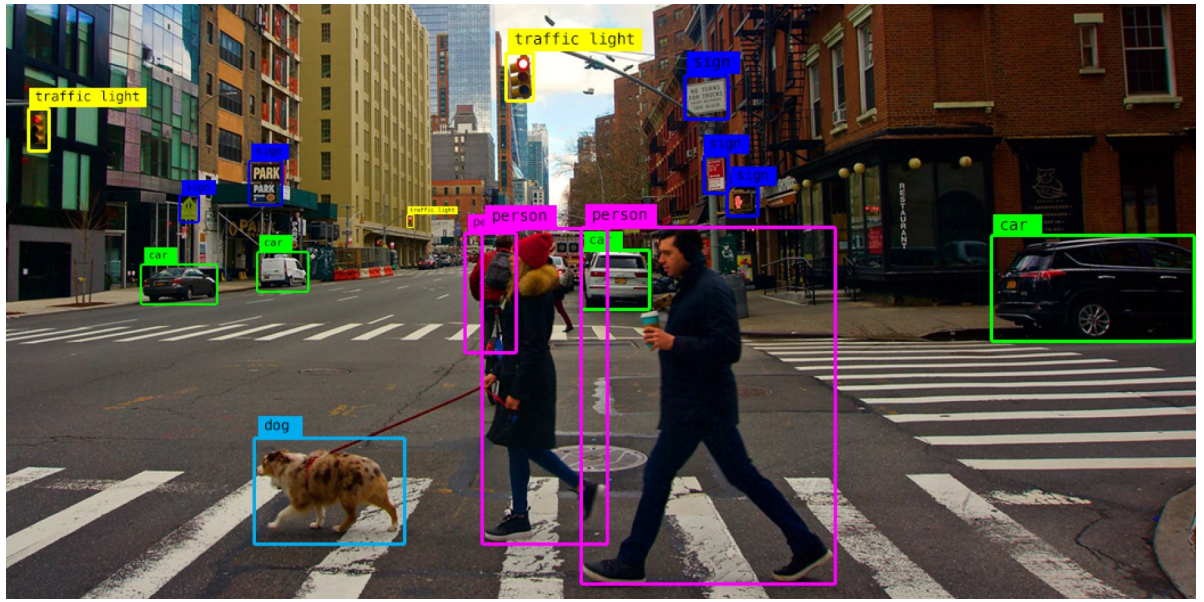
AI VIETNAM
All-in-One Course

Object Detection
(Part 1: Traditional Algorithms)



Vinh Dinh Nguyen - PhD in Computer Science
Hung-An & Minh-Duc Bui - TA

Year 2023

Outline

- Introduction to Object Detection
- Applications of Object Detection
- Object Detection Milestones
- Traditional Object Detectors
- CNNs for Image Classification
- Image Classifier to Object Detectors with CNN
- CNN Limitations and Spatial Outputs
- Assignment

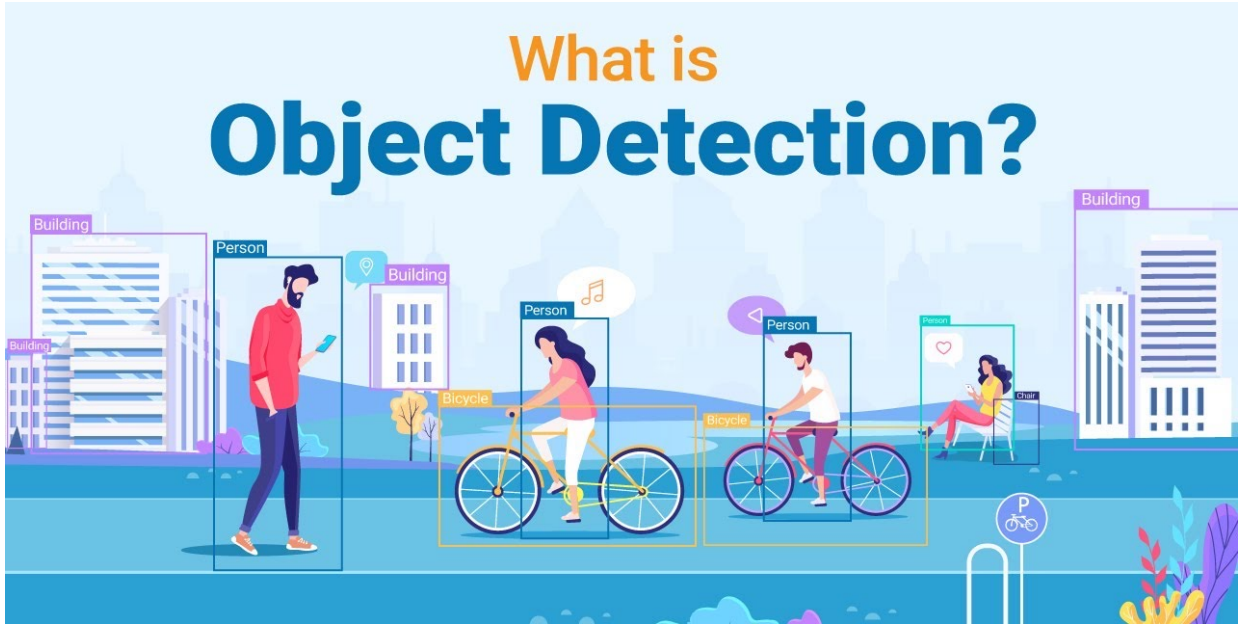
Outline

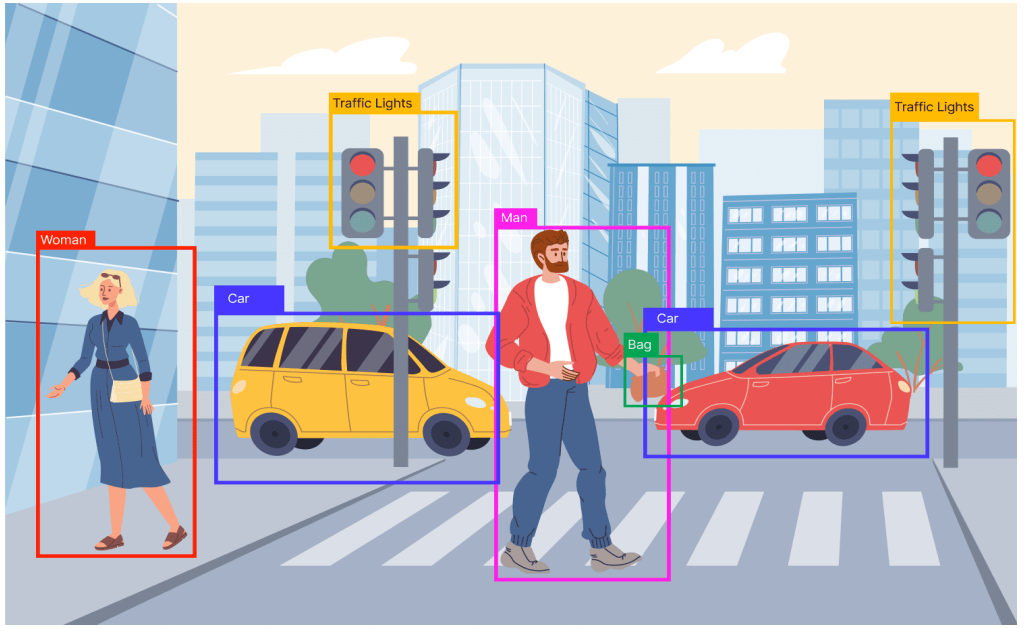
- Introduction to Object Detection
- Applications of Object Detection
- Object Detection Milestones
- Traditional Object Detectors
- CNNs for Image Classification
- Image Classifier to Object Detectors with CNN
- CNN Limitations and Spatial Outputs
- Assignment

What is an Object Detection?

The objective of object detection is to develop computational models and techniques that provide one of the most basic pieces of information needed by computer vision applications:

What objects are where?

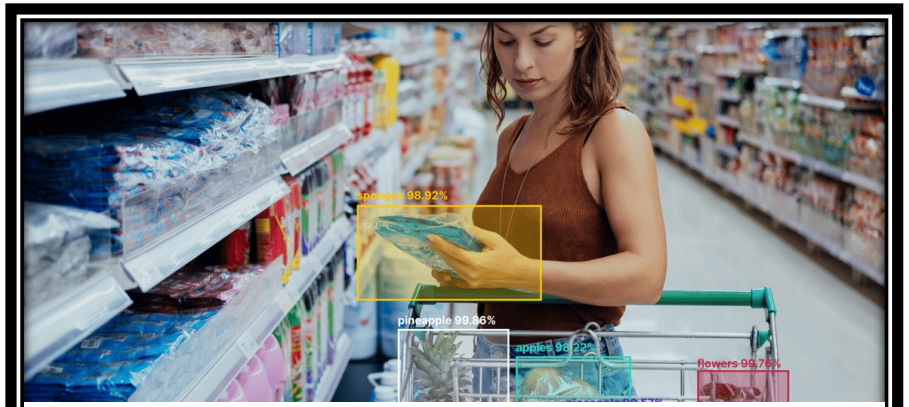




Outline

- Introduction to Object Detection
- Applications of Object Detection
- Object Detection Milestones
- Traditional Object Detectors
- CNNs for Image Classification
- Image Classifier to Object Detectors with CNN
- CNN Limitations and Spatial Outputs
- Assignment

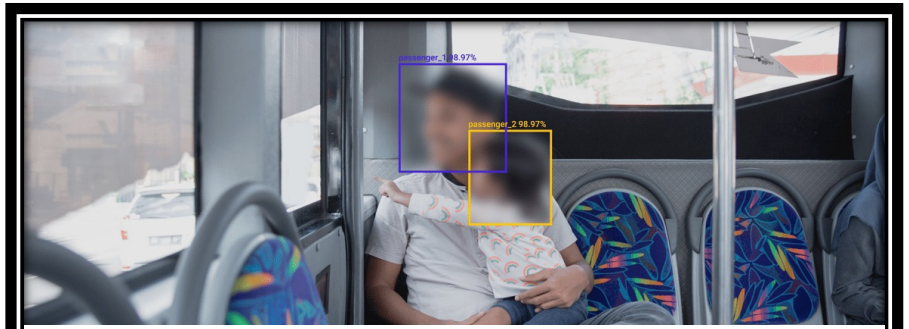
Object Detection for Businesses



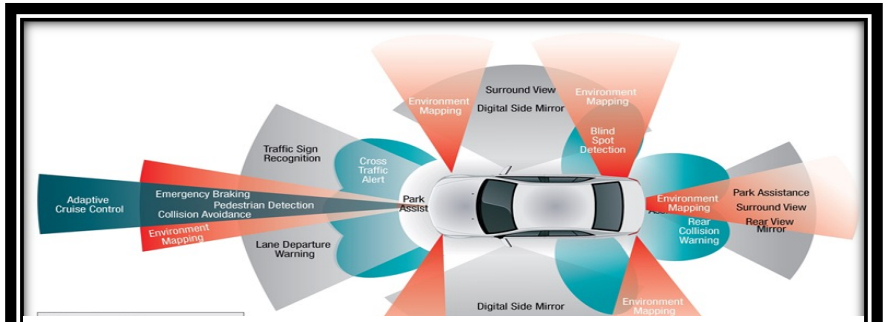
Retail

Industrial Use Cases

Medical



Transportation



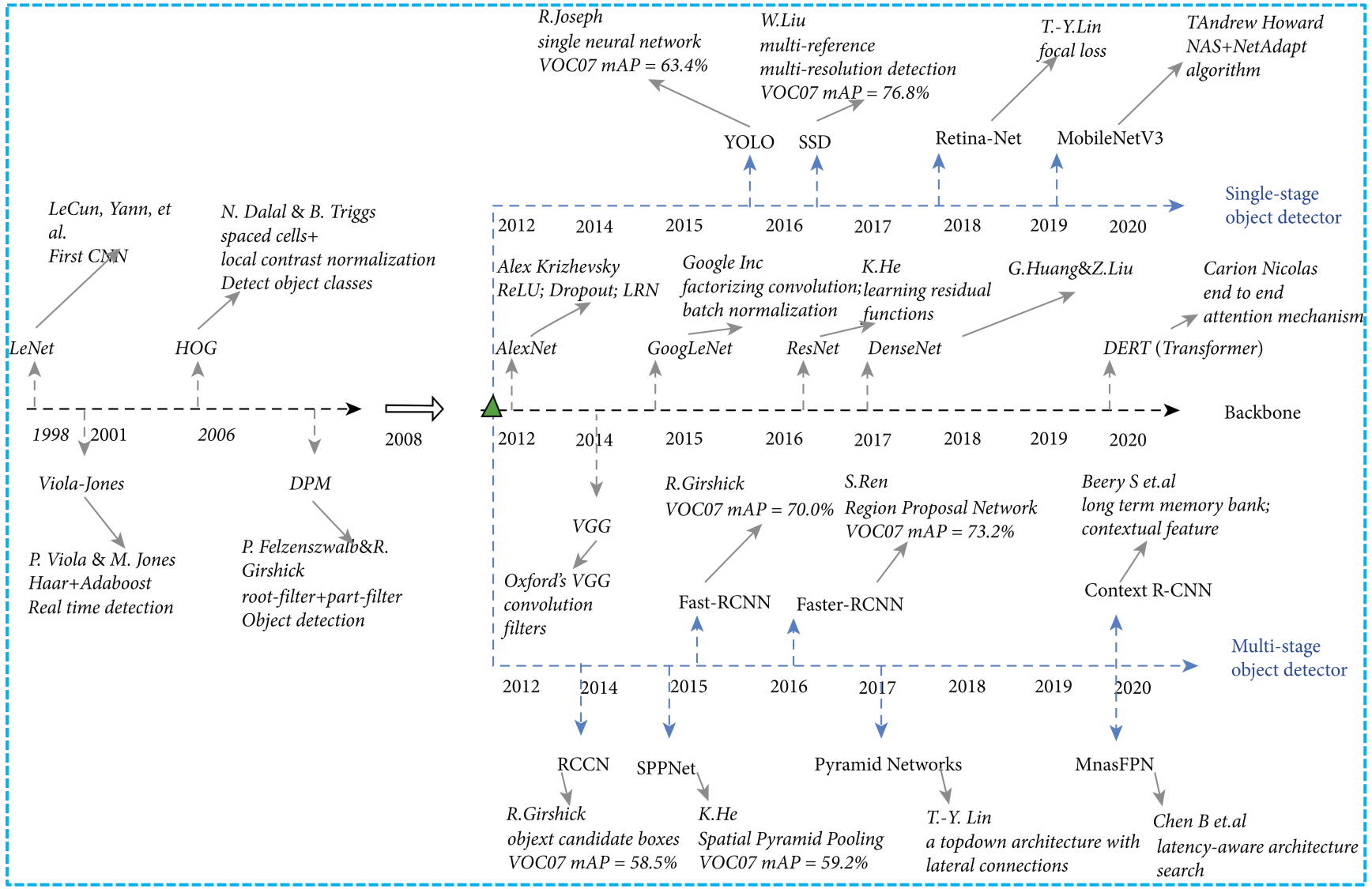
Autonomous Vehicle

...

Outline

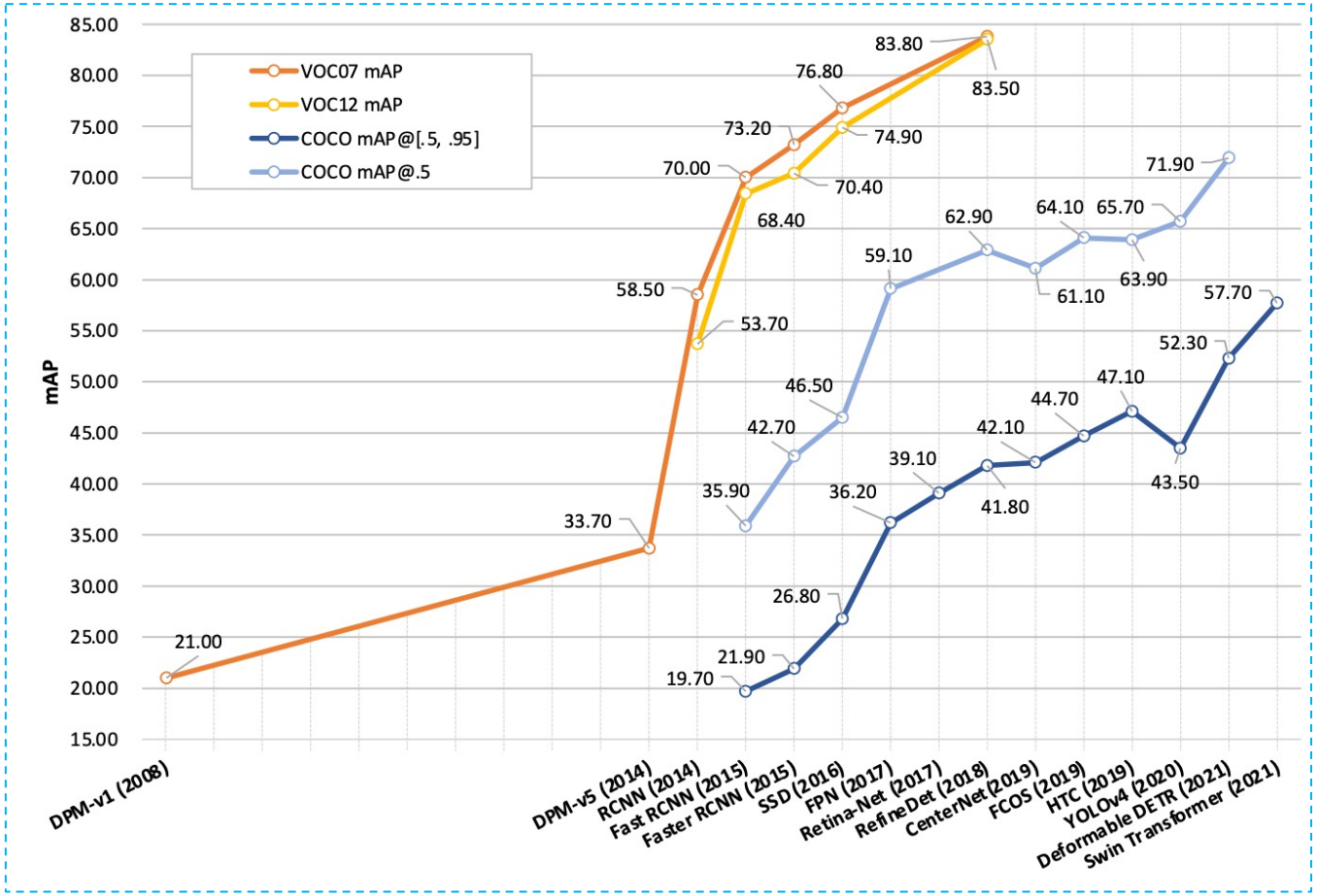
- Introduction to Object Detection
- Applications of Object Detection
- Object Detection Milestones
- Traditional Object Detectors
- CNNs for Image Classification
- Image Classifier to Object Detectors with CNN
- CNN Limitations and Spatial Outputs
- Assignment

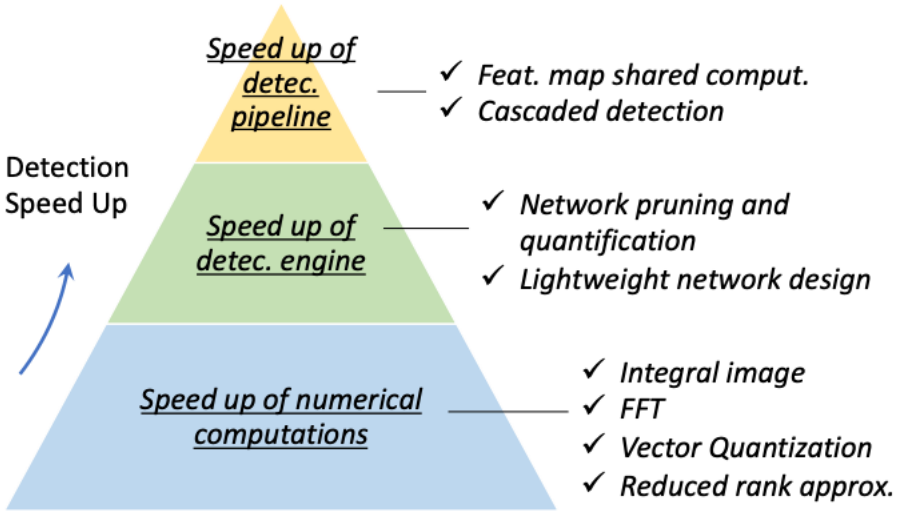
Object Detection Milestones



Shaoxi Li et al. "Survey on Deep Learning-Based Marine Object Detection", 2021

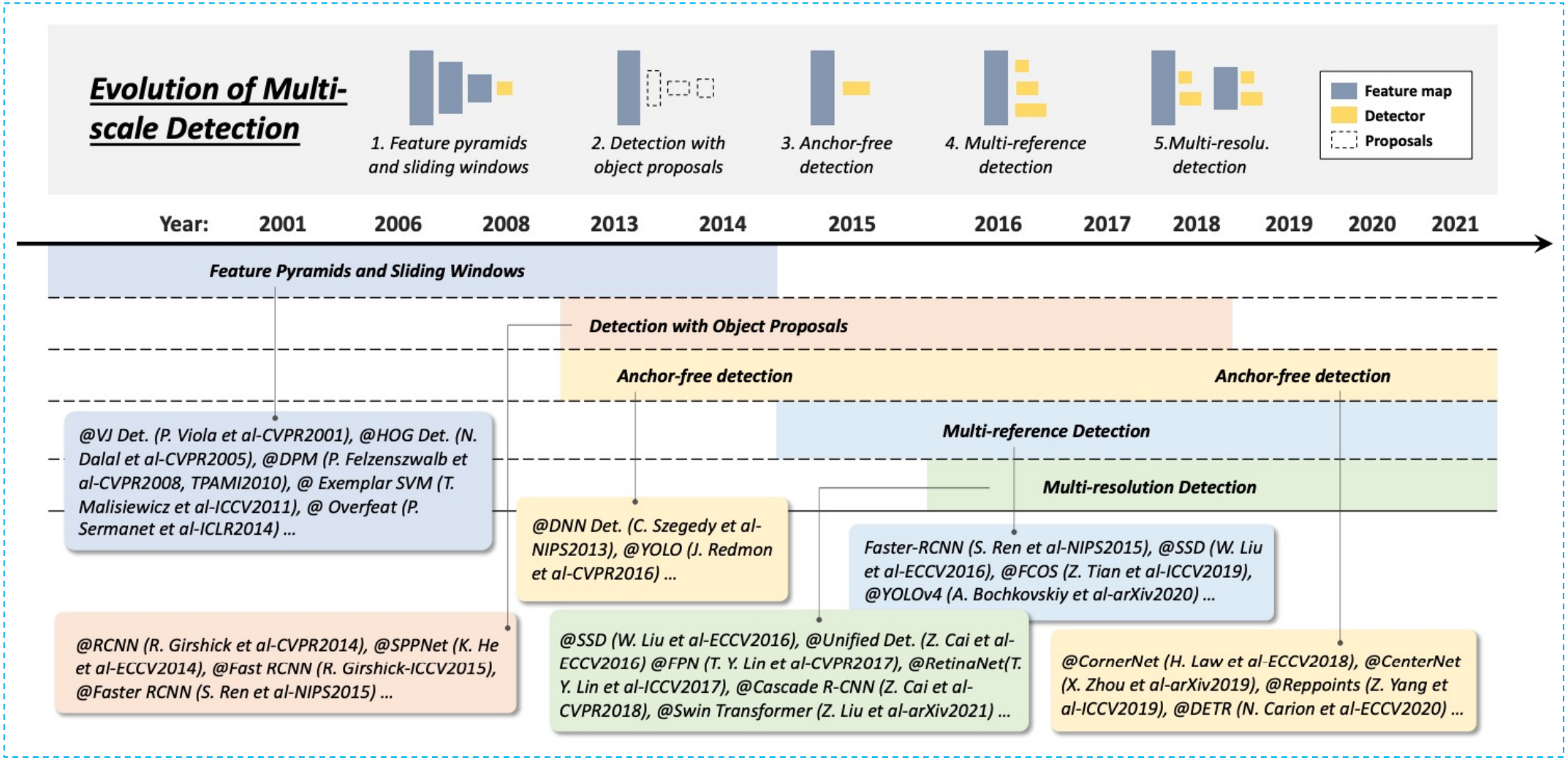
Object Detection Milestones





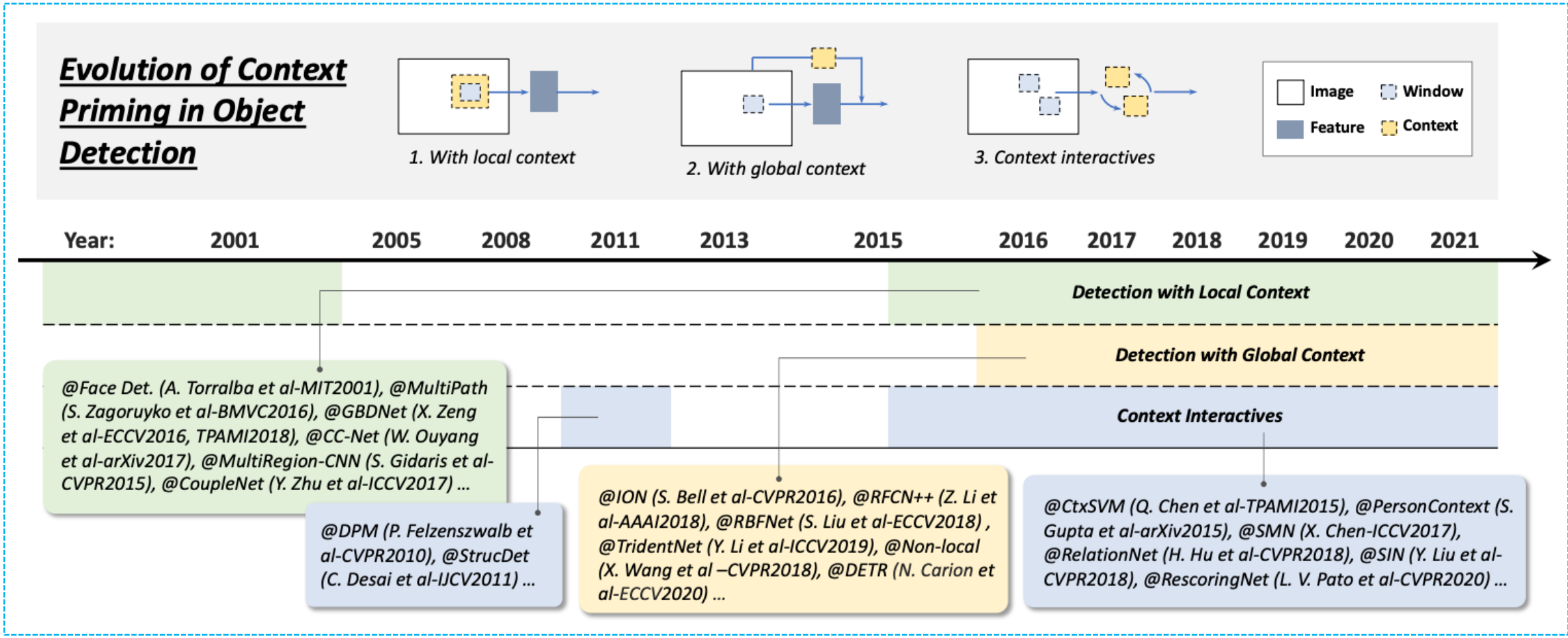
Z. Zou, K. Chen, Z. Shi, Y. Guo and J. Ye, "Object Detection in 20 Years: A Survey," in Proceedings of the IEEE, vol. 111, no. 3, pp. 257-276, March 2023, doi: 10.1109/JPROC.2023.3238524.

Object Detection Milestones



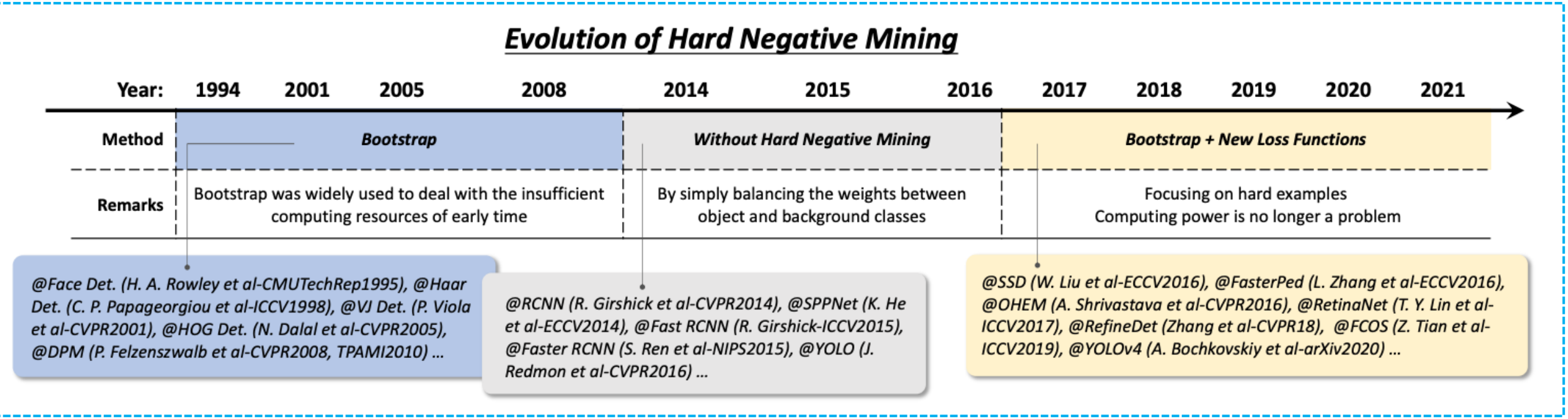
Z. Zou, K. Chen, Z. Shi, Y. Guo and J. Ye, "Object Detection in 20 Years: A Survey," in Proceedings of the IEEE, vol. 111, no. 3, pp. 257-276, March 2023, doi: 10.1109/JPROC.2023.3238524.

Object Detection Milestones



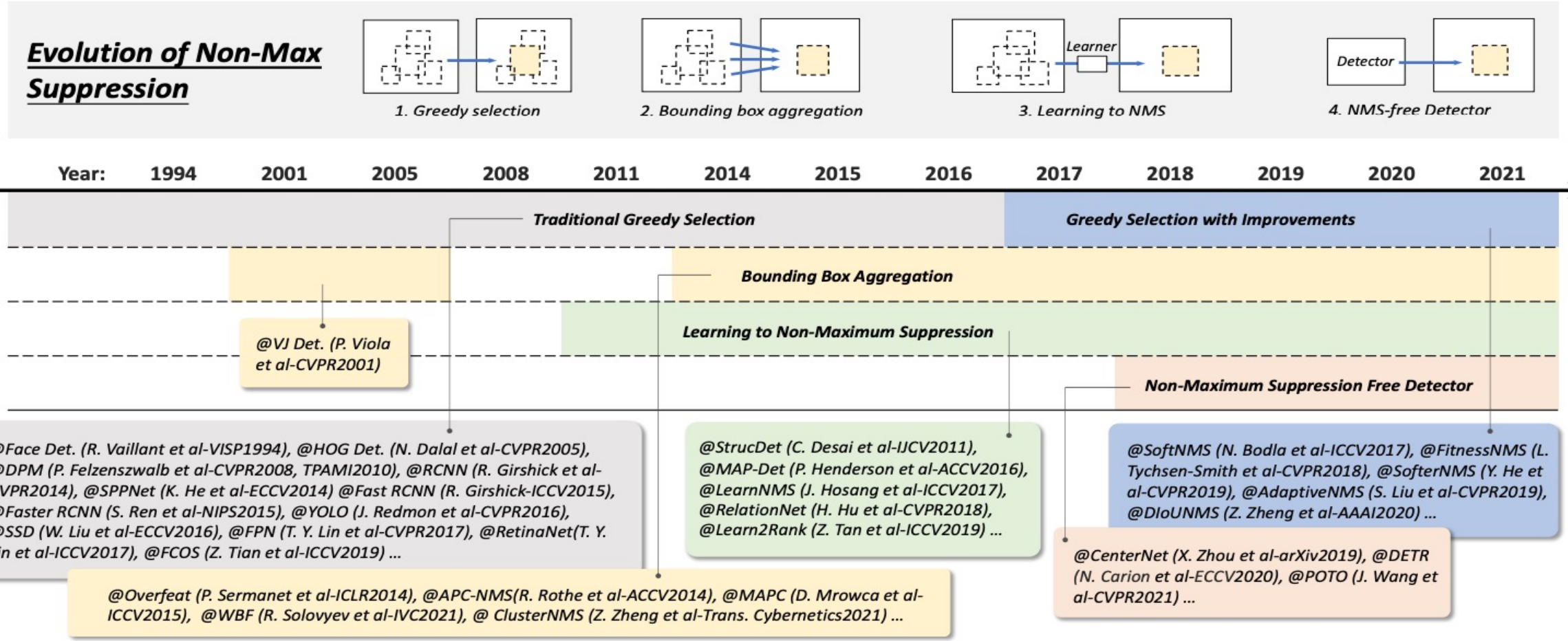
Z. Zou, K. Chen, Z. Shi, Y. Guo and J. Ye, "Object Detection in 20 Years: A Survey," in Proceedings of the IEEE, vol. 111, no. 3, pp. 257-276, March 2023, doi: 10.1109/JPROC.2023.3238524.

