AI TRAINING SERIES

AI ARTIFICIAL INTELLIGENCE

What is Artificial Intelligence? How Does AI Work?

CONTENTS

CONVOLUTIONAL NEURAL NETWORK

- CONVOLUTIONAL LAYER
- POOLING LAYER
- FC LAYER

CNN USE CASES

- CLASSIFICATION
- DETECTION&SEGMENTATION

PROBLEMS ON ANN

REQUIRES MANY PARAMETERS

28x28 image → (28×28+1)×10=7,850 ≈ 7.8K

640x480 image → (640×480+1)×10=3,072,010 ≈ 3M

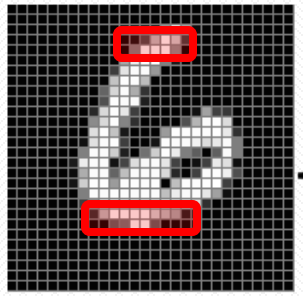
REQUIRES LOTS OF MEMORY AND COMPUTATION COST

WEIGHTS BIAS

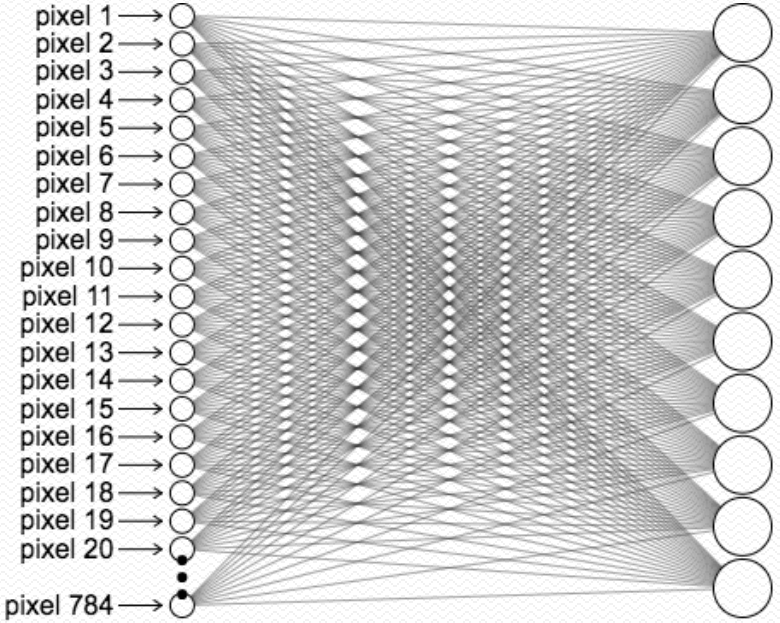
$$(N_{in} + 1) \cdot N_{out}$$

TAKES 1D FORM OF INPUT ONLY

Lose spatial information!



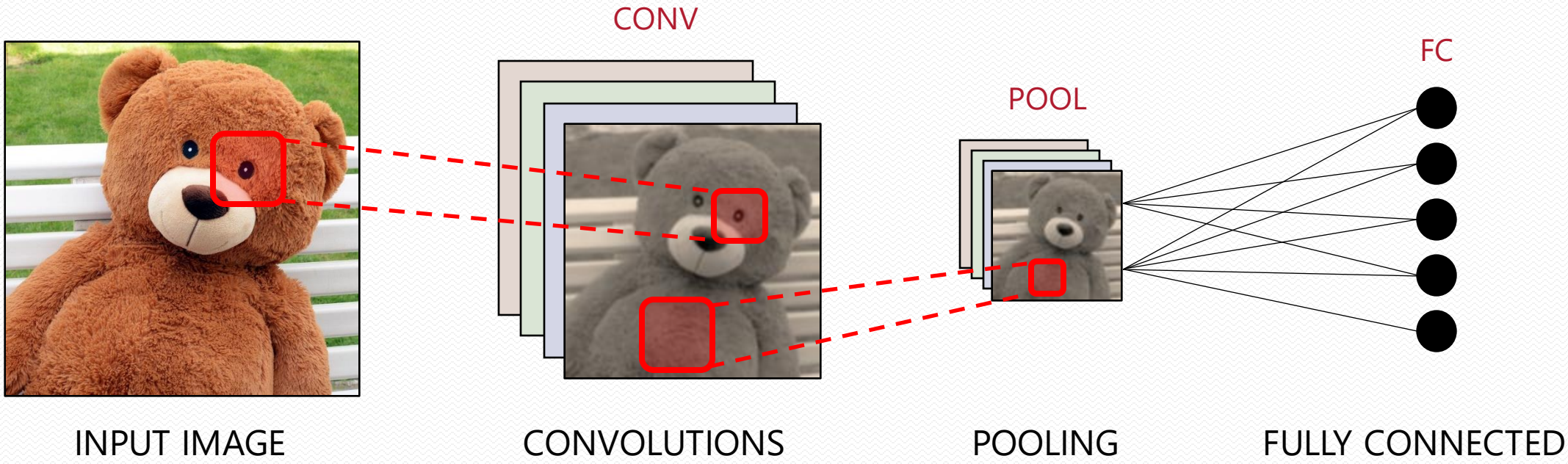
MISSING SPATIAL RELATIONSHIP



CONVOLUTIONAL NEURAL NETWORK

SPECIFIC TYPE OF NEURAL NETWORKS

MAINLY USED FOR IMAGE CLASSIFICATION
GENERALLY COMPOSED OF THE 3 LAYERS



CONVOLUTION

MATHEMATICAL OPERATION

Slides one function over another
e.g., Photo filters

IMAGE

$1_{\times 1}$	$1_{\times 0}$	$1_{\times 1}$	0	0
$0_{\times 0}$	$1_{\times 1}$	$1_{\times 0}$	1	0
$0_{\times 1}$	$0_{\times 0}$	$1_{\times 1}$	1	1
0	0	1	1	0
0	1	1	0	0

CONVOLVED FEATURE

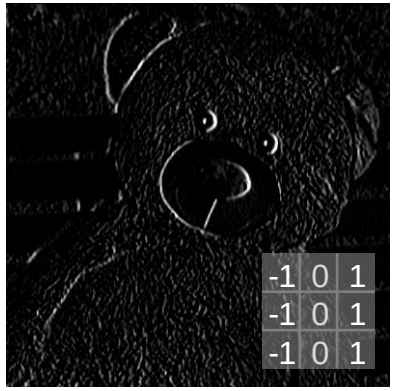
4		

ORIGINAL

SHARPEN

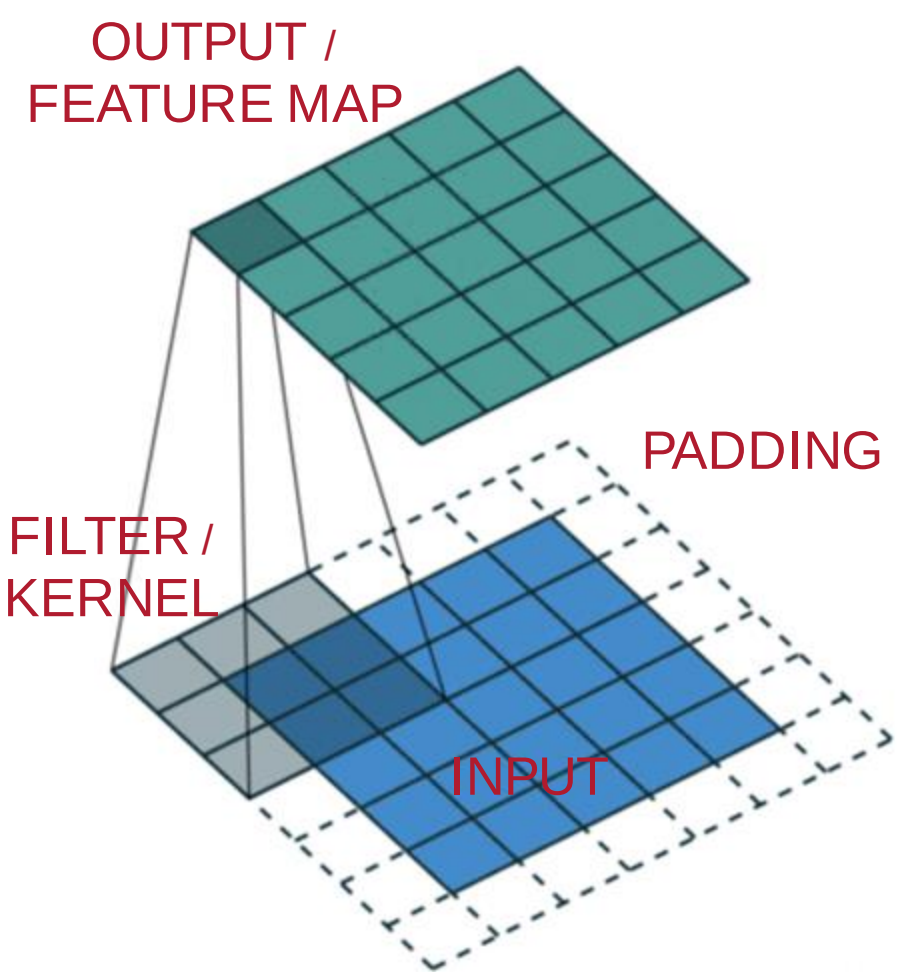
EMBOSS

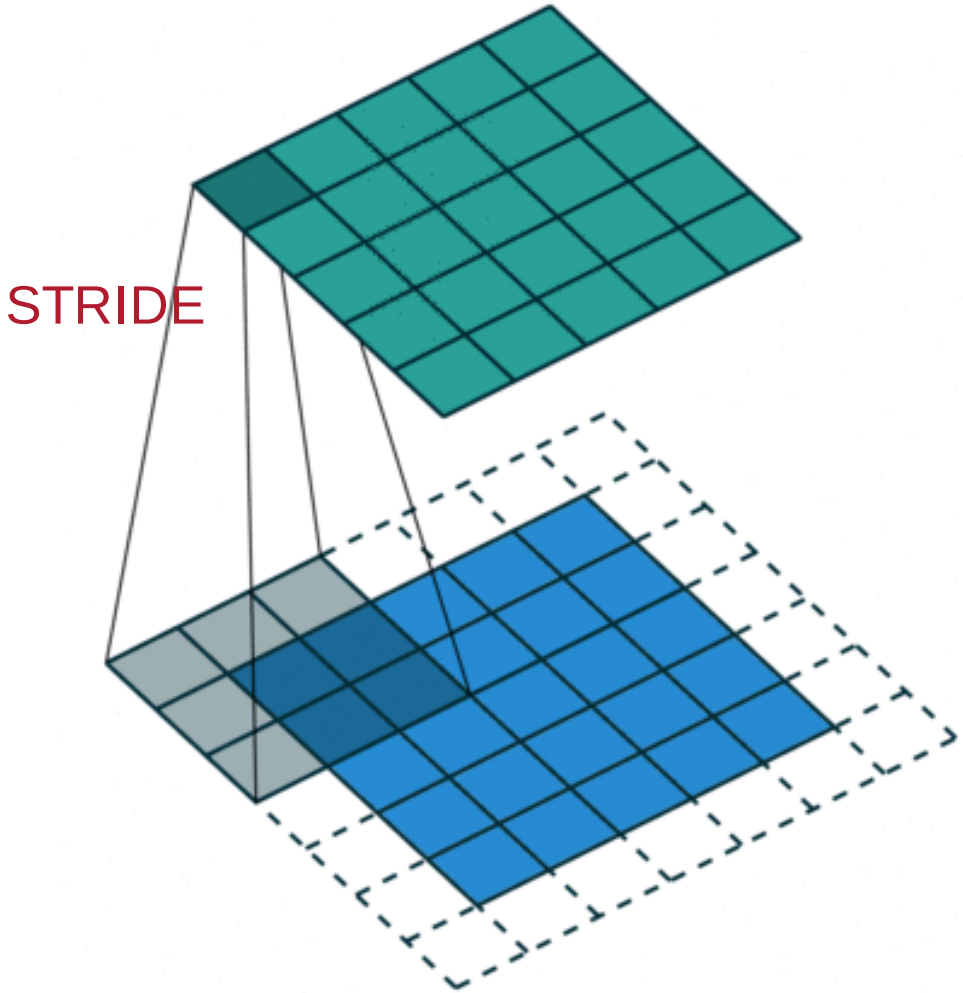
EXTRACT HORIZONTAL LINE



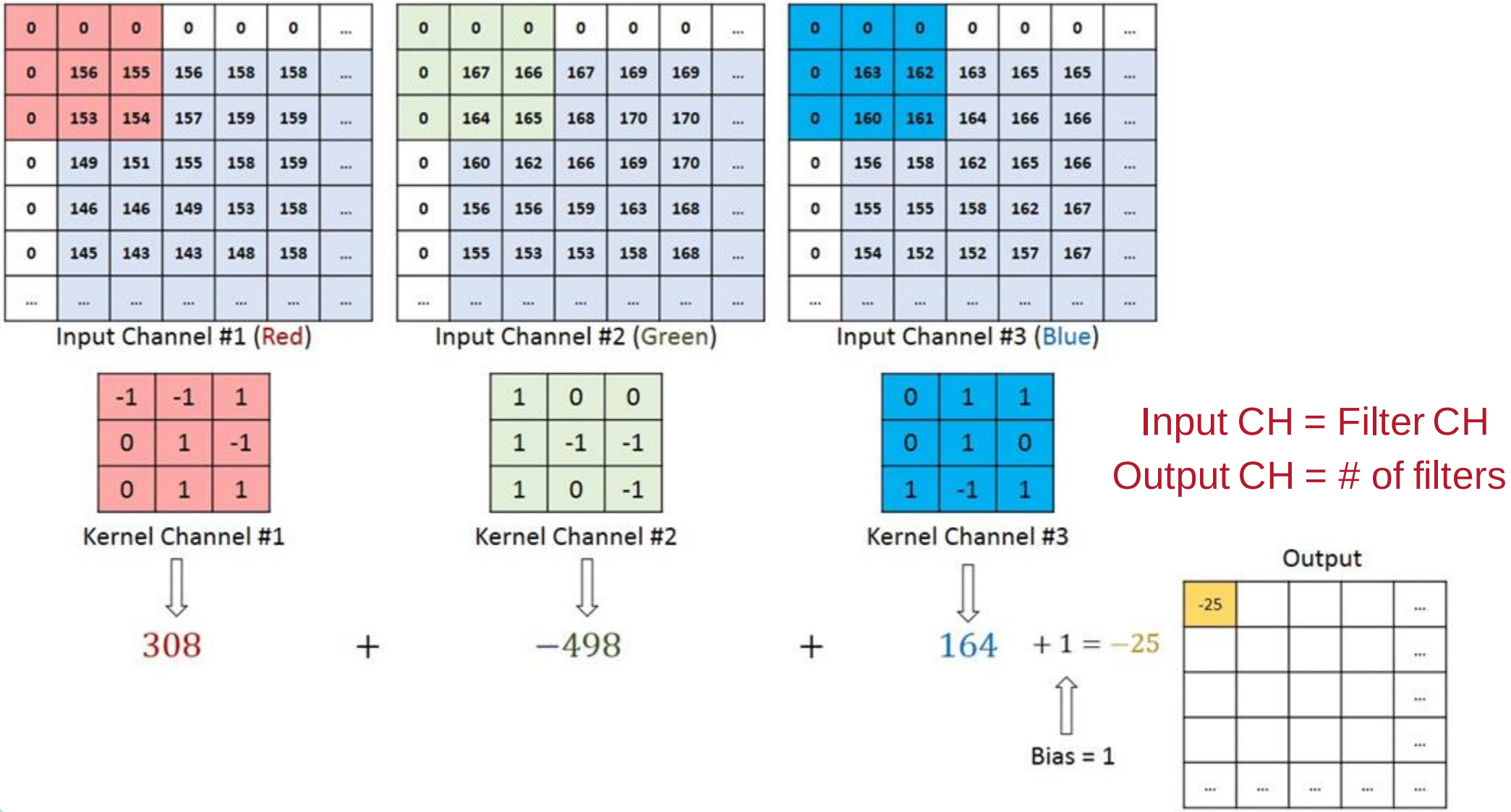
EXTRACT VERTICAL LINE

CONVOLUTIONAL LAYER / CONV





CONVOLUTION : MULTI-CHANNEL



POOLING LAYER / POOL

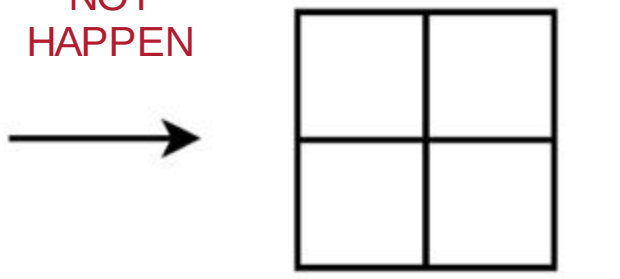
REDUCE THE SPATIAL SIZE

CONTROL OVERFITTING

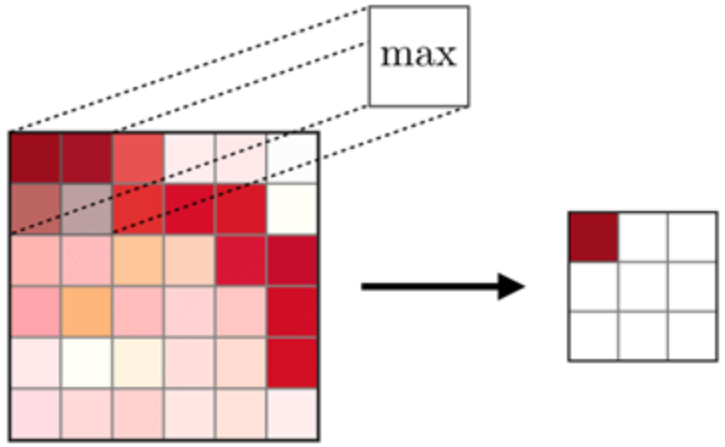
1	1	2	4
5	6	7	8
3	2	1	0
1	2	3	4

FEATURE MAP

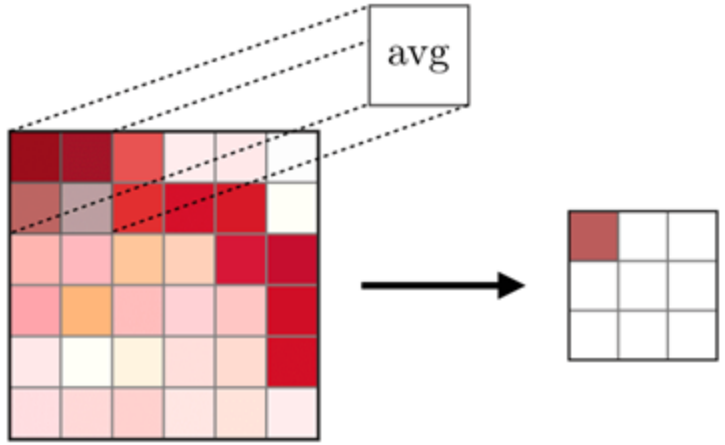
TRAINING NOT HAPPEN



POOLED FEATURE MAP

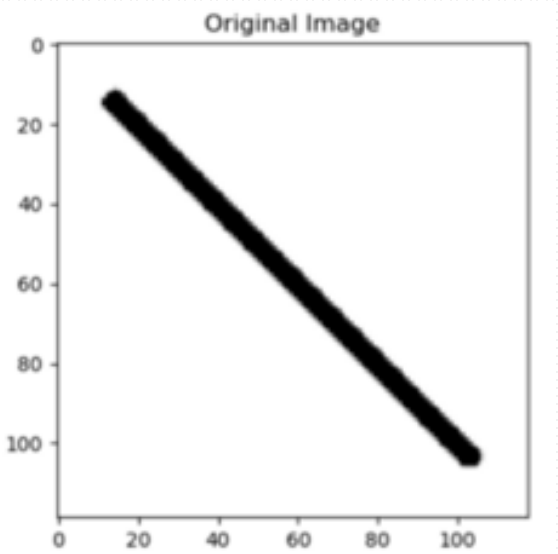


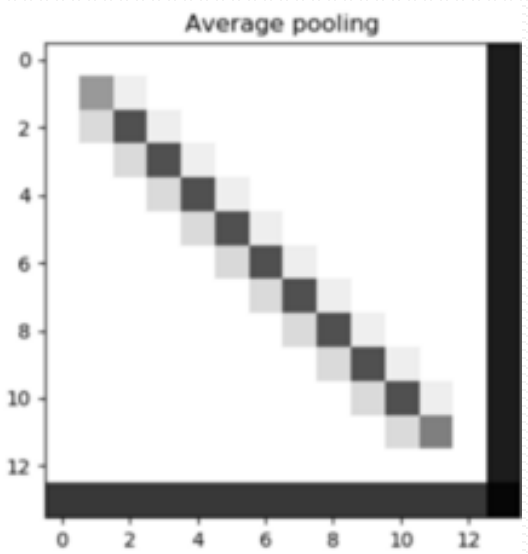
MAX POOLING

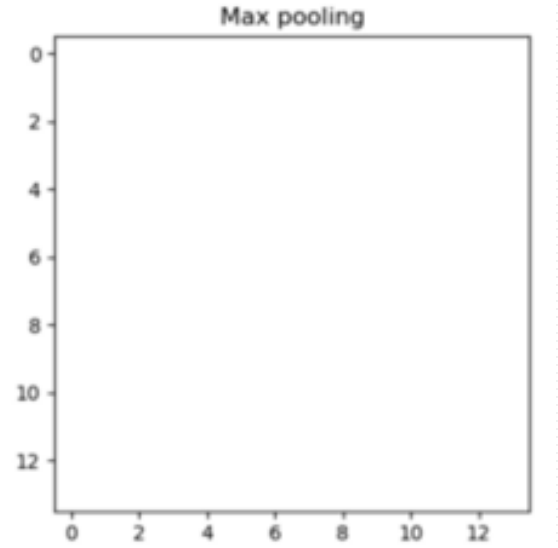


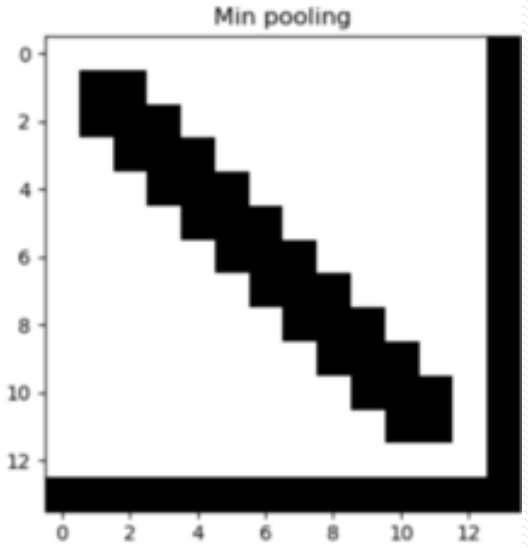
AVERAGE POOLING

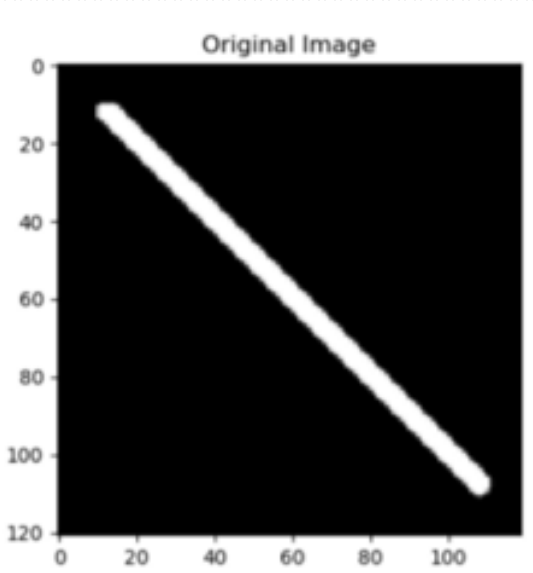
Compare Pooling algorithm (cont.)

