# Semantic Segmentation with PyTorch

CMPUT 328

Nilanjan Ray

# Agenda

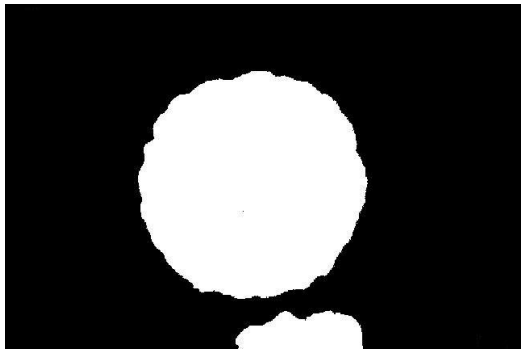
- We will look at two notebooks semantic segmentation
- For simplicity, we will do 2-class segmentation: foreground and background segmentation for a “flower.jpg” image
- Our first attempt will be to train a convnet on classification task and then use it for segmentation
  - In this process, we will learn one important feature: using a convolution in place of a fully connected layer
  - The advantage of using a fully convolutional layer is that the network can accept an image of any size as an input
- One limitation of applying a classification convent for segmentation is that the output would be smaller in size than the input
- Our next attempt will overcome this low-resolution issue by introducing an important type of convolution operation called “transposed convolution” (aka “upsampling”)
  - We will also learn about an important aspect about *labels* in training a “segmentation net” that is different from training a “classification net”

# Semantic segmentation example

For this lab we will do the simplest type of segmentation: foreground and background segmentation

