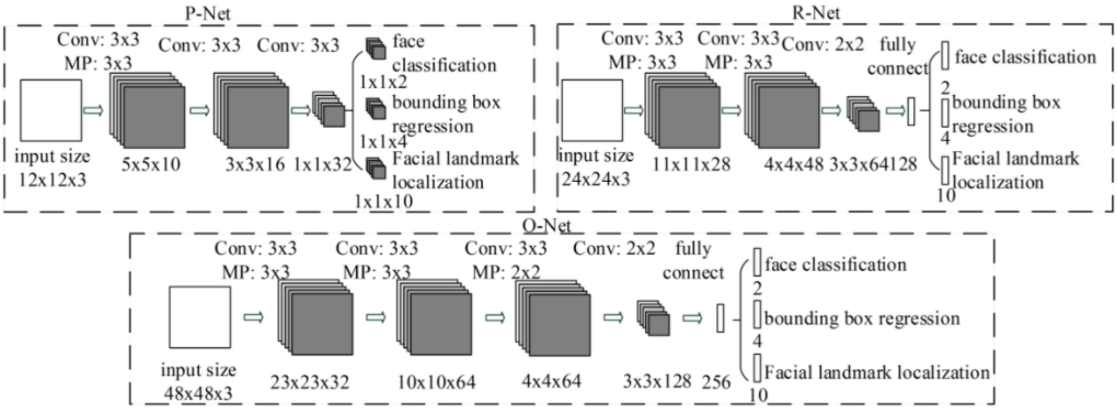
MTCNN (Multi-Task Cascaded Convolutional Neural Networks)

