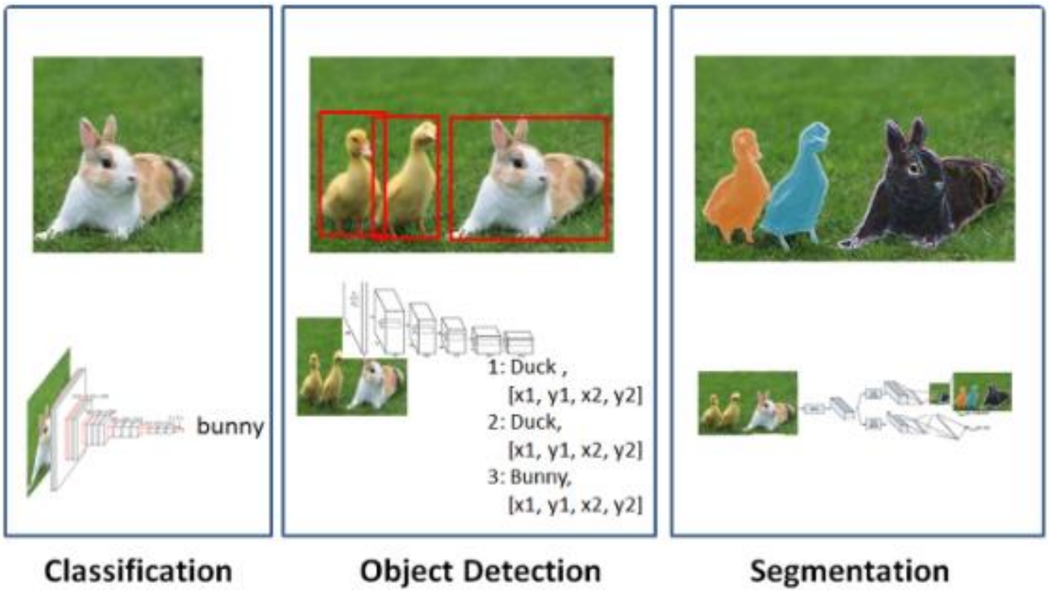
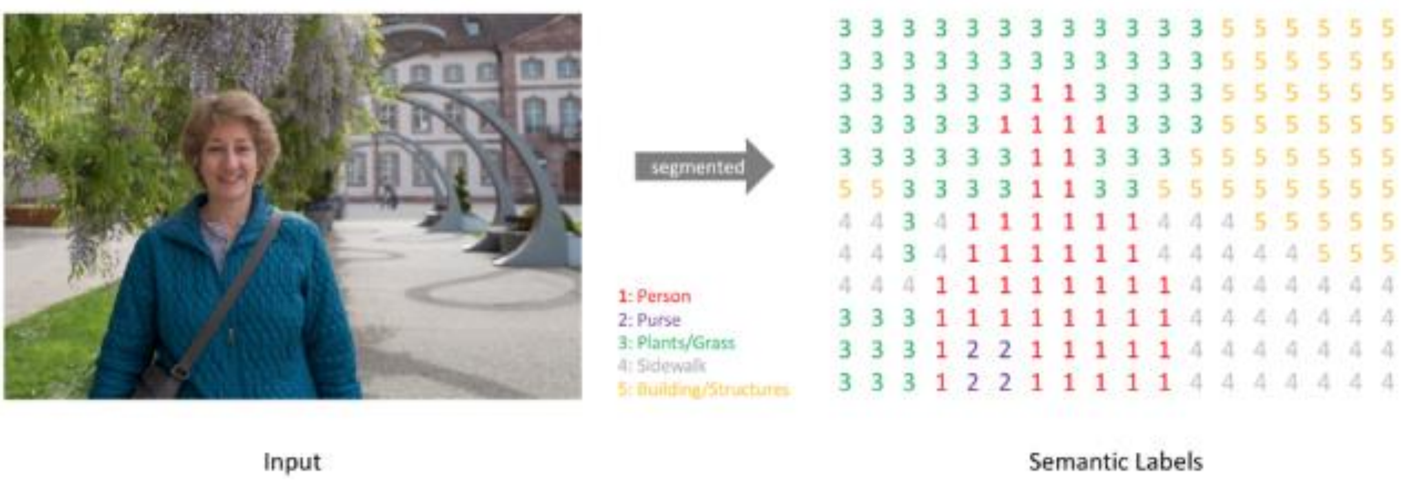
디지털 영상처리 연구실 연구보고서

