Baidu-III Project-II

AI for CV Group
2020

Week 7-8 II Face Detection & Recognition

Contents:

I. Face Detection

A. Introduction
B. A real example: FaceBoxes
C. Code of FaceBoxes
D. Advices for Projects

II. Face Recognition

E. What Is Recognition
F. Face Recognition Procedure
G. A real example: LightCNN

I. Face Detection

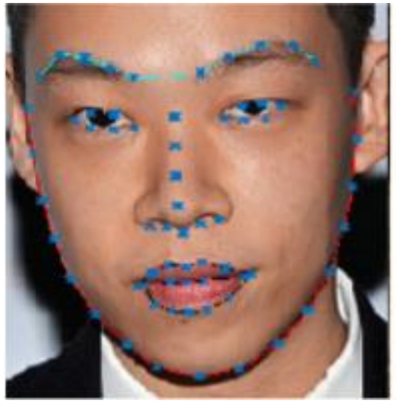
I. Face Detection

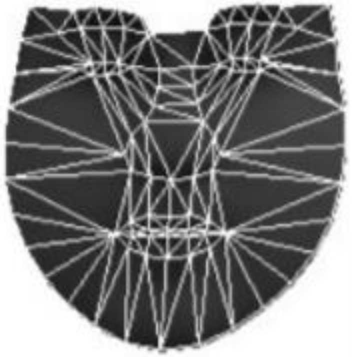
A. Introduction:

Red: Have to know before interview

1998~

Development: ASM — Viola Jones — MTCNN — SSH — BlazeFace





- AAM / CLM
- PCA

I. Face Detection

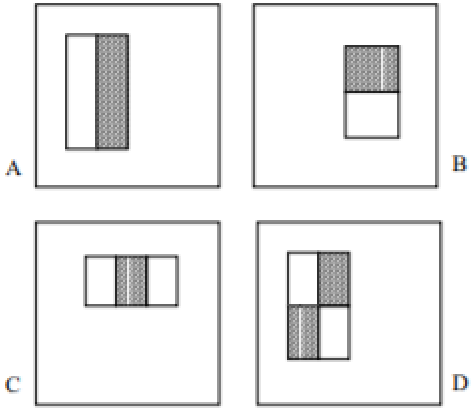
A. Introduction:

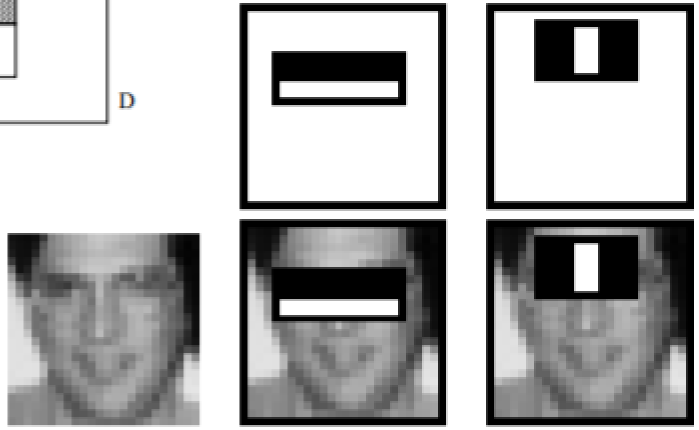
Pink: Better to know before interview

Blue: Code!!

