Convolution Neural Networks - CNN
Part II

Deep Neural Network

Session 21

Pramod Sharma
pramod.sharma@prasami.com

2

Agenda

- Introduction
- Classical Networks
- Network in Network
- Inception Network
- Transfer Learning
- Object Detection

11/22/2025

pra-sâmi

3

Classic Networks

- LeNet-5
- AlexNet
- VGG

4

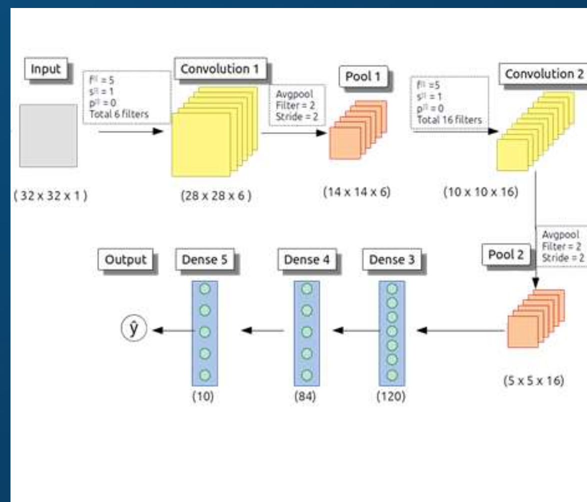
SOTA Networks

- ResNet
- DenseNet
- Unet

5

LeNet - 5

- LeCun et. Al., 1998 – Gradient based learning applied to document recognition
- A number of Conv and Pool layers stacked together
- Followed by dense layers
- Softmax activation to predict probabilities
- Original LeNet -5 had 32 x 32 x 1 images and was used for handwriting dataset
- Had Average Pooling and used Tanh activation

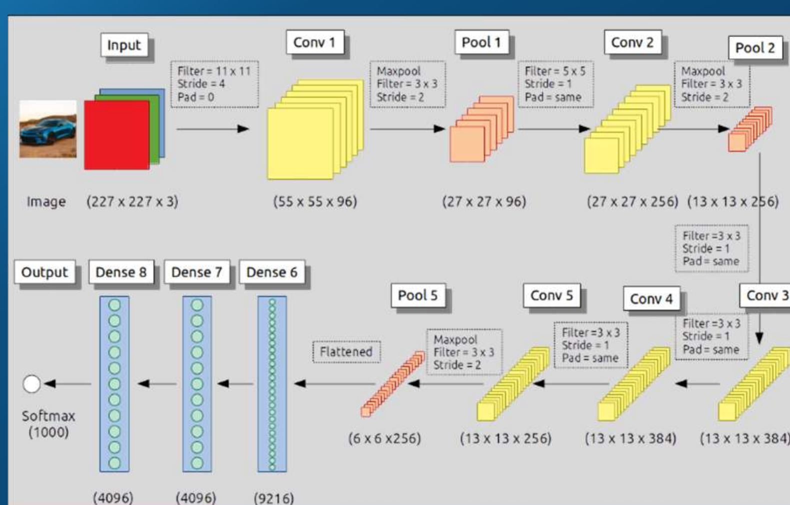


11/22/2025

pra-sâmi

6

AlexNet



- Alex net was considered very deep back then
  - It used ReLU
- First one to use 'Local Response Norm' and prove that it's not a good idea

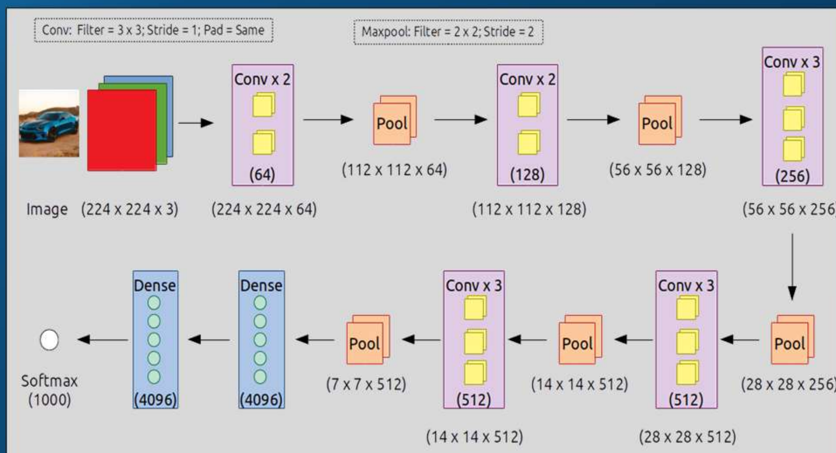
11/22/2025

pra-sâmi

7

VGG-16

- Standardized the parameters
- It had 16 layers with weights
- Uniformity made it very attractive for researchers



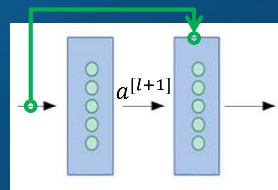
8

Those were Classical Networks

9

Residual Block

$a^{[l]}$



$a^{[l+2]}$

Short cut/ Skip Connection

$a^{[l+2]} = g(z^{[l+2]} + a^{[l]})$

$a^{[l]} \rightarrow Aggregation \rightarrow ReLU \rightarrow a^{[l+1]} \rightarrow Aggregation \rightarrow ReLU \rightarrow a^{[l+2]}$

$z^{[l+1]} = a^{[l]} * W^{[l+1]} + b^{[l+1]}$

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

$z^{[l+2]} = a^{[l+1]} * W^{[l+2]} + b^{[l+2]}$

$a^{[l+2]} = g(z^{[l+2]})$

Main Path

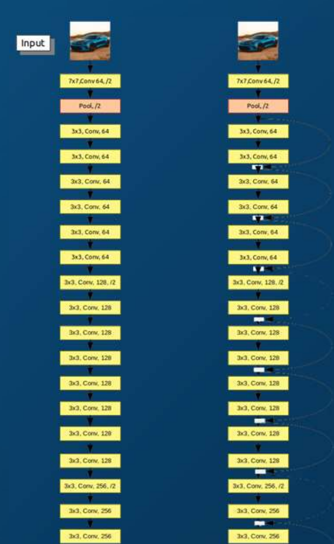
11/22/2025

pra-sâmi

10

ResNet

- Deeper networks had vanishing gradient problems
- Most networks resulted in higher errors and lesser accuracy as the depth increased
- ReLU activations solved it to some extent
- As networks became deeper (more layers), it lead to higher classification error
- It was not due to over-fitting as, as training errors were higher too!
- Expectation was that network with more layers should be as good if not better!
- Deeper networks are not good handling identity function (Output same as input)
- ResNet Architecture addressed it



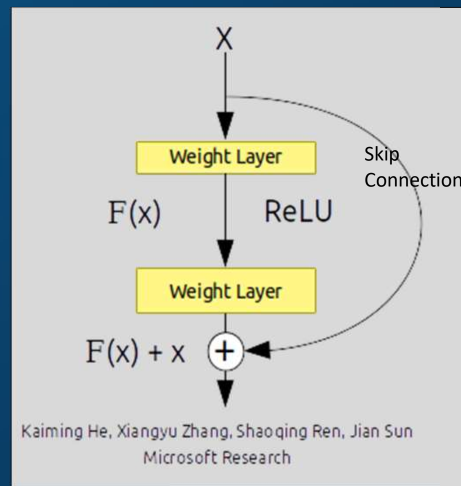
11/22/2025

pra-sâmi

11

ResNet – Building block

- For normal convolutions:
  - F(a) = F(a) + a
- In case of Pooling
  - F(a) = F(a) + a . Ws
  - Where Ws is matrix of <previous layer size> x <size of layer L+2>