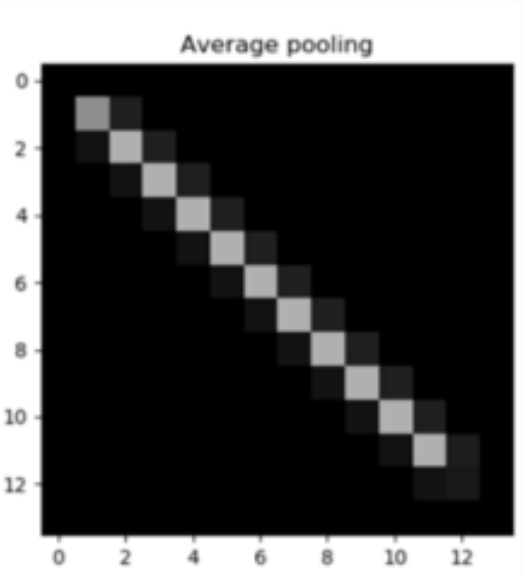


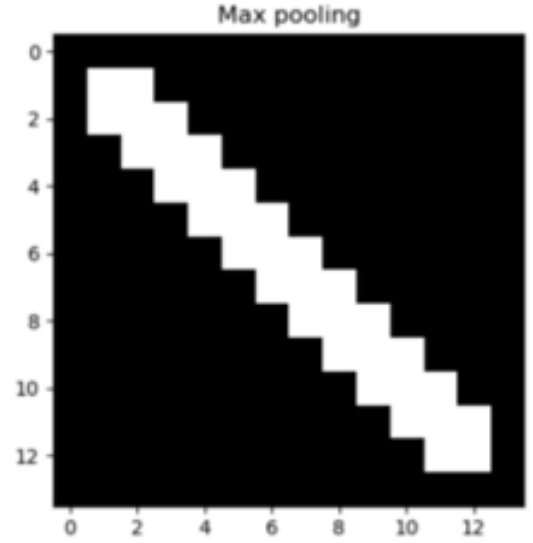


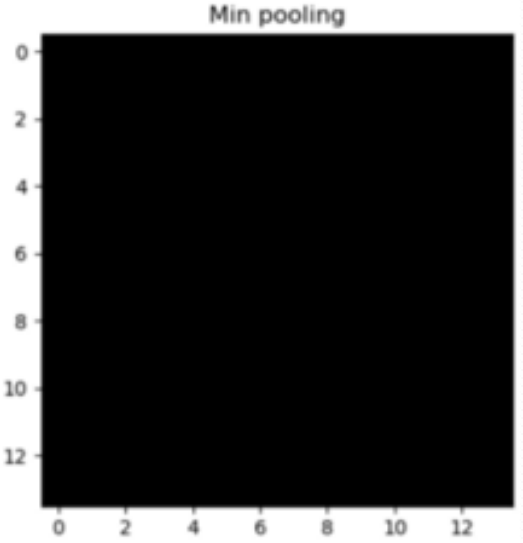




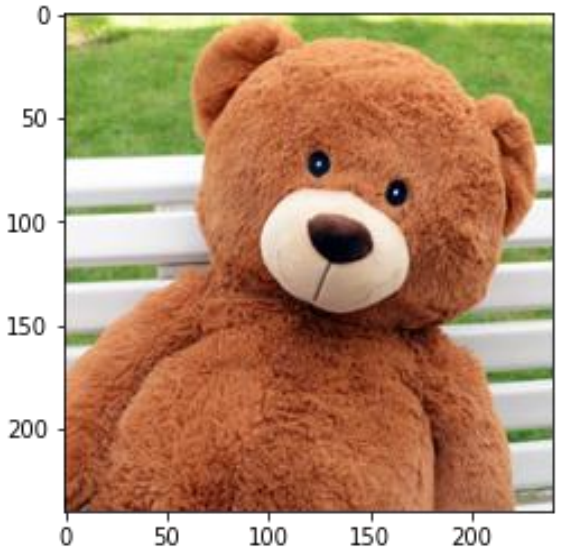




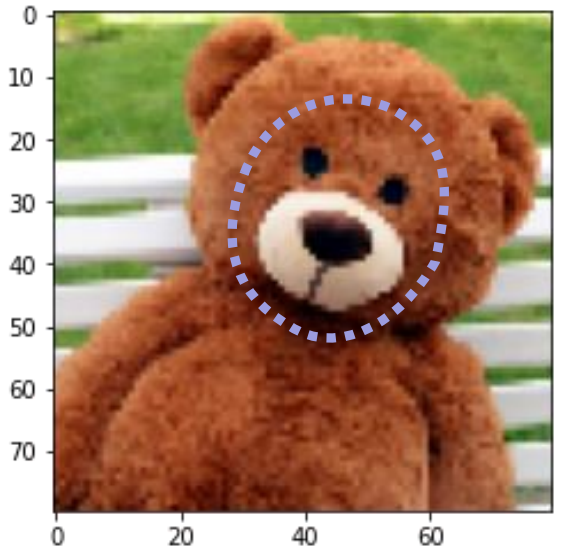




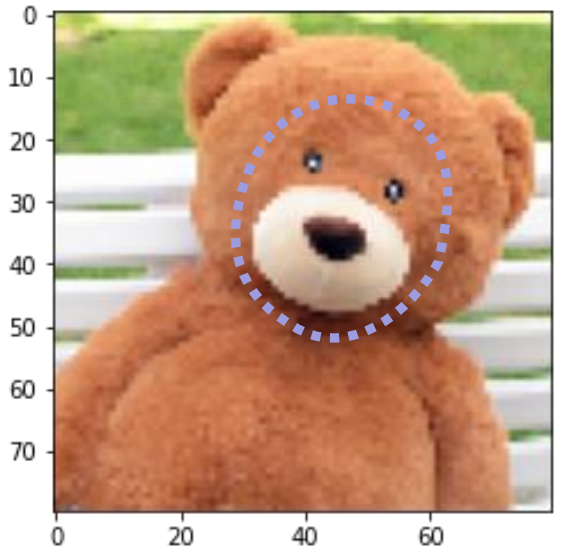
COMPARE POOLING ALGORITHM



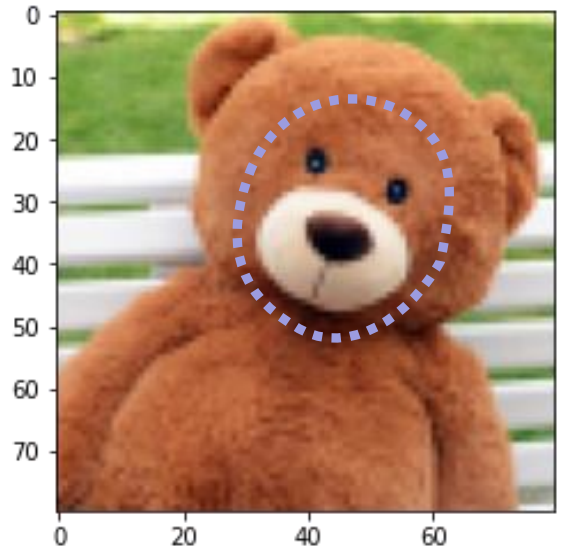
ORIGINAL



MIN

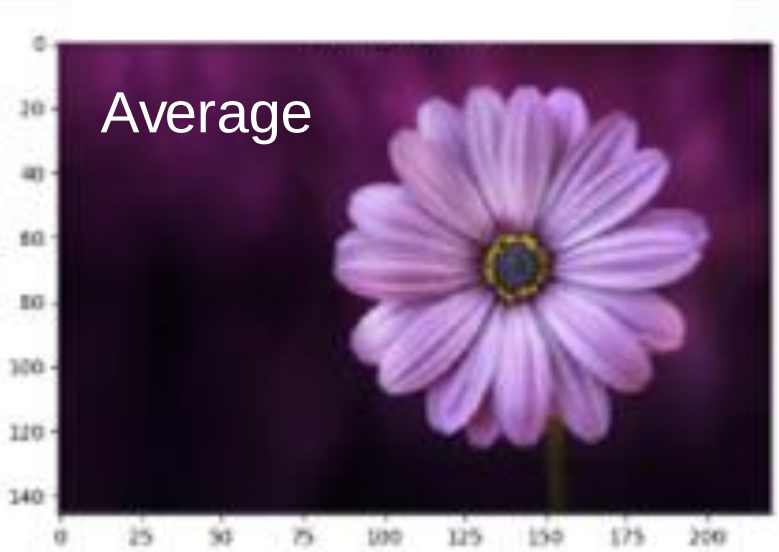


MAX



AVERAGE

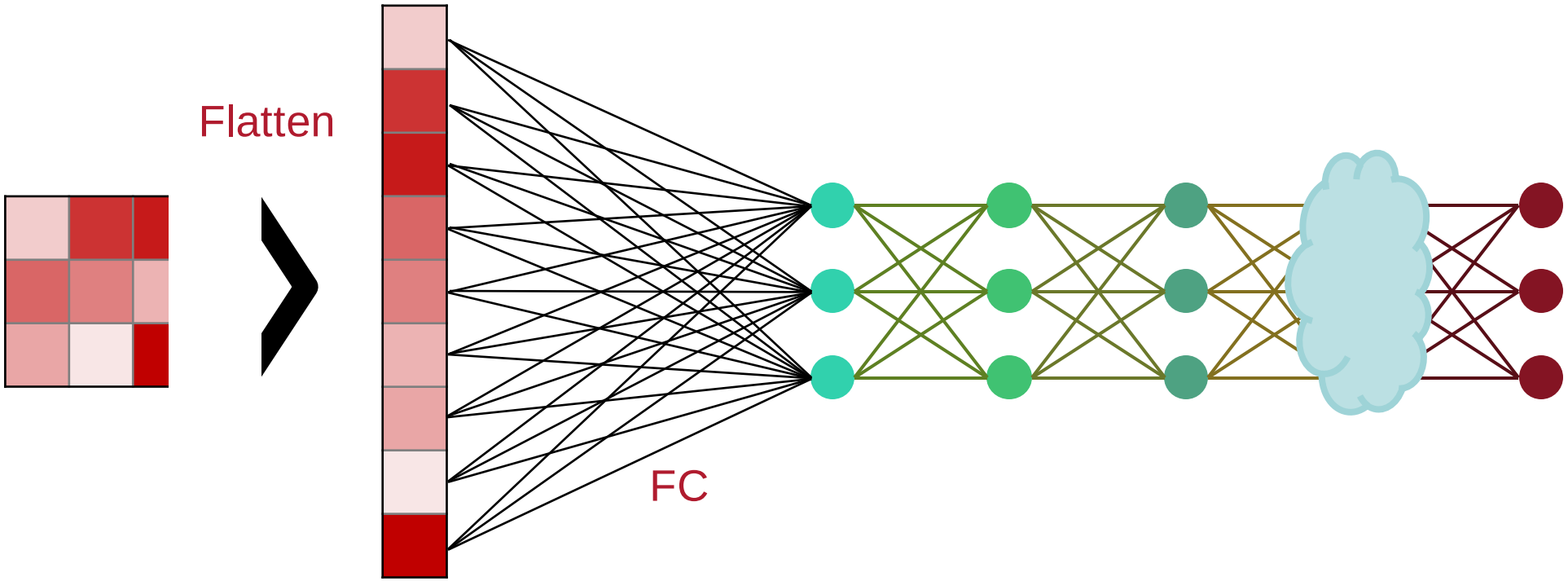
Compare Pooling algorithm



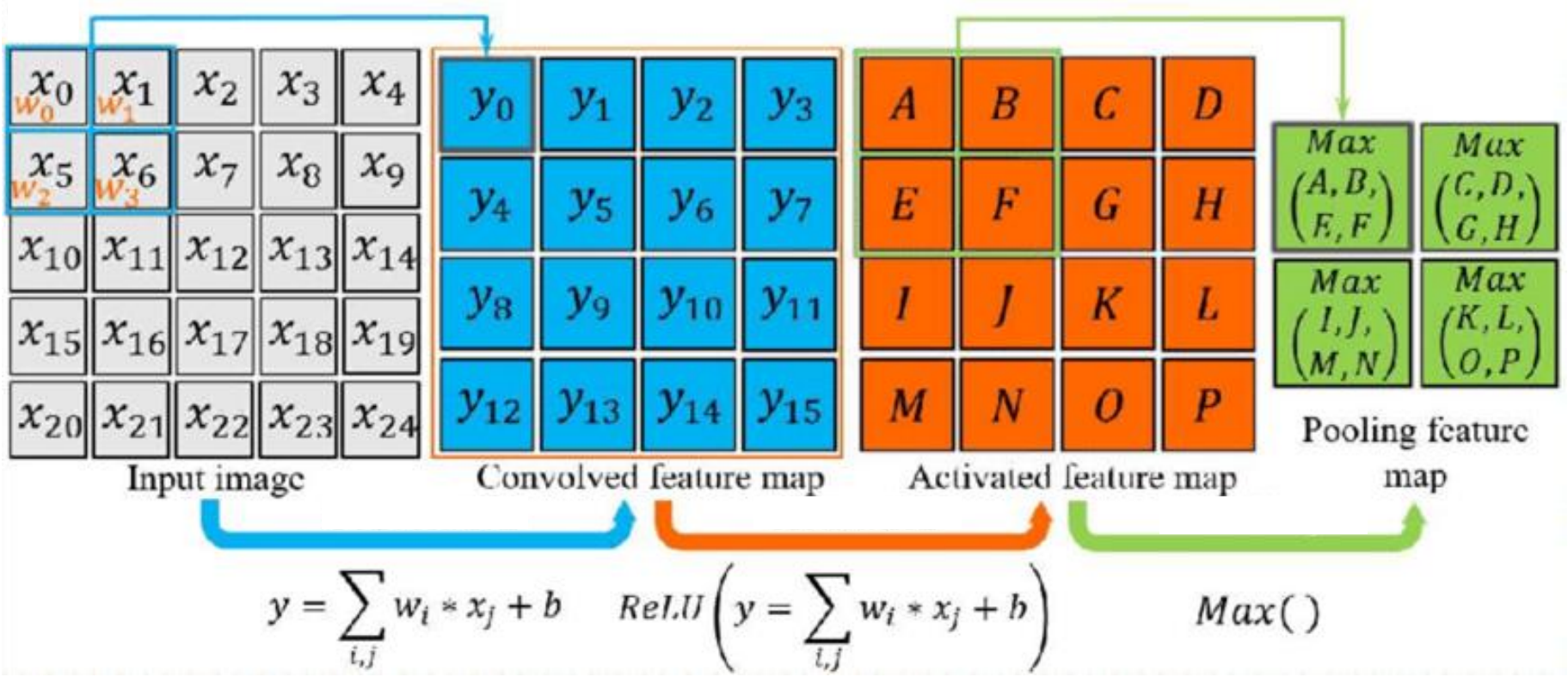
FULLY CONNECTED LAYER / FC

WORKS ON FLATTENED INPUTS WHERE EACH INPUT CONNECTS TO ALL NEURONS

FC LAYERS ARE USUALLY FOUND TOWARDS THE END OF CNN ARCHITECTURES

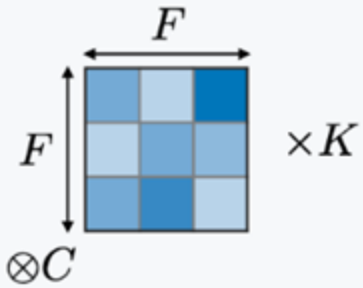
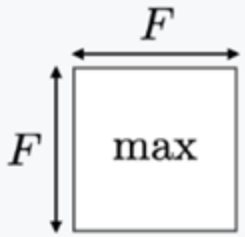
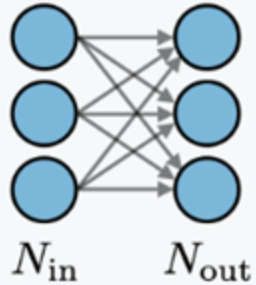


PUT ALL TOGETHER

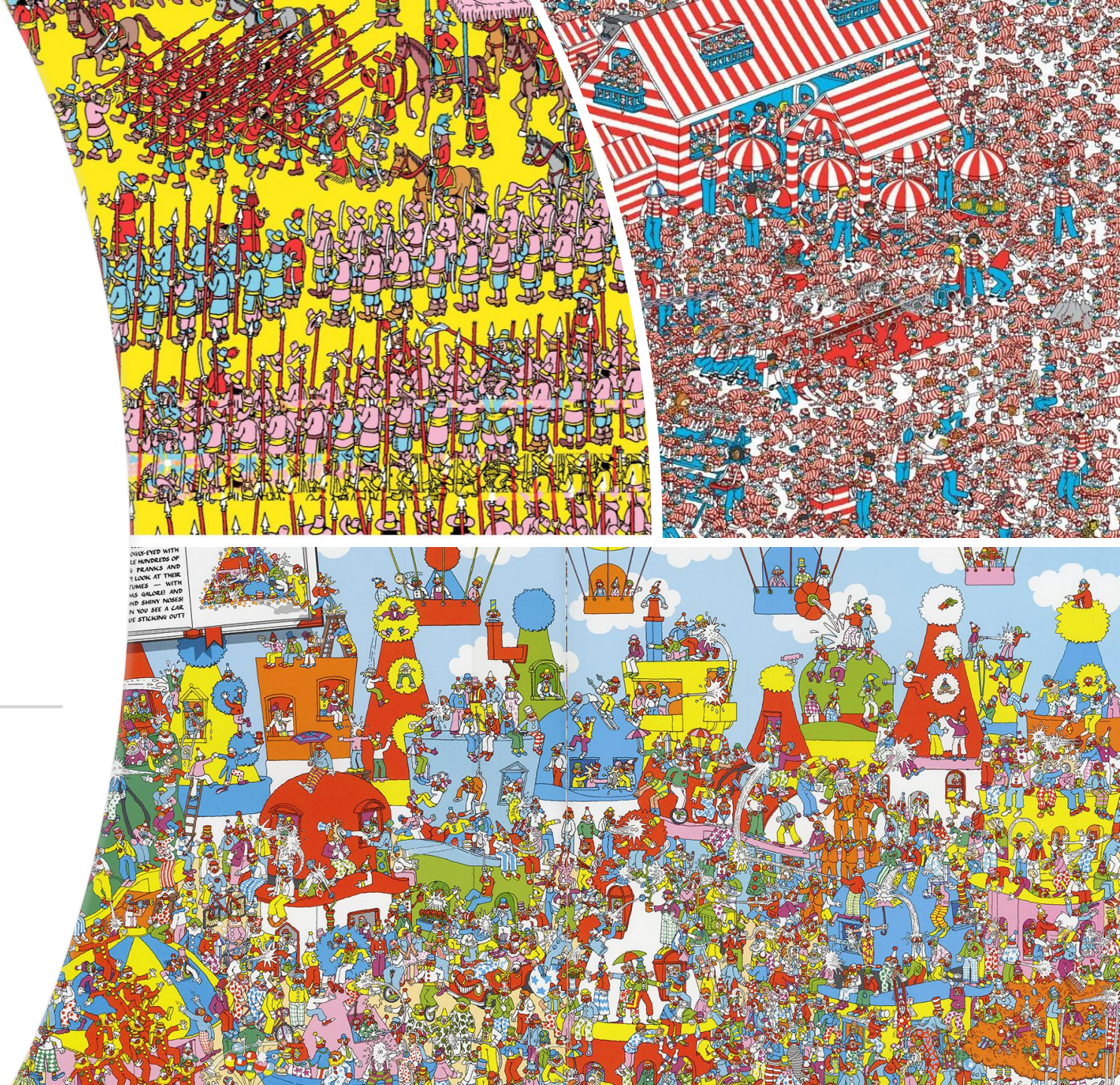


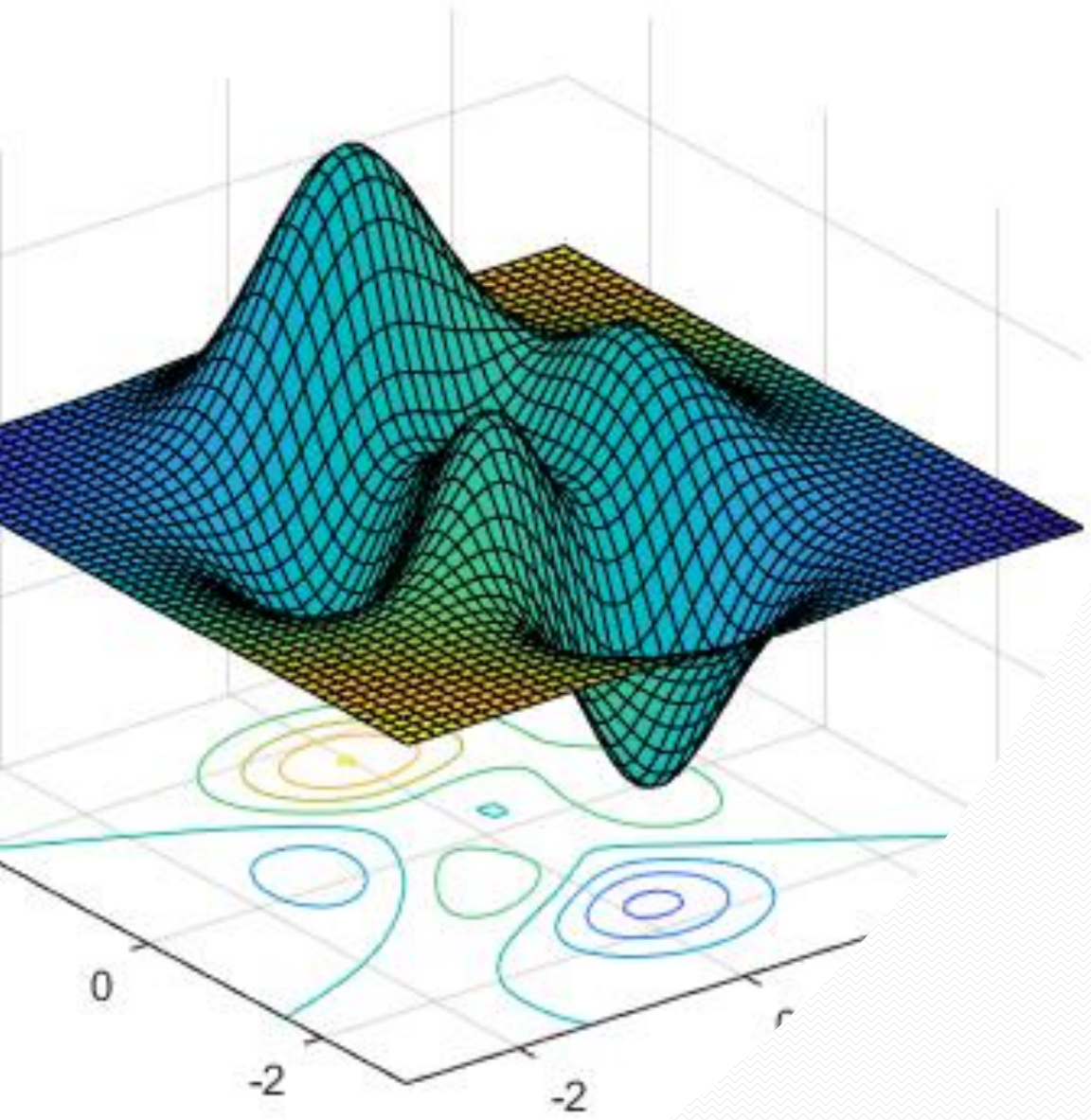
CONVOLUTION ACTIVATION POOLING

Determine the *N* of parameters

	CONV	POOL	FC
Illustration			
Input size	$I \times I \times C$	$I \times I \times C$	N_{in}
Output size	$O \times O \times K$	$O \times O \times C$	N_{out}
Number of parameters	$(F \times F \times C + 1) \cdot K$	0	$(N_{\text{in}} + 1) \times N_{\text{out}}$
Remarks *S*: Stride	• One bias parameter per filter • In most cases, $S < F$ • A common choice for K is $2C$	• Pooling operation done channel-wise • In most cases, $S = F$	• Input is flattened • One bias parameter per neuron • The number of FC neurons is free of structural constraints

