로봇을 위한 인공지능 기반 물체검출

한양대학교
최준원

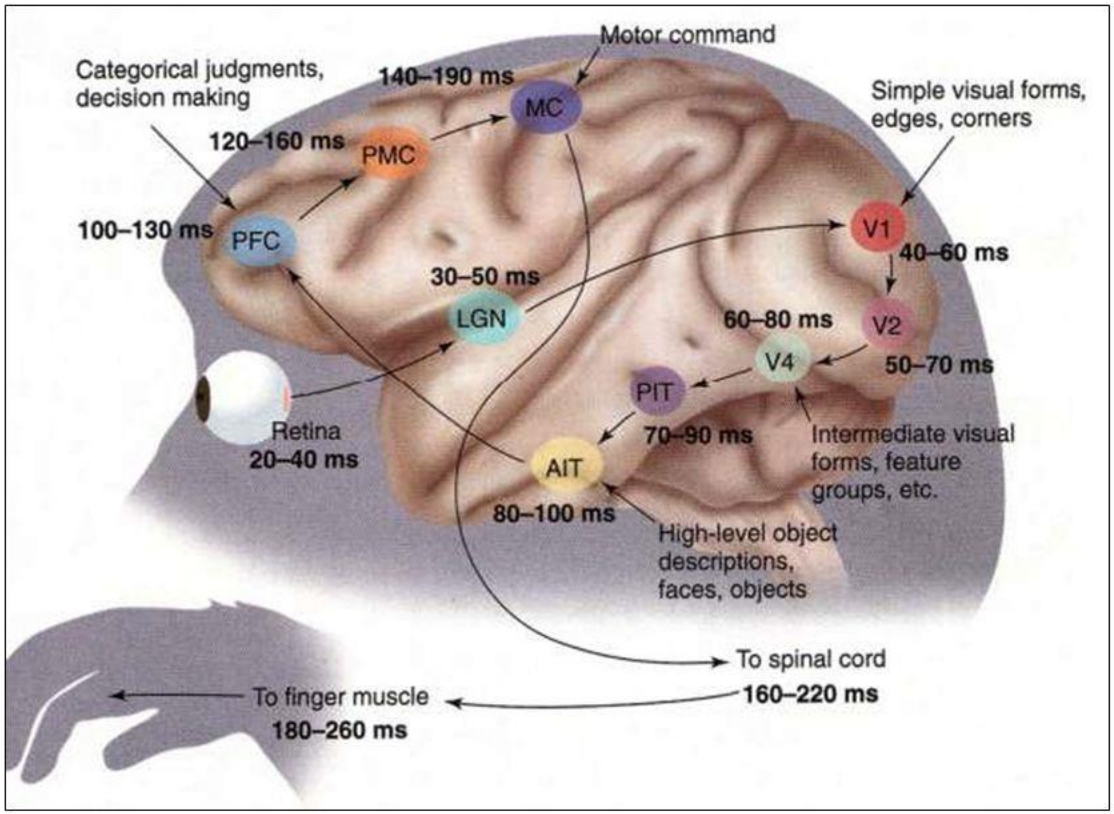
Contents

- CNN
- Region based object detector
  - R-CNN [1] (CVPR 2014)
  - Fast R-CNN [2] (ICCV 2015)
  - Faster R-CNN [3] (NIPS 2015)
- Single stage object detector
  - SSD: Single Shot multibox Detector [4] (ECCV 2016)

[1] Girshick, Ross, et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2014.
[2] Girshick, Ross. "Fast r-cnn." Proceedings of the IEEE international conference on computer vision. 2015.
[3] Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." Advances in neural information processing systems. 2015.
[4] Liu, Wei, et al. "Ssd: Single shot multibox detector." European conference on computer vision. Springer, Cham, 2016.

Convolutional neural network

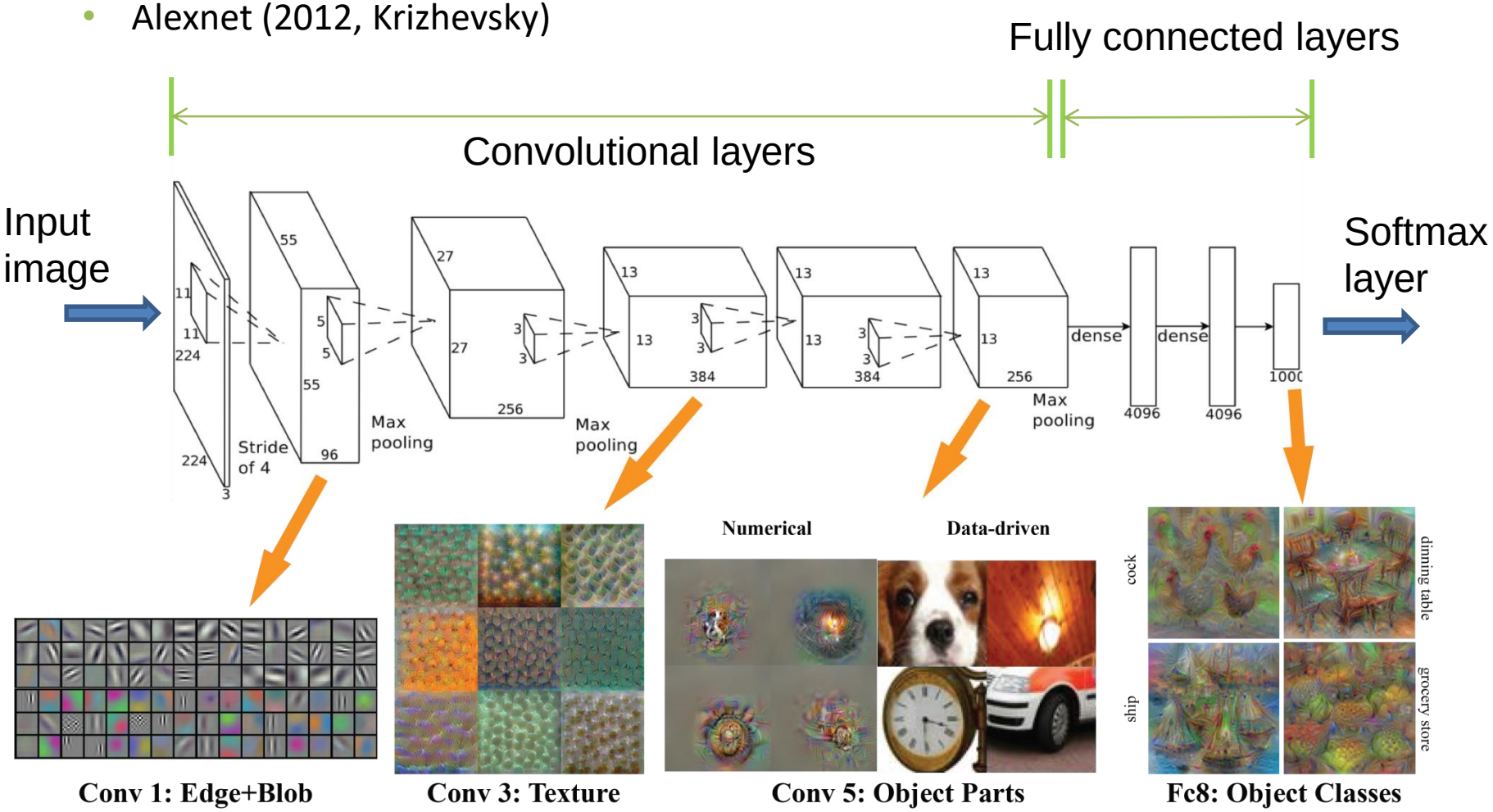
- Motivation
  - Inspired by human's biological data processing
  - Human visual system



Latences neuronales pour le ttt visuel (Thorpe & Fabre-thorpe, 2001)

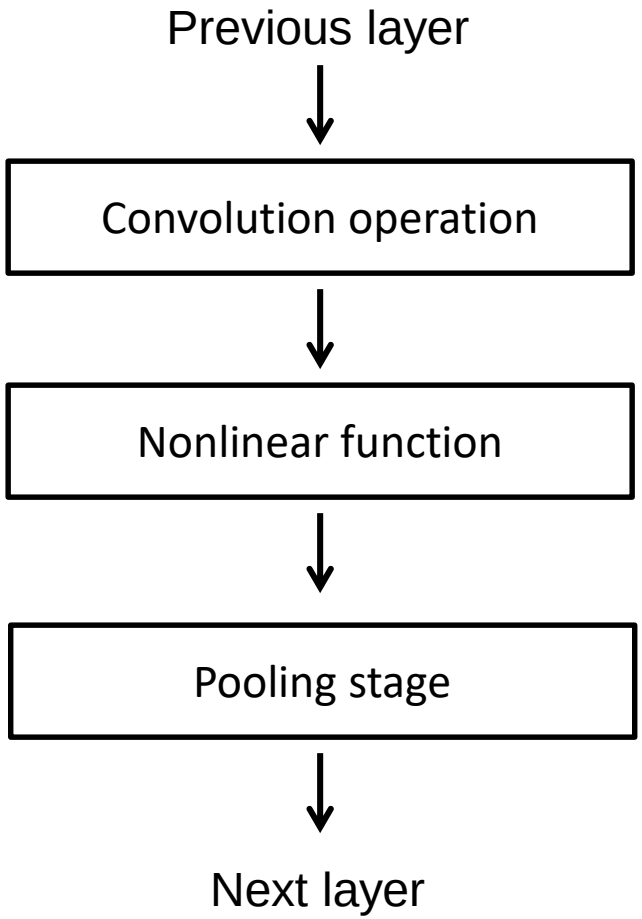
Convolutional neural network

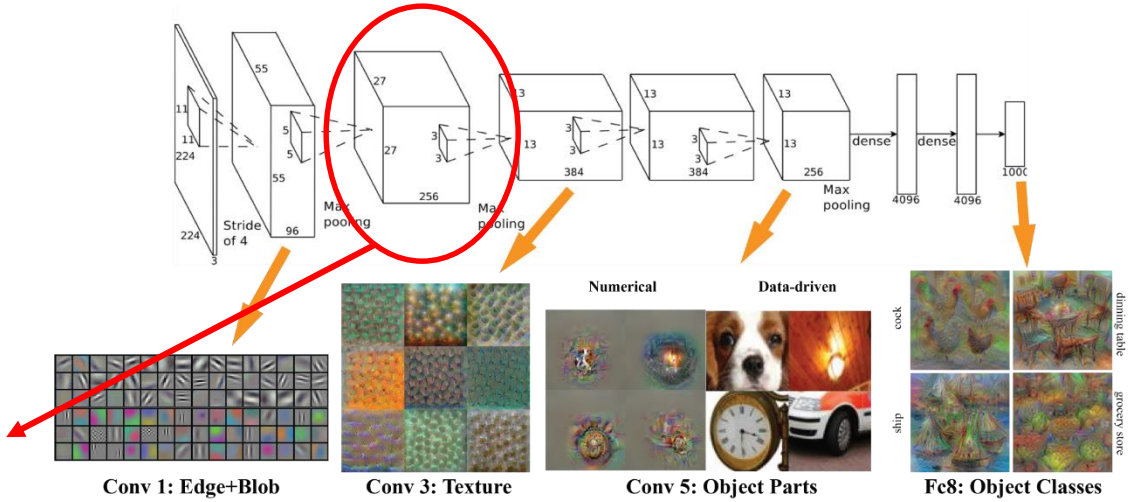
- Excellent for the data with grid-like topology
  - Alexnet (2012, Krizhevsky)



Convolutional neural network

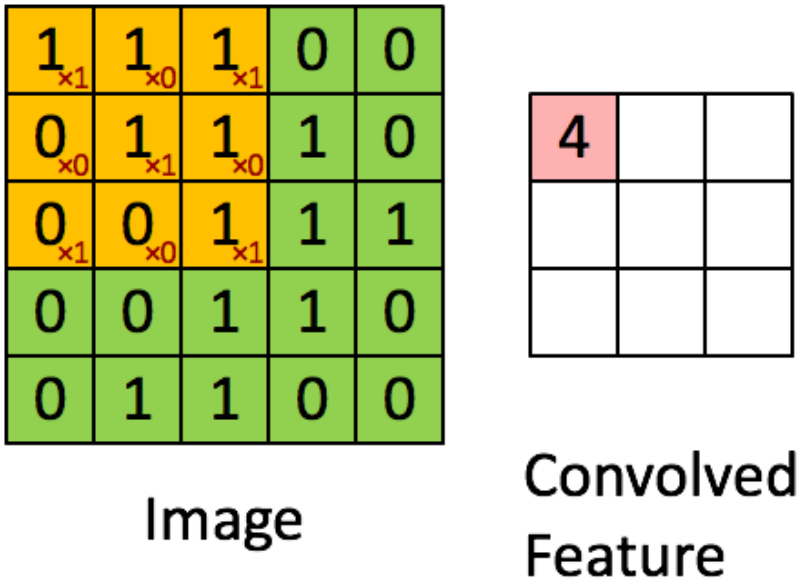
- Key steps for CNN





Building blocks of CNN

- Convolution with 3-dimensional kernel
  - Small two-dimensional filters are applied for each position of data.
  - We can skip sweeping convolution with stride k
  - Captures local connectivity
  - **Parameter sharing** significantly reduces the number of parameters that should be learned.



