Module 2 :

Deep Networks

Convolutional Neural Networks

Géraldine Conti, August 2020

Discussion Session

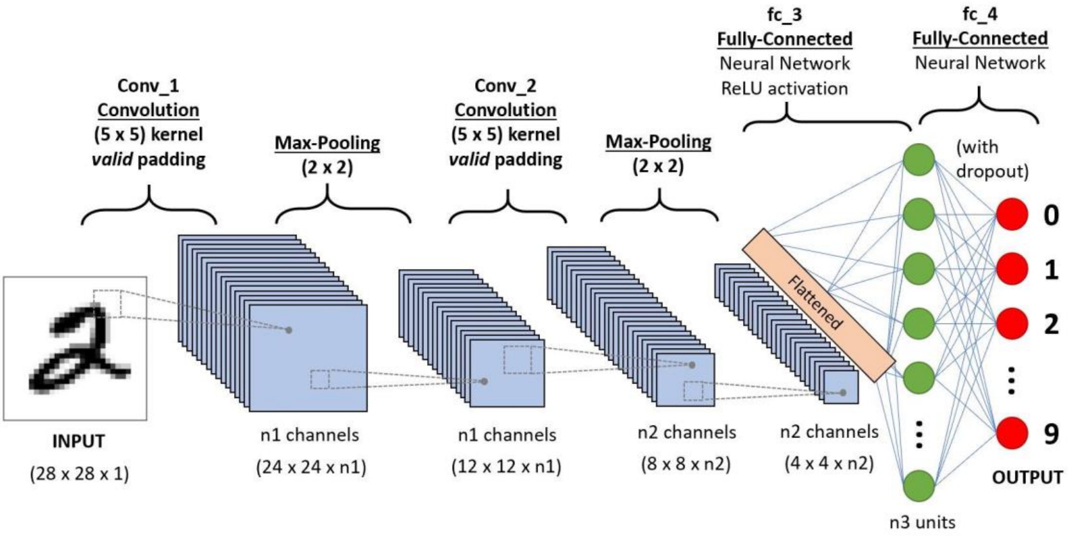
- Review of Notebook 5
    - Vanishing/exploding gradients
    - Xavier/He initializations
    - Leaky ReLU, ELU, SELU
    - Batch normalization
    - Gradient clipping
    - Reusing pretrained layers
    - Faster optimizers
    - Learning rate scheduling
    - Regularization
    - Dropout

Bibliography

- Deep Learning book (Goodfellow, Bengio, Courville)
- Machine Learning @ Stanford (Prof Andrew Ng)
- Hands-On Machine Learning with Scikit-Learn & Tensorflow (Aurélien Géron)

Learning Objectives

- CNN components
- Most important architectures
- Object Detection
- Face Detection



Convolutional Neural Networks

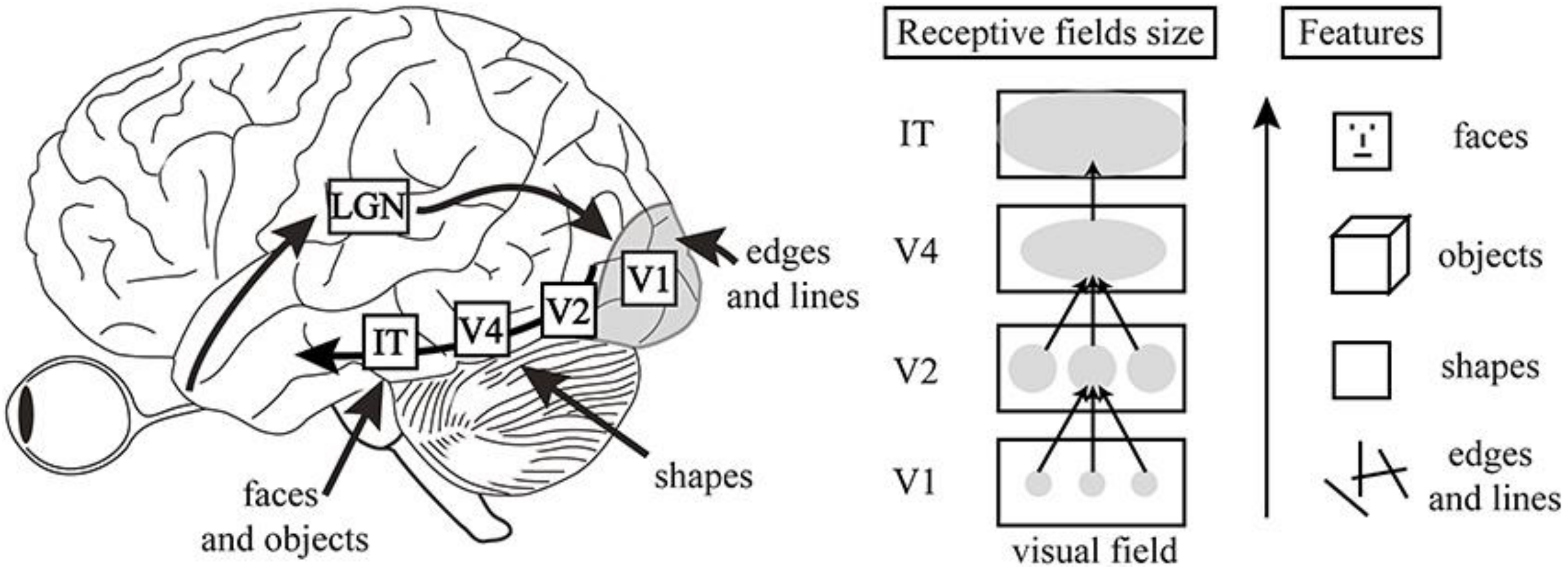
- Convolutional layer
- Padding, stride
- Filters
- Pooling layer

Introduction

- Convolutional Neural Networks (CNNs) emerged from the study for the brain's visual cortex
- Used in image recognition since the 1980s
  - Milestone (1998) with LeNet-5 architecture (LeCun)
- Huge improvements in the last few years due to :
  - Increased computational power
  - Amount of available training data
  - Tricks for training deep nets
- Everywhere: image search services, self-driving cars, automatic video classification, voice recognition, natural language processing,…

Visual Cortex

- Nobel prize in physiology (1981) : many neurons in the visual cortex react only to visual stimuli located in a limited region of the visual field (local receptive field)
  - Receptive fields overlap and tile the whole visual field

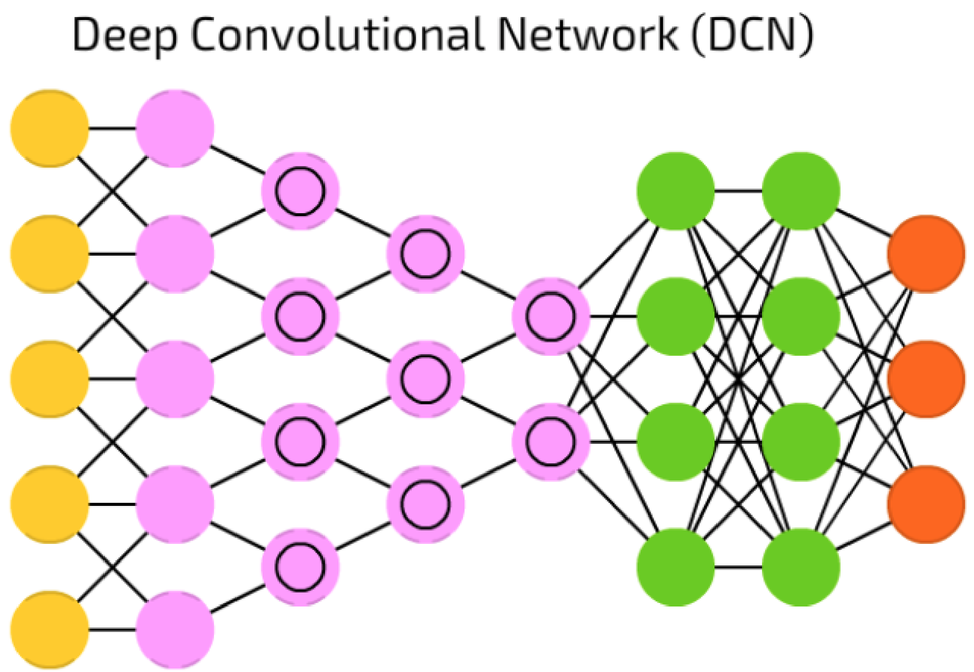


- Some neurons (with same receptive field) react to different line orientations (horizontal lines, lines with angle,…)

- Some neurons have larger receptive fields and react to more complex patterns : idea that higher-level neurons are based on the outputs of neighboring lower-level neurons

CNN

- inspired by the organization of the animal visual cortex



- **Convolutional and pooling cells** introduced in LeNet-5
    - used to process and simplify input data
    - weight sharing between *local regions*

Use cases

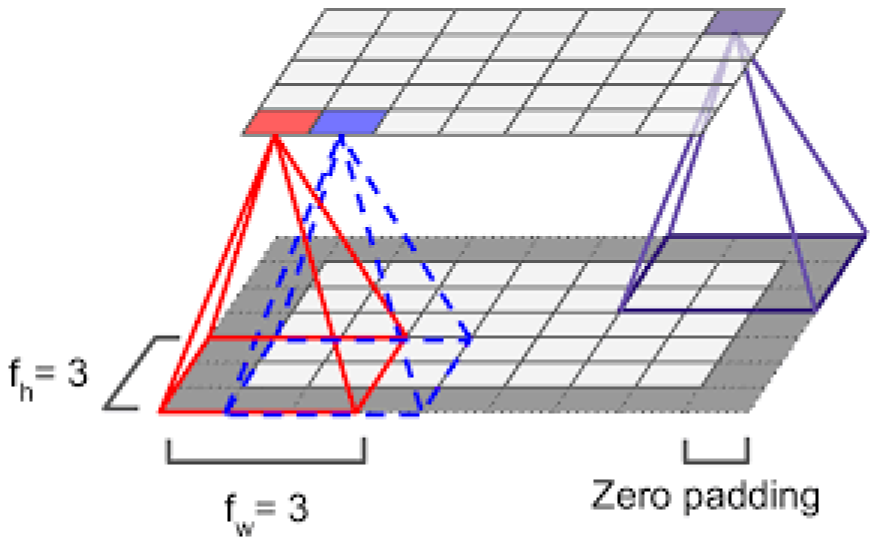
- well suited for computer vision tasks
  - Image classification
  - Object detection

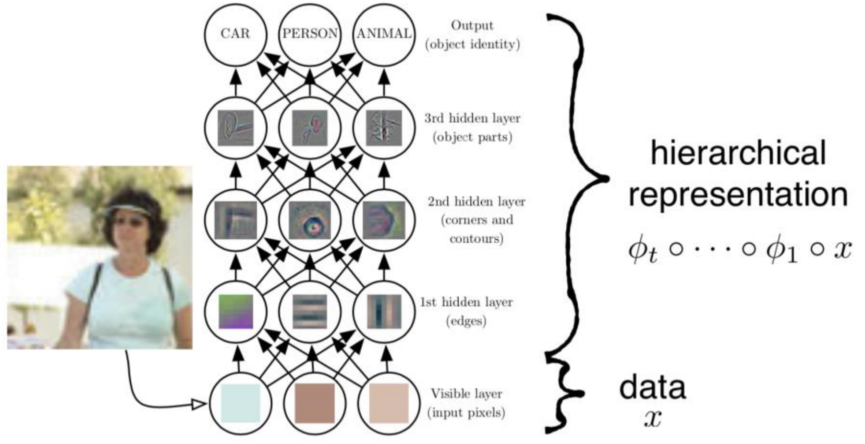
- More generally, specialized Neural Network for data arranged on a grid
  - Images
  - DNA sequences
  - …

	A	C	G	T	W	S	M	K	R	Y	B	D	H	V	N	Z
A	1	0	0	0	1/2	0	1/2	0	1/2	0	0	1/3	1/3	1/3	1/4	0
C	0	1	0	0	0	1/2	1/2	0	0	1/2	1/3	0	1/3	1/3	1/4	0
G	0	0	1	0	0	1/2	0	1/2	1/2	0	1/3	1/3	0	1/3	1/4	0
T	0	0	0	1	1/2	0	0	1/2	0	1/2	1/3	1/3	1/3	0	1/4	0