2001

Development: ASM — Viola Jones — MTCNN — SSH — BlazeFace





- Haar Feature
- Integral Image
- Adaboost
- Cascade

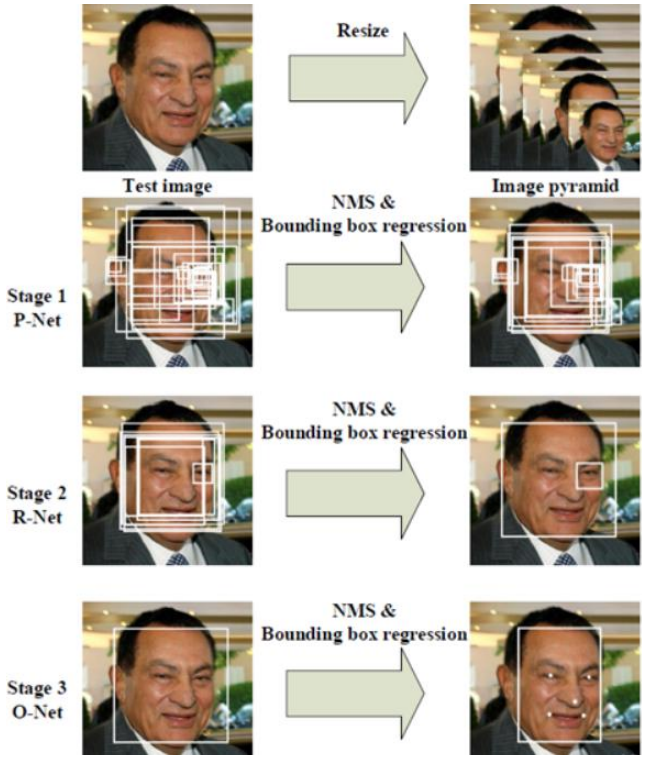
I. Face Detection

A. Introduction:

Pink: Better to know before interview

2015

Development: ASM — Viola Jones — **MTCNN** — SSH — BlazeFace



- Accurate and fast
- 3 stages / 3 networks

P-Net: Proposal Net

R-Net: Refined Net

O-Net: Output Net

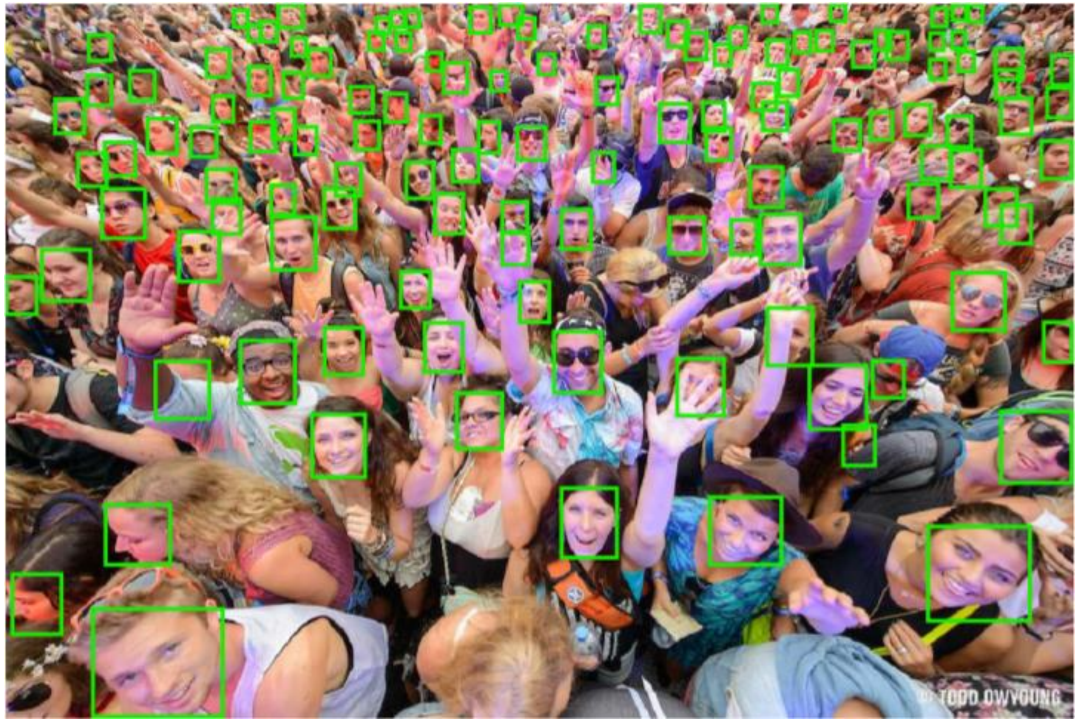
I. Face Detection

A. Introduction:

Green: Good to know before interview

2017

Development: ASM — Viola Jones — MTCNN — SSH — BlazeFace



- Accurate and fast
- Small faces in wild
- SSD-like structure

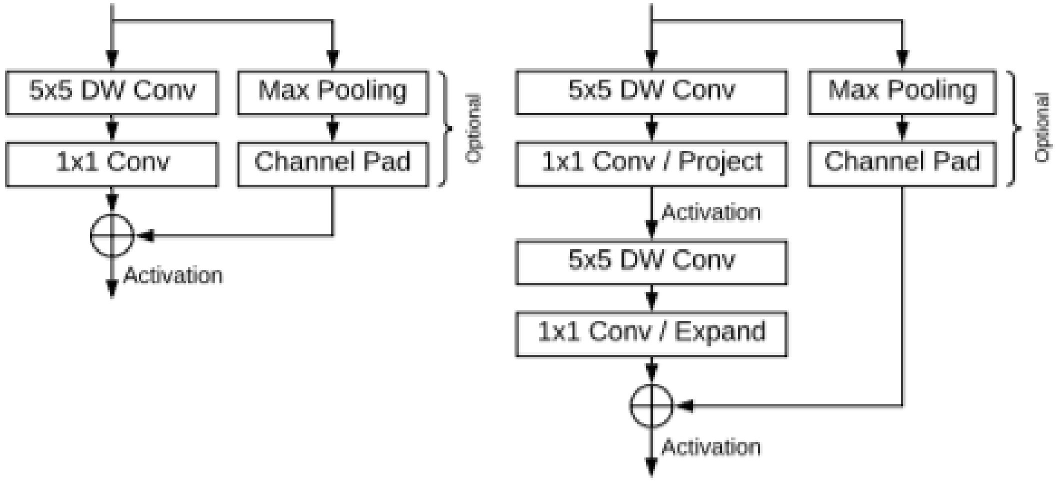
I. Face Detection

A. Introduction:

Green: Good to know before interview

2019

Development: ASM — Viola Jones — MTCNN — SSH — BlazeFace



- Accurate and extremely fast!
- Small faces in wild
- Latest tech in face detection
- Realtime in mobile devices

I. Face Detection

A. Introduction:

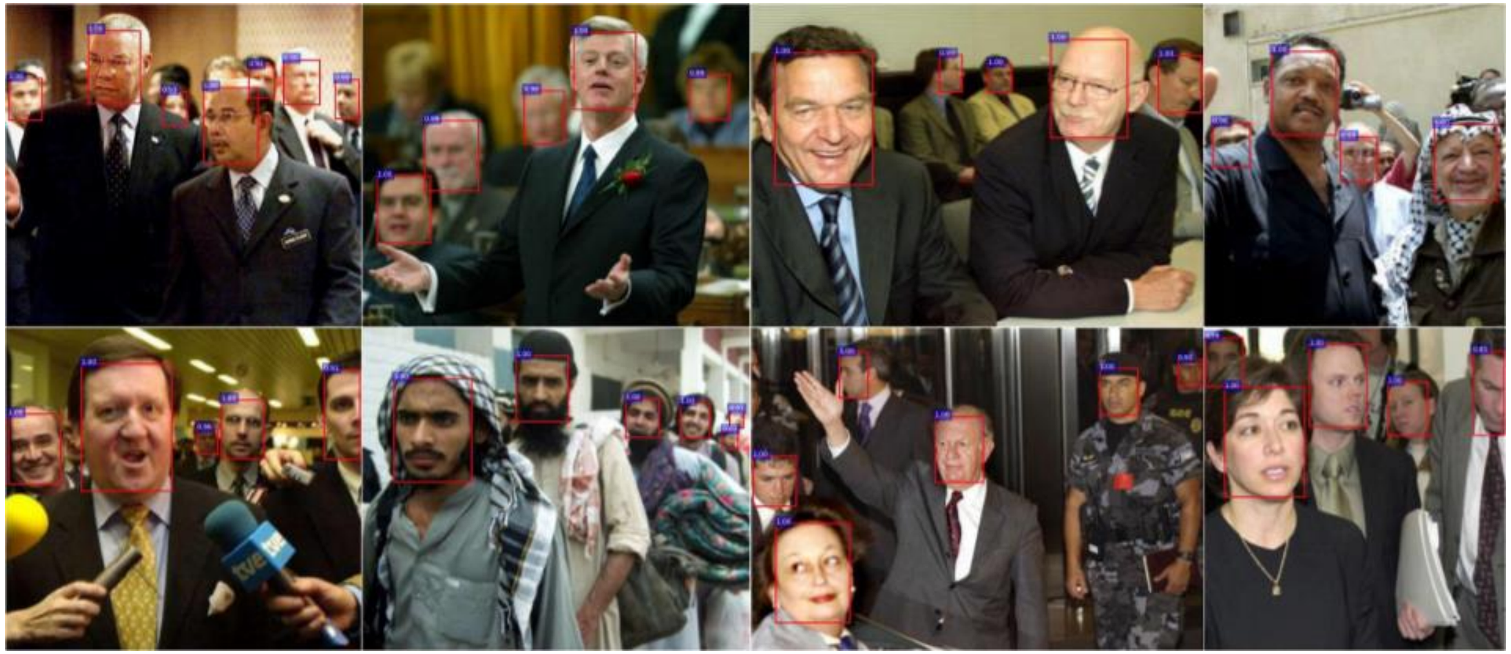
Trend: Small face in the wild
Real time in mobile devices
[Anti-spoof]

Companies: Megvii (Face++)
Sensetime
Google
Economic Companies
[Alibaba / Amazon / VISA / …]

I. Face Detection

B. A Real Example, FaceBoxes [2017-2018]:

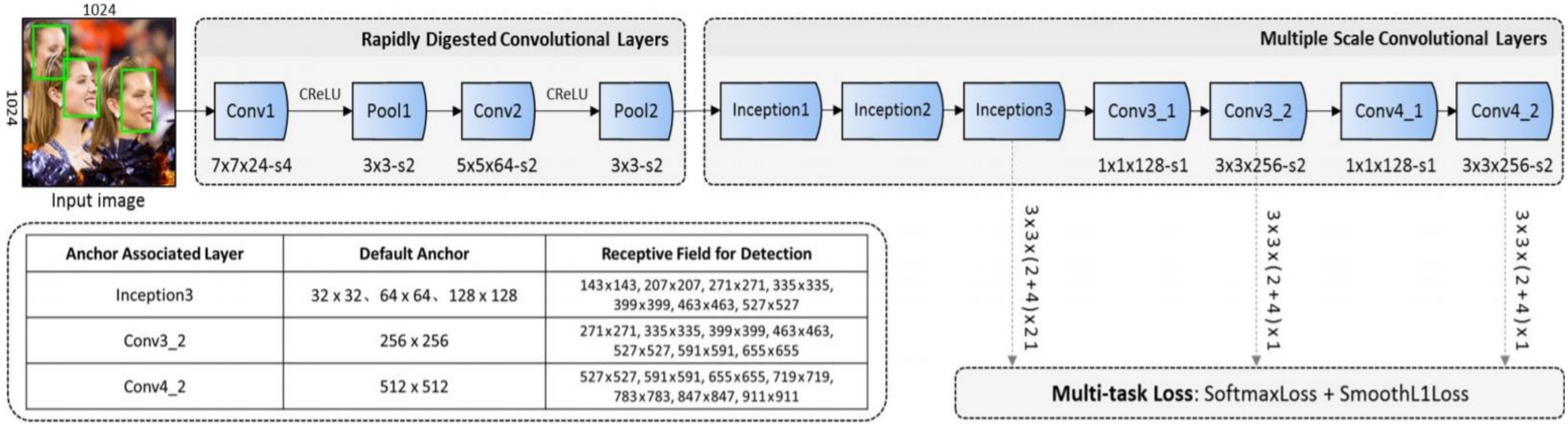
Features: Fast! Real-time on CPU
100+FPS on GPU
Practical & Acceptable



I. Face Detection

B. A Real Example, **FaceBoxes** [2017-2018]:

Structure:

