

CNN for CV

AI for CV Group
2019



Week 12. Image Transfer

Contents:

I. Image Segmentation

- A. FCN
- B. UNet/ENet
- C. Mask RCNN
- D. Developments

II. Image Style Transfer

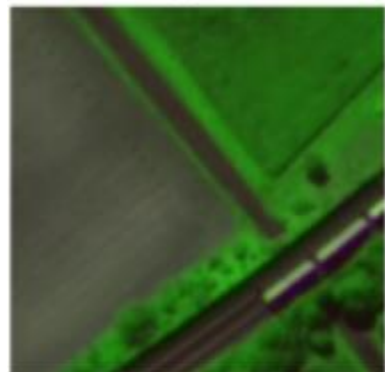
- E. Image Style Transfer: Perceptual Loss
- F. Feature Mimicking/Distillation

III. Other Applications

- G. Others

I. Image Segmentation

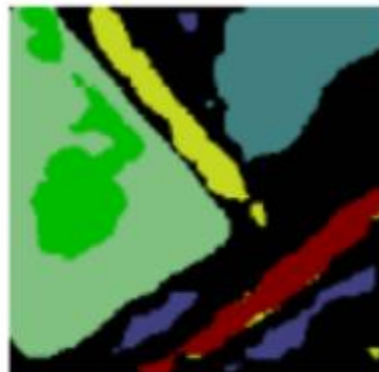
Input



Label



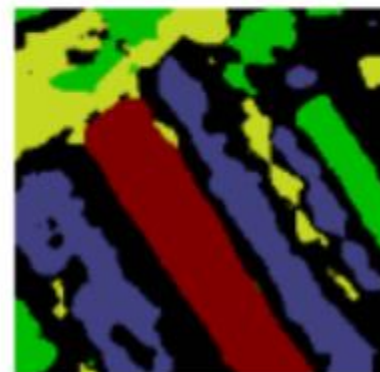
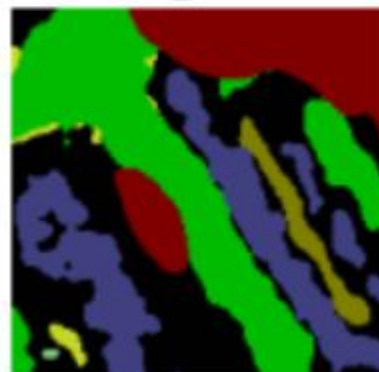
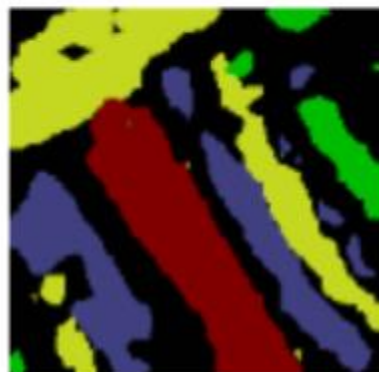
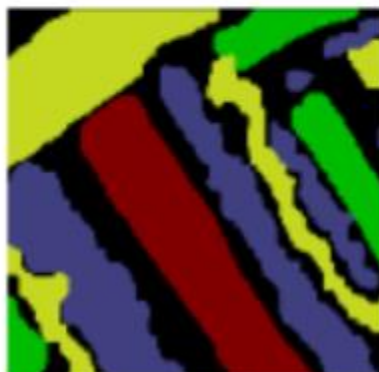
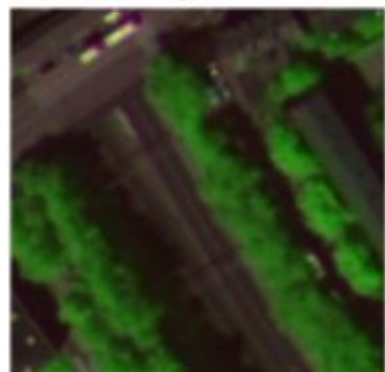
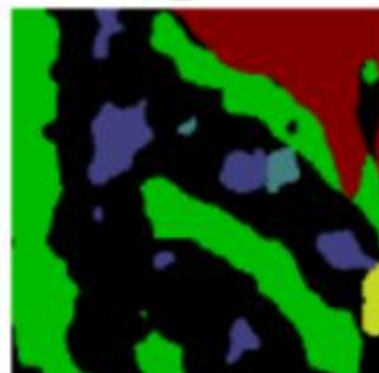
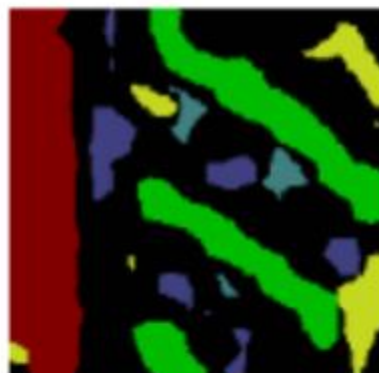
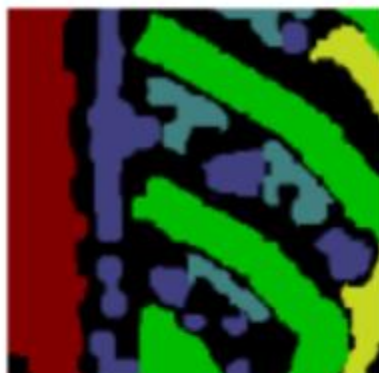
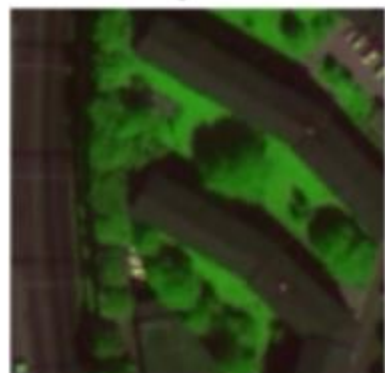
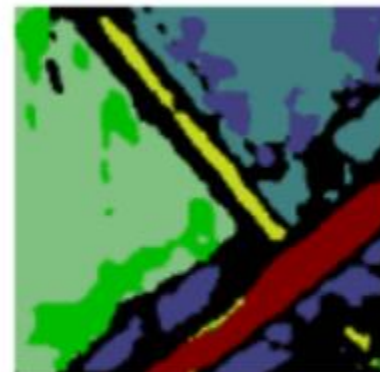
FCN