Flower image

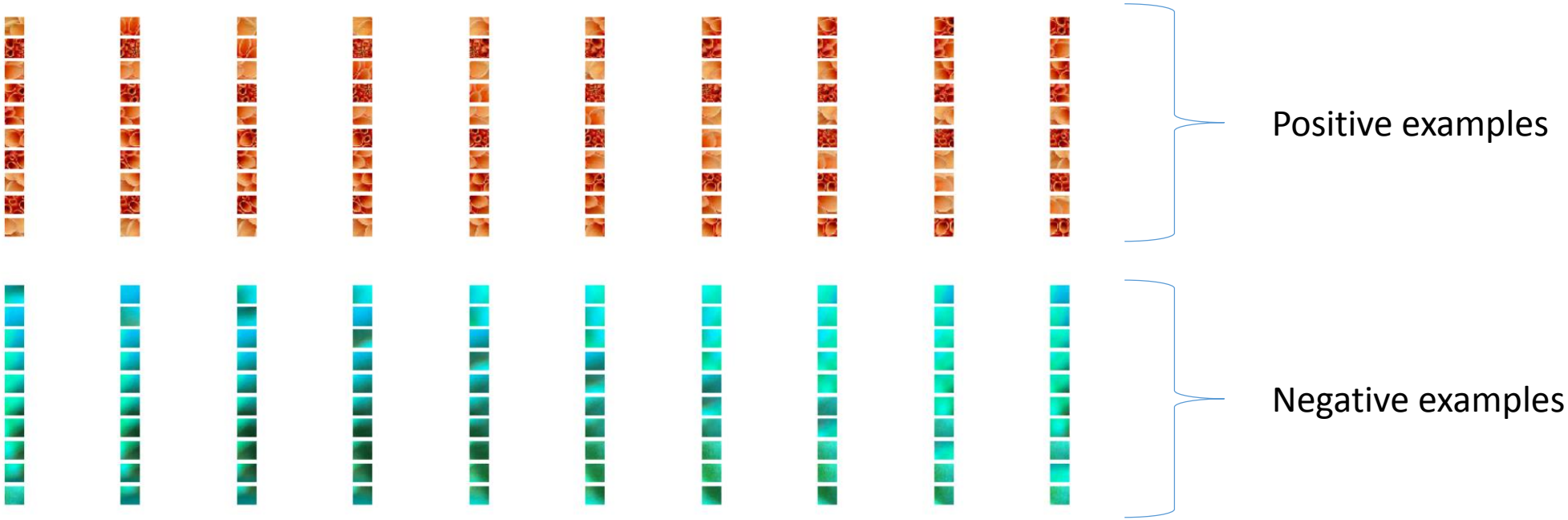
Semantic Segmentation



Binary foreground-background segmentation

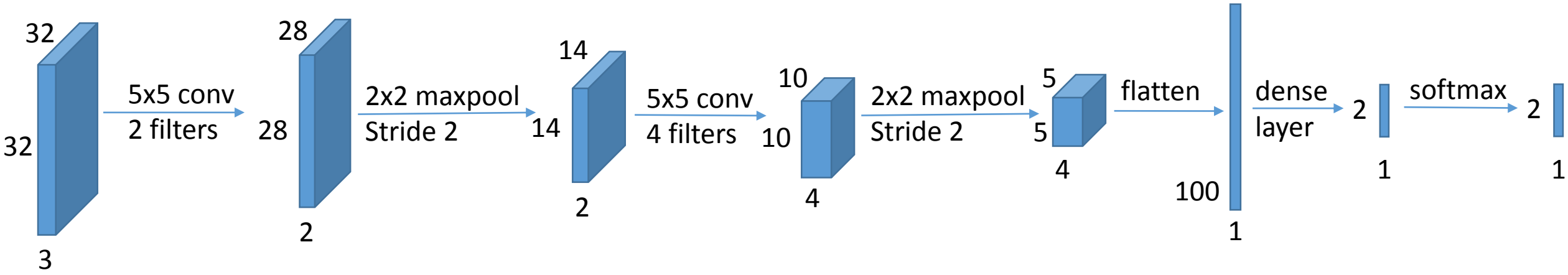
# Segmentation as image classification

Training dataset: 32-by-32-by-3 RGB images with positive or negative labels



# Segmentation as image classification…

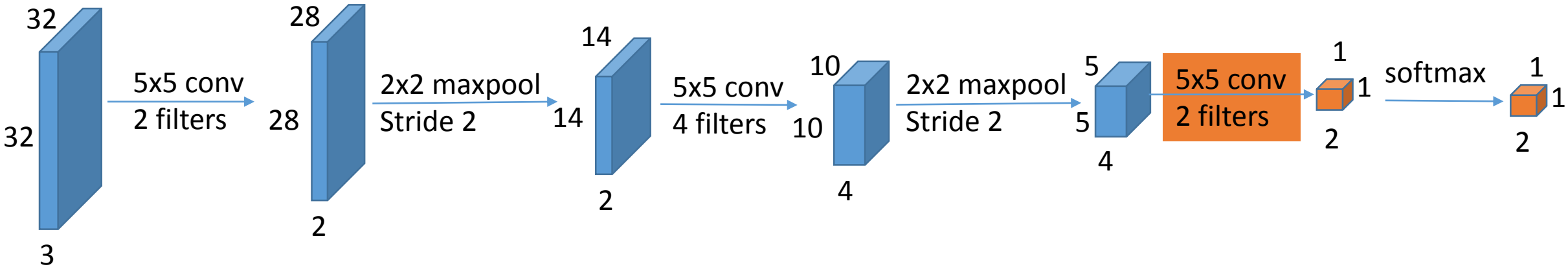
Train a convnet on the training set




This classification net can only take a fixed size input image.

# Segmentation as image classification…

Let’s convert the dense layer to a convolution layer:




This classification net can take an input of any width and height.

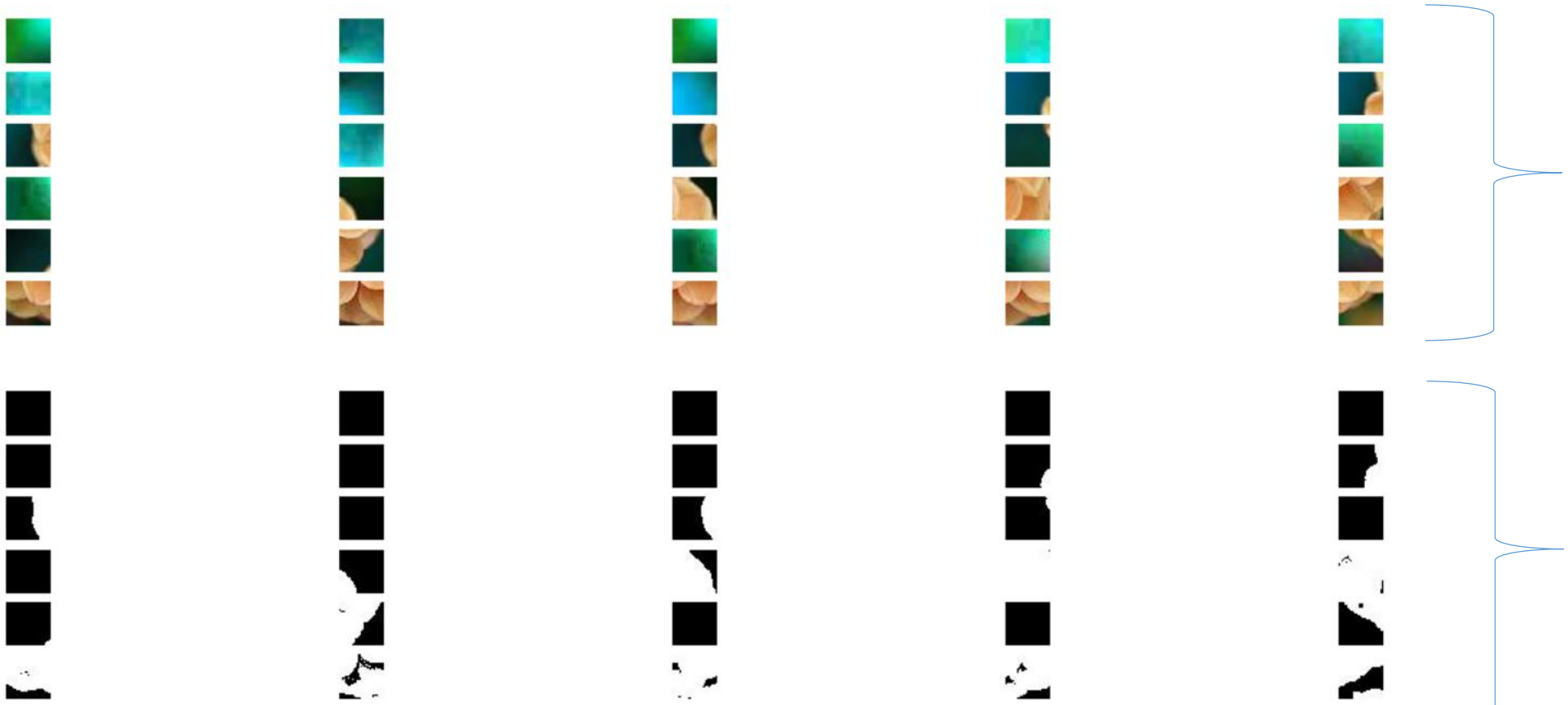
But, output image will have approximately 4x lower resolution.