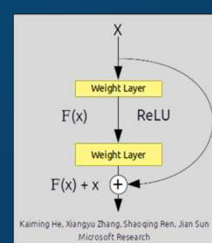
11/22/2025

pra-sâmi

12

ResNet – Building block

- if F(x) becomes zero, it is at least x
  - Relies on making identity function explicit
  - Simply, Input 'x' is processed by two conv. layers as earlier
  - Then 'x' is added to the output before applying ReLU



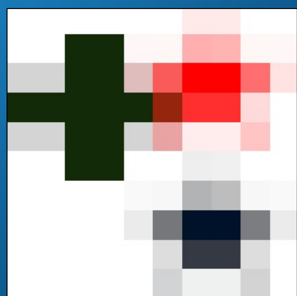
- Thus it is catering to both.
  - Old abstracts are retained and additional abstracts if any are added!
- Early layers are trying to learn some low-level features such as edges, corners etc.,
  - Later layers are focusing on high level abstractions such as wheels, wind shield, etc…
  - Subsequent layers may degrade or obfuscate these reliable signals
  - ResNet architecture gives the network a more explicit codes the output of the block defaulting to its input x, if F(x) is zero
- In short, don't forget what you have already learnt, at least….

11/22/2025

pra-sâmi

13

1 x 1 Convolution – Network in Network



255	255	255	255	255	255	255	255	255	255
255	255	18	18	255	255	255	255	255	255
211	211	18	18	223	249	255	255	255	255
18	18	18	18	18	150	255	255	255	255
211	211	18	18	211	232	255	255	255	255
255	255	18	18	255	255	238	240	255	255
255	255	255	255	249	247	178	189	247	249
255	255	255	255	235	118	0	0	124	235
255	255	255	255	255	221	53	53	221	255
255	255	255	255	255	210	239	239	211	255

$\star \boxed{2} \Rightarrow$

510	510	510	510	510	510	510	510	510	510
510	510	36	36	510	510	510	510	510	510
422	422	36	36	446	498	510	510	510	510
36	36	36	36	36	300	510	510	510	510
422	422	36	36	422	464	510	510	510	510
510	510	36	36	510	510	476	480	510	510
510	510	510	510	498	494	356	378	494	498
510	510	510	510	470	236	0	0	248	470
510	510	510	510	510	442	106	106	442	510
510	510	510	510	510	420	478	478	422	510

Not so obvious in a single layer...

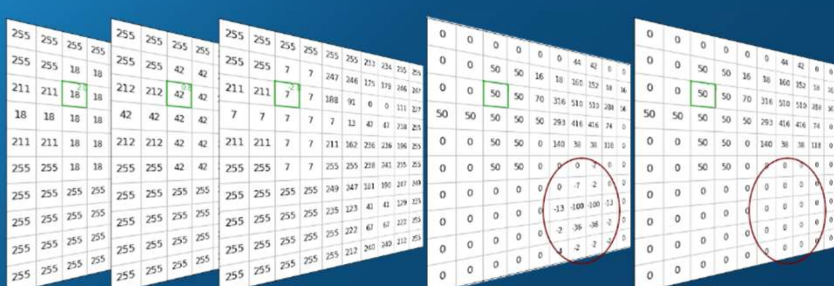
Lin et al., 2013 Network in Network

11/22/2025

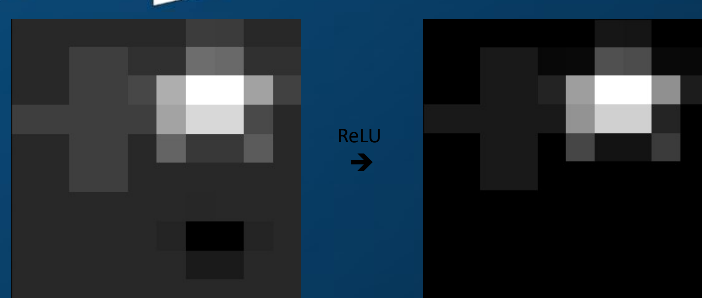
pra-sâmi

14

1 x 1 Convolution – multiple layers



Nonlinearity is introduced over multiple layers...

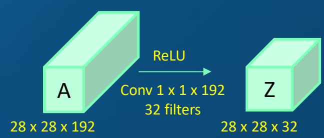


11/22/2025

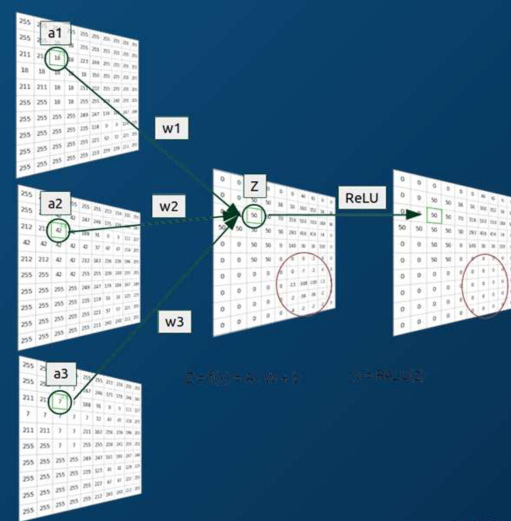
15

Network in Network

- Another advantage is that it can be used to reduce dimensions
- Thus allowing us to shrink or expand or keep the averages of the channels,
- Of course, it permits us to add non-linearity



Network in Network



11/22/2025

16

Inception Network - Acknowledgements

- Takes inspiration from movie “Inception”… “We need to go deeper”

Going deeper with convolutions

Christian Szegedy
Google Inc.

Wei Liu
University of North Carolina, Chapel Hill

Yangqing Jia
Google Inc.

Pierre Sermanet
Google Inc.

Scott Reed
University of Michigan

Dragomir Anguelov
Google Inc.

Dumitru Erhan
Google Inc.

Vincent Vanhoucke
Google Inc.

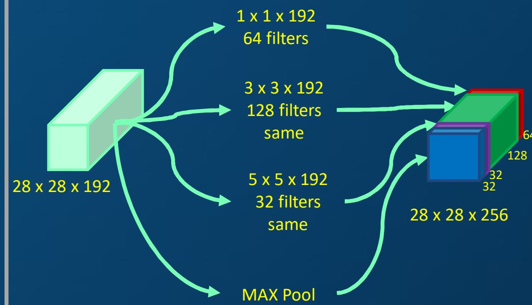
Andrew Rabinovich
Google Inc

11/22/2025

17

Inception Network – Building Block

- We are always faces with challenge of selecting the filters, pooling and their respective sizes
- Engineers though of a solution of adding all together and let the network decide what works best
- Enter combination of filters
- It has problem of computational cost
- Note that you have to use Padding with stride of one in the MaxPool layer to match the dimensions



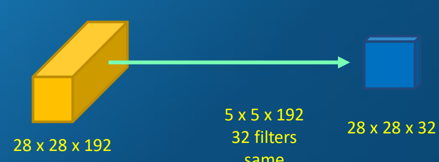
11/22/2025

pra-sâmi

18

Inception Network – Computational Cost

- Let's take one filter as an example



- Overall computations:
  - 5 x 5 x 192 x 28 x 28 x 32 = 120,422,400
  - Say = 120 million
- A very computationally heavy operation

