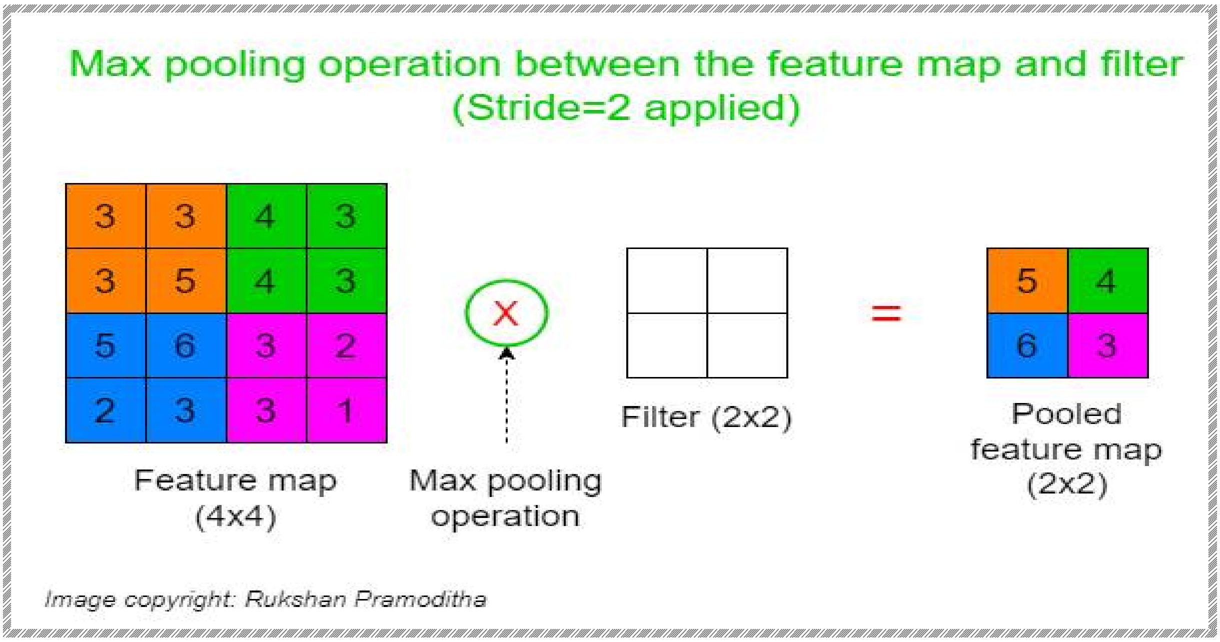
2 layers of 3×3 filters already covered the 5×5 area



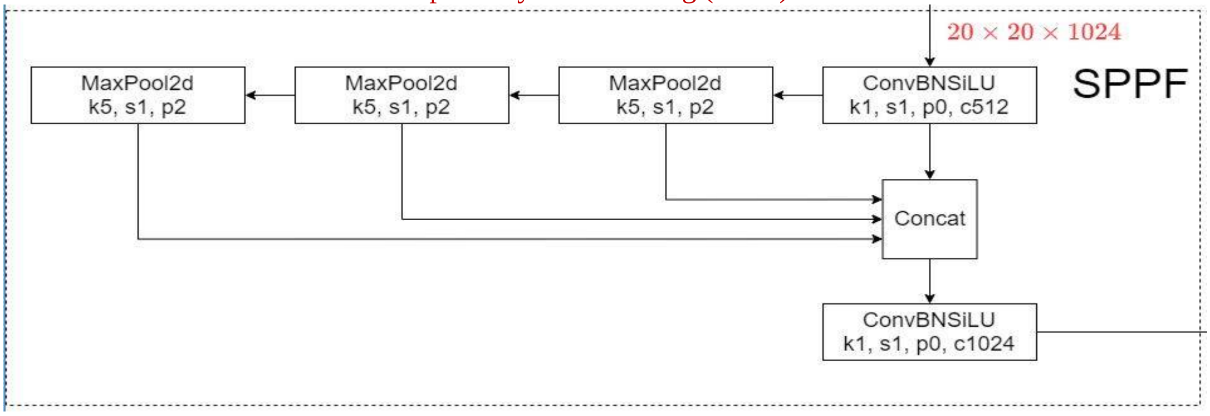


Operation (Pooling Maximum)

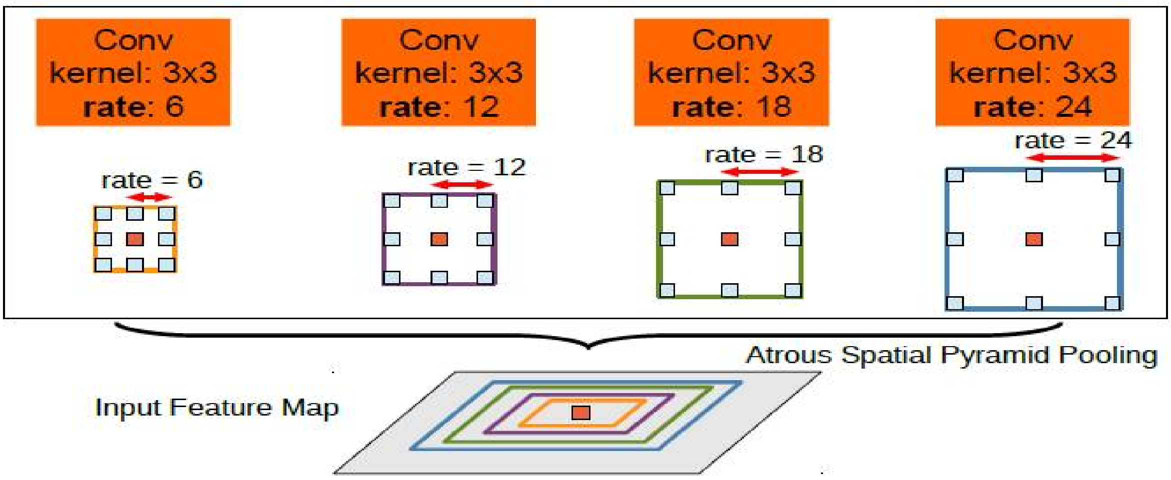


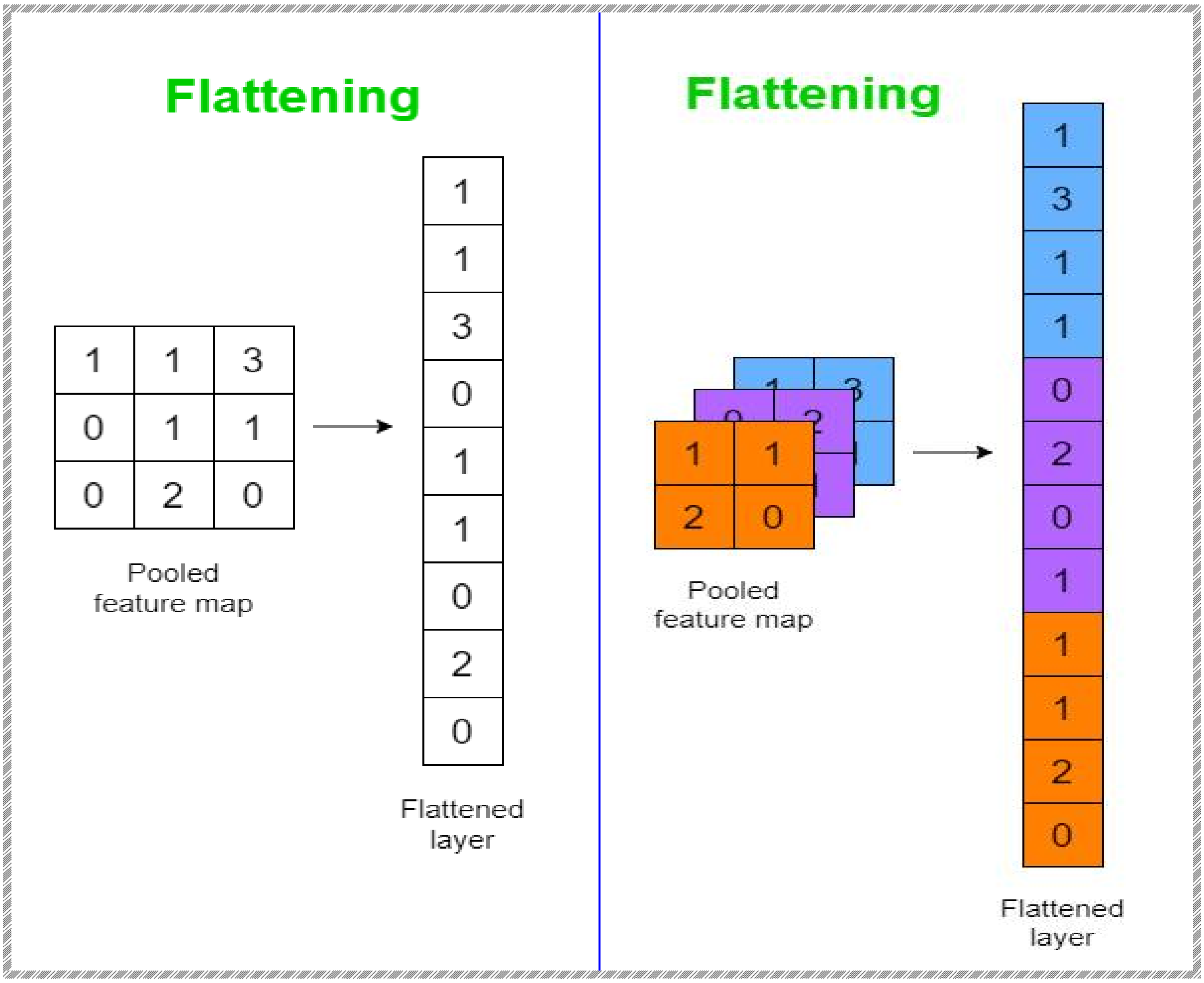


Spatial Pyramid Pooling (SPPF)

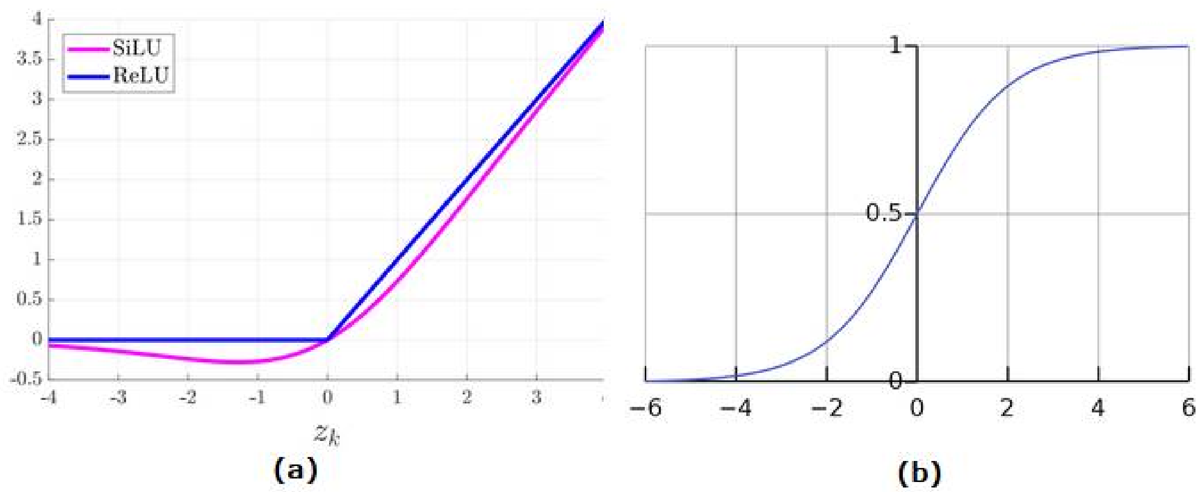


Atrous Spatial Pyramid Pooling (AtrosSpaPyramPool)



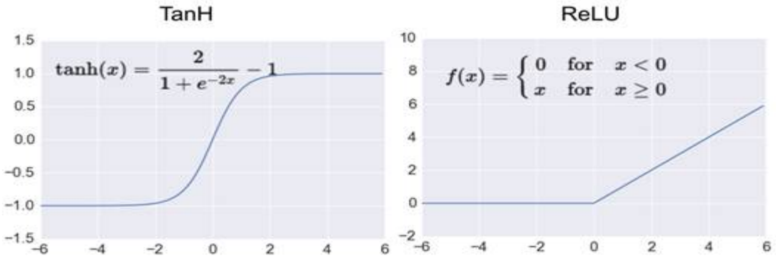


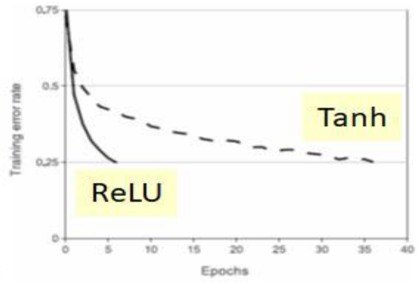
Activation functions (a) SiLU (b) Sigmoid



Sigmoid Linear Unit (or **swish** activation function)

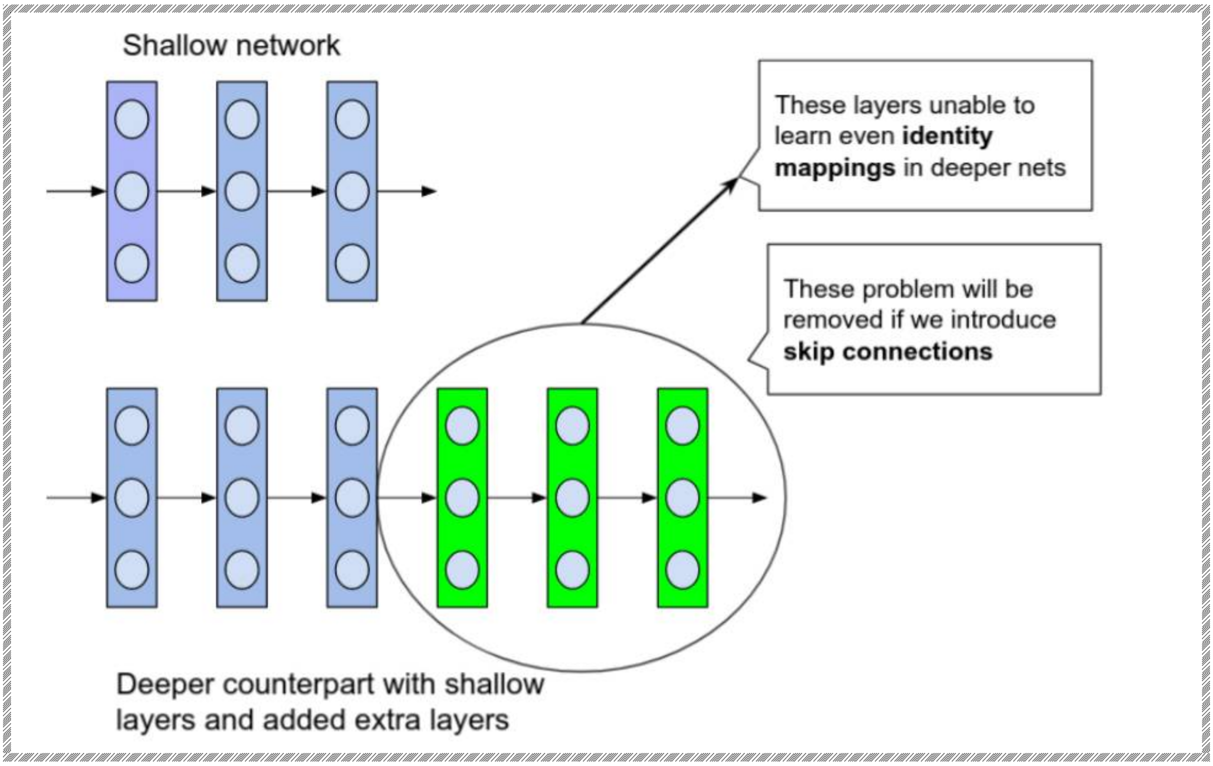
ReLU



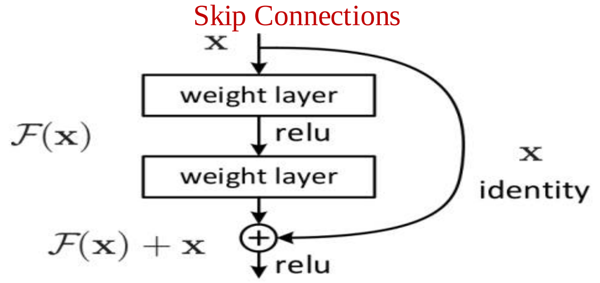


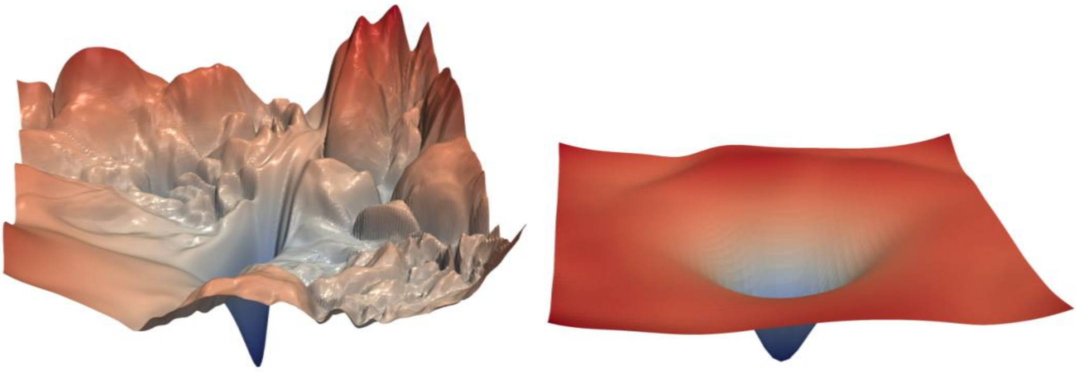
Activation function

- ReLU is introduced in AlexNet
- Before Alexnet, Tanh was popular TF
- ReLU is six times faster than Tanh