월리를 찾아라 실습

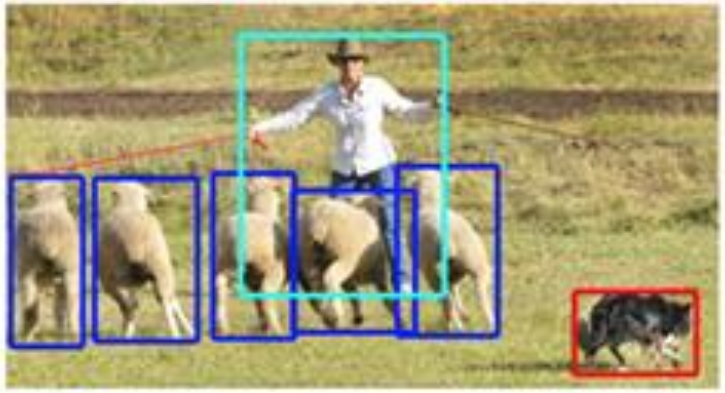




USE CASES

CNN USE CASES

CLASSIFICATION



DETECTION

SEGMENTATION

NEURAL STYLE TRANSFER

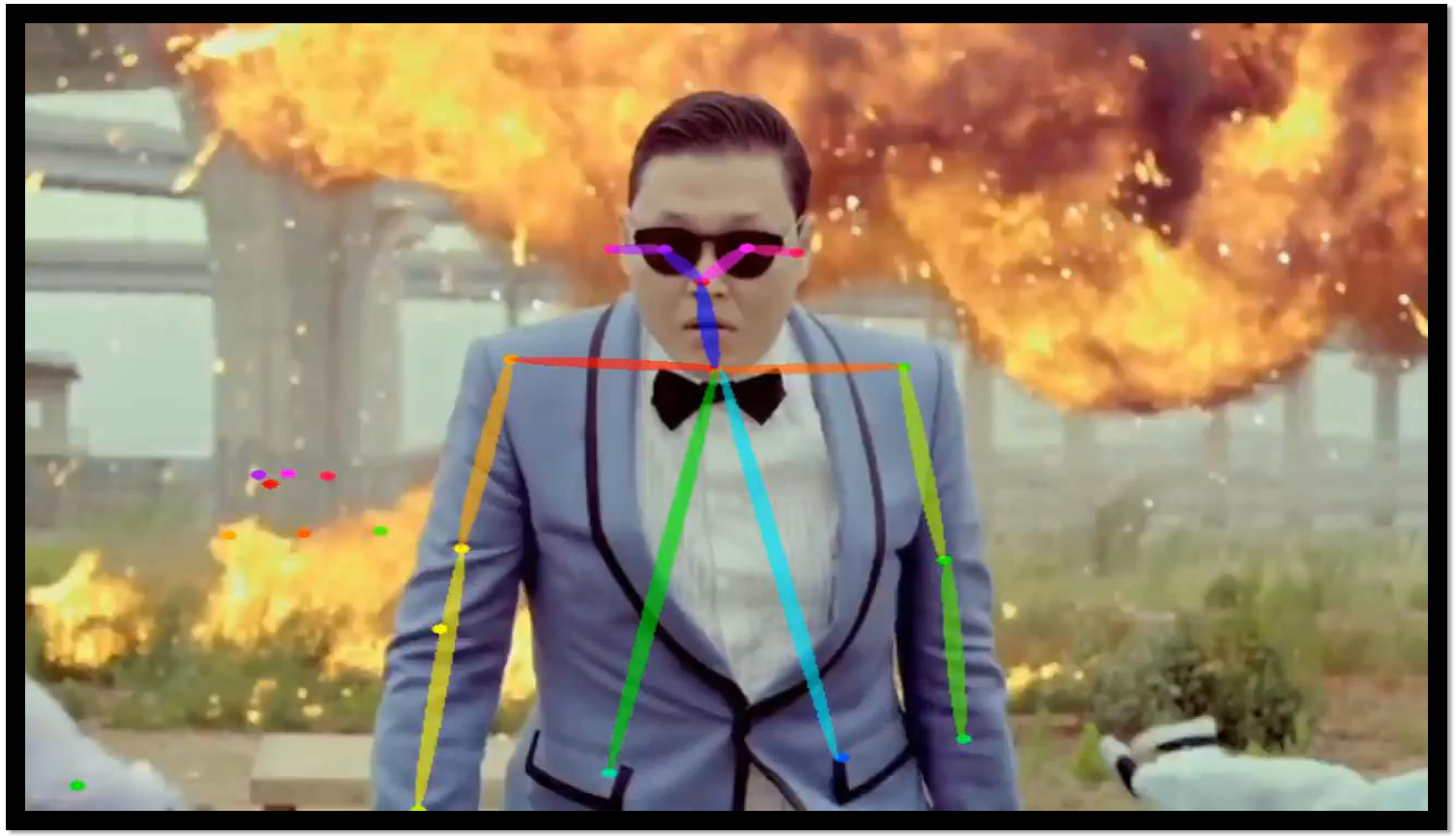
INPAINTING

COLORIZATION

CNN USE CASES / CONT.

SUPER RESOLUTION

MONO DEPTH



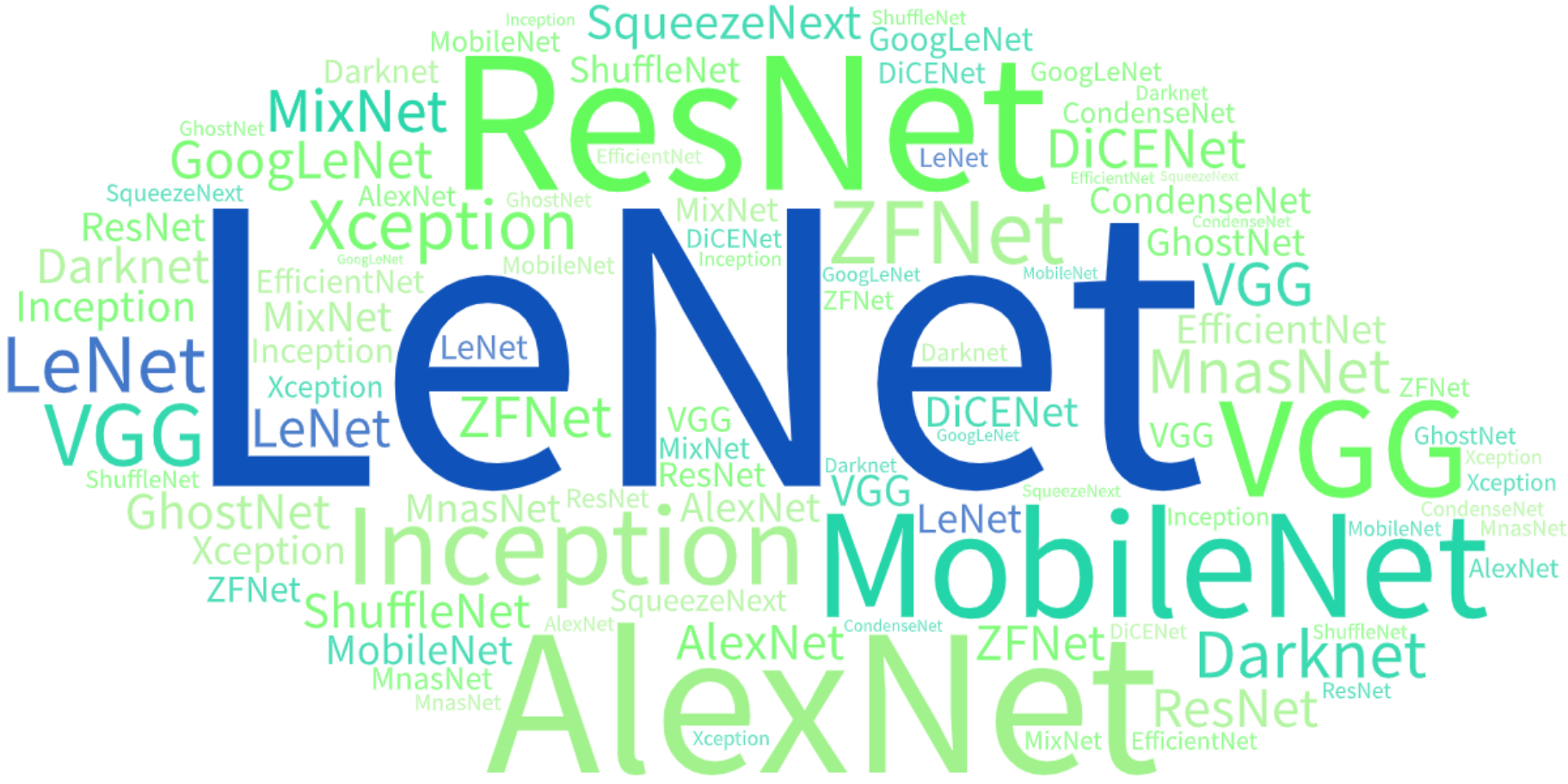
HUMAN POSE ESTIMATION

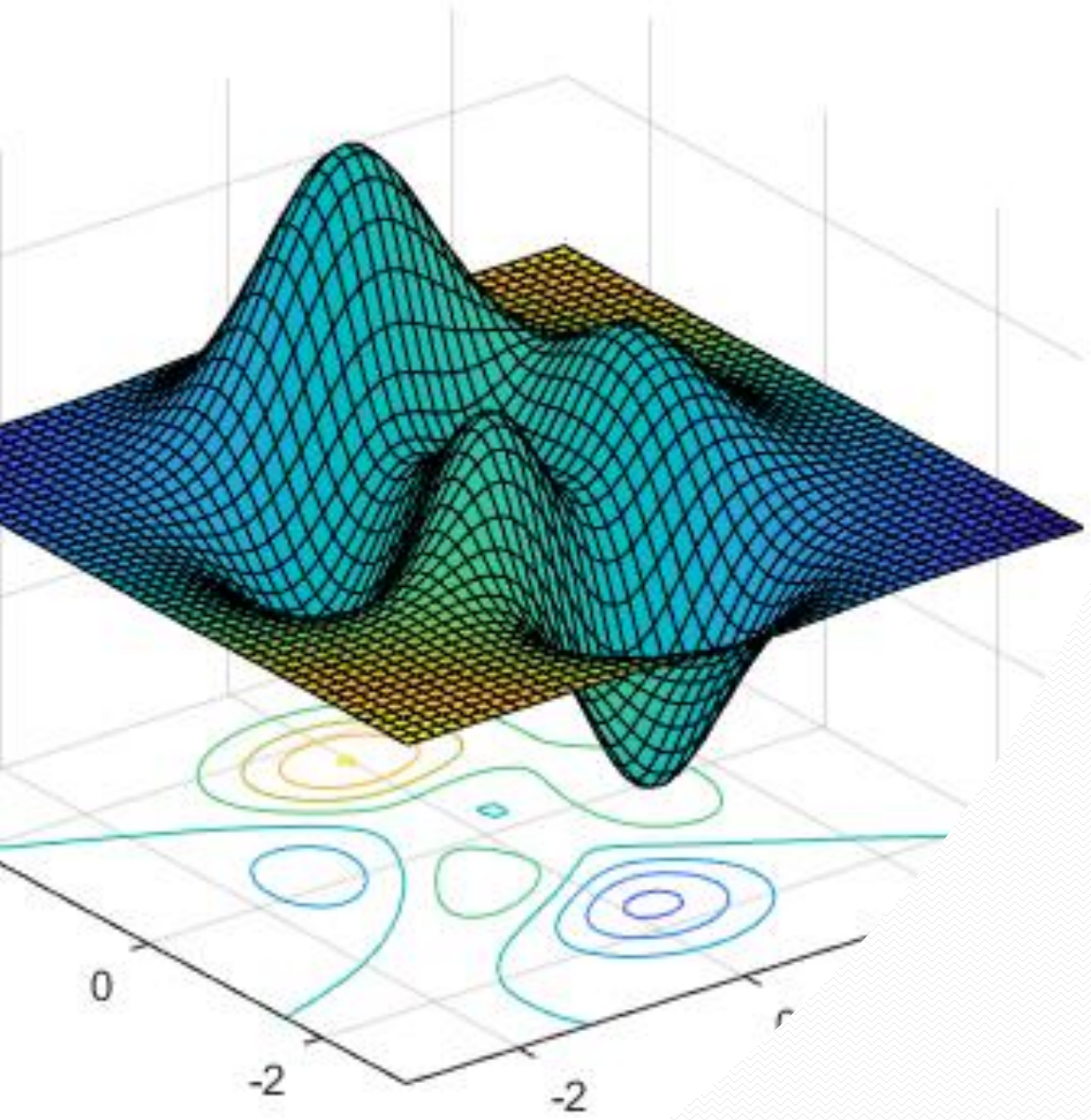
CNN USE CASES / CONT.

A young boy is playing basketball. 	Two dogs play in the grass. 	A dog swims in the water. 	A little girl in a pink shirt is swinging. 
A group of people walking down a street. 	A group of women dressed in formal attire. 	Two children play in the water. 	A dog jumps over a hurdle. 

IMAGE CAPTIONING

NETWORKS WITH CNN





Classification

IMAGENET LARGE SCALE VISUAL RECOGNITION CHALLENGE ILSVRC

2010 - NEC-UIUC Lin et al.

2011 - XRCE Florent Perronnin, Jorge Sanchez

2012 - AlexNet

2013 - ZFNet

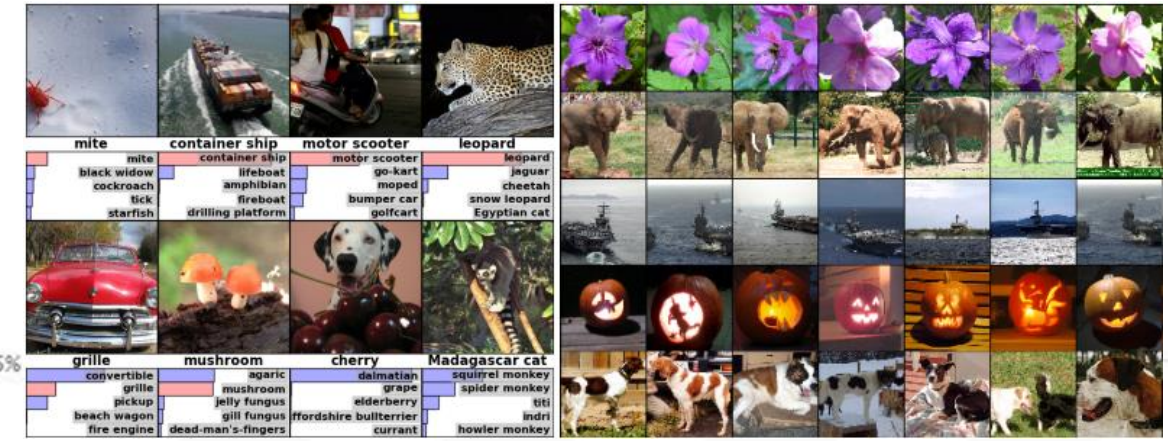
2014 – GoogLeNet

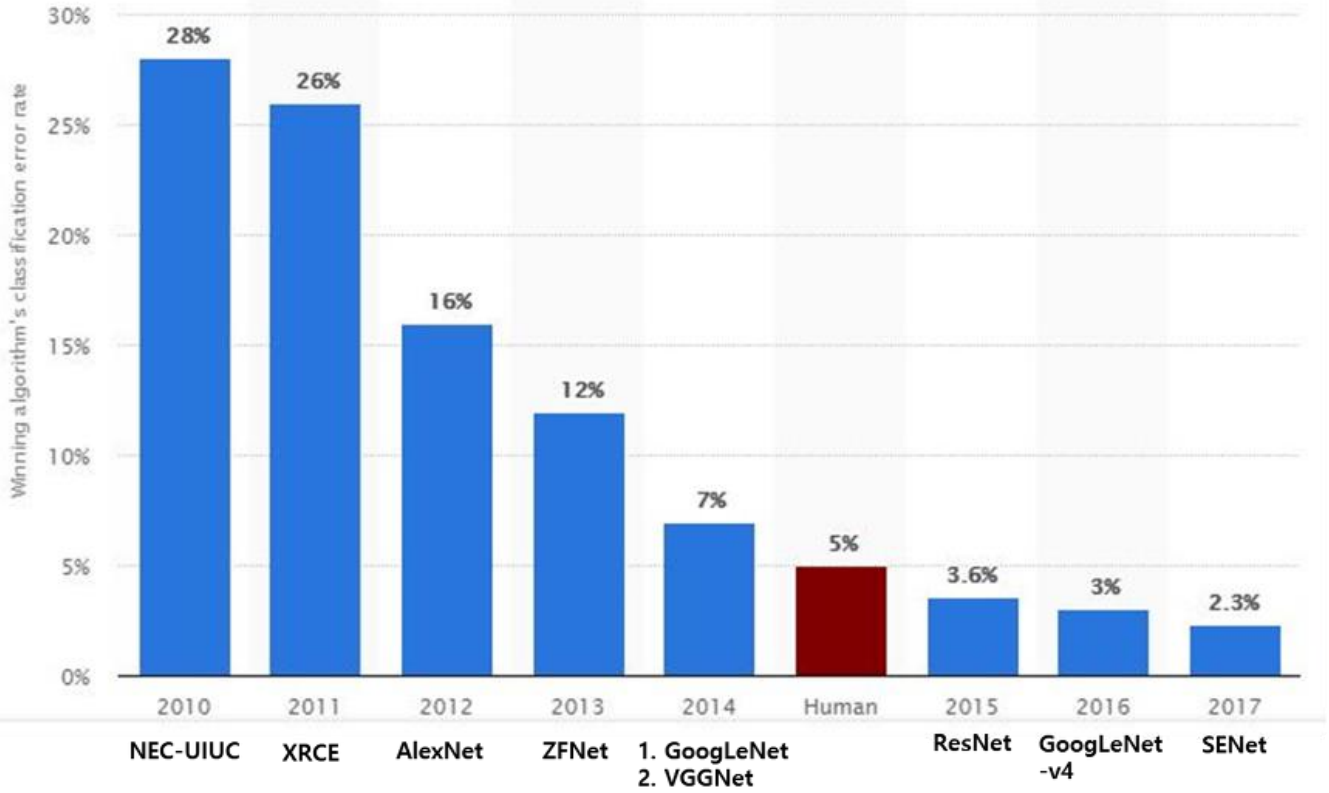
VGGNet Second Winner

2015 - ResNet

2016 - GoogLeNet-v4

2017 - SENet





LeNET-5 ARCHITECTURE

First CNN

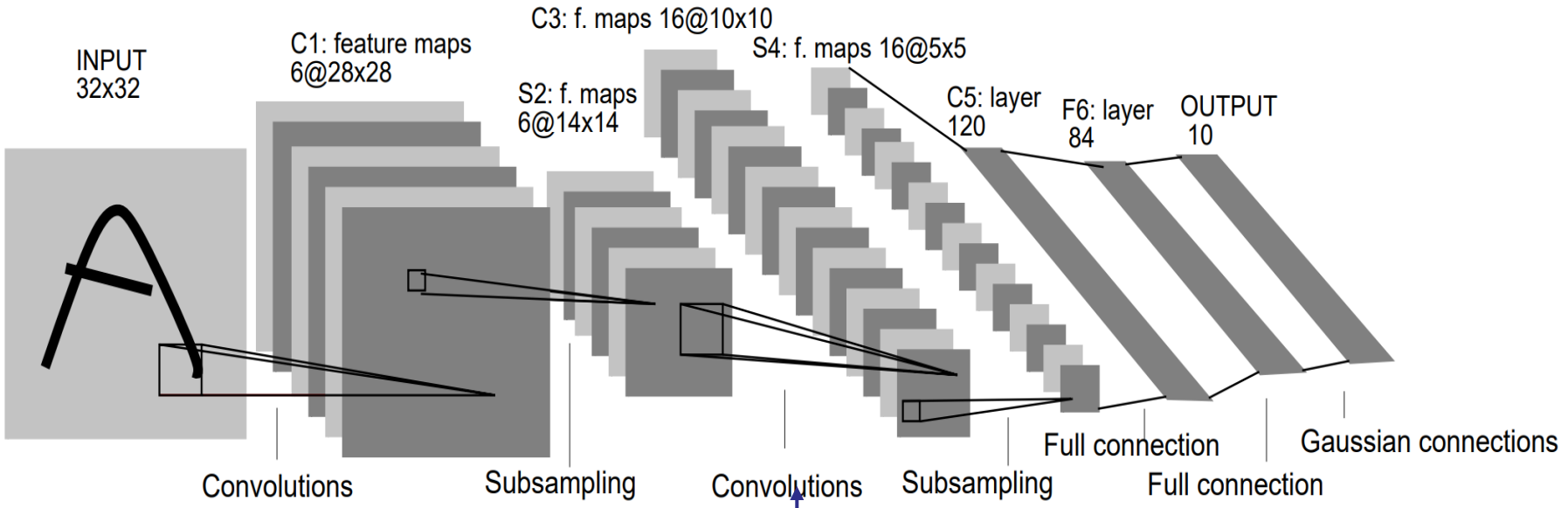


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	X				X	X	X			X	X	X	X		X	X
1	X	X				X	X	X			X	X	X	X		X
2	X	X	X				X	X	X			X		X	X	X
3		X	X	X			X	X	X	X			X		X	X
4			X	X	X			X	X	X	X		X	X		X
5				X	X	X			X	X	X	X		X	X	X

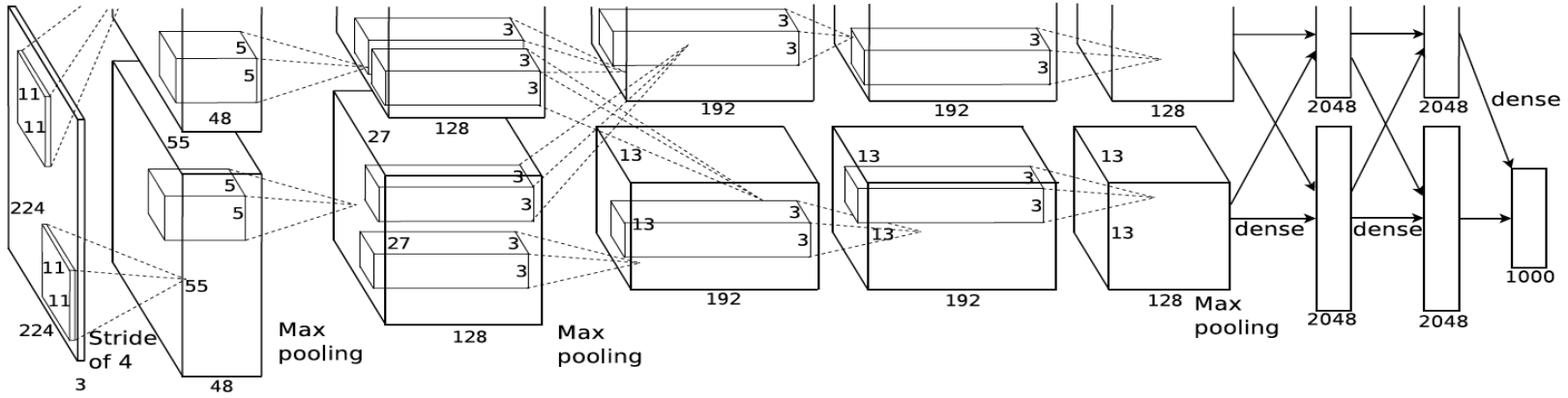
TABLE I

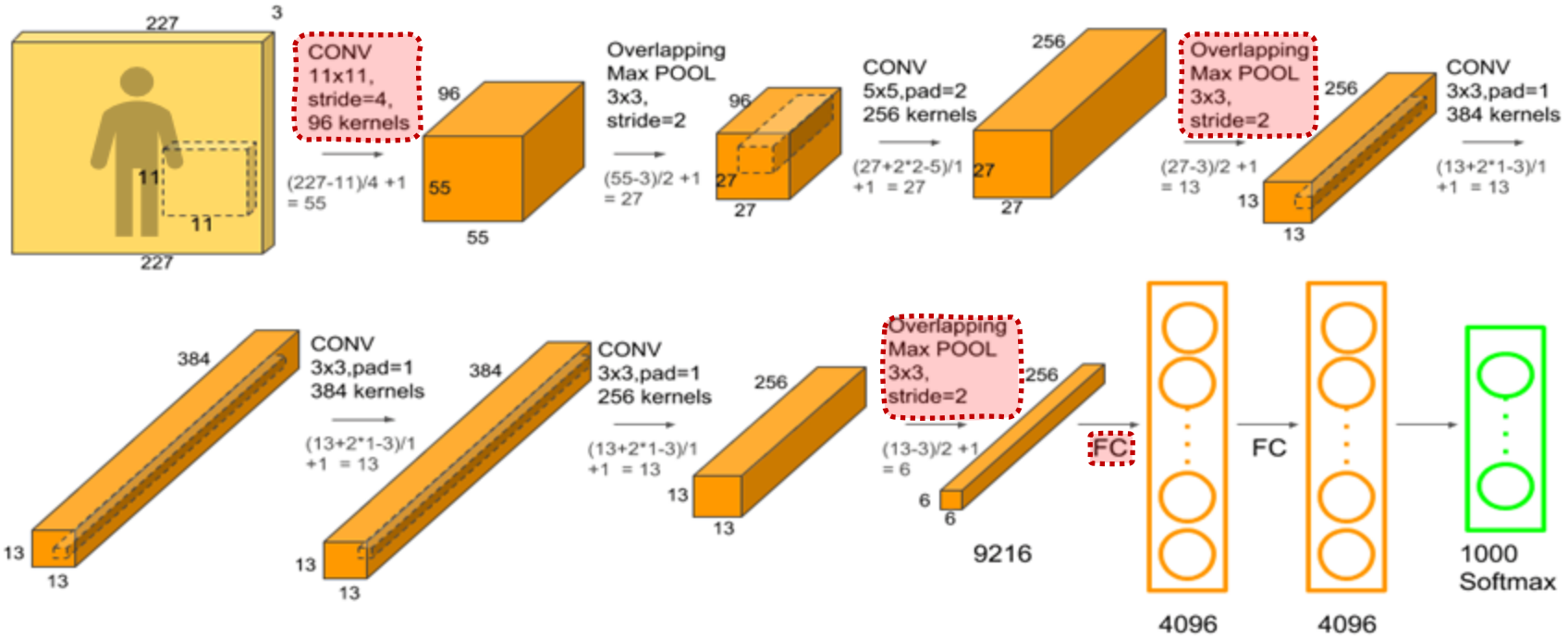
EACH COLUMN INDICATES WHICH FEATURE MAP IN S2 ARE COMBINED BY THE UNITS IN A PARTICULAR FEATURE MAP OF C3.