Building blocks of CNN

- Max-pooling
  - Pick the strongest activation in the region
  - Max-pooling allows the feature invariant to translation

Max-pooling

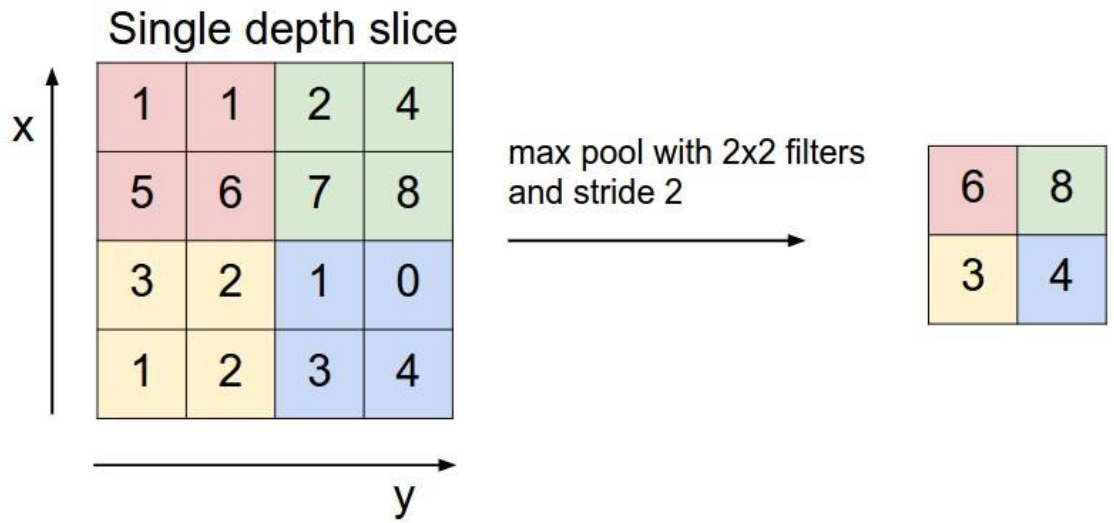
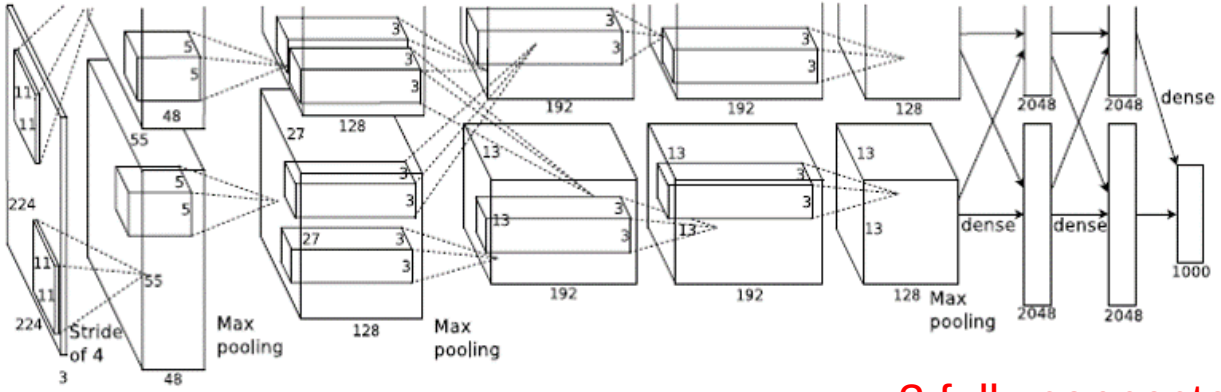


Image Classification via CNN

- Alexnet (2012, Krizhevsky)



5 convolutional layers

3 fully connected layers

- VGGnet (2014)
  - 3x3 convolutional layers are cascaded.

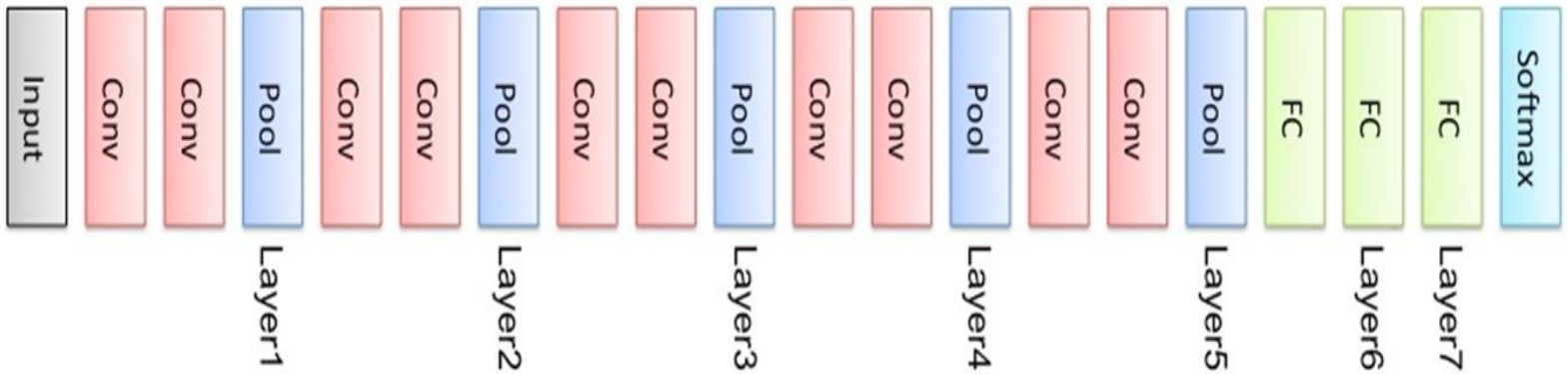
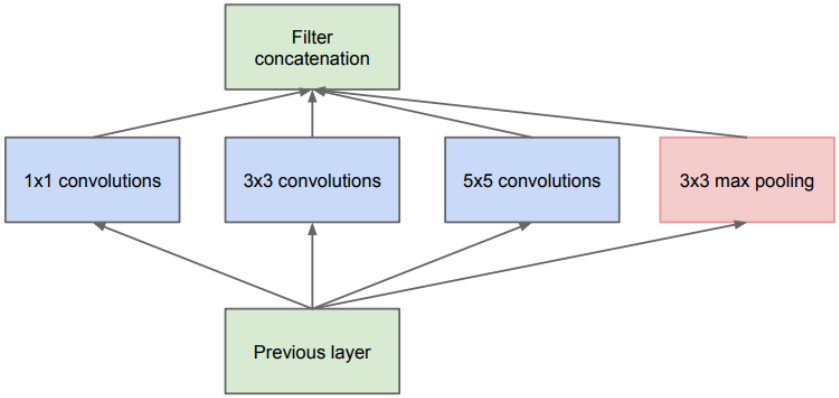
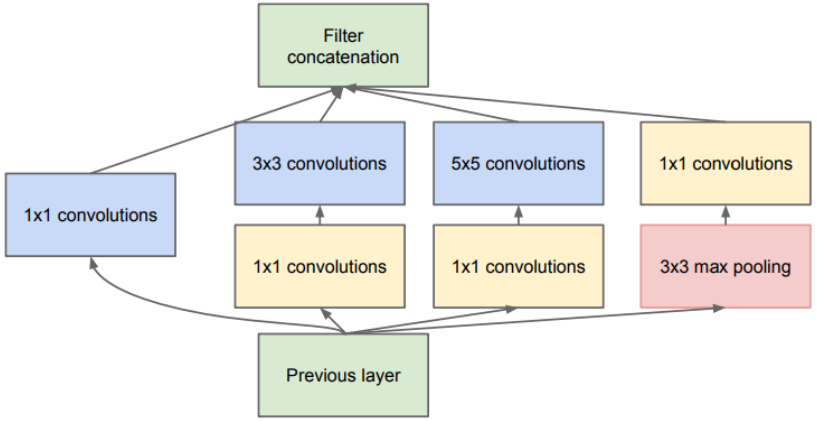


Image Classification via CNN

- GooleNet (2014)
  - Use inception module





- ResidualNet (2015)
  - Skip connection is introduced to improve network optimization.

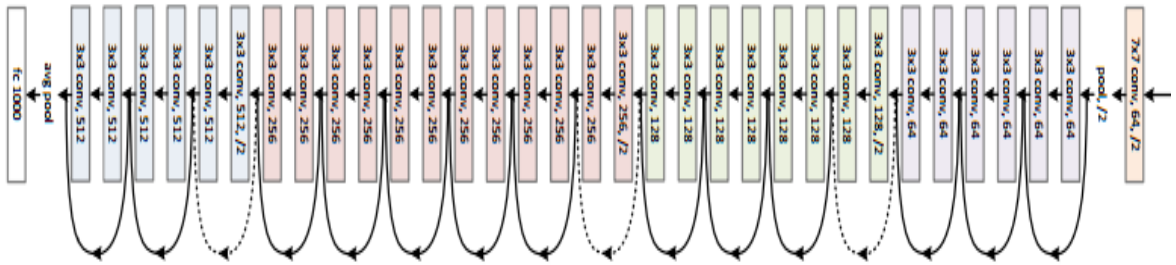
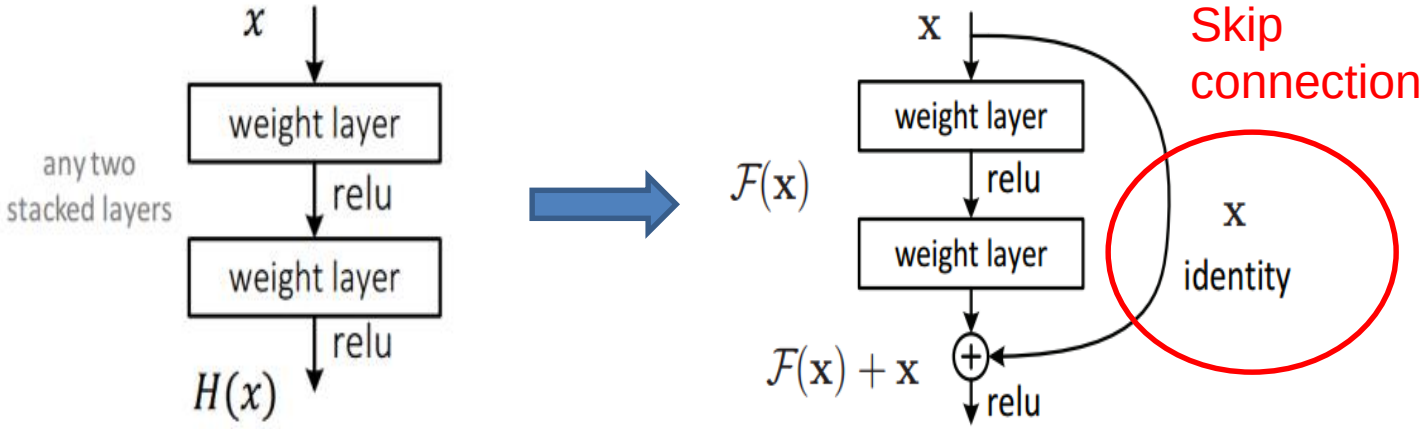
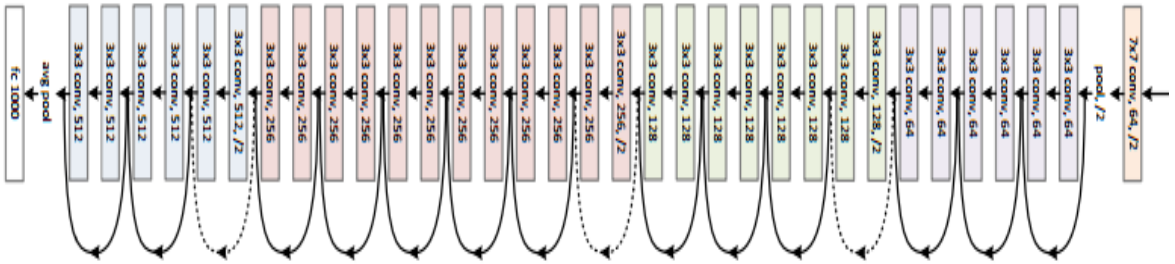