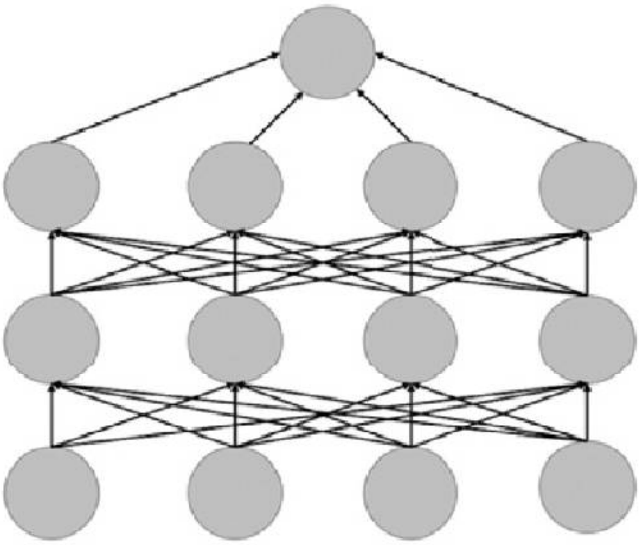


Skip Connections

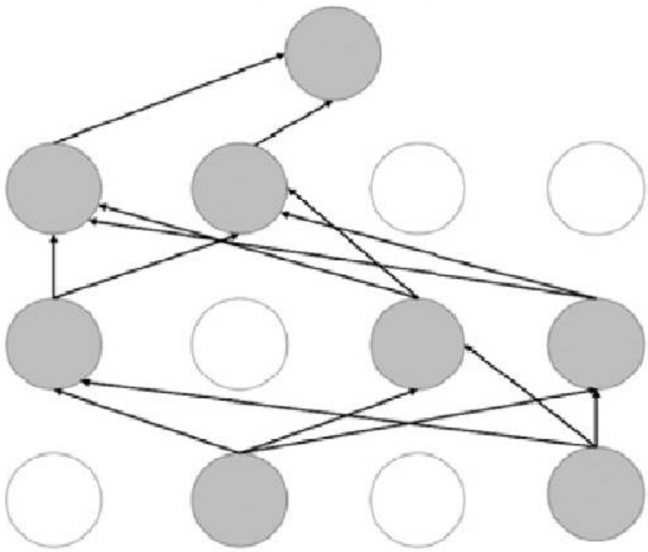




(a) without skip connections (b) with skip connections

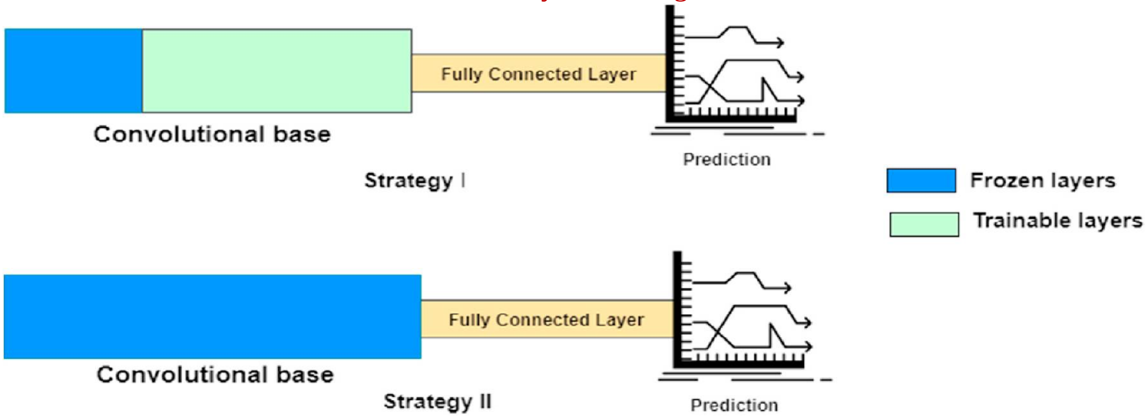


Standard
Fully connectedNet



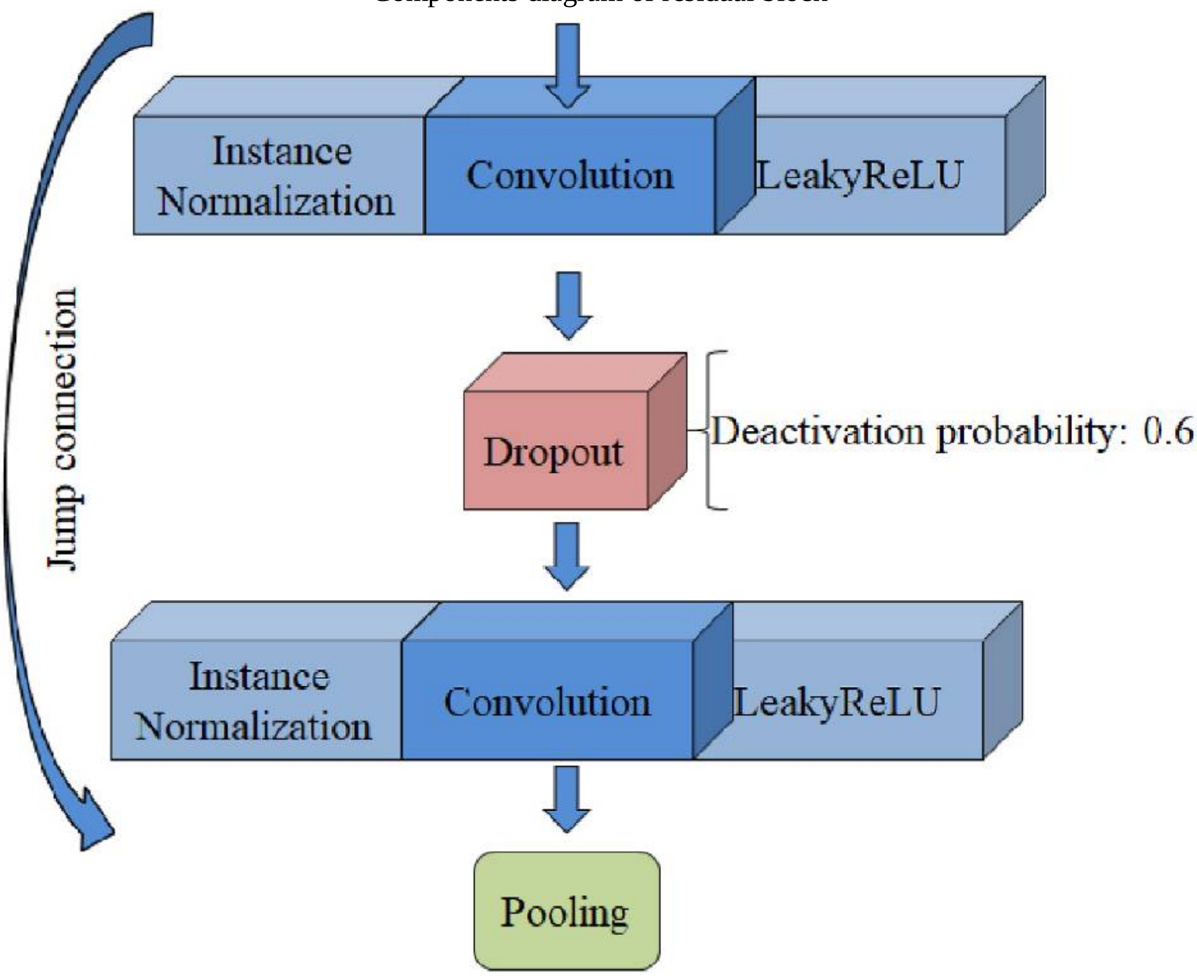
Drop out Net

- Layer Freezing



- Concept of Layer Freezing
- Strategy I is followed when the dataset is dissimilar to the dataset in the pre-trained model
- Strategy II is followed when the dataset is similar to the dataset in the pre-trained model

Components diagram of residual block

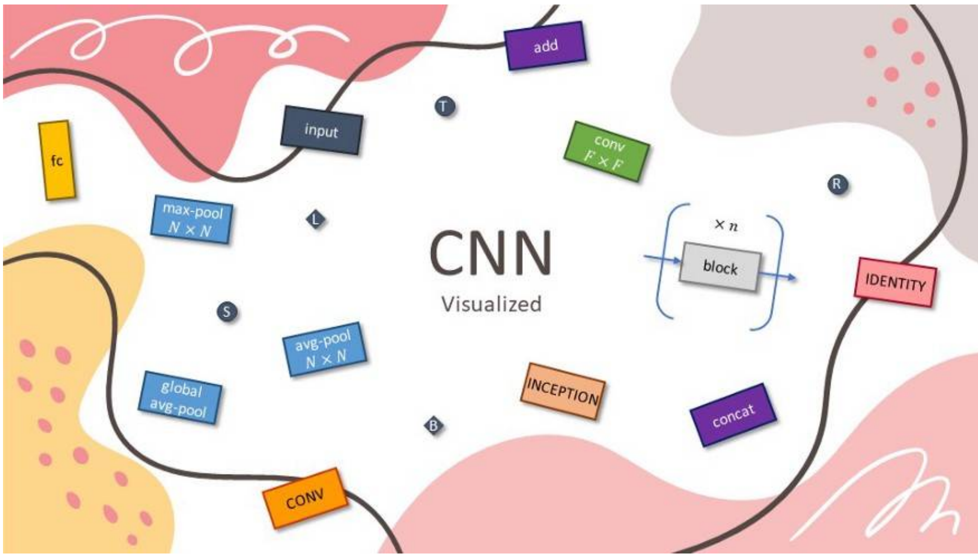


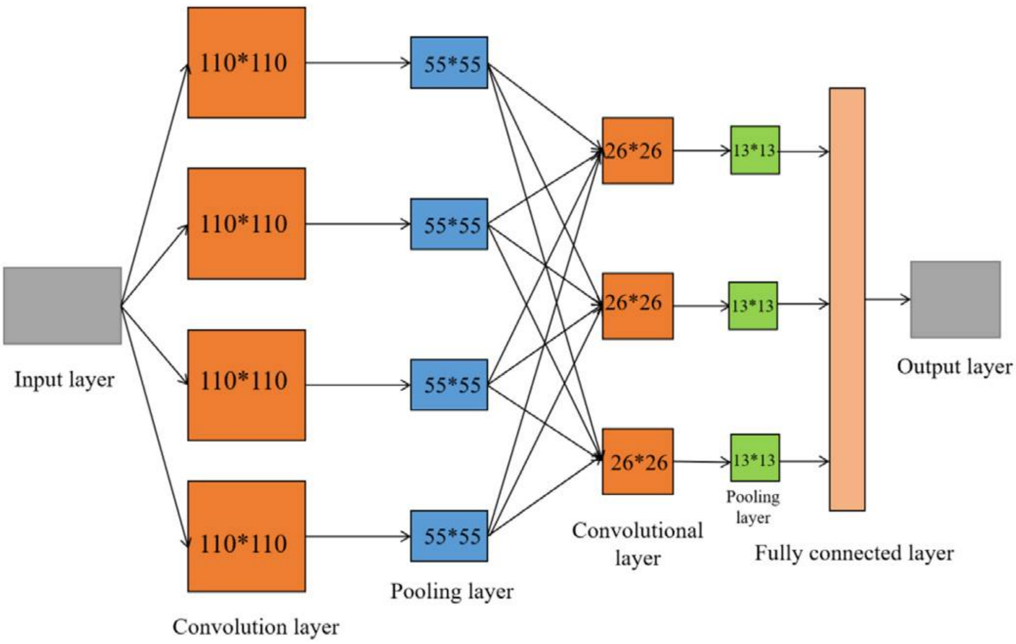
III. Architectures andPerformance measures

for typical $$$-ConvNNs

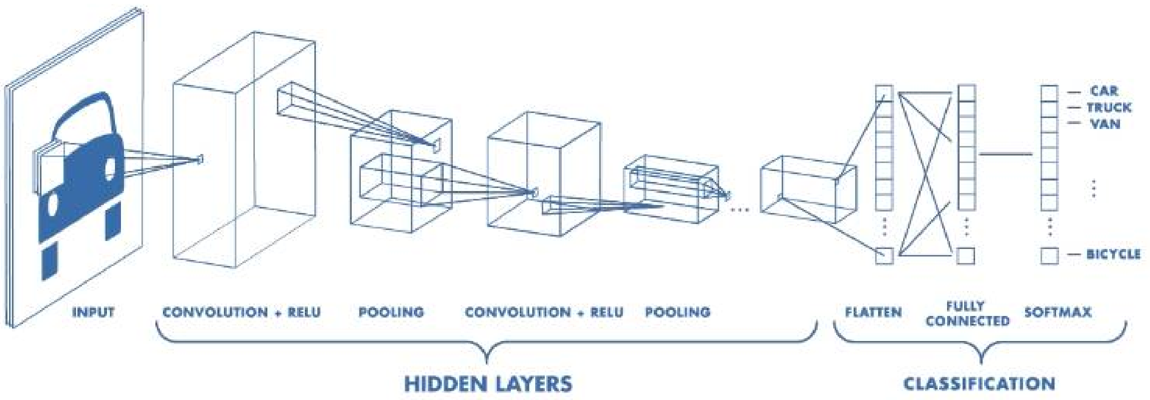
Named CNN	version	Depth	Parameters	Size (MB)	Top-1 Accuracy	Top-5 Accuracy
Xception	Xception	81	22.9M	88	79.0%	94.5%
VGG	VGG16	16	138.4M	528	71.3%	90.1%
	VGG19	19	143.7M	549	71.3%	90.0%
ResNet	ResNet50	107	25.6M	98	74.9%	92.1%
	ResNet50V2	103	25.6M	98	76.0%	93.0%
	ResNet101	209	44.7M	171	76.4%	92.8%
	ResNet101V2	205	44.7M	171	77.2%	93.8%
	ResNet152	311	60.4M	232	76.6%	93.1%
	ResNet152V2	307	60.4M	232	78.0%	94.2%
Inception	InceptionV3	189	23.9M	92	77.9%	93.7%
	InceptionResNetV2	449	55.9M	215	80.3%	95.3%
MobileNet	MobileNet	55	4.3M	16	70.4%	89.5%
	MobileNetV2	105	3.5M	14	71.3%	90.1%
DenseNet	DenseNet121	242	8.1M	33	75.0%	92.3%
	DenseNet169	338	14.3M	57	76.2%	93.2%
	DenseNet201	402	20.2M	80	77.3%	93.6%
NASNe	NASNetMobile	389	5.3M	23	74.4%	91.9%
	NASNetLarge	533	88.9M	343	82.5%	96.0%
EfficientNet	EfficientNetB0	132	5.3M	29	77.1%	93.3%
	EfficientNetB1	186	7.9M	31	79.1%	94.4%
	EfficientNetB2	186	9.2M	36	80.1%	94.9%
	EfficientNetB3	210	12.3M	48	81.6%	95.7%
	EfficientNetB4	258	19.5M	75	82.9%	96.4%
	EfficientNetB5	312	30.6M	118	83.6%	96.7%
	EfficientNetB6	360	43.3M	166	84.0%	96.8%
	EfficientNetB7	438	66.7M	256	84.3%	97.0%
	EfficientNetV2B0	-	7.2M	29	78.7%	94.3%
	EfficientNetV2B1	-	8.2M	34	79.8%	95.0%
	EfficientNetV2B2	-	10.2M	42	80.5%	95.1%
	EfficientNetV2B3	-	14.5M	59	82.0%	95.8%
	EfficientNetV2S	-	21.6M	88	83.9%	96.7%
	EfficientNetV2M	-	54.4M	220	85.3%	97.4%
	EfficientNetV2L	-	119.0M	479	85.7%	97.5%
ConvNeXt	ConvNeXtTiny	-	28.6M	109.42	81.3%	-
	ConvNeXtSmall	-	50.2M	192.29	82.3%	-
	ConvNeXtBase	-	88.5M	338.58	85.3%	-
	ConvNeXtLarge	-	197.7M	755.07	86.3%	-
	ConvNeXtXLarge	-	350.1M	1310	86.7%	-

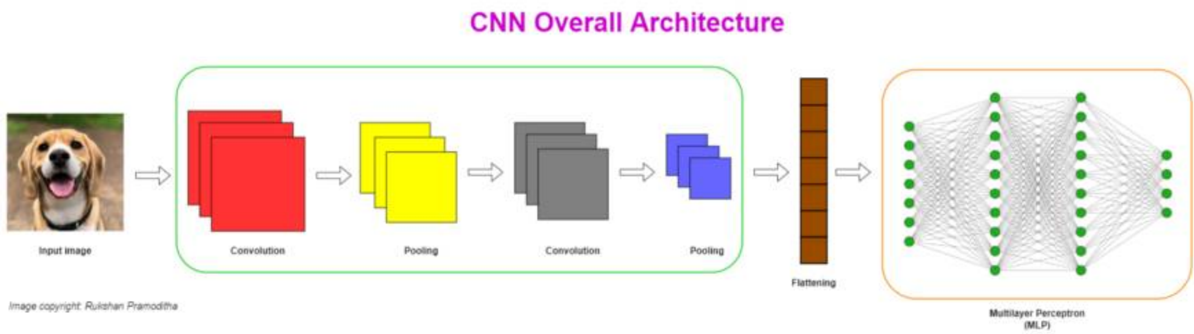
IV. Anatomy and characteristics of typical $$$-CNN





Architecture of a CNN (sequential)



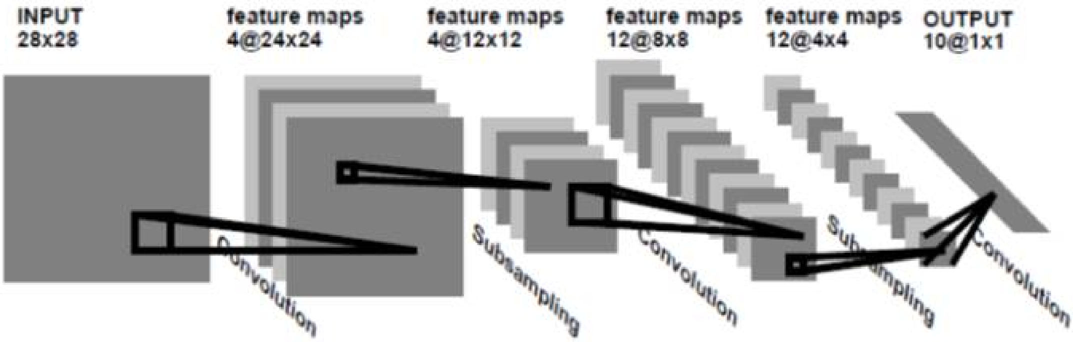


CNN Architecture	Default Input	Default Output	Number of Layers	Number of Parameters	Activation Function	New Additional Perks
LeNet-5	32x32x1	10	5	60K	tanh	Convolution Layer
AlexNet	224x224x3	1000	8	60M	ReLU	Local Response Normalization
VGG-16	224x224x3	1000	16	138M	ReLU	Very deep but still single thread
Inception-v1	224x224x3	1000	22	7M	ReLU	Auxiliary Classifiers & Inception Module
ResNet-50	224x224x3	1000	50	26M	ReLU	Batch Normalization & Residual Blocks

Summary on Error Rate

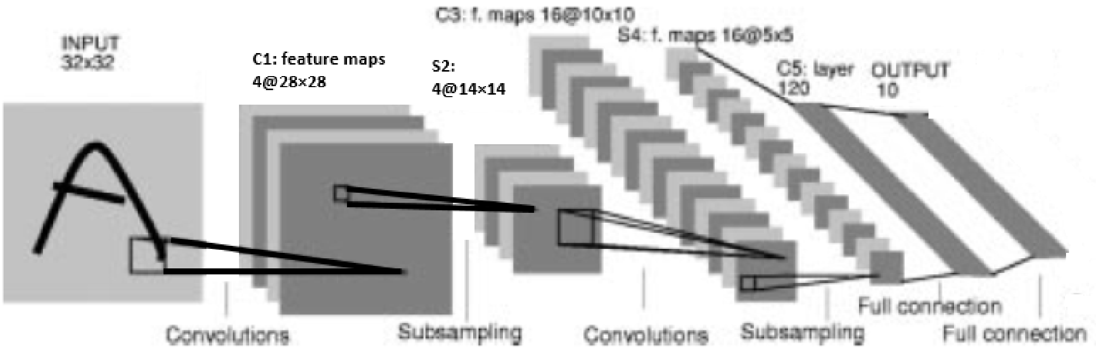
1. Baseline Linear Classifier: 8.4%
2. One-Hidden-Layer Fully Connected Multilayer NN: 3.6% to 3.8%
3. Two-Hidden-Layer Fully Connected Multilayer NN: 2.95% to 3.05%
4. LeNet-1: 1.7%
5. LeNet-4: 1.1%
6. LeNet-5: 0.95%
7. Boosted LeNet-4: 0.7%

LeNet-1



- 28×28 input image >
  Four 24×24 feature maps convolutional layer (5×5 size) >
  Average Pooling layers (2×2 size) >
  Eight 12×12 feature maps convolutional layer (5×5 size) >
  Average Pooling layers (2×2 size) >
  Directly fully connected to the output

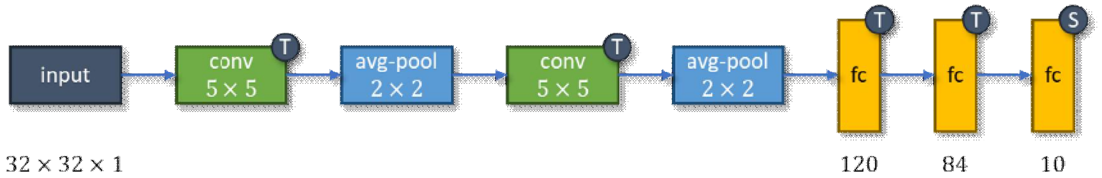
LeNet-4



Lenet-4

- 32×32 input image >
- Four 28×28 feature maps convolutional layer (5×5 size) >
- Average Pooling layers (2×2 size) >
- Sixteen 10×10 feature maps convolutional layer (5×5 size) >
- Average Pooling layers (2×2 size) >
- Fully connected to 120 neurons >
- Fully connected to 10 outputs

