Recognizing faces - 2

Image pyramid and neural networks

- With the neural networks, a classifier may be trained directly using preprocessed and normalized face and nonface training subwindows.
- Rowley et al (http://www.cs.cmu.edu/~har/) use the preprocessed 20x20 subwindows as the input to a neural network. The final decision is made to classify the 20x20 subwindow into face and nonface.

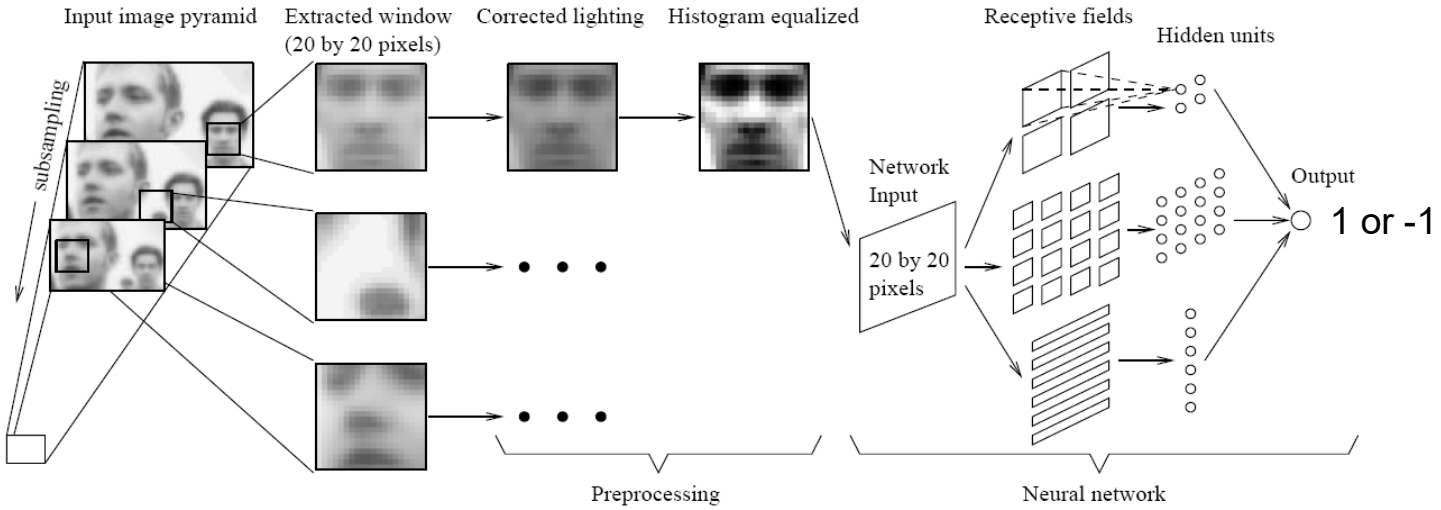


Figure 1: The basic algorithm used for face detection.

There are three types of hidden units: four which look at 10 x 10-pixel subregions, 16 which look at 5 x 5-pixel subregions, and six which look at overlapping 20 x 5-pixel horizontal stripes of pixels.

[Li-Jain-05, Rowley-Baluja-Kanade-98]

Image pyramid and neural networks

Original windows

Lighting corrected windows

Histogram equalized windows

Figure 4: Example face images (the authors), randomly mirrored, rotated, translated, and scaled by small amounts.

[Rowley-Baluja-Kanade-98]

Image pyramid and neural networks

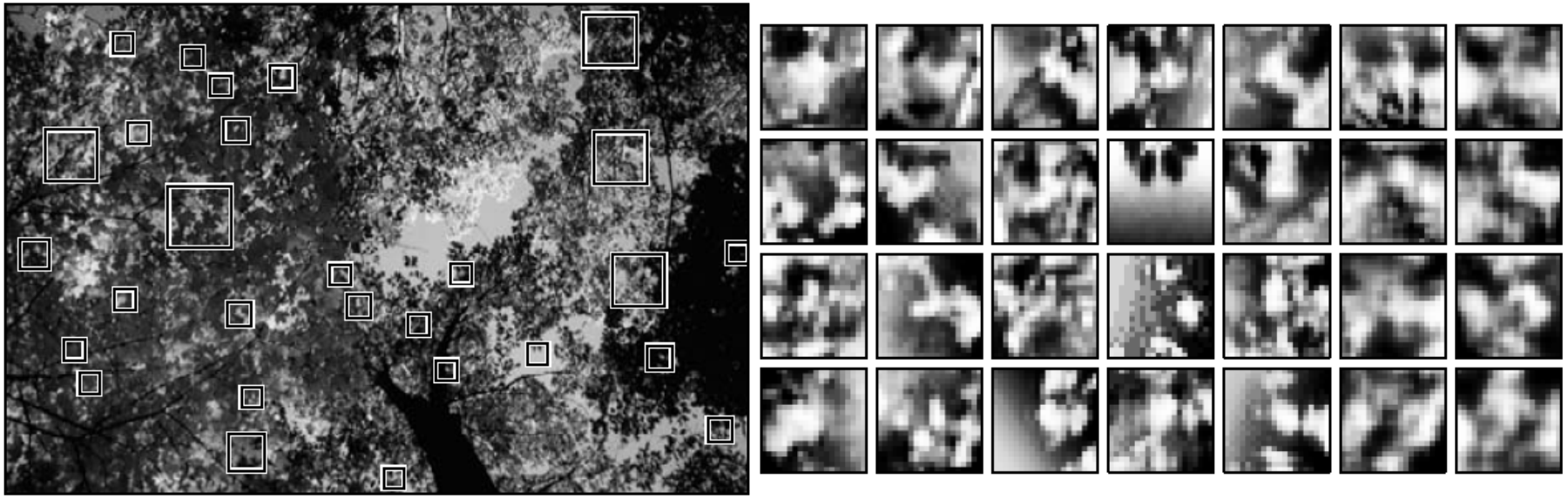


Figure 5: During training, the partially-trained system is applied to images of scenery which do not contain faces (like the one on the left). Any regions in the image detected as faces (which are expanded and shown on the right) are errors, which can be added into the set of negative training examples.

[Rowley-Baluja-Kanade-98]

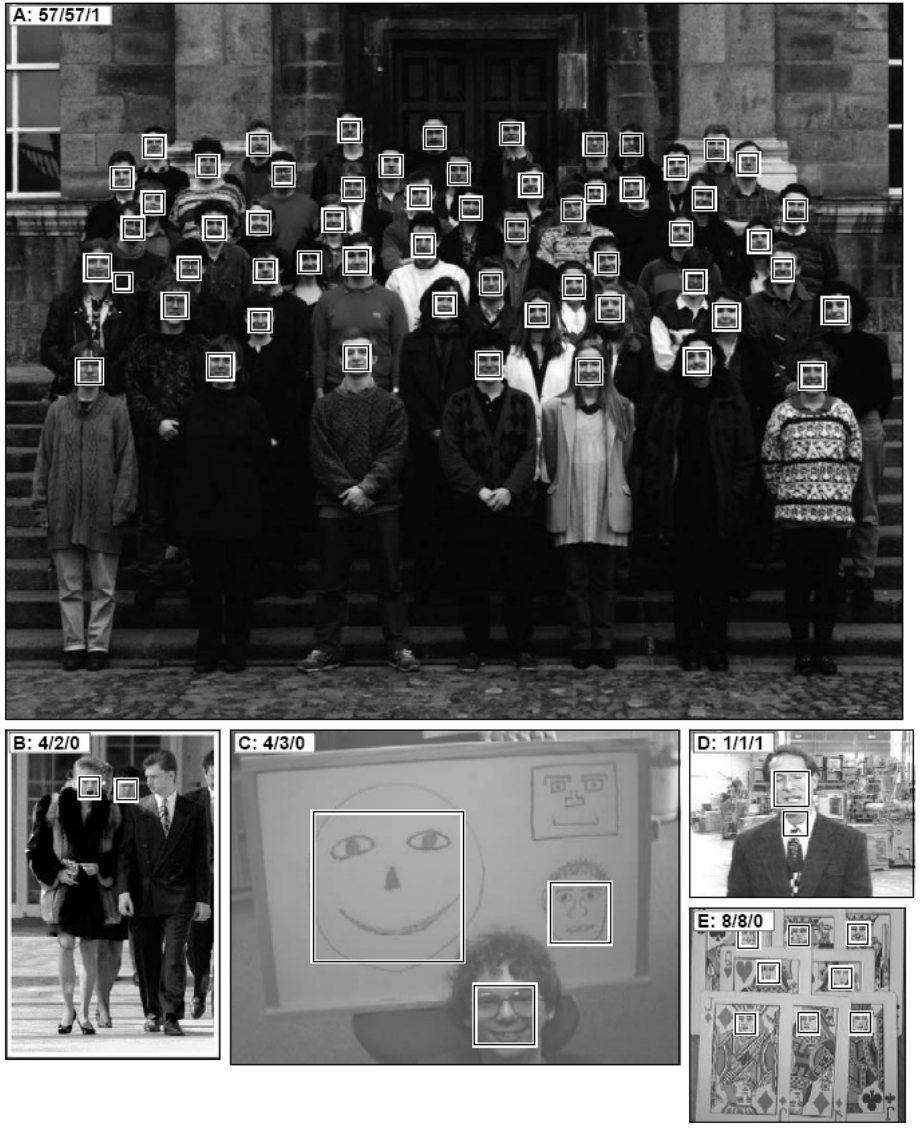


Figure 11: Output obtained from System 11 in Table 1 on images from Test Set 1. For each image, three numbers are shown: the number of faces in the image, the number of faces detected correctly, and the number of false detections. Some notes on specific images: Faces are missed in B (one due to occlusion, one due to large angle) and C (the stylized drawing was not detected at the same locations and scales by the two networks, and so is lost in the AND). False detections are present in A and D. Although the system was trained only on real faces, some hand drawn faces are detected in C and E. A was obtained from the World Wide Web, B and E were provided by Sung and Poggio at MIT, C is a CCD image, and D is a digitized television image.

[Rowley-Baluja-Kanade-98]

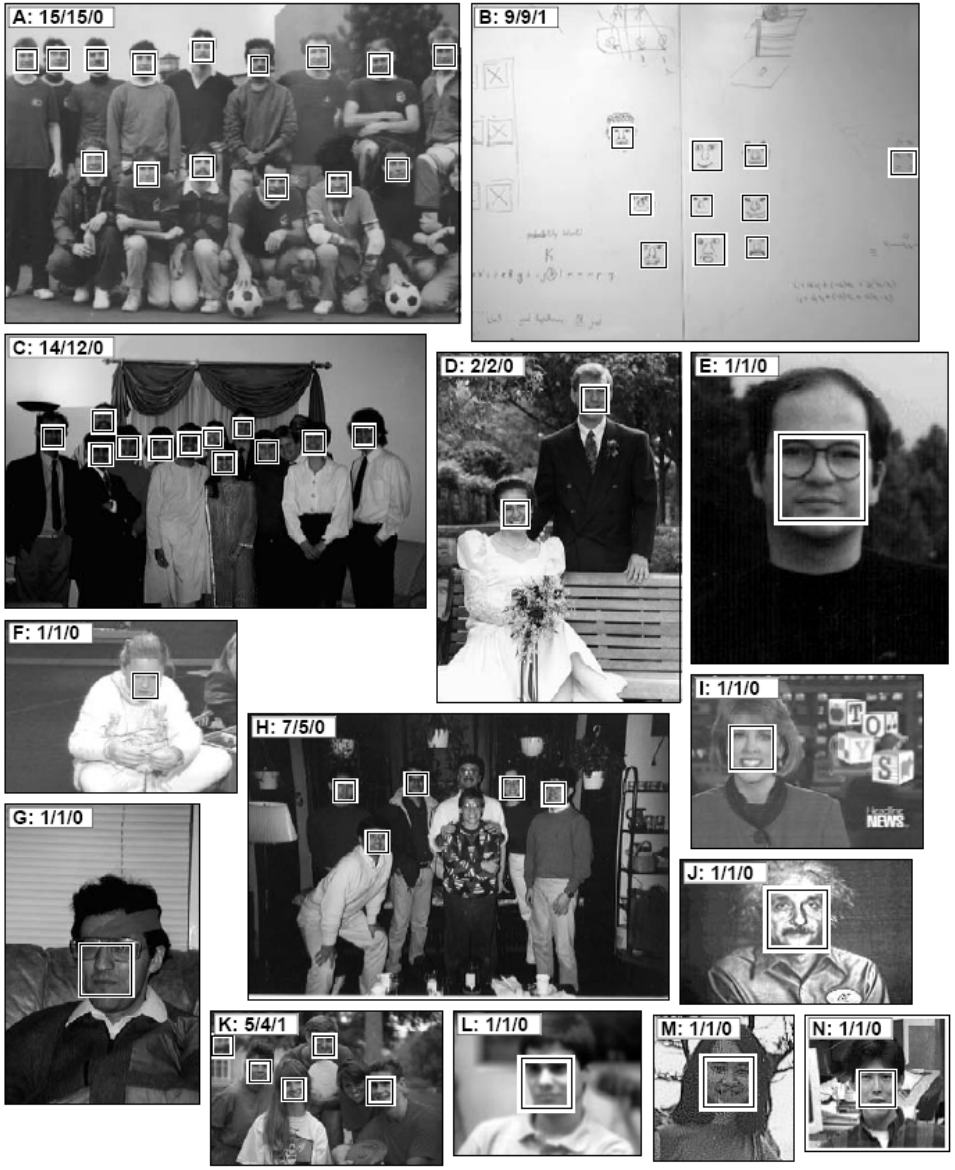


Figure 12: Output obtained in the same manner as the examples in Fig. 11. Some notes on specific images: Faces are missed in C (one due to occlusion, one due to large angle), H (reflections off of glasses made the eyes appear brighter than the rest of the face), and K (due to large angle). False detections are present in B and K. Although the system was trained only on real faces, hand drawn faces are detected in B. A, B, J, K, and L were provided by Sung and Poggio at MIT, C, D, E, G, H, and M were scanned from photographs, F and I are digitized television images, and N is a CCD image.

