CV Topic 16 Object Detection using Deep Learning

Dr. V Masilamani

masila@iiitdm.ac.in

Department of Computer Science and Engineering
IIITDM Kancheepuram
Chennai-127

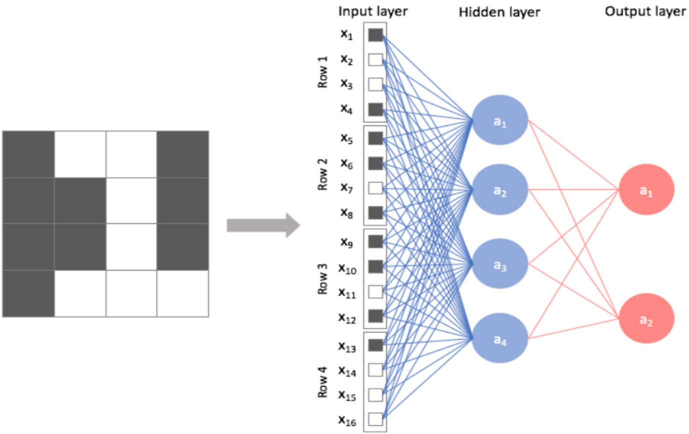
Outline

What is CNN

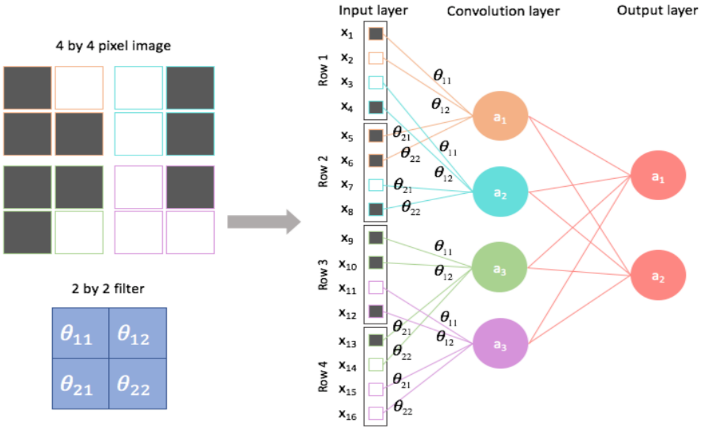
Convolution Neural Network(CNN) is an ANN in which the follwing layers are present

- Convolution Layer(at least 1)
- Fully connected layer
- Pooling Layer
- Un pooling layer

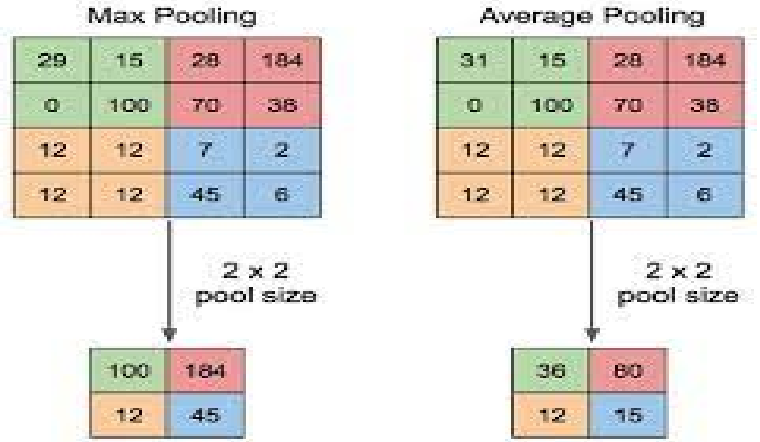
Fully convolutional layer



Convolution layer

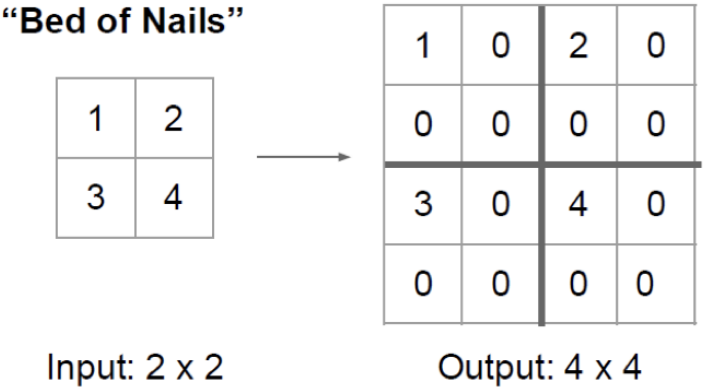


What is CNN (cont.)

