Volumetric Segmentation of 3D Images

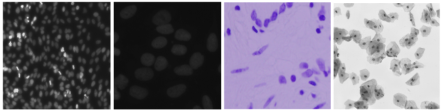
Shubham Sharma(M.Sc.-Ph.D.(OR), IIT Bombay)
M.Sc. Ph.D.(OR) Phase-1 Project Presentation

November 26, 2019

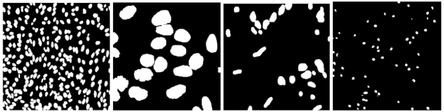
Index

Segmentation

- Process of partitioning or segmenting different objects of interests from an image
- Pixel wise classification of images into different segments
- Main goal is to change the representation of the image in a more meaningful image that is easier to analyze



(a) Original Images



(b) Corresponding labels

Figure 1: An example of nuclei segmentation; Source:Kaggle

Application of segmentation

1. In case of medical images it :-
   - Can be used in locating and detecting tumors and other pathologies
   - Can be used in anomaly detection
   - Can be used in surgery planning
   - Used in Intra-surgery navigation
2. Can be used in case of object detection for:-
   - Face detection
   - Locating images for satellite images
3. Can be used in case of recognition tasks for:-
   - Face detection
   - Locating images for satellite images

2D images

- This is an image to us

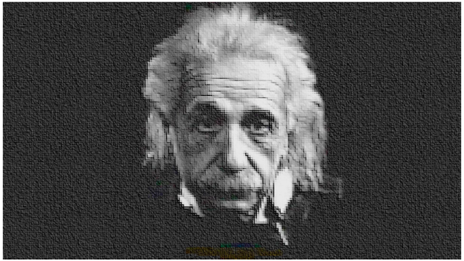


Figure 2: How humans see image; Source:Google

2D images

- This is an image to a computer

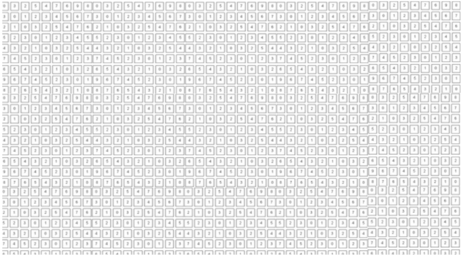


Figure 3: How computers see image; Source:Google

2D images

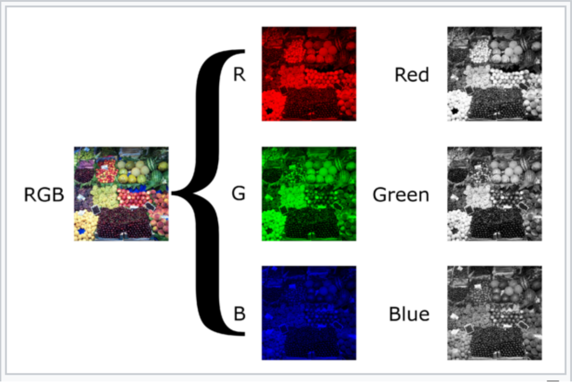


Figure 4: Composition of RGB from 3 Grayscale images; Source:Wikipedia

3D images

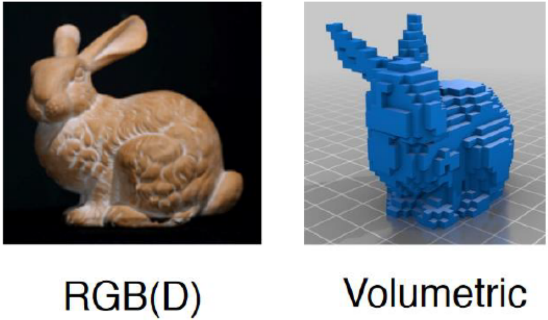


Figure 5: Example of RGB(D) and volumetric image; Source:Wikipedia

2D and 3D convolutions

- 2D convolutions

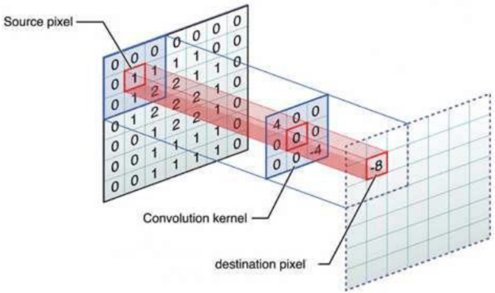


Figure 6: 2D Convolutions; Source:Wikipedia

2D and 3D convolutions

- Uses of 2D convolutions in image processing

Operation	Kernel ω	Image result g(x,y)
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Box blur	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur 3 × 3	$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	
Gaussian blur 5 × 5	$\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$	

