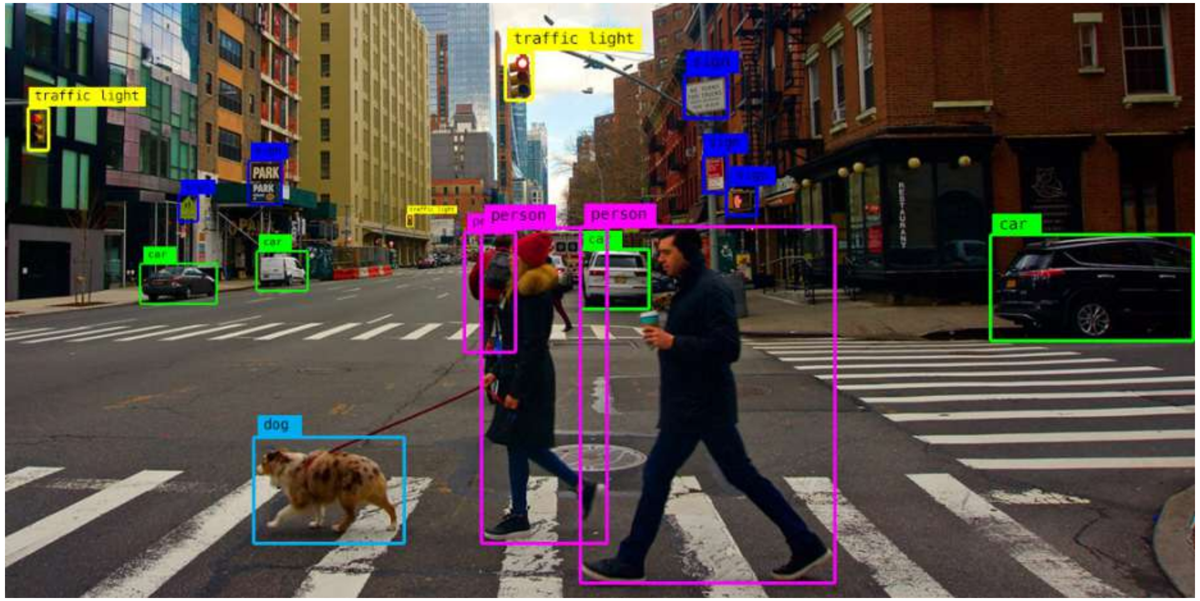
AI VIET NAM
@aivietnam.edu.vn

Object Detection

(Part 2: CNN-based Algorithms)



[Code & Data]

Vinh Dinh Nguyen - PhD in Computer Science

Outline

- CNN Limitations
- Region Based Convolutional Neural Networks
- Spatial Pyramid Pooling
- Fast R-CNN
- Faster RCNN
- YOLOv1-v2

AI VIET NAM
@aivietnam.edu.vn

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,...

Classification Pipeline

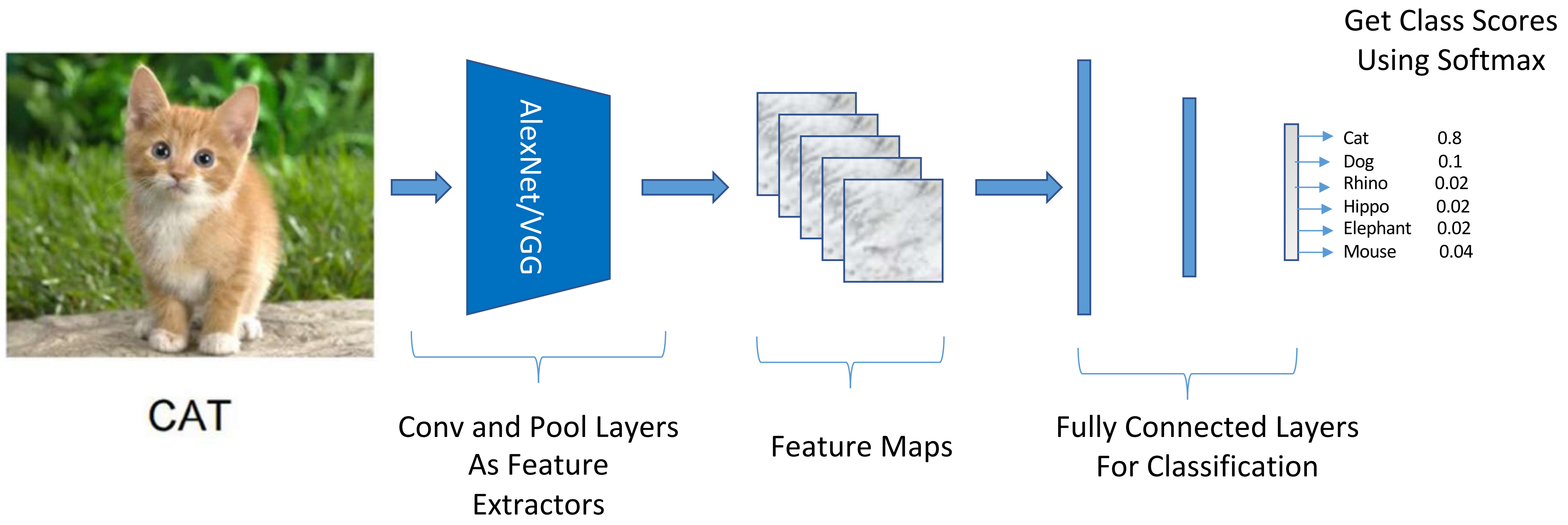
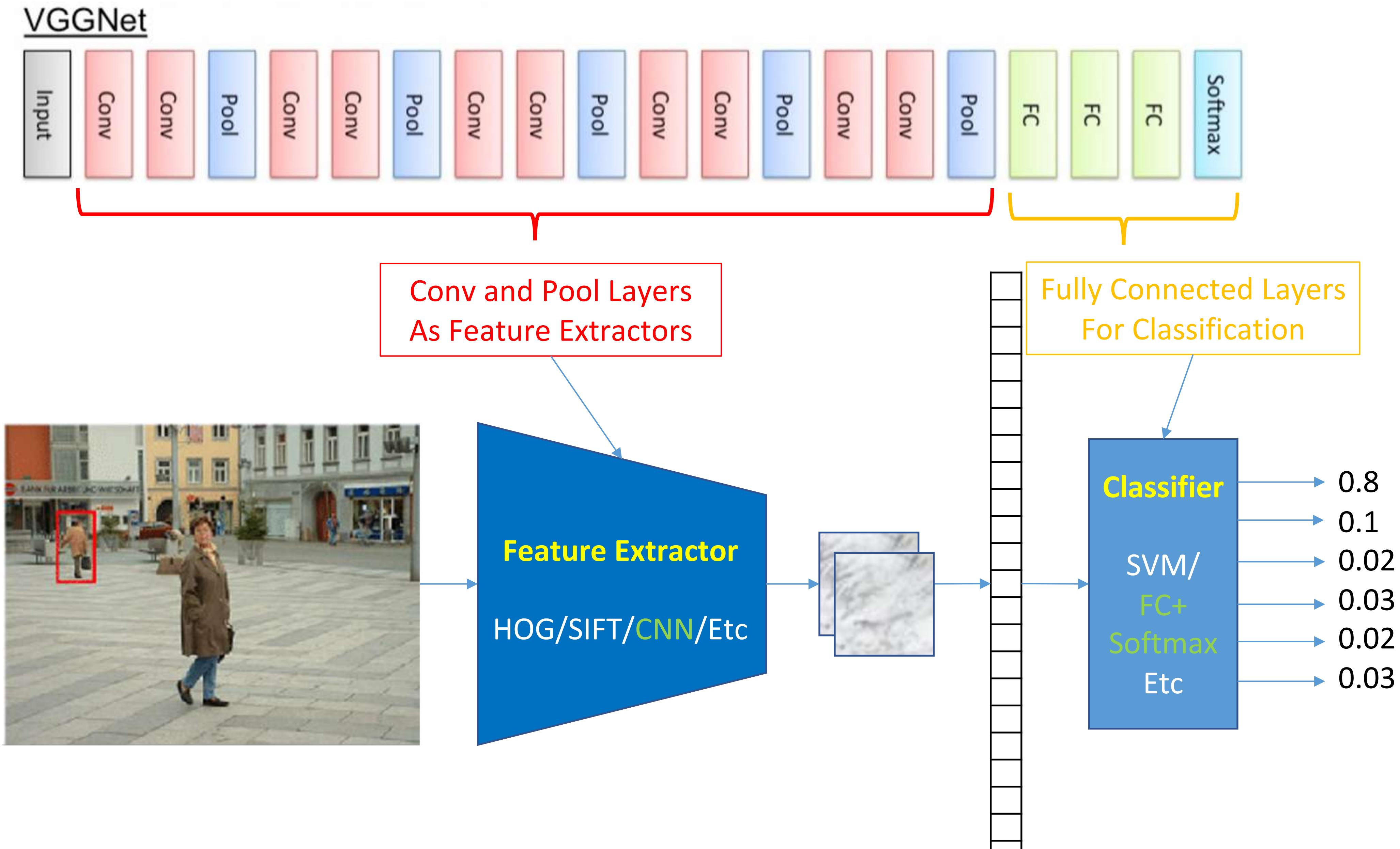


Image Credit - http://host.robots.ox.ac.uk/pascal/VOC/voc2012/examples/index.html

Classification Pipeline



AI VIET NAM
@aivietnam.edu.vn

Ideas for *Localization* using ConvNets

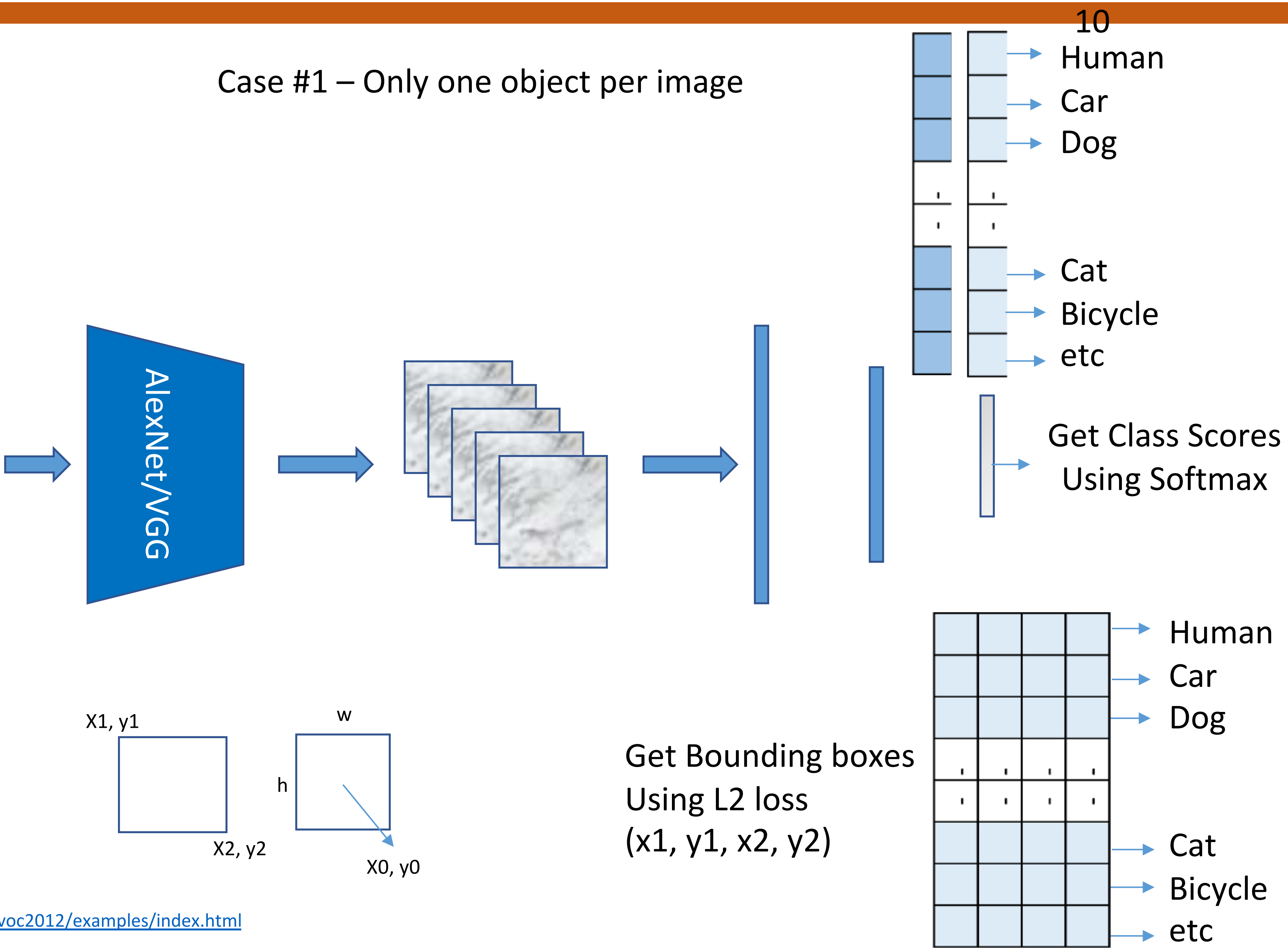
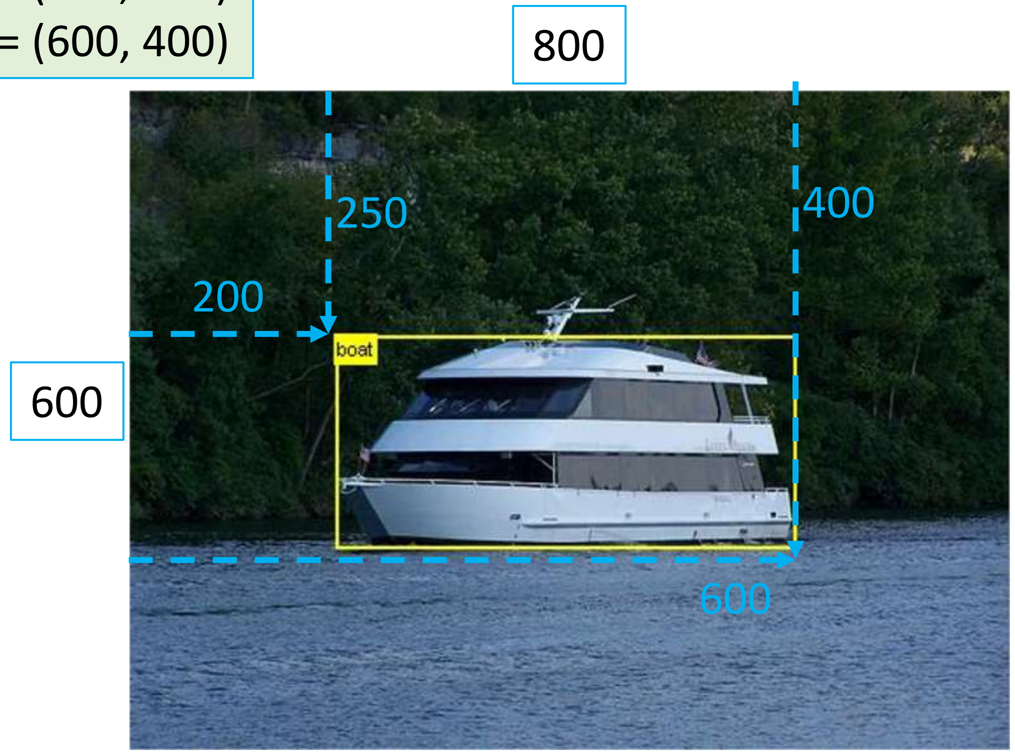
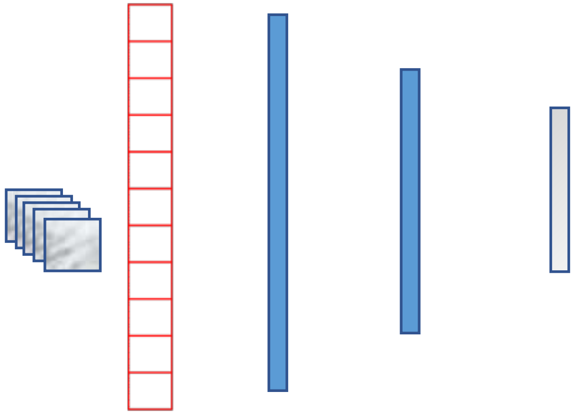


Image Credit - http://host.robots.ox.ac.uk/pascal/VOC/voc2012/examples/index.html

Bounding Box Regression Training

(x1,y1) = (200, 250)
(x2,y2) = (600, 400)

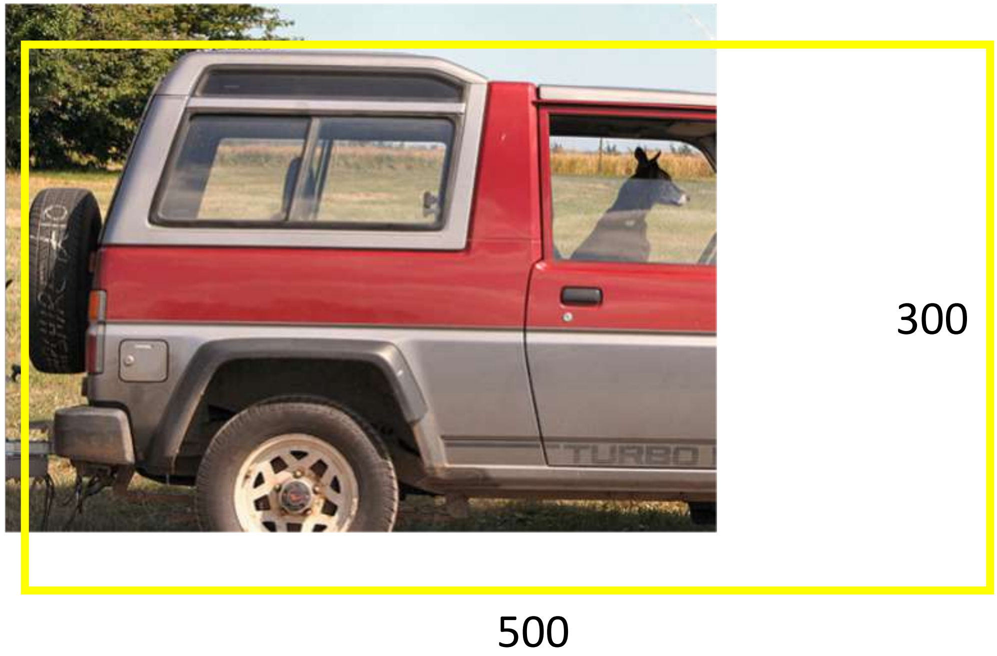


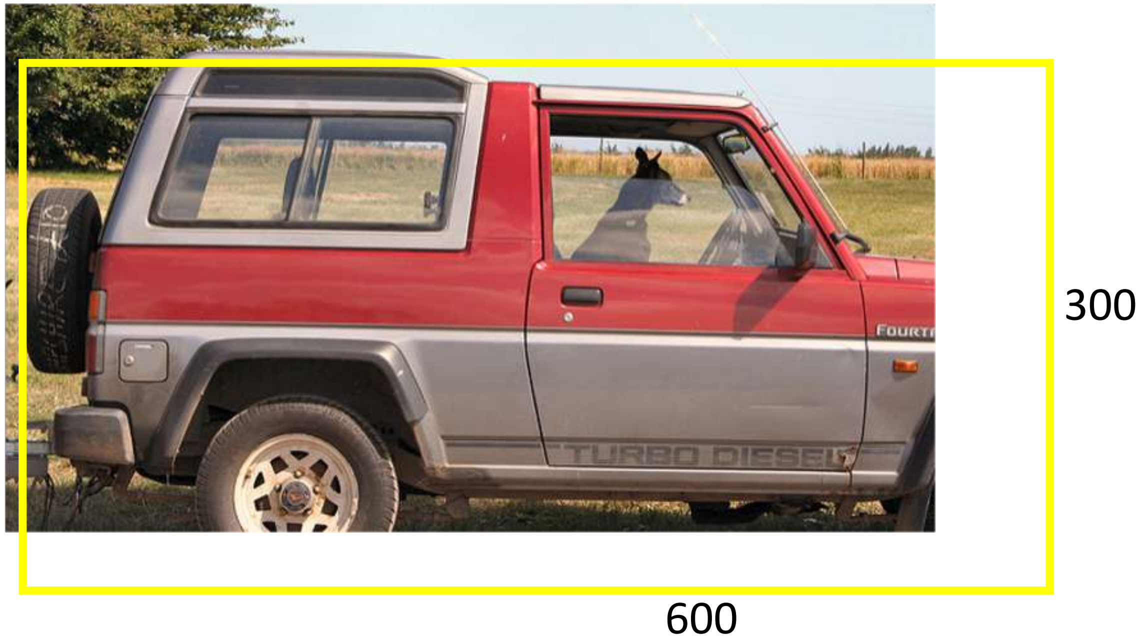


	x1	y1	x2	y2	L2 Loss				
Expected	200	250	600	400					
Prediction	0	0	800	600	$(200-0)^2$	$(250-0)^2$	$(600-800)^2$	$(400-600)^2$	182500
	100	150	700	450	$(200-100)^2$	$(250-150)^2$	$(600-700)^2$	$(400-450)^2$	32500
	210	245	590	405	$(200-210)^2$	$(250-245)^2$	$(600-590)^2$	$(400-405)^2$	250
	200	250	600	400	$(200-200)^2$	$(250-250)^2$	$(600-600)^2$	$(400-400)^2$	0

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

About Bounding Boxes





Ideas for *Localization* using ConvNets

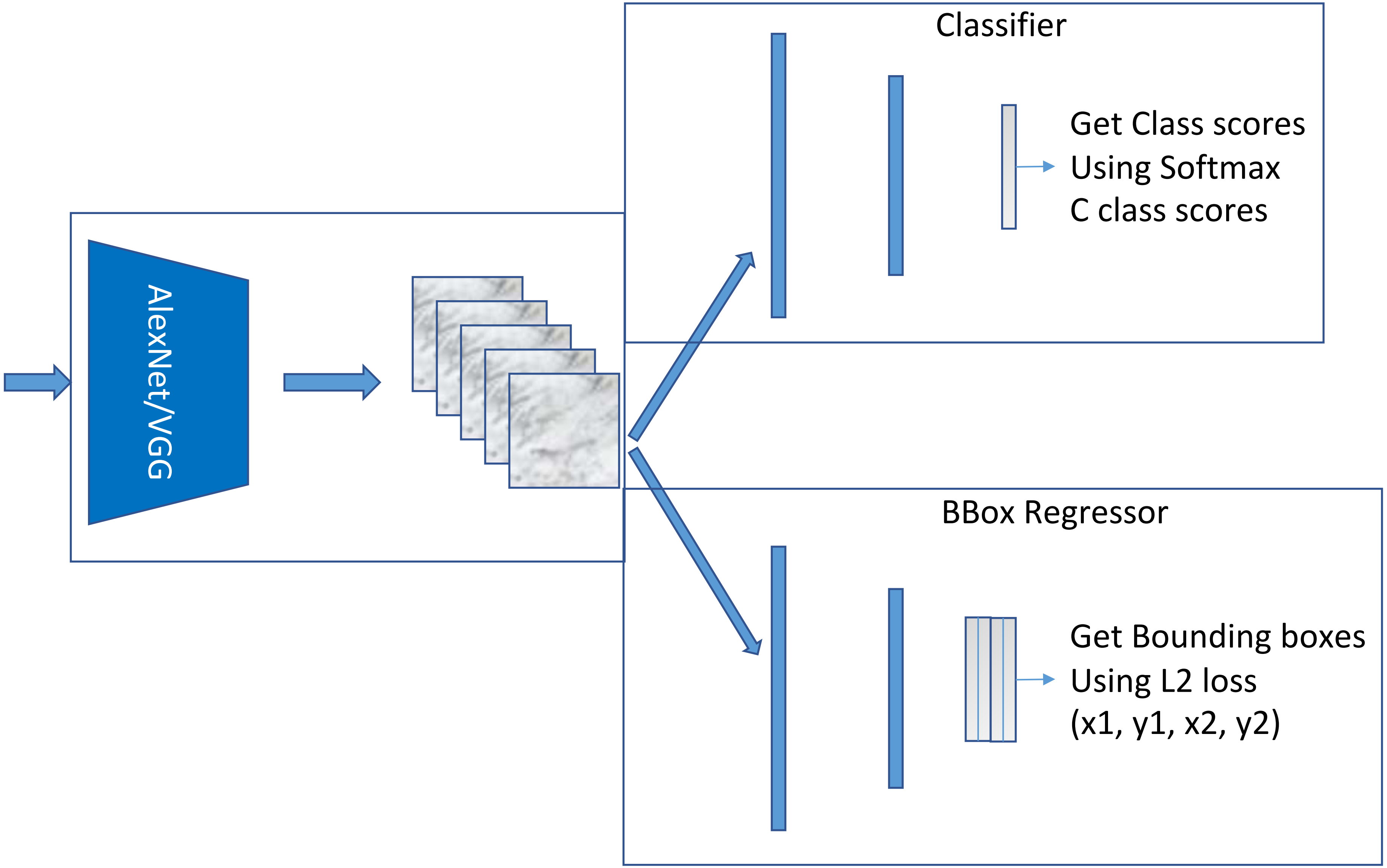
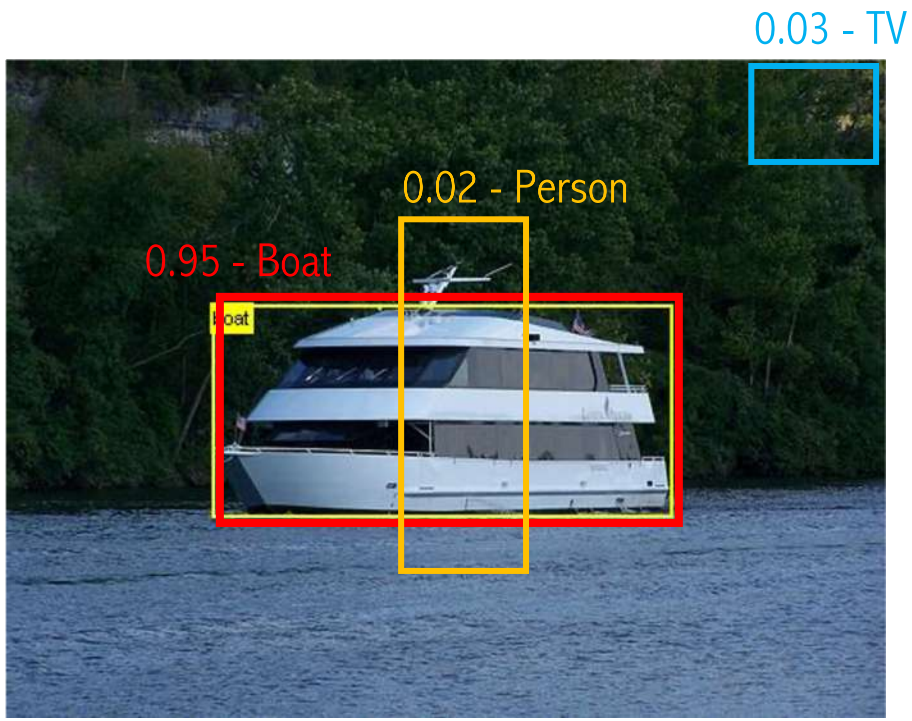
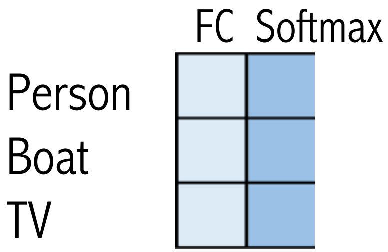


Image Credit - http://host.robots.ox.ac.uk/pascal/VOC/voc2012/examples/index.html

AI VIET NAM
@aivietnam.edu.vn

Combining Results





Class	Conf	Bbox coordinates			
Person	0.02	380	200	430	400
Boat	0.95	210	245	590	405
TV	0.03	700	10	790	100

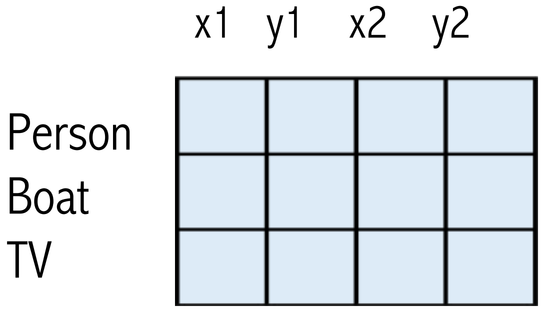
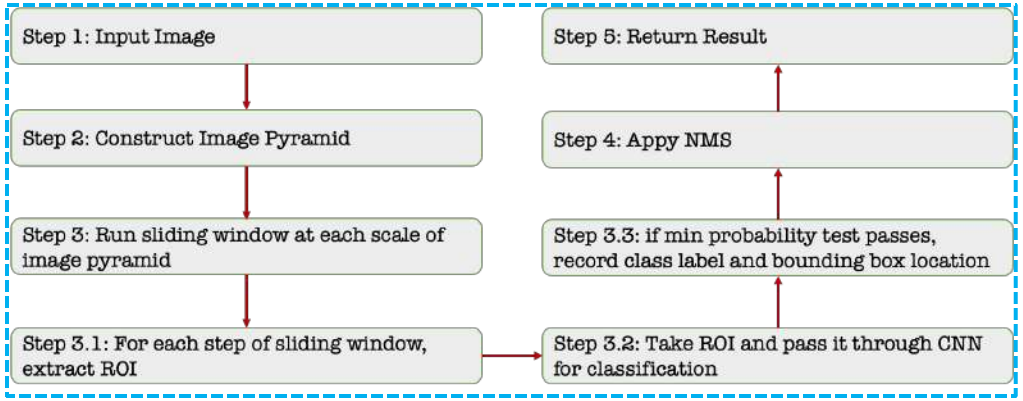
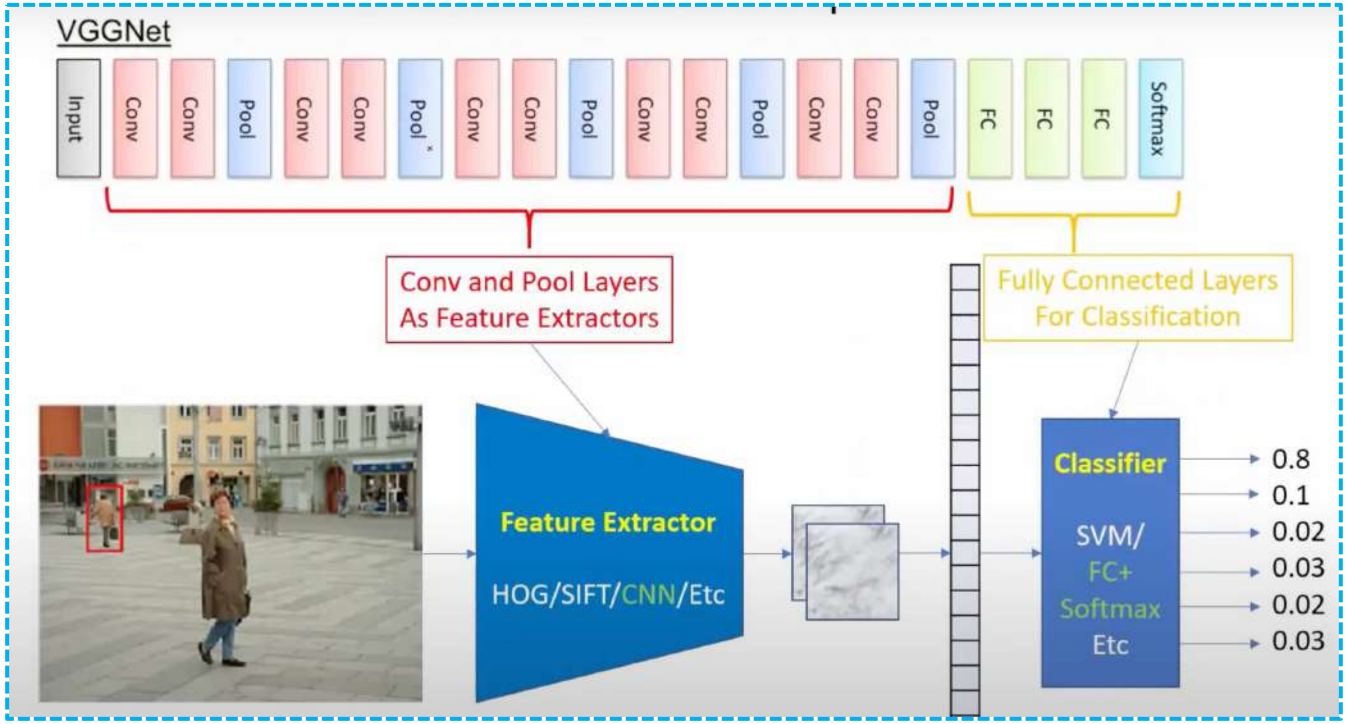
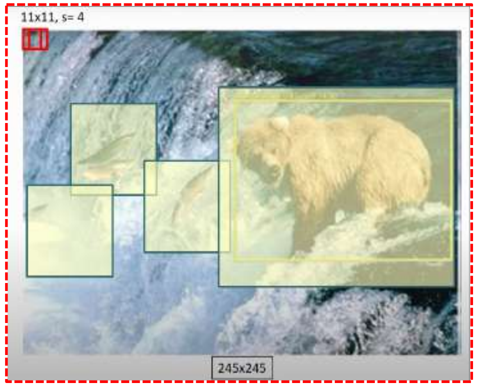


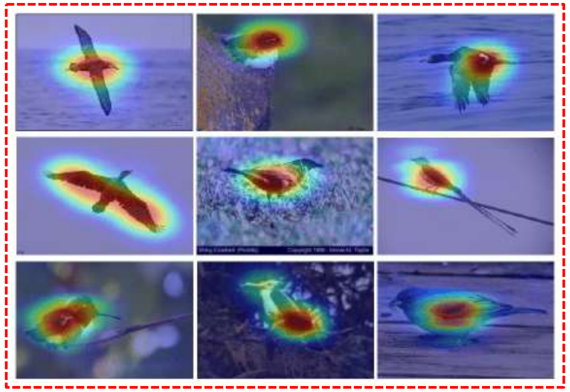
Image Credit - http://host.robots.ox.ac.uk/pascal/VOC/voc2012/examples/index.html

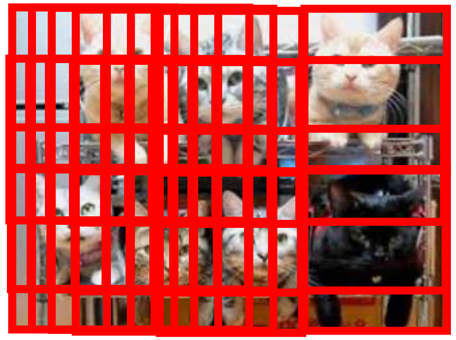
Convolutional Neural Network’s Limitations









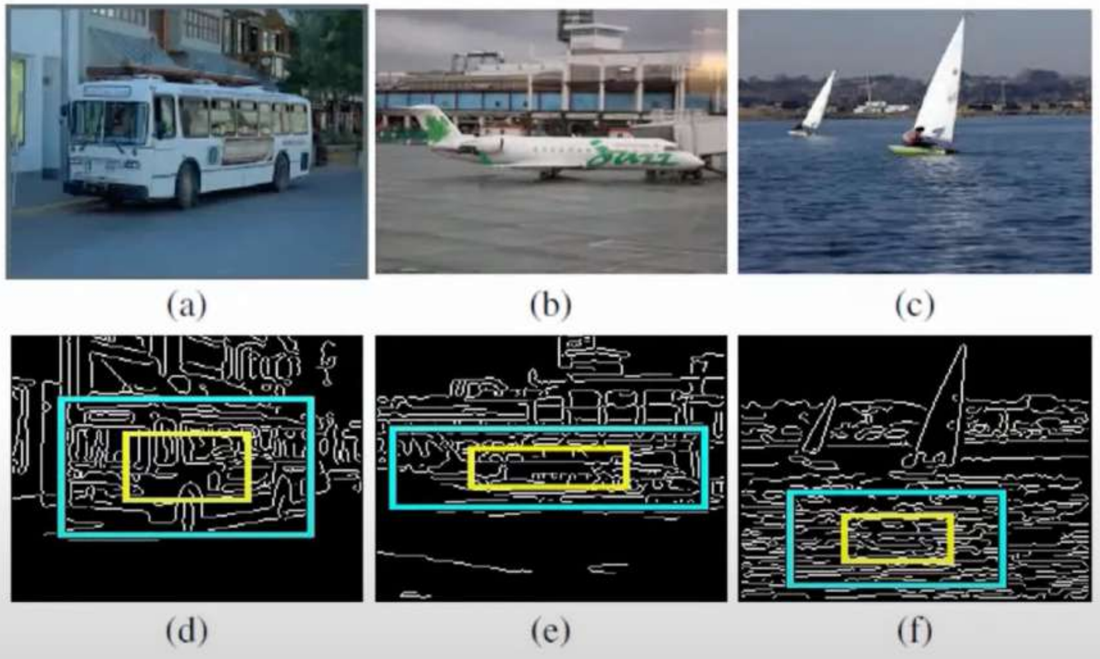


We need to focus on object only, not entire image

AI VIET NAM
@aivietnam.edu.vn

Region Proposal Methods

Colour Contrast



Edge Density

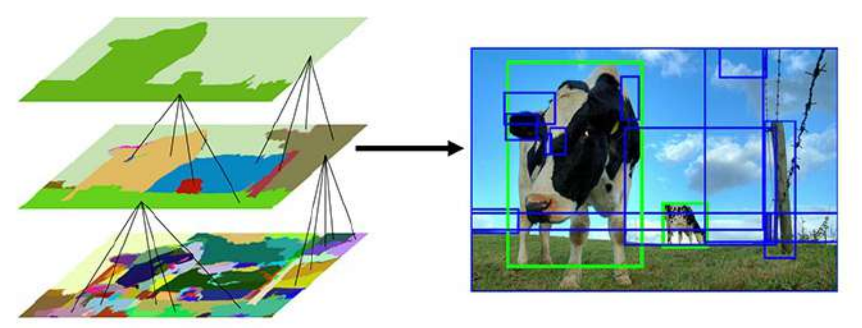


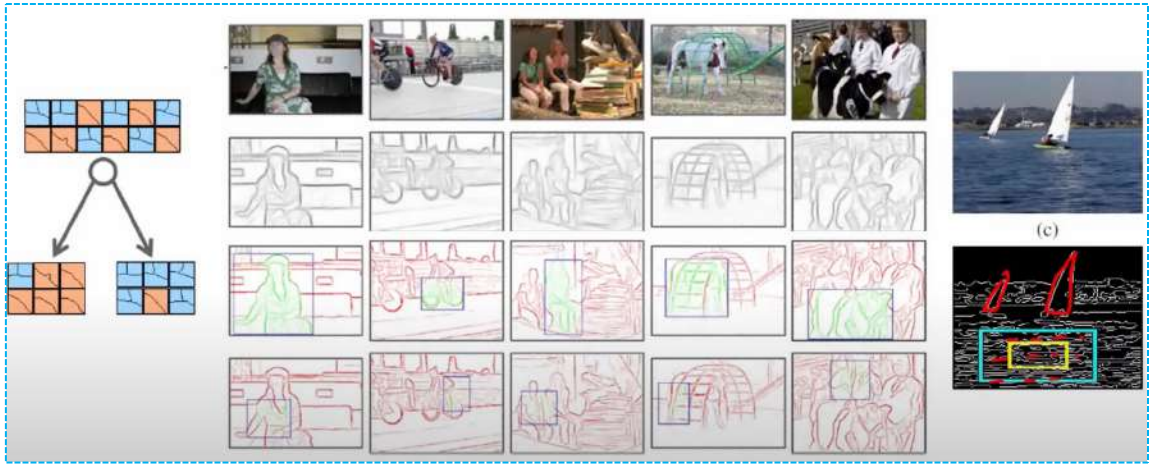
Image Segmentation

Method		Time	Repeatability	Recall	Detection	mAP
Objectness[1]	O	3	·	★	·	25.0/25.4
CPMC[4]	C	250	-	★★	★	29.9/30.7
Endres2010[9]	E	100	-	★★	★★	31.2/31.6
Sel.Search[30]	SS	10	★★	★★★	★★	31.7/32.3
Rahtu2011[24]	R1	3	·	·	★	29.6/30.4
Rand.Prim[22]	RP	1	★	★	★	30.5/30.9
Bing[6]	B	0.2	★★★	★	·	21.8/22.4
MCG[3]	M	30	★	★★★	★★	**32.4/32.7**
Ranta.2014[25]	R4	10	★★	·	★	30.7/31.3
EdgeBoxes[33]	EB	0.3	★★	★★★	★★	31.8/32.2
Uniform	U	0	·	·	·	16.6/16.9
Gaussian	G	0	·	·	★	**27.3/28.0**
SlidingWindow	SW	0	★★★	·	·	20.7/21.5
Superpixels	SP	1	★	·	·	11.2/11.3

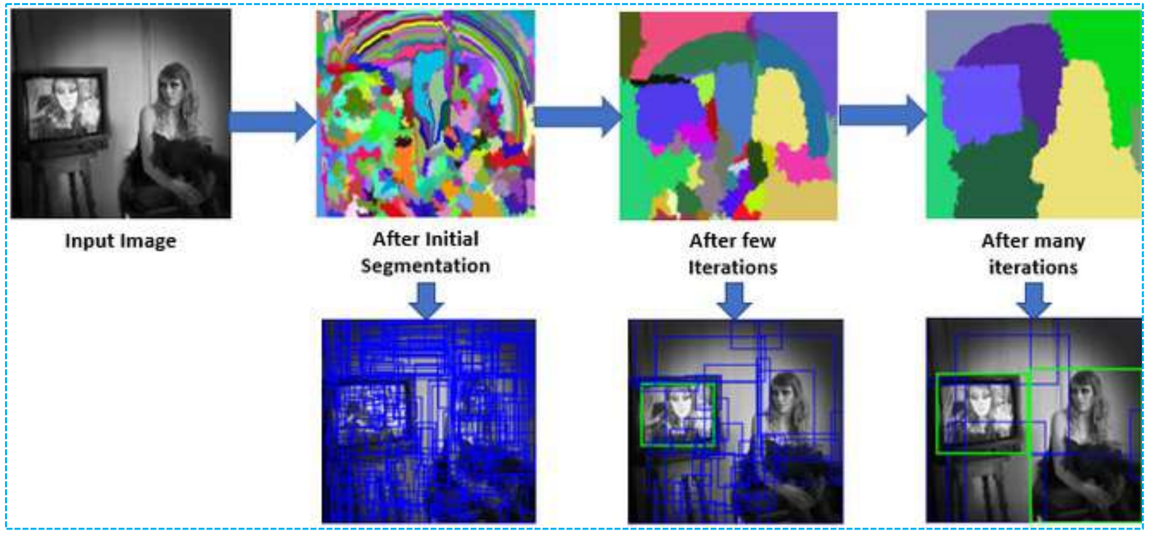
Jan Hosang, Rodrigo Benenson, Bernt Schiele, "How good are detection proposals, really?", 2014

AI VIET NAM
@aivietnam.edu.vn

Edge Boxes & Selective Search



Zitnick, C.L., Dollár, P. (2014). Edge Boxes: Locating Object Proposals from Edges.



How to Apply Edge Boxes / Selective Search for CNN?

AI VIET NAM
@aivietnam.edu.vn

Region Proposal Algorithms

Region proposal algorithms identify prospective objects in an image using segmentation. In segmentation, we group adjacent regions which are similar to each other based on some criteria such as color, texture etc.

RBG Image

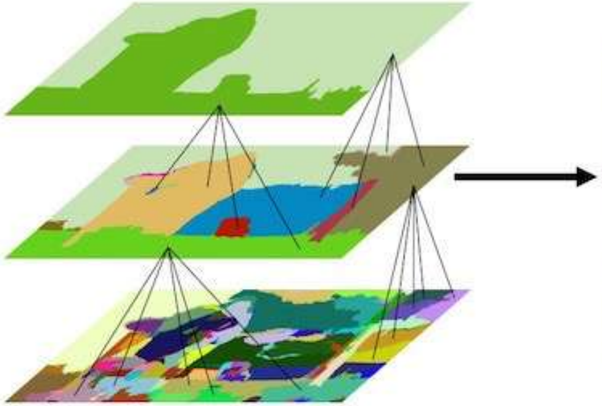
Segmentation Result

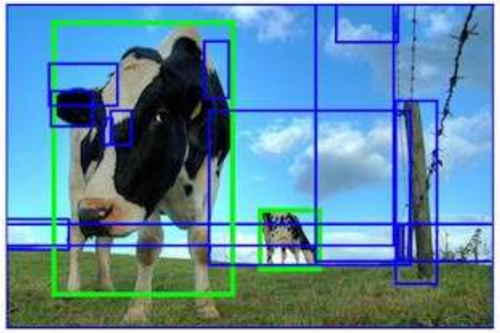
Can we use segmented parts in this image as region proposals? The answer is no and there are two reasons why we cannot do that

1.Most of the actual objects in the original image contain 2 or more segmented parts
2.Region proposals for occluded objects such as the plate covered by the cup or the cup filled with coffee cannot be generated using this method

Selective Search

Selective search uses oversegments from Felzenszwalb and Huttenlocher's method as an initial seed. An oversegmented image looks like this.





Selective Search uses 4 similarity measures based on color, texture, size and shape compatibility.

1. Add all bounding boxes corresponding to segmented parts to the list of regional proposals
2. Group adjacent segments based on similarity
3. Go to step 1

25×3 = 75-dimensional color descriptor

$$s_{color}(r_i, r_j) = \sum_{k=1}^{n} min(c_i^k, c_j^k)$$

10x8x3 = 240-dimensional feature descriptor

$$s_{texture}(r_i, r_j) = \sum_{k=1}^{n} min(t_i^k, t_j^k)$$

$$s_{size}(r_i, r_j) = 1 - \frac{size(r_i) + size(r_j)}{size(im)}$$

$$s_{fill}(r_i, r_j) = 1 - \frac{size(BB_{ij}) - size(r_i) - size(r_j)}{size(im)}$$

Final Similarity

$$\boxed{s(r_i, r_j) = a_1 s_{color}(r_i, r_j) + a_2 s_{texture}(r_i, r_j) + a_3 s_{size}(r_i, r_j) + a_4 s_{fill}(r_i, r_j)}$$

AI VIET NAM
@aivietnam.edu.vn

Selective Search

Selective search uses oversegments from Felzenszwalb and Huttenlocher's method as an initial seed. An oversegmented image looks like this.

Dogs: top 250 region proposals

Breakfast Table: top 200 region proposals

1. Add all bounding boxes corresponding to segmented parts to the list of regional proposals
2. Group adjacent segments based on similarity
3. Go to step 1

https://learnopencv.com/selective-search-for-object-detection-cpp-python/

Selective Search

```python
def genereate_region_by_selective_search(img_file):

    img_ss = cv2.imread(img_file)


    ss = cv2.ximgproc.segmentation.createSelectiveSearchSegmentation()
    ss.setBaseImage(img_ss)
    ss.switchToSelectiveSearchFast()
    # "... extract around 2000 region proposals (we use selective search's "fast mode" in all experiments)."
    rects = ss.process()

    print('Found',len(rects),'boxes...')
    for i, rect in (enumerate(rects)):
      if i>2000:
        break
      x, y, w, h = rect
      cv2.rectangle(img_ss, (x, y), (x+w, y+h), (100, 255, 100), 1)

    cv2_imshow(img_ss)
        # close image show window
```

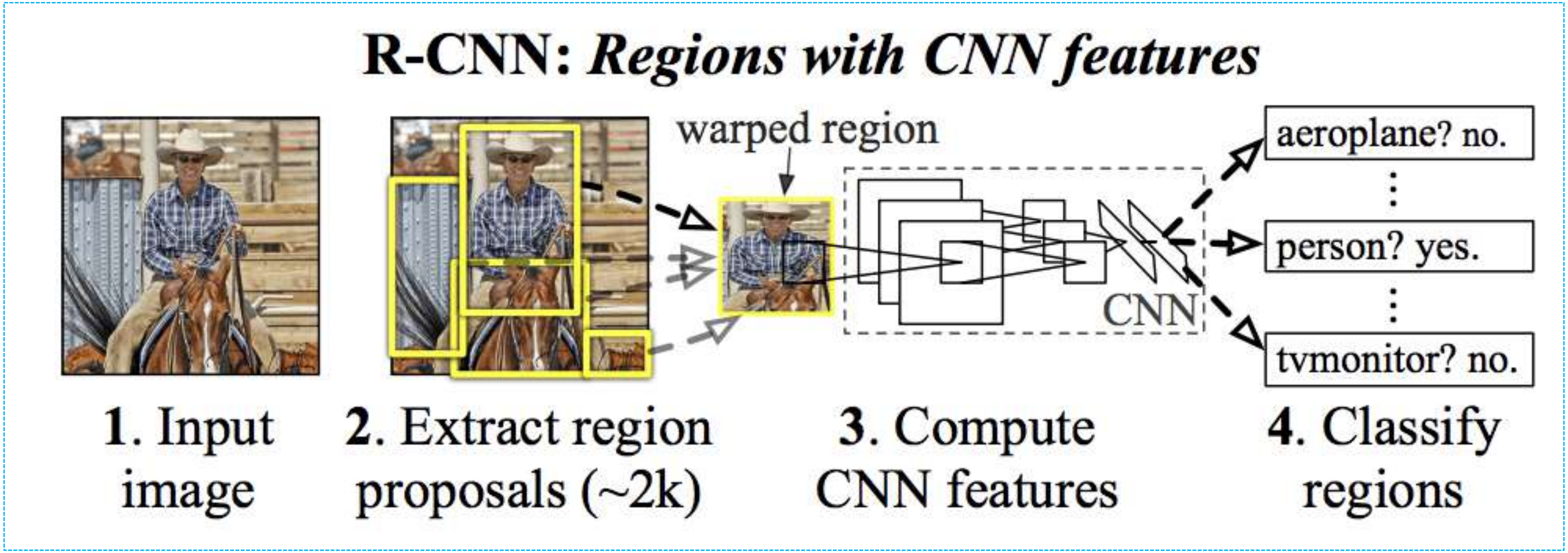
https://learnopencv.com/selective-search-for-object-detection-cpp-python/

Outline

- CNN Limitations
- Region Based Convolutional Neural Networks
- Spatial Pyramid Pooling
- Fast R-CNN
- Faster RCNN
- YOLOv1-v2

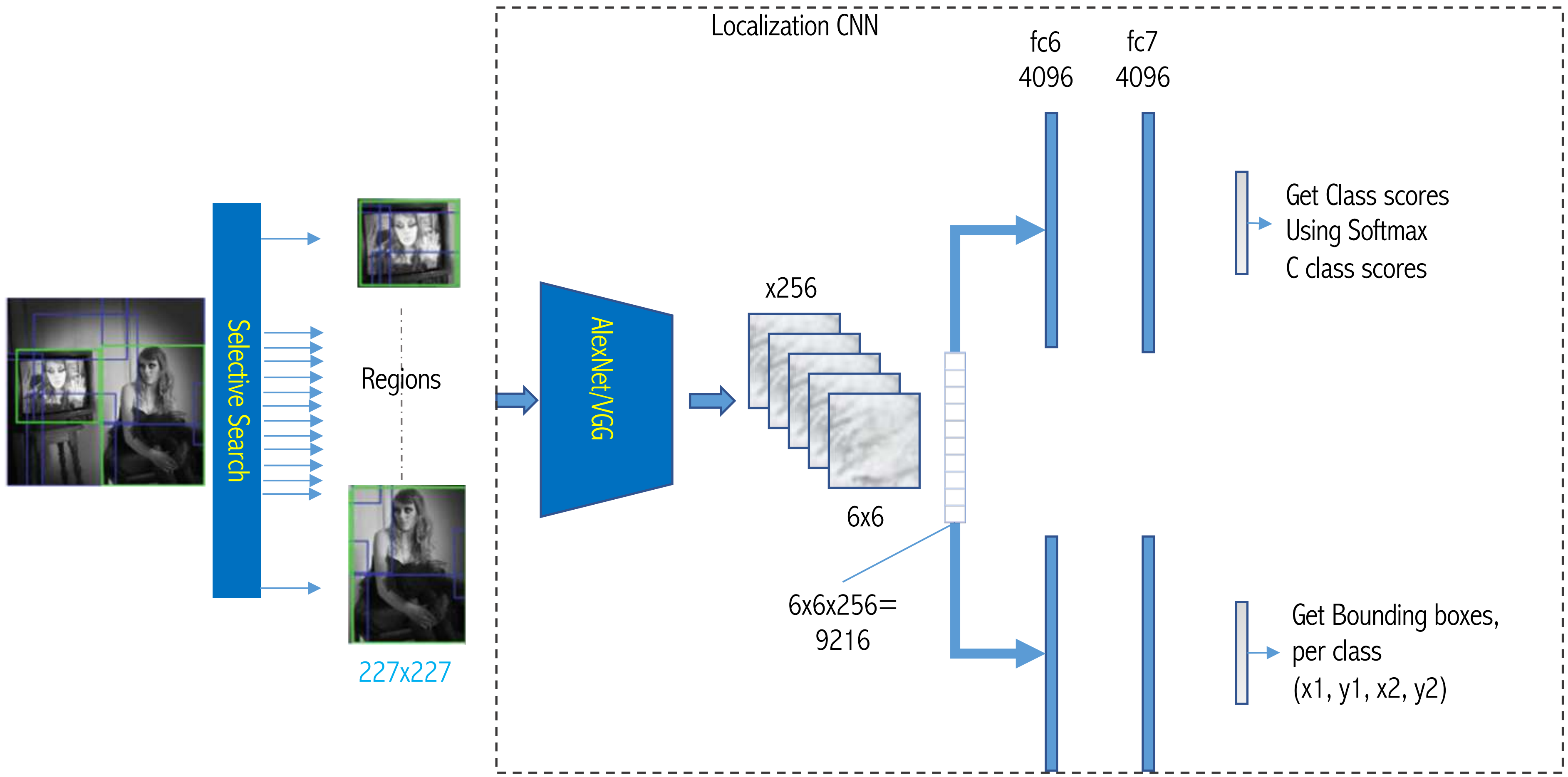
AI VIET NAM
@aivietnam.edu.vn

R-CNN



Ross Girshick, et al. from UC Berkeley titled "Rich feature hierarchies for accurate object detection and semantic segmentation."