Object Detection Milestones



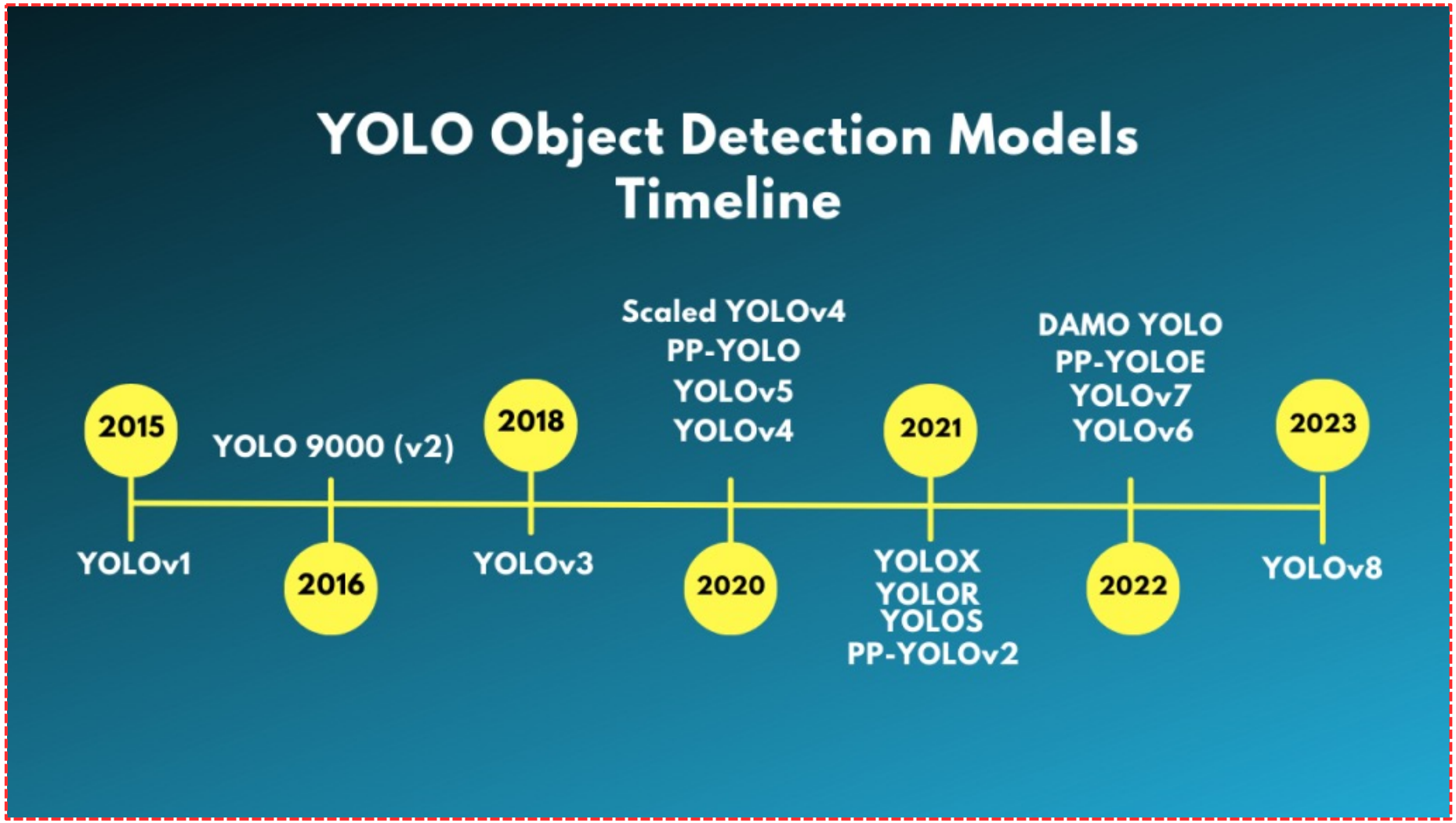
Z. Zou, K. Chen, Z. Shi, Y. Guo and J. Ye, "Object Detection in 20 Years: A Survey," in Proceedings of the IEEE, vol. 111, no. 3, pp. 257-276, March 2023, doi: 10.1109/JPROC.2023.3238524.

Object Detection Milestones



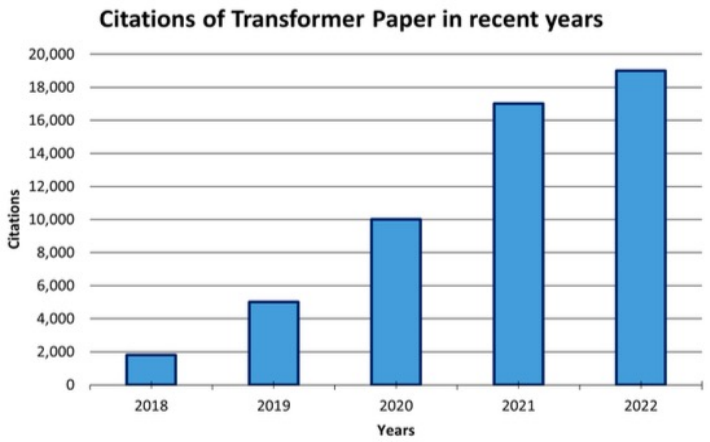
Z. Zou, K. Chen, Z. Shi, Y. Guo and J. Ye, "Object Detection in 20 Years: A Survey," in Proceedings of the IEEE, vol. 111, no. 3, pp. 257-276, March 2023, doi: 10.1109/JPROC.2023.3238524.

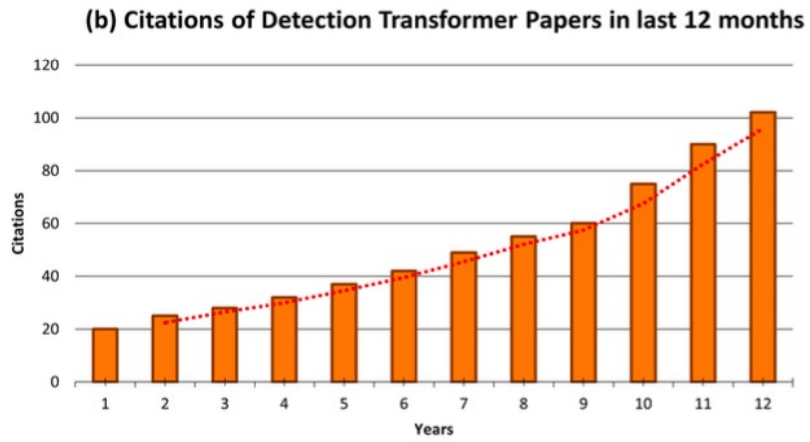
Object Detection Milestones

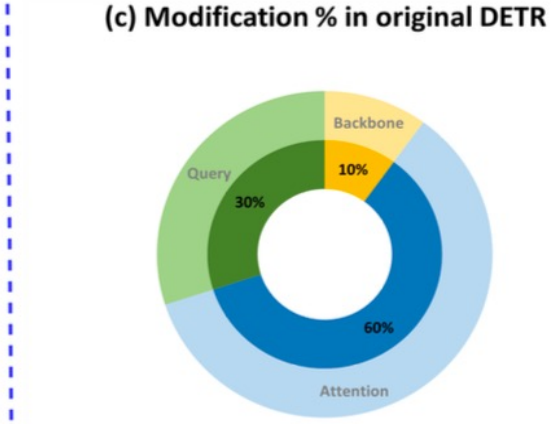


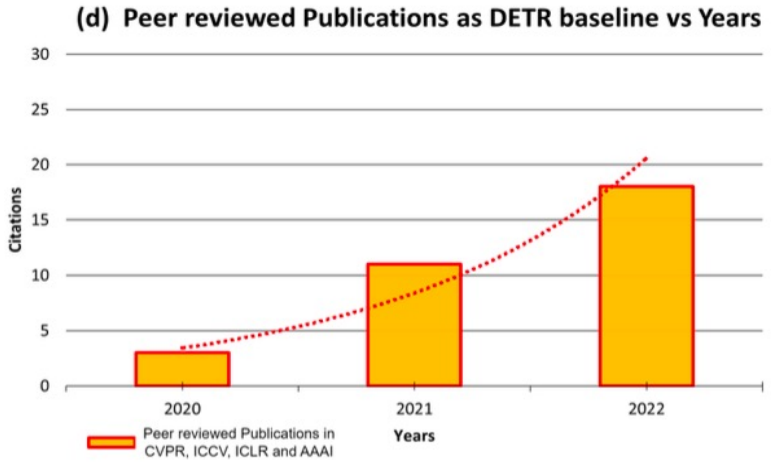
Z. Zou, K. Chen, Z. Shi, Y. Guo and J. Ye, "Object Detection in 20 Years: A Survey," in Proceedings of the IEEE, vol. 111, no. 3, pp. 257-276, March 2023, doi: 10.1109/JPROC.2023.3238524.

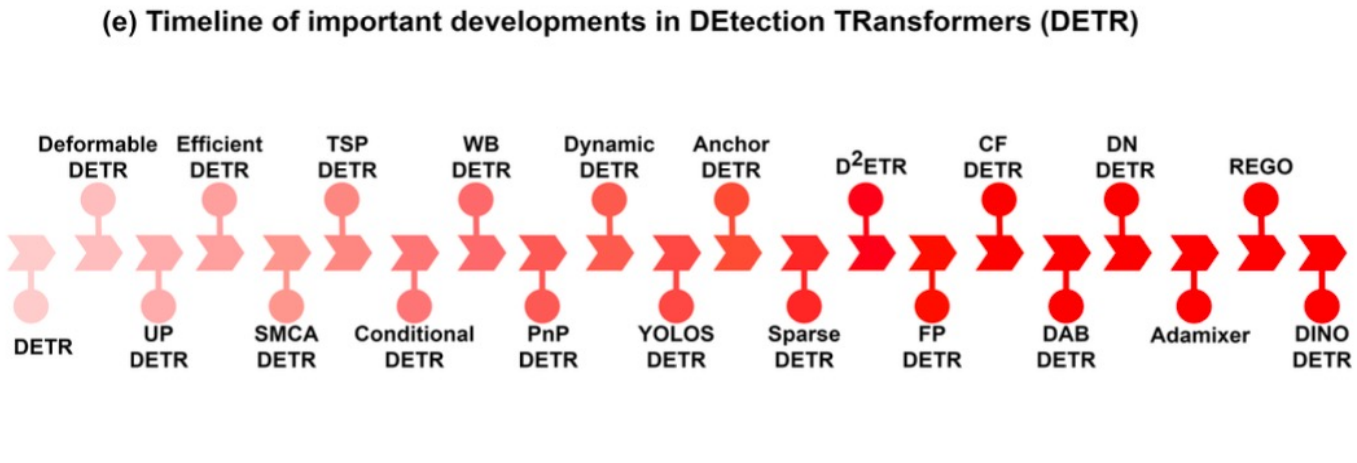
Object Detection: Transformer





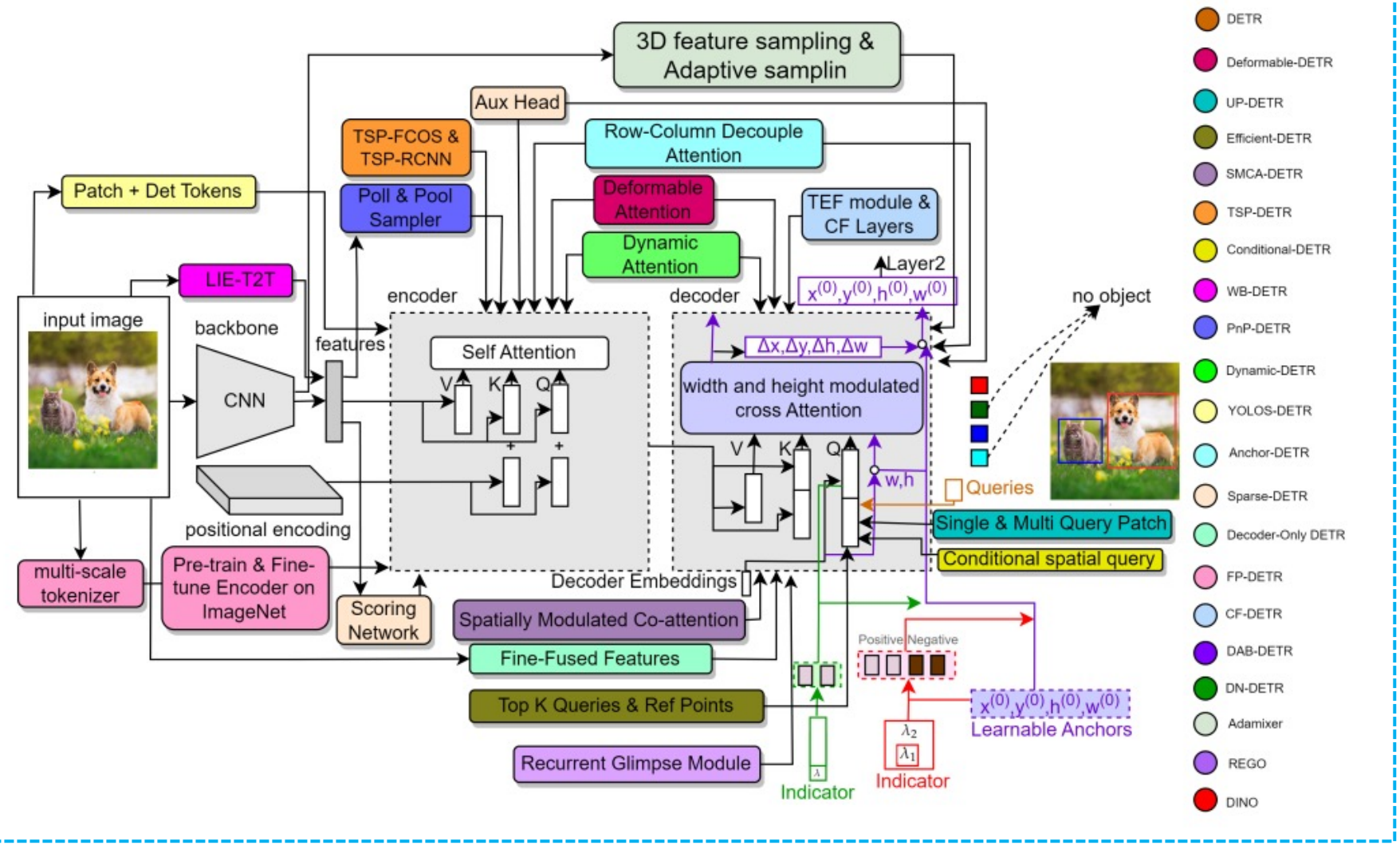






arXiv:2306.04670 [cs.CV]

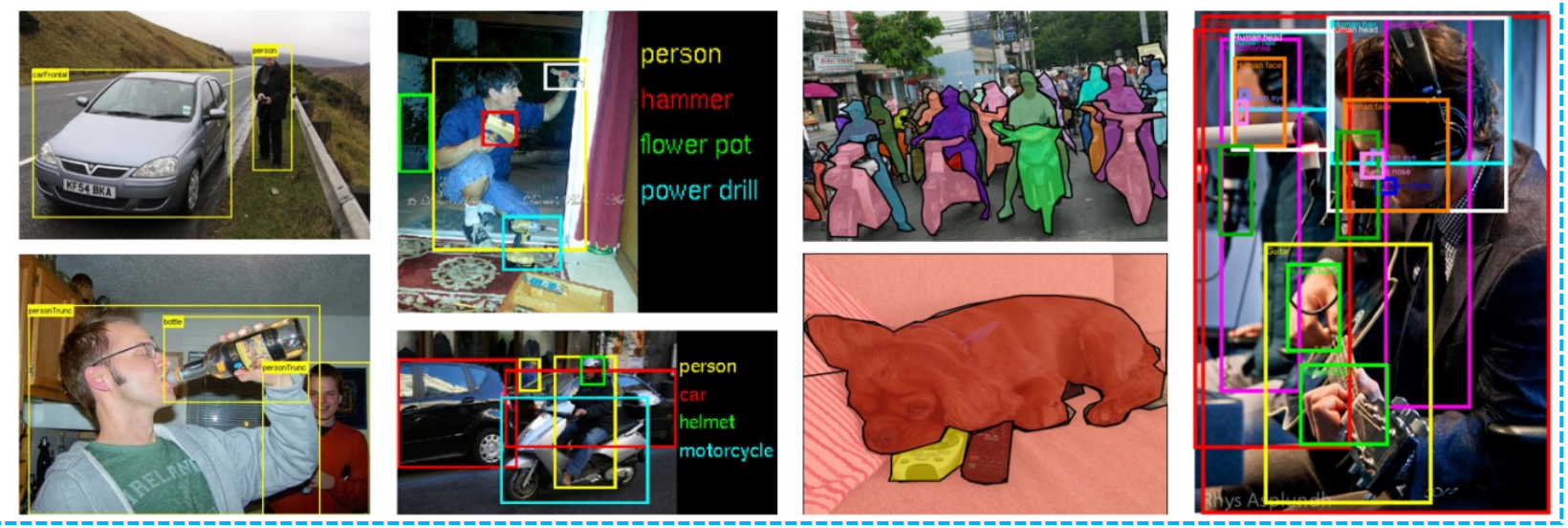
Object Detection: Transformer



An overview of the Detection Transformer (DETR) and its modifications proposed by recent methods to improve performance and training convergence.

arXiv:2306.04670 [cs.CV]

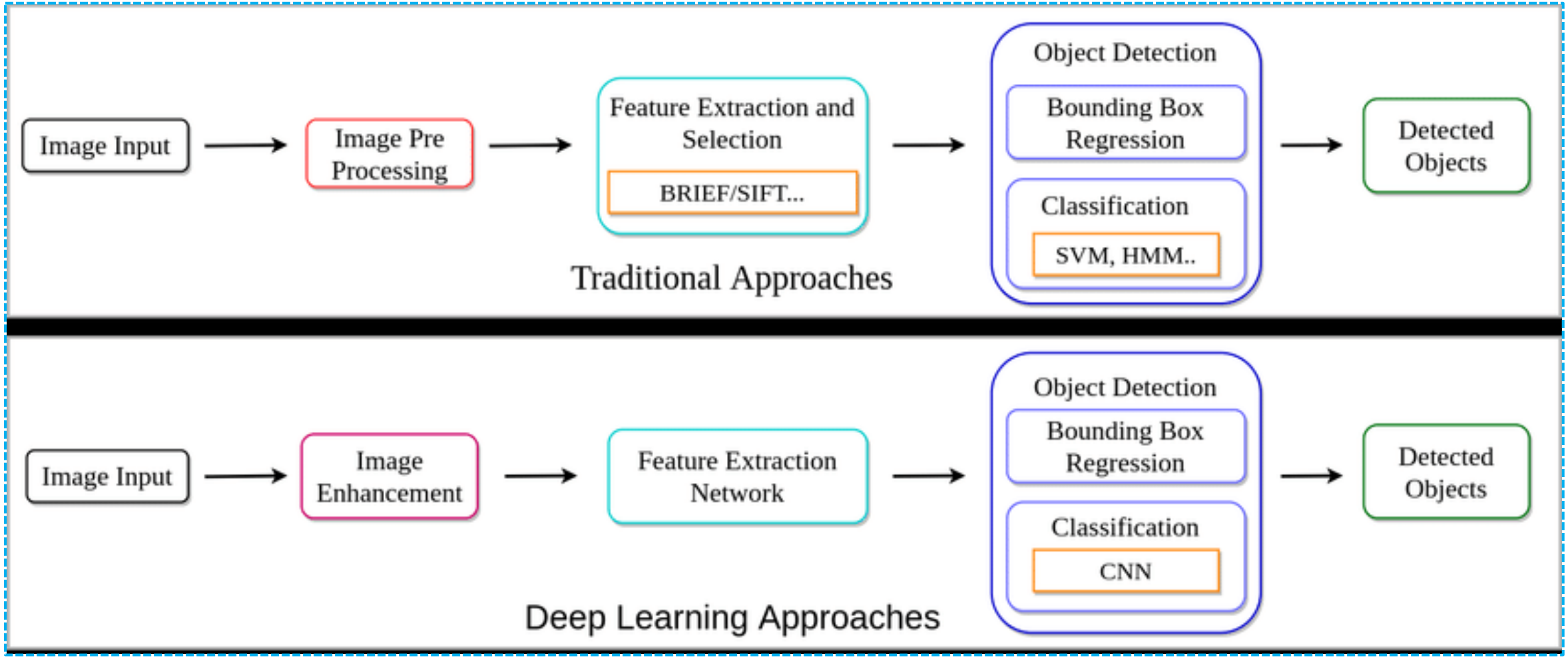
Well-known Object Detection Datasets



Z. Zou, K. Chen, Z. Shi, Y. Guo and J. Ye, "Object Detection in 20 Years: A Survey," in Proceedings of the IEEE, vol. 111, no. 3, pp. 257-276, March 2023, doi: 10.1109/JPROC.2023.3238524.

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

Traditional Vs. Deep Learning



Object Detection Milestones

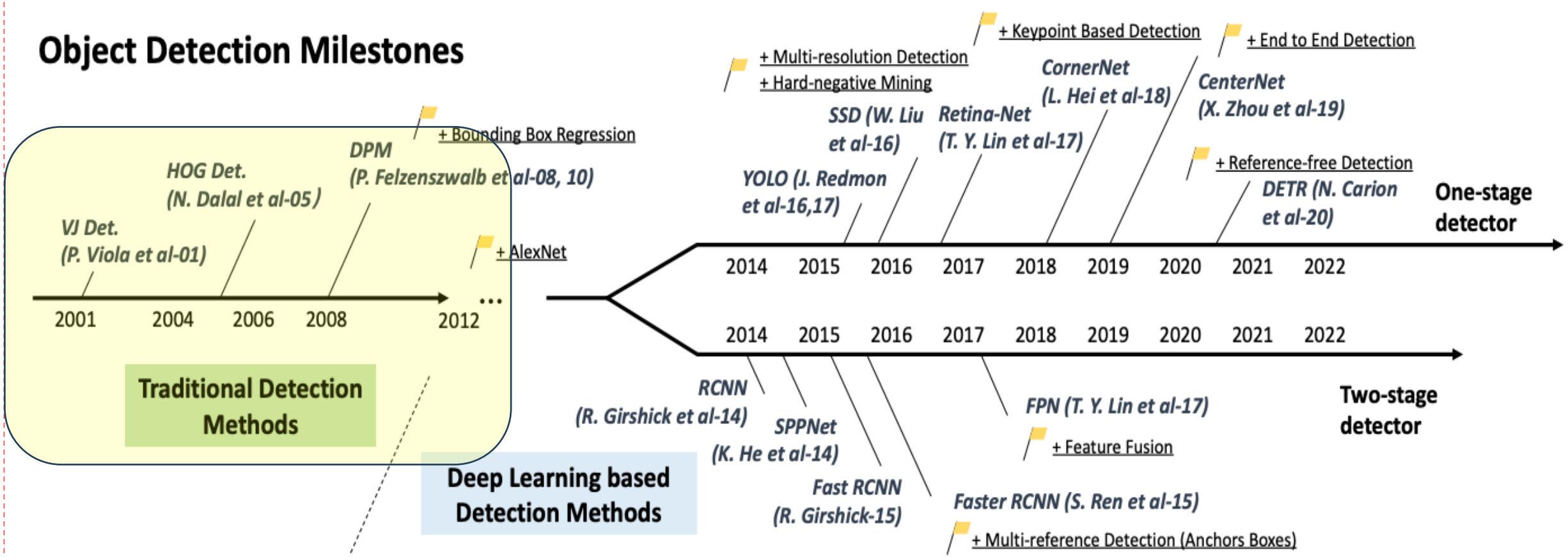
Milestones: Traditional Detectors
Viola Jones Detectors, SVM + HOG & DPM

Milestones: CNN based Two-stage Detectors
RCNN, SPPNet, Faster RCNN, Faster RCNN,..

Milestones: CNN based One-stage Detectors
YOLO, SSD, RetinaNet, CornerNet, Center Net,..

Milestones: Transformer for OD
DETR, D-DETR, DINO,...

Object Detection Milestones



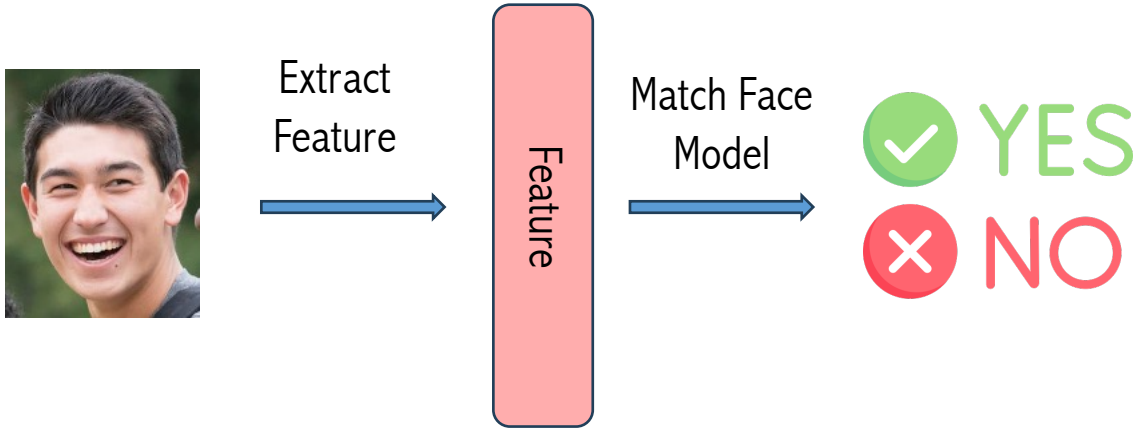
Z. Zou, K. Chen, Z. Shi, Y. Guo and J. Ye, "Object Detection in 20 Years: A Survey," in Proceedings of the IEEE, vol. 111, no. 3, pp. 257-276, March 2023, doi: 10.1109/JPROC.2023.3238524.

Outline

- Introduction to Object Detection
- Applications of Object Detection
- Object Detection Milestones
- Traditional Object Detectors
- CNNs for Image Classification
- Image Classifier to Object Detectors with CNN
- CNN Limitations and Spatial Outputs
- Assignment

Traditional Detectors

Slides windows of differents sizes across image.
At each location match window to face model



Features:
How to extract feature?
Which features represent face well?

Classifier:
How to build a model and classify features as face or not?

What are good features?

Interest Point (Edge, Corners, SIFT,..)

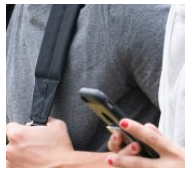
Facial Components (Templates)?

- Extreme Fast to Compute
- Millions of windows in an image

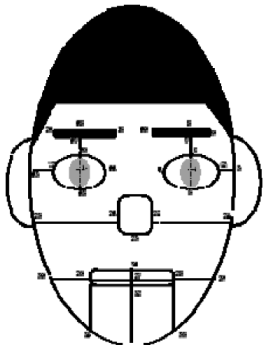
The key aspect in face recognition is detecting relevant features in human face like eyes, eyebrows, nose, lips. So how do we detect these features in real time/in an image ?

Discriminative Face / None Face

$\neq$

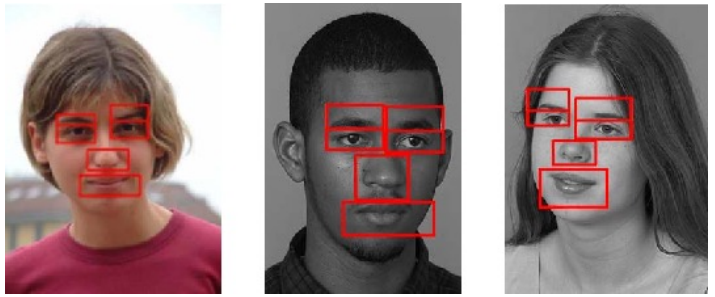


Methods for Face Detection



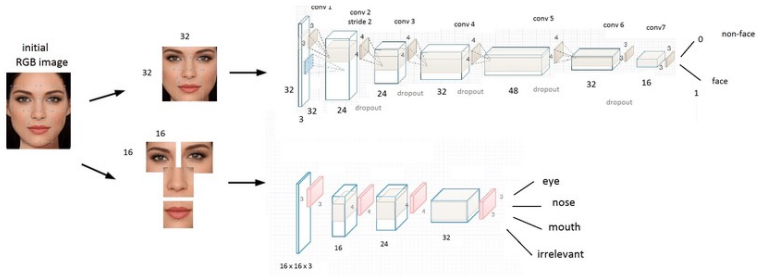
Knowledge Based

- Rule based (Ex: X must have eyes, x must have a nose)
- Too many rules and variables with this method



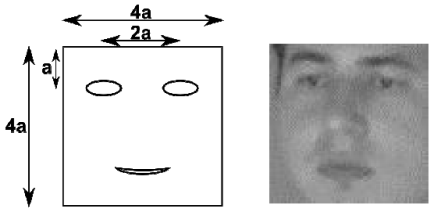
Feature Based

Locate and extract structural features in the face
Find a differential between facial and non facial regions in an image



Appearance Based

Learn the characteristics of a face. Example: CNN's
Accuracy depends on training data (which can be scarce)

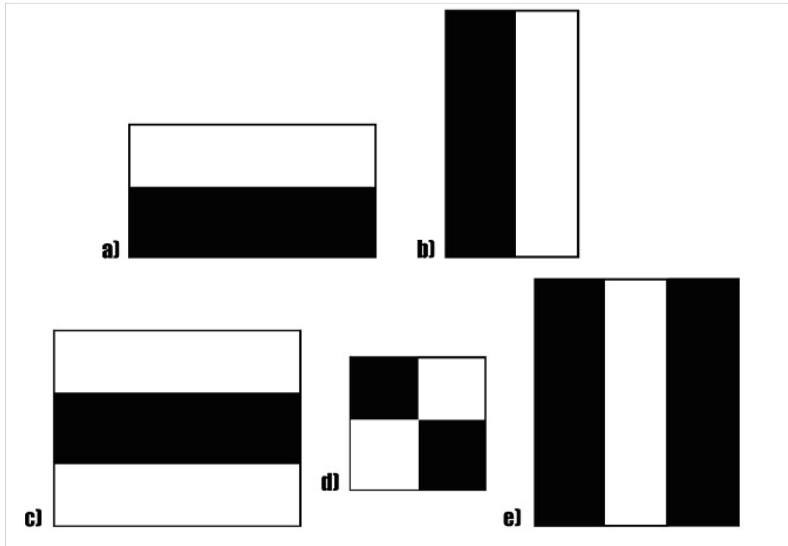


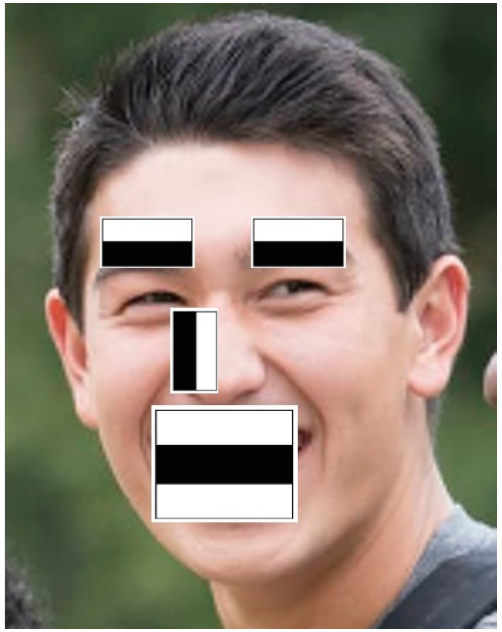
Template

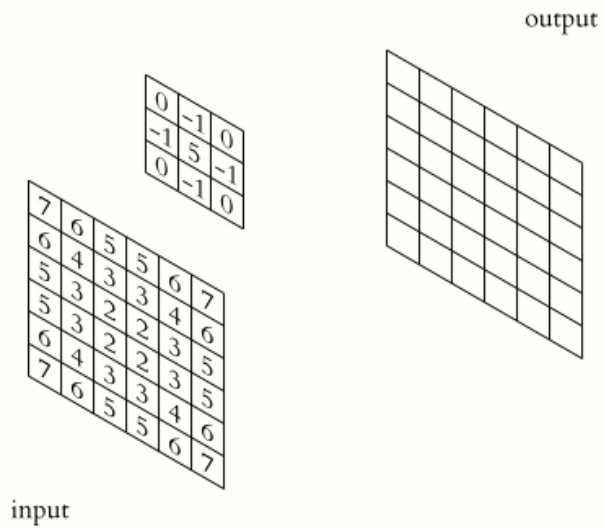
Using predefined templates for edge detection. Quick and easy
A trade off for speed over accuracy

Viola Jones Detectors : Haar Feature

Running on a 700MHz Pentium III CPU, the detector was tens or even hundreds of times faster than other algorithms in its time under comparable detection accuracy



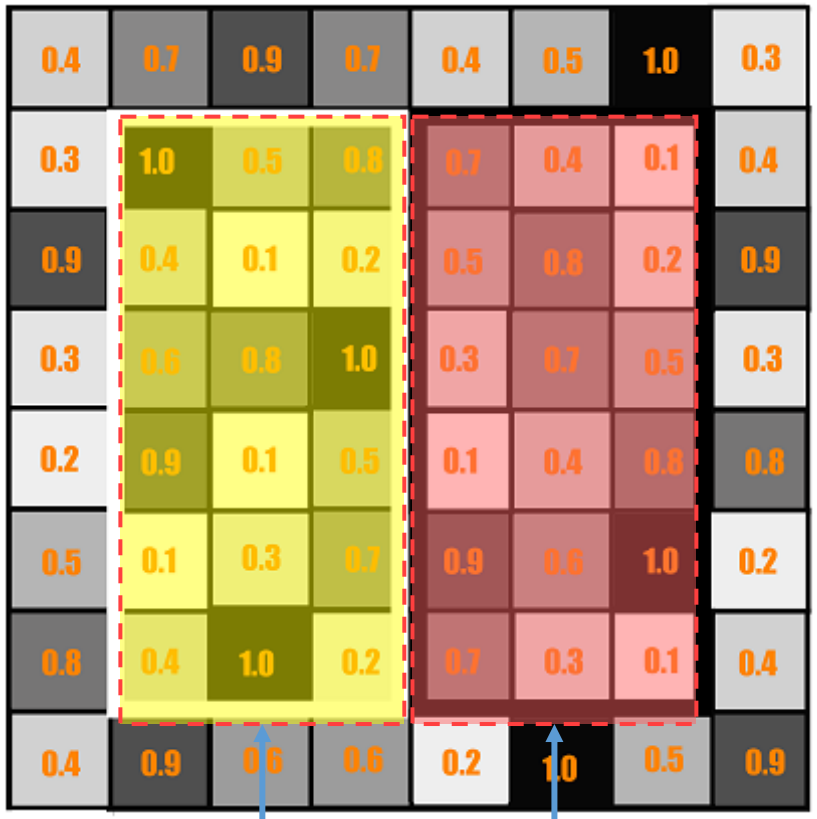




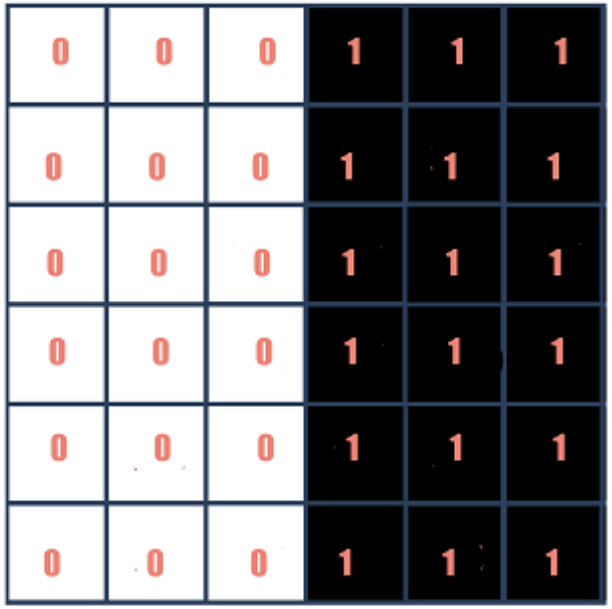
Viola-Jones uses 24*24 as base window size and calculates the above features all over the image shifting by 1 PX (Template + Feature).
There are over 160,000 possible feature combinations that can fit into a 24x24 pixel image, and over 250,000 for a 28x28 image

P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, Kauai, HI, USA, 2001, pp. I-I, doi: 10.1109/CVPR.2001.990517.