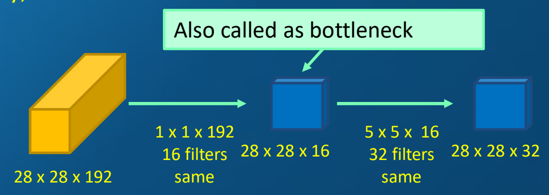
11/22/2025

pra-sâmi

19

Inception Network – Computational Cost

- Alternatively,



- Overall computations

  = {(1 x 1 x 192) x (28 x 28 x 16)} + {(5 x 5 x 16) x (28 x 28 x 32)} = 2,408,448 + 10,035,200 = 12,443,648 Say = 12 million

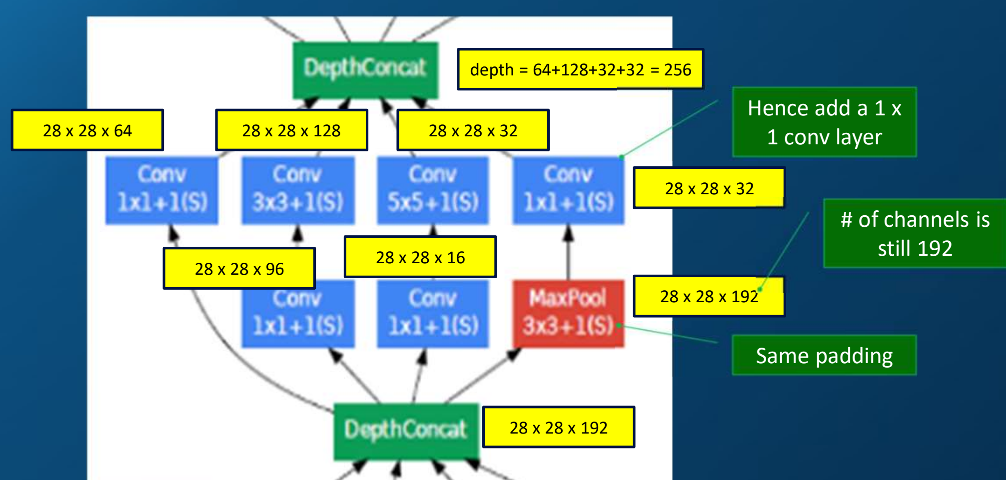
- Reduced by 10 times!
- Caution: the size of bottleneck layer to be chosen carefully too much shrinking may harm the performance
- Also Helping us in reducing the number of channels!

11/22/2025

pra-sâmi

20

Inception Module

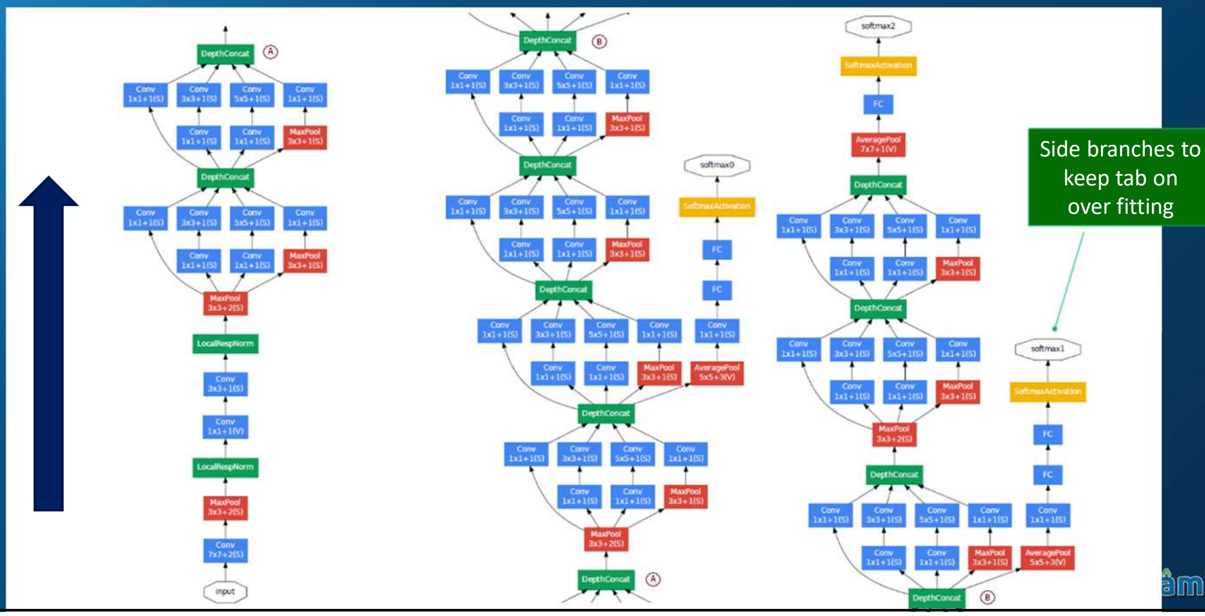


11/22/2025

pra-sâmi

21

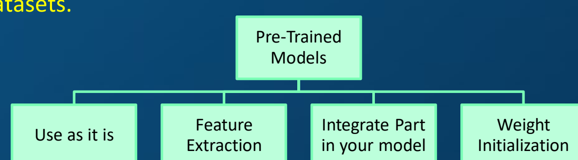
Complete Network - GoogLeNet



22

Transfer Learning

- May take days or even weeks to train on very large datasets.
- In AI and ML world, its customary to publish one's work in open source
  - Open source large datasets, pre-trained models and weights available



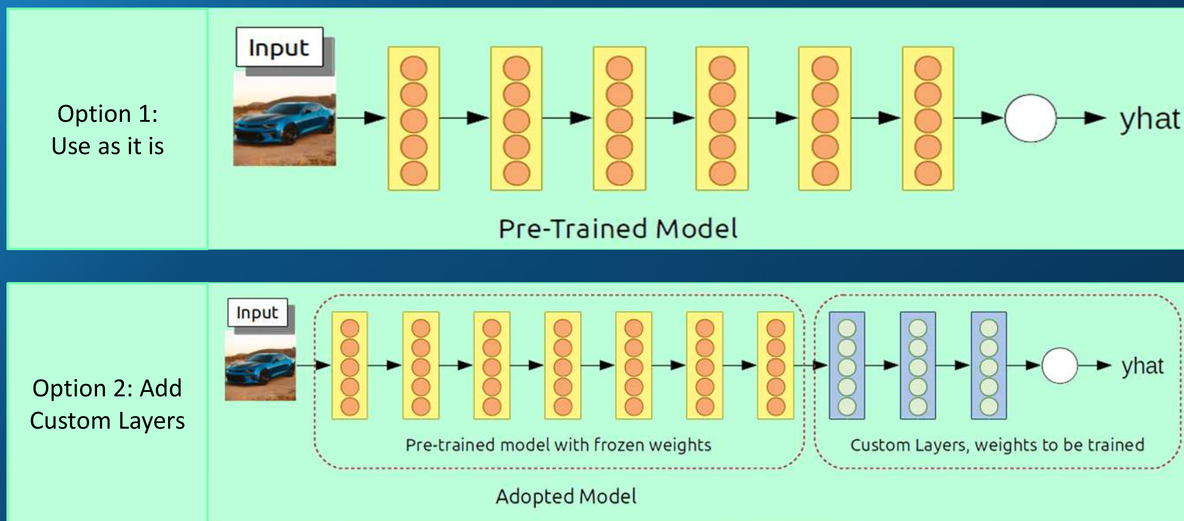
- Especially helpful in cases where we have limited pictures
- The models are complex and have multiple classes
  - Image net → 1000 classes (ImageNet Large Scale Visual Recognition Challenge, or ILSVRC or ImageNet)
  - A range of high-performing models available
- Use top performing model directly, or integrated into a new model
- Of course with some modifications to last few layers
- Most pre-trained models APIs are available