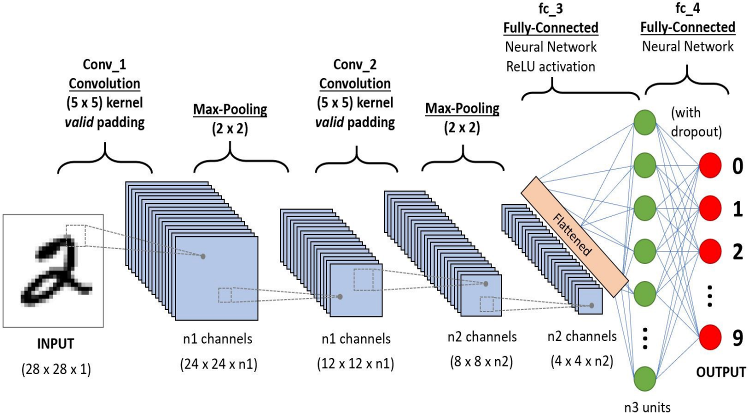


What is CNN (cont.)

An example of Un-pooling

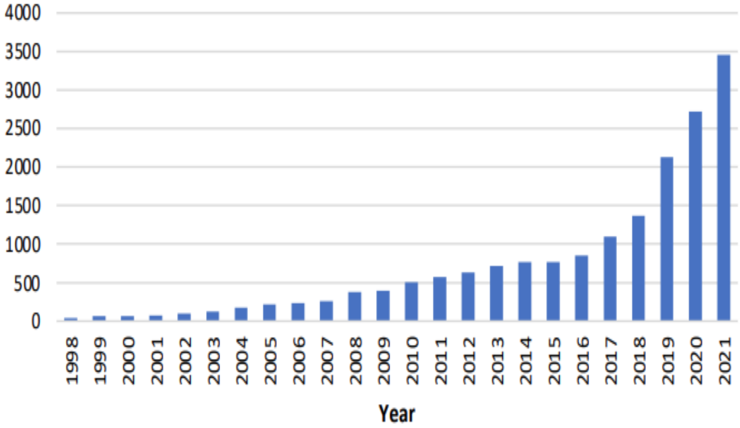


An Example CNN

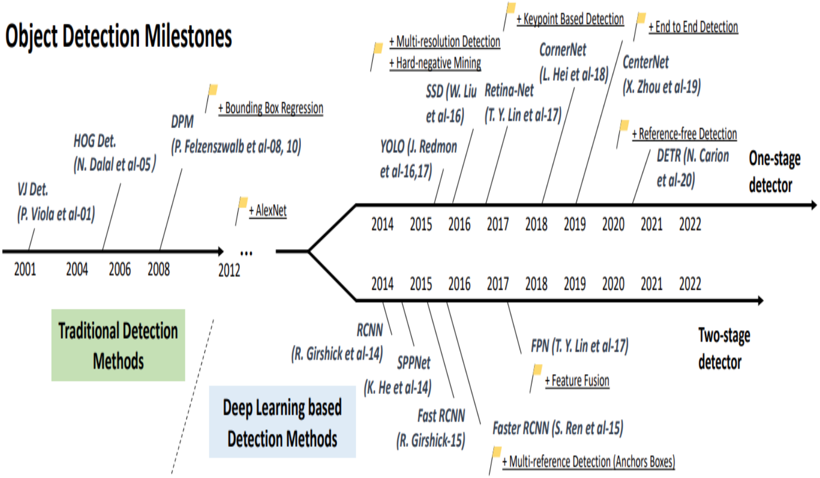


Evolution of Object Detection

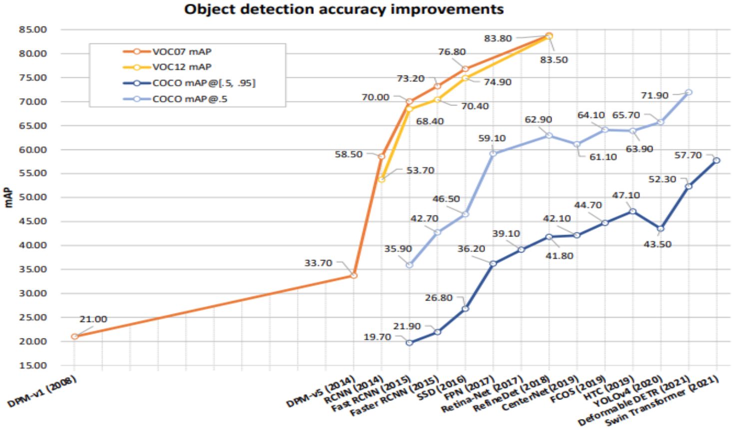
- Number of papers on **Object Detection from 1998 to 2021**



Evolution of Object Detection (cont.)



Evolution of Object Detection (cont.)



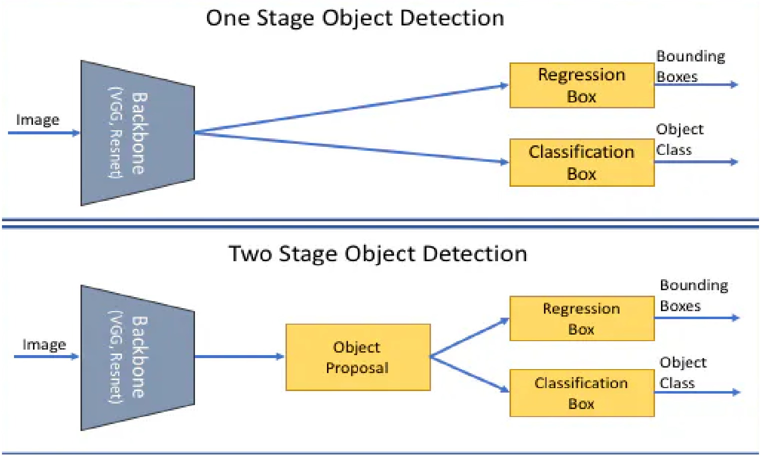
Evolution of Object Detection (cont.)

Dataset	train		validation		trainval		test	
	images	objects	images	objects	images	objects	images	objects
VOC-2007	2,501	6,301	2,510	6,307	5,011	12,608	4,952	14,976
VOC-2012	5,717	13,609	5,823	13,841	11,540	27,450	10,991	-
ILSVRC-2014	456,567	478,807	20,121	55,502	476,688	534,309	40,152	-