http://vision.stanford.edu/cs598_spring07/papers/Lecun98.pdf

AlexNet ARCHITECTURE

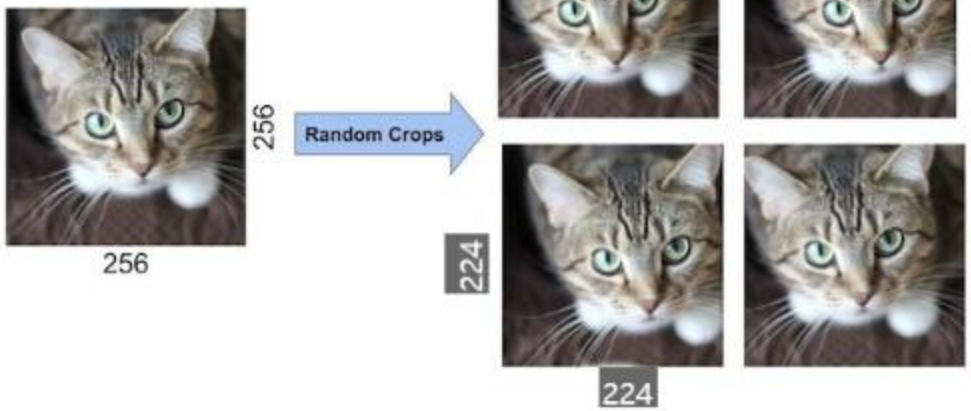
ReLU + Local response normalization
Multiple GPU
Overlapping pooling
Dropout
Data augmentation

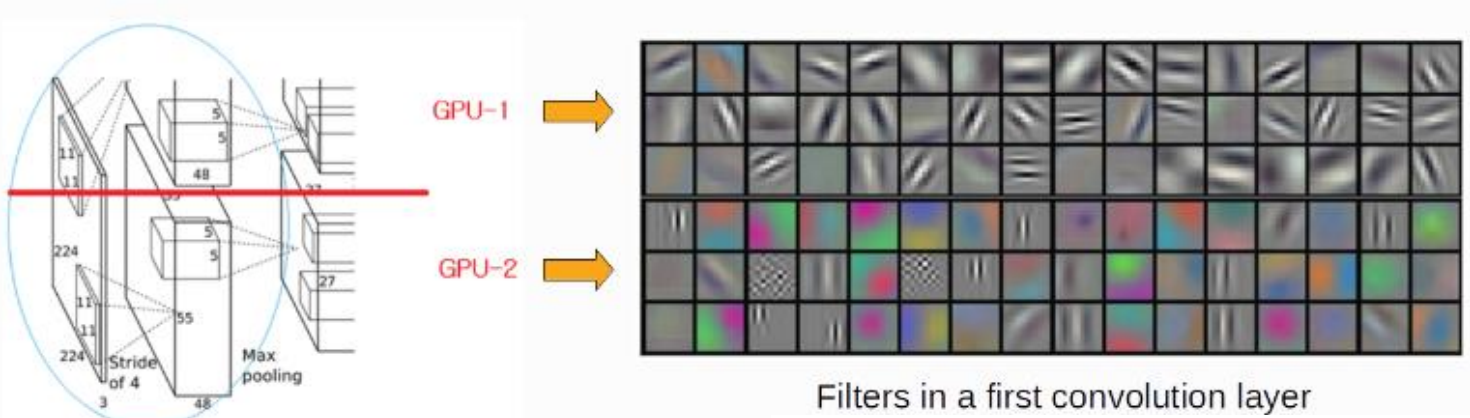


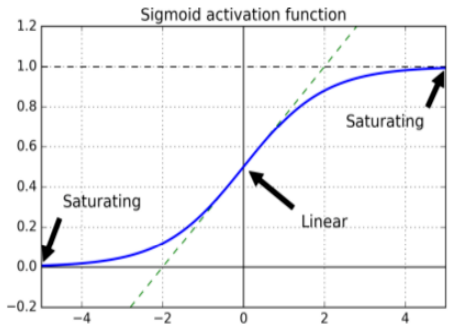


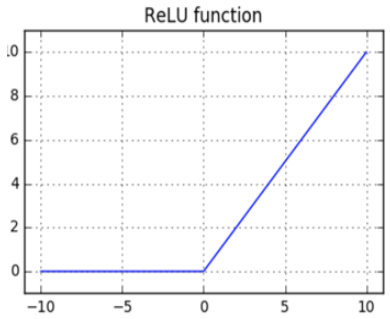
https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

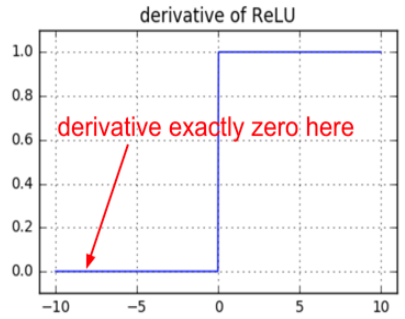
AlexNet ARCHITECTURE

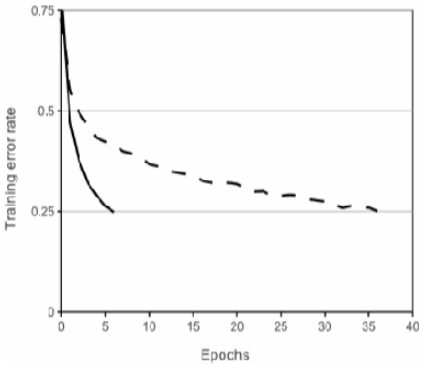


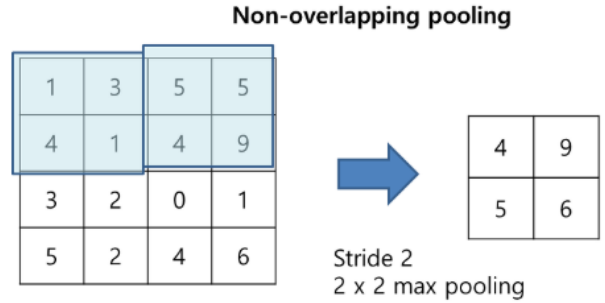


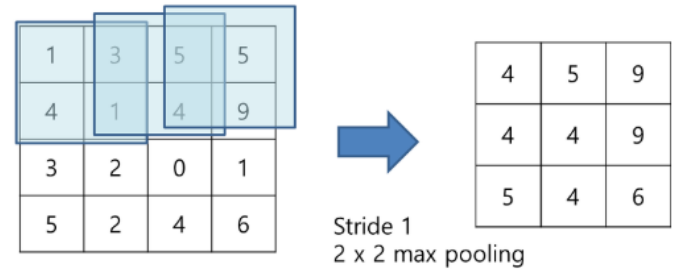












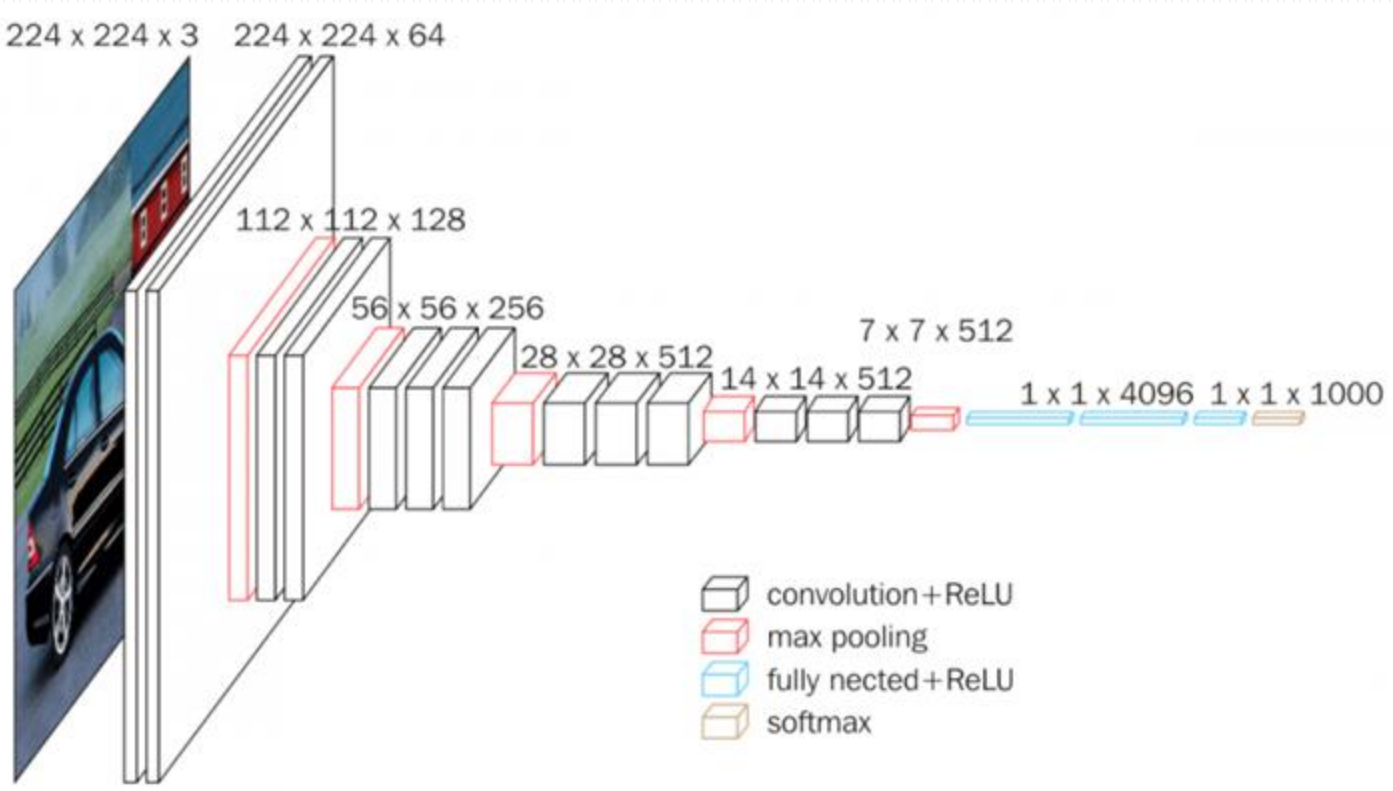
https://oi.readthedocs.io/en/latest/computer_vision/cnn/alexnet.html

VGG, 2014

Factorizing Conv.
Depth increased

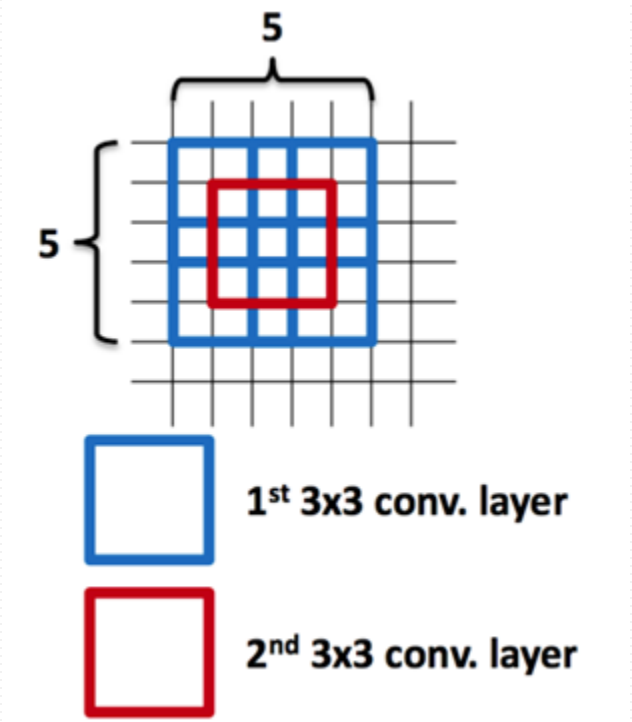
- The key is to see how deeper the depth of the network affects performance

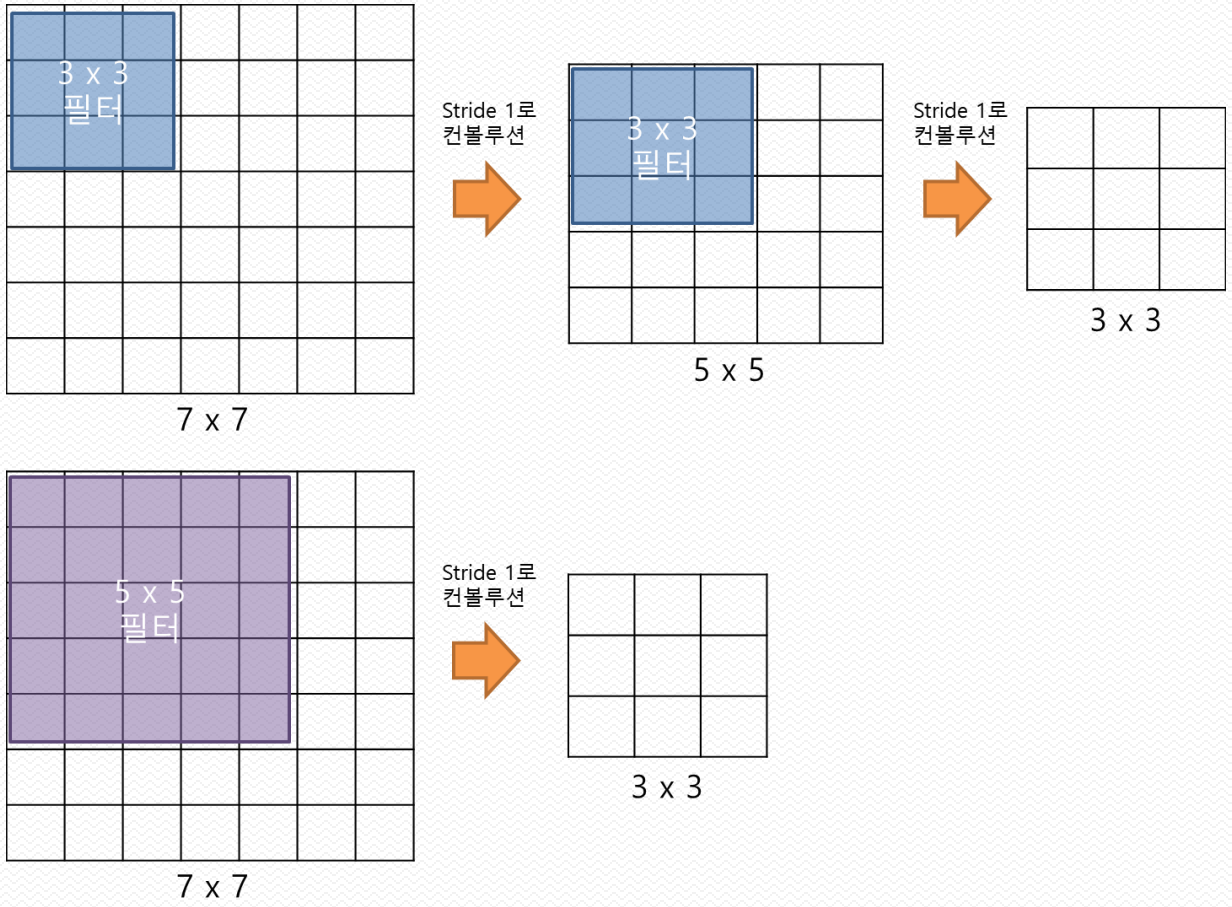
ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					



VGGNet Architecture

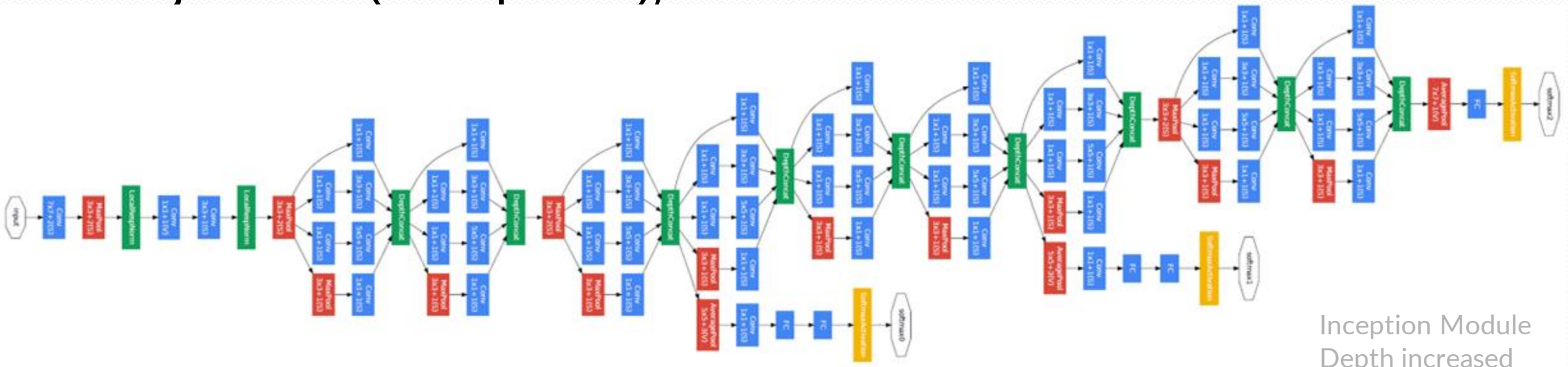
- 3x3 conv. Layer
  - Stacked conv. layers have a large receptive field
    - 3x3 (x2) – 5x5 receptive field
    - 3x3 (x3) – 7x7 receptive field
  - Less parameters to learn