11/22/2025

pra-sāmi

23

Transfer Learning Options

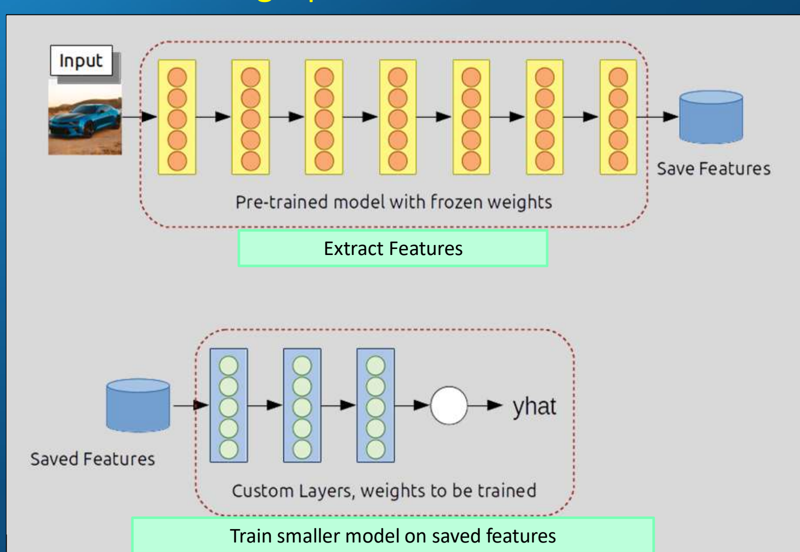


11/22/2025

pra-sâmi

24

Transfer Learning Option : 3



- Feel free to experiment by training frozen layers as well!
- If you have more data more layers could be used.
- It there is lots and lots of data, use this model to initialize and train all the weights
- These models are so well trained, it advantage to use existing weights!!

11/22/2025

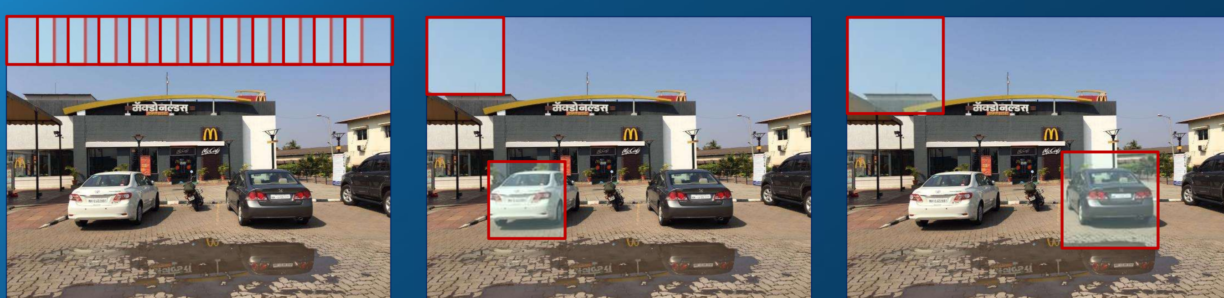
pra-sâmi

25

Object Localization

26

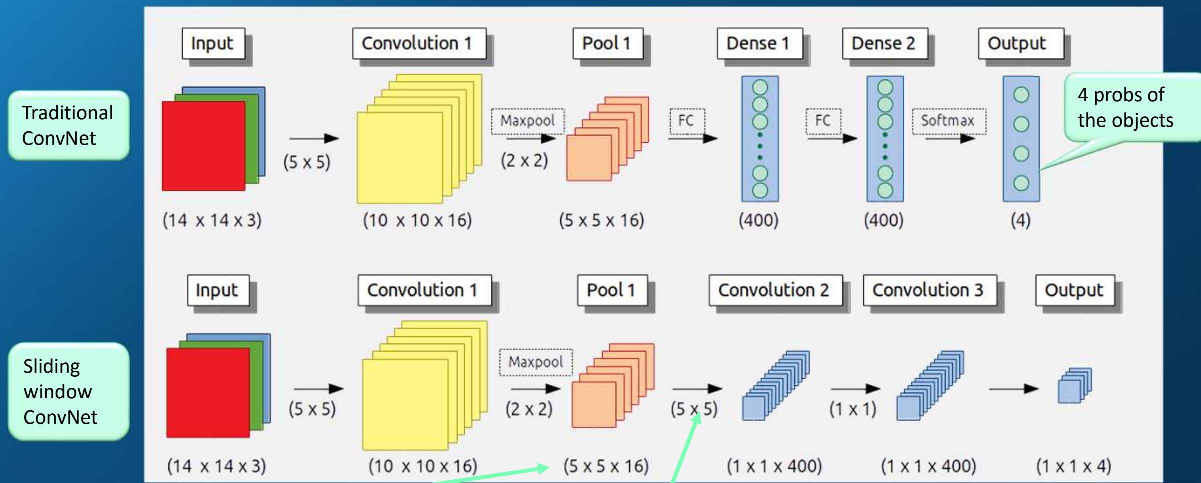
Sliding Window Detection



- Analyzing for all these windows is resource consuming....
- We can convert logic to some what similar to convolutional networks and achieve better efficiencies.

27

Sliding Window Convolution way…

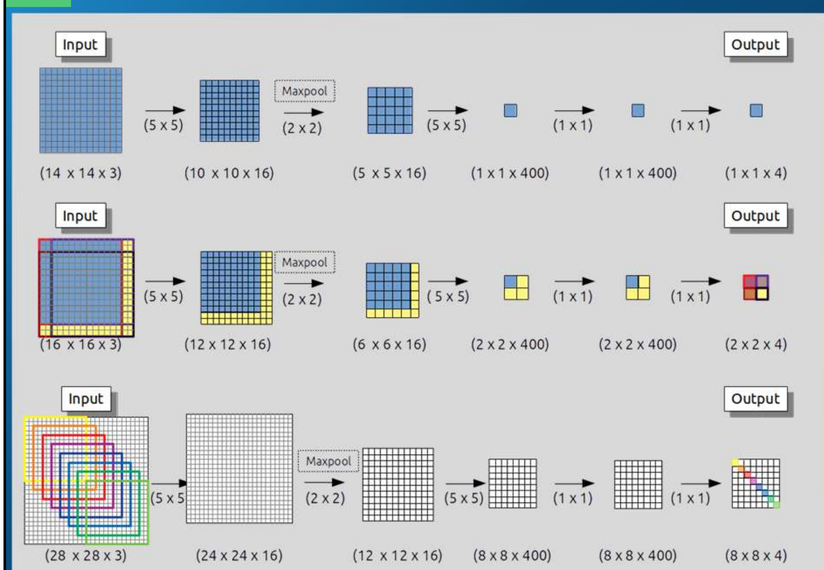


- Each 5 x5 x 16 layer is applied 5 x 5 x 16 filter and some activation to get 1 x 1 x 400 nodes
- Mathematically its same as fully connected layer!!