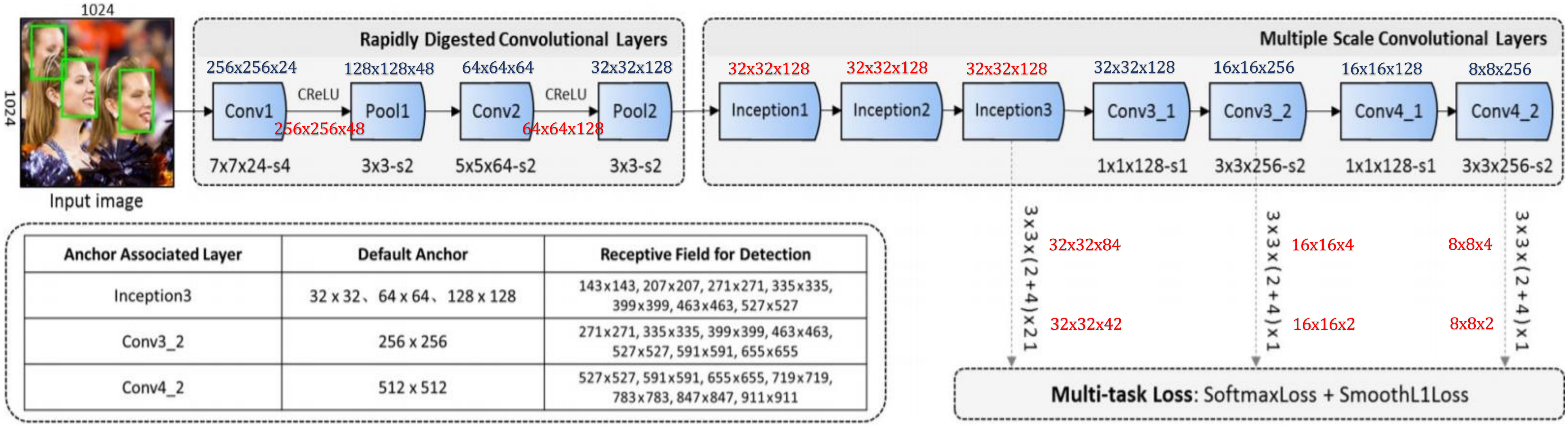


Anchor Associated Layer	Default Anchor	Receptive Field for Detection
Inception3	32 x 32、64 x 64、128 x 128	143x143, 207x207, 271x271, 335x335, 399x399, 463x463, 527x527
Conv3_2	256 x 256	271x271, 335x335, 399x399, 463x463, 527x527, 591x591, 655x655
Conv4_2	512 x 512	527x527, 591x591, 655x655, 719x719, 783x783, 847x847, 911x911

I. Face Detection

B. A Real Example, FaceBoxes [2017-2018]:

Structure:

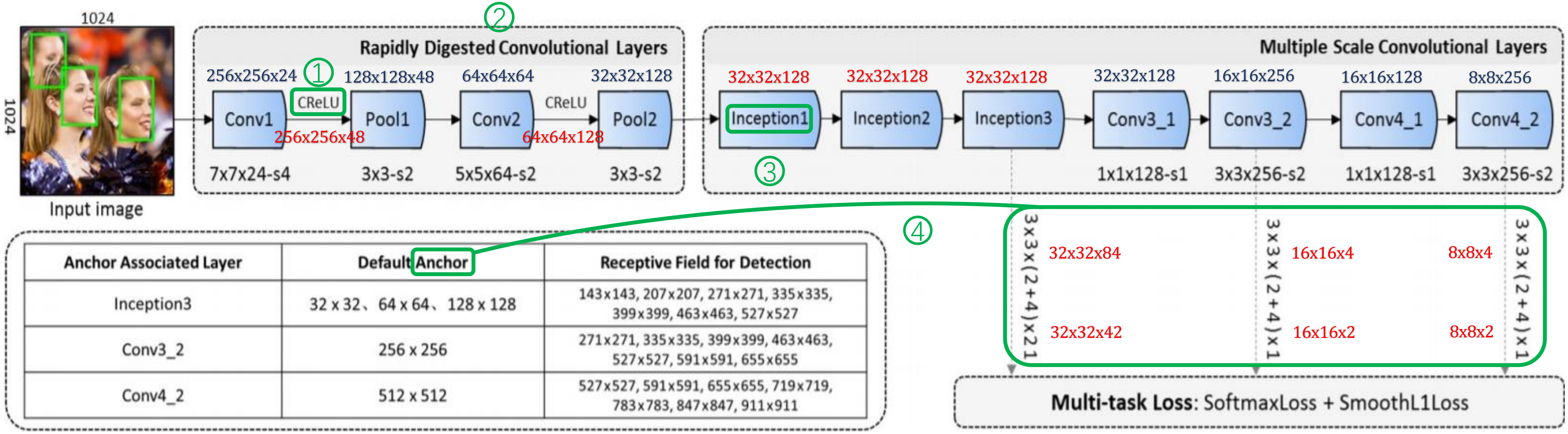


Anchor Associated Layer	Default Anchor	Receptive Field for Detection
Inception3	32 x 32、64 x 64、128 x 128	143x143, 207x207, 271x271, 335x335, 399x399, 463x463, 527x527
Conv3_2	256 x 256	271x271, 335x335, 399x399, 463x463, 527x527, 591x591, 655x655
Conv4_2	512 x 512	527x527, 591x591, 655x655, 719x719, 783x783, 847x847, 911x911

I. Face Detection

B. A Real Example, FaceBoxes [2017-2018]:

Structure:

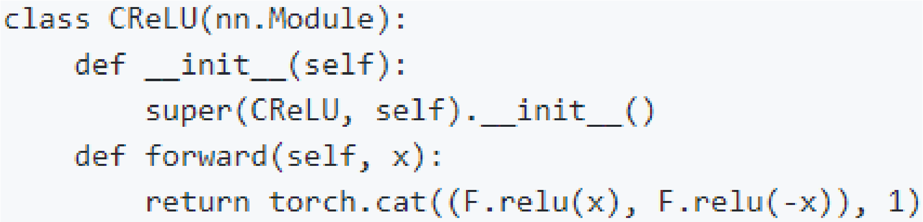


Anchor Associated Layer	Default Anchor	Receptive Field for Detection
Inception3	32 x 32、64 x 64、128 x 128	143x143, 207x207, 271x271, 335x335, 399x399, 463x463, 527x527
Conv3_2	256 x 256	271x271, 335x335, 399x399, 463x463, 527x527, 591x591, 655x655
Conv4_2	512 x 512	527x527, 591x591, 655x655, 719x719, 783x783, 847x847, 911x911

I. Face Detection

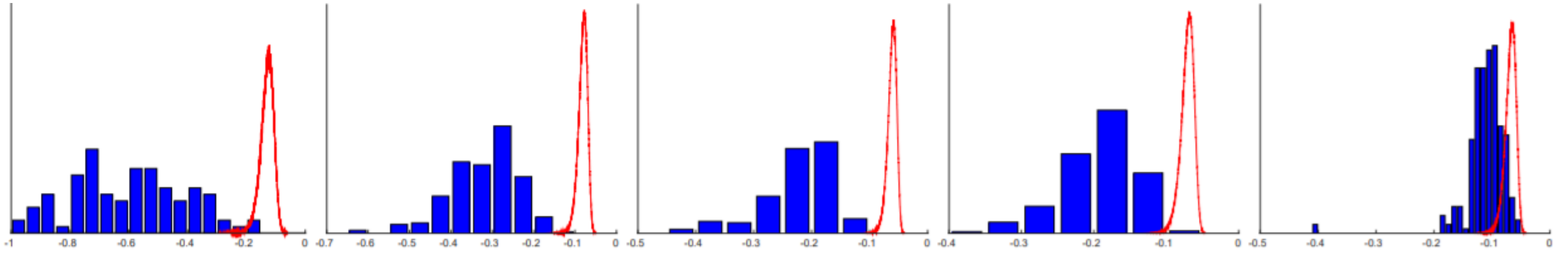
B. A Real Example, FaceBoxes [2017-2018]:

Structure: 1. CRelu [2016]



```
class CReLU(nn.Module):
    def __init__(self):
        super(CReLU, self).__init__()
    def forward(self, x):
        return torch.cat((F.relu(x), F.relu(-x)), 1)
```

- **Reduce parameters / Accelerate system: $\frac{n}{2}$** filters, n channels
- **Filters of lower conv layers form pairs**

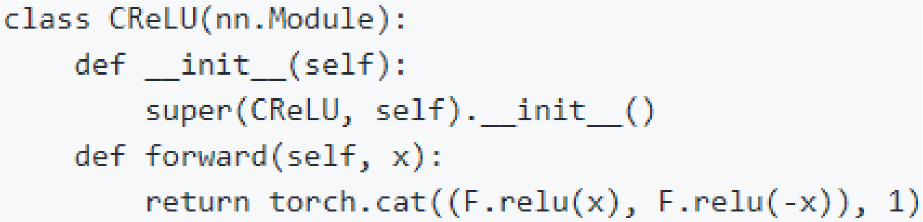


(a) conv1 (b) conv2 (c) conv3 (d) conv4 (e) conv5

I. Face Detection

B. A Real Example, FaceBoxes [2017-2018]:

Structure: 1. CReLU [2016]



```
class CReLU(nn.Module):
    def __init__(self):
        super(CReLU, self).__init__()
    def forward(self, x):
        return torch.cat((F.relu(x), F.relu(-x)), 1)
```