Paper -: LINK

Task-: Face detection and alignment

A blog to refer -: LINK, Three task details blog

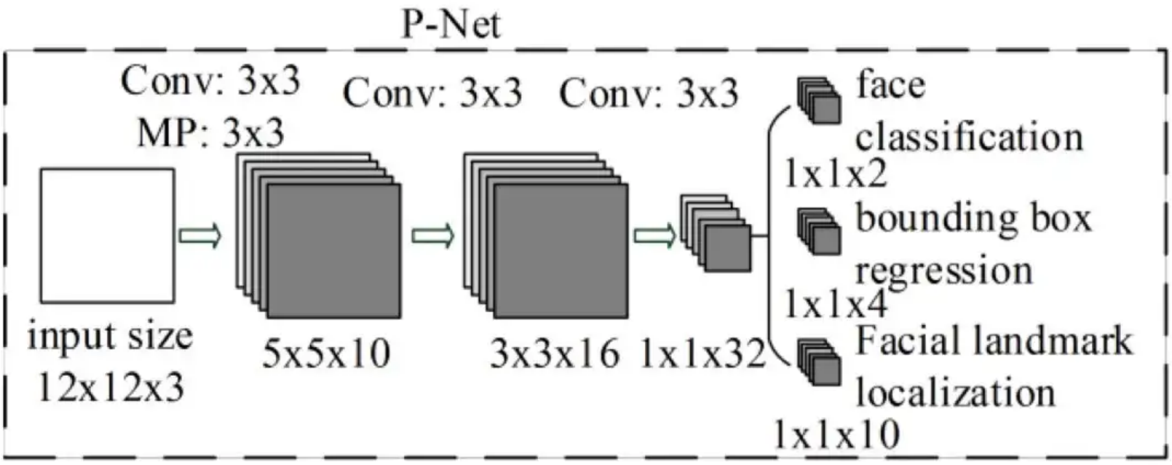


Three Networks used in MTCNN

Three Stages of MTCNN

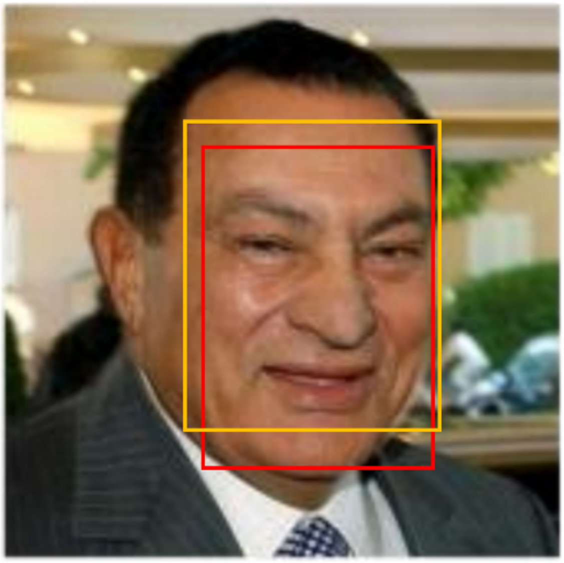
The first step is to take the image and resize it to different scales in order to build an image pyramid, which is the input of the following three-staged cascaded network.

1. **Stage 1: The Proposal Network (P-Net)**

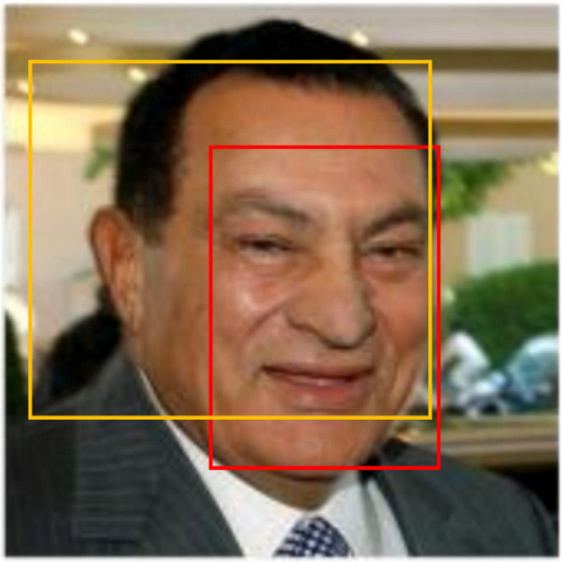


The Scaled Images are passed to the network one by one. This will help to detect faces of various sizes.

- This first stage is a fully convolutional network (FCN). which means it has no FC layers all the layers are convolution layers.
- Input kernel size of 12 * 12.
- The 12 * 12 kernel slides above the scaled Images with a stride of 2.
- The network is trained to generate bounding boxes for the face.
- Delete boxes with a lower confidence level.
- We will have to standardize the coordinate system, converting all the coordinate systems to that of the actual, “un-scaled” image.
- Apply NMS (Non-maximum Suppression), subsequently, we calculate the area of each of the kernels, as well as the overlapping area between each kernel and the kernel with the highest score. The kernels that overlap a lot with the high-scoring kernel get deleted.



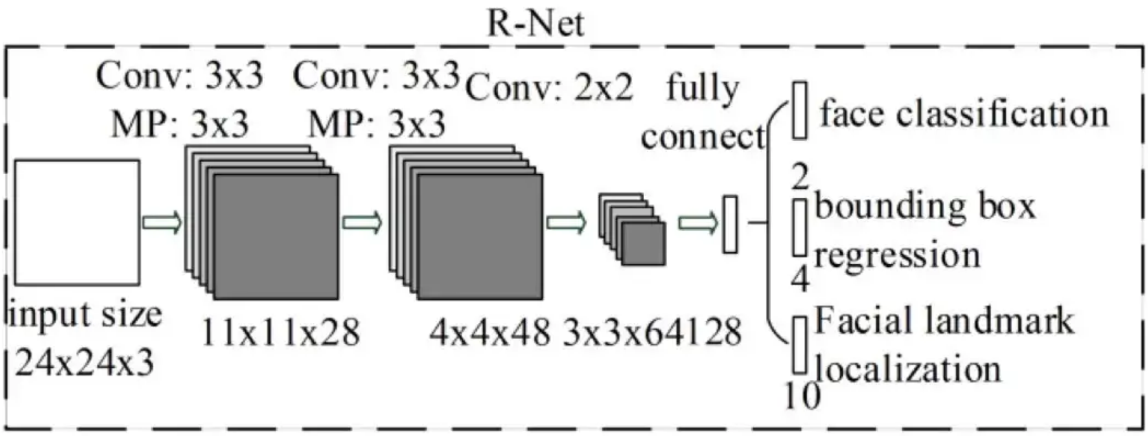
Large overlap, yellow box gets deleted



Small overlap, yellow box remains

- After having the bounding boxes (or candidate windows), it is passed to the next stage of R-Net.