11/22/2025

28

Convolution Implementation of Object Detection

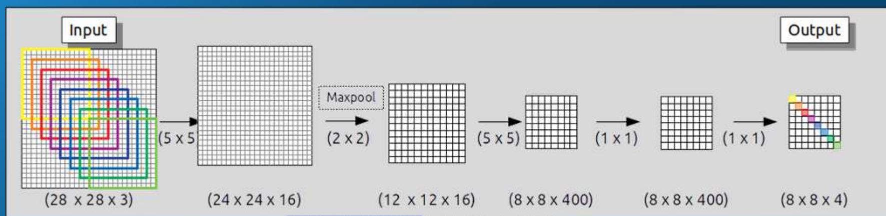


- The computations are shared across the windows
- Results of each of region (1 x 1) are available using the convolution
- For bigger image size, output also increases
- This is telling us if in respective region, target object is present or not!

OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks

29

Convolution instead of Sliding Window.



- Hence, by moving 14x14 region over the entire image we would know location of the region with maximum probability of containing a car.
- Issue remains that size of bounding box (region) is predefined
- Chances are that it is not very accurate.

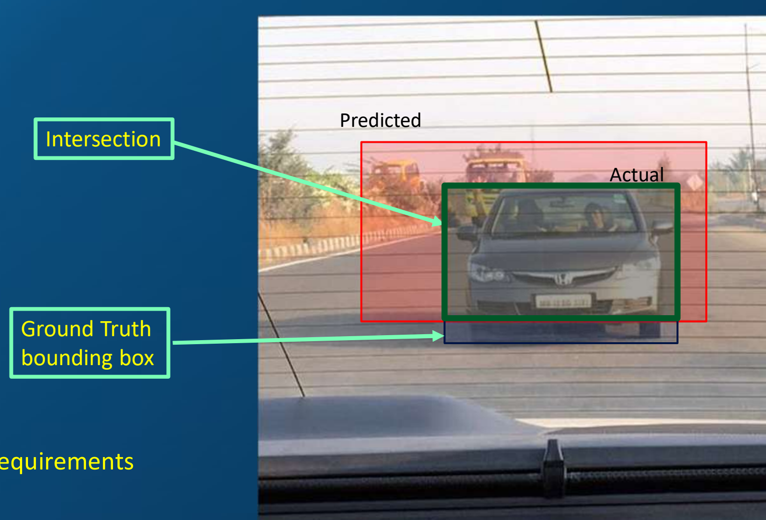
11/22/2025

pra-sâmi

30

Intersection over Union - IoU

- IoU = $\frac{Area\ of\ intersection}{Area\ of\ union}$



- IoU > 0.5 Acceptable
- IoU = 1.0 Perfect
- IoU > 0.6 for little stringent requirements

11/22/2025

pra-sâmi

31

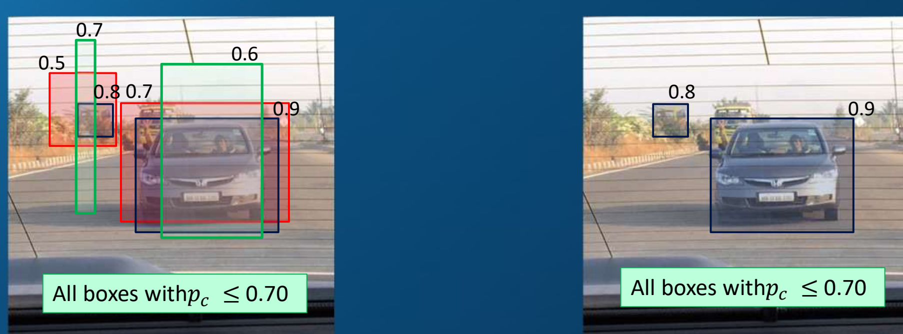
Non Max Suppression

- Multiple windows will detect objects
- In fact, every window will have some probability of having a car
- First reject all windows where probability is less than some predefined level say $p_c \leq 0.70$
- Thereafter, suppress all rectangle where IOU is above some limit (0.5) ➔ Blue rectangles are retained

11/22/2025

32

Non Max Suppression



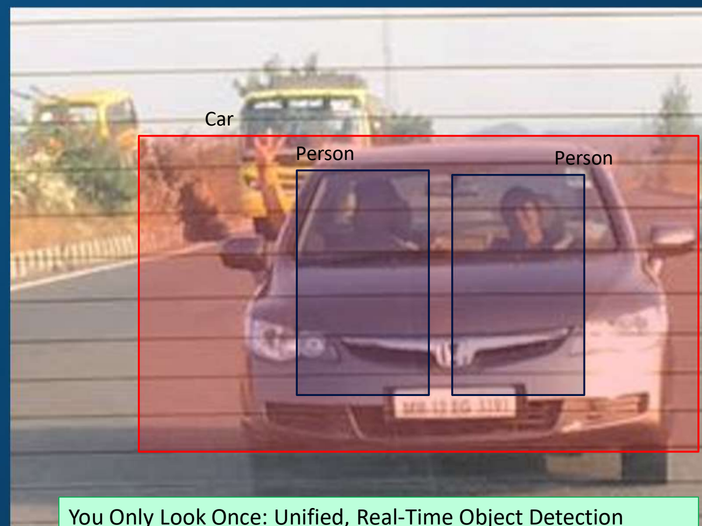
- Pick the box with highest C_n for that class
  - Discard any box with high IOU with this box
- If you are trying to identify multiple objects, say Cars, Pedestrians, Motorcycles output vector will have more dimensions
  - $p_c, C_1, C_2, C_3, x_1, y_1, h_1, w_1, \ldots.$

11/22/2025

33

Anchor Boxes

- Any anchor can be defined with
  - Presence : in any object is present in the anchor
  - Box location: mid point (x, y), height and width of the box
  - Class: What class is present- Car/person/motorcycle
- Fully defined anchor for three class
  - $p_c, b_x, b_y, b_h, b_w, c_1, c_2, c_3 \implies 8$ values
- $$\hat{y} = \left\{ \begin{array}{c} Presence \\ Box\ location \\ Class \end{array} \right\} = \left\{ \begin{array}{c} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{array} \right\}$$



You Only Look Once: Unified, Real-Time Object Detection
Joseph Redmon , Santosh Divvala , Ross Girshick , Ali Farhad

11/22/2025

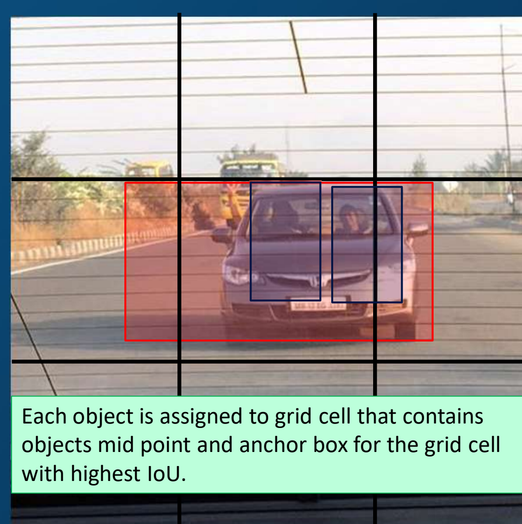
pra-sâmi

34

YOLO – You Only Look Once - Training and Data Preparation

- Assume our image is divided in 3 x 3 grid
  - Real implementation : 16 x 16 or 19 x 19
- Assume we have only two anchor box per cell
  - i.e. not more than two items in a cell
- Thus ŷ will be 3 x 3 x 16 or 3 x 3 x 2 x 8
- $$\hat{y} = \left\{ \begin{array}{c} p_{c1} \\ b_{x1} \\ b_{y1} \\ b_{h1} \\ b_{w1} \\ c_{11} \\ c_{21} \\ c_{31} \\ p_{c2} \\ b_{x2} \\ b_{y2} \\ b_{h2} \\ b_{w2} \\ c_{12} \\ c_{22} \\ c_{32} \end{array} \right\} = \begin{array}{cccccc c} 0 & 0 & 0 & & 1 & 1 & 0 \\ - & - & - & & 0.4 & 0.3 & - \\ - & - & - & & 0.5 & 0.4 & - \\ - & - & - & & 0.3 & 0.2 & - \\ - & - & - & & 0.4 & 0.3 & - \\ - & - & - & & 0 & 0 & - \\ - & - & - & & 1 & 1 & - \\ - & - & - & & 0 & 0 & - \\ 0 & 0 & 0 & \cdots & 1 & 0 & 0 \\ - & - & - & & 0.5 & - & - \\ - & - & - & & 0.7 & - & - \\ - & - & - & & 0.2 & - & - \\ - & - & - & & 0.35 & - & - \\ - & - & - & & 1 & - & - \\ - & - & - & & 0 & - & - \\ - & - & - & & 0 & - & - \end{array}$$