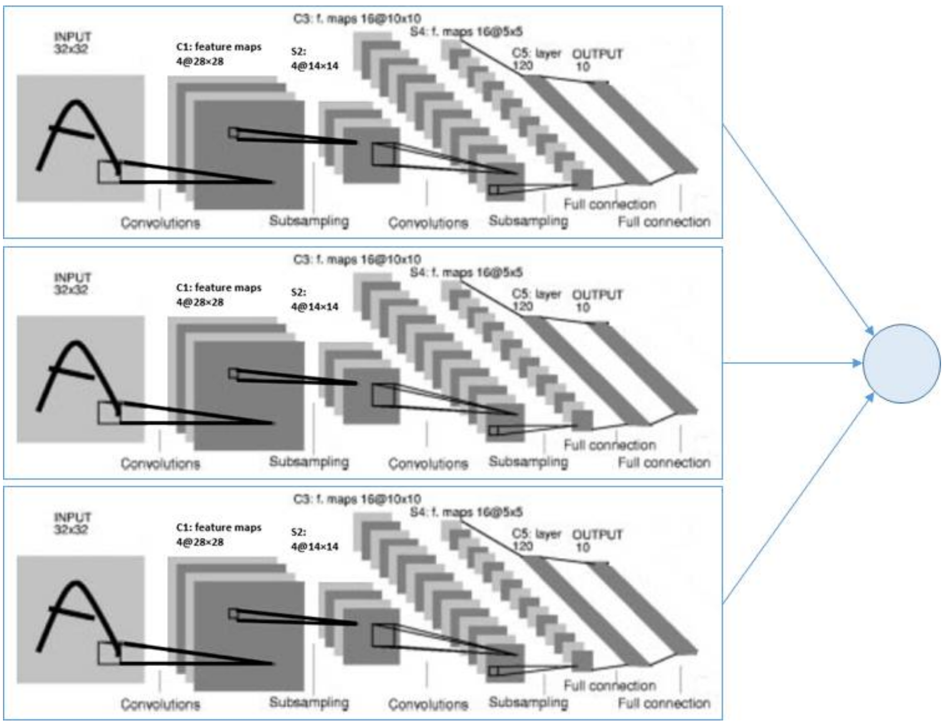
LeNet-5 (1998)



- 60,000 parameters
- Network trained on greyscale 32×32 digit images
- Tries to recognize them as one of ten digits (0 to 9)
- Each convolution layer is followed by a 2×2 average-pooling
- Every layer has tanh activation function except the last (which has softmax)
  - 2 convolution layers with kernel size 5×5, followed by
  - 3 fully connected layers

✓ 60 million parameters
✓ Winner of the ImageNet competition in 2012
✓ 1000 classes.
✓ Default: AlexNet accepts colored images with dimensions 224×224

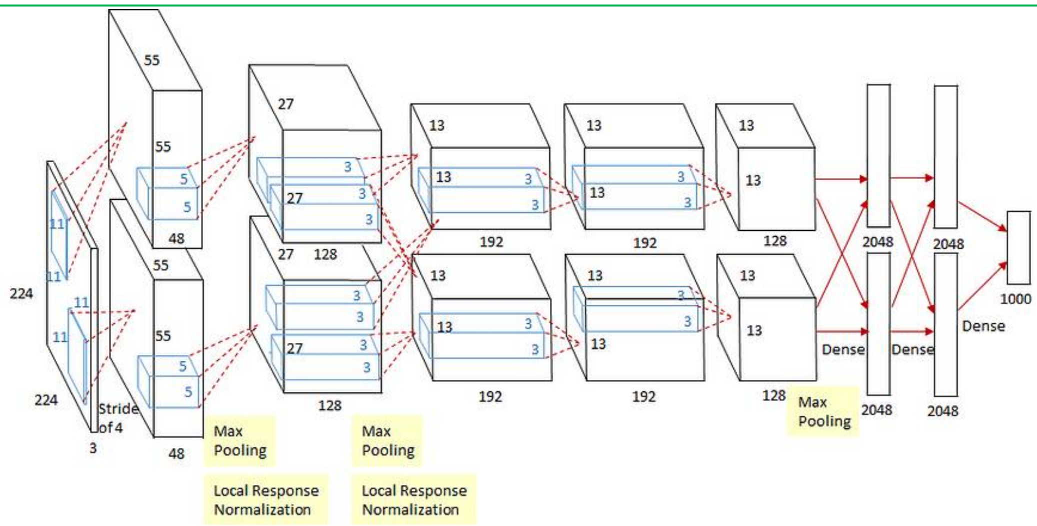
Boosted LeNet-4



+ Boosting is a technique to combine the results from several/many weak classifiers to obtain more accurate values.
- ✓ In LeNet-4, the outputs of three LeNet-4 are simply added together
- ✓ one with maximum value would be the predicted classification class
- ✓ And there is an enhancement that when the first net has a high confidence answer, the other nets would not be called.

+ With boosting, the **error rate** on test data is **0.7%** which is even smaller than that of LeNet-5.

Alex Architecture (2012)



Architecture of Alex Net in object mode

- ✓ Input: 224×224×3 ; original :

227×227×3 if padded during1st convolution

- ✓ Total parameters trained: 60 million

1st: Convolutional Layer: 2 groups of 48 kernels, size 11×11×3 (stride: 4, pad: 0)
Outputs 55×55 ×48 feature maps ×2 groups
Then **3×3 Overlapping Max Pooling (stride: 2)**
Outputs 27×27 ×48 feature maps ×2 groups
Then **Local Response Normalization**
Outputs 27×27 ×48 feature maps ×2 groups

2nd: Convolutional Layer: 2 groups of 128 kernels of size 5×5×48 (stride: 1, pad: 2)
Outputs 27×27 ×128 feature maps ×2 groups
Then **3×3 Overlapping Max Pooling (stride: 2)**
Outputs 13×13 ×128 feature maps ×2 groups
Then **Local Response Normalization**
Outputs 13×13 ×128 feature maps ×2 groups

3rd: Convolutional Layer: 2 groups of 192 kernels of size 3×3×256 (stride: 1, pad: 1)

Outputs 13×13 ×192 feature maps ×2 groups

- **4th: Convolutional Layer: 2 groups of 192 kernels of size 3×3×192 (stride: 1, pad: 1)**
  Outputs 13×13 ×192 feature maps ×2 groups

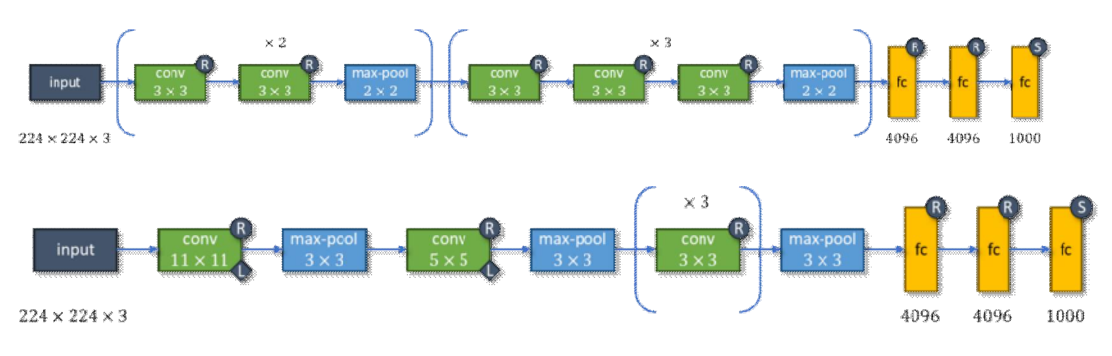
- **5th: Convolutional Layer: 256 kernels of size 3×3×192 (stride: 1, pad: 1)**
  Outputs 13×13 ×128 feature maps ×2 groups
  Then **3×3 Overlapping Max Pooling (stride: 2)**
  Outputs 6×6 ×128 feature maps ×2 groups

- **6th: Fully Connected (Dense) Layer of**
  4096 neurons

- **7th: Fully Connected (Dense) Layer of**
  4096 neurons

- **8th: Fully Connected (Dense) Layer of**
  Outputs 1000 neurons (since there are 1000 classes)
  Softmax is used for calculating the loss.

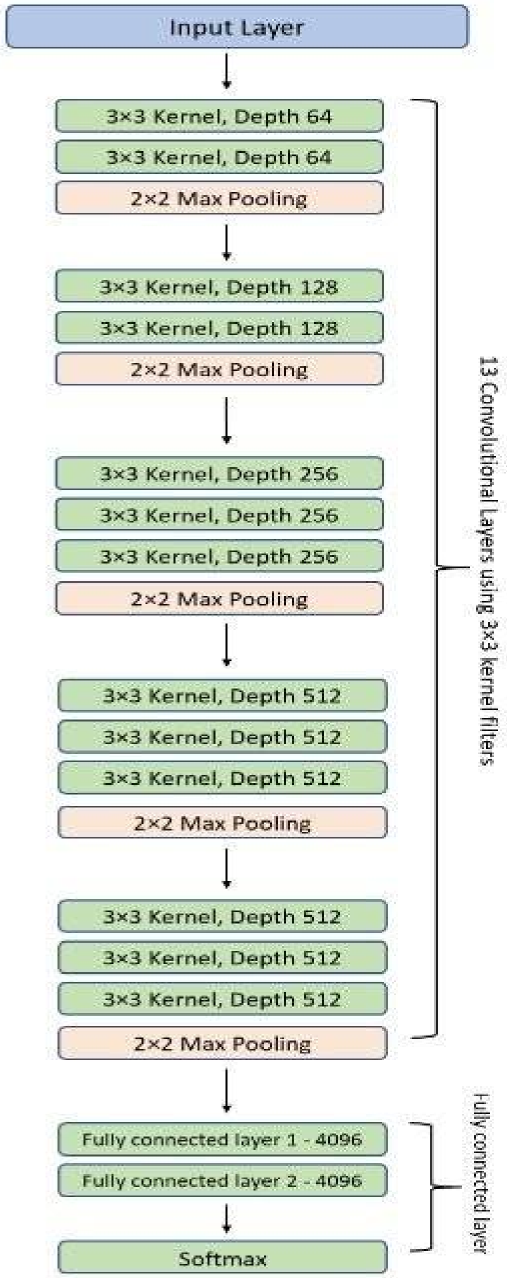
VGG-16 (2014)



- VGG-16 has a deep CNN architecture
- Accepts colored images with dimensions 224×224
- Outputs one of the 1000 classes.
- 138 million parameters

16 layers:

- 13 convolution layers with kernel size 3×3, followed by
- 3 fully connected layers.



VGG-16 Network structure.

NO. Layers Type 0utput
1 2D Convolution_1 of Block_1 3 × 3 64, 224, 224
2 2D Convolution_2 of Block_1 3 × 3 64, 224, 224
Max pool

3 2D Convolution_1 of Block_2 3 × 3 128, 112, 112
4 2D Convolution_2 of Block_2 3 × 3 128, 112, 112
Max pool

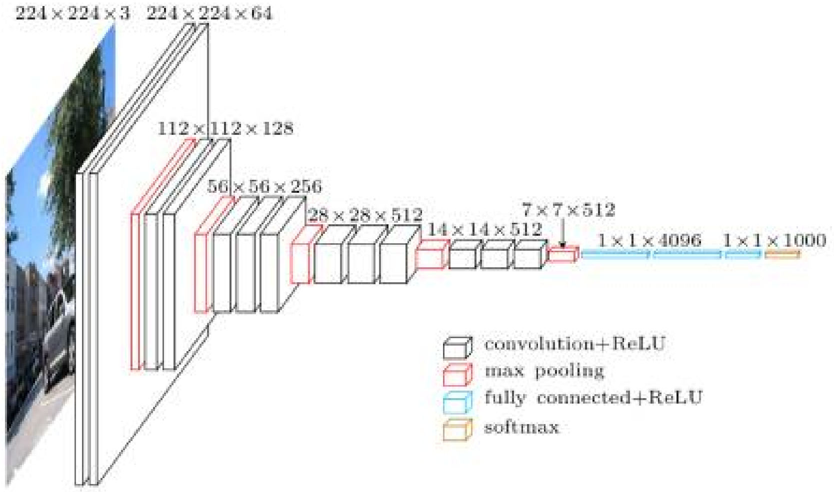
5 2D Convolution_1 of Block_3 3 × 3 256, 56, 56
6 2D Convolution_2 of Block_3 3 × 3 256, 56, 56
7 2D Convolution_3 of Block_3 3 × 3 256, 56, 56
Max pool

8 2D Convolution_1 of Block_4 3 × 3 512, 28, 28
9 2D Convolution_2 of Block_4 3 × 3 512, 28, 28
10 2D Convolution_3 of Block_4 3 × 3 512, 28, 28
Max pool

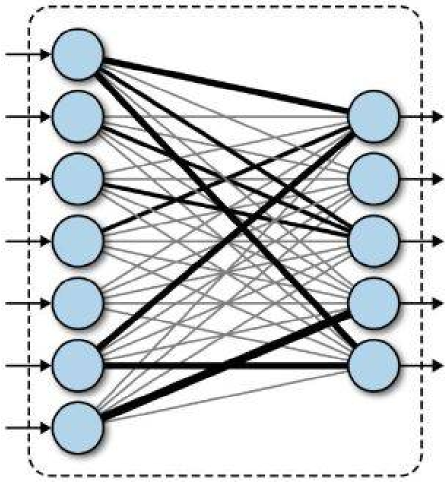
11 2D Convolution_2 of Block_5 3 × 3 512, 14, 14
12 2D Convolution_3 of Block_5 3 × 3 512, 14, 14
13 2D Convolution_4 of Block_5 3 × 3 512, 14, 14
Max pool

14 FC1 Fully connected 4096
15 FC2 Fully connected 4096
16 FC3 Fully connected 1000
Soft max

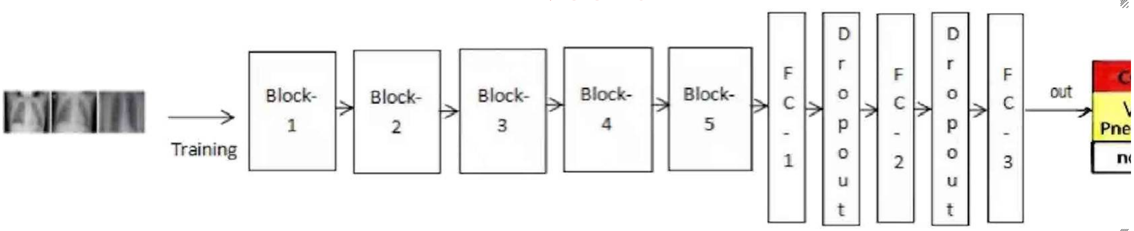
VGG16-Example



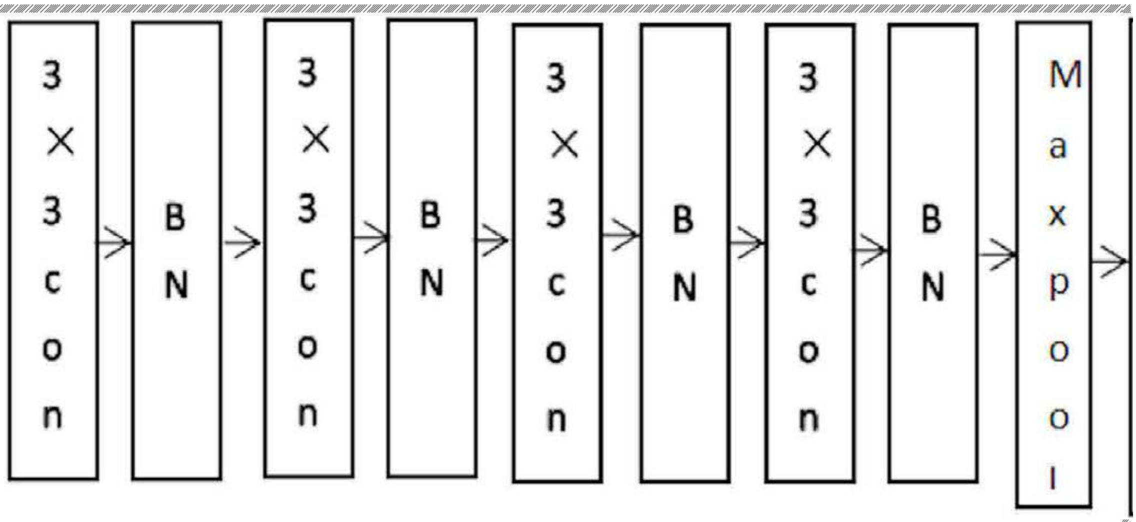
Fully Connected Layers VGG16 Architecture



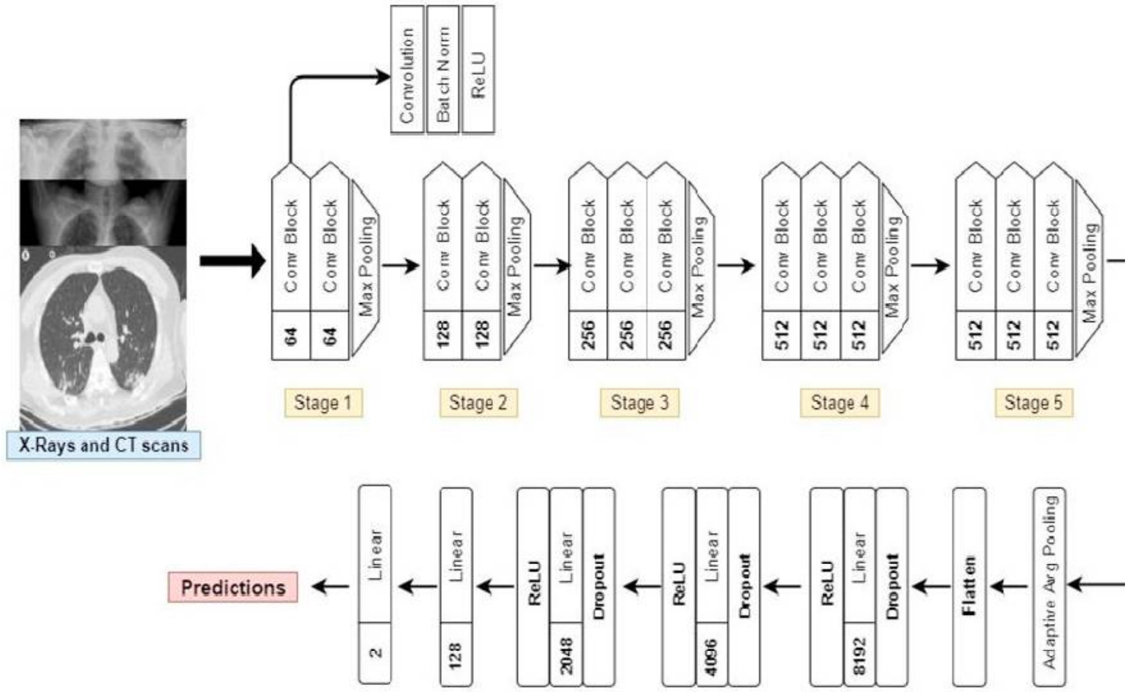
BND-VGG-19



Block-3, Block-4 and Block-5 of BND-VGG-19

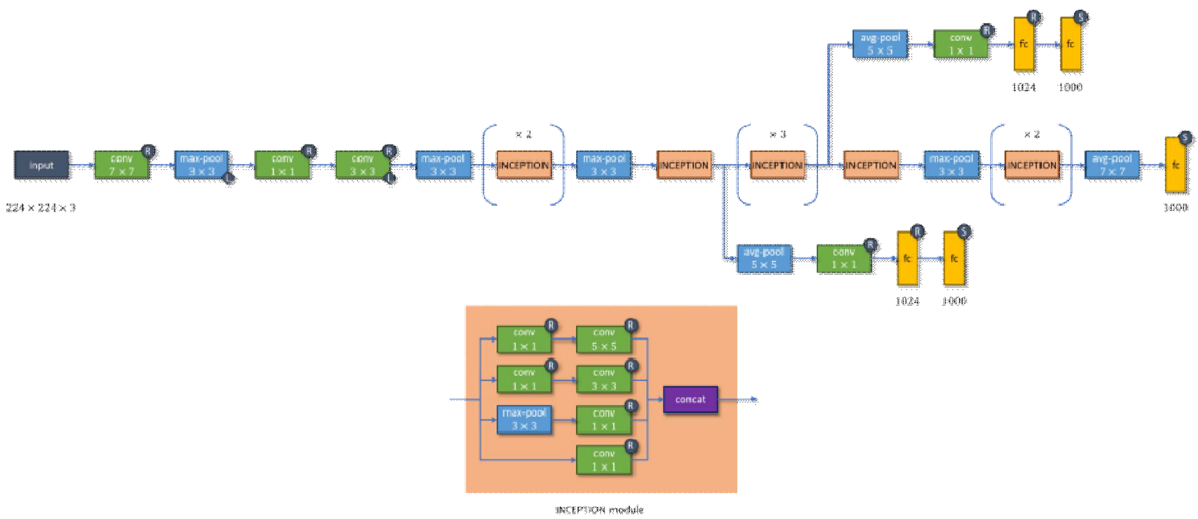


3 × 3 con
BN
3 × 3 con
BN
3 × 3 con
BN
3 × 3 con
BN
Maxpool



X-Rays and CT scans
Convolution
Batch Norm
ReLU
Conv Block 64
Conv Block 64
Max Pooling
Stage 1
Conv Block 128
Conv Block 128
Max Pooling
Stage 2
Conv Block 256
Conv Block 256
Conv Block 256
Max Pooling
Stage 3
Conv Block 512
Conv Block 512
Conv Block 512
Max Pooling
Stage 4
Conv Block 512
Conv Block 512
Conv Block 512
Max Pooling
Stage 5
Adaptive Avg Pooling
Flatten
ReLU
Linear 8192
Dropout
ReLU
Linear 4096
Dropout
ReLU
Linear 2048
Dropout
Linear 128
Linear 2
Predictions