[Rowley-Baluja-Kanade-98]

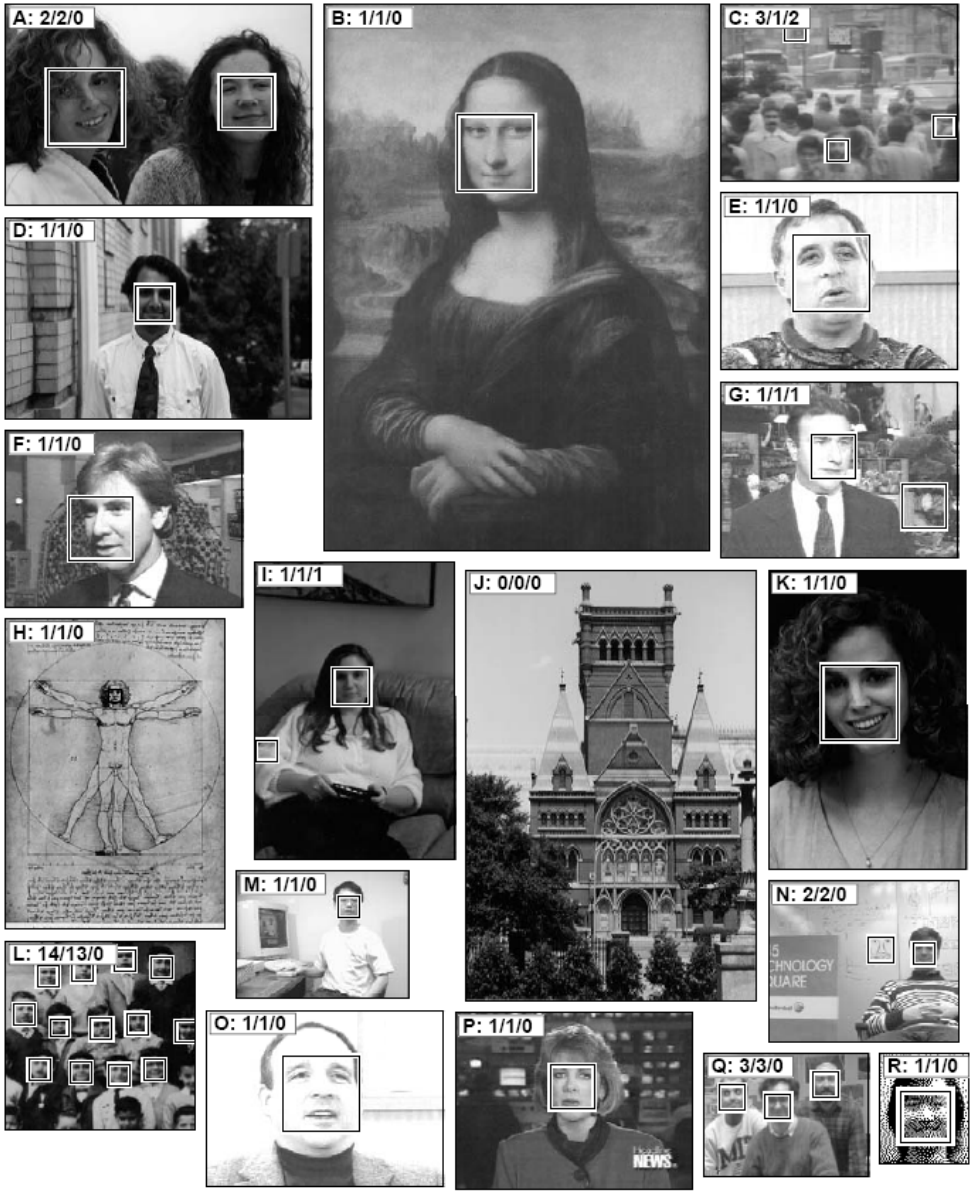


Figure 13: Output obtained in the same manner as the examples in Fig. 11. Some notes on specific images: Faces are missed in C (due to blurriness) and L (due to partial occlusion of the chin). False detections are present in C, G, and I. Although the system was trained only on real faces, hand drawn faces are detected in H and N. A, D, I, J, and K were scanned from photographs, B, H, and L were obtained from the World Wide Web, C, E, F, G, O, and P are digitized television images. M, N, and Q were provided by Sung and Poggio at MIT, and R is a dithered CCD image.

[Rowley-Baluja-Kanade-98]

Rotated faces

- Instead of upright, frontal faces, a router network can be trained to process each input window so that orientation can be estimated.
- Once the orientation is estimated, the input window can be prepared for detector neural network.

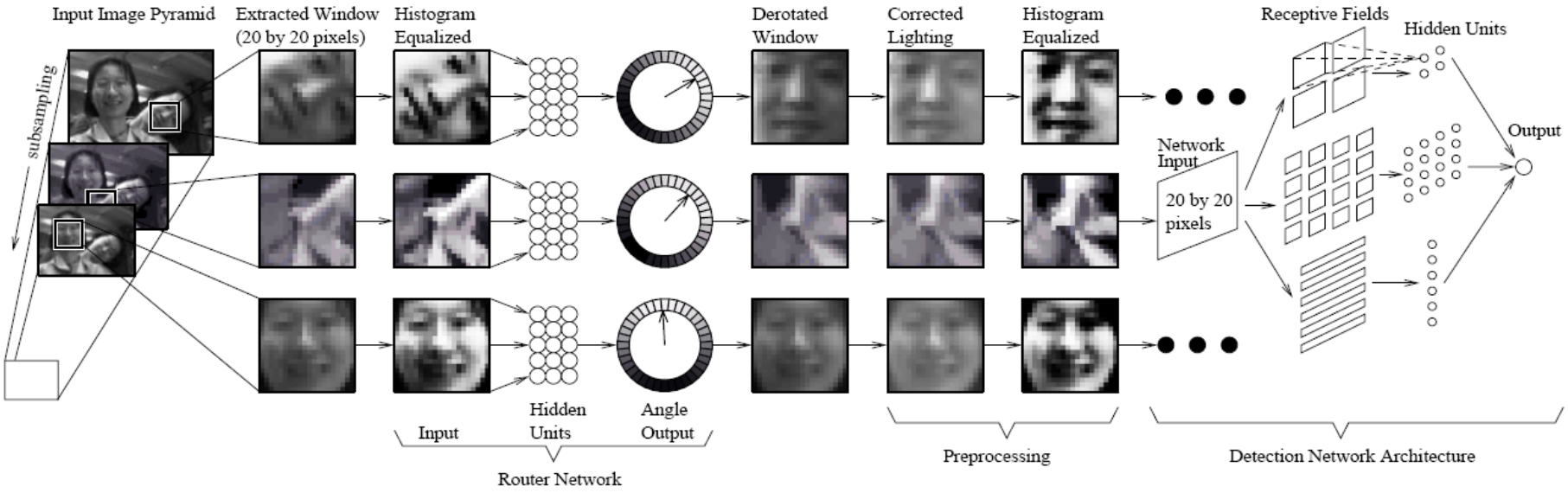


Figure 2. Overview of the algorithm.

[Rowley-Baluja-Kanade-98cvpr]