Conv Layer Output is 3 x 3 x 2 x 8



Each object is assigned to grid cell that contains objects mid point and anchor box for the grid cell with highest IoU.

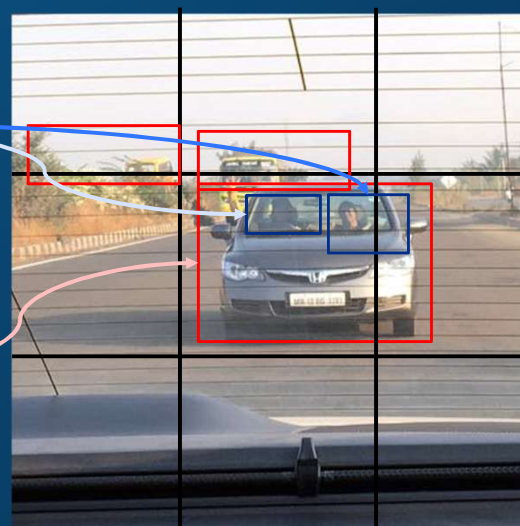
11/22/2025

pra-sâmi

35

YOLO – You Only Look Once - Predictions

$$\hat{y} = \begin{array}{c} p_{c1} \\ b_{x1} \\ b_{y1} \\ b_{h1} \\ b_{w1} \\ c_{11} \\ c_{21} \\ c_{31} \\ p_{c2} \\ b_{x2} \\ b_{y2} \\ b_{h2} \\ b_{w2} \\ c_{12} \\ c_{22} \\ c_{32} \end{array} = \begin{array}{ccccccc} 0 & 0 & 0 & & 1 & 1 & 0 \\ - & - & - & & 0.4 & 0.3 & - \\ - & - & - & & 0.5 & 0.4 & - \\ - & - & - & & 0.3 & 0.2 & - \\ - & - & - & & 0.4 & 0.3 & - \\ - & - & - & & 0 & 0 & - \\ - & - & - & & 1 & 1 & - \\ - & - & - & & 0 & 0 & - \\ 0 & 0 & 0 & \cdots & 1 & 0 & 0 \\ - & - & - & & 0.5 & - & - \\ - & - & - & & 0.7 & - & - \\ - & - & - & & 0.2 & - & - \\ - & - & - & & 0.35 & - & - \\ - & - & - & & 1 & - & - \\ - & - & - & & 0 & - & - \\ - & - & - & & 0 & - & - \end{array}$$



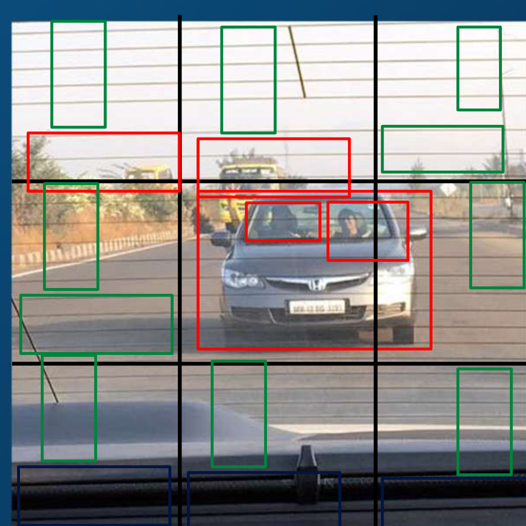
11/22/2025

pra-sâmi

36

YOLO – You Only Look Once - Predictions

- Get bounding boxes for each of the cells…
- Bounding boxes may overflow
  - We have not given any grid locations
- Except for those in red every one else would have low probability
- Keep Red ones and remove others.



11/22/2025

pra-sâmi

37

YOLO8 – Most Stable Version (2023)

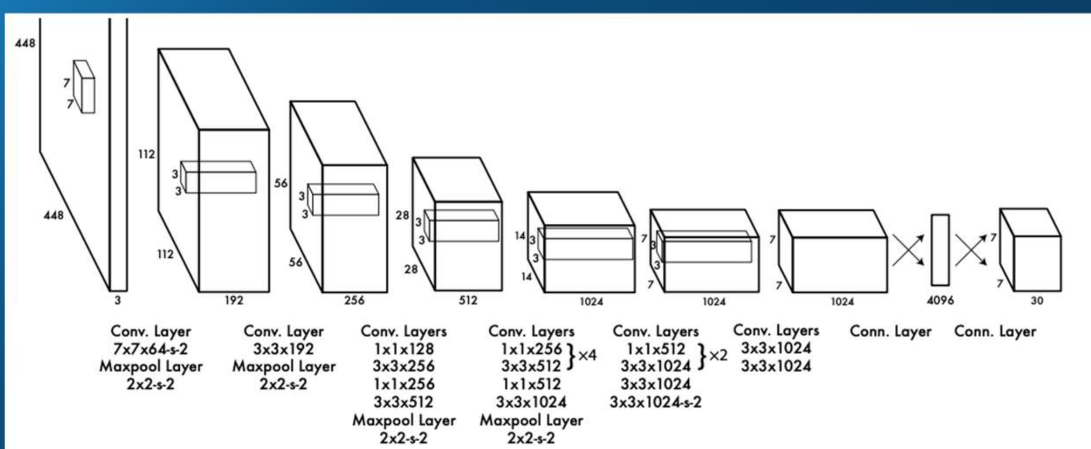
- The input image is resized to 640 × 640, which is the standard recommended size for YOLOv8 models.
- A single end-to-end convolutional neural network processes the input image.
- The model filters predictions using confidence thresholds and non-maximum suppression (NMS).
- Output: an anchor-free detection head, generating bounding boxes, class probabilities, and objectness scores across multiple feature scales (not a fixed 7×7×30 tensor previous version).
- SiLU (Swish) activation function across most layers for improved gradient flow and performance.
- The final detection layer applies linear outputs for box coordinates and sigmoid activations for objectness and class probabilities.
- YOLOv8 does not use Sum of Squares Error; instead, it uses an advanced composite loss function
- Typical training hyperparameters include batch size = 16–64, momentum = 0.937, and weight decay = 0.0005 (as per Ultralytics defaults).
- Bye… Bye… dropout! it uses architectural regularization and large-scale augmentation.
- Extensive data augmentation, including random scaling, translation, flipping, mosaic augmentation, HSV color augmentation, and more…

11/22/2025

pra-sâmi

38

YOLO V1



11/22/2025

pra-sâmi

39

Now Darknet -53

- Starting YOLO version 3.0 started using Darknet-53
  - Other networks can also be used