Convolutional Layer (CONV)

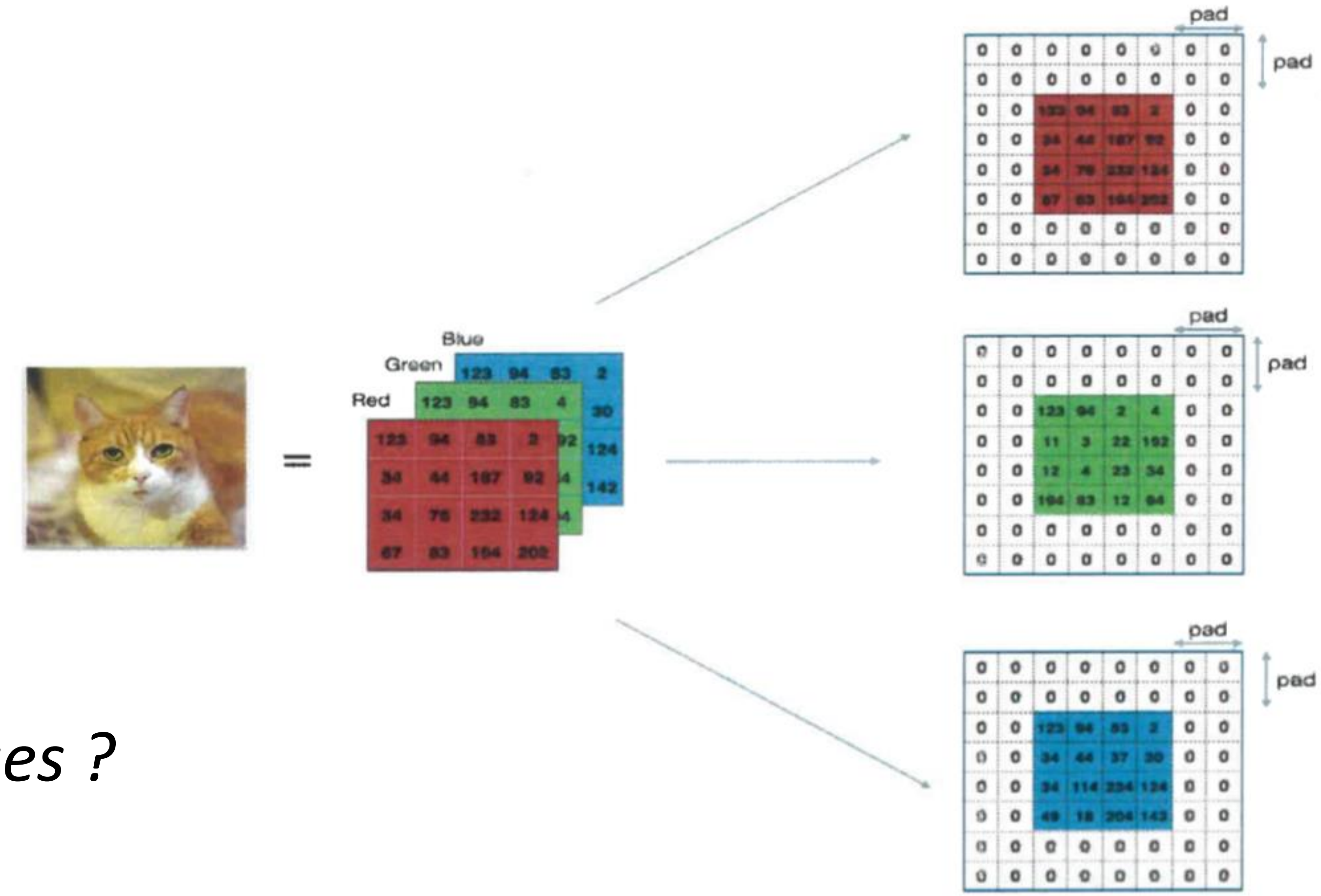
- Neurons in the 1st convolutional layer are not connected to every single pixel in the input image, but only to pixels in their receptive fields
- Neurons in the 2nd layer are connected only to neurons located within a small rectangle in the first layer
- Low-level features in the first hidden layer, higher-level features in the next hidden layer,…





Padding

- Adds zeros around the border of an image

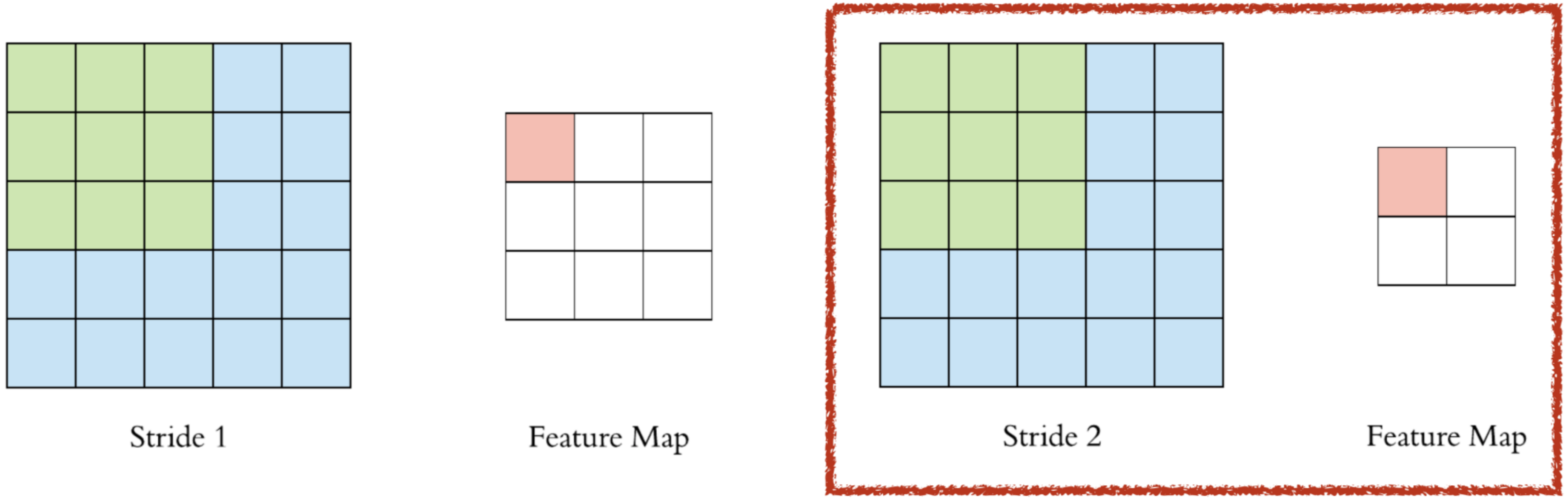


Use cases ?

Stride

- By how much you move the filter

- Use cases :
  - Connect a large input layer to a much smaller layer by spacing out the receptive fields (reduce dimensionality)



increasing stride from 1 to 2

Filters

- Also called convolutional kernels
- used to detect features
- Examples :
  - Vertical filter : neurons using these weights will ignore everything in their receptive field except for the central vertical line
  - Idem for a horizontal filter

1	0	-1
1	0	-1
1	0	-1

Vertical

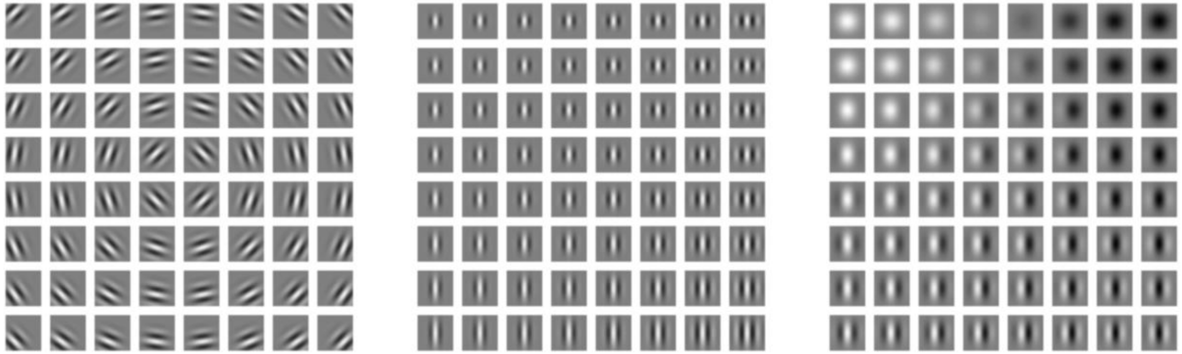
1	1	1
0	0	0
-1	-1	-1

Horizontal

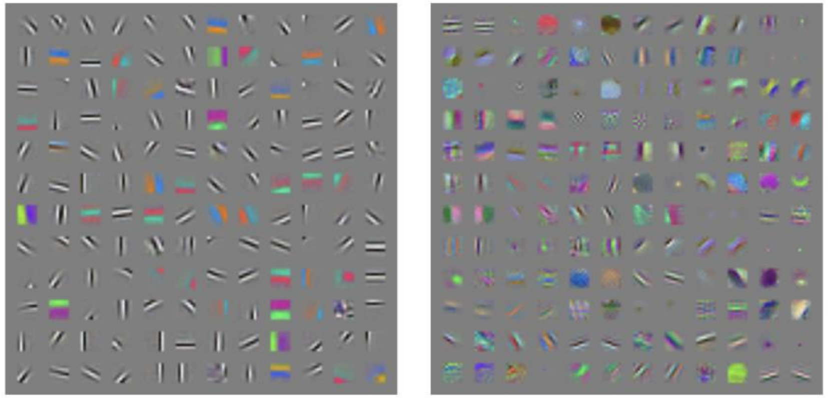
Kernel Initialization

- With random weights

- With hand-designed features

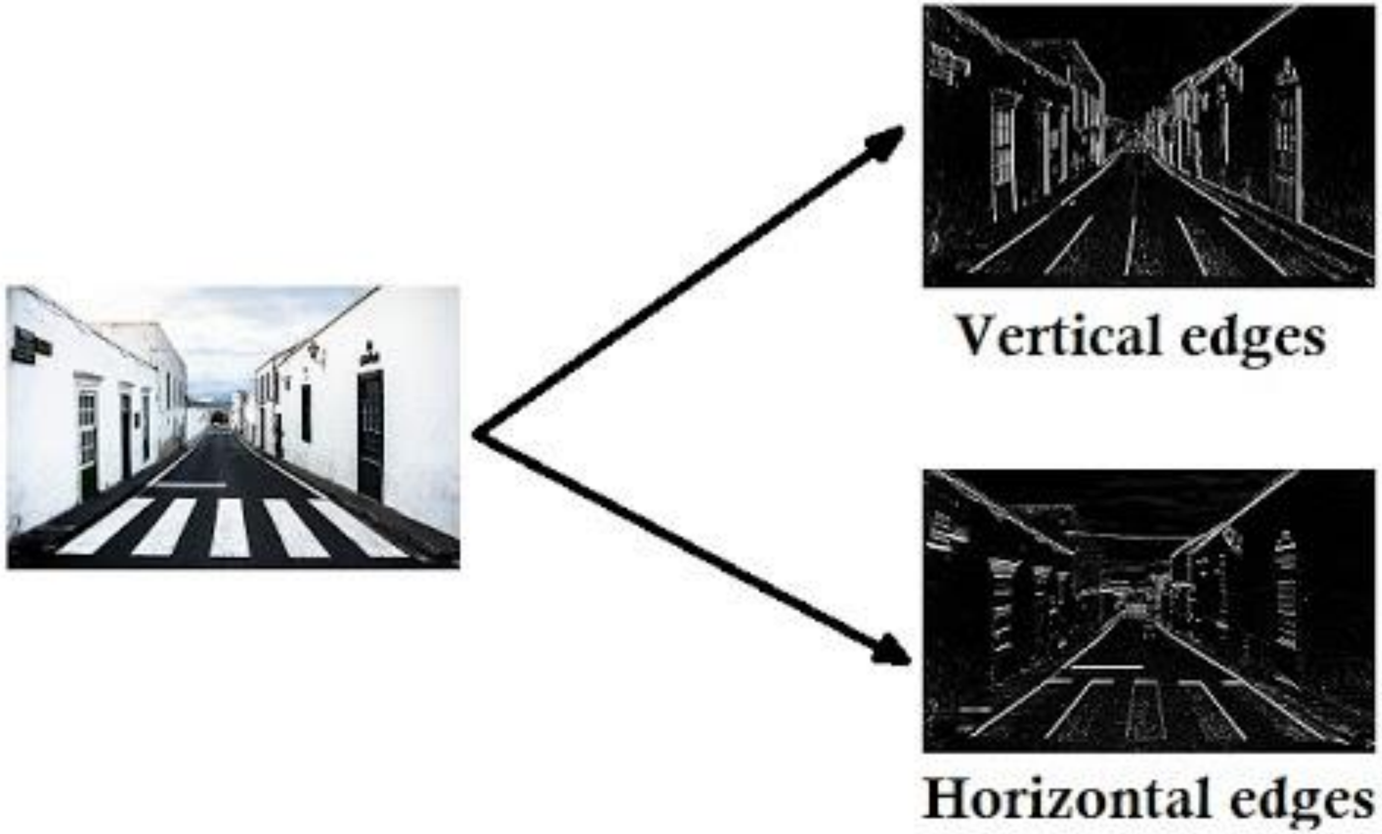


- With unsupervised learning algorithms (eg. apply K-means clustering to patches, then use centroids as kernels)