Image Classification via CNN

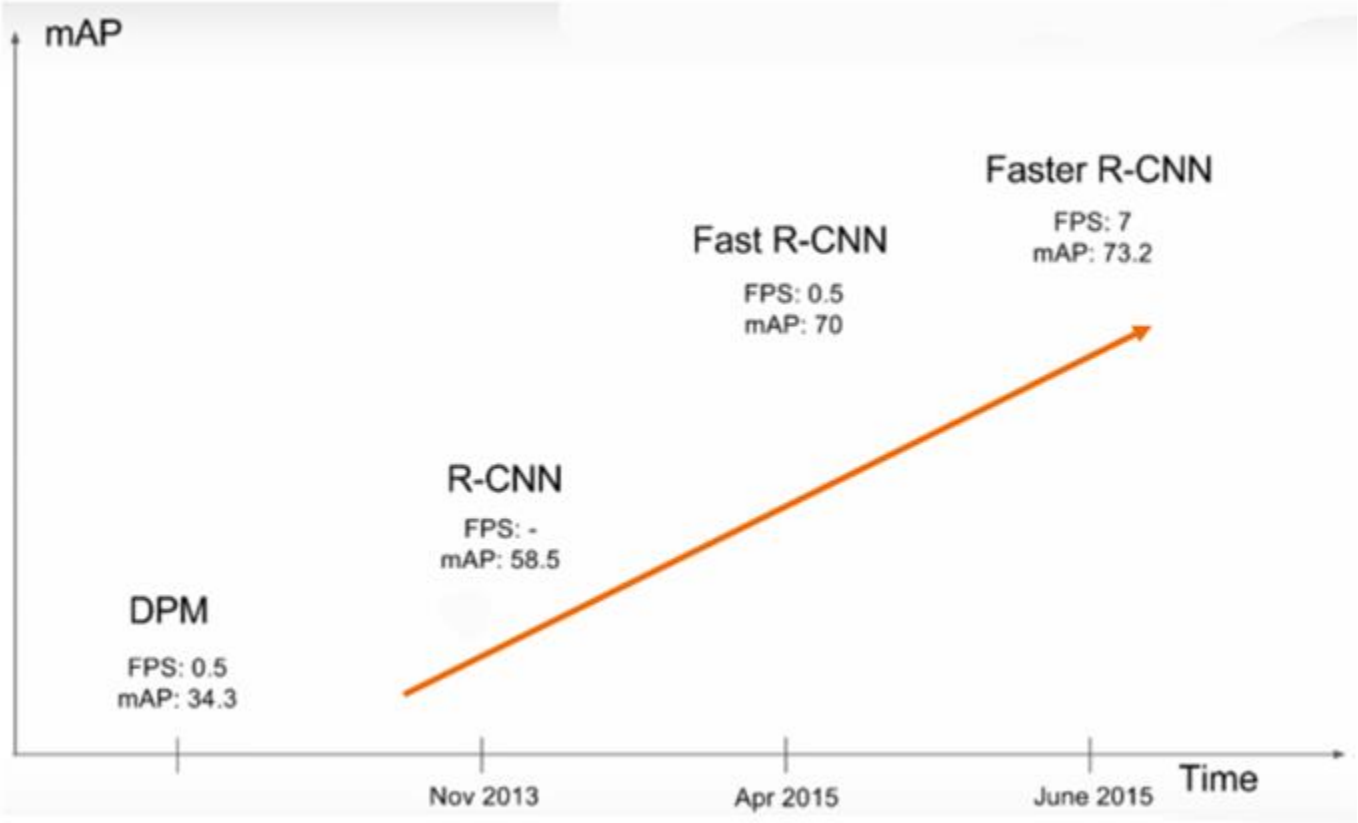
- ResidualNet (2015)
  - Skip connection is introduced to improve network optimization.
  - The residual function $F(x) = H(x) - x$ is learned rather than the original function $H(x)$.





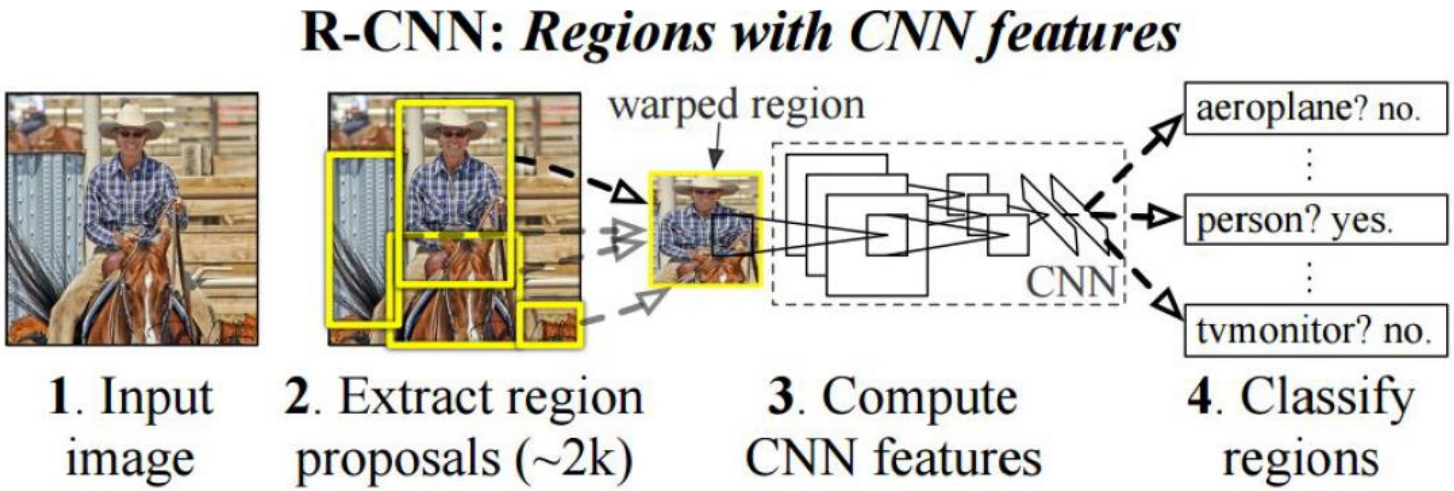
Region-Based Object Detector

- R-CNN (Region-based CNN)
  - First object detector using CNN
    - Speed (FPS) and accuracy comparison



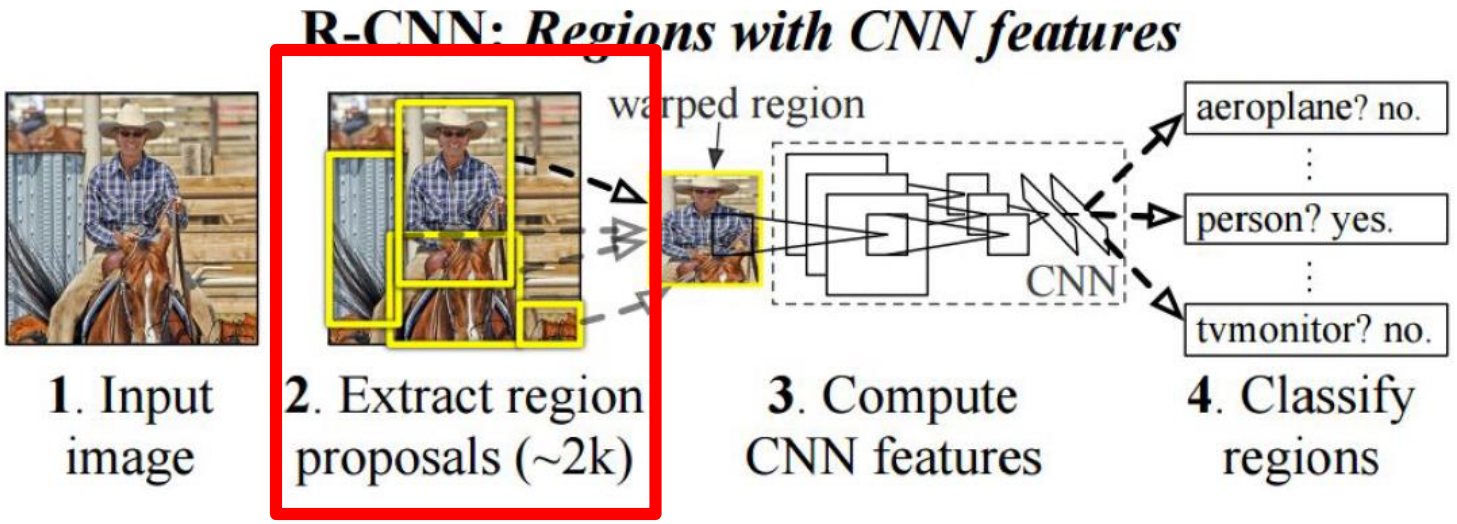
Region-Based Object Detector

- R-CNN (Region-based CNN)
  - First object detection method using CNN.
  - The **selective search** algorithm is used for region proposals.
  - **AlexNet** is used for feature extraction.
  - **Support vector machine(SVM)** is used for classification and the bounding-box regression is used to improve localization performance.



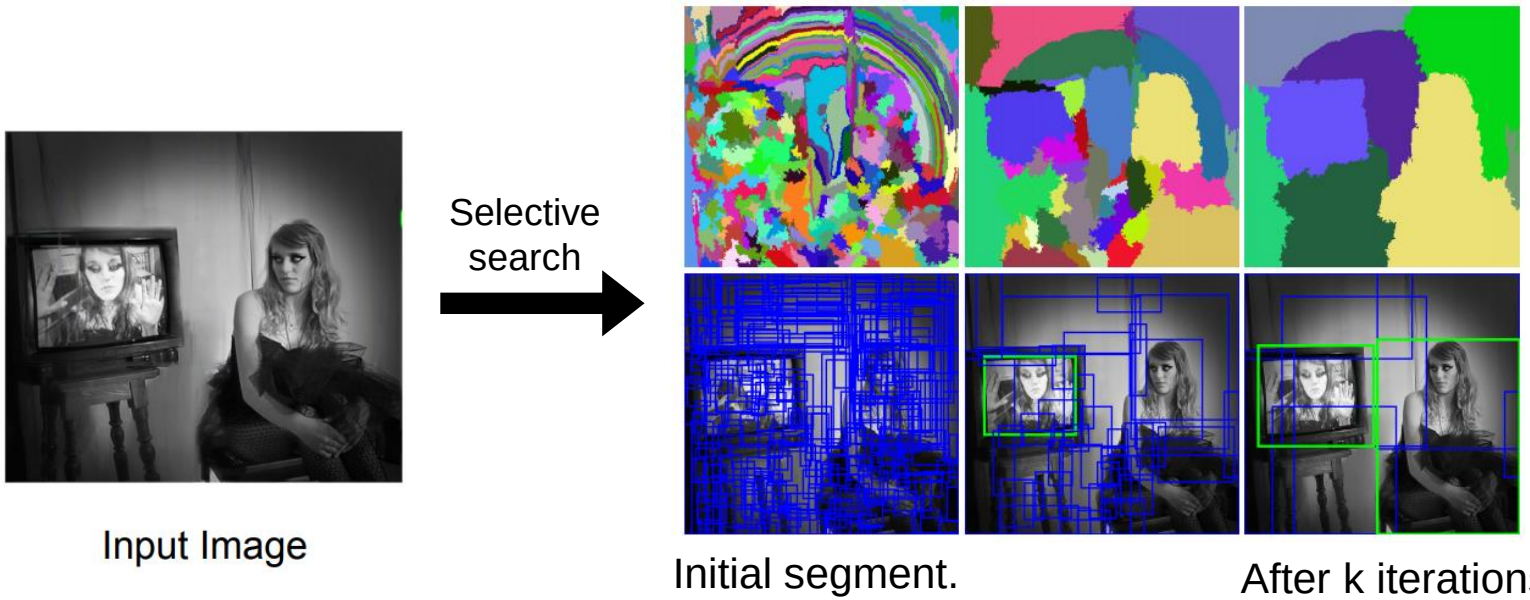
Region-Based Object Detector

- R-CNN
  - Algorithm procedure
    1) **Region proposal**
       - **Selective search**
    2) Crop & wrap region
    3) CNN for feature extraction
    4) Classification (SVM) and box regression



Region-Based Object Detector

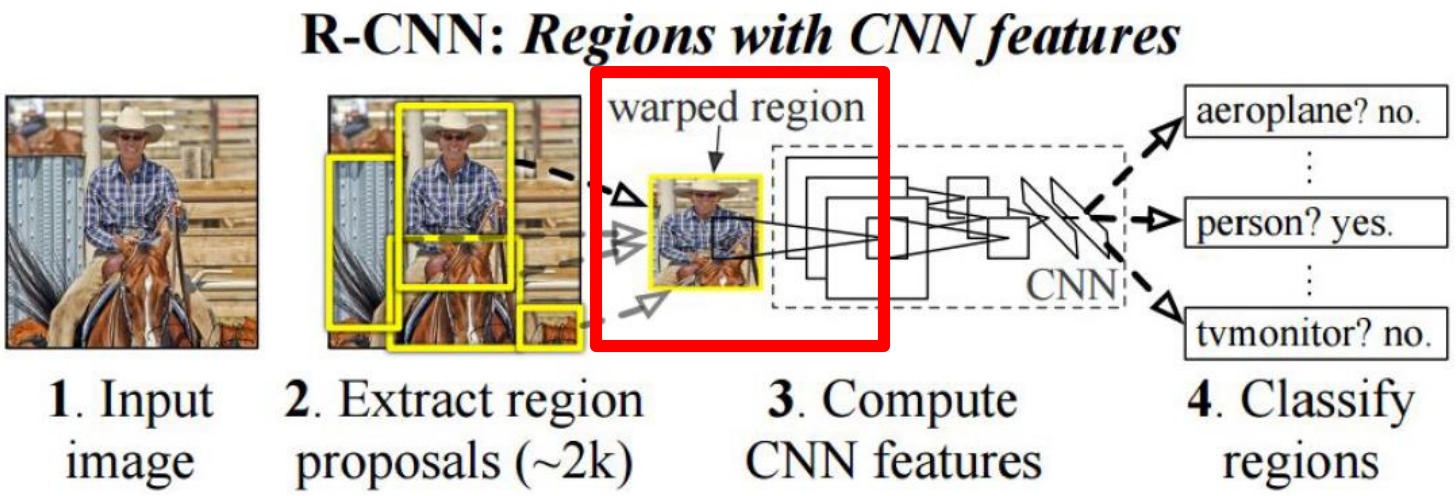
- R-CNN
  - Algorithm procedure
    1) **Region proposal**
       - **Selective search [5]**
         1. Generate initial sub-segmentation.
         2. Recursively combine similar regions into larger ones.
         3. Use the generated regions to produce candidate object locations.