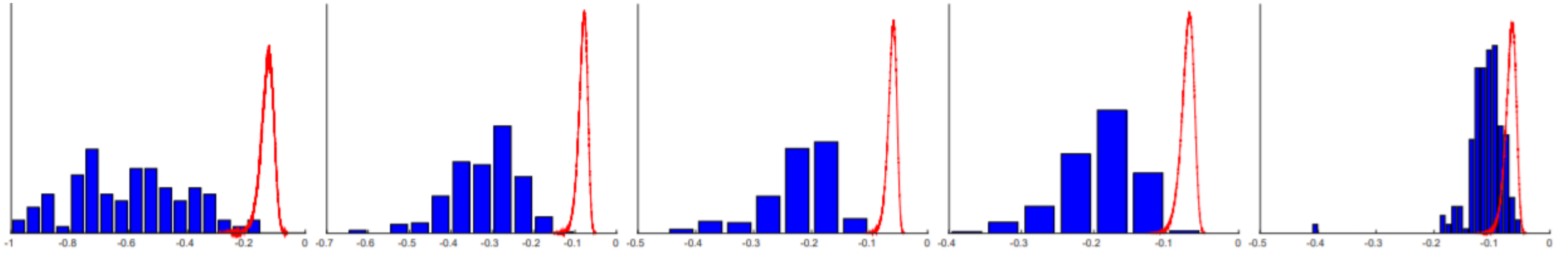
- **Don't be bewildered by CELU**

  CELU: continuously differentiable exponential linear units

  CReLU: concatenated rectified linear units

- **Reduce parameters / Accelerate system:** $\frac{n}{2}$ filters, n channels

- **Filters of lower conv layers form pairs**

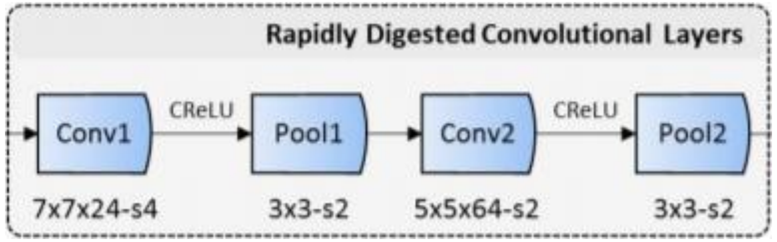


(a) conv1 (b) conv2 (c) conv3 (d) conv4 (e) conv5

I. Face Detection

B. A Real Example, FaceBoxes [2017-2018]:

Structure: 2. Rapid Digested Convolutional Layers

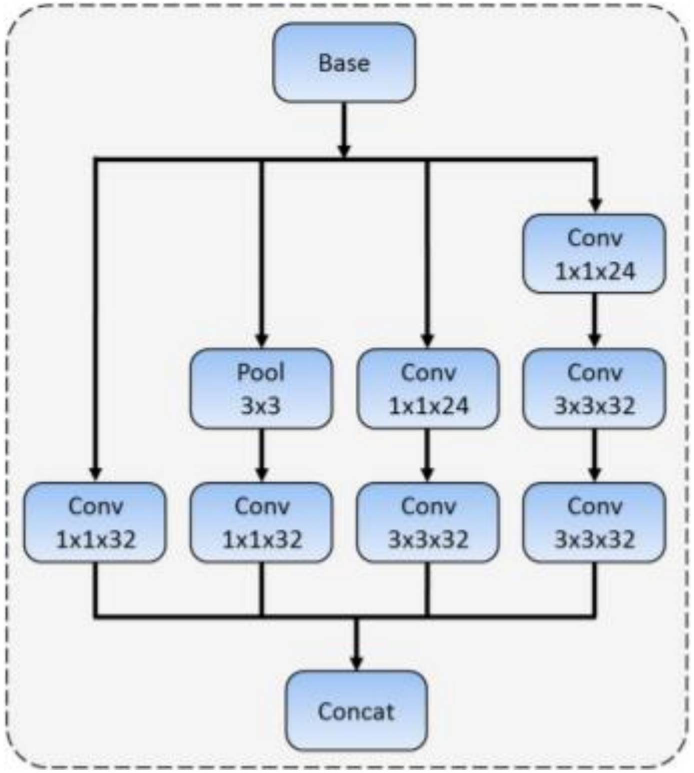


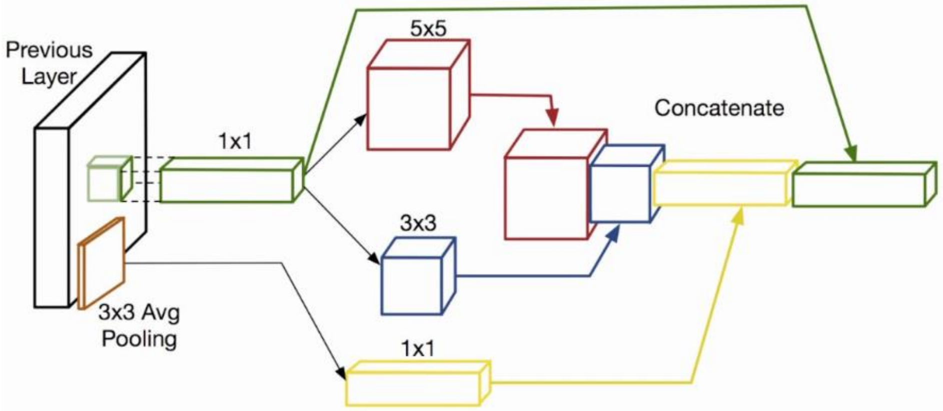
- Rapidly shrink the input size: 7x7
- Carefully choose filter size: bigger for getting info; smaller for speeding up
- Reducing channel number: CReLU