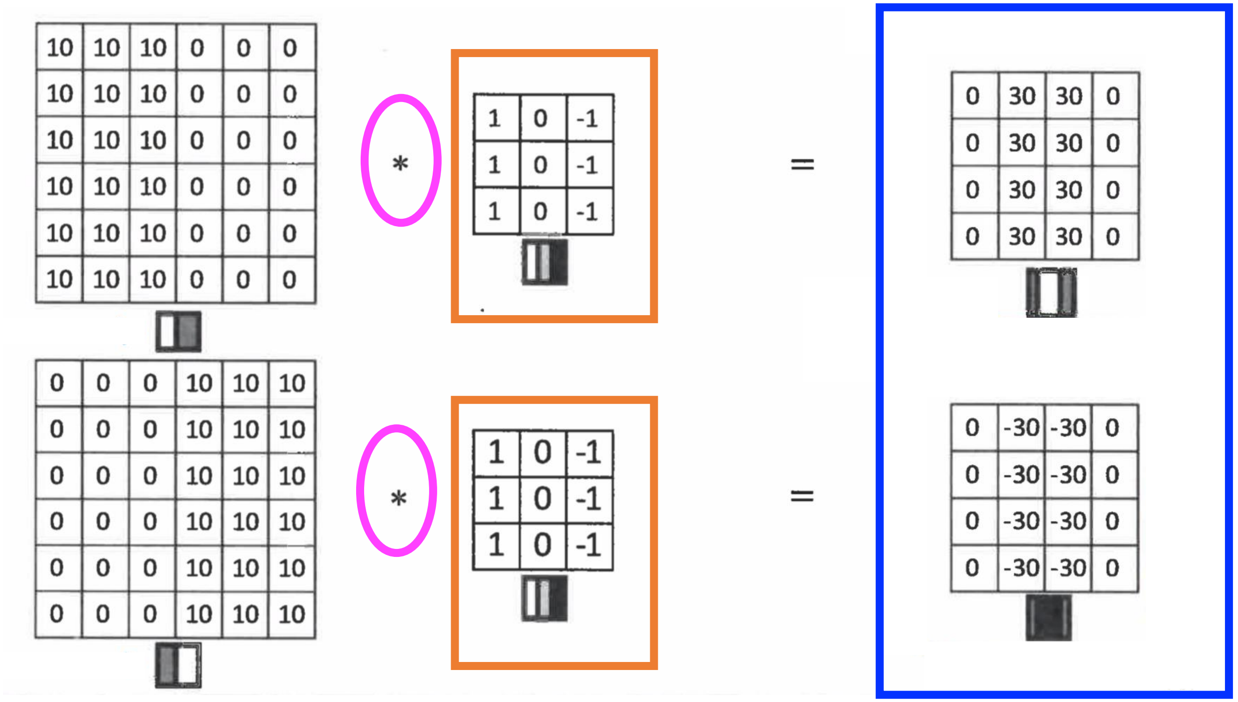


Feature Map

- A full layer of neurons using the same filter gives a feature map highlighting the areas in an image that are most similar to the filter



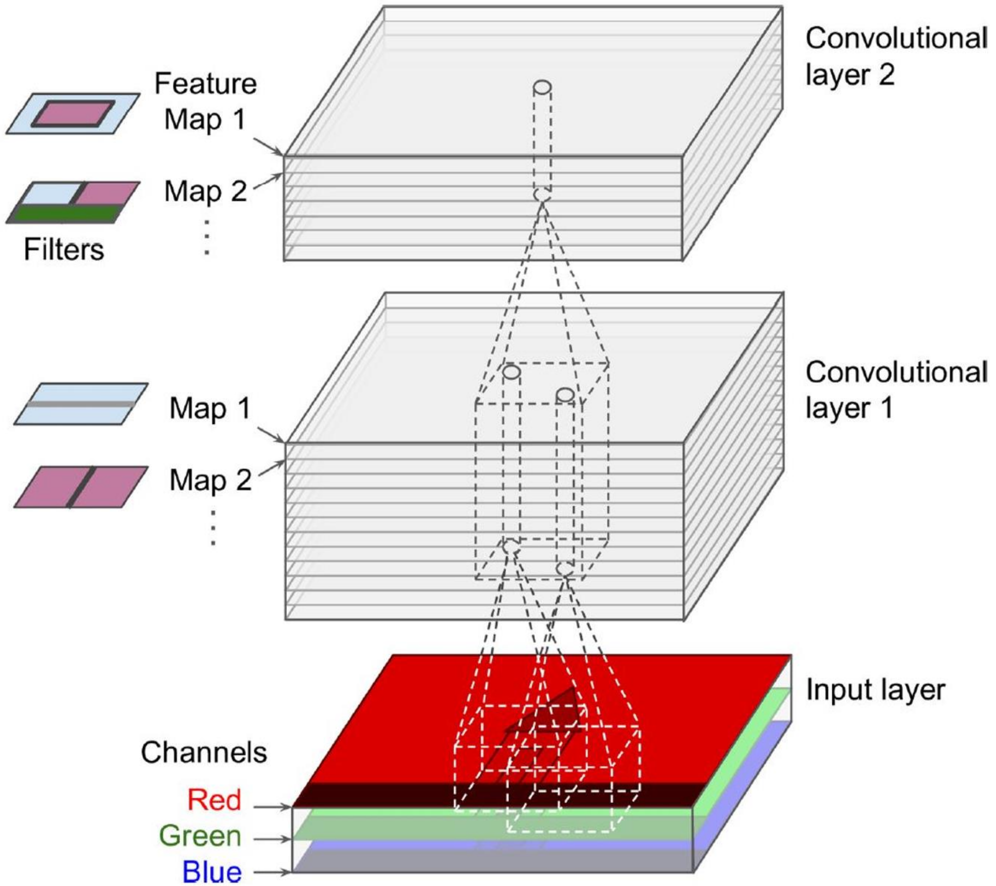
Convolution



$n \times n$ image $\quad f \times f$ filter $\quad \text{out} = n - f + 1$

Stacking Multiple Feature Maps

- Each convolutional layer composed of several feature maps of equal sizes
- Within one feature map, all neurons share the same parameters (weights and bias term)



- Parameters : $\boxed{\text{(filter width * filter height * filter depth +1) * filter number}}$

Exercise

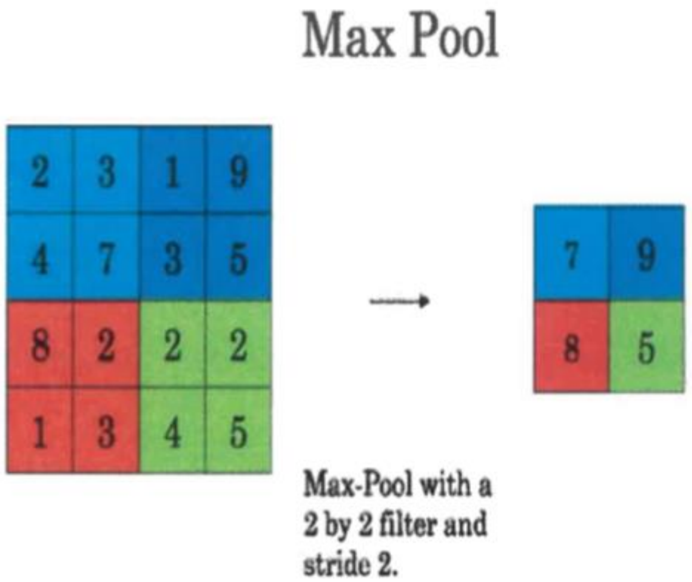
- 300x300 RGB image, how many parameters does a hidden layer have (incl. bias) in the following cases:

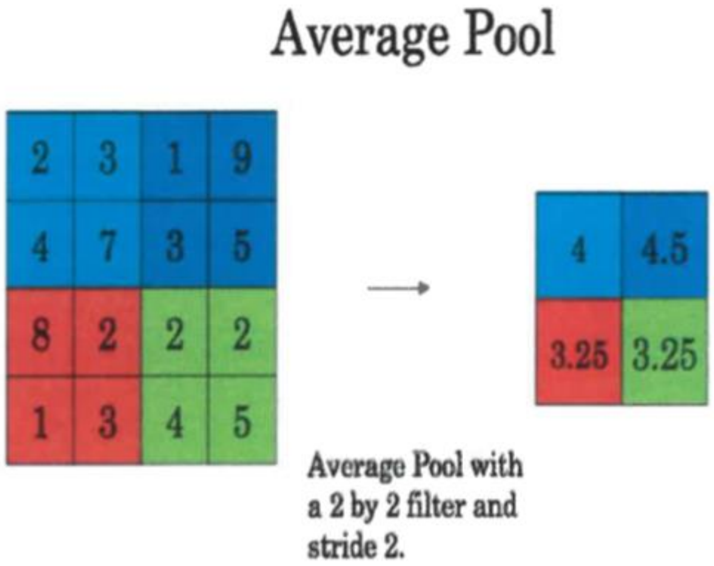
a) Hidden layer with 100 neurons, each fully connected to the input (*no convolution*)

b) Hidden *convolution* layer with 100 filters that are each 5x5

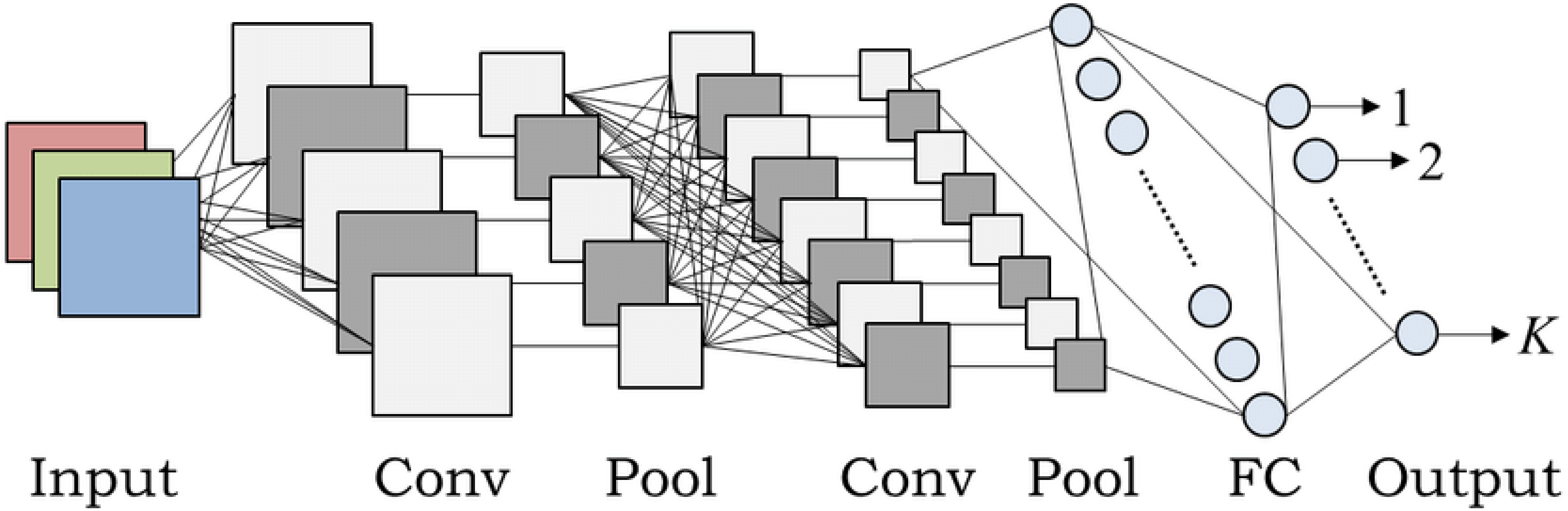
Pooling Layer

- Goal is to subsample the input image
  - Reduce the computational load, memory usage, number of parameters (hence overfitting)
- Neurons connected to the outputs of a limited number of neurons in the previous layer located within a small rectangular receptive field
- Pooling neuron has no weights, it aggregates the inputs with an aggregation function (max, mean)





CNN Architectures



- Stack a few CONV layers (each one followed by ReLU), then a POOL layer, then few CONV layers (+ReLU), then POOL layer, …
- Image gets smaller and smaller, but also deeper and deeper (with more feature maps) due to the CONV layers
- At the top of the stack, a regular feedforward NN is added with a few fully connected layers (+ReLU)
- Final output is the prediction (softmax giving class probabilities)

One Look Is Worth A Thousand Words--

One look at our line of Republic, Firestone, Miller and United States tires can tell you more than a hundred personal letters or advertisements.