Inception Net



- 7 million parameters
- Last layer of main classifier and two auxiliary classifiers are equipped with a softmax activation function
- All others have ReLu
- 1st and 3rd are Sconvolution layers
- 2nd and 7th inception modules
- Followed by a 3×3 max-pooling
- Last inception module is followed by a 7×7 average-pooling
- LRN is applied after the 1st max-pooling and 3rd convolution layer

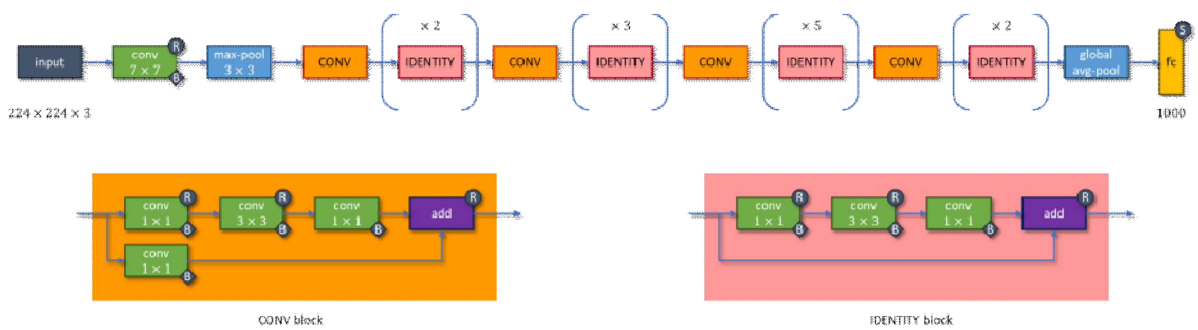
\+ Inception-v1 tackles this issue by adding two auxiliary classifiers connected to intermediate layers, with the hope to increase the gradient signal that gets propagated back. During training, their loss gets added to the total loss of the network with a 0.3 discount weight. At inference time, these auxiliary networks are discarded

Auxiliary classifiers are branched out after the 3rd and 6th inception modules, each starts with a 5×5 average-pooling and followed by 3 layers:

- 1 convolution layer with 1×1 kernel size,
- 2 fully connected layers

✓ Default Inception-v1 accepts 224×224colored images

✓ Outputs one of 1000 classes



When deeper networks can start converging, a degradation problem has been exposed: with the network depth increasing,

- Accuracy gets saturated and then degrades rapidly
- ! Unexpectedly, such degradation is not caused by overfitting (usually indicated by lower training error and higher testing error) since adding more layers to a suitably deep network leads to higher training error

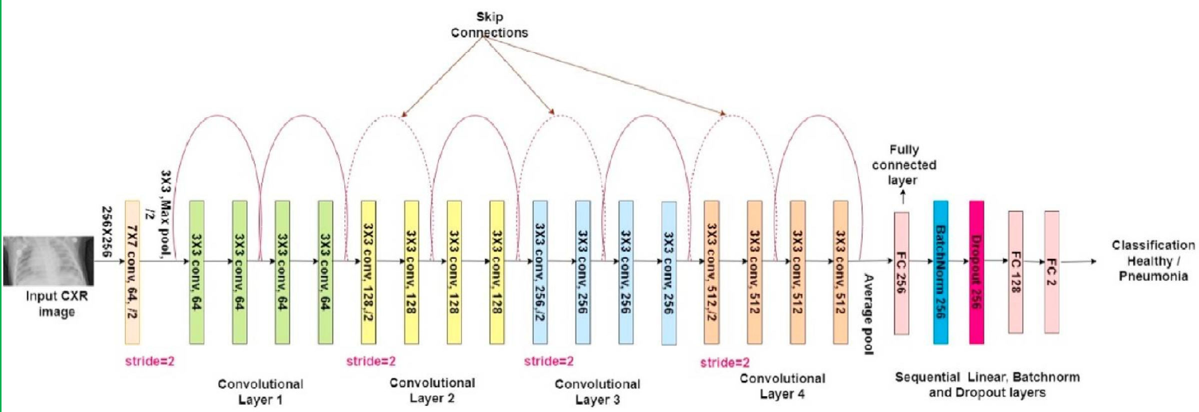
Notice that both residual blocks have 3 layers.

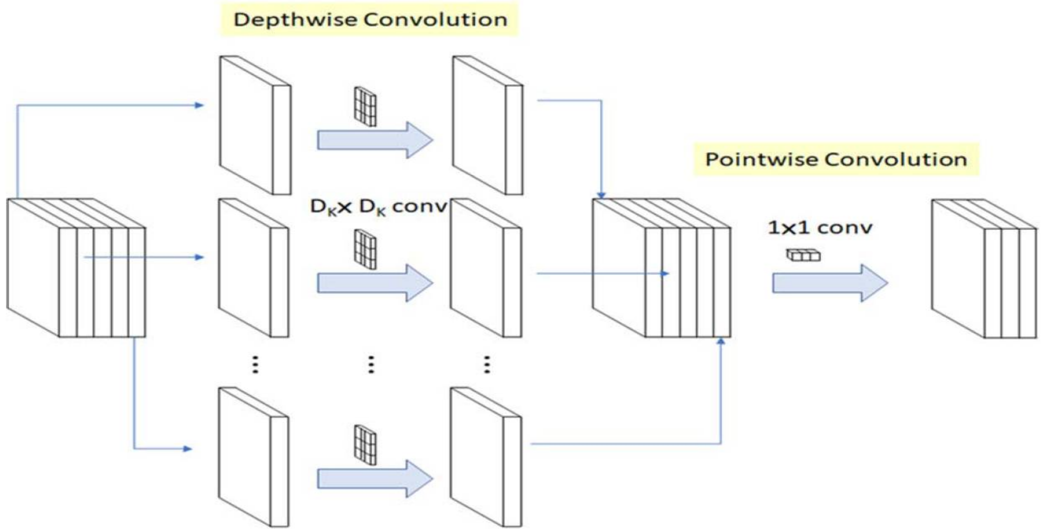
ResNet-50 has 50 layers and 26 million parameters

- ✓ 1 convolution layer with BN then ReLU is applied, followed by
- ✓ 9 layers that consist of 1 convolution block and 2 identity blocks, followed by
- ✓ 12 layers that consist of 1 convolution block and 3 identity blocks, followed by
- ✓ 18 layers that consist of 1 convolution block and 5 identity blocks, followed by
- ✓ 9 layers that consist of 1 convolution block and 2 identity blocks, followed by
- ✓ 1 fully connected layer with softmax

- o The first convolution layer is followed by a 3×3 max-pooling
- o Last identity block is followed by a global-average-pooling
- o Default ResNet-50 accepts colored images with dimensions 224×224 and outputs one of the 1000 classes.

Architecture customized ResNet model

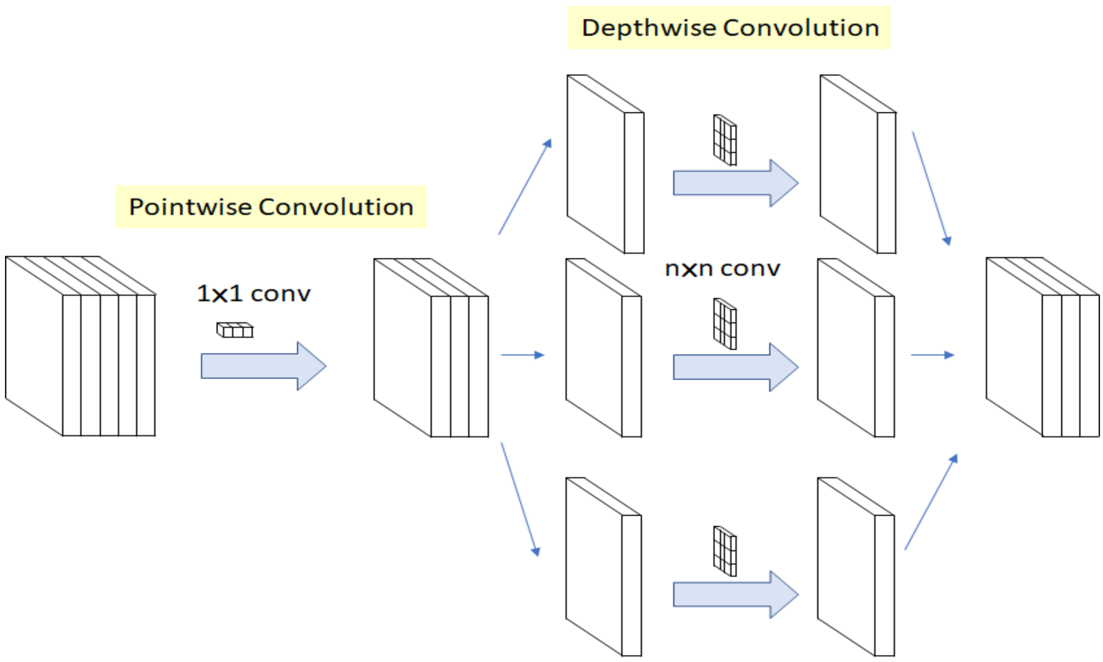




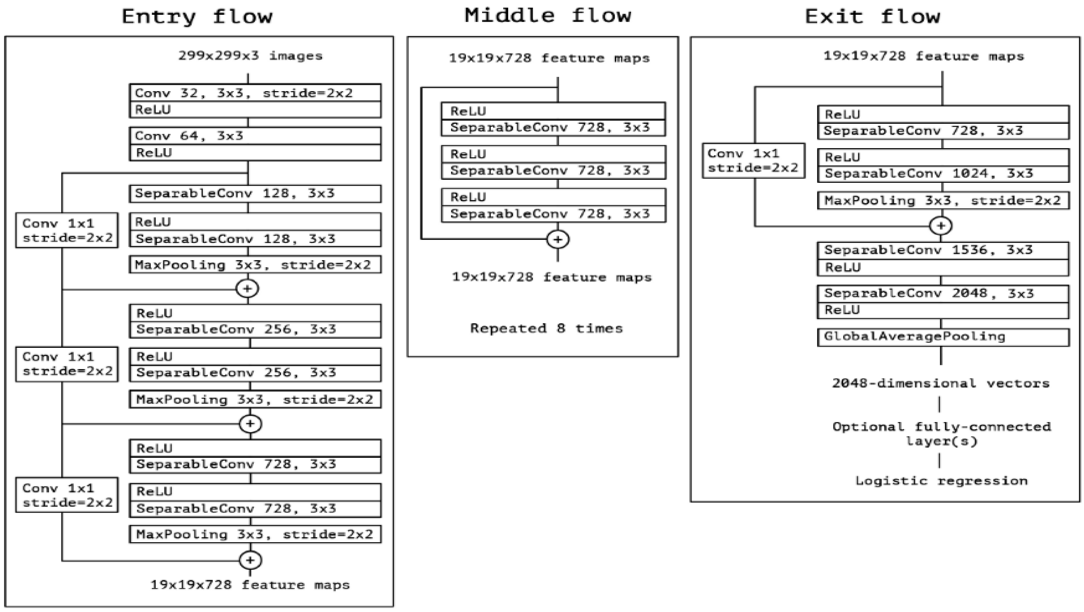
MobileNet module Depthwise Convolution and Pointwise convolution (2017)

MobileNet is a lightweight architecture consisting of 30 layers for image classification (71% accuracy)
I. Layer of convolution with 2 strides
Ii. Depthwise convolution layer
Iii. Pointwise convolution layer which doubles the number of channels
Iv. Depthwise convolution layer with 2 strides
V. Pointwise convolution layer which doubles the number of channels

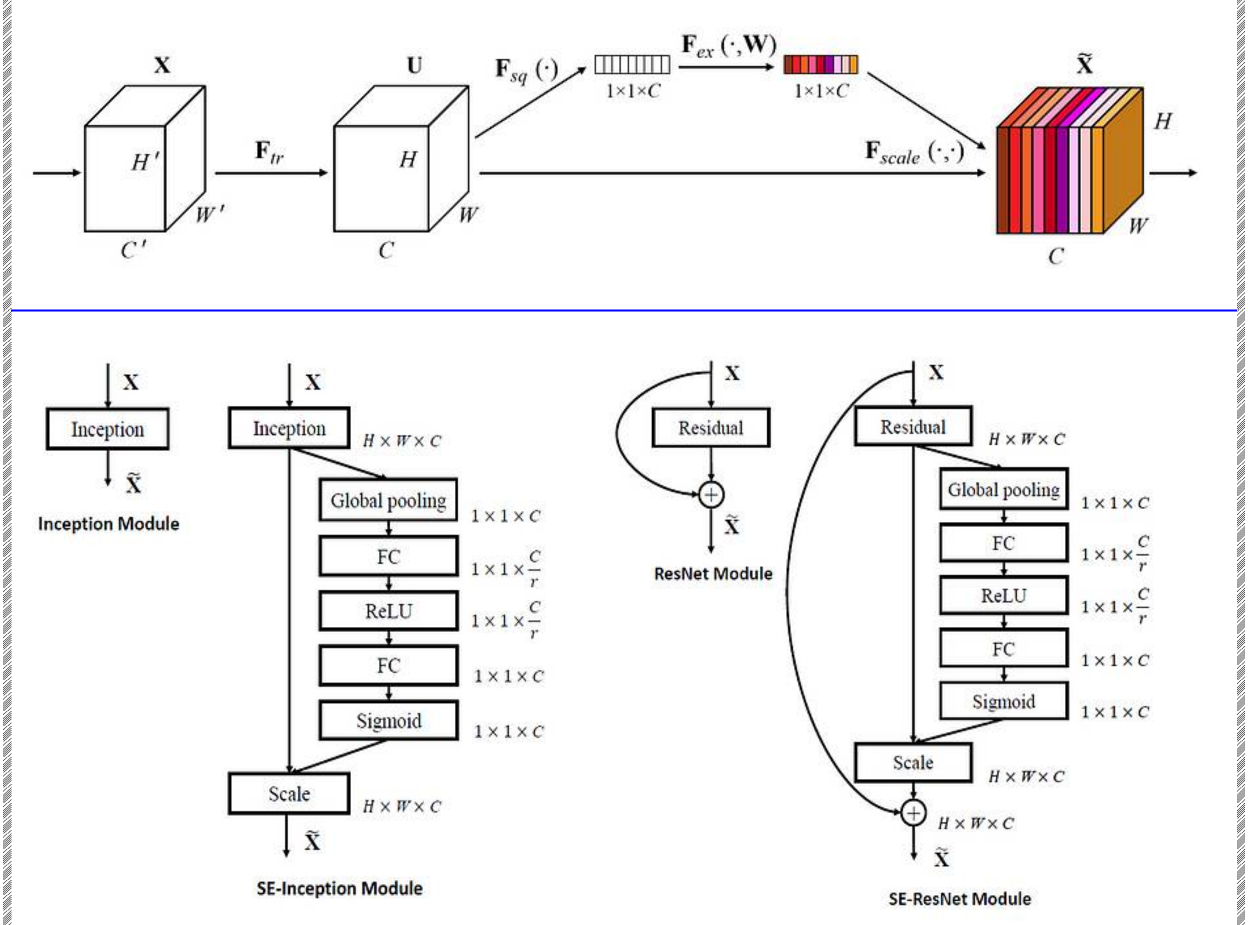
Xception



Architecture. Xception (Entry Flow > Middle Flow > Exit Flow)



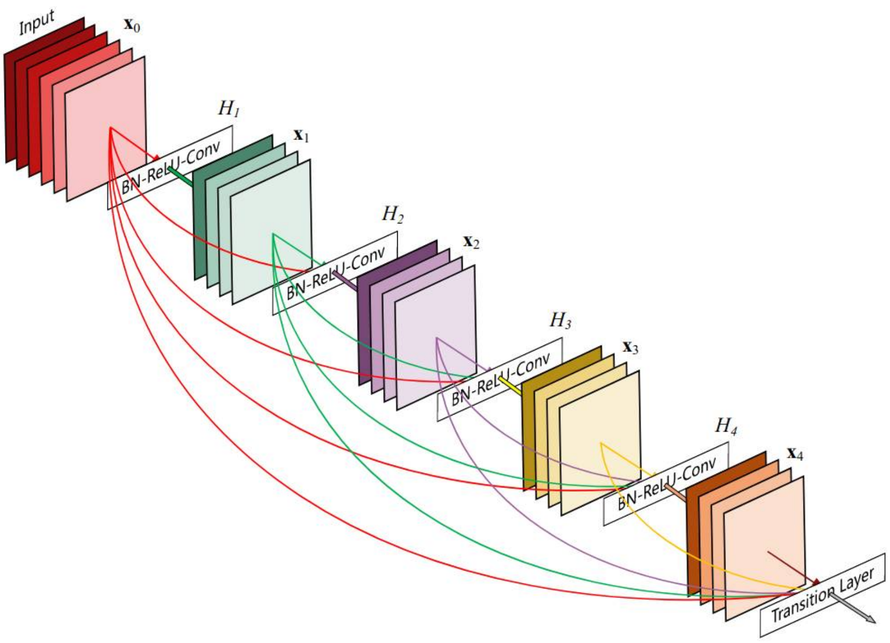
Squeeze-and-Excitation Net (SqExNet)