I. Face Detection

B. A Real Example, FaceBoxes [2017-2018]:

Structure: 3. Inception





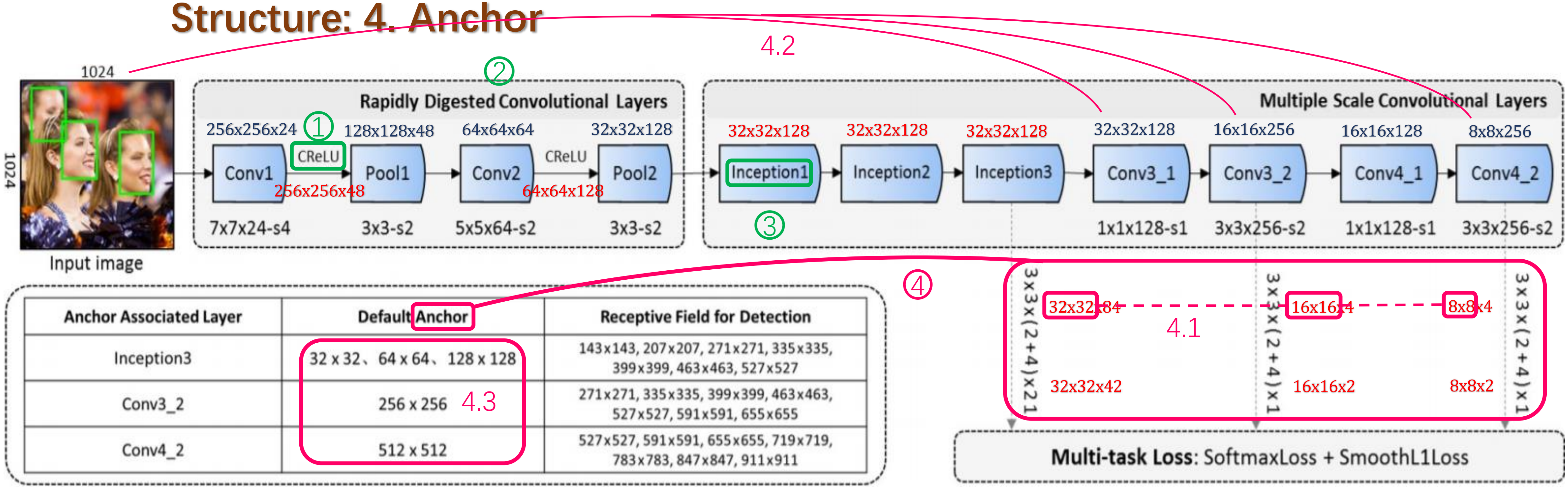
- **1x1 Conv:**
  reduce dimension

- **Inception Module:**
  multiple resolution

I. Face Detection

B. A Real Example, FaceBoxes [2017-2018]:

Structure: 4. Anchor

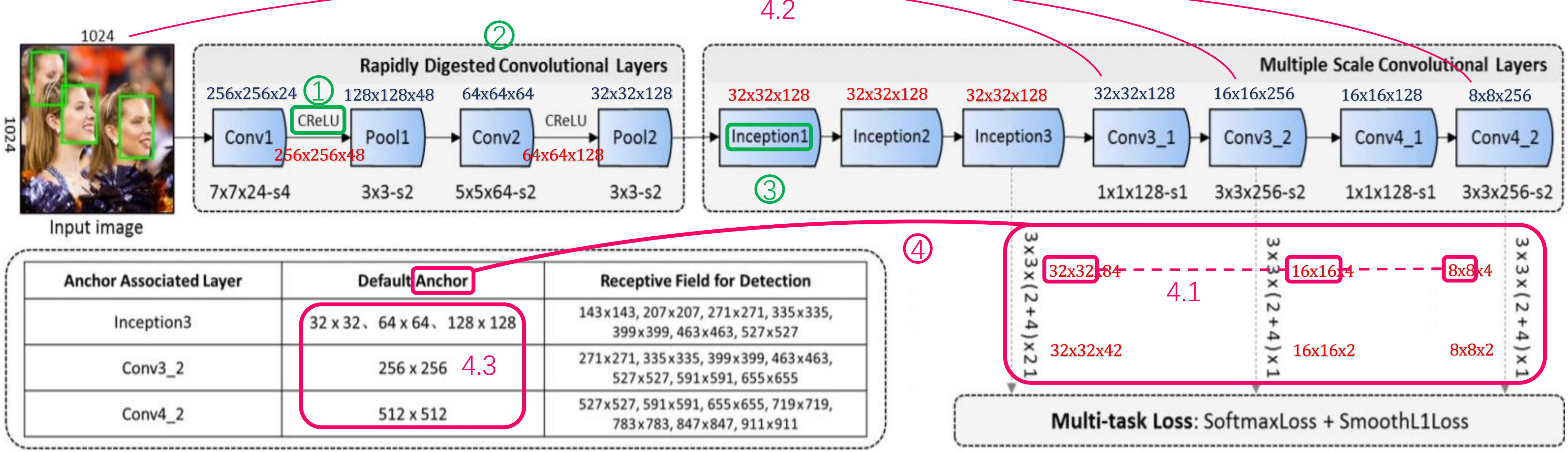


Anchor Associated Layer	Default Anchor	Receptive Field for Detection
Inception3	32 x 32、64 x 64、128 x 128	143x143, 207x207, 271x271, 335x335, 399x399, 463x463, 527x527
Conv3_2	256 x 256	271x271, 335x335, 399x399, 463x463, 527x527, 591x591, 655x655
Conv4_2	512 x 512	527x527, 591x591, 655x655, 719x719, 783x783, 847x847, 911x911

I. Face Detection

B. A Real Example, FaceBoxes [2017-2018]:

Structure: 4. Anchor



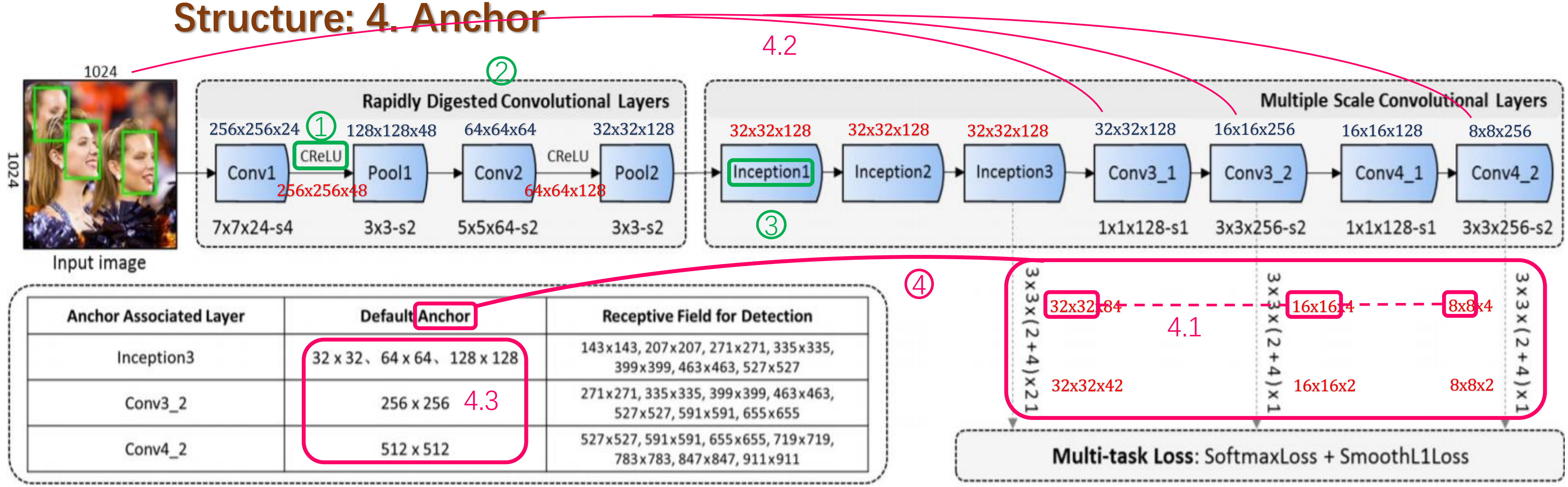
Anchor Associated Layer	Default Anchor	Receptive Field for Detection
Inception3	32 x 32、64 x 64、128 x 128	143x143, 207x207, 271x271, 335x335, 399x399, 463x463, 527x527
Conv3_2	256 x 256	271x271, 335x335, 399x399, 463x463, 527x527, 591x591, 655x655
Conv4_2	512 x 512	527x527, 591x591, 655x655, 719x719, 783x783, 847x847, 911x911

- 4.1 feature maps: 3 scales: 32 x 32, 16 x 16, 8 x 8

I. Face Detection

B. A Real Example, FaceBoxes [2017-2018]:

Structure: 4. Anchor



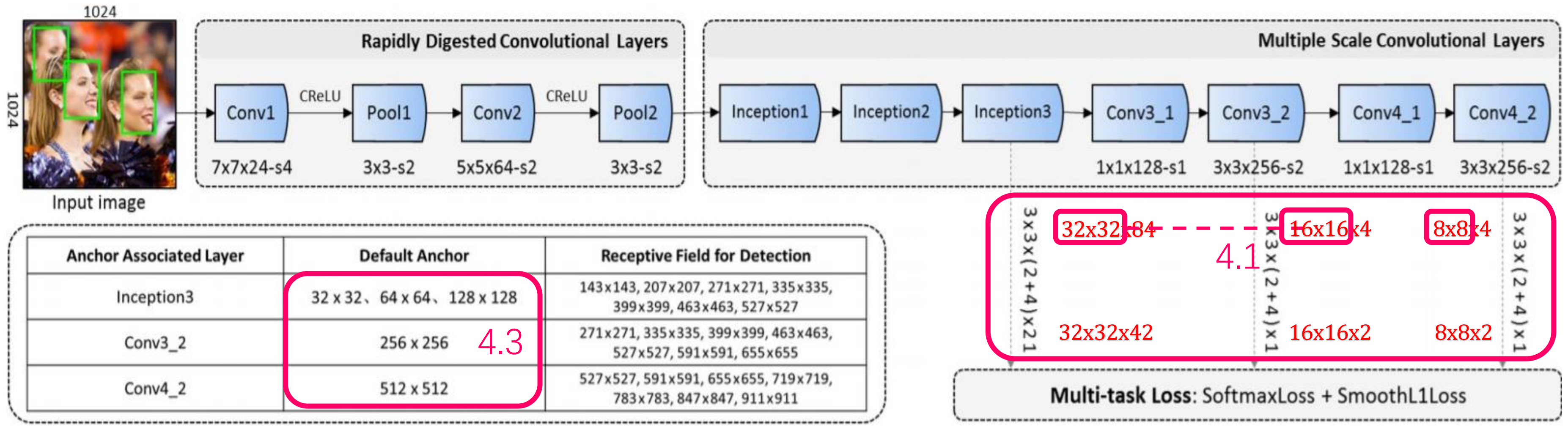
Anchor Associated Layer	Default Anchor	Receptive Field for Detection
Inception3	32 x 32、64 x 64、128 x 128	143x143, 207x207, 271x271, 335x335, 399x399, 463x463, 527x527
Conv3_2	256 x 256	271x271, 335x335, 399x399, 463x463, 527x527, 591x591, 655x655
Conv4_2	512 x 512	527x527, 591x591, 655x655, 719x719, 783x783, 847x847, 911x911

- **4.2 skips: 1024 / 32 = 32; 1024 / 64 = 16; 1024 / 128 = 8**

I. Face Detection

B. A Real Example, **FaceBoxes** [2017-2018]:

Structure: 4. Anchor



Anchor Associated Layer	Default Anchor	Receptive Field for Detection
Inception3	32 x 32、64 x 64、128 x 128	143x143, 207x207, 271x271, 335x335, 399x399, 463x463, 527x527
Conv3_2	256 x 256	271x271, 335x335, 399x399, 463x463, 527x527, 591x591, 655x655
Conv4_2	512 x 512	527x527, 591x591, 655x655, 719x719, 783x783, 847x847, 911x911