Let’s go through “ImSegFromClassification” notebook.

# Segmentation using convolution transpose

- Convolution transpose is a type of convolution
  - See: https://nrupatunga.github.io/2016/05/14/convolution-arithmetic-in-deep-learning-part-1/
  - https://nrupatunga.github.io/convolution-2/
- Convolution transpose is more commonly known as “upsampling” layer
- Let’s call this net “Segmentation Net”

# Training set for segmentation net

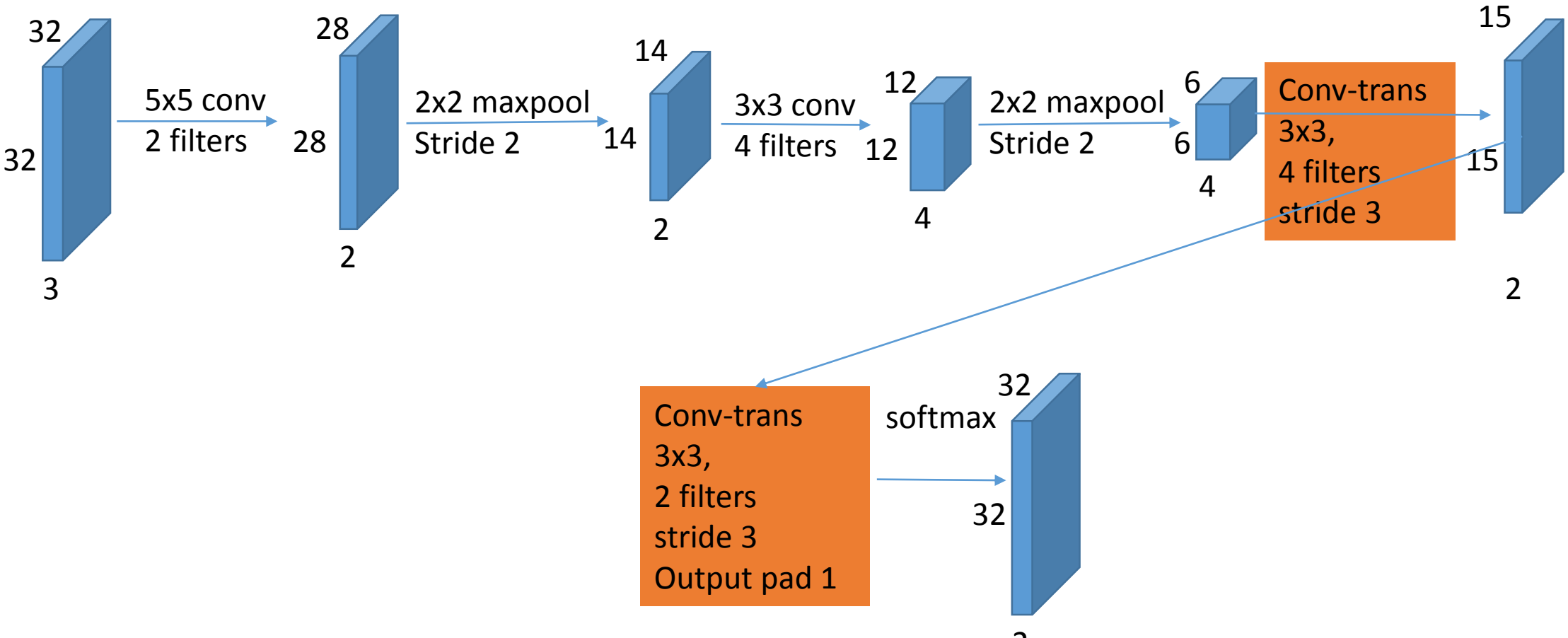



A training image has a corresponding label image of same height and width.

# Segmentation net architecture

This network can take input of any height and width.

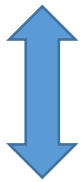
Output image does not lose resolution!



# Convolution transpose layer

Roughly speaking, the size of the output of conv-trans layer would be determined by the following relationship:

output = conv(input, kernel, stride)



input = conv_trans(output, kernel, stride)

# Exercise

Design this architecture and train; then segment the flower image with this architecture.