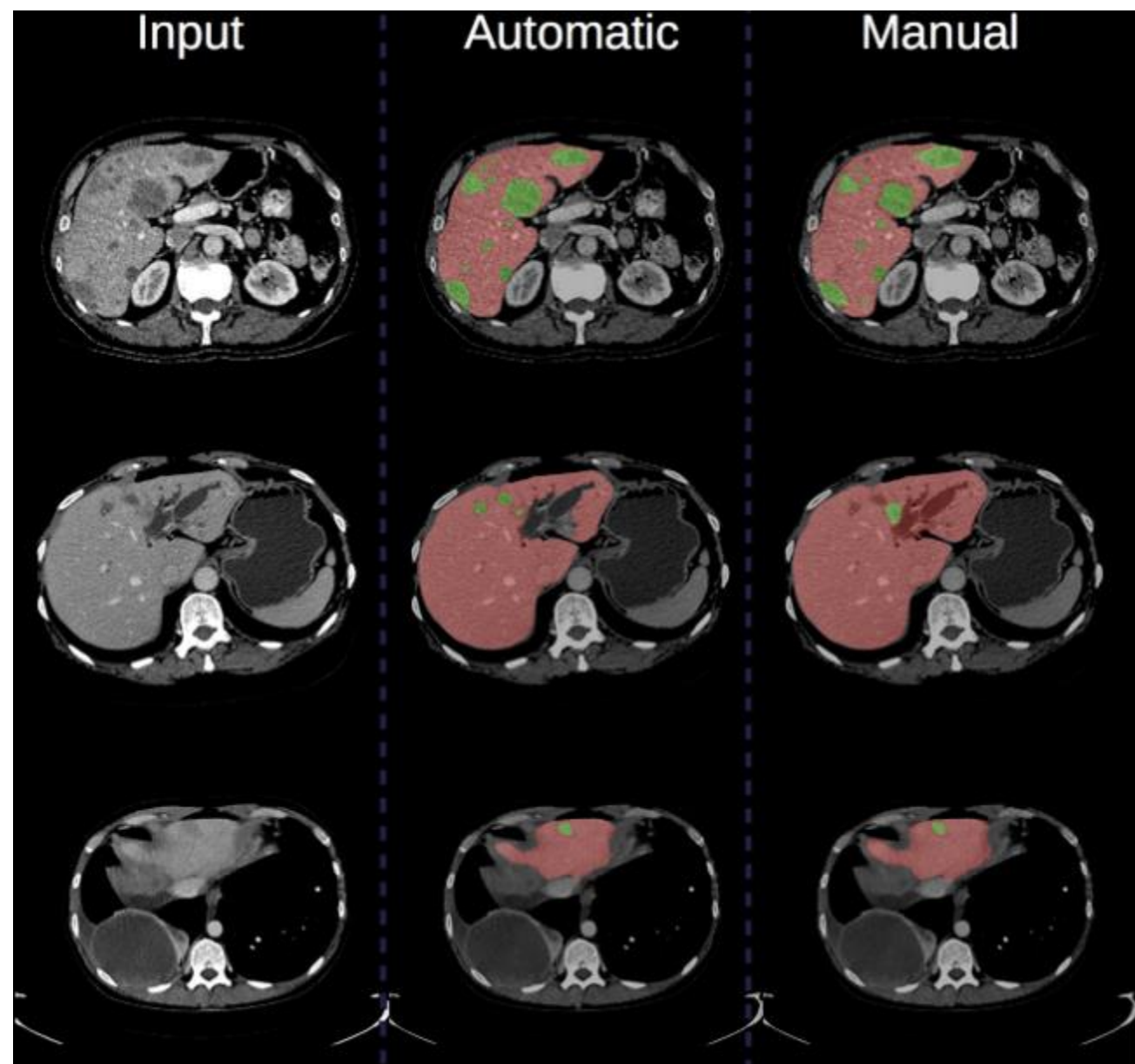


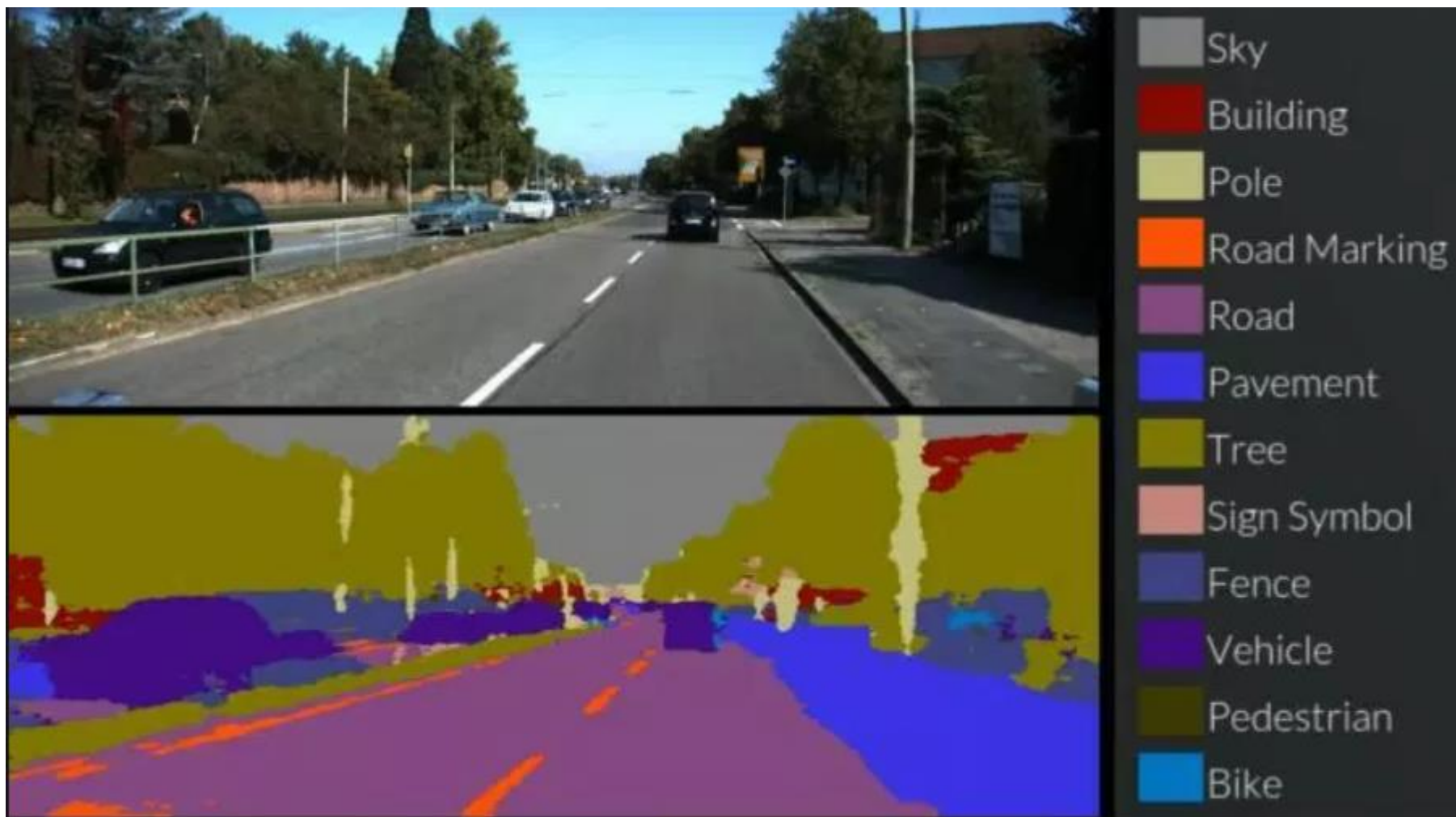
Segnet



UNet





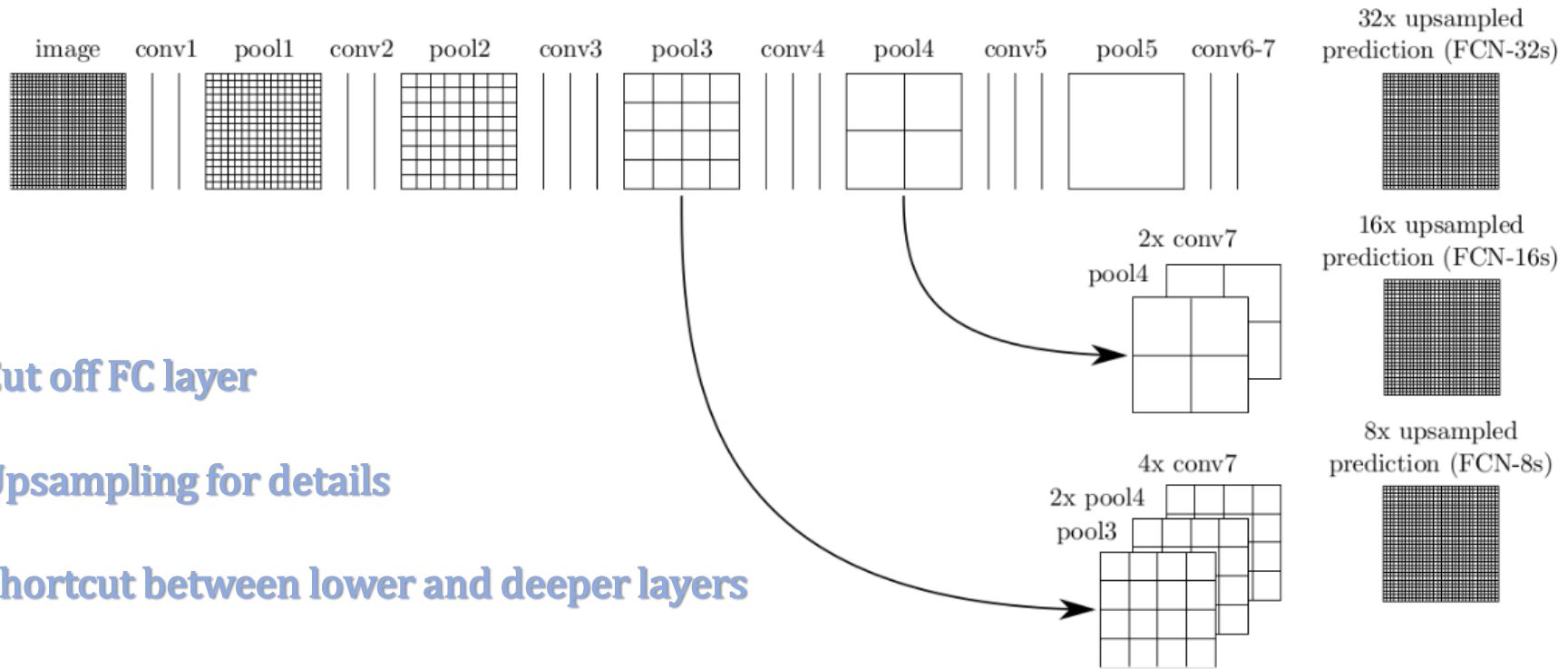


I. Image Segmentation