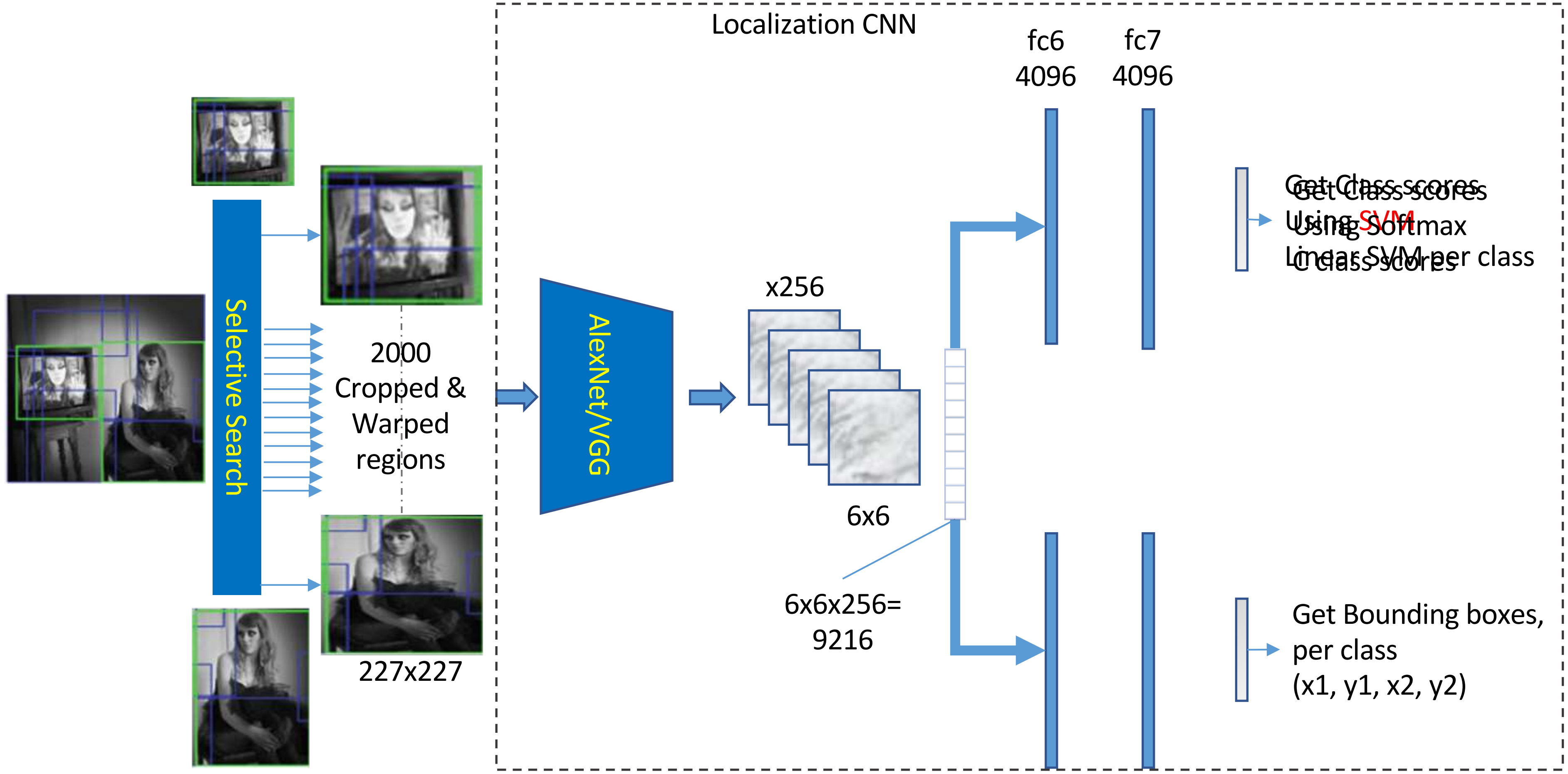
RCNN - Region proposals with CNNs



Rich feature hierarchies for accurate object detection and semantic segmentation - Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik

AI VIET NAM
@aivietnam.edu.vn

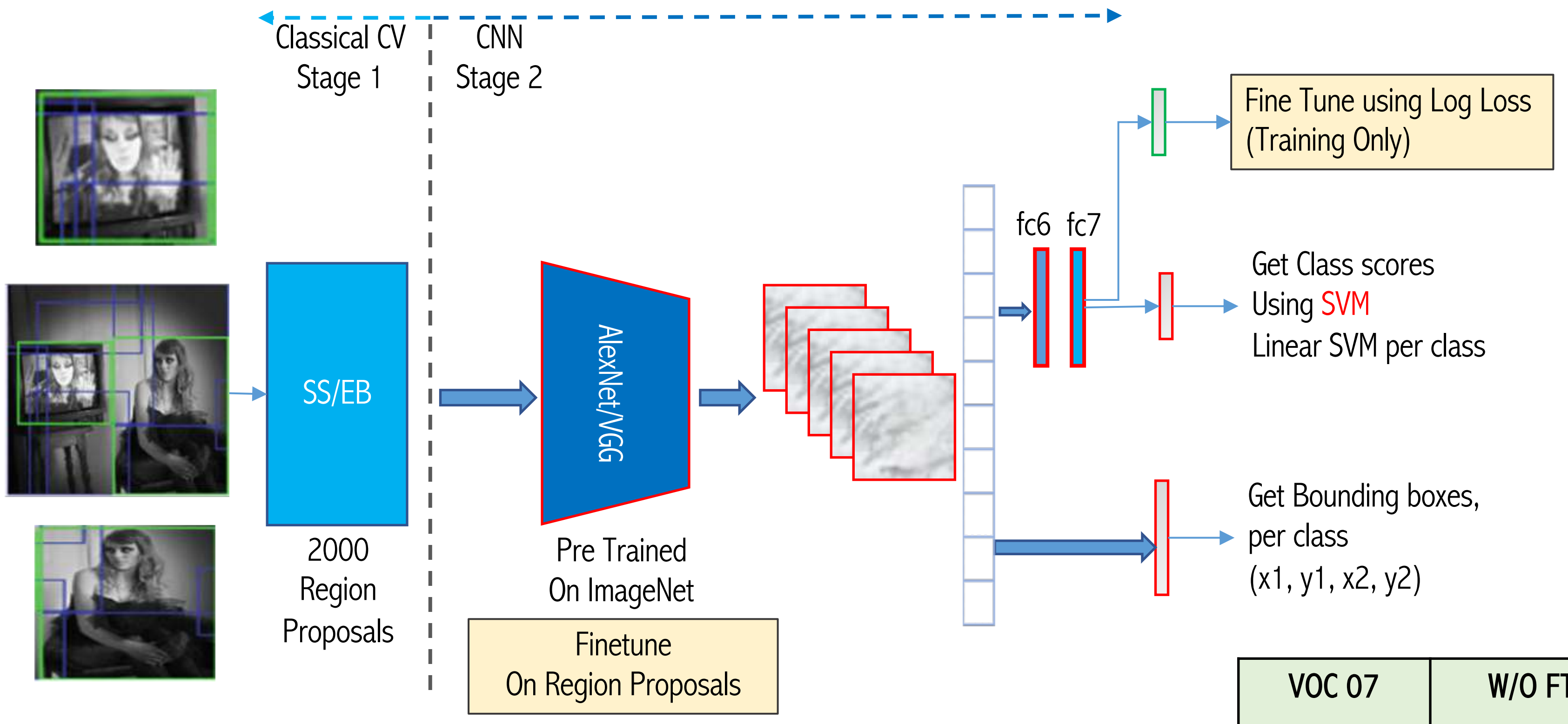
RCNN - Region proposals with CNNs



Rich feature hierarchies for accurate object detection and semantic segmentation - Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik

AI VIET NAM
@aivietnam.edu.vn

RCNN - Region proposals with CNNs



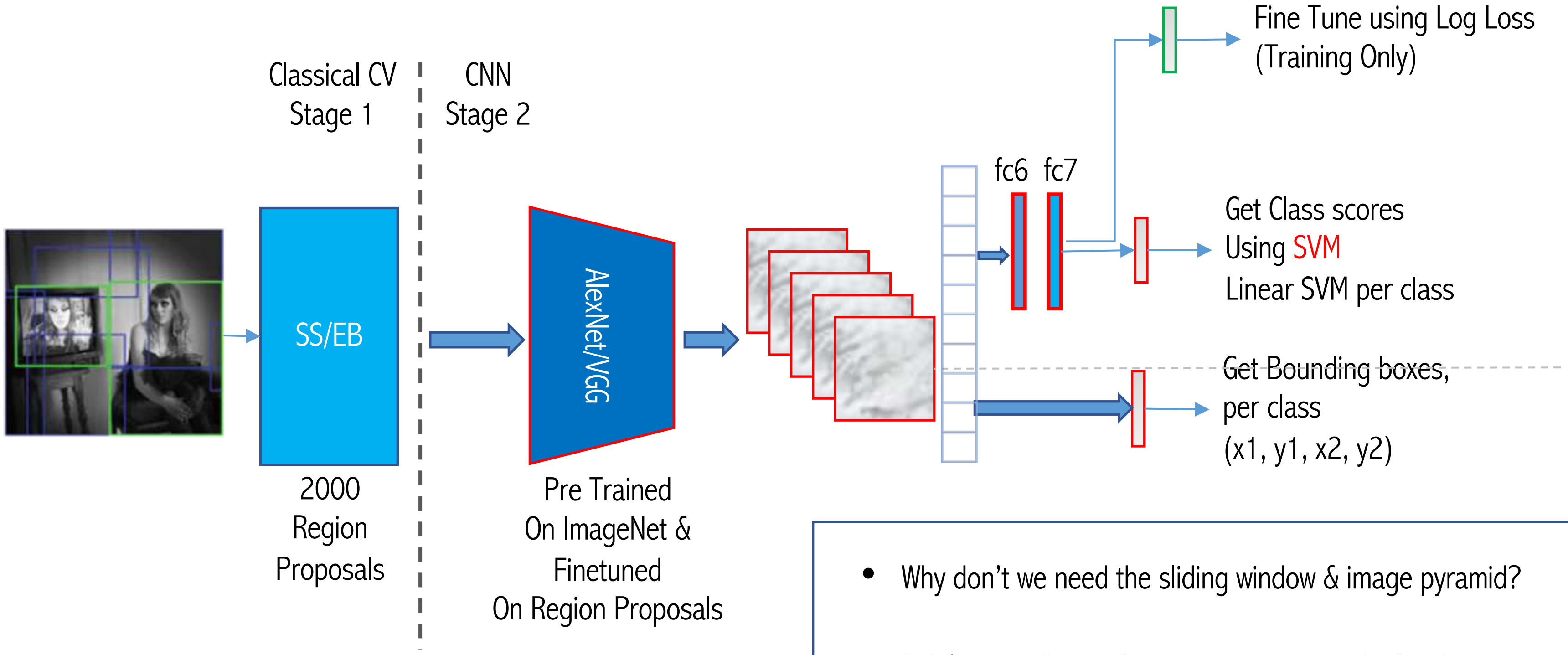
VOC 07	W/O FT	W FT
pool5	44.2	47.3
fc6	46.2	53.1
fc7	44.7	54.2

Alex Net
- Before finetune: 44%
- After finetune: 54%
- Adding bounding box regression: 58%
    - VGG: 66%

Rich feature hierarchies for accurate object detection and semantic segmentation - Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik

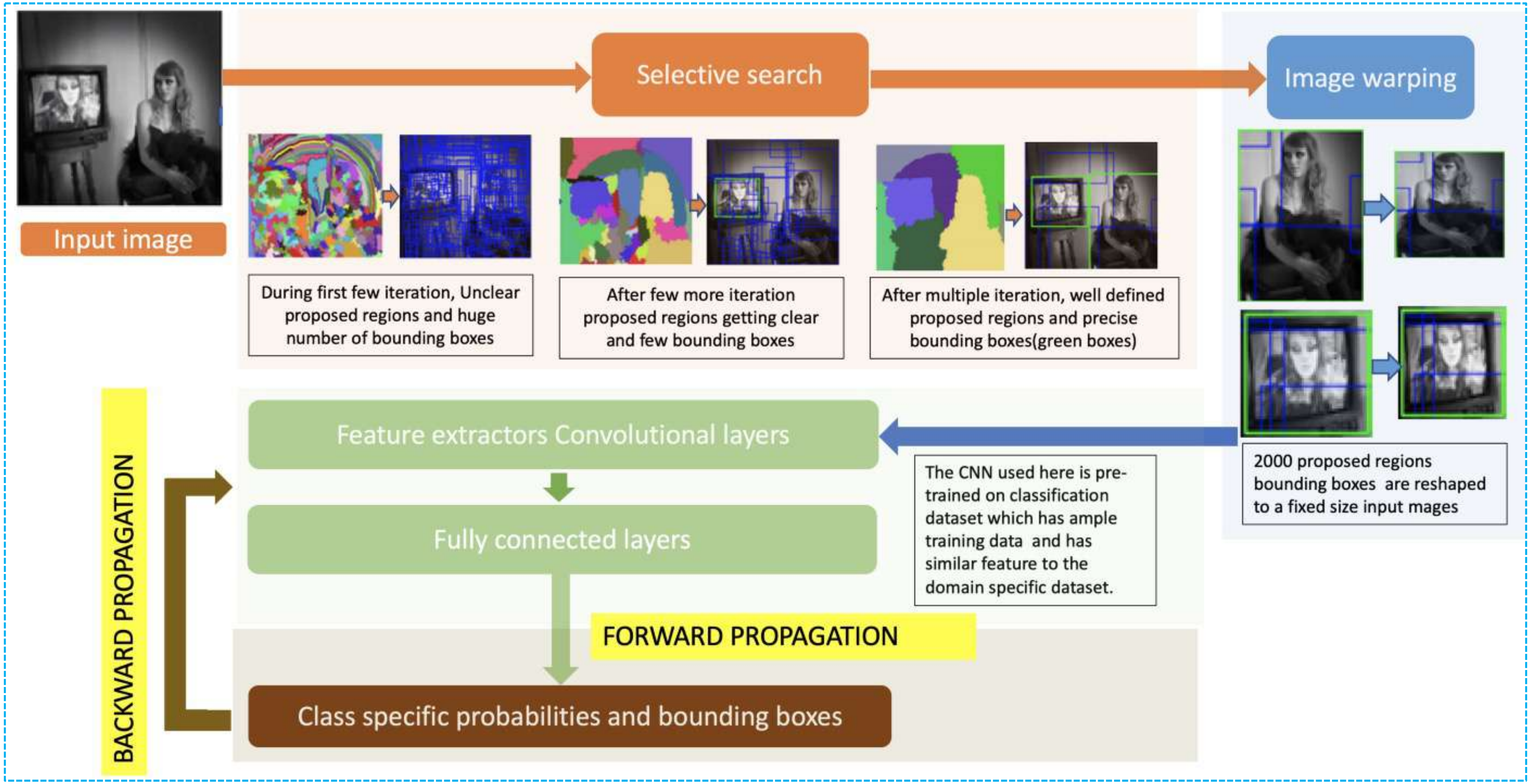
AI VIET NAM
@aivietnam.edu.vn

RCNN - Region proposals with CNNs



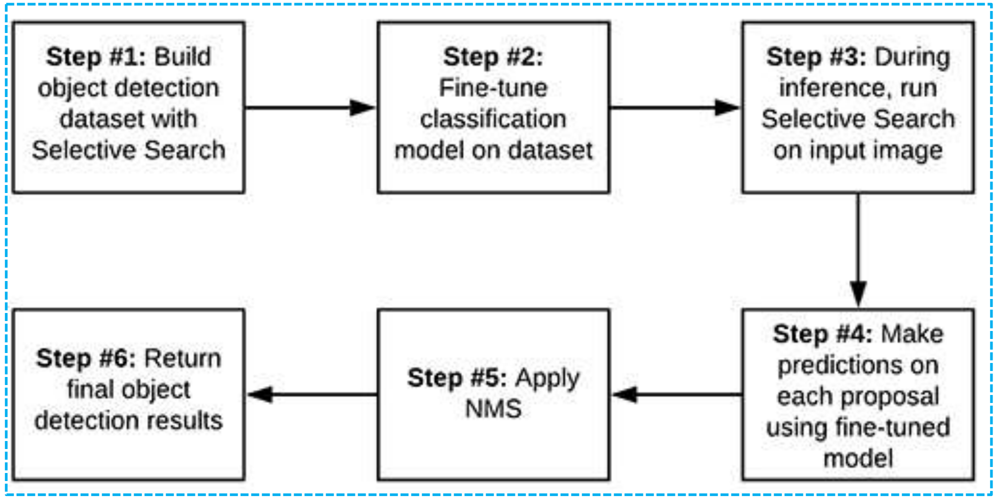
- Why don't we need the sliding window & image pyramid?
- Didn't we end up with too many inputs to the localization network?

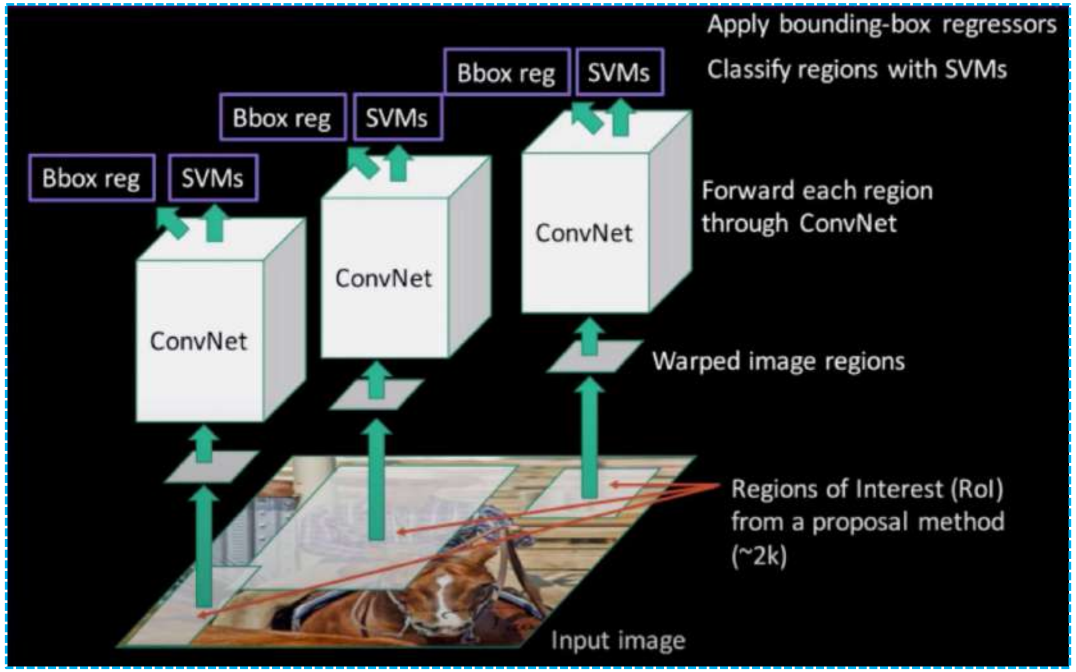
RCNN - Region proposals with CNNs



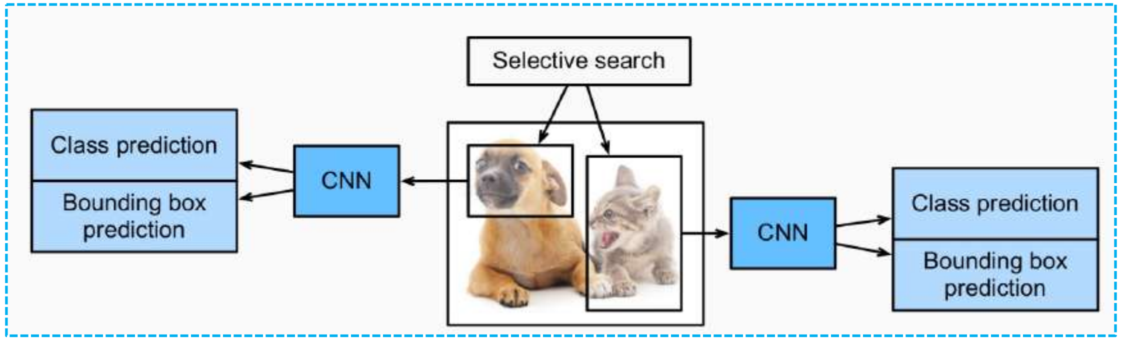
AI VIET NAM
@aivietnam.edu.vn

RCNN - Region proposals with CNNs





1.It consumes a huge amount of time, storage, and computation power.
2.It has a complex multi-stage training pipeline(3 stage — Log loss, SVM, and BBox Regressor's L2 loss).



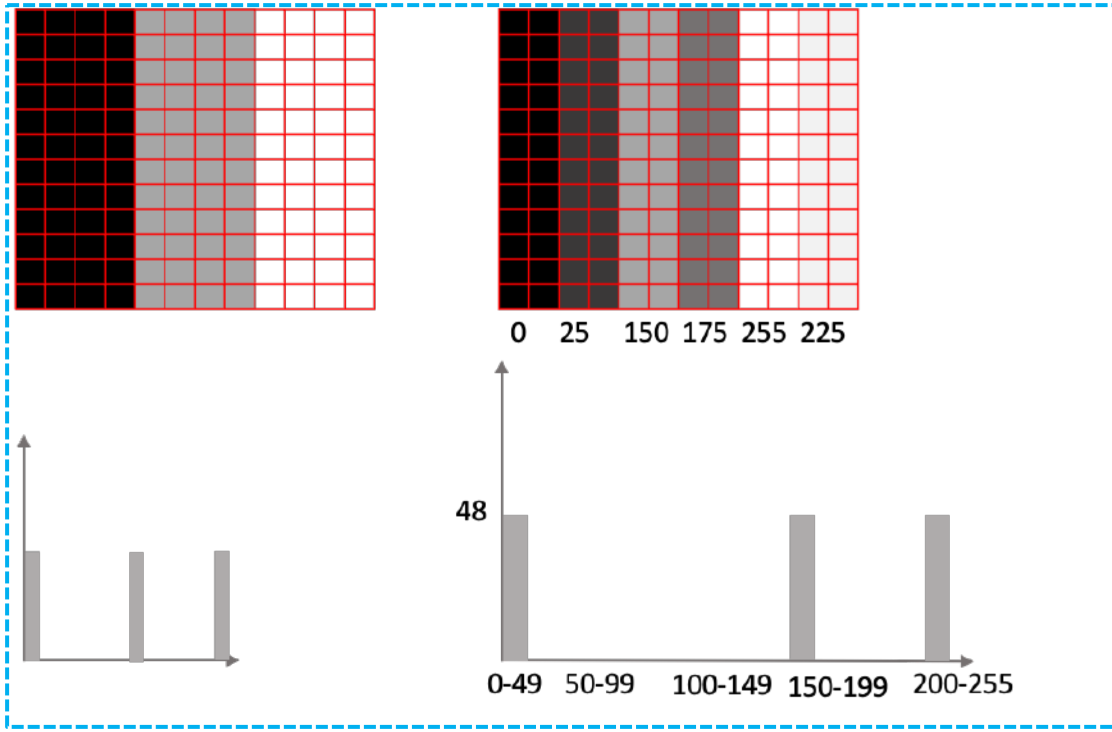
Why not consider running the CNN just once per image and then find a way to share that computation across the ~2000 proposals?

Outline

- CNN Limitations
- Region Based Convolutional Neural Networks
- Spatial Pyramid Pooling
- Fast R-CNN
- Faster RCNN
- YOLOv1-v2

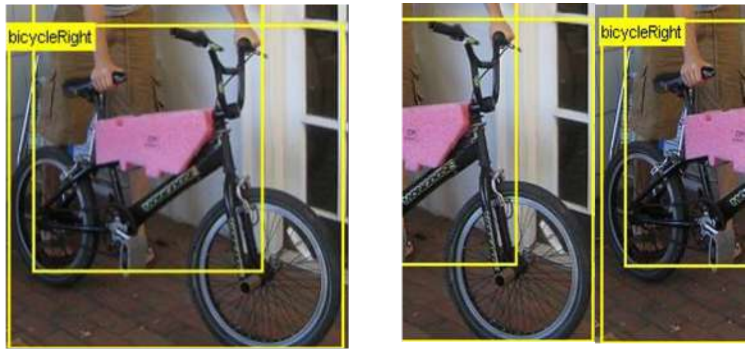
Spatial Pyramid Matching

Histograms of Images - Bins



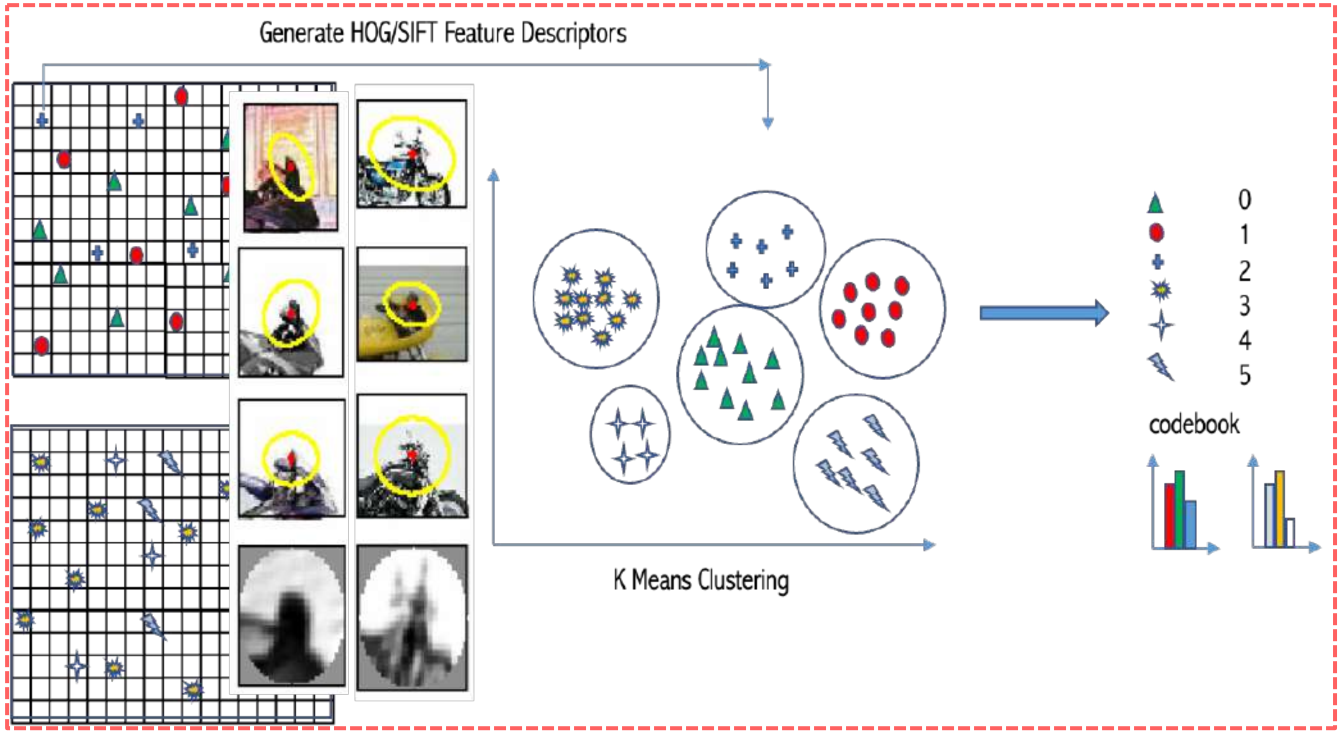
Codebook

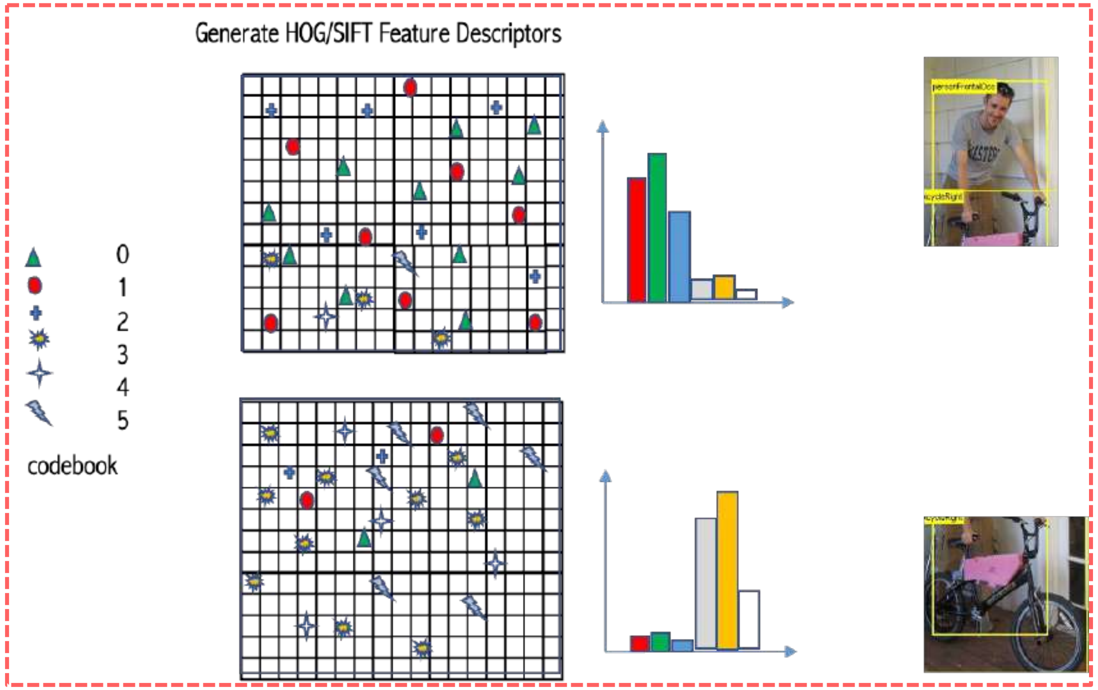
0-49	0	48
50-99	1	0
100-149	2	0
150-199	3	48
200-255	4	48

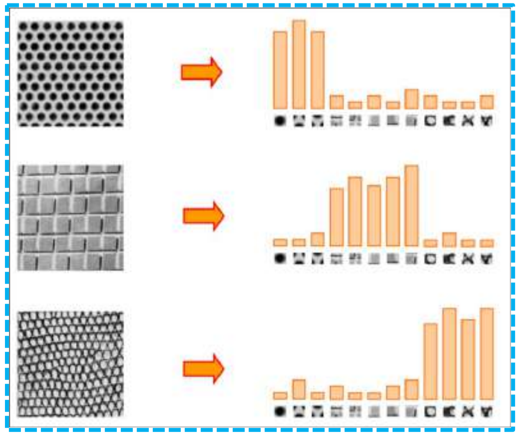


Spatial Pyramid Matching

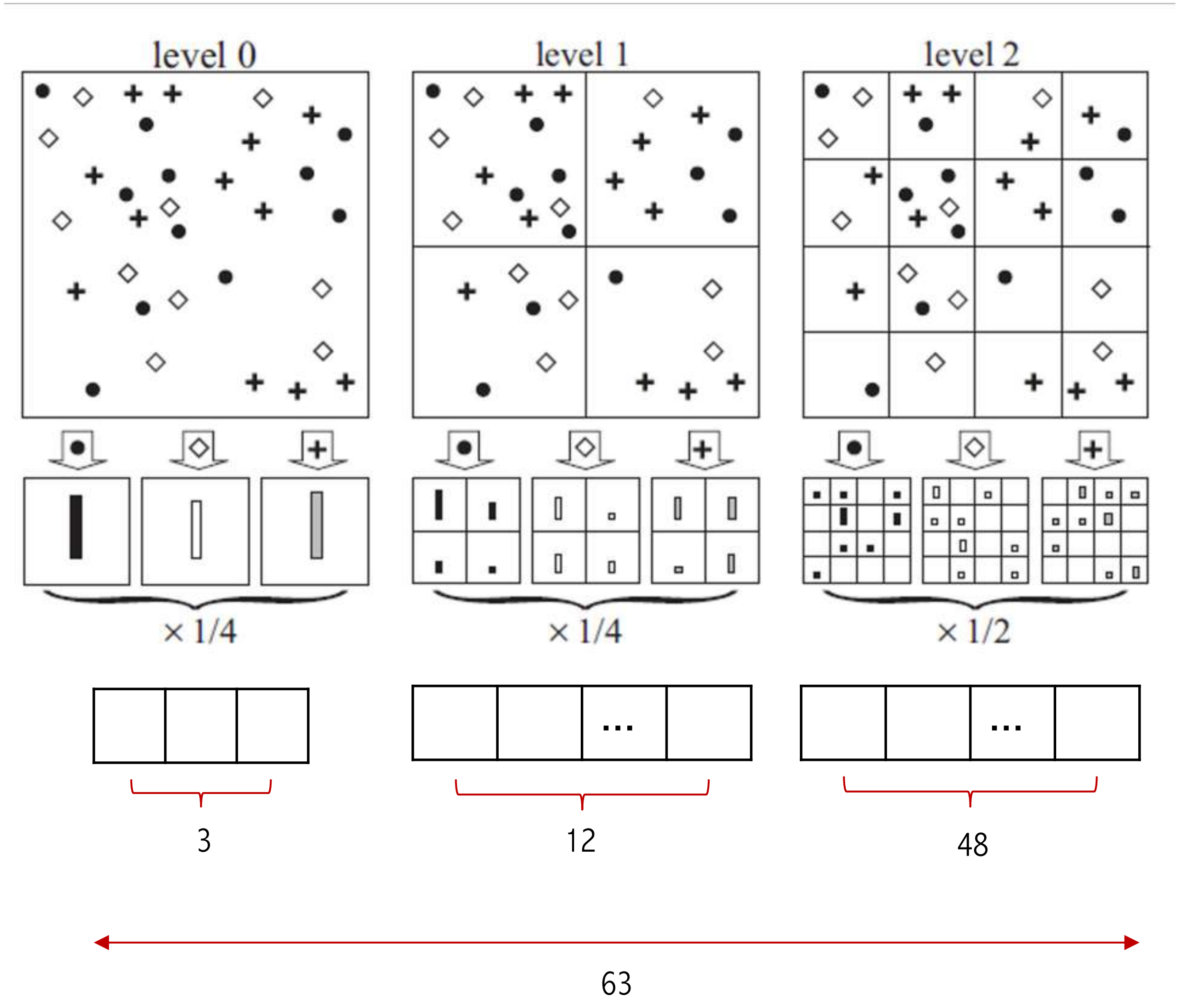
Bag of Visual Words – K Means Clustering



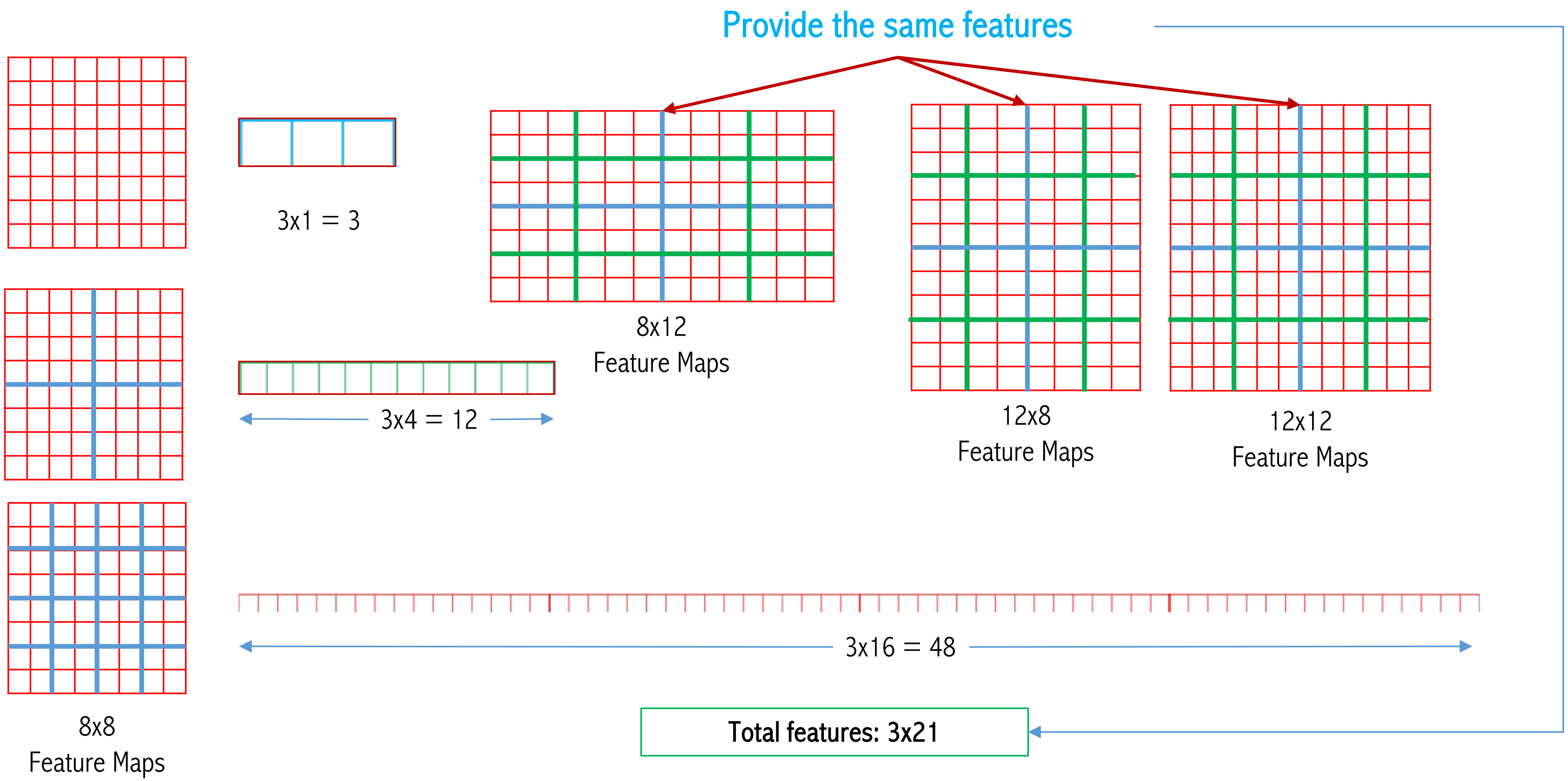




Spatial Pyramid Matching

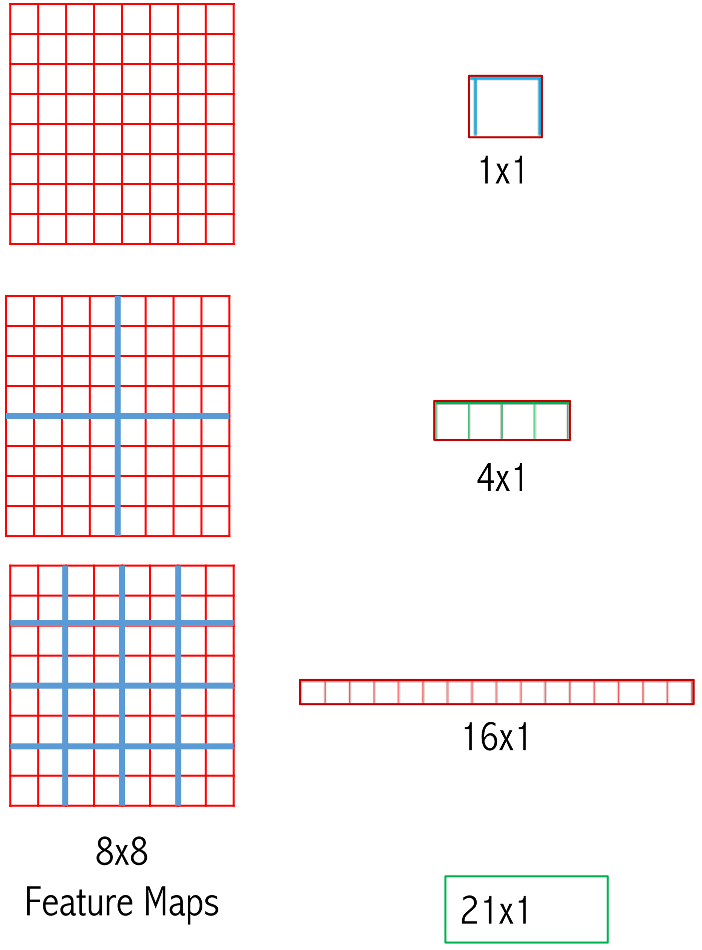


Spatial Pyramid Matching



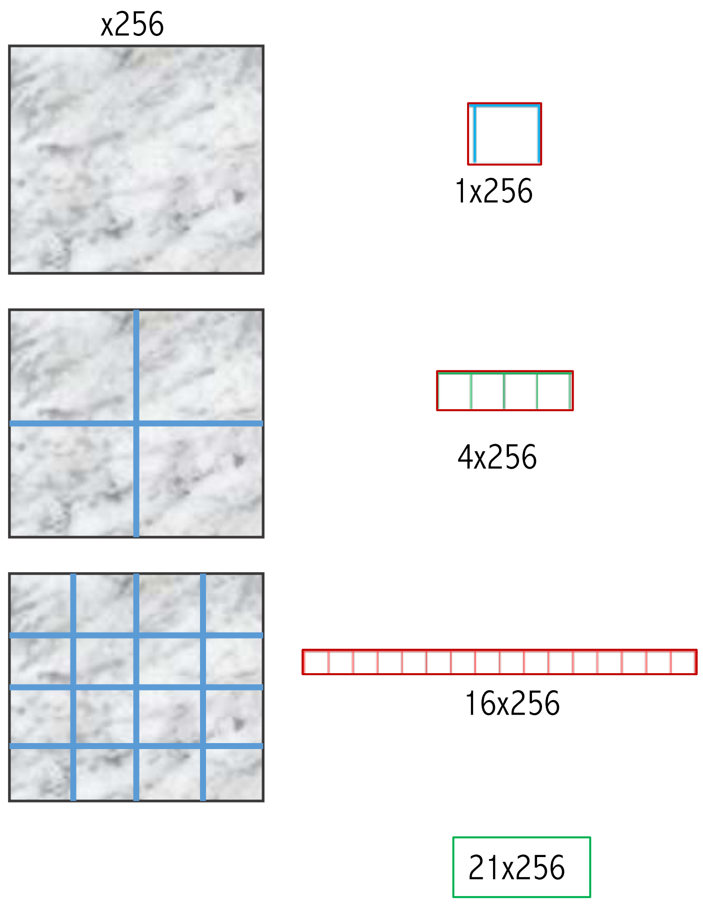
Any Size and Aspect Ratio

Spatial Pyramid Pooling

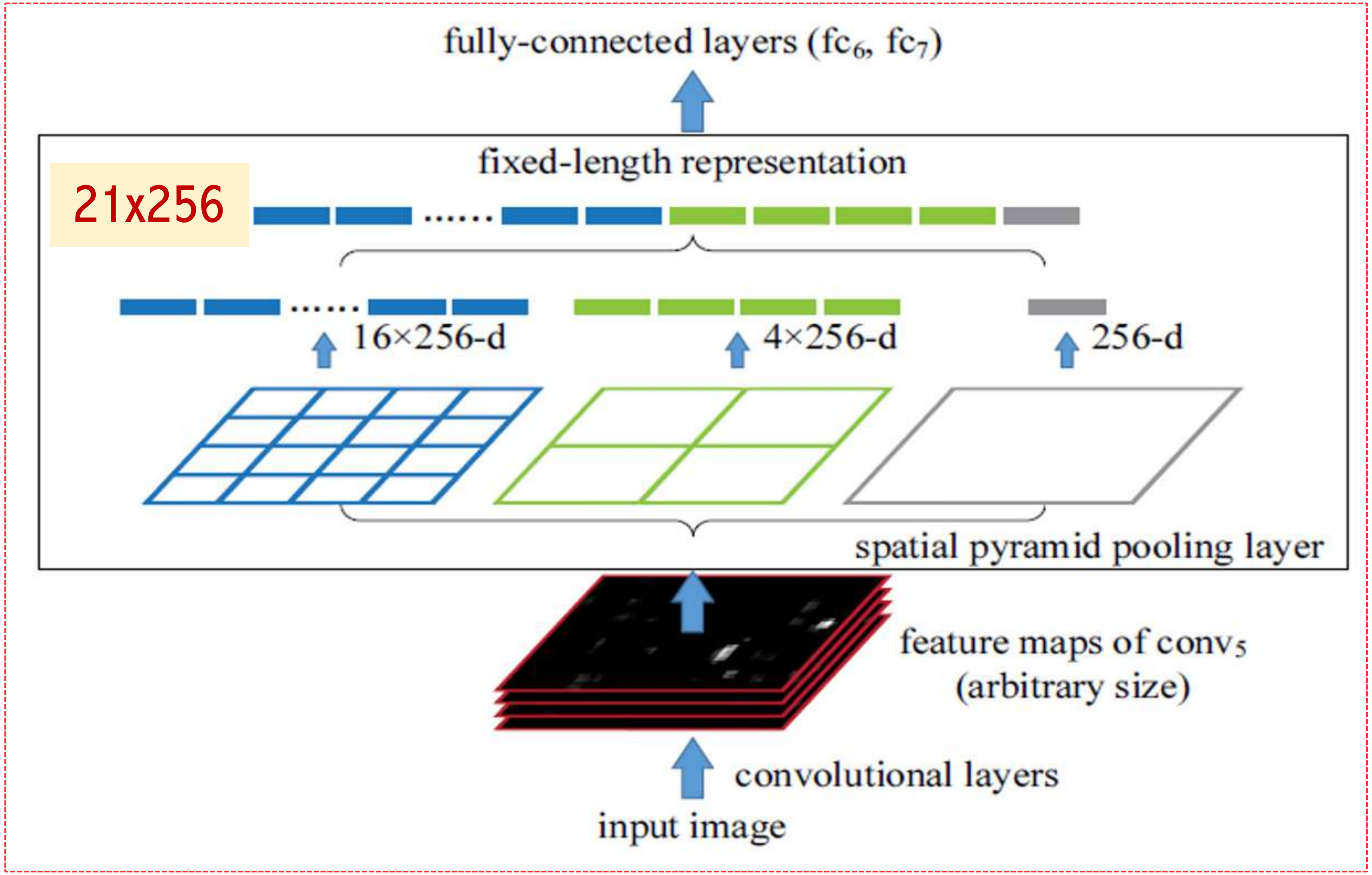


- Identifying features
- K-means clustering
- Codebooks
- Histograms

Just Max-Pool



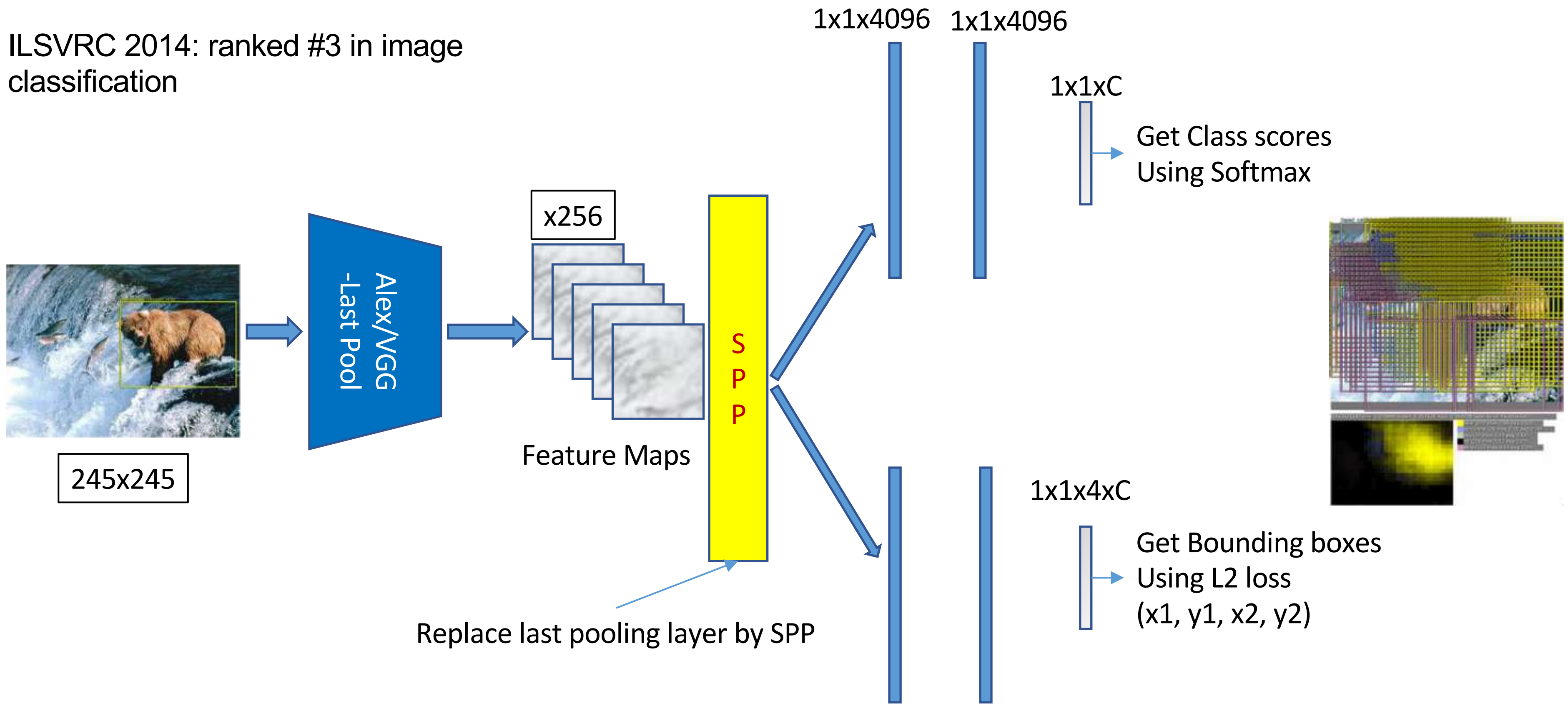
Spatial Pyramid Pooling



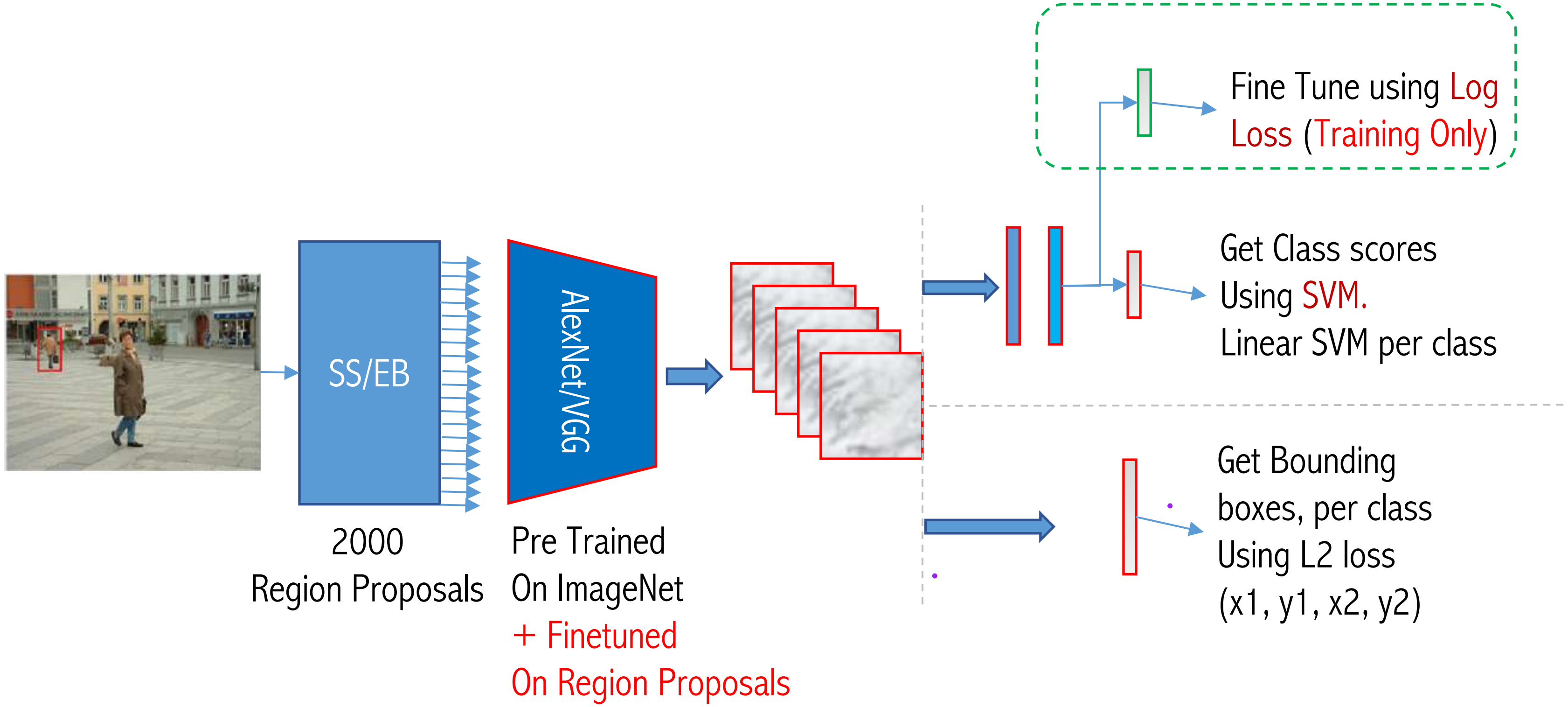
AI VIET NAM
@aivietnam.edu.vn

SPPNet = SPP + Overfeat for Classification

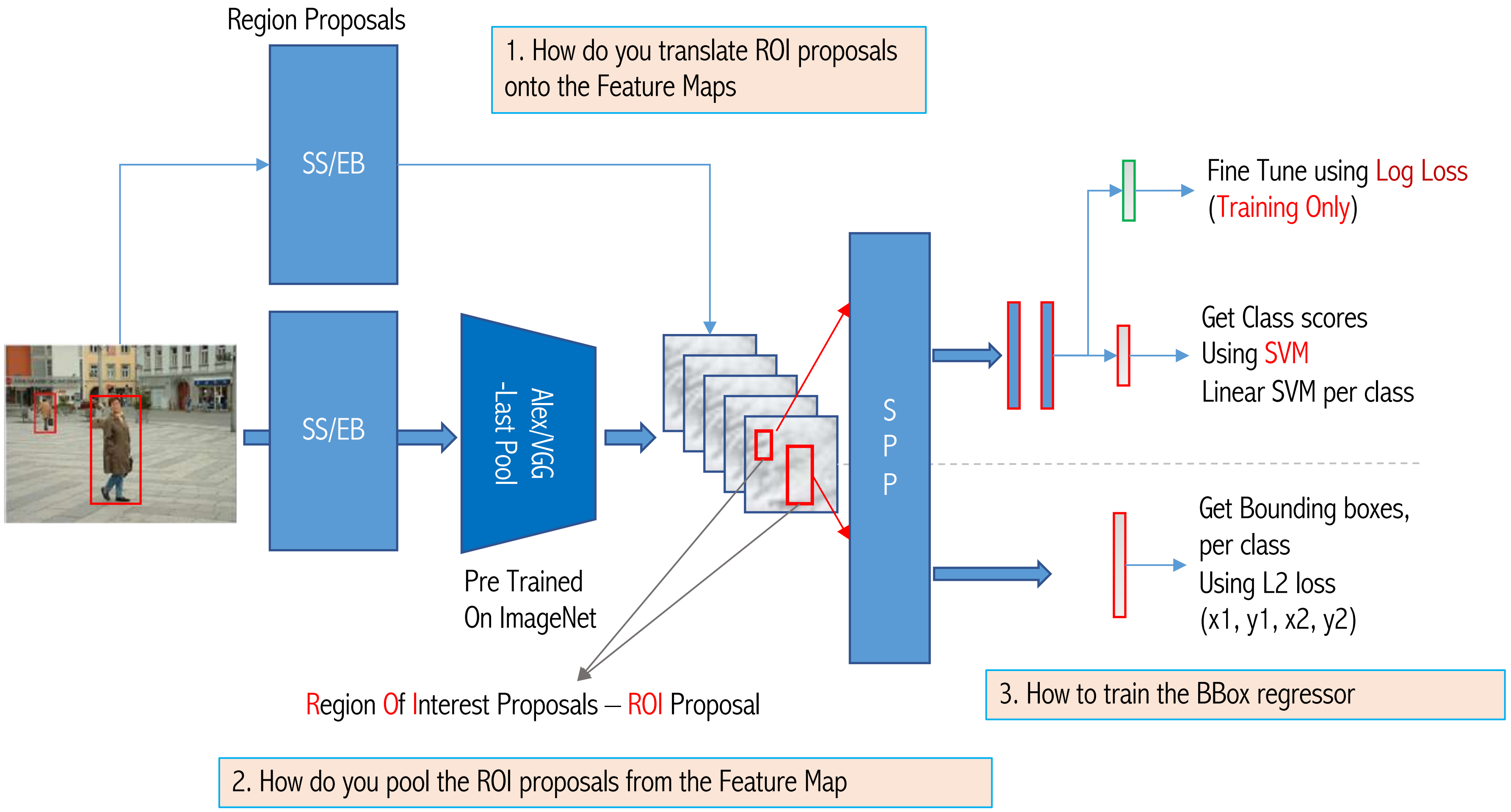
ILSVRC 2014: ranked #3 in image classification



