

Master's thesis

**Automatic Segmentation Pipeline
for Lung Immunofluorescent Images
Using Deep Learning**

Shu Isaka

Division of Electrical and Electronic Engineering
Graduate School of Engineering
Mie University

Contents

1	Introduction	1
1.1	Background	1
1.2	Research Objective	2
2	Related Works	4
2.1	Lung Development Analysis	4
2.2	Deep Learning Based Image Segmentation	5
2.2.1	History of Convolutional Neural Networks	5
2.2.2	Typical Layers in Convolutional Neural Networks	6
3	Experimental Materials	12
3.1	Lung Immunofluorescent Confocal Images	12
3.2	Image Annotation	12
4	Methods	17
4.1	Development of Annotation System	17
4.1.1	Motivation	17
4.1.2	System Requirements	17
4.1.3	System Architecture	18
4.1.4	Processing Flow	21
4.2	Construction of Dataset	22
4.2.1	Manual Image Annotation	22
4.2.2	Preprocessing	22
4.3	Learning Process	24
4.3.1	Experimental Environment and Parameters	24
4.3.2	Network Architecture	25
4.4	Evaluation Process	32
4.4.1	Postprocessing	32
4.4.2	Evaluation Method	32
4.5	Synthetic Image Generation	33

5 Results and Discussions	34
5.1 Segmentation	34
5.1.1 Experimental Results	34
5.1.2 Discussion	34
5.2 Synthetic Image Generation	39
5.2.1 Experimental Results	39
5.2.2 Discussion	39
6 Concluding Remarks	41
6.1 Conclusion	41
6.2 Further Works	41
Acknowledgment	43
Reference	44
Publication List	47
Appendix A Detailed Immunofluorescence Method	49
Appendix B Segmentation Results	51

List of Figures

1.1	Pipeline for automated lung image segmentation and lung image analysis	3
2.1	Input Features and Convolution Filter.	7
2.2	Convolution.	7
2.3	Max Pooling.	8
2.4	Fully Connected Layer	9
2.5	Definition of Input Features, Convolution Filter, and Output Features . .	10
2.6	Padded Input Features of Transposed Convolution	10
2.7	Transposed Convolution	11
3.1	Website of LungMAP.	13
3.2	Example of Lung Immunofluorescent Confocal Image	14
3.3	Object Examples	15
3.4	Example of Segmentation Label Image and Segmented Image	16
4.1	Screenshot of web based annotation system.	19
4.2	Entity Relationship Diagram	20
4.3	Dataflow diagram of web based annotation system.	21
4.4	Drawing Polygon for Segmentation.	22
4.5	Conceptual diagram of padding and cropping.	23
4.6	Augmentation example for cropped images.	23
4.7	Network Structure of Deep Lab v3+	31
5.1	Bar Chart of Test Results	35
5.2	Bar Chart of Train Results	36
5.3	Learning Curve	38
5.4	Caption	38
5.5	Synthetic Image Generation	40
B.1	FCN8s	51
B.2	SegNet	52
B.3	Deep Lab v3+	53
B.4	U-Net (lr=0.001, loss func.=MSE)	54

B.5	U-Net (lr=0.005, loss func.=MSE)	55
B.6	U-Net (lr=0.005, loss func.=Dice Loss)	56
B.7	U-Net (lr=0.005, loss func.=Gen. Dice Loss)	57

List of Tables

4.1	System Environment for Web-based Annotation Tool	18
4.2	Experimental Environment for Segmentation	24
4.3	Parameters for training multi-class deep learning segmentation models.	24
4.4	Meanings of Symbols in Network Architecture Table	25
4.5	FCN8s	26
4.6	SegNet	27
4.7	U-Net	28
4.8	DeepLab v3+	29
4.9	Layer 1	29
4.10	Layer 2	30
4.11	Layer 3	30
4.12	Layer 4	31
4.13	Parameters for a Deep Convolutional Generative Adversarial Network.	33
4.14	Parameters for Segmentation Mask Generator by U-Net.	33
5.1	Test Results	35
5.2	Training Results	36
5.3	Results	37

Chapter 1

Introduction

1.1 Background

Alveolar development is a critical stage of respiratory system (lung) development to increase the surface area of alveoli which supports the gas exchanging process for air-breathing. Prematurely born infants are born at an early stage of lung development so that they has great risk for bronchopulmonary dysplasia (BPD). Thus, the maturity of the lung at birth is an inescapable element of lung diseases. However, this respiratory system development involves complex interactions such as interactive gene networks and dynamic cross-talk among multiple cell types. Although these mechanisms should be unraveled, knowledge of the molecular and cellular biology responsible for lung development is severely lacking [1–3]. Therefore, it is important to understand the lung development mechanism at the molecular/cellular level.