김우헌

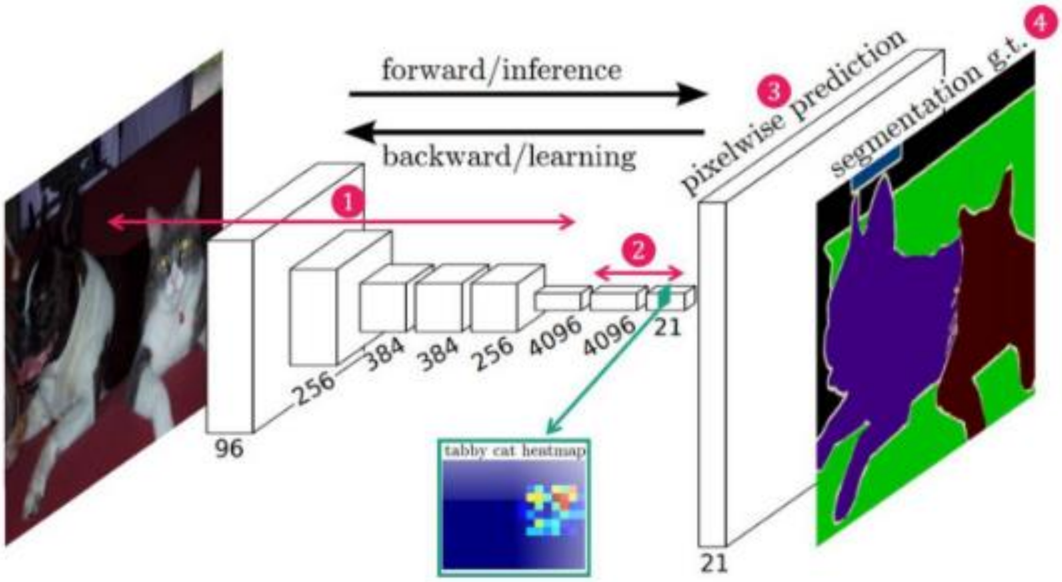
##image segmentation

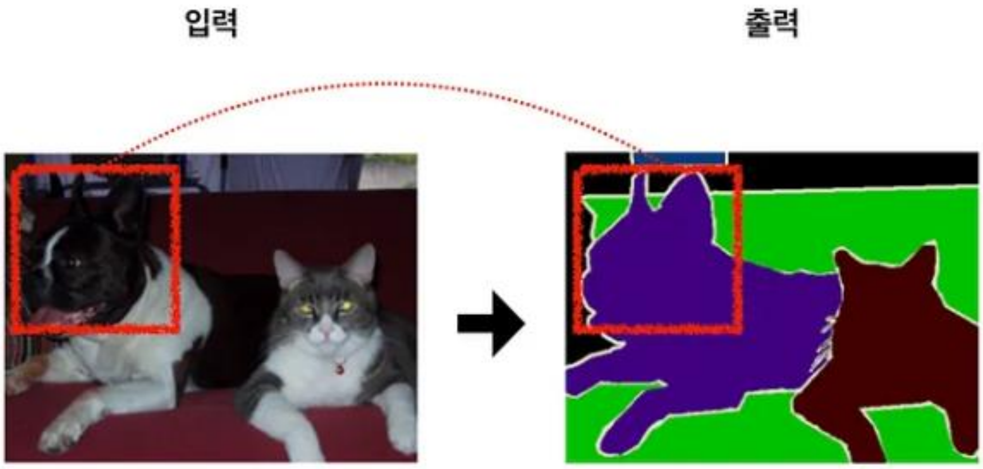




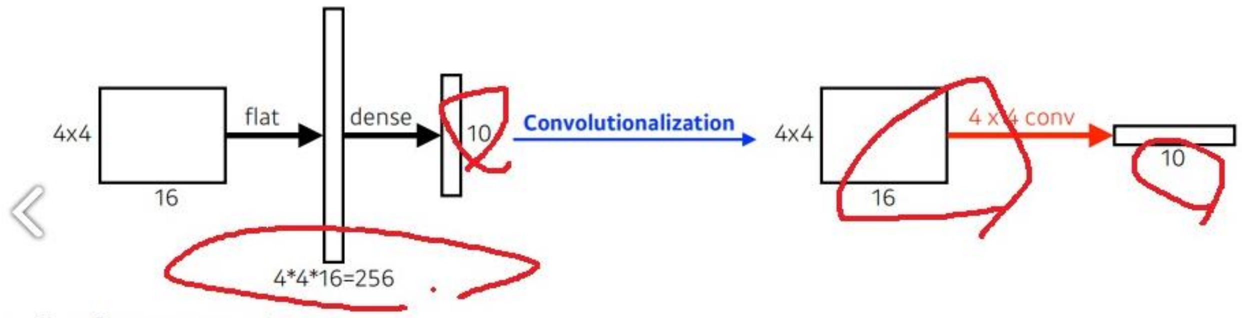
```
3 3 3 3 3 3 3 3 3 3 3 3 5 5 5 5 5 5
3 3 3 3 3 3 3 3 3 3 3 3 5 5 5 5 5 5
3 3 3 3 3 3 1 1 3 3 3 3 5 5 5 5 5 5
3 3 3 3 3 1 1 1 1 3 3 3 5 5 5 5 5 5
3 3 3 3 3 3 1 1 3 3 3 5 5 5 5 5 5 5
5 5 3 3 3 3 1 1 3 3 5 5 5 5 5 5 5 5
4 4 3 4 1 1 1 1 1 1 4 4 4 5 5 5 5 5