RCNN – Two stage-based methods



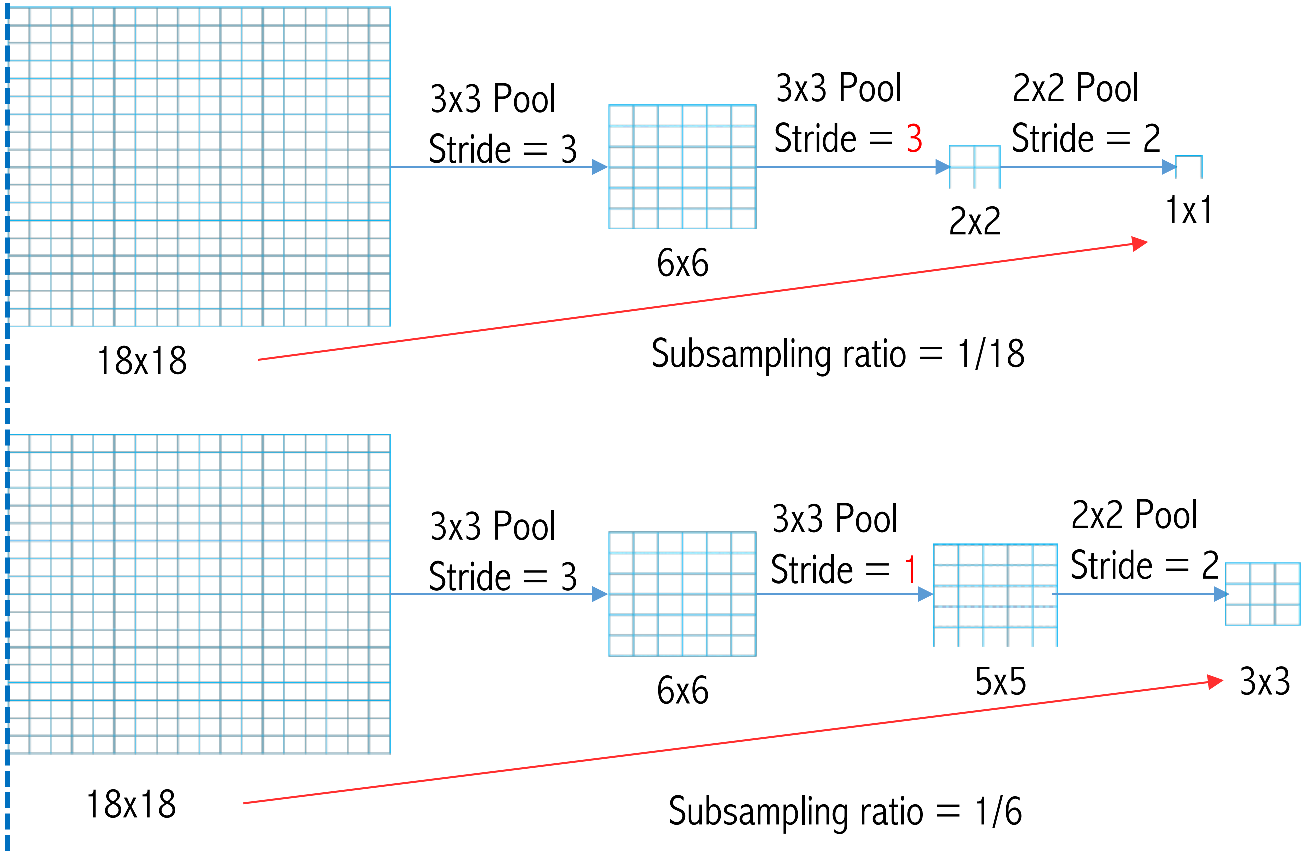
SPP – Two stage-based methods

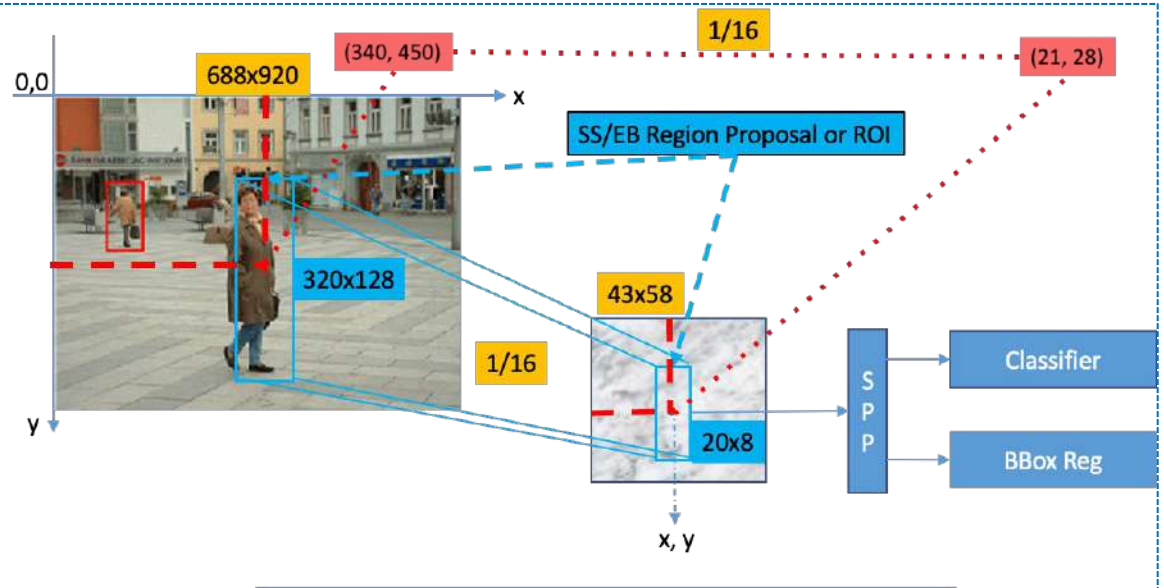


AI VIET NAM
@aivietnam.edu.vn

Subsampling Ratio

1. How do you translate ROI proposals onto the Feature Maps

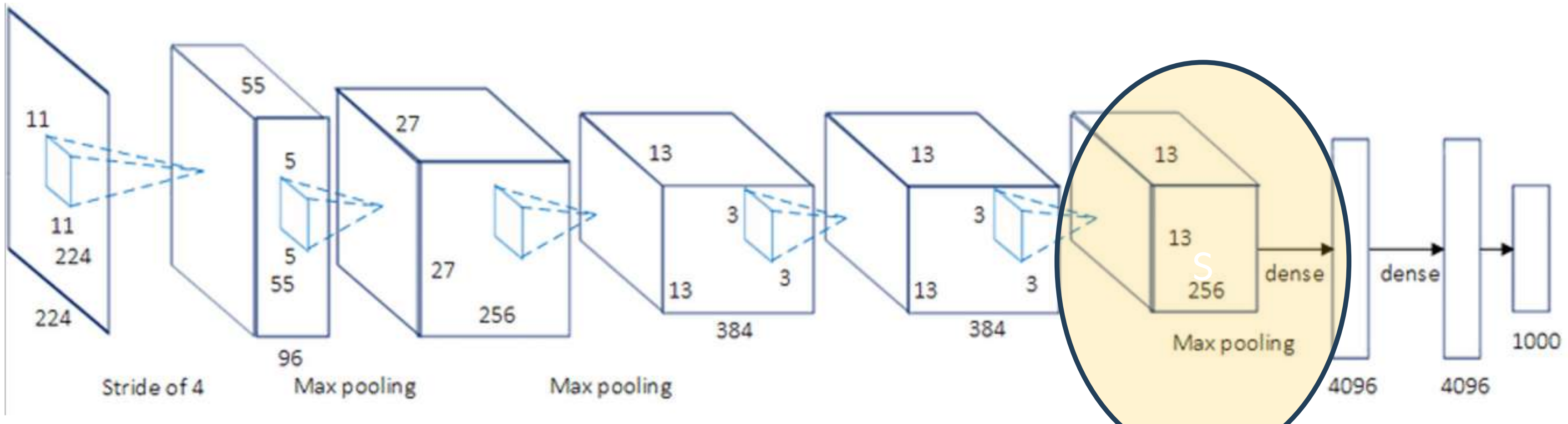




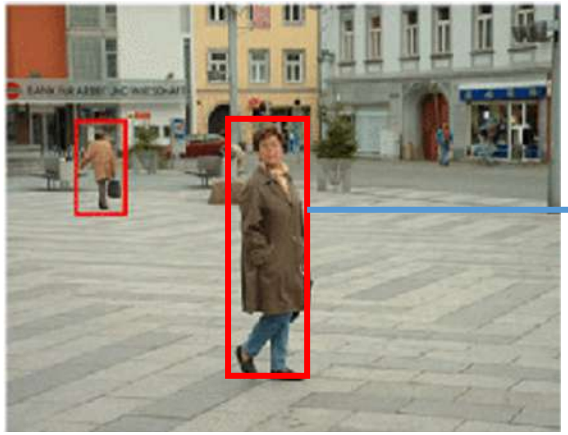
1. How do you translate ROI proposals onto the Feature Maps

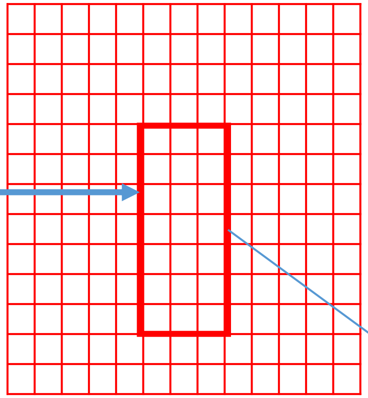
AI VIET NAM
@aivietnam.edu.vn

AlexNet Subsampling Ratio



2. How do you pool the ROI proposals from the Feature Map





13x13
Feature Maps

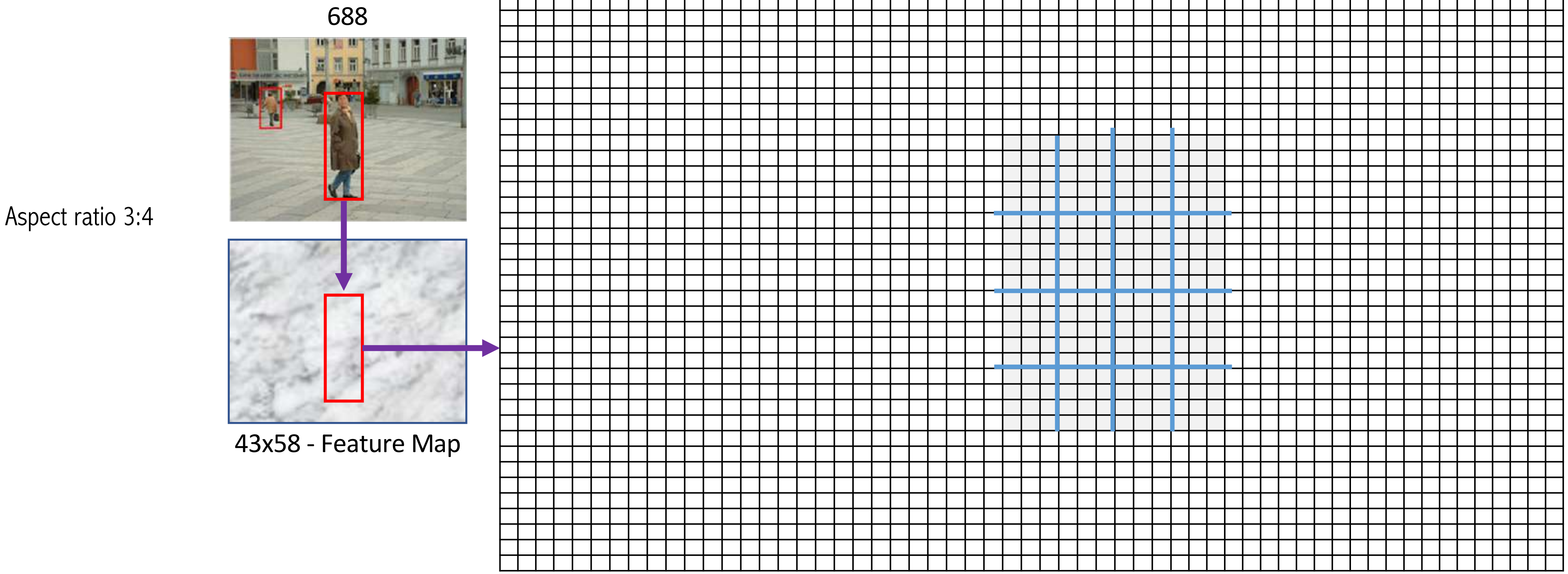
Spatial Pyramid Pooling
Level 1, Level 2, Level 3

6x6 grid. It is impossible

How to solve

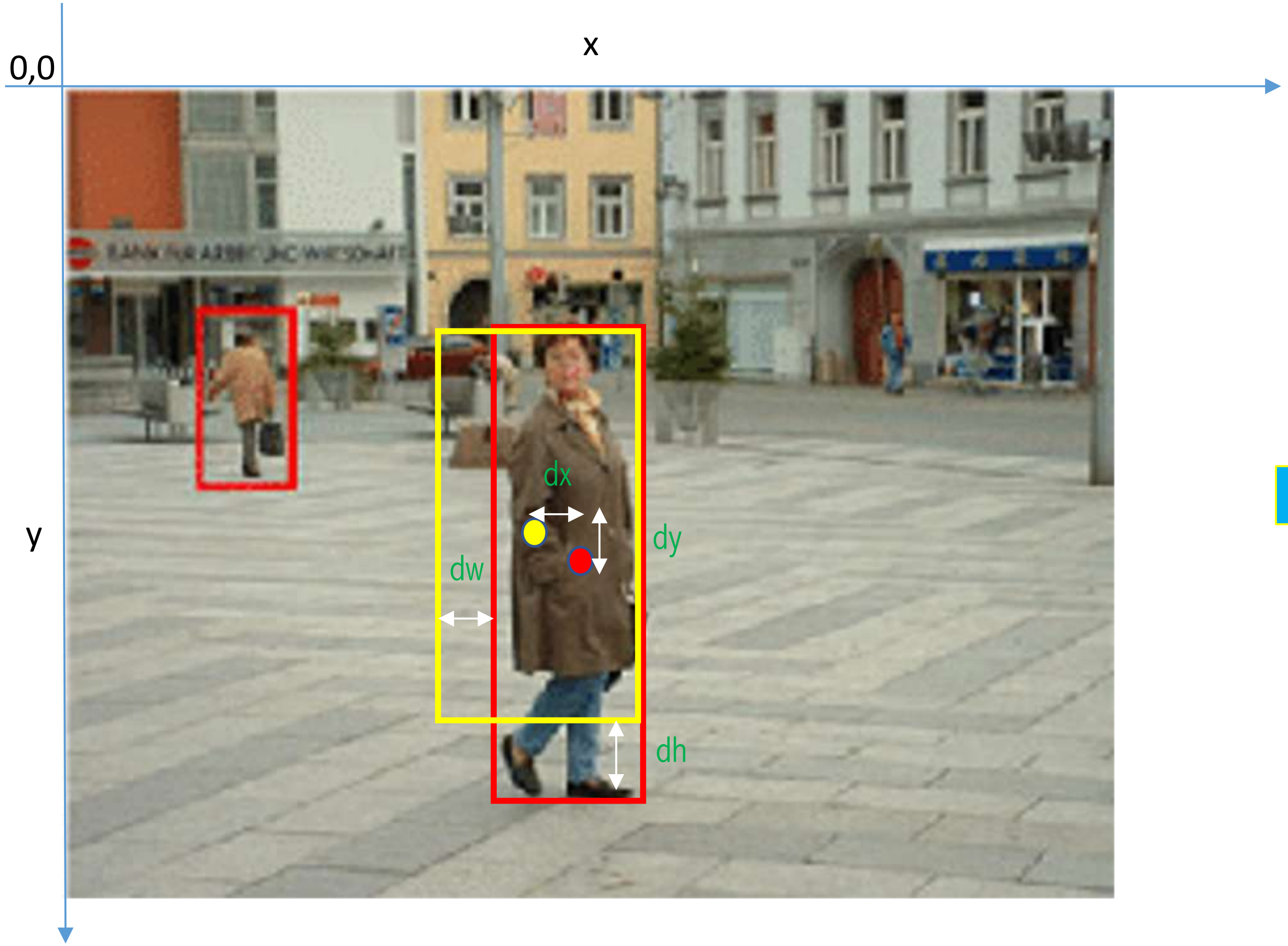
AI VIET NAM
@aivietnam.edu.vn

SPP on Region Proposals

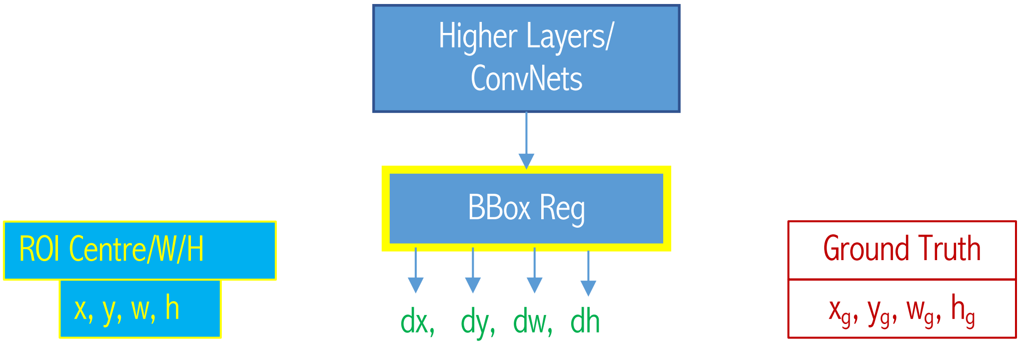


Three level in Practice - {6x6, 3x3, 2x2, 1x1}

BBox Regression Training



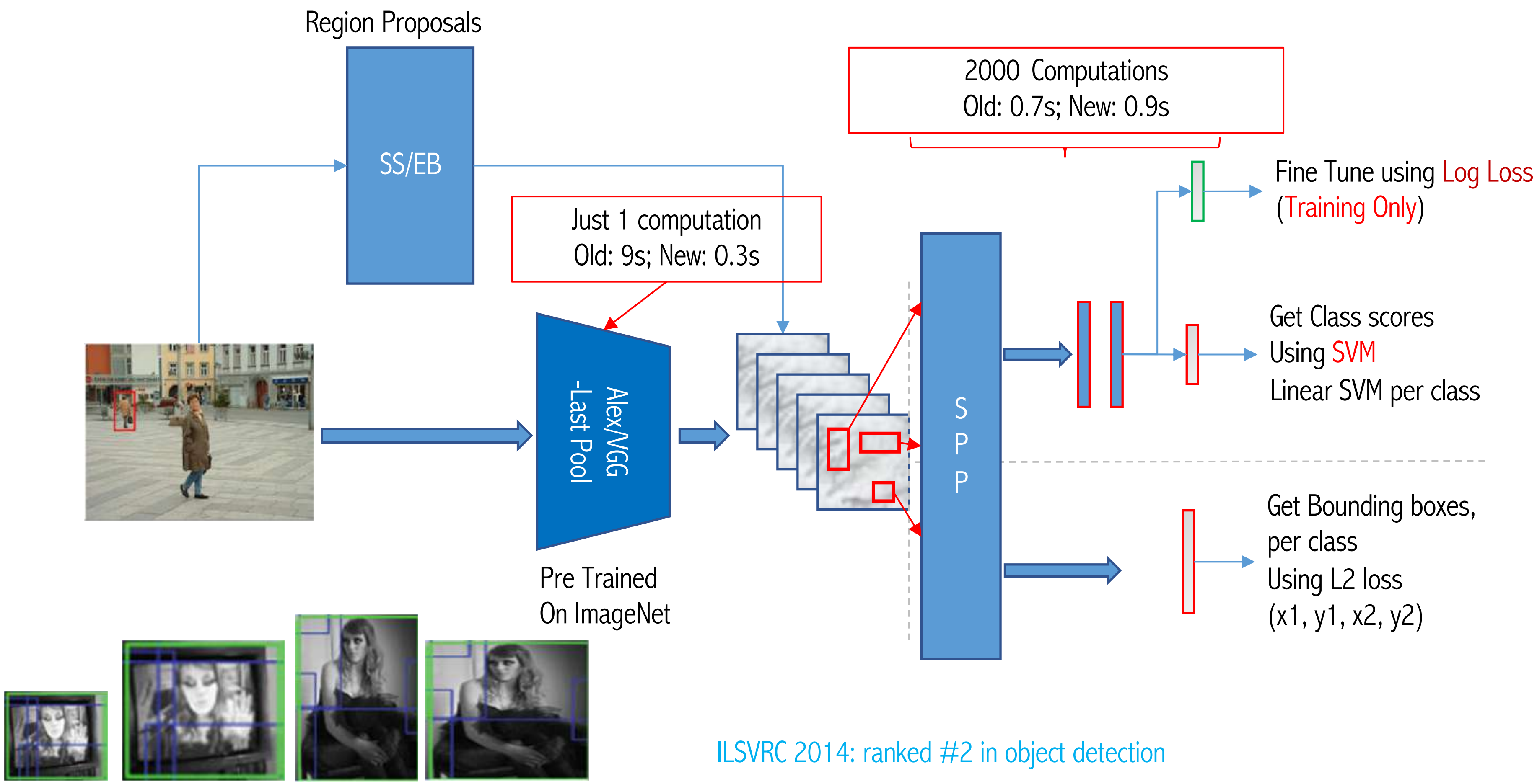
3. How to train the BBox regressor



$$(x + dx - x_g)^2 = 0$$

AI VIET NAM
@aivietnam.edu.vn

SPP – 2 Stage Network - Inference



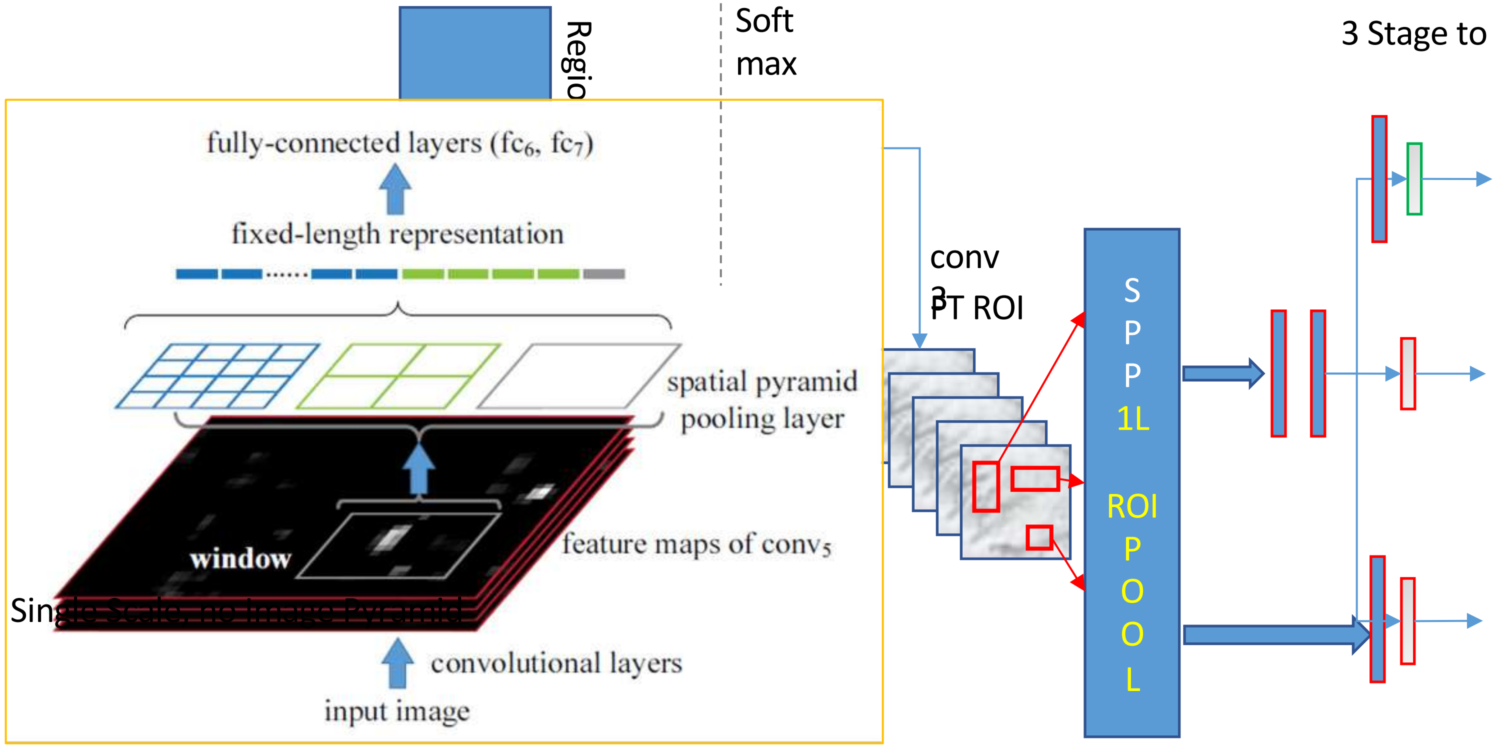
ILSVRC 2014: ranked #2 in object detection

Speed-up: SPP vs RCNN=>Fast RCNN

Basic Shapes → Complex Shapes

FC + Soft max

3 Stage to 1 Stage



Fine Tune using Log Loss (~~Training Only~~) + Softmax for Classification

Get Class scores Using SVM
Linear SVM per class

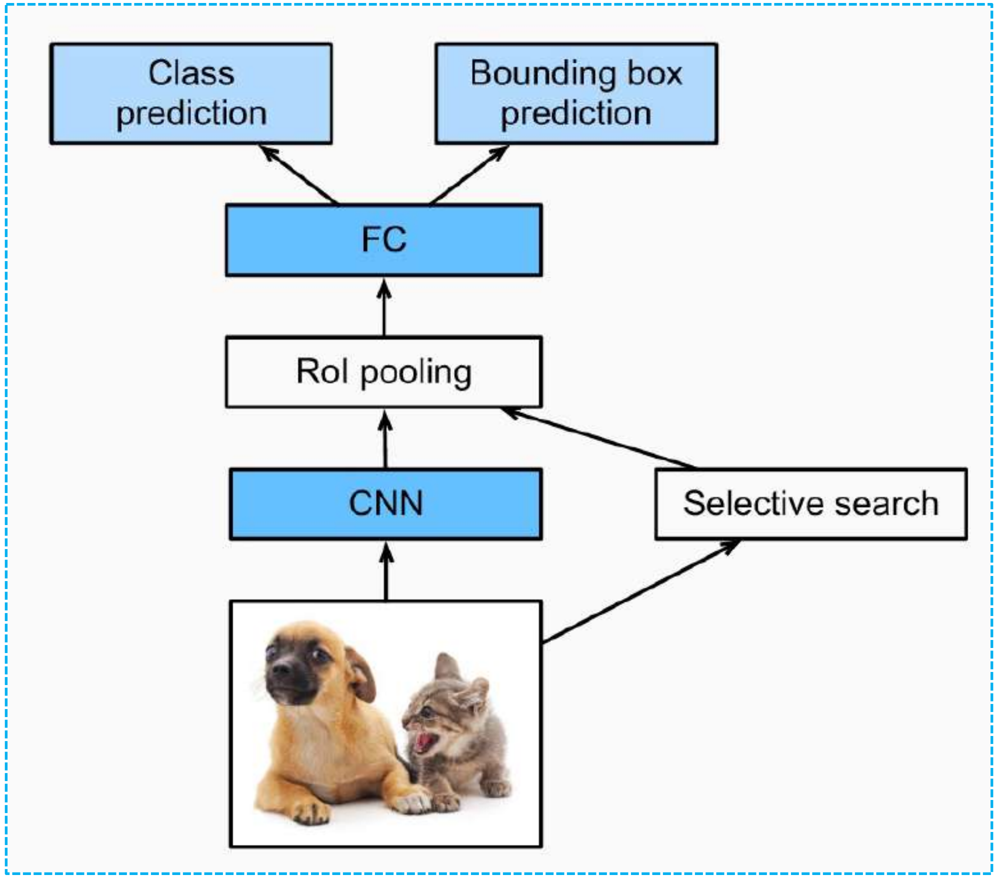
Get Bounding boxes, per class
Using ~~L2 Loss~~
Smooth L1 loss
(x, y, h, w)

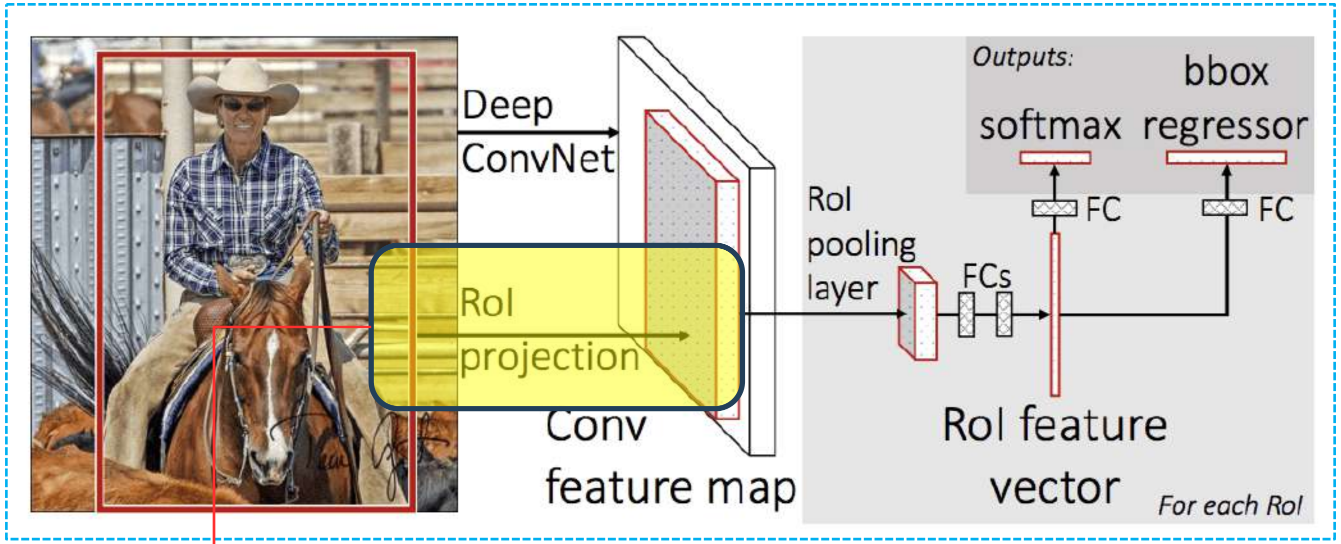
if |a-b|<1, $(a-b)^2/2$
else, |a-b| - 1/2

	SPP (1-sc) (ZF-5)	SPP (5-sc) (ZF-5)	R-CNN (Alex-5)
$pool_5$	43.0	44.9	44.2
fc_6	42.5	44.8	46.2
$ftfc_6$	52.3	53.7	53.1
$ftfc_7$	54.5	55.2	54.2
$ftfc_7$ bb	58.0	**59.2**	58.5
conv time (GPU)	0.053s	0.293s	8.96s
fc time (GPU)	0.089s	0.089s	0.07s
total time (GPU)	0.142s	0.382s	9.03s
speedup (*vs.* RCNN)	**64×**	**24×**	-

Table 9: Detection results (mAP) on Pascal VOC 2007. "ft" and "bb" denote fine-tuning and bounding box regression.

Fast R-CNN



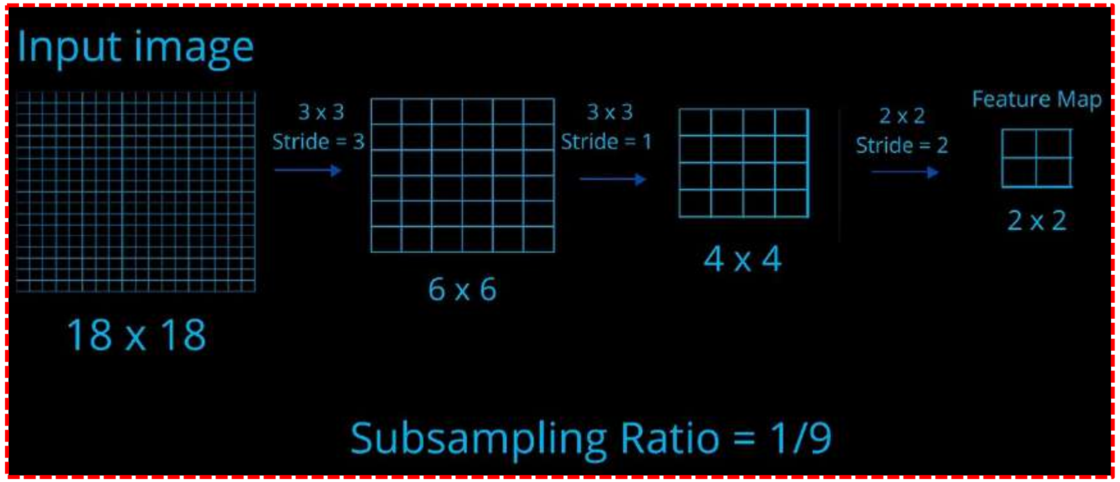


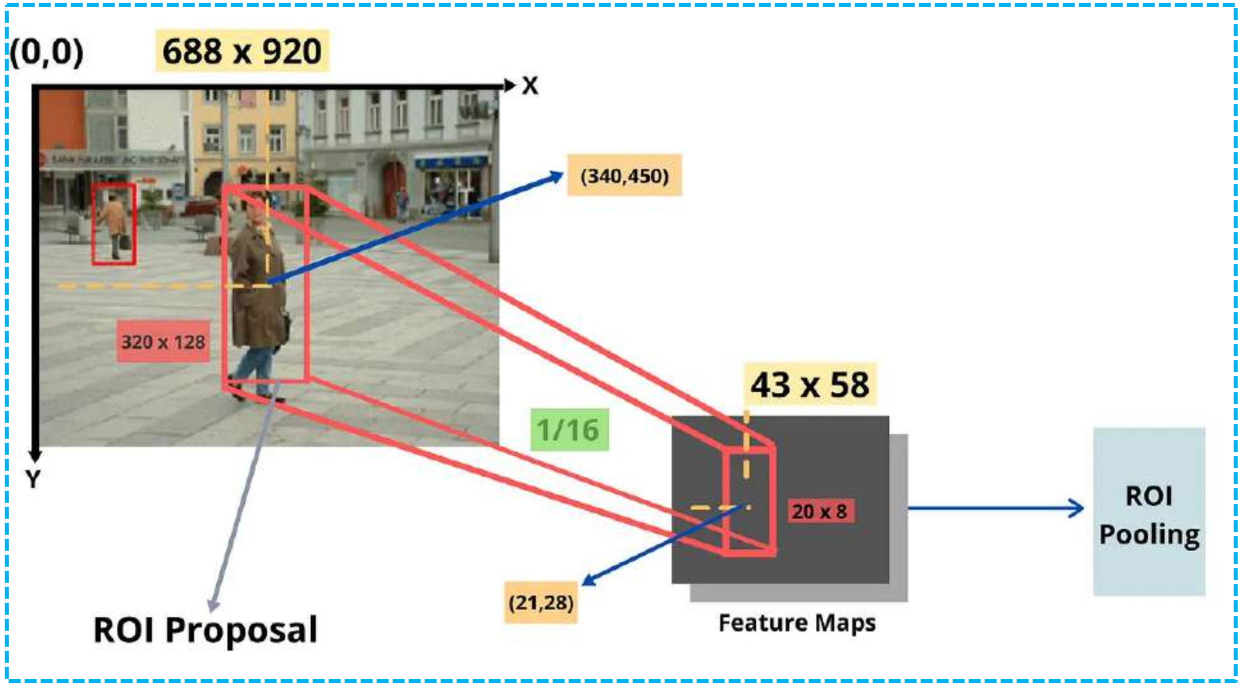
New terminology

Fast R-CNN, which was developed a year later after R-CNN, solves these issues very efficiently and is about 146 times faster than the R-CNN during the test time.

AI VIET NAM
@aivietnam.edu.vn

Sub-Sampling Ratio & Roi Projection



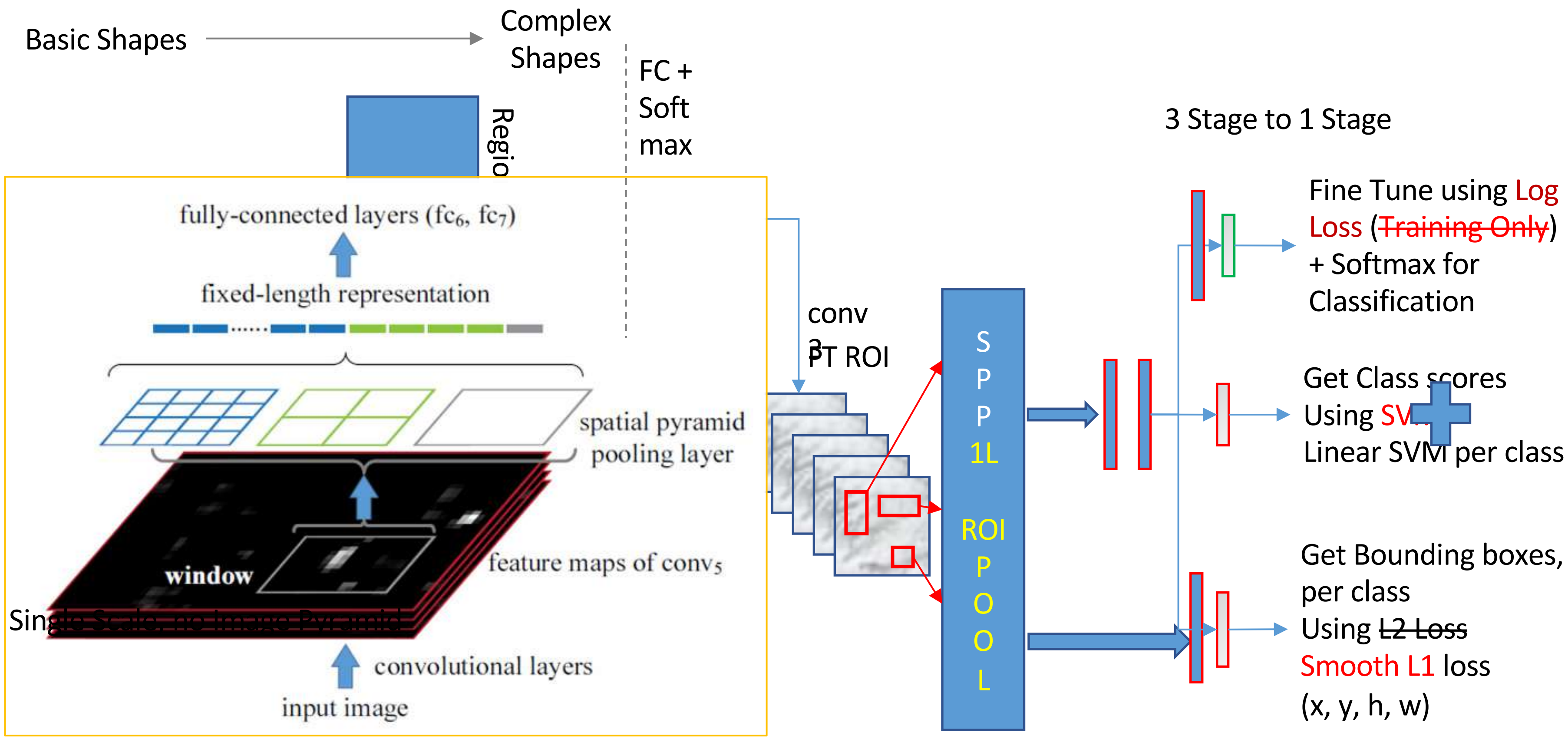


The idea of ROI projection is that we get the coordinates of the bounding box from the ROI proposal and we need to project them onto the feature maps by projecting the ROI proposal with respect to the subsampling ratio.

Outline

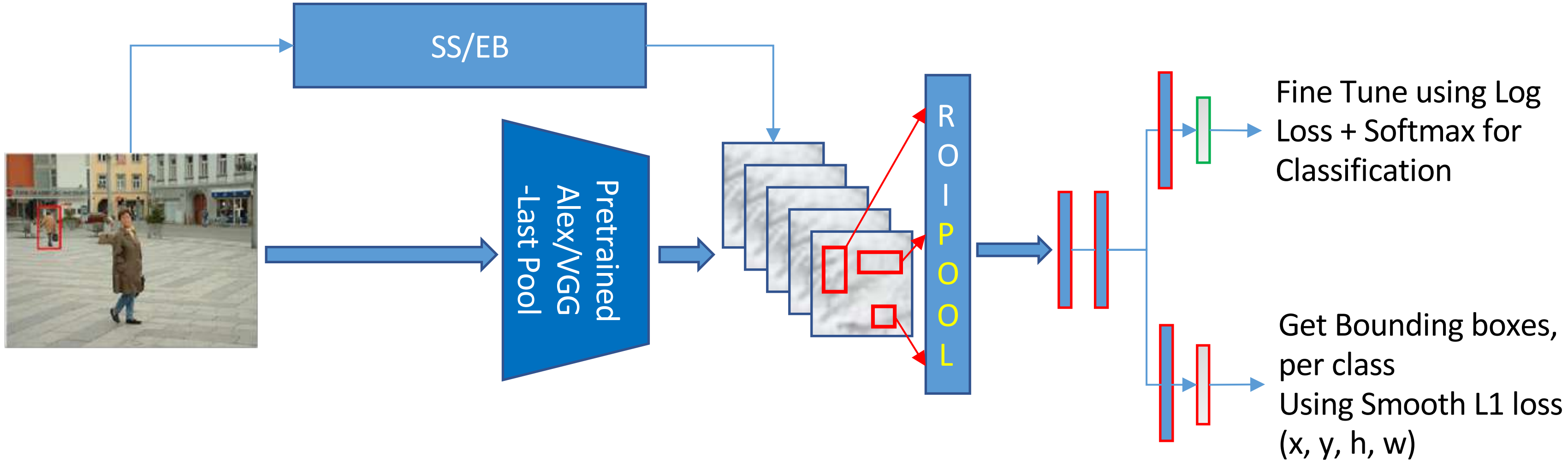
- CNN Limitations
- Region Based Convolutional Neural Networks
- Spatial Pyramid Pooling
- Fast R-CNN
- Faster RCNN
- YOLOv1-v2

RCNN -> SPPNet -> Fast RCNN



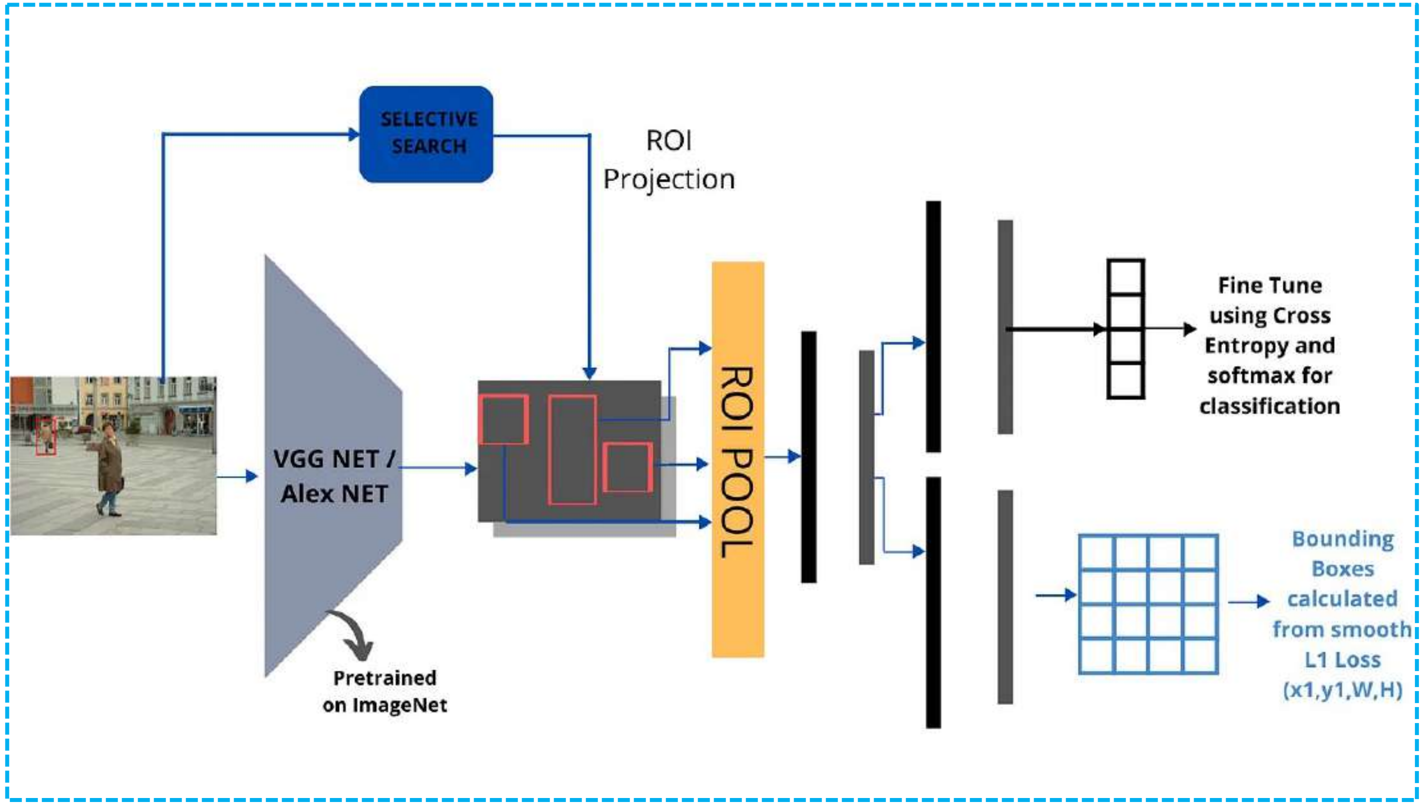
if |a-b|<1, $(a-b)^2/2$
else, |a-b| - 1/2

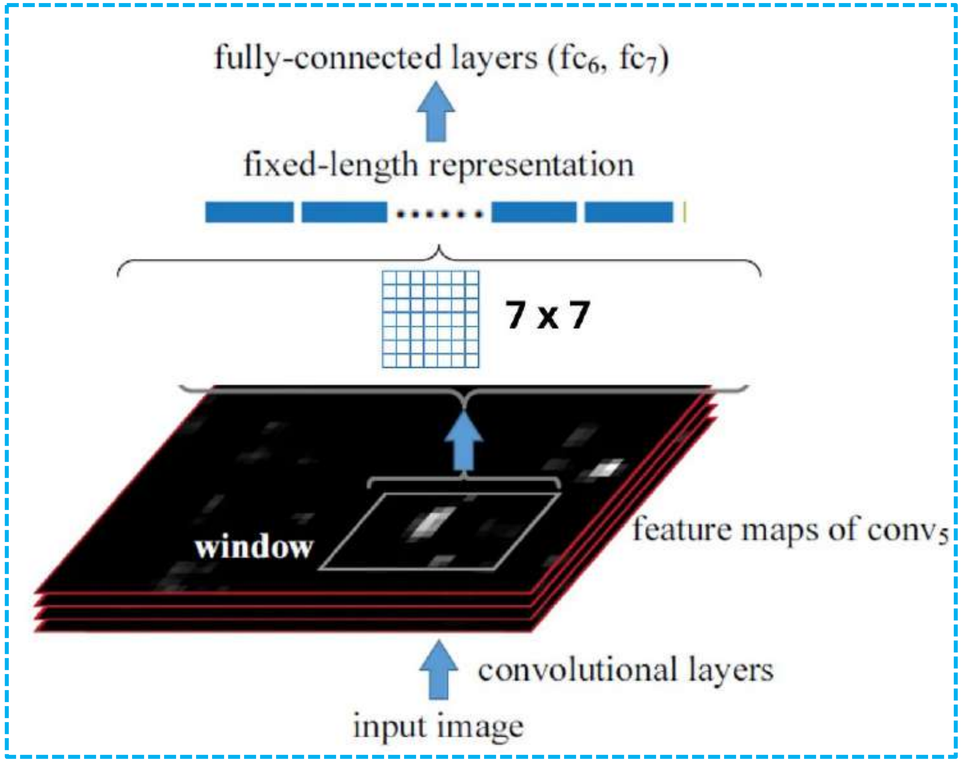
RCNN -> SPPNet -> Fast RCNN



AI VIET NAM
@aivietnam.edu.vn

Fast R-CNN





	RCNN	Fast RCNN
Training Time	84 hours	9.5 hours
(Speedup)	1x	8.8x
Test time per image	47 seconds	0.32 seconds
(Speedup)	1x	146x
Test time per image with Selective Search	50 seconds	2 seconds
(Speedup)	1x	25x

So why don't we reuse those same CNN results for region proposals instead of running a separate selective search algorithm?

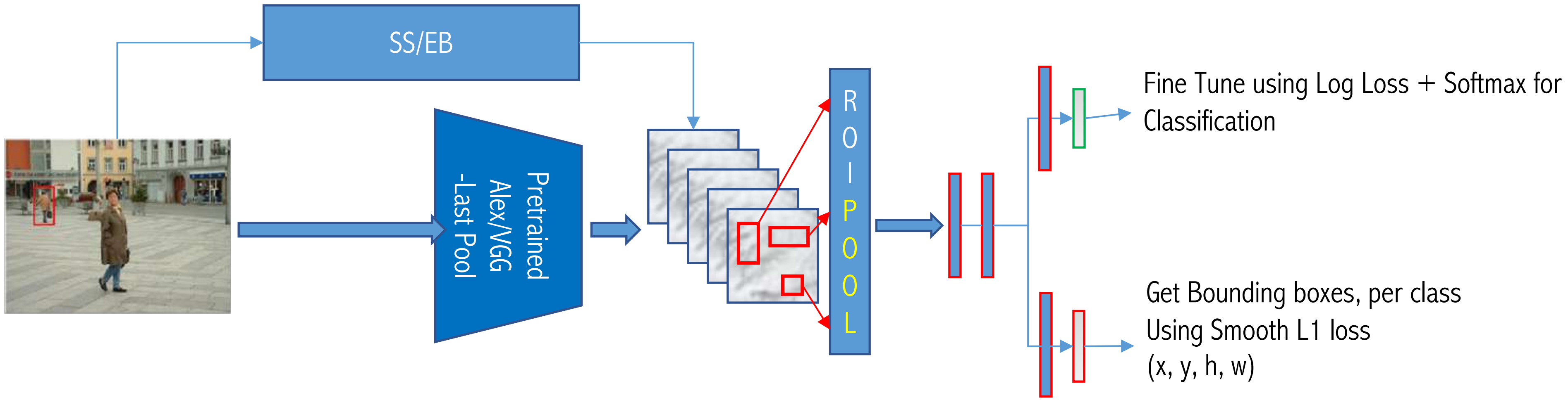
Outline

- CNN Limitations
- Region Based Convolutional Neural Networks
- Spatial Pyramid Pooling
- Fast R-CNN

- Faster RCNN
- YOLOv1-v2

AI VIET NAM
@aivietnam.edu.vn

Criteria for replacing SS



< 2000 Region Proposals

As fast as SS or better

As Accurate as SS or better

Should be able to propose Overlapping ROIs with different Aspect Ratios and Scale

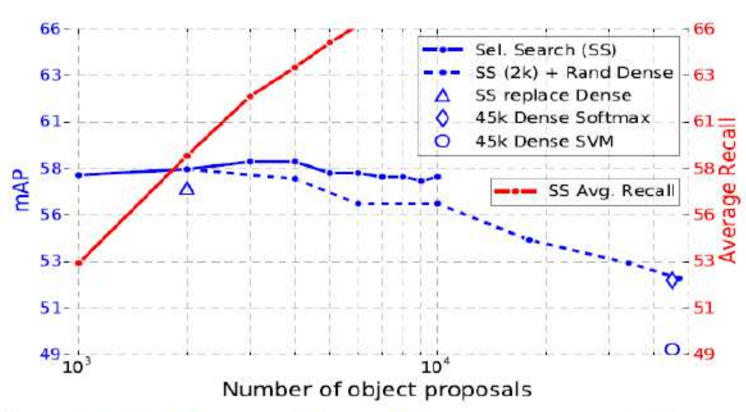
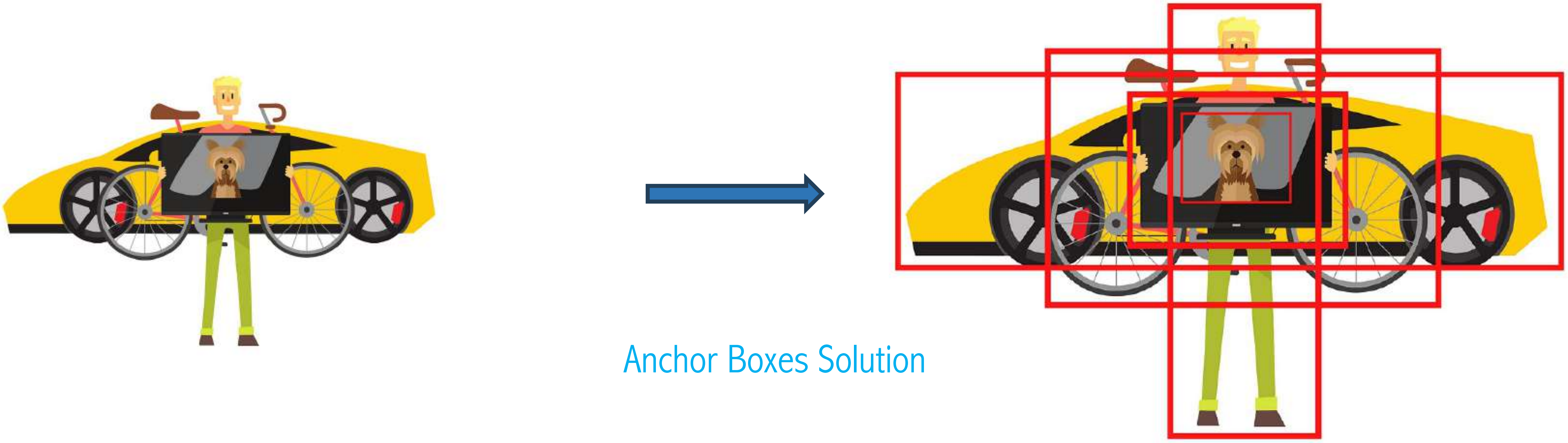


Figure 3. VOC07 test mAP and AR for various proposal schemes.