Viola Jones Detectors : Haar Feature



White region Dark region



=

SUM OF THE DARK PIXELS/NUMBER OF DARK PIXELS - SUM OF THE LIGHT PIXELS/NUMBER OF THE LIGHT PIXELS

(0.7 + 0.4 + 0.1 + 0.5 + 0.8 + 0.2 + 0.3 + 0.7 + 0.5 + 0.1 + 0.4 + 0.8 + 0.9 + 0.6 + 1.0 + 0.7 + 0.3 + 0.1)/18

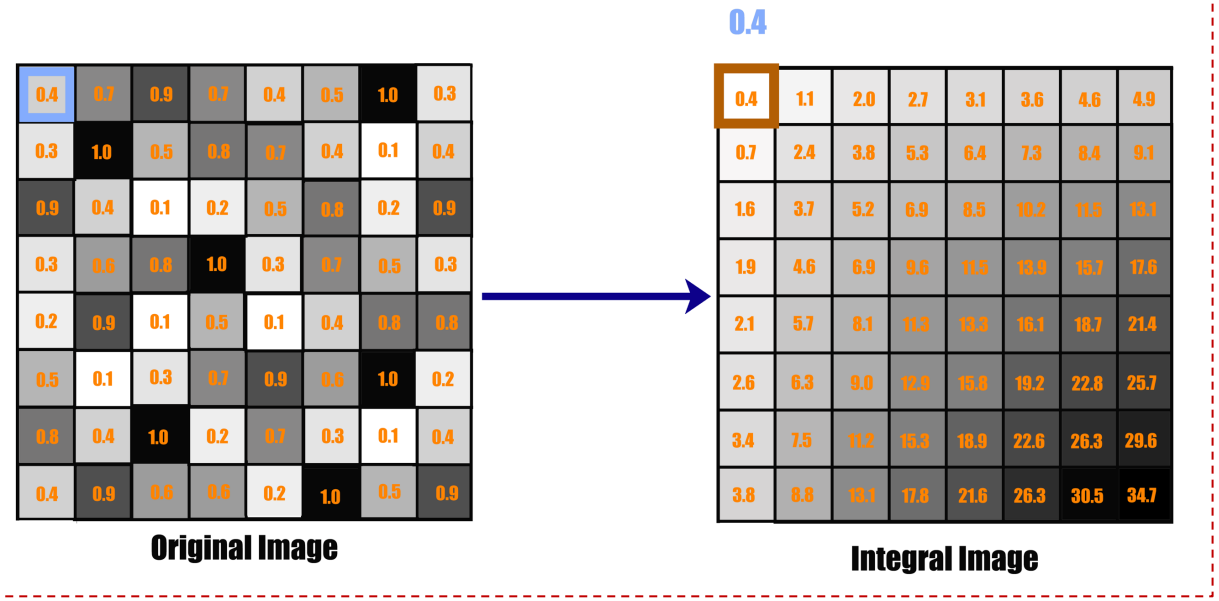
-

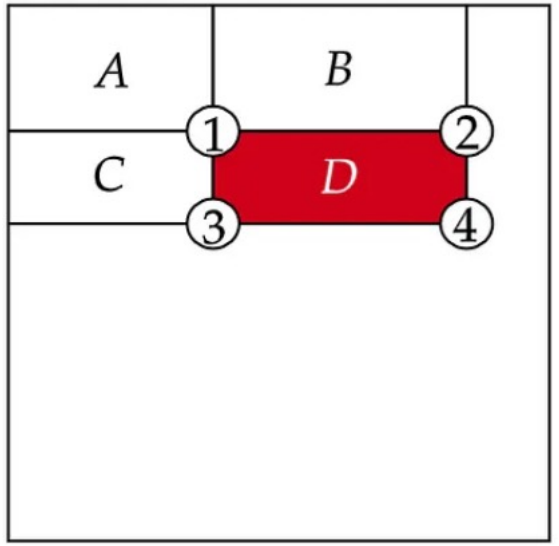
(1.0 + 0.5 + 0.8 + 0.4 + 0.1 + 0.2 + 0.6 + 0.8 + 1.0 + 0.9 + 0.1 + 0.5 + 0.1 + 0.3 + 0.7 +0.4 + 1.0 + 0.2)/18

0.51-0.53 = -0.02

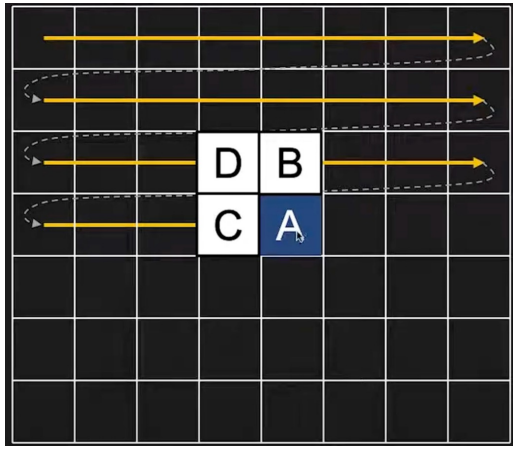
https://towardsdatascience.com/face-detection-with-haar-cascade-727f68dafd08

Viola Jones Detectors : Intergral Image





- The value of the integral image at point 1 is the sum of the pixels in rectangle A
- The value at point 2 is A + B
- The value at point 3 is A + C
- The value at point 4 is A + B + C + D.
- Therefore, the sum of pixels in region D can simply be computed as : 4+1−(2+3).



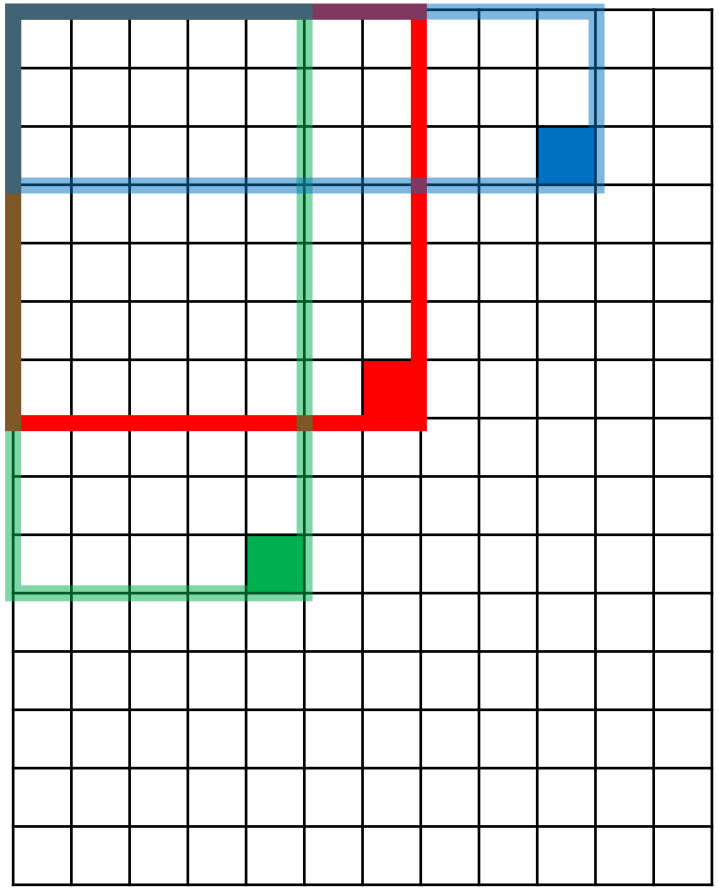
Raster Scanning

Viola Jones Detectors : Intergral Image

The trick is to compute an "integral image." Every pixel is the sum of its neighbors to the upper left.

Sequentially compute using:

$$I(x,y) = I(x,y) + I(x-1,y) + I(x,y-1) - I(x-1,y-1)$$



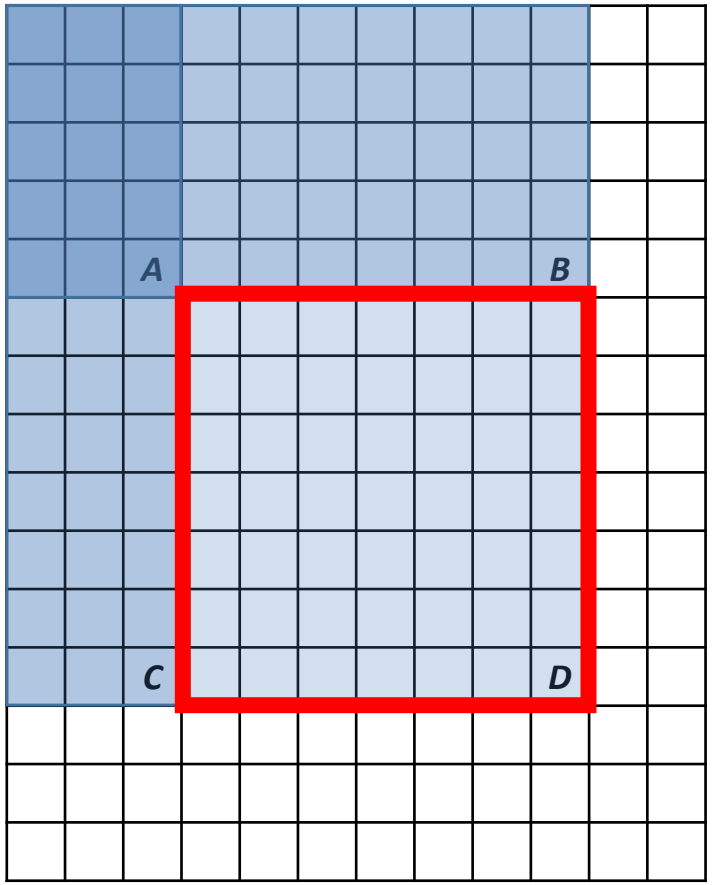
Viola Jones Detectors : Intergral Image

Solution is found using:

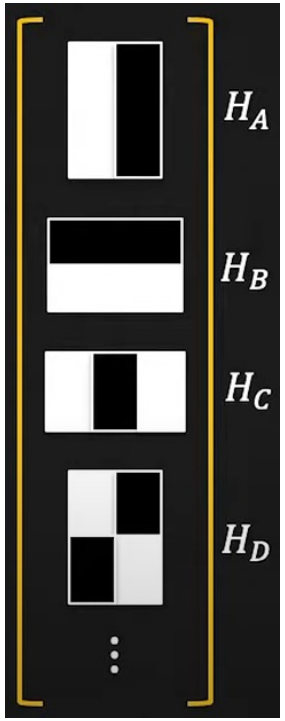
$$A + D - B - C$$

What if the position of the box lies between pixels?

Use bilinear interpolation.

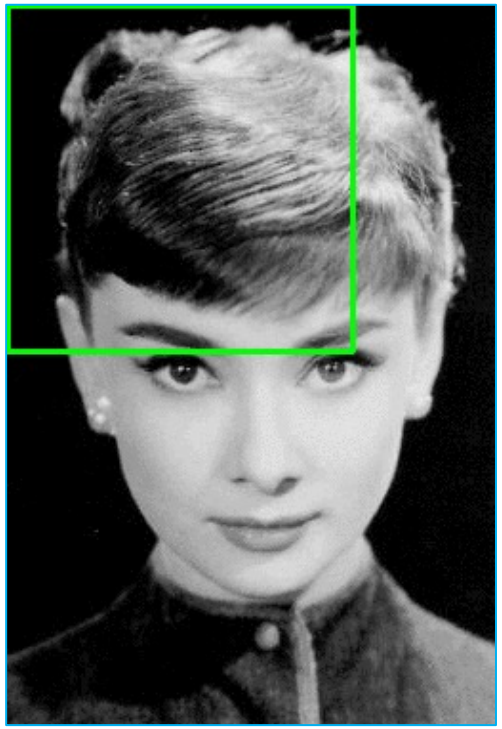


Viola Jones Detectors : Haar Feature



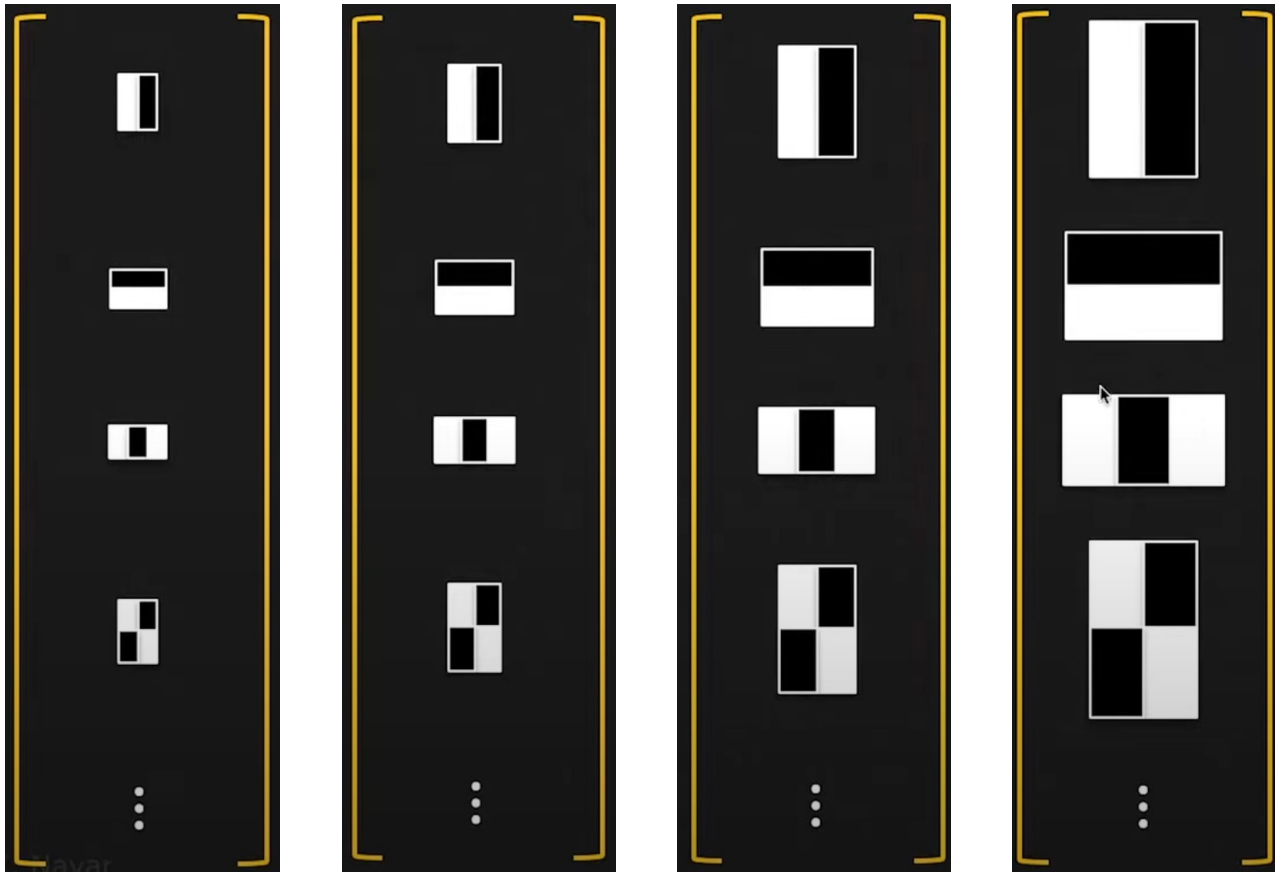
Haar Filters

Haar Features



AI VIETNAM
All-in-One Course

Haar Features for Multiple Scales



$$V_A[i,j] = \sum_m \sum_n I[m-i, n-j] H_A[m,n]$$

Compute Haar Features at different scales to detect faces of different sizes

As stated previously there can be approximately 160,000 + feature values within a detector at 24*24 base resolution which need to be calculated. But it is to be understood that only a few set of features will be useful among all these features to identify a face.

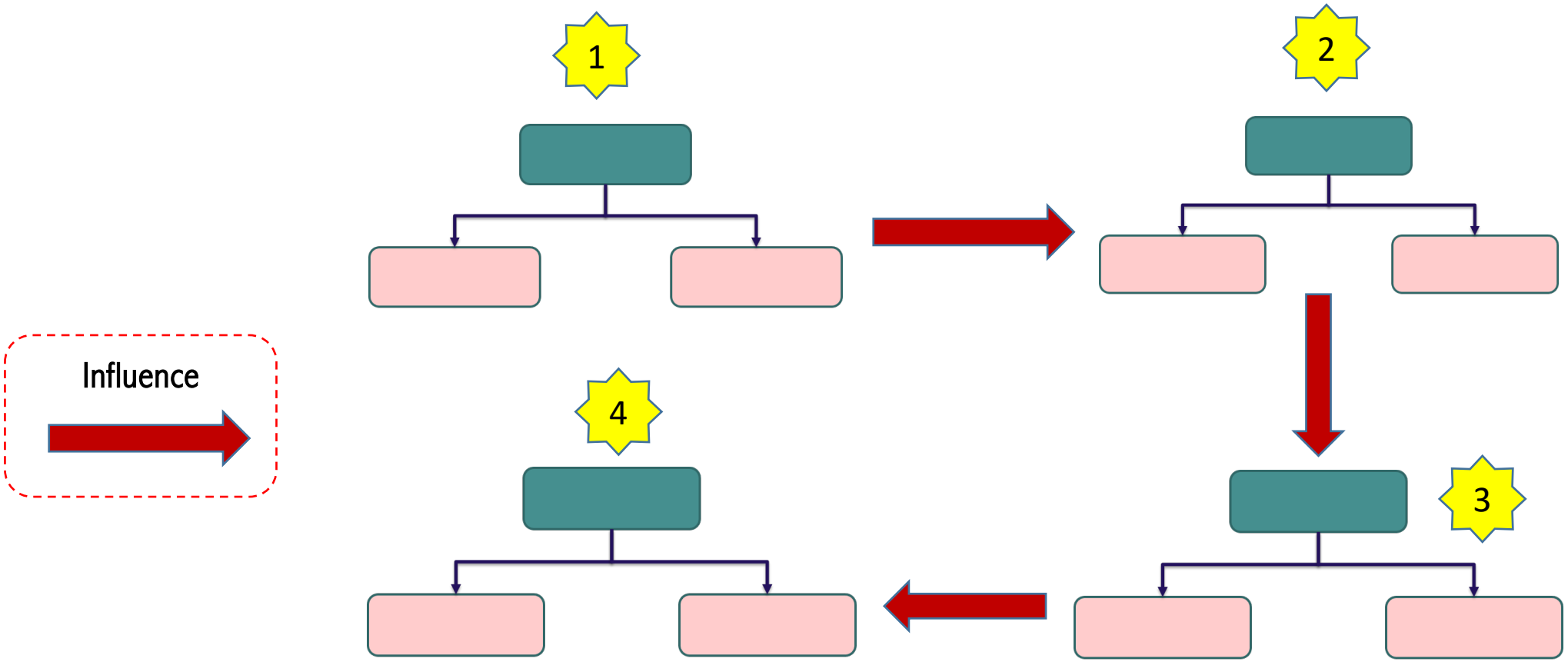

Relevant feature


Irrelevant feature

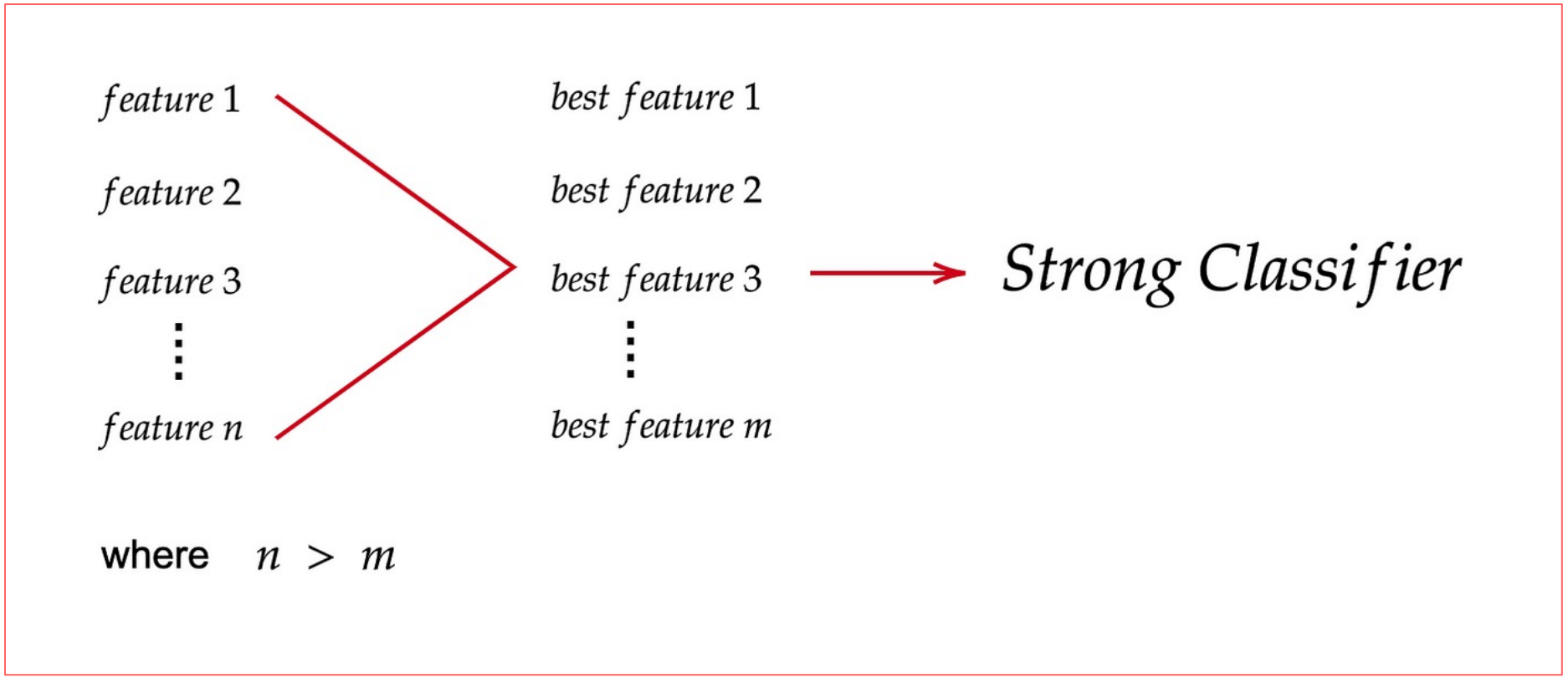
Feature detecting a vertical edge is useful detecting a nose but irrelevant detecting a lip.

Can we create a good classifier using just a small subset of all possible features?

AdaBoost: FOREST OF STUMP

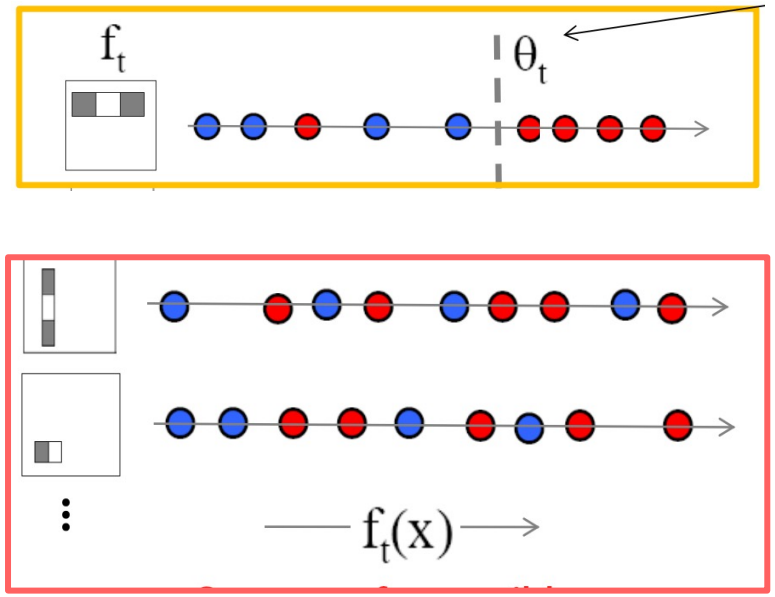


Adaboost for Face Detection



The goal of using the AdaBoost algorithm is to extract the best features from n features

AdaBoost for Feature+Classifier Selection



θ^t is a threshold for classifier h^t

Resulting weak classifier:

$$h_t(x) = \begin{cases} +1 & \text{if } f_t(x) > \theta_t \\ 0 & \text{otherwise} \end{cases}$$

For next round, reweight the examples according to errors, choose another filter/threshold combo.

- Each weak classifier works on exactly one rectangle feature.
- Each weak classifier has 3 associated variables
  1. its threshold θ
  2. its polarity p
  3. its weight α

  h(x) = 1 if p*f(x) < pθ, else 0

  used for the combination step

  The code does not actually compute h.
- The polarity can be 0 or 1
- The weak classifier computes its one feature f
  - When the polarity is 1, we want f > θ for face
  - When the polarity is 0, we want f < θ for face
- The weight will be used in the final classification by AdaBoost.

Want to select the single rectangle feature and threshold that best separates positive (faces) and negative (non-faces) training examples, in terms of weighted error

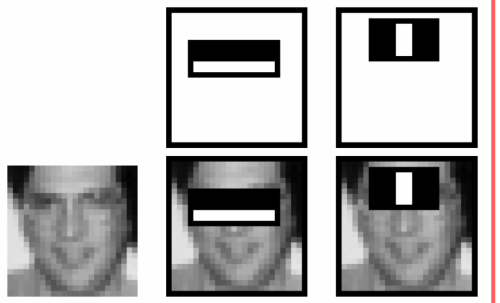
1. Initialized weights for each training example
2. Normalized all the weights
3. Then we selected the best weak classifier based on the weighted error of the training examples
4. Then updated the weights according to the error of the chosen best weak classifier
5. Repeated steps 2–4 N times, where N is the desired number of weak classifiers

AdaBoost for Feature+Classifier selection

- Given example images $(x_1, y_1), \ldots, (x_n, y_n)$ where $y_i = 0, 1$ for negative and positive examples respectively.
- Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0, 1$ respectively, where m and l are the number of negatives and positives respectively.
- For $t = 1, \ldots, T$:
  1. Normalize the weights,
  $$w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$$
  so that w_t is a probability distribution.
  2. For each feature, j, train a classifier h_j which is restricted to using a single feature. The error is evaluated with respect to w_t, $\epsilon_j = \sum_i w_i \, |h_j(x_i) - y_i|$.
  3. Choose the classifier, h_t, with the lowest error ϵ_t.
  4. Update the weights:
  $$w_{t+1,i} = w_{t,i} \beta_t^{1-e_i}$$
  where $e_i = 0$ if example x_i is classified correctly, $e_i = 1$ otherwise, and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.
- The final strong classifier is:
$$h(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$$
where $\alpha_t = \log \frac{1}{\beta_t}$

The AdaBoost algorithm for classifier learning. Each round t of boosting selects one feature from the 180,000 potential feature

First two features selected by boosting:



This feature combination can yield 100% detection rate and 50% false positive rate

A 200-feature classifier can yield 95% detection rate and a false positive rate of 1 in 14084

AdaBoost Algorithm modified by Viola Jones

- Given example images $(x_1, y_1), \ldots, (x_n, y_n)$ where $y_i = 0, 1$ for negative and positive examples respectively.
- Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0, 1$ respectively, where m and l are the number of negatives and positives respectively.
- For $t = 1, \ldots, T$:

1. Normalize the weights,

$$w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$$

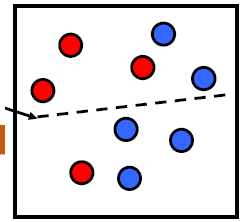
so that w_t is a probability distribution.

2. For each feature, j, train a classifier h_j which is restricted to using a single feature. The error is evaluated with respect to w_t, $\epsilon_j = \sum_i w_i \left| h_j(x_i) - y_i \right|$. sum over training samples
3. Choose the classifier, h_t, with the lowest error ϵ_t.
4. Update the weights:

$$w_{t+1,i} = w_{t,i} \beta_t^{1-e_i}$$

where $e_i = 0$ if example x_i is classified correctly, $e_i = 1$ otherwise, and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.

NOTE: Our code uses equal weights for all samples



$\{x_1, \ldots x_n\}$

For T rounds: meaning we will construct T weak classifiers

Normalize weights

Find the best threshold and polarity for each feature, and return error.

Re-weight the examples:
Incorrectly classified -> more weight
Correctly classified -> less weight

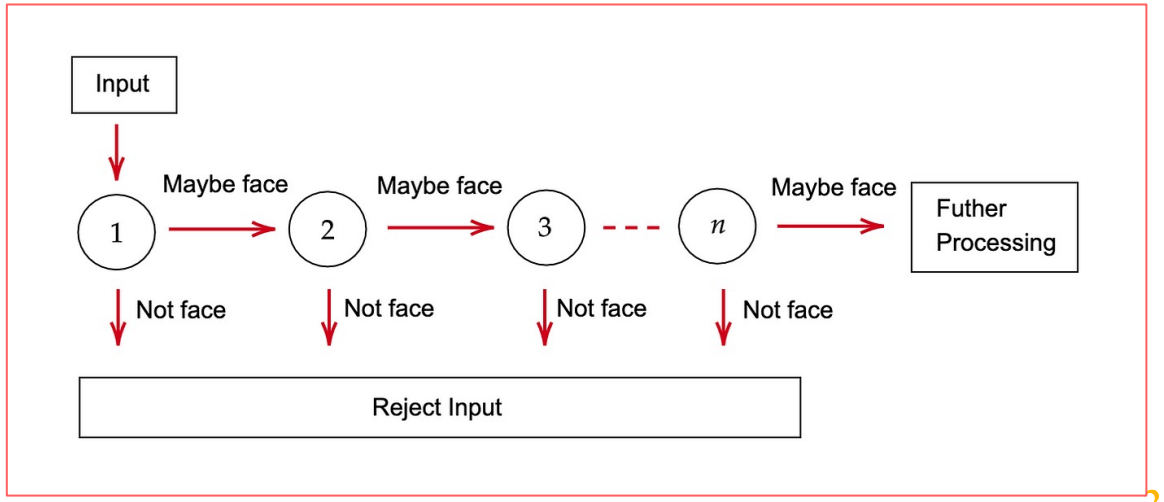
The Cascade Classifier

In an image, most of the image is a non-face region

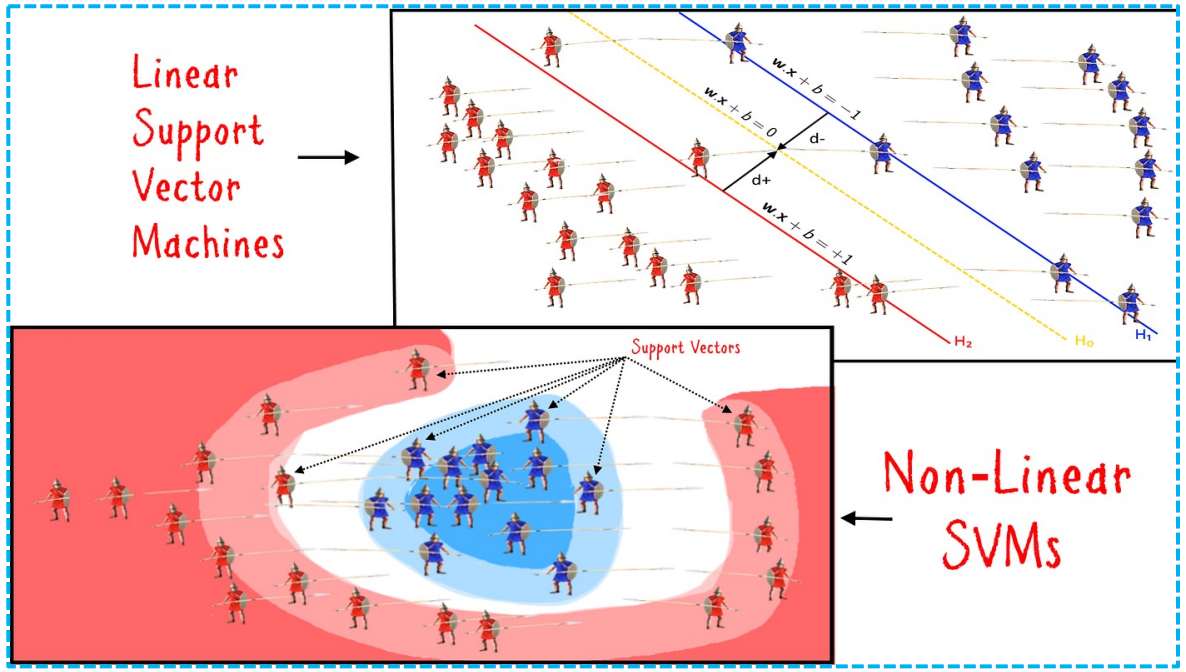
The key idea is to reject sub-windows that do not contain faces while identifying regions that do. Since the task is to identify properly the face, we want to minimize the false negative rate, i.e the sub-windows that contain a face and have not been identified as such.