4 4 3 4 1 1 1 1 1 1 4 4 4 4 4 5 5 5
4 4 4 1 1 1 1 1 1 1 1 4 4 4 4 4 4 4
3 3 3 1 1 1 1 1 1 1 1 4 4 4 4 4 4 4
3 3 3 1 2 2 1 1 1 1 1 4 4 4 4 4 4 4
3 3 3 1 2 2 1 1 1 1 1 4 4 4 4 4 4 4
```

#FCN(Fully Convolutional Networks for Semantic Segmentation)





1. Convolutionalization



# of parameters

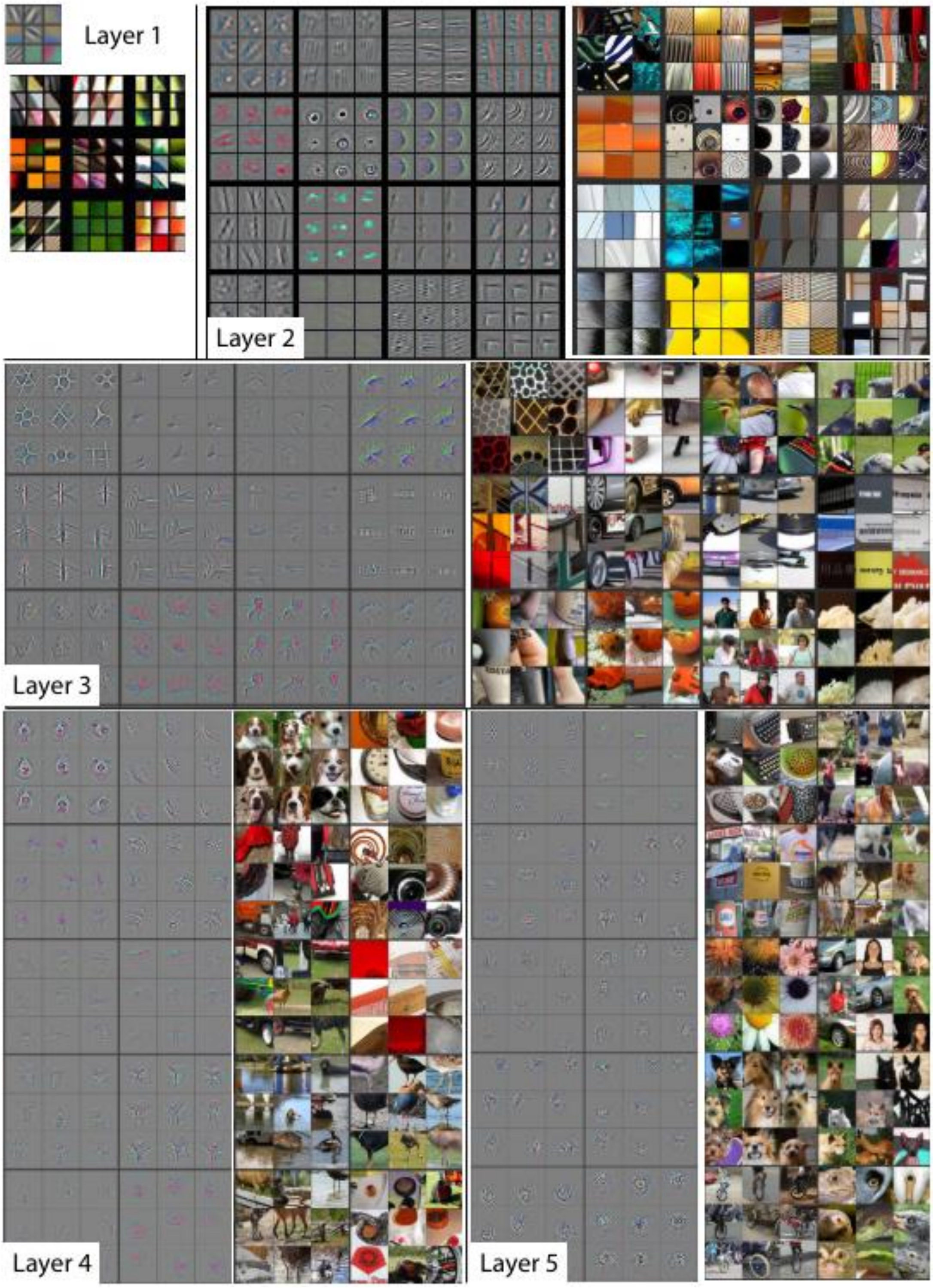
- Left: $4 \times 4 \times 16 \times 10 = 2{,}560$
- Right: $4 \times 4 \times 16 \times 10 = 2{,}560$