A. FCN [2015, Jonathan]:

- **3 Trends:**

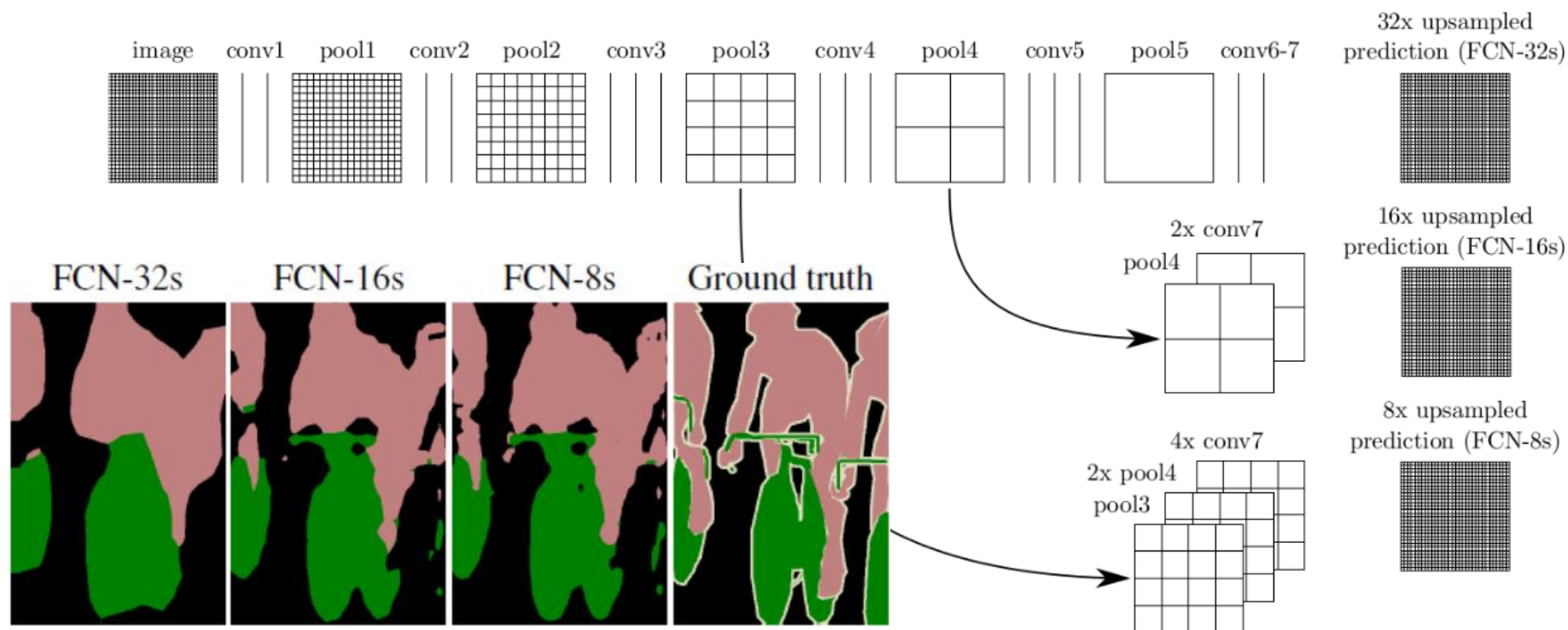
- **Cut off FC layer**
- **Upsampling for details**
- **Shortcut between lower and deeper layers**