Under these circumstances, the Molecular Atlas of Lung Development Program (LungMAP) was funded by the National Heart, Lung and Blood Institute (NHLBI) to unravel the respiratory system development mechanism [1, 4]. LungMAP aimed to build,

1. A mouse lung atlas that integrates gene expression, imaging, and anatomic analysis,
2. A human lung atlas to validate the molecular profile in lung cell types, and
3. An integrated, publicly accessible, and expandable database called “BREATH”.

BREATH database includes RNA-sequence, proteomics, lipidomics, metabolomics, imaging data and so on. In particular, fine-resolution imaging data is required at cellular and molecular levels to build a lung development atlas [2]. Therefore, lung immunofluorescent confocal microscopy, high-resolution imaging data which stains specific proteins, are used for analyzing the lung development mechanism in LungMAP. These images are color stained by antibodies to mark the specific proteins so that we can identify the regions of interest.

However, in many cases, these biomedical images could not be analyzed as it is because

they should be annotated by the experts to describe the detailed information. Lung immunofluorescent confocal image is also needed to be annotated and segmented to mark structural regions of interest [1, 5]. Moreover, even though the annotations are necessary, the number of specialists who can annotate it is limited and there will be also time constraints. Thus, knowledge limitation and time constraints are the big problem in biomedical image analysis.

Recently, there are lots of researches based on deep learning methods such as the convolutional neural networks (CNNs) for biomedical image analysis to solve these problems [6–8]. However, one of the drawbacks of such deep learning models is that these models need plenty of labeled training data. This labeled data also requires the manual annotations by the specialists the same as traditional biomedical image analysis. Moreover, some of the methods require both segmentation and classification tools simultaneously [7–9]. Lung immunofluorescent confocal image also requires not only classification but also segmentation task. This pixel-wise classification is generally called semantic segmentation.

However, there is only a few annotated lung immunofluorescent confocal images because of the previously noted problems. Thus, environmental improvement of annotating process for lung immunofluorescent confocal images and automation of semantic segmentation using few training dataset is an important task.

1.2 Research Objective

This dissertation aims to construct the pipeline shown in Figure 1.1 for automated lung image segmentation and lung development analysis. This pipeline includes,

- I. **Data Collection** for obtaining the annotations of lung images,
- II. **Structure Segmentation** for level 1 segmentation identifying tissue structure,
- III. Individual Segmentation for level 2 segmentation identifying cell structure in level 1 segmented objects,
- IV. Image Feature Extraction for calculating the image features, and
- V. Analysis for combining the obtained image information with the gene expression data.

This paper describes I and II (shown as red marks in Figure 1.1) as the first step of this pipeline. For the data collection part, a web-based image annotation tool was developed. For the structure segmentation part, the state-of-the-art deep learning CNNs were applied to on semantic segmentation for lung immunofluorescent confocal images, and evaluation experiments were conducted. Furthermore, the feasibility of deep convolutional generative adversarial network (DCGAN) [10] was discussed for the synthetic lung immunofluorescent image generation to improve the semantic segmentation accuracy.