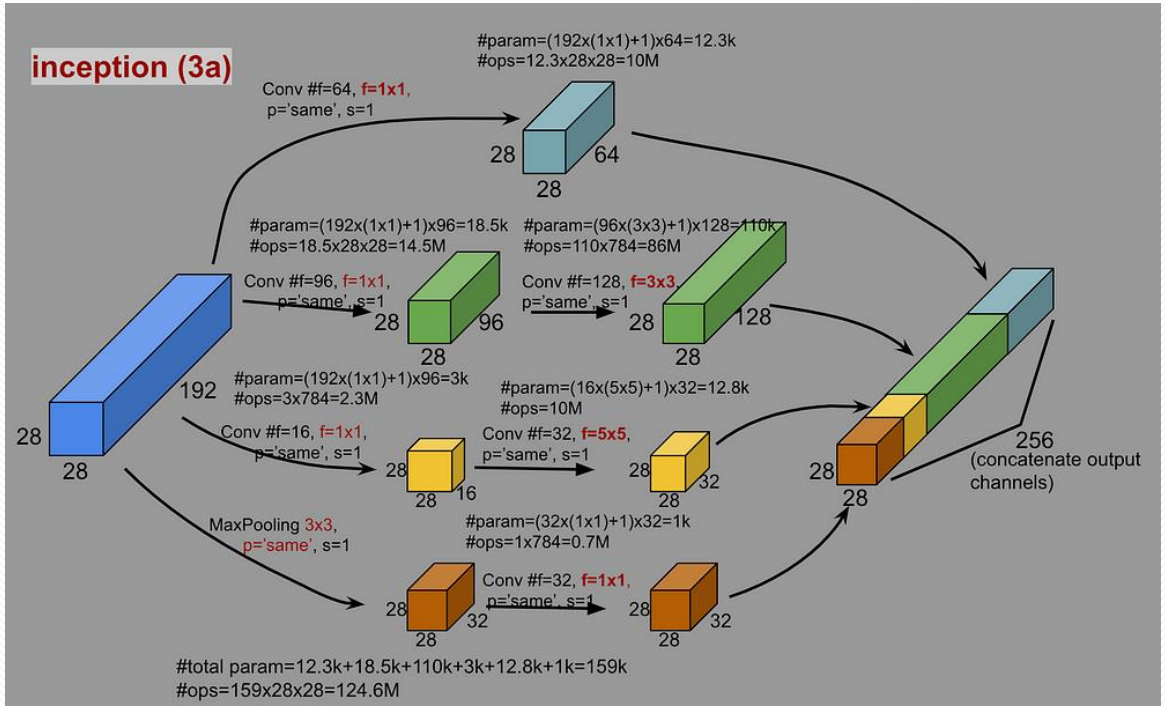


https://bskyvision.com/504

GoogLeNet (Inception), 2014



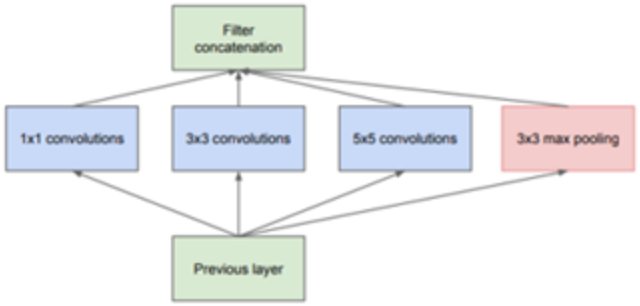
Inception Module
Depth increased



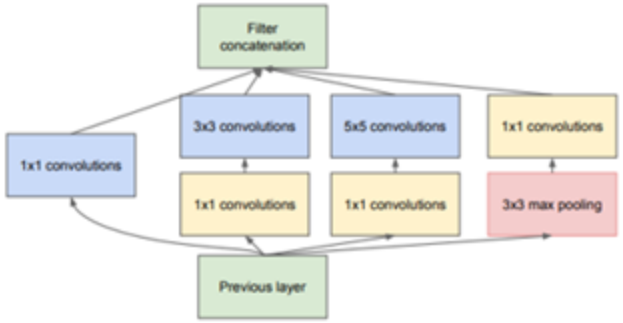
https://arxiv.org/pdf/1409.4842.pdf

GoogLeNet Architecture

1. Inception Module



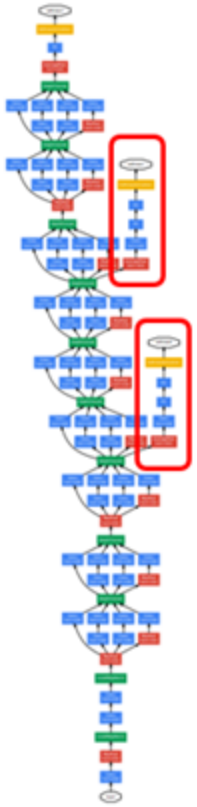
(a) Inception module, naïve version



(b) Inception module with dimensionality reduction

Figure 2: Inception module

2. Auxiliary Classifier



AvgPool

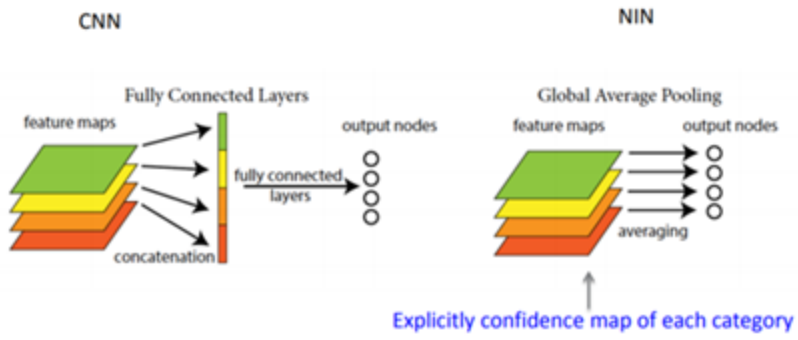
1x1Conv

FC

FC

Softmax

3. FC layer → Global Average Pooling

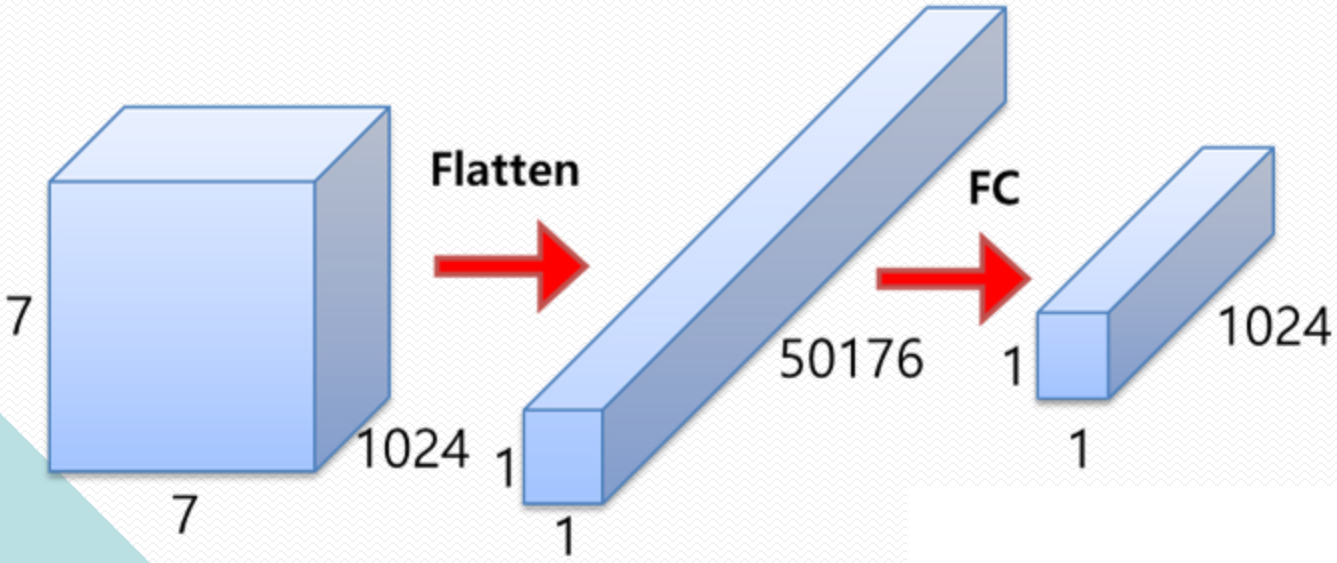


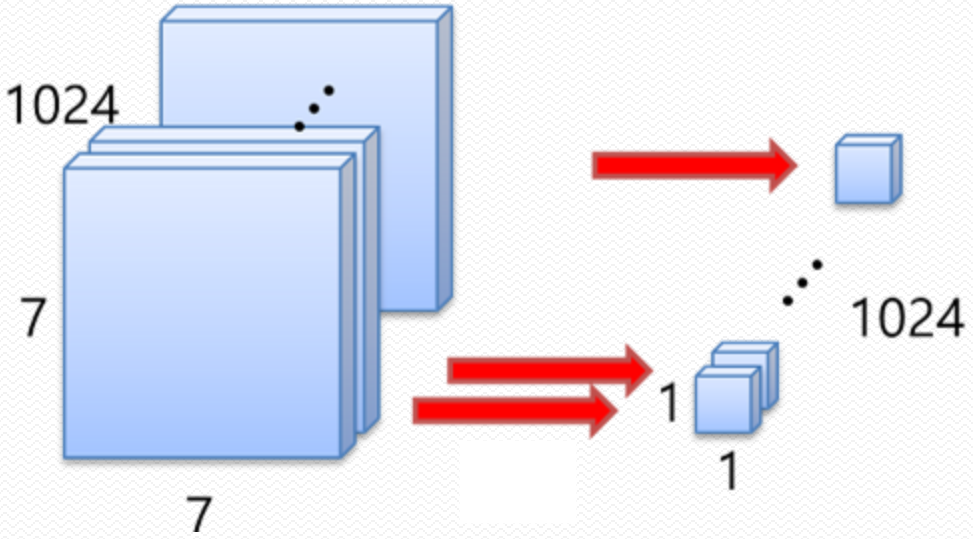
Save a large portion of parameters

inception (5b)		7×7×1024
avg pool	7×7/1	1×1×1024
dropout (40%)		1×1×1024

Global Average Pooling

- FC (Fully Connected)
  - Weights: 7 x 7 x 1024 x 1024 = 51.3M
- GAP (Global Average Pooling)
  - Weights: 0

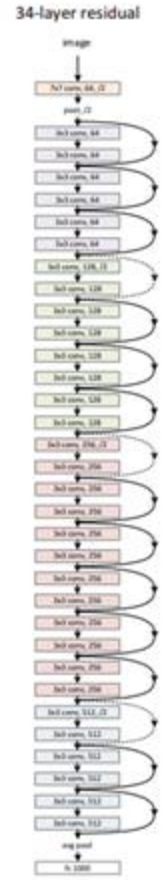




ResNet, 2015

- Revolution of depth

Residual Network
Great Depth
Better recognition than human



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

ResNet, 152 layers
(ILSVRC 2015)

VGG, 19 layers
(ILSVRC 2014)

AlexNet, 8 layers
(ILSVRC 2012)

Network Design

- Simple design
  - All 3x3 conv (almost)
  - No Hidden FC
- Spatial size /2 → # filters x2

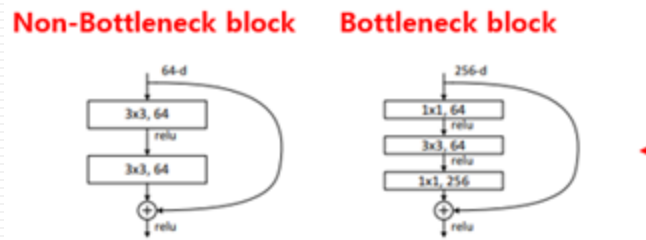


Figure 5. A deeper residual function $\mathcal{F}$ for ImageNet. Left: a building block (on 56×56 feature maps) as in Fig. 3 for ResNet-34. Right: a "bottleneck" building block for ResNet-50/101/152.

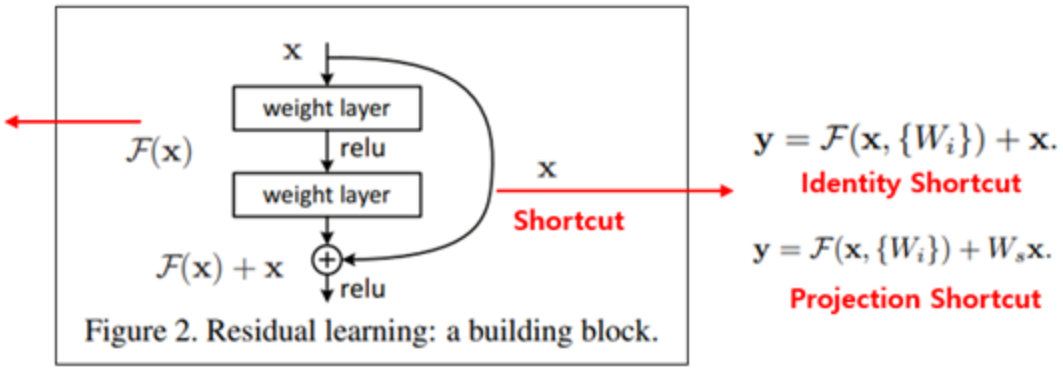


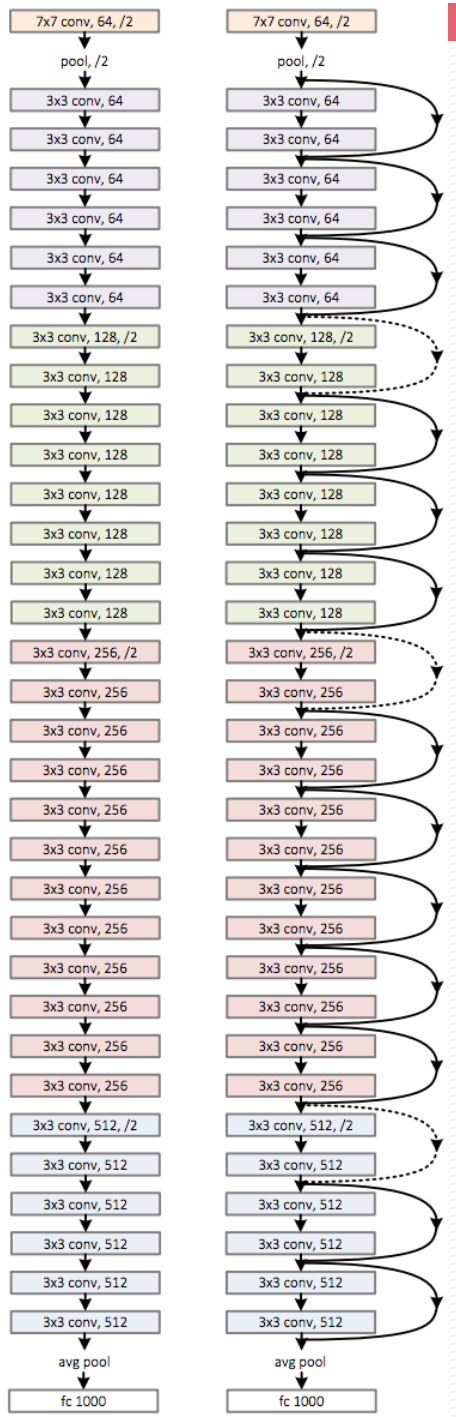
Figure 2. Residual learning: a building block.

$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + \mathbf{x}.$

Identity Shortcut

$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + W_s\mathbf{x}.$

Projection Shortcut

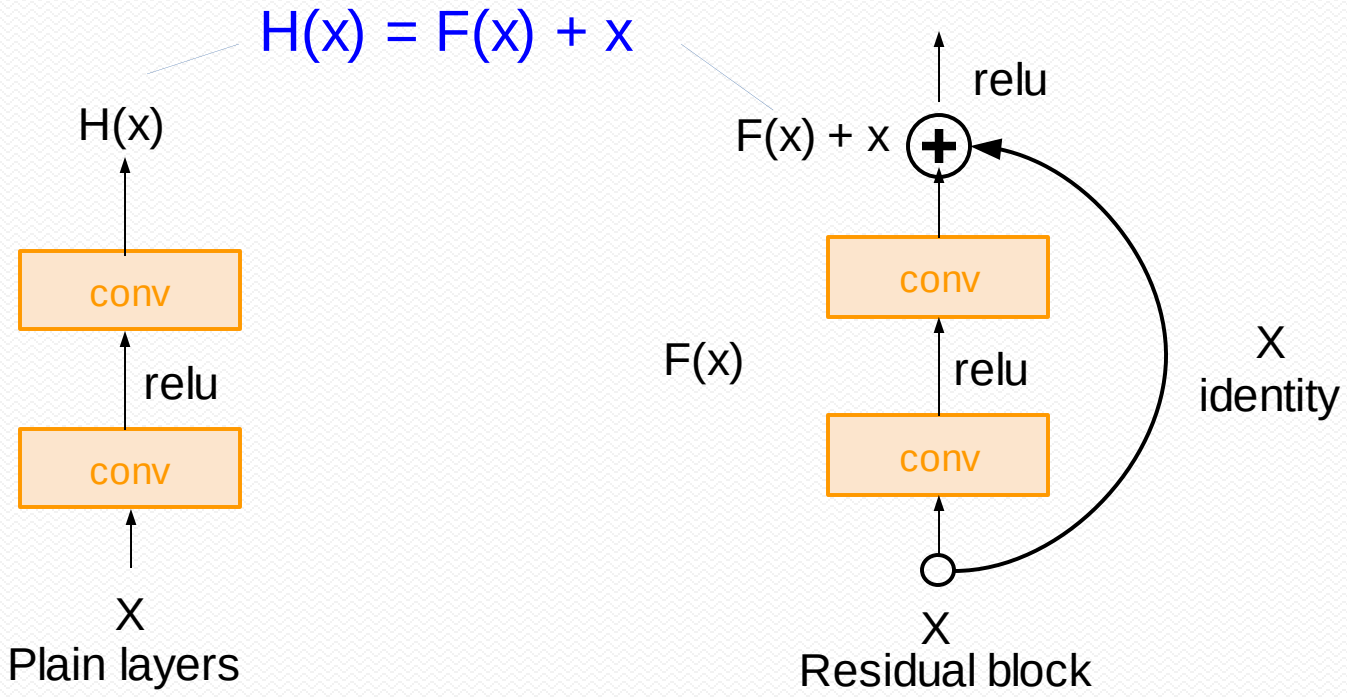


Identity shortcut

Projection shortc

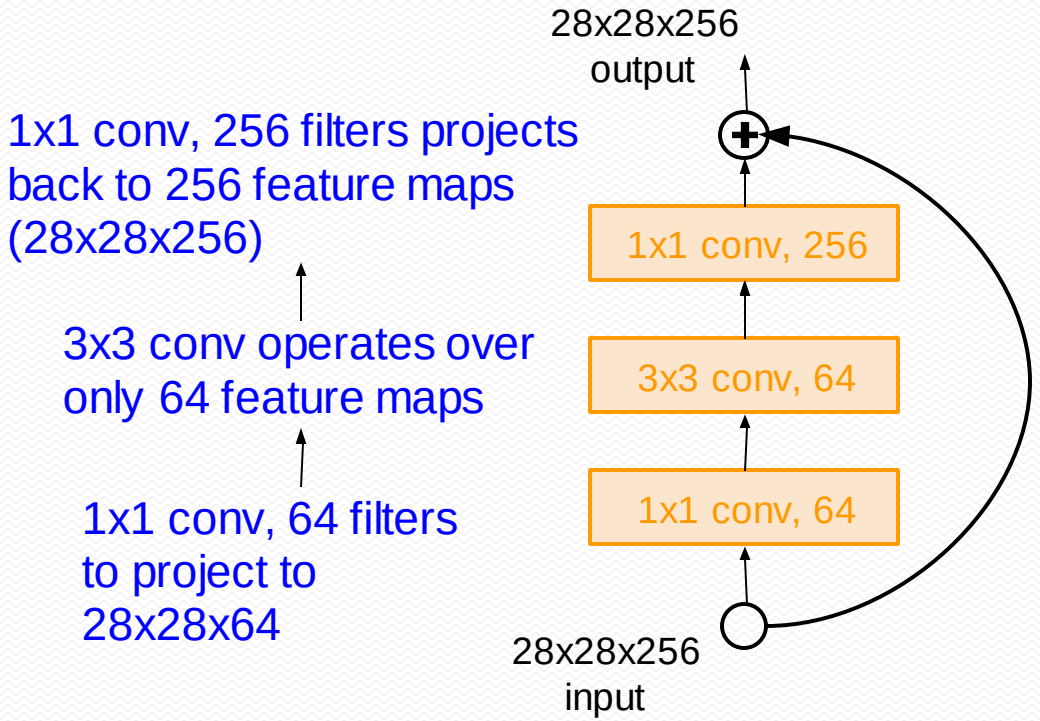
Residual Block

- Use layers to fit residual *"F(x) = H(x) - x"* instead of *"H(x)"* directly



Residual Block (cont.)

- For deeper networks…
  - use “bottleneck” layer
  - improve efficiency
  - (similar to GoogLeNet)
  - e.g. ResNet-50+

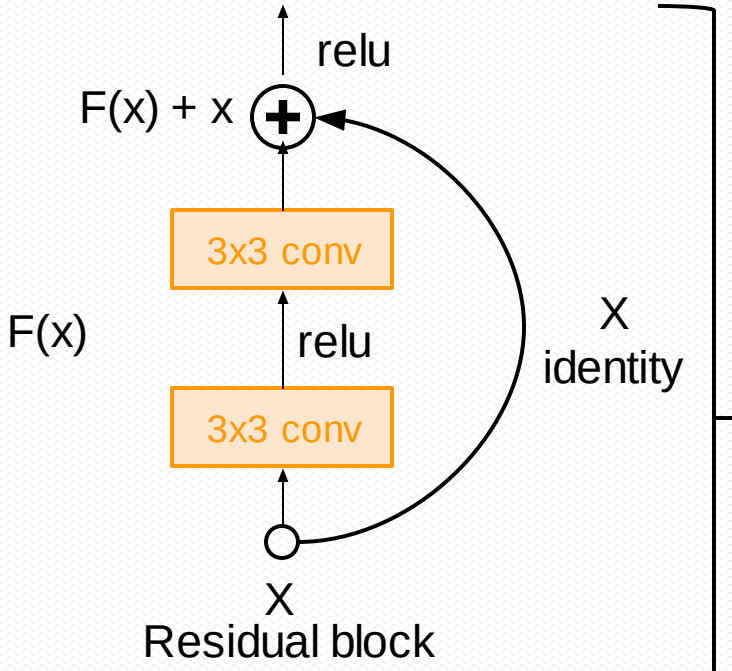


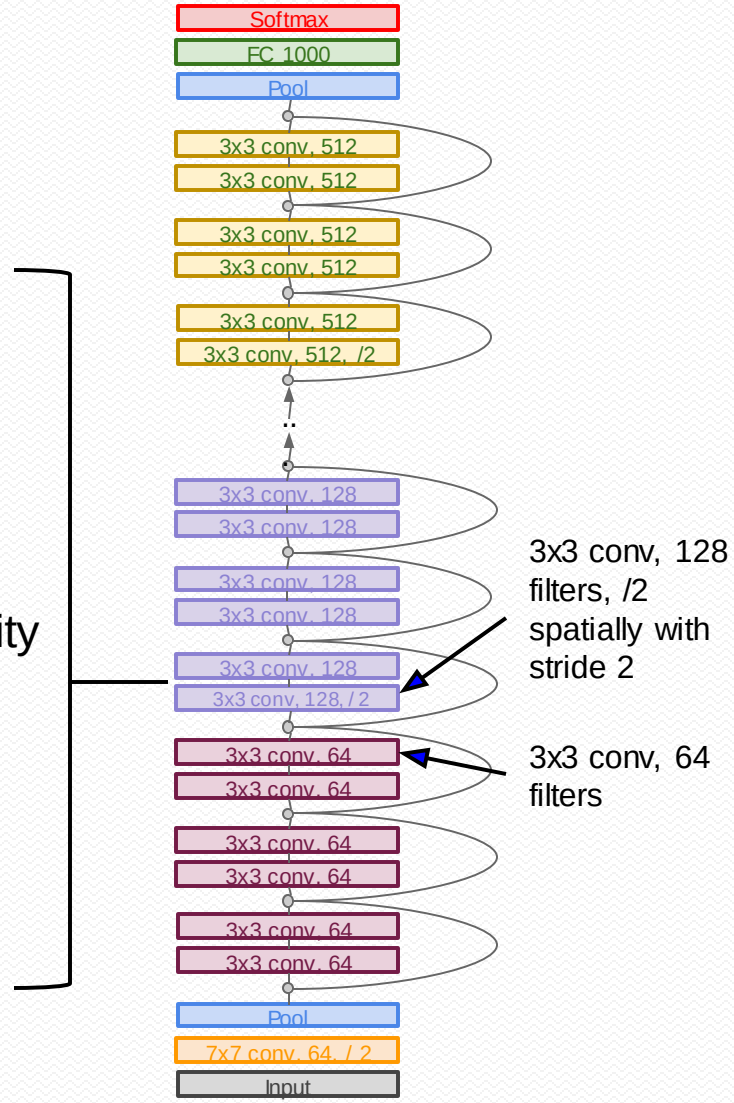
Residual Block x Keras

```
def conv4_layer(x):
    shortcut = x

    for i in range(6):
        if (i == 0):
            x = Conv2D(256, (1, 1), strides=(2, 2),
                       padding='valid')(x)
            x = BatchNormalization()(x)
            x = Activation('relu')(x)

            x = Conv2D(256, (3, 3), strides=(1, 1),
                       padding='same')(x)
            x = BatchNormalization()(x)
            x = Activation('relu')(x)

            x = Conv2D(1024, (1, 1), strides=(1,
1),
                       padding='valid')(x)
            x = BatchNormalization()(x)

            shortcut = Conv2D(1024, (1, 1),
                              strides=(2, 2),

padding='valid')(shortcut)
            shortcut =
BatchNormalization()(shortcut)

            x = Add()([x, shortcut])
            x = Activation('relu')(x)

            shortcut = x
        else:
            x = Conv2D(256, (1, 1), strides=(1,
1),
                       padding='valid')(x)
            x = BatchNormalization()(x)
            x = Activation('relu')(x)

            x = Conv2D(256, (3, 3), strides=(1,
1),
                       padding='same')(x)
            x = BatchNormalization()(x)
            x = Activation('relu')(x)

            x = Conv2D(1024, (1, 1), strides=(1,
1),
                       padding='valid')(x)
            x = BatchNormalization()(x)

            x = Add()([x, shortcut])
            x = Activation('relu')(x)

            shortcut = x

    return x
```

Full ResNet architecture

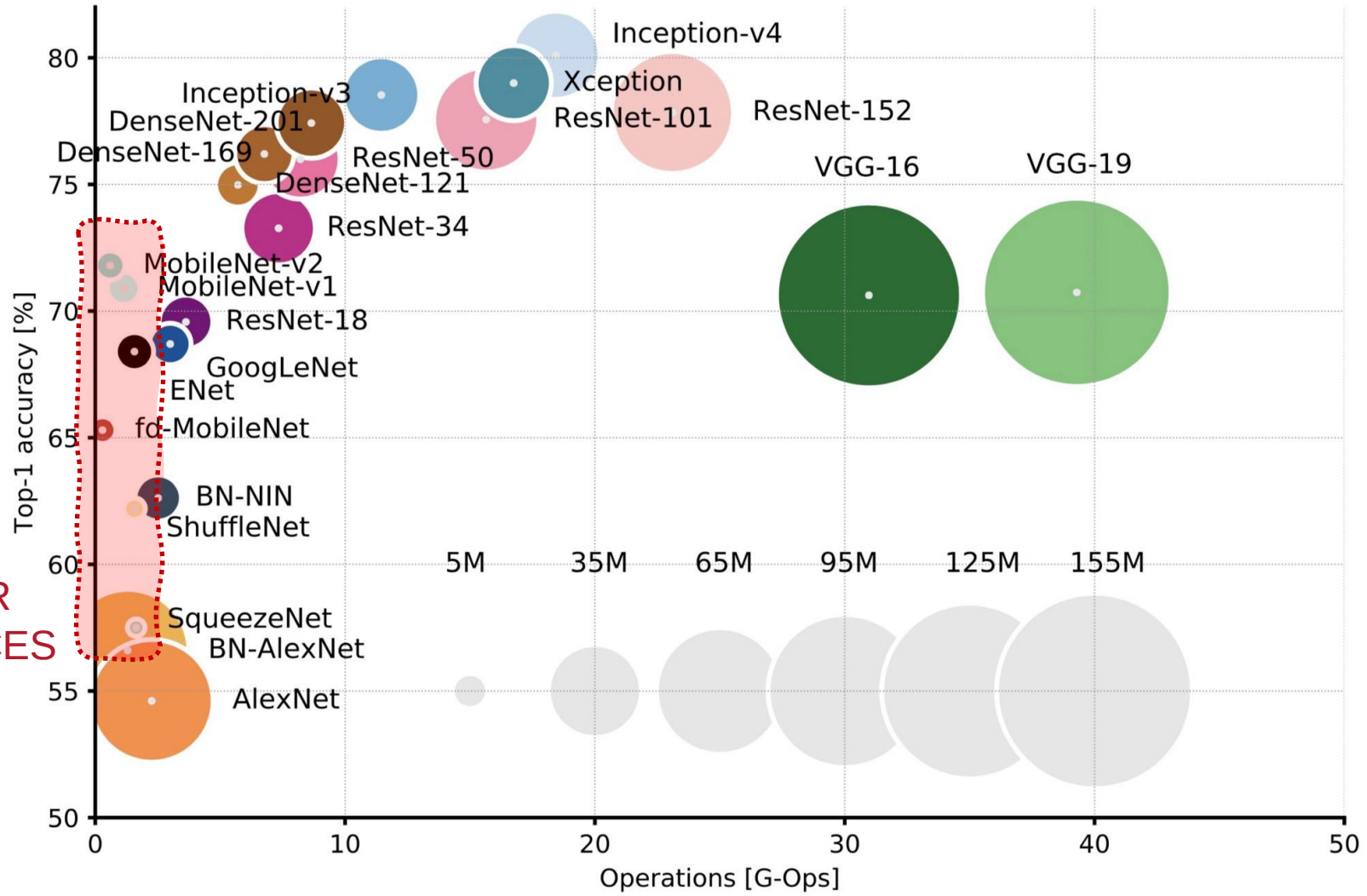
- Stack residual blocks
  - Consist of two 3x3 conv layers
- Periodically,
  - Double # of filters
  - Downsample spatially
  - By using stride of 2
  - (2 in each dimension)