[5] Uijlings, Jasper RR, et al. "Selective search for object recognition." *International journal of computer vision* 104.2 (2013): 154-171.

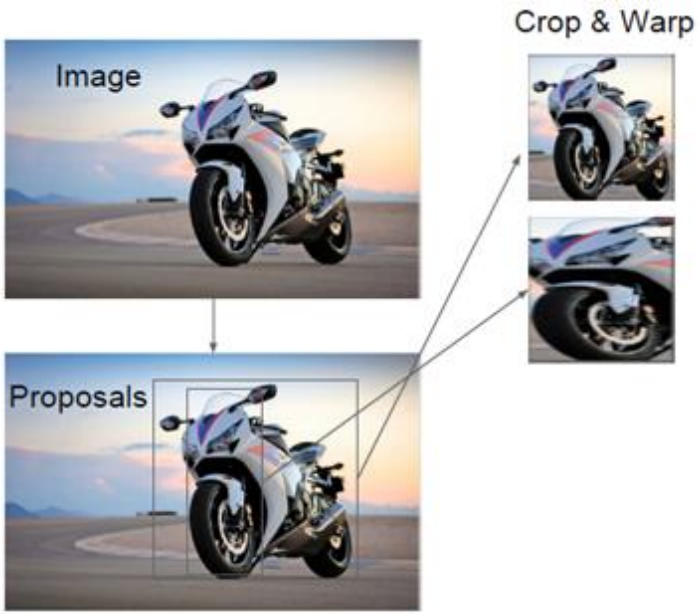
Region-Based Object Detector

- R-CNN
  - Algorithm procedure
    1) Region proposal (selective search)
    2) **Crop & wrap region**
    3) CNN for feature extraction
    4) Classification (SVM) and box regression



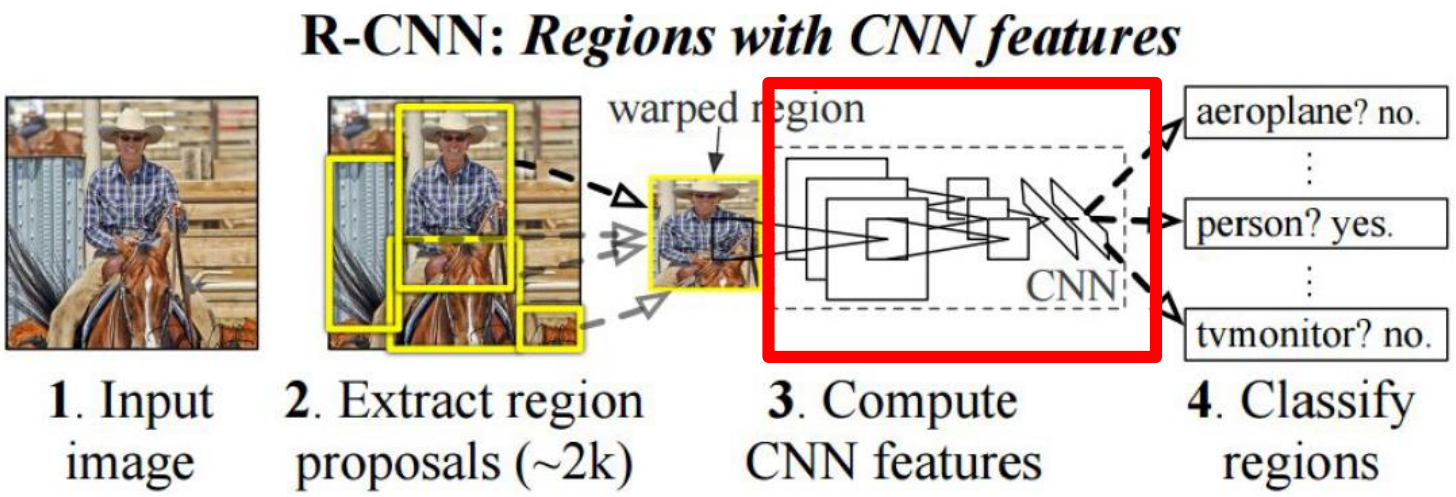
Region-Based Object Detector

- R-CNN
  - Algorithm procedure
    1) Region proposal (selective search)
    2) **Crop & wrap region**
       - Crop regions from input image.
       - Since a CNN takes a fixed-size image, we wrap a cropped region into a 227 x 227 RGB images.



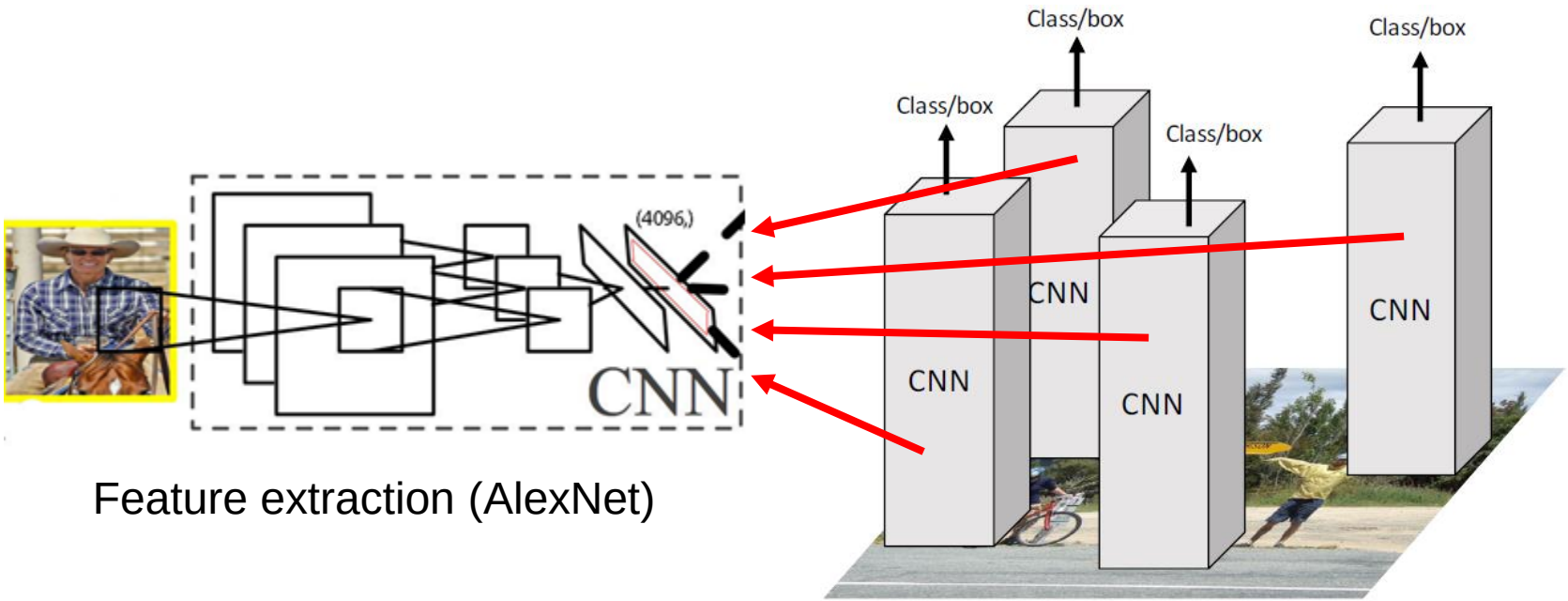
Region-Based Object Detector

- R-CNN
  - Algorithm procedure
    1) Region proposal (selective search)
    2) Crop & wrap region
    3) **CNN for feature extraction**
    4) Classification (SVM) and box regression



Region-Based Object Detector

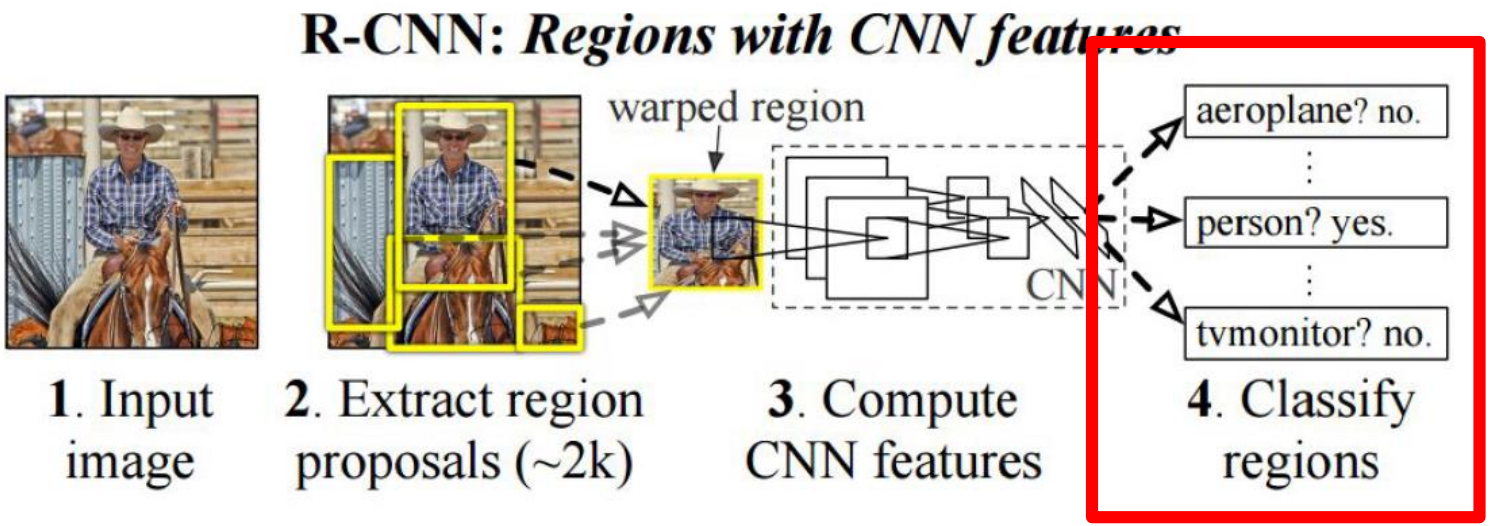
- R-CNN
  - Algorithm procedure
    1) Region proposal (selective search)
    2) Crop & wrap region
    3) **CNN for feature extraction**
       - Use CNN (AlexNet) to extract a 4096-dimensional feature.