Rotated faces

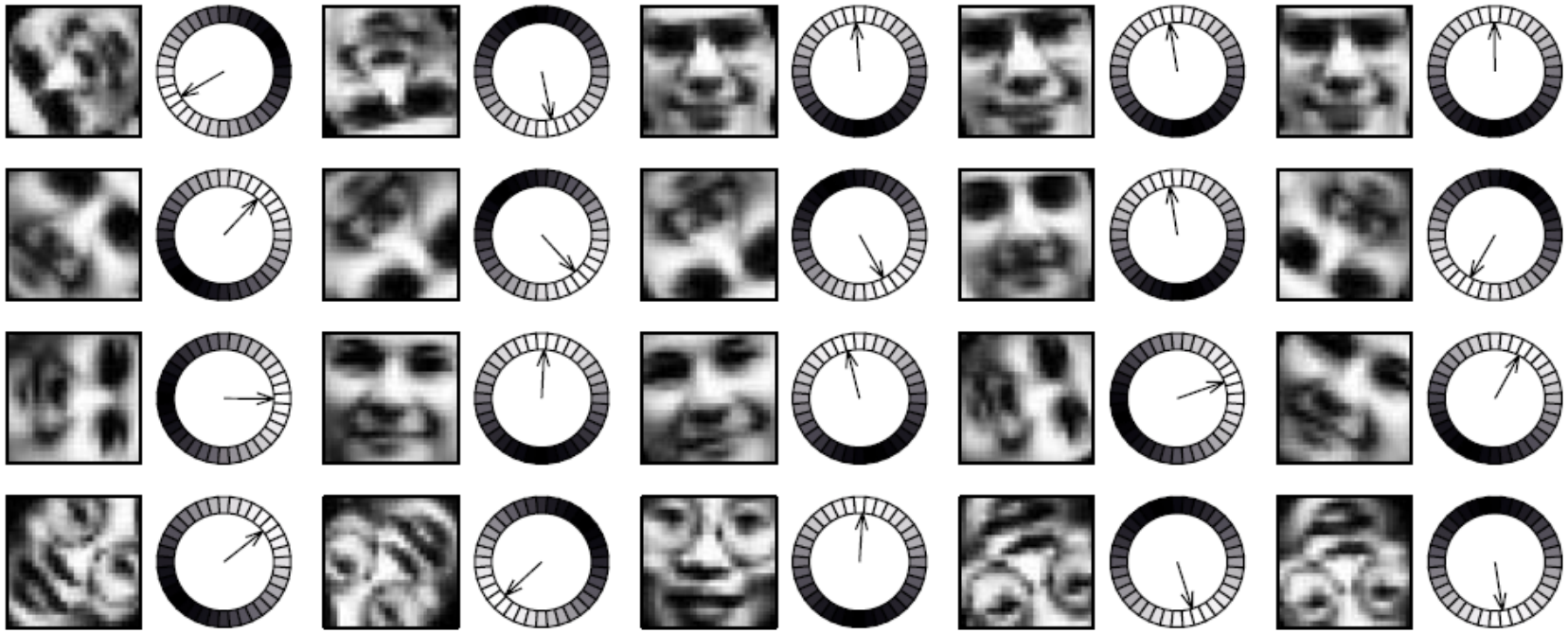


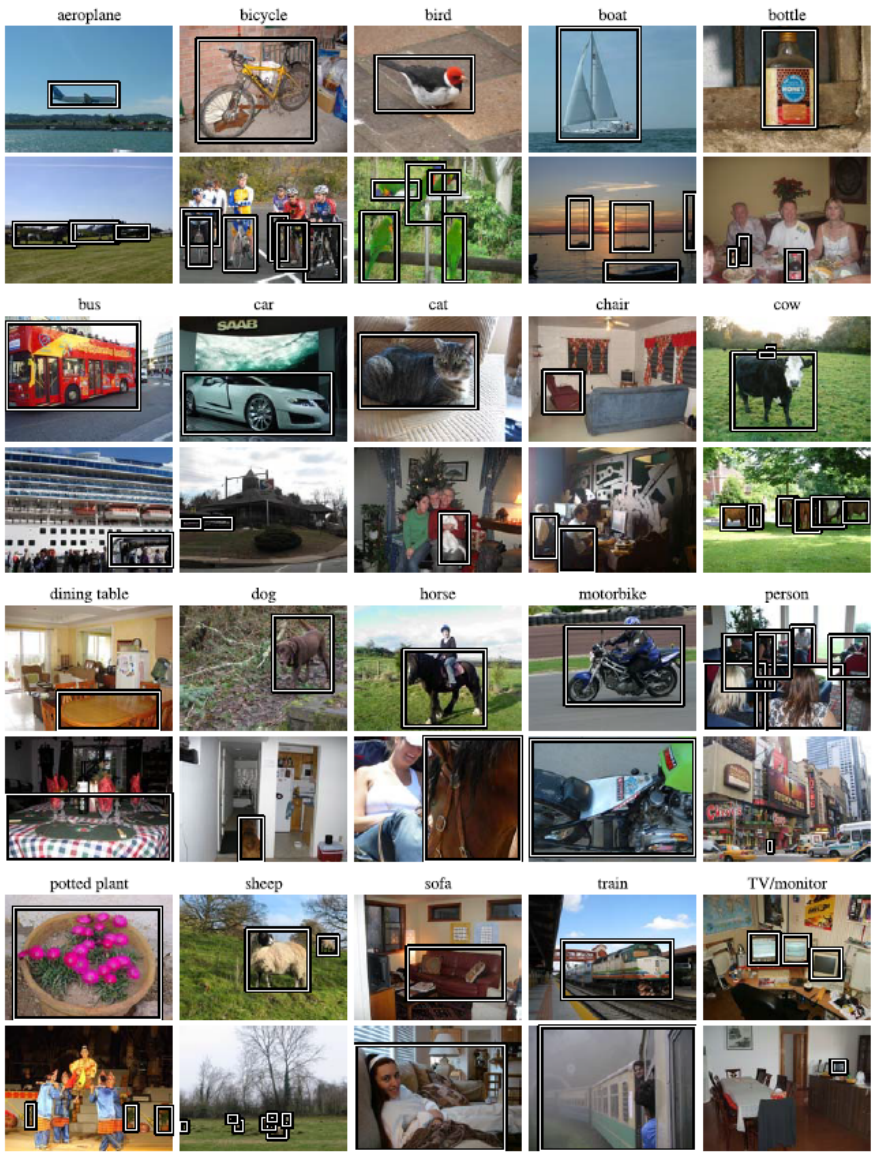
Figure 3. Example inputs and outputs for training the router network.

[Rowley-Baluja-Kanade-98cvpr]

[Rowley-Baluja-Kanade-98cvpr]

Object Detection

Example images

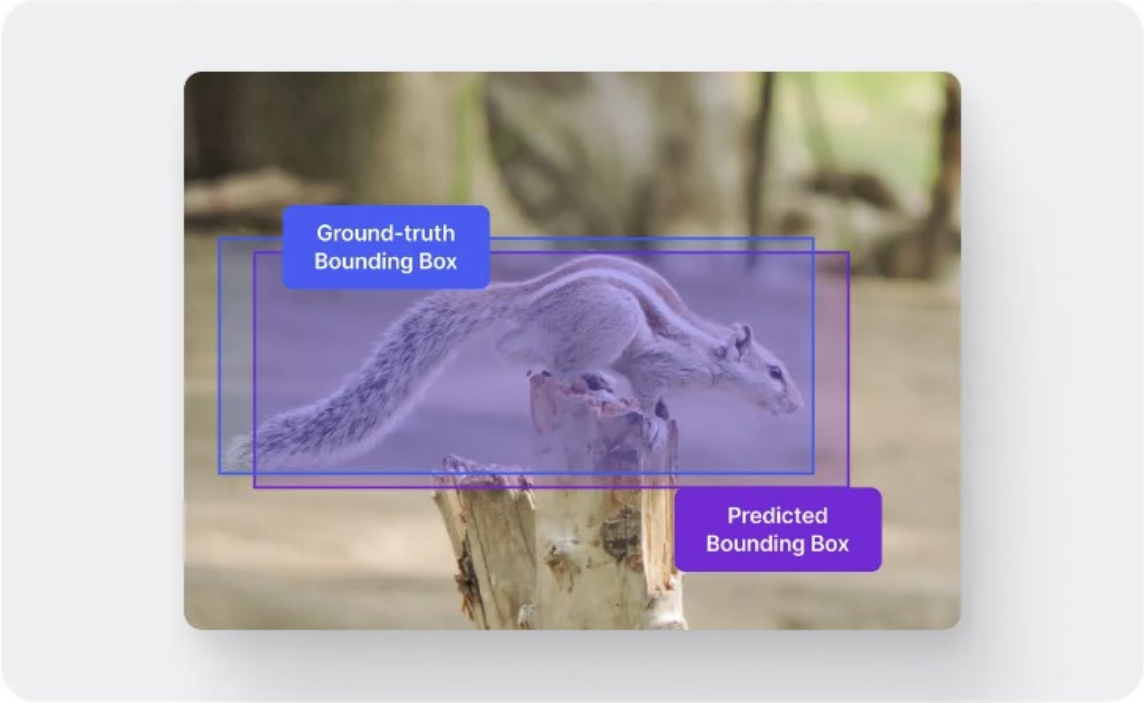


input = image.
output: rectangle (bounding box) + (label)
multiply

Fig. 1 Example images from the VOC2007 dataset. For each of the 20 classes annotated, two examples are shown. *Bounding boxes* indicate all instances of the corresponding class in the image which are marked as "non-difficult" (see Sect. 3.3)—bounding boxes for the other classes are available in the annotation but not shown. Note the wide range of pose, scale, clutter, occlusion and imaging conditions

Intersection over Union (IoU)

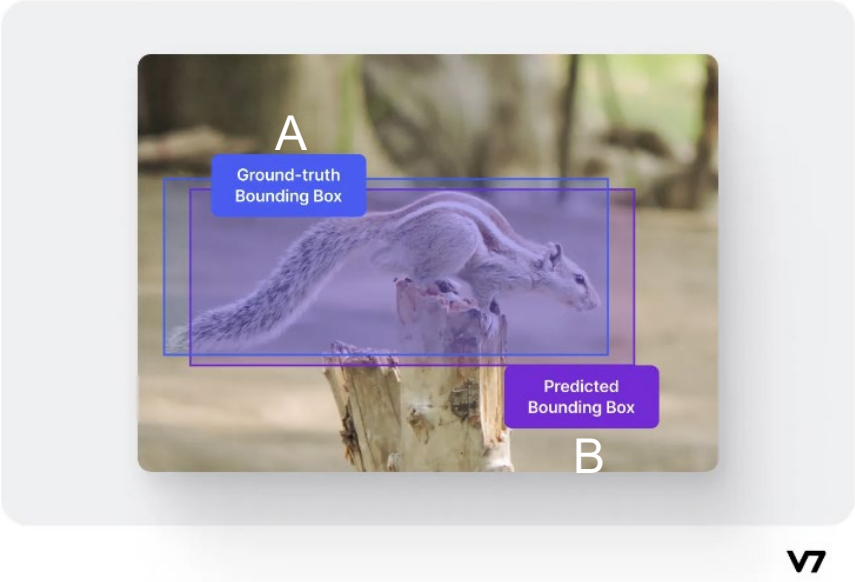
- IoU quantifies how well the model can distinguish objects from their backgrounds in an image. It measures the localization/prediction accuracy.



Source: https://www.v7labs.com/blog/intersection-over-union-guide

Intersection over Union (IoU)

- IoU calculates the union of the ground truth bounding box (A) and the predicted bounding box (B) as the denominator.
- IoU computes the intersection of the ground truth bounding box (A) and the predicted bounding box (B) as the numerator.



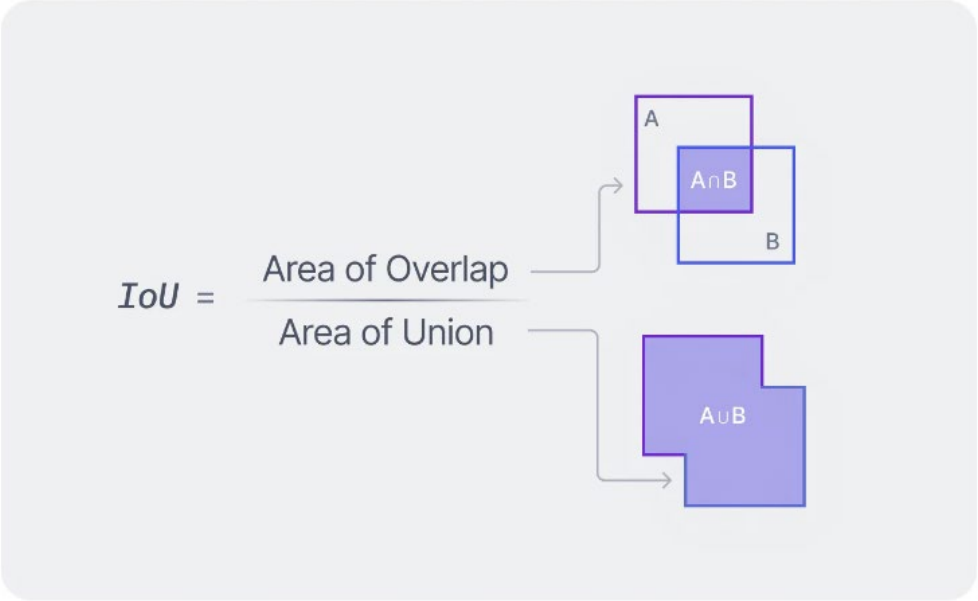
$$\text{IoU} = \frac{|A \cap B|}{|A| \cup |B|}$$

$$\text{IoU} = \frac{TP}{TP + FN + FP}$$

Source: https://www.v7labs.com/blog/intersection-over-union-guide

Intersection over Union (IoU)

- IoU calculates the union of the ground truth bounding box (A) and the predicted bounding box (B) as the denominator.
- IoU computes the intersection of the ground truth bounding box (A) and the predicted bounding box (B) as the numerator.



[0, 1]

$$\text{IoU} = \frac{|A \cap B|}{|A| \cup |B|}$$

$$\text{IoU} = \frac{TP}{TP + FN + FP}$$

Source: https://www.v7labs.com/blog/intersection-over-union-guide

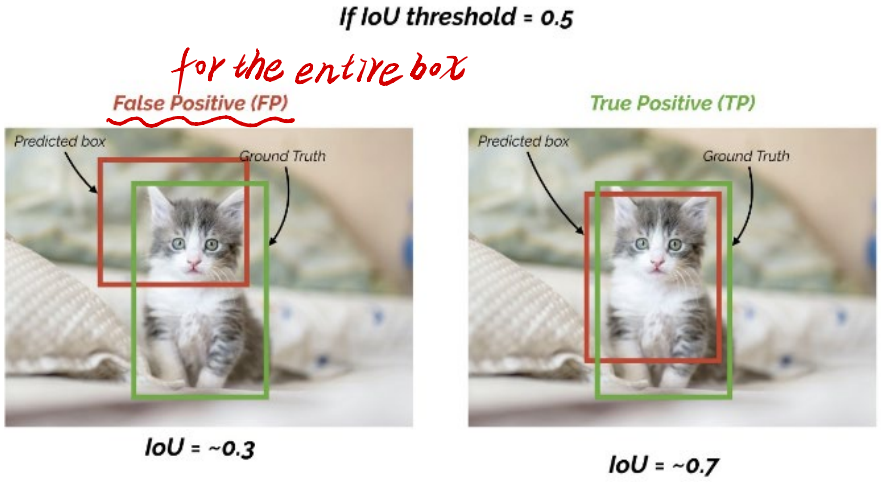
Intersection over Union (IoU)

- IoU comparative performance

Source: https://www.v7labs.com/blog/intersection-over-union-guide

Intersection over Union (IoU)

- IoU comparative performance



Calculating IoU threshold

?

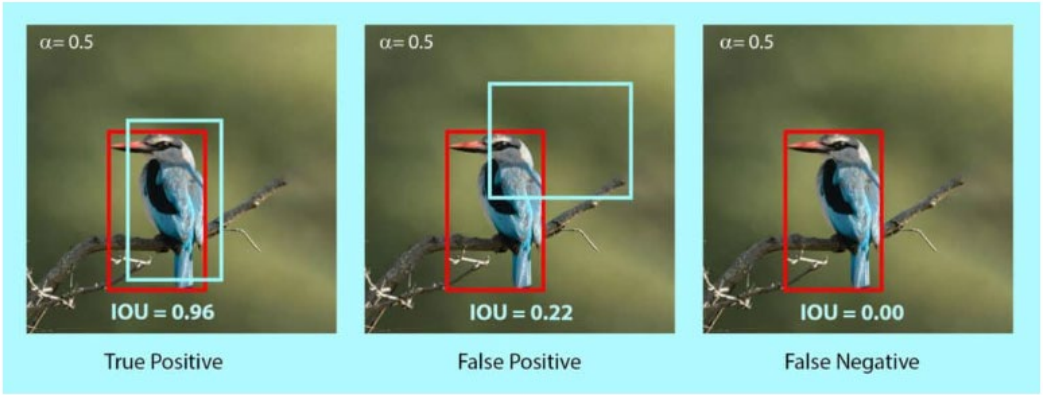


Fig: Object detection predictions based on IoU

Source: https://www.v7labs.com/blog/mean-average-precision
https://learnopencv.com/mean-average-precision-map-object-detection-model-evaluation-metric/

Non-Maximum Suppression (NMS)

- Non-maximum suppression (NMS) selects one bounding box out of many overlapping bounding boxes.