Figure 7: The figure shows example of different convolution operations on an image; source: Wikipedia

2D and 3D convolutions

- 3D convolutions

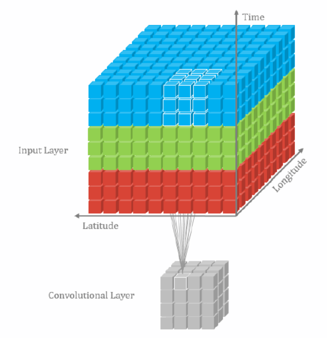


Figure 8: 3D convolutions; source: Wikipedia

U-net

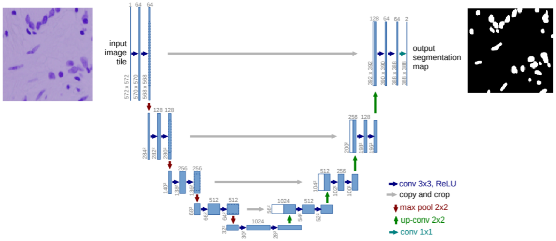


Figure 9: The U-net[1] architecture. Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on the top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations(concatenation); Source:google

3D U-net

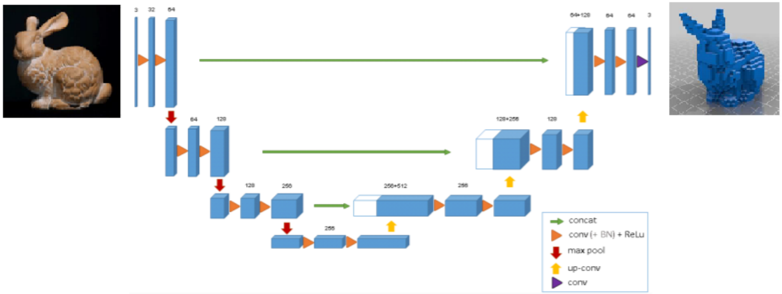


Figure 10: The 3D U-net[2] architecture. Blue boxes represent feature maps. The number of channels is denoted above each feature map.; Source:google

Data-set: Heart

- **Heart Data-set**
  - The Cardiac dataset from Medicaldecation.com contains 20 training images and 10 test images
  - Has x and y dimensions as 320 and 320 and z(depth) varying from 90-130 in the different training images
  - The intensities of pixels are ranging from 0-2000 roughly
  - The target is left atrium.

Data-set: Heart

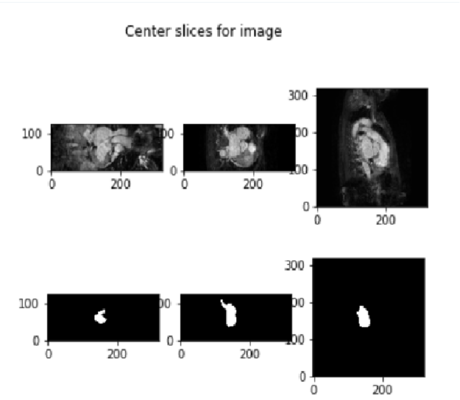


Figure 11: Original and Target image

Data-set: Spleen

- **Spleen Data-set**
  - The Spleen data-set contains 41 training images 20 testing images
  - Has x and y dimensions as 512 and 512 and z(depth) varying from 40-168 roughly in the different training images
  - The intensities of pixels are ranging from -1024 to 3072 roughly
  - The challenge is large ranging foreground size

Data-set: Spleen

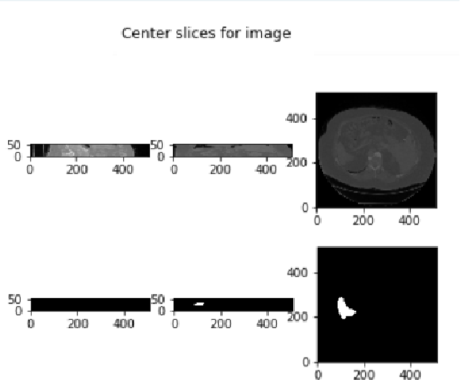


Figure 12: Original and Target image

Loss functions

- There are two loss functions that are used in this project:
  - Cross-entropy loss

$$CE\_Loss = -(y_i log(\hat{y}_i) + (1 - y_i) log(1 - \hat{y}_i)) \tag{1}$$

  - Dice loss

$$Dice\_Coefficient = \frac{2|A \cap B|}{|A| + |B|} \tag{2}$$

$$Dice\_Loss = 1 - Dice\_Coefficient \tag{3}$$

- ★ $|A \cap B|$ is common elements between set A and B
- ★ $|A|$ is the number of elements for set A(and likewise for set B).
- ★ $|A \cap B|$ can be approximated with element wise multiplication of target and the resultant matrix.