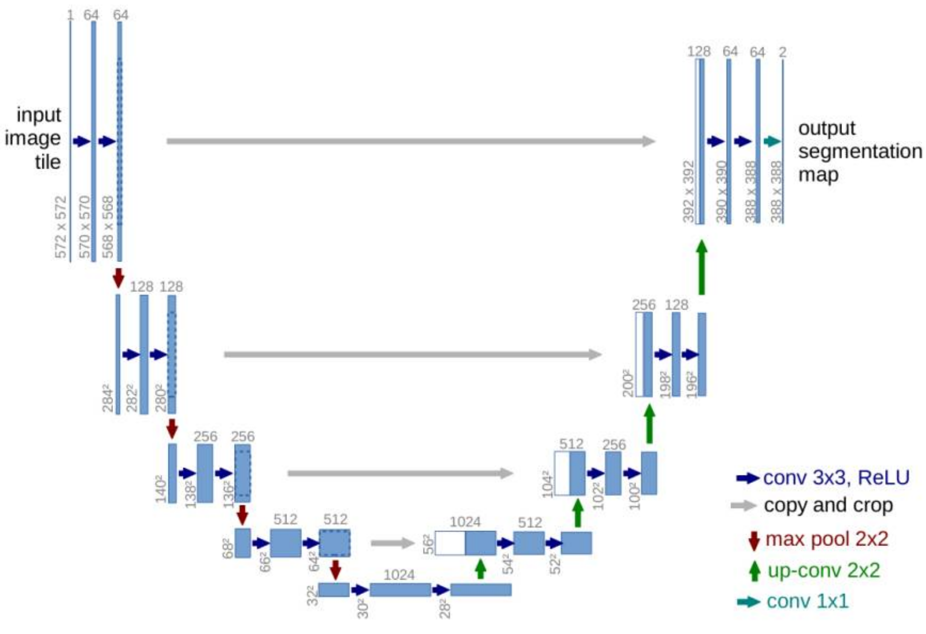
Output size	ResNet-50	SE-ResNet-50	SE-ResNeXt-50 (32 × 4d)
112 × 112	conv, 7 × 7, 64, stride 2		
56 × 56	max pool, 3 × 3, stride 2		
	$\begin{bmatrix} \text{conv}, 1\times1, 64 \\ \text{conv}, 3\times3, 64 \\ \text{conv}, 1\times1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} \text{conv}, 1\times1, 64 \\ \text{conv}, 3\times3, 64 \\ \text{conv}, 1\times1, 256 \\ fc, [16, 256] \end{bmatrix} \times 3$	$\begin{bmatrix} \text{conv}, 1\times1, 128 & \\ \text{conv}, 3\times3, 128 & C=32 \\ \text{conv}, 1\times1, 256 & \\ fc, [16, 256] & \end{bmatrix} \times 3$
28 × 28	$\begin{bmatrix} \text{conv}, 1\times1, 128 \\ \text{conv}, 3\times3, 128 \\ \text{conv}, 1\times1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} \text{conv}, 1\times1, 128 \\ \text{conv}, 3\times3, 128 \\ \text{conv}, 1\times1, 512 \\ fc, [32, 512] \end{bmatrix} \times 4$	$\begin{bmatrix} \text{conv}, 1\times1, 256 & \\ \text{conv}, 3\times3, 256 & C=32 \\ \text{conv}, 1\times1, 512 & \\ fc, [32, 512] & \end{bmatrix} \times 4$
14 × 14	$\begin{bmatrix} \text{conv}, 1\times1, 256 \\ \text{conv}, 3\times3, 256 \\ \text{conv}, 1\times1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} \text{conv}, 1\times1, 256 \\ \text{conv}, 3\times3, 256 \\ \text{conv}, 1\times1, 1024 \\ fc, [64, 1024] \end{bmatrix} \times 6$	$\begin{bmatrix} \text{conv}, 1\times1, 512 & \\ \text{conv}, 3\times3, 512 & C=32 \\ \text{conv}, 1\times1, 1024 & \\ fc, [64, 1024] & \end{bmatrix} \times 6$
7×7	$\begin{bmatrix} \text{conv}, 1\times1, 512 \\ \text{conv}, 3\times3, 512 \\ \text{conv}, 1\times1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} \text{conv}, 1\times1, 512 \\ \text{conv}, 3\times3, 512 \\ \text{conv}, 1\times1, 2048 \\ fc, [128, 2048] \end{bmatrix} \times 3$	$\begin{bmatrix} \text{conv}, 1\times1, 1024 & \\ \text{conv}, 3\times3, 1024 & C=32 \\ \text{conv}, 1\times1, 2048 & \\ fc, [128, 2048] & \end{bmatrix} \times 3$
1 × 1	global average pool, 1000-d fc, softmax		

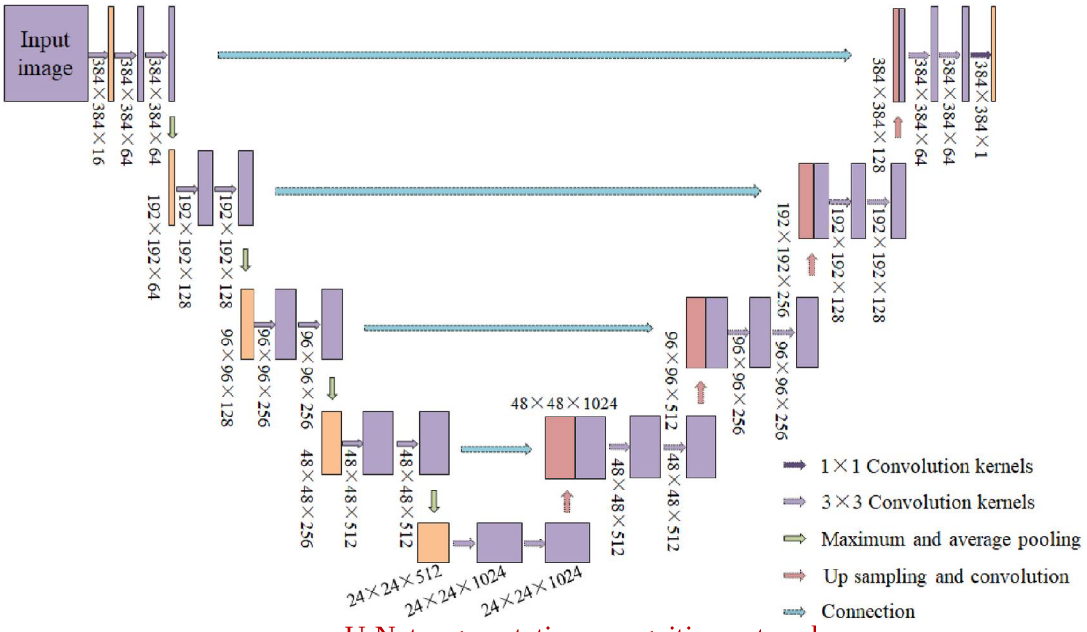
ResNet-50 (Left), SE-ResNet-50 (Middle), SE-ResNeXt-50 (32×4d) (Right)

A 5-layer dense block



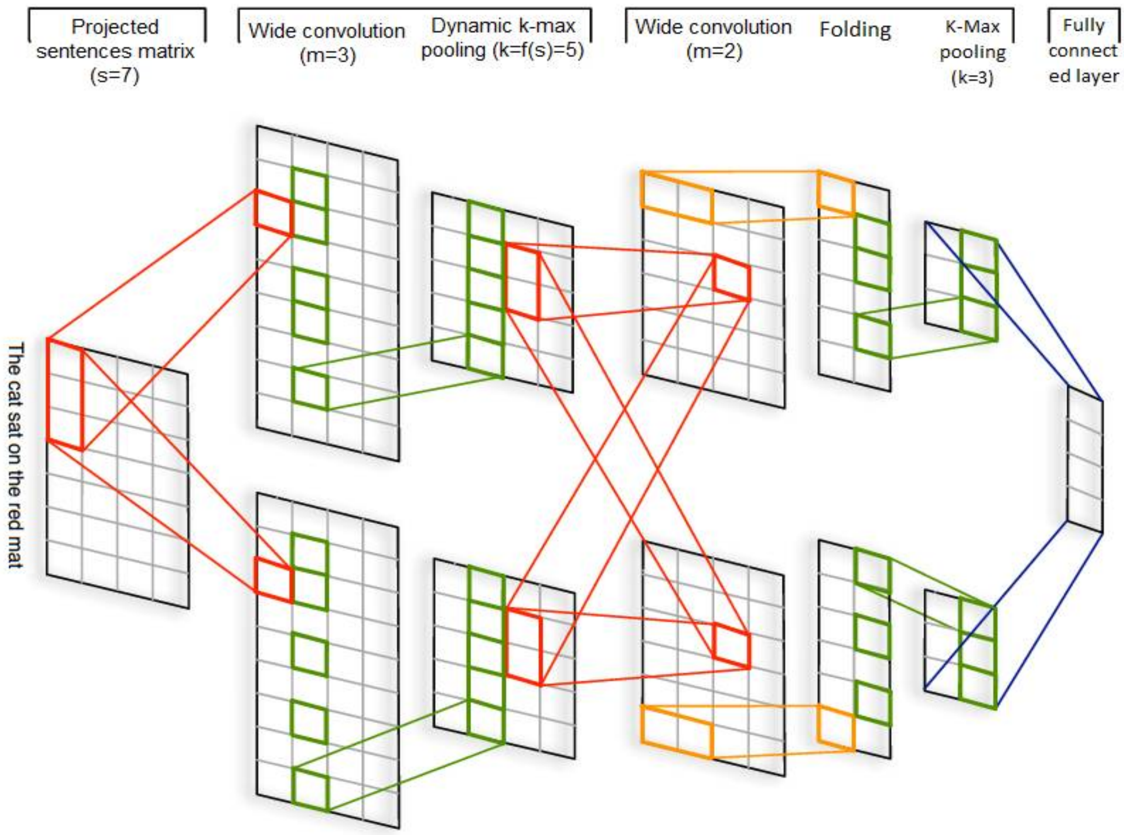
U-Net architecture



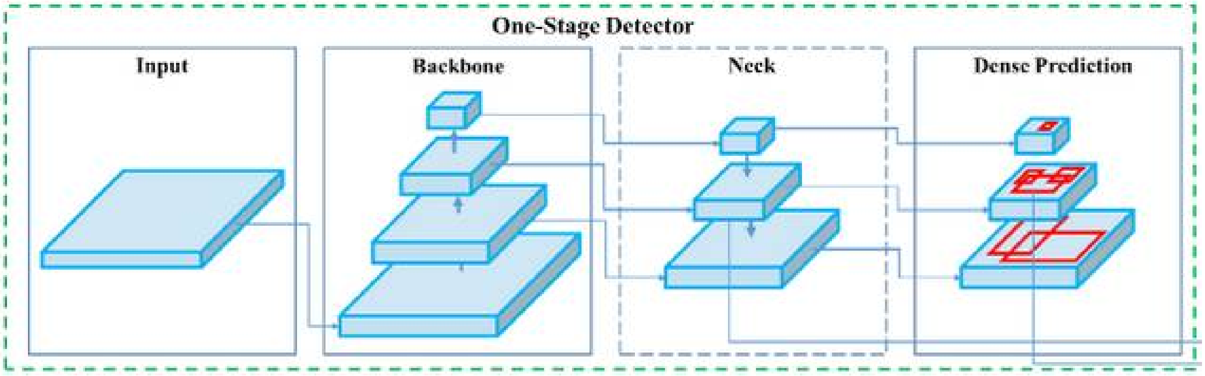


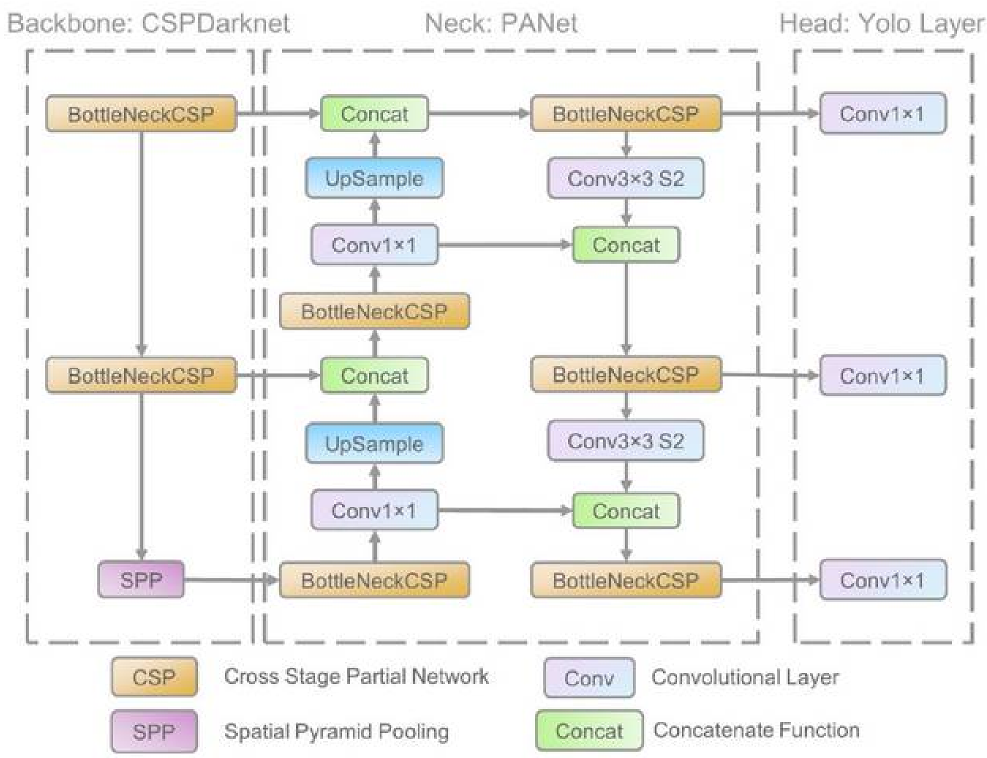
U-Net segmentation recognition network

Dynamic CNN (DynCNN)



YOLO

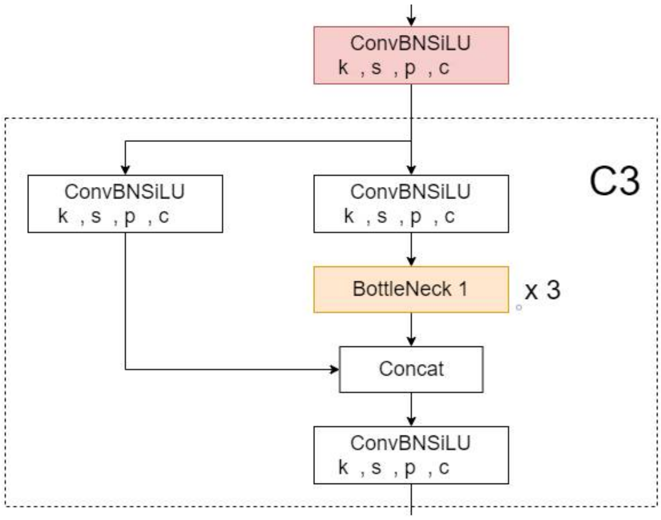


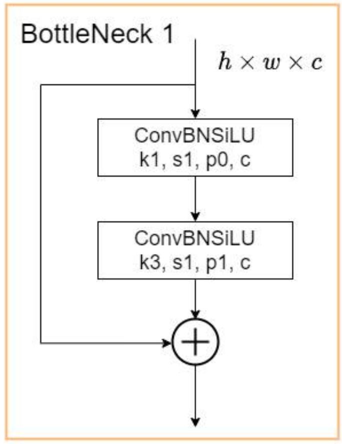


- ✓ YOLOv5 uses CSP-Darknet53 as its backbone
- ✓ CSP-Darknet53 is just the convolutional network **Darknet53** used as the backbone for YOLOv3
- ✓ **Cross Stage Partial** (CSP) network strategy.

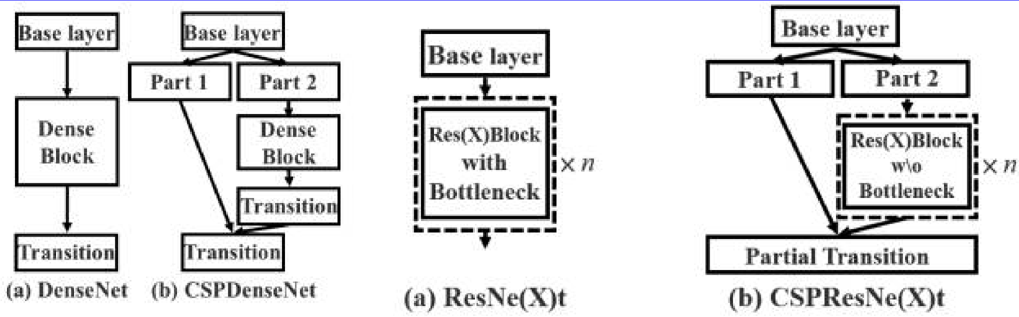
Cross Stage Partial Network

- + Preserves the advantage of DenseNet's feature reuse characteristics and helps reducing the excessive amount of redundant gradient information by truncating the gradient flow

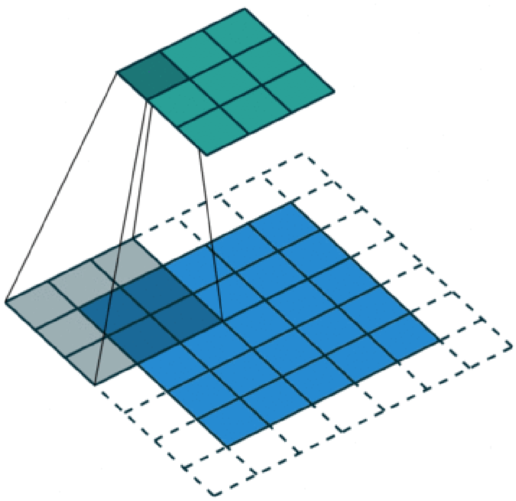




BottleNeckCSP module architecture; source: YOLOv5 github repo

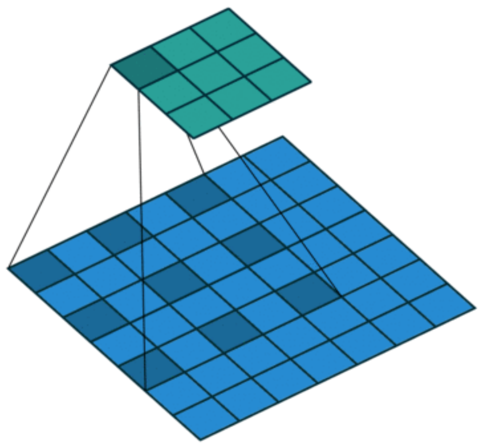


DilatedNet — Dilated Convolution (Semantic Segmentation)

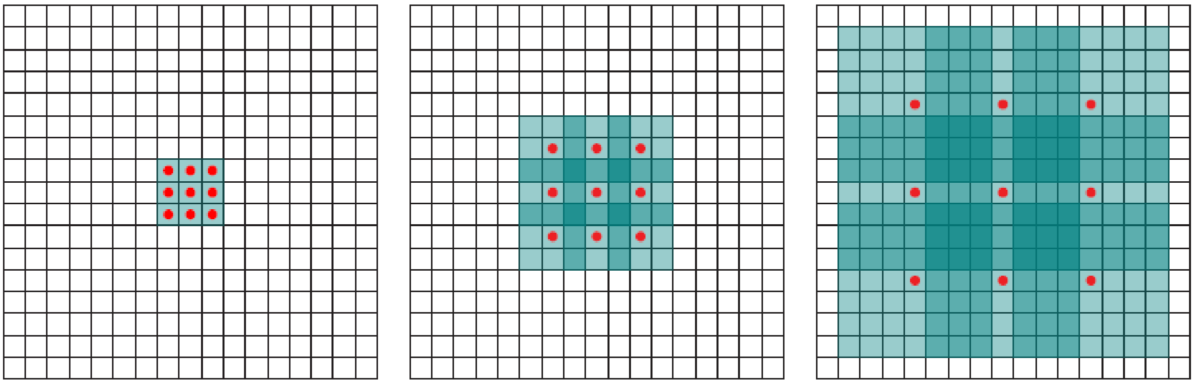


Standard Convolution (l=1)