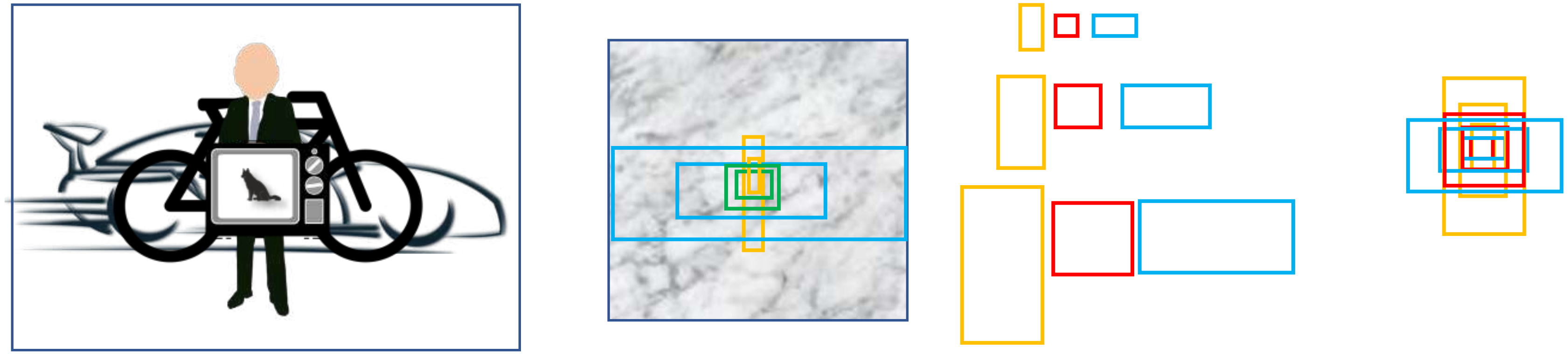
Role of Region Proposals

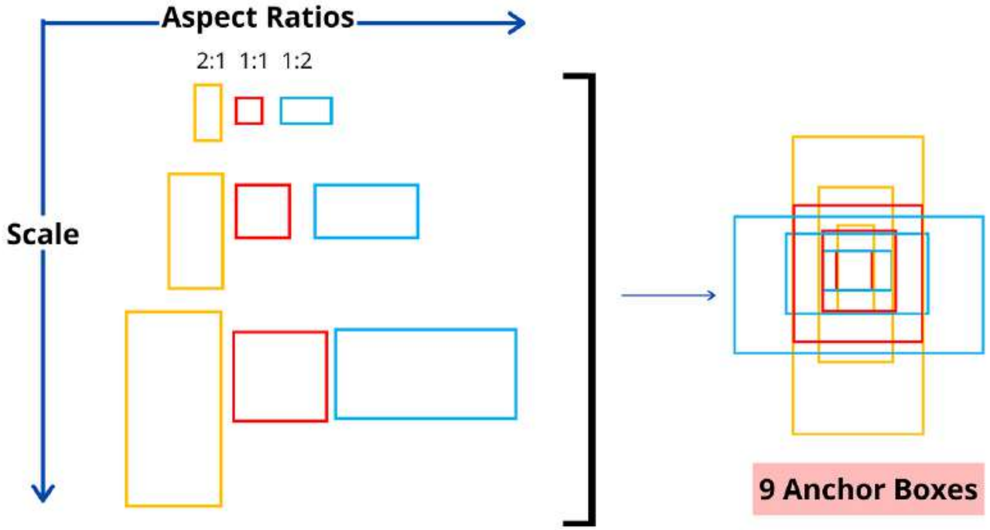
Overlapping ROIs



From the image, we see a lot of objects overlapping each other. We see a car, a bicycle, a person holding a television, and a dog inside this television. The selective search could solve this problem but we end up with a huge number of ROIs. We need to think of an idea that efficiently solves this.

Anchor Boxes



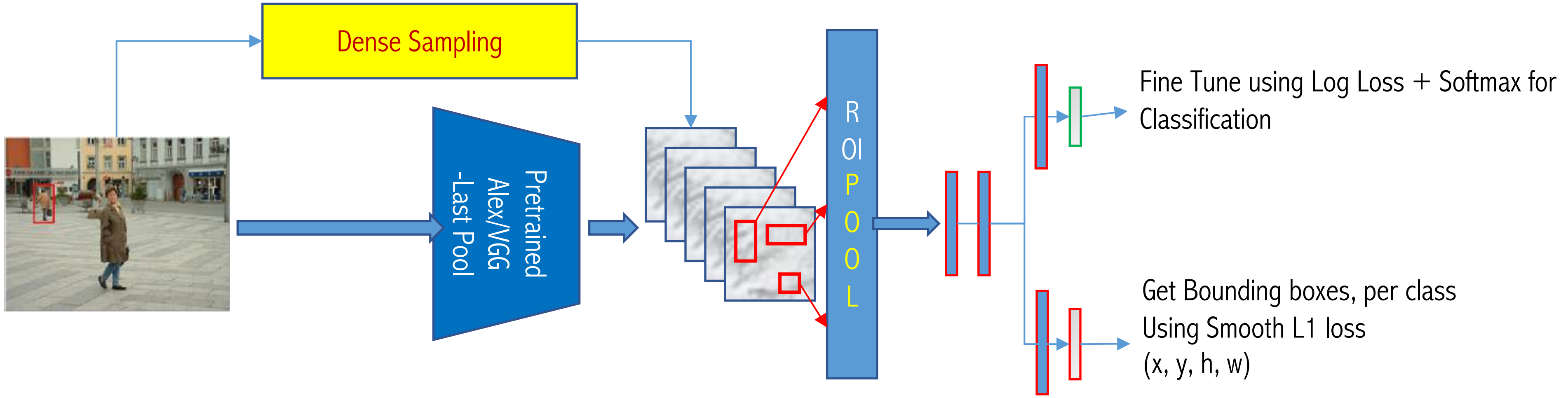


Any object in the image can be detected using boxes of 3 different scales and 3 different aspect ratios.

This could be a technique that can be used to solve our purpose of replacing the region proposal.

Solutions for Replace Region Proposals

Removing Selective Search and applying a sliding window on top of the Feature Maps. But with this, we end detecting mostly objects of a single scale.

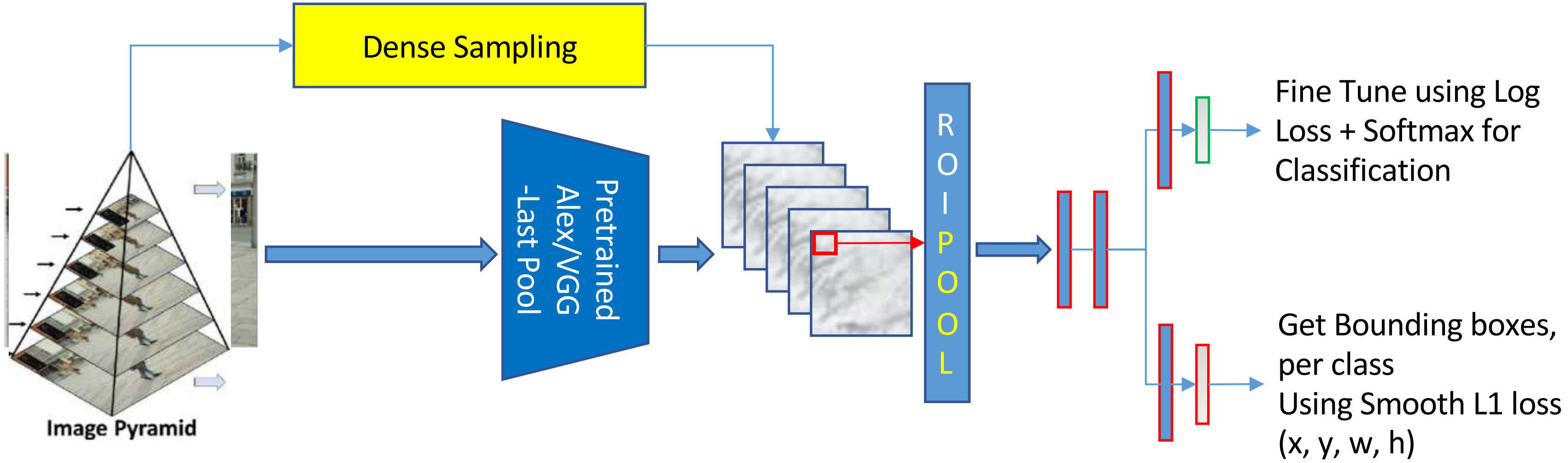


Fast RCNN + Sliding Window

AI VIET NAM
@aivietnam.edu.vn

Solutions for Replace Region Proposals

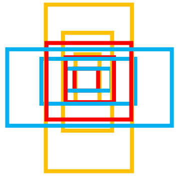
To take care of multiple scales, we have to use Image Pyramids at the input. But using images of 5 different scales (by which almost every object can be detected) makes the network 4 times slower.



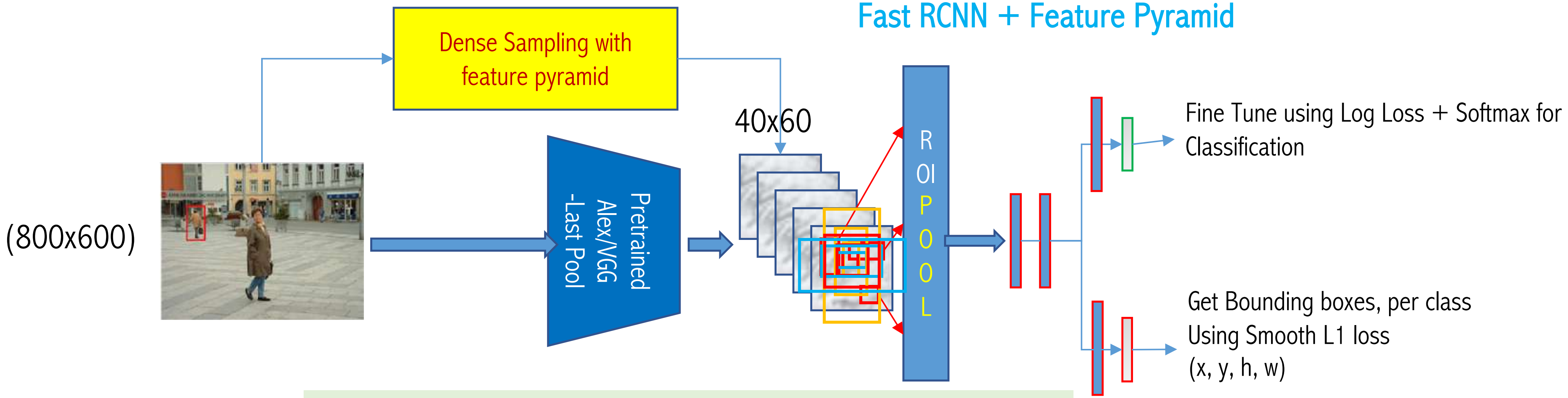
Fast RCNN + Sliding Window + Image Pyramid

Solutions for Replace Region Proposals

Another option is to use sliding windows of different sizes (9, as shown above) on the Feature Map. This concept is called the Feature Pyramid. This involves the use of sliding windows of 9 different sizes on top of the feature maps.



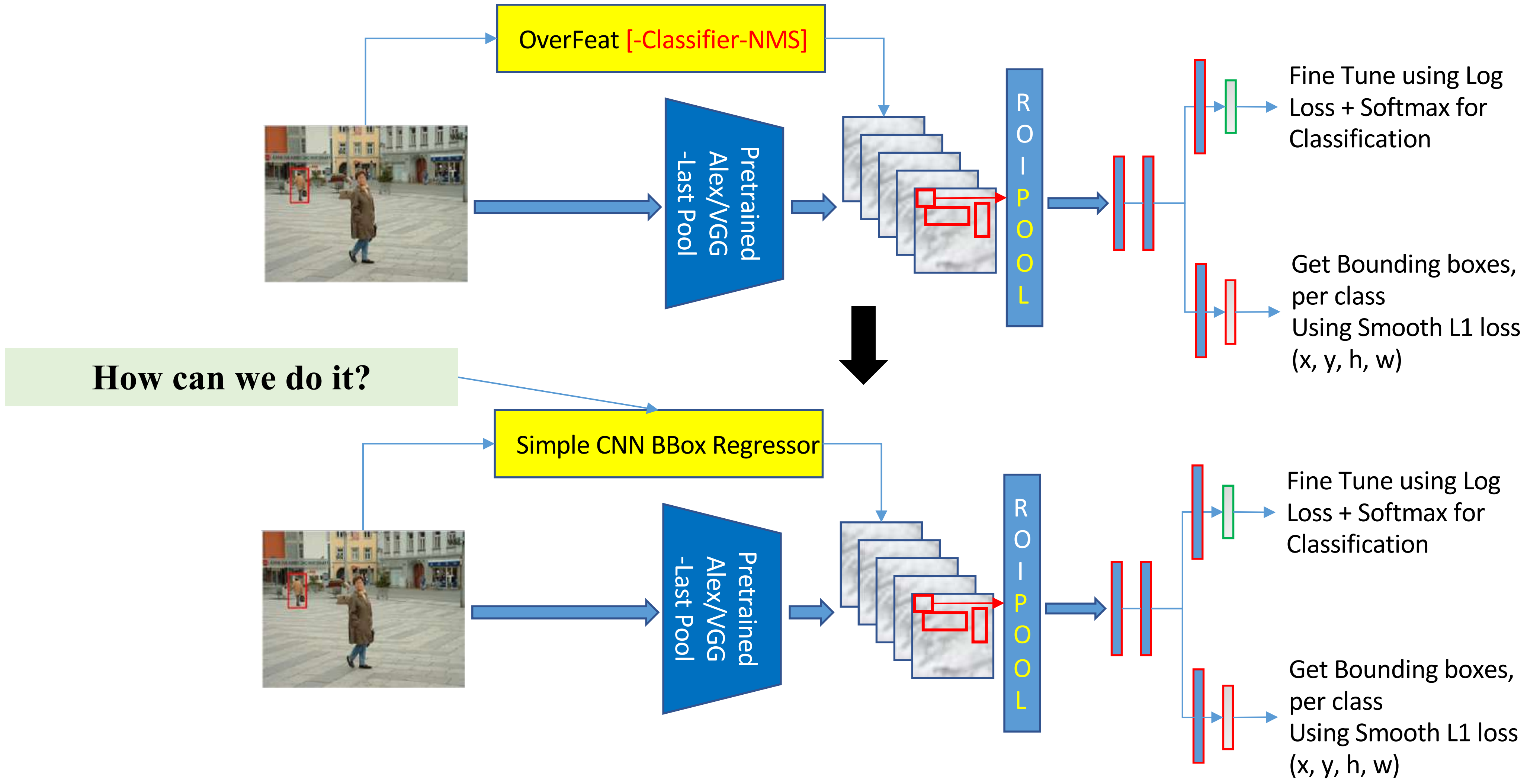
Fast RCNN + Feature Pyramid



- At least 40x60x9 =~ 20,000 proposals -> time consuming.
- Backpropagating through those many proposals is difficult/time consuming

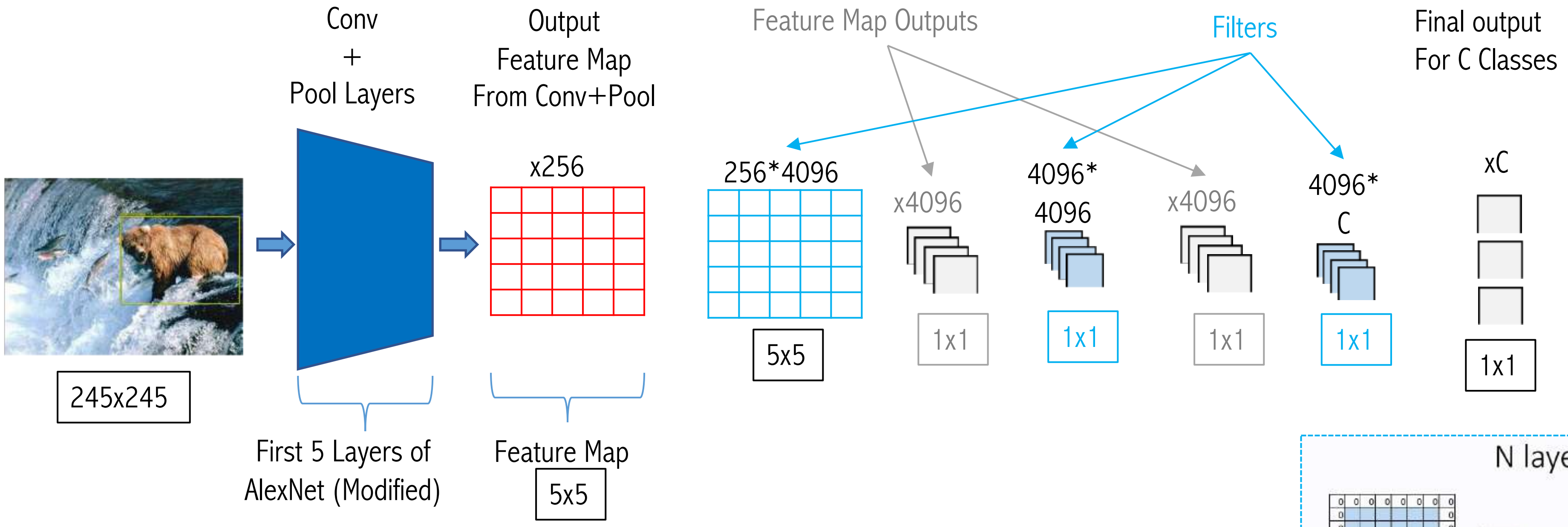
AI VIET NAM
@aivietnam.edu.vn

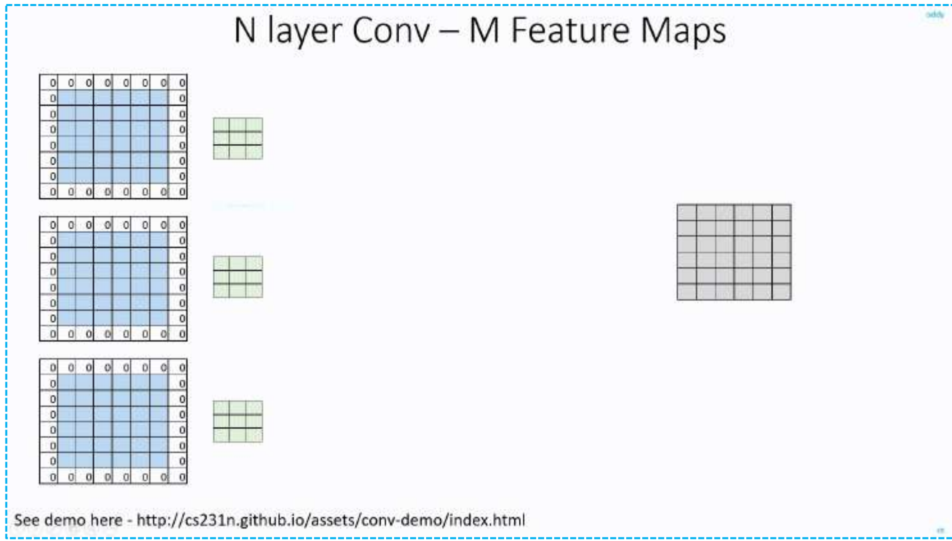
Fast RCNN + Neural Network



Overfeat Classification

Fully Connected layer implemented as a convolution layer





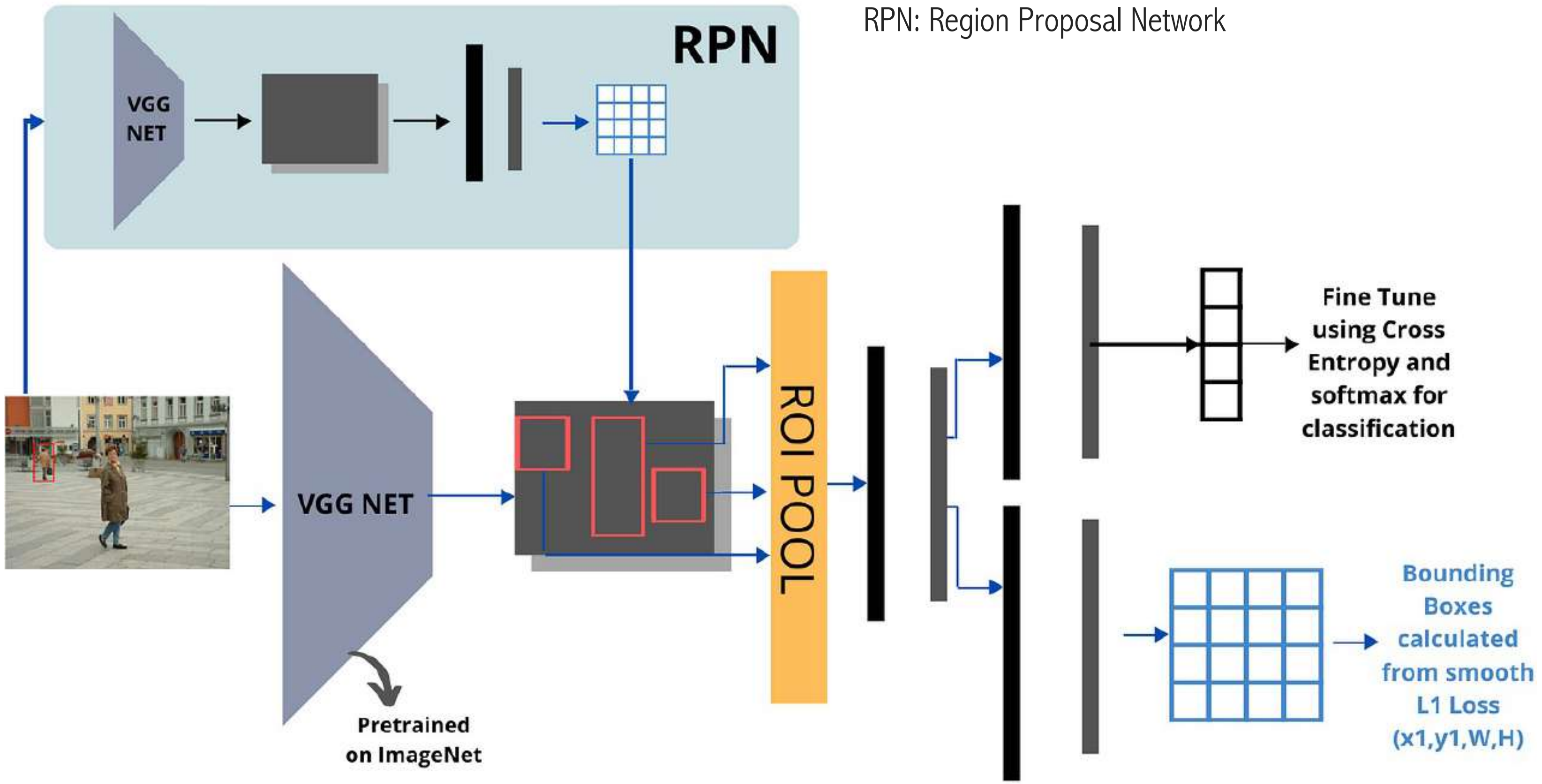
OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks – Sermanet et al

AI VIET NAM
@aivietnam.edu.vn

Solutions for Replace Region Propsals

Consider using a simple CNN BBox regressor in place of Selective Search to get the approximate region proposals of the image which could further be fed to the underlying Fast R-CNN architecture. This is the core idea behind Faster R-CNN.

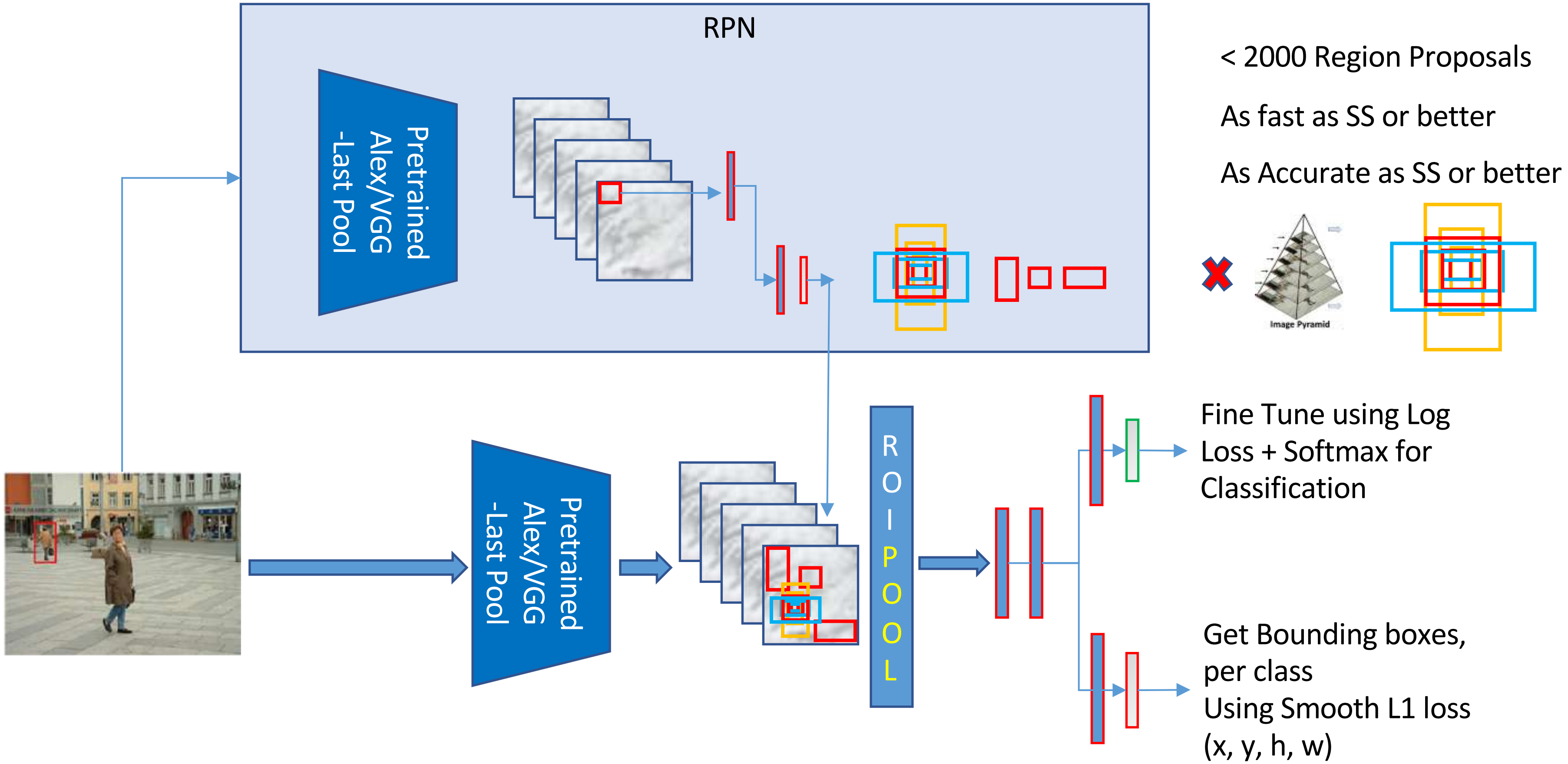
RPN: Region Proposal Network



AI VIET NAM
@aivietnam.edu.vn

Fast RCNN + RPN

How can we do it?



AI VIET NAM
@aivietnam.edu.vn

Review: Ideas for *Localization* using ConvNets

Absolute Bounding Box Regression

Case #1 – Only one object per image

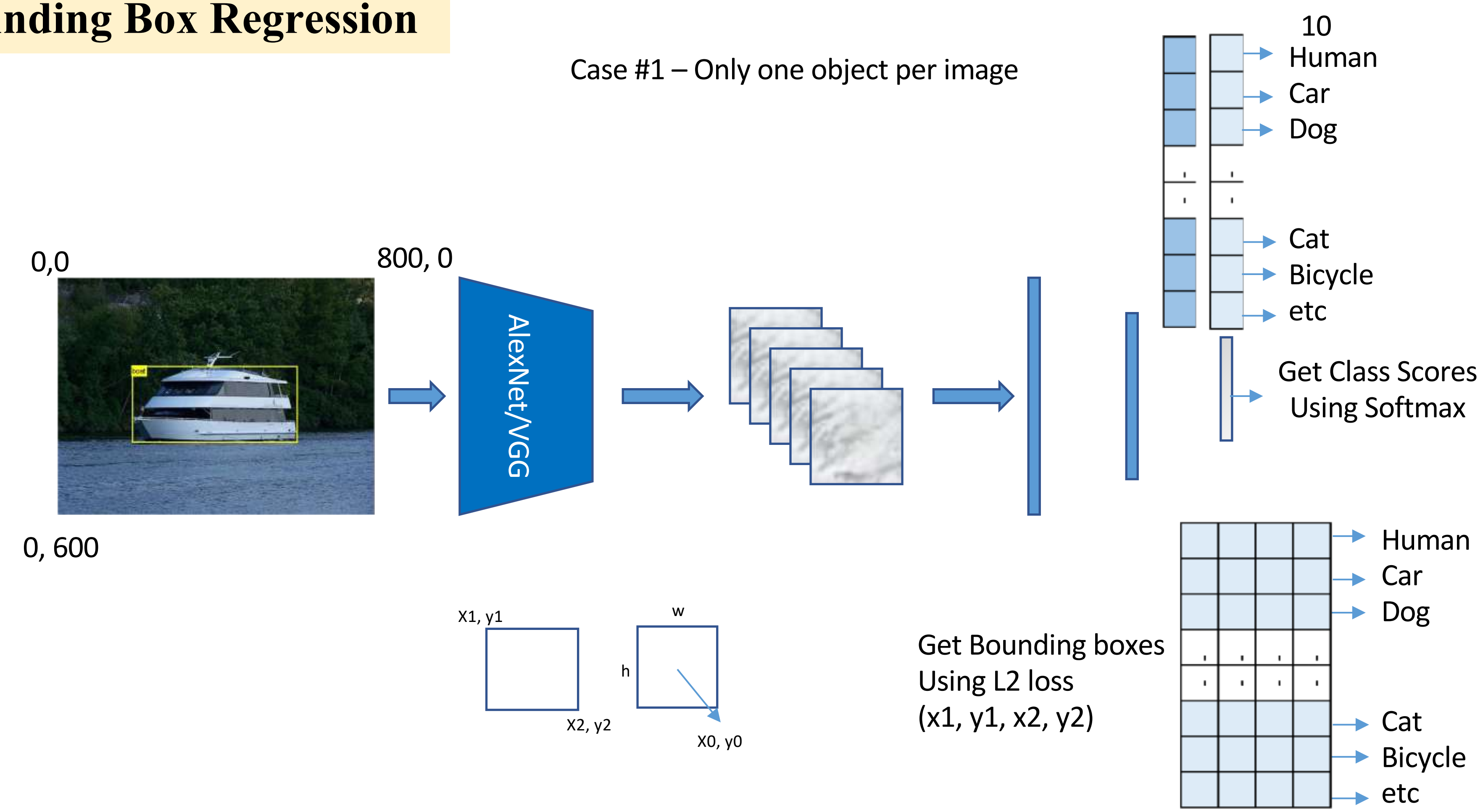
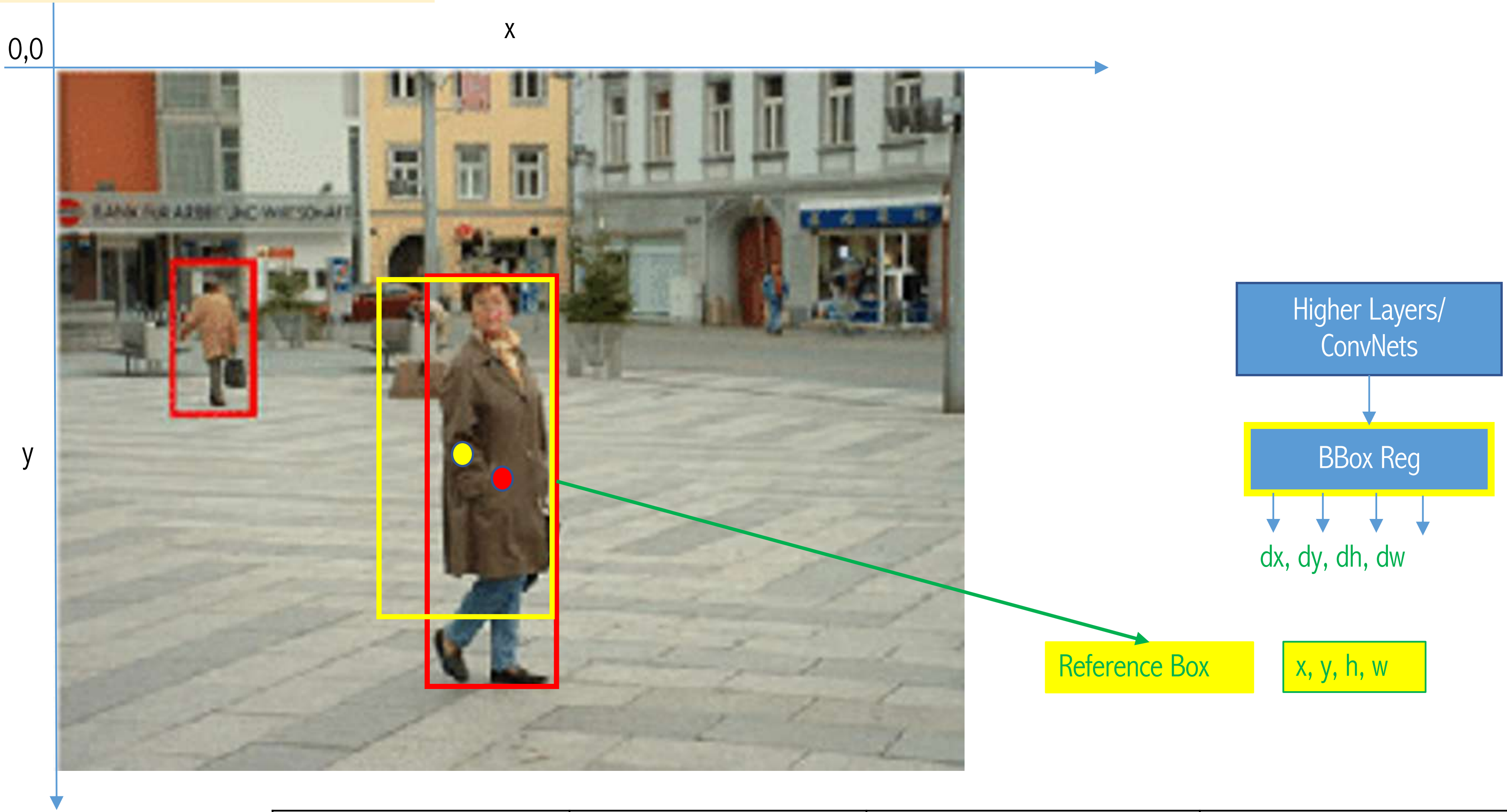


Image Credit - http://host.robots.ox.ac.uk/pascal/VOC/voc2012/examples/index.html

AI VIET NAM
@aivietnam.edu.vn

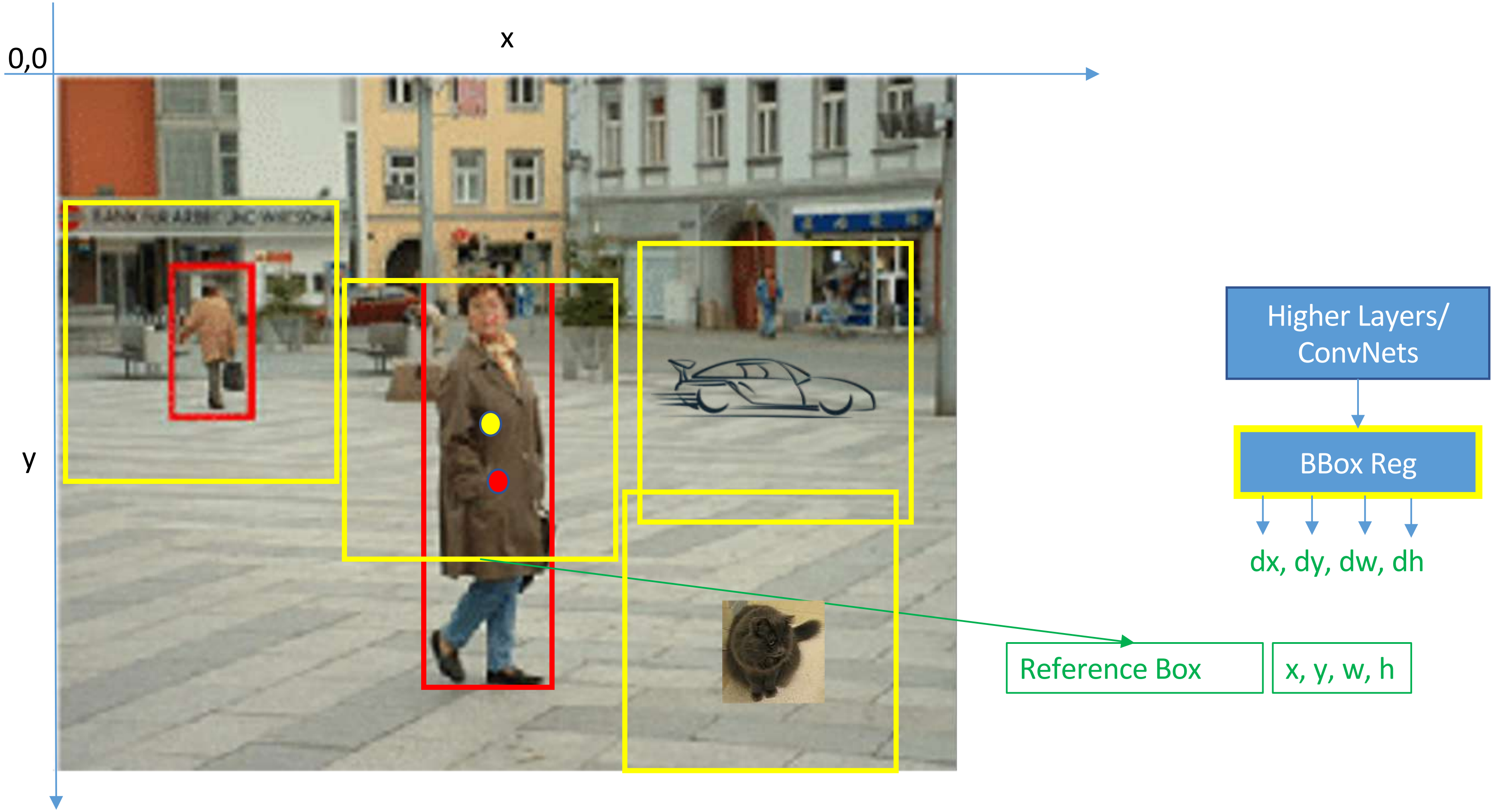
BBox Regression - Relative

Relative Bounding Box Regression



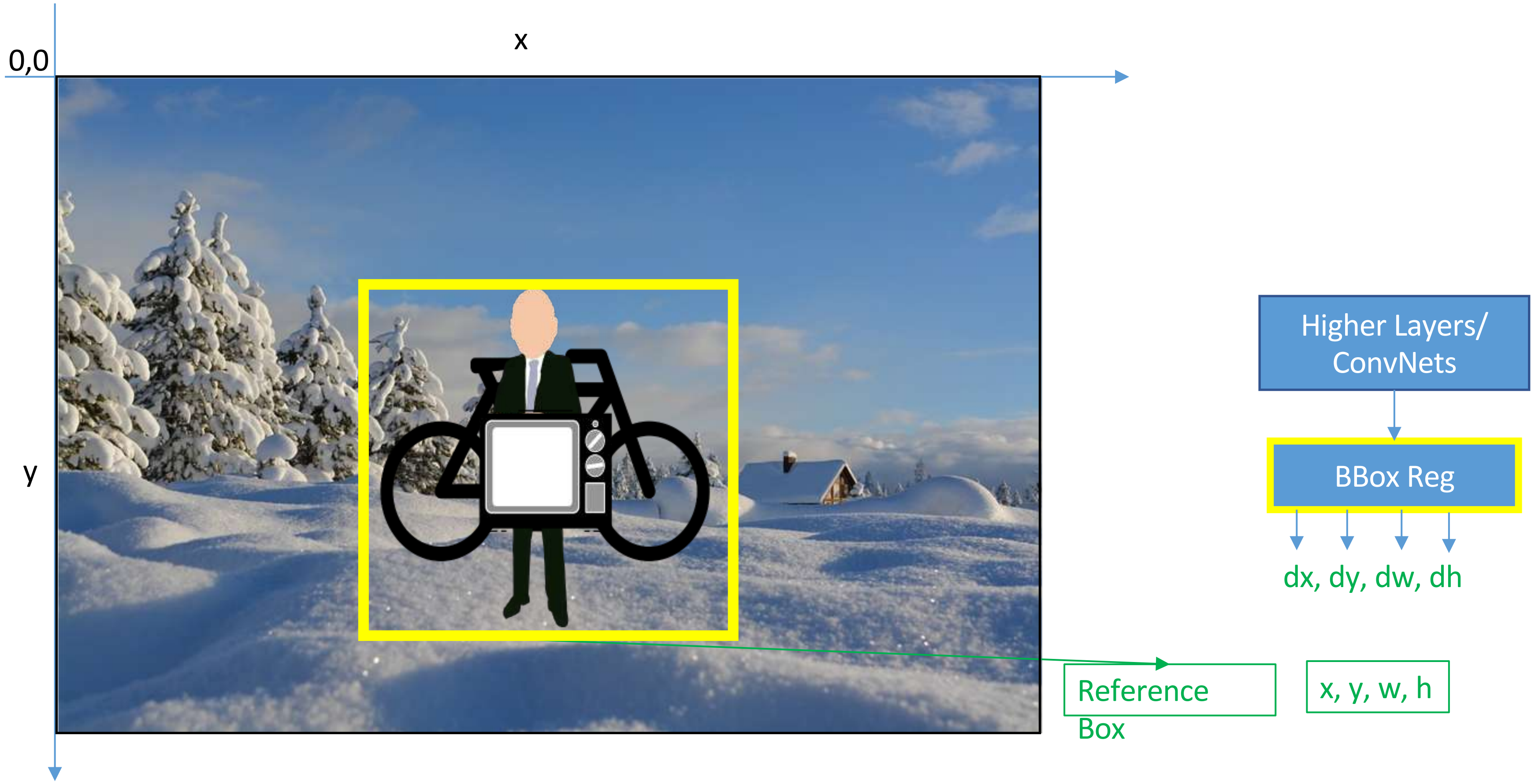
ROI reference				Bbox deltas				Predicted				Expected			
x	y	h	w	dx	dy	dh	dw	x	y	h	w	x	y	h	w
160	240	150	150	18	-22	-30	-125	178	218	120	25	180	220	120	30

Sliding Window as Reference Box

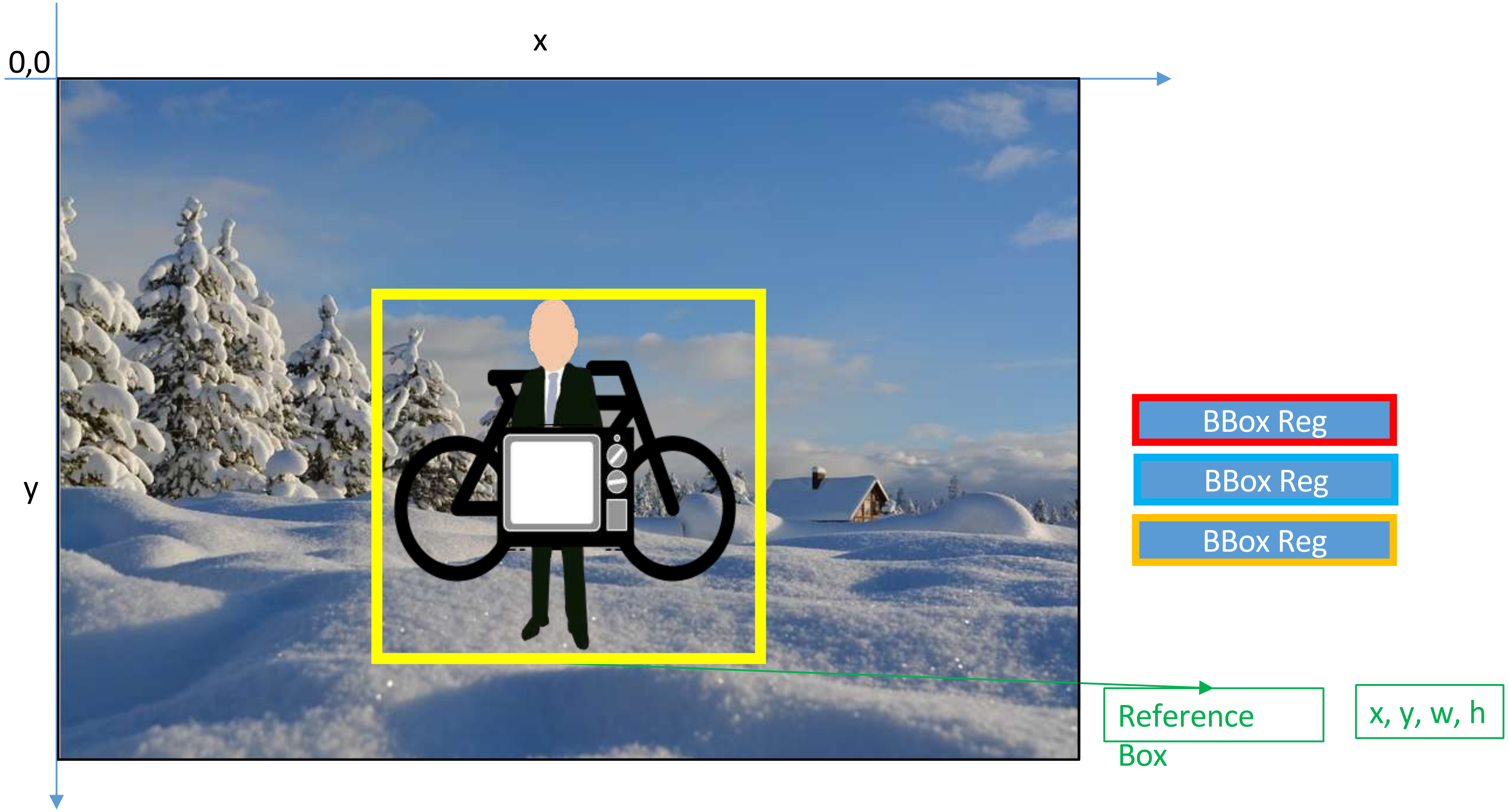


AI VIET NAM
@aivietnam.edu.vn

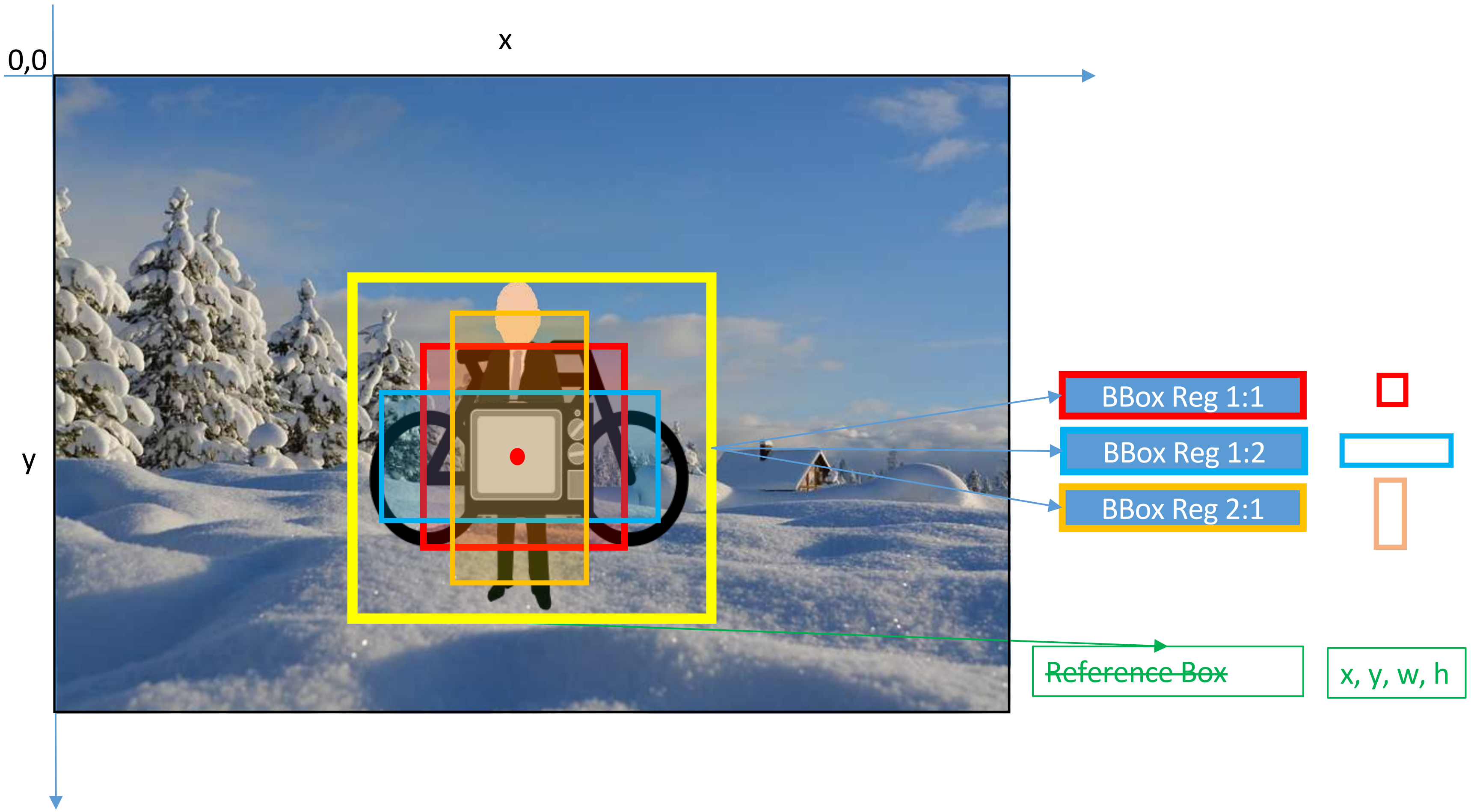
Sliding Window as Reference Box



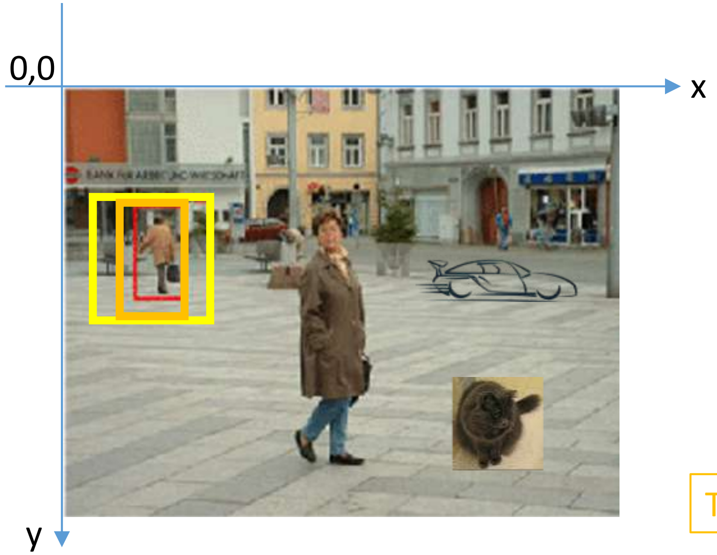
Sliding Window as Reference Box



Sliding Window as Reference Box



Square ROI to Rectangular Proposal



Higher Layers/ ConvNets

BBox Reg 2:1

dx, dy, dw, dh

ROI Centre/W/H

x, y, w, h

Ground Truth

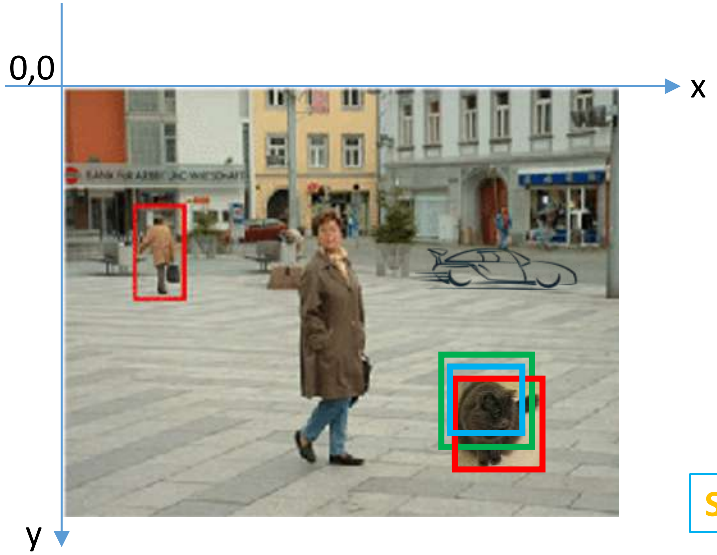
x, y, w, h

Tall Box Centre/W/H

x, y, w, h

ROI reference				Bbox deltas				Predicted				Expected			
x	y	h	w	dx	dy	dh	dw	x	y	h	w	x	y	h	w
160	240	130	65	18	-22	-10	-32	178	218	120	33	180	220	120	30

Square ROI to Rectangular Proposal



Higher Layers/ ConvNets

BBox Reg 1:1

dx, dy, dw, dh

ROI Centre/W/H

x, y, w, h

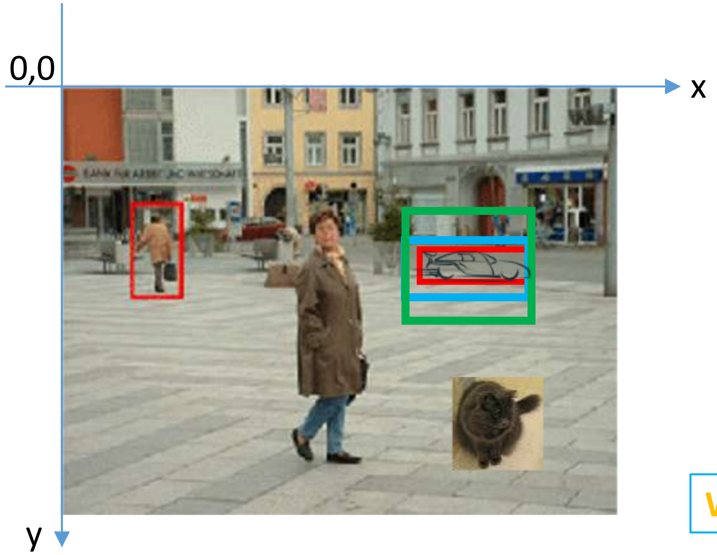
Ground Truth

x, y, w, h

Square Box Centre/W/H

x, y, w, h

Square ROI to Rectangular Proposal



ROI Centre/W/H

x, y, w, h

Wide Box Centre/W/H

x, y, w, h

Higher Layers/ ConvNets

BBox Reg 1:2

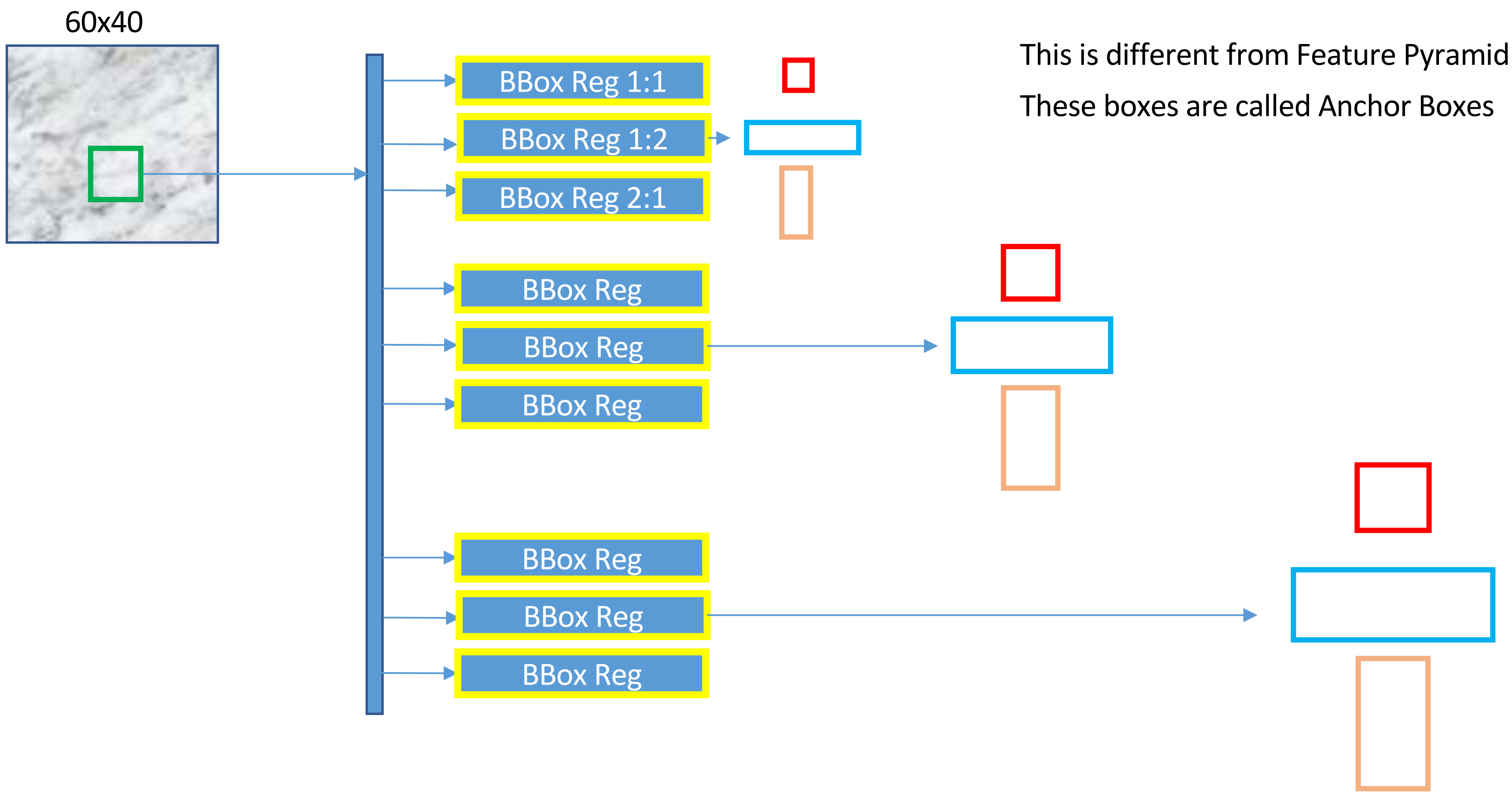
dx, dy, dw, dh

Ground Truth

x, y, w, h

AI VIET NAM
@aivietnam.edu.vn

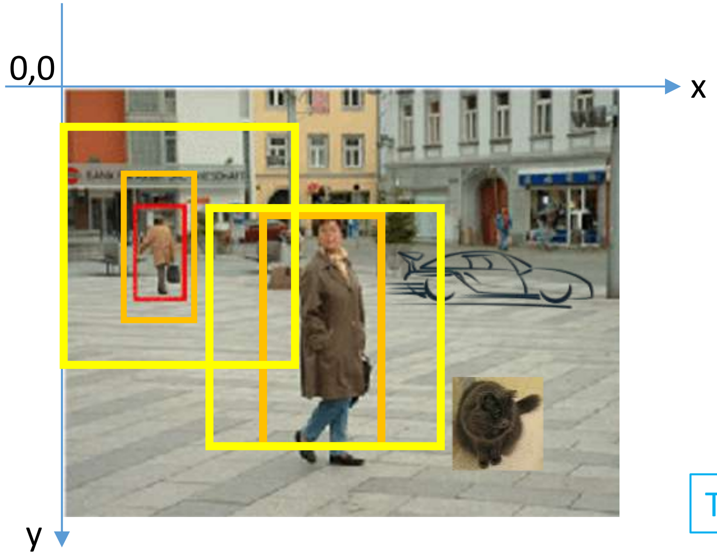
Multiple BBox Reg using Reference Boxes



This is different from Feature Pyramid

These boxes are called Anchor Boxes

Bigger Objects?