Source: https://learnopencv.com/non-maximum-suppression-theory-and-implementation-in-pytorch/

Non-Maximum Suppression (NMS)

- Input – a list P of prediction BBoxes with c as the predicted confidence score. The overlap threshold for IoU is T.
- Output – a list K of filtered prediction BBoxes.

- There are three steps for NMS.
- Step 1: Select the prediction S with the highest confidence score, remove S from P, and add it to the final prediction K, which is initially empty.

Source: https://learnopencv.com/non-maximum-suppression-theory-and-implementation-in-pytorch/

Non-Maximum Suppression (NMS)

- Step 2: Calculate the IoU score of this BBox S with every other BBoxes in P. If the IoU score is greater than T (e.g., 0.6-0.7) for any BBox R present in P, then remove R from P.
- Step 3: If P is non-empty, go to Step 1; otherwise, return K.

Before NMS (Source: That '70s Show)

After NMS (Source: That '70s Show)

Source: https://learnopencv.com/non-maximum-suppression-theory-and-implementation-in-pytorch/

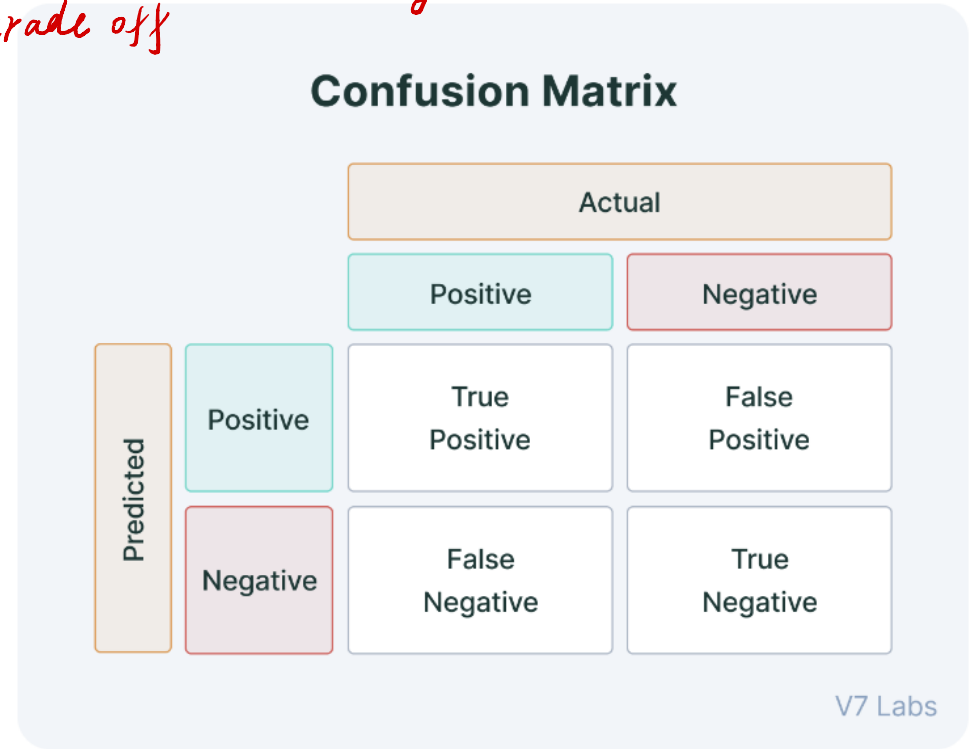
Non-Maximum Suppression (NMS)

- Example (in class)

Source: https://learnopencv.com/non-maximum-suppression-theory-and-implementation-in-pytorch/

Mean Average Precision (mAP)

- Mean average precision, mAP, evaluates the object detection model's performance and is computed based on IoU, Precision, and Recall.

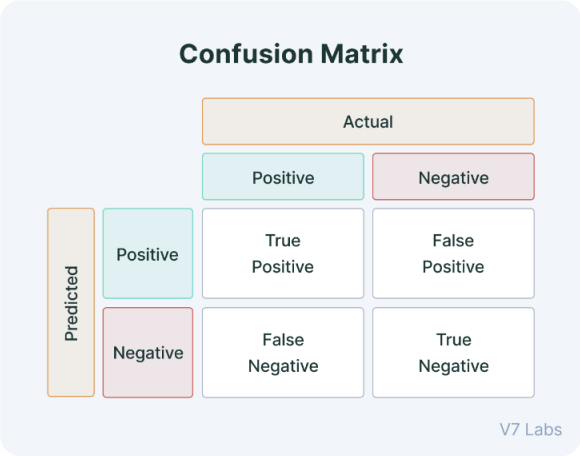


Confusion matrix

Source: https://www.v7labs.com/blog/mean-average-precision
https://learnopencv.com/mean-average-precision-map-object-detection-model-evaluation-metric/

Mean Average Precision (mAP)

- There are four items (in the confusion matrix) based on the IoU prediction threshold.
- True Positives (TP): The model predicts a label. The label matches the ground truth correctly.
- True Negatives (TN): The model does not predict the label at a location. The location is not a part of the ground truth.



Confusion matrix

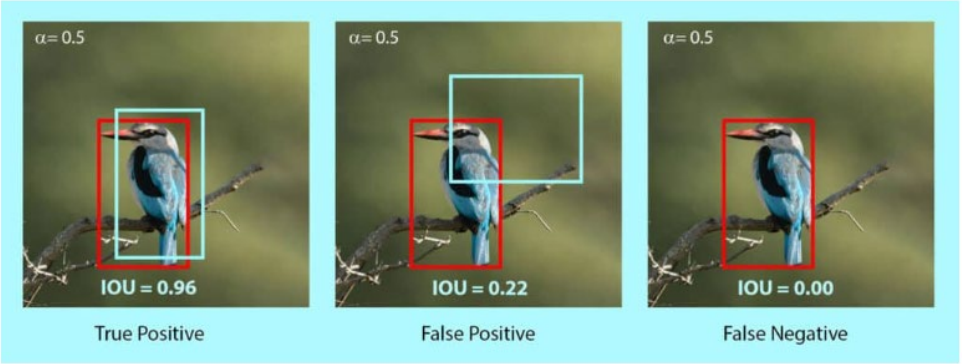
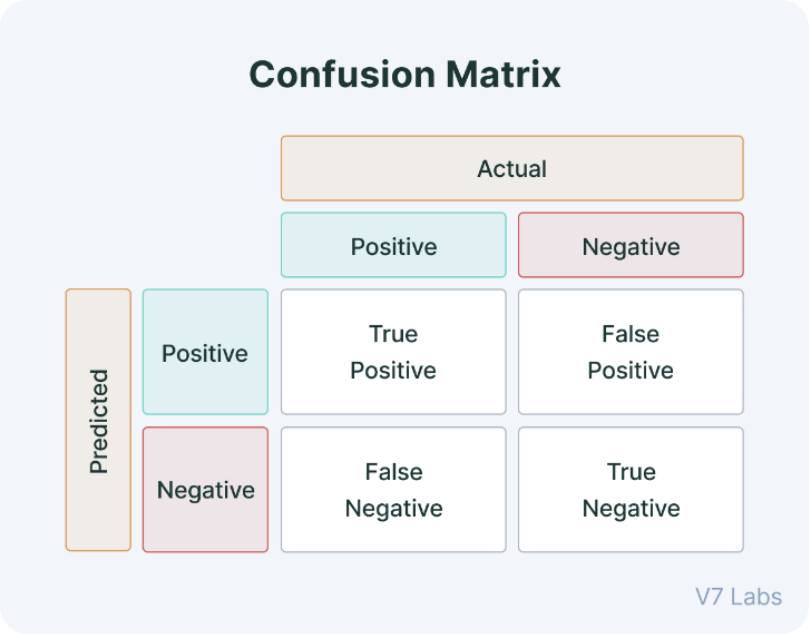


Fig: Object detection predictions based on IoU

Source: https://www.v7labs.com/blog/mean-average-precision
https://learnopencv.com/mean-average-precision-map-object-detection-model-evaluation-metric/

Mean Average Precision (mAP)

- False Positives (FP): The model predicts a label, but the label is not a part of the ground truth (Type I Error).
- False Negatives (FN): The model does not predict a label at a location. But the location is part of the ground truth. (Type II Error).



Confusion matrix

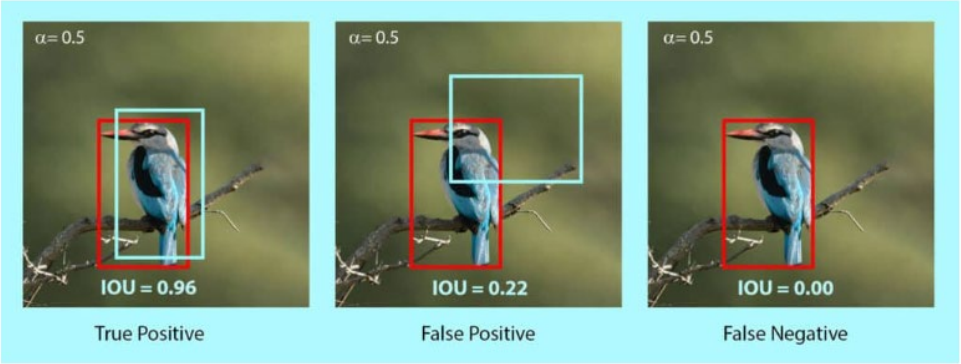


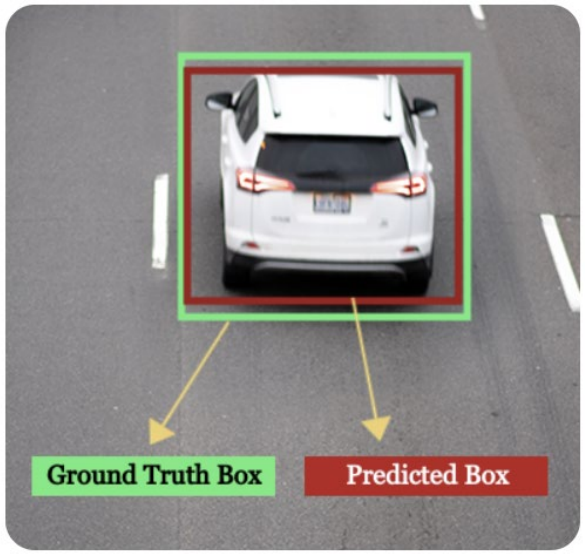
Fig: Object detection predictions based on IoU

Source: https://www.v7labs.com/blog/mean-average-precision
https://learnopencv.com/mean-average-precision-map-object-detection-model-evaluation-metric/

Mean Average Precision (mAP)

- Intersection over Union (IoU)

Intersection over Union



Ground truth box vs predicted box

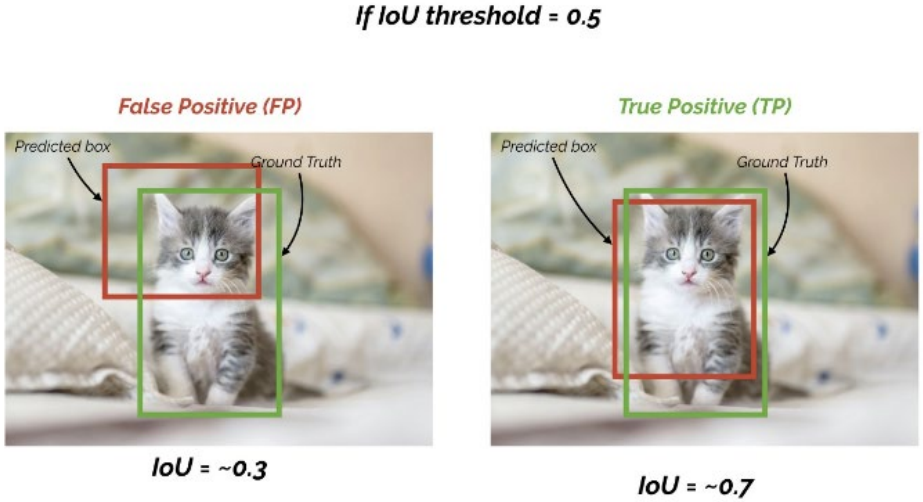
$$\text{IoU} = \frac{|A \cap B|}{|A| \cup |B|}$$

$$\text{IoU} = \frac{TP}{TP + FN + FP}$$

Source: https://www.v7labs.com/blog/mean-average-precision

Mean Average Precision (mAP)

- Precision measures how well you can find true positives (TP) out of all positive predictions (TP + FP), where (TP + FP) = all predictions. Precision = TP / (TP + FP), ranges 0-1.
- Precision is calculated using the IoU threshold in the object detection tasks.