WE WILL PROVE THEIR VALUE BEFORE YOU INVEST ONE DOLLAR IN THEM.

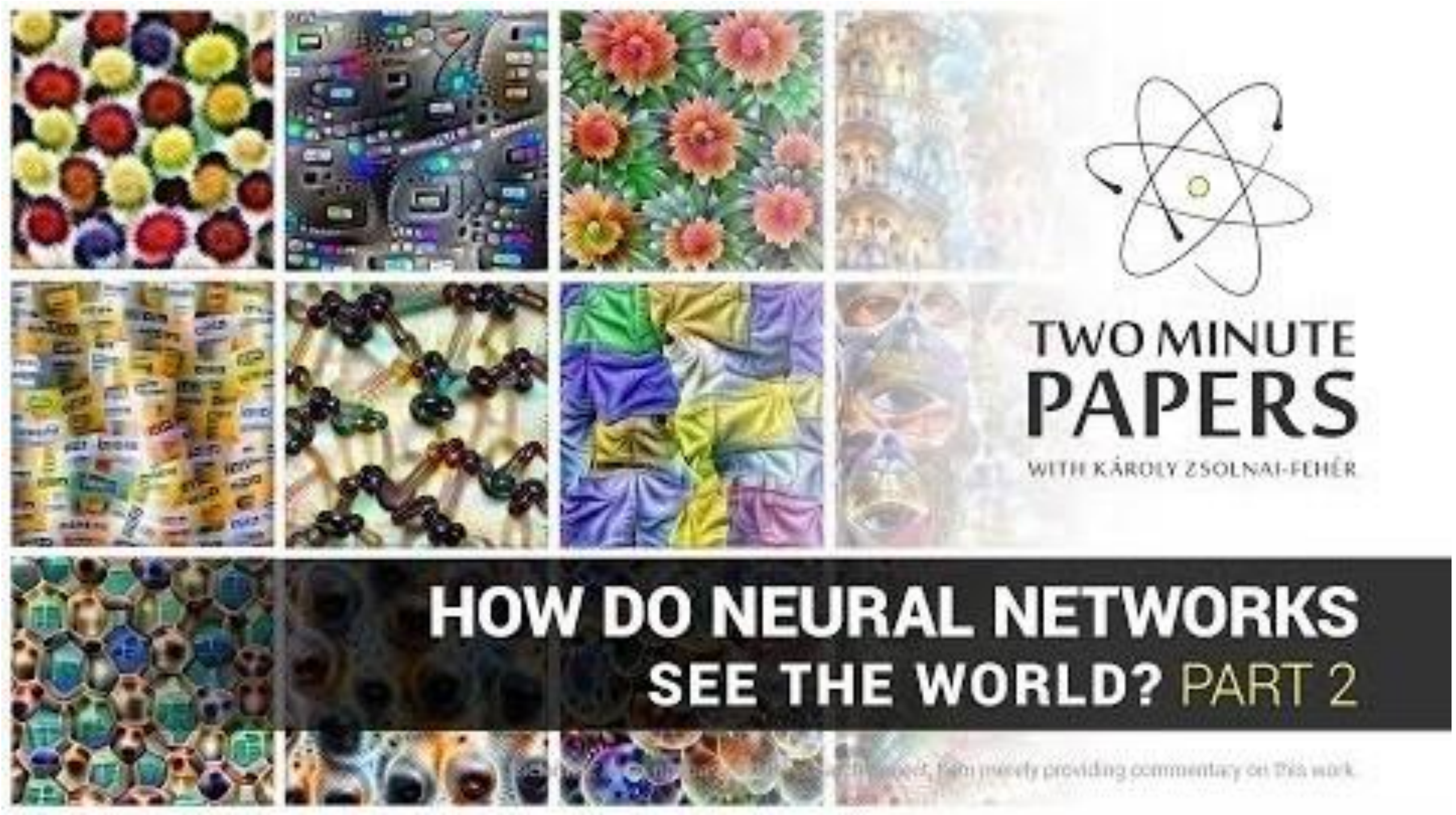
Ever consider buying Supplies from a catalog?

What's the use! Call and see what you are buying. One look at our display of automobile and motorcycle accessories will convince you of the fact.

THAT WE HAVE EVERYTHING FOR THE AUTO

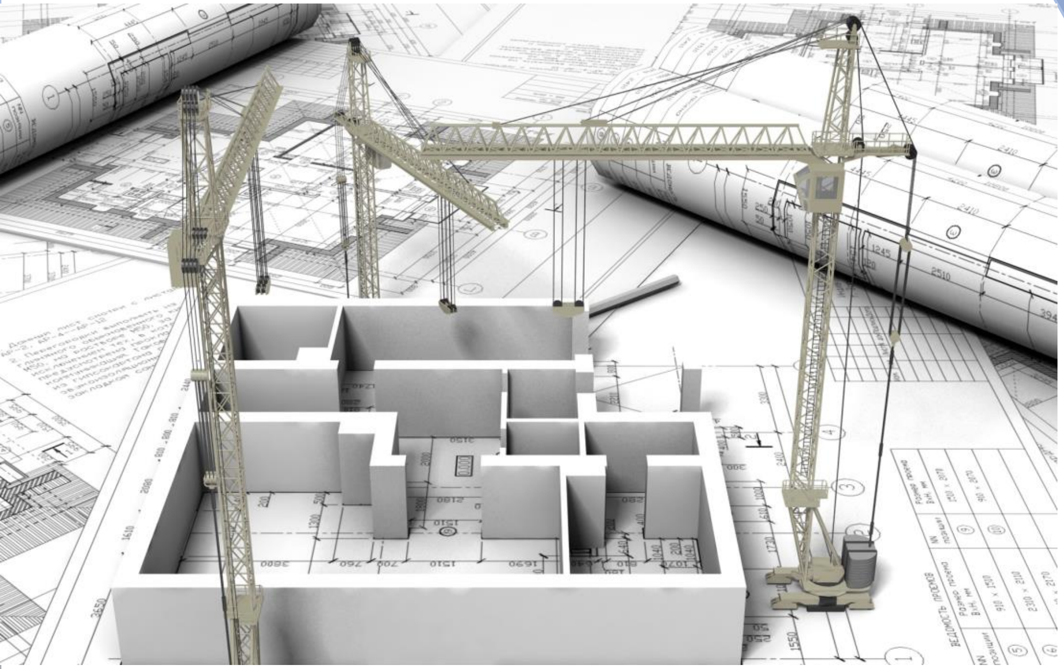
Piqua Auto Supply House

133 N. Main St.—Piqua, O.



Two-Minute Papers

CNN Famous Architectures



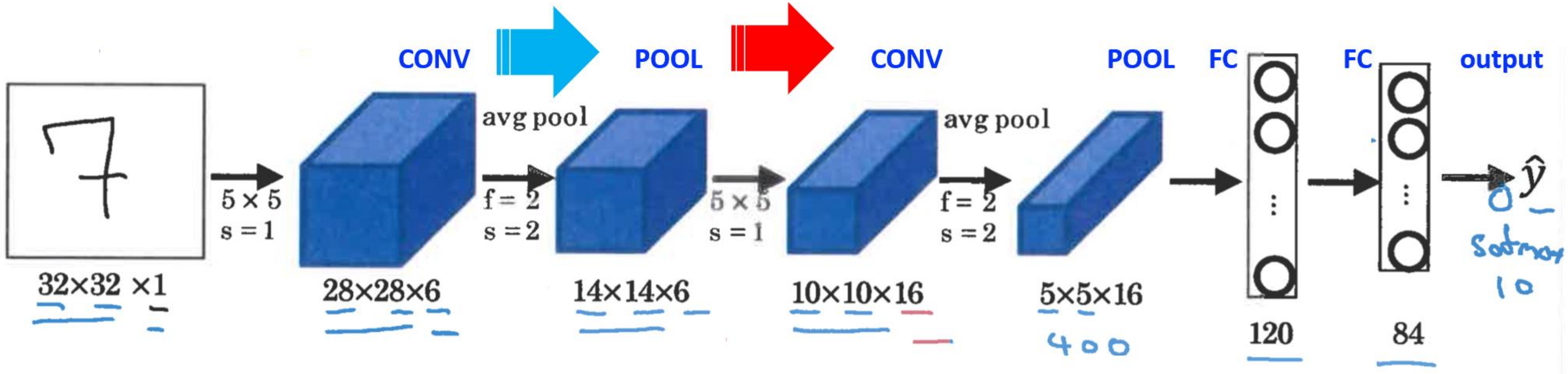
1. LeNet-5
2. AlexNet
3. GoogLeNet
4. ResNet

1. LeNet-5 (1998)

- Used for hand-written digit recognition (MNIST)
  - Images (28x28) zero-padded to 32x32 and normalized
  - The rest of the network does not use padding, which is why the size keeps shrinking

- Activation functions : sigmoid/tanh

- 60K parameters



Exercise

- Fill the table with the parameters corresponding to the figure of the LeNet-5 architecture (previous slide)

Type	Feature Map	Size	Kernel size	Stride	Activation
Input image	1	32x32	-	-	-
Convolution					
Average pooling					
Convolution					
Average pooling					
FC					
FC	-		-	-	
Output	-		-	-	

2. AlexNet (2012)

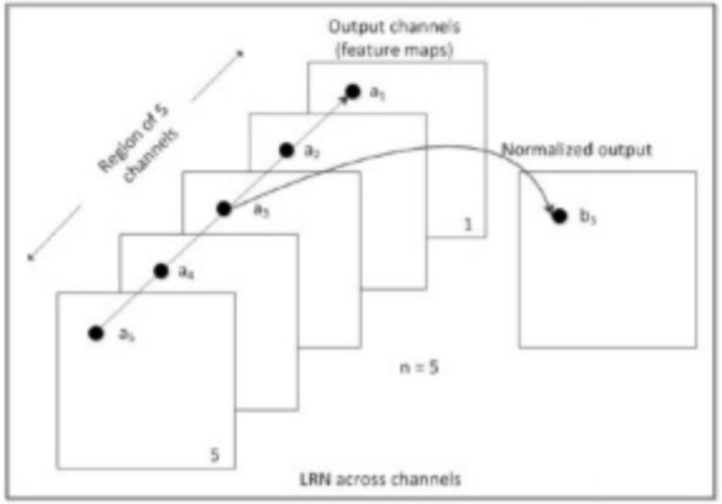
- Quite similar to LeNet_5, only much larger and deeper
- First one to stack CONV layers directly on top of each other instead of stacking a POOL layer on top of each CONV layer
- Activation functions : ReLU
- 60mio parameters
- To reduce overtiftting :
  - 50% dropout in FC layers
  - data augmentation
- Local Response Normalization used a normalization step