- **4.3 min_sizes [anchor size]:**

min_sizes	skips	feature maps
[32x32, 64x64, 128x128]	32	32x32
[256x256]	64	16x16
[512x512]	128	8x8

I. Face Detection

B. A Real Example, FaceBoxes [2017-2018]:

Structure: 4. Anchor

- **4.3 min_sizes [anchor size]:**

min_sizes	skips	feature maps
[32x32, 64x64, 128x128]	32	32x32
[256x256]	64	16x16
[512x512]	128	8x8

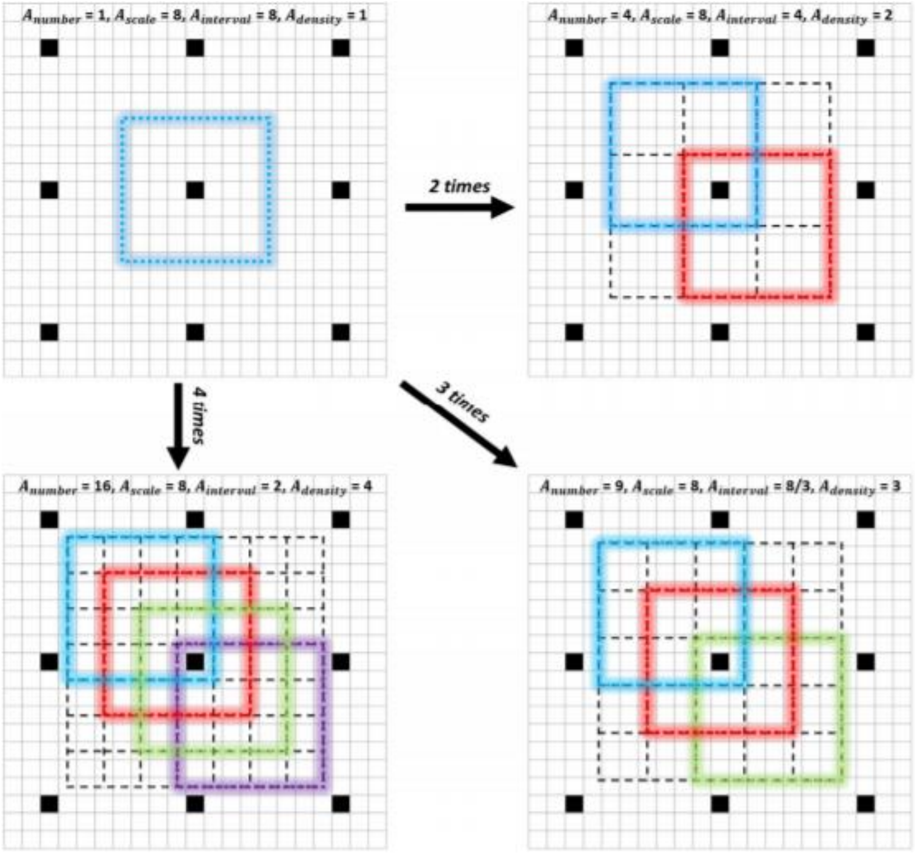
I. Face Detection

B. A Real Example, FaceBoxes [2017-2018]:

Structure: 4. Anchor

- **4.3 min_sizes [anchor size]:**

min_sizes	skips	feature maps
[32x32, 64x64, 128x128]	32	32x32
[256x256]	64	16x16
[512x512]	128	8x8


A_{number} = 1, A_{scale} = 8, $A_{interval}$ = 8, $A_{density}$ = 1
2 times
A_{number} = 4, A_{scale} = 8, $A_{interval}$ = 4, $A_{density}$ = 2
4 times
3 times
A_{number} = 16, A_{scale} = 8, $A_{interval}$ = 2, $A_{density}$ = 4
A_{number} = 9, A_{scale} = 8, $A_{interval}$ = 8/3, $A_{density}$ = 3

I. Face Detection

C. Code of FaceBoxes

- **whole structure**
- **prior_box**
- **faceboxes**
- **multibox_loss**

Latest Advanced Code

- Accelerated by CPP
- Structure Refined by SSD
- Added Facial Keypoints

I. Face Detection

D. Advices for Projects

- **BlazeFace**: Google 2019 [Assembled in Google Mediapipe]
  Faster than Face Boxes; real-time in mobile devices
  No official code released. **Be No.1 in the world!**

- **FaceBoxes Revised**:
  Input size of official FaceBoxes.PyTorch is fixed.
  Anchor sizes are fixed.
  Let's fix it!

II. Face Recognition

II. Face Recognition

E. What Is Recognition

- **Object**
- **Expression**
- **Face**

II. Face Recognition

E. What Is Recognition

- **Object**

Classification

- **Expression**
- **Face**

II. Face Recognition

E. What Is Recognition

• **Object**	Classification
• **Expression**	Classification
• **Face**	

II. Face Recognition

E. What Is Recognition

• **Object**	Classification
• **Expression**	Classification/Regression
• **Face**	

II. Face Recognition

E. What Is Recognition

- **Object** Classification

AffecNet

- **Expression** Classification/Regression

Aff-Wild

- **Face**

II. Face Recognition

E. What Is Recognition

• **Object**	Classification	
• **Expression**	Classification/Regression	AffecNet Aff-Wild
• **Face**	“Recognition”	

II. Face Recognition

E. What Is Recognition

• **Object**	Classification	
		AffecNet
• **Expression**	Classification/Regression	
		Aff-Wild
• **Face**	“Recognition”	

Recognition: The procedure of extracting wanted info from data

II. Face Recognition

F. Face Recognition Procedure

• **Object**	Classification
	AffecNet
• **Expression**	Classification/Regression
	Aff-Wild
• **Face**	"Recognition"

Let's see this "real recognition" in algorithm level

Recognition: The procedure of extracting wanted info from data

II. Face Recognition

E. What Is Recognition

- **Object** — Classification

AffecNet

- **Expression** — Classification/Regression

Aff-Wild

- **Face** — “Recognition”

Let’s see this “real recognition” in algorithm level

Recognition in Algorithm — **Extract features & Retrieve**

II. Face Recognition

F. Face Recognition Procedure

F0. Two Steps

Feature Database + Face Recognition

II. Face Recognition

F. Face Recognition Procedure

F1. Construct the Database

0. Face Detection

1. Facial Feature Extraction

2. Database

II. Face Recognition

F. Face Recognition Procedure

F1. Construct the Database

0. Face Detection

1. Facial Feature Extraction

2. Database

- Data preparation
- Collected under control

II. Face Recognition

F. Face Recognition Procedure

F1. Construct the Database

0. Face Detection

1. Facial Feature Extraction

2. Database

- Face represents: Vector
- How to get: VGG / Resnet / ...
- Algorithm: classification

II. Face Recognition

F. Face Recognition Procedure

F1. Construct the Database

0. Face Detection

1. Facial Feature Extraction

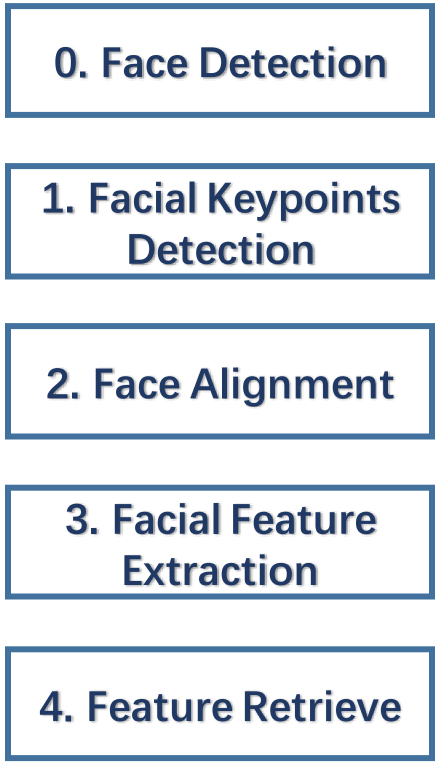
2. Database

- Not interested in building
- Mad of retrieving

II. Face Recognition

F. Face Recognition Procedure

F2. Face Recognition



0. Face Detection

1. Facial Keypoints Detection

2. Face Alignment

3. Facial Feature Extraction

4. Feature Retrieve

II. Face Recognition

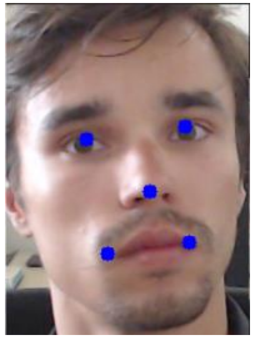
F. Face Recognition Procedure

F2. Face Recognition

- 0. Face Detection
- **1. Facial Keypoints Detection**
- 2. Face Alignment
- 3. Facial Feature Extraction
- 4. Feature Retrieve