Calculating IoU threshold

- The precision value may vary based on the model's IoU threshold.

Source: https://www.v7labs.com/blog/mean-average-precision

Mean Average Precision (mAP)

- Recall measures how well you can find true positives (TP) out of all ground truths (TP + FN), where (TP + FN) = all ground truths. Recall = TP / (TP + FN), ranges 0-1.
- With the paired values of Precision and Recall, the precision-recall curve for each class can be computed.
- The average precision, AP, is measured using the 11-point interpolation method or the area under the precision-recall curve.
- The mAP is calculated by finding the Average Precision, AP, for each class and then averaging over a number of classes

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

Mean Average Precision Formula

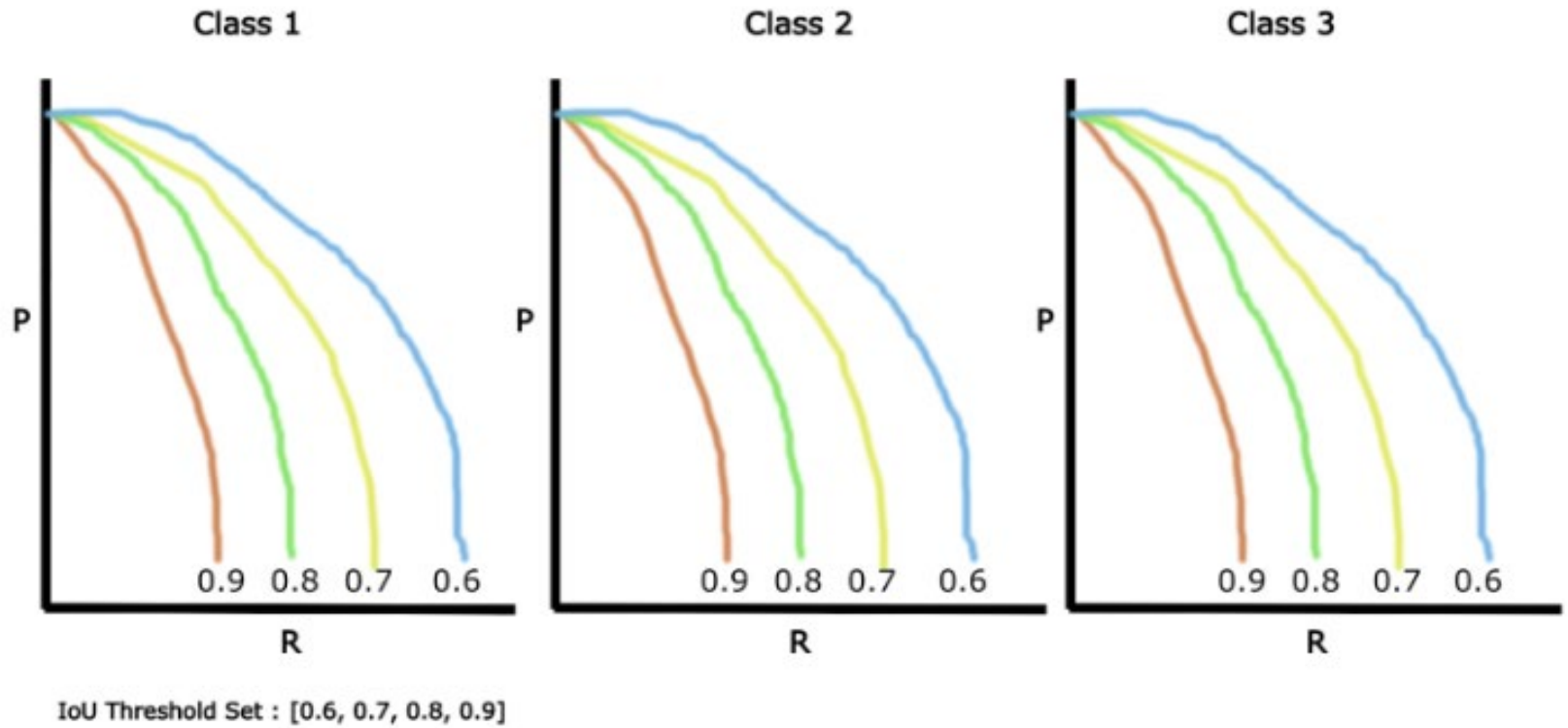
Source: https://www.v7labs.com/blog/mean-average-precision

Mean Average Precision (mAP)

- mAP for each model, N = number of classes.

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

Mean Average Precision Formula



Source: https://www.v7labs.com/blog/mean-average-precision

Mean Average Precision (mAP)

- The precision-recall curve is obtained by plotting the model's precision and recall values as a function of the model's confidence score threshold.
- mAP is used as a standard metric to analyze the accuracy of an object detection model.
- The primary challenge metric in the COCO 2017 challenge is calculated as follows:
  - AP is calculated for the IoU threshold of 0.5 for each class by calculating the precision at every recall value (0 to 1 with a step size of 0.01).
  - It is repeated for IoU thresholds of 0.55,0.60,…,0.95.
  - Average is taken over all 80 classes and all 10 thresholds.

Source: https://www.v7labs.com/blog/mean-average-precision

suggest possible areas in an image that contains an object.

R-CNN: Combining region proposals with CNN (Convolutional Neural Network)

- Object detection system overview

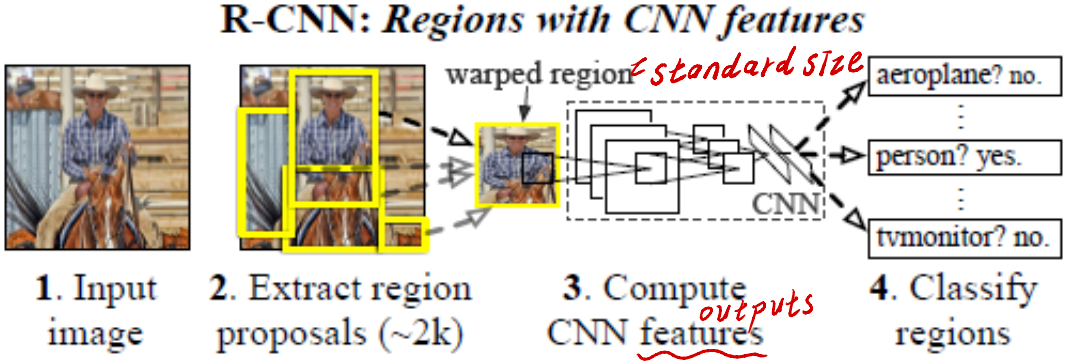


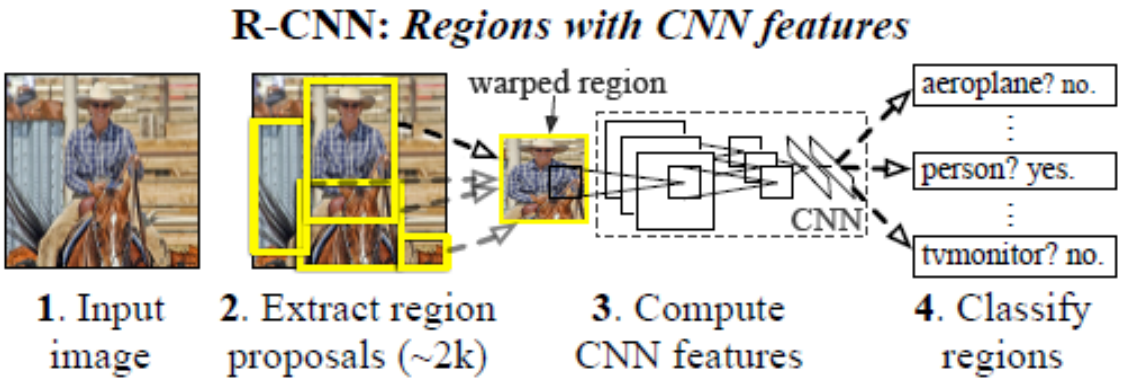
Figure 1: Object detection system overview. Our system (1) takes an input image, (2) extracts around 2000 bottom-up region proposals, (3) computes features for each proposal using a large convolutional neural network (CNN), and then (4) classifies each region using class-specific linear SVMs. R-CNN achieves a mean average precision (mAP) of **53.7% on PASCAL VOC 2010**. For comparison, [39] reports 35.1% mAP using the same region proposals, but with a spatial pyramid and bag-of-visual-words approach. The popular deformable part models perform at 33.4%. On the 200-class **ILSVRC2013 detection dataset, R-CNN's mAP is 31.4%**, a large improvement over OverFeat [34], which had the previous best result at 24.3%.

Source: Rich feature hierarchies for accurate object detection and semantic segmentation, R. Girshick, J. Donahue, T. Darrell, J. Malik, CVPR 2014

R-CNN: Regions with CNN features

- R-CNN is the first to show that a CNN can lead to dramatically higher object detection performance. It uses a region paradigm (region proposals) for object detection.

- At test time, R-CNN generates around 2000 category-independent region proposals for the input image, extracts a fixed-length feature vector from each proposal using a CNN, and then classifies each region with category-specific linear SVMs.

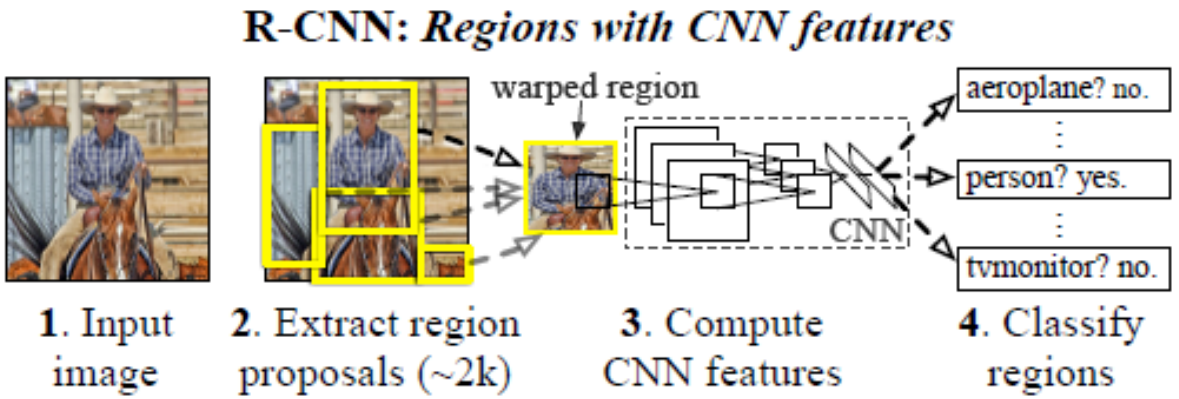


Source: Rich feature hierarchies for accurate object detection and semantic segmentation, R. Girshick, J. Donahue, T. Darrell, J. Malik, CVPR 2014

R-CNN = look twice
① region proposal.
② classification.

R-CNN: Regions with CNN features

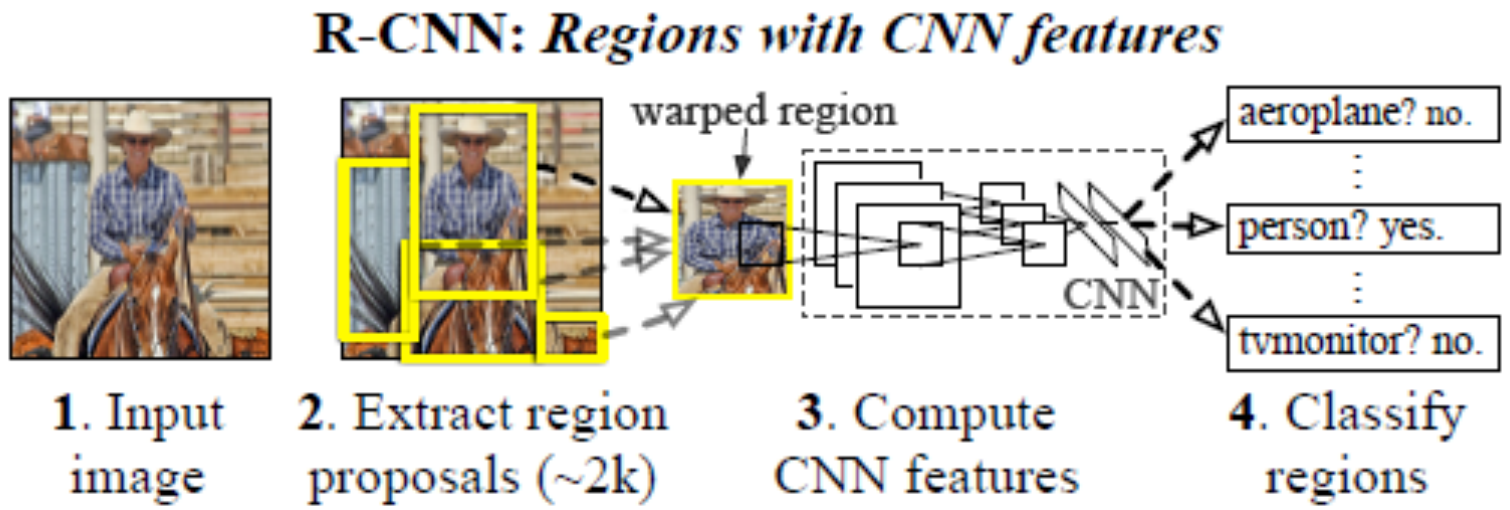
- R-CNN consists of three modules.
- The first generates category-independent region proposals.
- These proposals define the set of candidate detections available to the object detector, e.g., 2000 region proposals during test time.
- Regardless of the size or the aspect ratio of the candidate region, all pixels are warped into a tight bounding box around it to the required size.