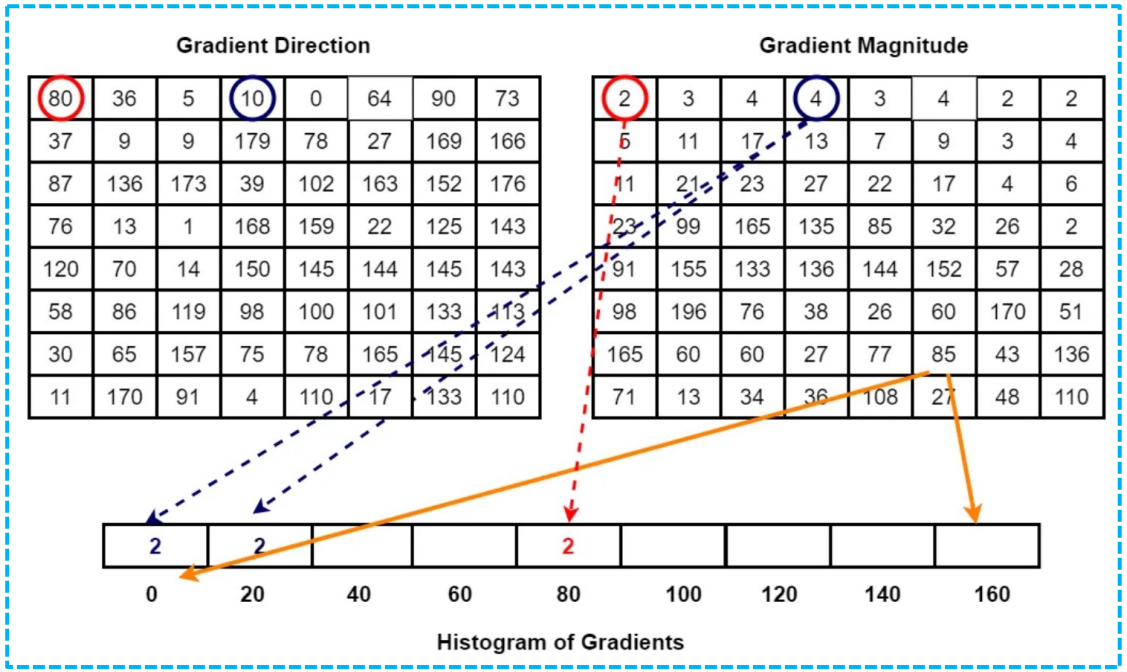
A series of classifiers are applied to every sub-window. These classifiers are simple decision trees :

- if the first classifier is positive, we move on to the second
- if the second classifier is positive, we move on to the third.
- Any negative result at some point leads to a rejection of the sub-window as potentially containing a face

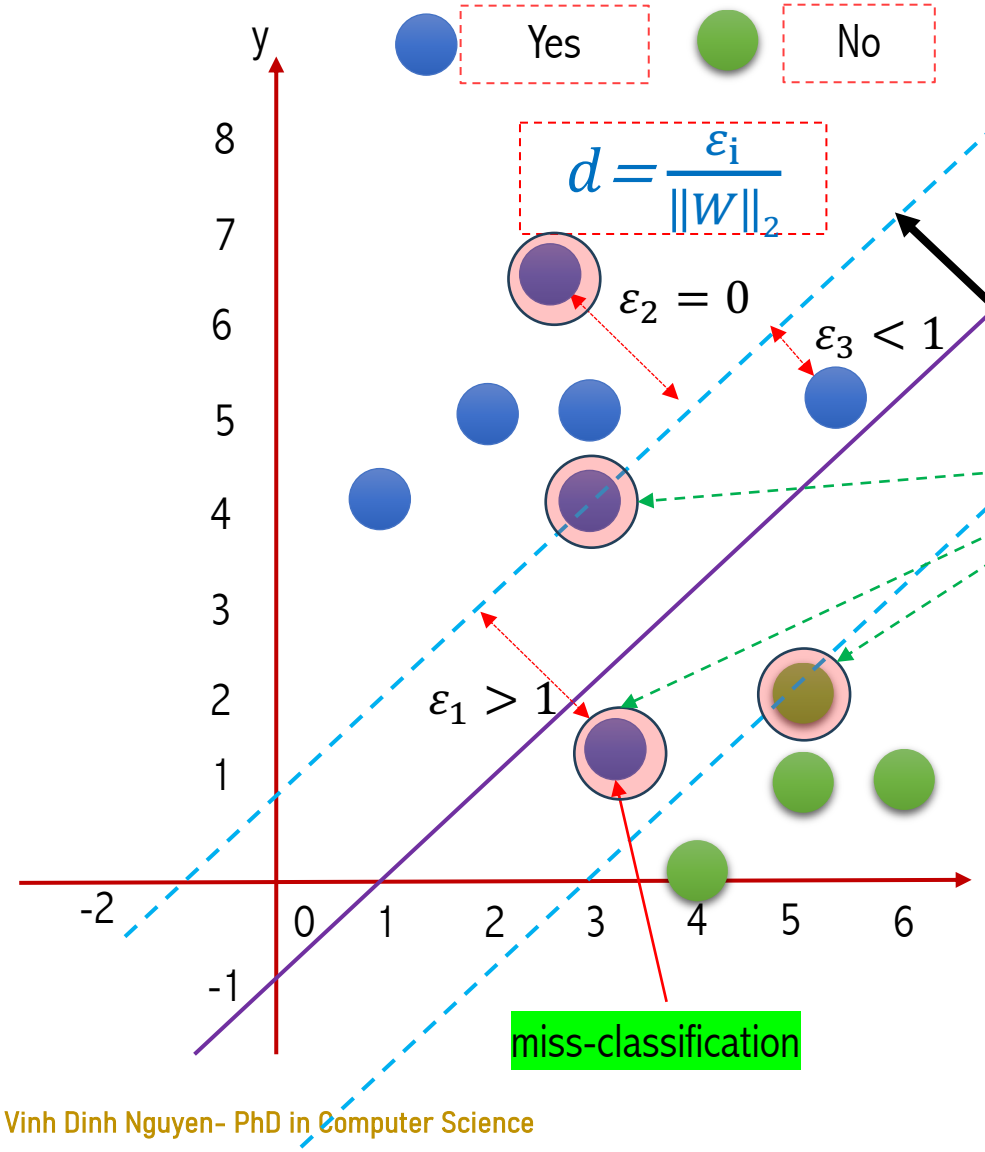


SVM and HOG for Object Detection





Support Vector Machine



$w^T X + b = 1$

$w^T X + b = 0$

$w^T X + b = -1$

$$d = \frac{|2|}{\sqrt{A^2+B^2}} = \frac{2}{\|W\|_2}$$

Support Vector

Slack variable in SVM

$$\text{Max } \frac{2}{\|W\|_2} \text{ such that } Y * (w^T X + b) \geq 1$$

$$\min \frac{1}{2} \|W\|_2^2 + C \sum_i^N \varepsilon_i \text{ such that } Y * (w^T X + b) \geq 1 - \varepsilon_i$$

Allow for miss classification

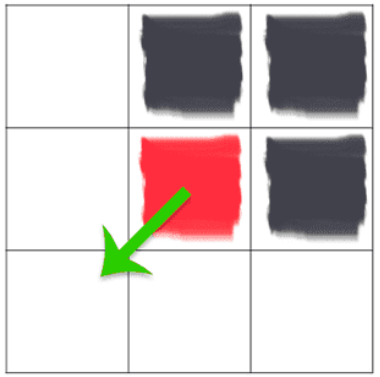
ε_i is a distance measure of the data points from their corresponding blue line.

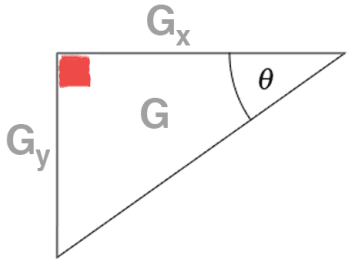
C controls how much weight should set on the misclassification data

Histogram of Oriented Gradient

An image gradient is a directional change in the intensity or color in an image. The gradient of the image is one of the fundamental building blocks in image processing. For example, the Canny edge detector uses image gradient for edge detection.

255	0	0
255		0
255	255	255





$$\theta = \arctan2(G_y, G_x) \times \left(\frac{180}{\pi}\right)$$

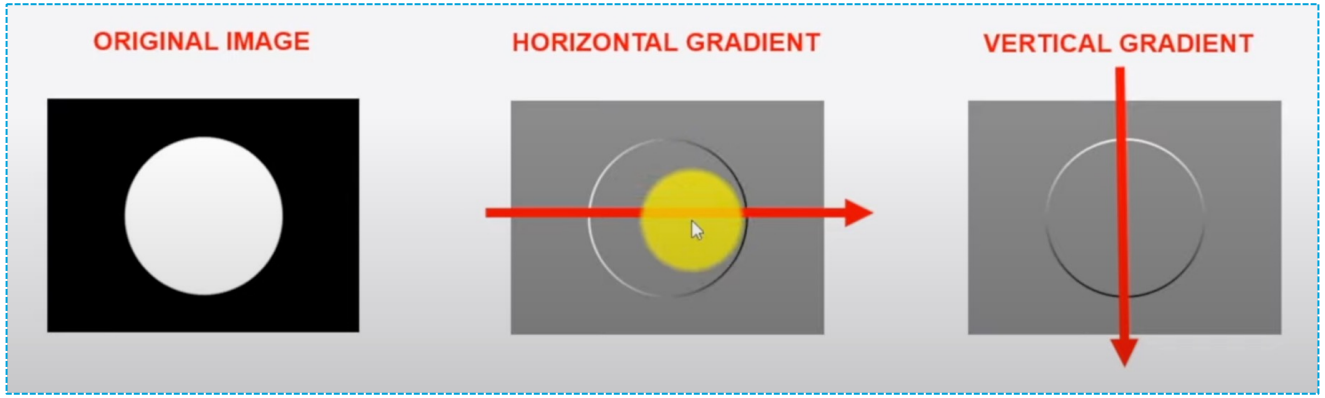
$$G_x = I(x+1, y) - I(x-1, y)$$

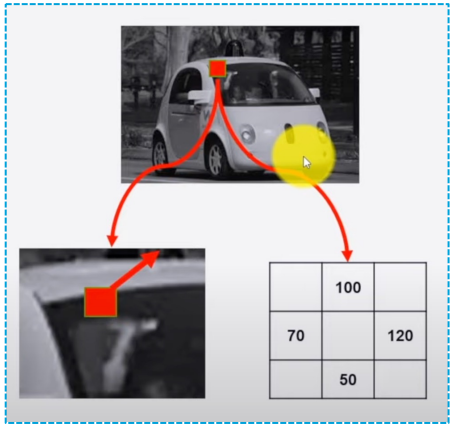
$$G_y = I(x, y+1) - I(x, y-1)$$

$$G = \sqrt{G_x^2 + G_y^2}$$

Histogram of Oriented Gradient

An image gradient is a directional change in the intensity or color in an image. The gradient of the image is one of the fundamental building blocks in image processing. For example, the Canny edge detector uses image gradient for edge detection.





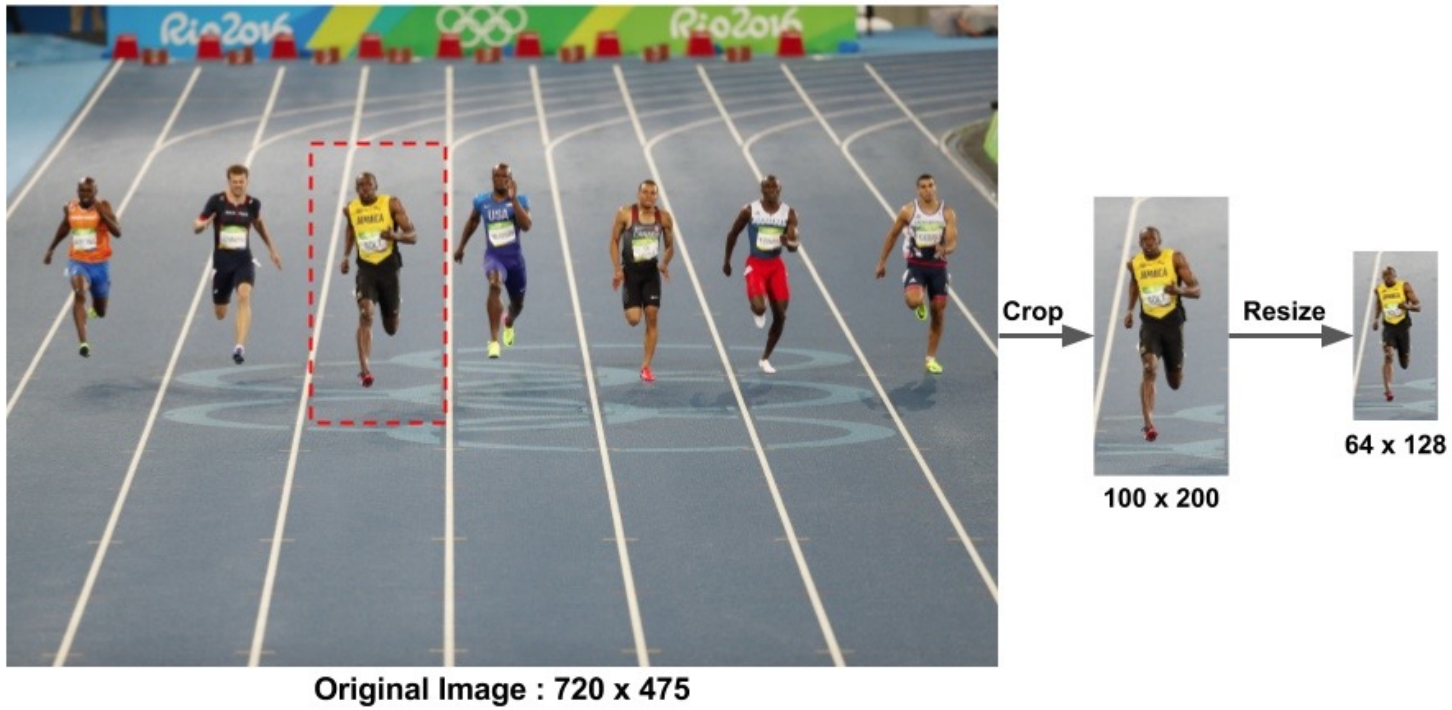
-1	0	1

-1
0
1

Gradient Magnitude = $\sqrt{(50)^2 + (50)^2} = 70.1$
Gradient Angle = $\tan^{-1}(50/50) = 45°$

Histogram of Oriented Gradient

Step 1: preprocessing and compute gradient

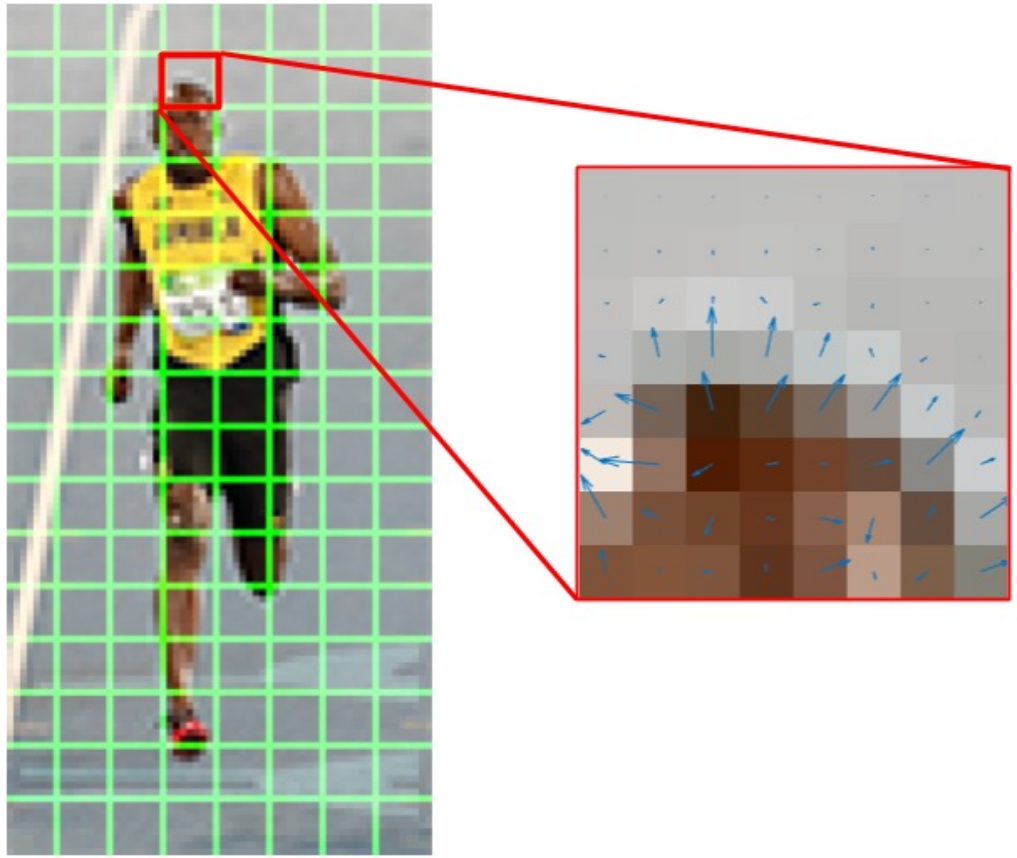


Left : Absolute value of x-gradient. Center : Absolute value of y-gradient. Right : Magnitude of gradient.

In the case of the HOG feature descriptor, the input image is of size 64 x 128 x 3 = 24576 and the output feature vector is of length 3780

Histogram of Oriented Gradient

Step 2: Calculate Histogram of Gradients in 8×8 cells



2	3	4	4	3	4	2	2
5	11	17	13	7	9	3	4
11	21	23	27	22	17	4	6
23	99	165	135	85	32	26	2
91	155	133	136	144	152	57	28
98	196	76	38	26	60	170	51
165	60	60	27	77	85	43	136
71	13	34	23	108	27	48	110

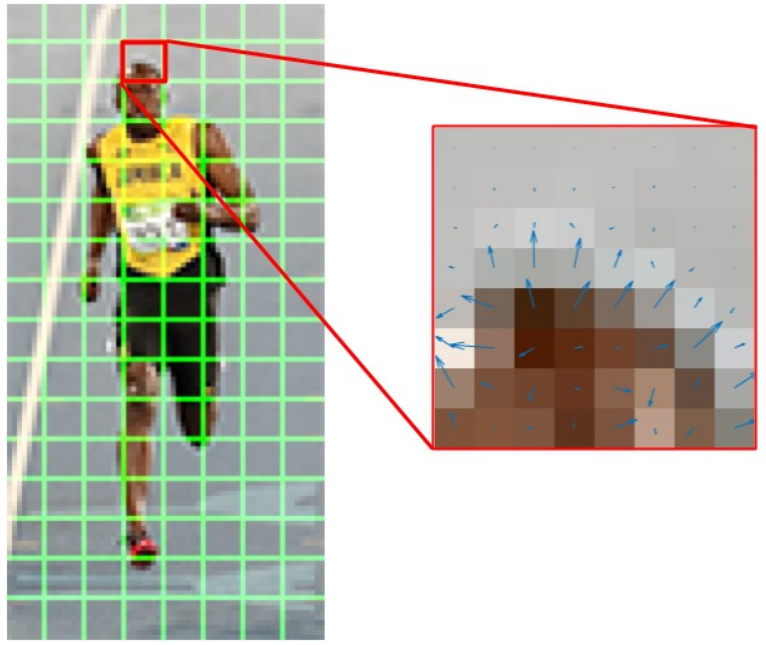
Gradient Magnitude

80	36	5	10	0	64	90	73
37	9	9	179	78	27	169	166
87	136	173	39	102	163	152	176
76	13	1	168	159	22	125	143
120	70	14	150	145	144	145	143
58	86	119	98	100	101	133	113
30	65	157	75	78	165	145	124
11	170	91	4	110	17	133	110

Gradient Direction

Histogram of Oriented Gradient

Step 3: Compute Histogram of Oriented Gradient

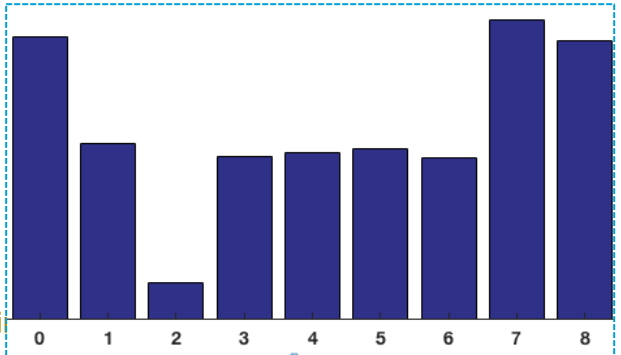


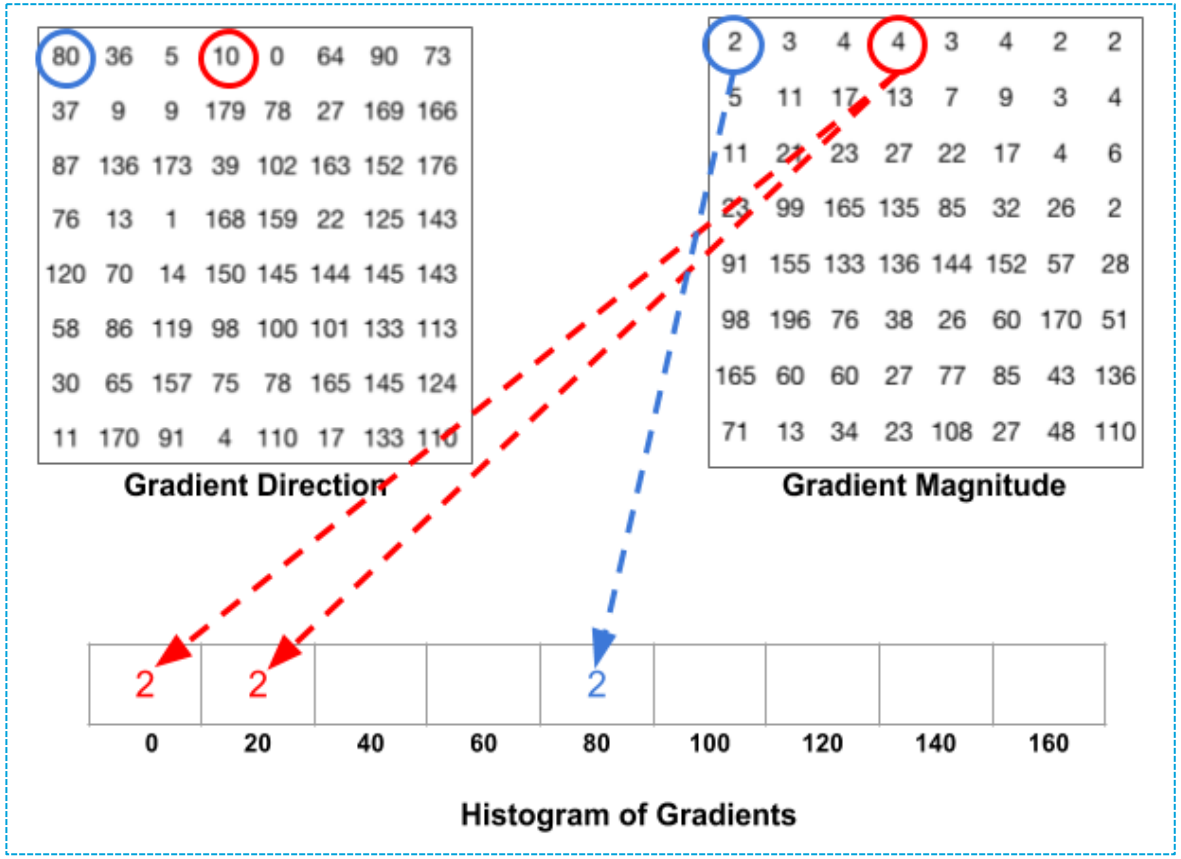
2	3	4	4	3	4	2	2
5	11	17	13	7	9	3	4
11	21	23	27	22	17	4	6
23	99	165	135	85	32	26	2
91	155	133	136	144	152	57	28
98	196	76	38	26	60	170	51
165	60	60	27	77	85	43	136
71	13	34	23	108	27	48	110

Gradient Magnitude

80	36	5	10	0	64	90	73
37	9	9	179	78	27	169	166
87	136	173	39	102	163	152	176
76	13	1	168	159	22	125	143
120	70	14	150	145	144	145	143
58	86	119	98	100	101	133	113
30	65	157	75	78	165	145	124
11	170	91	4	110	17	133	110

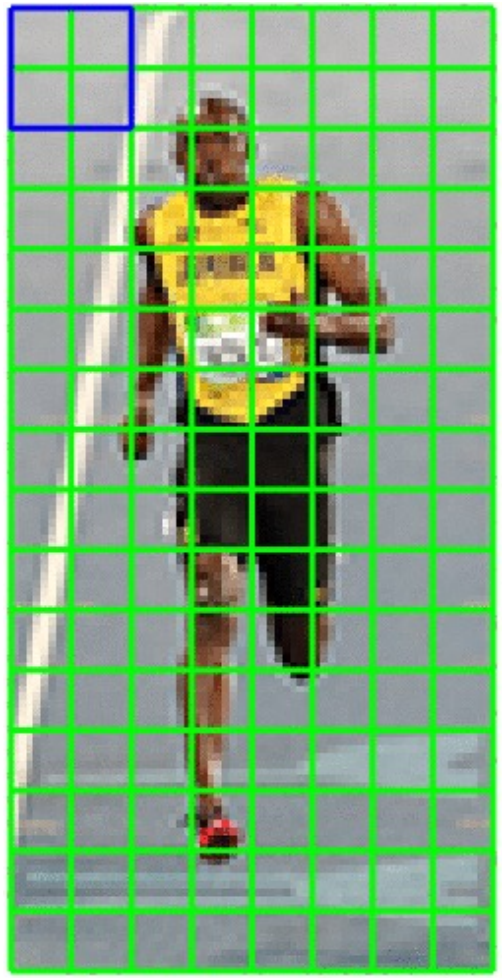
Gradient Direction





Histogram of Oriented Gradient

Step 4: 16×16 Block Normalization



Gradients for highlighted Bock (having 36 features):
$V = [a_1, a_2, a_3, .., a_{36}]$

Root of the sum of squares:

$$k = \sqrt{a_1^2 + a_2^2 + a_3^2 + \ldots + a_{36}^2}$$

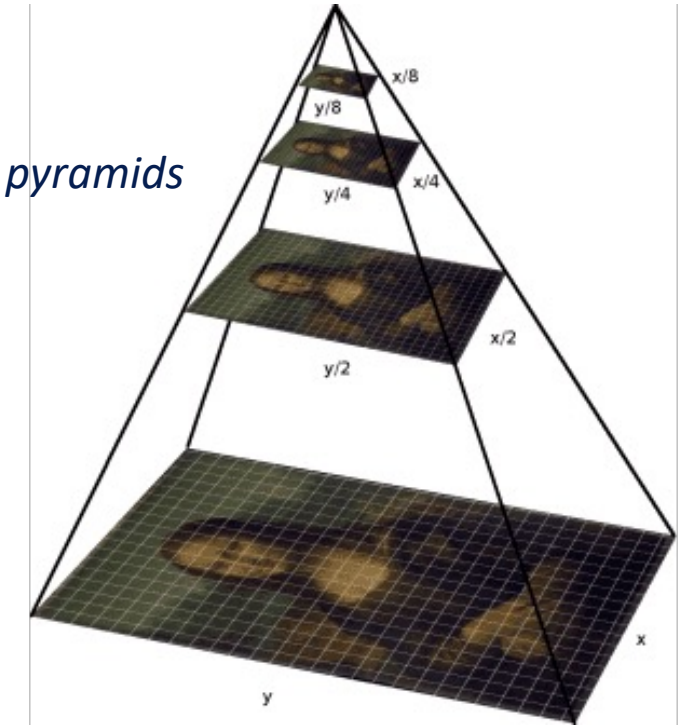
Normalized vector $= \left(\frac{a_1}{k}, \frac{a_2}{k}, \frac{a_3}{k}, \ldots, \frac{a_{36}}{k}\right)$

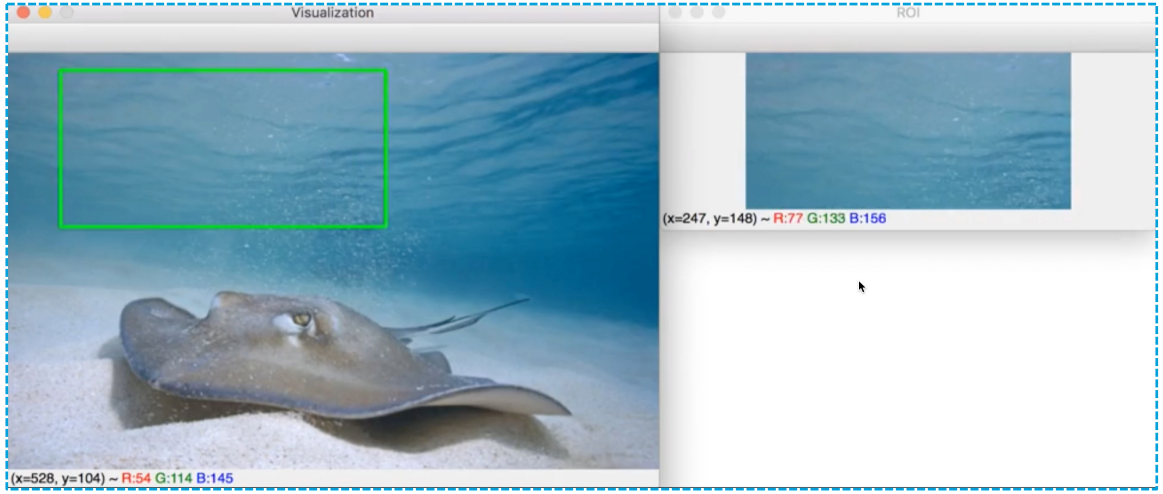
Normalized vectors created for all 105 blocks, having 36 features each image Feature Descriptor: 1 x 3780 matrix

SVM + HOG

Advanced techniques:

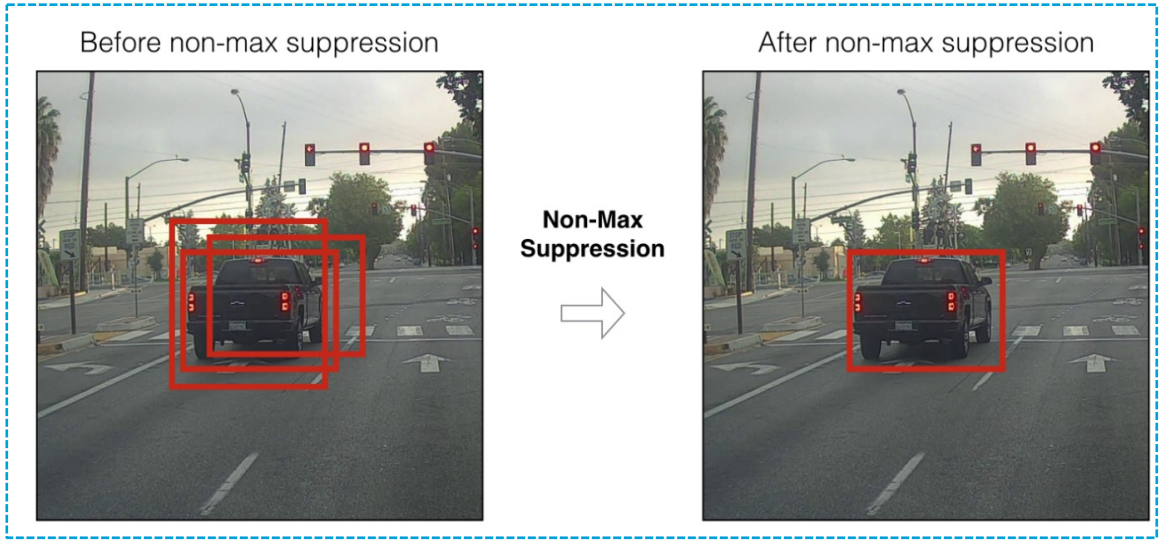
Image pyramids





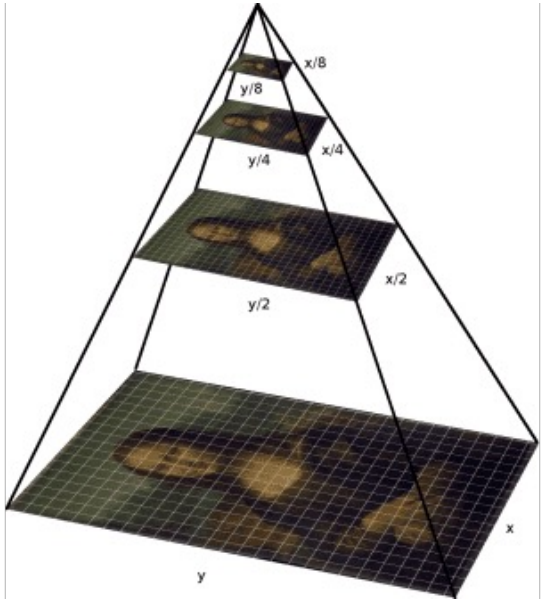
Sliding Window

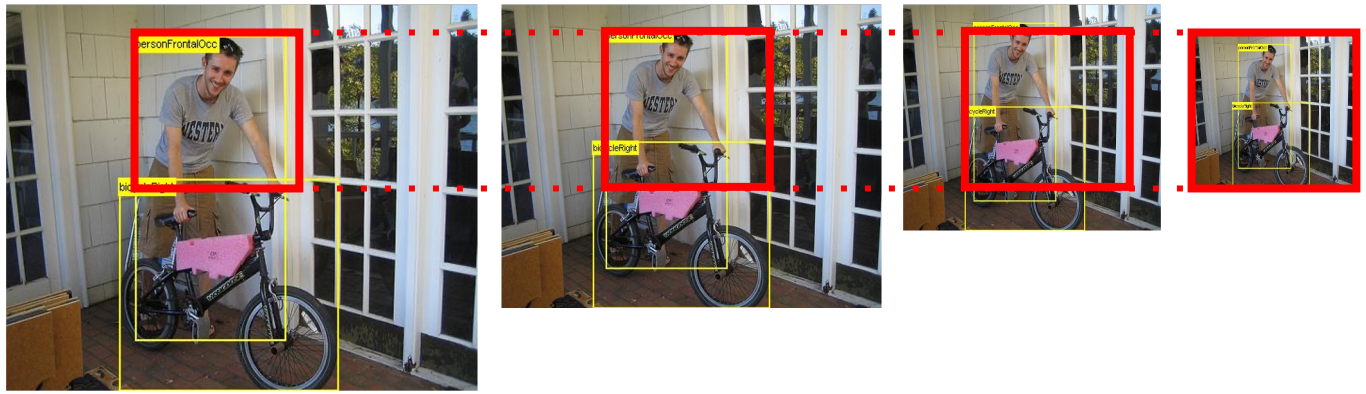
non-maxima suppression



SVM + HOG

Image pyramids





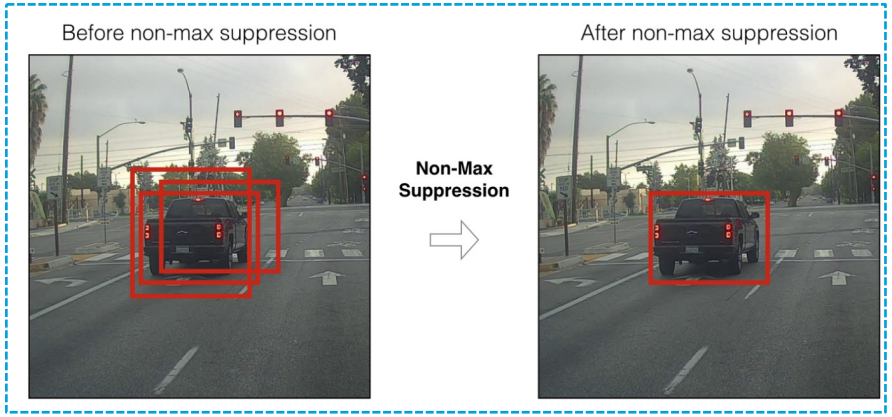
Smaller objects

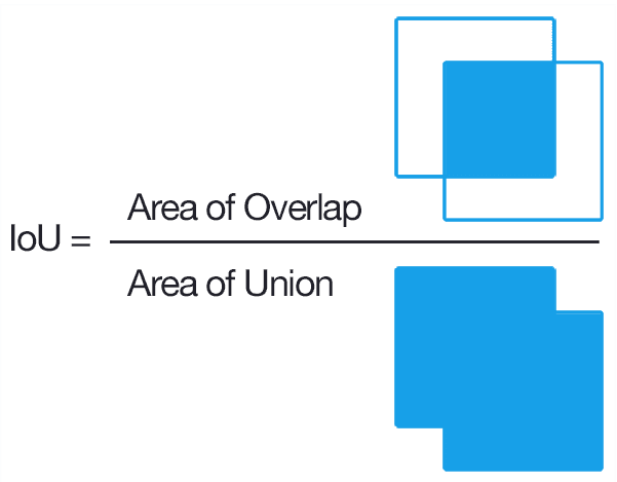
Sliding Window – Location
Image Pyramid - Scale