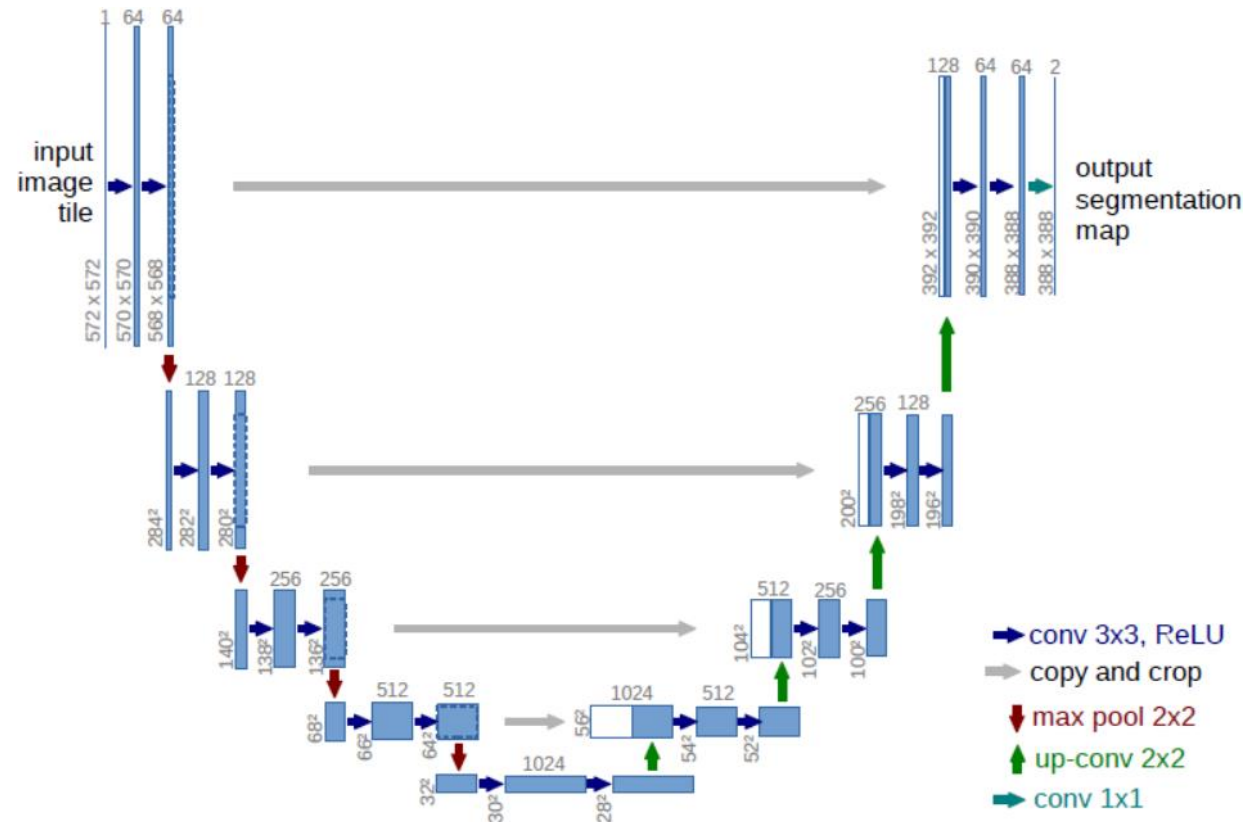


I. Image Segmentation

A. FCN [2015, Jonathan]:

- **Effects:**



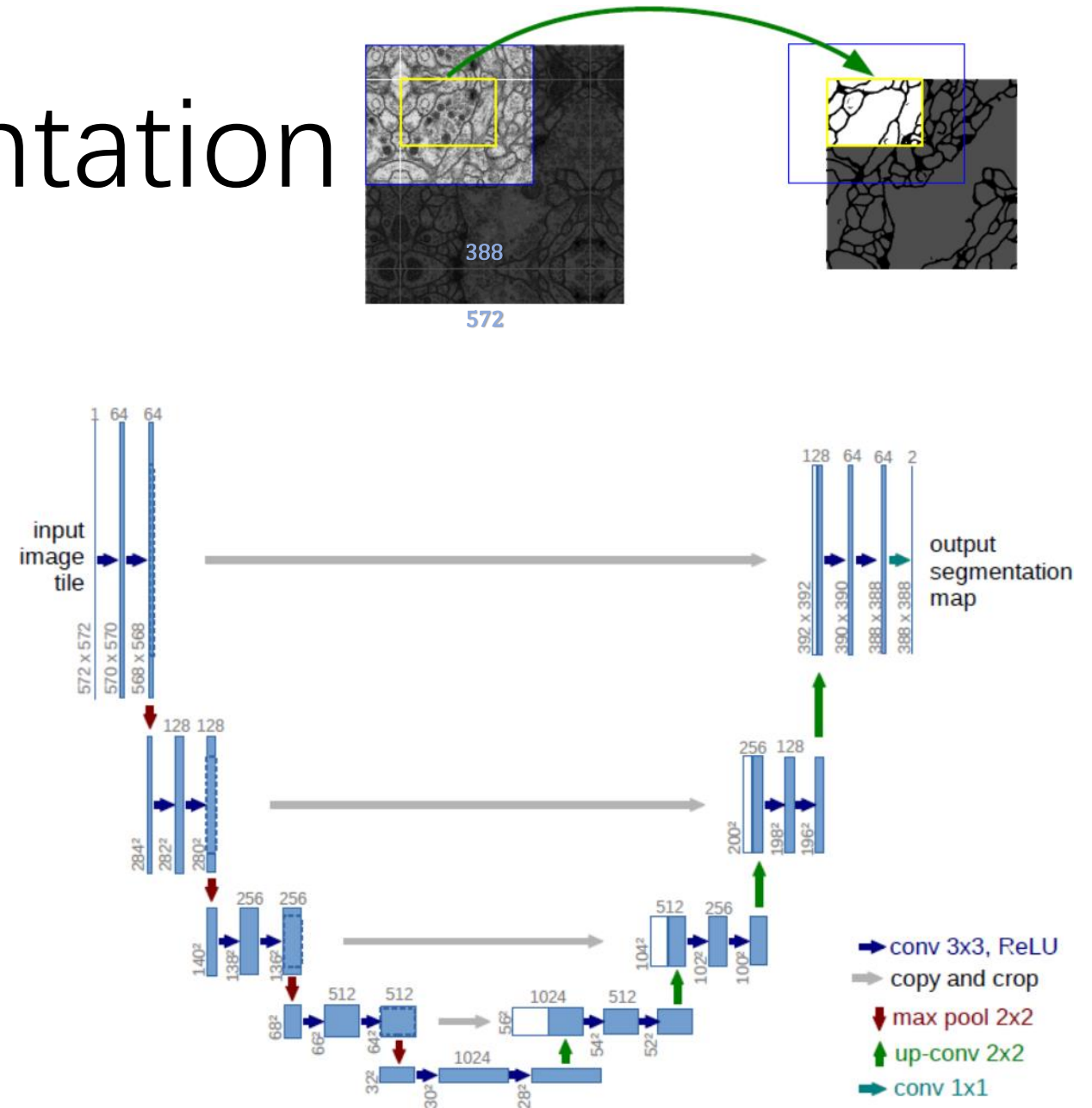


I. Image Segmentation

B. U-Net / E-Net:

B1. U-Net [2015, Olaf]

- **Tile Strategy:**
 - Use padding in mirroring way
 - To deal with borders
 - Just use valid parts
- **U-like Structure**
 - Combine lower & higher info
 - Remove FC layers

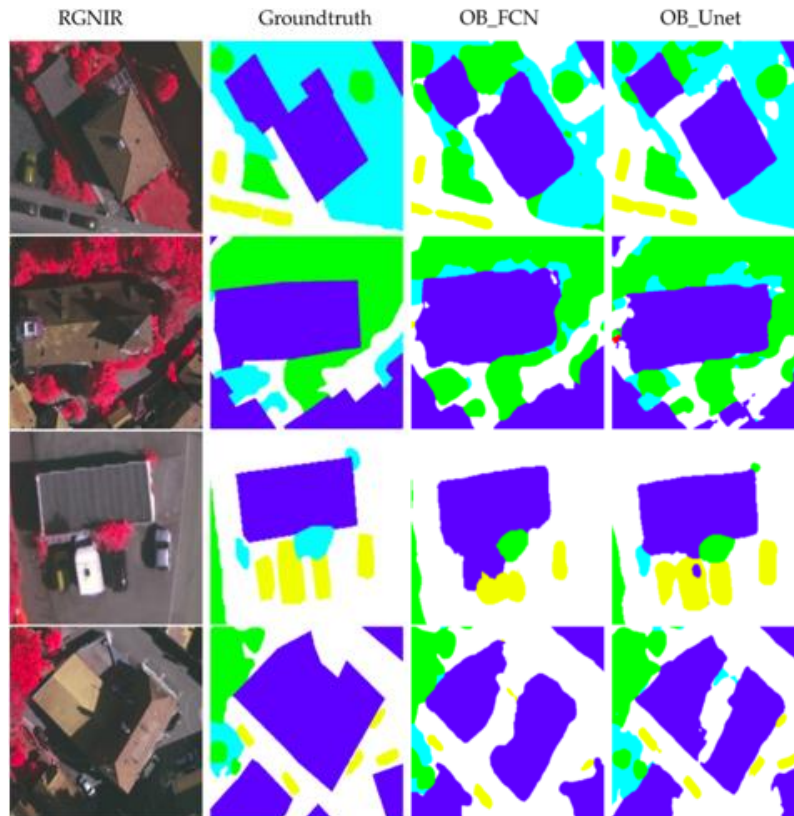


I. Image Segmentation

B. U-Net / E-Net:

B1. U-Net [2015, Olaf]

- **Effects:**

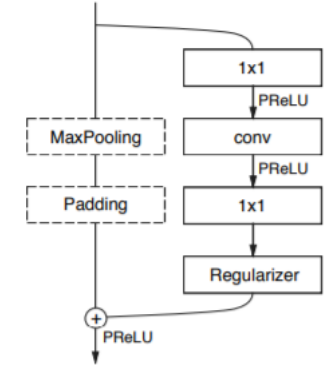
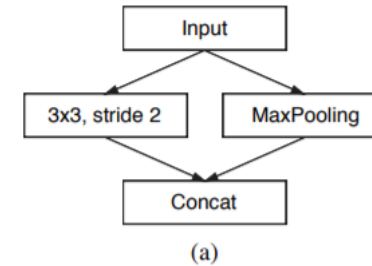
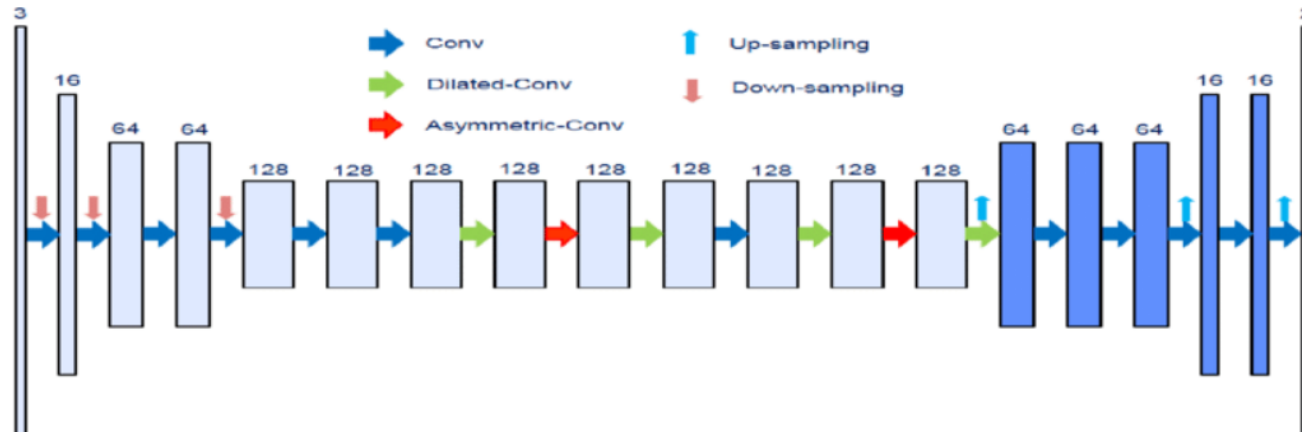


I. Image Segmentation

B. U-Net / E-Net:

B2. E-Net [2016, Adam]

- **Structure:**



Name	Type	Output size
initial		$16 \times 256 \times 256$
bottleneck1.0	downsampling	$64 \times 128 \times 128$
4 × bottleneck1.x		$64 \times 128 \times 128$
bottleneck2.0	downsampling	$128 \times 64 \times 64$
bottleneck2.1		$128 \times 64 \times 64$
bottleneck2.2	dilated 2	$128 \times 64 \times 64$
bottleneck2.3	asymmetric 5	$128 \times 64 \times 64$
bottleneck2.4	dilated 4	$128 \times 64 \times 64$
bottleneck2.5		$128 \times 64 \times 64$
bottleneck2.6	dilated 8	$128 \times 64 \times 64$
bottleneck2.7	asymmetric 5	$128 \times 64 \times 64$
bottleneck2.8	dilated 16	$128 \times 64 \times 64$
Repeat section 2, without bottleneck2.0		
bottleneck4.0	upsampling	$64 \times 128 \times 128$
bottleneck4.1		$64 \times 128 \times 128$
bottleneck4.2		$64 \times 128 \times 128$
bottleneck5.0	upsampling	$16 \times 256 \times 256$
bottleneck5.1		$16 \times 256 \times 256$
fullconv		$C \times 512 \times 512$