ILSVRC-2017	456,567	478,807	20,121	55,502	476,688	534,309	65,500	-
MS-COCO-2015	82,783	604,907	40,504	291,875	123,287	896,782	81,434	-
MS-COCO-2017	118,287	860,001	5,000	36,781	123,287	896,782	40,670	-
Objects365-2019	600,000	9,623,000	38,000	479,000	638,000	10,102,000	100,000	1,700,00
OID-2020	1,743,042	14,610,229	41,620	303,980	1,784,662	14,914,209	125,436	937,327

Object Detection using Deep Learning

- Object detection involves
  - Object Localization (Regression Problem)
  - Object category classification
- Approach 1:
  - Do region proposals
  - using region proposals, do object classification and regression
- Approach 2:
  - Do regression and classification together

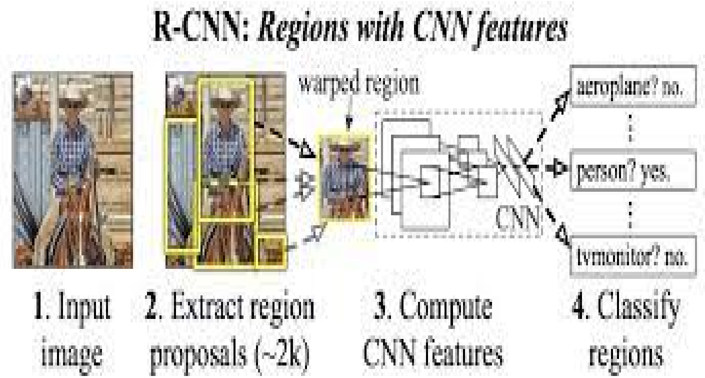
Object Detection using Deep Learning (cont.)