Feature extraction (AlexNet)

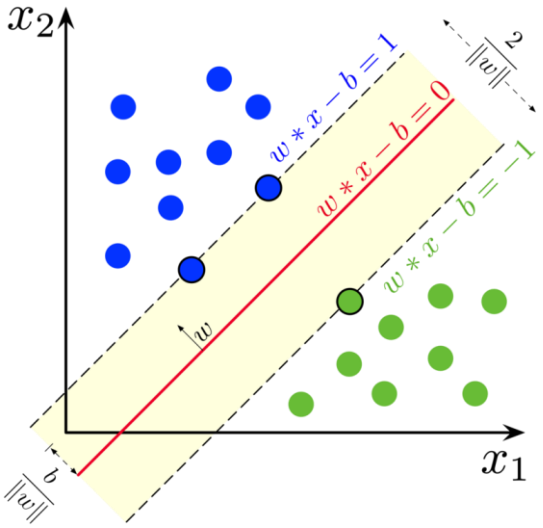
Region-Based Object Detector

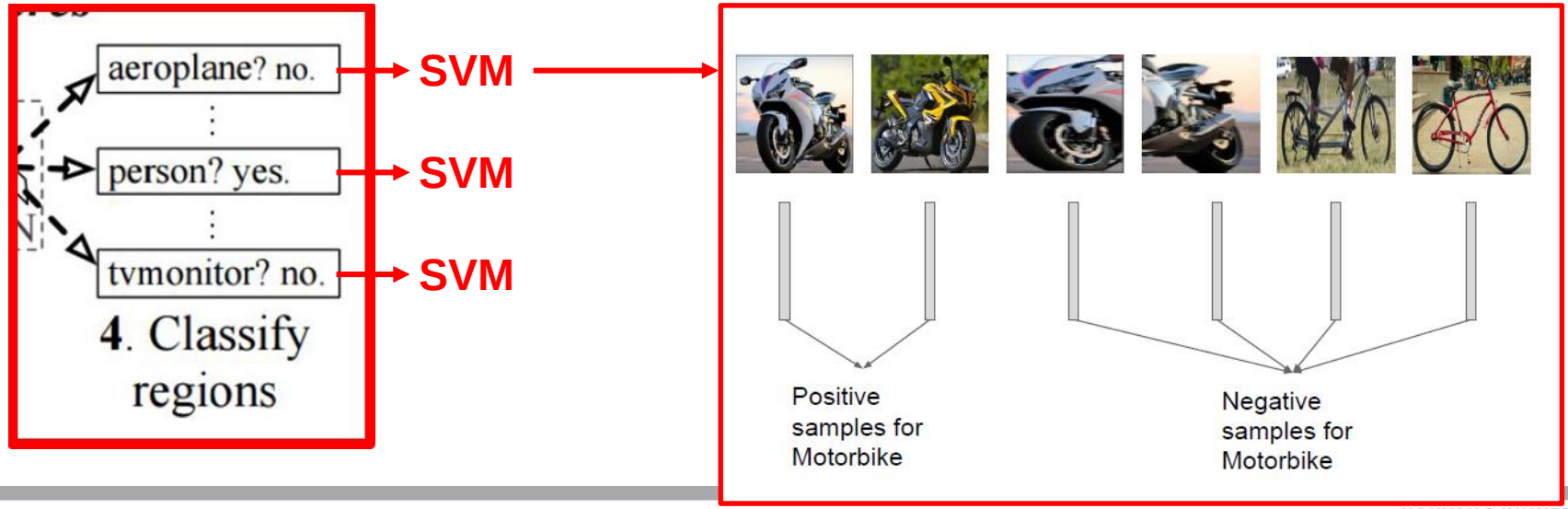
- R-CNN
  - Algorithm procedure
    1) Region proposal (selective search)
    2) Crop & wrap region
    3) CNN for feature extraction
    4) **Classification (SVM)** and box regression



Region-Based Object Detector

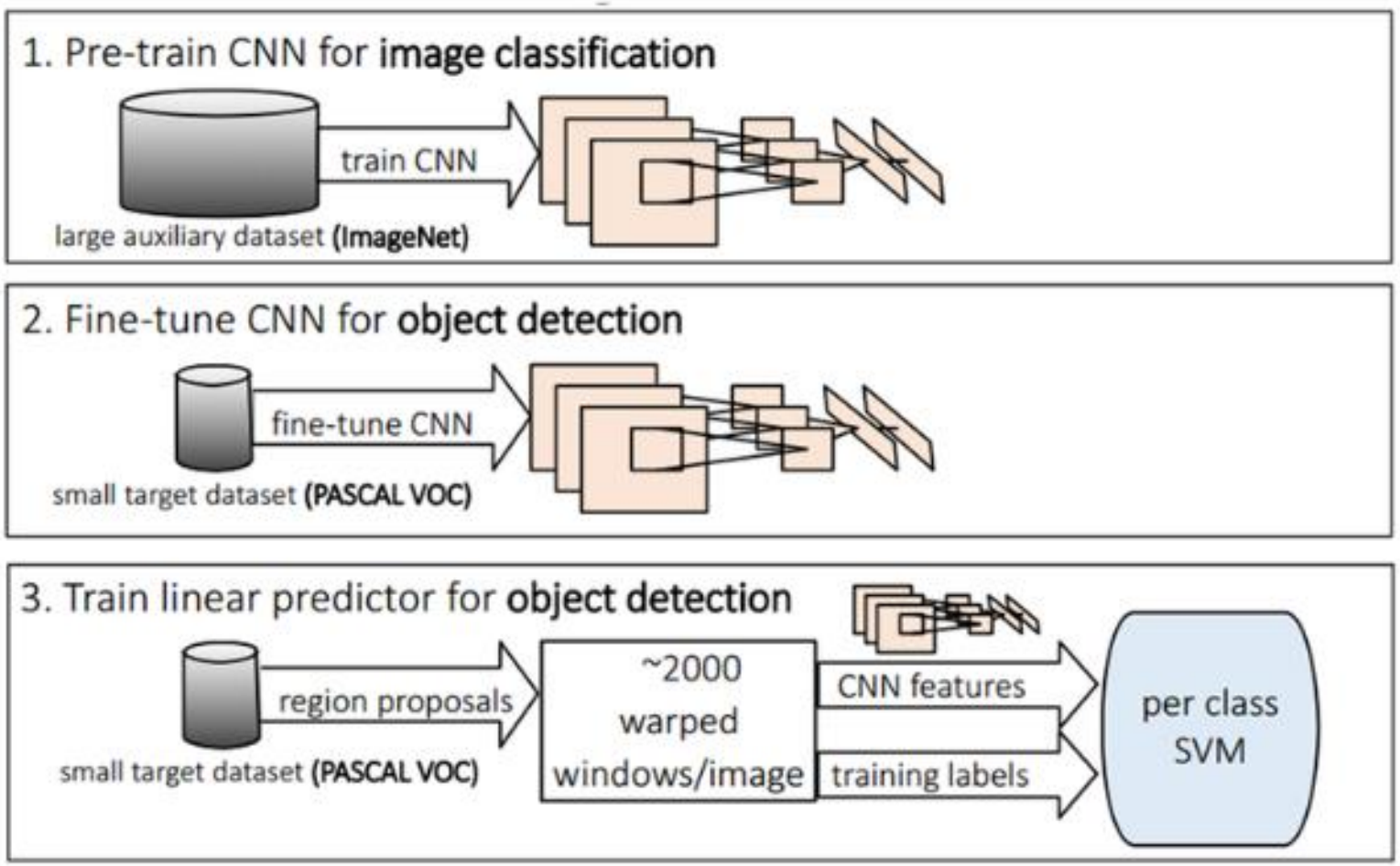
- R-CNN
  - Algorithm procedure
    1) Region proposal (selective search)
    2) Crop & wrap region
    3) CNN for feature extraction
    4) **Classification (SVM)** and box regression
       - **Train SVM for each class**





Region-Based Object Detector

- R-CNN
  - Training procedure (Multi-stage training)

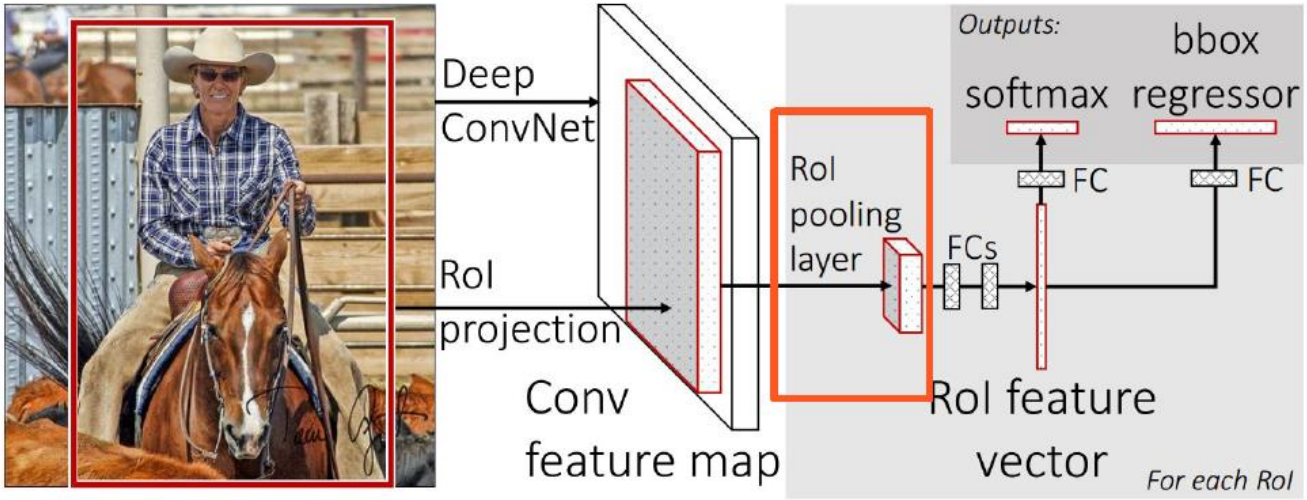


Region-Based Object Detector

- R-CNN
  - Limitations
    - Complex training procedure (multi-stage training)
    - Slow inference time (2k cropped images should be fed into CNN)

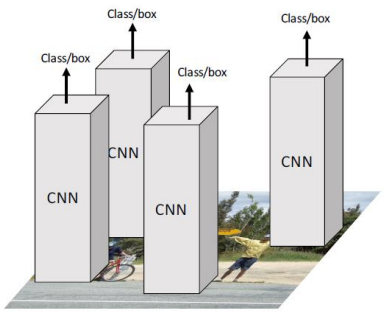
Region-Based Object Detector

- Fast R-CNN
  - The Fast R-CNN builds on R-CNN to efficiently classify object proposals.
    - It is 9x faster in training and 213x faster in testing than R-CNN.
    - It use a **region of interest(ROI) projection** for reduce computation.
    - Bounding box regression is into network and single stage training is used with **multi-task loss**.

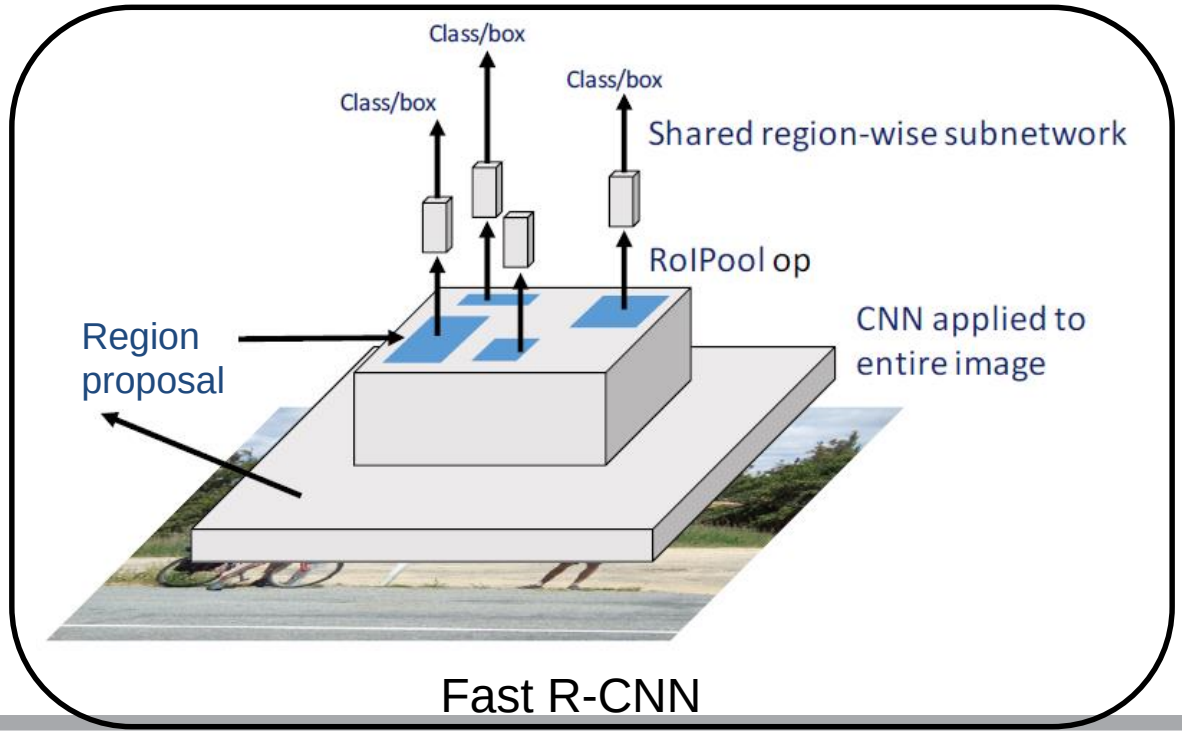


Region-Based Object Detector

- Fast R-CNN
  - **ROI Pooling**
    - R-CNN extract CNN feature vectors independently for each region proposal.
      - High computational cost
    - Fast R-CNN aggregates them into one CNN forward pass over the entire image and the region proposals share this feature matrix.