Dilated Convolution (l=2)



Dilated Convolution (l=2)

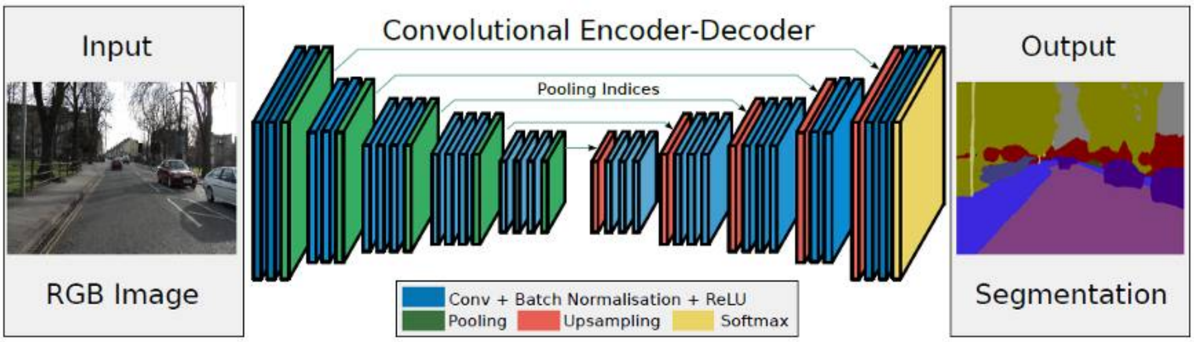


Receptive field is larger compared with the standard one
l=1 (left), l=2 (Middle), l=4 (Right)

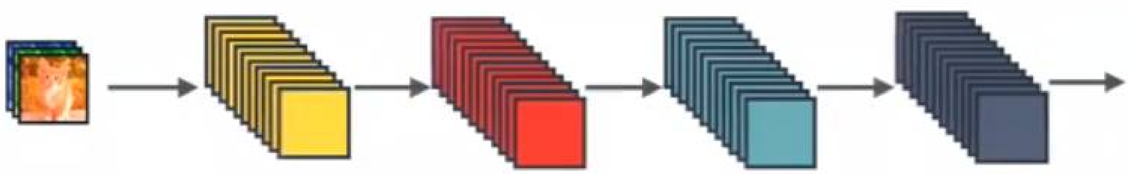
Layer	1	2	3	4	5	6	7	8
Convolution	3×3	3×3	3×3	3×3	3×3	3×3	3×3	1×1
Dilation	1	1	2	4	8	16	1	1
Truncation	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
Receptive field	3×3	5×5	9×9	17×17	33×33	65×65	67×67	67×67
Output channels								
Basic	C	C	C	C	C	C	C	C
Large	$2C$	$2C$	$4C$	$8C$	$16C$	$32C$	$32C$	C

Basic and Large **Multi-Scale Context Aggregation (i.e. Context Module)**

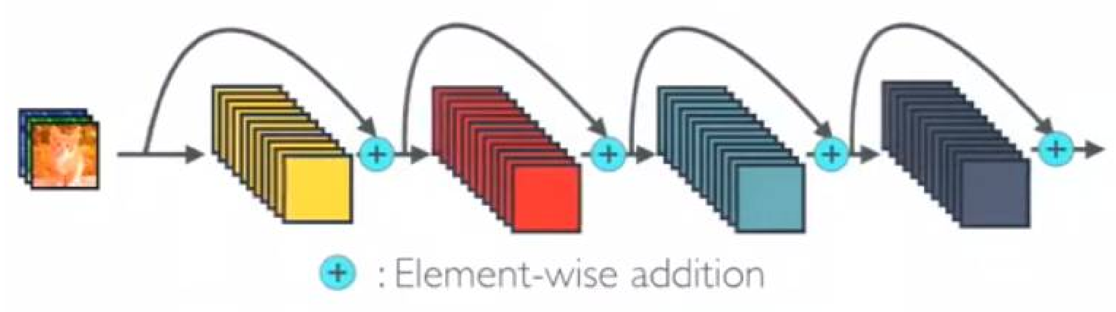
SegNet Architecture: Encoder Decoder - pixelwise classification layer

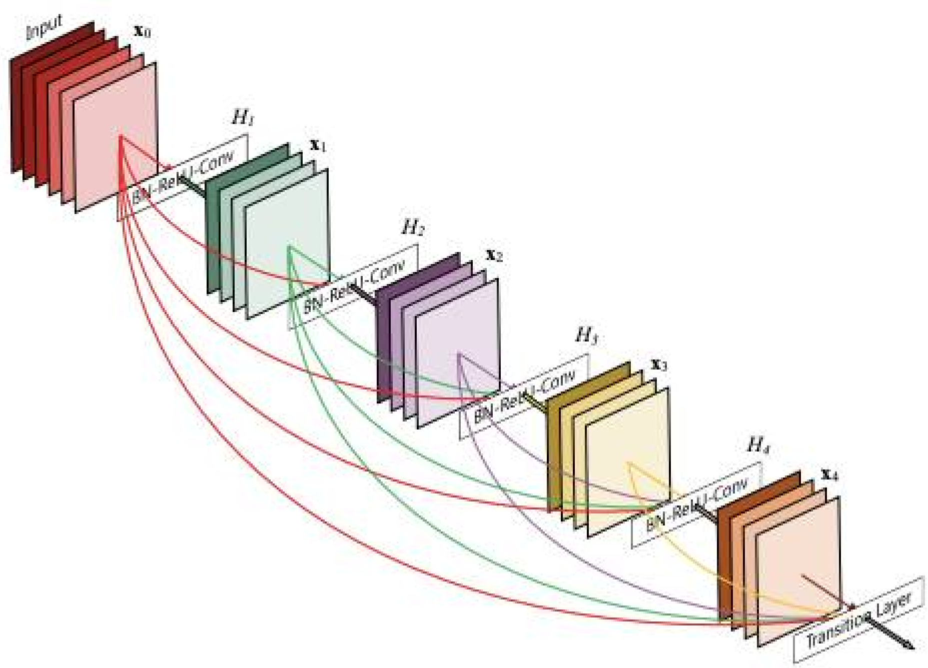


CNNNet



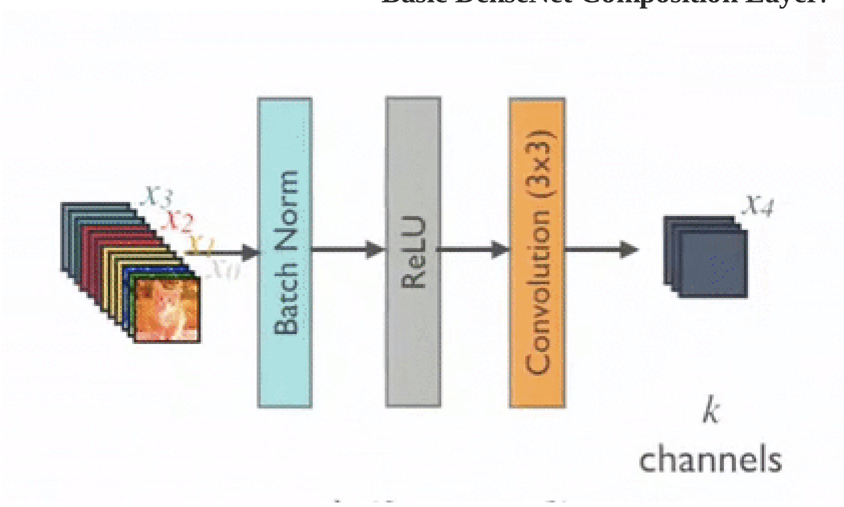
Dense Net



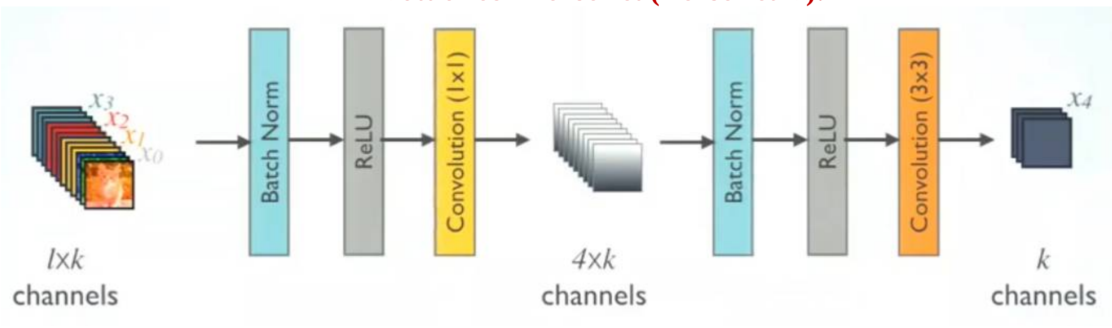


- ✓ Layers in network receive feature maps from all the preceding layers
- + Network will be thinner and more compact

Basic DenseNet Composition Layer:



BottleNeck DenseNet (DenseNet-B):



Multiple Dense Blocks with Transition Layers

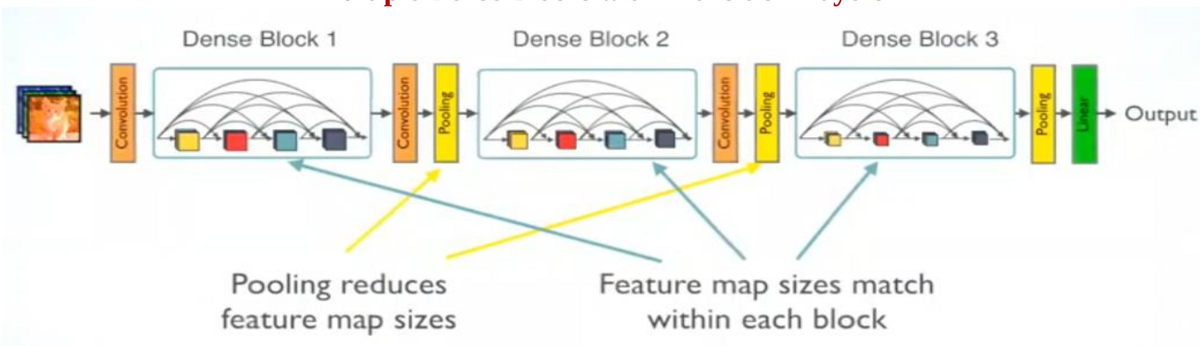
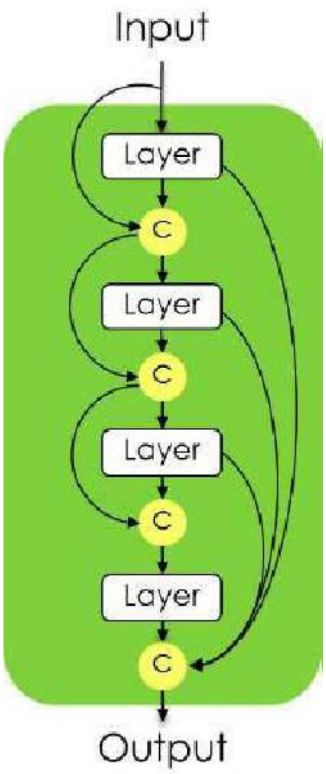


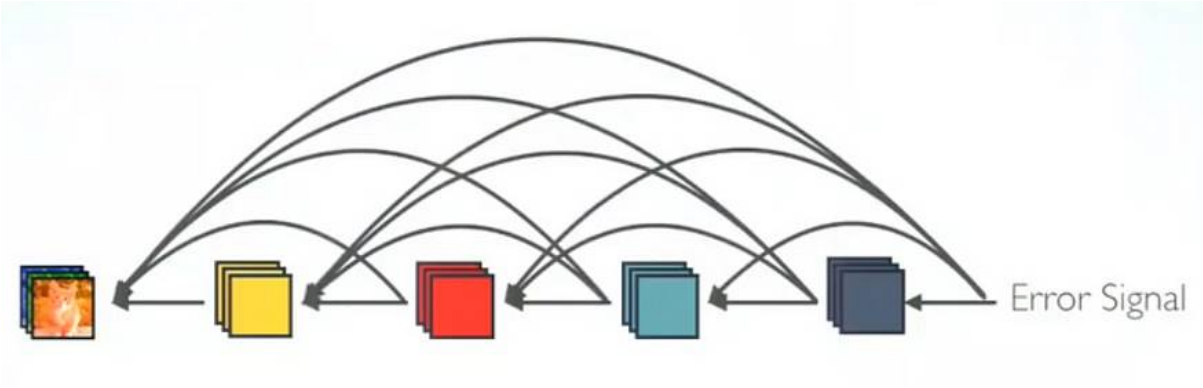
Diagram of a dense block of 4 layers



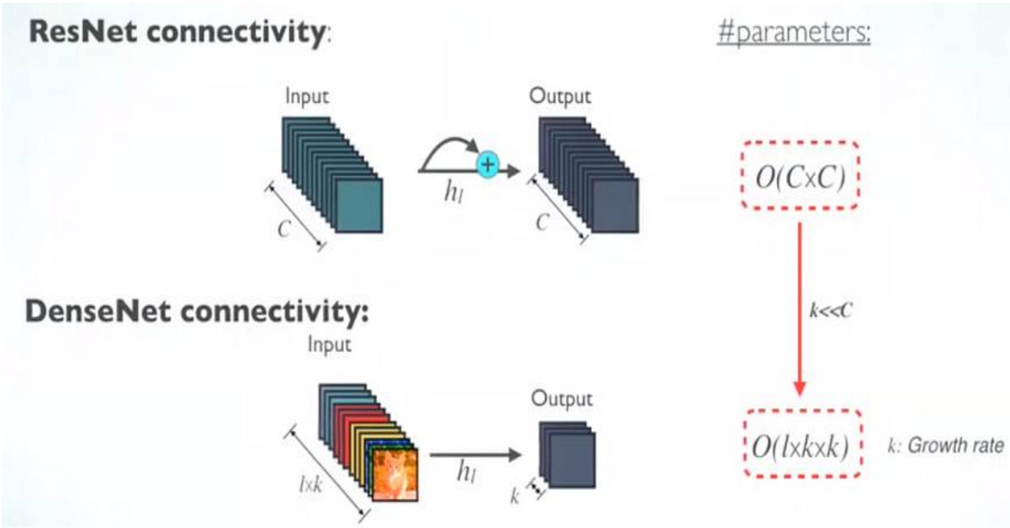
- Outputs of the 4 layers,
- Thus contains 4 □ k feature maps