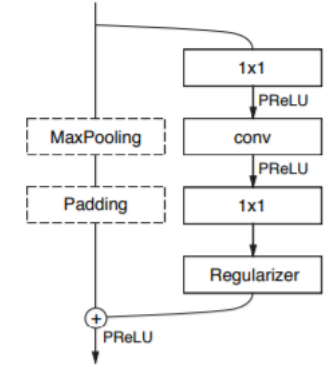
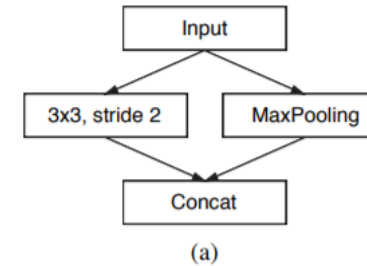
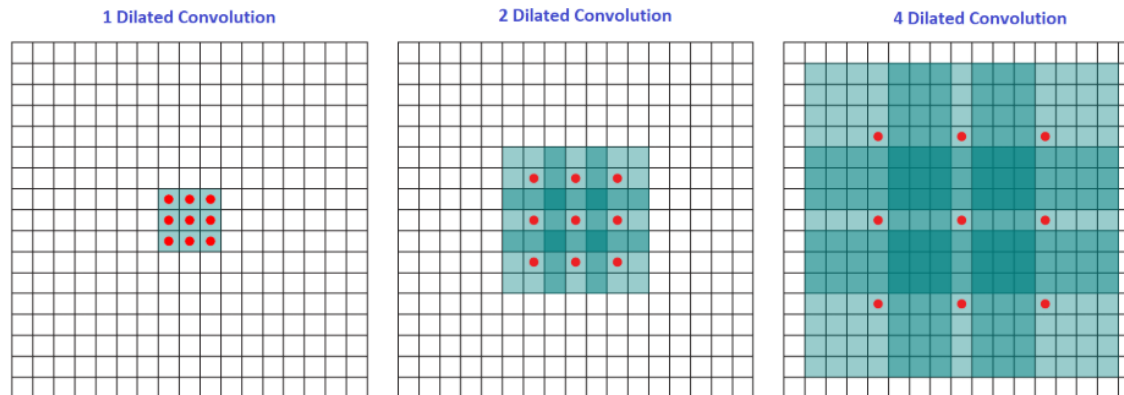
I. Image Segmentation

B. U-Net / E-Net:

B2. E-Net [2016, Adam]

- **Features:**

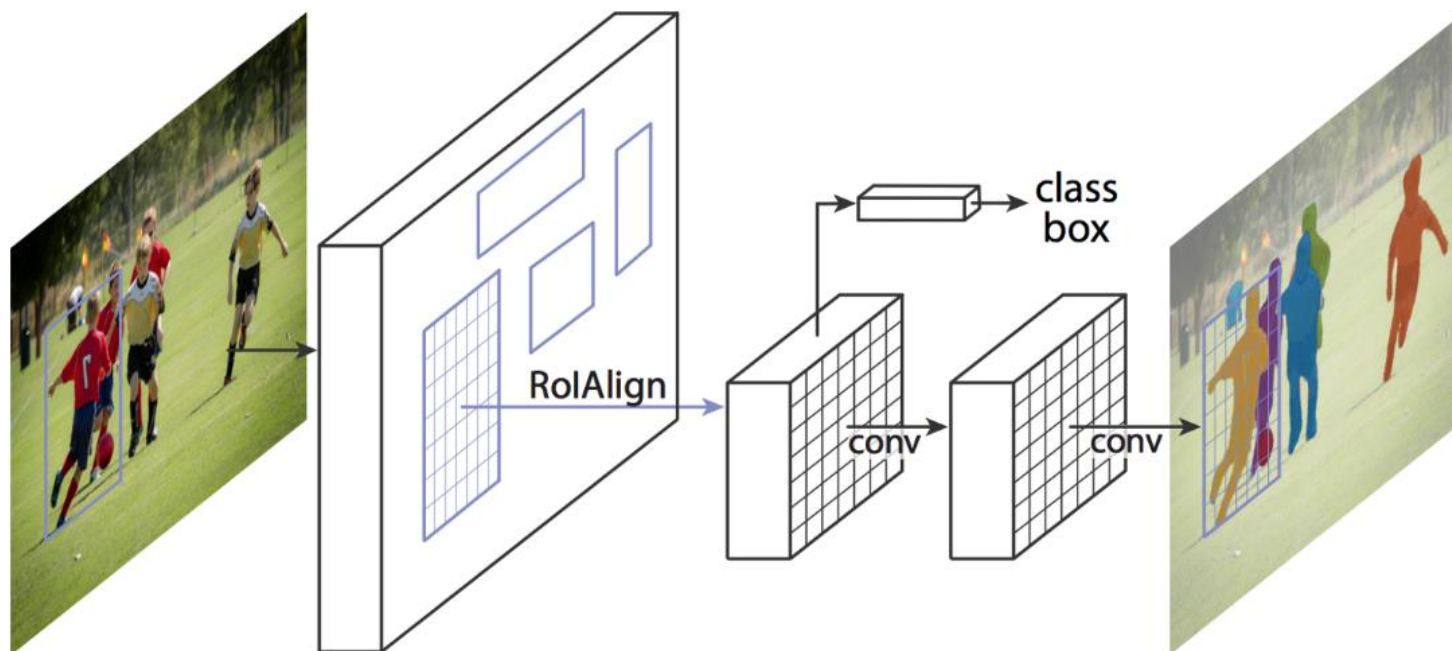
- Real Time: Altered bottleneck / Asymmetric conv
- Unbalanced Encoder & Decoder
- **Dilated conv**



Name	Type	Output size
initial		$16 \times 256 \times 256$
bottleneck1.0	downsampling	$64 \times 128 \times 128$
4 × bottleneck1.x		$64 \times 128 \times 128$
bottleneck2.0	downsampling	$128 \times 64 \times 64$
bottleneck2.1		$128 \times 64 \times 64$
bottleneck2.2	dilated 2	$128 \times 64 \times 64$
bottleneck2.3	asymmetric 5	$128 \times 64 \times 64$
bottleneck2.4	dilated 4	$128 \times 64 \times 64$
bottleneck2.5		$128 \times 64 \times 64$
bottleneck2.6	dilated 8	$128 \times 64 \times 64$
bottleneck2.7	asymmetric 5	$128 \times 64 \times 64$
bottleneck2.8	dilated 16	$128 \times 64 \times 64$
<i>Repeat section 2, without bottleneck2.0</i>		
bottleneck4.0	upsampling	$64 \times 128 \times 128$
bottleneck4.1		$64 \times 128 \times 128$
bottleneck4.2		$64 \times 128 \times 128$
bottleneck5.0	upsampling	$16 \times 256 \times 256$
bottleneck5.1		$16 \times 256 \times 256$
fullconv		$C \times 512 \times 512$