Experiments 1

- **Result 1**
  - Every image is resized to (1, 128, 128, 128) from cardiac dataset
  - Network 1 is made by taking the no of layers same as that of a simple U-net
    - No of trainable parameters 22,578,946
    - Trained with cross-entropy loss with adam optimizer with a learning rate of 1e-4
    - Trained for around 100 epochs
  - Network 2 is made by some layers from network 1
    - No of trainable parameters 5,602,306
    - Trained with dice loss with adam optimizer with a learning rate of 1e-5 with decaying factor of 1e-1 after every 100 epochs
    - Trained for around 100 epochs

Result 1: with cross-entropy loss

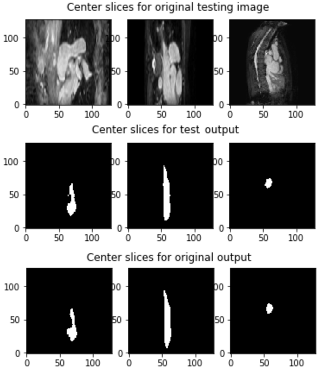


Figure 13: Segmentation on test image when cross-entropy loss is used

Result 1: with dice loss

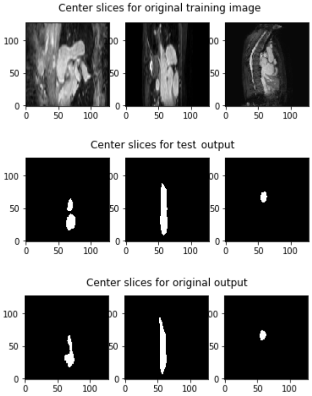


Figure 14: Segmentation on test images when dice loss is used

Experiments 2

- Tried to maintain the aspect ratio for the images
- Cardiac images are resized to (1, 160, 160, 64)
- Spleen images are resized to (1, 128, 128, 32)

Result 2: Cardiac images with cross-entropy loss

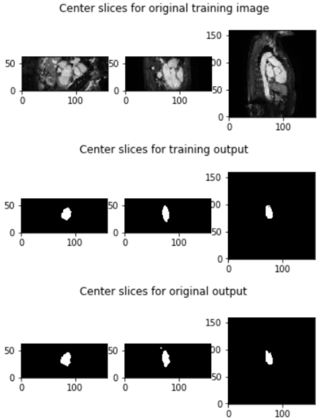


Figure 15: Segmentation on test image when cross-entropy loss is used

Result 2: Cardiac images with dice loss

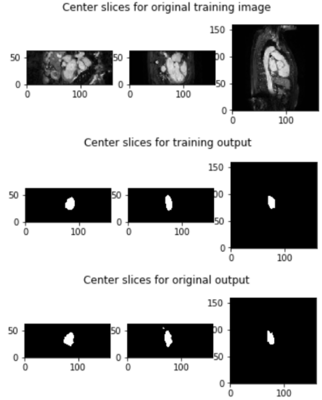


Figure 16: Segmentation on test image when dice loss is used

Result 2: Spleen images with cross-entropy loss

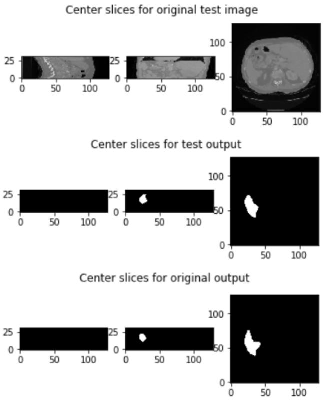


Figure 17: Segmentation on test image when cross-entropy loss is used

Result 2: Spleen images with dice loss

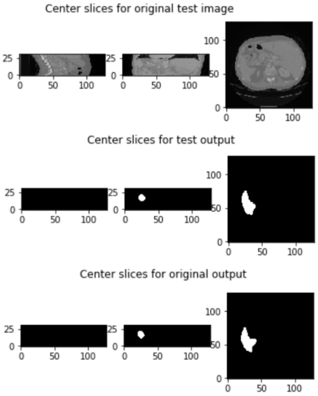


Figure 18: Segmentation on test image when dice loss is used

Results

Dice coefficient			
Data-set	**Loss Function**	**Train images**	**Test images**
Cardiac	Cross-entropy	0.879	0.768
	Dice loss	0.939	0.759
Spleen	Cross-entropy	0.863	0.816
	Dice loss	0.923	0.796