Higher Layers/ ConvNets

BBox Reg

dx, dy, dw, dh

SW Centre/W/H

x, y, w, h

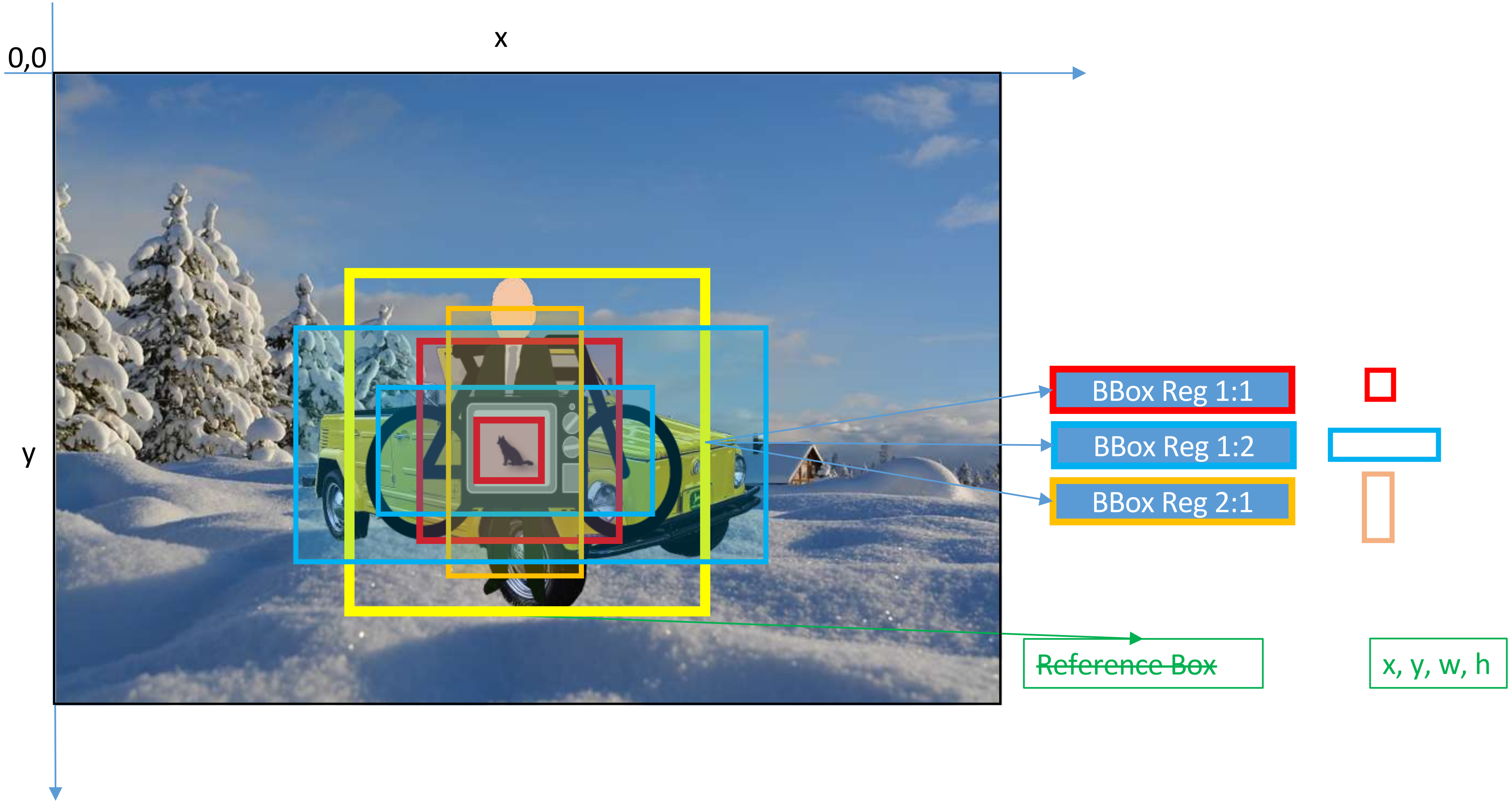
Tall Box Centre/W/H

x, y, w, h

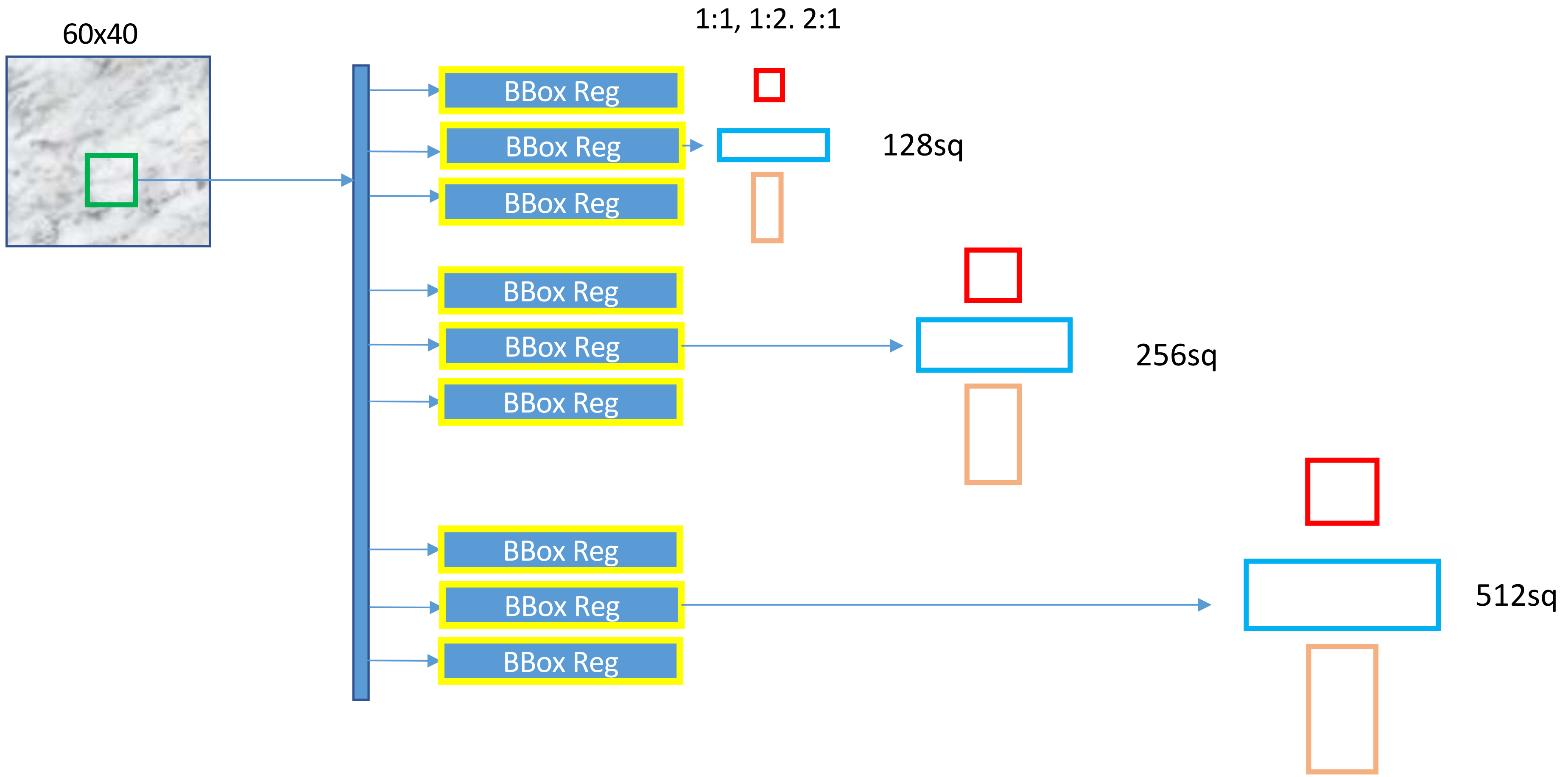
Ground Truth

x, y, w, h

Bigger Objects?

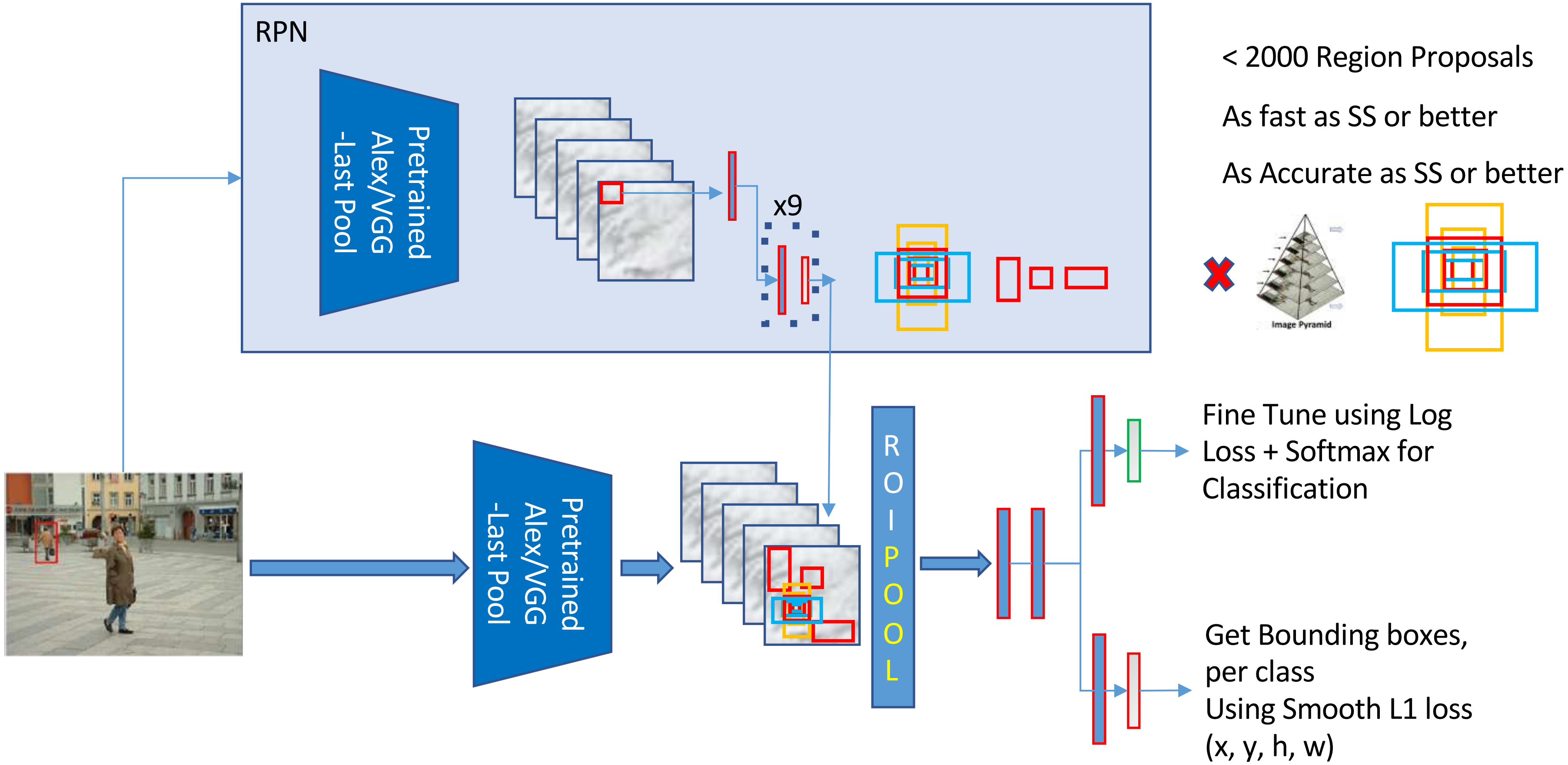


Multiple BBox Reg using Anchor Boxes



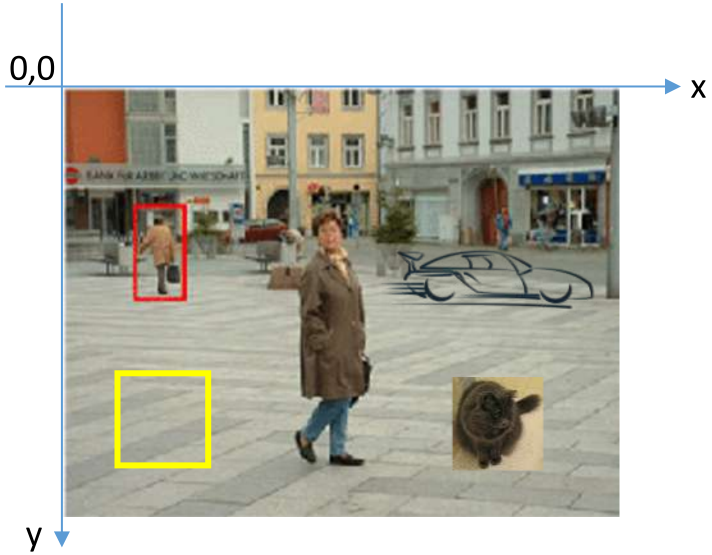
AI VIET NAM
@aivietnam.edu.vn

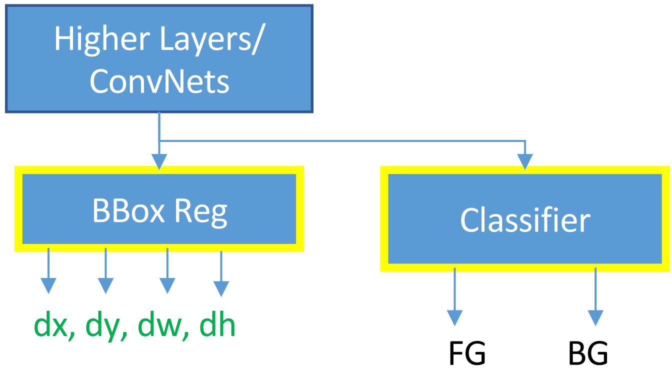
Fast RCNN + RPN



AI VIET NAM
@aivietnam.edu.vn

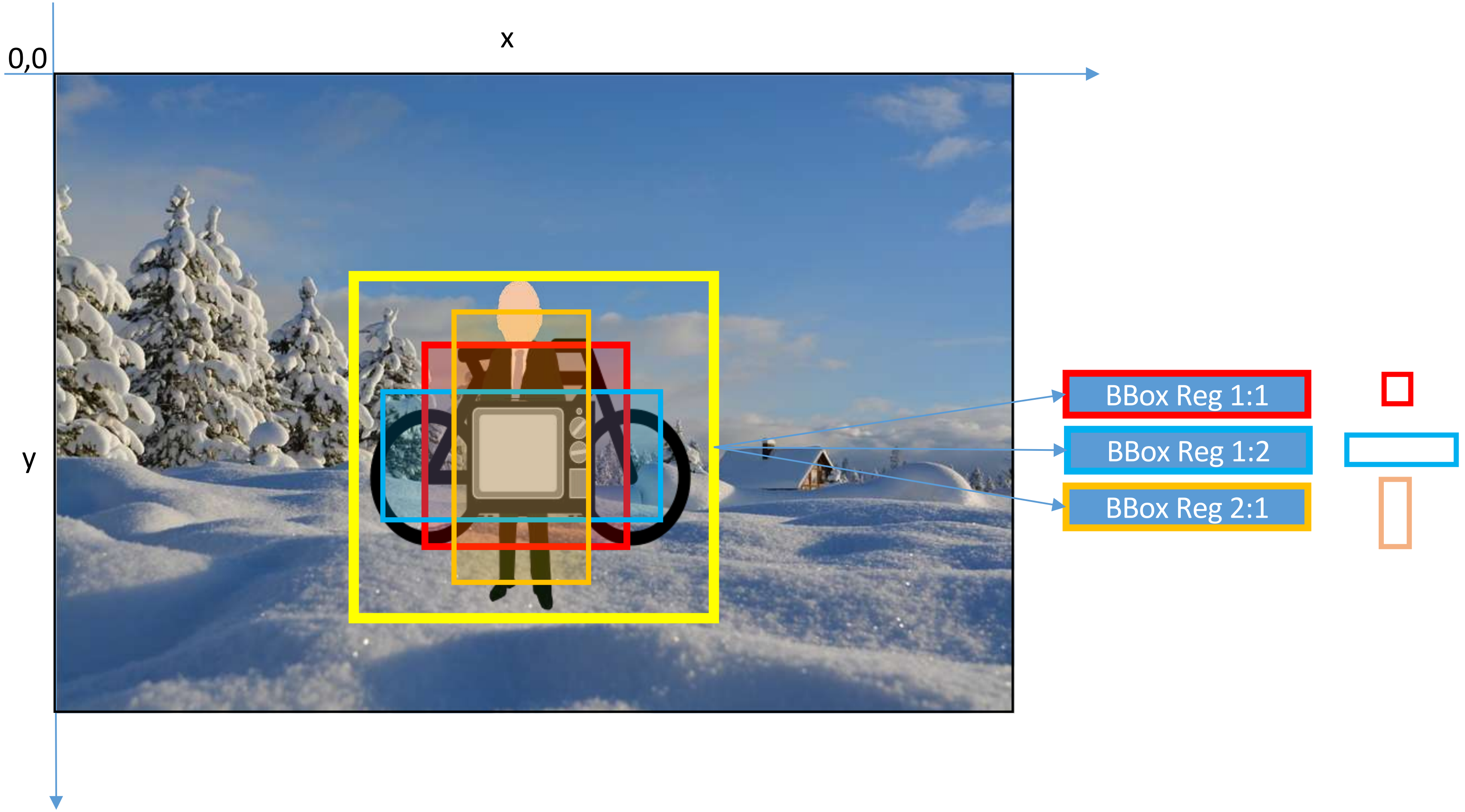
How to reduce number of proposals?



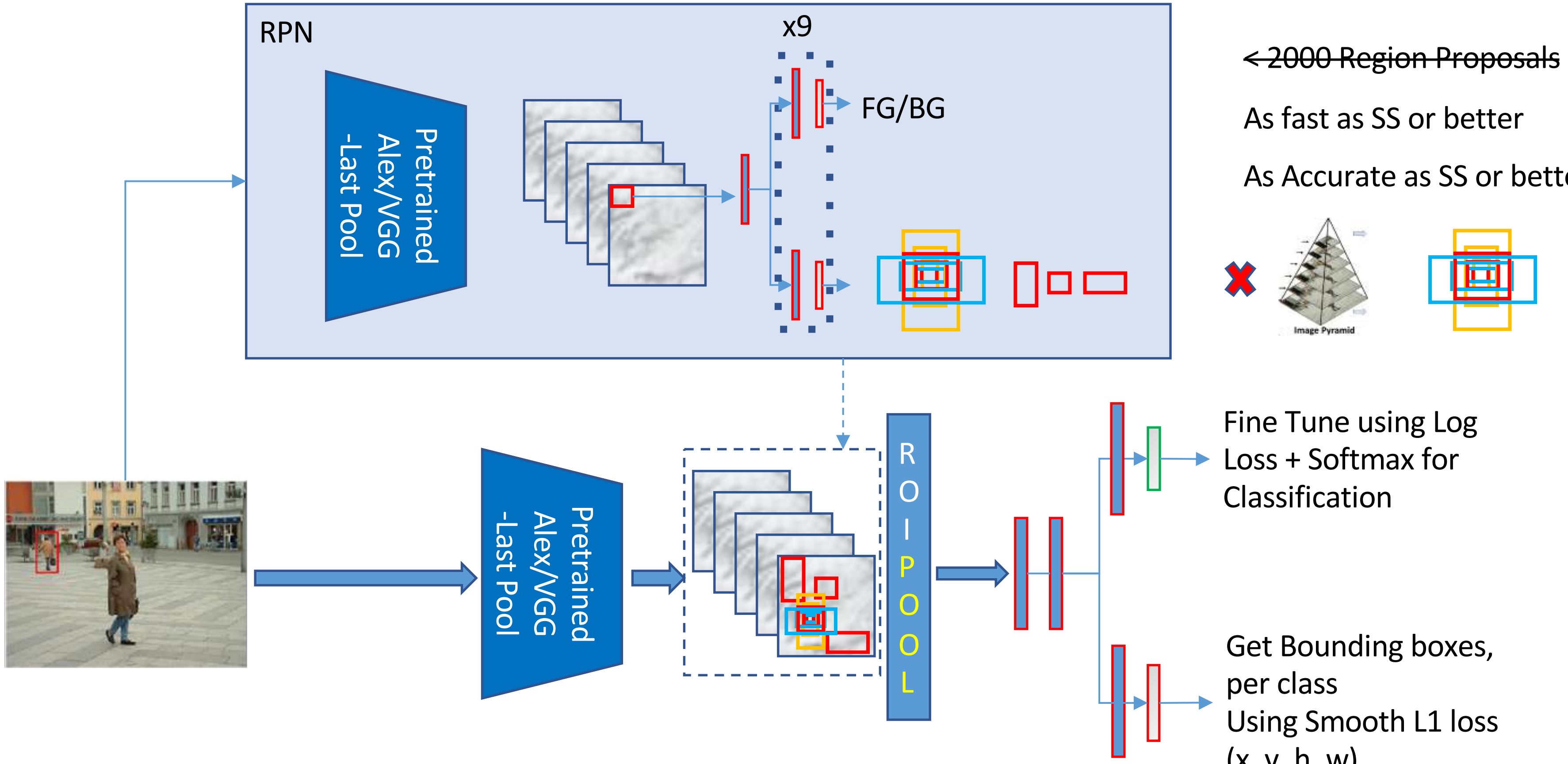


AI VIET NAM
@aivietnam.edu.vn

Faster RCNN – Training Anchor Boxes - Labelling



Fast RCNN + RPN



~~< 2000 Region Proposals~~

As fast as SS or better

As Accurate as SS or better

Fine Tune using Log Loss + Softmax for Classification

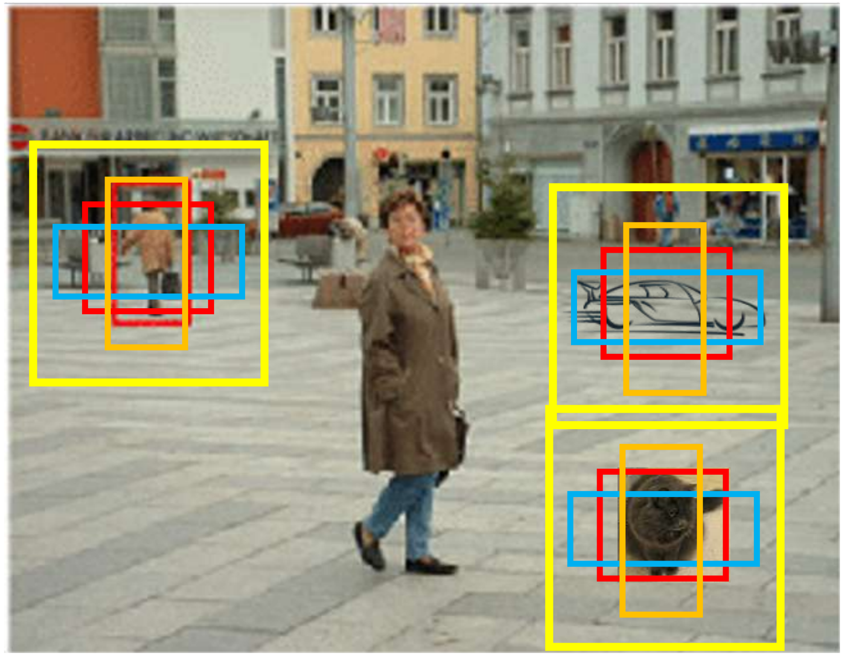
Get Bounding boxes, per class
Using Smooth L1 loss
(x, y, h, w)

Faster RCNN – Training Anchor Boxes - Labelling

BBox Reg 1:1

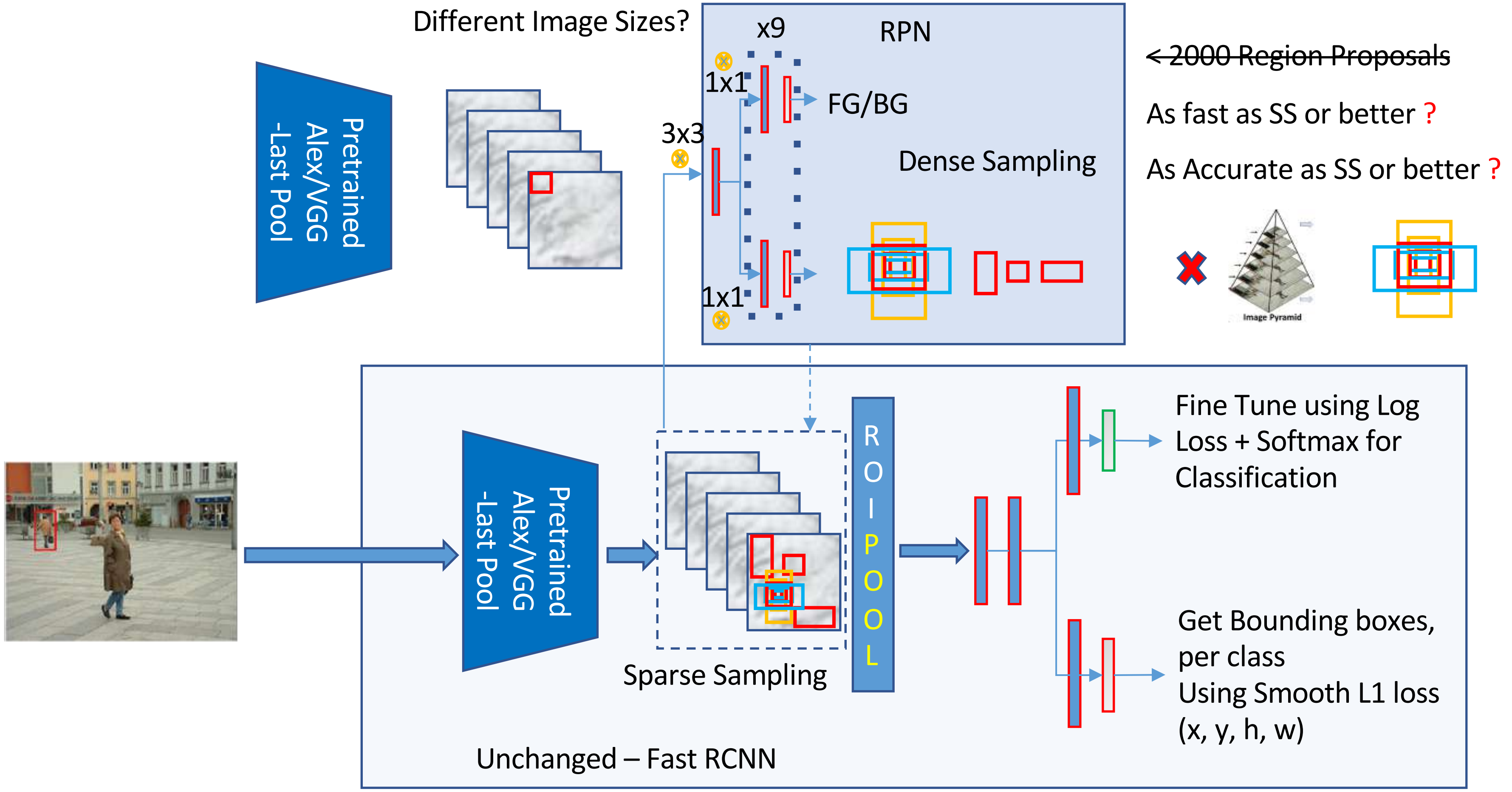
BBox Reg 1:2

BBox Reg 2:1



AI VIET NAM
@aivietnam.edu.vn

Fast RCNN + RPN = Faster RCNN

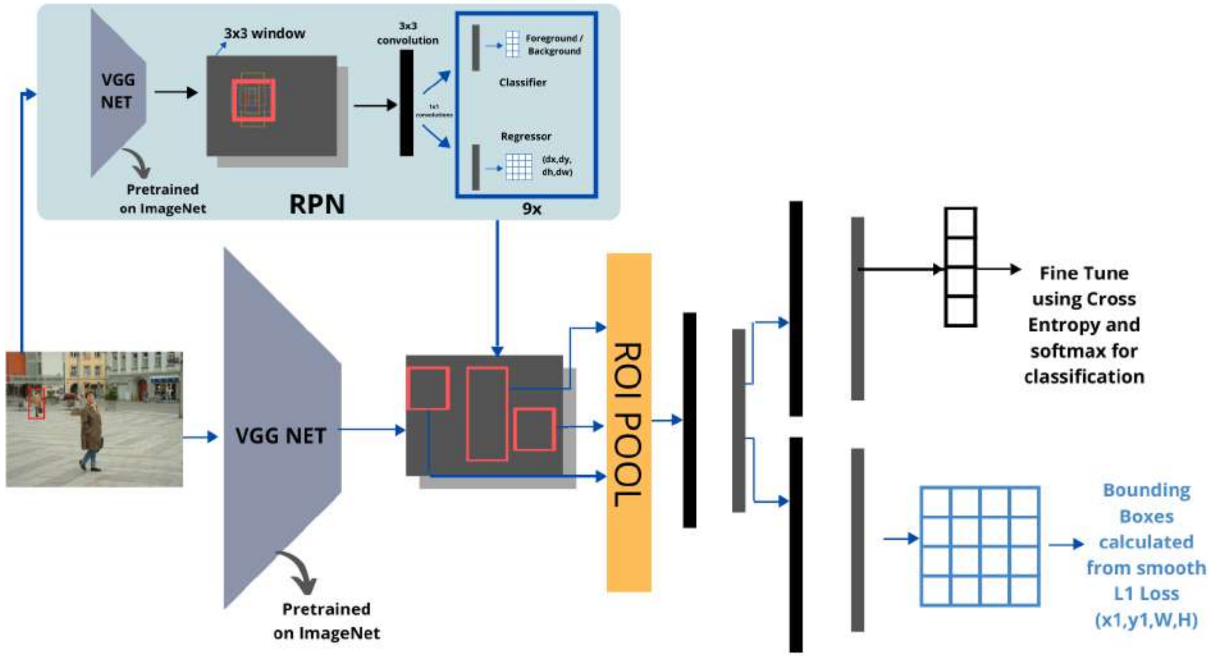


AI VIET NAM
@aivietnam.edu.vn

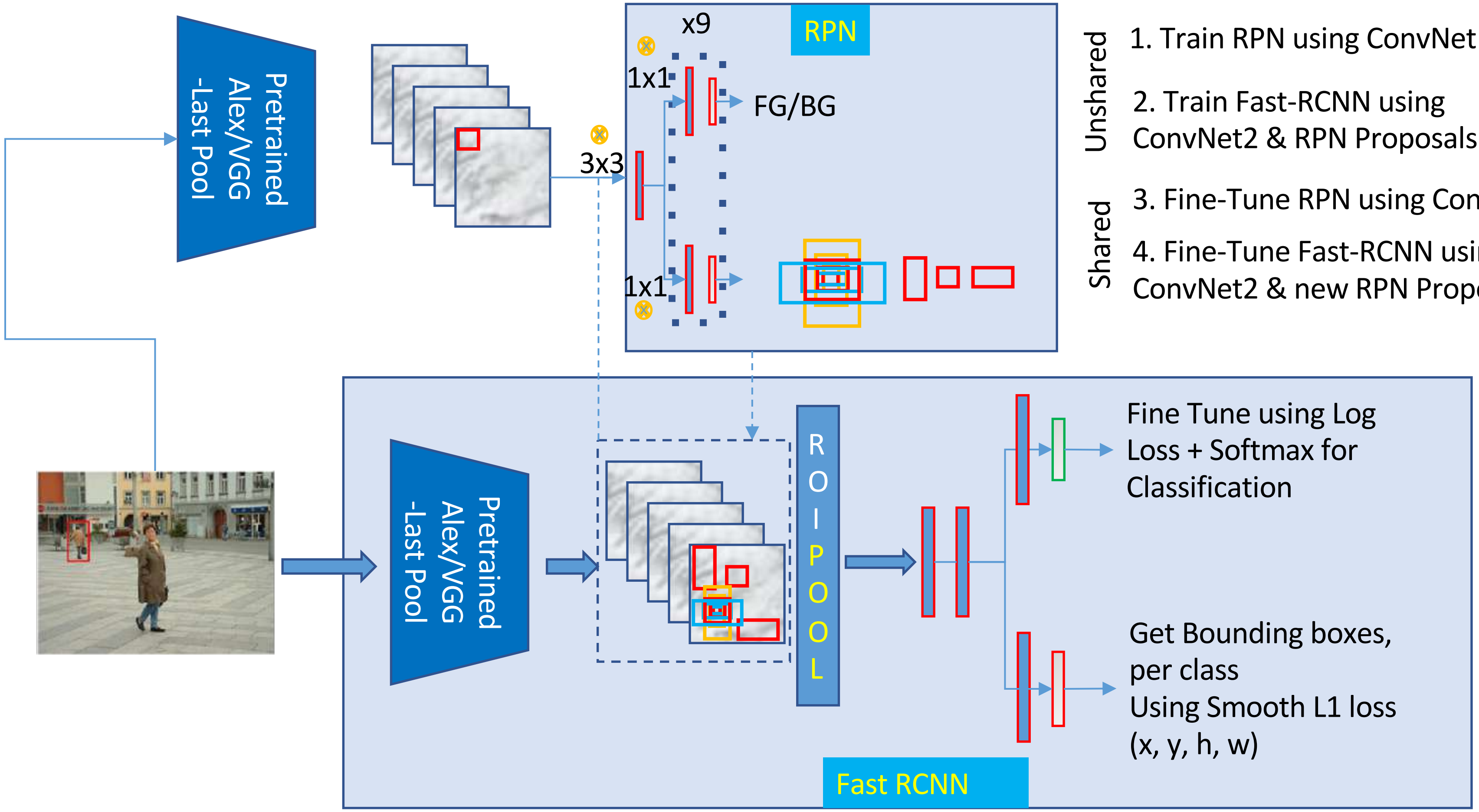
Faster & Accurate than SS?

Time in ms

model	system	conv	proposal	region-wise	total	rate	mAP
VGG	SS + Fast R-CNN	146	1510	174	1830	0.5 fps	66.9
VGG	RPN + Fast R-CNN	141	**10**	47	**198**	**5 fps**	69.9



Faster RCNN - Training



Unshared
1. Train RPN using ConvNet1
2. Train Fast-RCNN using ConvNet2 & RPN Proposals

Shared
3. Fine-Tune RPN using ConvNet2
4. Fine-Tune Fast-RCNN using ConvNet2 & new RPN Proposals

Object Detection Milestones

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

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

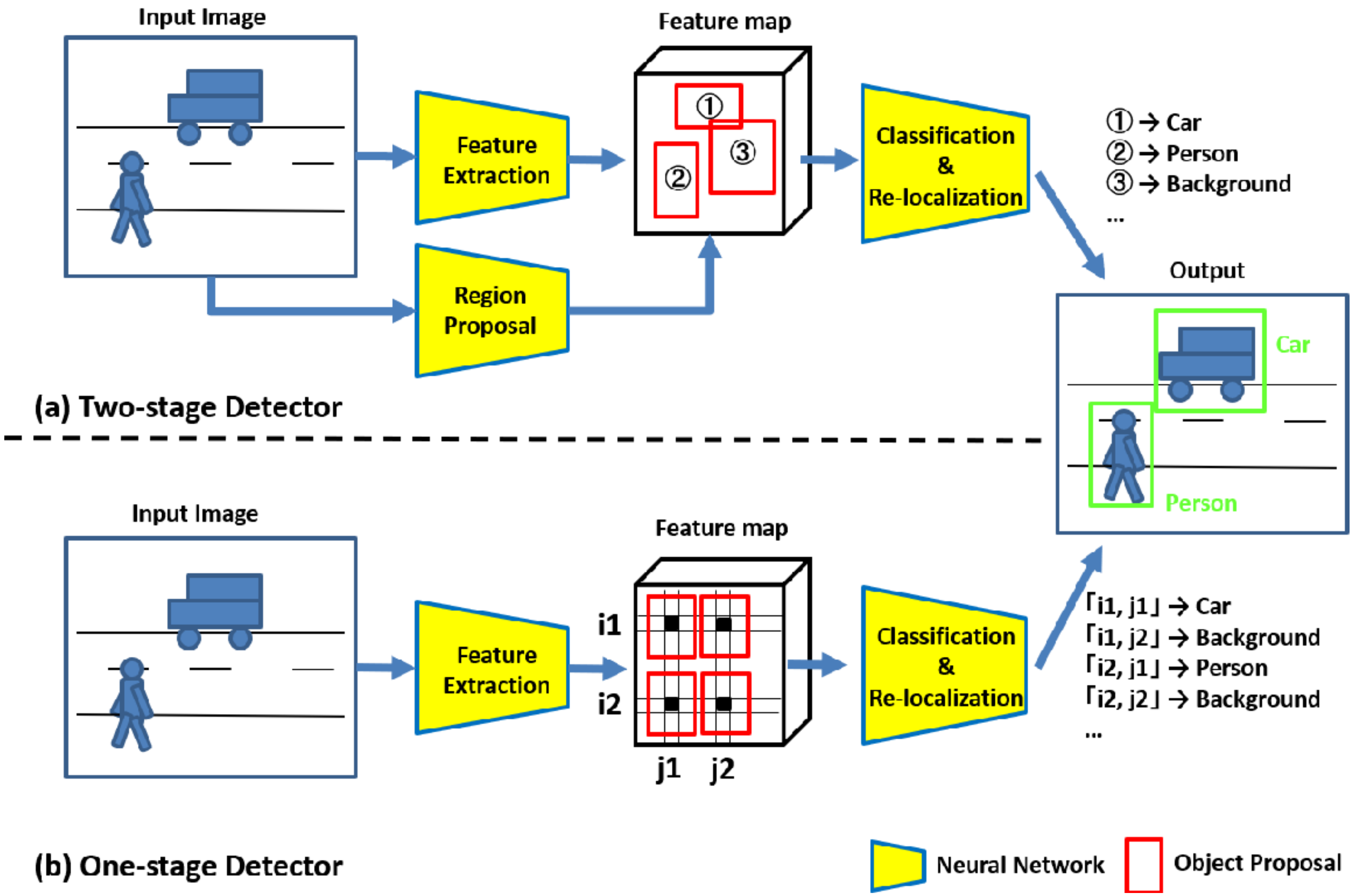
Milestones: CNN based One-stage Detectors
YOLO, SSD, RetinaNet, CornerNet, Center Net,..

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

Outline

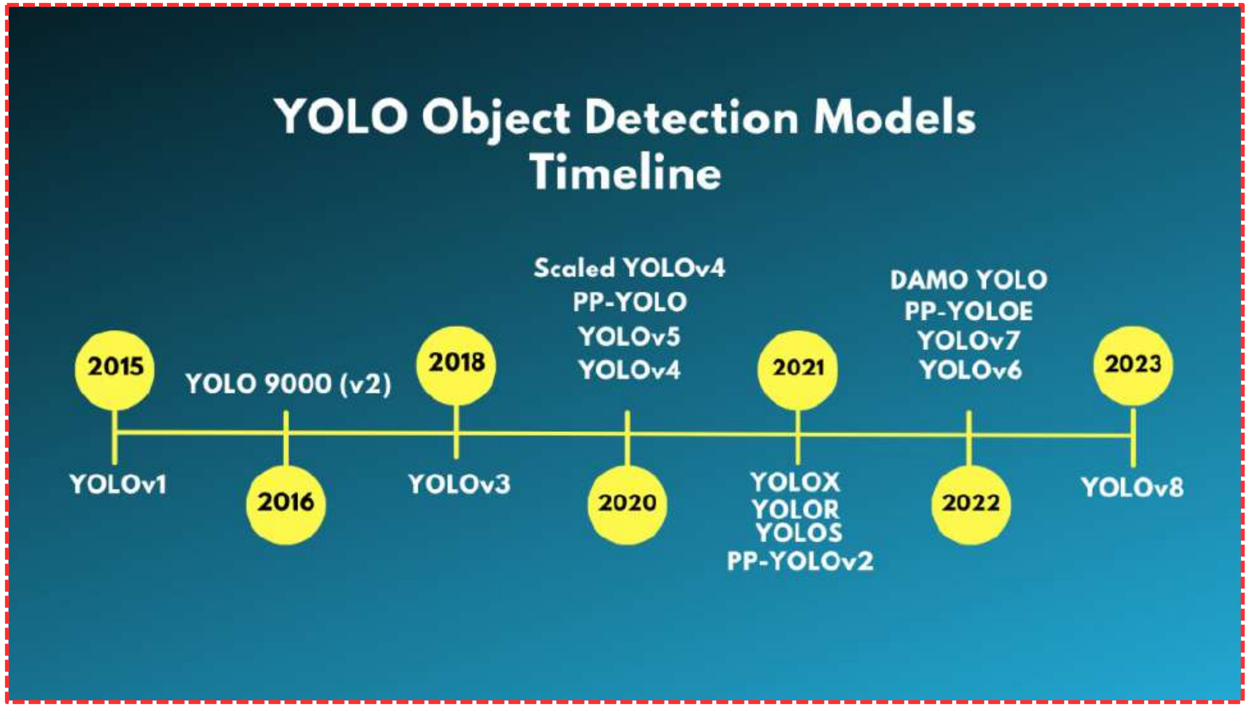
- ➢ CNN Limitations
- ➢ Region Based Convolutional Neural Networks
- ➢ Spatial Pyramid Pooling
- ➢ Fast R-CNN
- ➢ Faster RCNN
- ➢ YOLOv1-v2

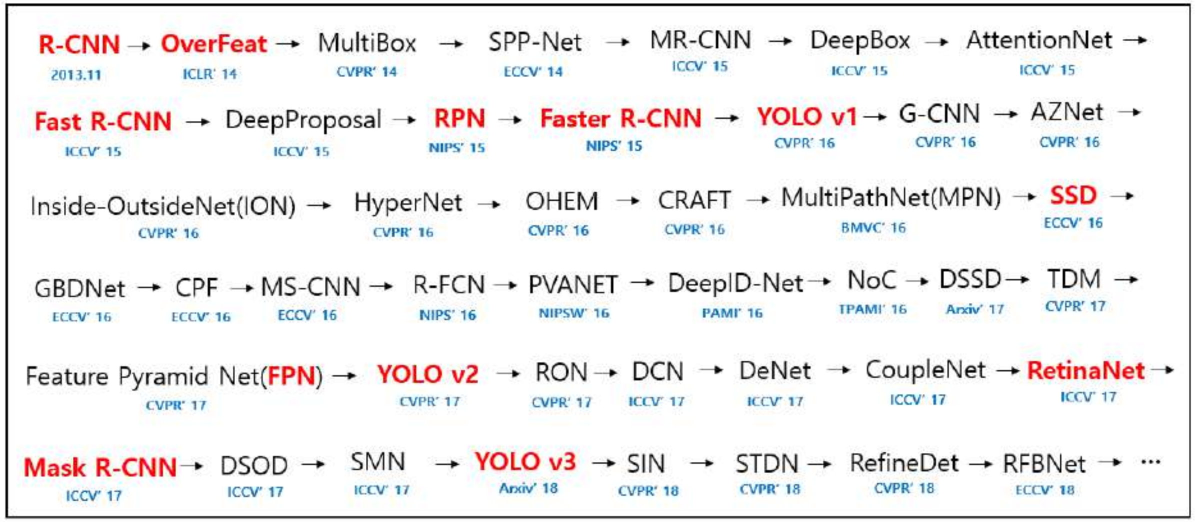
One-stage vs. Two-stage Detectors



AI VIET NAM
@aivietnam.edu.vn

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.

Why Study YOLO?

YOLO is an abbreviation for the term 'You Only Look Once'. This is an algorithm that detects and recognizes various objects in a picture (in real-time). Object detection in YOLO is done as a regression problem and provides the class probabilities of the detected images.

YOLO algorithm employs convolutional neural networks (CNN) to detect objects in real-time. As the name suggests, the algorithm requires only a single forward propagation through a neural network to detect objects.

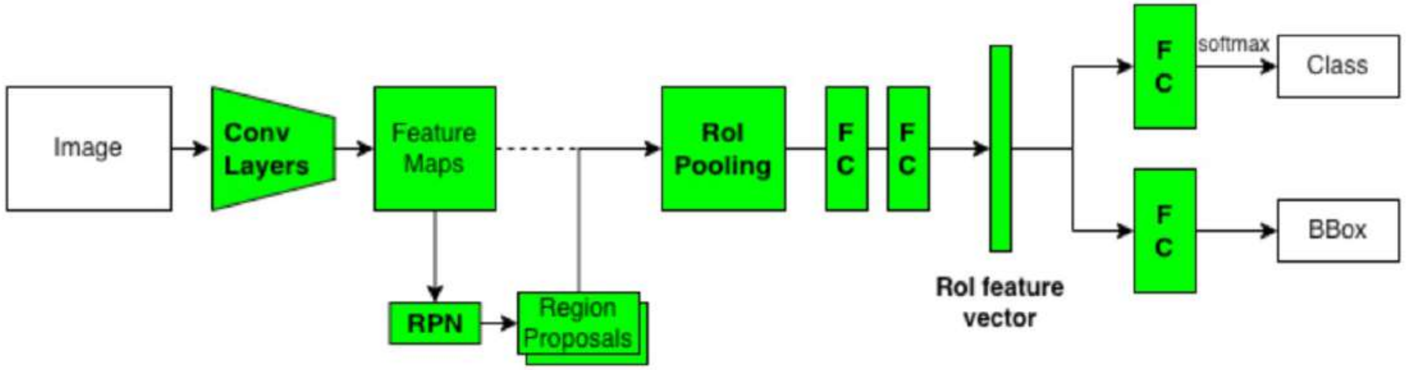
- **Speed**: This algorithm improves the speed of detection because it can predict objects in real-time.
- **High accuracy:** YOLO is a predictive technique that provides accurate results with minimal background errors.
- **Learning capabilities**: The algorithm has excellent learning capabilities that enable it to learn the representations of objects and apply them in object detection.

For small datasets and limited computational power, YOLOv8 might be a better choice as it has been optimized for speed and accuracy. However, if you have more significant datasets and more complex object detection tasks, DETR could be a better fit due to its ability to handle object detection without pre-defined anchor boxes

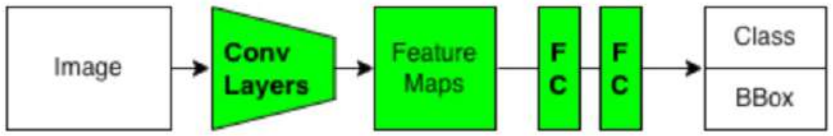
AI VIET NAM
@aivietnam.edu.vn

YOLO: Motivations

In 2015, Joseph Redmon (University of Washington) developed YOLO. One of his co-authors, Ross Girshick (Microsoft Research), published a paper for Faster R-CNN around the same time. They probably shared common ideas in computer vision research as there are some similarities between YOLO and Faster R-CNN. For example, both models apply convolutional layers on input images to generate feature maps. However, Faster R-CNN uses a two-stage object detection pipeline, while YOLO has no separate region proposal step and is much faster than Faster R-CNN.



Faster R-CNN pipeline



YOLO pipeline

YOLO: Motivations

In 2015, Joseph Redmon (University of Washington) developed YOLO. One of his co-authors, Ross Girshick (Microsoft Research), published a paper for Faster R-CNN around the same time. They probably shared common ideas in computer vision research as there are some similarities between YOLO and Faster R-CNN. For example, both models apply convolutional layers on input images to generate feature maps. However, Faster R-CNN uses a two-stage object detection pipeline, while YOLO has no separate region proposal step and is much faster than Faster R-CNN.

YOLO has many versions (variants). Joseph Redmon developed the first three versions of YOLO: YOLOv1, v2, and v3. Then, he quit.

After YOLOv3, different groups of people developed their versions of YOLO:

- YOLOv4 by Alexey Bochkovskiy, et al.
- YOLOv5 by Ultralytics
- YOLOv6 by Meituan
- YOLOX by Zheng Ge et al.
- YOLOv7 by Chien-Yao Wang et al. (The same people from YOLOv4)

Joseph Redmon · Feb 20, 2020
@pjreddie · **Follow**
"We shouldn't have to think about the societal impact of our work because it's hard and other people can do it for us" is a really bad argument.

> **Roger Grosse** @RogerGrosse
> Replying to @kevin_zakka and @hardmaru
> To be clear, I don't think this is a positive step. Societal impacts of AI is a tough field, and there are researchers and organizations that study it professionally. Most authors do not have expertise in the area and won't do good enough scholarship to say something meaningful.

Joseph Redmon
@pjreddie · **Follow**

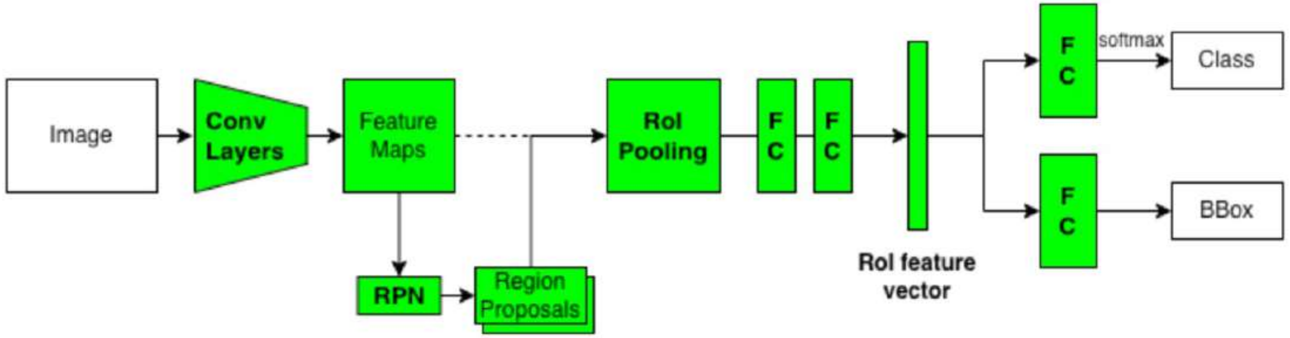
I stopped doing CV research because I saw the impact my work was having. I loved the work but the military applications and privacy concerns eventually became impossible to ignore.

> **Roger Grosse** @RogerGrosse
> Replying to @skoularidou
> What's an example of a situation where you think someone should decide not to submit their paper due to Broader Impacts reasons?