2. Deconvolution (Upsampling)

    1. Bilinear Interpolation

    2. Backwards convolution

3. Skip Architecture



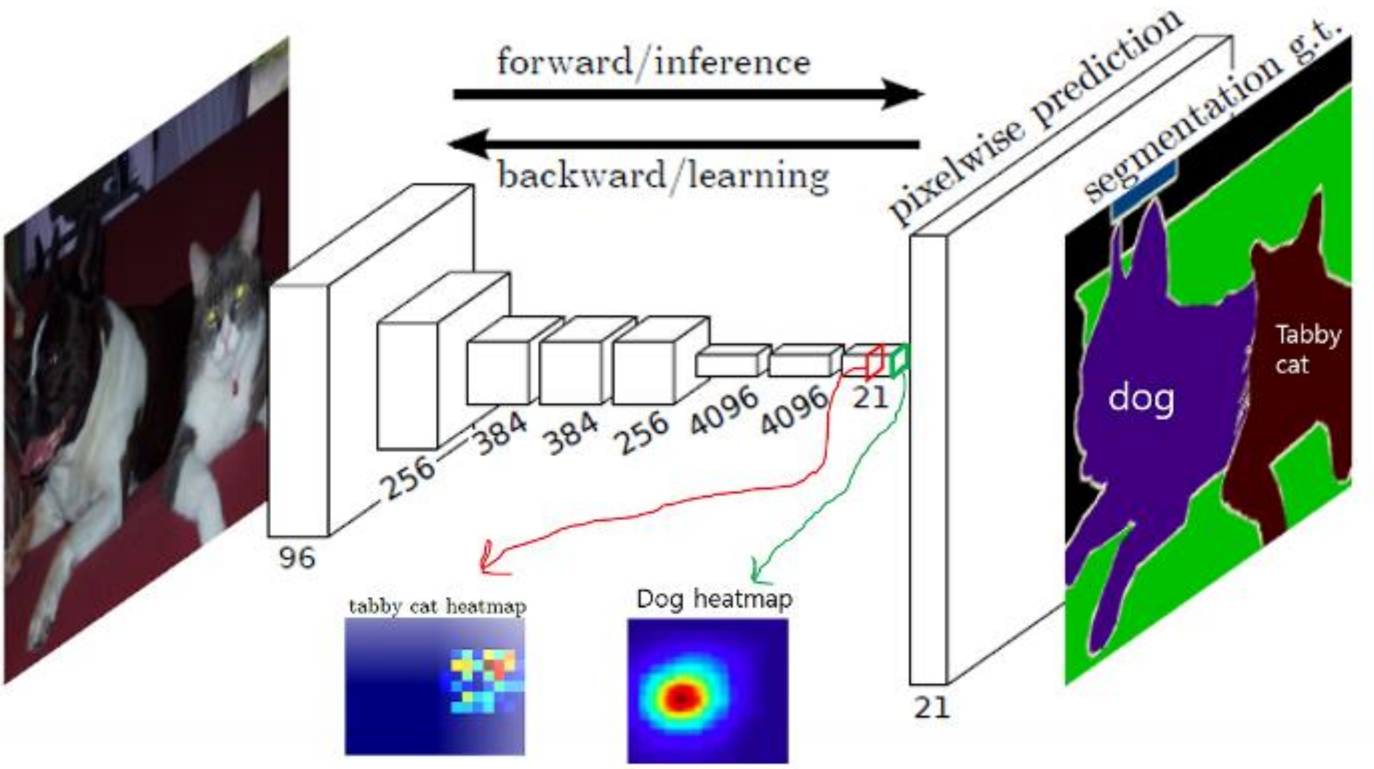

forward/inference
backward/learning
pixelwise prediction
segmentation g.t.
96
256
384
384
256
4096
4096
21
21
dog
Tabby cat
tabby cat heatmap
Dog heatmap