Table 1: Dice coefficient corresponding to different loss functions for both the data-sets

Future works

- A combination of both the losses can be taken

$$Loss = (\alpha) \times Dice\_Loss + (1 - \alpha) \times CE\_Loss \tag{4}$$

- Will have to tune the value of α and also the parameters in network architecture

Other works

- VGG13[4] like network has been trained for 3D MNIST
- Contains 10000 training images and 2000 test images
- The network was trained for 60 epochs with learning rate 1e-5 and adam optimizer

Other works

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv3d-1       [-1, 64, 16, 16, 16]           1,792
       BatchNorm3d-2       [-1, 64, 16, 16, 16]             128
              ReLU-3       [-1, 64, 16, 16, 16]               0
            Conv3d-4       [-1, 64, 16, 16, 16]         110,656
       BatchNorm3d-5       [-1, 64, 16, 16, 16]             128
              ReLU-6       [-1, 64, 16, 16, 16]               0
         MaxPool3d-7          [-1, 64, 8, 8, 8]               0
            Conv3d-8         [-1, 128, 8, 8, 8]         221,312
       BatchNorm3d-9         [-1, 128, 8, 8, 8]             256
             ReLU-10         [-1, 128, 8, 8, 8]               0
           Conv3d-11         [-1, 128, 8, 8, 8]         442,496
      BatchNorm3d-12         [-1, 128, 8, 8, 8]             256
             ReLU-13         [-1, 128, 8, 8, 8]               0
        MaxPool3d-14         [-1, 128, 4, 4, 4]               0
           Conv3d-15         [-1, 256, 4, 4, 4]         884,992
      BatchNorm3d-16         [-1, 256, 4, 4, 4]             512
             ReLU-17         [-1, 256, 4, 4, 4]               0
           Conv3d-18         [-1, 256, 4, 4, 4]       1,769,728
      BatchNorm3d-19         [-1, 256, 4, 4, 4]             512
             ReLU-20         [-1, 256, 4, 4, 4]               0
        MaxPool3d-21         [-1, 256, 2, 2, 2]               0
           Conv3d-22         [-1, 512, 2, 2, 2]       3,539,456
      BatchNorm3d-23         [-1, 512, 2, 2, 2]           1,024
             ReLU-24         [-1, 512, 2, 2, 2]               0
           Conv3d-25         [-1, 512, 2, 2, 2]       7,078,400
      BatchNorm3d-26         [-1, 512, 2, 2, 2]           1,024
             ReLU-27         [-1, 512, 2, 2, 2]               0
        MaxPool3d-28         [-1, 512, 1, 1, 1]               0
           Linear-29                  [-1, 100]          51,300
             ReLU-30                  [-1, 100]               0
           Linear-31                   [-1, 10]           1,010
================================================================
Total params: 14,104,982
Trainable params: 14,104,982
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.02
Forward/backward pass size (MB): 16.27
Params size (MB): 53.81
Estimated Total Size (MB): 70.09
----------------------------------------------------------------
```

Figure 19: Summary of the network used for classification

References I

[1] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[2] Özgün Çiçek, Ahmed Abdulkadir, Soeren S Lienkamp, Thomas Brox, and Olaf Ronneberger. 3d u-net: learning dense volumetric segmentation from sparse annotation. In *International conference on medical image computing and computer-assisted intervention*, pages 424–432. Springer, 2016.

[3] Amber L Simpson, Michela Antonelli, Spyridon Bakas, Michel Bilello, Keyvan Farahani, Bram van Ginneken, Annette Kopp-Schneider, Bennett A Landman, Geert Litjens, Bjoern Menze, et al. A large annotated medical image dataset for the development and evaluation of segmentation algorithms. *arXiv preprint arXiv:1902.09063*, 2019.

References II

[4] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[5] Xiaofan Xu, David Corrigan, Alireza Dehghani, Sam Caulfield, and David Moloney. 3d object recognition based on volumetric representation using convolutional neural networks. In *International conference on articulated motion and deformable objects*, pages 147–156. Springer, 2016.