COSE474 Deep Learning

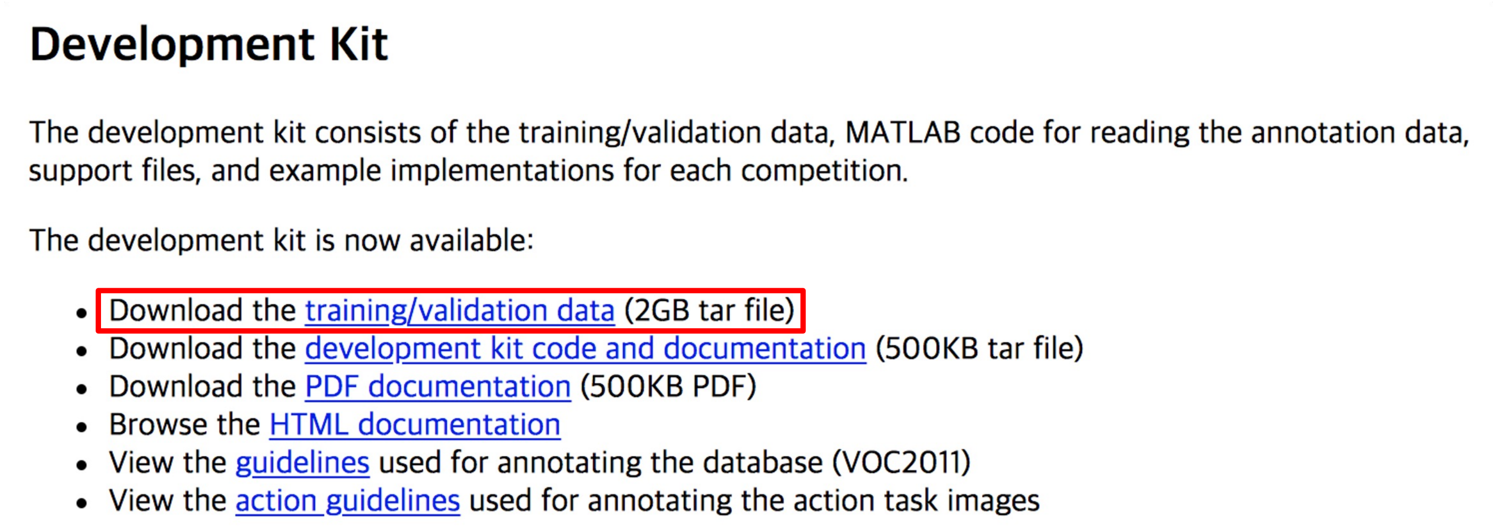
Project #3: Encoder-Decoder Implementation

Seungryong Kim

Computer Vision Lab. (CVLAB)
Department of Computer Science and Engineering
Korea University

Pascal VOC

- **Download Pascal VOC 2012 Dataset**
  - http://host.robots.ox.ac.uk/pascal/VOC/voc2012/



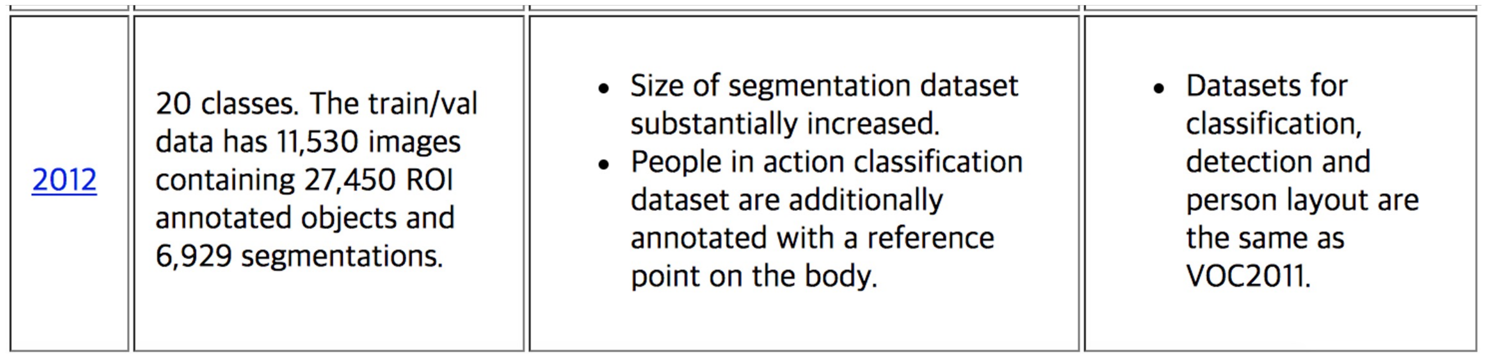
Development Kit

The development kit consists of the training/validation data, MATLAB code for reading the annotation data, support files, and example implementations for each competition.