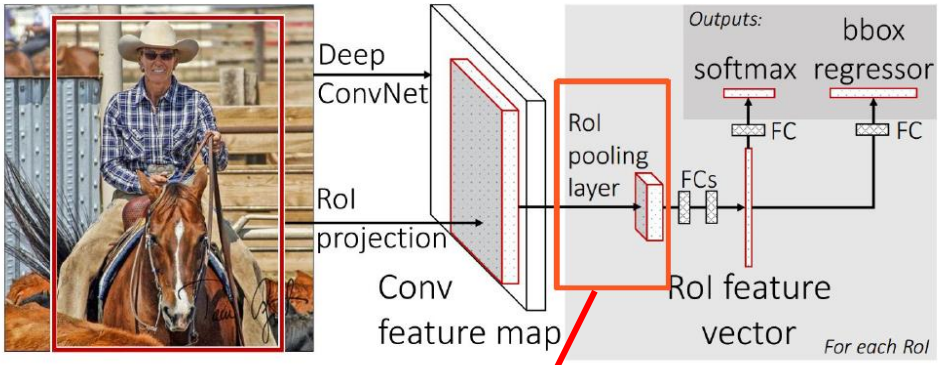
R-CNN



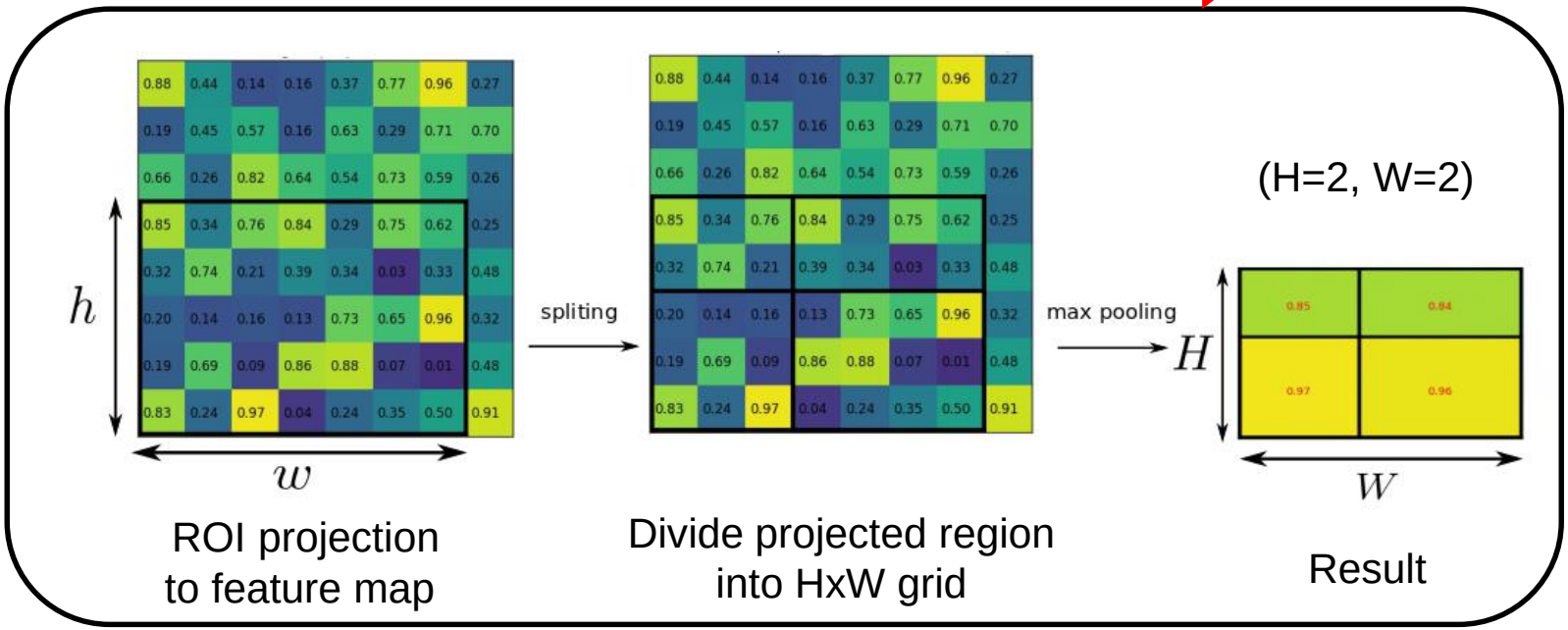
Fast R-CNN

Region-Based Object Detector

- Fast R-CNN
  - **ROI Pooling**
    - Perform max-pooling on non-uniform sizes inputs to obtain fixed-size feature maps. (e.g. 2×2)



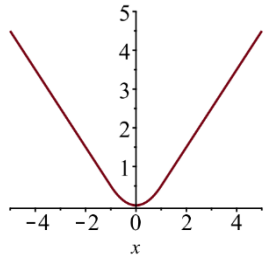
ROI Pooling layer



Region-Based Object Detector

$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases}$$

- Fast R-CNN
  - Bounding box regression with **multi-task loss**



$$L(p, u, t^u, v) = L_{\text{cls}}(p, u) + \lambda[u \geq 1] L_{\text{loc}}(t^u, v)$$

$$-\log p_u$$

Classification loss

$$\sum_{i \in \{\text{x,y,w,h}\}} \text{smooth}_{L_1}(t_i^u - v_i)$$

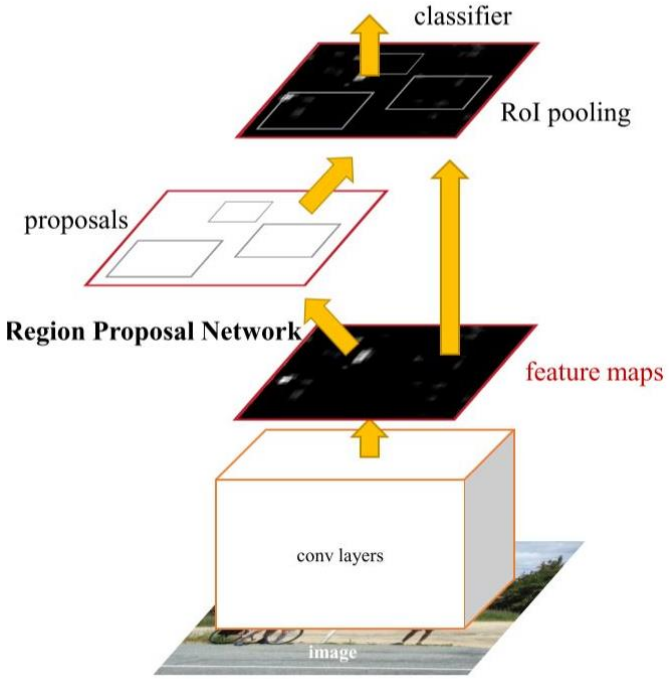
Bounding box regression loss

(The smooth L1 loss is adopted here and it is claimed to be less sensitive to outliers.)

Symbol	Explanation
u	True class label, $u \in 0, 1, \ldots, K$; by convention, the catch-all background class has $u = 0$.
p	Discrete probability distribution (per RoI) over K + 1 classes: $p = (p_0, \ldots, p_K)$, computed by a softmax over the K + 1 outputs of a fully connected layer.
v	True bounding box $v = (v_x, v_y, v_w, v_h)$.
t^u	Predicted bounding box correction, $t^u = (t_x^u, t_y^u, t_w^u, t_h^u)$.

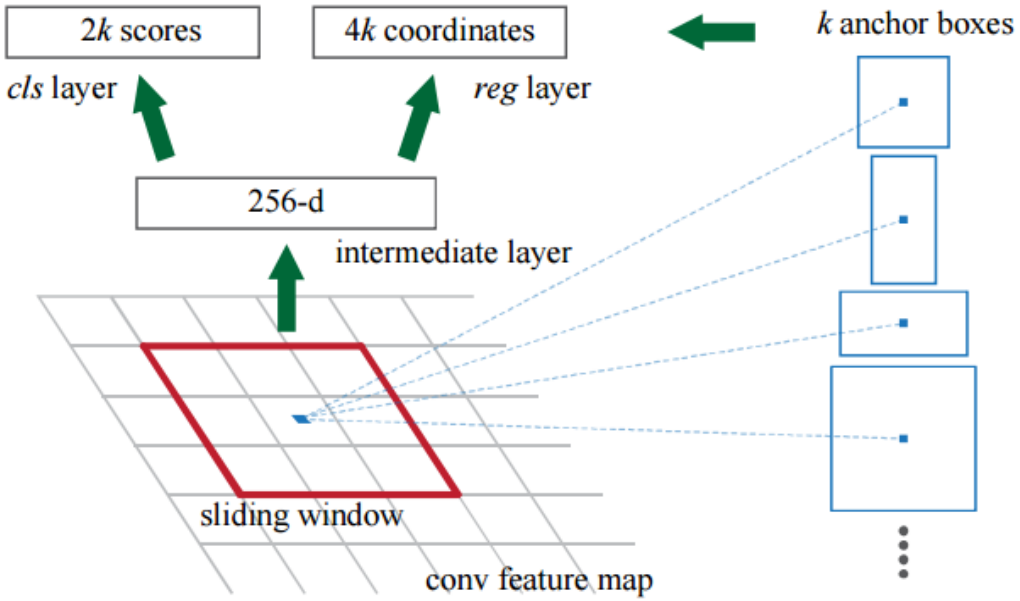
Region-Based Object Detector

- Faster R-CNN
  - End-to-end training
  - The Faster-RCNN is composed of two modules.
    1. **Region proposal network (RPN)**.
    2. Fast R-CNN detector.
  - RPN is trained to produce region proposals directly.
  - After RPN, use Roi pooling and an upstream classifier and bounding box regressor like Fast R-CNN.



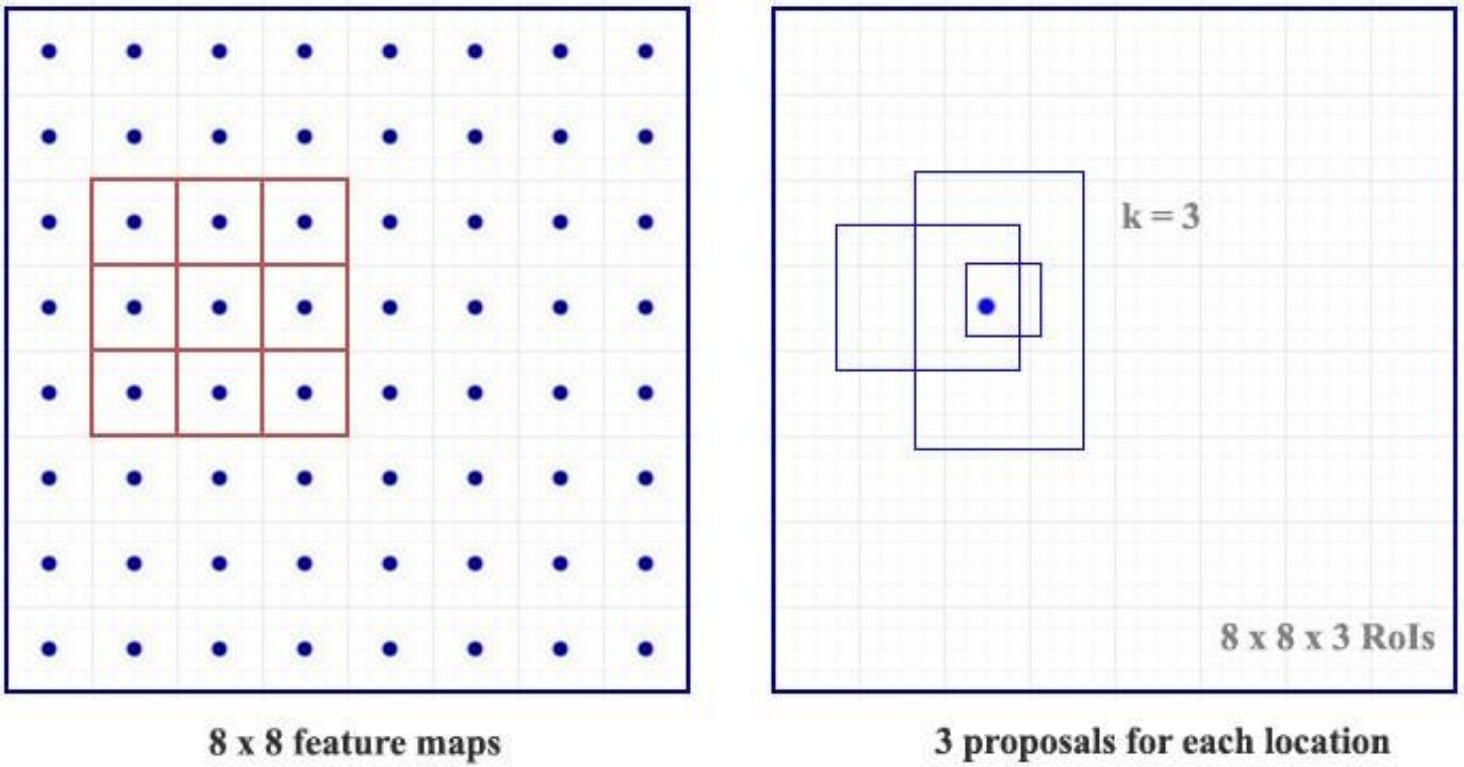
Region-Based Object Detector

- Faster R-CNN
  - Region proposal network (RPN)
    - The sliding window suggests whether or not there is an object in the area and the coordinate value.
    - Multi-Scale anchors are used as regression references.