I. Image Segmentation

C. Mask-RCNN [2017, He]:

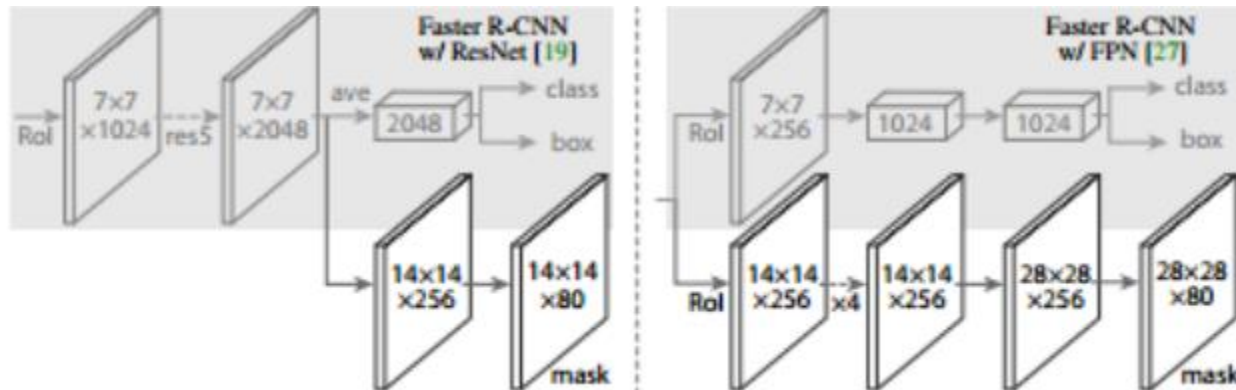


I. Image Segmentation

C. Mask-RCNN [2017, He]:

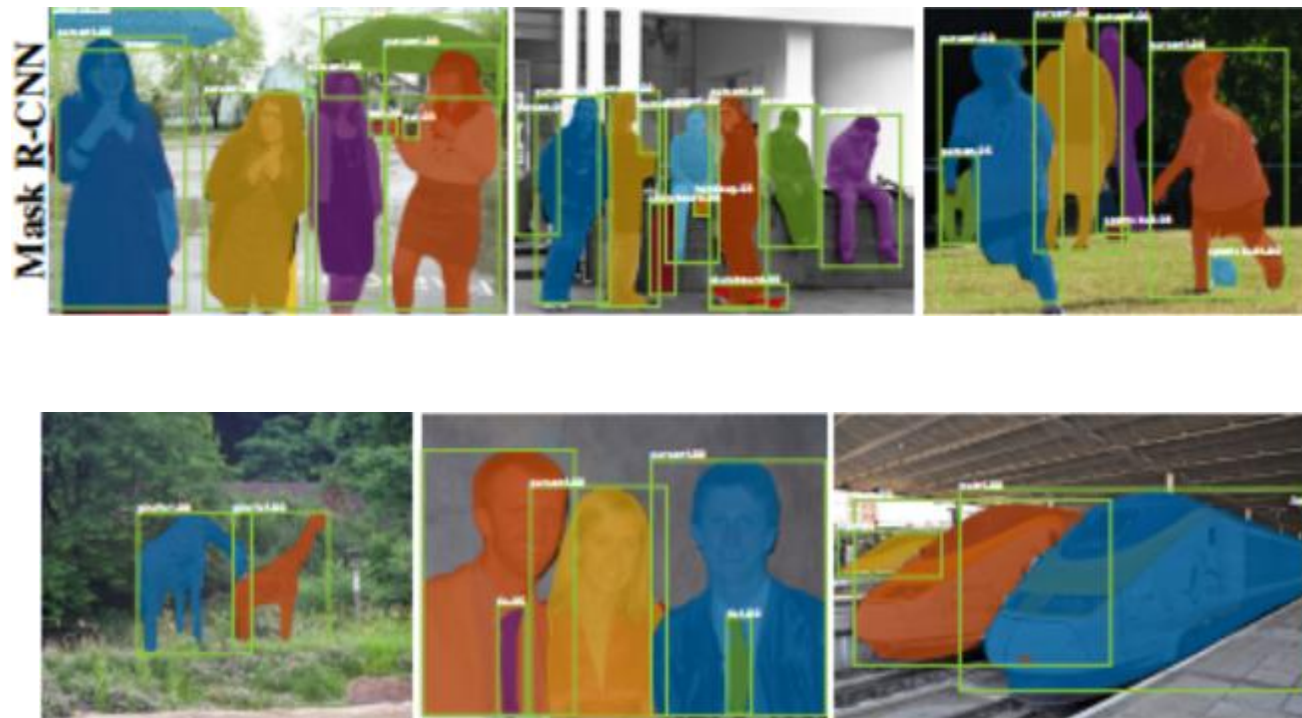
- **Features:**

- Use FPN as backbone
- Add FCN for each proposal as mask branch
- **ROI Align**
- Classify, mask and detect separately / FCN: classify with mask together
- 5 fps



I. Image Segmentation

C. Mask-RCNN [2017, He]:



I. Image Segmentation

D. Developments:

- FCN
- Upsampling method: Deconv —> Interpolation
- FCN with CRF / other traditional methods
- Dilated conv
- Backbone dev: VGG, Resnet, ...
- Pyramid
- Multi-stage: ICNet [Cascade]
- Semi/non-supervised learning [[A paper](#)]

II. Image Style Transfer

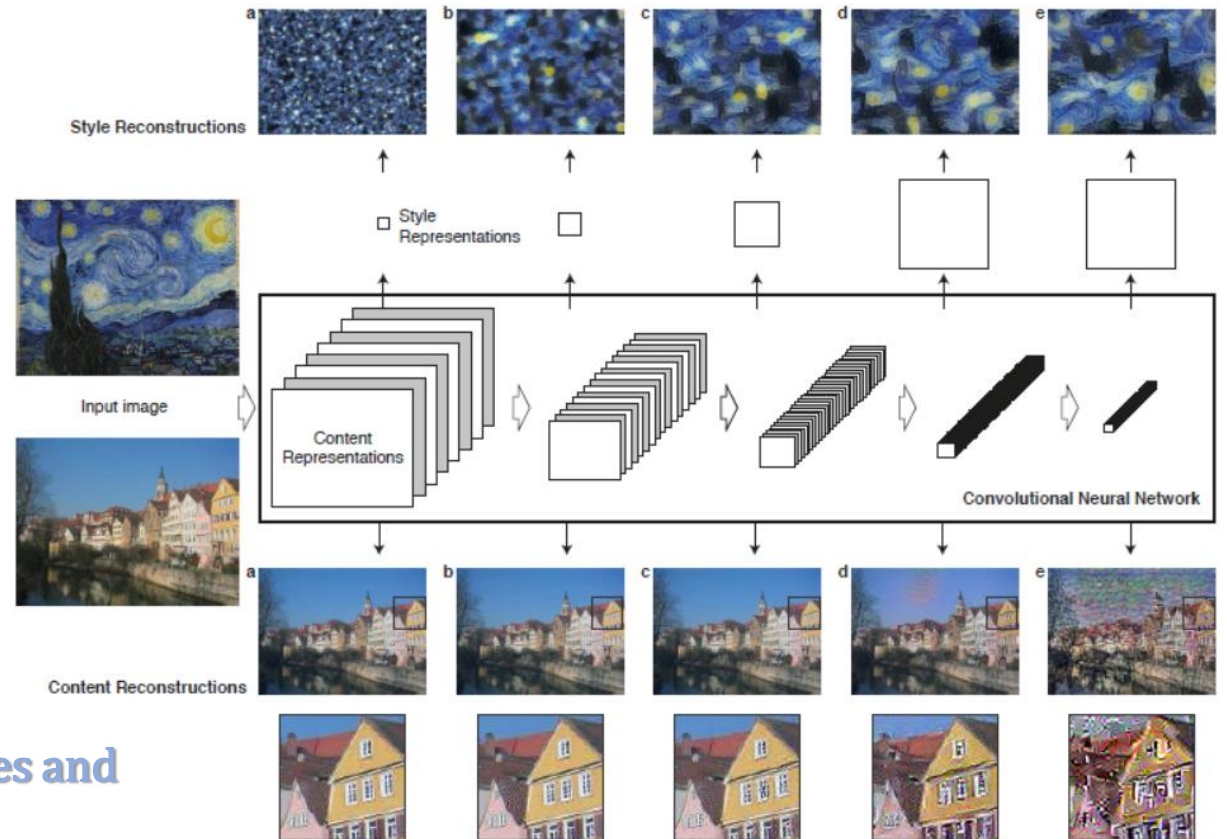
II. Image Style Transfer

E. Image Style Transfer

E1. 1st Trial [2015, Gatys]

- **Arguments:**

- Images can be represented by contents and styles
- The higher a layer is, the more semantic info we'll get;
The lower a layer is, the more local info we'll get.
- We can transfer an image's style by minimizing the loss of their styles and contents.



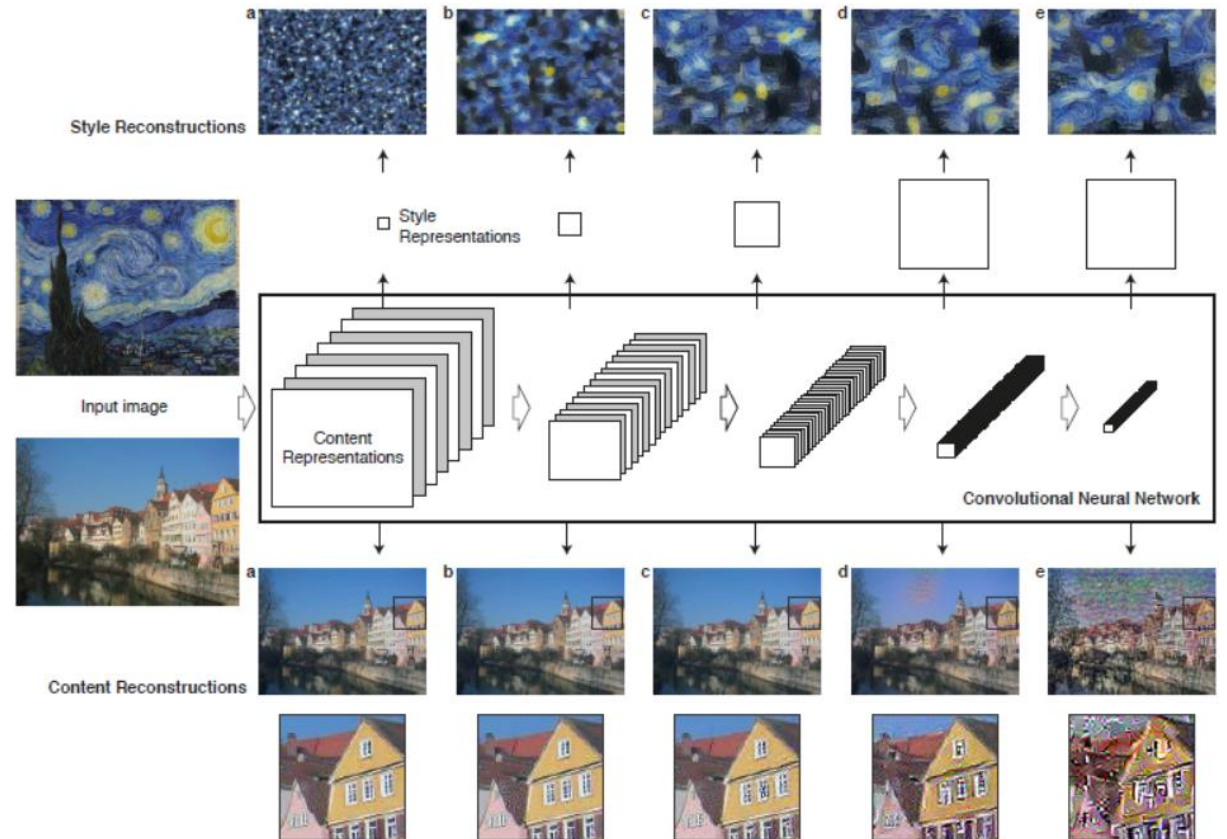
II. Image Style Transfer

E. Image Style Transfer

E1. 1st Trial [2015, Gatys]