Source: Rich feature hierarchies for accurate object detection and semantic segmentation, R. Girshick, J. Donahue, T. Darrell, J. Malik, CVPR 2014

R-CNN: Regions with CNN features

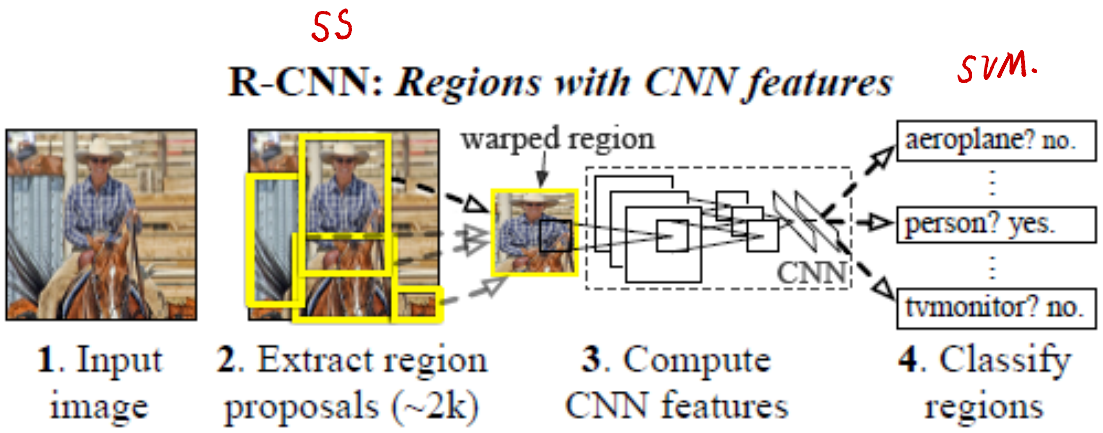
- The second module is a convolutional neural network (CNN, shared-weights) that extracts a fixed-length, 4096-dimensional feature vector from each region. Features are computed by forward propagating a 227x227 RGB image through five convolutional layers and two fully connected layers.



Source: Rich feature hierarchies for accurate object detection and semantic segmentation, R. Girshick, J. Donahue, T. Darrell, J. Malik, CVPR 2014

R-CNN: Regions with CNN features

- The third module is a set of class-specific linear SVMs, act as object detectors. R-CNN scores each extracted feature vector for each class using the SVM trained for that class. So, each region is scored.
- A non-maximum suppression is applied (for each class independently) that rejects a region if it has an IoU overlap with a higher-scoring selected region larger than a learned threshold.



Source: Rich feature hierarchies for accurate object detection and semantic segmentation, R. Girshick, J. Donahue, T. Darrell, J. Malik, CVPR 2014

先把整张图像放入一个CNN提取特征矩阵.
再输入region，获得特征矩阵中对应的一小块，
利用此子特征矩阵来对region进行分类

Fast R-CNN

- In R-CNN, numerous candidate object locations (proposals) must be processed (features are extracted from each object proposal at test time). As such, R-CNN is computationally slow because it performs a ConvNet forward pass for each object proposal without sharing computation.

- Fast R-CNN streamlines the training process by using a single-stage training algorithm using a multi-task loss that jointly learns to classify and refine their spatial locations. Fast R-CNN can efficiently classify object proposals using deep convolutional networks.

Source: Fast R-CNN, R. Girshick, ICCV 2015

Fast R-CNN

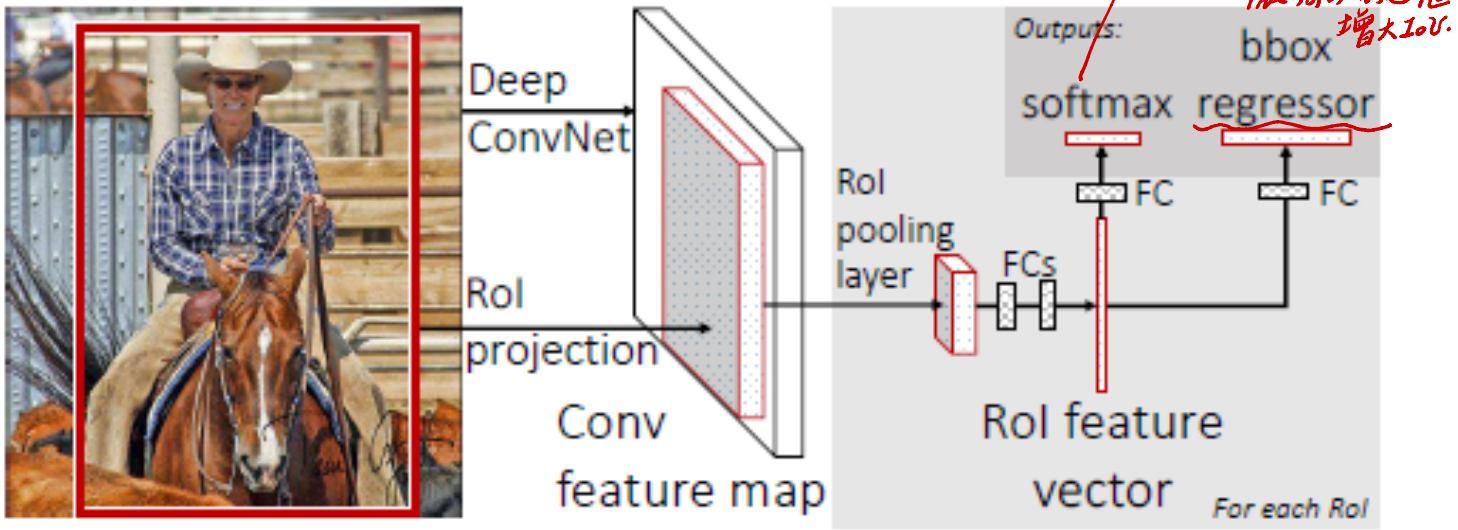


Figure 1. Fast R-CNN architecture. An input image and multiple regions of interest (RoIs) are input into a fully convolutional network. Each RoI is pooled into a fixed-size feature map and then mapped to a feature vector by fully connected layers (FCs). The network has two output vectors per RoI: softmax probabilities and per-class bounding-box regression offsets. The architecture is trained end-to-end with a multi-task loss.

Source: Fast R-CNN, R. Girshick, ICCV 2015

Fast R-CNN

- A Fast R-CNN network inputs an entire image and a set of object proposals as input. The network first processes the whole image with several convolutional (conv) and max pooling layers to produce a conv feature map in a single CNN.

- For each object proposal, a region of interest (RoI) pooling layer extracts a fixed-length feature vector from the feature map by (1) passing/resizing the region proposals to the feature map and (2) performing max pooling in the resized region proposals.

- The computational cost is drastically reduced by sharing convolutions across proposals.

Source: Fast R-CNN, R. Girshick, ICCV 2015

Fast R-CNN

- Each feature vector is fed into a sequence of fully connected (FC) layers that finally branch into two sibling output layers:

  - one that produces softmax probability estimates over K object classes plus a catch-all “background” class and

  - another layer that outputs four real-valued numbers for each of the K object classes. Each set of 4 values encodes refined bounding-box positions for one of the K classes.

Source: Fast R-CNN, R. Girshick, ICCV 2015

Fast R-CNN

	Fast R-CNN			R-CNN			SPPnet
	S	**M**	**L**	**S**	**M**	**L**	†**L**
train time (h)	**1.2**	2.0	9.5	22	28	84	25
train speedup	**18.3**×	14.0×	8.8×	1×	1×	1×	3.4×
test rate (s/im)	0.10	0.15	0.32	9.8	12.1	47.0	2.3
▷ with SVD	**0.06**	0.08	0.22	-	-	-	-
test speedup	98×	80×	146×	1×	1×	1×	20×
▷ with SVD	169×	150×	**213**×	-	-	-	-
VOC07 mAP	57.1	59.2	**66.9**	58.5	60.2	66.0	63.1
▷ with SVD	56.5	58.7	66.6	-	-	-	-

Table 4. Runtime comparison between the same models in Fast R-CNN, R-CNN, and SPPnet. Fast R-CNN uses single-scale mode. SPPnet uses the five scales specified in [11]. †Timing provided by the authors of [11]. Times were measured on an Nvidia K40 GPU.

method	classifier	S	M	L
R-CNN [9, 10]	SVM	**58.5**	**60.2**	66.0
FRCN [ours]	SVM	56.3	58.7	66.8
FRCN [ours]	softmax	57.1	59.2	**66.9**

Table 8. Fast R-CNN with softmax vs. SVM (VOC07 mAP).

Source: Fast R-CNN, R. Girshick, ICCV 2015

You Only Look Once: YOLO

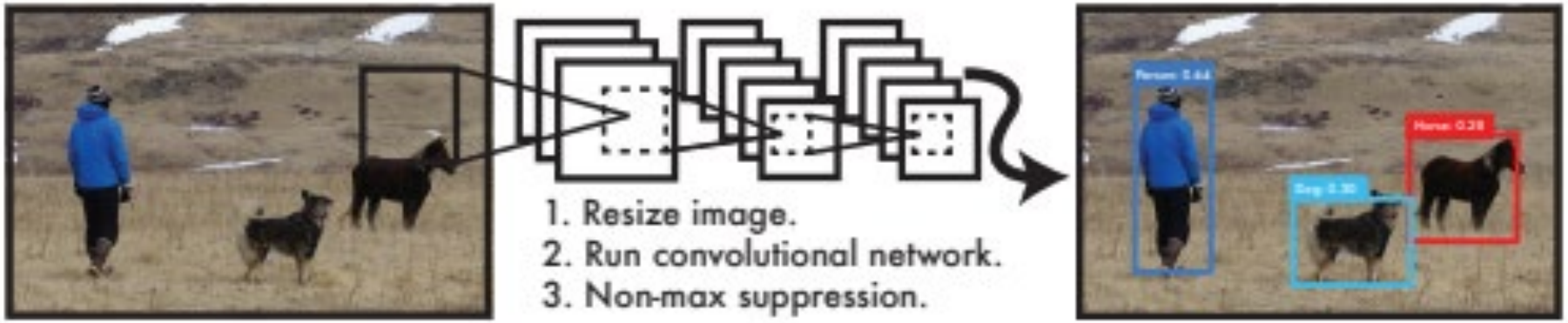
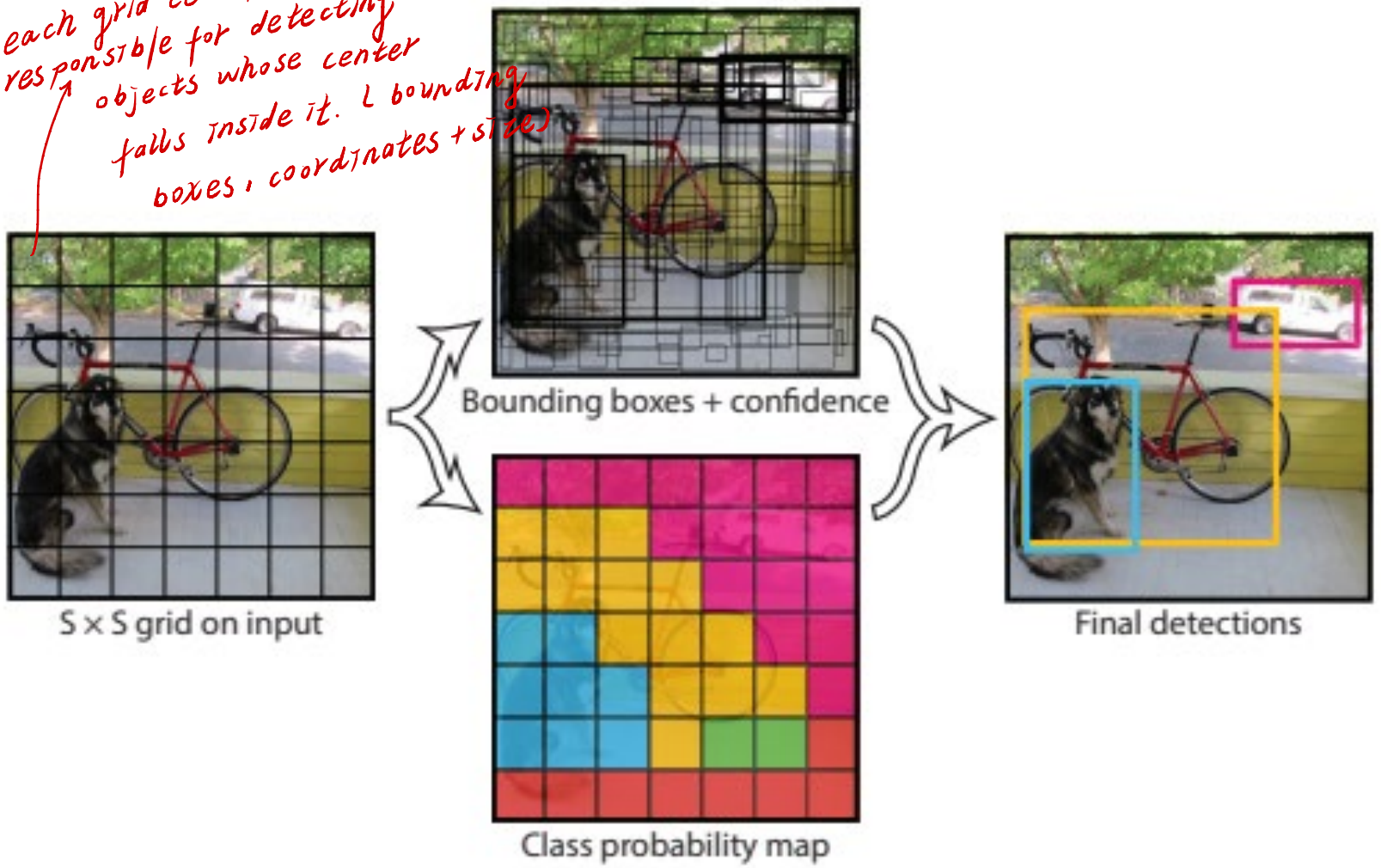