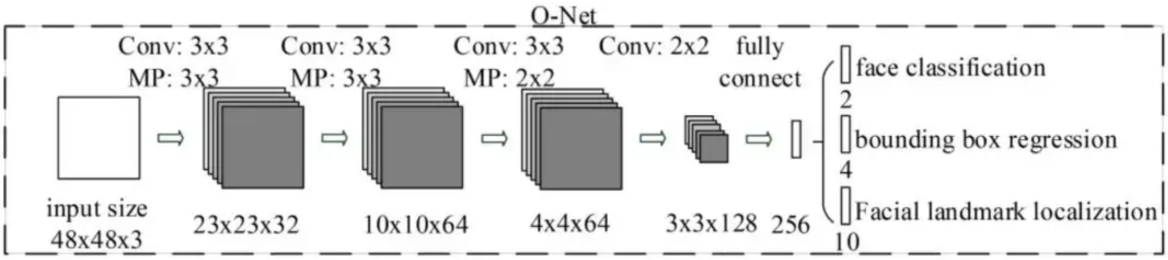
2. **Stage 2: The Refine Network (R-Net)**



R-Net's output is similar to that of P-Net: It includes the coordinates of the new, more accurate bounding boxes, as well as the confidence level of each of these bounding boxes.

- Once again, we get rid of the boxes with lower confidence and perform NMS on every box to further eliminate redundant boxes. Since the coordinates of these new bounding boxes are based on the P-Net bounding boxes, we need to convert them to the standard coordinates.
- After standardizing the coordinates, we reshape the bounding boxes to a square to be passed on to O-Net.

3. **Stage 3: The Output Network (O-Net)**



- The bounding boxes are resized to 48 * 48 pixels, then they are passed into O-Net.
- The outputs of O-Net are slightly different from that of P-Net and R-Net. O-Net provides 3 outputs: the coordinates of the bounding box (out[0]), the coordinates of the 5 facial landmarks (out[1]), and the confidence level of each box (out[2]).
- Once again, we get rid of the boxes with lower confidence levels and standardize both the bounding box coordinates and the facial landmark coordinates.
- Finally, we run them through the last NMS.
- At this point, there should only be one bounding box for every face in the image.