Region-Based Object Detector

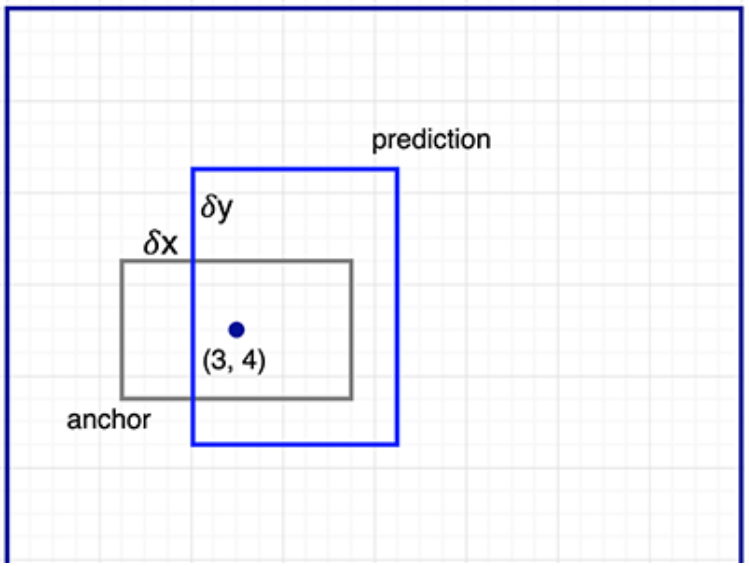
- Faster R-CNN
  - **Region proposal network (RPN)**
    - For each location in the feature maps, RPN makes k guesses.
    - Therefore RPN outputs 4×k coordinates and 2×k scores per location.



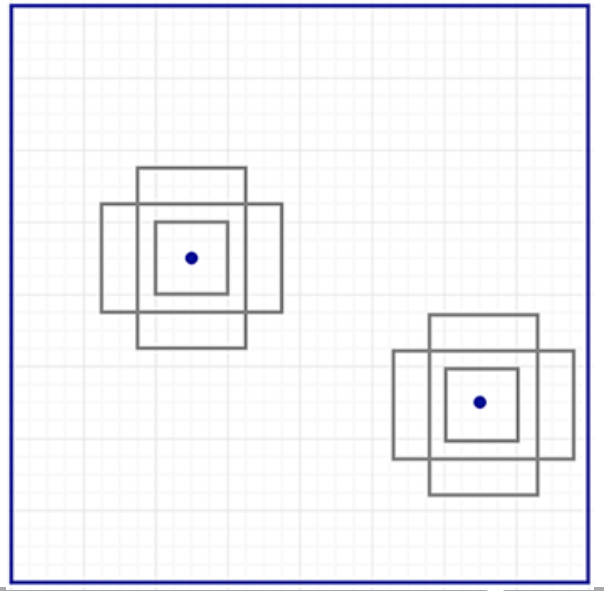
Region-Based Object Detector

- Faster R-CNN
  - **Region proposal network (RPN)**
    - Since we just need one to be correct, we will be better off if our initial guesses have default shapes and size.
    - Instead, it predicts offsets like δx, δy that are relative to the top left corner of some reference boxes called **anchors**.

Example of offset prediction with anchor boxes



Example of *3* anchor boxes



Region-Based Object Detector

p^* : Ground truth label
p : Predicted score
t^* : GT bbox
t : Predicted bbox offset

- Faster R-CNN
  - **Region proposal network (RPN)**
    - Training strategy for RPN
      - Assign a binary class label (of being an object or not) to each anchor.
      - Assign a positive label to two kinds of anchors:
        - » The anchors with the highest IoU (Intersection-over-union) with a ground-truth box.
        - » An anchor that has an IoU overlap higher than 0.7 with any ground-truth box.
    - Assign a negative label to a non-positive anchor if its IoU ratio is lower than 0.3 for all ground-truth boxes.

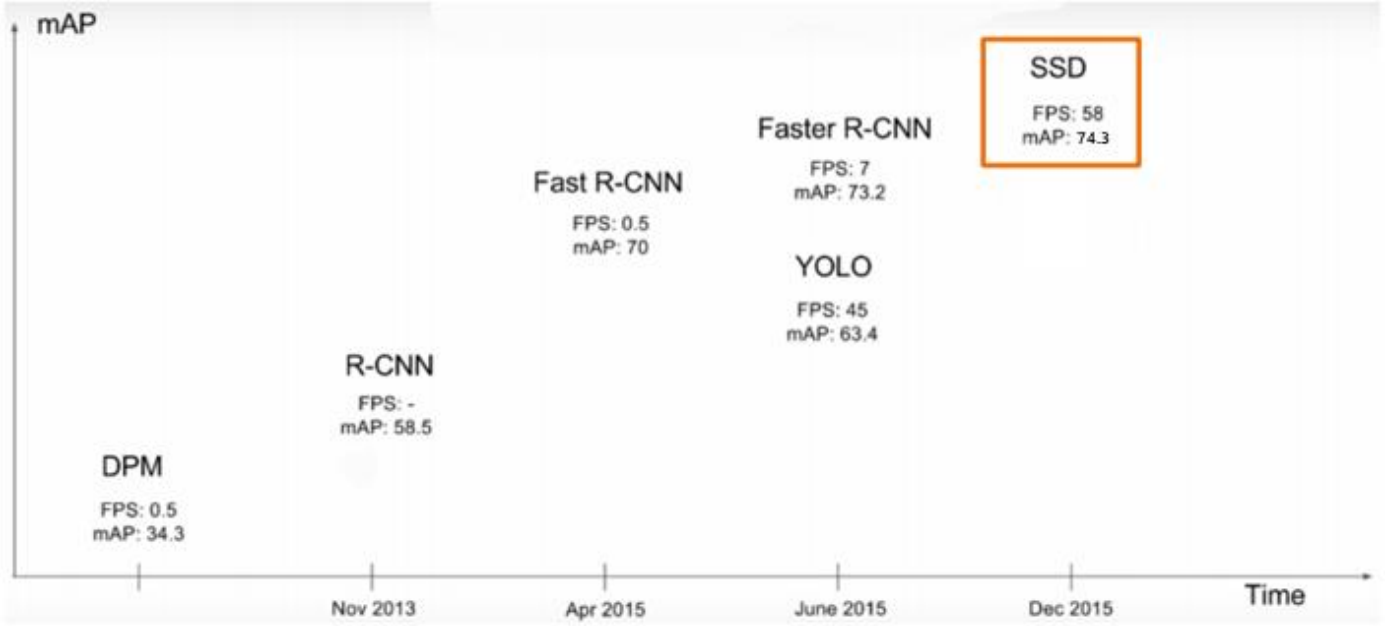
$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i, p_i^*) + \lambda\frac{1}{N_{reg}}\sum_i p_i^* L_{reg}(t_i, t_i^*).$$

Region-Based Object Detector

- Faster R-CNN
  - Limitations of two-stage detector
    - Still slow inference time
      - Operates at only 7 frames per second (FPS)

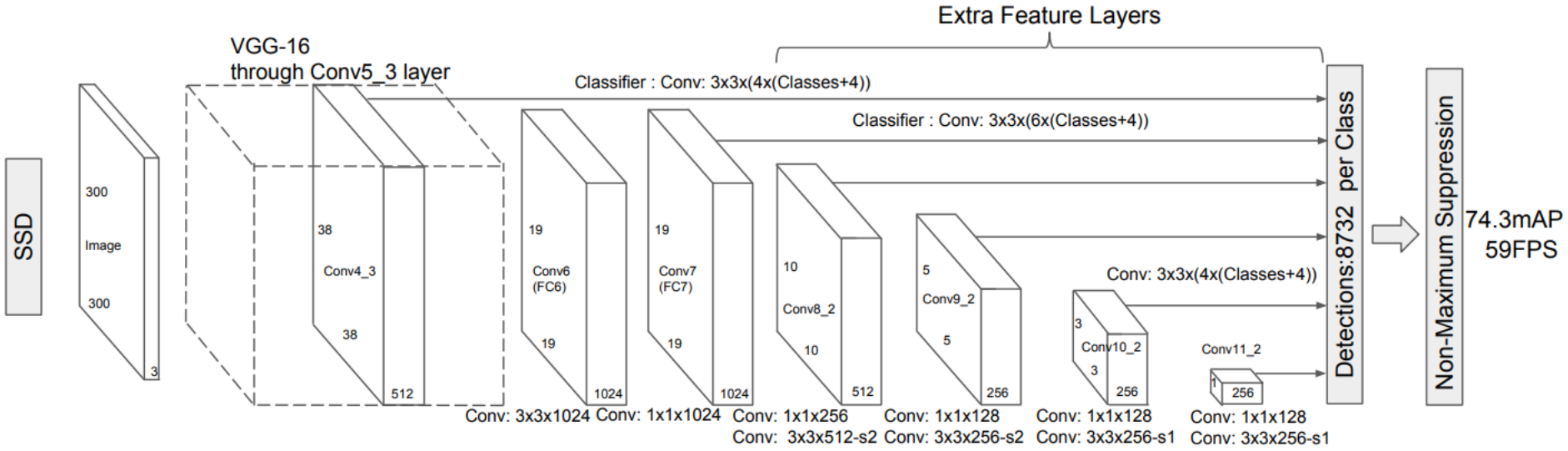
Single Shot Object Detector

- SSD: Single Shot multibox Detector
  - Single stage detector
    - Predict object categories and offsets in bounding box locations at once.
  - Achieves significant improvement in speed for high-accuracy detection.
    - SSD : 58 FPS with mAP 74.3% on VOC2007 test
      - Faster R-CNN 7 FPS with mAP 73.2%
      - YOLO 45 FPS with mAP 63.4%



Single Shot Object Detector

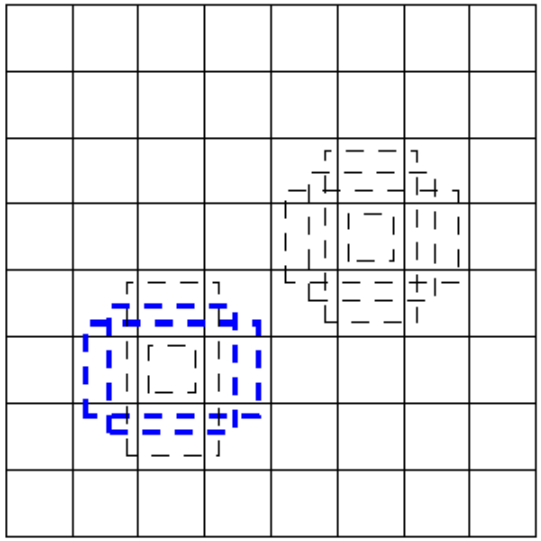
- SSD: Single Shot multibox Detector
  - Structural features
    - The core of SSD is predicting category scores and box offsets for a fixed set of **default bounding boxes (= anchor boxes)** using **small convolutional filters** applied to feature maps.
    - Makes predictions on **multiple feature maps** with different resolutions to handle objects of different sizes.



Single Shot Object Detector

- SSD: Single Shot multibox Detector
  - Structural features
    - **Default bounding boxes (= anchor boxes)**
      - Similar to the anchors of Faster R-CNN, with the difference that SSD applies them on several feature maps of different resolutions.

Example of *4* default boxes



(b) 8×8 feature map

Single Shot Object Detector

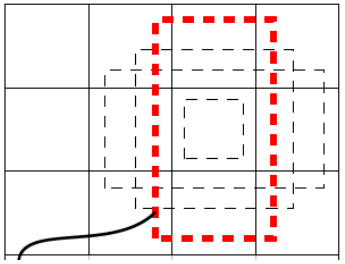
- SSD: Single Shot multibox Detector
  - Structural features
    - **Default bounding boxes (= anchor boxes)**
      - Choosing scales and aspect ratios for default boxes;
        - » If m feature maps are used for detection, the scale of the default boxes for kth feature map is;

$$s_k = s_{\min} + \frac{s_{\max} - s_{\min}}{m - 1}(k - 1), \quad k \in [1, m]$$

$s_{\min} = 0.1$
$s_{\max} = 0.9$

      - At each scale, different aspect ratios are considered;