+ Advantages of DenseNet

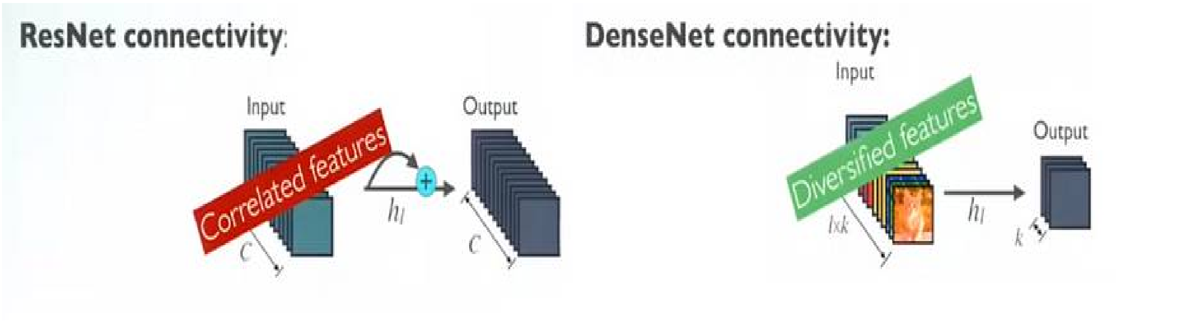
Strong Gradient Flow

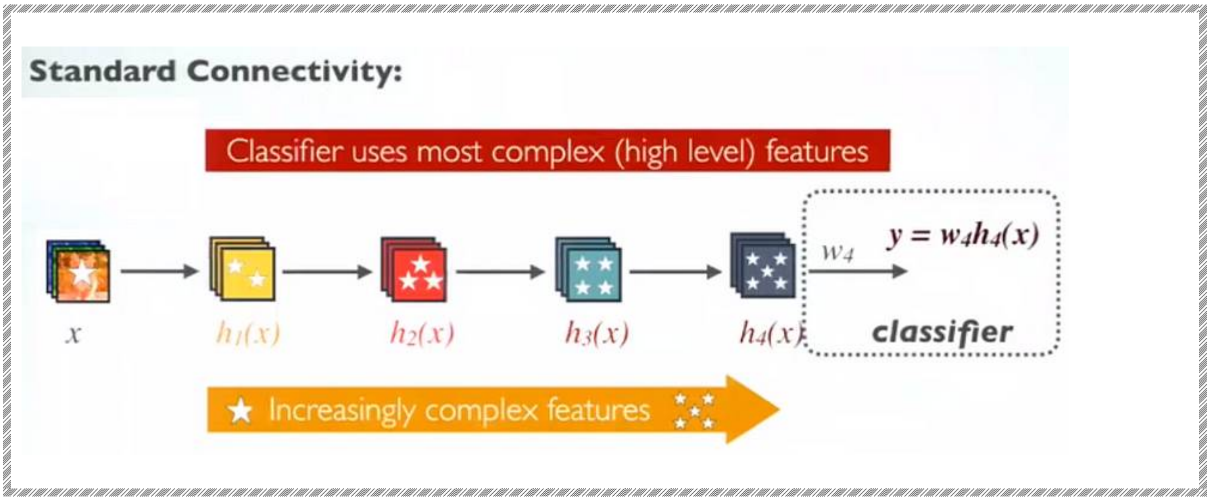


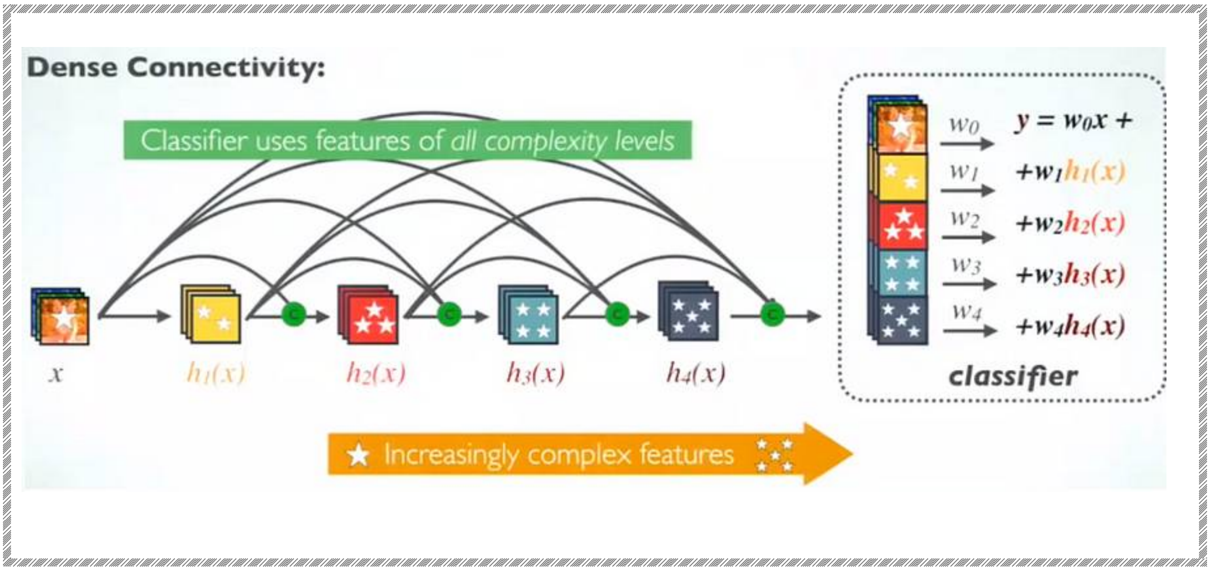
Parameter & Computational Efficiency

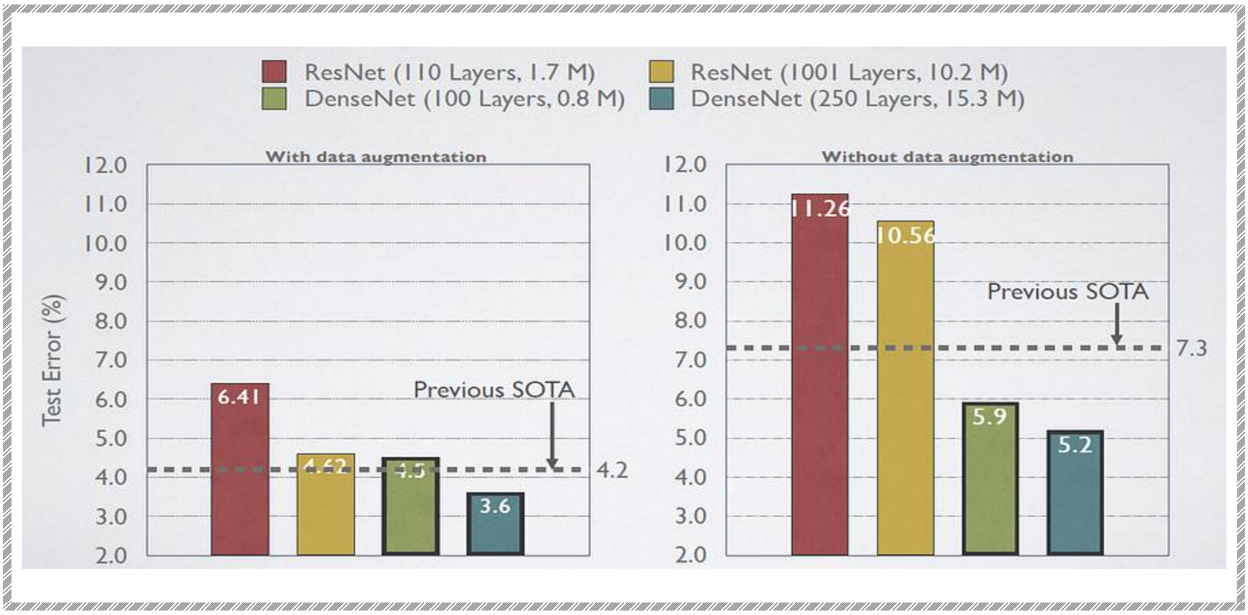


More Diversified Features in DenseNet



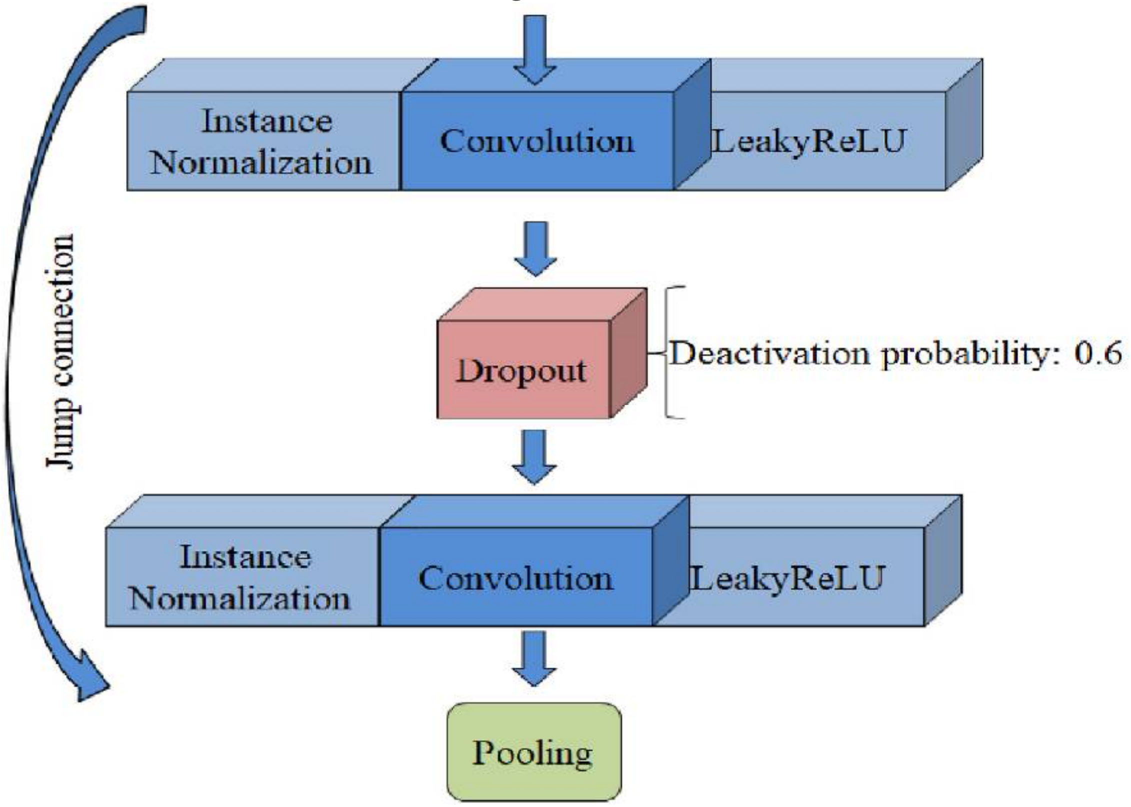

Standard Connectivity:
Classifier uses most complex (high level) features
$y = w_4 h_4(x)$
w_4
x
$h_1(x)$
$h_2(x)$
$h_3(x)$
$h_4(x)$
classifier
Increasingly complex features


Dense Connectivity:
Classifier uses features of all complexity levels
x
$h_1(x)$
$h_2(x)$
$h_3(x)$
$h_4(x)$
w_0 $y = w_0 x +$
w_1 $+w_1 h_1(x)$
w_2 $+w_2 h_2(x)$
w_3 $+w_3 h_3(x)$
w_4 $+w_4 h_4(x)$
classifier
Increasingly complex features

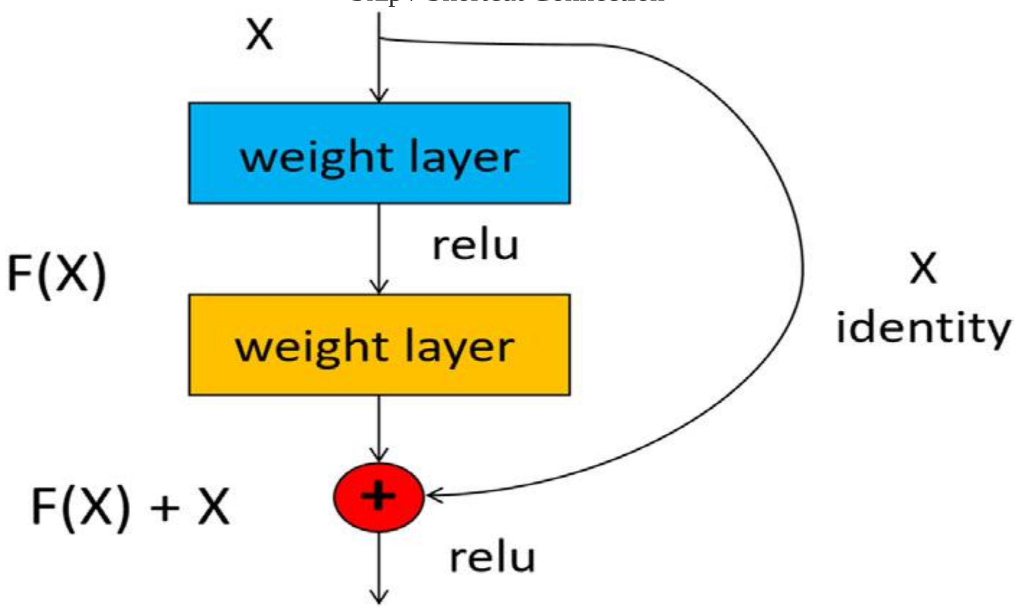

ResNet (110 Layers, 1.7 M)
ResNet (1001 Layers, 10.2 M)
DenseNet (100 Layers, 0.8 M)
DenseNet (250 Layers, 15.3 M)
With data augmentation
Without data augmentation
Test Error (%)
12.0
11.0
10.0
9.0
8.0
7.0
6.0
5.0
4.0
3.0
2.0
6.41
4.62
4.5
3.6
Previous SOTA
4.2
11.26
10.56
5.9
5.2
Previous SOTA
7.3

Residual Network (ResNet)

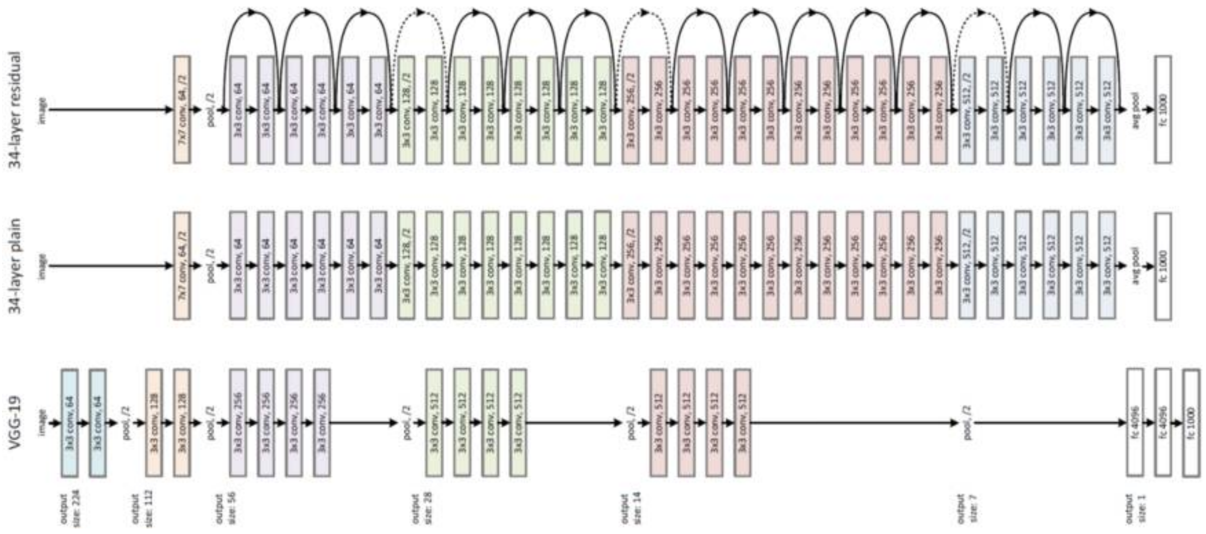
Structural diagram of residual block



Skip / Shortcut Connection



ResNet Architecture

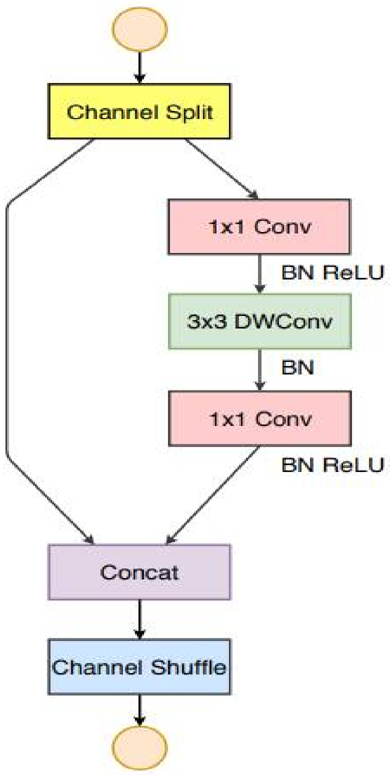


- ✓ 34-layer ResNet with Skip / Shortcut Connection (Top)
- ✓ 34-layer Plain Network (Middle)
- ✓ 19-layer VGG-19 (Bottom)
  - o a state-of-the-art approach in ILSVRC 2014

- ✓ For ResNet, there are 3 types of skip / shortcut connections when the input dimensions are smaller than the output dimensions.
  - o Shortcut performs identity mapping, with extra zero padding for increasing dimensions. Thus, no extra parameters
  - o The projection shortcut is used for increasing dimensions only, the other shortcuts are identity. Extra parameters are needed.
  - o All shortcuts are projections. Extra parameters are more than that of middle.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

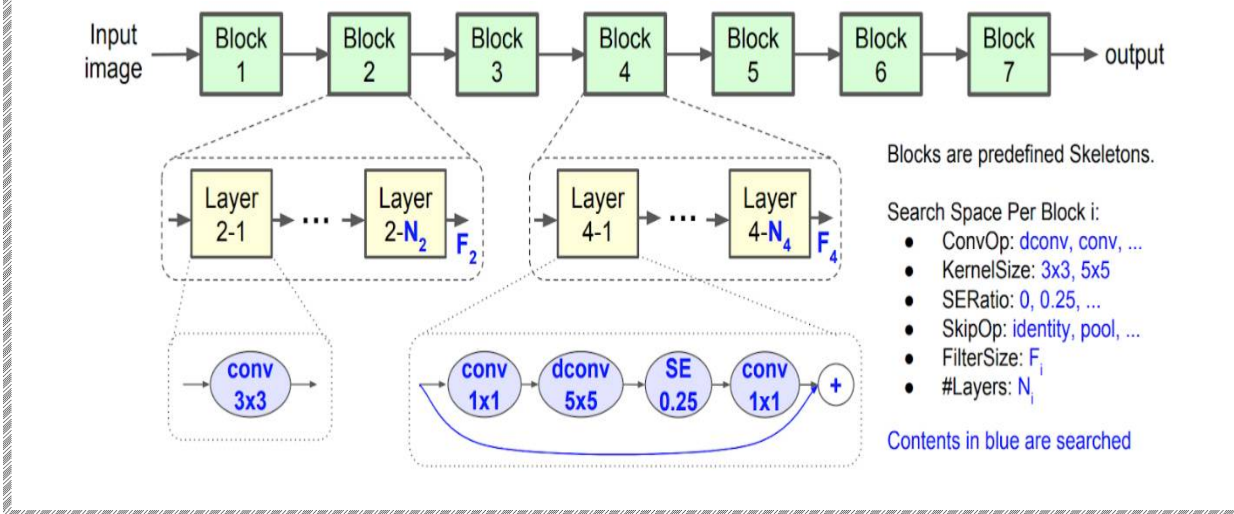
ShuffleNet v2 architecture (2018)



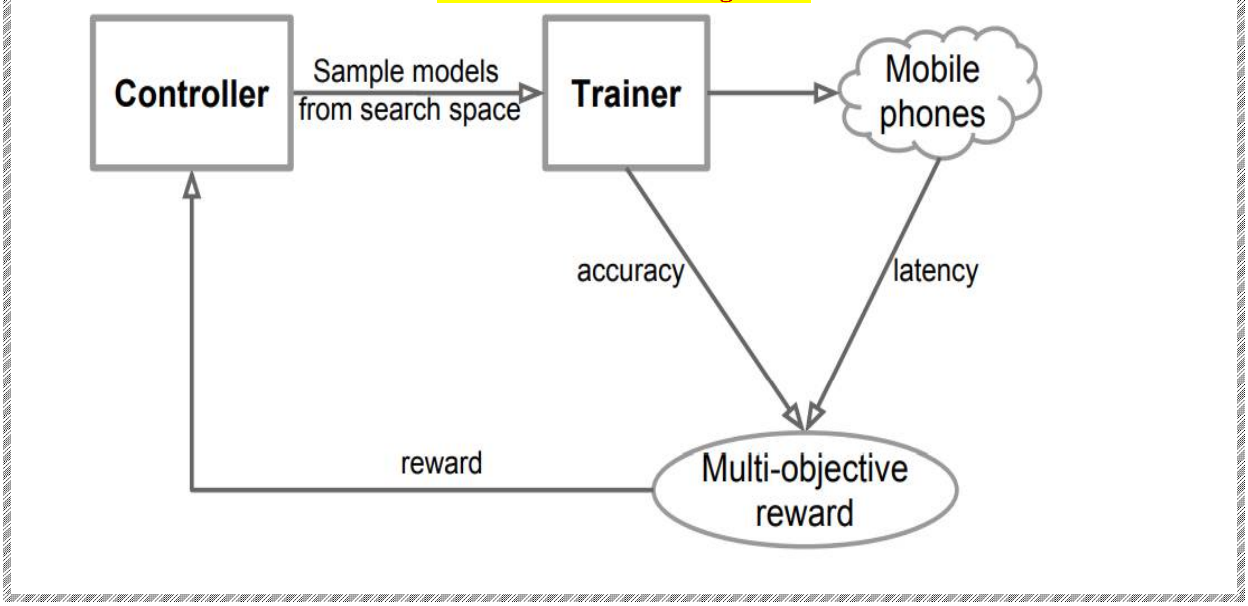
ShuffleNet v2 architecture table

Layer	Output size	KSize	Stride	Repeat	Output channels			
					0.5×	1×	1.5×	2×
Image	224×224				3	3	3	3
Conv1 MaxPool	112×112 56×56	3×3 3×3	2 2	1	24	24	24	24
Stage2	28×28 28×28		2 1	1 3	48	116	176	244
Stage3	14×14 14×14		2 1	1 7	96	232	352	488
Stage4	7×7 7×7		2 1	1 3	192	464	704	976
Conv5	7×7	1×1	1	1	1024	1024	1024	2048
GlobalPool	1×1	7×7						
FC					1000	1000	1000	1000
FLOPs					41M	146M	299M	591M
# of Weights					1.4M	2.3M	3.5M	7.4M

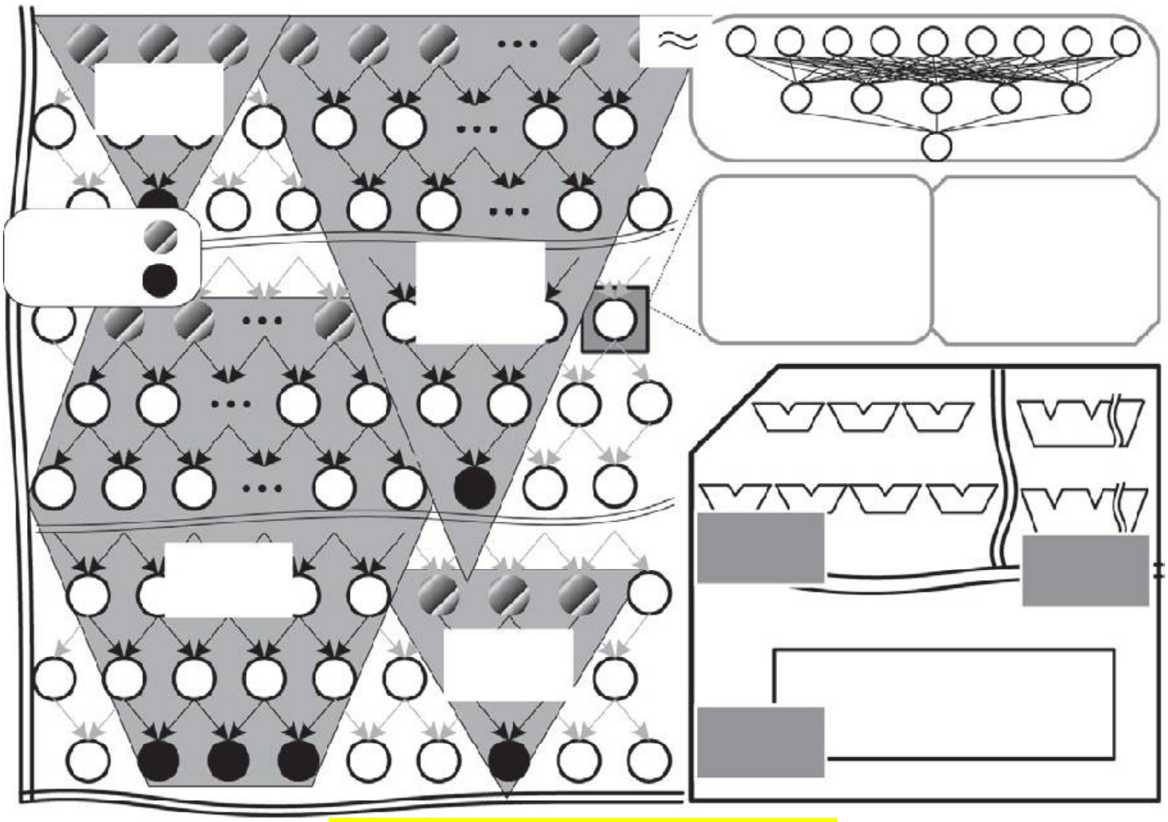
MnasNet (2019)