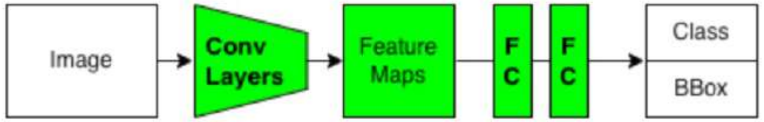
AI VIET NAM
@aivietnam.edu.vn

YOLO: Motivations

- Residual blocks
- Bounding box regression
- Intersection Over Union (IOU)



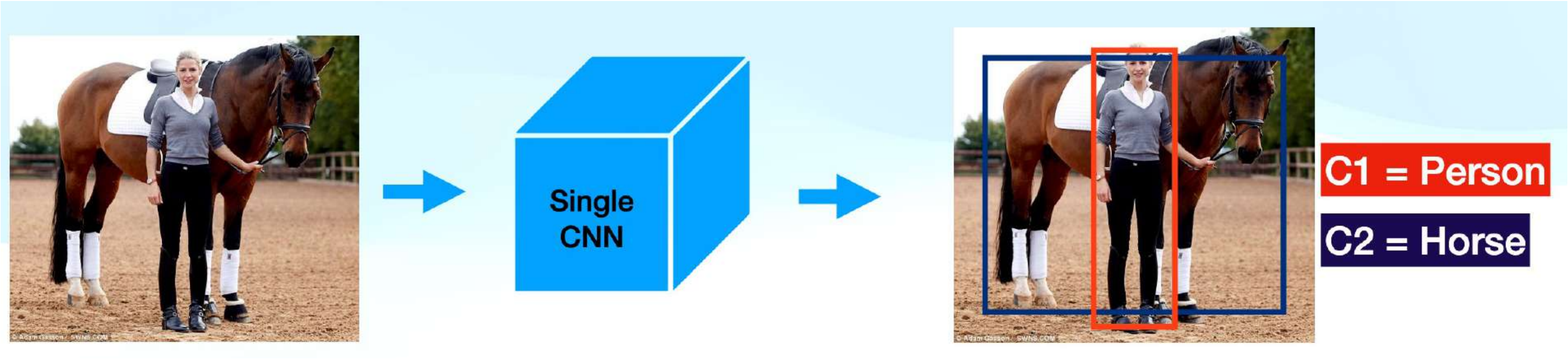
Faster R-CNN pipeline



YOLO pipeline

AI VIET NAM
@aivietnam.edu.vn

YOLO: Motivations



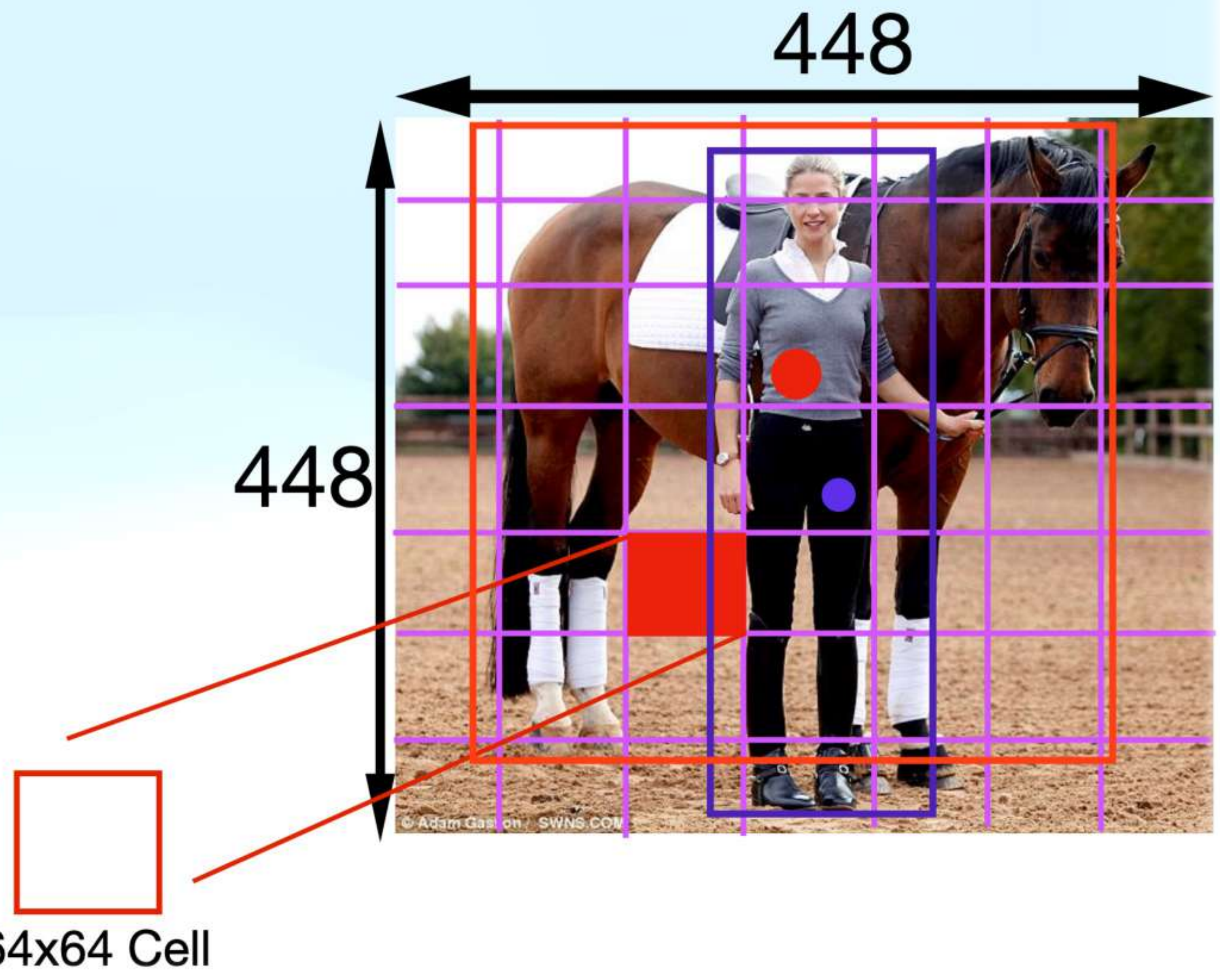
Single stage regression problem

https://github.com/MLForNerds/YOLO-OBJECT-DETECTION-TUTORIALS

YOLO: Motivations

Steps in YOLO

- Take input image
- Resize to 448x448
- Divide into SxS Grid cells
- S=7 in paper
- Each cell is responsible for predicting one object
- Which cells are responsible for person and horse?

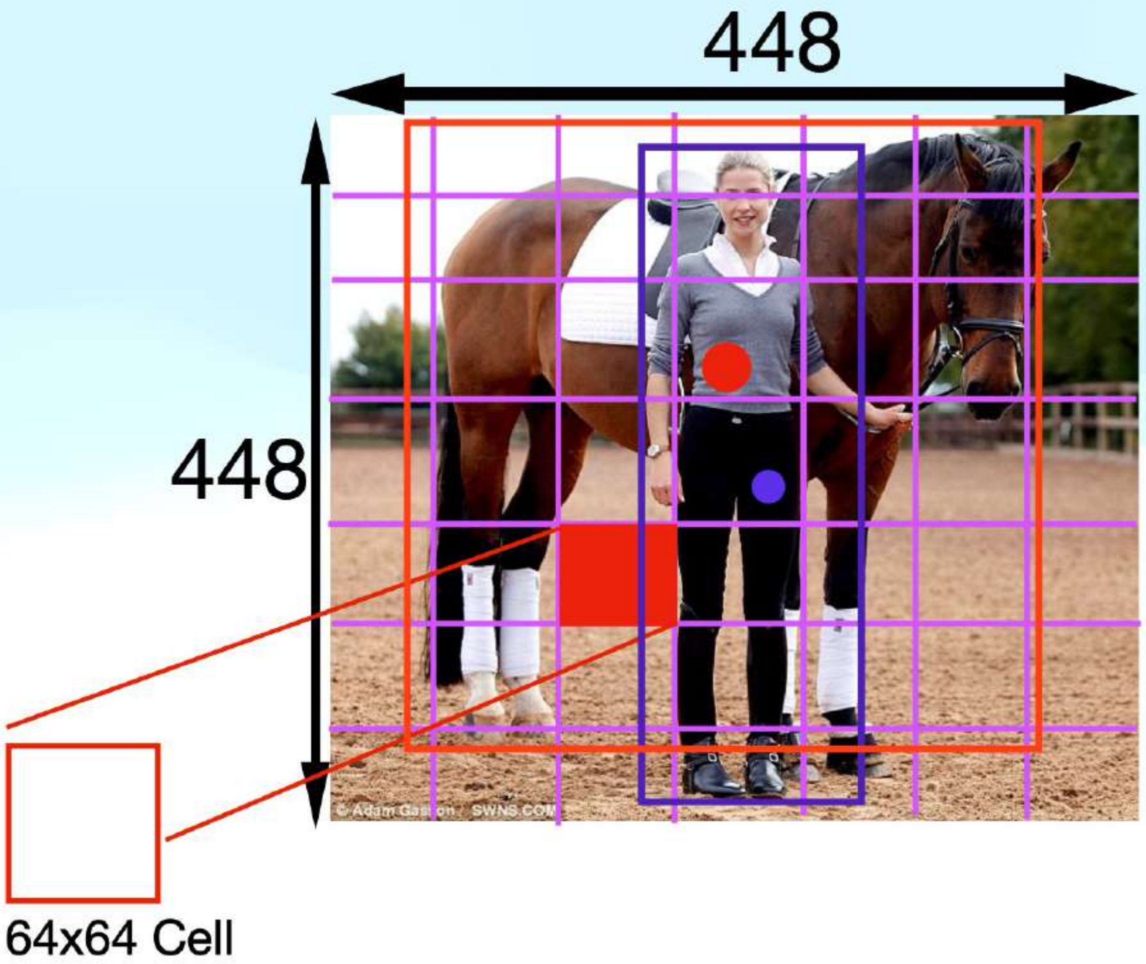


AI VIET NAM
@aivietnam.edu.vn

YOLO: Motivations

Steps in YOLO

- Take input image
- Resize to 448x448
- Divide into SxS Grid cells
- S=7 in paper
- Each cell is responsible for predicting one object
- Which cell is responsible?
- Where Center of object falls into



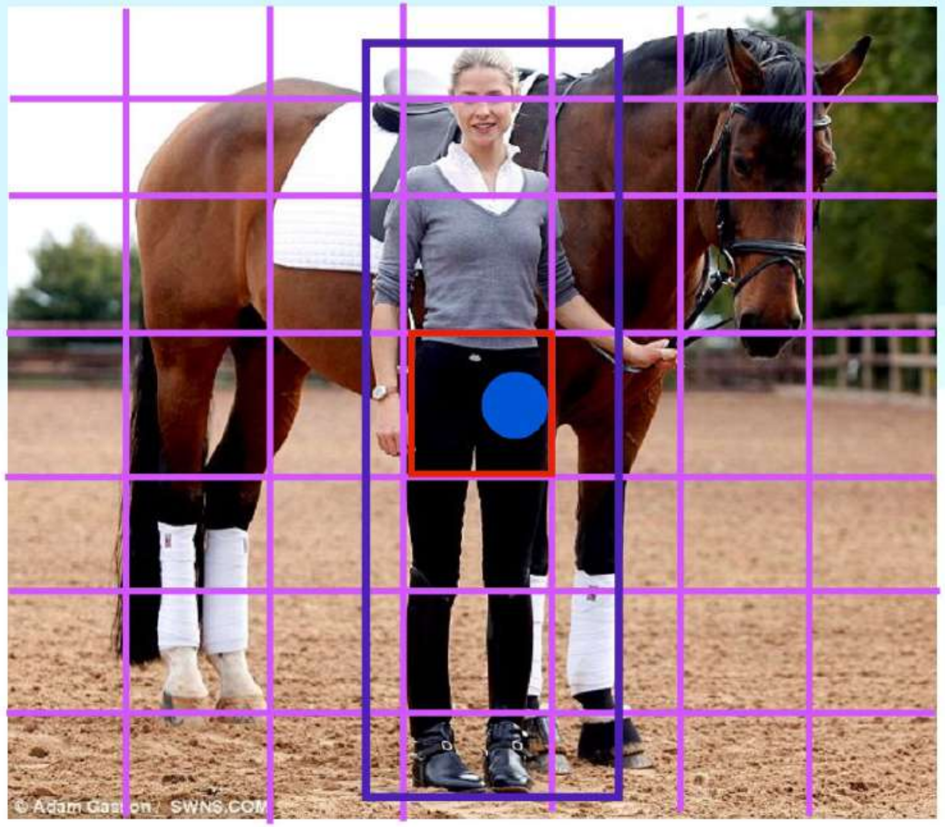
AI VIET NAM
@aivietnam.edu.vn

YOLO: Motivations

Bounding boxes

- Box - x,y,w,h
- How are these encoded?
- Relative to Grid cell that the object Center falls into.

(200, 311, 142, 250)



Bounding Boxes

- Center point (x,y): Relative to anchor that (x,y) falls into.

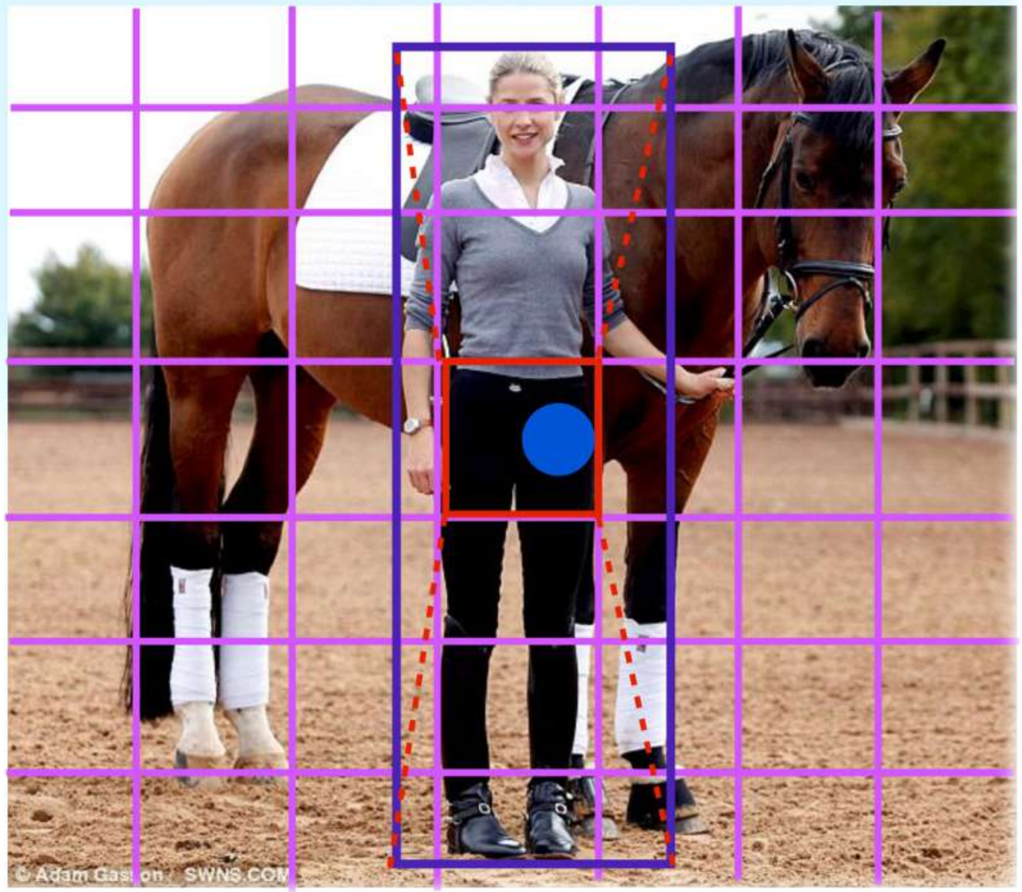
$$\Delta x = (x - x_a)/64$$
$$\Delta y = (y - y_a)/64$$

(x_a, y_a): the coordinate of left-top point

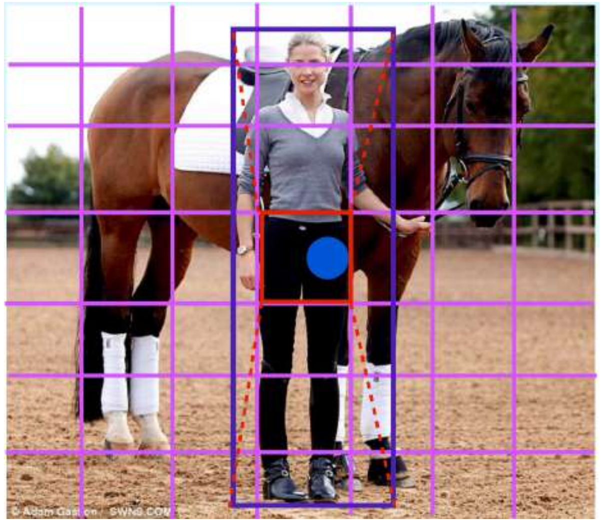
- Width/height (w,h): relative to the whole image

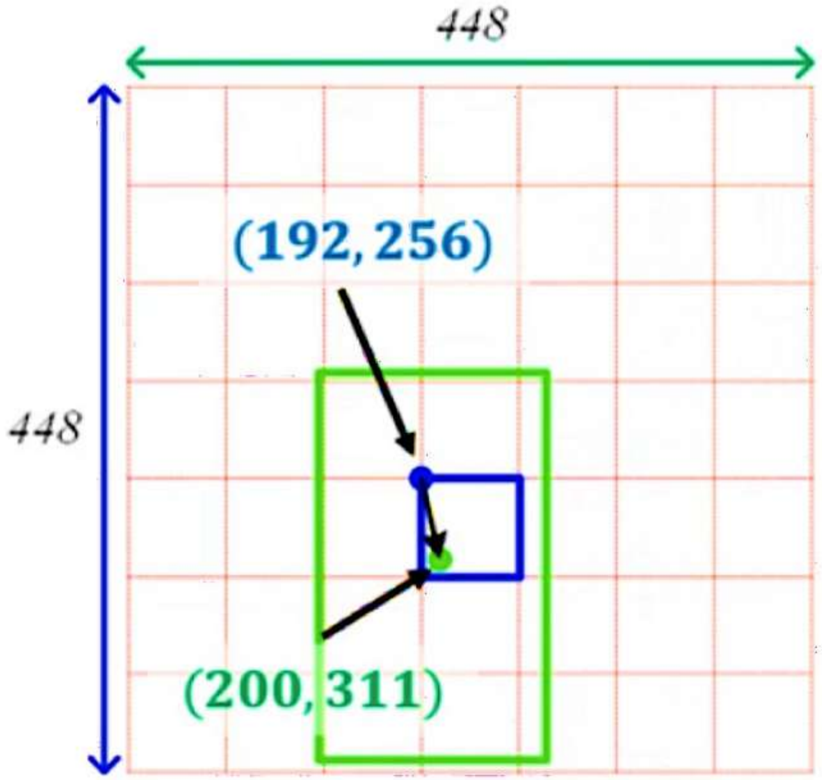
$$\Delta w = w/448$$
$$\Delta h = h/448$$

$(200, 311, 142, 250)$



Bounding Boxes





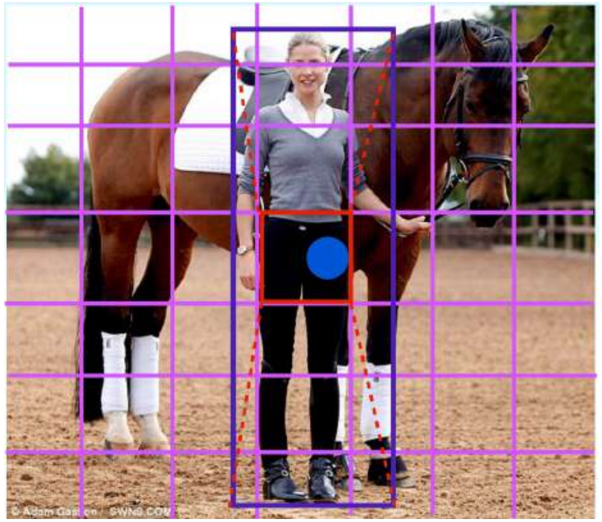
$$\Delta x = \frac{200 - 192}{64} \approx 0.13$$

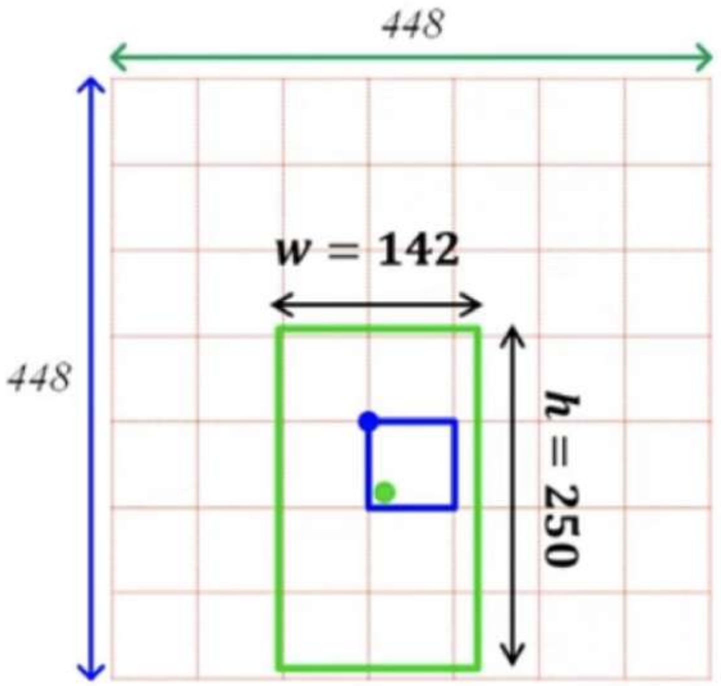
$$\Delta y = \frac{311 - 256}{64} \approx 0.87$$

$$\Delta w = w/448$$

$$\Delta h = h/448$$

Example Calculation: GT/Target Values





$$\Delta x = \frac{200 - 192}{64} \approx 0.13$$

$$\Delta y = \frac{311 - 256}{64} \approx 0.87$$

$$\Delta w = \frac{142}{448} \approx 0.31$$

$$\Delta h = \frac{250}{448} \approx 0.56$$

$$(200, 311, 142, 250) \Rightarrow (0.13,\ 0.87,\ 0.31,\ 0.56)$$

YOLO: Label Encoding

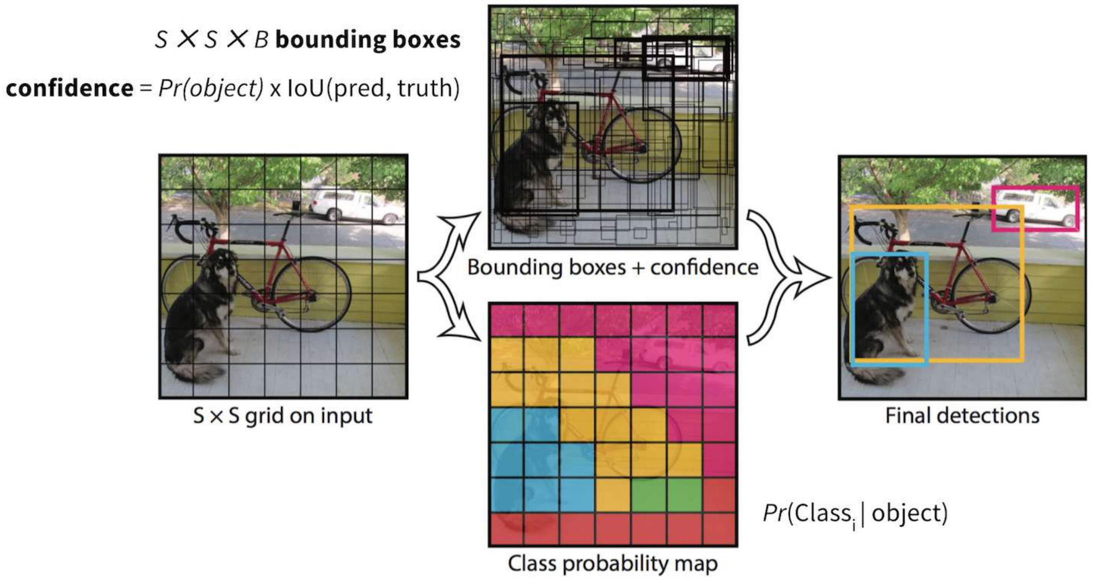
- For every Grid cell (Anchor box), we need to create targets/labels
- No Object - All zeros
- Object - Relative values w.r.t grid
- Classes - one-hot encoding
- Ex: (x,y,w,h,c) = (0.9,0.7,0.1,0.1,1.0)
    - Classes = (1.0, 0,0,…0) - 20 values

	$(\Delta\hat{x}, \Delta\hat{y}, \Delta\hat{w}, \Delta\hat{h}, \hat{c})$	$(\hat{p}_1, \hat{p}_2, \dots, \hat{p}_{20})$
A_1	(0 0 0 0 0)	(0 0 ... 0)
A_2	(0 0 0 0 0)	(0 0 ... 0)
	... ...	
A_{11}	(0.9 0.7 0.1 0.1 1.0)	(0 ... 1.0 ...)
	... ...	$\hat{p}_{14=\text{person}}$
A_{32}	(0.1 0.8 0.3 0.5 1.0)	(0 ... 1.0 ...)
	... ...	

AI VIET NAM
@aivietnam.edu.vn

YOLO: Motivations

- Residual blocks
- Bounding box regression
- Intersection Over Union (IOU)



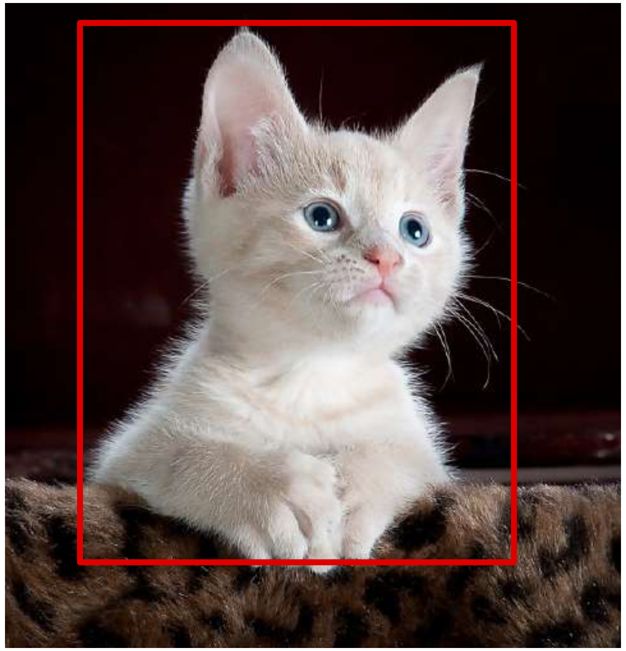
YOLO: Motivations

Image Classification

Is this a cat or a dog?

Neural Network Output:
Cat = 1
Dog = 0

Object Localization



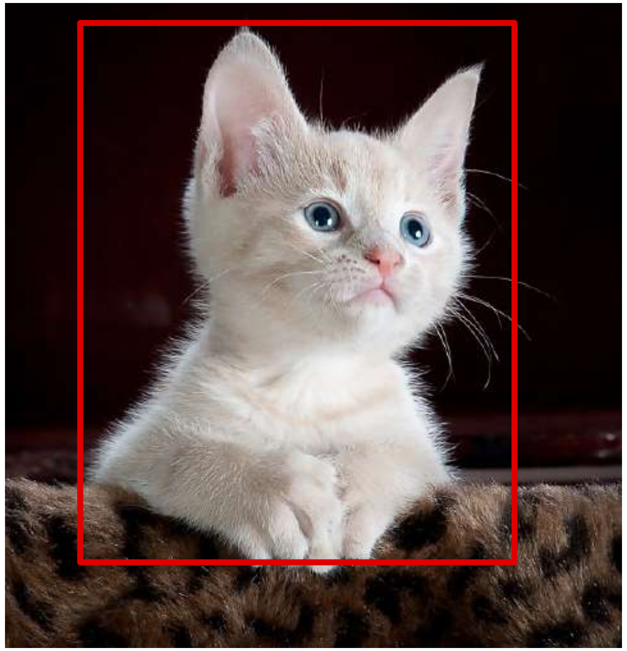
Where extactly is the cat in the image?

Neural Network Output:
Cat = 1
Dog = 0

Bounding box

YOLO: Motivations

Object Localization



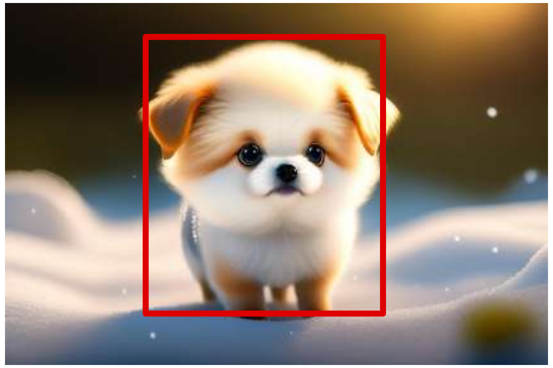
Where exactly is the cat in the image?

$$\begin{bmatrix} P_c \\ B_x \\ B_y \\ B_w \\ B_h \\ C_1 \\ C_2 \end{bmatrix} \quad \begin{bmatrix} 1 \\ 50 \\ 70 \\ 60 \\ 70 \\ 1 \\ 0 \end{bmatrix}$$

C_1: Cat class
C_2: Dog class

Neural Network Output:
Cat = 1
Dog = 0

Bounding box

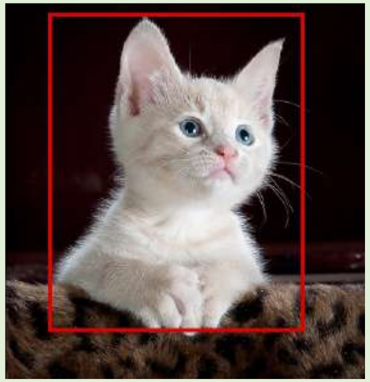


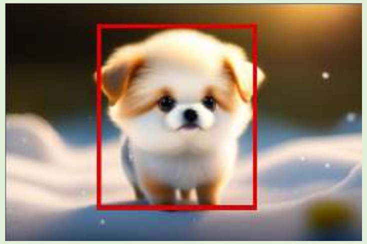
$$\begin{bmatrix} P_c \\ B_x \\ B_y \\ B_w \\ B_h \\ C_1 \\ C_2 \end{bmatrix} \quad \begin{bmatrix} 1 \\ 30 \\ 28 \\ 28 \\ 82 \\ 0 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} P_c \\ B_x \\ B_y \\ B_w \\ B_h \\ C_1 \\ C_2 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

YOLO: Motivations

X: Train



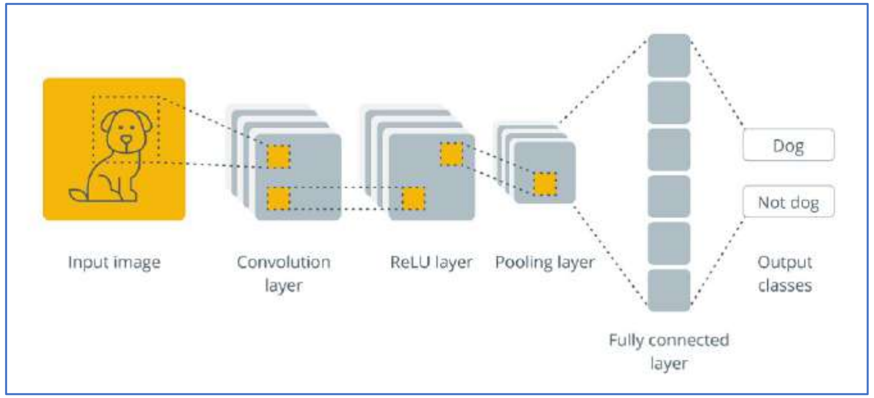


Y: Train

$$\begin{bmatrix} P_c \\ B_x \\ B_y \\ B_w \\ B_h \\ C_1 \\ C_2 \end{bmatrix} \begin{bmatrix} 1 \\ 50 \\ 70 \\ 60 \\ 70 \\ 1 \\ 0 \end{bmatrix}$$

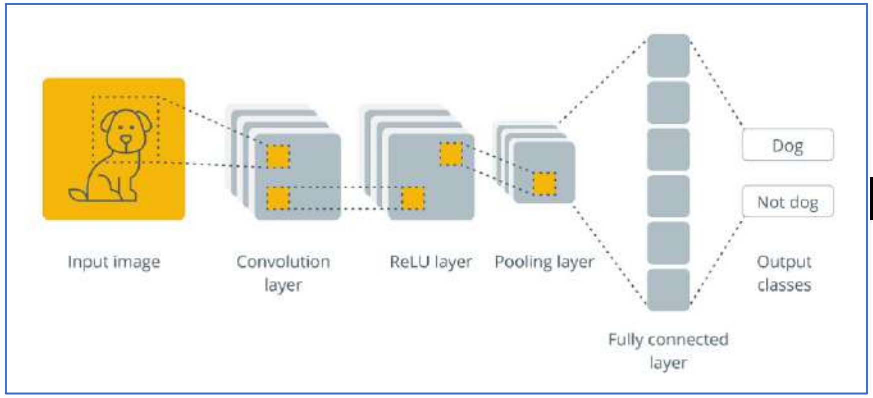
$$\begin{bmatrix} P_c \\ B_x \\ B_y \\ B_w \\ B_h \\ C_1 \\ C_2 \end{bmatrix} \begin{bmatrix} 1 \\ 30 \\ 28 \\ 28 \\ 82 \\ 0 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} P_c \\ B_x \\ B_y \\ B_w \\ B_h \\ C_1 \\ C_2 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$



$$\begin{bmatrix} P_c \\ B_x \\ B_y \\ B_w \\ B_h \\ C_1 \\ C_2 \end{bmatrix}$$

YOLO: Motivations

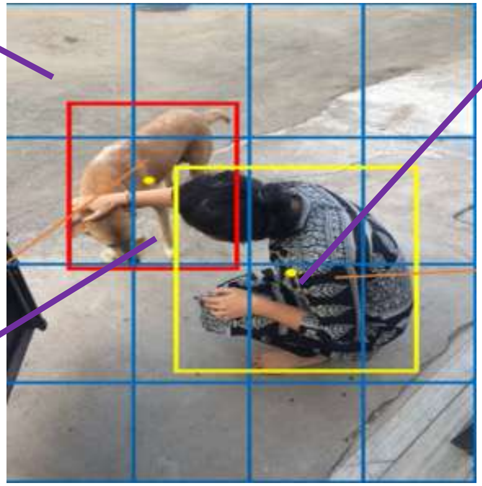


$$\begin{bmatrix} 1 \\ 50 \\ 70 \\ 60 \\ 70 \\ 1 \\ 0 \end{bmatrix}$$

This works OK only for single object. What about multiple objects in an image

YOLO: Motivations

$$\begin{bmatrix} P_c \\ B_x \\ B_y \\ B_w \\ B_h \\ C_1 \\ C_2 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$



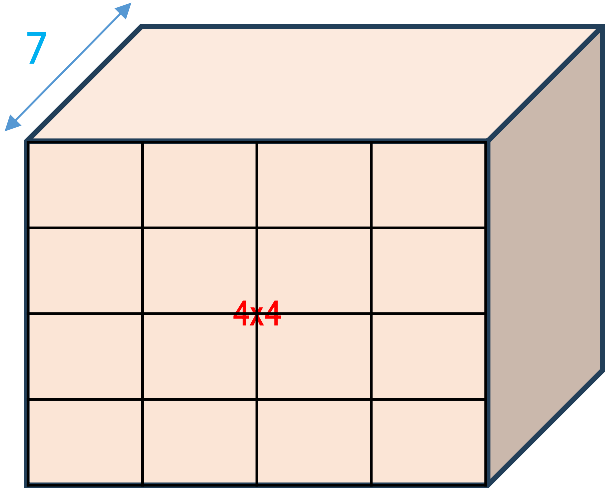
$$\begin{bmatrix} 1 \\ 0.32 \\ 0.02 \\ 3.0 \\ 2.0 \\ 0 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 \\ 0.05 \\ 0.3 \\ 2.0 \\ 1.3 \\ 1 \\ 0 \end{bmatrix}$$

(0,0)

(1,1)

C_1: Dog class
C_2: Human class

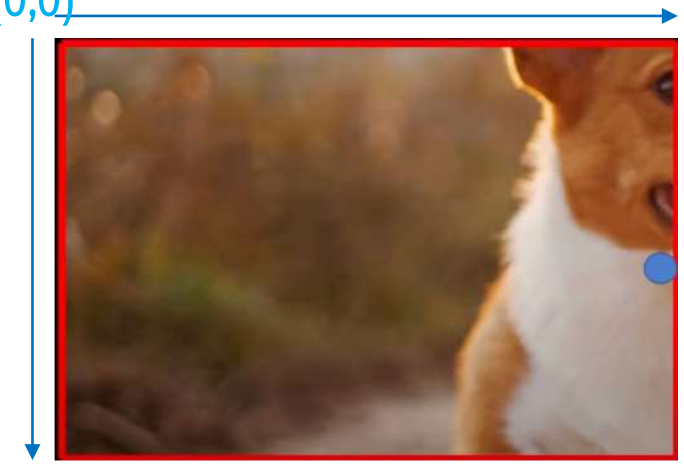


AI VIET NAM
@aivietnam.edu.vn

YOLO: Motivations

How the labels look like?

(0,0)



(1,1)

Each output and label will be relative to the cell!
Each bounding box for each cell will have:
[x,y,w,h] = [0.95, 0.55, 0.5, 1.0]

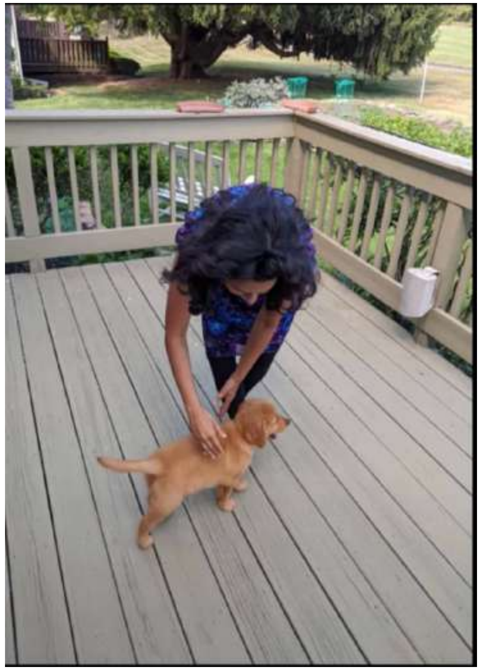
$$\text{Label}_{\text{cell}} = [\underbrace{c_1, c_2, \ldots, c_{20}}_{\text{20 classes}}, p_c, \overbrace{x, y, w, h}^{\text{Bounding Box}}]$$

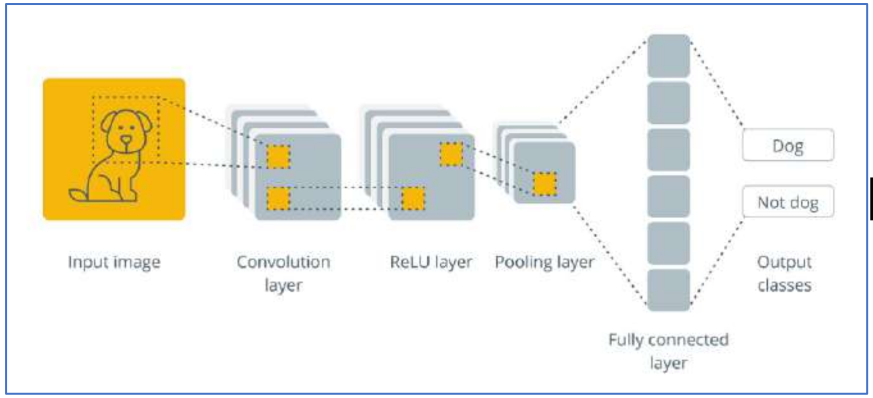
Probability that there is an object (1 or 0)

AI VIET NAM
@aivietnam.edu.vn

YOLO: Motivations

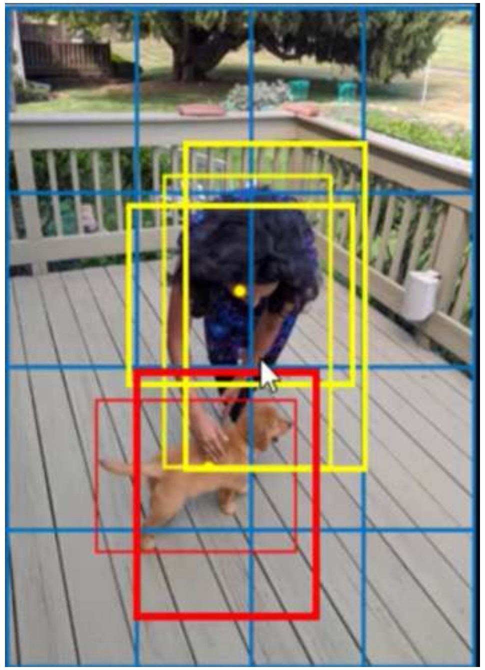
How the prediction look like?





$$\begin{bmatrix} P_c \\ B_x \\ B_y \\ B_w \\ B_h \\ C_1 \\ C_2 \end{bmatrix}$$

16 vectors



Multiple bounding boxes

$$\text{Prediction}_{\text{cell}} = [\underbrace{c_1, c_2, \ldots, c_{20}}_{\text{20 classes}}, p_{c1}, \overbrace{x, y, w, h}^{\text{Bounding Box}}]$$

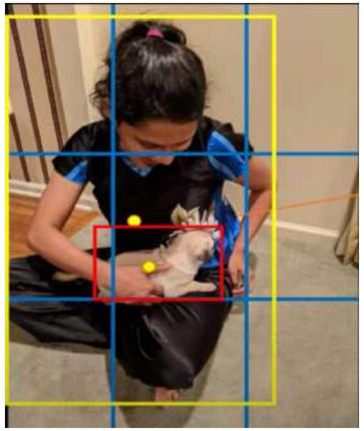
Probability that there is an object (1 or 0)

YOLO: Motivations

Non Maximum Suppression

Algorithm 1 Non-Max Suppression

procedure NMS(B,c)
  $B_{nms} \leftarrow \emptyset$ Initialize empty set
  for $b_i \in B$ **do** => Iterate over all the boxes
    $discard \leftarrow$ False Take boolean variable and set it as false. This variable indicates whether b(i) should be kept or discarded
    for $b_j \in B$ **do** Start another loop to compare with b(i)
      if same$(b_i, b_j) > \lambda_{\mathbf{nms}}$ **then** If both boxes having same IOU
        if score$(c, b_j) >$ score(c, b_i) **then**
          $discard \leftarrow$ 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.
      $B_{nms} \leftarrow B_{nms} \cup b_i$
  return B_{nms} Do the same procedure for remaining boxes and return the final list

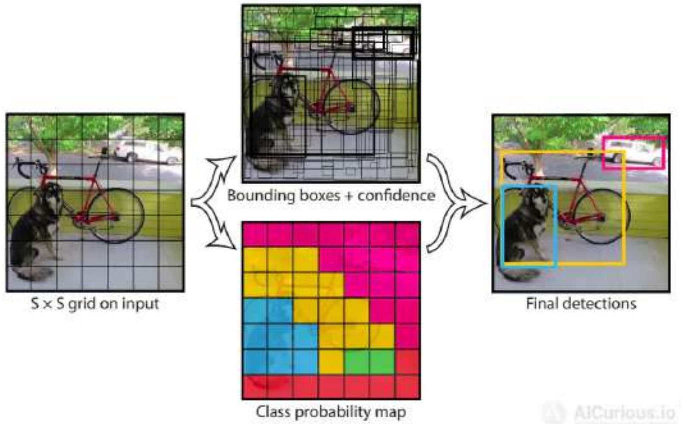


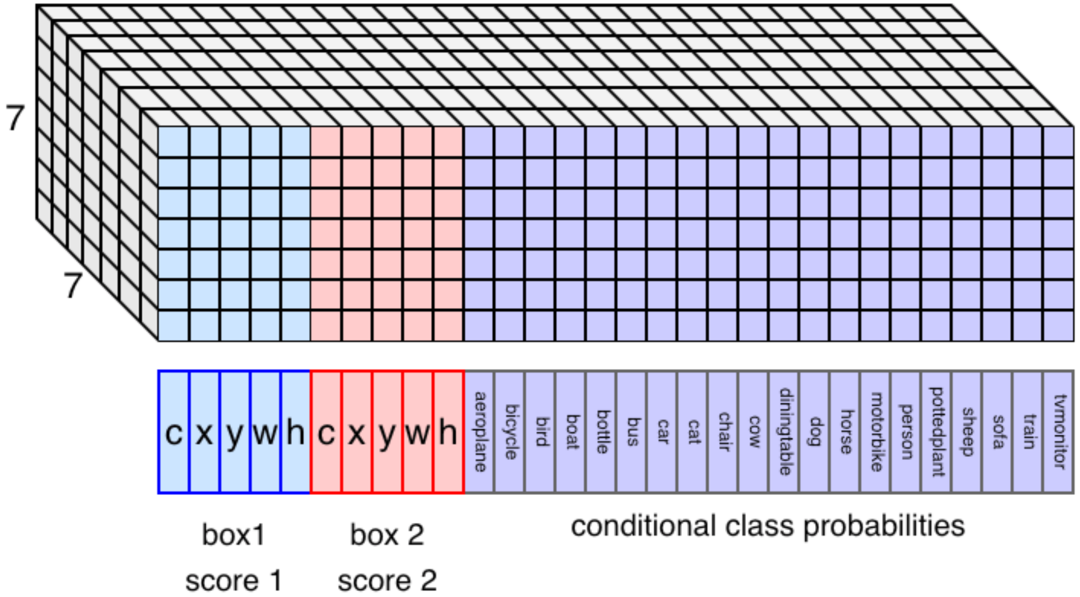
What if one grid cell has center of two objects

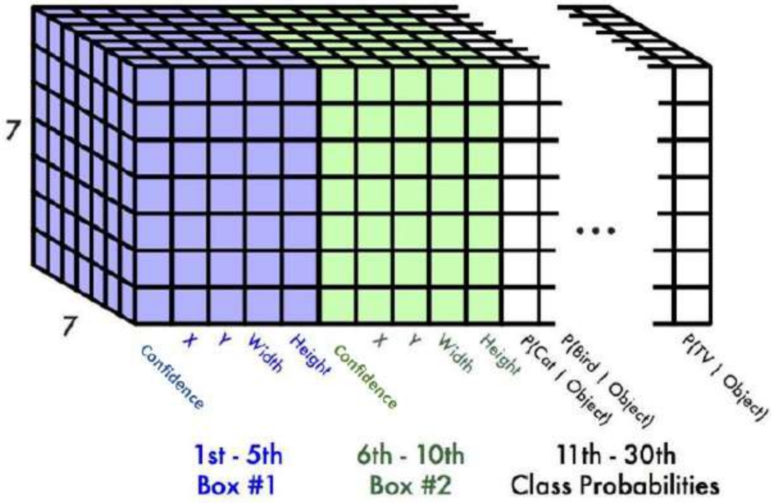
AI VIET NAM
@aivietnam.edu.vn

YOLO-v1 Architecture

- The input image is divided into an S×S grid (S=7)
- Each grid cell predicts B bounding boxes (B=2) and confidence scores for those boxes
- Each bounding box consists of 5 predictions: x, y, w, h, and confidence
- Each grid cell also predicts conditional class probabilities, P(Class i | Object). (Total number of classes=20)



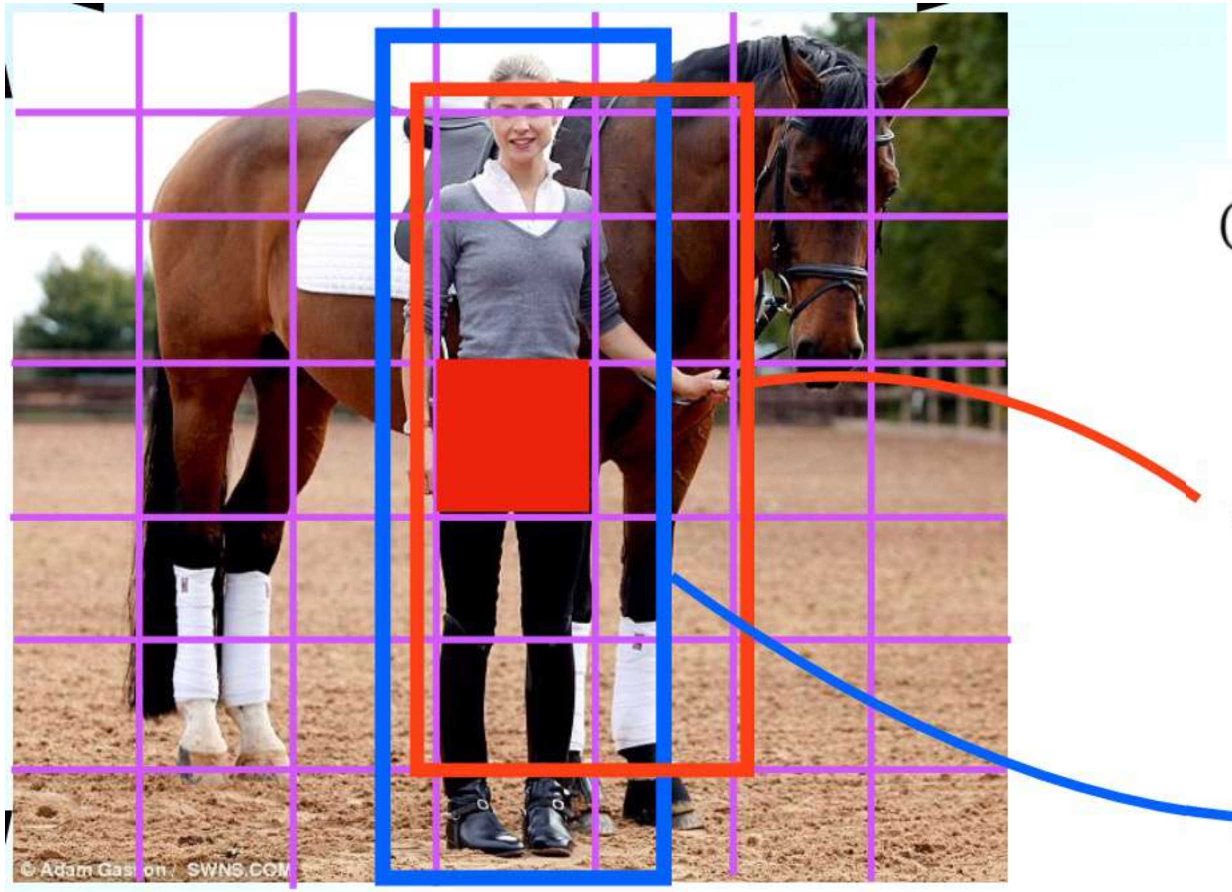




The output size : 7×7×(2×5+20)=1470

AI VIET NAM
@aivietnam.edu.vn

Output Parsing



Consider the box with highest confidence score per each grid

$(\Delta x_1, \Delta y_1, \Delta w_1, \Delta h_1)$,c1 $(\Delta x_2, \Delta y_2, \Delta w_2, \Delta h_2)$,c2

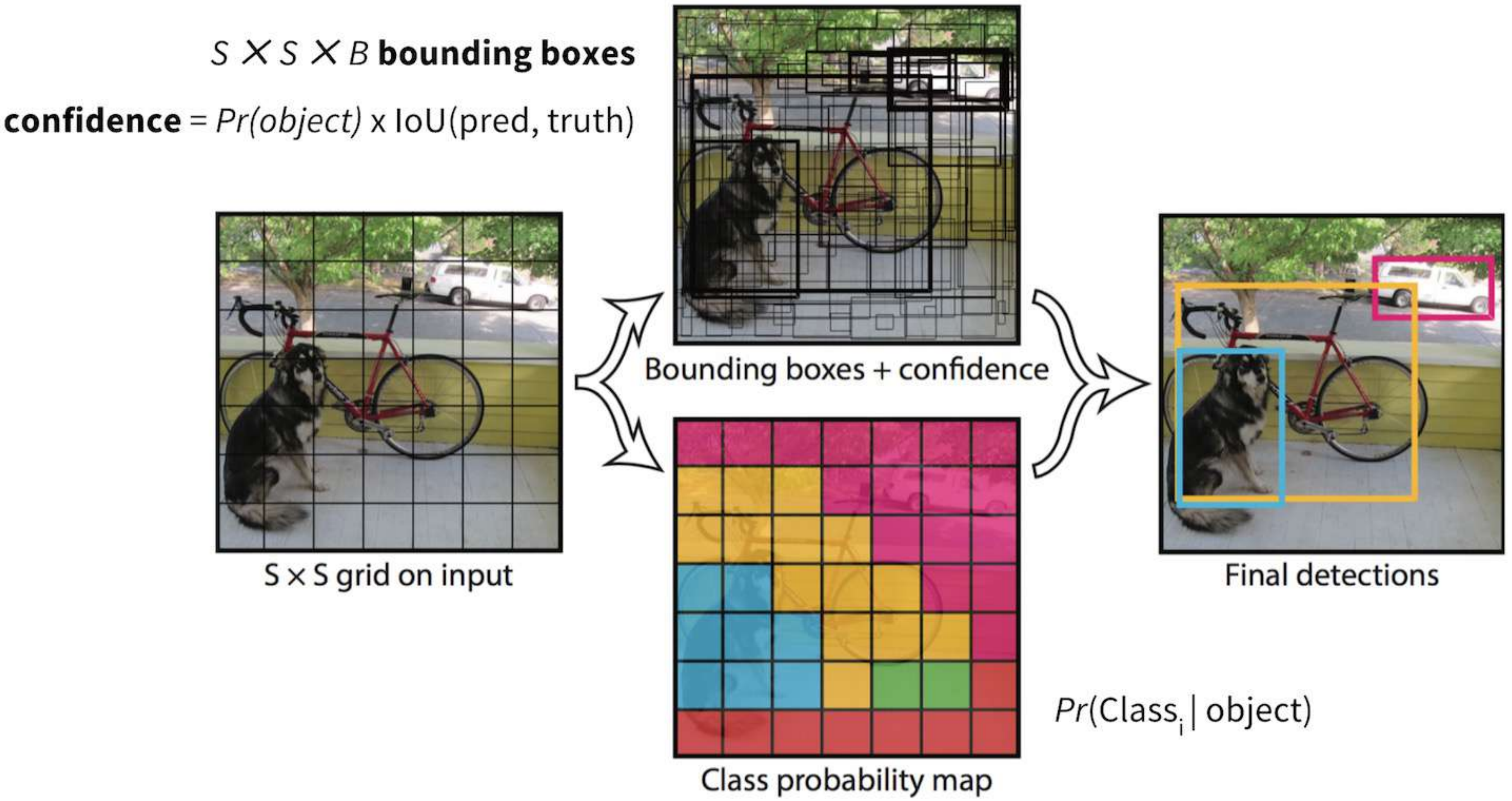
$$\begin{cases} x_1 = \Delta x_1 \times 64 + x_a \\ y_1 = \Delta y_1 \times 64 + y_a \\ w_1 = \Delta w_1 \times 448 \\ h_1 = \Delta h_1 \times 448 \end{cases} \qquad \begin{cases} x_2 = \Delta x_2 \times 64 + x_a \\ y_2 = \Delta y_2 \times 64 + y_a \\ w_2 = \Delta w_2 \times 448 \\ h_2 = \Delta h_2 \times 448 \end{cases}$$

object class $l = \text{argmax}(p_1, p_2, \ldots, p_{20})$

class confidence $\hat{c}_1 = c_1 \times p$ $\quad \hat{c}_2 = c_2 \times p$

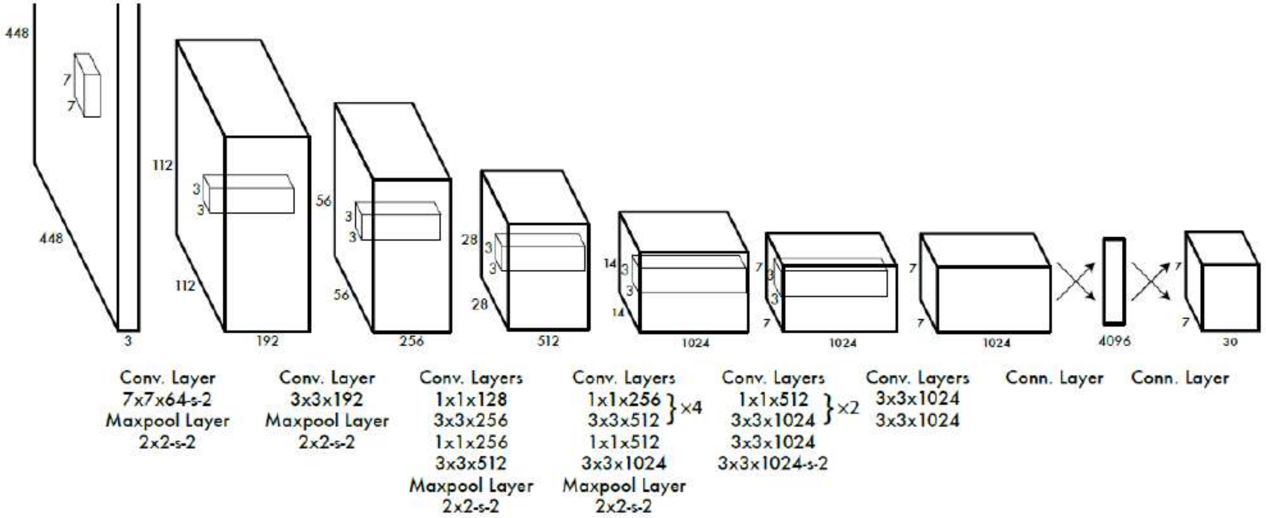
$p = \max(p_1, p_2, \ldots, p_{20})$

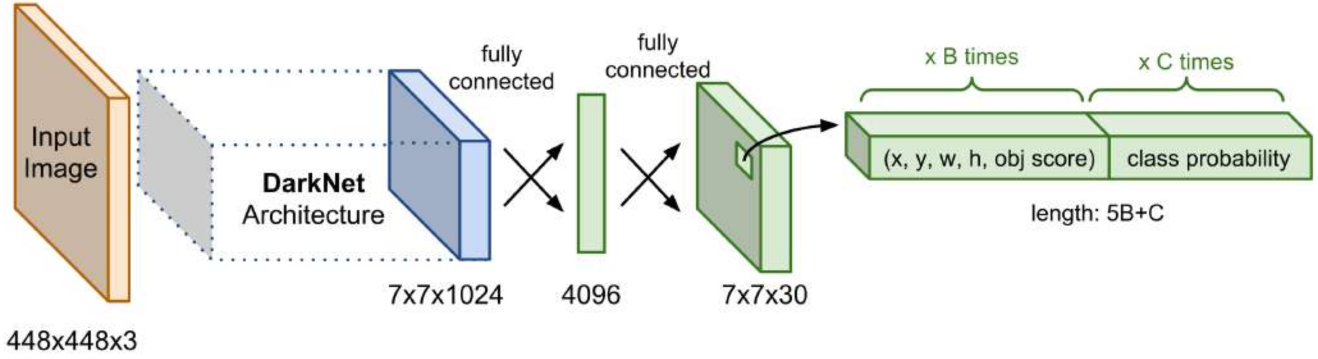
YOLOv1: Summary



AI VIET NAM
@aivietnam.edu.vn

YOLO-v1 Architecture





The YOLO pipeline is simple.