- loss = loss1+loss2+loss3

$$loss_1 = -\sum_{i=0}^{S^2}\sum_{j=0}^{B} W_{ij}^{obj}[\hat{C}_i^j \log\left(C_i^j\right) + (1-\hat{C}_i^j)\log\left(1-C_i^j\right)] - \lambda_{noobj}\sum_{i=0}^{S^2}\sum_{j=0}^{B}(1-W_{ij}^{obj})[\hat{C}_i^j \log\left(C_i^j\right) + (1-\hat{C}_i^j)\log\left(1-C_i^j\right)]$$

$$loss_2 = -\sum_{i}^{s^2}\sum_{j}^{B} W_{ij}^{obj}\sum_{c=1}^{C}[\hat{p}_i^j(c)\log(p_i^j(c)) - (1-\hat{p}_i^j(c))\log(1-p_i^j(c))]$$

$$loss_3 = 1 - IOU + \frac{\rho^2\left(b, b^{gt}\right)}{c^2} + \frac{16}{\pi^4}\frac{\left(\arctan\frac{w^{gt}}{h^{gt}} - \arctan\frac{w}{h}\right)^4}{1 - IOU + \frac{4}{\pi^2}\left(\arctan\frac{w^{gt}}{h^{gt}} - \arctan\frac{w}{h}\right)^2}$$

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Table 1. **Darknet-53.**

11/22/2025

pra-sâmi

40

R-CNN

- RCNN has nothing to do with RNN (Recurrent neural networks).

- R-CNN is short for "Region-based Convolutional Neural Networks."
  - Takes in input image
  - Extracts around 2000 bottom-up region proposals
  - Computes features for each proposal using a large convolutional neural network (CNN)
  - Classifies each region using class-specific linear SVMs

- This network was slow, hence
  - Spate of other proposals are going on
  - Fast RCNN
    - Convolutional implementation of sliding window
  - Faster R-CNN
    - Use Convolutional Network to propose regions

11/22/2025

pra-sâmi

41

Dense Net

pra-sâmi

42

Acknowledgement

Gao Huang
Cornell University

Zhuang Liu
Tsinghua

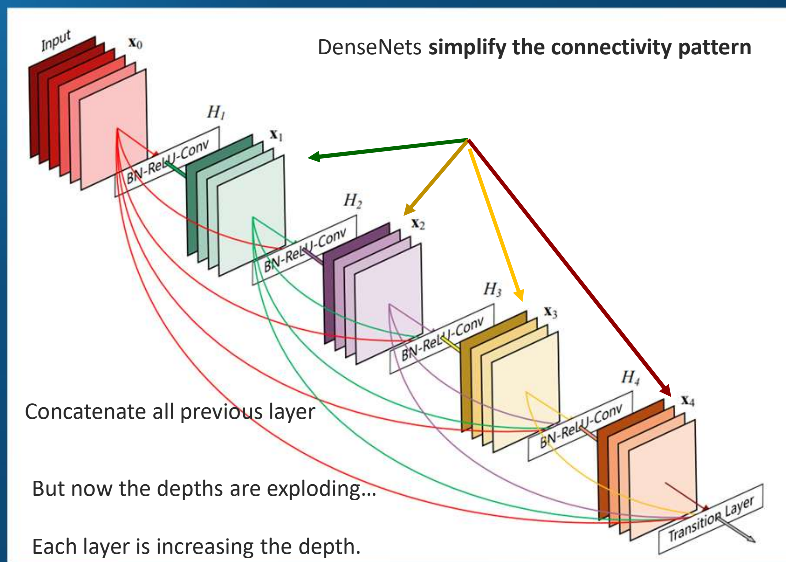
Laurens van der Maaten
Facebook AI Research

Kilian Q. Weinberger
Cornell University

pra-sâmi

43

A 5-layer dense block with a growth rate of k = 4.

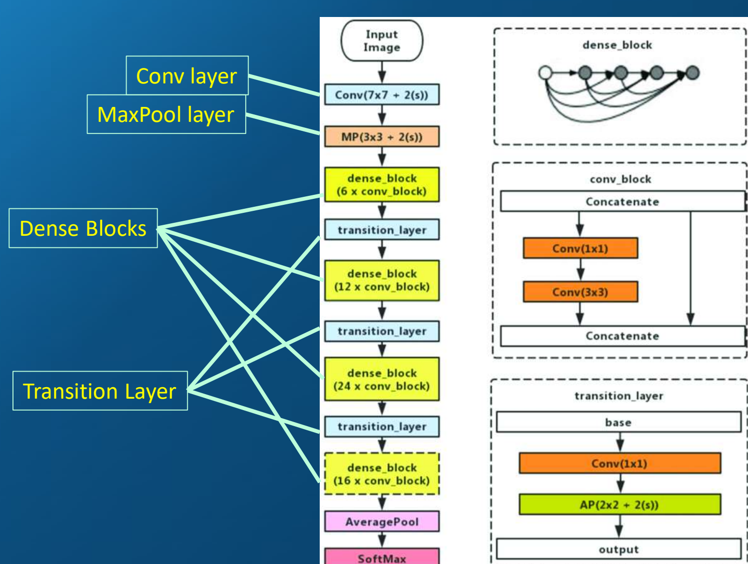


11/22/2025

pra-sâmi

44

DenseNet 121 Architecture



11/22/2025

pra-sâmi

45

DenseNet Architectures

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	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 6$
Transition Layer (1)	56 × 56	1 × 1 conv			
	28 × 28	2 × 2 average pool, stride 2			
Dense Block (2)	28 × 28	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 12$
Transition Layer (2)	28 × 28	1 × 1 conv			
	14 × 14	2 × 2 average pool, stride 2			
Dense Block (3)	14 × 14	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 24$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 48$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 64$
Transition Layer (3)	14 × 14	1 × 1 conv			
	7 × 7	2 × 2 average pool, stride 2			
Dense Block (4)	7 × 7	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 16$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 48$
Classification Layer	1 × 1	7 × 7 global average pool			
		1000D fully-connected, softmax			

46

Why Change?

- Traditional feed-forward neural networks connect the output of the layer to the next layer using:
  - Activations (a^[l]) = g (a^[l−1] * W^[l] + b^[l])
- ResNet modified them a bit:
  - Activations (a^[l]) = g (a^[l−1] * W^[l] + b^[l] + a^[l−2])
- DenseNets require fewer parameters than an equivalent traditional CNN
- Some variations of ResNets have proven that many layers are barely contributing and can be dropped
- Inception Nets have proven that it’s a good idea to concatenate layers
- Vanishing Gradients were always problems
  - In DenseNets each layer has direct access to the gradients from the loss function and the original input image

47

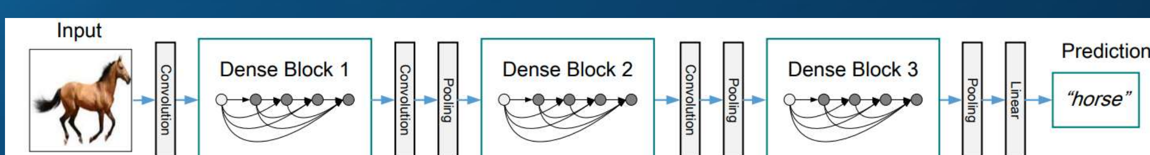
DenseNets

- DenseNets : do not sum the output feature maps of the layer with the incoming feature maps but concatenate them:
  - Activations ($a^{[l]}$) = g ([$a^{[0]}$, $a^{[1]}$, $a^{[2]}$, …, $a^{[l-2]}$, $a^{[l-1]} * W^{[l]}$] + $b^{[l]}$)
- But Activations between various layers would have different shape
  - To solve, DenseNets divide them in blocks
  - Shape remain same in one DenseBlock
- Transition Layers: Layers in-between Dense Layers changing dimensions from one block to another block:
  - Apply 1 x 1, pooling, BatchNorm etc.