Type	Feature Map	Size	Kernel size	Stride	Activation
Input	3 (RGB)	227x227	-	-	-
Convolution	96	55x55	11x11	4	ReLU
Max Pooling	96	27x27	3x3	2	-
Convolution	256	27x27	5x5	1	ReLU
Max Pooling	256	13x13	3x3	2	-
Convolution	384	13x13	3x3	1	ReLU
Convolution	384	13x13	3x3	1	ReLU
Convolution	256	13x13	3x3	1	ReLU
Max Pooling	256	6x6	3x3	2	-
Fully connected	-	4096	-	-	ReLU
Fully connected	-	4096	-	-	ReLU
Fully connected	-	1000	-	-	Softmax

Local Response Normalization (LRN)

- Neurobiology : capacity of a neuron to reduce the activity of its neighbors (lateral inhibition)

- In DNNs : inhibition carries out local contrast enhancement so that locally maximum pixel values are used as excitation for the next layers
  - Neuron a3 that most strongly activates will inhibit neurons (a1,a2,a4,a5) at the same location but in neighboring feature maps



- This encourages different feature maps to specialize, pushing them apart and forcing them to explore a wider range of features

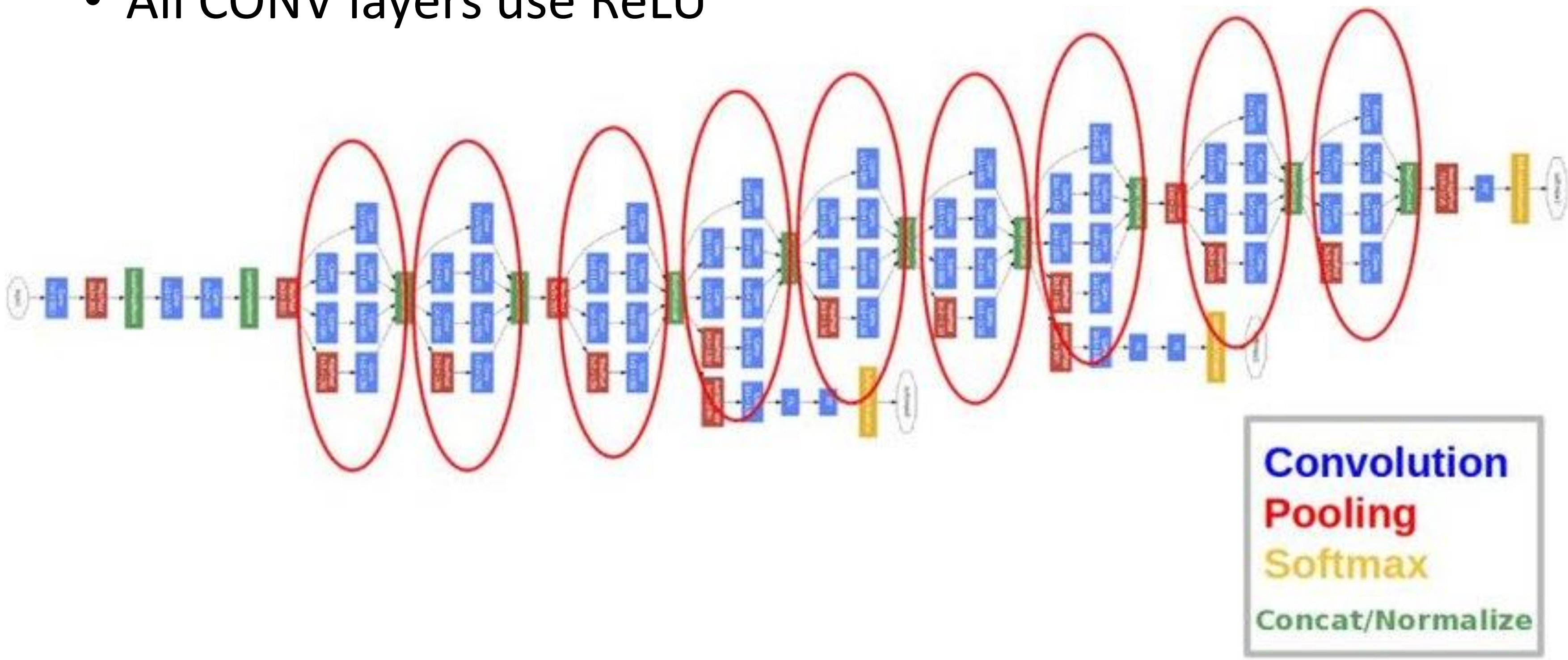
Exercise

- Draw the scheme corresponding to the AlexNet architecture (previous slide)

3. GoogLeNet (2014)

- Much deeper than previous CNNs thanks to sub-networks called inception modules
  - These use parameters much more efficiently : 6 mio of parameters instead of 60mio for AlexNet

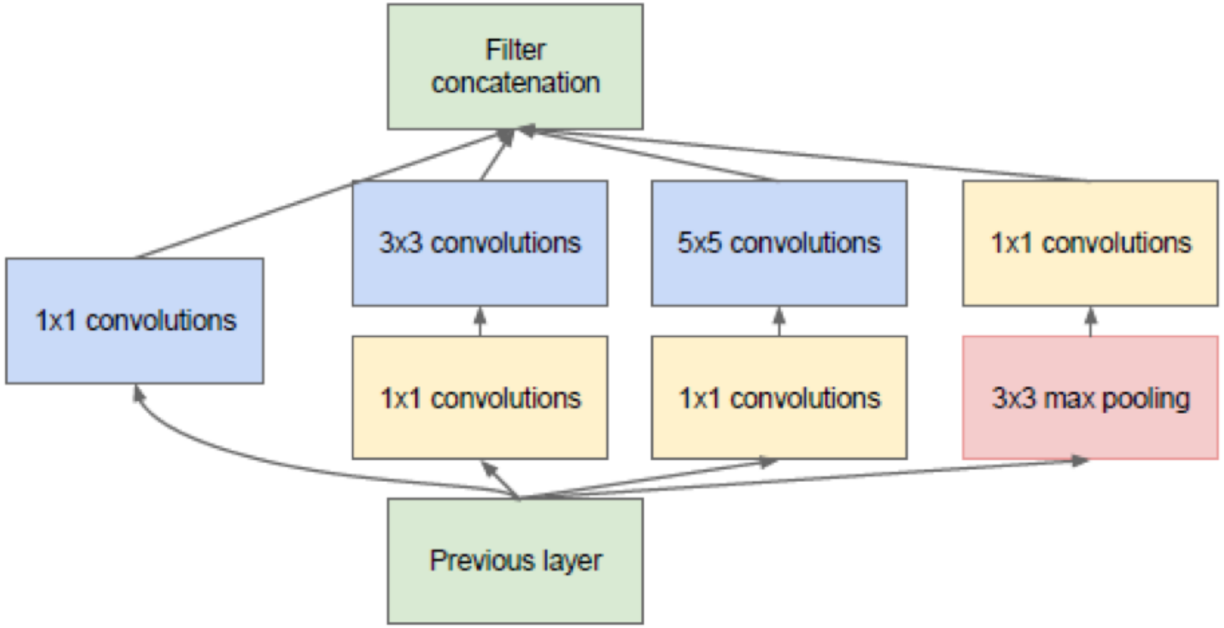
- Architecture :
  - 9 inception modules included
  - All CONV layers use ReLU



Inception Modules

- Input signal copied and fed to 4 layers
  - Second set of CONV layers has different kernel sizes to capture patterns at different scales

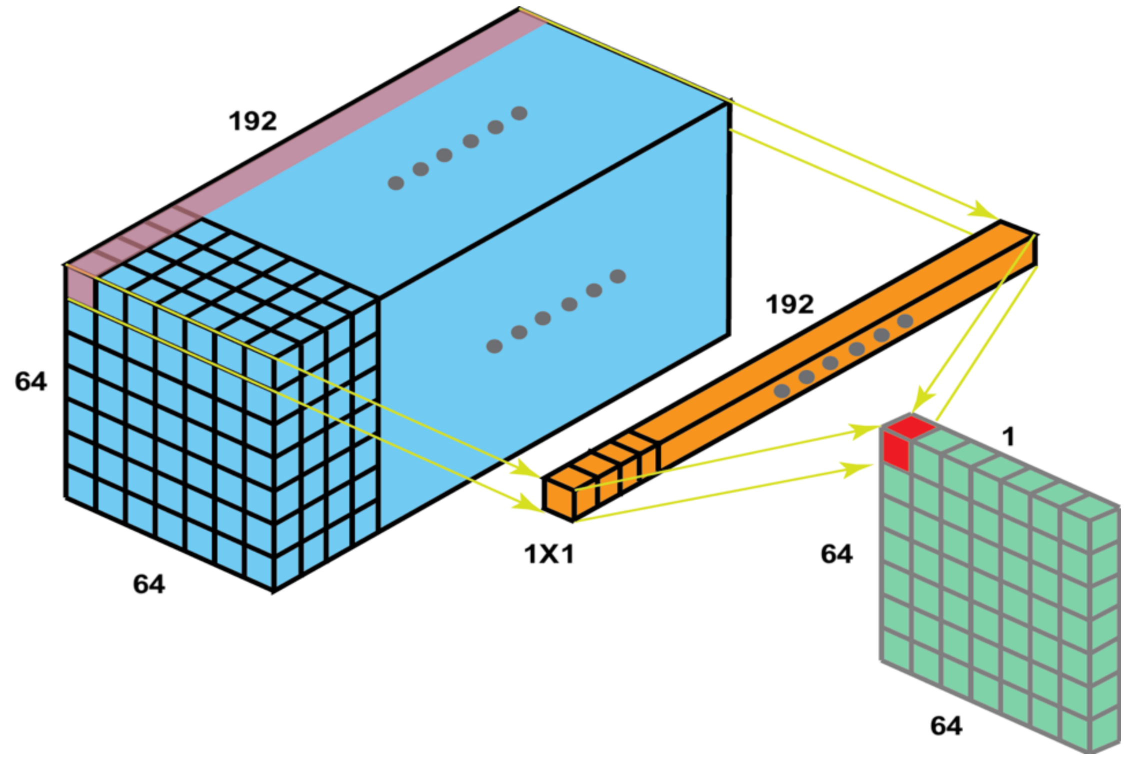
- Output : concatenate the four outputs (i.e. stack the feature maps)



Why use CONV layers with 1x1 kernel ?