그림14

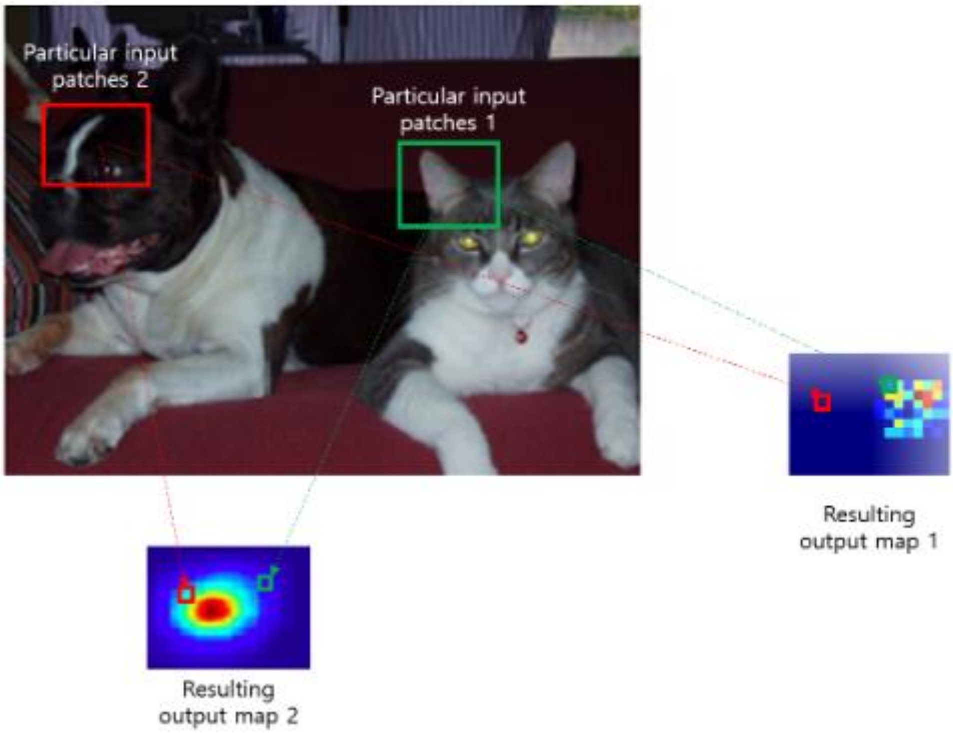