Object Detection using two Stages

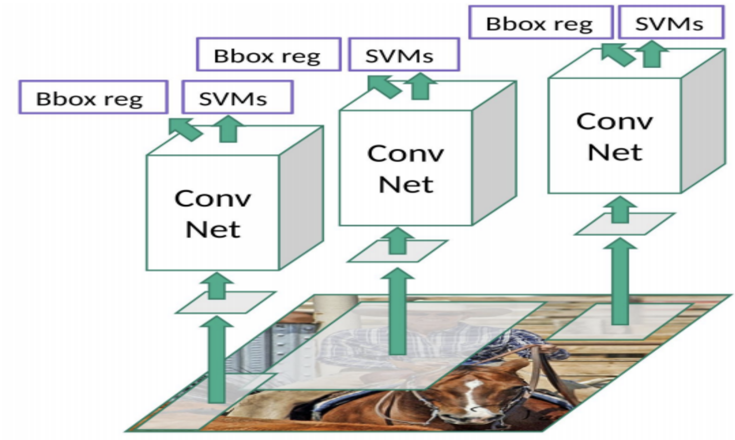
- Region Proposal CNN (R-CNN)
- Fast R-CNN
- Faster R-CNN

RCNN



R-CNN: Regions with CNN features
warped region
aeroplane? no.
person? yes.
tvmonitor? no.
CNN
1. Input image
2. Extract region proposals (~2k)
3. Compute CNN features
4. Classify regions

RCNN (cont.)



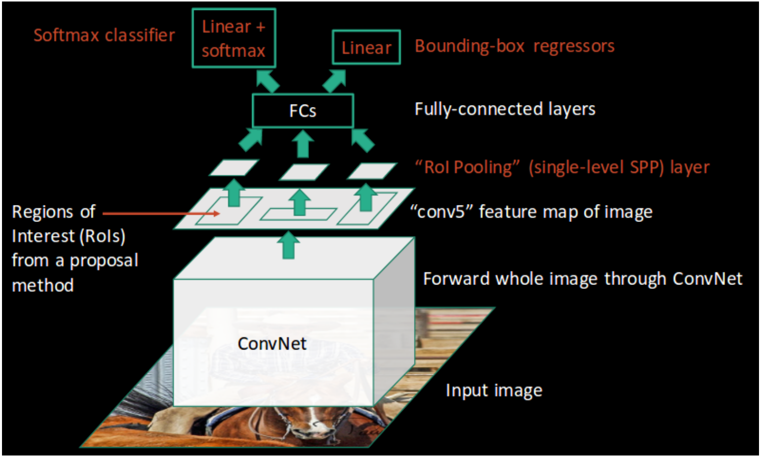
Bbox reg
SVMs
Bbox reg
SVMs
Bbox reg
SVMs
Conv Net
Conv Net
Conv Net

RCNN (cont.)

Steps of RCNN

1. Extracts a set of regions from the given image using selective search (2k regions)
2. Compute features for each of th region seleted in step 1
3. Build classifier that uses the features computed in the previous step, and output the object type for each region
4. Build regressor that uses the features computed in step 2, and predicts the center, width and height of the bounding box for each region

Fast RCNN



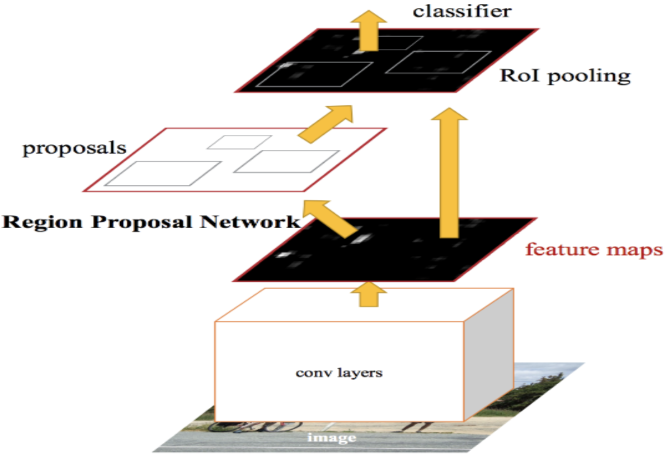
Softmax classifier
Linear + softmax
Linear
Bounding-box regressors
FCs
Fully-connected layers
"RoI Pooling" (single-level SPP) layer
Regions of Interest (RoIs) from a proposal method
"conv5" feature map of image
Forward whole image through ConvNet
ConvNet
Input image

Fast RCNN (cont.)