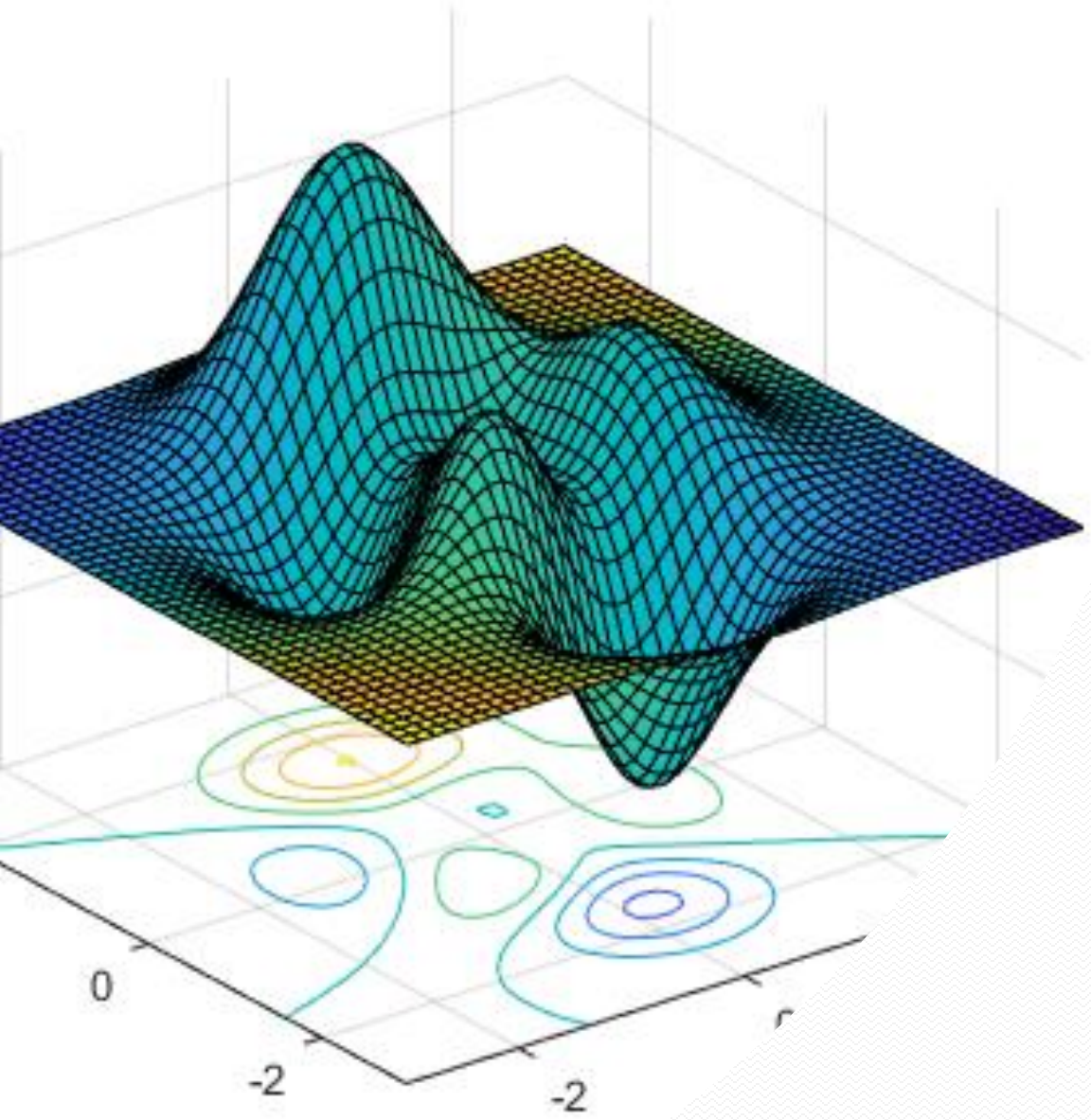




TOP 1 VS. OPERATIONS, PARAMETERS

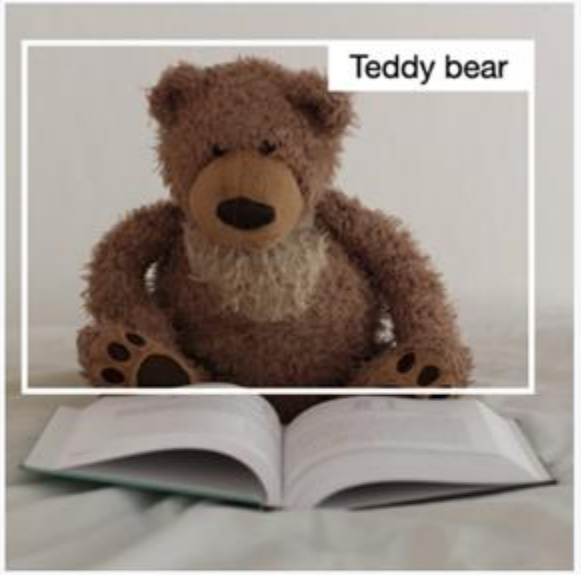
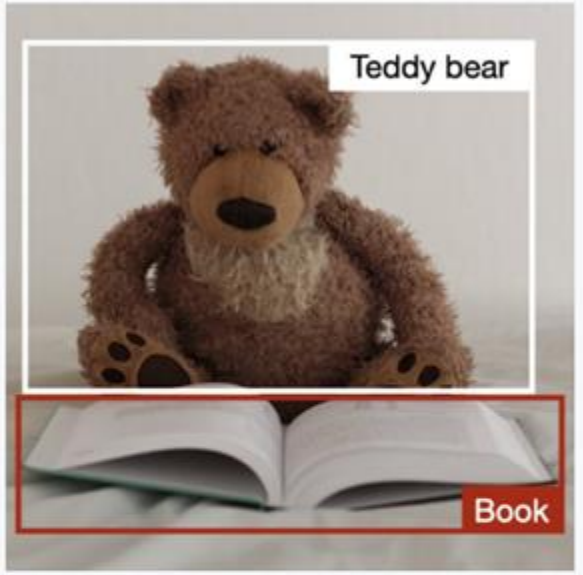
SUITED FOR MOBILE DEVICES



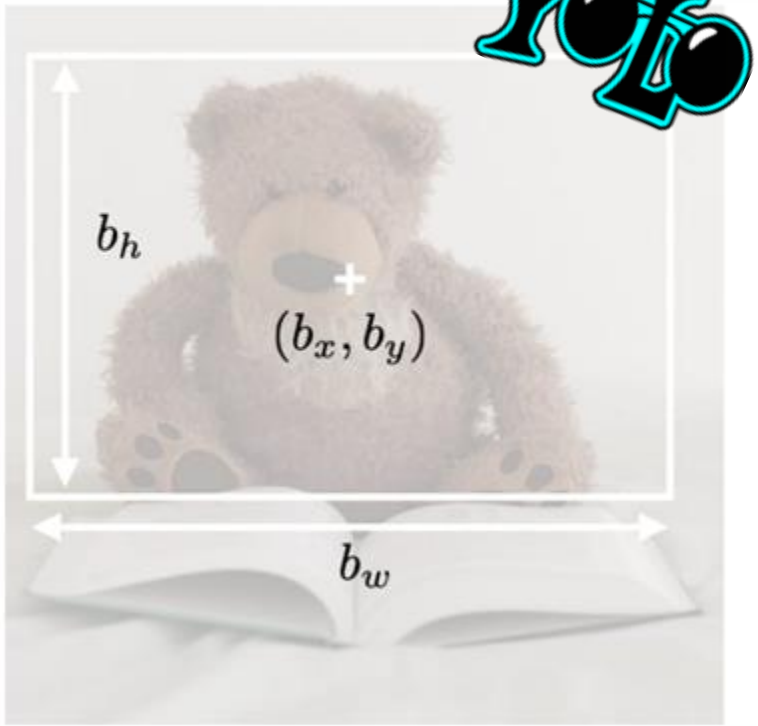
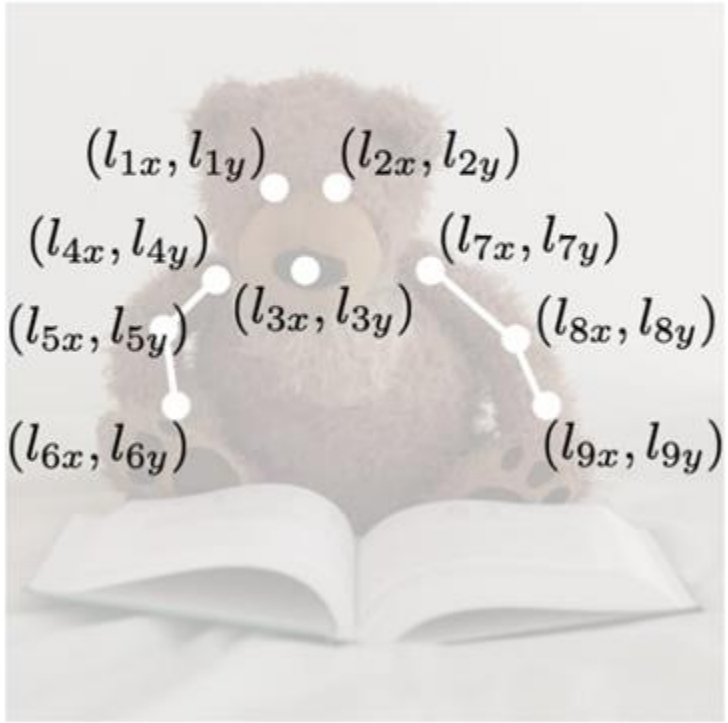


Object Detection

Object Detection

Image classification	Classification w. localization	Detection
		
• Classifies a picture • Predicts probability of object	• Detects an object in a picture • Predicts probability of object and where it is located	• Detects up to several objects in a picture • Predicts probabilities of objects and where they are located
Traditional CNN	Simplified YOLO, R-CNN	YOLO, R-CNN

Object Detection (cont.)

Bounding box detection	Landmark detection
• Detects the part of the image where the object is located	• Detects a shape or characteristics of an object (e.g. eyes) • More granular
	
Box of center (b_x, b_y), height b_h and width b_w	Reference points (l_{1x}, l_{1y}), ..., (l_{nx}, l_{ny})

OBJECT DETECTION

2-STAGE

REGION PROPOSAL

SELECTIVE SEARCH
REGIONAL PROPOSAL NETWORK

PLAIN CNNS

CLASSIFICATION

FOR EACH PROPOSED REGION

MULTI-CLASS CLASSIFICATION

BOUNDING BOX REGRESSION

1-STAGE

CNN

CONV LAYERS

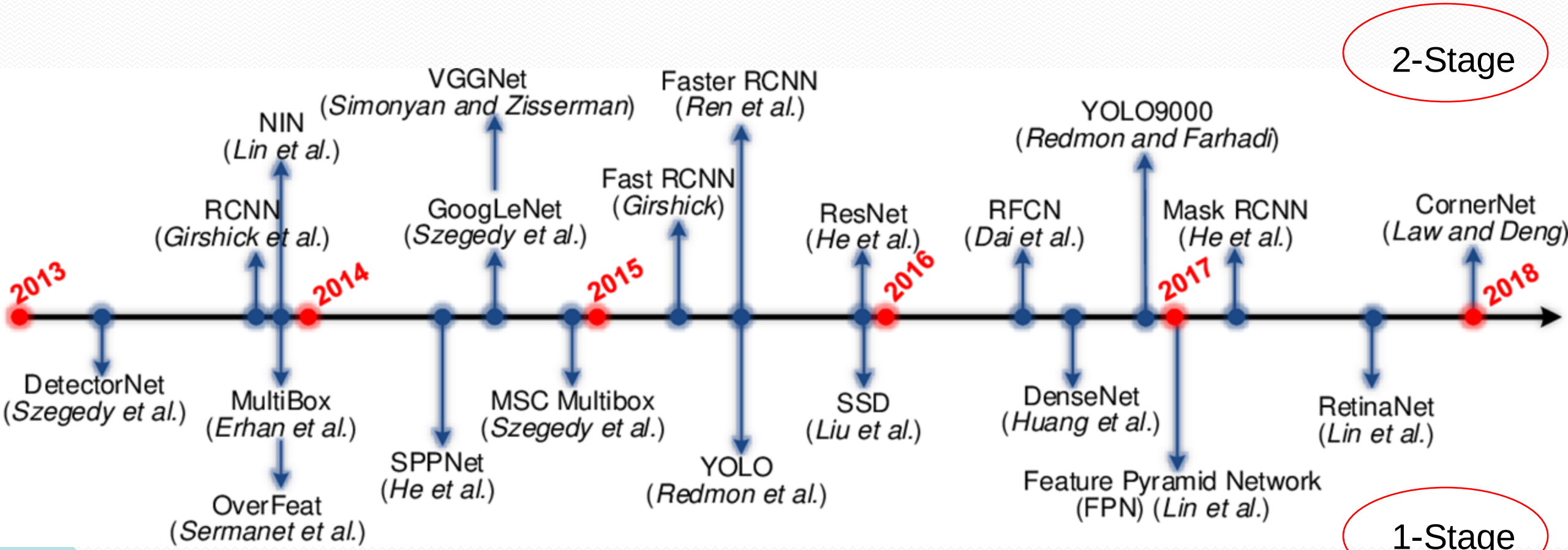
FEATURE EXTRACTION

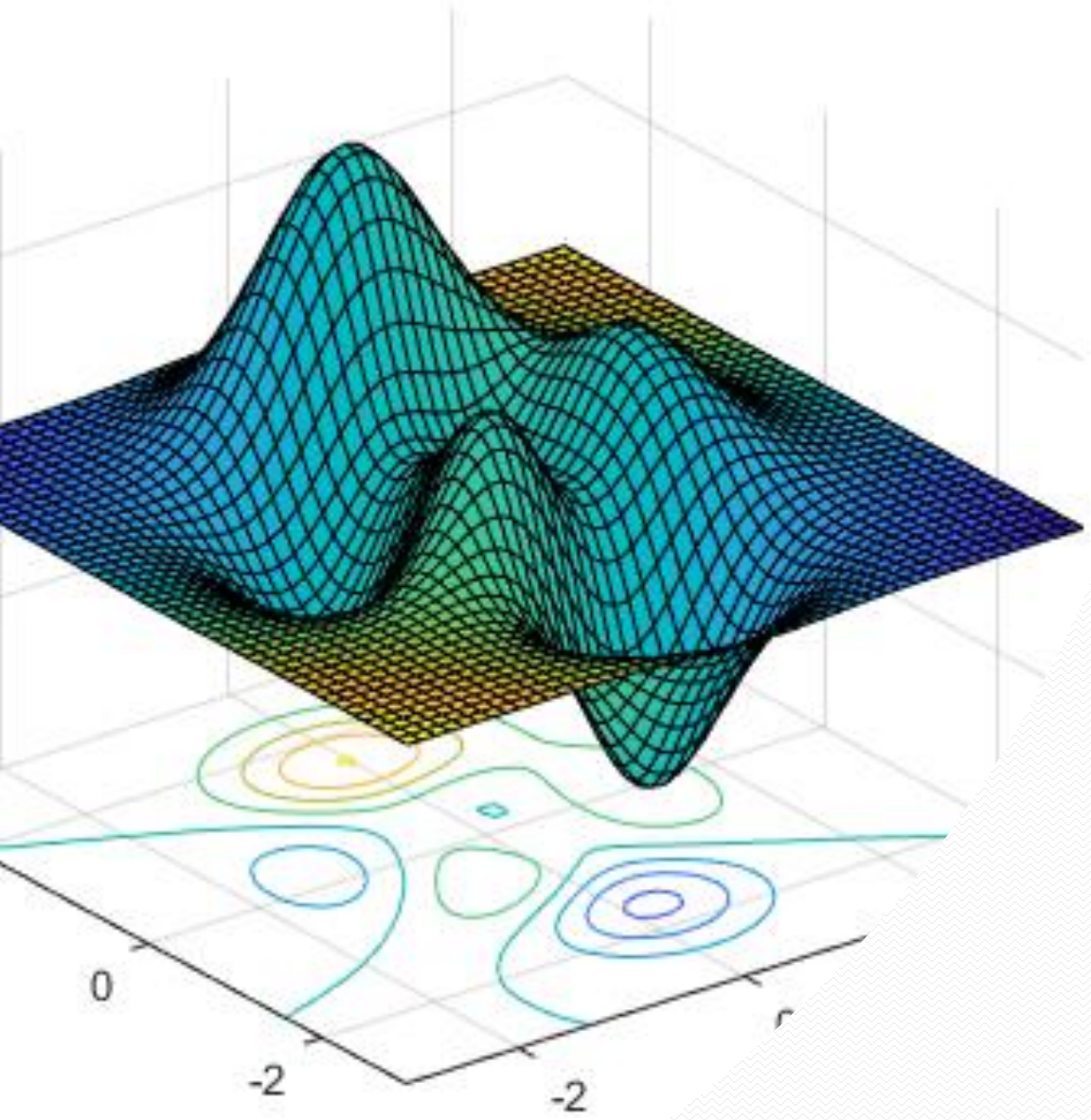
...

FEATURE MAPS

FOR EACH GRID / SPATIAL LOCATION

1-Stage & 2-stage Detector (cont.)





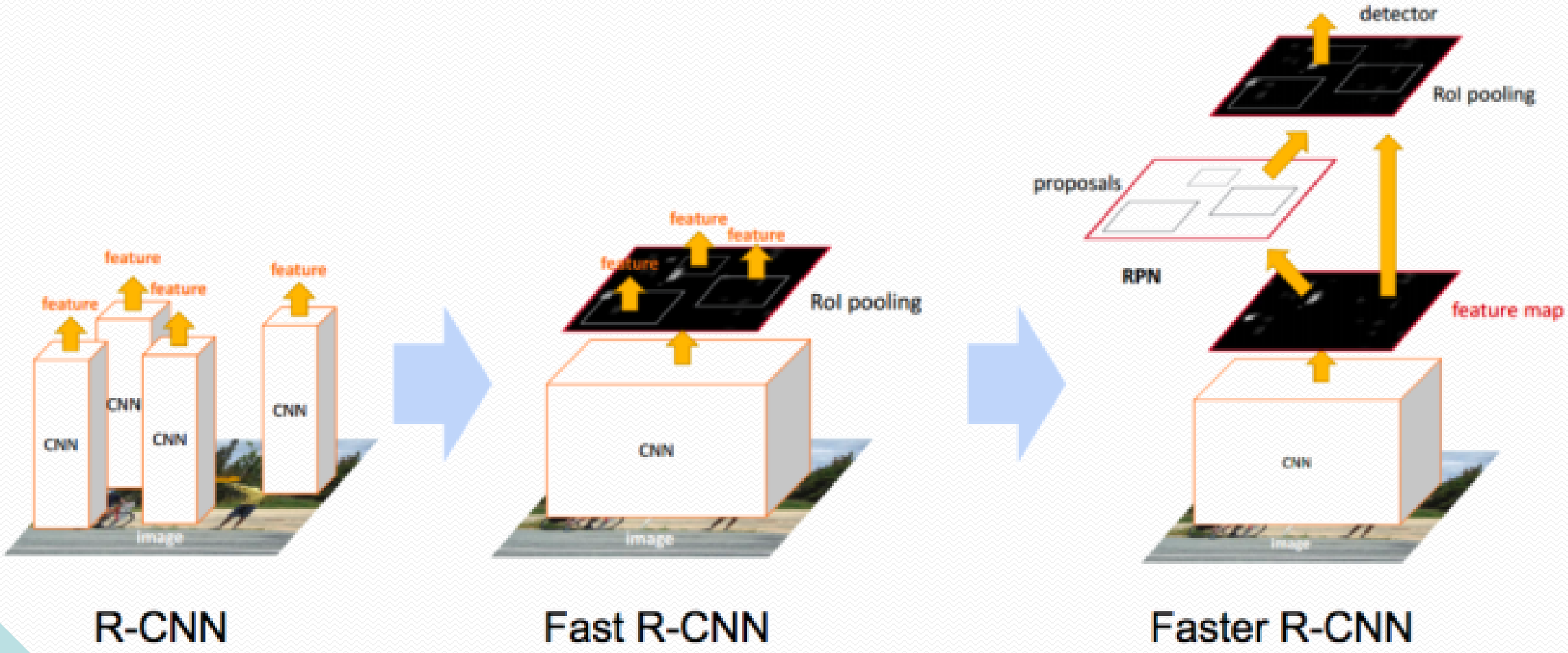
2 stage

NMS(Non-Maximum Suppression)

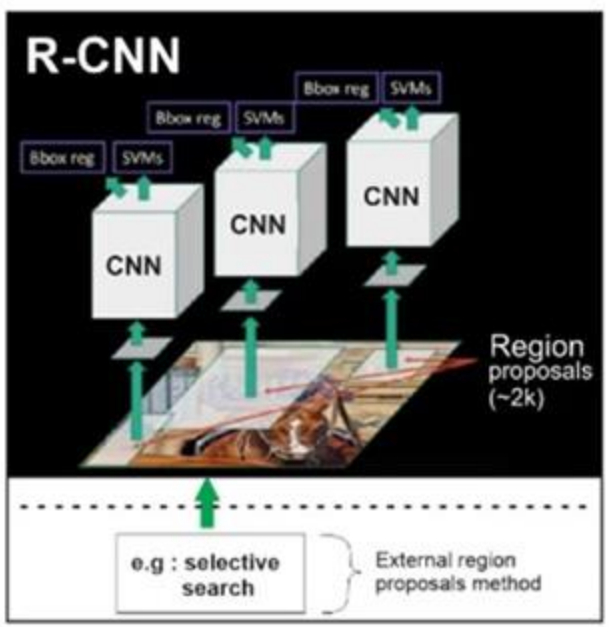
- If there are multiple boxes on the same object, only the box with the highest score is left

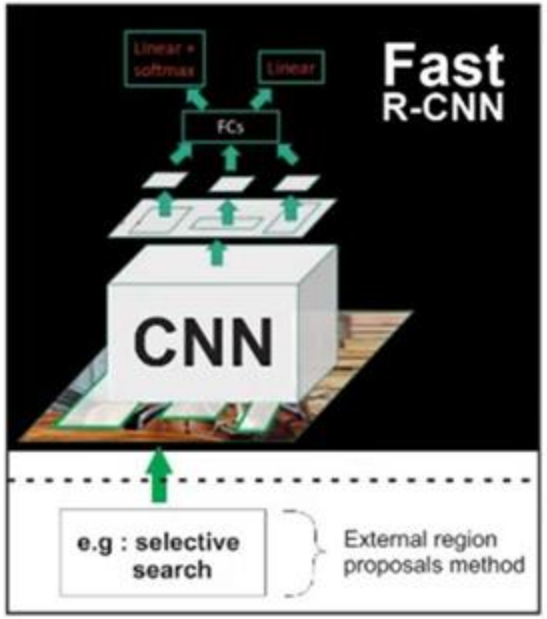
$$IoU = \frac{\text{Intersection}}{\text{Union}}$$