Reinforcement search algorithm

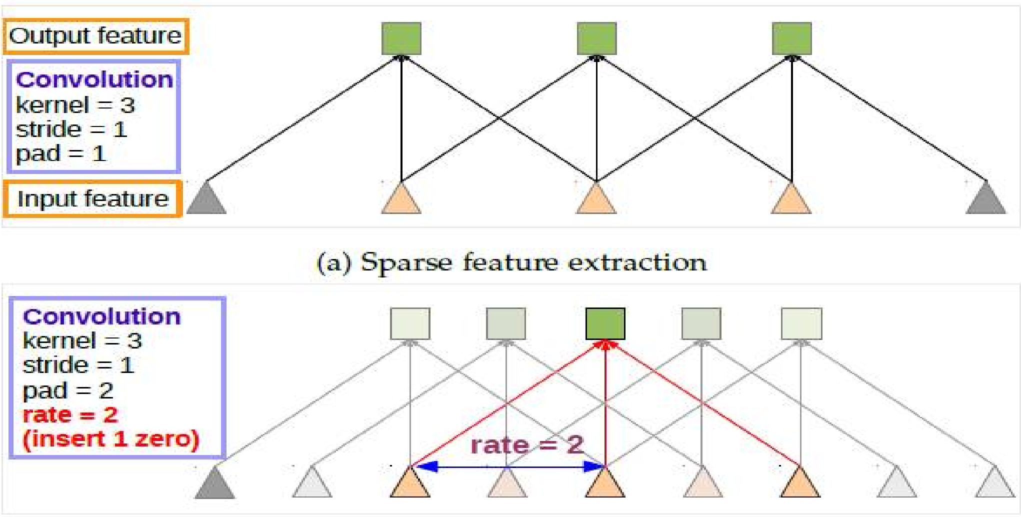


Layout of PE array with DiaNet topology:



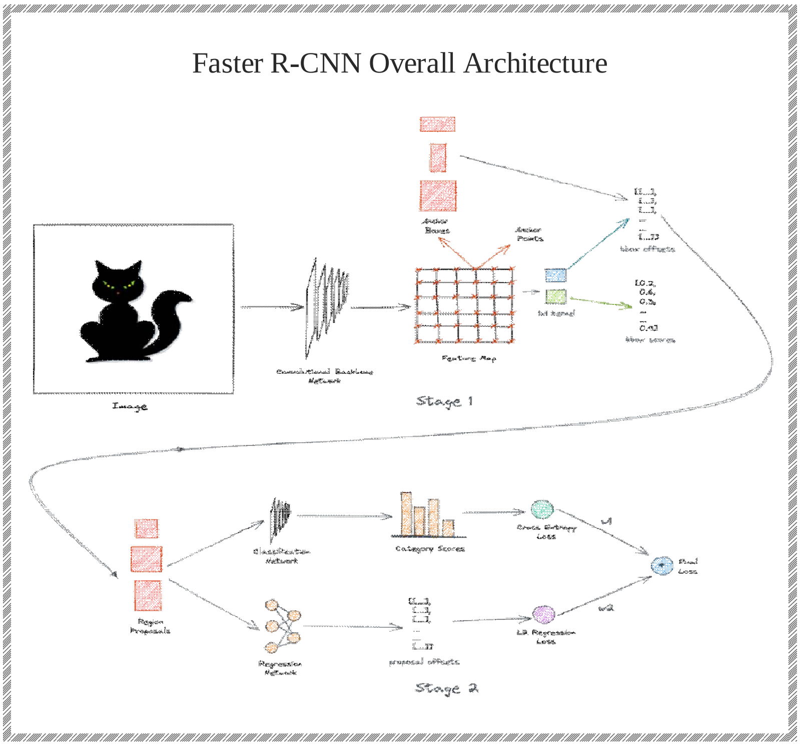
A large scale PE arrays can be reconfigured partitioned into various tasks, which are executed in independent DiaNets parallelly

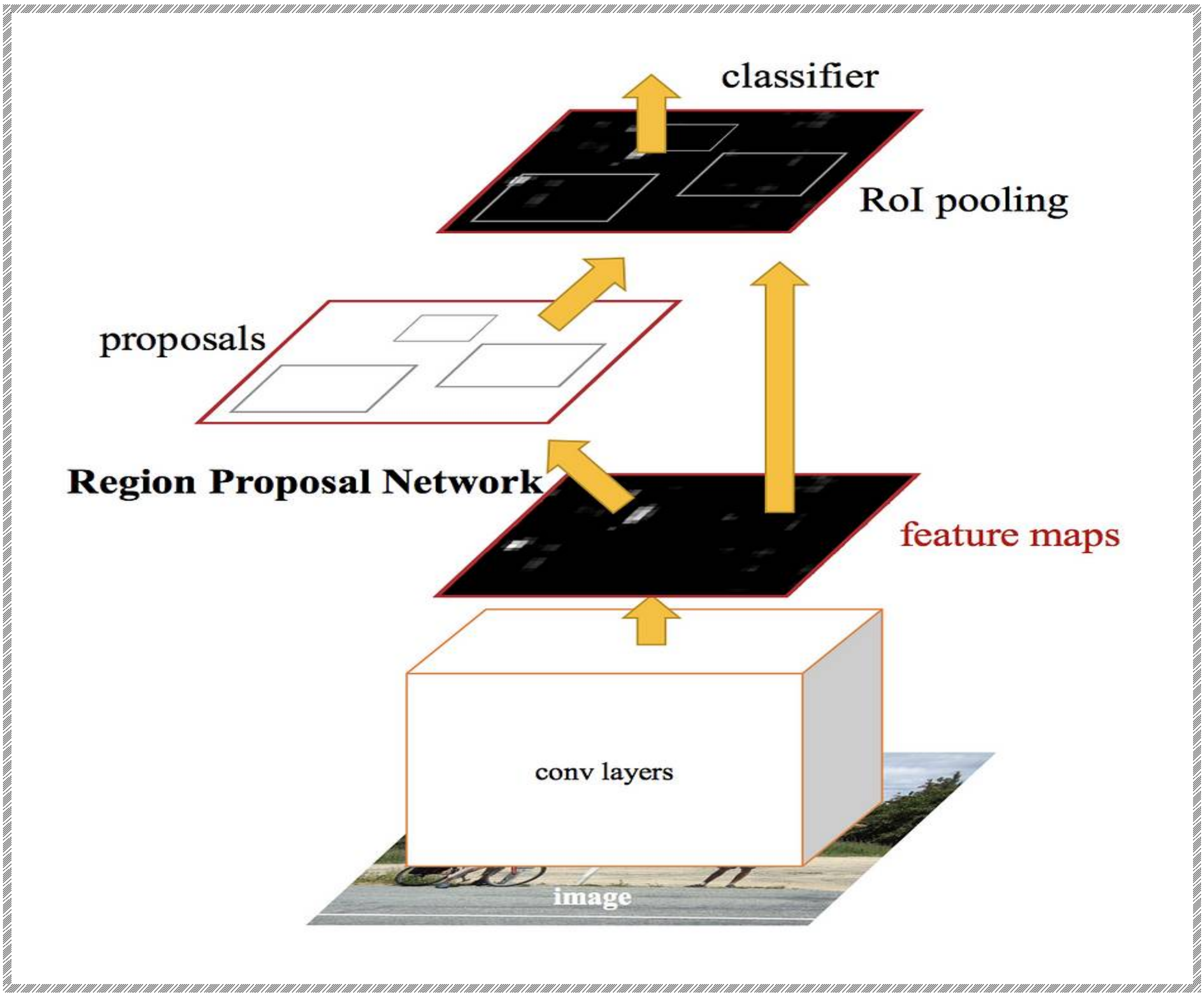
Standard Convolution (Top) Atrous Convolution (Bottom) Feature Extraction



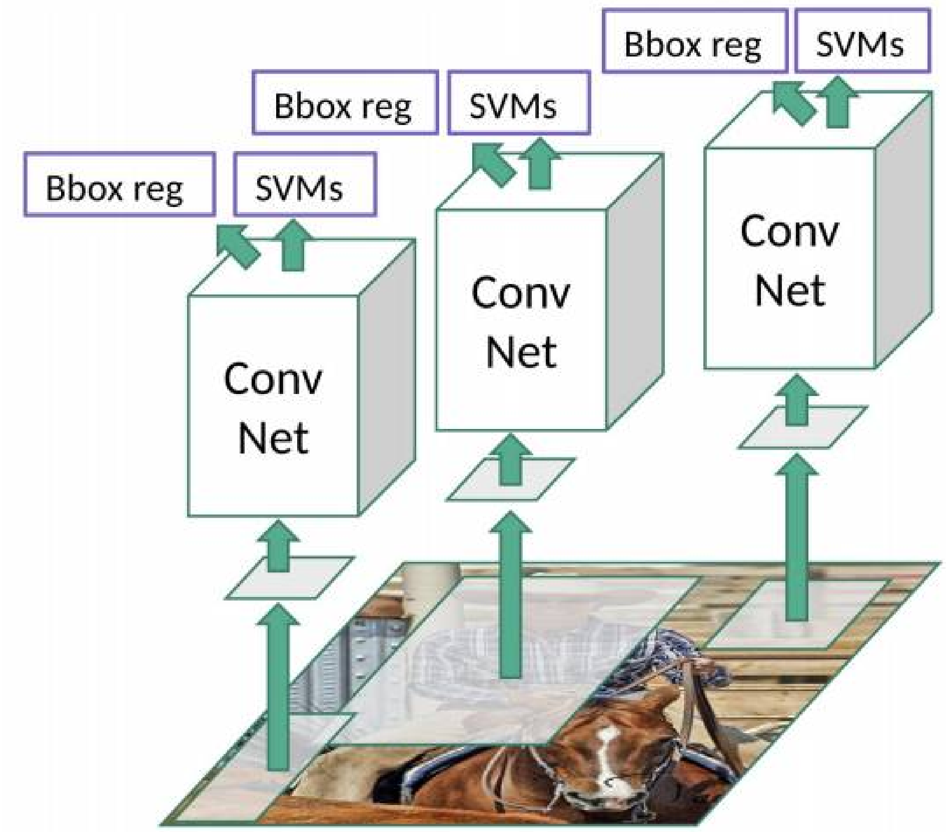
(a) Sparse feature extraction

(b) Dense feature extraction

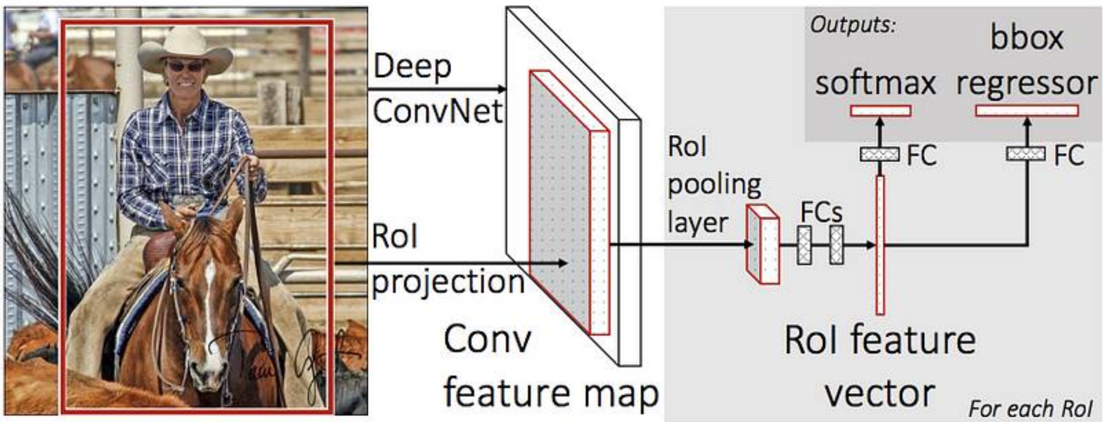

Faster R-CNN Overall Architecture
Image
Stage 1
Stage 2


classifier
RoI pooling
proposals
Region Proposal Network
feature maps
conv layers
image

R(Regions with)-CNN



Fast R-CNN



V. xAiProbes

for ConvNN, CapsNN

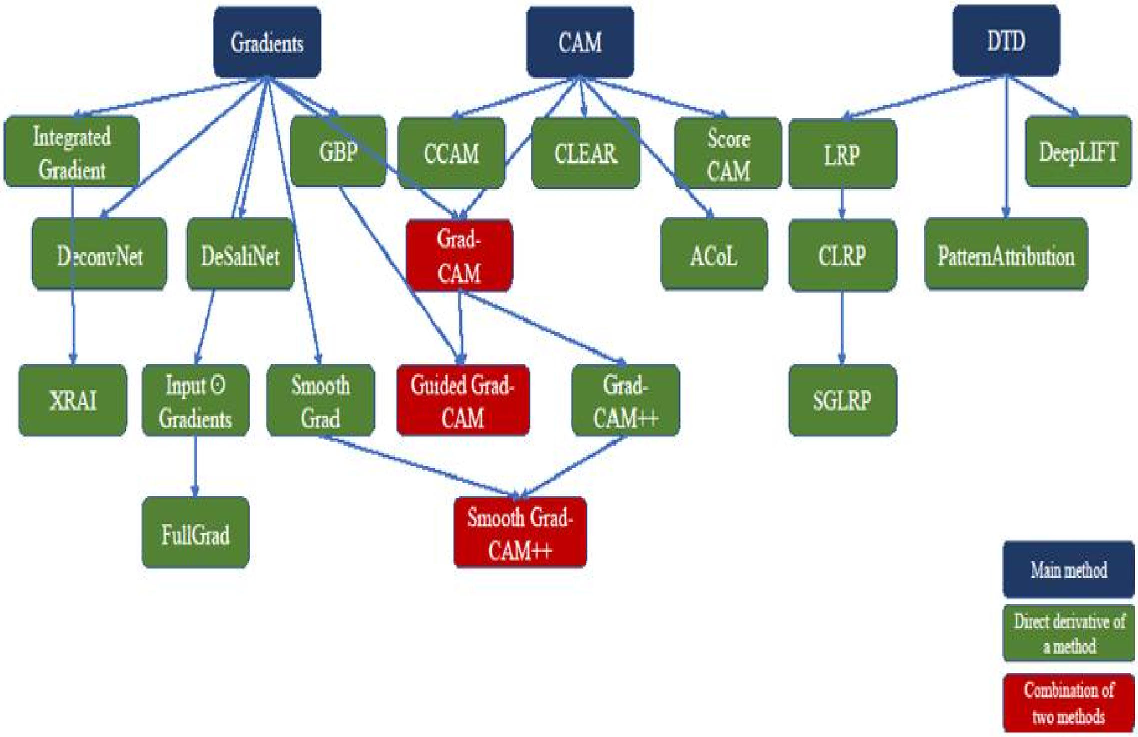
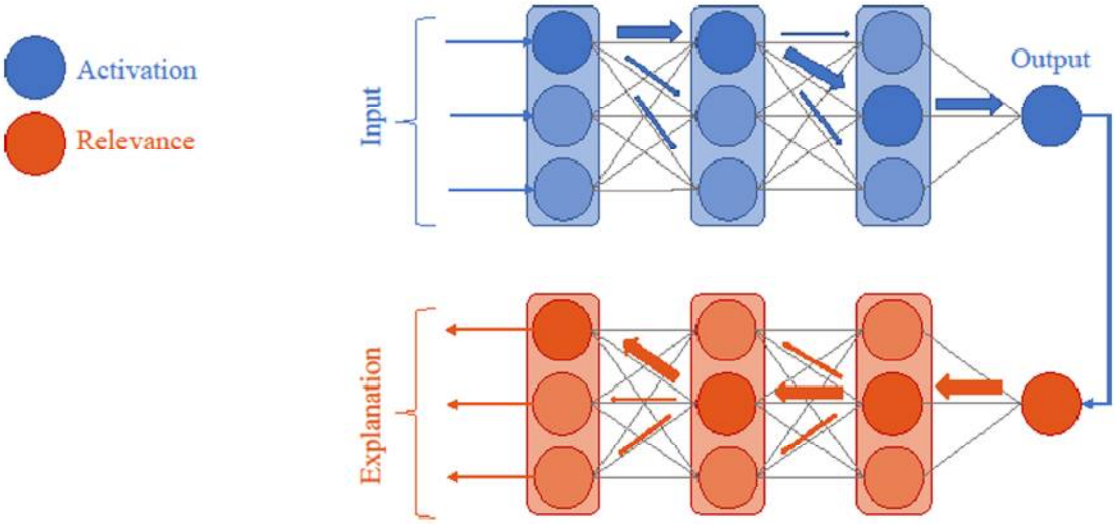
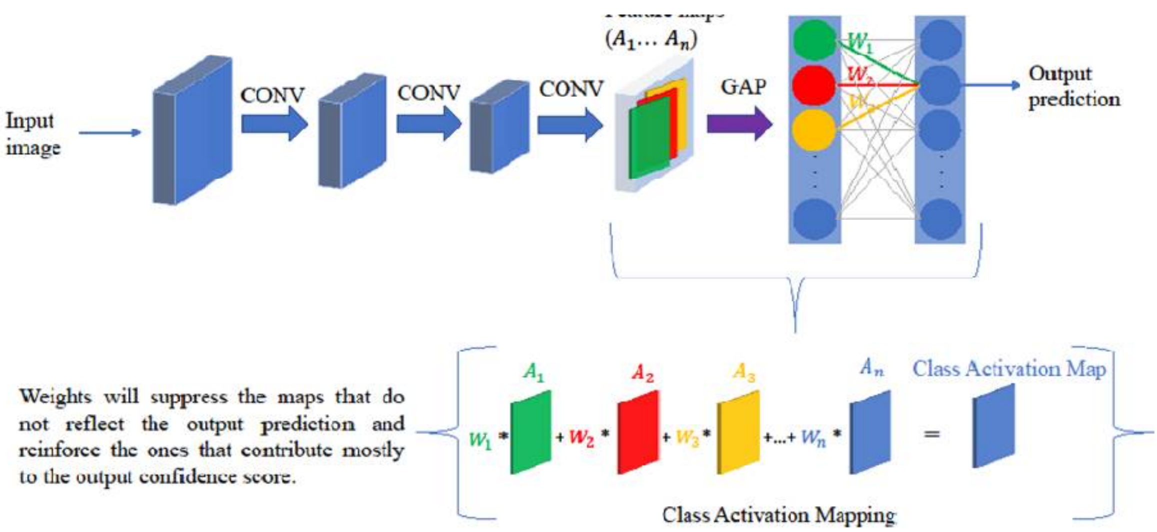


Chart of post-hoc visualisation techniques

Ref	Displays 73 (2022) 102239; /doi.org/10.1016/j.displa.2022.102239
Ti	A review of visualisation-as-explanation techniques for convolutionalneural networks and their evaluation
Au	Elhassan Mohamed , Konstantinos Sirlantzis, Gareth Howells



LRP relevance redistribution technique



Class Activation Map (CAM) generation process