48

DenseNets

- Every layer has access to its preceding feature maps
  - i.e. to the collective knowledge
  - Each layer is then adding a new information
- DenseNet layers are very narrow (e.g., 12 filters per layer)
  - Adding only a small set of feature-maps to the “collective knowledge” of the network
  - Keep the remaining feature-maps unchanged
  - The final classifier makes a decision based on all feature-maps in the network



49

Type of DenseNets

- DenseNets-B
  - Regular DenseNets that take advantage of 1x1 convolution to reduce the feature maps size
  - Then apply the 3x3 convolution
  - B stands for bottleneck

- DenseNets-BC
  - Another little incremental step to DenseNets-B, to reduce the number of output feature maps
  - The compression factor (theta) determines the reduction.
  - Instead of having m feature maps at a certain layer, we will have theta*m.
  - Theta is in the range [0–1].
  - DenseNets will remain the same when theta=1, and will be DenseNets-B otherwise.

11/22/2025

pra-sâmi

50

Reflect…

- Which of the following is true about AlexNet?
  - a) It uses 15 layers including fully connected layers
  - b) It introduced the concept of Residual Learning
  - c) It was the first CNN to win the ImageNet Large Scale Visual Recognition Challenge (ILSVRC)
  - d) It uses a 5x5 kernel in the first convolutional layer

- Answer: c) It was the first CNN to win the ImageNet Large Scale Visual Recognition Challenge (ILSVRC)

- What is the key innovation introduced by ResNet?
  - a) Use of deeper convolution layers
  - b) Use of 1x1 convolution kernels
  - c) Introduction of skip connections (residual connections)
  - d) Global average pooling for dimensionality reduction

- Answer: c) Introduction of skip connections (residual connections)

- Which of the following is true about ImageNet?
  - a) It is a dataset consisting of 10 million images
  - b) It contains over 22,000 object categories
  - c) It focuses on medical image segmentation
  - d) It contains only grayscale images

- Answer: b) It contains over 22,000 object categories

- What is the primary characteristic of VGGNet architecture?
  - a) It uses a large number of filters in each layer
  - b) It uses very small 3x3 filters in convolutional layers
  - c) It introduced skip connections
  - d) It employs global average pooling instead of fully connected layers

- Answer: b) It uses very small 3x3 filters in convolutional layers

11/22/2025

pra-sâmi

51

Reflect…

- What was the main innovation introduced by Google's Inception Net?
  - a) Introduction of the "bottleneck" layers
  - b) Use of parallel filters of different sizes in the same layer (Inception module)
  - c) Use of large convolution filters for all layers
  - d) Introduction of Dense blocks
- Answer: b) Use of parallel filters of different sizes in the same layer (Inception module)
- What is the key innovation of Faster R-CNN over Fast R-CNN?
  - a) It uses an RPN (Region Proposal Network) for faster region proposals
  - b) It replaces convolution layers with fully connected layers
  - c) It combines object detection and segmentation in one model
  - d) It removes the need for bounding box regression
- Answer: a) It uses an RPN (Region Proposal Network) for faster region proposals
- How does YOLO differ from traditional object detection models?
  - a) YOLO performs object detection by scanning the image in patches
  - b) YOLO predicts both class probabilities and bounding boxes in a single pass
  - c) YOLO uses a sliding window technique for localization
  - d) YOLO uses fully connected layers for region proposal
- Answer: b) YOLO predicts both class probabilities and bounding boxes in a single pass
- What is the primary characteristic of DenseNet?
  - a) It uses dilated convolutions to increase the receptive field
  - b) It uses skip connections from every layer to every other layer
  - c) It stacks convolutional layers without any pooling layers
  - d) It uses separable convolutions to reduce computational cost
- Answer: b) It uses skip connections from every layer to every other layer

11/22/2025

pra-sâmi

52

Reflect…

- Why does ResNet's performance degrade when the depth of the network increases, without residual connections?
  - a) The network begins to overfit due to an excessive number of parameters
  - b) The gradient vanishes as it backpropagates through the layers, making training ineffective
  - c) It reduces computational complexity too much, leading to poor feature extraction
  - d) It uses too many skip connections, leading to exploding gradients
- Answer: b) The gradient vanishes as it backpropagates through the layers, making training ineffective
- In DenseNet, how does feature reuse occur across layers?
  - a) Each layer receives the feature maps of all preceding layers as input
  - b) Feature maps from selected layers are concatenated to form the final feature vector
  - c) The output of each layer is summed with the output of the previous layer
  - d) DenseNet shares weights between alternate layers to reduce the number of parameters
- Answer: a) Each layer receives the feature maps of all preceding layers as input
- In Faster R-CNN, what is the role of the Region Proposal Network (RPN)?
  - a) To classify the entire image and then crop regions of interest
  - b) To predict regions that are most likely to contain objects, which are then classified by the detection network
  - c) To directly classify each pixel of the image into object categories
  - d) To generate bounding boxes based on edge detection algorithms
- Answer: b) To predict regions that are most likely to contain objects, which are then classified by the detection network
- Which domain is U-Net primarily designed for?
  - a) Object detection
  - b) Natural language processing
  - c) Image segmentation, especially in biomedical images
  - d) Image classification
- Answer: c) Image segmentation, especially in biomedical images

11/22/2025

pra-sâmi

53

11/22/2025

pra-sâmi

EXTRA MATERIAL

pra-sâmi

55

Tips

Data vs. Feature Engineering

- Depending upon size of data, you may need to do feature engineering
- More data, lesser feature engineering

Benchmark Performance

- For benchmarking ➔Ensamble
  - Create multiple model (3 to 25 models)
  - Train them independently
  - Average out the results (ŷ)
- Rarely used in production due to cost considerations
- Multi-crop at the test time

11/22/2025

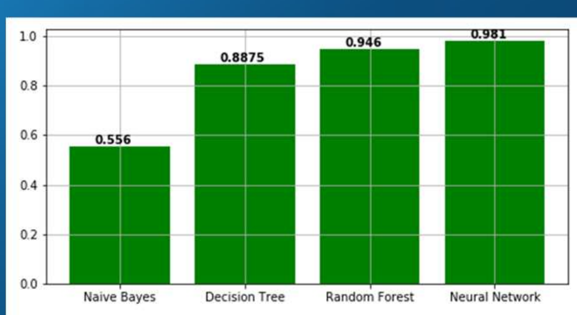
pra-sâmi

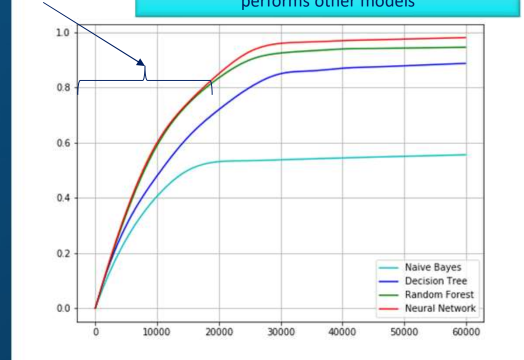
56

Relative performance of models

Small amount of data performance are comparable

As data size grows Neural networks out performs other models





11/22/2025

pra-sâmi