Figure 1: The YOLO Detection System. Processing images with YOLO is simple and straightforward. Our system (1) resizes the input image to 448 × 448, (2) runs a single convolutional network on the image, and (3) thresholds the resulting detections by the model's confidence.

Ref Source: https://arxiv.org/abs/1506.02640

You Only Look Once: YOLO

each grid cell is responsible for detecting objects whose center falls inside it. 2 bounding boxes, coordinates + size)



Ref Source: https://arxiv.org/abs/1506.02640

You Only Look Once: YOLO

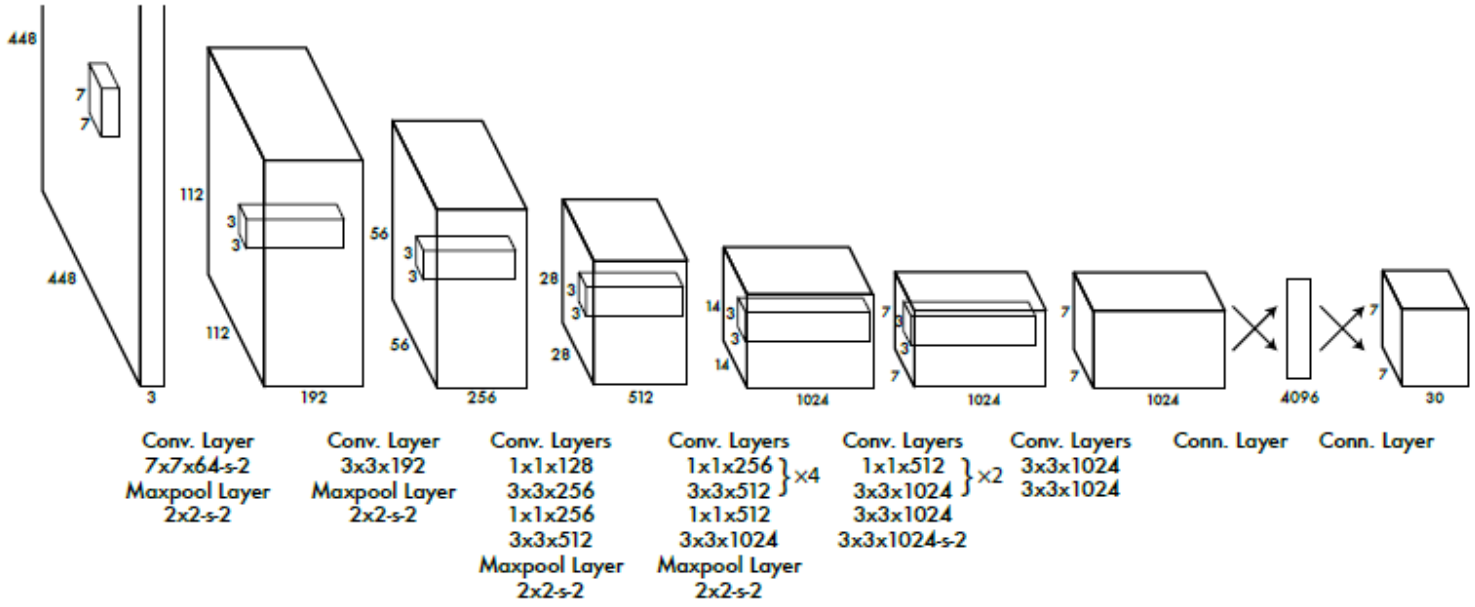


Figure 3: The Architecture. Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1×1 convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution (224×224 input image) and then double the resolution for detection.

Ref Source: https://arxiv.org/abs/1506.02640

You Only Look Once: YOLO

	mAP	**FPS**
Fast YOLO	52.7	155
YOLO	63.4	45
Faster R-CNN	73.2	7

Ref Source: https://arxiv.org/abs/1506.02640

move region proposals from CPU to GPU

Faster R-CNN

- Proposals are the computational bottleneck in detection systems. Fast region-based CNNs take advantage of GPUs, while the region proposal methods used in research are implemented on the CPU, making such runtime comparisons unfair. Re-implementing proposal computation for the GPU is an obvious way to accelerate it.

- Faster R-CNN uses Region Proposal Networks (RPNs) that share convolutional layers with the object detection networks.

Source: Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun, NIPS 2015

Faster R-CNN

- The main observation is that the convolutional (conv) feature maps used by region-based detectors, like Fast R-CNN, can also be used to generate region proposals.

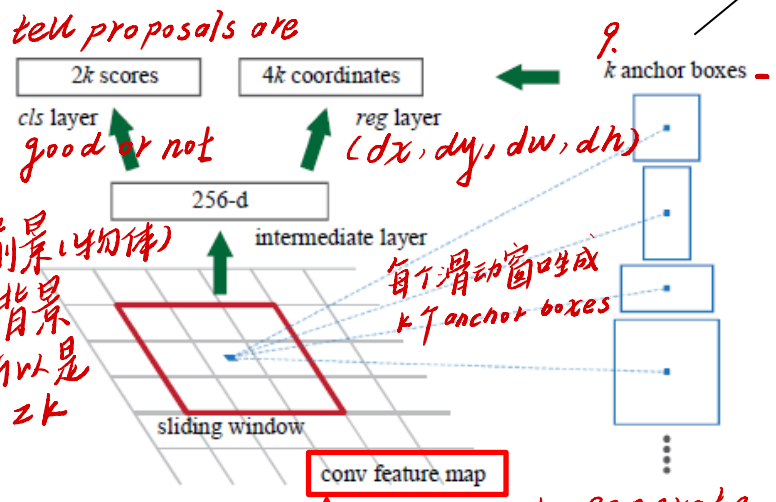
- On top of these conv features, RPNs (Region Proposal Networks) are constructed by adding two additional conv layers: (1) one that encodes each conv map position into a short (e.g., 256-d) feature vector and (2) a second that, at each conv map position, outputs an objectness score (whether the box contains an object) and regressed bounds (box transformation parameters) for k region proposals at that location, k = 9 typically.

Source: Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun, NIPS 2015

Faster R-CNN

A Region Proposal Network (RPN) takes an image (of any size) as input and outputs a set of rectangular object proposals, each with an objectness score.

3 scales and 3 aspect ratios are used, yielding k = 9 anchors at each sliding position for classification and regression.



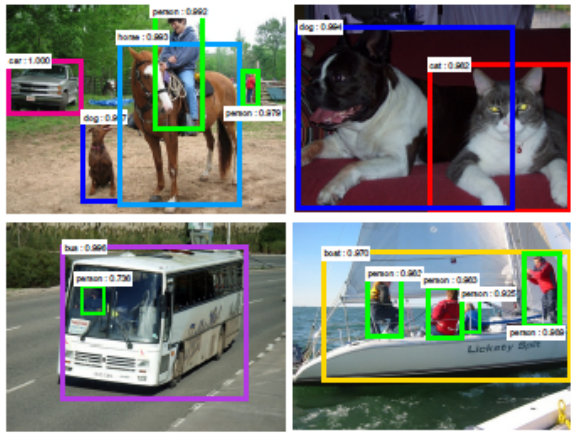


Figure 1: **Left**: Region Proposal Network (RPN). **Right**: Example detections using RPN proposals on PASCAL VOC 2007 test. Our method detects objects in a wide range of scales and aspect ratios.

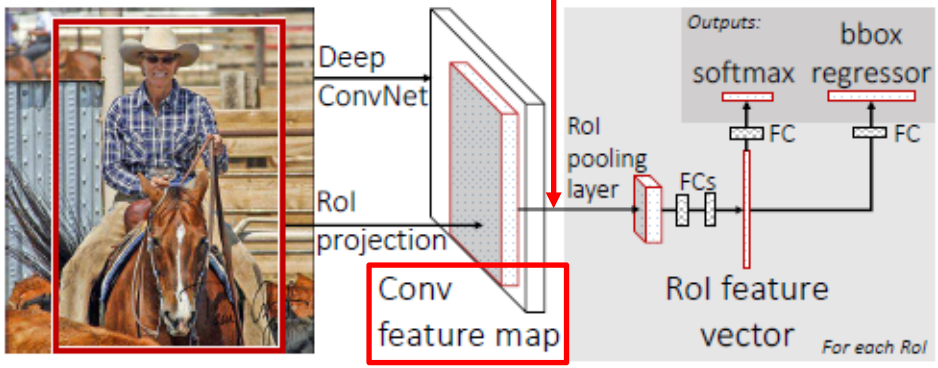


Figure 1. Fast R-CNN architecture.