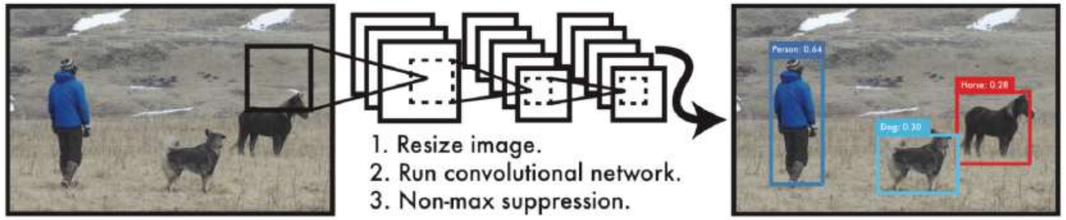


Figure 1 of the paper

1. It resizes input images to 448 x 448.
2. It runs a single convolutional network on the input images.
3. It thresholds the resulting detections by the model's confidence.

The YOLO is a network was "*inspired by*" GoogleNet. It has 24 convolutional layers working for feature extractors and 2 dense layers for doing the predictions. This architecture works upon is called Darknet. There is a fast version of YOLO called "Tiny-YOLO" which only has 9 convolution layers

YOLO-v1 Architecture

	Type	Size	Filters	Stride	Output
	Conv.	7 x 7 x 3	64	2	224 x 224 x 64
	max pool	2 x 2			112 x 112 x 64
	Conv.	3 x 3 x 64	192	1	112 x 112 x 192
	max pool	2 x 2			56 x 56 x 192
	Conv.	1 x 1 x 192	128	1	56 x 56 x 128
	Conv.	3 x 3 x 128	256	1	56 x 56 x 256
	Conv.	1 x 1 x 256	256	1	56 x 56 x 256
	Conv.	3 x 3 x 256	512	1	56 x 56 x 512
	max pool	2 x 2			28 x 28 x 512
4 x	Conv.	1 x 1 x 512	256	1	28 x 28 x 256
	Conv.	3 x 3 x 256	512	1	28 x 28 x 512
	Conv.	1 x 1 x 512	512	1	28 x 28 x 512
	Conv.	3 x 3 x 512	1024	1	28 x 28 x1024
	max pool	2 x 2			14 x 14 x1024
2 x	Conv.	1 x 1 x 1024	512	1	14 x 14 x 512
	Conv.	3 x 3 x 512	1024	1	14 x 14 x1024
	Conv.	3 x 3 x 1024	1024	1	14 x 14 x1024
	Conv.	3 x 3 x 1024	1024	2	7 x 7 x1024
	Conv.	3 x 3 x 1024	1024	1	7 x 7 x1024
	Conv.	3 x 3 x 1024	1024	1	7 x 7 x1024

6

$4 \times 2 = 8$

2

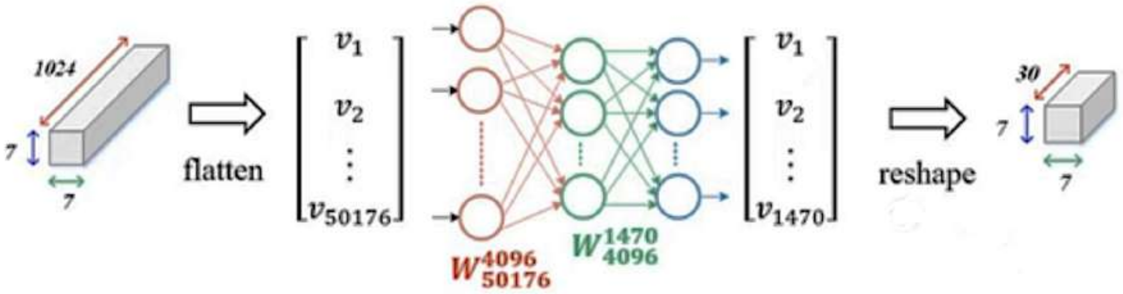
$2 \times 2 = 4$

4

$6 + 8 + 2 + 4 + 4 = 24$

Flatten the last conv map 7x7x1024 to 50176 feature vector

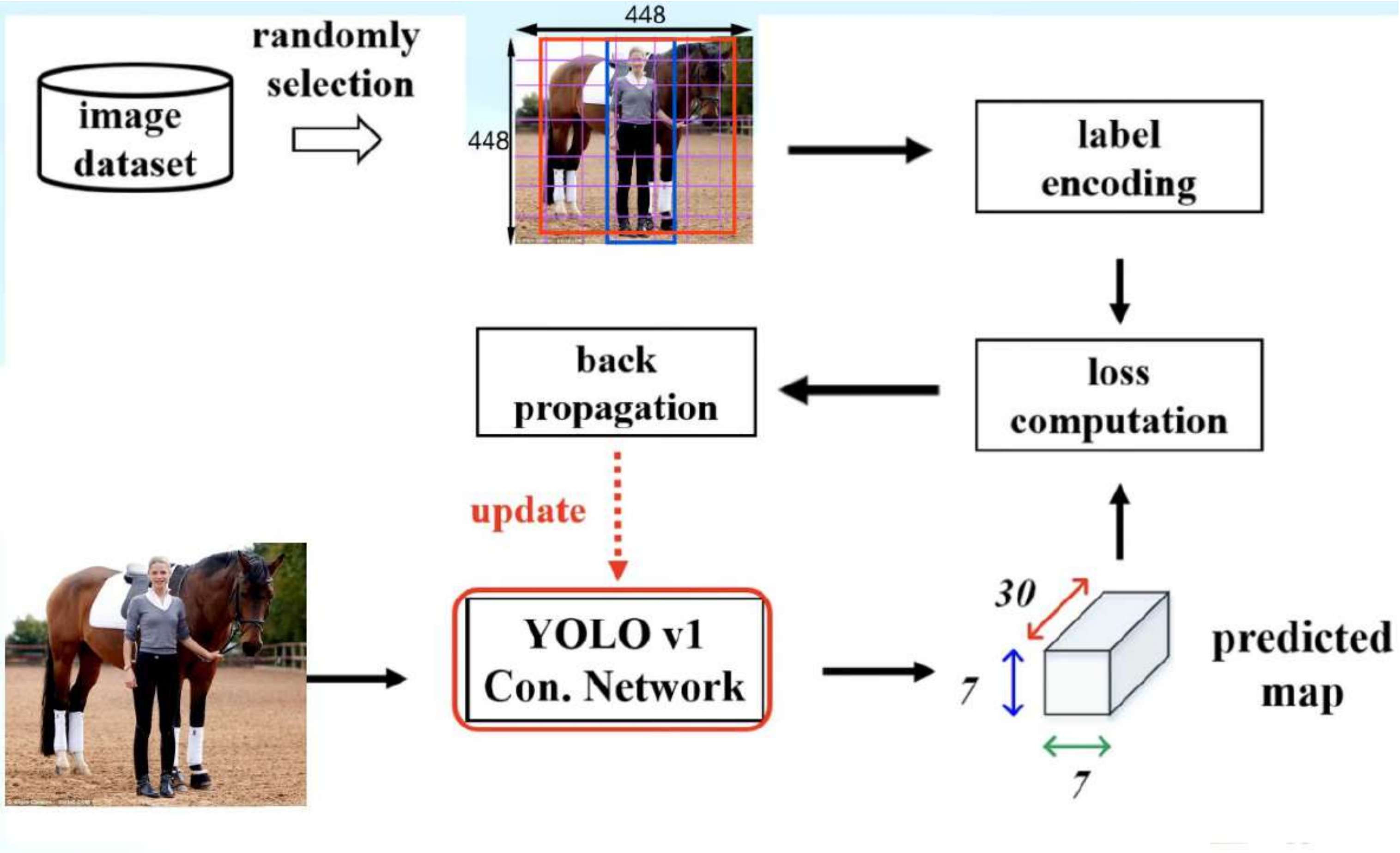
- Pass through 2 fully connected layers
- Output - 1470 feature vector
- Reshape 1470 vector to 7x7x30 feature map



AI VIET NAM
@aivietnam.edu.vn

Training Process

Dataset: Pascal VOC - 20 classes
• Network pretrained on Imagenet at 224x224
• Actual training on 448x448 on VOC dataset

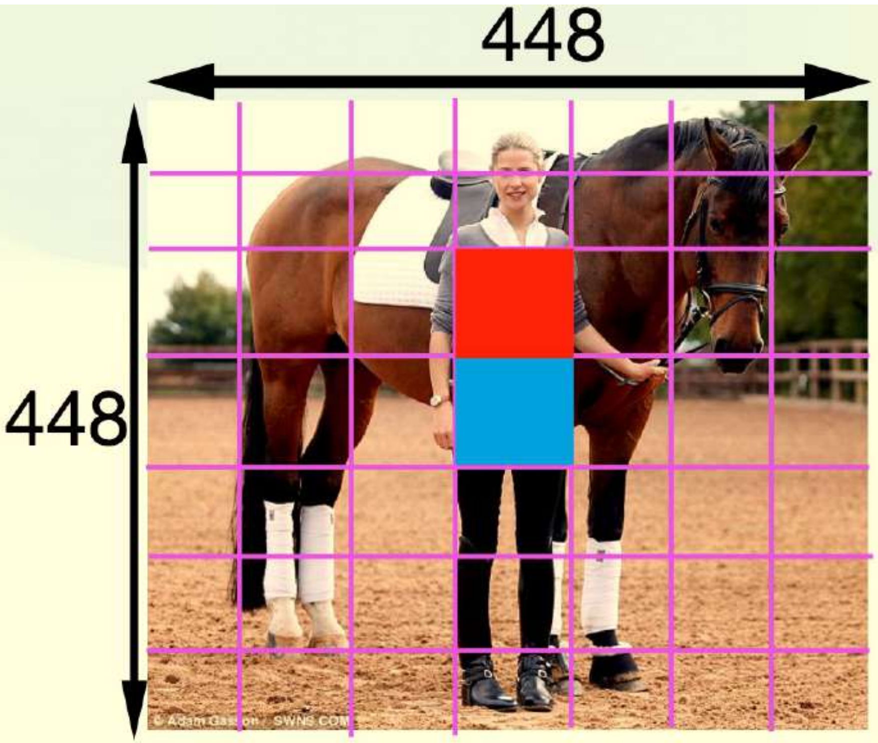


AI VIET NAM
@aivietnam.edu.vn

Loss Function

- Loss L is the sum of losses over all grid cells SxS.
- Put more importance on grid cells that contain objects
- Decrease the importance of grid cells having no objects
- Ex: 2 object cells, 47 no-object cells

$$\boldsymbol{L} = \sum_{i=1}^{S^2} L_i$$



$$\boldsymbol{L} = \sum_{i=1}^{S^2} \boxed{1_i^{obj}} \times L_{i,obj} + \underbrace{\lambda_{no\_obj}}_{0.5} \sum_{i=1}^{S^2} \boxed{1_i^{no\_obj}} \times L_{i,no\_obj}$$

$1_i^{obj} = 1$ if i^{th} grid is **object anchor**

$1_i^{no\_obj} = 1$ if i^{th} grid is **no-object anchor**

AI VIET NAM
@aivietnam.edu.vn

Loss for object cells

Loss = Confidence loss + classification loss + Box Regression loss
• Put more weightage on box parameters

$$\boldsymbol{L_{i,obj}} = \underbrace{\boxed{\lambda_{coord}}}_{=5} \times \underline{L^{box}_{i,obj}} + L^{conf}_{i,obj} + L^{cls}_{i,obj}$$

$$L^{box}_{i,obj} = \left(\Delta x_i^* - \Delta\hat{x}_i\right)^2 + \left(\Delta y_i^* - \Delta\hat{y}_i\right)^2 + \left(\sqrt{\Delta w_i^*} - \sqrt{\Delta\widehat{w}_i}\right)^2 + \left(\sqrt{\Delta h_i^*} - \sqrt{\Delta h_i}\right)^2$$

- $(\Delta\hat{x}_i, \Delta\hat{y}_i, \Delta\widehat{w}_i, \hat{h}_i)$: ground-truth box
- $(\Delta x_i^*, \Delta y_i^*, \Delta w_i^*, \Delta h_i^*)$: **responsible** predicted box that has the largest IoU with ground-truth box

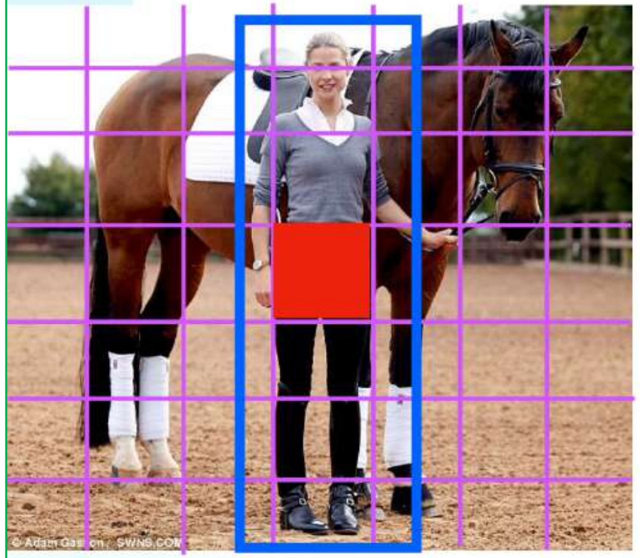
AI VIET NAM
@aivietnam.edu.vn

Loss for object cells

Loss = Confidence loss + classification loss + Box Regression loss
• Put more weightage on box parameters

$$L_{i,obj} = \underbrace{\boxed{\lambda_{coord}}}_{=5} \times L_{i,obj}^{box} + \underline{L_{i,obj}^{conf}} + L_{i,obj}^{cls}$$

$$L_{i,obj}^{conf} = (c_i^* - \hat{c}_i)^2 = (0.9 - 1.0)^2$$



Loss for object cells

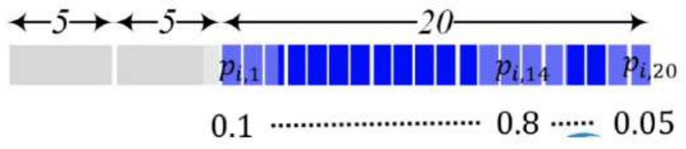
Loss = Confidence loss + classification loss + Box Regression loss
- Put more weightage on box parameters

$$\boldsymbol{L_{i,obj}} = \underbrace{\boxed{\lambda_{coord}}}_{=5} \times L_{i,obj}^{box} + L_{i,obj}^{conf} + \underline{L_{i,obj}^{cls}}$$

$$L_{i,obj}^{cls} = \sum_{c=1}^{20} \left(p_{i,c} - \hat{p}_{i,c}\right)^2$$

$$L_{i,obj}^{cls} = \left(p_{i,1} - \hat{p}_{i,1}\right)^2 + \cdots + \left(p_{i,14} - \hat{p}_{i,14}\right)^2 + \cdots + \left(p_{i,20} - \hat{p}_{i,20}\right)^2$$

$$L_{i,obj}^{cls} = (0.1 - 0.0)^2 + \cdots + (0.8 - \underset{}{\overset{\hat{p}_{i,14}}{1.0}})^2 + \cdots + (0.05 - 0.0)^2$$



$$\hat{\boldsymbol{p}}_{\boldsymbol{i,14=person}} = 1.0$$

YOLO-v1 Architecture

Loss Function

Regression loss

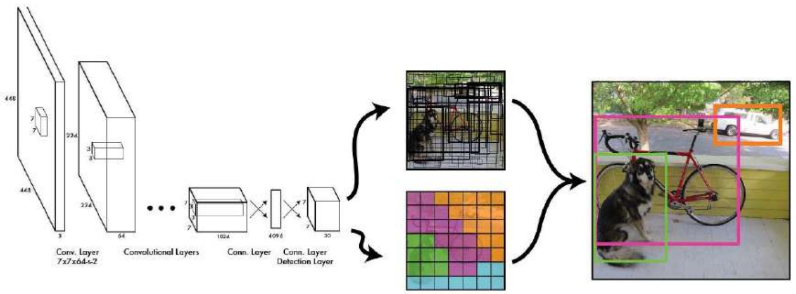
$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

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

Confidence loss

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

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



Classification loss

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

$$\mathcal{L} = \mathcal{L}_{\text{loc}} + \mathcal{L}_{\text{obj}} + \mathcal{L}_{\text{cls}}$$

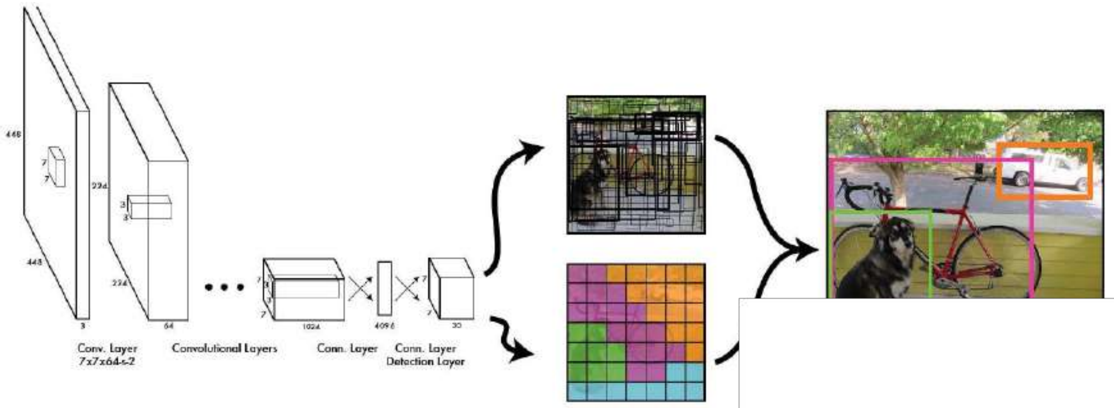
$\mathcal{L}_{\text{loc}}$ - loss function cho dự đoán vị trí bounding box so với ground truth.

$\mathcal{L}_{\text{obj}}$ - loss function cho dự đoán trong cell có object hay không.

$\mathcal{L}_{\text{cls}}$ - loss function cho dự đoán phân phối xác suất cho từng class.

AI VIET NAM
@aivietnam.edu.vn

YOLO-v1 Architecture



$$\text{box confidence score} = Pr(object) \cdot IOU_{pred}^{truth}$$

$$\text{conditional class probabilities} = Pr(class_i | object)$$

$$\text{class confidence score} = Pr(class_i) \cdot IOU_{pred}^{truth}$$

Loss Function

1 when there is object, 0 when there is no object (in the box j of the cell i)

Localization loss

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

Bounding box location (x, y) when there is object

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

Bounding box size (w, h) when there is object

Confidence loss

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

Confidence when there is object

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

Confidence when there is no object (in the box j of the cell i)

Classification loss

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

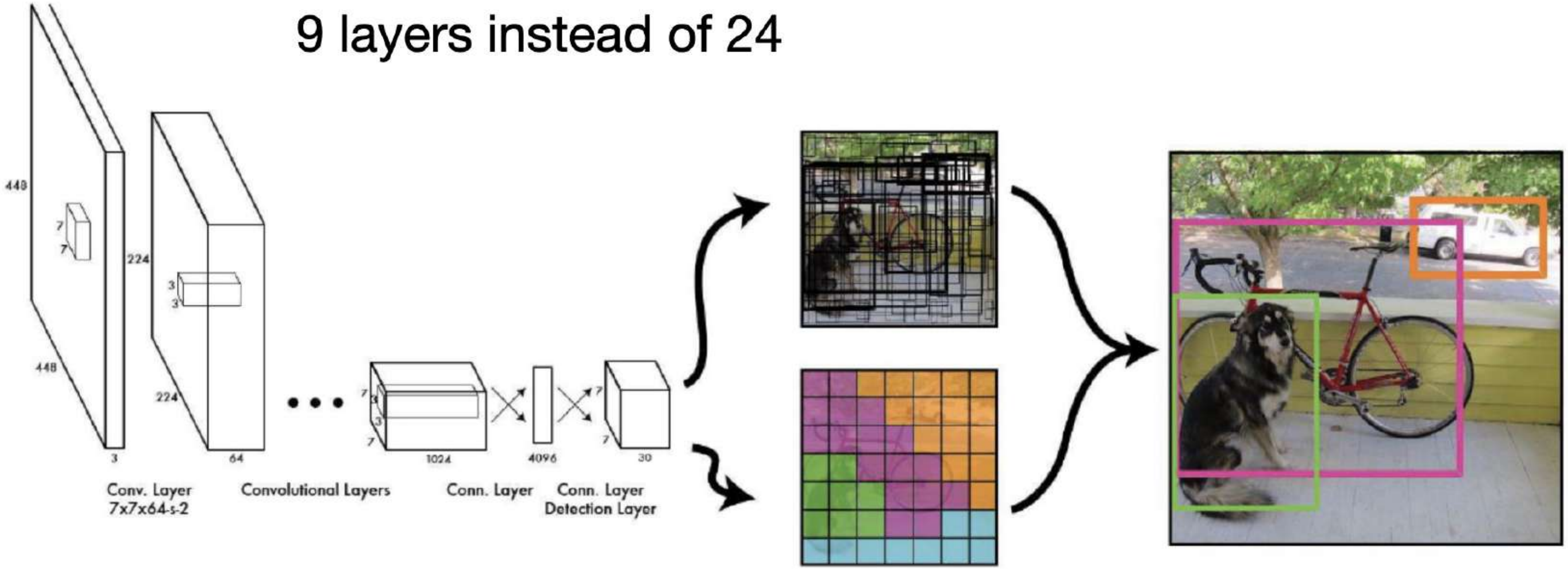
Class probabilities when there is object (in the cell i)

1 when there is no object, 0 when there is object

AI VIET NAM
@aivietnam.edu.vn

Fast YOLOv1

9 layers instead of 24



AI VIET NAM
@aivietnam.edu.vn

YOLO-v1 Limitations

- Strong Spatial Constraints: Each cell only predicts two bounding boxes and can only have two class

  YOLO struggles with small objects that appear in groups, like flocks of birds

- Relative Coarse Features:

  YOLO has many downsampling layers

  It struggles to generalize to objects in new or unsual aspect ratios or configurations

- Small-Object Localization

YOLO-v2 Motivations

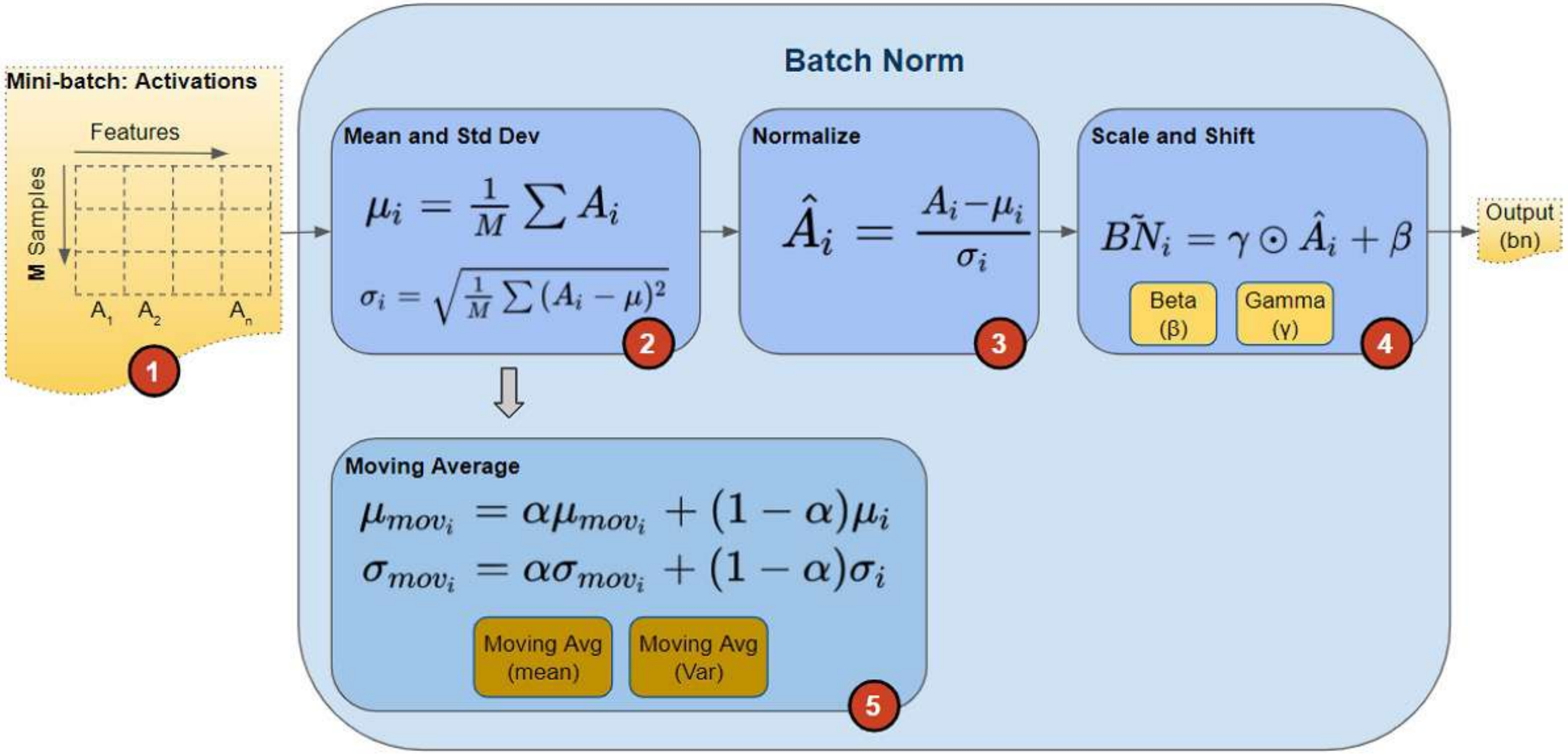
- YOLO v1 was faster than Faster R-CNN, but it was less accurate.
- YOLO v1's weakness was the bounding box accuracy. It didn't predict object locations and sizes well, particularly bad at spotting small objects.
- SSD, another single-stage detector, broke the record by being better (more accurate) than Faster R-CNN and even faster than YOLO v1.
- Authors wanted to make their object detector to recognize a wide variety of objects. Pascal VOC object detection dataset contains only 20 classes. They wanted their model to recognize much more classes of objects

AI VIET NAM
@aivietnam.edu.vn

Changes in YOLO-v2

Batch Normalization

In YOLO v2, they added Batch Normalization to all convolutional layers
- It improved mAP by 2%
- It helped method model to avoid overfitting

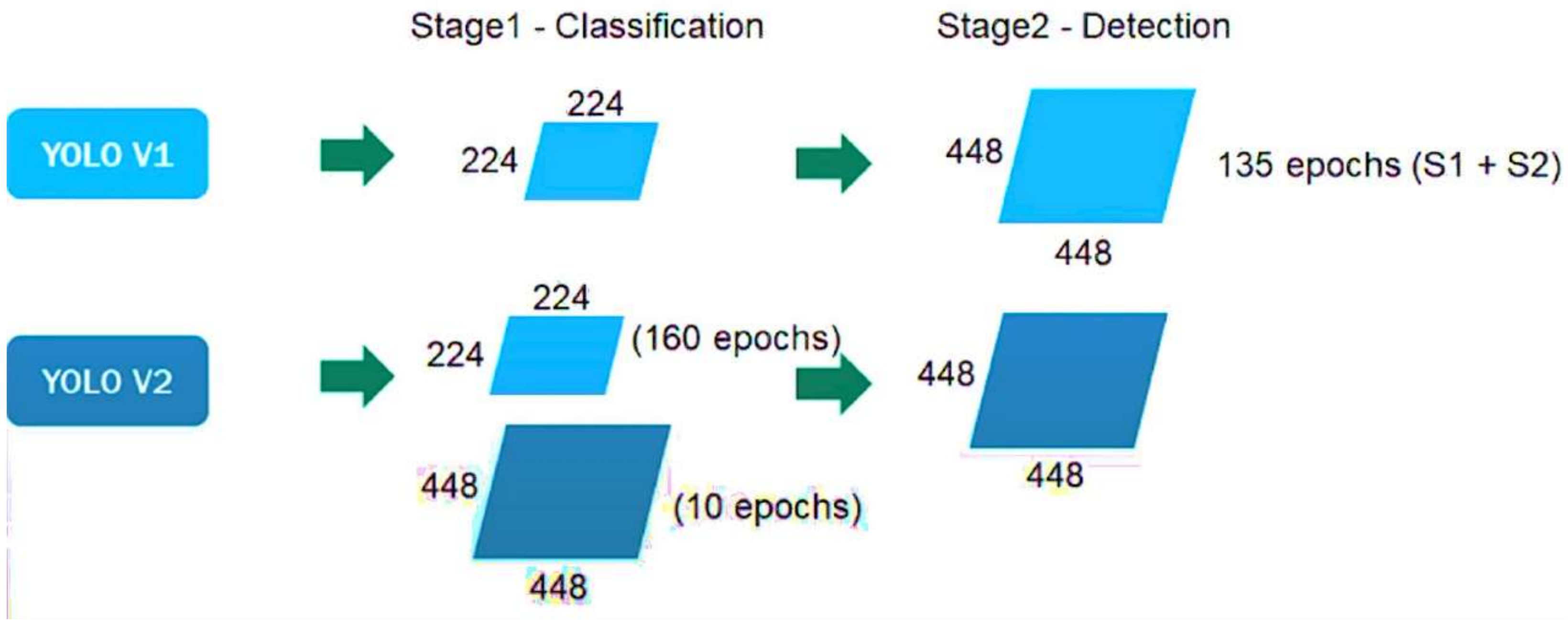


AI VIET NAM
@aivietnam.edu.vn

Changes in YOLO-v2

High-Resolution Classifier

- First, train the classifier on images of size 224 x 224
- Fine-tune the classifier on images of size 448 x 448 for 10 epochs on ImageNet
- Improve mAP by almost 4%

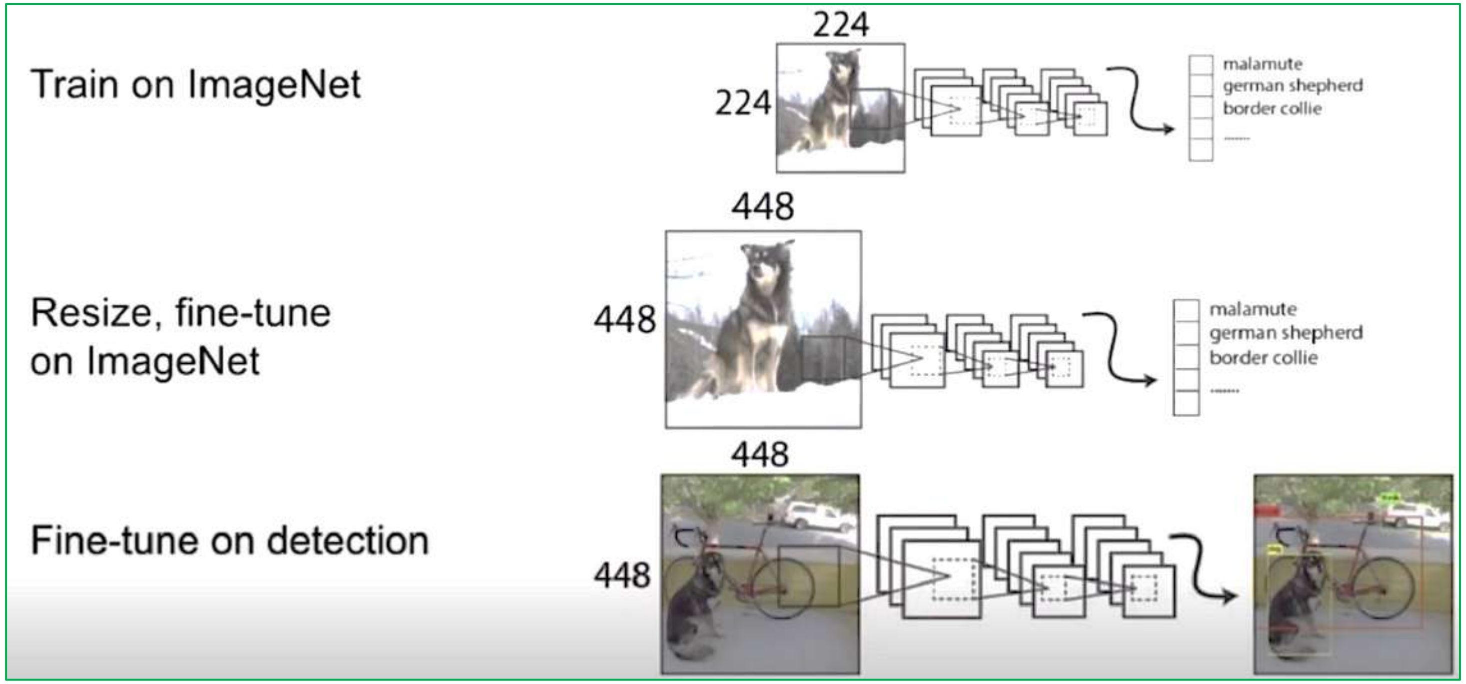


AI VIET NAM
@aivietnam.edu.vn

Changes in YOLO-v2

High-Resolution Classifier

- First, train the classifier on images of size 224 x 224
- Fine-tune the classifier on images of size 448 x 448 for 10 epochs on ImageNet
- Improve mAP by almost 4%



AI VIET NAM
@aivietnam.edu.vn

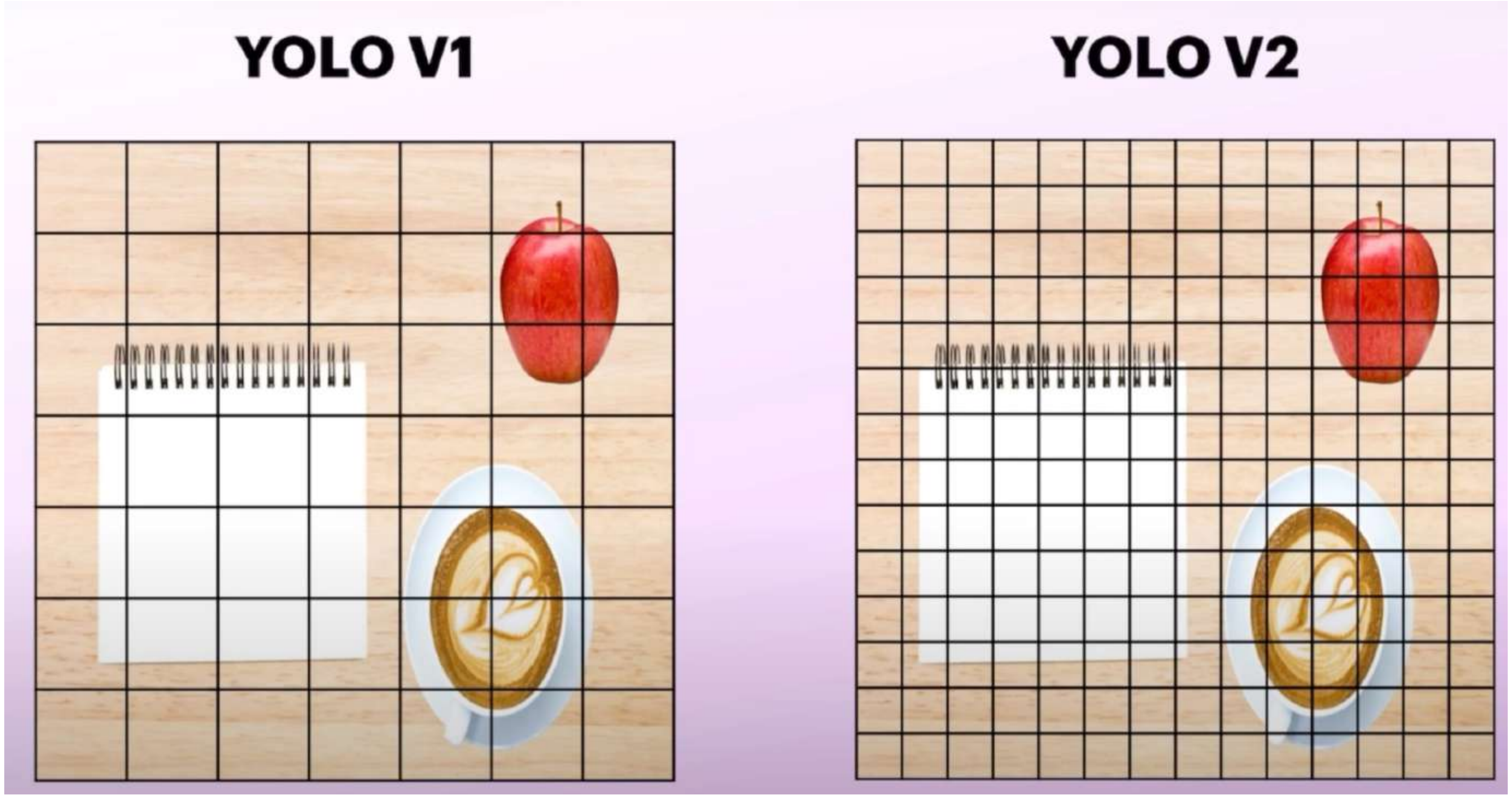
Changes in YOLO-v2

YOLOv1

Improve YOLOv1: high resolution feature map

Improve YOLOv1: high resolution feature map

Changes in YOLO-v2



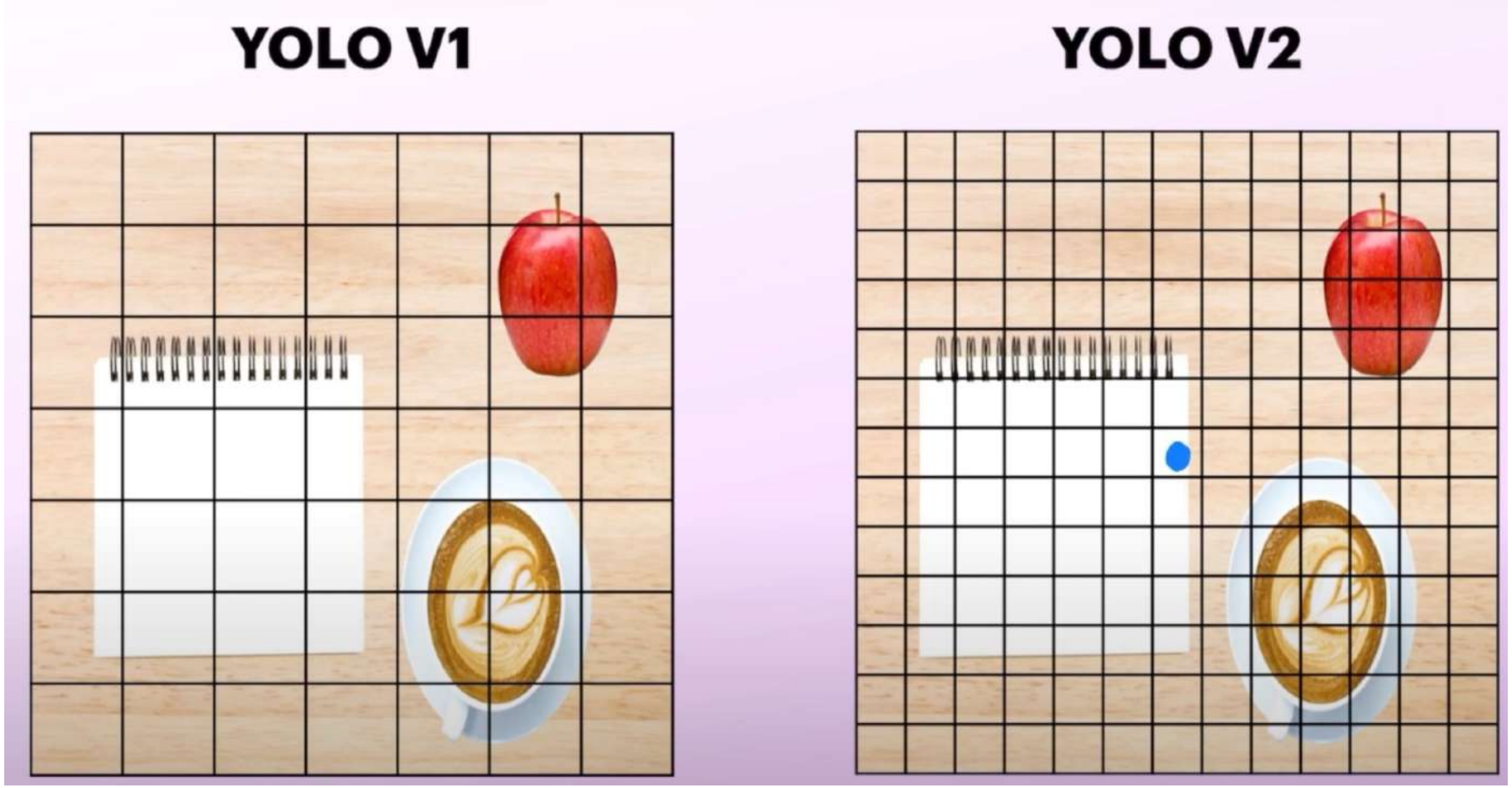
7x7

14x14

Problem: If you have even number of grid cells then there is no single center location

AI VIET NAM
@aivietnam.edu.vn

Changes in YOLO-v2



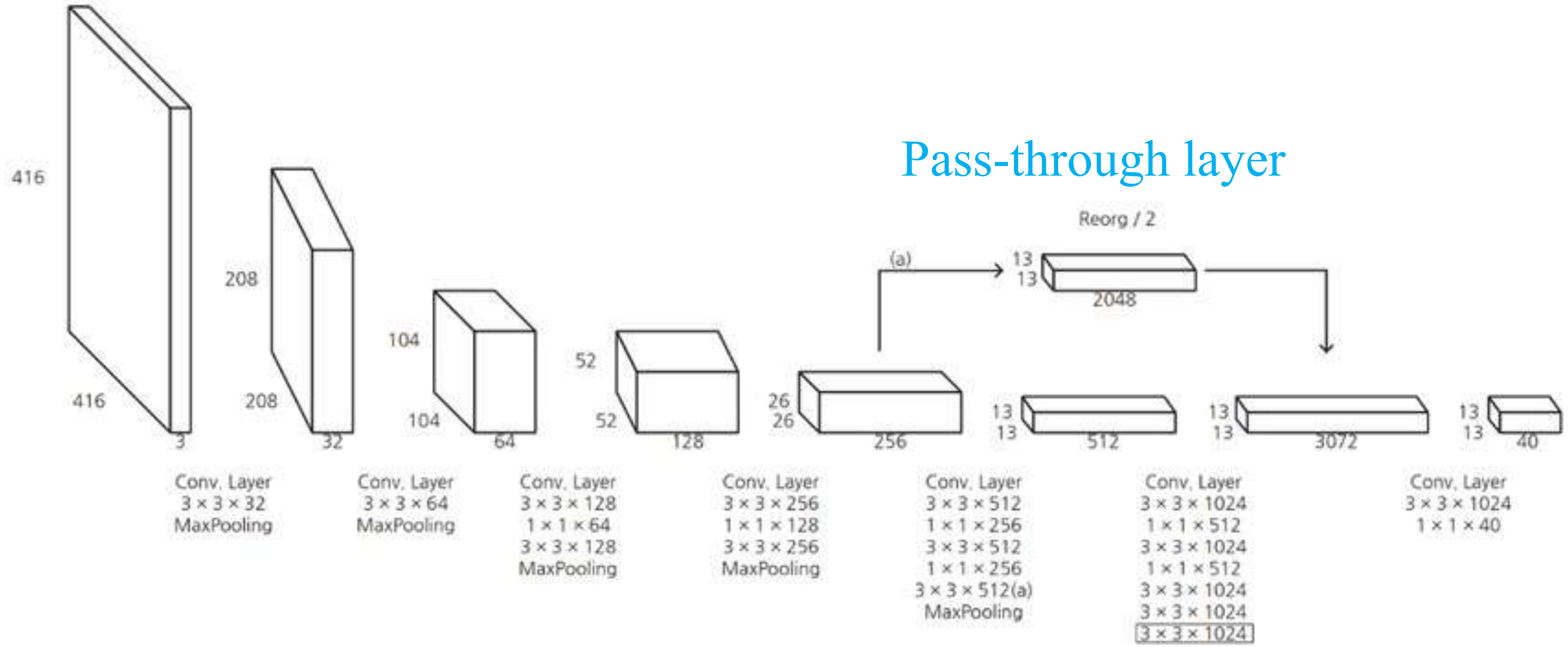
7x7

13x13

Problem: If you have even number of grid cells then there is no single center location

AI VIET NAM
@aivietnam.edu.vn

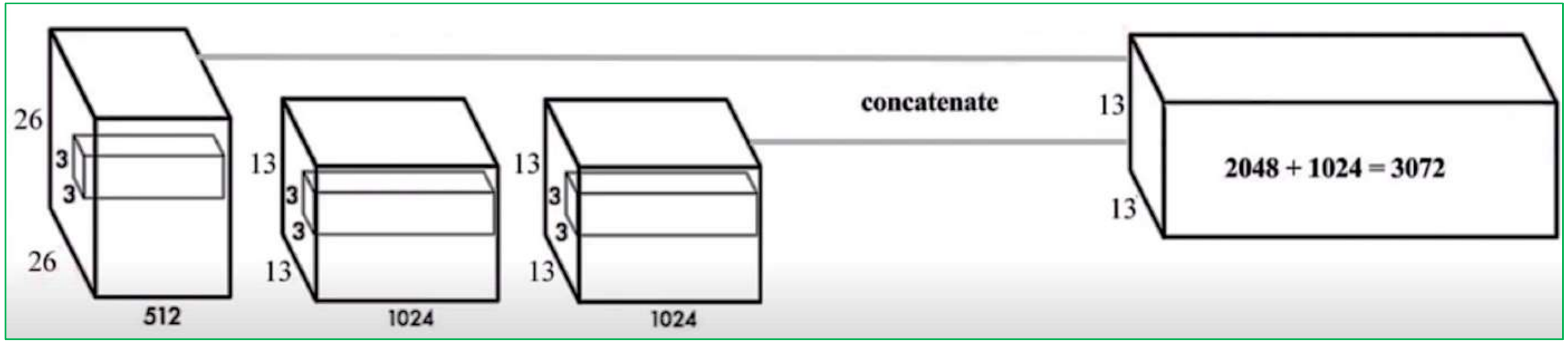
Changes in YOLO-v2

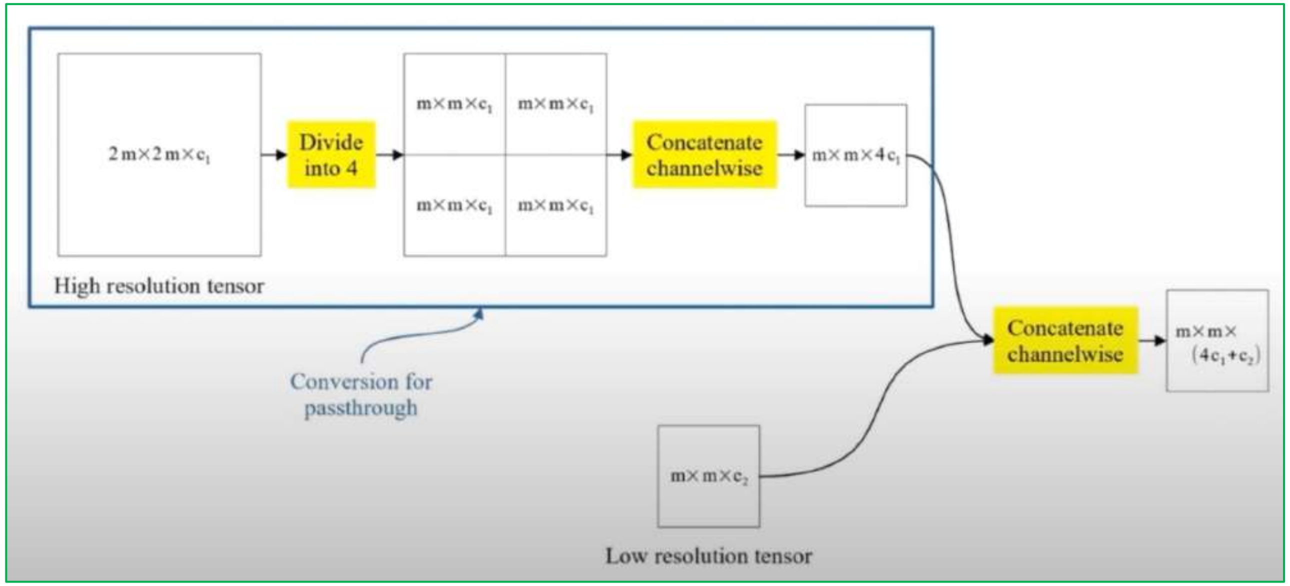


AI VIET NAM
@aivietnam.edu.vn

Changes in YOLO-v2

Pass-through layer





Changes in YOLO-v2

Convolutional with Anchor Boxes

YOLO v1 suffered from a low recall rate

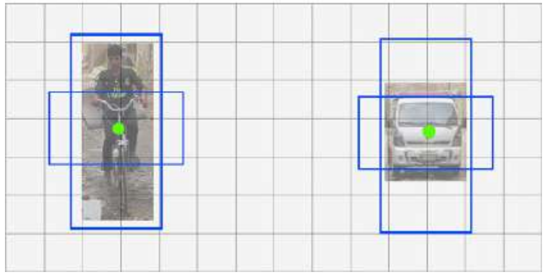
		Predicted	
		0	1
Actual	0	TN	FP
	1	FN	TP

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

YOLOv2 removes all fully connected layers and uses anchor boxes to predict bounding boxes

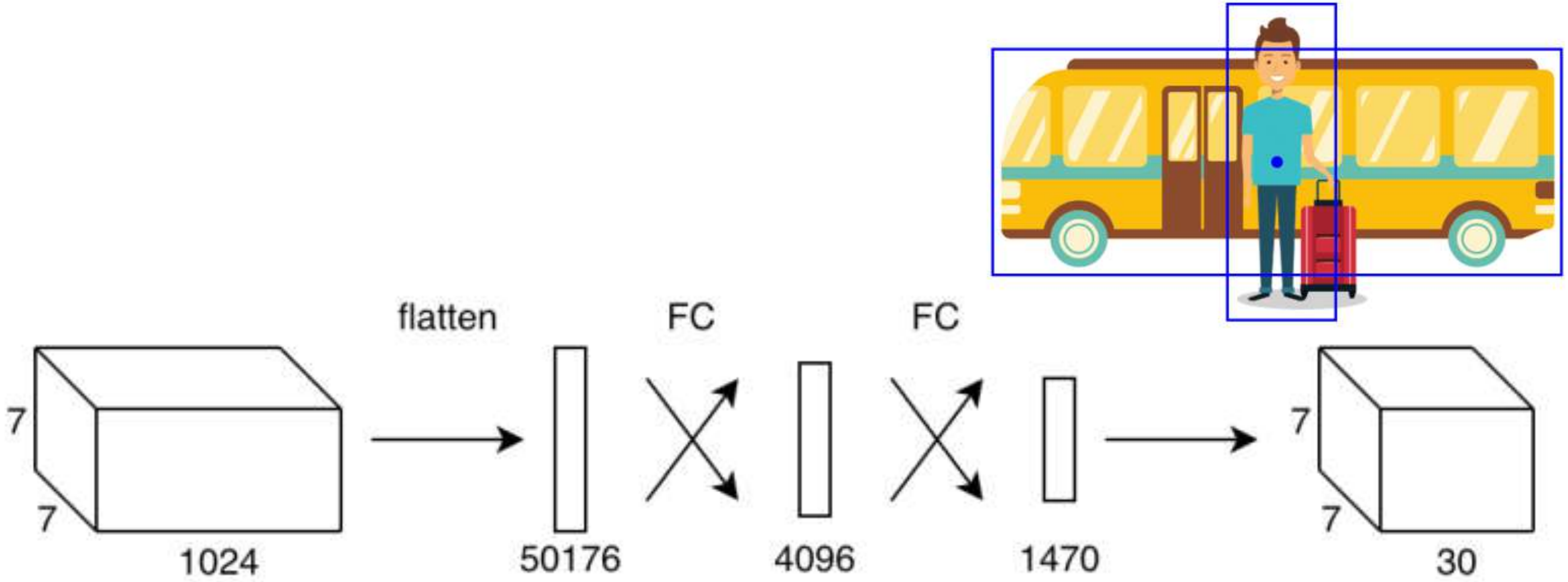
YOLO v1 only predicted two bounding boxes per grid cell, which means a total of 98 (= 7 x 7 x 2) bounding boxes per image, much lower than Faster R-CNN.



Method	mAP	FPS	batch size	# Boxes	Input resolution
Faster R-CNN (VGG16)	73.2	7	1	~ 6000	$\sim 1000 \times 600$
Fast YOLO	52.7	155	1	98	448×448
YOLO (VGG16)	66.4	21	1	98	448×448
SSD300	74.3	46	1	8732	300×300
SSD512	76.8	19	1	24564	512×512
SSD300	74.3	59	8	8732	300×300
SSD512	76.8	22	8	24564	512×512

AI VIET NAM
@aivietnam.edu.vn

YOLO-v1 vs. YOLO-v2



Two bounding boxes had to share the same class probabilities. As such, increasing the number of bounding boxes would not benefit much. On the contrary, Faster R-CNN and SSD predicted class probabilities for each bounding box, making it easier to predict multiple classes sharing a similar center location.