Larger objects

SVM + HOG

Non-maxima suppression





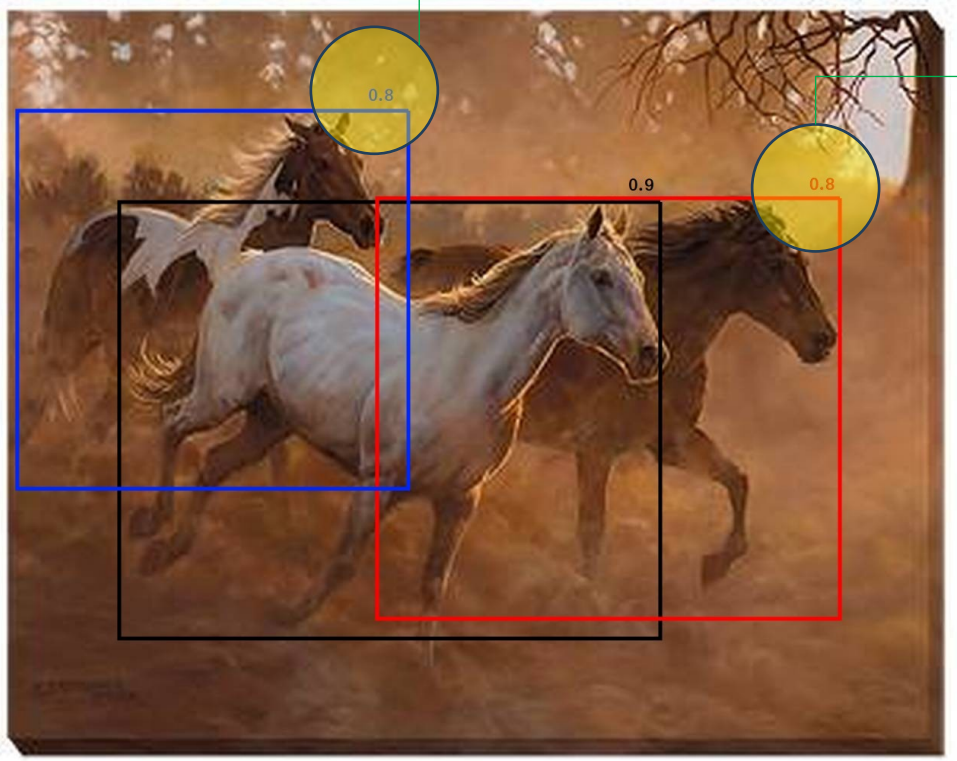
Algorithm 1 Non-Max Suppression

```
procedure NMS(B,c)
    B_nms ← ∅                                   Initialize empty set
    for b_i ∈ B do                              Iterate over all the boxes
        discard ← False                         Take boolean variable and set it as false. This variable indicates whether b(i) should be kept or discarded
        for b_j ∈ B do                          Start another loop to compare with b(i)
            if same(b_i, b_j) > λ_nms then      If both boxes having same IOU
                if score(c, b_j) > score(c, b_i) then
                    discard ← True              Compare the scores. If score of b(i) is less than that of b(j), b(i) should be discarded, so set the flag to True.
        if not discard then                     Once b(i) is compared with all other boxes and still the discarded flag is False, then b(i) should be considered. So add it to the final list.
            B_nms ← B_nms ∪ b_i
    return B_nms                                Do the same procedure for remaining boxes and return the final list
```

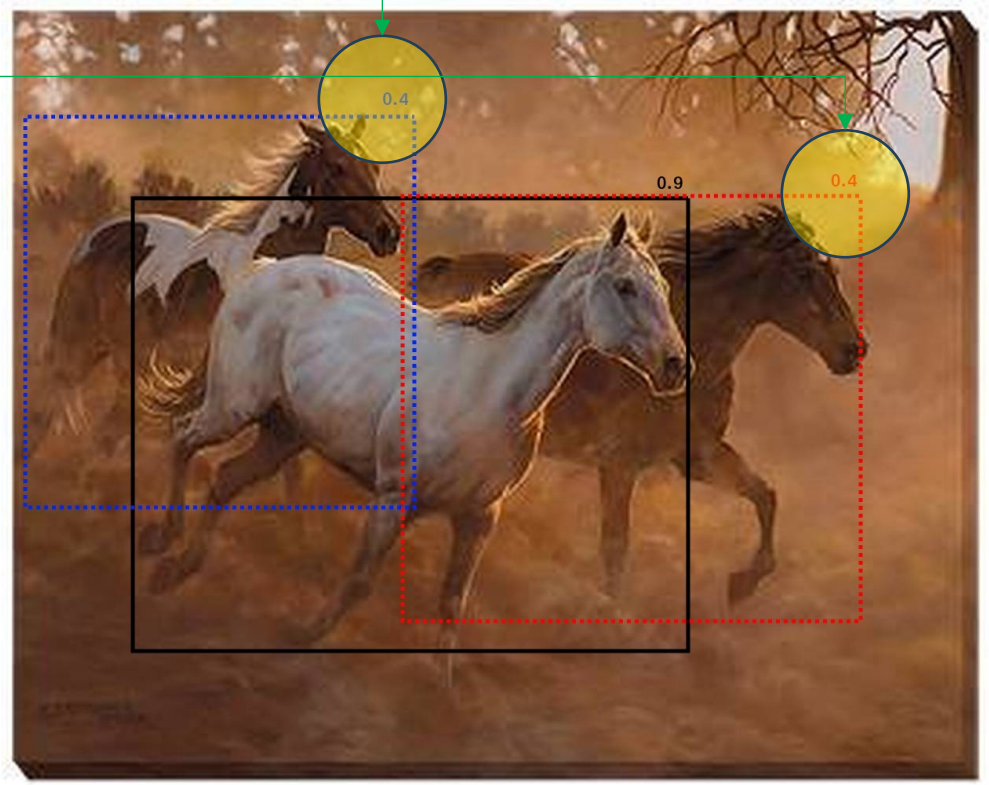
SVM + HOG

Non-maxima suppression

Hard NMS



Soft NMS



SVM + HOG

Non-maxima suppression

Hard NMS

Algorithm 1 Non-Max Suppression

```
procedure NMS(B,c)
    B_nms ← ∅                                   Initialize empty set
    for b_i ∈ B do                              Iterate over all the boxes
        discard ← False                         Take boolean variable and set it as false. This variable indicates whether b(i) should be kept or discarded
        for b_j ∈ B do                          Start another loop to compare with b(i)
            if same(b_i, b_j) > λ_nms then      If both boxes having same IOU
                if score(c, b_j) > score(c, b_i) then
                    discard ← True              Compare the scores. If score of b(i) is less than that of b(j), b(i) should be discarded, so set the flag to True.
        if not discard then                     Once b(i) is compared with all other boxes and still the discarded flag is False, then b(i) should be considered. So add it to the final list.
            B_nms ← B_nms ∪ b_i
    return B_nms                                Do the same procedure for remaining boxes and return the final list
```

Soft NMS

Input : $\mathcal{B} = \{b_1, .., b_N\}, \mathcal{S} = \{s_1, .., s_N\}, N_t$
$\mathcal{B}$ is the list of initial detection boxes
$\mathcal{S}$ contains corresponding detection scores
N_t is the NMS threshold

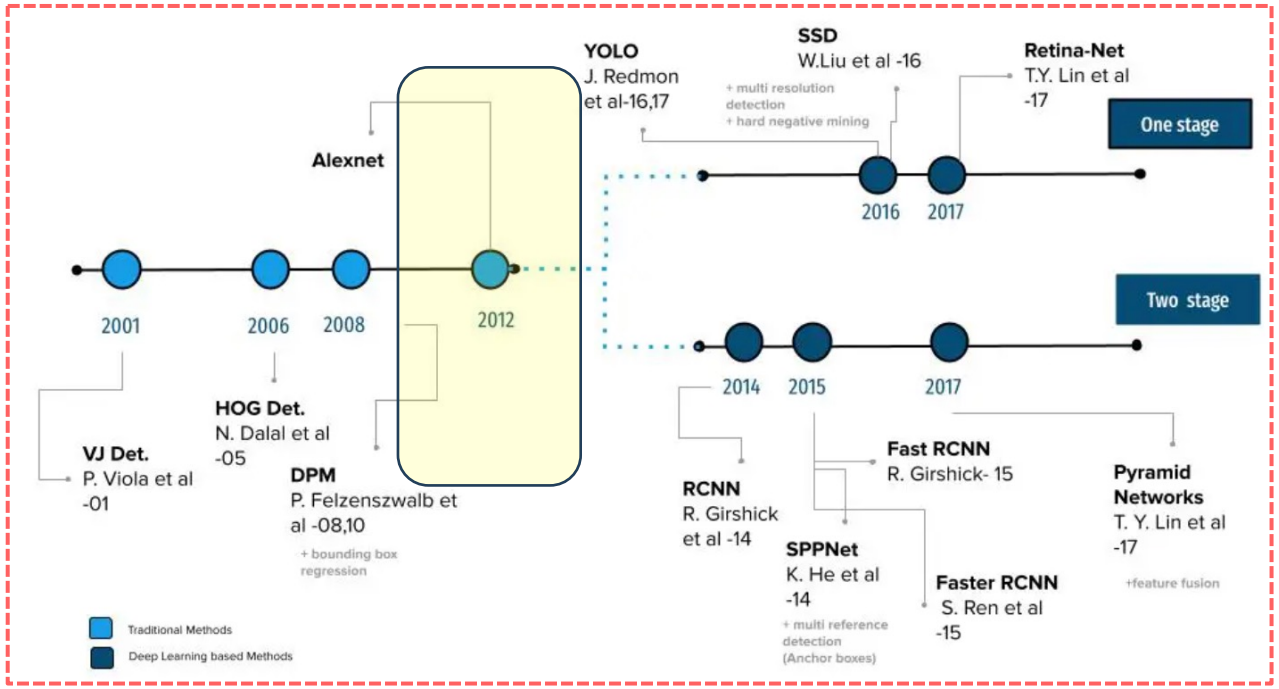
begin
  $\mathcal{D} \leftarrow \{\}$
  while $\mathcal{B} \neq$ *empty* **do**
    $m \leftarrow$ argmax $\mathcal{S}$
    $\mathcal{M} \leftarrow b_m$
    $\mathcal{D} \leftarrow \mathcal{D} \bigcup \mathcal{M}; \mathcal{B} \leftarrow \mathcal{B} - \mathcal{M}$
    for b_i *in* $\mathcal{B}$ **do**
      if $iou(\mathcal{M}, b_i) \geq N_t$ **then**
        $\mathcal{B} \leftarrow \mathcal{B} - b_i; \mathcal{S} \leftarrow \mathcal{S} - s_i$
      end NMS
      $s_i \leftarrow s_i f(iou(\mathcal{M}, b_i))$ Soft-NMS
    end
  end
  return $\mathcal{D}, \mathcal{S}$
end

Traditional Detectors: Summary

Most traditional object detection algorithms like Viola–Jones, and Histogram of Oriented Gradients (HOG)are relied on extracting handcrafted features like edges, corners, gradients from the image and classical machine learning algorithms.

The traditional computer vision approaches were in the game until 2010.

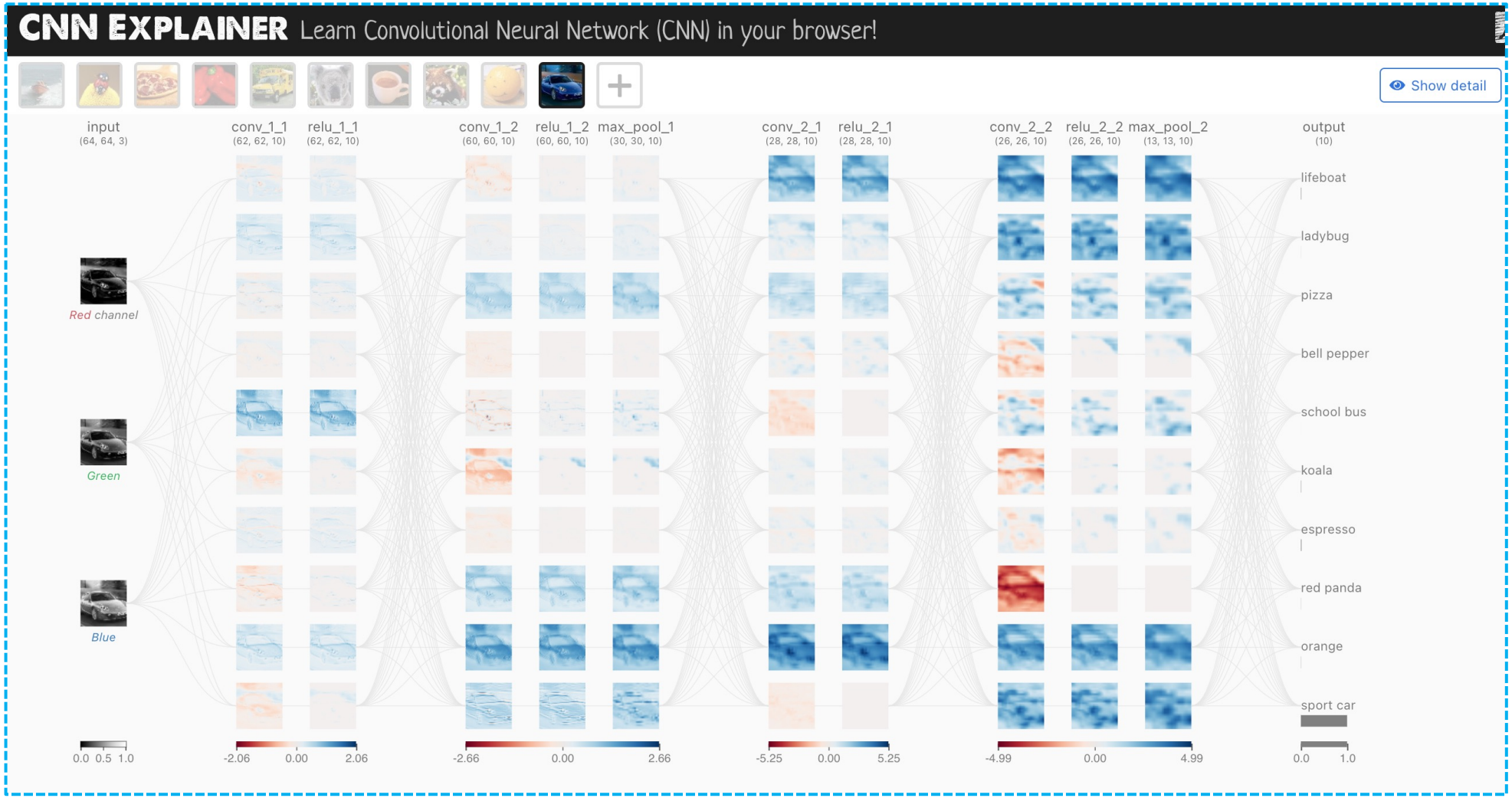
From 2012, a new era of convolutional neural networks started when AlexNet (an image classification network) won the ImageNet Visual Recognition challenge 2012. The accuracy of 84.7% as compared to the second-best with an accuracy of 73.8%.



Outline

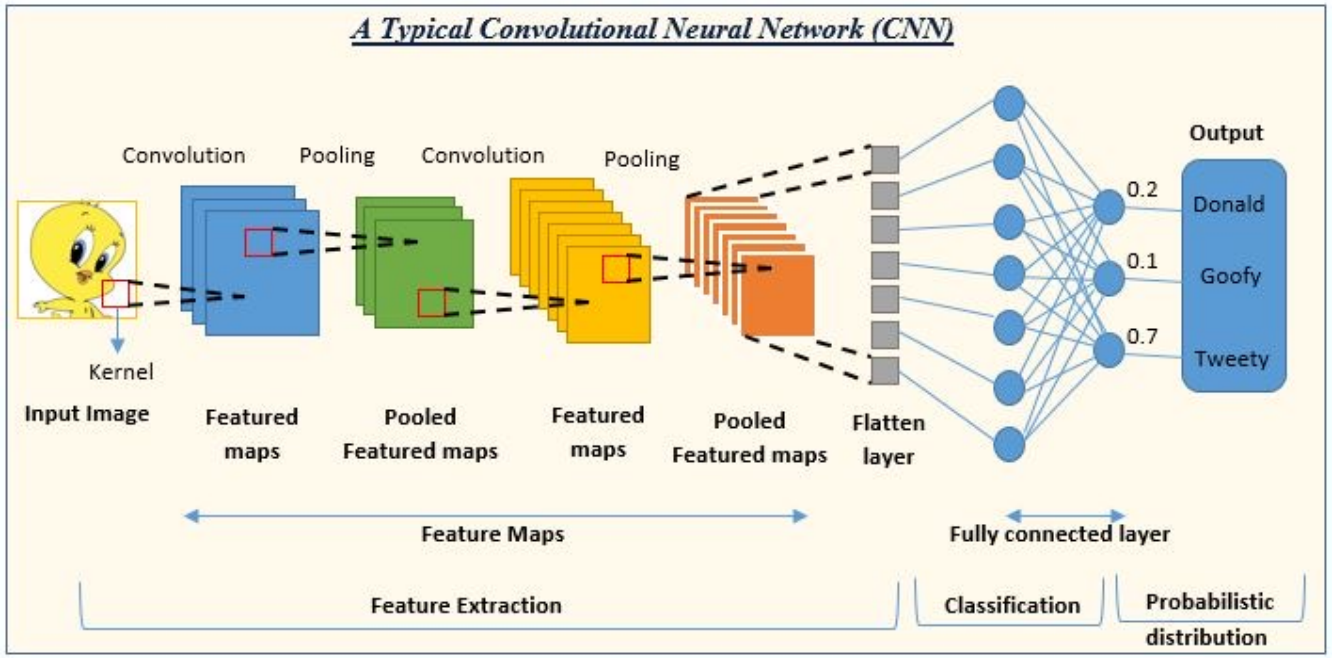
- Introduction to Object Detection
- Applications of Object Detection
- Object Detection Milestones
- Traditional Object Detectors
- CNNs for Image Classification
- Image Classifier to Object Detectors with CNN
- CNN Limitations and Spatial Outputs
- Assignment

Convolutional Neural Network



https://poloclub.github.io/cnn-explainer/

Convolutional Neural Network



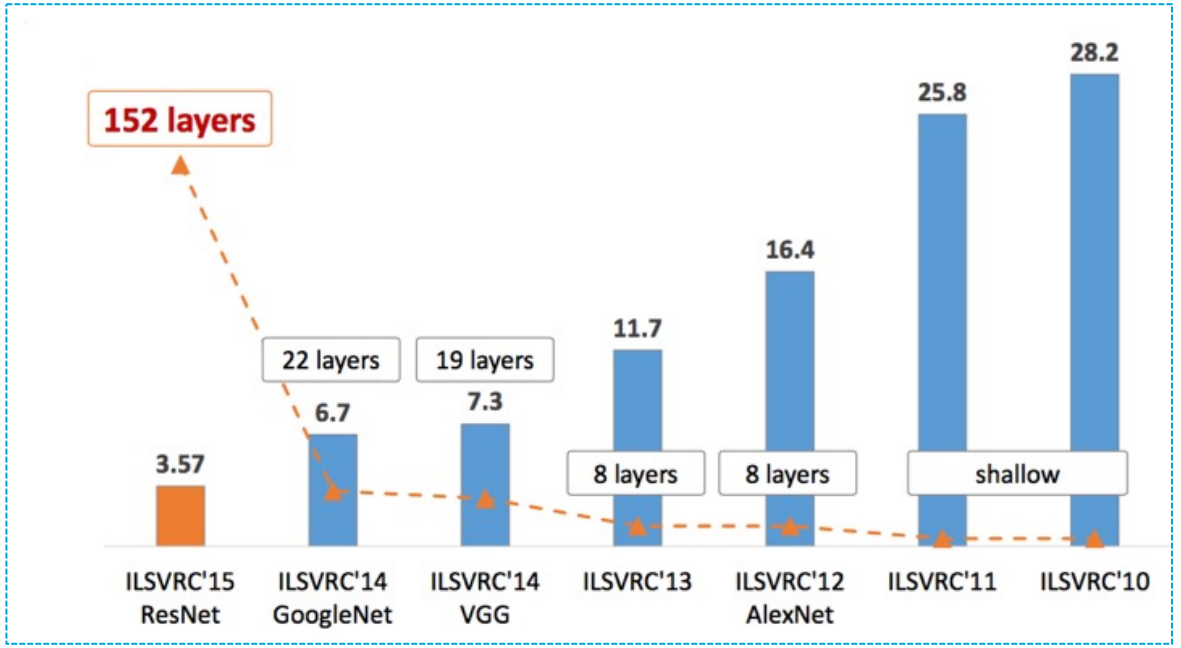
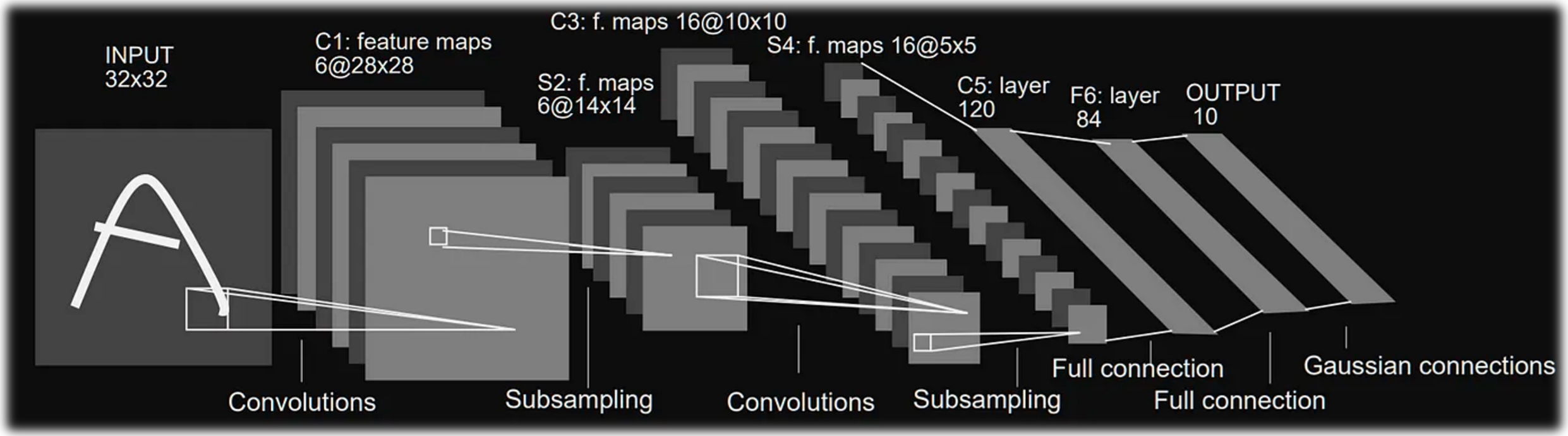


Image Classifier (IF)

LetNet for IF

'Hello World' in the domain of Convolution Neural Networks

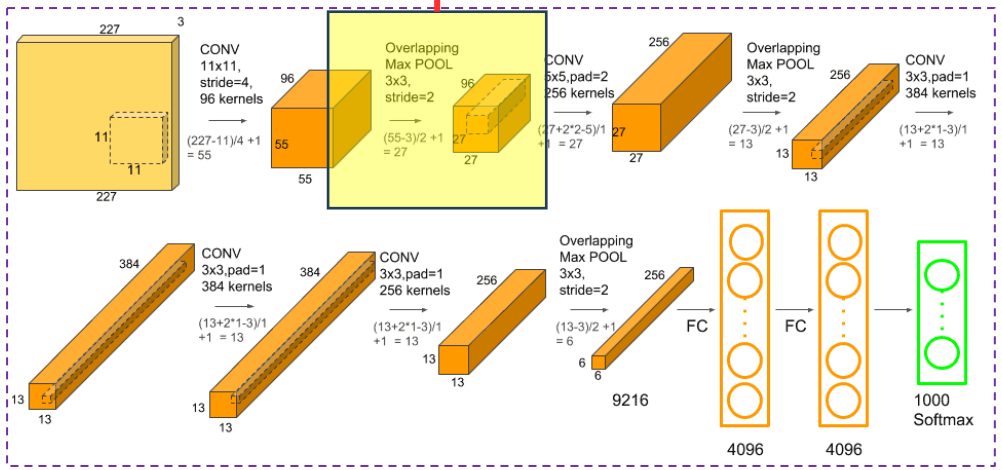


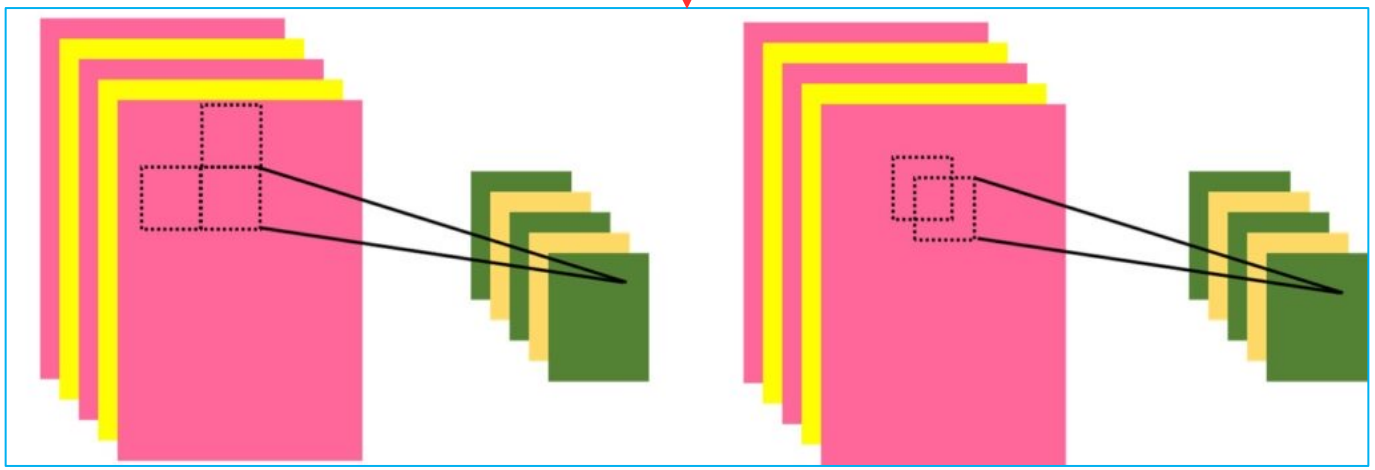
handwritten and machine-printed character recognition

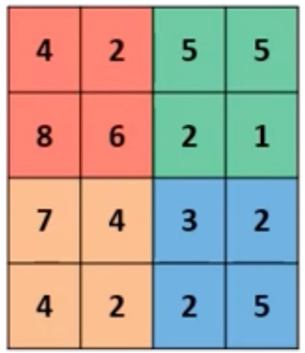
It was introduced by Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner in 1998

AlexNet for IF

New: Overlapping Max Pooling



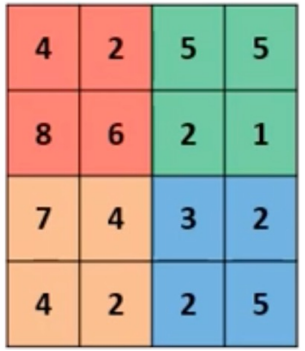




Max pooling

(2x2)
Stride 2

This overlapping nature of pooling helped reduce the top-1 error rate by 0.4% and top-5 error rate by 0.3% respectively when compared to using non-overlapping pooling windows of size 2×2 with a stride of 2 that would give same output dimensions.



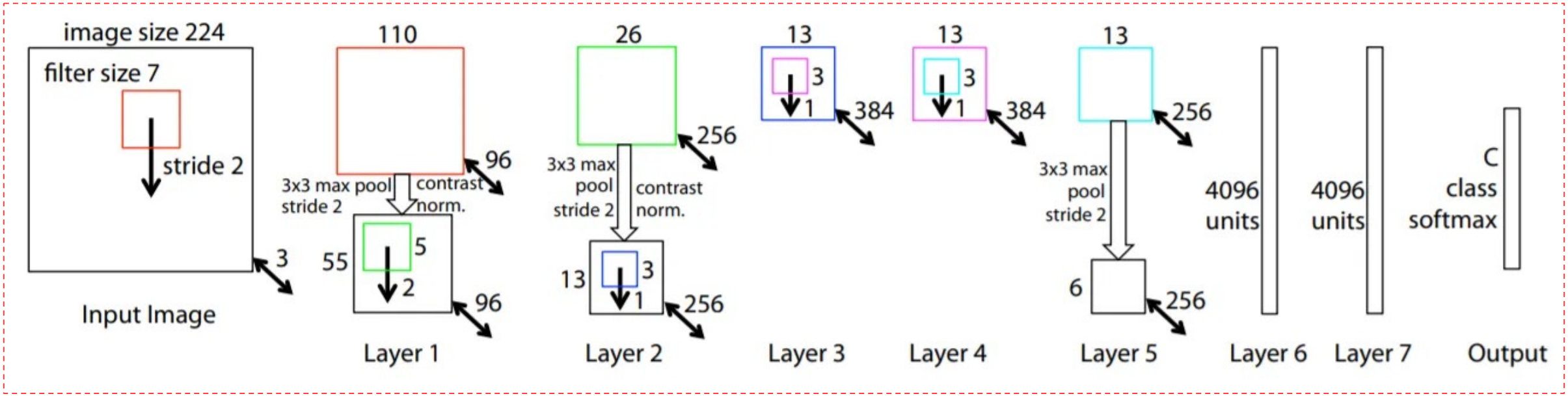
Overlap
max- pooling

(2x2)
Stride 1

AlexNet: Summary

Layer	# filters / neurons	Filter size	Stride	Padding	Size of feature map	Activation function
Input	-	-	-	-	227 x 227 x 3	-
Conv 1	96	11 x 11	4	-	55 x 55 x 96	ReLU
Max Pool 1	-	3 x 3	2	-	27 x 27 x 96	-
Conv 2	256	5 x 5	1	2	27 x 27 x 256	ReLU
Max Pool 2	-	3 x 3	2	-	13 x 13 x 256	-
Conv 3	384	3 x 3	1	1	13 x 13 x 384	ReLU
Conv 4	384	3 x 3	1	1	13 x 13 x 384	ReLU
Conv 5	256	3 x 3	1	1	13 x 13 x 256	ReLU
Max Pool 3	-	3 x 3	2	-	6 x 6 x 256	-
Dropout 1	rate = 0.5	-	-	-	6 x 6 x 256	-
Fully Connected 1	-	-	-	-	4096	ReLU
Dropout 2	rate = 0.5	-	-	-	4096	-
Fully Connected 2	-	-	-	-	4096	ReLU
Fully Connected 3	-	-	-	-	1000	Softmax

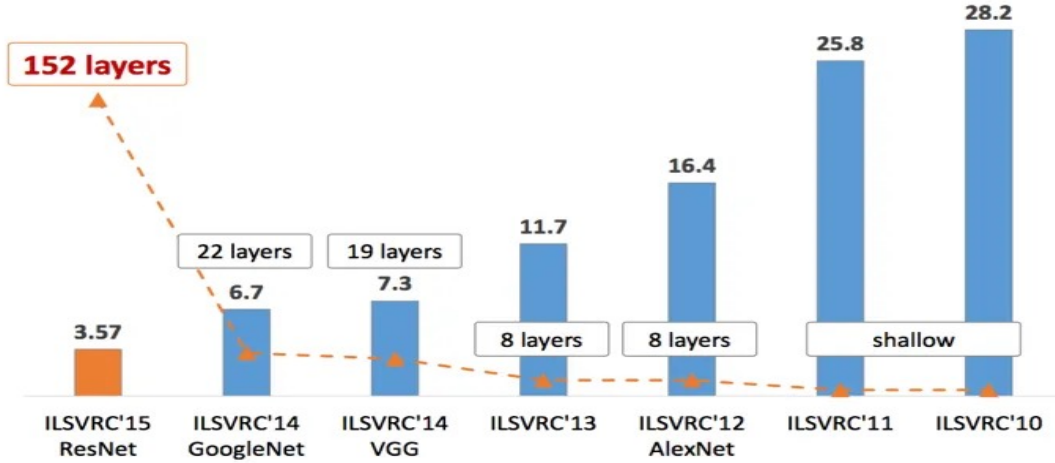
ZFNet for IF

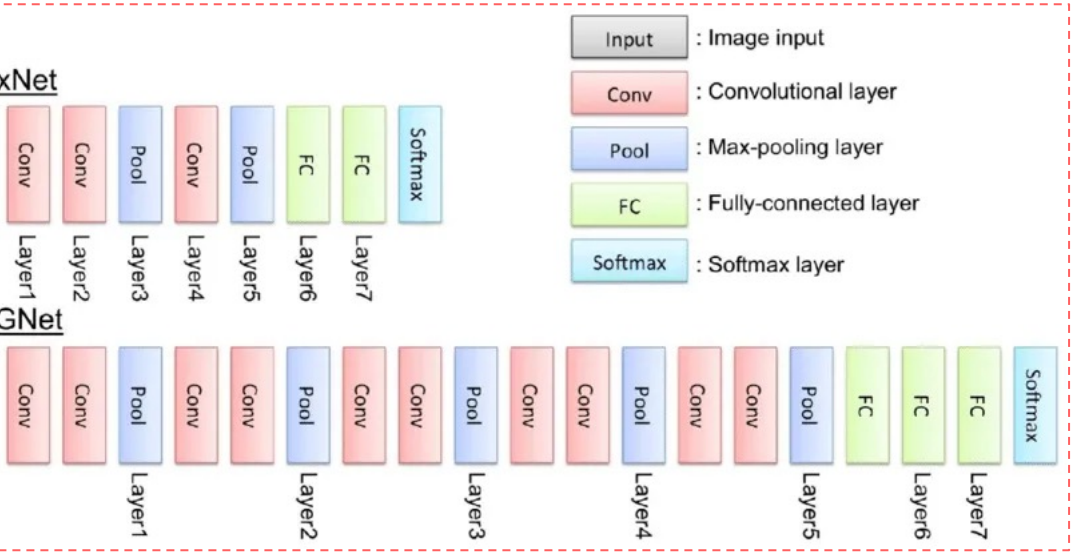


By visualizing the convolutional network, ZFNet become the Winner of ILSVLC 2013 in image classification by fine-tuning the AlexNet invented in 2012

ZFNet was invented by Rob Fergus and Matthew D. Zeiler

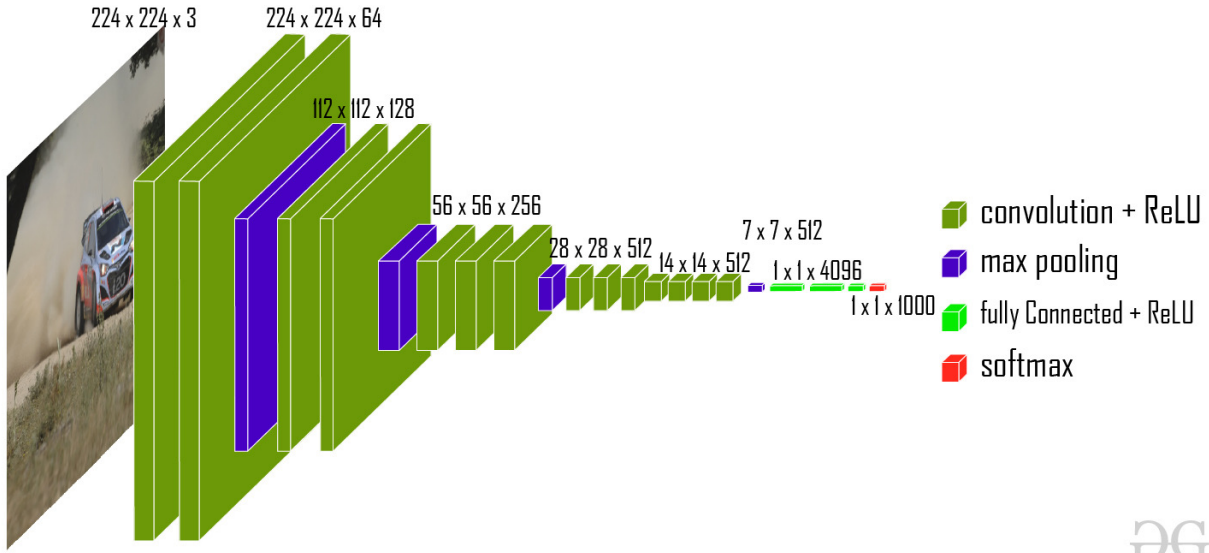
VGGNet for IF





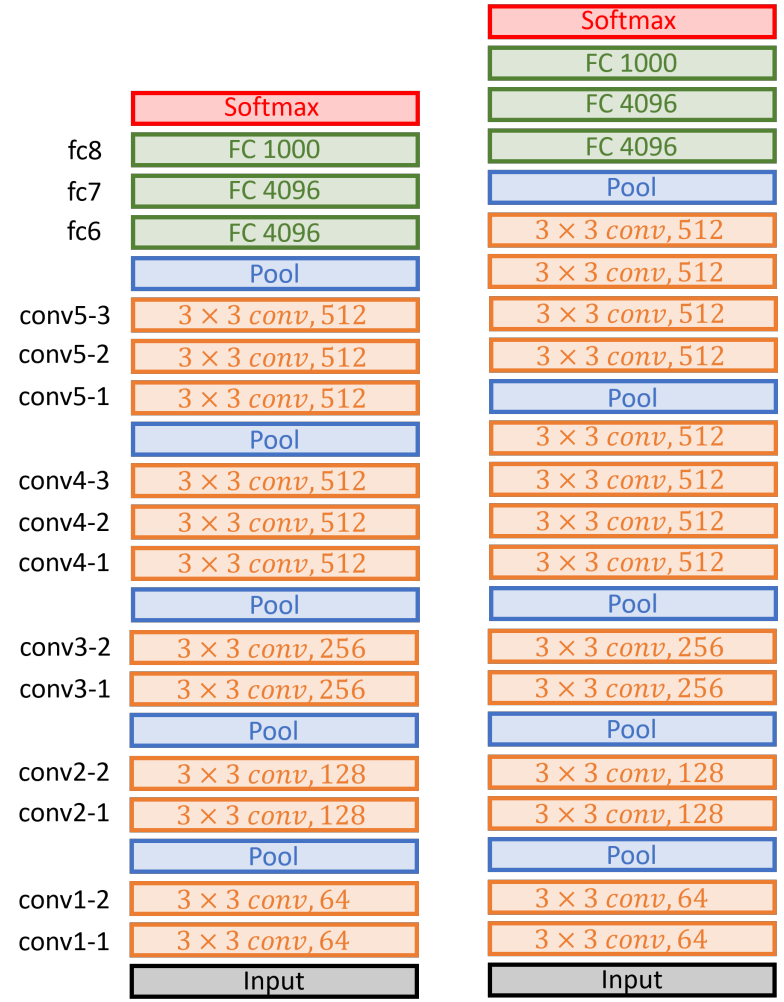
The full name of VGG is the **Visual Geometry Group**, which belongs to the Department of Science and Engineering of Oxford University.