The development kit is now available:

- Download the training/validation data (2GB tar file)
- Download the development kit code and documentation (500KB tar file)
- Download the PDF documentation (500KB PDF)
- Browse the HTML documentation
- View the guidelines used for annotating the database (VOC2011)
- View the action guidelines used for annotating the action task images

- Latest version (2012)



2012	20 classes. The train/val data has 11,530 images containing 27,450 ROI annotated objects and 6,929 segmentations.	• Size of segmentation dataset substantially increased. • People in action classification dataset are additionally annotated with a reference point on the body.	• Datasets for classification, detection and person layout are the same as VOC2011.

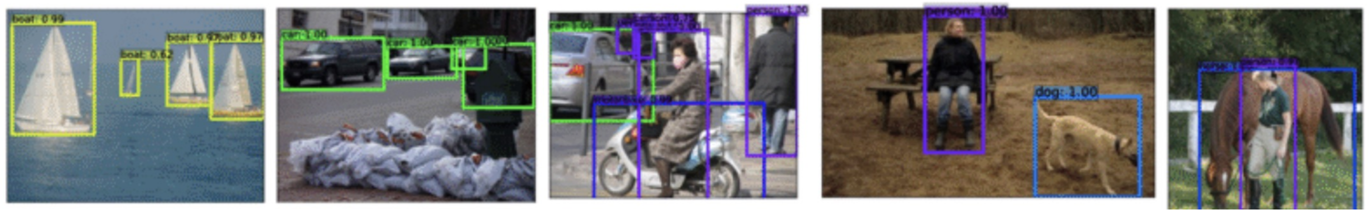
Pascal VOC

- **Pascal VOC 2012 Dataset**

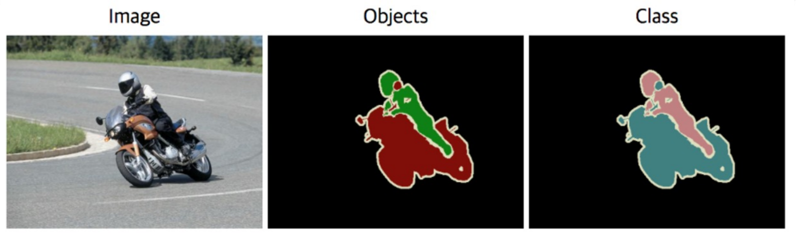
- Pascal VOC 2012 has 20 classes. The train/validation data has 11,530 images containing 27,450 ROI annotated objects and 6,929 segmentations.
- **Classes**
  - Person : person
  - Animal : bird, cat, cow, dog, horse, sheep
  - Vehicle : aeroplane, bicyble, boat, bus, car, motorbike, train
  - Indoor : bottle, chair, dining table, potted plant, sofa, tv/monitor

Pascal VOC

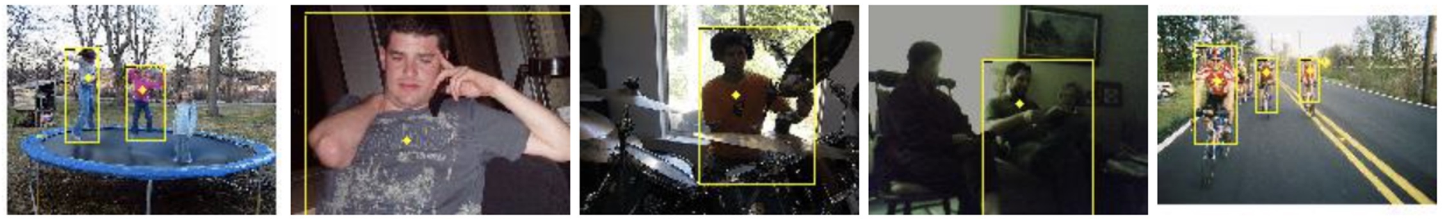
- **Pascal VOC 2012 Dataset**
  - There are main object recognition competitions: classification, detection, segmentation, action classification, and a competition on large scale recognition run by ImageNet.
  - Classification/Detection Competitions