1x1 convolutions

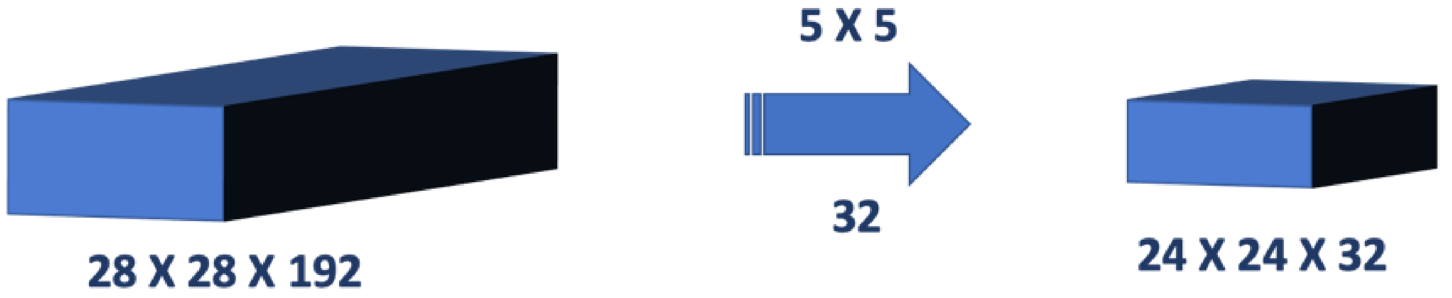
- These layers do not capture any features since they look at only one pixel at a time.
- Reduces the depth, but keep the height and width of the feature map



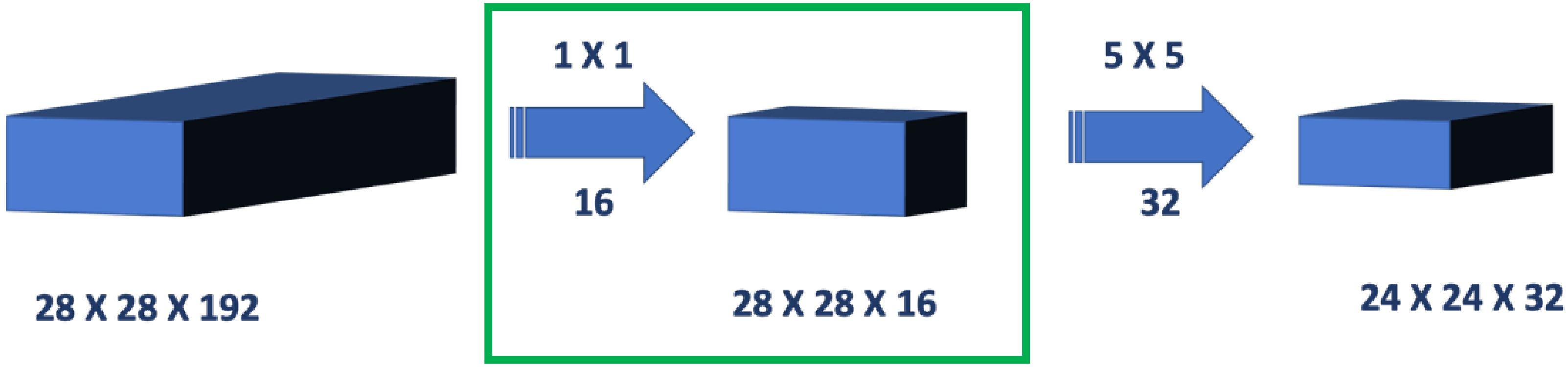
- Use cases :
  - Dimensionality reduction (GoogleNet)
  - Build deeper networks (ResNet)

1x1 convolutions : dimensionality reduction

- These layers do not capture any features since they look at only one pixel at a time.



Number of Operations : (28X28X32) X (5X5X192) = 120.422 Million Ops



Number of Operations for 1 X 1 Conv Step : (28X28X16) X (1X1X192) = 2.4 Million Ops
Number of Operations for 5 X 5 Conv Step : (28X28X32) X (5X5X16) = 10 Million Ops
Total Number of Operations = 12.4 Million Ops

4. ResNet (2015)

- Extremely deep CNN composed of 152 layers
- Very deep NNs suffer from vanishing gradients : skip connections

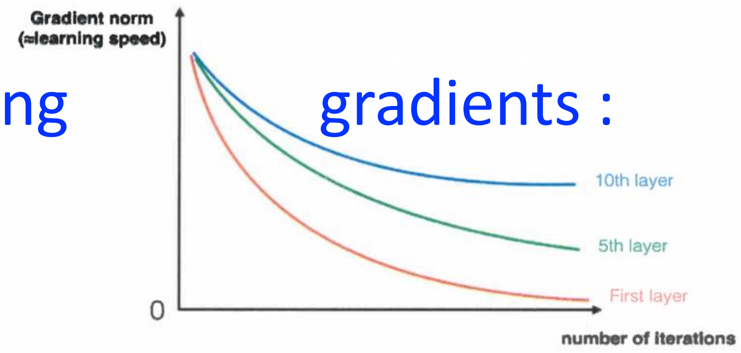
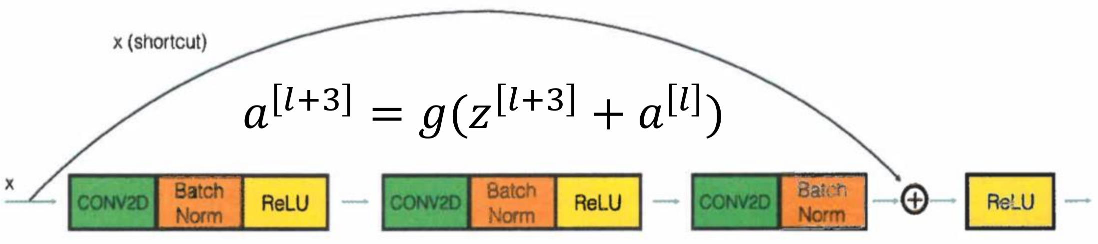


Figure 1 : Vanishing gradient

- Residual training : the signal feeding into a layer is also added to the output of a layer located a bit higher up the stack
  - Network forced to model : f(x) = h(x) – x rather than h(x)
- Architecture : stack of residual units, where each residual unit is a small NN with a skip connection

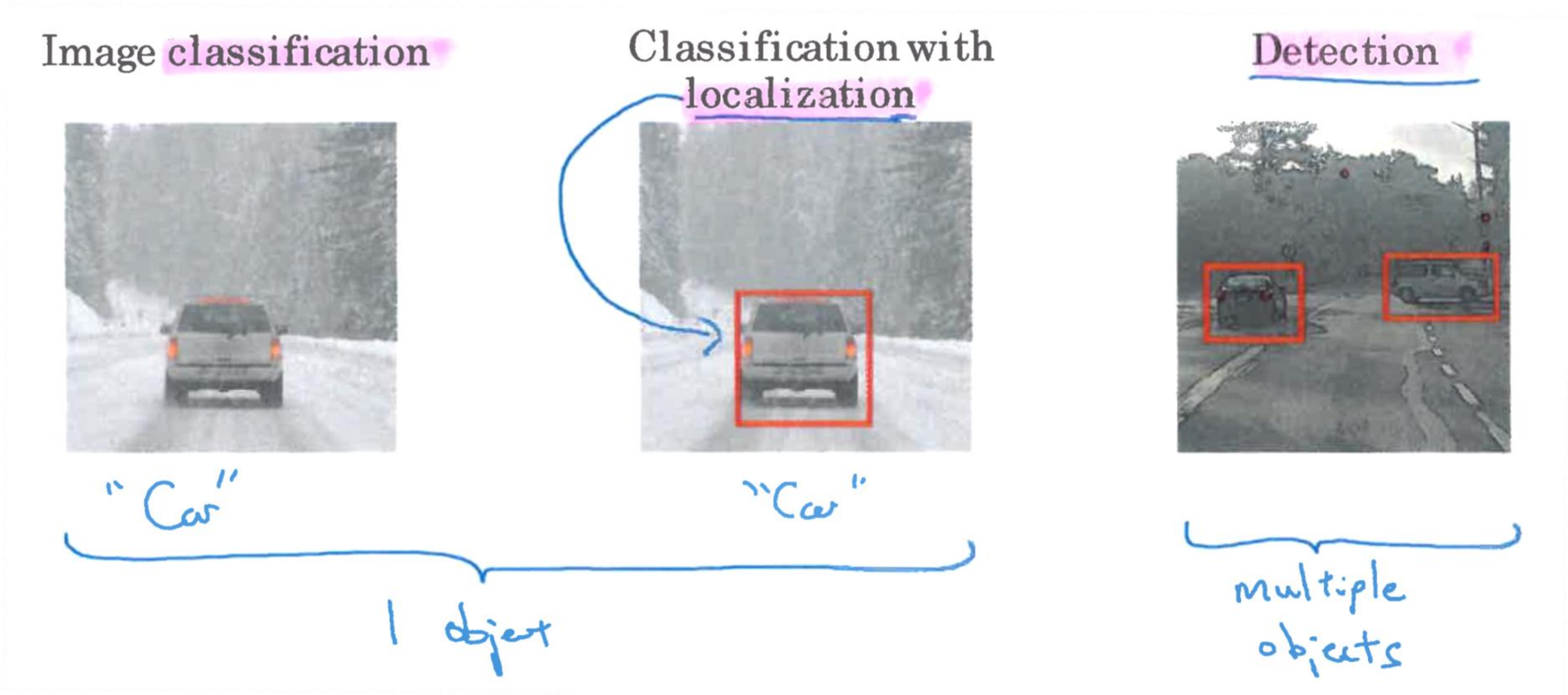


Some reading

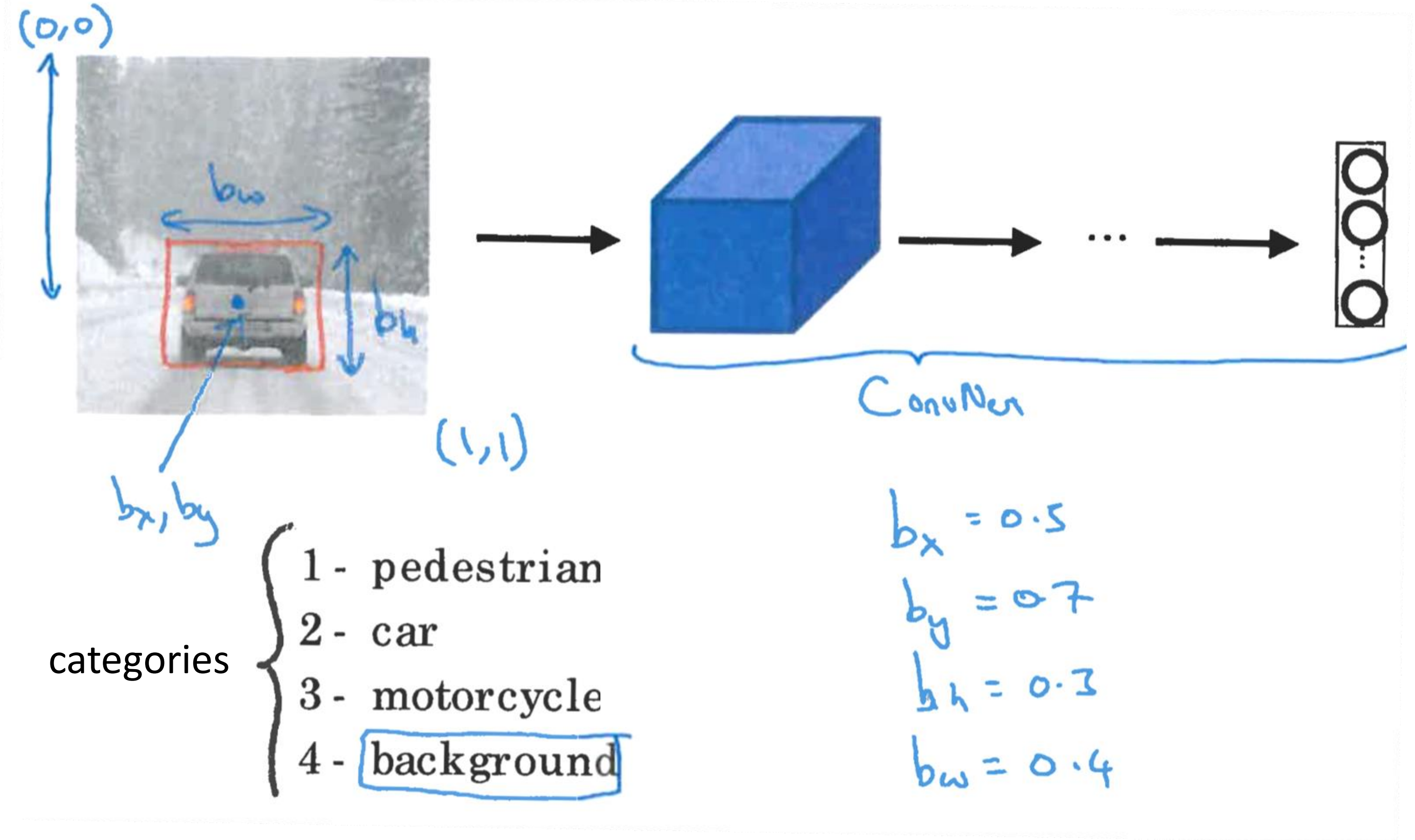
10 Architectures

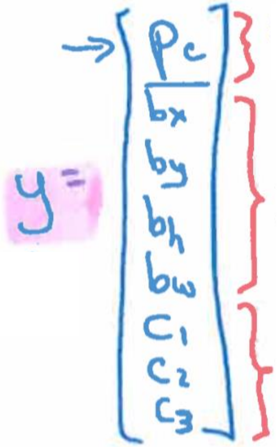
Object Detection

Object Detection



2. Classification with Localization



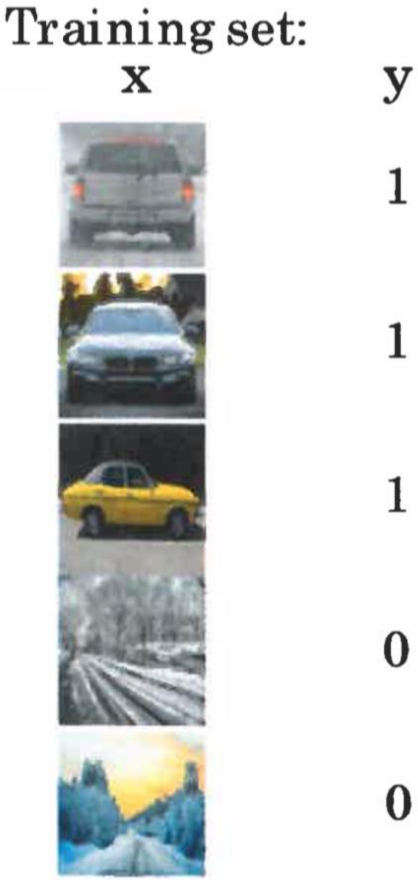


is there any object ?