The original purpose of VGG's research on the depth of convolutional networks is to understand how the depth of convolutional networks affects the accuracy and accuracy of large-scale image classification and recognition.

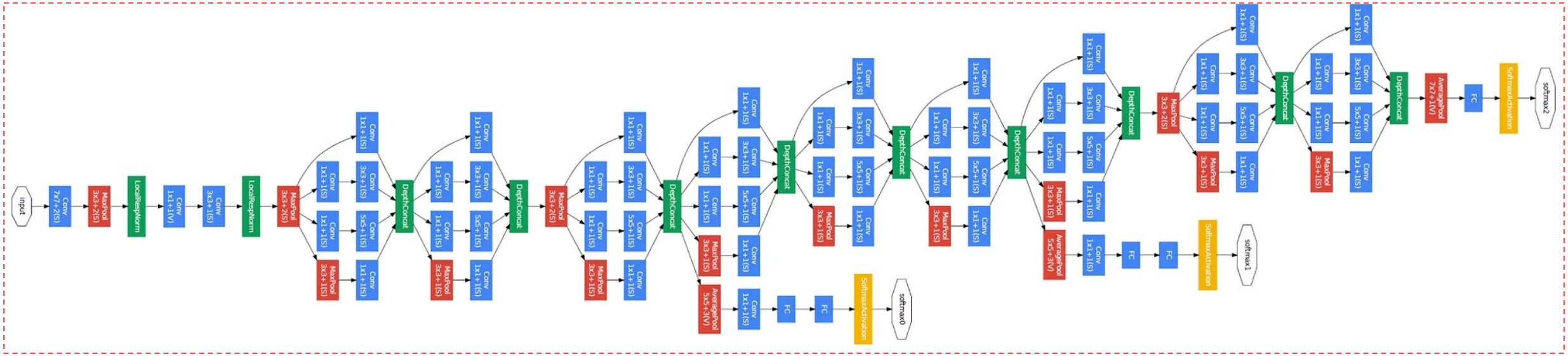


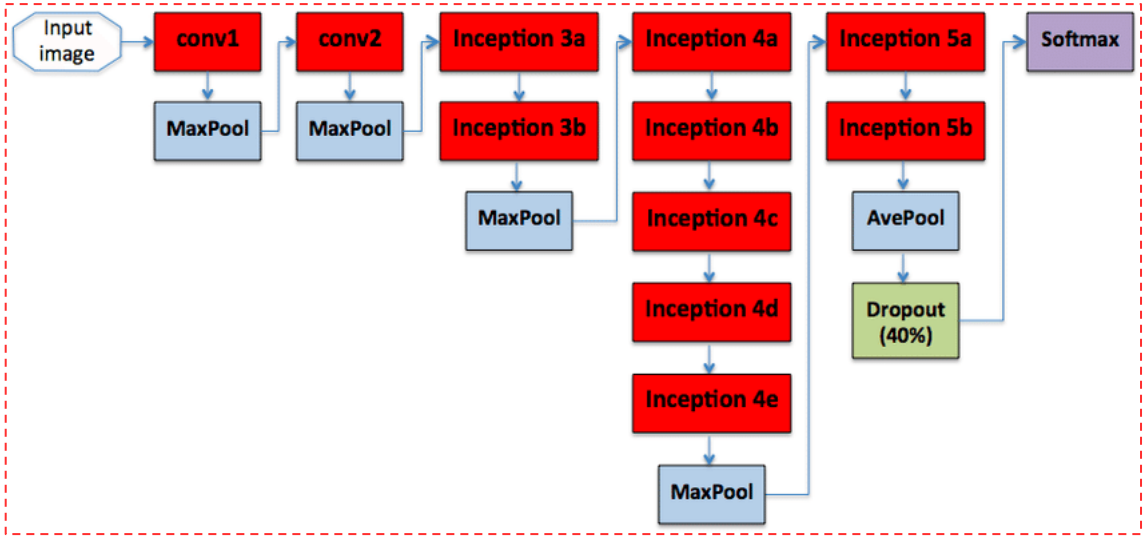
VGG16Net for IF

Layer		Feature Map	Size	Kernel Size	Stride	Activation
Input	Image	1	224 x 224 x 3	-	-	-
1	2 X Convolution	64	224 x 224 x 64	3x3	1	relu
	Max Pooling	64	112 x 112 x 64	3x3	2	relu
3	2 X Convolution	128	112 x 112 x 128	3x3	1	relu
	Max Pooling	128	56 x 56 x 128	3x3	2	relu
5	2 X Convolution	256	56 x 56 x 256	3x3	1	relu
	Max Pooling	256	28 x 28 x 256	3x3	2	relu
7	3 X Convolution	512	28 x 28 x 512	3x3	1	relu
	Max Pooling	512	14 x 14 x 512	3x3	2	relu
10	3 X Convolution	512	14 x 14 x 512	3x3	1	relu
	Max Pooling	512	7 x 7 x 512	3x3	2	relu
13	FC	-	25088	-	-	relu
14	FC	-	4096	-	-	relu
15	FC	-	4096	-	-	relu
Output	FC	-	1000	-	-	Softmax



GoogleLeNet for IF



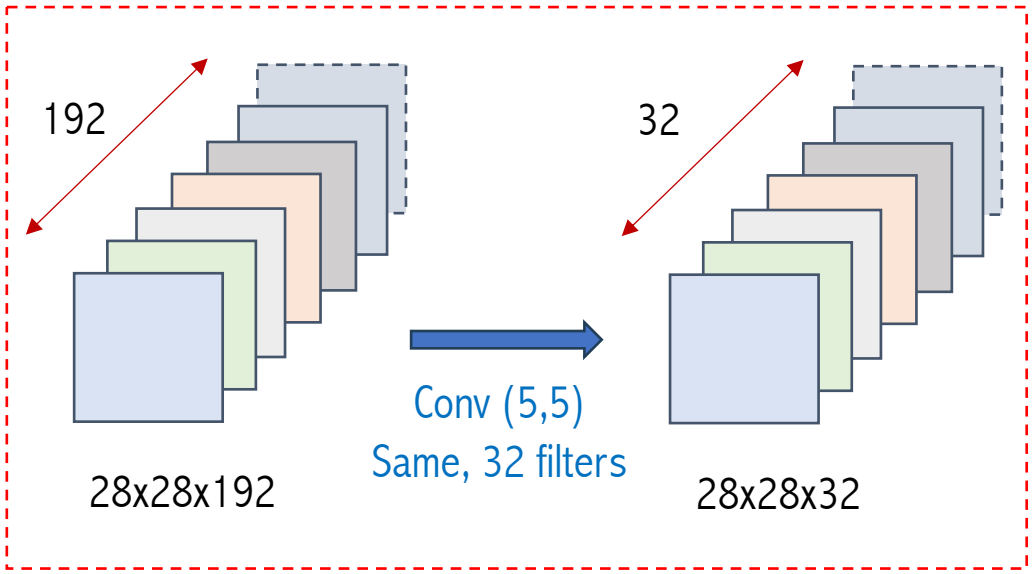


There are a total of 22 layers, It is already a very deep model compared with previous AlexNet, ZFNet, and VGGNet

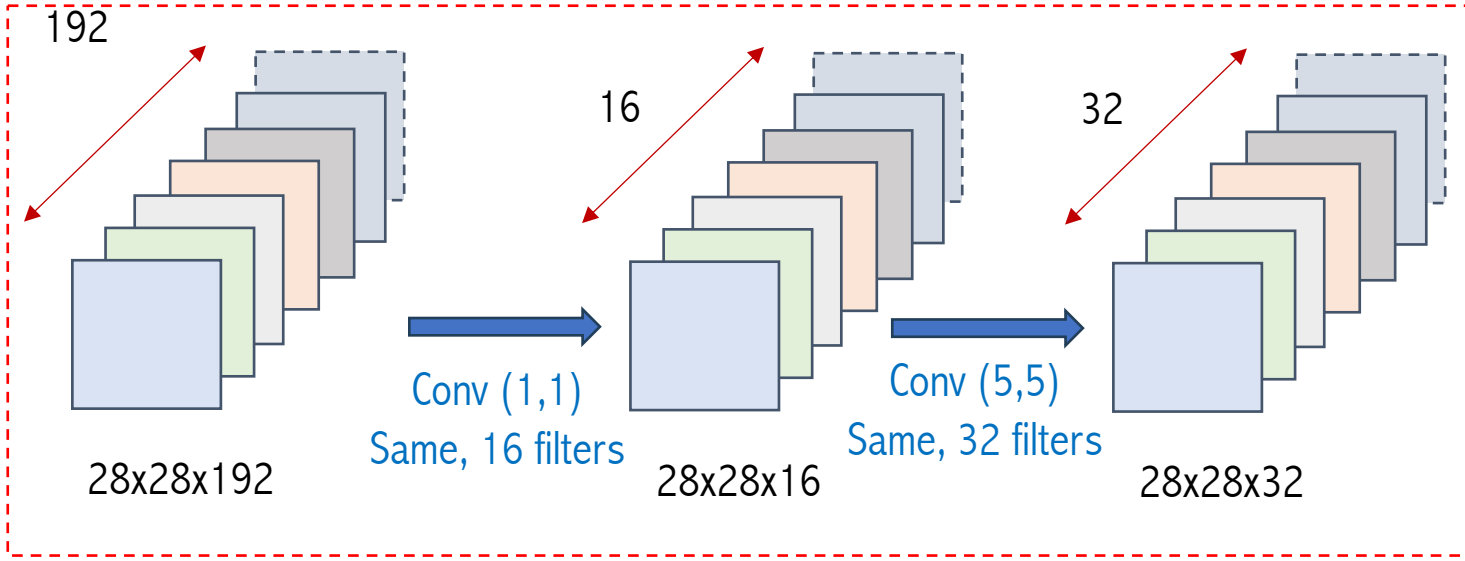
GoogleLeNe for IF

New: 1×1 convolution is used as a dimension reduction module to reduce the computation

5x5 convolution without using 1x1 bottleneck convolution



5x5 convolution using 1x1 bottleneck convolution



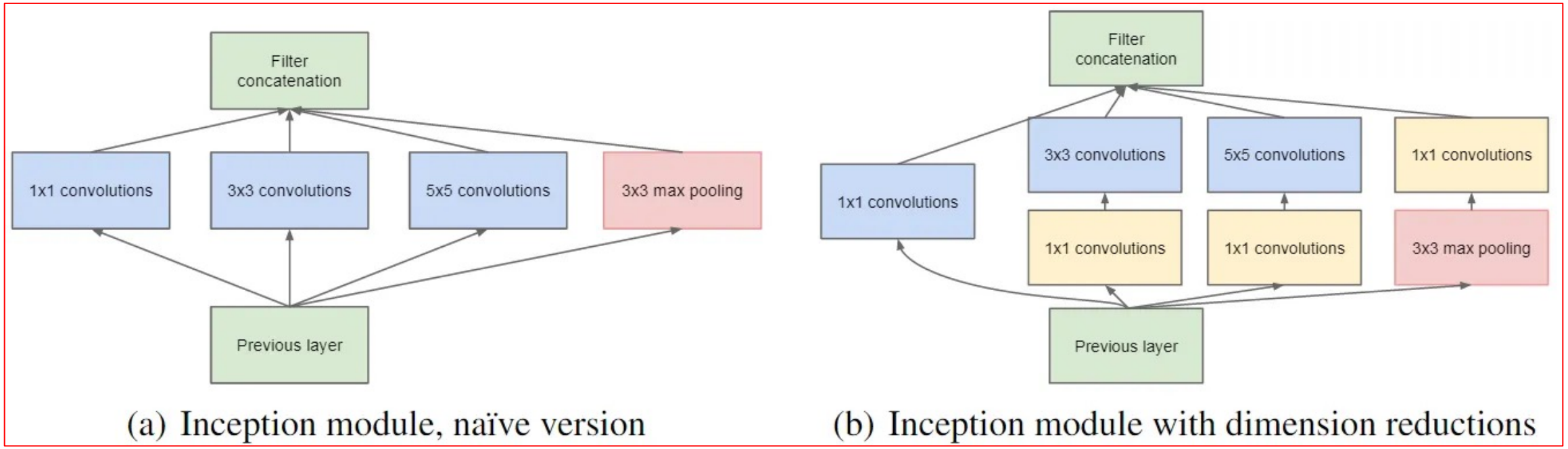
Number of parameters =28x28 (size of the input image)x 5x5(size of filter) x 192 (channels) x 32(no of filters) = 120,422,400 operations

Number of parameters =28 x 28 x 16 x 192 + 28 x 28 x 32 x 5 x 5 x 16 ~ 12.4M

As the authors said that the idea of the name "Inception" comes from famous internet meme below: WE NEED TO GO DEEPER

GoogLeNet for IF

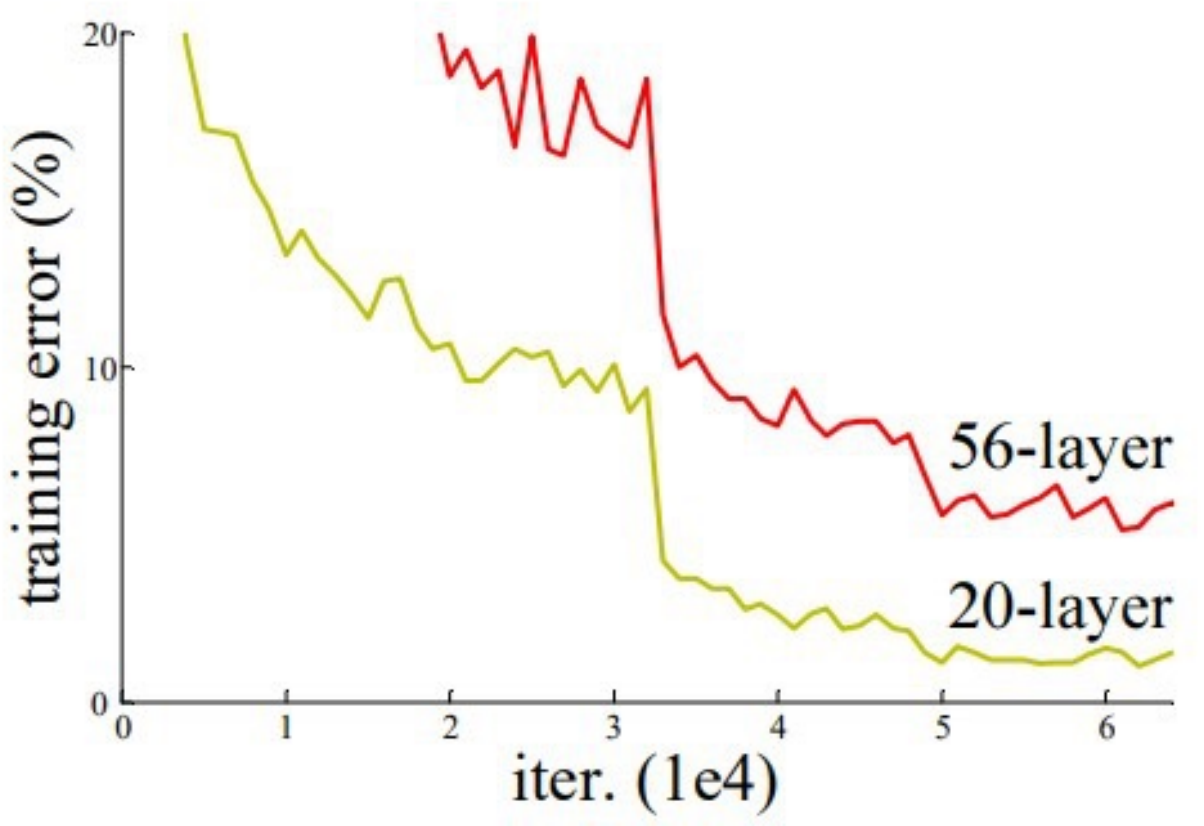
New: Inception Module

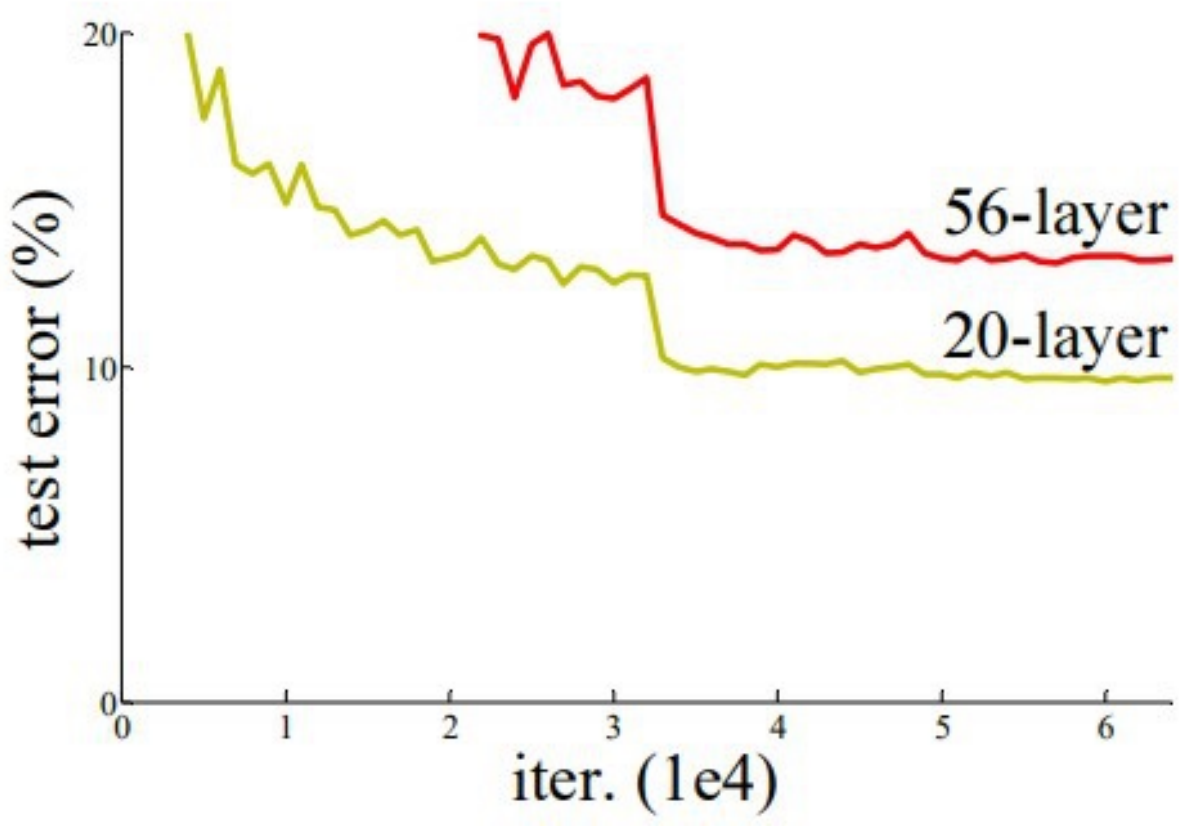


Inception V1 Module

Inception V1 Optimizated

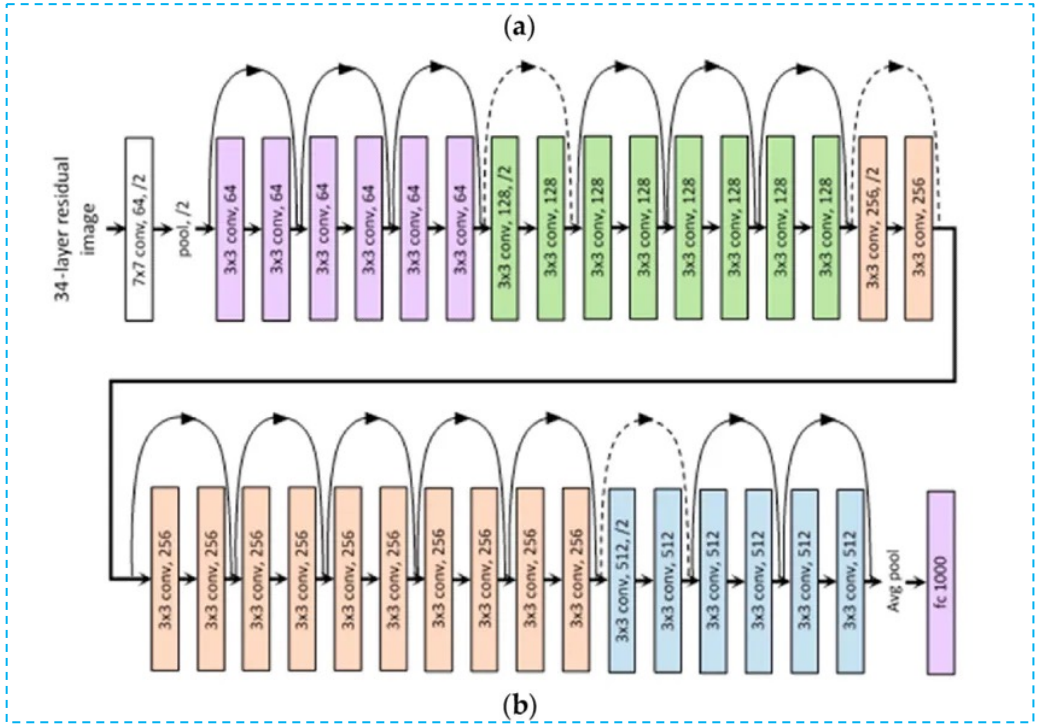
Residual Network (ResNet) for IF

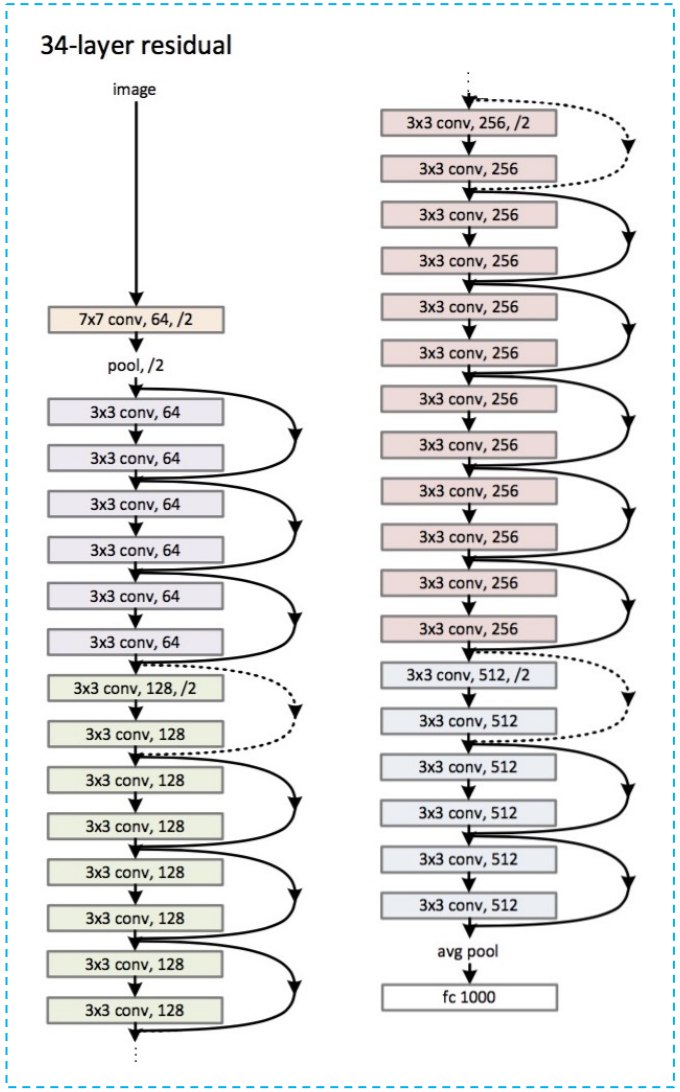




The Residual Blocks idea was created by this design to address the issue of the vanishing/exploding gradient

Residual Network (ResNet) for IF

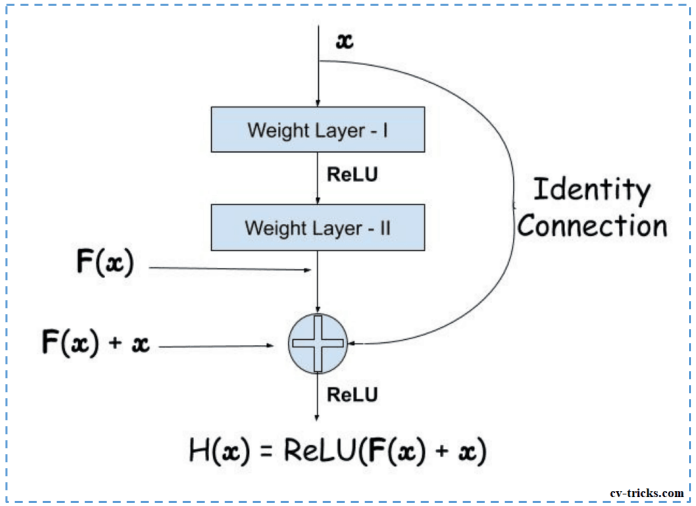




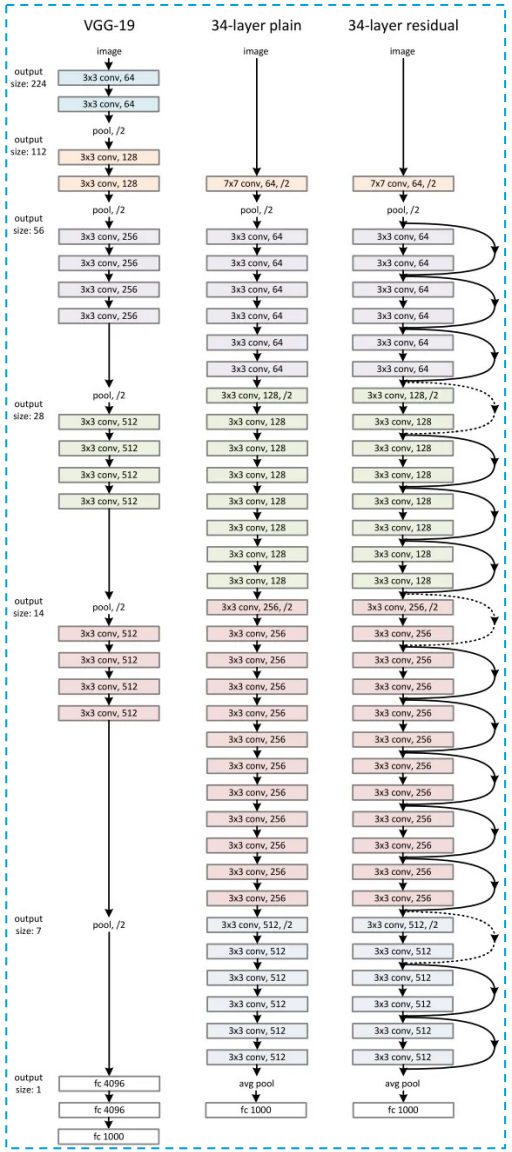
A novel architecture called **Residual Network** was launched by Microsoft Research experts in 2015 with the proposal of **ResNet.**

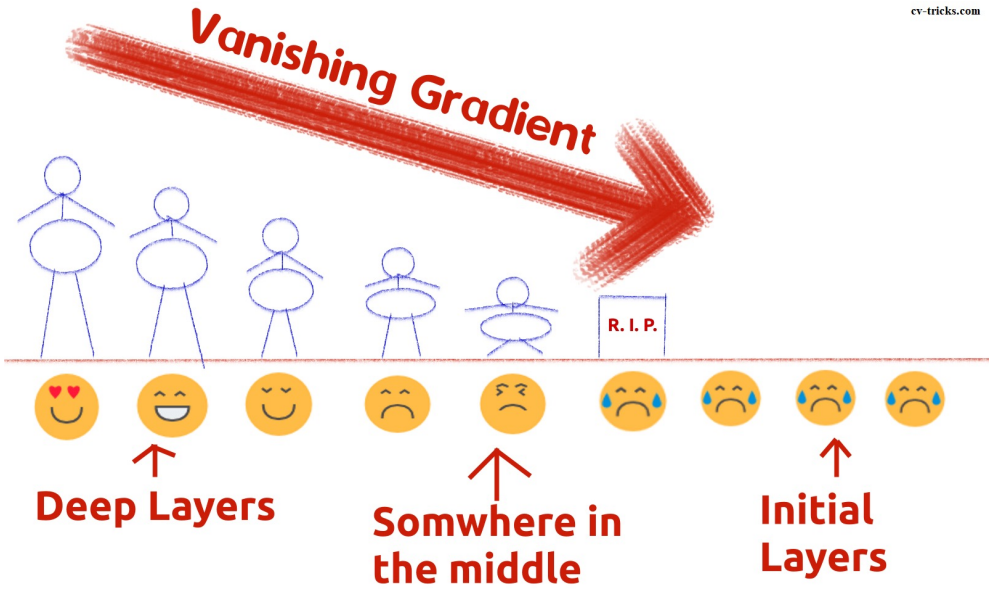
Residual Network (ResNet34) for IF

New: Skip Connection



The intuition is that learning f(x) = 0 has to be easy for the network.