  - Segmentation Competition

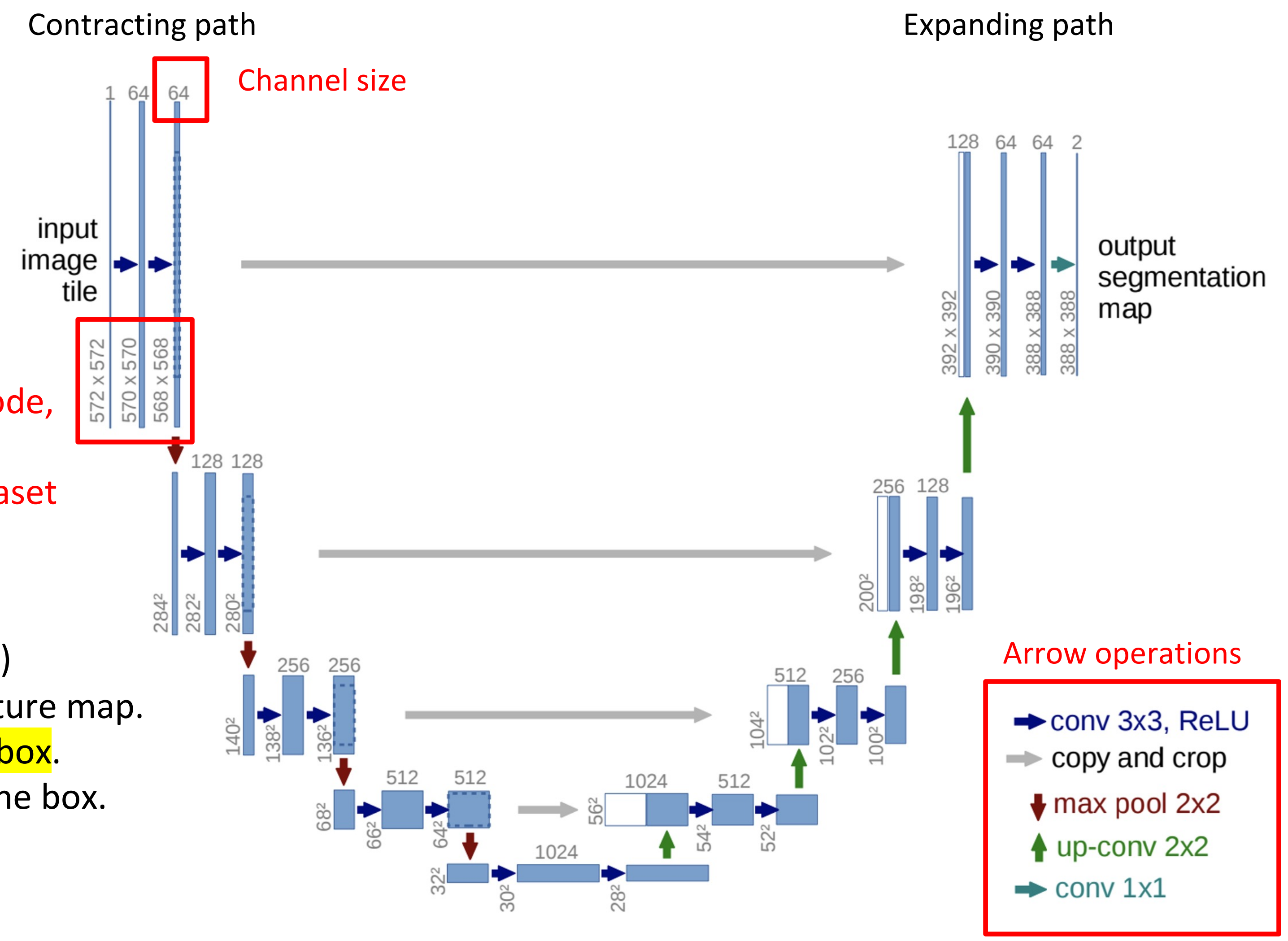


  - Action Classification Competition



Original U-Net

- **Train "*UNet*"**
  - Network Architecture

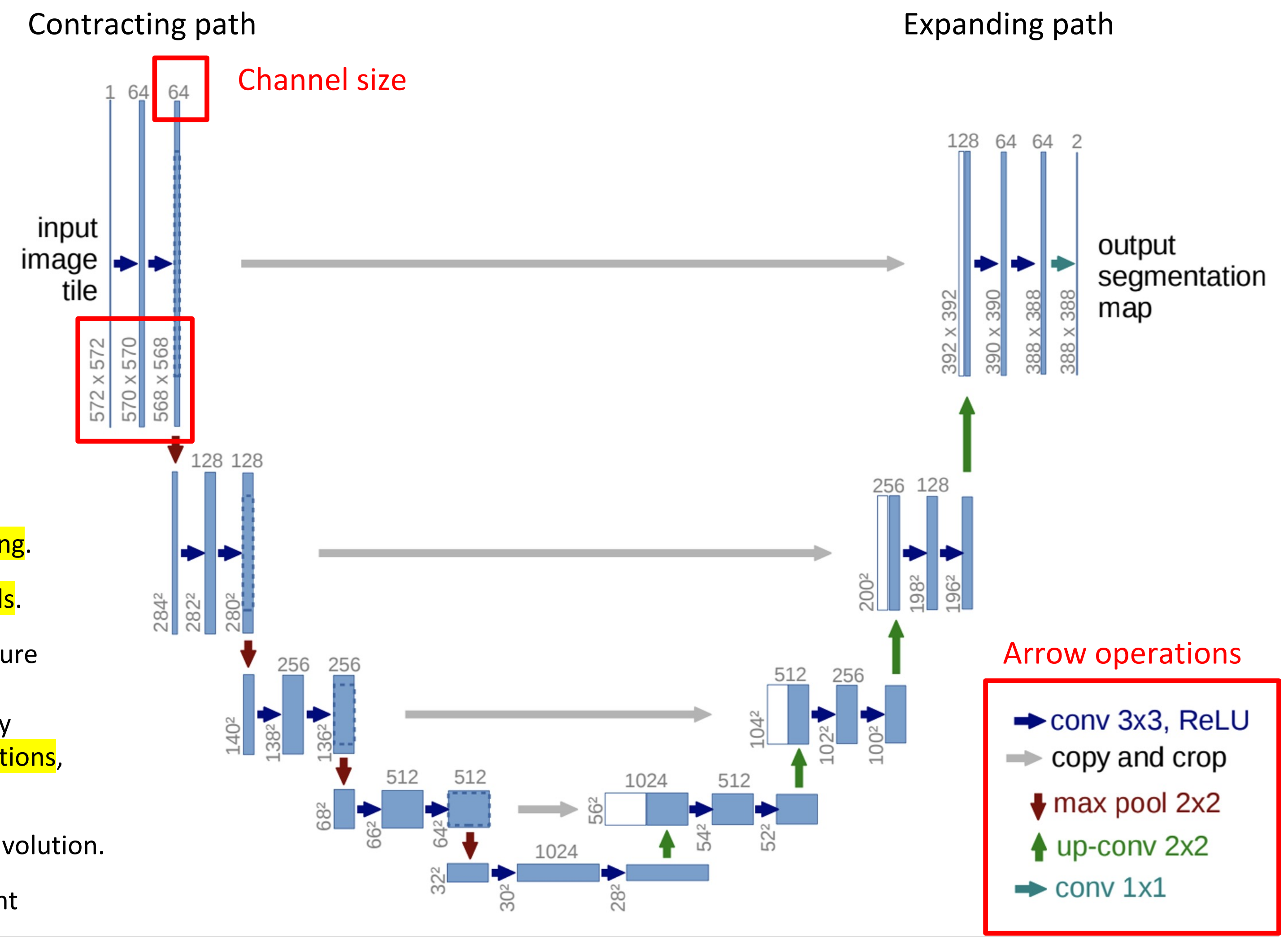


1. U-net architecture
   (example for 32x32 pixels in the lowest resolution)
2. Each blue box corresponds to a multi-channel feature map.
3. The number of channels is denoted on top of the box.
4. The x-y-size is provided at the lower left edge of the box.
5. White boxes represent copied feature maps.
6. The arrows denote the different operations.