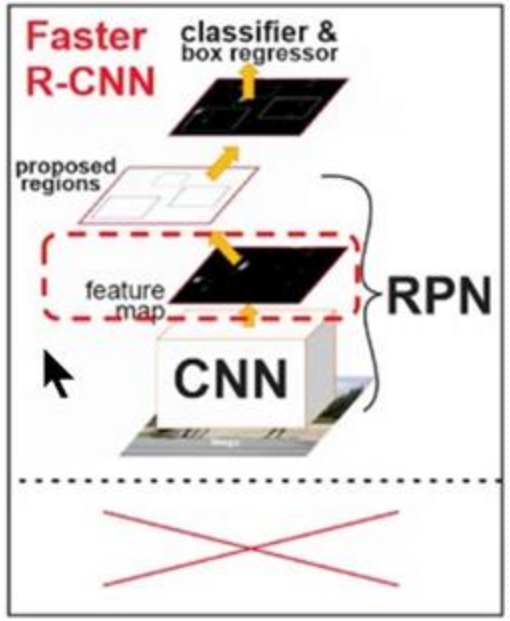
Evolution from R-CNN to Faster R-CNN



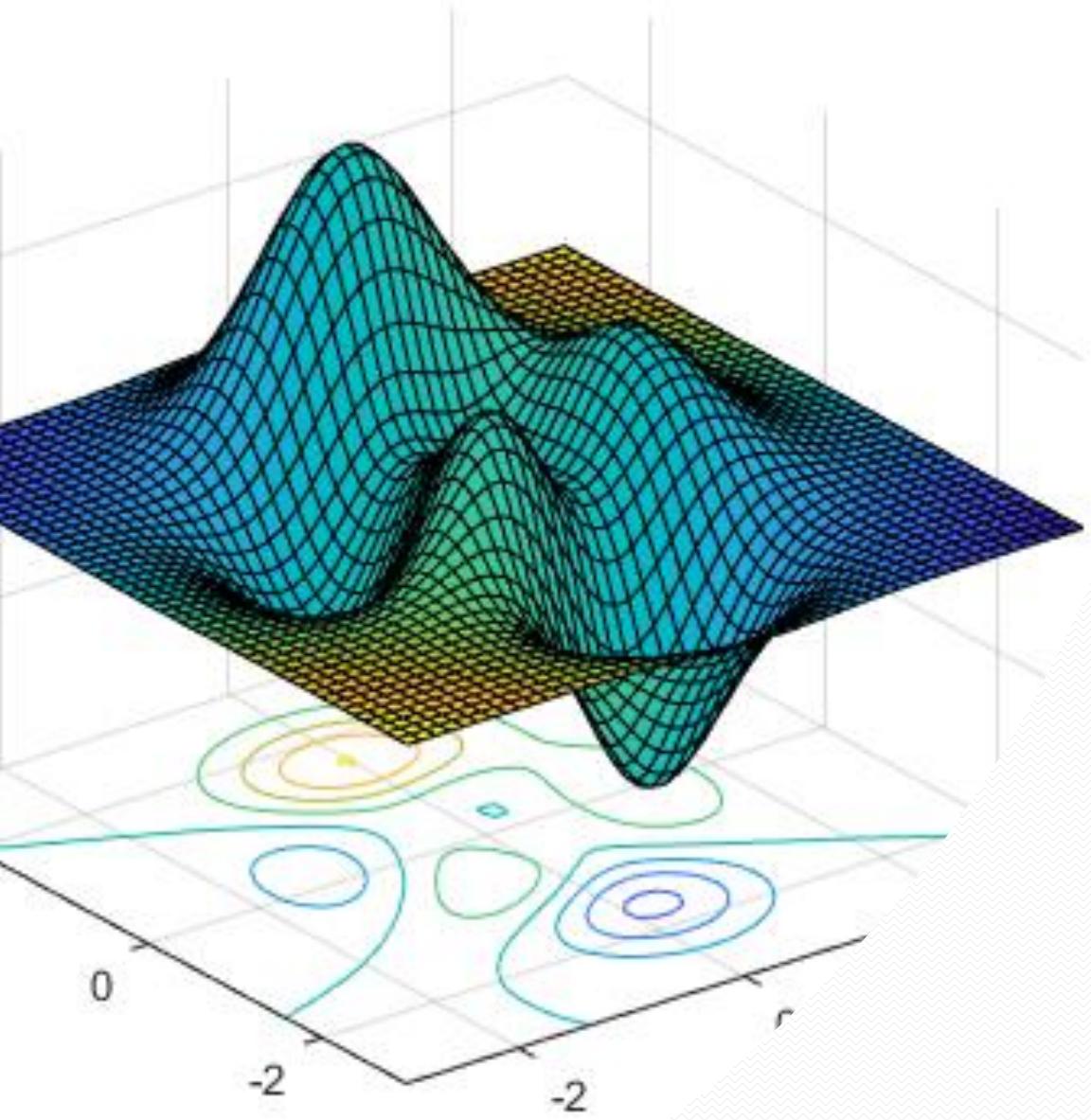
Compare performance





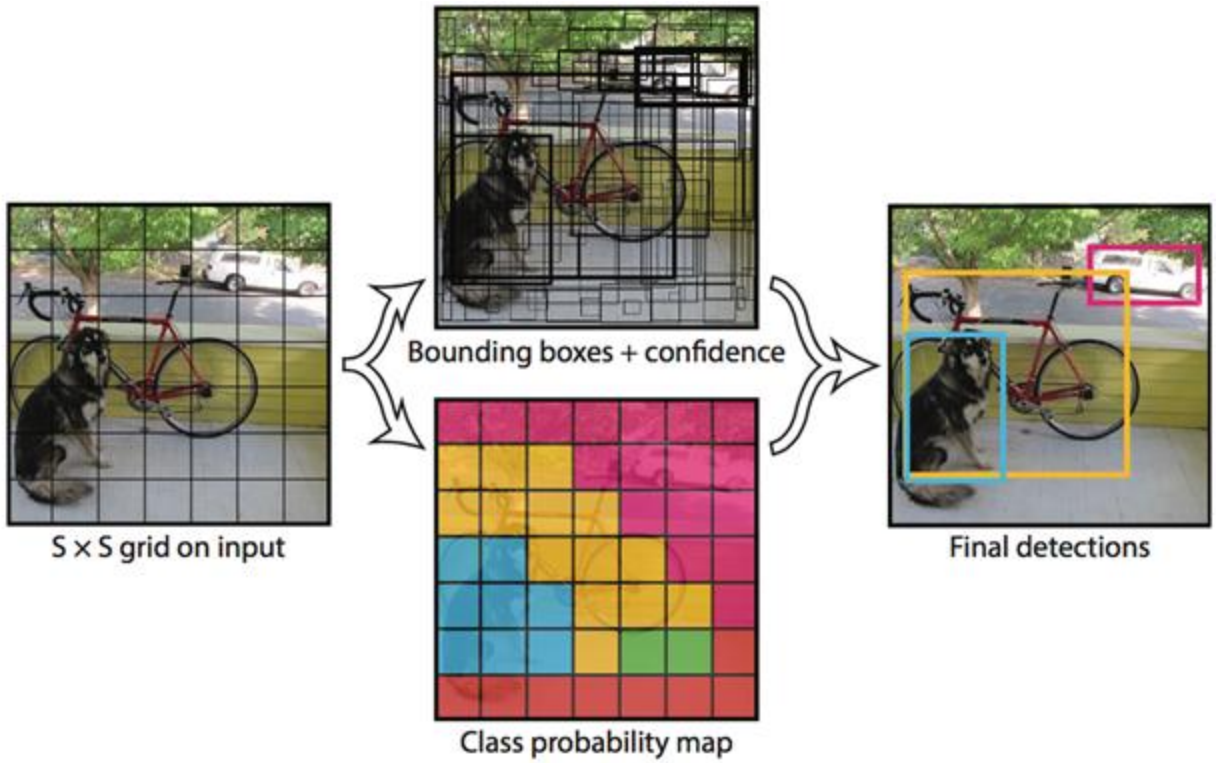


	R-CNN	Fast R-CNN	Faster R-CNN
Test time per image	50 seconds	2 seconds	0.2 seconds
Speed-up	1x	25x	250x
mAP (VOC 2007)	66.0%	66.9%	66.9%

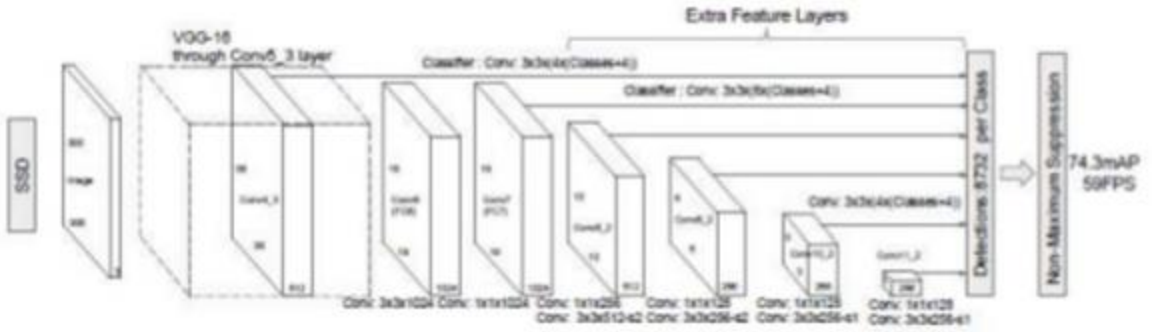


1 stage

1 STAGE OBJECT DETECTION



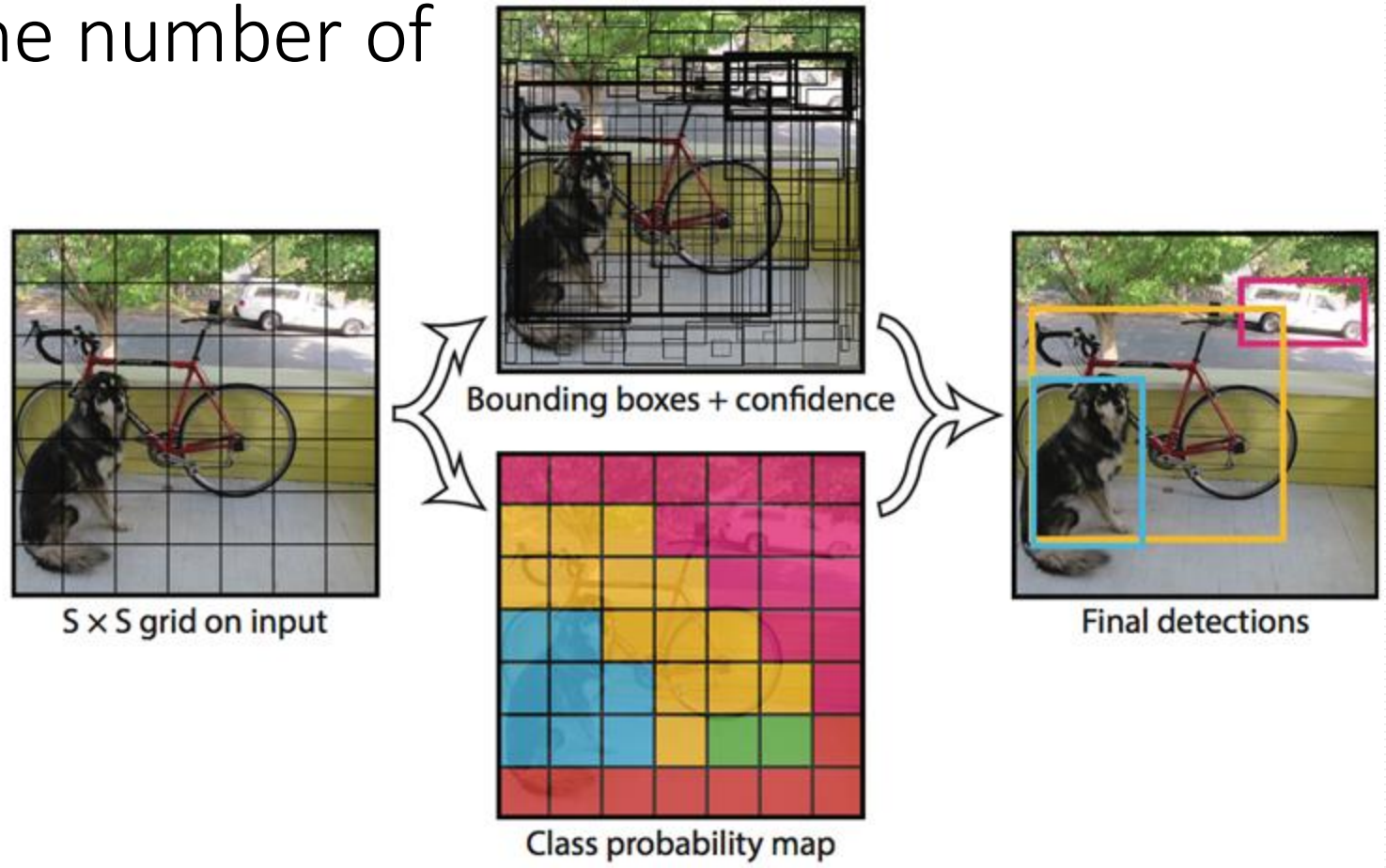
YOLO



SSD

Yolo

- Predict one set of class probabilities per grid cell, regardless of the number of boxes B
- Image
  - **S** x **S** grids
- Grid cell
  - **B**: Bboxes & Confidence score
  - **C**: class probabilities
  - with regard to #classes



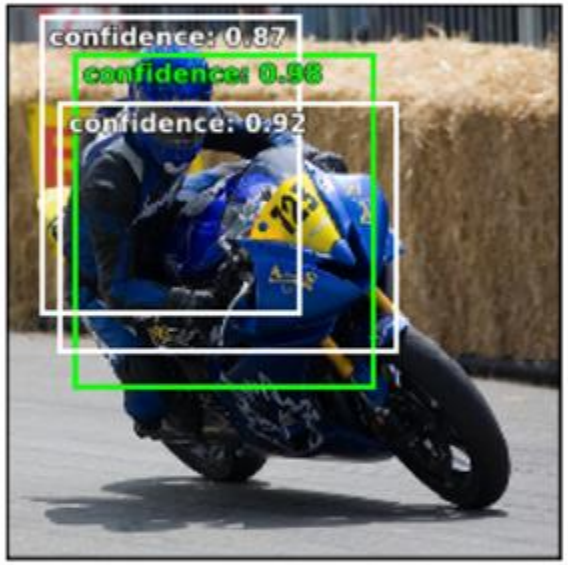
x, y, w, h, confidence

Predictions on a grid (cont.)

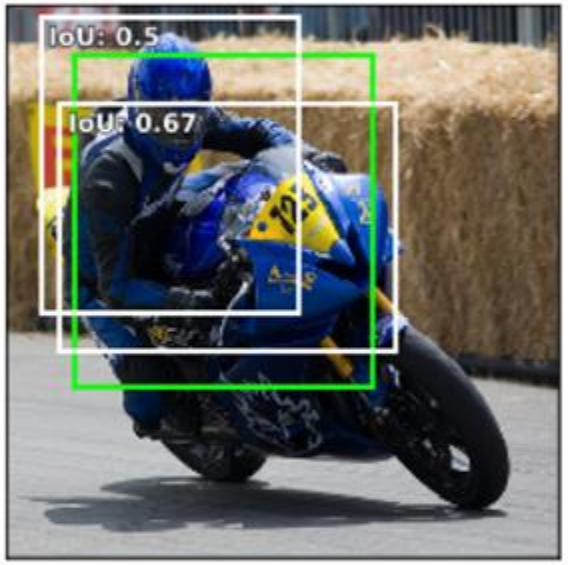
Repeat with next highest confidence prediction until no more boxes are being suppressed

For each class...

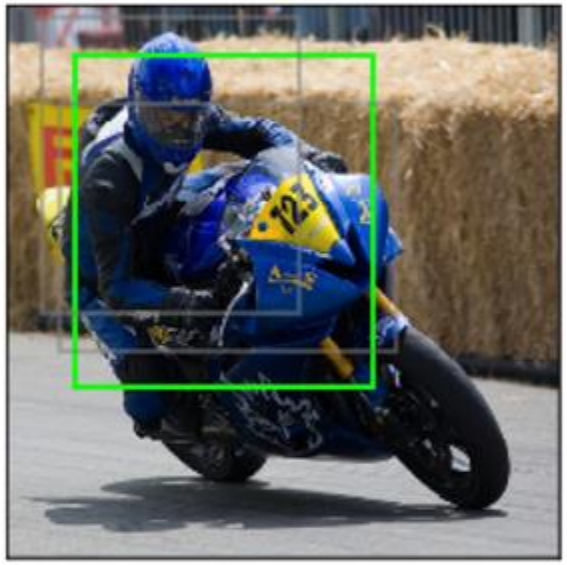
After filtering out low confidence predictions, we may still be left with ***redundant*** **detections**



Select the bounding box prediction with the **highest confidence**



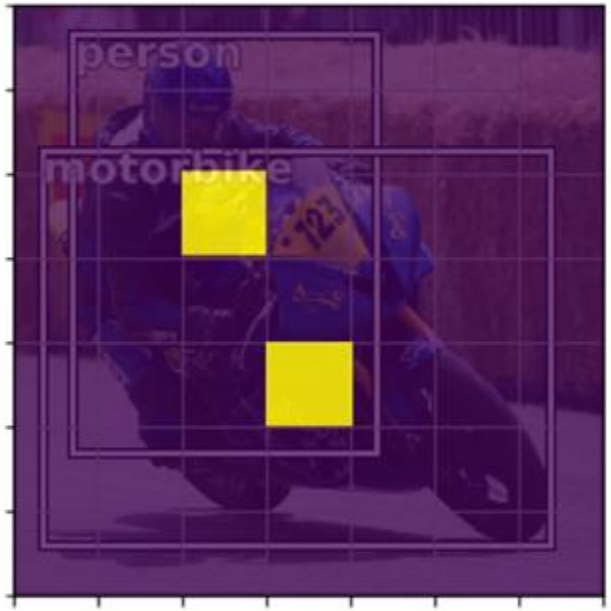
Calculate the IoU between the selected box and all remaining predictions

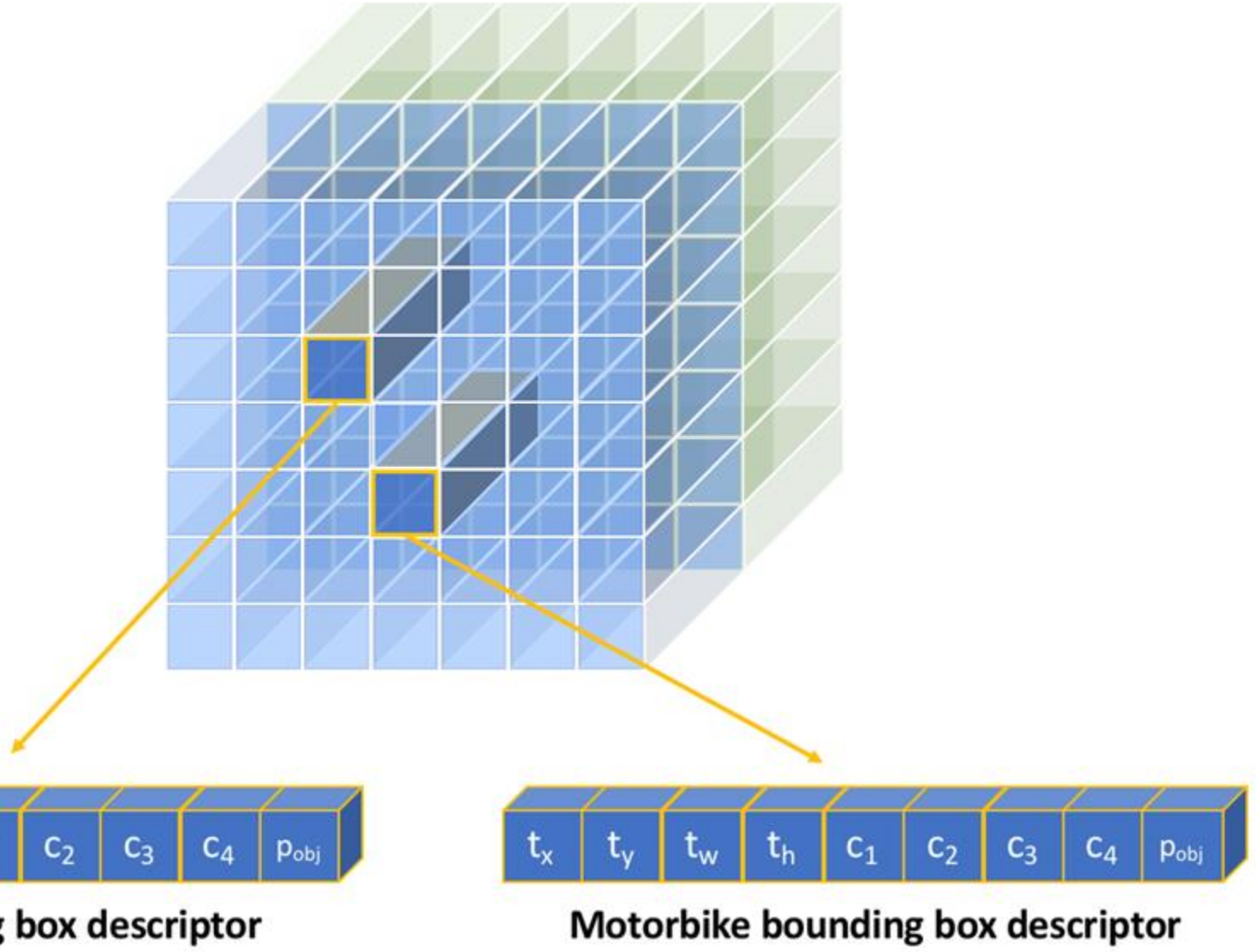


Remove any boxes which have an IoU score above some defined threshold

Predictions on a grid (cont.)

multiple objects can be detected in parallel

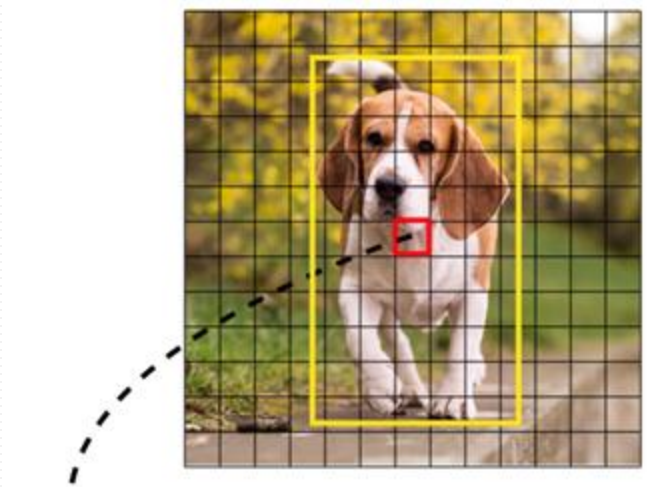




Yolo V3(Prediction Across Scales)