Source: Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun, NIPS 2015

Faster R-CNN

Table 2: Detection results on **PASCAL VOC 2007 test set**. The detector is Fast R-CNN and VGG-16. Training data: "07": VOC 2007 trainval, "07+12": union set of VOC 2007 trainval and VOC 2012 trainval. For RPN, the train-time proposals for Fast R-CNN are 2k. †: this was reported in [5]; using the repository provided by this paper, this number is higher (68.0±0.3 in six runs).

method	# proposals	data	mAP (%)	time (ms)
SS	2k	07	66.9†	1830
SS	2k	07+12	70.0	1830
RPN+VGG, shared	300	07	69.9	**198**
RPN+VGG, shared	300	07+12	**73.2**	**198**

Table 4: **Timing** (ms) on a K40 GPU, except SS proposal is evaluated in a CPU. "Region-wise" includes NMS, pooling, fc, and softmax. See our released code for the profiling of running time.

model	system	conv	proposal	region-wise	total	rate
VGG	SS + Fast R-CNN	146	1510	174	1830	0.5 fps
VGG	RPN + Fast R-CNN	141	**10**	47	**198**	**5 fps**
ZF	RPN + Fast R-CNN	31	**3**	25	**59**	**17 fps**

Source: Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun, NIPS 2015

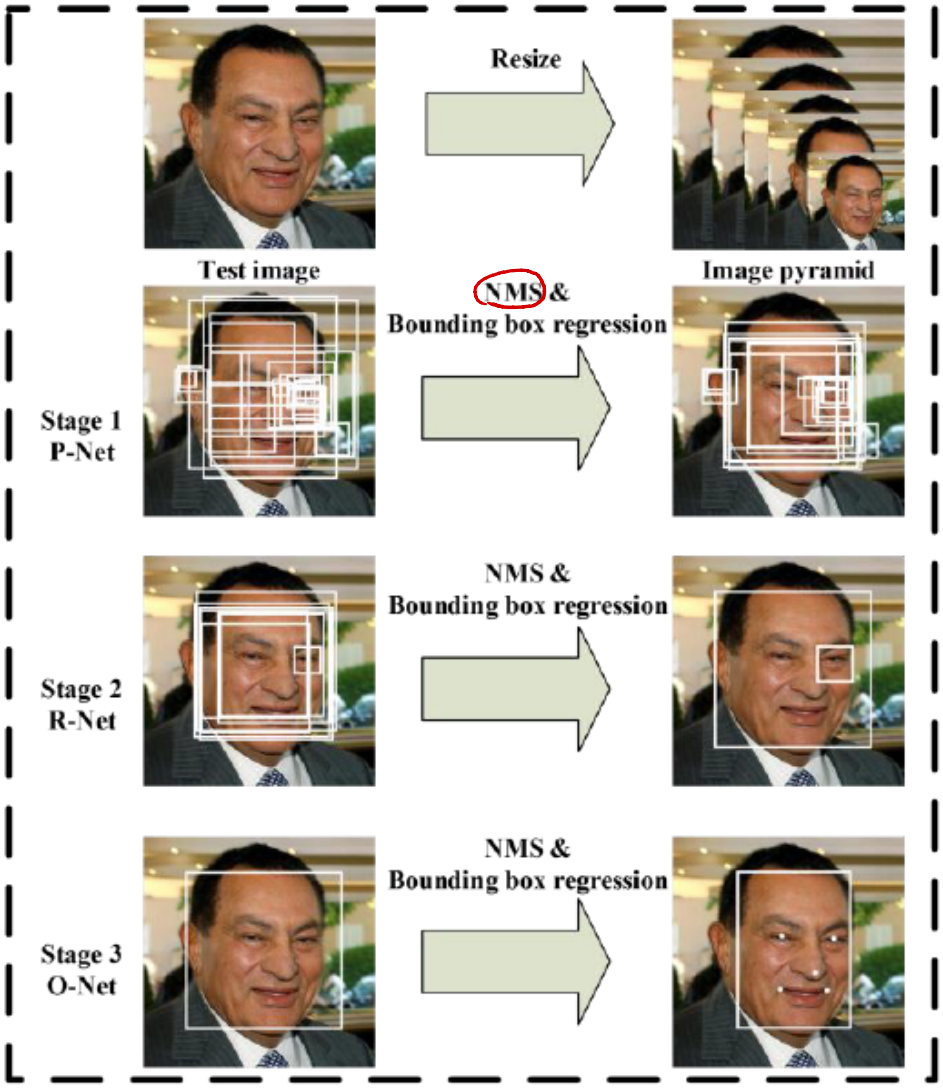
Face detection and facial landmark localization with MTCNN

- Multitask cascaded convolutional networks (MTCNN) use cascaded CNNs by multi-task learning (with three CNNs) for face detection, regression of bounding box parameters, and facial landmarks' positions.

- The proposed CNNs consist of three stages. In the first stage, it quickly produces candidate windows through a shallow CNN. Then, it refines the windows to reject a large number of non-face windows through a more complex CNN. Finally, it uses a more powerful CNN to refine the result and output facial landmark positions.

- It gives high accuracy in face detection with high average precision, ~0.915.

Source: Joint face detection and alignment using multitask cascaded convolutional networks, Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, Yu Qiao, IEEE Signal Processing Letters 23:10, 2016



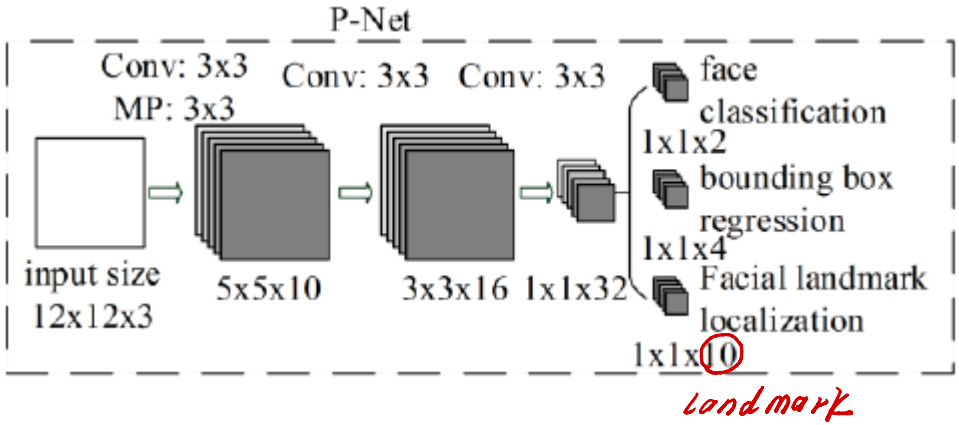
first pass

Fig. 1. Pipeline of our cascaded framework that includes three-stage multi-task deep convolutional networks. Firstly, candidate windows are produced through a fast Proposal Network (P-Net). After that, we refine these candidates in the next stage through a Refinement Network (R-Net). In the third stage, The Output Network (O-Net) produces final bounding box and facial landmarks position.

Source: Joint face detection and alignment using multitask cascaded convolutional networks, Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, Yu Qiao, IEEE Signal Processing Letters 23:10, 2016

- Stage 1: A fully convolutional network, called Proposal Network (P-Net), is used to obtain the candidate windows and their bounding box regression vectors.
- The estimated bounding box regression vectors are used to calibrate the candidates.
- After that, we employ non-maximum suppression (NMS) to merge highly overlapped candidates.

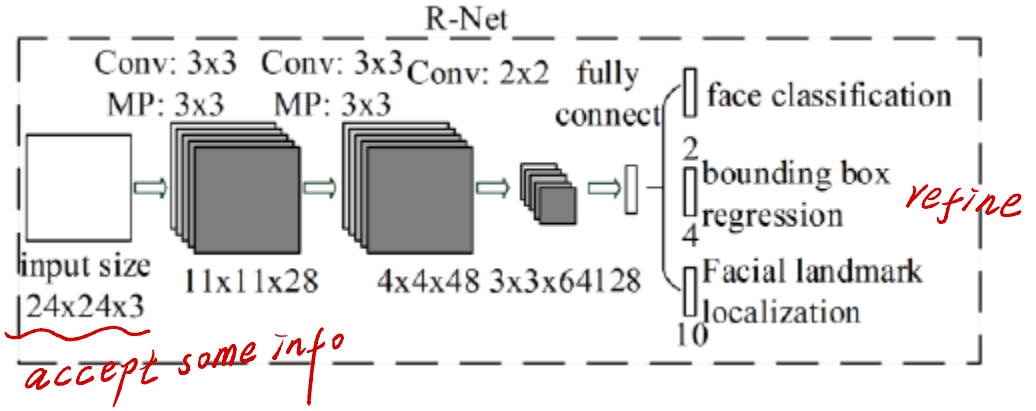
① For obtaining the candidate windows



Source: Joint face detection and alignment using multitask cascaded convolutional networks, Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, Yu Qiao, IEEE Signal Processing Letters 23:10, 2016

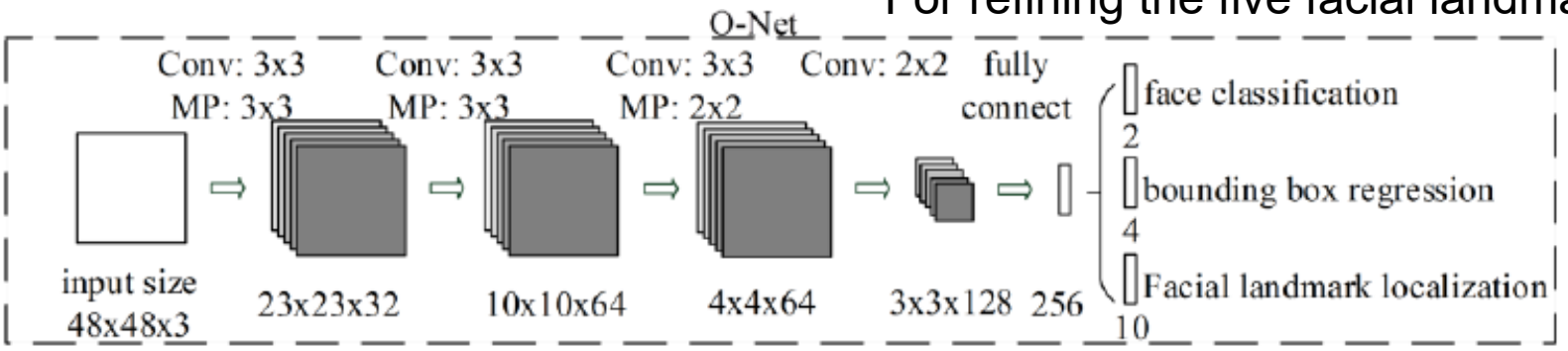
- Stage 2: all candidates are fed to another CNN, called Refine Network (R-Net), which further rejects a large number of false candidates, performs calibration with bounding box regression, and NMS candidate merge.

For rejecting the candidate windows



- Stage 3: This stage is similar to the second stage, but in this stage the authors aim to describe the face in more details. In particular, the network will output five facial landmarks' positions.

For refining the five facial landmarks



Source: Joint face detection and alignment using multitask cascaded convolutional networks, Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, Yu Qiao, IEEE Signal Processing Letters 23:10, 2016

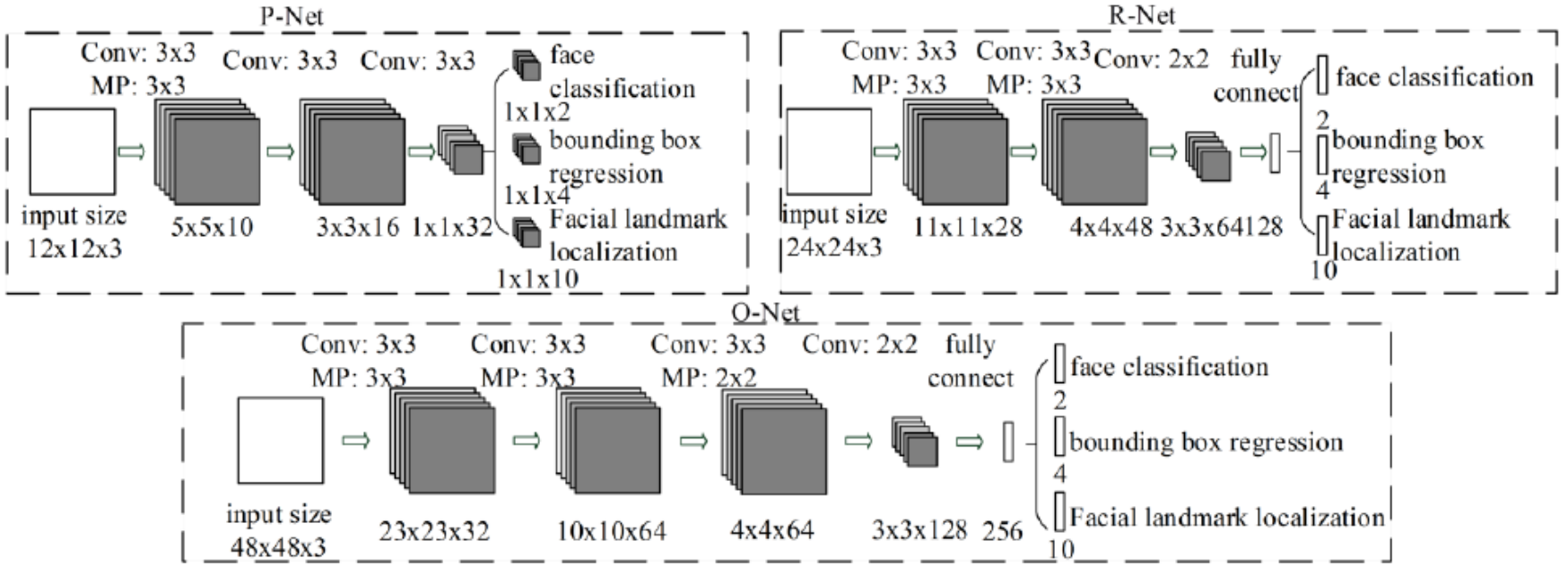


Fig. 2. The architectures of P-Net, R-Net, and O-Net, where "MP" means max pooling and "Conv" means convolution. The step size in convolution and pooling is 1 and 2, respectively.

Source: Joint face detection and alignment using multitask cascaded convolutional networks, Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, Yu Qiao, IEEE Signal Processing Letters 23:10, 2016