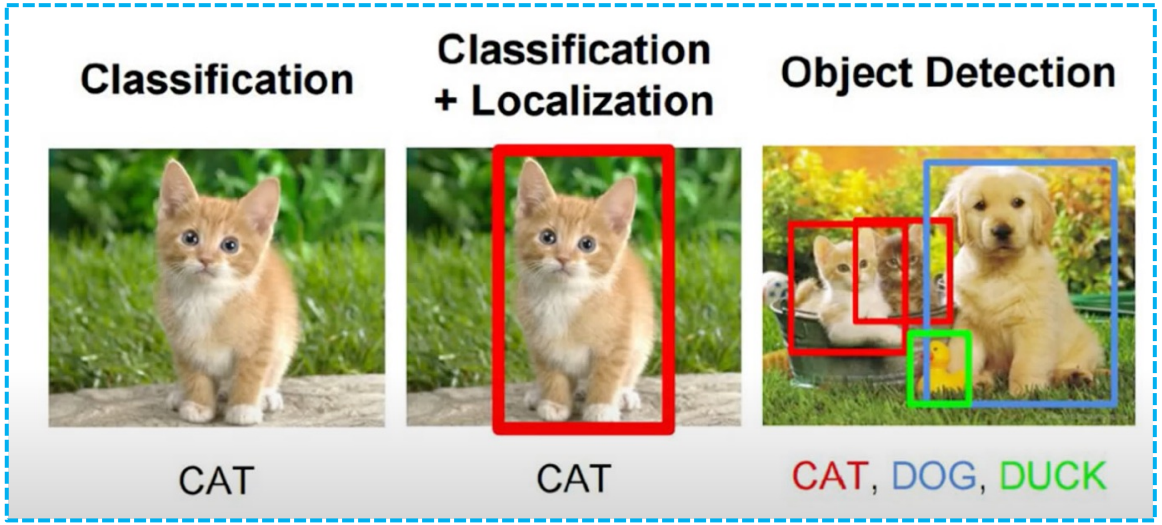


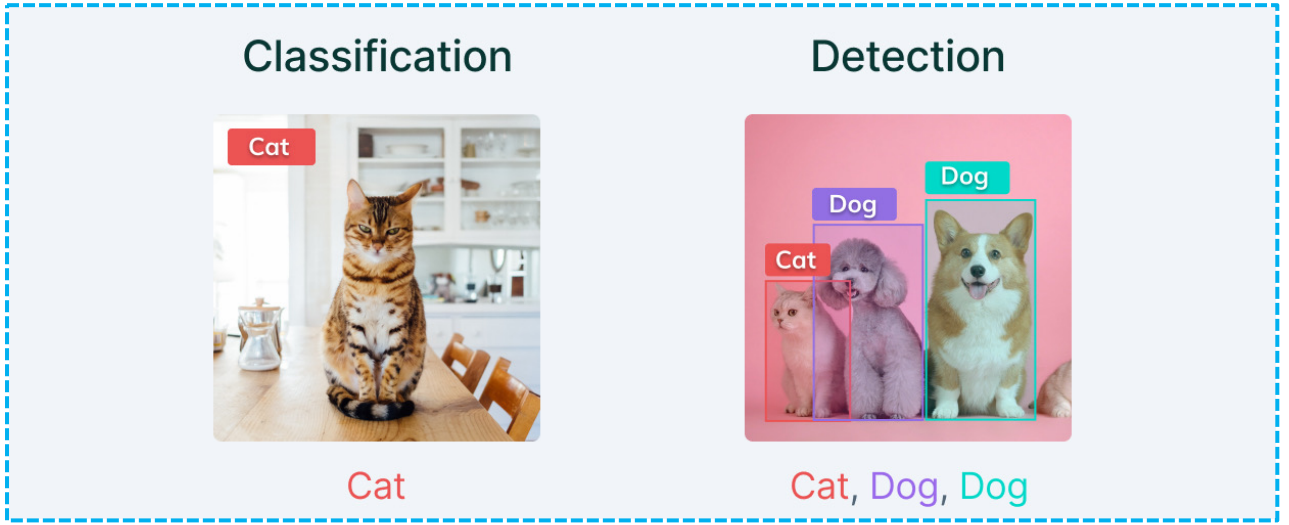


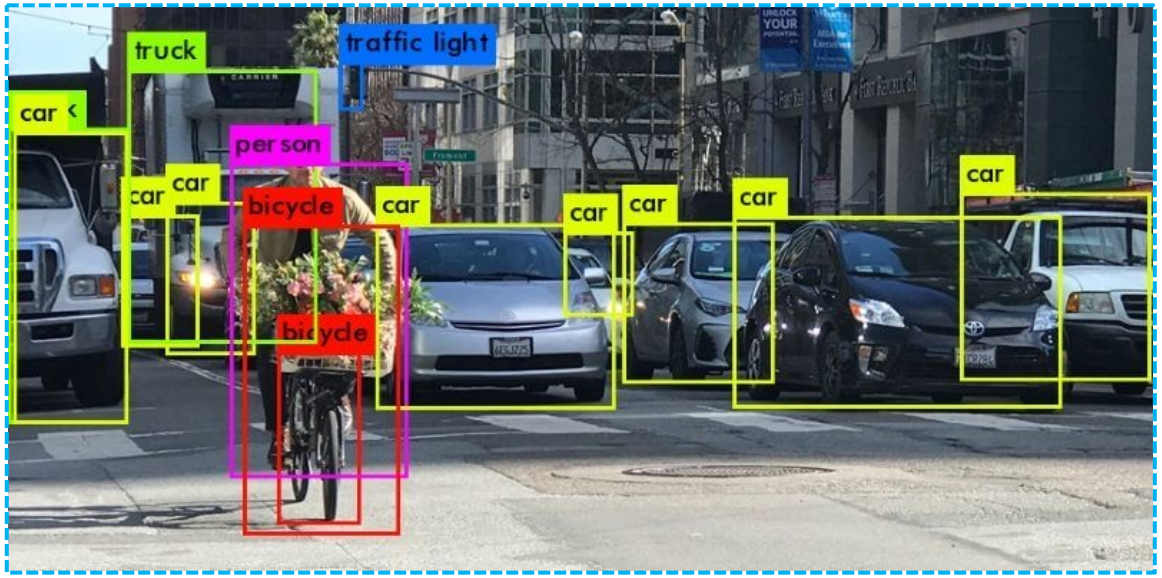
The VGG-19-inspired 34-layer plain network architecture used by ResNet is followed by the addition of the shortcut connection

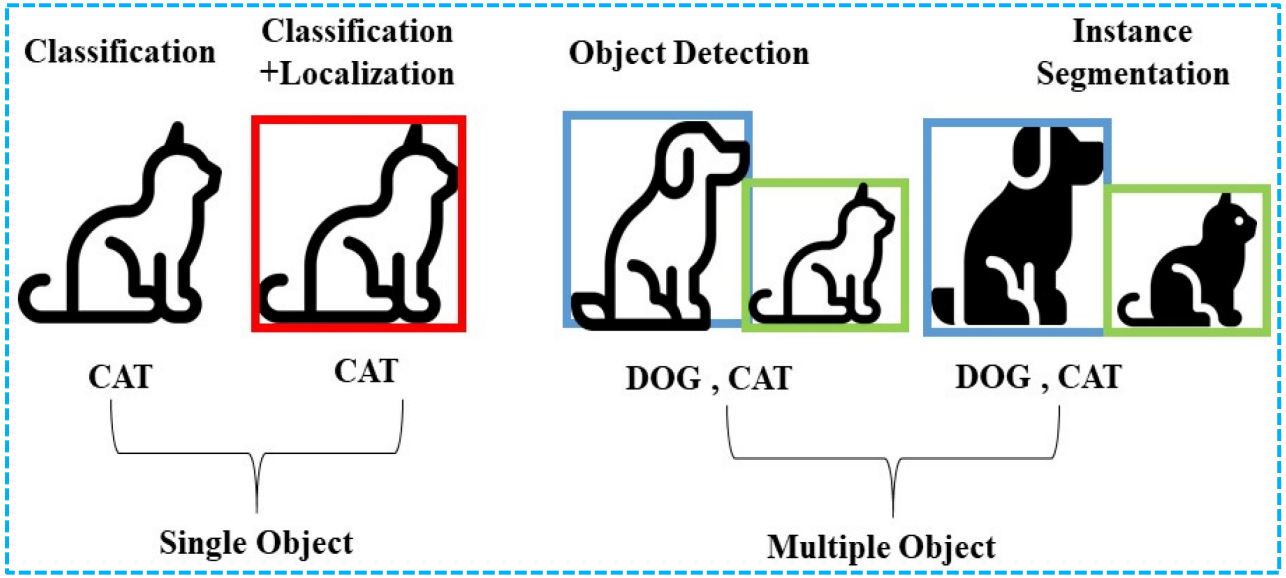
AI VIETNAM
All-in-One Course

Image Classification vs. Object Detection









Outline

- Introduction to Object Detection
- Applications of Object Detection
- Object Detection Milestones
- Traditional Object Detectors
- CNNs for Image Classification
- Image Classifier to Object Detectors with CNN
- CNN Limitations and Spatial Outputs
- Assignment

Image Classifier To Object Detector

Step 1: Input Image

Step 2: Construct Image Pyramid

Step 3: Run sliding window at each scale of image pyramid

Step 3.1: For each step of sliding window, extract ROI

Step 3.2: Take ROI and pass it through CNN for classification

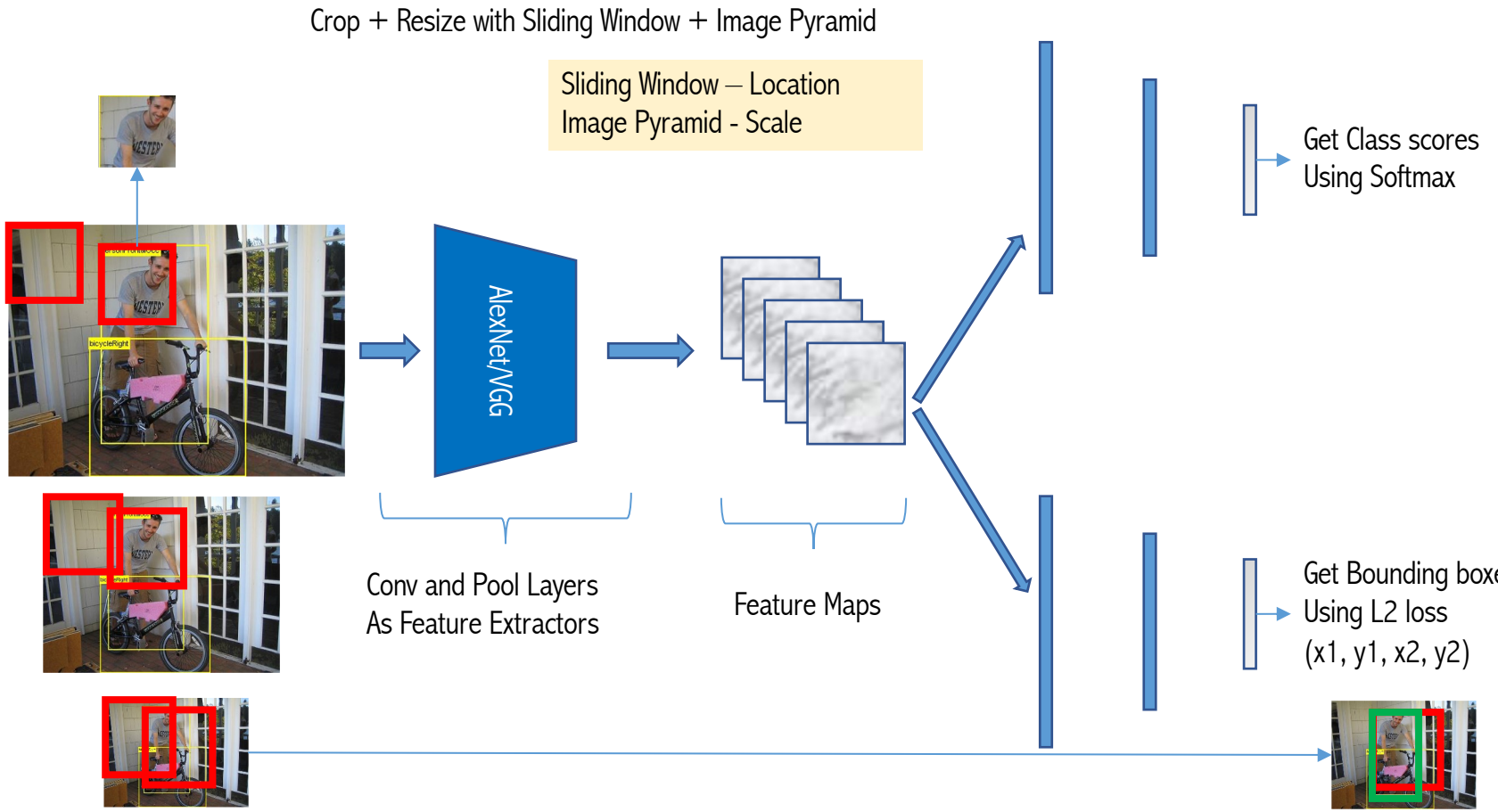
Step 3.3: if min probability test passes, record class label and bounding box location

Step 4: Appy NMS

Step 5: Return Result

Code: https://colab.research.google.com/drive/1EtxlG4XRgrsOF7N8ivHQ6JZvVRboQx-8?usp=sharing

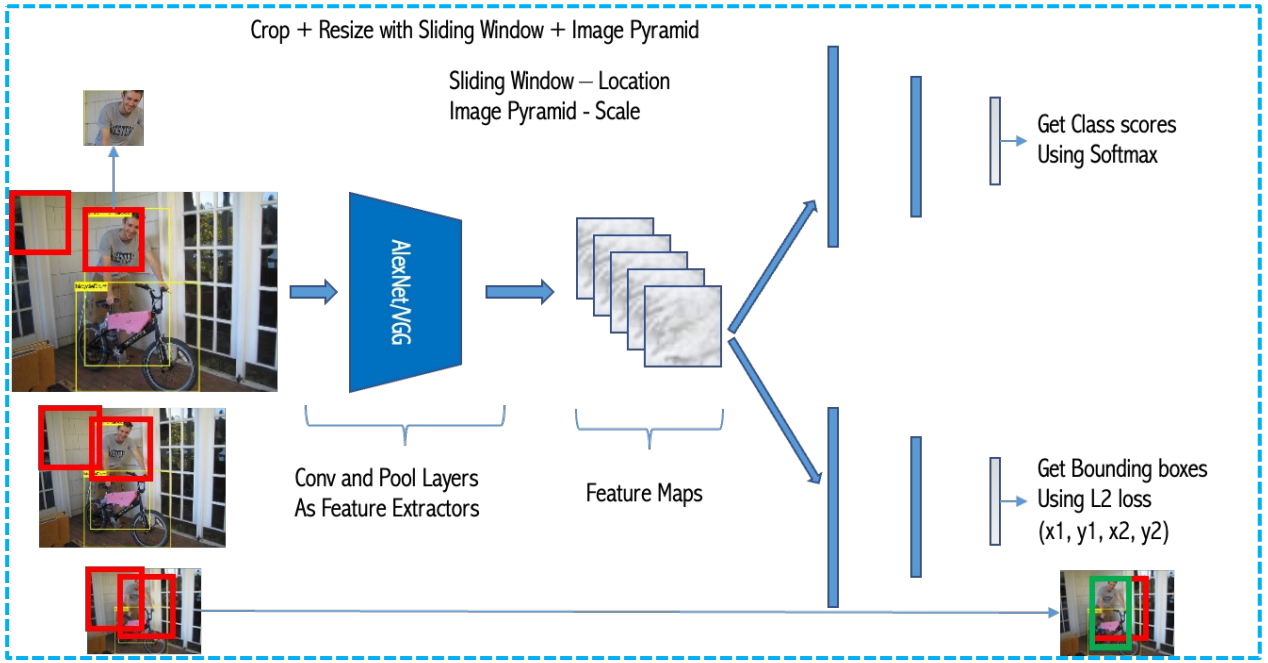
Ideas for Detection using CNN

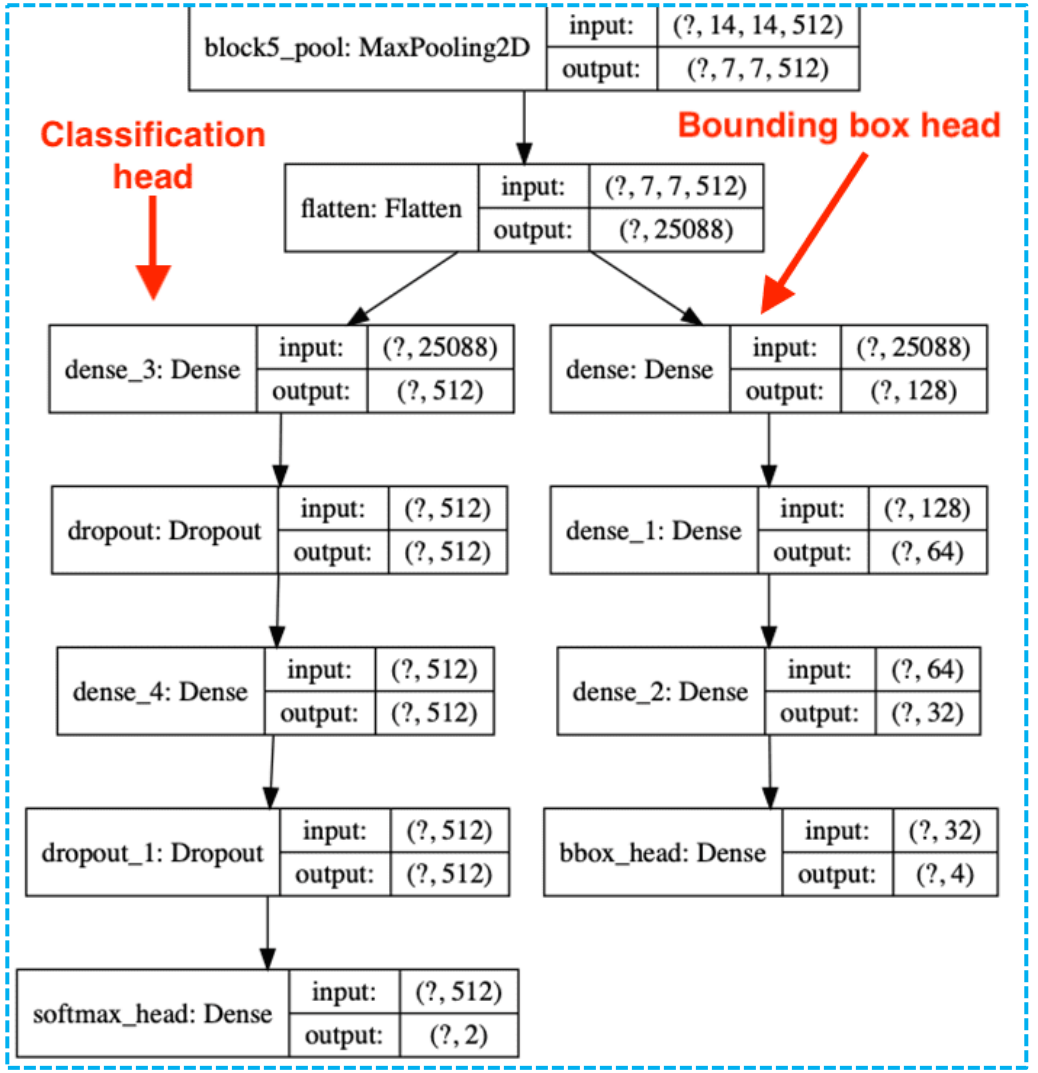


For example, to process an image of 800x800, if the sliding window size is 224, we will end up with 331,776 crops.

Image Credit - http://host.robots.ox.ac.uk/pascal/VOC/voc2012/examples/index.html

Ideas for Detection using CNN

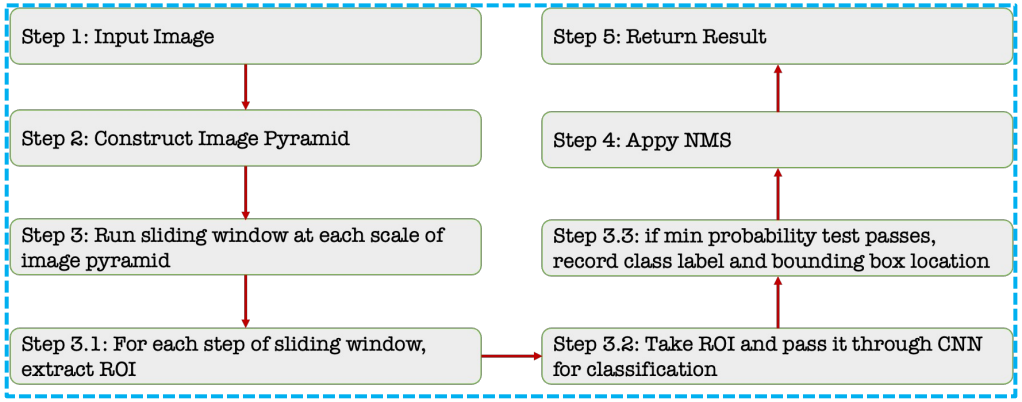


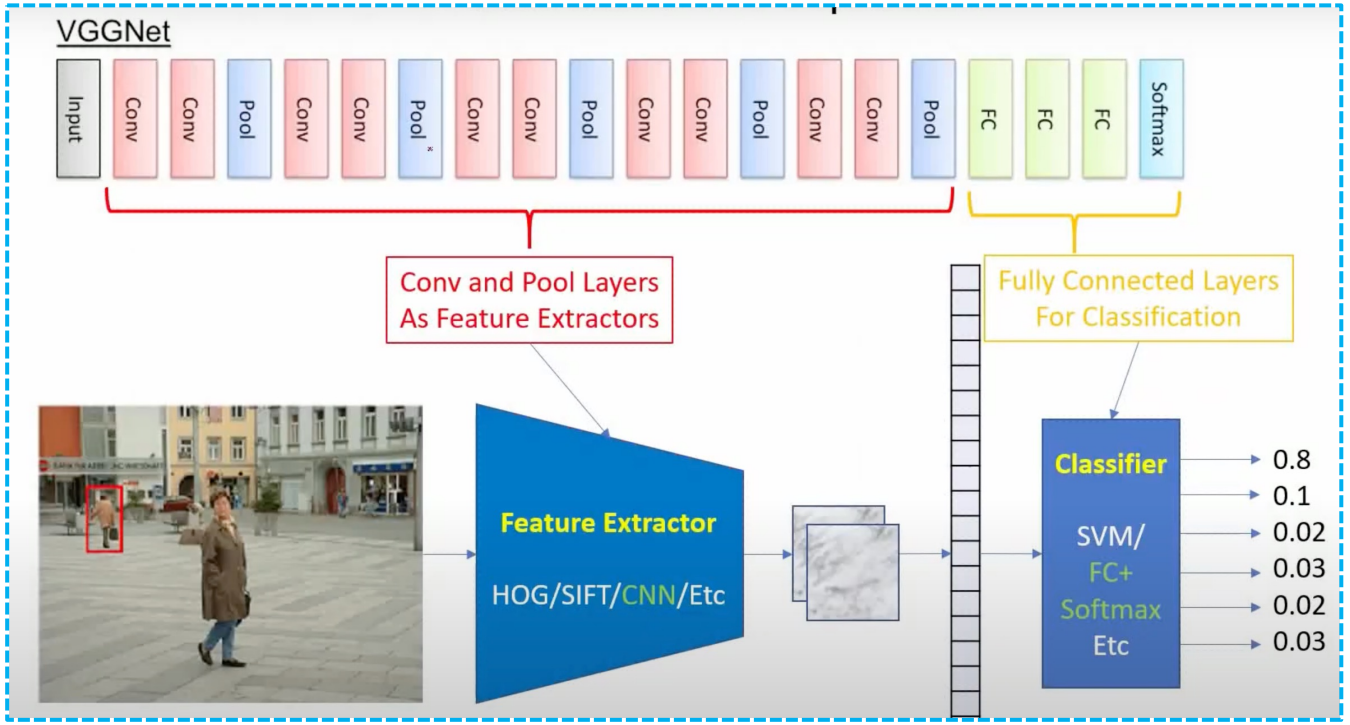


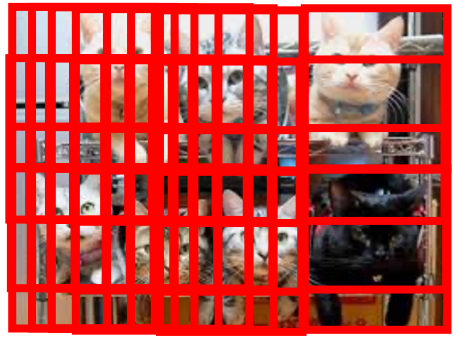
Outline

- Introduction to Object Detection
- Applications of Object Detection
- Object Detection Milestones
- Traditional Object Detectors
- CNNs for Image Classification
- Image Classifier to Object Detectors with CNN
- CNN Limitations and Spatial Outputs
- Assignment

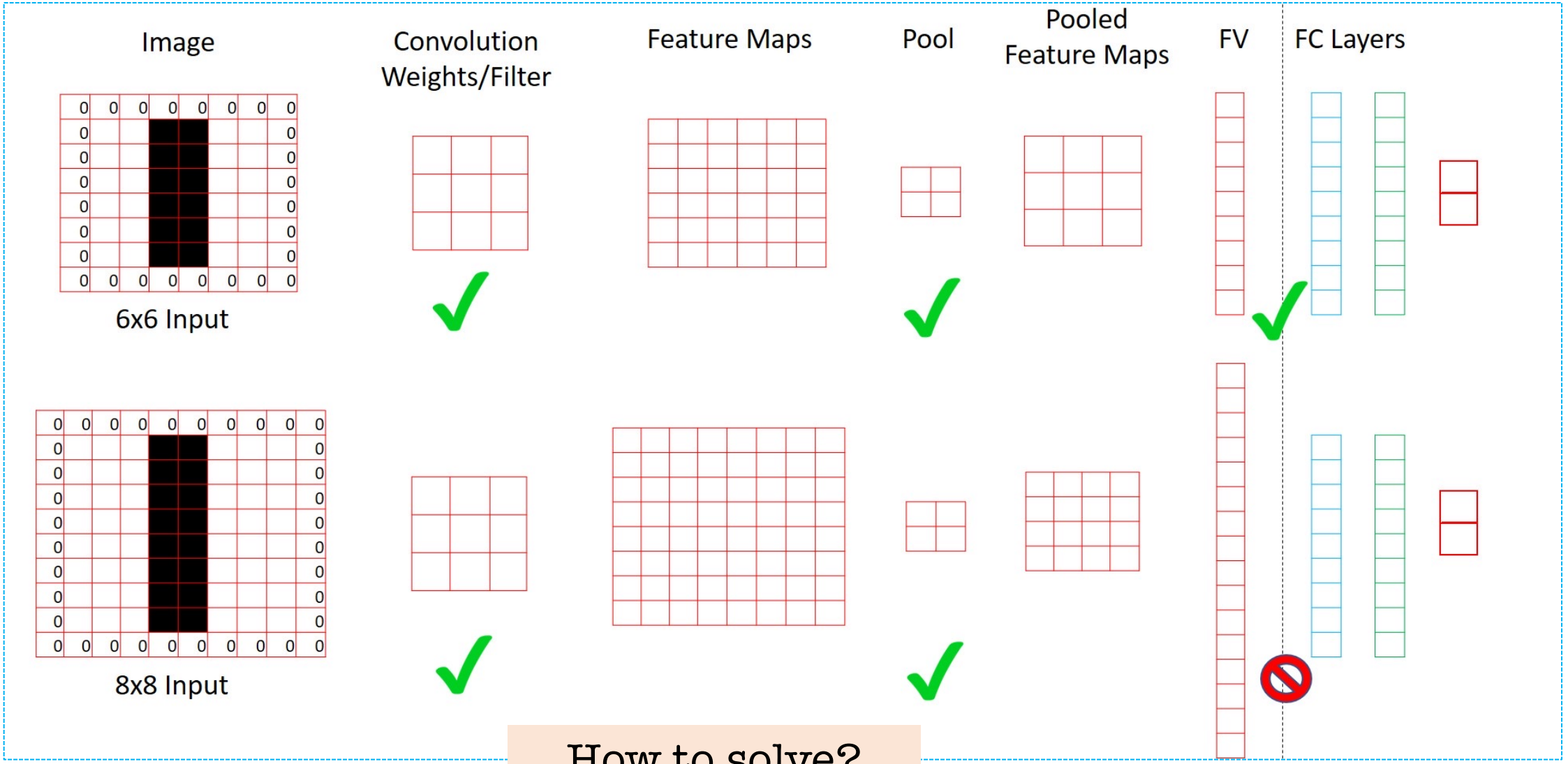
Image Classifier To Object Detector





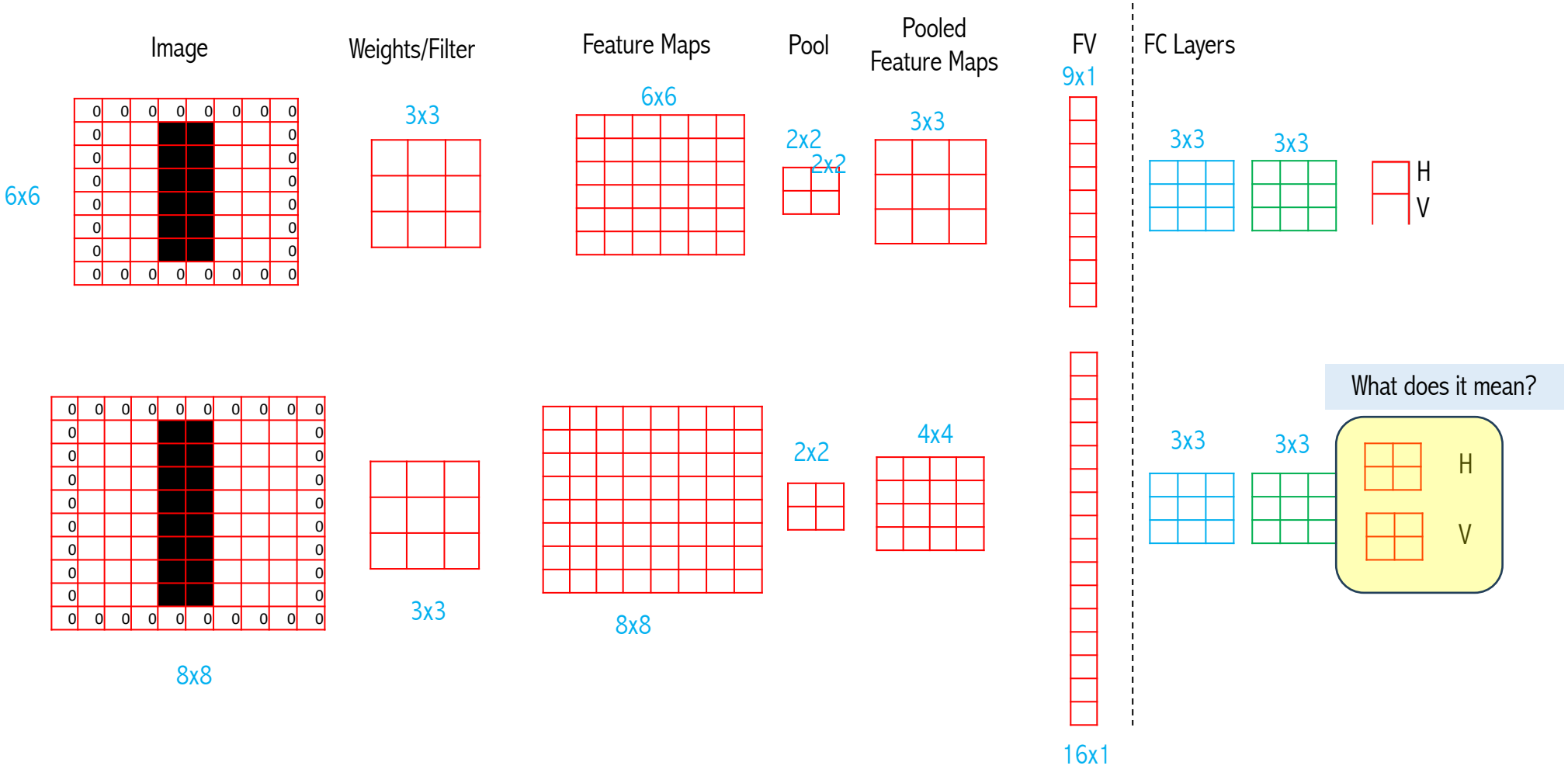


Problems: CNN Input Size Constraints

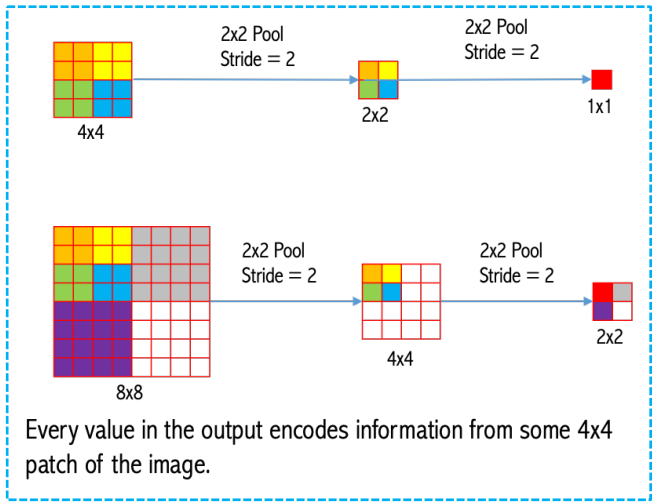


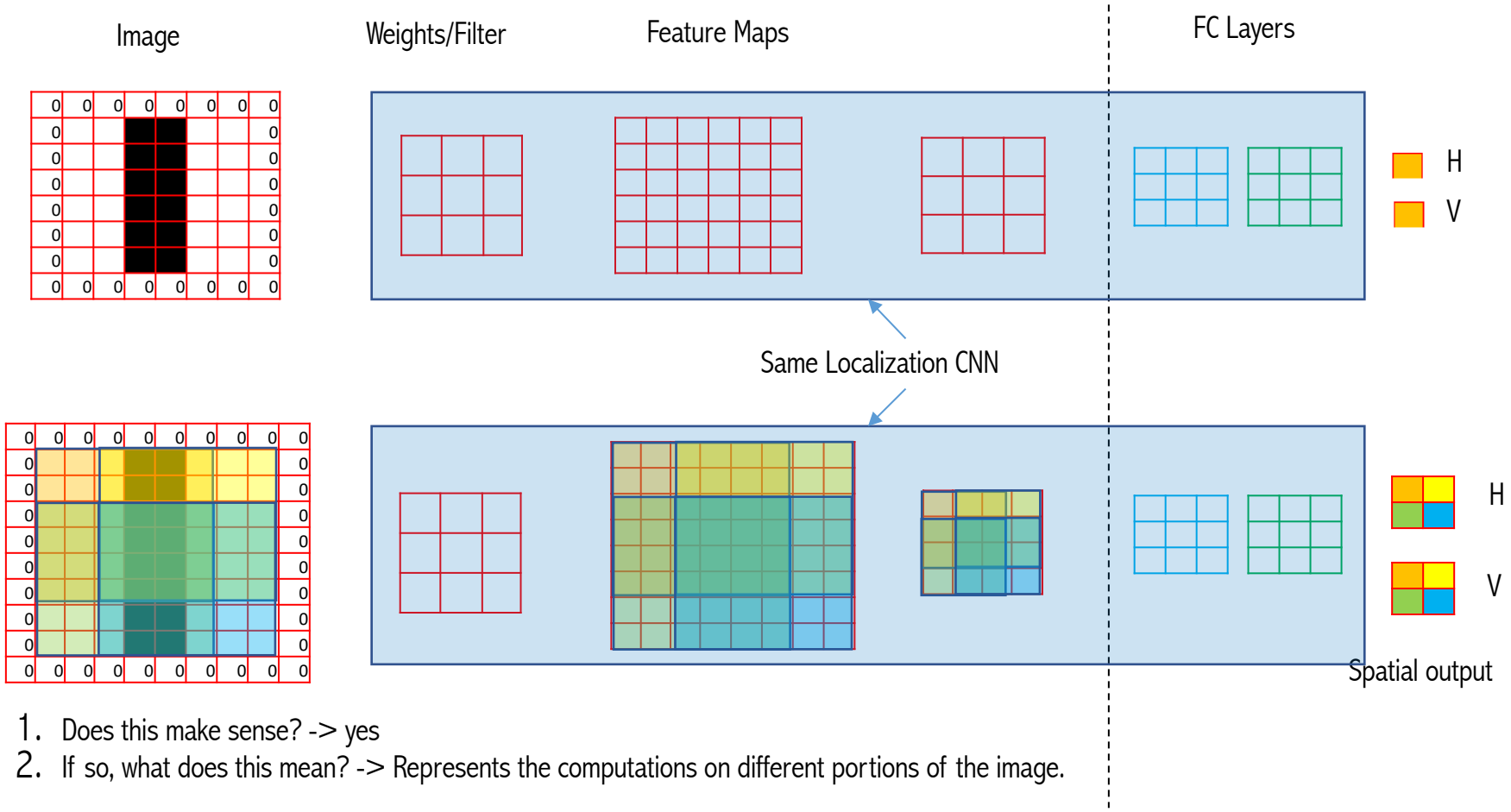
How to solve?