Example of default box



$$a_r \in \{1, 2, 3, \tfrac{1}{2}, \tfrac{1}{3}\}$$

$$(w_k^a = s_k\sqrt{a_r})$$

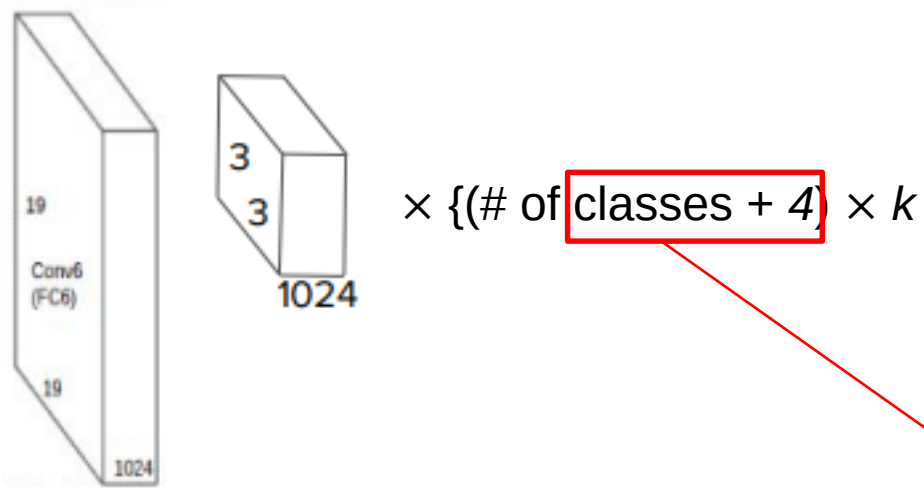
$$(h_k^a = s_k/\sqrt{a_r})$$

w_k^a : width of default box
h_k^a : height of default box

Single Shot Object Detector

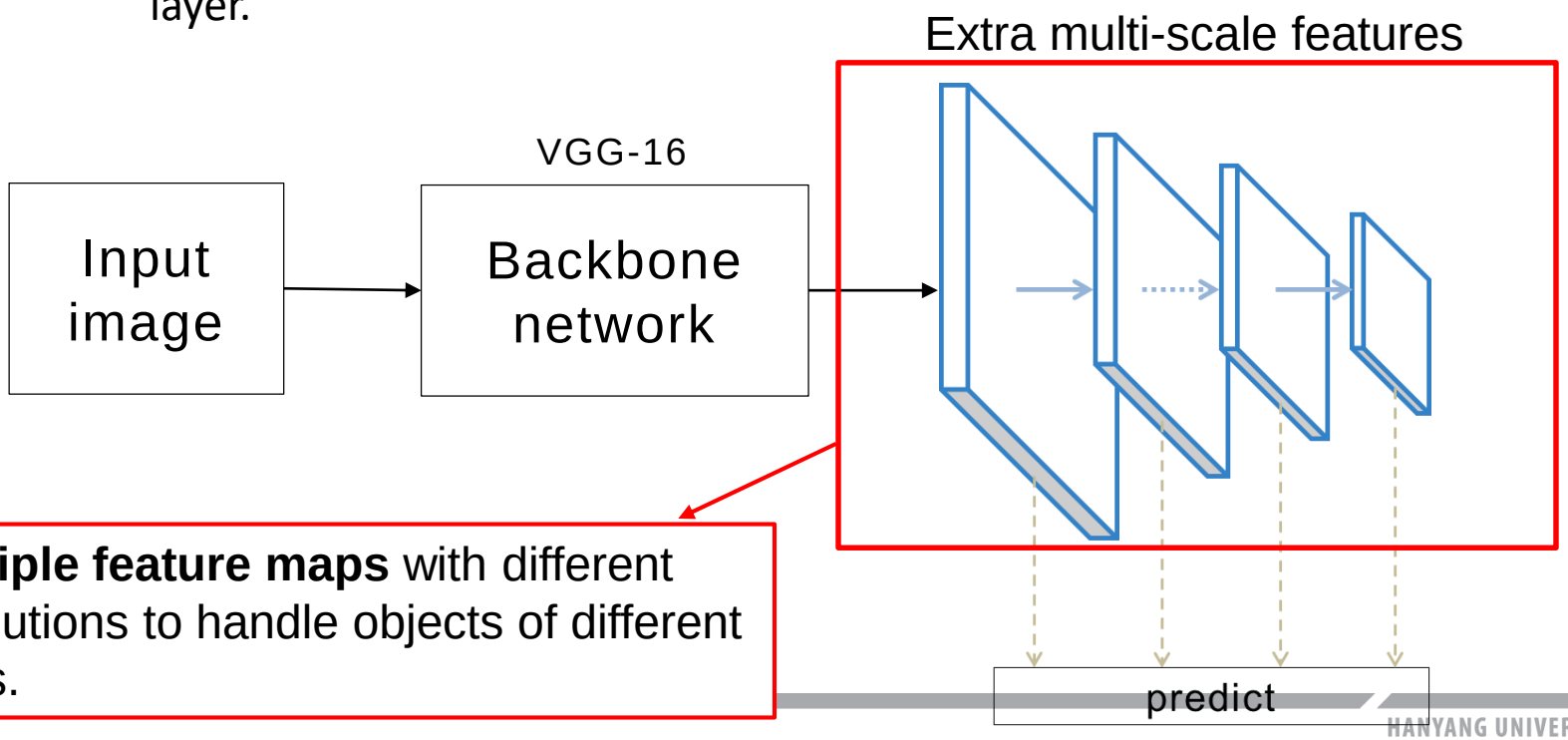
- SSD: Single Shot multibox Detector
  - Structural features
    - **Small convolutional filters for detection**
      - For each feature layer of m x n x p, we apply kernels of $3 \times 3 \times p \times \{(\#\text{ of classes} + 4) \times k\}$ to produce either a score for each class and *4* offset relative to a *k* default bounding box coordinates.

Example for conv6:



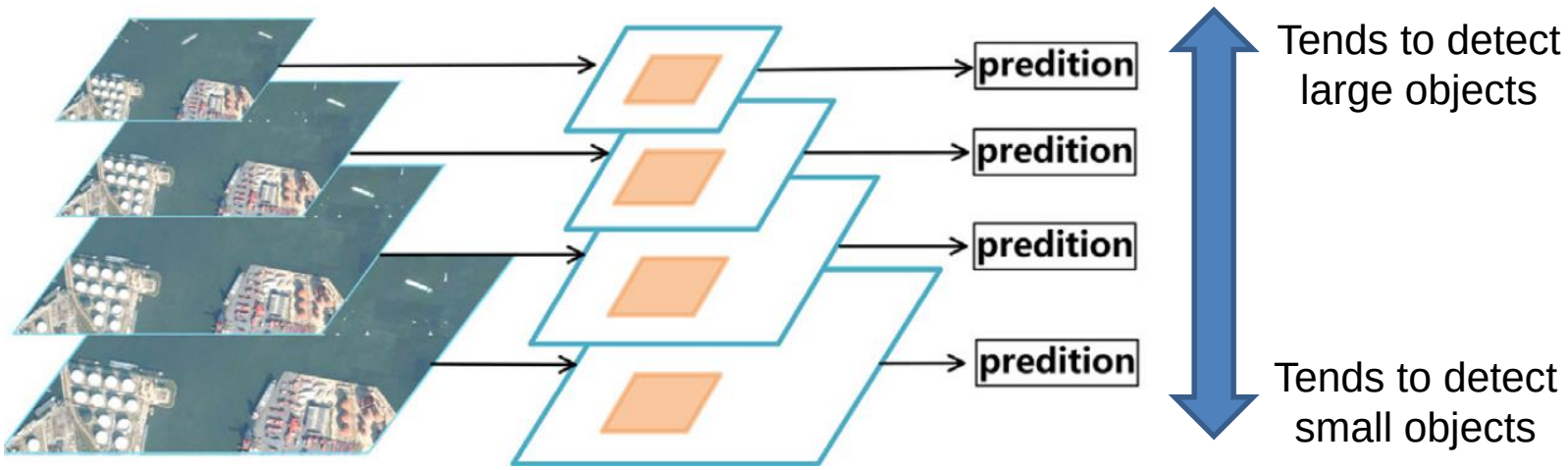
Single Shot Object Detector

- SSD: Single Shot multibox Detector
  - Structural features
    - **Multiple feature maps**
      - Add convolutional layers at the end of backbone network (VGG-16) and decrease the size of the features progressively.
      - Concatenate the detection results from each extra features at the last layer.



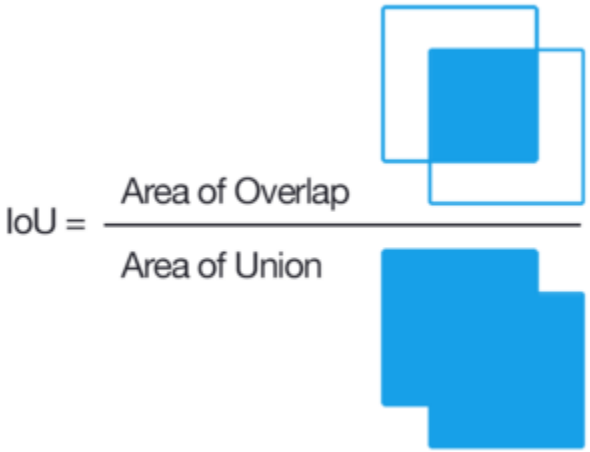
Single Shot Object Detector

- SSD: Single Shot multibox Detector
  - Structural features
    - **Multiple feature maps**
      - Add convolutional layers at the end of backbone network (VGG-16) and decrease the size of the features progressively.
      - Concatenate the detection results from each extra features at the last layer.



Single Shot Object Detector

- SSD: Single Shot multibox Detector
  - Training strategy
    - Matching strategy
      - For each ground truth box, we have to select from all the default boxes the ones that best fit in terms of location, aspect ratio and scale.
        - » We select the default box with best jaccard overlap (=ioU; intersection of union). Then every box has at least 1 correspondence.
        - » Default boxes with a jaccard overlap higher than 0.5 are also selected.



Single Shot Object Detector

- SSD: Single Shot multibox Detector
  - Training strategy
    - Loss function

$$L(x, c, l, g) = \frac{1}{N}\left(L_{conf}(x, c) + \alpha L_{loc}(x, l, g)\right)$$

Classification loss

$$-\sum_{i \in Pos}^{N} x_{ij}^{p} log(\hat{c}_i^p) - \sum_{i \in Neg} log(\hat{c}_i^0)$$

where $\hat{c}_i^p = \frac{\exp(c_i^p)}{\sum_p \exp(c_i^p)}$

c : class confidence (score)

Bounding box regression loss

(The smooth L1 loss is adopted.)

$$L_{loc}(x, l, g) = \sum_{i \in Pos}^{N} \sum_{m \in \{cx, cy, w, h\}} x_{ij}^{k} \text{smooth}_{\text{L1}}(l_i^m - \hat{g}_j^m)$$

$$\hat{g}_j^{cx} = (g_j^{cx} - d_i^{cx})/d_i^w \qquad \hat{g}_j^{cy} = (g_j^{cy} - d_i^{cy})/d_i^h$$

$$\hat{g}_j^{w} = \log\left(\frac{g_j^w}{d_i^w}\right) \qquad \hat{g}_j^{h} = \log\left(\frac{g_j^h}{d_i^h}\right)$$

g : Ground truth box
l : Predicted box
d : Default box

Single Shot Object Detector

- SSD: Single Shot multibox Detector
  - Training strategy
    - **Hard negative mining**
      - To handle significant imbalance between positive and negative training examples.
        - » Use negative samples with higher confidence score.
        - » Then the ratio of positive-negative samples is 3:1.
    - **Data augmentation**
      - To make the model more robust to various input object sizes and shapes, each training image is randomly sampled by one of the following options:
        - » Use the entire original input image.
        - » Sample a patch so that the minimum jaccard overlap with the objects is 0.1, 0.3, 0.5, 0.7, or 0.9.
        - » Randomly sample a patch.

Single Shot Object Detector

- SSD: Single Shot multibox Detector
  - Results
    - Effect of multiple features

Prediction source layers from:						mAP	# Boxes
conv4_3	conv7	conv8_2	conv9_2	conv10_2	conv11_2		
✔	✔	✔	✔	✔	✔	74.3	8732
✔	✔	✔	✔	✔		**74.6**	8764
✔	✔	✔	✔			73.8	8942
✔	✔	✔				70.7	9864
✔	✔					64.2	9025
	✔					62.4	8664

    - Effect of data augmentation & various scales of default boxes

	SSD300				
more data augmentation?		✔	✔	✔	✔
include $\{\frac{1}{2}, 2\}$ box?	✔		✔	✔	✔
include $\{\frac{1}{3}, 3\}$ box?	✔			✔	✔
use atrous?	✔	✔	✔		✔
VOC2007 `test` mAP	65.5	71.6	73.7	74.2	**74.3**

Results

- Results
  - Detection & speed performance
    - PASCAL VOC 2007 test set

SSD300 : 300x300 images as an input
SSD300 : 512x512 images as an input

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

    - COCO test-dev 2015

Method	data	Avg. Precision, IoU:			Avg. Precision, Area:			Avg. Recall, #Dets:			Avg. Recall, Area:		
		0.5:0.95	0.5	0.75	S	M	L	1	10	100	S	M	L
Fast [6]	train	19.7	35.9	-	-	-	-	-	-	-	-	-	-
Fast [24]	train	20.5	39.9	19.4	4.1	20.0	35.8	21.3	29.5	30.1	7.3	32.1	52.0
Faster [2]	trainval	21.9	42.7	-	-	-	-	-	-	-	-	-	-
ION [24]	train	23.6	43.2	23.6	6.4	24.1	38.3	23.2	32.7	33.5	10.1	37.7	53.6
Faster [25]	trainval	24.2	45.3	23.5	7.7	26.4	37.1	23.8	34.0	34.6	12.0	38.5	54.4
SSD300	trainval35k	23.2	41.2	23.4	5.3	23.2	39.6	22.5	33.2	35.3	9.6	37.6	56.5
SSD512	trainval35k	**26.8**	**46.5**	**27.8**	**9.0**	**28.9**	**41.9**	**24.8**	**37.5**	**39.8**	**14.0**	**43.5**	**59.0**