Original U-Net

- **Train “*UNet*”**
  - Network Architecture

- It consists of a contracting path (left side) and an expansive path (right side).
- The contracting path follows the typical architecture of a convolutional network.
- It consists of the repeated application of two 3x3 convolutions (unpadded convolutions), each followed by a rectified linear unit (ReLU) and a 2x2 max pooling operation with stride 2 for downsampling.
- At each downsampling step we double the number of feature channels.
- Every step in the expansive path consists of an upsampling of the feature map followed by a 2x2 convolution (up-convolution) that halves the number of feature channels, a concatenation with the correspondingly cropped feature map from the contracting path, and two 3x3 convolutions, each followed by a ReLU.
- The cropping is necessary due to the loss of border pixels in every convolution.
- At the final layer a 1x1 convolution is used to map each 64- component feature vector to the desired number of classes.
- In total the network has 23 convolutional layers.



Original U-Net

- **Train “*UNet*”**
  - Train “UNet” model with “Pascal VOC 2012” datasets, and conduct image segmentation
    - Optimize parameters with *Adam optimizer* and *cross Entropy Loss*
      - ■ Get “Pascal VOC 2012” Dataset using previous ppt slides.
        (*Tip : If data processing process takes too long, try using only 20 images first.)
    - Procedure

      1) Download Pascal VOC 2012 Dataset using page 2 in this slide.

      2) Fill in the blanks of the skeleton codes ('main_skeleton.py', 'modules_skeleton.py', 'Unet_skeleton.py')

      'main_skeleton.py', 'modules_skeleton.py': Refer to p2-7 of ‘Project2_COSE474_SeungryongKim.pdf’
      ‘Unet_skeleton.py’: Refer to p5-6 of ‘Project3_COSE474_SeungryongKim.pdf’

      3) Load the trained model, which is given as 'UNet_trained_model.pth'

      4) Train it with CPU or GPU, and screen capture the test accuracy.

      Note) As the trained checkpoint parameters are used, you will train model only 1 epoch.

Original U-Net

- **Train “*UNet*”**
  - Train “UNet” model with “Pascal VOC 2012” datasets, and conduct image segmentation
    - (30 points) Unet model code: ‘Unet_skeleton.py’
      - Check out the channel size and fill in the blanks of the convolutional layers
    - (30 points) train/test module: 'modules_skeleton.py'
      - Please understand train/test codes, fill out the blanks that get output out of model, loss, optimizer, and backpropagation.
    - (20 points) main code: ‘main_skeleton.py’
      - Please understand the loading model, saving model, model initialization, setting optimizer and loss in 'Project2_COSE474_SeungryongKim.pdf'.