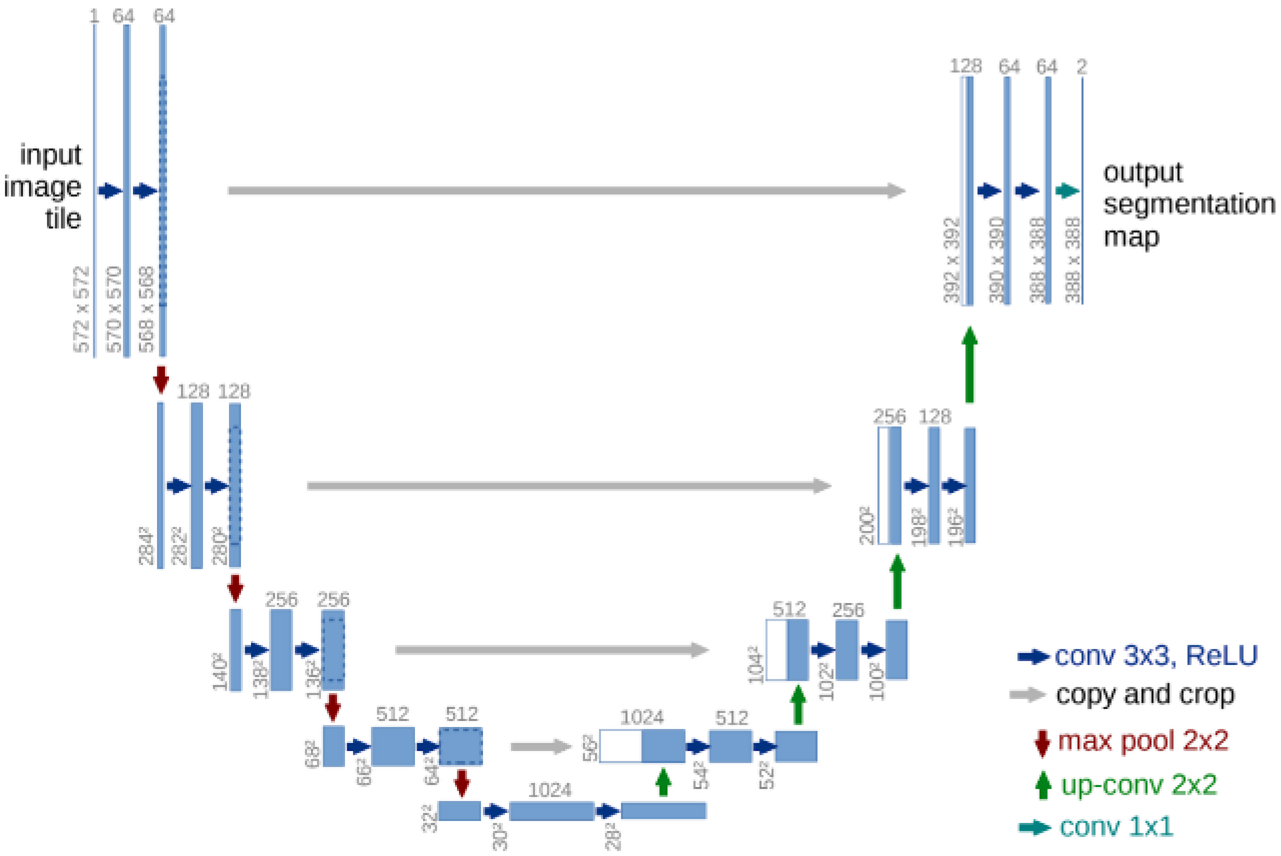
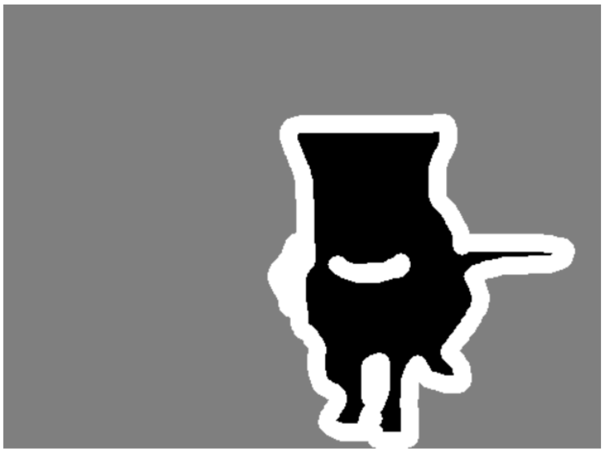