bounding box shape

categories

3. Detection

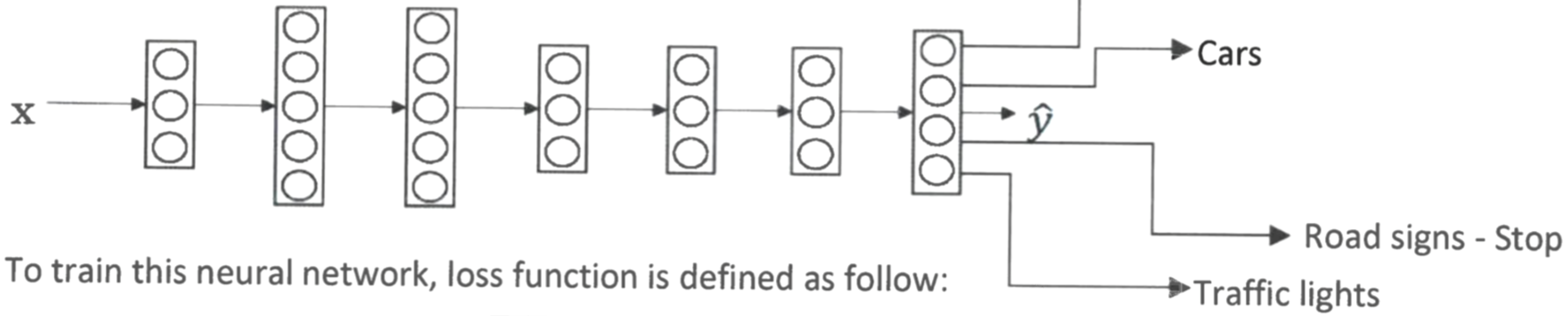


- use closely cropped images for the training set (object of interest centered)
- Define a sliding window
- Convolutional implementation

Multi-Task Learning

- NN does simultaneously several tasks

Neural Network architecture

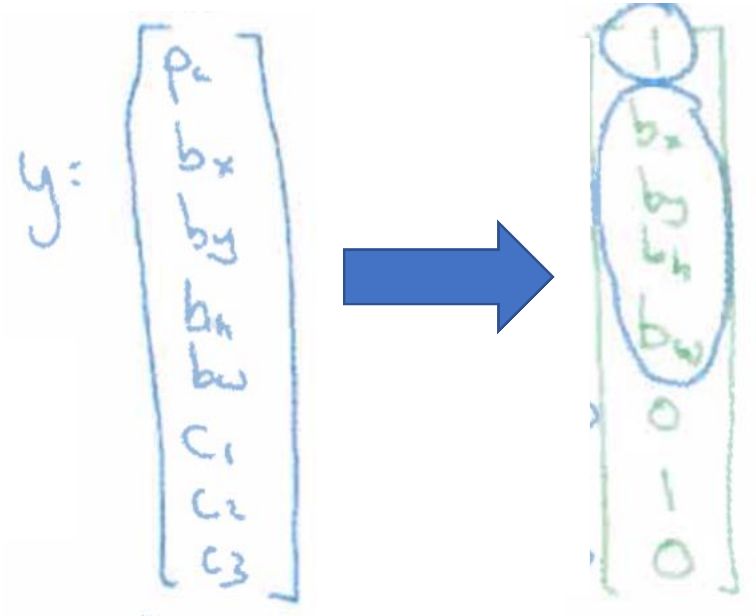


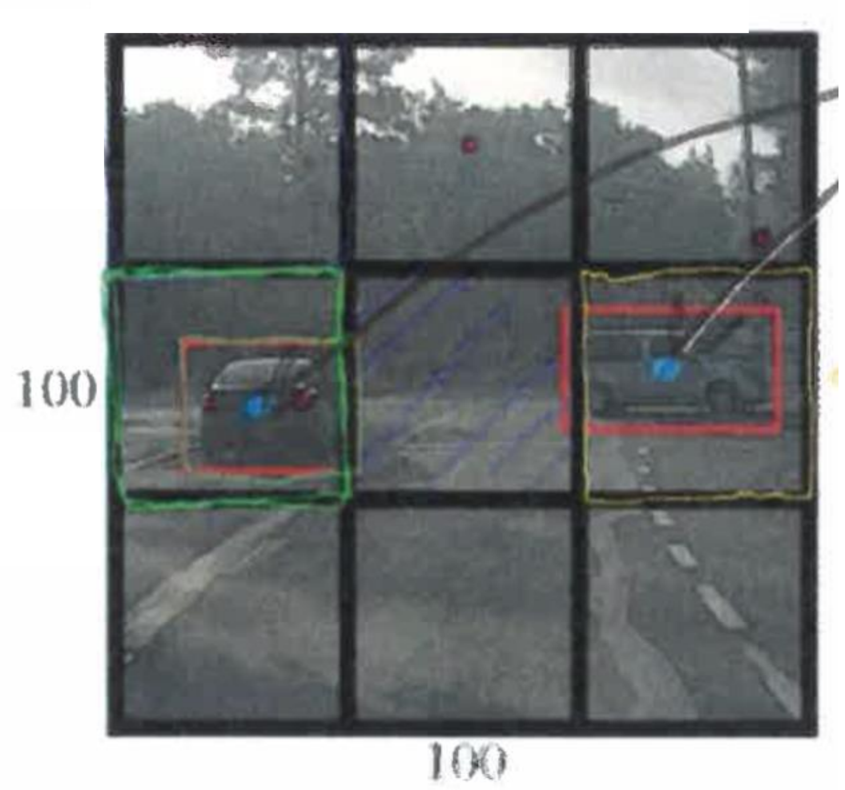
To train this neural network, loss function is defined as follow:

$$-\frac{1}{m}\sum_{i=1}^{m}\boxed{\sum_{j=1}^{4}}\left(y_j^{(i)}\log\left(\hat{y}_j^{(i)}\right)+\left(1-y_j^{(i)}\right)\log\left(1-\hat{y}_j^{(i)}\right)\right)$$

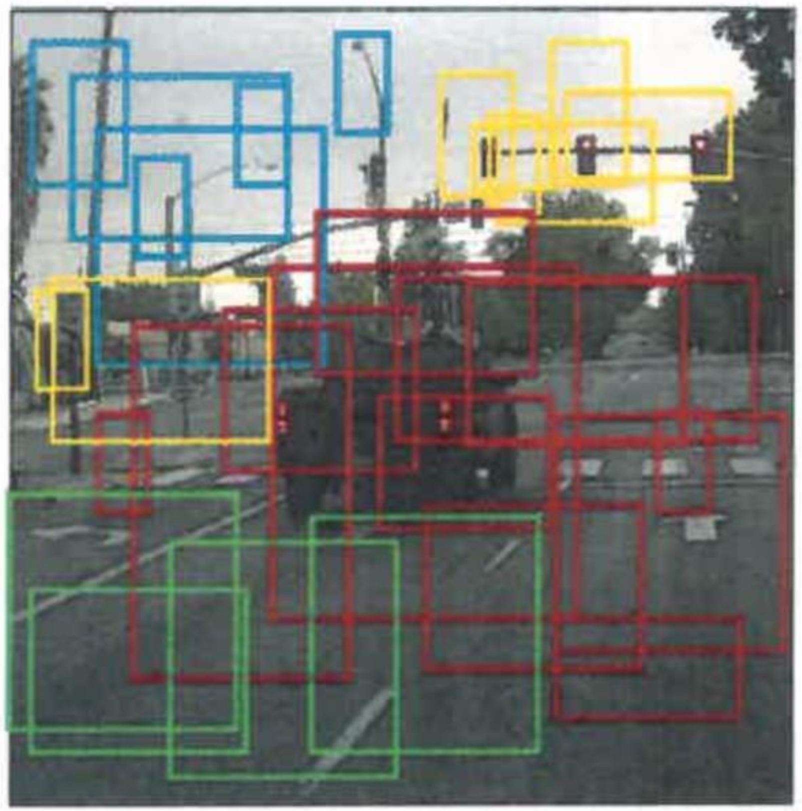
YOLO algorithm

- YOu Look Only once algorithm (YOLO)
- define a grid in the image
- apply the training to each cell