- **Questions:**

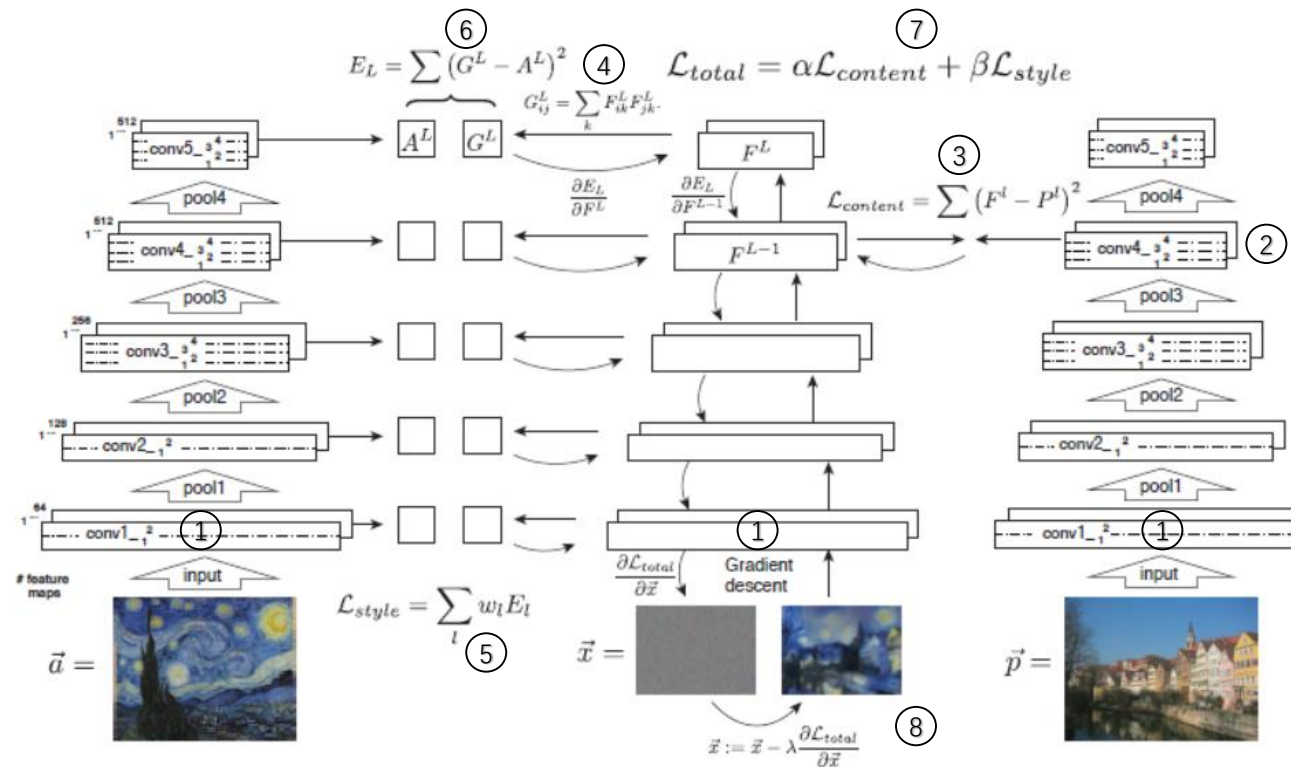
- How to represent contents?
- How to represent styles?
- How to minimizing the gap?
- How to combine content / style?



II. Image Style Transfer

E. Image Style Transfer

E1. 1st Trial [2015, Gatys]



- ① Vgg. Same network.
- ② Higher layer, ampler content info.
- ③ Content loss
- ④ Style representation: Gram Matrix
- ⑤ Different weights for different styles of different layers
- ⑥ Style loss
- ⑦ Combine contents and styles
- ⑧ Generated from white noise

II. Image Style Transfer

E. Image Style Transfer

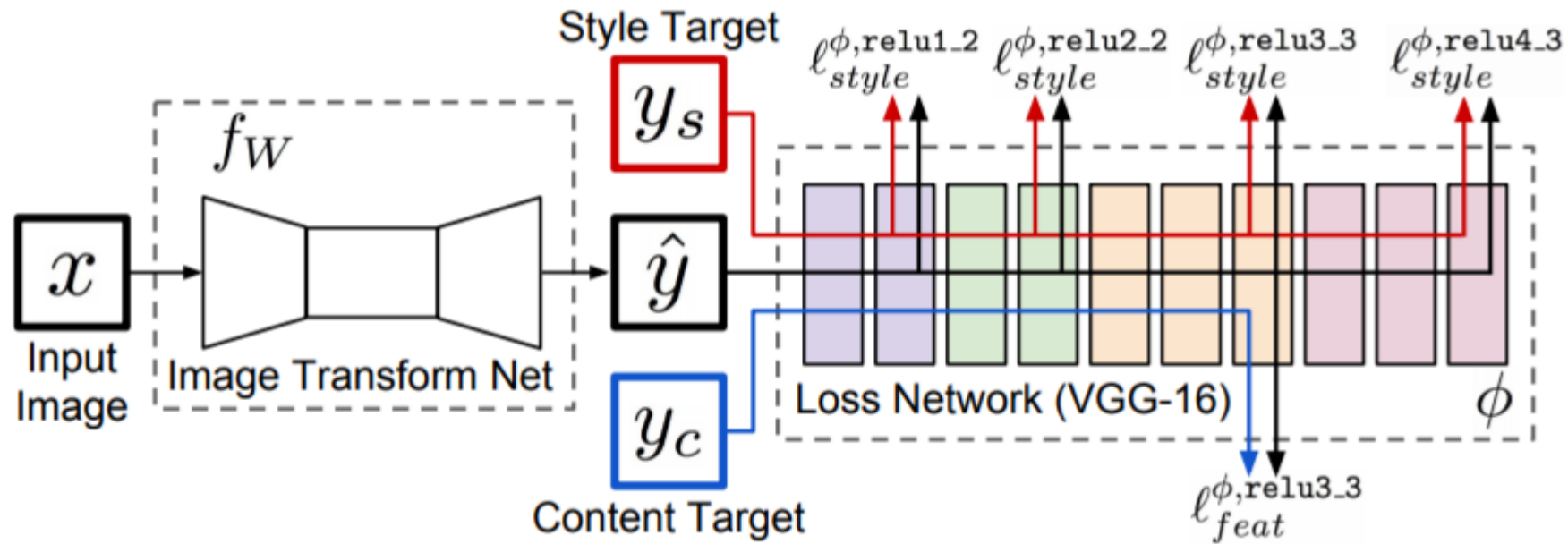
E1. 1st Trial [2015, Gatys]



II. Image Style Transfer

E. Image Style Transfer

E2. Make It Faster! [2016, Perceptual Loss, Justin]



II. Image Style Transfer

E. Image Style Transfer

E2. Make It Faster! [2016, Perceptual Loss, Justin]

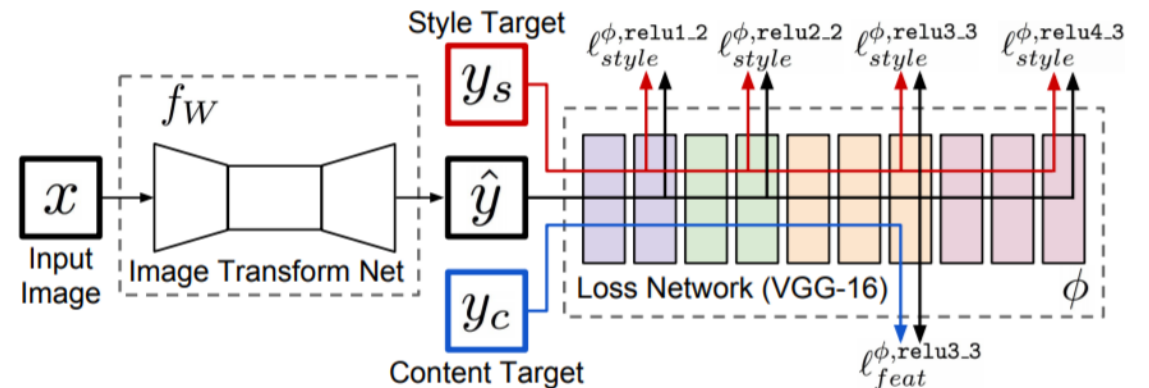
- **Import aspects:**

- **Perceptual Loss! Perceptual Loss! Perceptual Loss!**

- **Variation Regularization**

$$\hat{y} = \arg \min_y \lambda_c \ell_{feat}^{\phi, j}(y, y_c) + \lambda_s \ell_{style}^{\phi, J}(y, y_s) + \lambda_{TV} \ell_{TV}(y)$$

- **Functional network + Loss network**



II. Image Style Transfer

E. Image Style Transfer

E2. Make It Faster! [2016, Perceptual Loss, Justin]

- **Tips:**

- L1 / L2 / Perceptual / Gradient loss in Image Transferring

- L1: Good for details but bad for color

- L2: Good for color but bad for details

- Perceptual: Good for everything. Common in super resolution

- Gradient: Can decrease chessboard pattern tremendously

- **Thinking:**

- Assume your task is to improve the rate of face recognition under bad light condition.