Big Objects

Small Objects



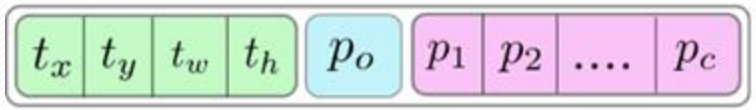
13 x 13

26 x 26

52 x 52

Prediction Feature Map

Attributes of a bounding box

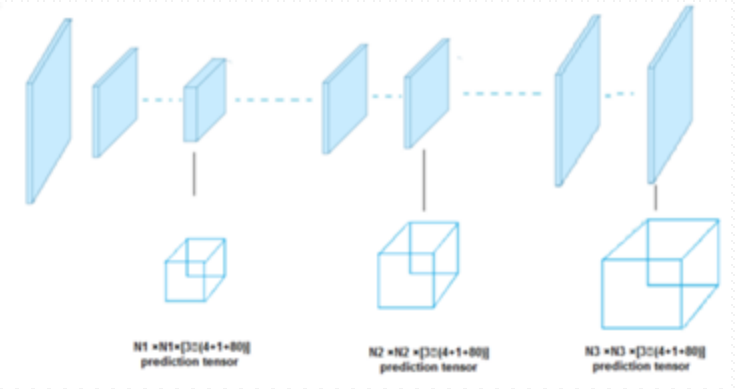


x B

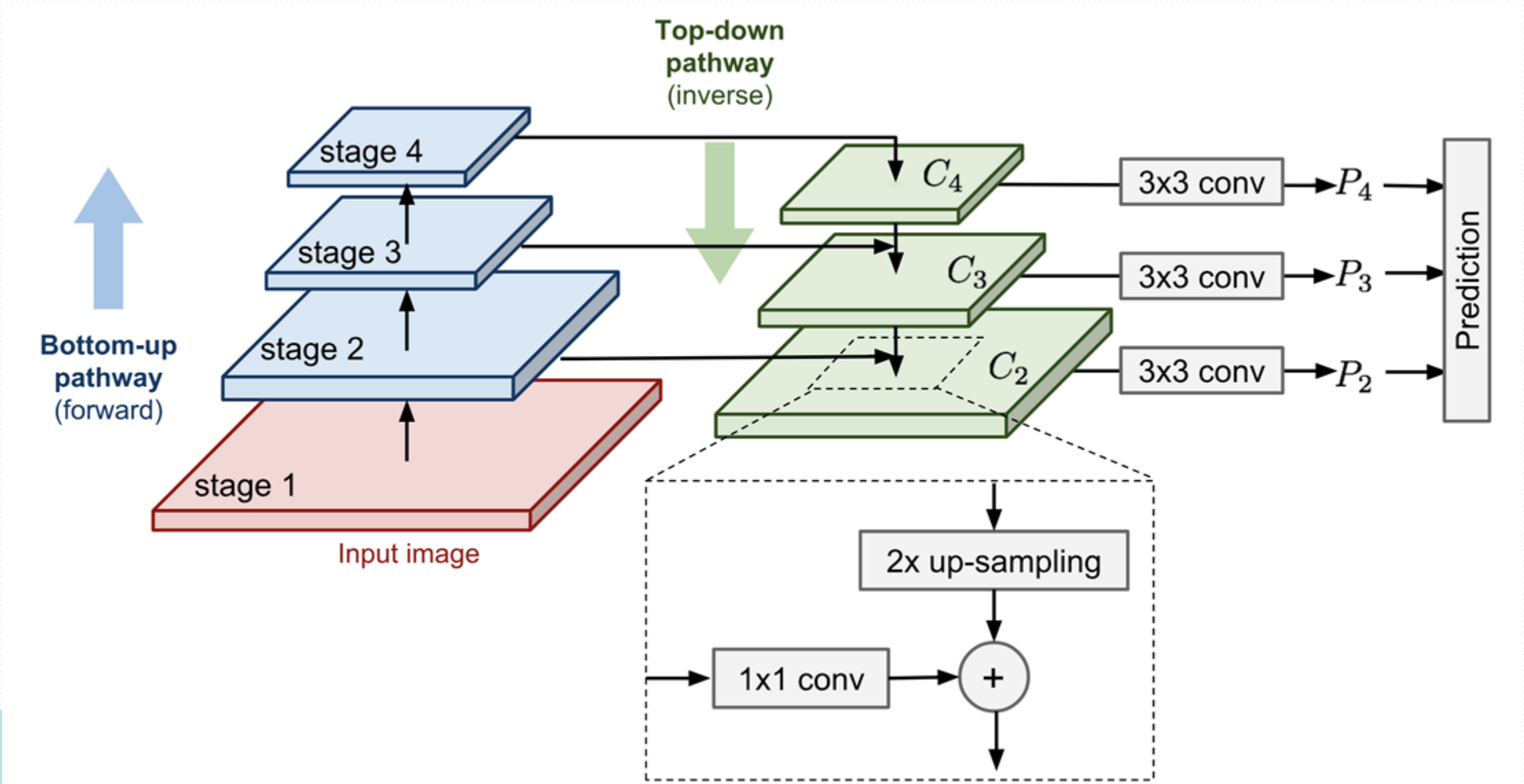
Box Co-ordinates

Objectness Score

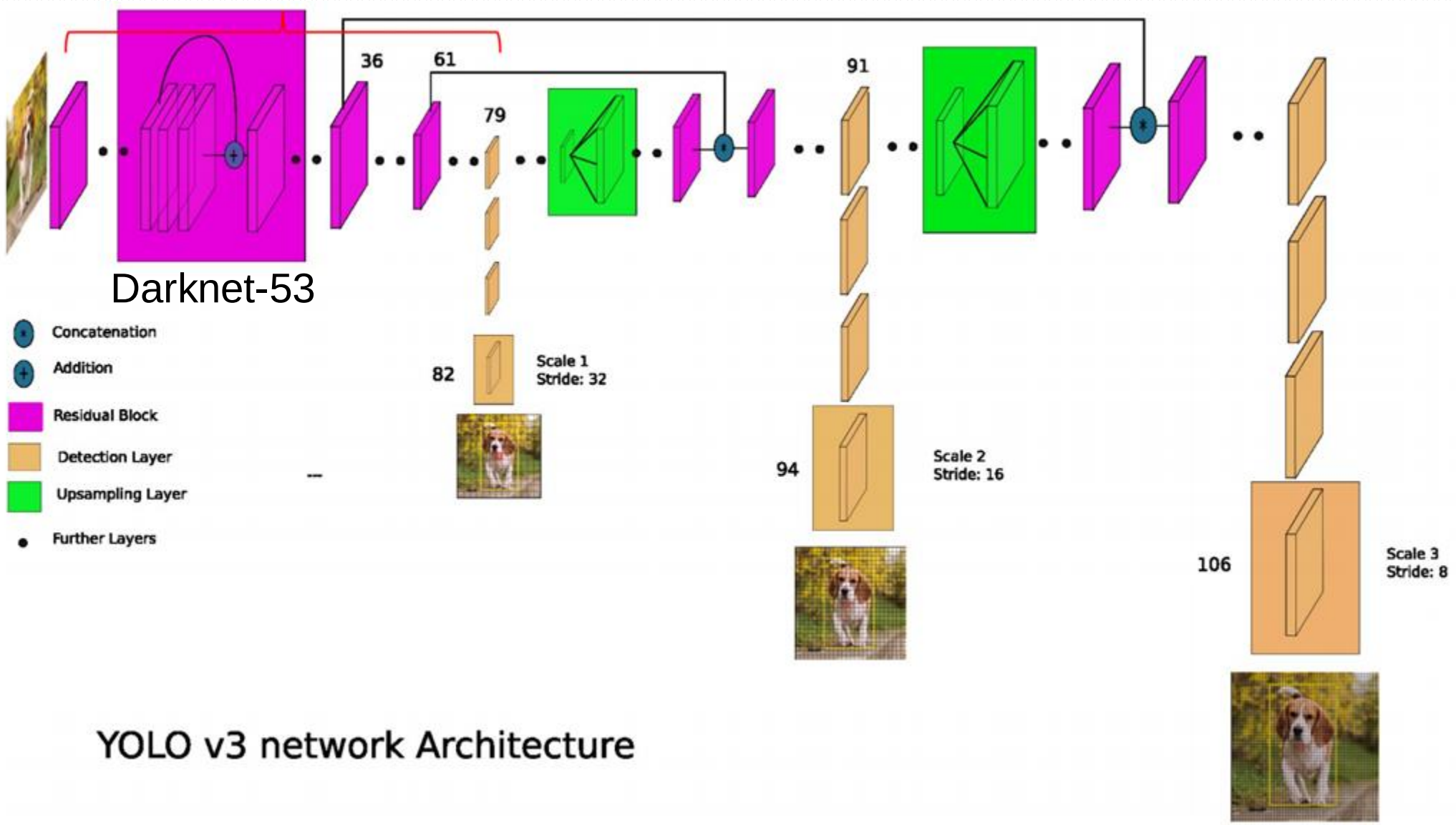
Class Scores



Feature Pyramid Networks



Network Architecture again...



THANK YOU

CONFUSION MATRIX

QUANTIFIABLE MEASURE THAT IS USED TO TRACK AND ASSESS THE STATUS OF A SPECIFIC PROCESS

Model classification & answer → TRUE / FALSE

- True Positive / TP
- False Positive / FP
- False Negative / FN
- True Negative / TN

		ACTUAL	
		TRUE	FALSE
PREDICTED	TRUE	**TRUE** POSITIVE	FALSE POSITIVE
	FALSE	FALSE NEGATIVE	**TRUE** NEGATIVE

PRECISION / RECALL

Precision $\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{\text{TP}}{\#\text{ predictions}}$

Proportion of what the actual true among the model classifies as true

Accuracy rate

Recall $\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{\text{TP}}{\#\text{ ground truths}}$

Proportion of what the model predicted to be true among the actual true

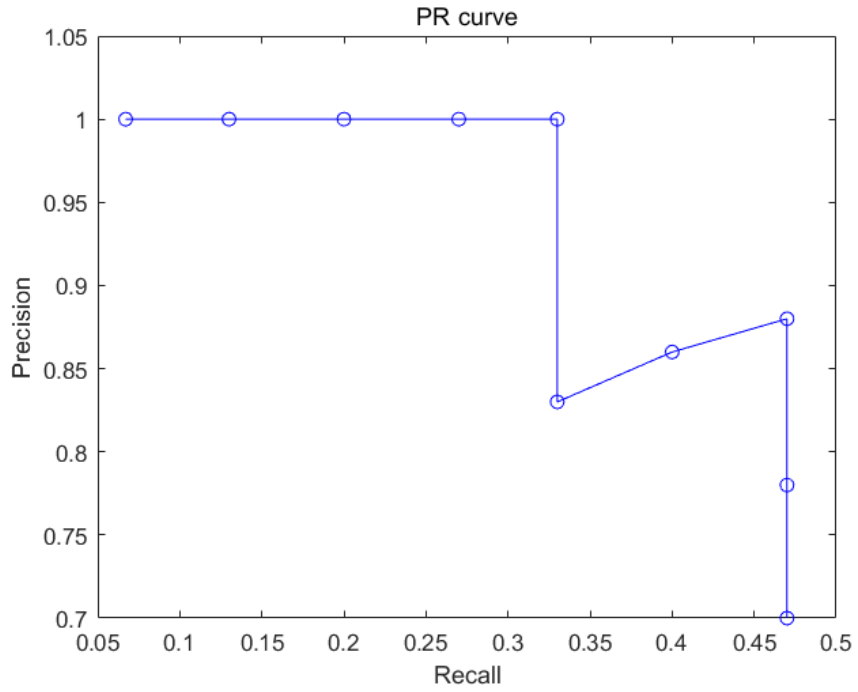
Detection rate

Precision = 67.7%

Recall = 33.3%

PR CURVE

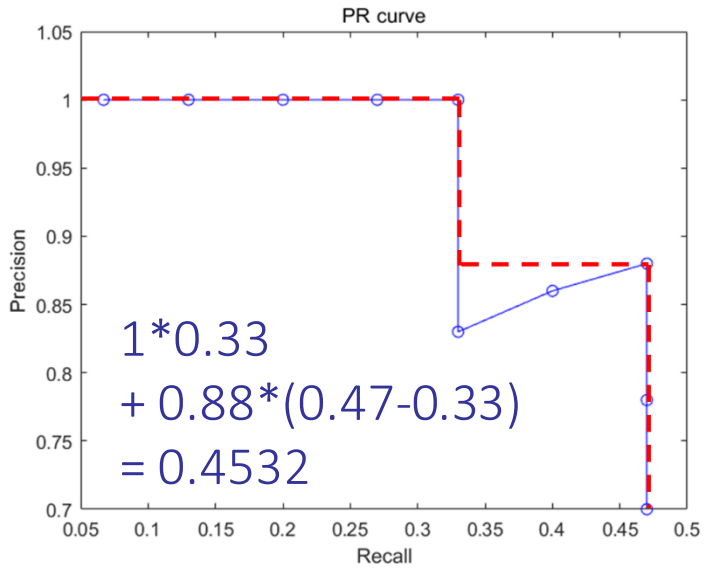
Detections	confidences	TP or FP	Precision	Recall
I	95%	TP	1/1=1	1/15=0.067
E	91%	TP	2/2=1	2/15=0.13
D	85%	TP	3/3=1	3/15=0.2
J	81%	TP	4/4=1	4/15=0.27
B	78%	TP	5/5=1	5/15=0.33
H	68%	FP	5/6=0.83	5/15=0.33
A	57%	TP	6/7=0.86	6/15=0.4
G	45%	TP	7/8=0.88	7/15=0.47
C	43%	FP	7/9=0.78	7/15=0.47
F	13%	FP	7/10=0.7	7/15=0.47



AP AVERAGE PRECISION / MAP MEAN AP

Measurement that combines recall and precision for ranked retrieval results

The general definition for the Average Precision (AP) is finding the area under the precision-recall curve above



$$\text{mAP} = \frac{1}{N}\sum_{i=1}^{N} \text{AP}_i$$

IOU INTERSECTION OVER UNION

EVALUATION METRIC USED TO MEASURE THE ACCURACY OF AN OBJECT DETECTOR

MOSTLY USED IN OBJECT DETECTION

PASCAL VOC challenge

$$IoU = \frac{TP}{(TP + FP + FN)}$$

PREREQUISITES

The ground-truth bounding boxes = hand-labeled bounding boxes

The predicted bounding boxes from the model.

$$IoU = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

GFLOP GIGA FLOATING POINT OPERATION

Amount of single precision giga-floating point operations calculated

Most of deep learning calculation is proceed in floating point

1 gigaFLOPS (GFLOPS) computer system is capable of performing one billion floating-point operations per second

Easy to represent a large dynamic range

Mostly used in picture information (graphics)

$$\mathrm{FLOPS} = \mathrm{cores} \times \mathrm{clock} \times \frac{\mathrm{FLOPs}}{\mathrm{cycle}}$$

Unit	Flops
kFLOPS	10^3
MFLOPS	10^6
GFLOPS	10^9
TFLOPS	10^{12}
PFLOPS	10^{15}
EFLOPS	10^{18}
ZFLOPS	10^{21}

DICE

Sørensen–Dice coefficient, Dice index

$$Dice = \frac{2\,|A \cap B|}{|A| + |B|} \qquad Dice = \frac{2 \times TP}{(TP + FP) + (TP + FN)}$$

$$|A \cap B| = \begin{bmatrix} 0.01 & 0.03 & 0.02 & 0.02 \\ 0.05 & 0.12 & 0.09 & 0.07 \\ 0.89 & 0.85 & 0.88 & 0.91 \\ 0.99 & 0.97 & 0.95 & 0.97 \end{bmatrix} * \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \xrightarrow{\text{element-wise multiply}} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0.89 & 0.85 & 0.88 & 0.91 \\ 0.99 & 0.97 & 0.95 & 0.97 \end{bmatrix} \xrightarrow{\text{sum}} 7.41$$

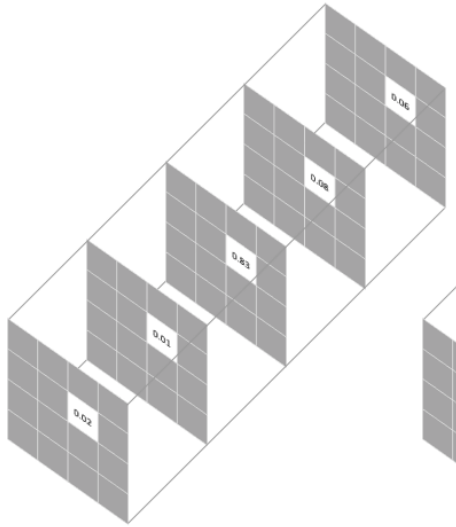
prediction target

$$|A| = \begin{bmatrix} 0.01 & 0.03 & 0.02 & 0.02 \\ 0.05 & 0.12 & 0.09 & 0.07 \\ 0.89 & 0.85 & 0.88 & 0.91 \\ 0.99 & 0.97 & 0.95 & 0.97 \end{bmatrix}^{2\ (optional)} \xrightarrow{\text{sum}} 7.82$$

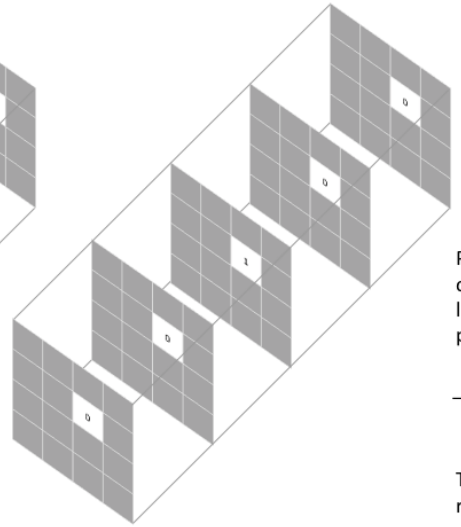
$$|B| = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}^{2\ (optional)} \xrightarrow{\text{sum}} 8$$

DICE LOSS

pixel-wise cross entropy loss



Prediction for a selected pixel



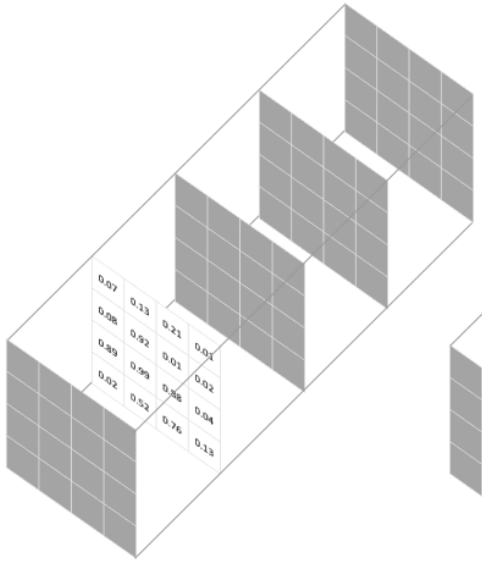
Target for the corresponding pixel

Pixel-wise loss is calculated as the log loss, summed over all possible classes

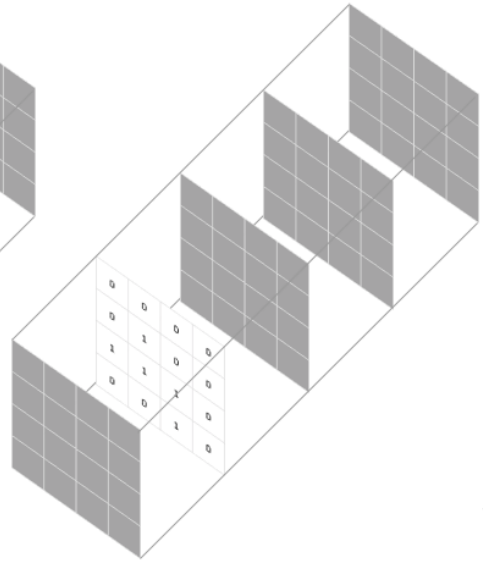
$$-\sum_{classes} y_{true} \log\left(y_{pred}\right)$$

This scoring is repeated over all **pixels** and averaged

Dice loss → 1 - Dice



Prediction for a selected class



Target for the corresponding class

Soft Dice coefficient is calculated for each class mask

$$1 - \frac{2\sum_{pixels} y_{true} y_{pred}}{\sum_{pixels} y_{true}^2 + \sum_{pixels} y_{pred}^2}$$

This scoring is repeated over all **classes** and averaged

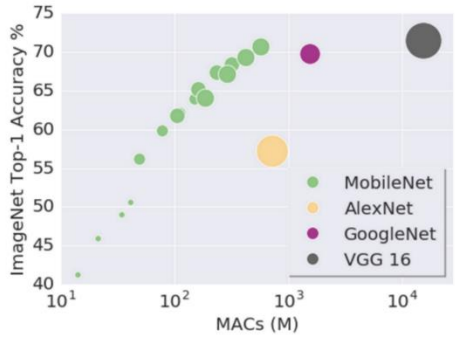
MOBILENET

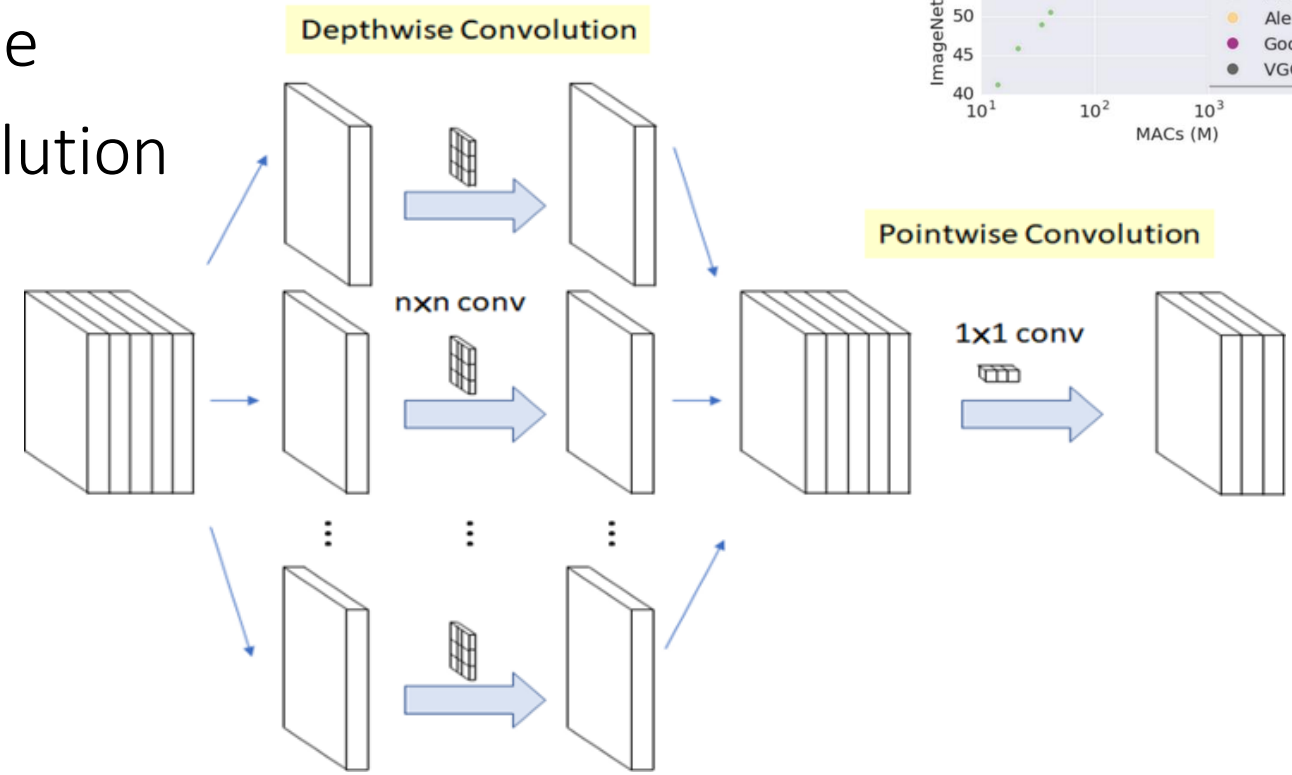
CNN ALGORITHM AIMED TO USE IN LIMITED COMPUTER PERFORMANCE AND BATTERY PERFORMANCE

FEATURES

- It can be used in mobile platform
- Minimizing computation time
- Depth-wise separable convolution
- Depth-wise convolution
- point-wise convolution

8~9 TIMES BETTER OPTIMIZED





Data Set

Challenge	Object Classes	Number of Images			Number of Annotated Objects	
		Train	Val	Test	Train	Val
PASCAL VOC Object Detection Challenge						
VOC07	20	2,501	2,510	4,952	6,301(7,844)	6,307(7,818)
VOC08	20	2,111	2,221	4,133	5,082(6,337)	5,281(6,347)
VOC09	20	3,473	3,581	6,650	8,505(9,760)	8,713(9,779)
VOC10	20	4,998	5,105	9,637	11,577(13,339)	11,797(13,352)
VOC11	20	5,717	5,823	10,994	13,609(15,774)	13,841(15,787)
VOC12	20	5,717	5,823	10,991	13,609(15,774)	13,841(15,787)
ILSVRC Object Detection Challenge						
ILSVRC13	200	395,909	20,121	40,152	345,854	55,502
ILSVRC14	200	456,567	20,121	40,152	478,807	55,502
ILSVRC15	200	456,567	20,121	51,294	478,807	55,502
ILSVRC16	200	456,567	20,121	60,000	478,807	55,502
ILSVRC17	200	456,567	20,121	65,500	478,807	55,502
MS COCO Object Detection Challenge						
MS COCO15	80	82,783	40,504	81,434	604,907	291,875
MS COCO16	80	82,783	40,504	81,434	604,907	291,875
MS COCO17	80	118,287	5,000	40,670	860,001	36,781
MS COCO18	80	118,287	5,000	40,670	860,001	36,781
Open Images Object Detection Challenge						
OID18	500	1,743,042	41,620	125,436	12,195,144	–

https://github.com/hoya012/deep_learning_object_detection/blob/master/assets/deep_learning_object_detection_dataset.PNG

- ICCV
- ICLR
- CVPR
- ECCV
- NIPS