➢ Most common dataset: 5 / 68 landmarks

➢ If you're enthusiastic: dlib (official + example)



II. Face Recognition

F. Face Recognition Procedure

F2. Face Recognition

0. Face Detection

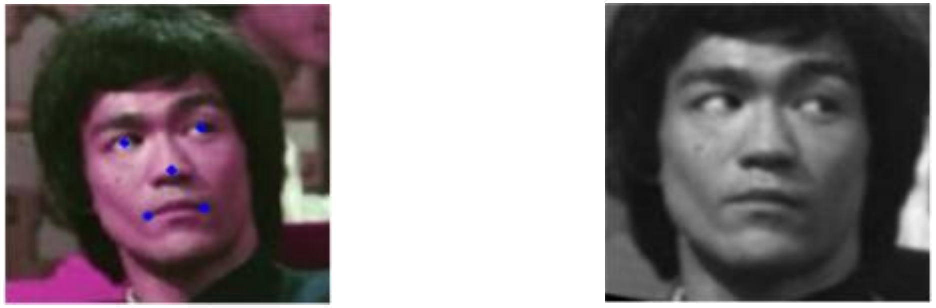
1. Facial Keypoints Detection

2. Face Alignment

3. Facial Feature Extraction

4. Feature Retrieve

- 人脸对齐
- Remove rotation / light influences / expression / …
- A practical example: dlib_alignment



II. Face Recognition

F. Face Recognition Procedure

F2. Face Recognition

0. Face Detection

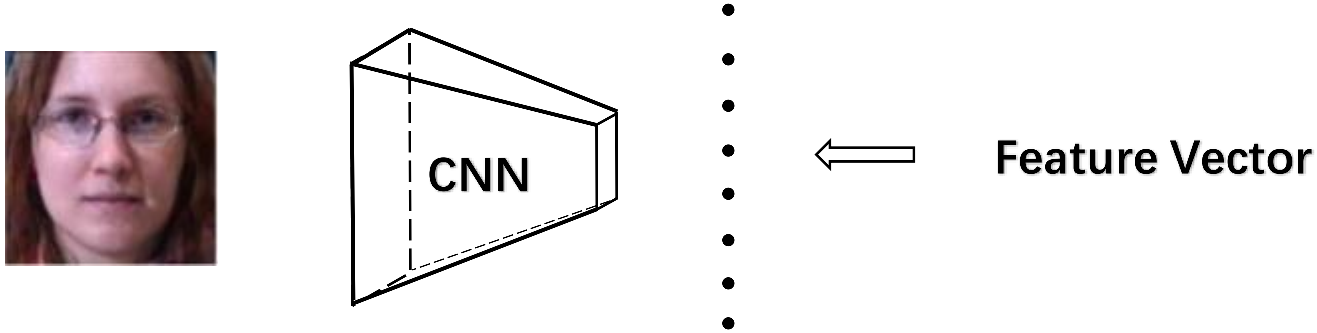
1. Facial Keypoints Detection

2. Face Alignment

3. Facial Feature Extraction

4. Feature Retrieve

- Historical: Geometrical Info / Handcrafted Feature
- Now: CNN-based feature (A trained network: VGG / ResNet…)
- A real example: LightCNN
- If you want more: TP-GAN/… + LightCNN + Perceptual Loss



II. Face Recognition

F. Face Recognition Procedure

Pink: Better to know

Green: Good to know

Red: Have to know

Blue: Code

F2. Face Recognition

0. Face Detection

1. Facial Keypoints Detection

2. Face Alignment

3. Facial Feature Extraction

4. Feature Retrieve

- Target: Retrieve vector most similar to our feature vector
- Problem: How to retrieve fast & accurate
- Methods: cosine distance (most common way)

  K-Dimensional Tree (# dimension is small)

  PCA (# of dimension is high)

  cluster / K-nary tree (large scale)

  ……

II. Face Recognition

G. A real example: LightCNN (2017)

G1. Framework

- Small — Medium — Large
- We use small as an e.g.

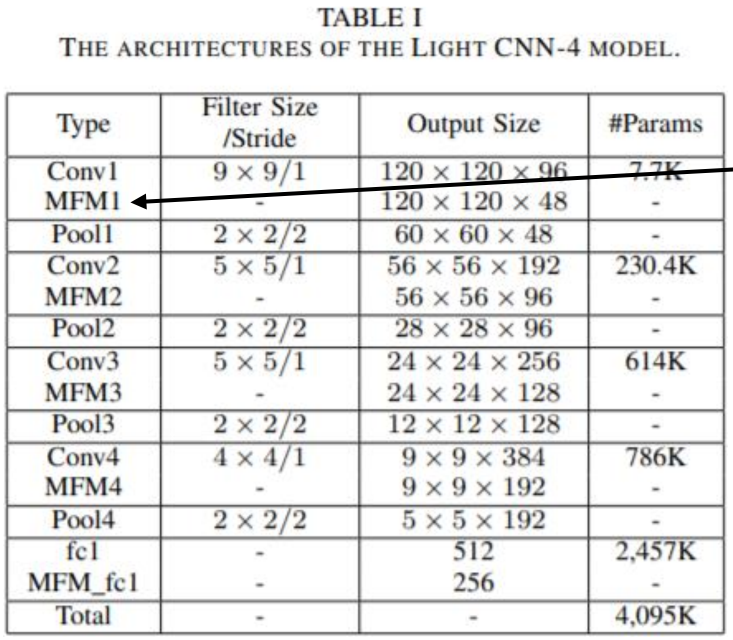
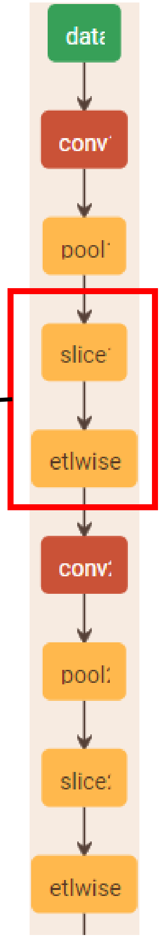
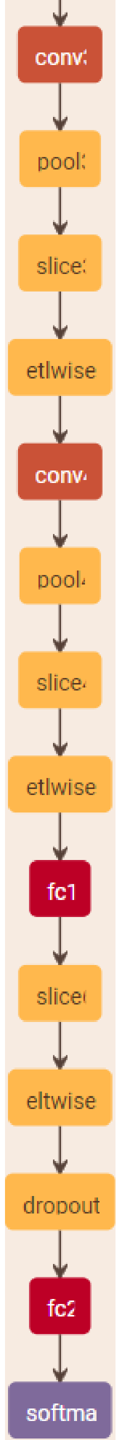


TABLE I
THE ARCHITECTURES OF THE LIGHT CNN-4 MODEL.

Type	Filter Size /Stride	Output Size	#Params
Conv1 MFM1	9 × 9/1 -	120 × 120 × 96 120 × 120 × 48	7.7K -
Pool1	2 × 2/2	60 × 60 × 48	-
Conv2 MFM2	5 × 5/1 -	56 × 56 × 192 56 × 56 × 96	230.4K -
Pool2	2 × 2/2	28 × 28 × 96	-
Conv3 MFM3	5 × 5/1 -	24 × 24 × 256 24 × 24 × 128	614K -
Pool3	2 × 2/2	12 × 12 × 128	-
Conv4 MFM4	4 × 4/1 -	9 × 9 × 384 9 × 9 × 192	786K -
Pool4	2 × 2/2	5 × 5 × 192	-
fc1 MFM_fc1	- -	512 256	2,457K -
Total	-	-	4,095K