- What can you do?



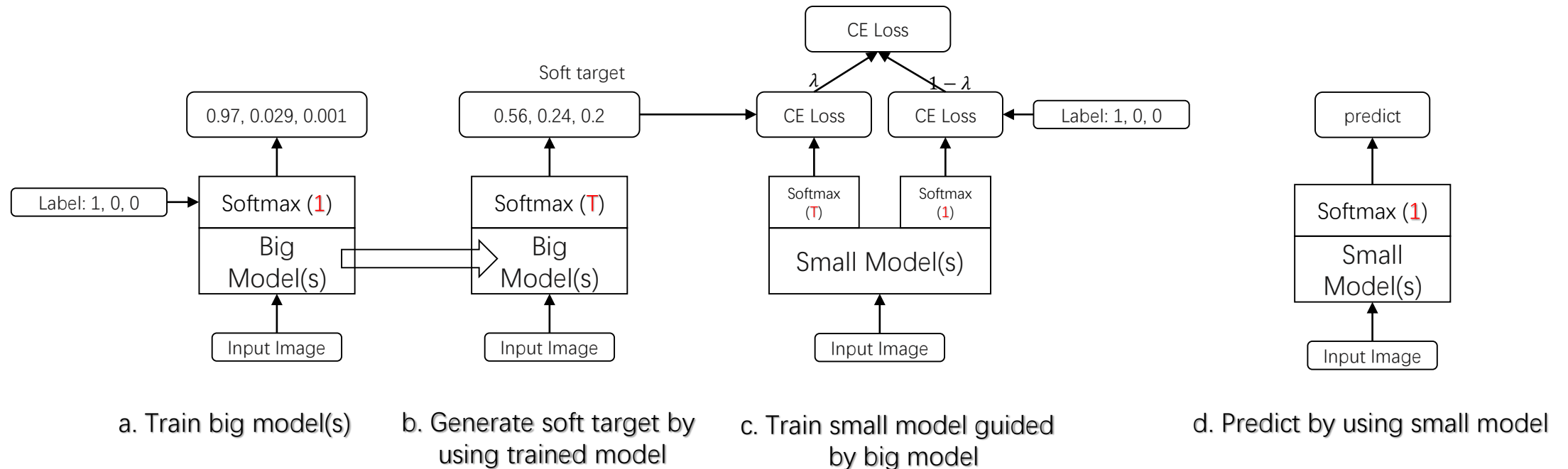
II. Image Style Transfer

F. Feature mimicking / Model distillation

- **Aim:**
 - Distill knowledge from bigger models
 - Use the distilled knowledge to guide the learning of smaller models
 - Use smaller models to mimic the effect of bigger models
- **Papers:**
 - Distilling the Knowledge in a Neural Network [2015, Hinton]
 - FitNets: Hints for Thin Deep Nets [2015, Remero, Bengio]
 - Mimicking Very Efficient Network for Object Detection [2017, Quanquan Li]

II. Image Style Transfer

F. Feature mimicking / Model distillation



III. Other Applications

III. Other Applications

G. Others

➤ [Image Enhancement \[2018\]](#)



1.JPG



2.JPG



3.JPG



4.JPG



5.JPG



6.JPG



7.JPG

Channel split;

Exposure combination;

High/Low frequency split



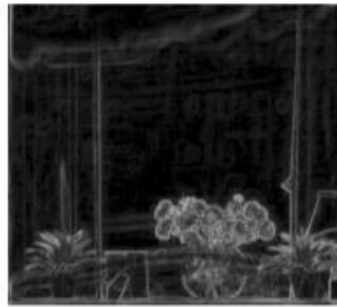
III. Other Applications

G. Others

➤ [Reflection Removal \[2018\]](#)



Input image



Input edge



Pred. edge by E-CNN



CEILNet (naive data)



I-CNN only



CEILNet

$$\mathbf{I} = \mathbf{B} + \mathbf{R}$$

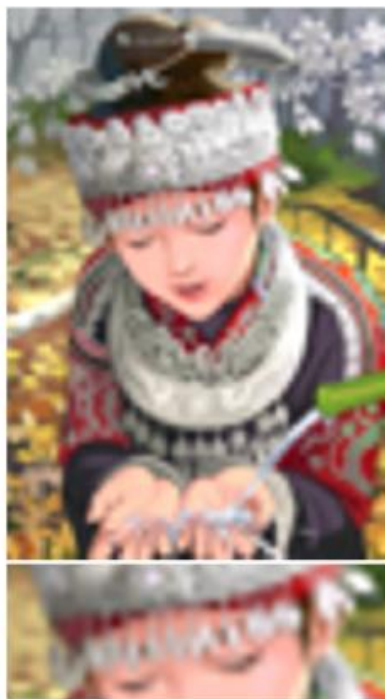
III. Other Applications

G. Others

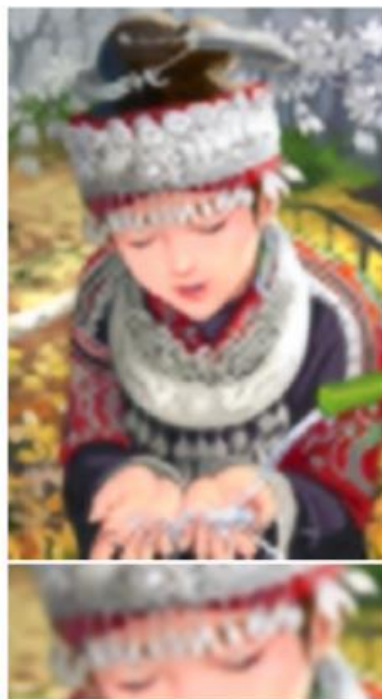
➤ Super Resolution



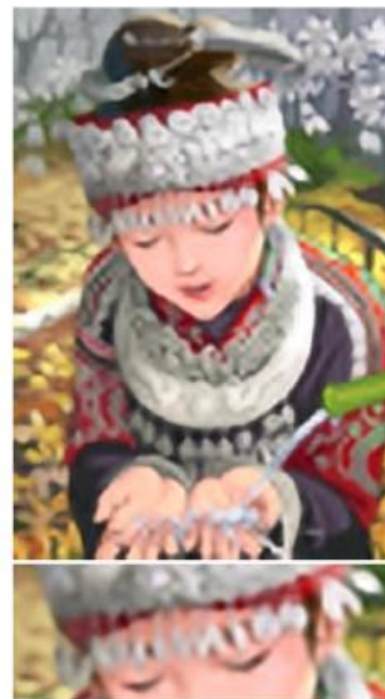
Ground Truth



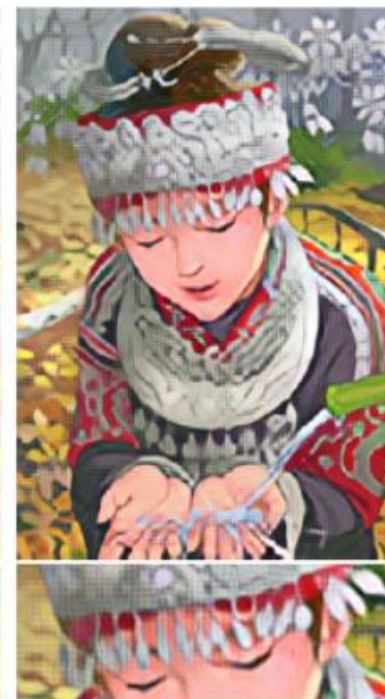
Bicubic



Ours (ℓ_{pixel})



SRCNN [13]



Ours (ℓ_{feat})