YOLO-v1 vs. YOLO-v2

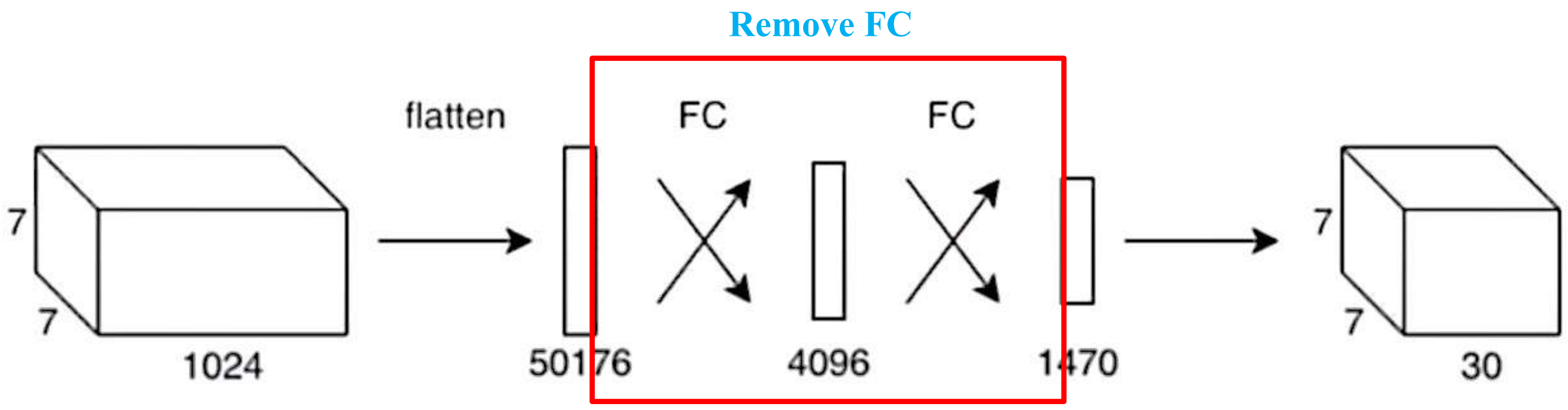
YOLOv1 was an anchor-free model that predicted the coordinates of B-boxes directly using fully connected layers in each grid cell.

Inspired by Faster-RCNN that predicts B-boxes using hand-picked priors known as anchor boxes, YOLOv2 also works on the same principle.

Unlike YOLOv1, wherein each grid cell, the model predicted one set of class probabilities per grid cell, ignoring the number of boxes B, YOLOv2 predicted class and objectness for every anchor box.

AI VIET NAM
@aivietnam.edu.vn

Problem with Bounding Boxes



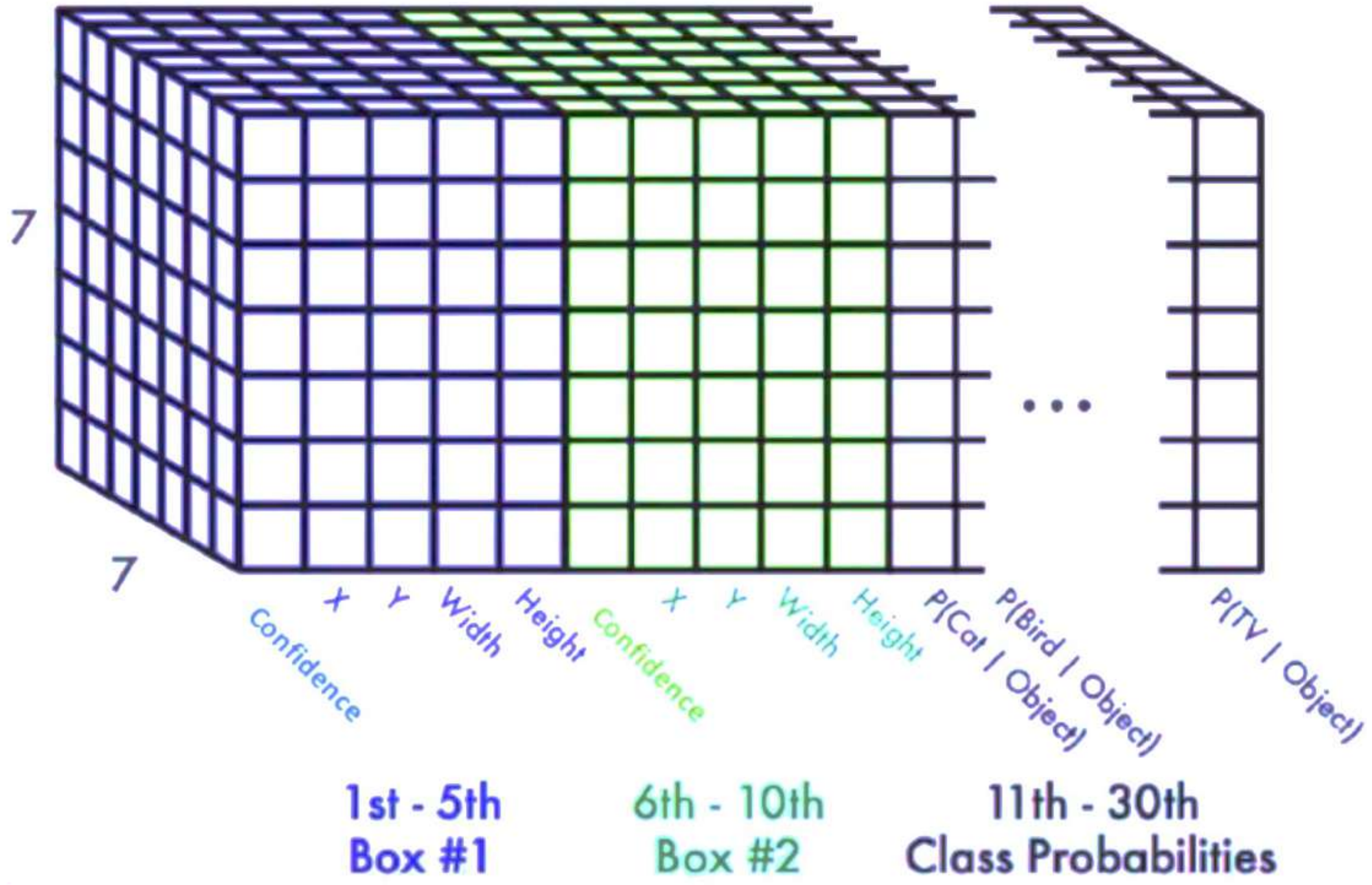
- With 13x13 grid cells - 13x13x2=338
- Should we increase the grids to 50x50?

Problem with fully connected layer => YOLOv2 is fully convolutional network

AI VIET NAM
@aivietnam.edu.vn

Problem with Bounding Boxes

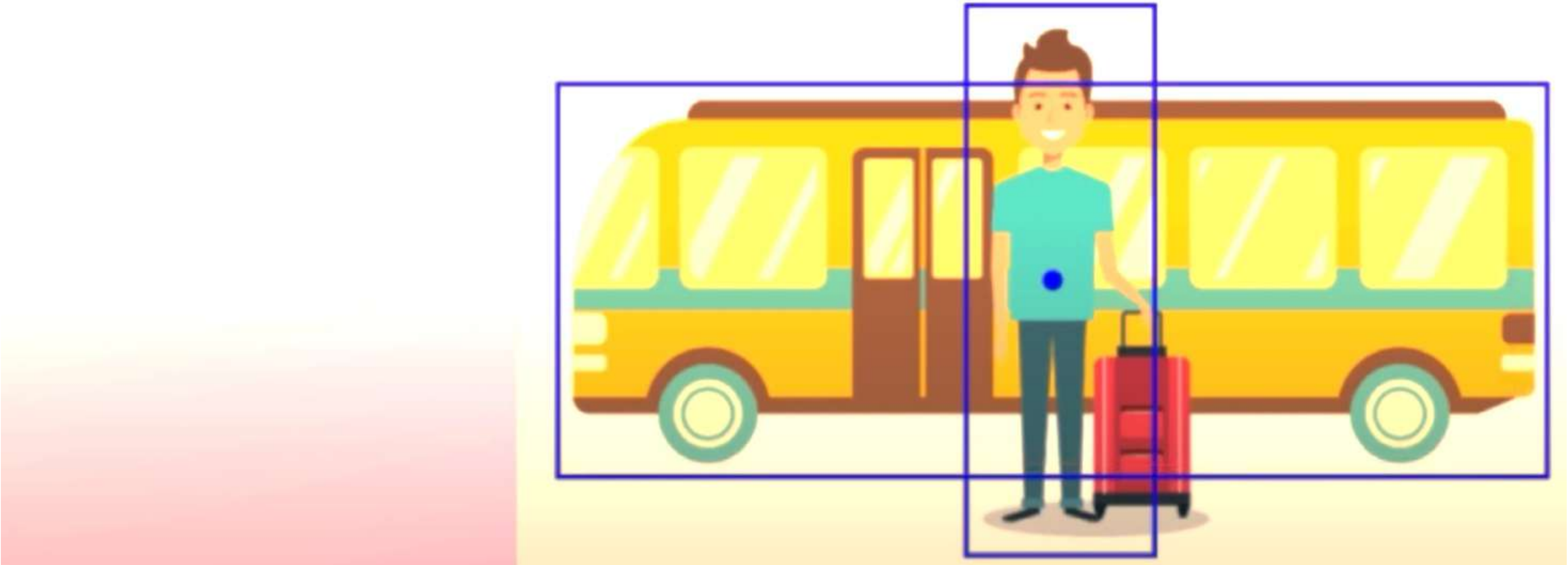
- 1 class per grid cell
- Limits the number of objects detected
- Solution: class prediction per box



AI VIET NAM
@aivietnam.edu.vn

Reason for Poor Localization

- Boxes are learnt relative to grid cell
- Objects can be of different shapes



Anchor box is a solution

AI VIET NAM
@aivietnam.edu.vn

Changes in YOLO-v2

Dimension Clusters

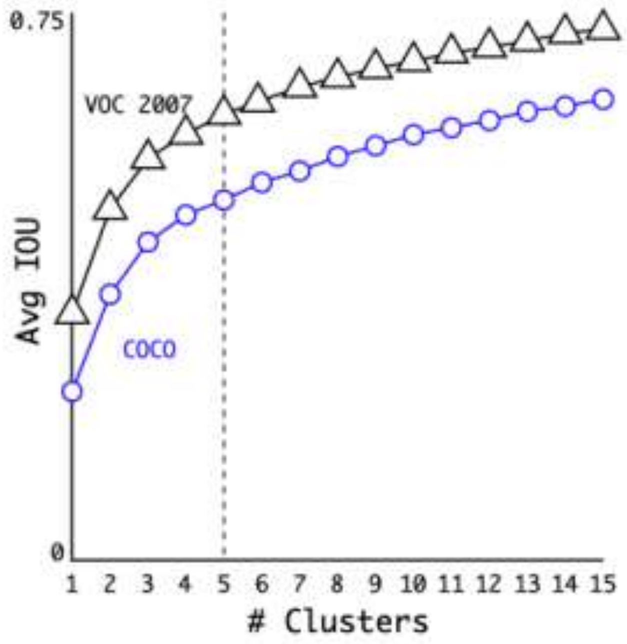
Dimension ClustersUnlike Faster-RCNN, which used hand-picked anchor boxes, YOLOv2 used a smart technique to find anchor boxes for the PASCAL VOC and MS COCO datasets.

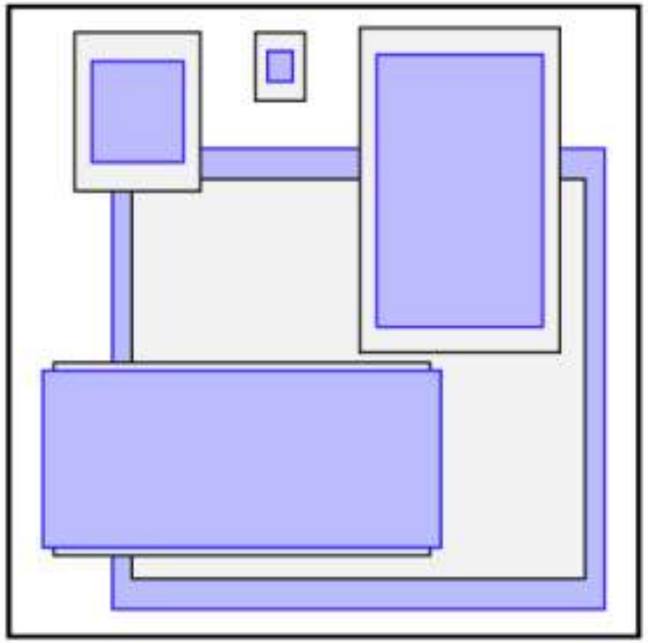
Redmon and Farhadi thought that instead of using hand-picked anchor boxes, we pick better priors that reflect the data more closely. It would be a great starting point for the network, and it would become much easier for the network to predict the detections and optimize faster.

Changes in YOLO-v2

Dimension Clusters

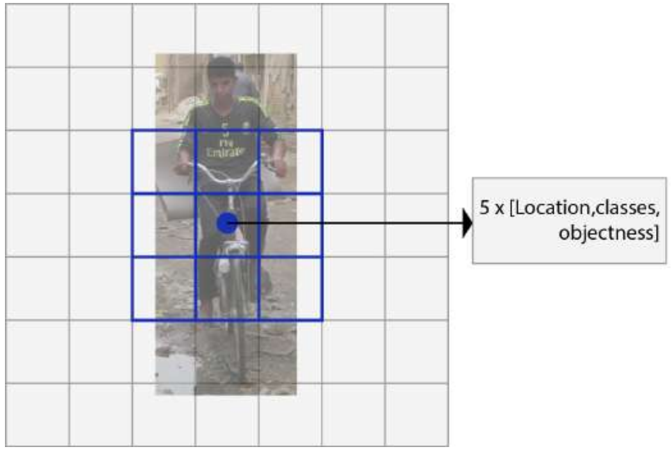
Using k-means clustering on the training set bounding boxes to find good anchor boxes or priors.





a) They picked the distance function as follows: d(box, centroid) = 1 − IOU(box, centroid).

b) They ran K-Means with a various value of *k* and found out that k=5 gives a good tradeoff between between model complexity and high recall.

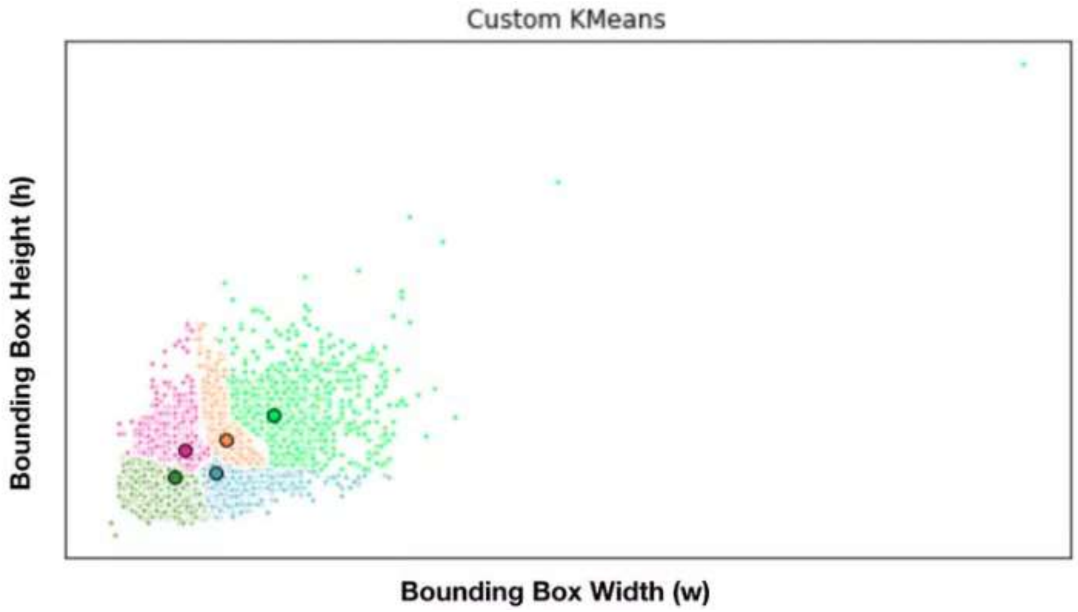


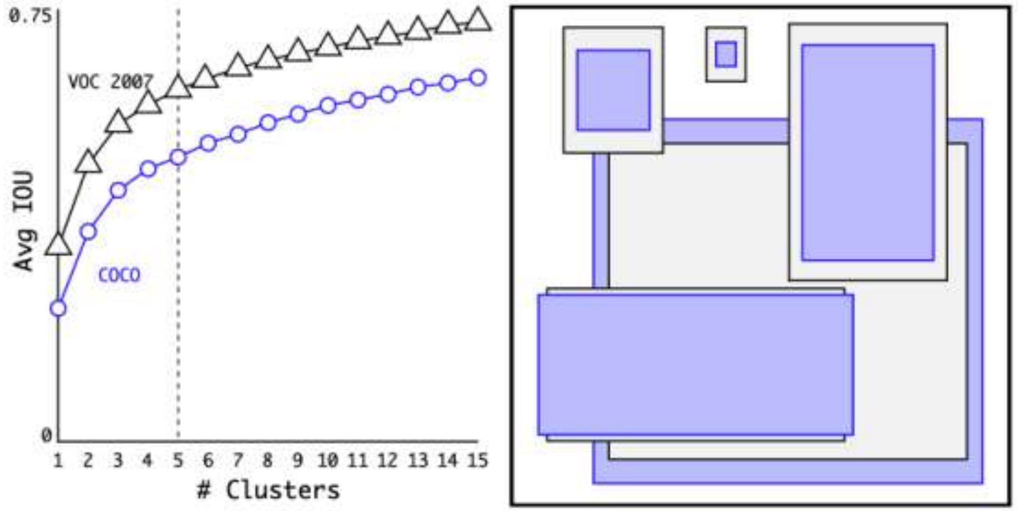
AI VIET NAM
@aivietnam.edu.vn

Changes in YOLO-v2

Dimension Clusters

Using k-means clustering on the training set bounding boxes to find good anchor boxes or priors.





AI VIET NAM
@aivietnam.edu.vn

Changes in YOLO-v2

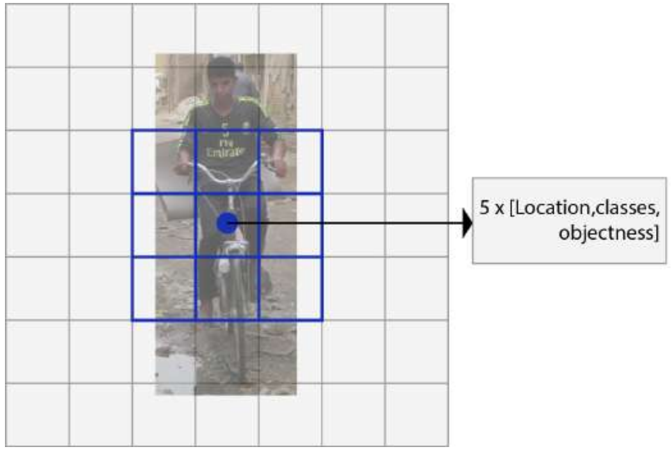
Dimension Clusters

Using k-means clustering on the training set bounding boxes to find good anchor boxes or priors.

Box Generation	Number of Anchors	Average IOU
Cluster Sum-Squared Distance	5	58.7
Cluster IOU	5	61.0
Anchor Boxes	9	60.9
Cluster IOU	9	67.2

a) They picked the distance function as follows: d(box, centroid) = 1 − IOU(box, centroid).

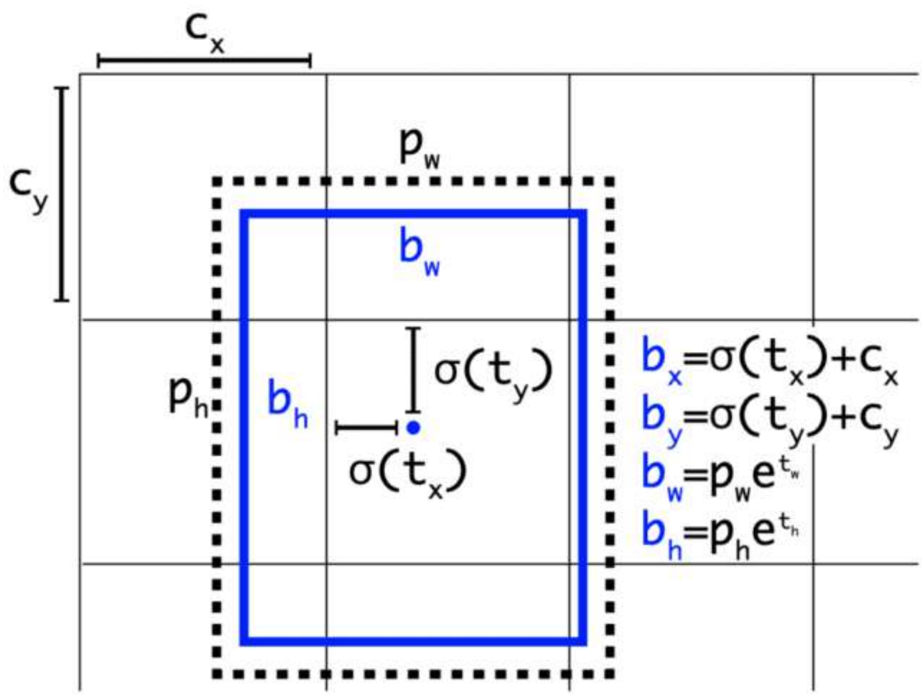
b) They ran K-Means with a various value of *k* and found out that k=5 gives a good tradeoff between between model complexity and high recall.



AI VIET NAM
@aivietnam.edu.vn

Changes in YOLO-v2

Direct Location Prediction vs. Offset location predictions



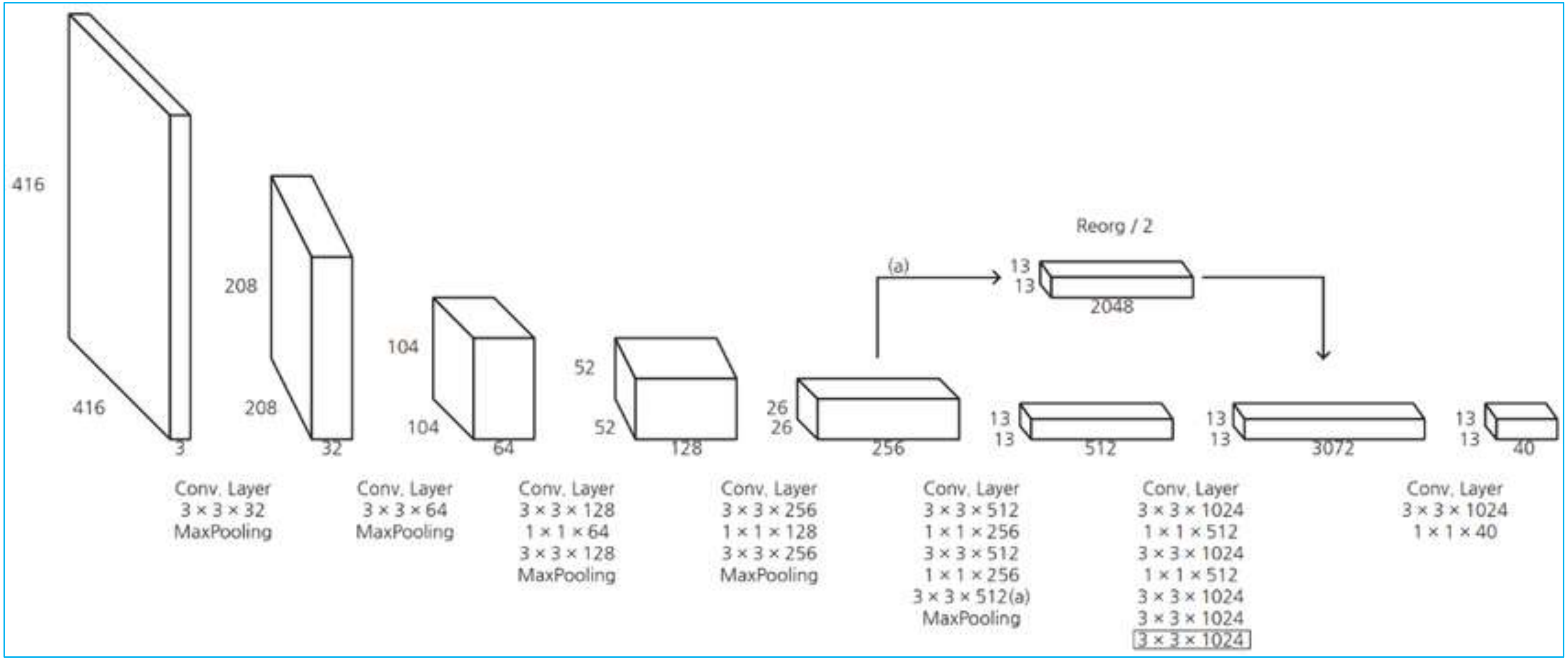
Instead of predicting the direct coordinates , they predict offsets to these bounding boxes during the training.

AI VIET NAM
@aivietnam.edu.vn

Changes in YOLO-v2

Add fine-grained feature

The idea is similar to the skip connections in ResNet



AI VIET NAM
@aivietnam.edu.vn

Changes in YOLO-v2

Multiple-Scale Training

Detection Frameworks	Training Data	mAP	FPS
Fast R-CNN	2007+2012	70.0	0.5
Faster R-CNN with VGG-16 backbone	2007+2012	73.2	7.0
Faster R-CNN with ResNet backbone	2007+2012	76.4	5.0
YOLOv1	2007+2012	63.4	45.0
SSD300	2007+2012	74.3	46.0
SSD500	2007+2012	76.8	19.0
YOLOv2 with input size 288 x 288	2007+2012	69.0	**91.0**
YOLOv2 with input size 352 x 352	2007+2012	73.7	81.0
YOLOv2 with input size 416 x 416	2007+2012	76.8	67.0
YOLOv2 with input size 480 x 480	2007+2012	77.8	59.0
YOLOv2 with input size 544 x 544	2007+2012	**78.6**	40.0

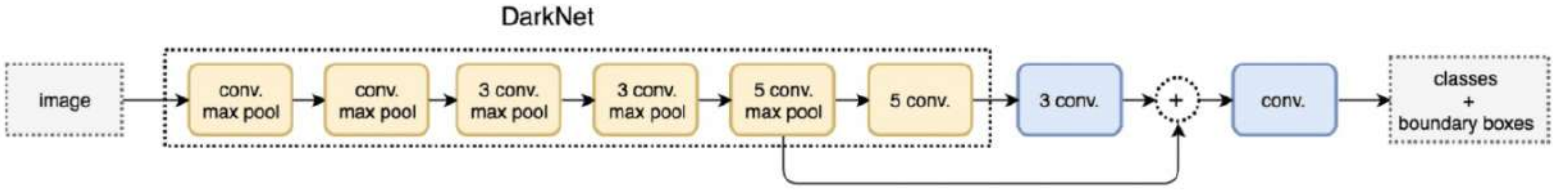
AI VIET NAM
@aivietnam.edu.vn

Changes in YOLO-v2

Light-weight backbone

Darknet-19 A fully convolutional model with 19 convolutional layers and five max-pooling layers was designed.

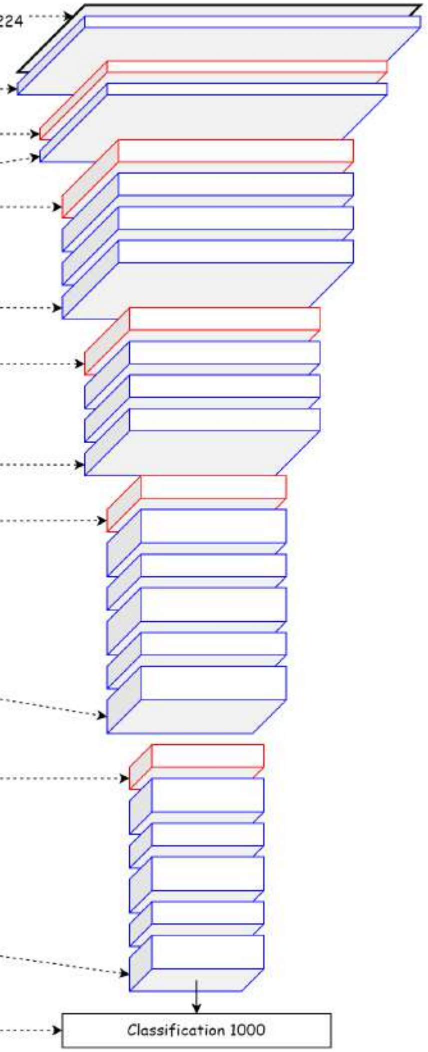
Type	Filters	Size/Stride	Output
Convolutional	32	3×3	224×224
Maxpool		$2 \times 2/2$	112×112
Convolutional	64	3×3	112×112
Maxpool		$2 \times 2/2$	56×56
Convolutional	128	3×3	56×56
Convolutional	64	1×1	56×56
Convolutional	128	3×3	56×56
Maxpool		$2 \times 2/2$	28×28
Convolutional	256	3×3	28×28
Convolutional	128	1×1	28×28
Convolutional	256	3×3	28×28
Maxpool		$2 \times 2/2$	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Maxpool		$2 \times 2/2$	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	1000	1×1	7×7
Avgpool		Global	1000
Softmax			

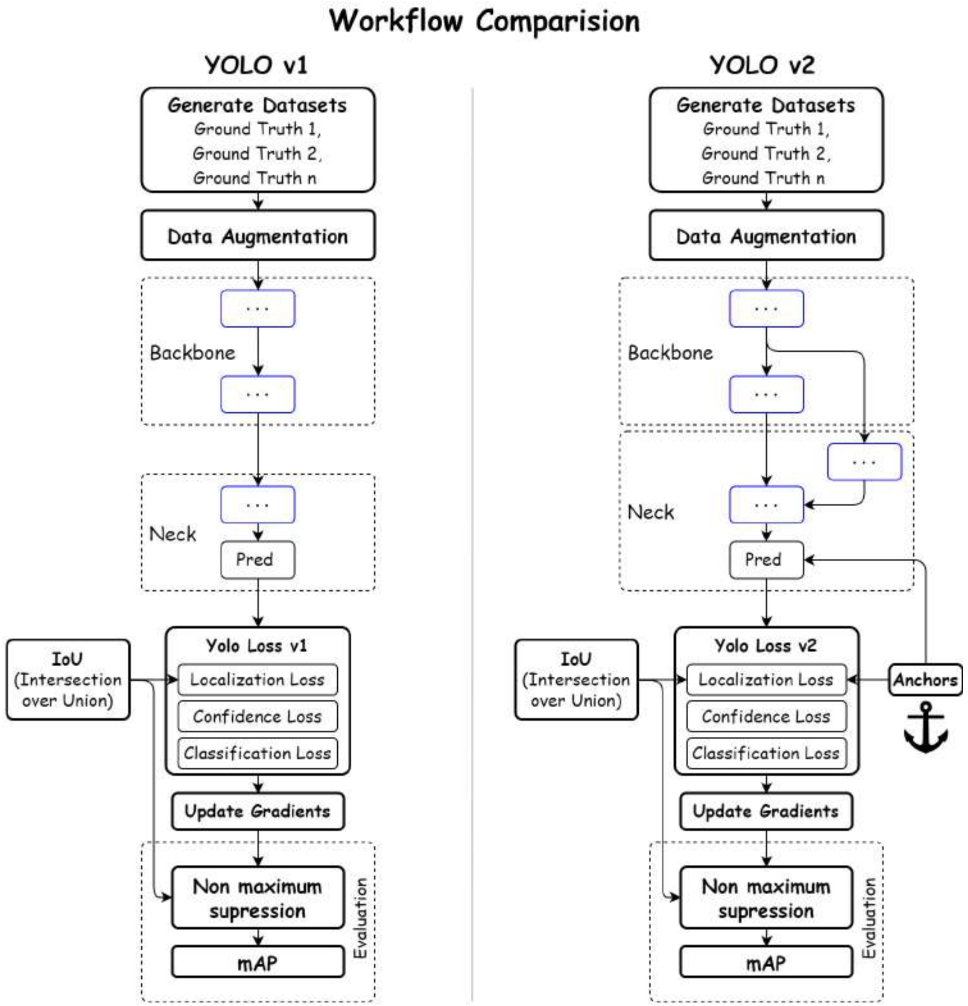


AI VIET NAM
@aivietnam.edu.vn

YOLO-v1 vs. YOLO-v2

Image	Filters	Size/Stride	Output
Convolutional	32	3 x 3	224 x 224
Maxpool		2 x 2 / 2	112 x 112
Convolutional	64	3 x 3	112 x 112
Maxpool		2 x 2 / 2	56 x 56
Convolutional	128	3 x 3	56 x 56
Convolutional	64	1 x 1	56 x 56
Convolutional	128	3 x 3	56 x 56
Maxpool		2 x 2 / 2	28 x 28
Convolutional	256	3 x 3	28 x 28
Convolutional	128	1 x 1	28 x 28
Convolutional	256	3 x 3	28 x 28
Maxpool		2 x 2 / 2	14 x 14
Convolutional	512	3 x 3	14 x 14
Convolutional	256	1 x 1	14 x 14
Convolutional	512	3 x 3	14 x 14
Convolutional	256	1 x 1	14 x 14
Convolutional	512	3 x 3	14 x 14
Maxpool		2 x 2 / 2	7 x 7
Convolutional	1024	3 x 3	7 x 7
Convolutional	512	1 x 1	7 x 7
Convolutional	1024	3 x 3	7 x 7
Convolutional	512	1 x 1	7 x 7
Convolutional	1024	3 x 3	7 x 7
Convolutional Avgpool Softmax	1000	1 x 1 Global	7 x 7



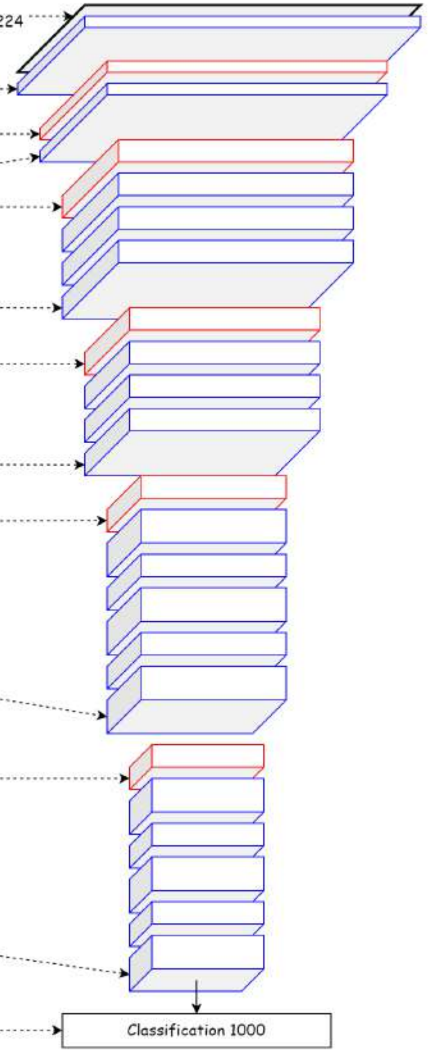


Evaluation will be explained in detail after the Yolo V3 Training.

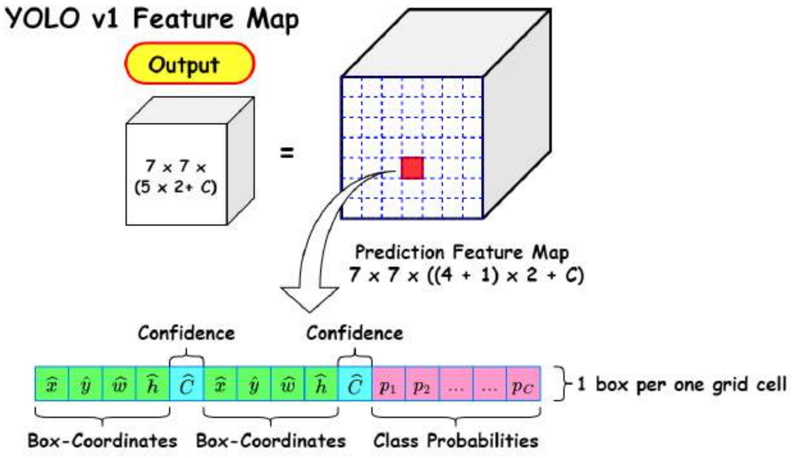
https://wikidocs.net/167705

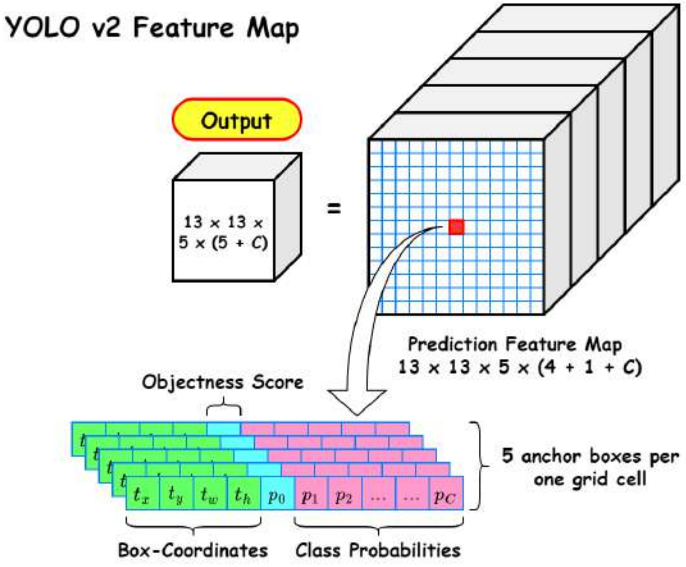
YOLO-v1 vs. YOLO-v2

Image	Filters	Size/Stride	Output
Convolutional	32	3 x 3	224 x 224
Maxpool		2 x 2 / 2	112 x 112
Convolutional	64	3 x 3	112 x 112
Maxpool		2 x 2 / 2	56 x 56
Convolutional	128	3 x 3	56 x 56
Convolutional	64	1 x 1	56 x 56
Convolutional	128	3 x 3	56 x 56
Maxpool		2 x 2 / 2	28 x 28
Convolutional	256	3 x 3	28 x 28
Convolutional	128	1 x 1	28 x 28
Convolutional	256	3 x 3	28 x 28
Maxpool		2 x 2 / 2	14 x 14
Convolutional	512	3 x 3	14 x 14
Convolutional	256	1 x 1	14 x 14
Convolutional	512	3 x 3	14 x 14
Convolutional	256	1 x 1	14 x 14
Convolutional	512	3 x 3	14 x 14
Maxpool		2 x 2 / 2	7 x 7
Convolutional	1024	3 x 3	7 x 7
Convolutional	512	1 x 1	7 x 7
Convolutional	1024	3 x 3	7 x 7
Convolutional	512	1 x 1	7 x 7
Convolutional	1024	3 x 3	7 x 7
Convolutional Avgpool Softmax	1000	1 x 1 Global	7 x 7



Head





https://wikidocs.net/167705

YOLO-v1 vs. YOLO-v2

Loss function

YOLO v1 Loss

$$
\begin{aligned}
&\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \left[(x_i - \boxed{\hat{x}_i})^2 + (y_i - \boxed{\hat{y}_i})^2 \right] \\
&+ \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \left[\left(\sqrt{w_i} - \boxed{\sqrt{\hat{w}_i}} \right)^2 + \left(\sqrt{h_i} - \boxed{\sqrt{\hat{h}_i}} \right)^2 \right] \\
&+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \left(C_i - \boxed{\hat{C}_i} \right)^2 \\
&+ \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{noobj} \left(C_i - \boxed{\hat{C}_i} \right)^2 \\
&+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{obj} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c) \right)^2
\end{aligned}
$$

YOLO v2 Loss

$$
\begin{aligned}
&\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \left[(x_i - \boxed{b_{xi}})^2 + (y_i - \boxed{b_{yi}})^2 \right] \\
&+ \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \left[\left(\sqrt{w_i} - \boxed{\sqrt{b_{wi}}} \right)^2 + \left(\sqrt{h_i} - \boxed{\sqrt{b_{hi}}} \right)^2 \right] \\
&+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \left(C_i - \boxed{\sigma(t_{oi})} \right)^2 \\
&+ \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{noobj} \left(C_i - \boxed{\sigma(t_{oi})} \right)^2 \\
&+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{obj} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c) \right)^2
\end{aligned}
$$

$$
\begin{aligned}
b_x &= \sigma(t_x) + c_x \\
b_y &= \sigma(t_y) + c_y \\
b_w &= p_w \, e^{t_w} \\
b_h &= p_h \, e^{t_h} \\
\Pr(\text{object}) \star IOU(b, \text{object}) &= \sigma(t_o)
\end{aligned}
$$

where

t_x, t_y, t_w, t_h	are predictions made by YOLO.
c_x, c_y	is the top left corner of the grid cell of the anchor.
p_w, p_h	are the width and height of the anchor.
c_x, c_y, p_w, p_h	are normalized by the image width and height.
b_x, b_y, b_w, b_h	are the predicted boundary box.
$\sigma(t_o)$	is the box confidence score.

https://wikidocs.net/167705

SoTA

Group DETR: Fast DETR Training with Group-Wise One-to-Many Assignment

Qiang Chen[1*], Xiaokang Chen[2*], Jian Wang[1], Shan Zhang[3]
Kun Yao[1], Haocheng Feng[1], Junyu Han[1], Errui Ding[1], Gang Zeng[2], Jingdong Wang[1†]
[1]Baidu VIS
[2]Key Lab. of Machine Perception (MoE), School of IST, Peking University
[3]Australian National University
{chenqiang13,wangjian33}@baidu.com
{fenghaocheng,hanjunyu,dingerrui,wangjingdong}@baidu.com
{pkucxk,gang.zeng}@pku.edu.cn, shan.zhang@anu.edu.au

arXiv:2207.13085v3 [cs.CV] 31 Aug 2023

Abstract

Detection transformer (DETR) relies on one-to-one assignment, assigning one ground-truth object to one prediction, for end-to-end detection without NMS post-processing. It is known that one-to-many assignment, assigning one ground-truth object to multiple predictions, succeeds in detection methods such as Faster R-CNN and FCOS. While the naive one-to-many assignment does not work for DETR, and it remains challenging to apply one-to-many assignment for DETR training. In this paper, we introduce Group DETR, a simple yet efficient DETR training approach that introduces a group-wise way for one-to-many assignment. This approach involves using multiple groups of object queries, conducting one-to-one assignment within each group, and performing decoder self-attention separately. It resembles data augmentation with automatically-learned object query augmentation. It is also equivalent to simultaneously training parameter-sharing networks of the same architecture, introducing more supervision and thus improving DETR training. The inference process is the same as DETR trained normally and only needs one group of queries without any architecture modification. Group DETR is versatile and is applicable to various DETR variants. The experiments show that Group DETR significantly speeds up the training convergence and improves the performance of various DETR-based models. Code will be available at https://github.com/Atten4Vis/GroupDETR.

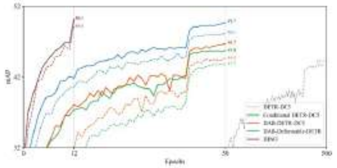


Figure 1. **Group DETR accelerates the training process for DETR variants.** The training convergence curves are obtained on COCO *val2017* [34] with ResNet-50 [22]. Dashed and bold curves correspond to the baseline models and the Group DETR counterparts. Best viewed in color.

crafted components, such as non-maximum suppression (NMS) [23] and anchor generation [44, 33, 43]. The architecture consists of a CNN [22] and transformer encoder [53], and a transformer decoder that consists of self-attention, cross-attention and FFNs, followed by class and box prediction FFNs. During training, one-to-one assignment, where one ground-truth object is assigned to one single prediction, is applied for learning to only promote the predictions assigned to ground-truth objects, and demote the duplicate predictions.

This work explores the solutions to accelerate the DETR training process. Previous solutions contain two main lines. The one line is to modify *cross-attention* so that informa-

Group DETR v2: Strong Object Detector with Encoder-Decoder Pretraining

Qiang Chen[1*], Jian Wang[1*], Chuchu Han[1*], Shan Zhang[2], Zexian Li[3], Xiaokang Chen[4], Jiahui Chen[3]
Xiaodi Wang[1], Shuming Han[1], Gang Zhang[1], Haocheng Feng[1], Kun Yao[1], Junyu Han[1], Errui Ding[1]
Jingdong Wang[1†]

[1]Baidu VIS [2]Australian National University [3]Beihang University [4]Peking University

arXiv:2211.03594v1 [cs.CV] 7 Nov 2022

Table 1. **Our method establishes a new SoTA on the COCO *test-dev* leaderboard.**

Method	#Params	Encoder Pretraining Data	Detector Pretraining Data	w/ Mask	mAP
Swin-L (HTC++) [16]	284M	IN-22K (14M)	n/a	✓	58.7
DyHead (Swin-L) [6]	213M	IN-22K (14M)	n/a	✓	60.6
Soft-Teacher (Swin-L) [25]	284M	IN-22K (14M)	COCO-unlabeled + O365	✓	61.3
GLIP (DyHead) [11]	≥284M	IN-22K (14M)	FourODs + GoldG + Cap24M	×	61.5
Florence (CoSwin-H) [29]	≥637M	FLD-900M (900M)	FLD-9M	×	62.4
GLIPv2 (CoSwin-H) [29]	≥637M	FLD-900M (900M)	FourODs + INBoxes + GoldG + CC15M + SBU	✓	62.4
SwinV2-G (HTC++) [15]	3.0B	IN-22K + ext-70M (84M)	O365	✓	63.1
DINO (Swin-L) [28]	218M	IN-22K (14M)	O365	×	63.3
BEIT-3 (ViTDet) [22]	1.9B	IN-22K + Image-Text (35M) + Text (160GB)	O365	✓	63.7
FD-SwinV2-G (HTC++) [23]	3.0B	IN-22K + IN-1K + ext-70M (85M)	O365	✓	64.2
FocalNet-H (DINO) [26]	746M	IN-22K (14M)	O365	×	64.3
Group DETR v2 (Our method)	629M	**IN-1K (1M)**	O365	×	**64.5**

All the results are achieved with test time augmentation. In the table, we follow the notations for various datasets used in DINO [28] and FocalNet [26]. 'w/ Mask' means using mask annotations when finetuning the detectors on COCO [13].

Abstract

*We present a strong object detector with encoder-decoder pretraining and finetuning. Our method, called Group DETR v2, is built upon a vision transformer encoder ViT-Huge [8], a DETR variant DINO [28], and an efficient DETR training method Group DETR [3]. The training process consists of self-supervised pretraining and finetuning a ViT-Huge encoder on ImageNet-1K, pretraining the detector on Object365, and finally finetuning it on COCO. Group DETR v2 achieves **64.5** mAP on COCO test-dev, and establishes a new SoTA on the COCO leaderboard*[1].

to achieve superior results on various vision tasks, including object detection. With supervised encoder-decoder pretraining on a large-scale dataset, Object365 [20], DINO [28] achieves a state-of-the-art result on COCO [13].

Our method, *Group DETR v2*, is built upon ViT-Huge, DINO, and Group DETR. We adopt an encoder-decoder pretraining and finetuning pipeline: pretraining and then finetuning a ViT-Huge encoder on ImageNet-1K [7], pretraining the detector, both the encoder and the decoder, on Object365, and finally finetuning it on COCO. Group DETR v2 achieves **64.5** mAP on COCO test-dev [13] (Table 1 and Table 2), setting a new record for COCO object

Detection Transformer with Stable Matching

**Shilong Liu[1,2*†], Tianhe Ren[2*], Jiayu Chen[3*],
Zhaoyang Zeng[2], Hao Zhang[2,4], Feng Li[2,4], Hongyang Li[2,5],
Jun Huang[3], Hang Su[1], Jun Zhu[1‡], Lei Zhang[2‡]**

[1] Dept. of Comp. Sci. and Tech., BNRist Center, State Key Lab for Intell. Tech. & Sys.,
Institute for AI, Tsinghua-Bosch Joint Center for ML, Tsinghua University
[2] International Digital Economy Academy (IDEA) [3] Platform of AI (PAI), Alibaba Group
[4] The Hong Kong University of Science and Technology
[5] South China University of Technology

liusl20@mails.tsinghua.edu.cn {rentianhe, zengzhaoyang}@idea.edu.cn {yunji.cjy, huangjun.hj}@alibaba-inc.com
{hzhangcx, fliay}@connect.ust.hk ftwangyeunglei@mail.scut.edu.cn {suhangss, dcszj}@mail.tsinghua.edu.cn leizhang@idea.edu.cn

Abstract

*This paper is concerned with the matching stability problem across different decoder layers in DEtection TRansformers (DETR). We point out that the unstable matching in DETR is caused by a multi-optimization path problem, which is highlighted by the one-to-one matching design in DETR. To address this problem, we show that the most important design is to use and only use positional metrics (like IOU) to supervise classification scores of positive examples. Under the principle, we propose two simple yet effective modifications by integrating positional metrics to DETR's classification loss and matching cost, named position-supervised loss and position-modulated cost. We verify our methods on several DETR variants. Our methods show consistent improvements over baselines. By integrating our methods with DINO, we achieve 50.4 and 51.5 AP on the COCO detection benchmark using ResNet-50 backbones under 1× (12 epochs) and 2× (24 epochs) training settings, achieving a new record under the same setting. We achieve **63.8** AP on COCO detection test-dev with a Swin-Large backbone. Our code will be made available at* https://github.com/IDEA-Research/Stable-DINO.

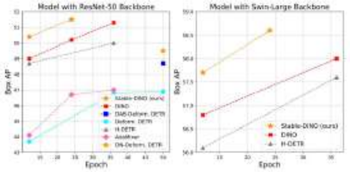


Figure 1: Comparison of our methods (named Stable-DINO in figures) and baselines. We compare models with ResNet-50 backbones in the left figure and models with Swin-Transformer Large backbones in the right figure. All models use a maximum 1/8 resolution feature map from a backbone, except AdaMixer uses a maximum 1/4 resolution feature map.

decades with the development of deep learning, especially the convolutional neural network (CNN) [36, 14, 16, 7].

Detection Transformer (DETR) [3] proposed a novel Transformer-based object detector, which attracted a lot of interest in the research community. It gets rid of the

SAP-DETR: Bridging the Gap between Salient Points and Queries-Based Transformer Detector for Fast Model Convergency

Yang Liu[1,3*] Yao Zhang[1,3*] Yixin Wang[2*] Yang Zhang[4] Jiang Tian[4]
Zhongchao Shi[4] Jianping Fan[4] Zhiqiang He[1,3,5†]
[1]Institute of Computing Technology (ICT), Chinese Academy of Sciences [2]Stanford University
[3]University of Chinese Academy of Sciences [4]AI Lab, Lenovo Research [5]Lenovo Ltd.
{liuyang20c,zhangyao215}@mails.ucas.ac.cn yixinwang@stanford.edu
{zhangyang20,tianjiang1,shizc2,jfan1,hezq}@lenovo.com

arXiv:2211.02006v2 [cs.CV] 15 Mar 2023

Abstract

Recently, the dominant DETR-based approaches apply central-concept spatial prior to accelerating Transformer detector convergency. These methods gradually refine the reference points to the center of target objects and imbue object queries with the updated central reference information for spatially conditional attention. However, centralizing reference points may severely deteriorate queries' saliency and confuse detectors due to the indiscriminative spatial prior. To bridge the gap between the reference points of salient queries and Transformer detectors, we propose SAlient Point-based DETR (SAP-DETR) by treating object detection as a transformation from salient points to instance objects. Concretely, we explicitly initialize a query-specific reference point for each object query, gradually aggregate them into an instance object, and then predict the distance from each side of the bounding box to these points. By rapidly attending to query-specific reference regions and the conditional box edges, SAP-DETR can effectively bridge the gap between the salient point and the query-based Transformer detector with a significant convergency speed. Experimentally, SAP-DETR achieves 1.4× convergency speed with competitive performance and stably promotes the SoTA approaches by ∼1.0 AP. Based on ResNet-DC-101, SAP-DETR achieves 46.9 AP. The code will be released at https://

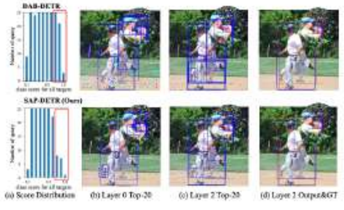


Figure 1. Comparison of SAP-DETR and DAB-DETR under 36 training epochs. (a) Statistics of the query count in different classification score intervals. (b) and (c) Distribution of reference points and the visualization of the query with top-20 classification score (blue proposal bounding boxes and red reference points) in different decoder layers. (d) Visualization of bounding boxes for positive queries (blue) and ground truth (red) during training process.

tors [6, 11, 14, 18, 20, 22] based on Convolutional Neural Networks (CNNs), have received widespread attention and made significant progress. Recently, Carion *et al.* [2] proposed a new end-to-end paradigm for object detection based on the Transformer [24], called DEtection TRansformer (DETR),

DETRs with Collaborative Hybrid Assignments Training

Zhuofan Zong Guanglu Song Yu Liu*
SenseTime Research
{zongzhuofan,liuyuisanai}@gmail.com
songguanglu@sensetime.com

arXiv:2211.12860v6 [cs.CV] 10 Aug 2023

Abstract

In this paper, we provide the observation that too few queries assigned as positive samples in DETR with one-to-one set matching leads to sparse supervision on the encoder's output which considerably hurt the discriminative feature learning of the encoder and vice visa for attention learning in the decoder. To alleviate this, we present a novel collaborative hybrid assignments training scheme, namely Co-DETR, to learn more efficient and effective DETR-based detectors from versatile label assignment manners. This new training scheme can easily enhance the encoder's learning ability in end-to-end detectors by training the multiple parallel auxiliary heads supervised by one-to-many label assignments such as ATSS and Faster RCNN. In addition, we conduct extra customized positive queries by extracting the positive coordinates from these auxiliary heads to improve the training efficiency of positive samples in the decoder. In inference, these auxiliary heads are discarded and thus our method introduces no additional parameters and computational cost to the original detector while requiring no hand-crafted non-maximum suppression (NMS). We conduct extensive experiments to evaluate the effectiveness of the proposed approach on DETR variants, including DAB-DETR, Deformable-DETR, and DINO-Deformable-DETR. The state-of-the-art DINO-Deformable-DETR with Swin-L can be improved from 58.5% to 59.5% AP on COCO val. Surprisingly, incorporated with ViT-L backbone, we achieve 66.0% AP on COCO test-dev and 67.9% AP on LVIS val, outperforming previous methods by clear margins with much fewer model sizes. Codes are available at https://github.com/Sense-X/Co-DETR.

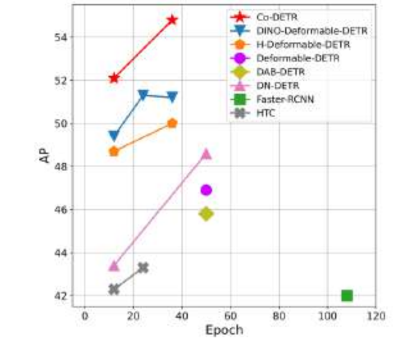


Figure 1. Performance of models with ResNet-50 on COCO val. Co-DETR outperforms other counterparts by a large margin.

a series of variants [31, 37, 44] such as ATSS [41], RetinaNet [21], FCOS [32], and PAA [17] lead to the significant breakthrough of object detection task. One-to-many label assignment is the core scheme of them, where each ground-truth box is assigned to multiple coordinates in the detector's output as the supervised target cooperated with proposals [11, 27], anchors [21] or window centers [32]. Despite their promising performance, these detectors heavily rely on many hand-designed components like a non-maximum suppression procedure or anchor generation [1]. To conduct a more flexible end-to-end detector, DEtection TRansformer (DETR) [1] is proposed to view the object detection as a set prediction problem and introduce the one-to-one set matching scheme based on a transformer encoder-decoder