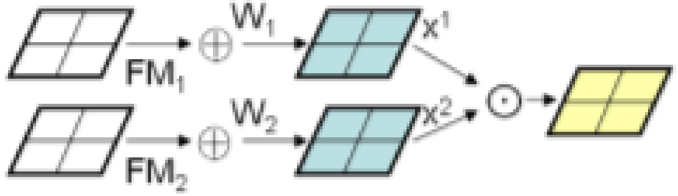




II. Face Recognition

G. A real example: LightCNN (2017)

G2. MFM: Max-Feature Map

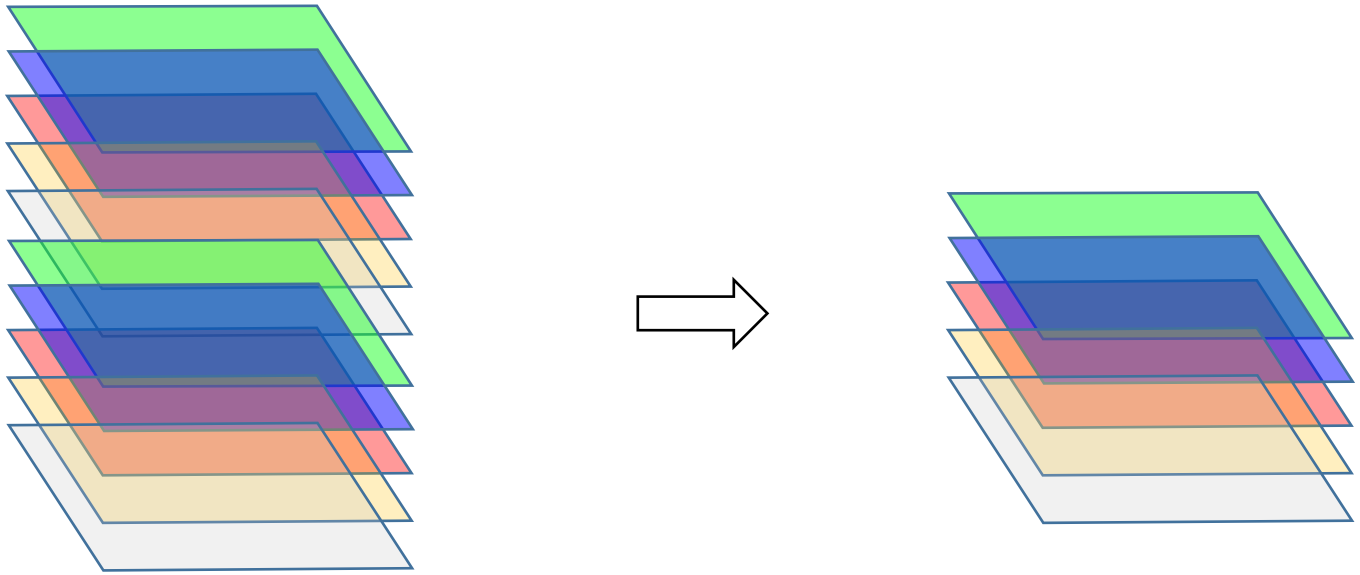


$$\hat{x}_{ij}^{k} = \max(x_{ij}^{k}, x_{ij}^{k+N})$$

II. Face Recognition

G. A real example: LightCNN (2017)

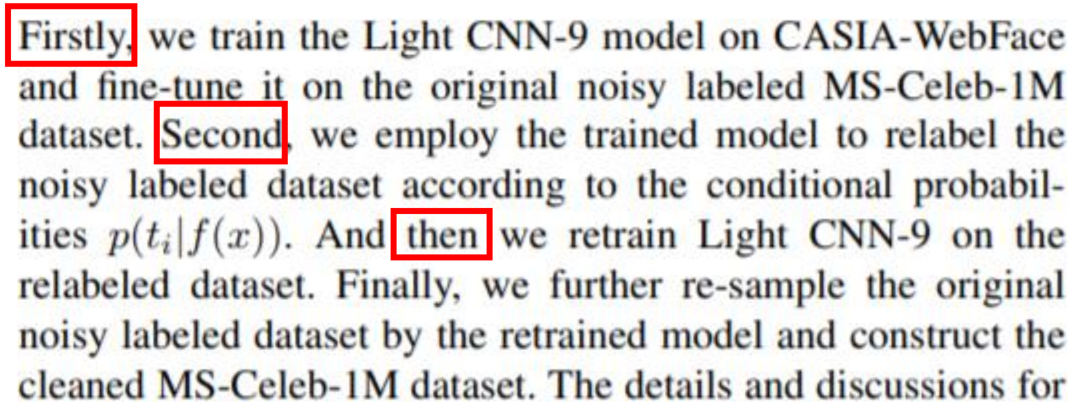
G2. MFM: Max-Feature Map



II. Face Recognition

G. A real example: LightCNN (2017)

G3. Semantic Bootstrapping for Noisy Labels

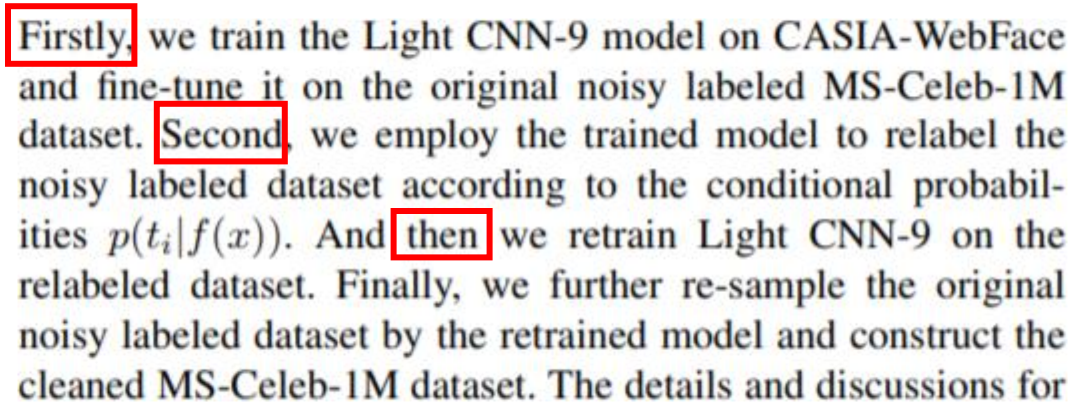


Firstly, we train the Light CNN-9 model on CASIA-WebFace and fine-tune it on the original noisy labeled MS-Celeb-1M dataset. Second, we employ the trained model to relabel the noisy labeled dataset according to the conditional probabilities $p(t_i|f(x))$. And then we retrain Light CNN-9 on the relabeled dataset. Finally, we further re-sample the original noisy labeled dataset by the retrained model and construct the cleaned MS-Celeb-1M dataset. The details and discussions for

II. Face Recognition

G. A real example: LightCNN (2017)

G3. Semantic Bootstrapping for Noisy Labels



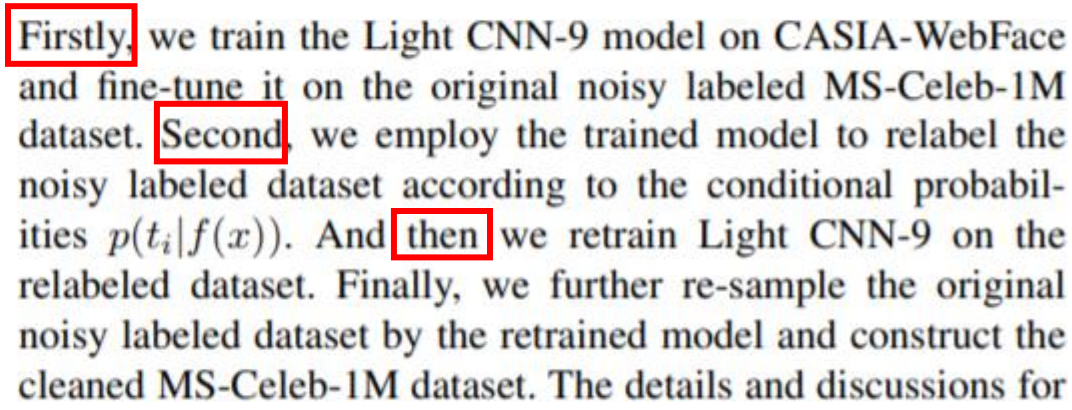
Firstly, we train the Light CNN-9 model on CASIA-WebFace and fine-tune it on the original noisy labeled MS-Celeb-1M dataset. Second, we employ the trained model to relabel the noisy labeled dataset according to the conditional probabilities $p(t_i|f(x))$. And then we retrain Light CNN-9 on the relabeled dataset. Finally, we further re-sample the original noisy labeled dataset by the retrained model and construct the cleaned MS-Celeb-1M dataset. The details and discussions for

- **How to deal with bad dataset**

II. Face Recognition

G. A real example: LightCNN (2017)

G3. Semantic Bootstrapping for Noisy Labels



Firstly, we train the Light CNN-9 model on CASIA-WebFace and fine-tune it on the original noisy labeled MS-Celeb-1M dataset. Second, we employ the trained model to relabel the noisy labeled dataset according to the conditional probabilities $p(t_i|f(x))$. And then we retrain Light CNN-9 on the relabeled dataset. Finally, we further re-sample the original noisy labeled dataset by the retrained model and construct the cleaned MS-Celeb-1M dataset. The details and discussions for

- **How to deal with bad dataset**

➢ Train

➢ Use it

➢ Train again