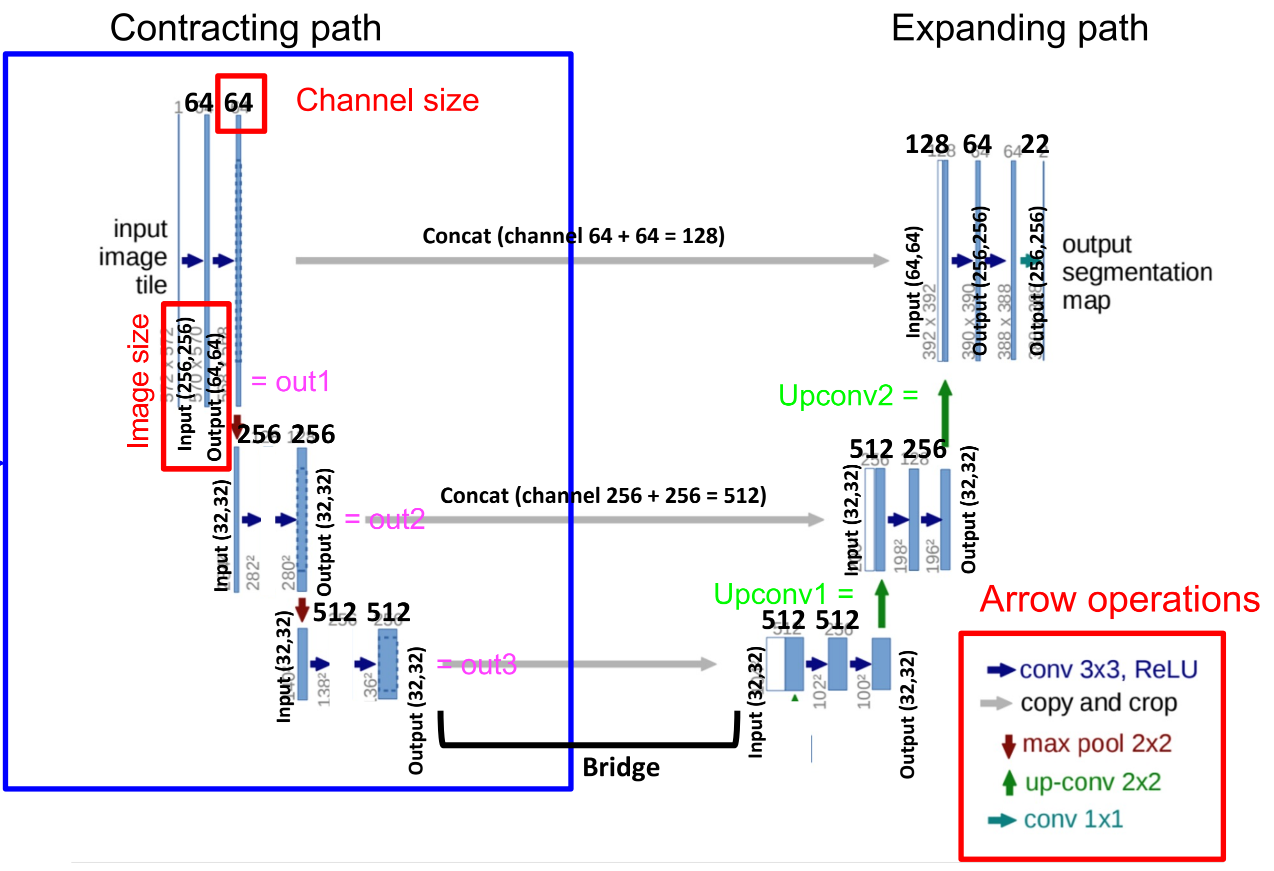
Original U-Net

- **Implement “*ResNet-encoder-Unet*”**
  - Replace the encoder of original U-net with ResNet-50

ResNet-50

Layer number	Network
Layer 1	7x7 conv, channel = 64, stride = 2 3x3 max pool, stride = 2
Layer 2	[1x1 conv, channel = 64, 3x3 conv, channel = 64, 1x1 conv, channel = 256] x 2 [1x1 conv, channel = 64, stride = 2 3x3 conv, channel = 64, 1x1 conv, channel = 256] x 1
Layer 3	[1x1 conv, channel = 128, 3x3 conv, channel = 128, 1x1 conv, channel = 512] x 4

U-net



Note) In layer 2, reduce an activation map in half by using the strided convolution (stride = 2) at the last Residual block.

Original U-Net

- **Implement "*ResNet-encoder-Unet*"**
  - Train "ResNet-encoder-Unet" model with "Pascal VOC 2012" datasets, and conduct image segmentation

    Tip) If a data processing step takes too long, try using only 20 images first

    - Optimize parameters with *Adam optimizer* and *cross Entropy Loss*
    - Procedure

      1) Download Pascal VOC 2012 Dataset using page 2 in this slide.

      2) Fill in the blanks of the skeleton code ('resnet_encoder_unet_skeleton.py')

      Refer to the page 5, 6, and 9 in this slide and Resnet-50 code ('resnet50_skeleton.py') of the previous assignment

      3) Load the trained model , which is given as 'resnet_unet_trained_model.pth'

      4) Train it with CPU or GPU, and screen capture the test accuracy.

      Note) As the trained checkpoint parameters are used, you will train model only 1 epoch.

Original U-Net

- **Implement "*ResNet-encoder-Unet*"**
  - Train "ResNet-encoder-Unet" model with "Pascal VOC 2012" datasets, and conduct image segmentation
    - (20 points) ResNet-encoder-Unet model code: 'resnet_encoder_unet_skeleton.py'
      - Please understand the architecture of the Unet in page 5, 6, and 9.
      - Check out the concatenation functions in pytorch.

    - (30 points) train/test module: 'modules_skeleton.py'

      This is the same as the codes in p8

      - Please understand train/test codes, fill out the blanks that get output out of model, loss, optimizer, and backpropagation.

    - (20 points) main code: 'main_skeleton.py'
      - Please understand the loading model, saving model, model initialization, setting optimizer and loss in 'Project2_COSE474_SeungryongKim.pdf'.

Encoder-Decoder Implementation

Due on Dec. 5 (Mon.), 03:29 pm (in Blackboard)

(late policy: 25% off per a day late)

You must submit the **code** with the **report**.
(1 page with free format, including the description of your code, results, and discussions)

The report should be written in **English**.

Please do NOT copy your friends' and internet sources.

Please start your project EARLY.

Thank you!

Q & A