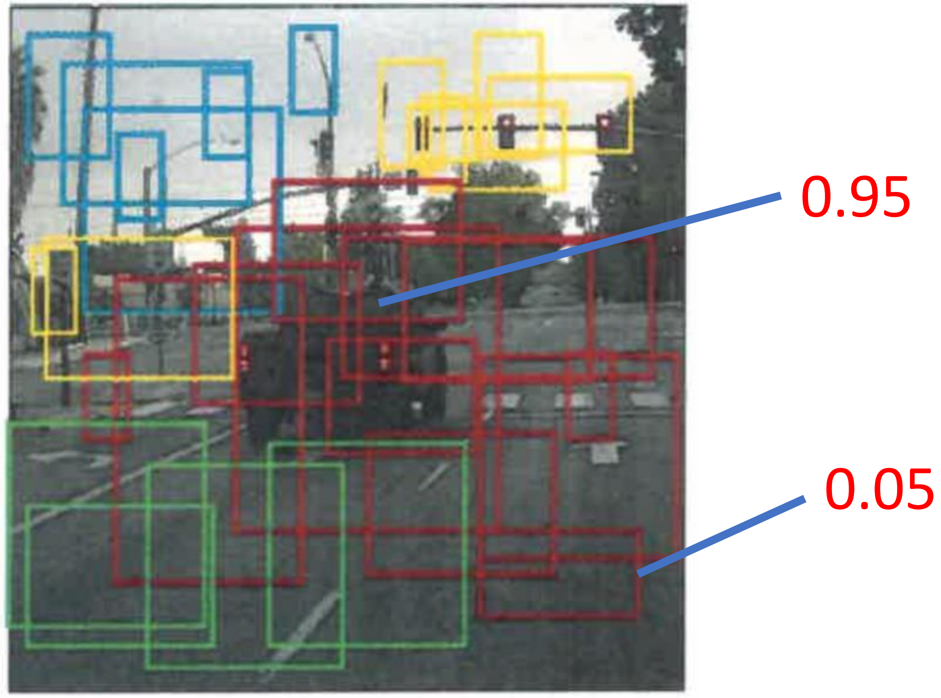
YOLO bounding boxes



- Filter the boxes using :
  1) score thresholding
  2) non-max suppression

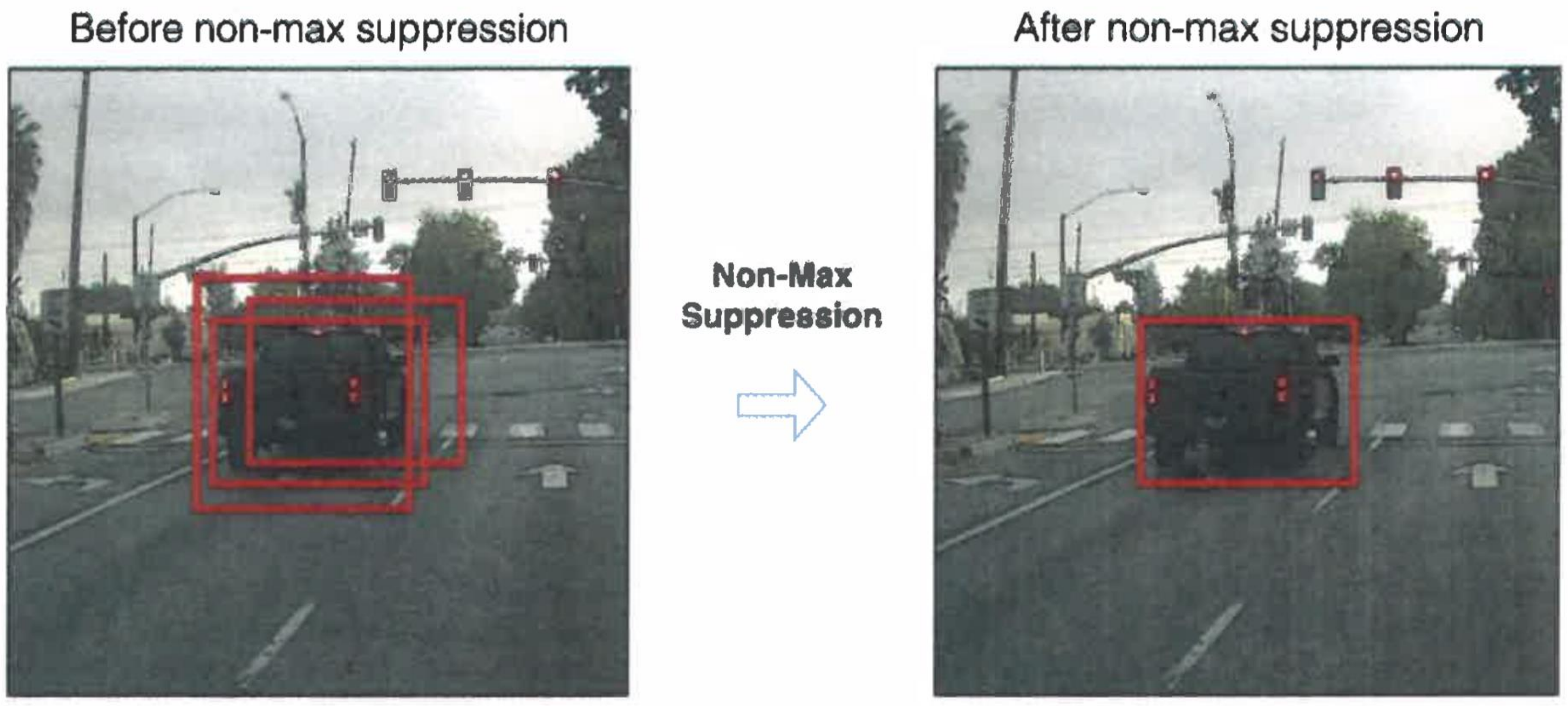
1. Score Thresholding

- Throw away boxes that have detected a class with a score less than the threshold (*0.6 for example*)



2. Non-max suppression

- ensures that an object is detected only once
  - keep the largest p_C output
  - discard any remaining box with intersection over union (IoU)>0.5

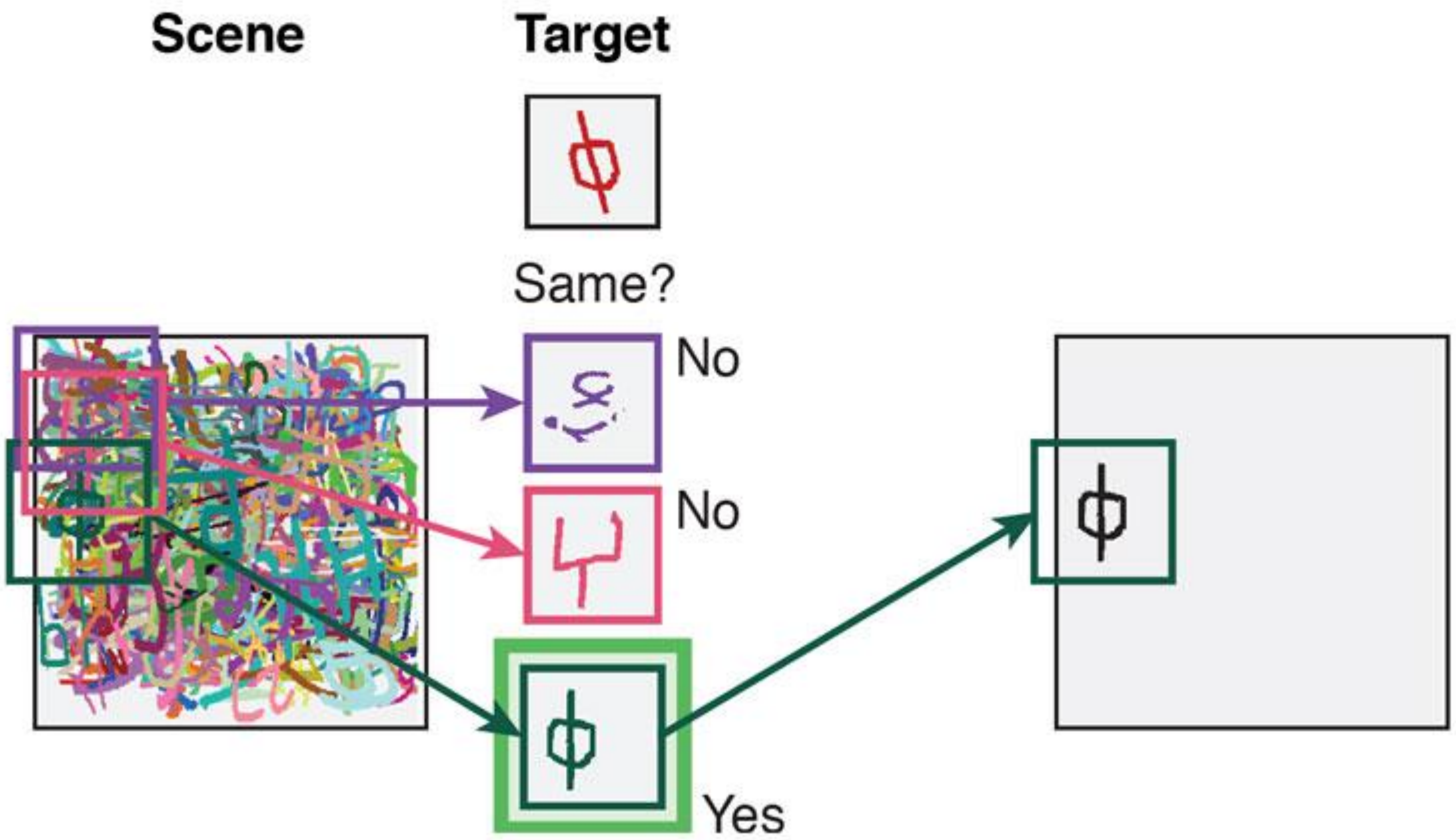


Face Detection

One-shot Learning

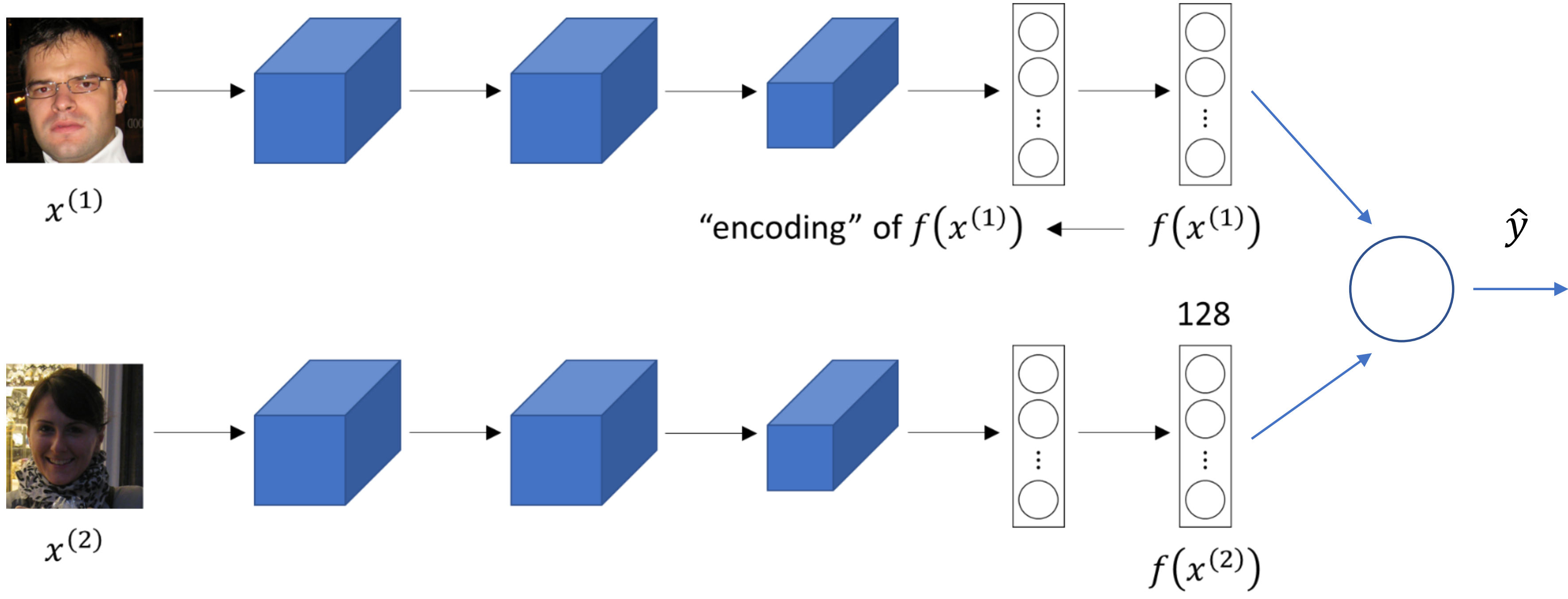
- Learn from one example to recognize the person again (very small training set)

- Learn a similarity function d (degree of difference between images)



Siamese Network

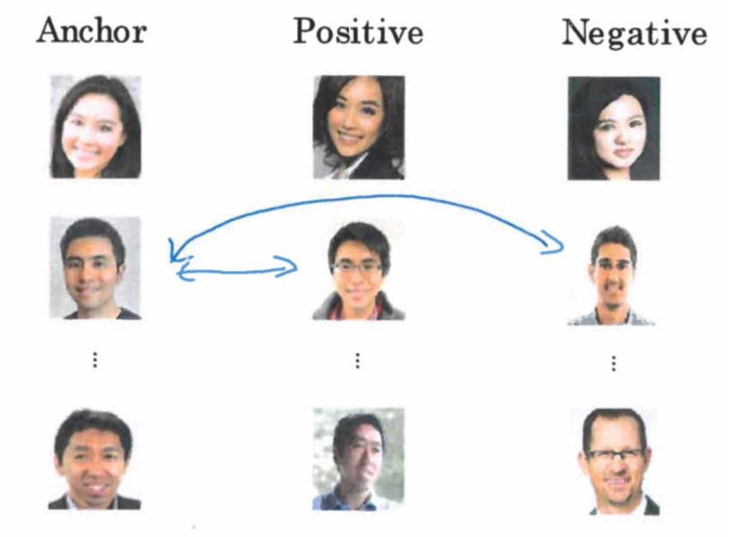
- Network used to learn the function d



$$d\left(x^{(1)}, x^{(2)}\right) = \left\| f\left(x^{(1)}\right) - f\left(x^{(2)}\right) \right\|^2$$

Triplet Loss

- Loss function where a baseline input (anchor) is compared to a positive input and a negative input :



- Often used for learning similarity
- Loss function is described using a Euclidean distance function

$$\mathcal{L}(A, P, N) = \max\Big(\| \mathrm{f}(A) - \mathrm{f}(P) \|^2 - \| \mathrm{f}(A) - \mathrm{f}(N) \|^2 + \alpha, 0 \Big)$$

- Cost function :

$$\mathcal{J} = \sum_{i=1}^{M} \mathcal{L}\left(A^{(i)}, P^{(i)}, N^{(i)} \right)$$

Security

Watch out with your algorithms !

Two-Minute Papers

Quiz

https://b.socrative.com/login/student/

Room : CONTI6128