FaceNet

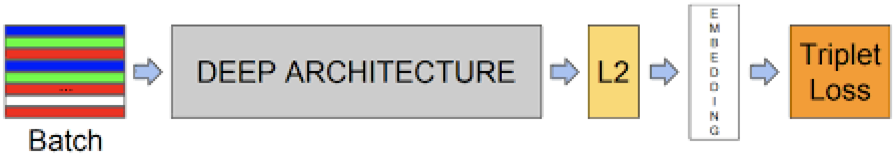
Founded -: 2015

Paper -: LINK

Task -: Face Embedding

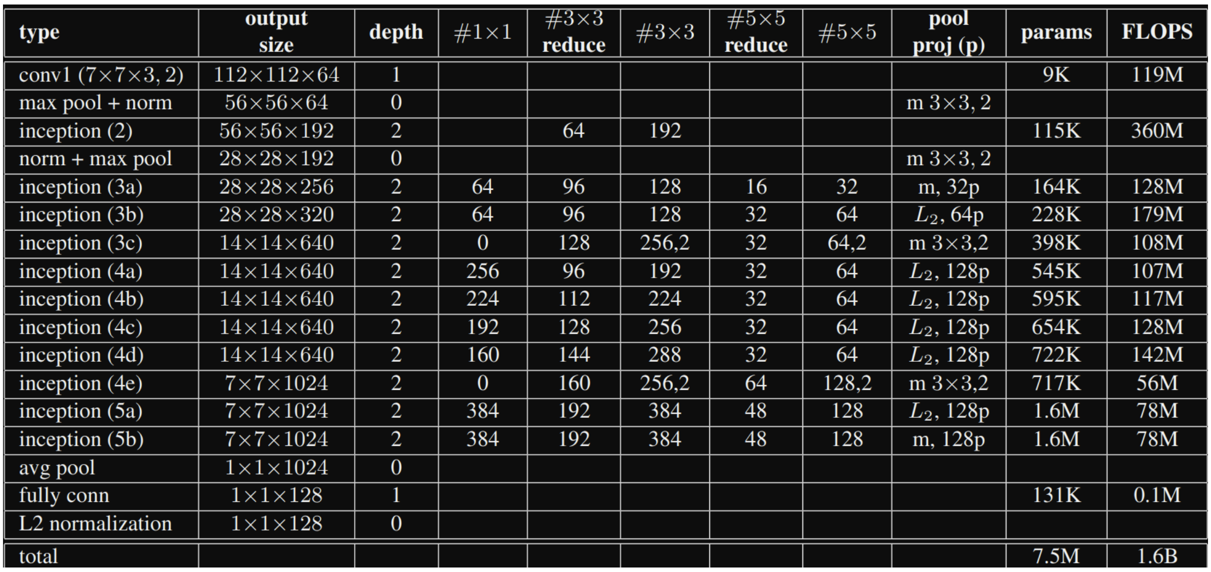
Blogs -: LINK1, LINK2, LINK3

FaceNet takes an image of the person's face as input and outputs a vector of 128 numbers which represent the most important features of a face. In machine learning, this vector is called **embedding**.



Deep Convolution Neural Network (DCNN)

- 22 Layer Architecture

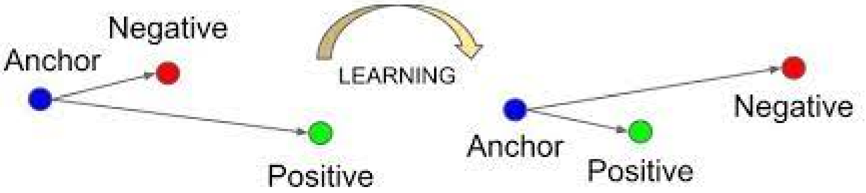


type	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj (p)	params	FLOPS
conv1 (7×7×3, 2)	112×112×64	1							9K	119M
max pool + norm	56×56×64	0						m 3×3, 2		
inception (2)	56×56×192	2		64	192				115K	360M
norm + max pool	28×28×192	0						m 3×3, 2		
inception (3a)	28×28×256	2	64	96	128	16	32	m, 32p	164K	128M
inception (3b)	28×28×320	2	64	96	128	32	64	L_2, 64p	228K	179M
inception (3c)	14×14×640	2	0	128	256,2	32	64,2	m 3×3,2	398K	108M
inception (4a)	14×14×640	2	256	96	192	32	64	L_2, 128p	545K	107M
inception (4b)	14×14×640	2	224	112	224	32	64	L_2, 128p	595K	117M
inception (4c)	14×14×640	2	192	128	256	32	64	L_2, 128p	654K	128M
inception (4d)	14×14×640	2	160	144	288	32	64	L_2, 128p	722K	142M
inception (4e)	7×7×1024	2	0	160	256,2	64	128,2	m 3×3,2	717K	56M
inception (5a)	7×7×1024	2	384	192	384	48	128	L_2, 128p	1.6M	78M
inception (5b)	7×7×1024	2	384	192	384	48	128	m, 128p	1.6M	78M
avg pool	1×1×1024	0								
fully conn	1×1×128	1							131K	0.1M
L2 normalization	1×1×128	0								
total									7.5M	1.6B

- DCNN will transform a face picture into a face features (Vector) in 128d.

Triplet-Loss

- **The triplet-loss function use three pictures during evaluations.**
- **The anchor is an arbitrary picture (some person).**
- **The picture that is positive belong to the same class (same person). The picture that is negative belong to various class (different person) from the anchor.**
- **The triple loss reduces the distance among the anchor and positive picture while increasing distance among anchor and negative picture.**


Negative
Anchor
Positive
LEARNING
Negative
Anchor
Positive