Steps of Fast RCNN

1. Give the entire image as input to ConvNet to get feature map
2. Build Region Proposal Net(RPN) to predict regions from the features computed in the step 1
3. Do ROI pooling: Do the max pooing of features over the proposed regions such that that resultant sizes for all regions are the same
4. Build Classifier and regressor using features of ROI, computed in the previous step

Faster RCNN


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

Faster RCNN (cont.)

Steps of Faster RCNN

1. Give input image to the ConvNet which returns feature maps for the image
2. Apply Region Proposal Network (RPN) on these feature maps and get object proposals
3. Apply ROI pooling layer to bring down all the proposals to the same size
4. Finally, pass these proposals to a fully connected layer in order to classify and predict the bounding boxes for the image

Object Detection using one Stage

- You Only Look Once (YOLO) (all five versions)
- Single Short Detector(SSD)

YOLO

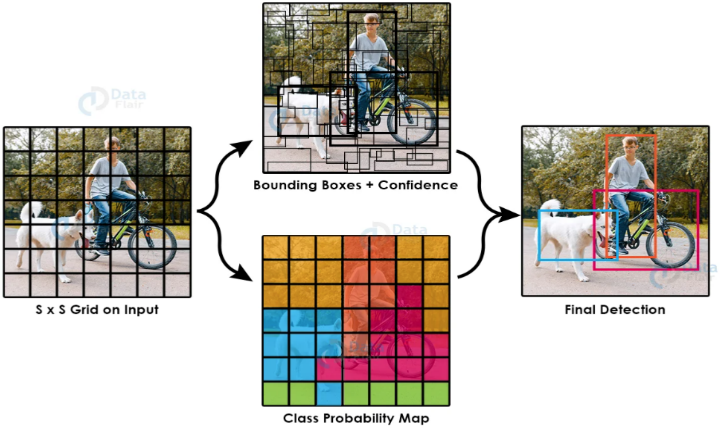


Figure 1: YOLO Object Detection

- Yolo divides the image into $S \times S$ cells(blocks)

YOLO (cont.)

- For each cell
  - Find $((x, y), w, h, c, p(c_1), ..p(c_k))$
    - (x, y) is the enter of box predicted
    - w, h are width and height of the box
    - c is confidence score for the cell to have an object
    - $p(c_i)$ is the probability of the object(which is prsent in the cell) to belong to class c_i
- YOLO is influenced by googleNet
- YOLO consists of
  - 24 convolution layers to extract features
  - Max pooling layers
  - Two fully connected layers to predict object class and bounding box

YOLO (cont.)

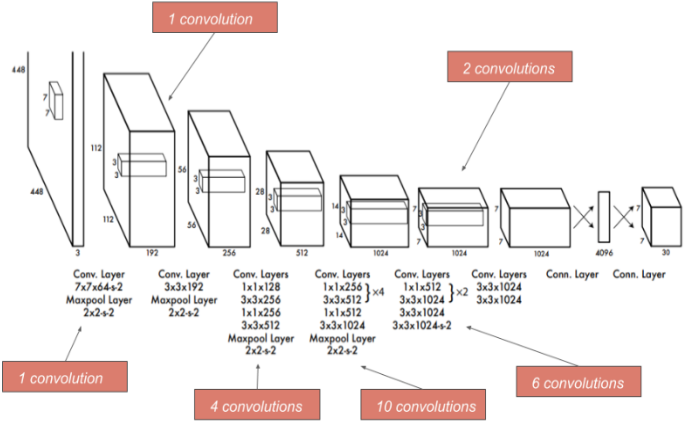


Figure 2: YOLO Architecture

YOLO (cont.)

Confidence Loss:

1 if object **appears** in **cell i** and **j-th box** detects it, **0** otherwise

Ground truth confidence score

Set to 0.5 to **decrease** the loss for **empty** boxes

$$\sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} (C_i - \hat{C}_i)^2 + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{noobj} (C_i - \hat{C}_i)^2$$

For each **grid cell**

For each **box**

Predicted confidence score

1 if there is **no** object in the **i-th cell**, **0** otherwise

Localization Loss:

$$\lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$+ \lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

Classification Loss:

$$\sum_{i=0}^{S^2} \mathbb{1}_i^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2$$

- Loss = Confidence Loss + Localization Loss + Classification Loss

Acknowledgement

Images and other details are taken from various internet sources.