ConvNets input size constraints – FC as Conv



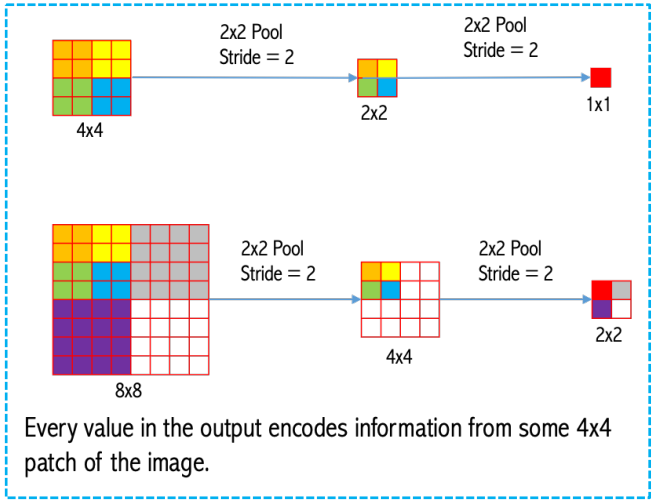
ConvNets input size constraints – FC as Conv

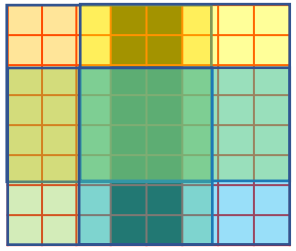


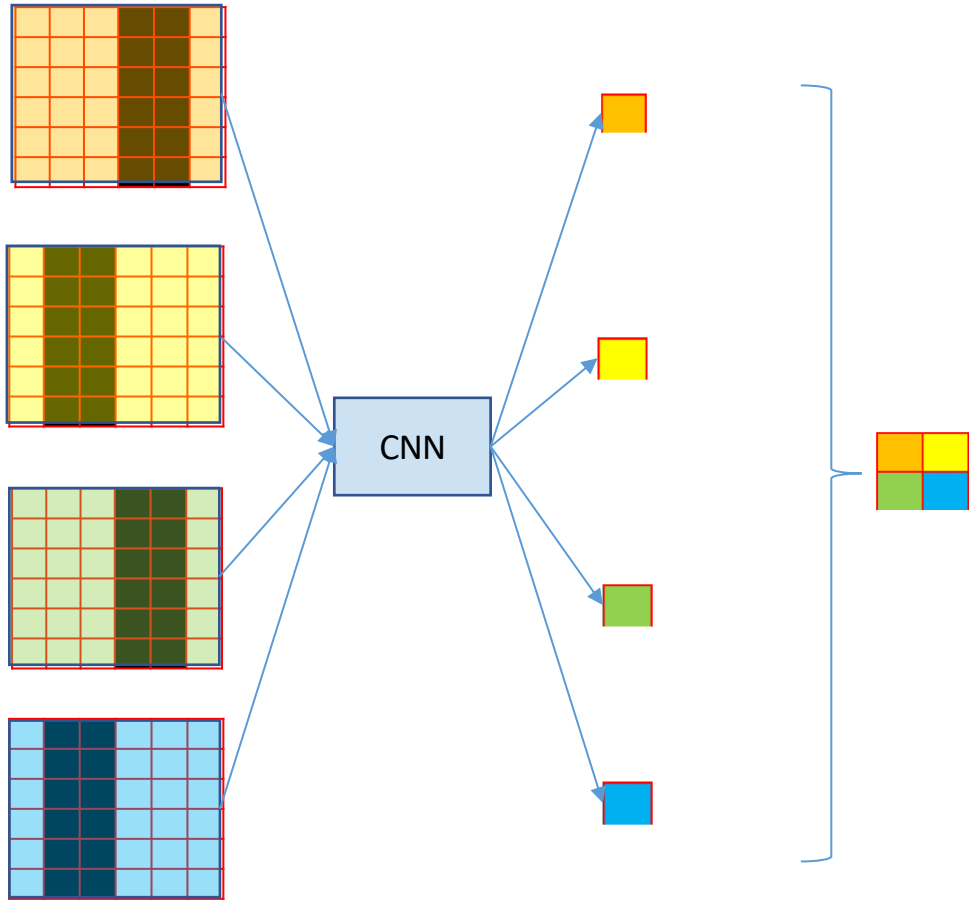


1. Does this make sense? -> yes
2. If so, what does this mean? -> Represents the computations on different portions of the image.

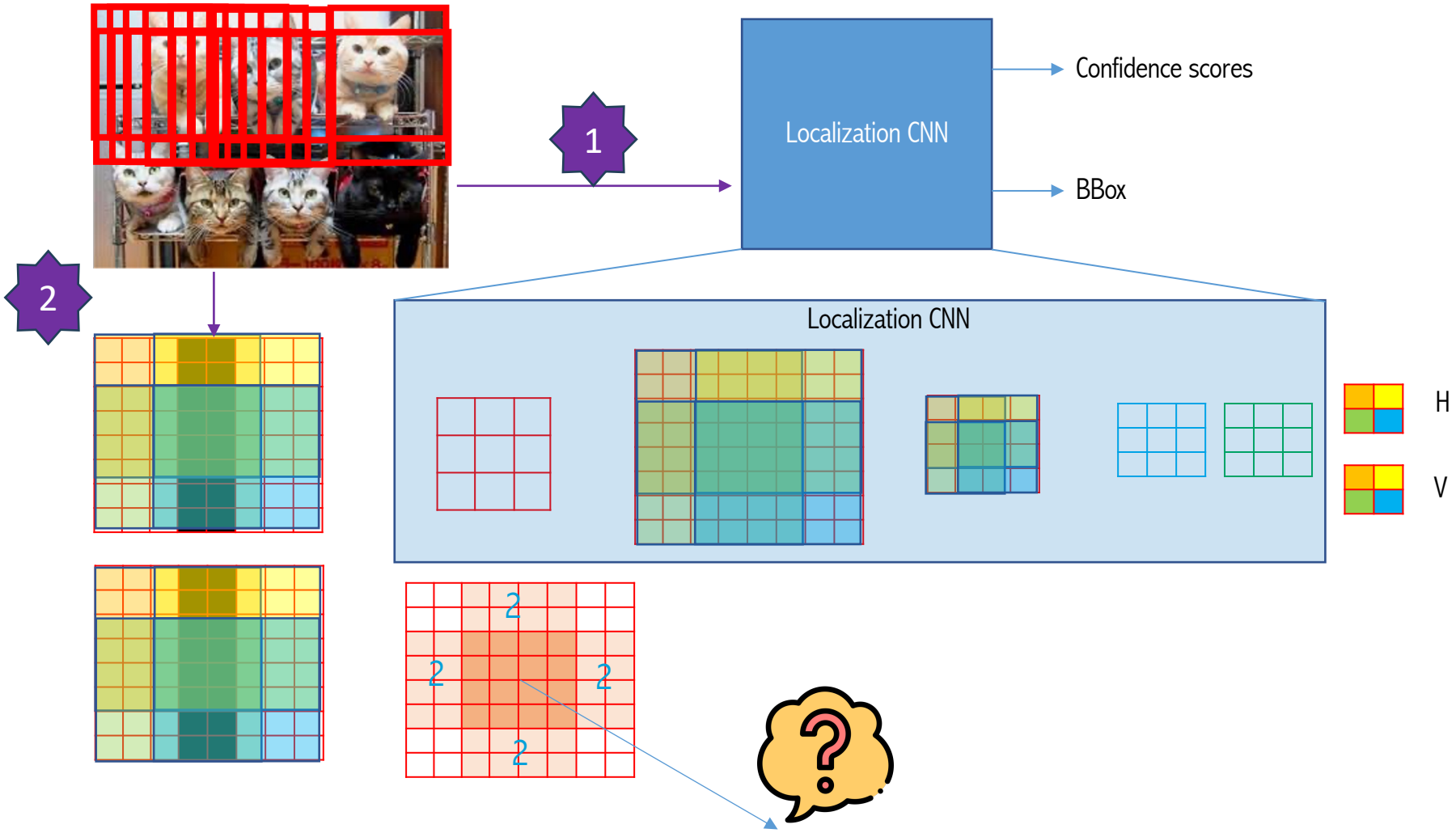
ConvNets input size constraints – FC as Conv



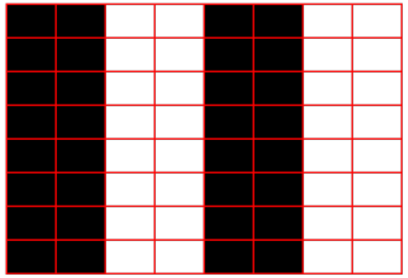




ConvNets and Sliding Window Efficiency



ConvNets and Sliding Window Efficiency



0	0	255	255	0	0	255	255
0	0	255	255	0	0	255	255
0	0	255	255	0	0	255	255
0	0	255	255	0	0	255	255
0	0	255	255	0	0	255	255
0	0	255	255	0	0	255	255
0	0	255	255	0	0	255	255
0	0	255	255	0	0	255	255

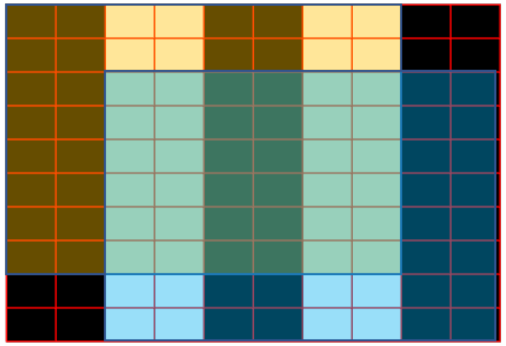
8x8

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

3x3

-765	-765	765	765	-765	-765
-765	-765	765	765	-765	-765
-765	-765	765	765	-765	-765
-765	-765	765	765	-765	-765
-765	-765	765	765	-765	-765
-765	-765	765	765	-765	-765

6x6



0	0	255	255	0	0	255	255	0	0
0	0	255	255	0	0	255	255	0	0
0	0	255	255	0	0	255	255	0	0
0	0	255	255	0	0	255	255	0	0
0	0	255	255	0	0	255	255	0	0
0	0	255	255	0	0	255	255	0	0
0	0	255	255	0	0	255	255	0	0
0	0	255	255	0	0	255	255	0	0
0	0	255	255	0	0	255	255	0	0
0	0	255	255	0	0	255	255	0	0

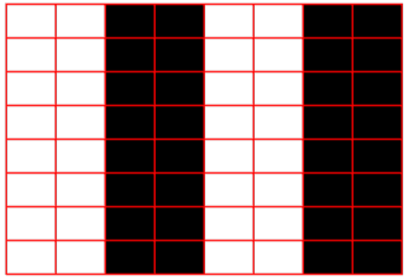
10x10

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

3x3

-765	-765	765	765	-765	-765	765	765
-765	-765	765	765	-765	-765	765	765
-765	-765	765	765	-765	-765	765	765
-765	-765	765	765	-765	-765	765	765
-765	-765	765	765	-765	-765	765	765
-765	-765	765	765	-765	-765	765	765
-765	-765	765	765	-765	-765	765	765
-765	-765	765	765	-765	-765	765	765

8x8



255	255	0	0	255	255	0	0
255	255	0	0	255	255	0	0
255	255	0	0	255	255	0	0
255	255	0	0	255	255	0	0
255	255	0	0	255	255	0	0
255	255	0	0	255	255	0	0
255	255	0	0	255	255	0	0
255	255	0	0	255	255	0	0

8x8

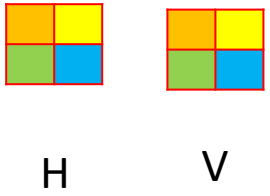
1	0	-1
1	0	-1
1	0	-1

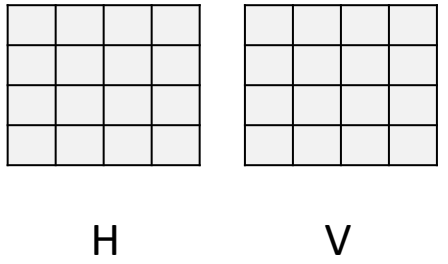
3x3

765	765	-765	-765	765	765
765	765	-765	-765	765	765
765	765	-765	-765	765	765
765	765	-765	-765	765	765
765	765	-765	-765	765	765
765	765	-765	-765	765	765

6x6

Spatial Ouput for Image Pyramids





Spatial Outputs: Summary

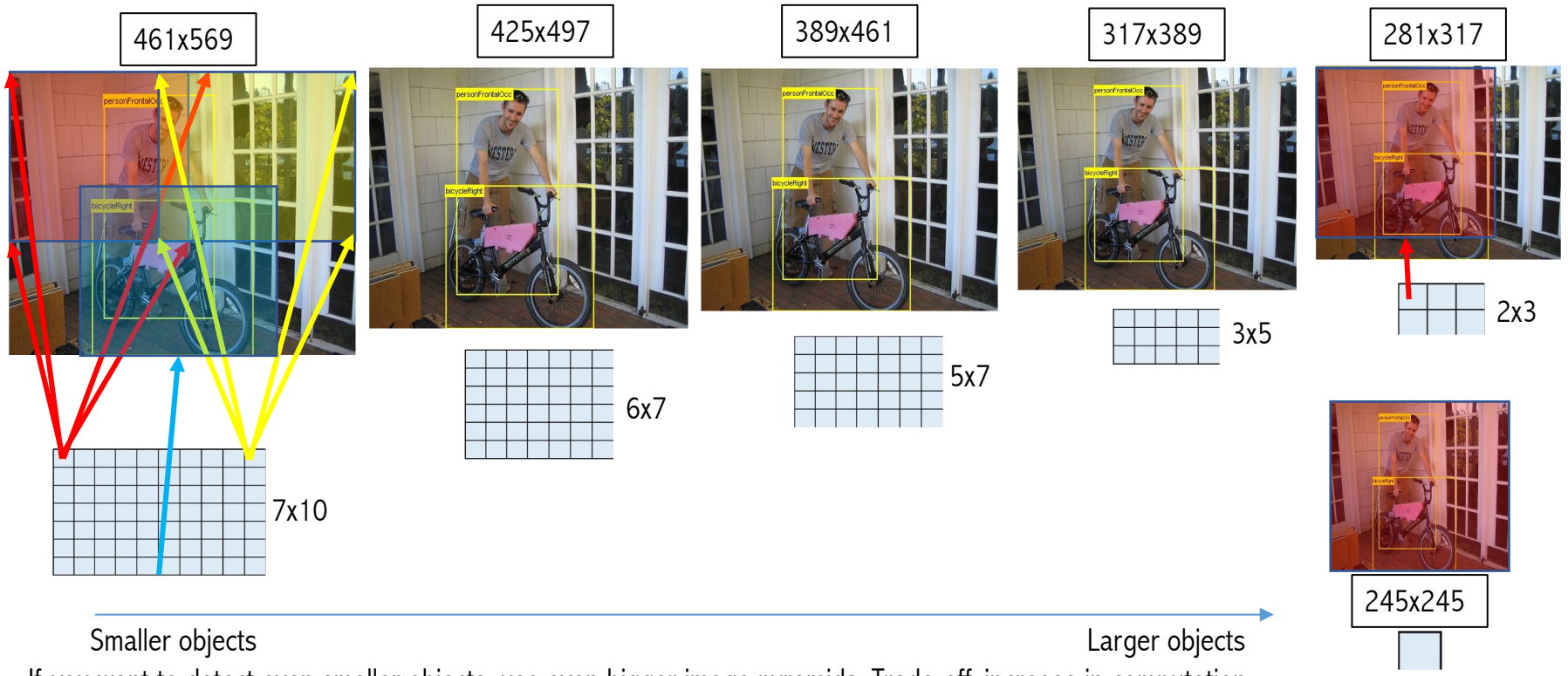
With Spatial Outputs, we can detect different objects at different locations of the image. Below figure shows a 2x3 Spatial Output for a sample image.

OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks

~~Sliding Window Crop~~ FC as Conv (No input size constraint) + Spatial Output + Image Pyramid

Resolution = 36

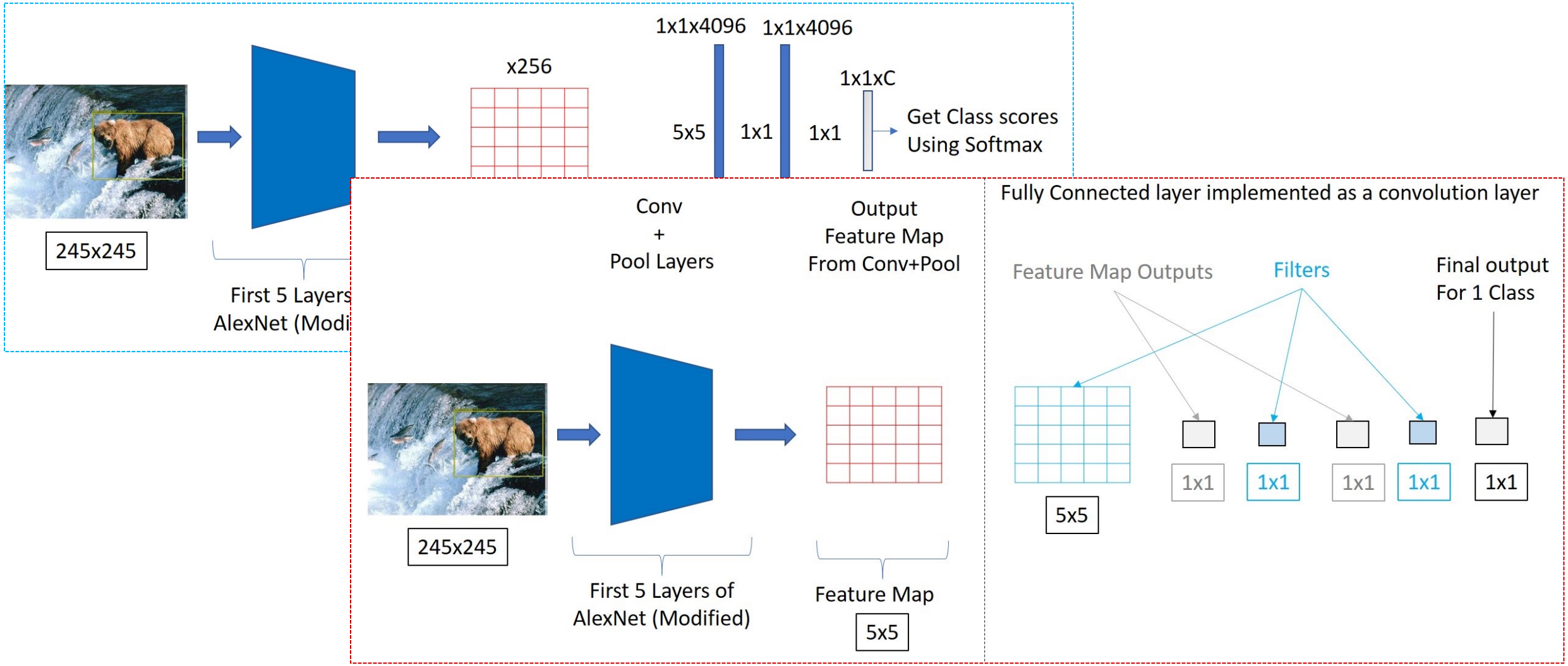
How to modify localization framework to convert FC as Conv?



1. Use the same localization network, without using the Sliding Window crops at different locations.
2. No input size constraint, be able to use the Image pyramids.
3. Use Image Pyramids, we will get the Spatial Output, which will give us detections at different locations of the image.
4. The entire network is using Convolution operations, it is way more efficient than taking crops.

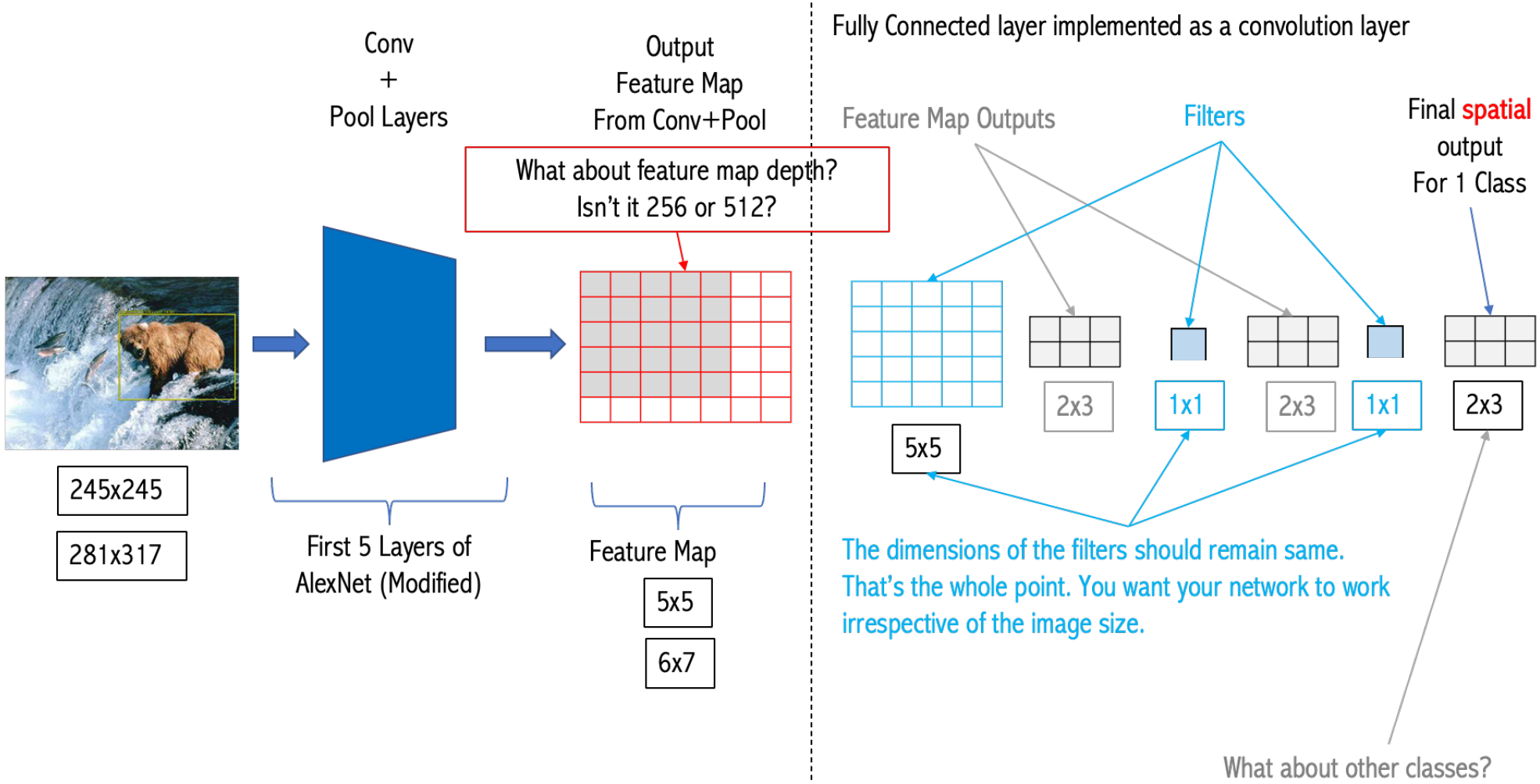
If you want to detect even smaller objects, use even bigger image pyramids. Trade-off, increase in computation

Overfeat Classification



OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks – Sermanet et al

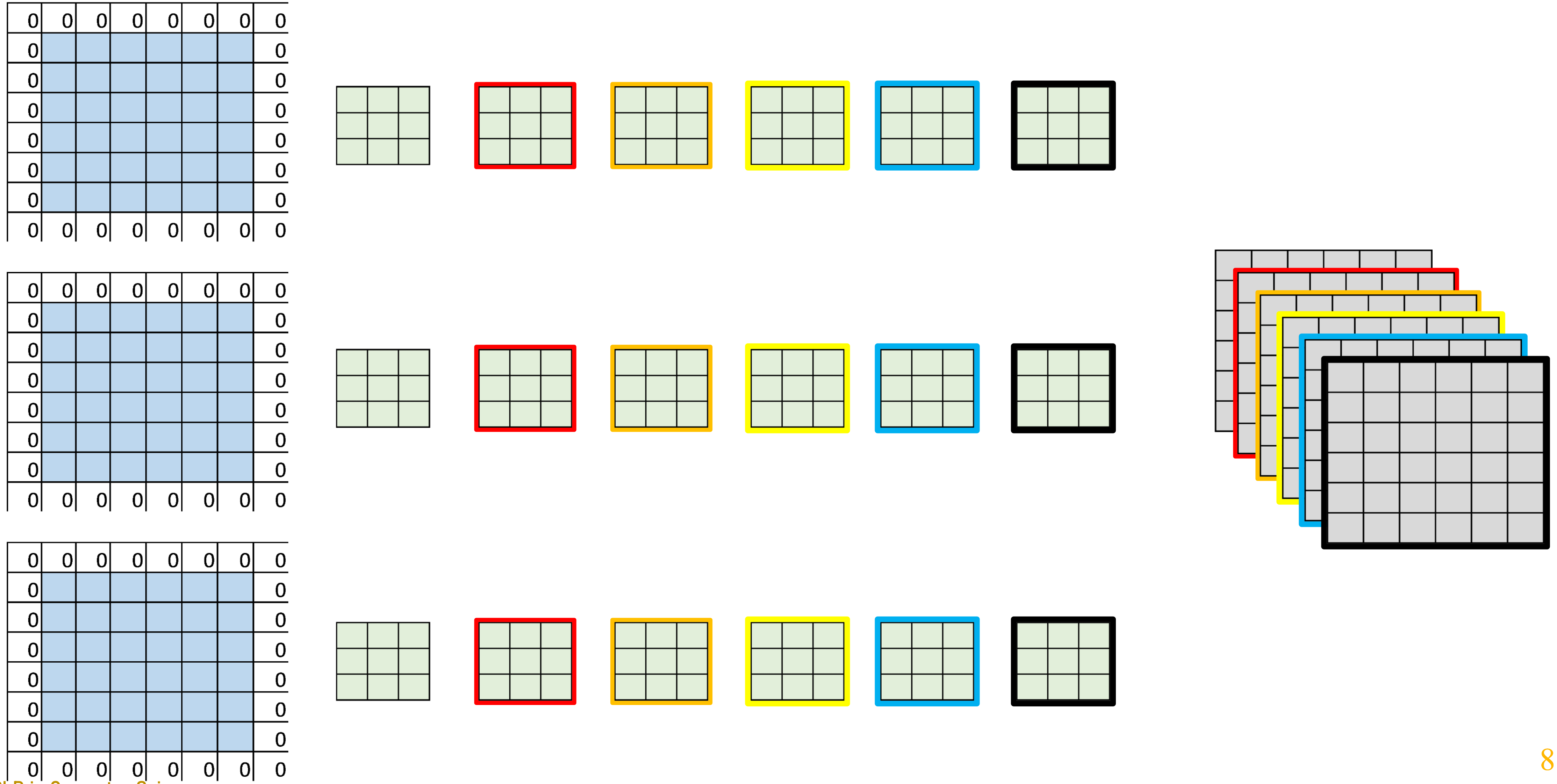
Overfeat Classification



OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks – Sermanet et al

Images Credit – Overfeat paper & https://towardsdatascience.com/object-localization-in-overfeat-5bb2f7328b62

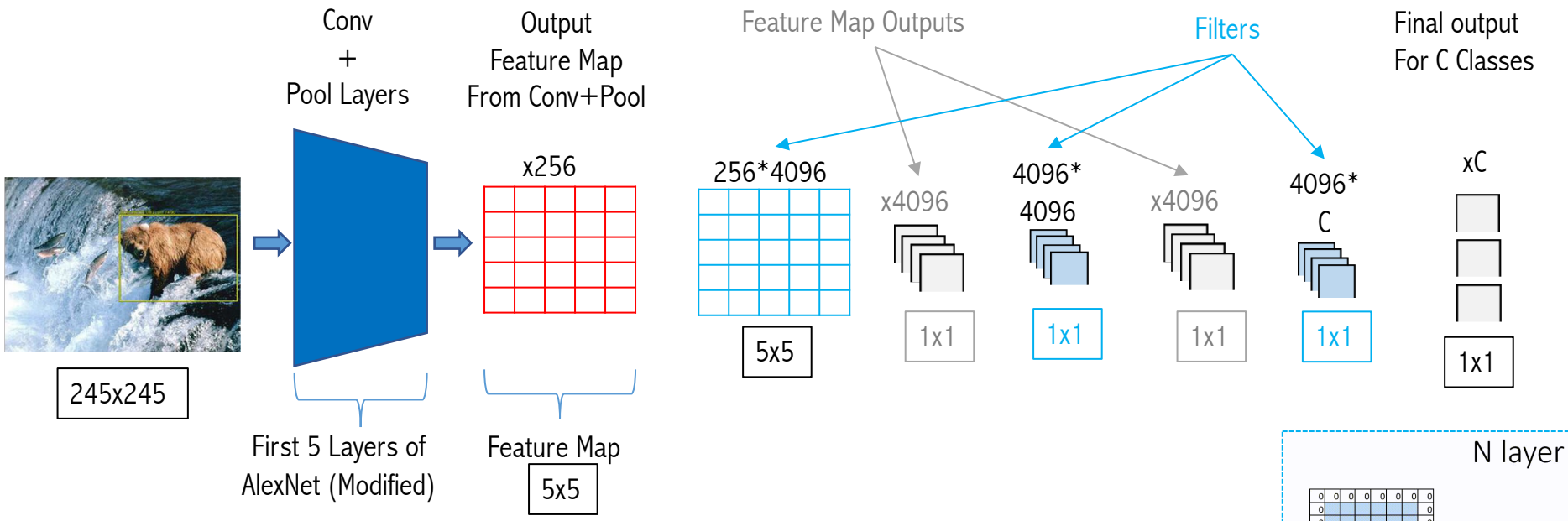
N layer Conv – M Feature Maps

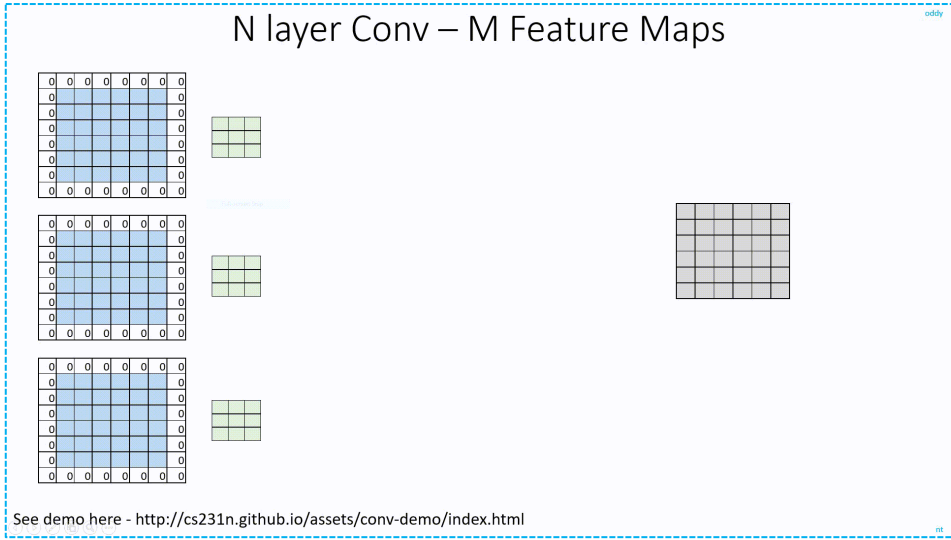


AI VIETNAM
All-in-One Course

Overfeat Classification

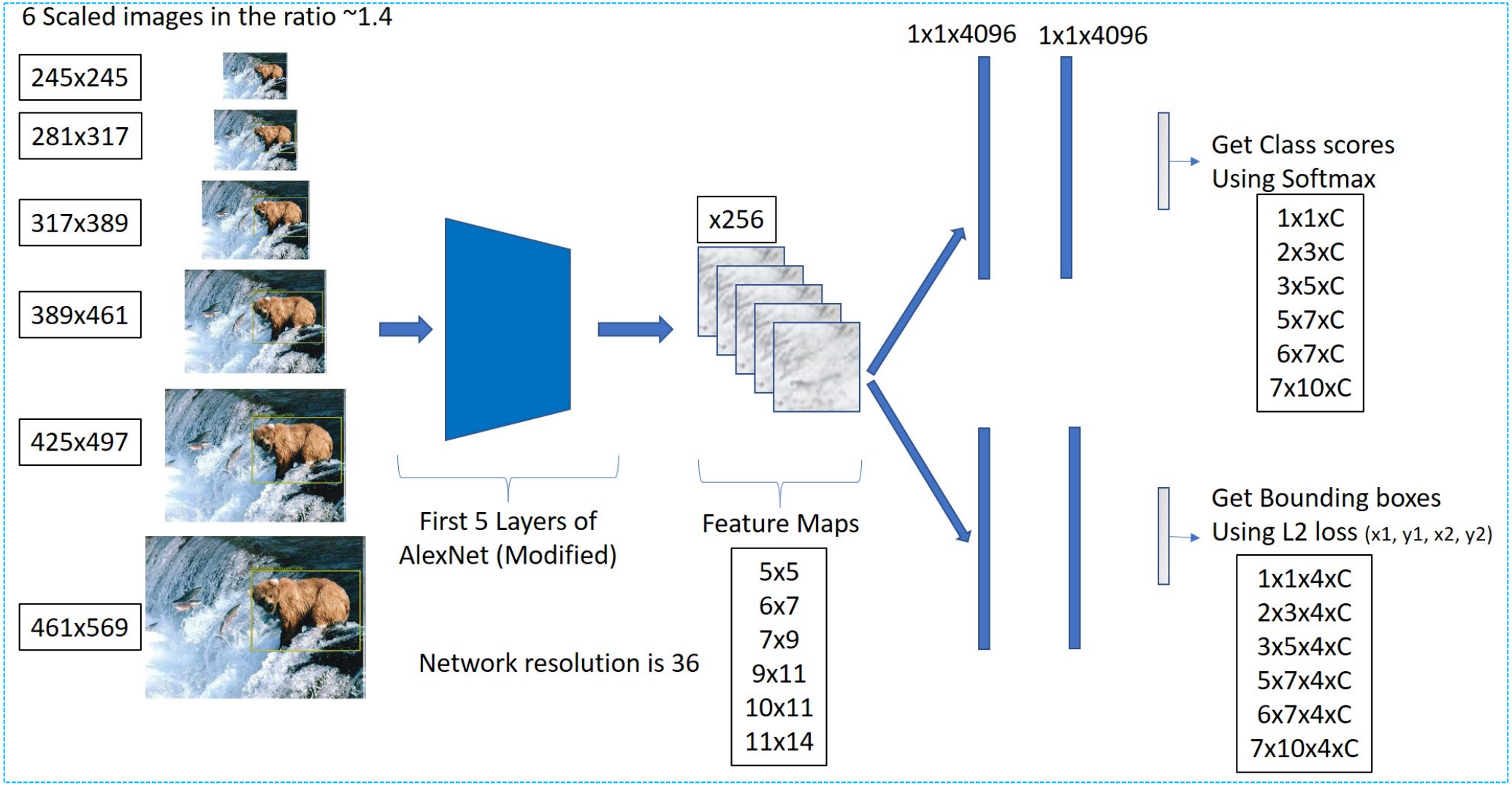
Fully Connected layer implemented as a convolution layer





OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks – Sermanet et al

Overfeat Classification



Outline

- Introduction to Object Detection
- Applications of Object Detection
- Object Detection Milestones
- Traditional Object Detectors
- CNNs for Image Classification
- Image Classifier to Object Detectors with CNN
- CNN Limitations and Spatial Outputs
- Assignment


Thank
you