Particular input patches 2
Particular input patches 1
Resulting output map 1
Resulting output map 2

그림15

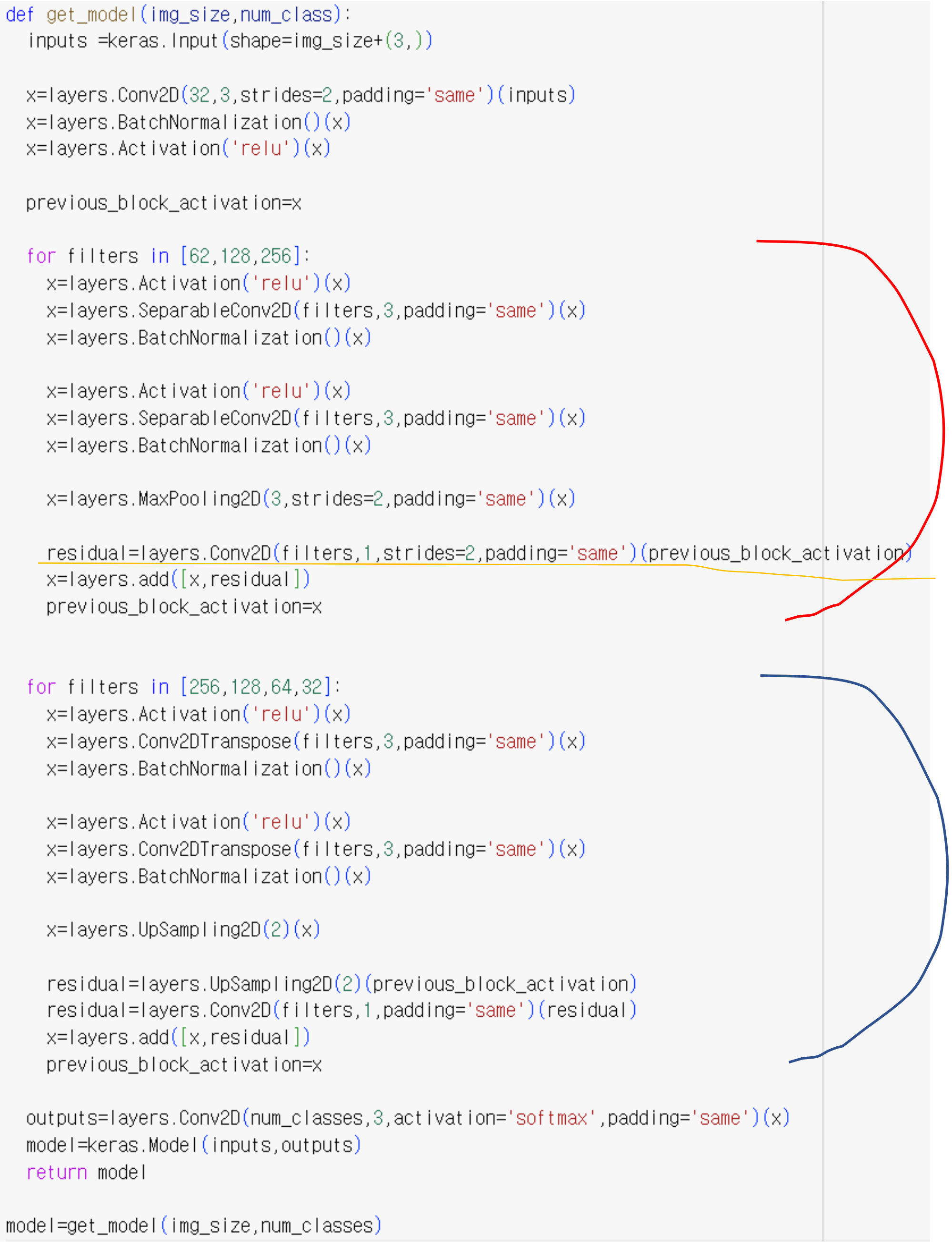
#U-Net 기반 image segmentation



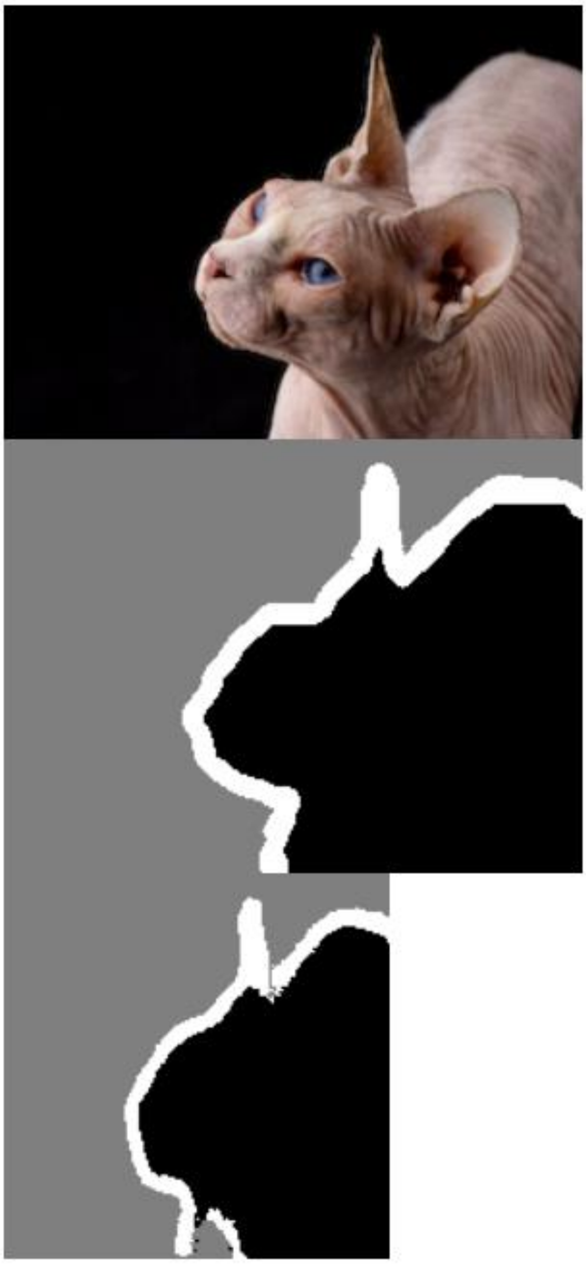
->Oxford-IIIT Pets 데이터셋 사용

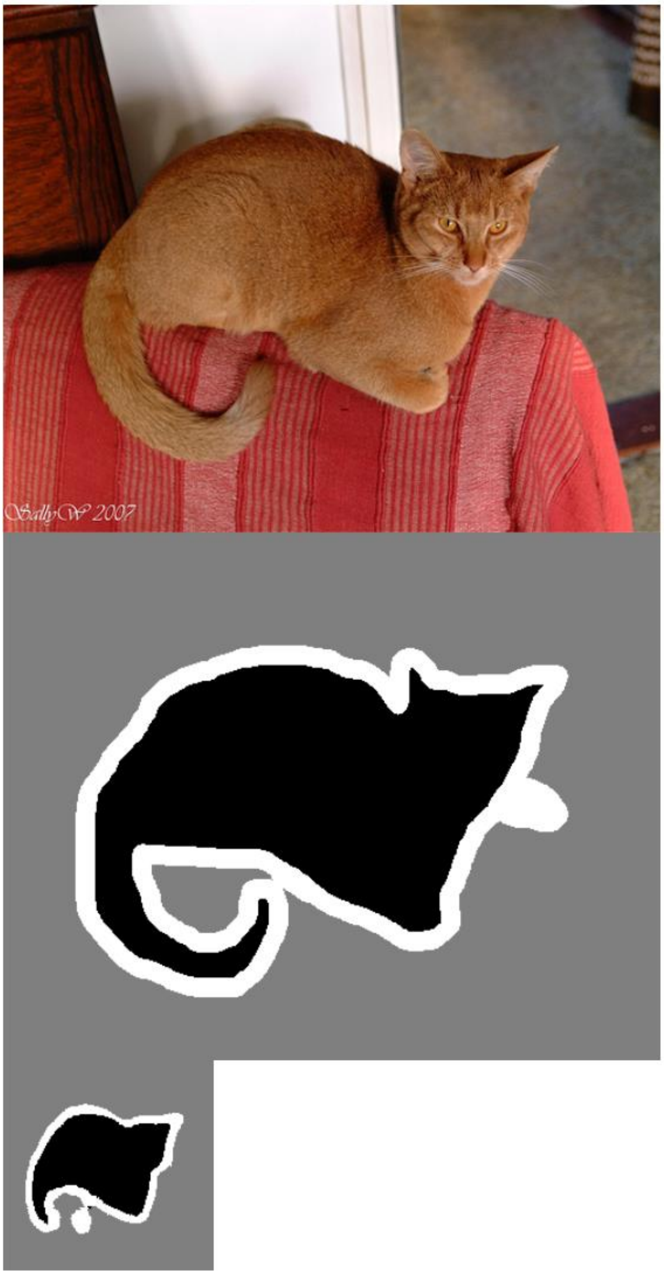


->7000개 정도의 data set



```
def get_model(img_size,num_class):
  inputs =keras.Input(shape=img_size+(3,))

  x=layers.Conv2D(32,3,strides=2,padding='same')(inputs)
  x=layers.BatchNormalization()(x)
  x=layers.Activation('relu')(x)

  previous_block_activation=x

  for filters in [62,128,256]:
    x=layers.Activation('relu')(x)
    x=layers.SeparableConv2D(filters,3,padding='same')(x)
    x=layers.BatchNormalization()(x)

    x=layers.Activation('relu')(x)
    x=layers.SeparableConv2D(filters,3,padding='same')(x)
    x=layers.BatchNormalization()(x)

    x=layers.MaxPooling2D(3,strides=2,padding='same')(x)

    residual=layers.Conv2D(filters,1,strides=2,padding='same')(previous_block_activation)
    x=layers.add([x,residual])
    previous_block_activation=x


  for filters in [256,128,64,32]:
    x=layers.Activation('relu')(x)
    x=layers.Conv2DTranspose(filters,3,padding='same')(x)
    x=layers.BatchNormalization()(x)

    x=layers.Activation('relu')(x)
    x=layers.Conv2DTranspose(filters,3,padding='same')(x)
    x=layers.BatchNormalization()(x)

    x=layers.UpSampling2D(2)(x)

    residual=layers.UpSampling2D(2)(previous_block_activation)
    residual=layers.Conv2D(filters,1,padding='same')(residual)
    x=layers.add([x,residual])
    previous_block_activation=x

  outputs=layers.Conv2D(num_classes,3,activation='softmax',padding='same')(x)
  model=keras.Model(inputs,outputs)
  return model

model=get_model(img_size,num_classes)
```





SallyW 2007