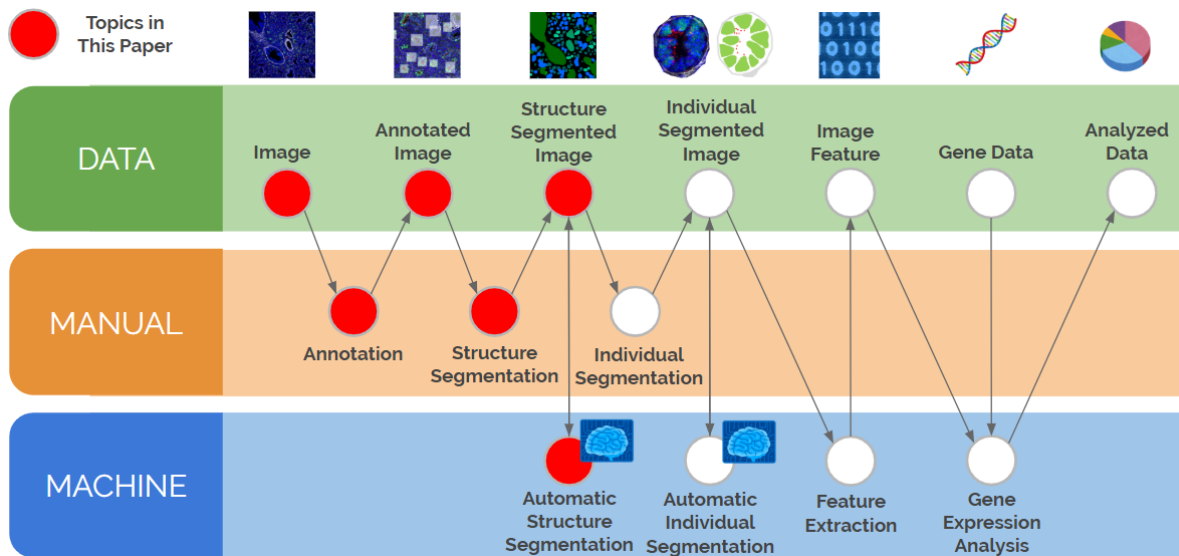


Figure 1.1: Pipeline for automated lung image segmentation and lung image analysis

The rest of this paper is organized as follows. Chapter 2 describes the related works on biomedical image analysis and deep learning-based semantic segmentation methods. Chapter 3 shows the image resources used in this research. Chapter 4 provides the dataset construction and the experimental setup using deep learning models, which is based on semantic segmentation and synthetic image generation, for the lung immunofluorescent confocal images. Chapter 5 shows experimental results and discusses the remaining challenges. Finally, Chapter 6 concludes this dissertation.

Chapter 2

Related Works

2.1 Lung Development Analysis

In the field of developmental biology, an important challenge is to unravel the mechanisms of organ formation at a gene, cellular level and differentiation of the diverse cell types that compose the embryo [3]. In particular, the lung is a complex organ with high cellular heterogeneity, and its development and maintenance are required for interactive gene networks and dynamic cross-talk among multiple cell types. Furthermore, it is difficult to understand the cause of abnormal lung development and adult lung disorder at the molecular level until we understand its normal development [1]. Therefore, many research projects aim to clarify the lung development/formation at gene/cellular/molecular level, and these projects are important for the establishment of a new medical treatment or the development of medical science [1, 3, 11, 12]. In particular, imaging data provides us the foundation to build the lung atlas. Therefore, it is required to understand these visual data and identify a specific region in the images by annotation [1, 13].

In the past research projects which focus on immunofluorescent confocal images, such as the researches by the Human Protein Atlas [14], immunofluorescent images were annotated manually by a specialist for the analyses [13]. Although the LungMAP team has developed an automated image annotation tool [1], the tool asks the user to input the region of interests for identification and annotation. In other words, the manual process is employed to annotate the lung immunofluorescent images. The number of tissue images is increasing, and also, 3D images of the lung will be used for the analysis in the future. It is hard to annotate all of them manually. Thus, the demand for an automated annotation tool without manual input is now growing, and it would be of help to researchers [15].

2.2 Deep Learning Based Image Segmentation

2.2.1 History of Convolutional Neural Networks

Deep Convolutional Neural Networks (CNNs) was originally proposed as the handwritten character recognition method by Le Cun et al. in 1998 [16]. Their networks, which is also called LeNet, consists of convolution layers, sub-sampling layers, and fully connected layers. It was designed to ensure some degree of shift, scale, and distortion invariance with local receptive fields, weight replication, and spatial or temporal subsampling [16].

In 2012, AlexNet, proposed by Krizhevsky et al., marked the highest performance in the ILSVRC-2012 competition. Compared with LeNet-5 [16] that has 60K parameters, AlexNet [17] contains 60M parameters and it is 1,000 times greater than LeNet-5. These enormous parameters could be calculated on Graphic Processing Unit (GPU) in 2012, therefore, CNNs were improved with the rapid growth of processing units.

In 2015, Long et al. proposed fully convolutional networks (FCN) and it marked 62.2% mean IoU¹ (improved 20% relatively) in PASCAL VOC 2012 [18]. They replaced fully connected layers in contemporary classification networks (AlexNet [17], the VGG net [19], and GoogLeNet [20]) with convolution layers. Their work allowed CNNs which were typically used for classification tasks to do semantic segmentation, pixels-to-pixels class prediction.

In the same year, Badrinarayanan et al. proposed SegNet, which was a deep convolutional encoder-decoder architecture for semantic segmentation [21]. SegNet only stores the max-pooling indices of the feature maps and used them in its decoder network. Thus, GPU inference memory is smaller than other networks like FCN, which stores a feature map extracted in the encoder network.

Moreover, Ronneberger et al. proposed U-Net which was used for biomedical image segmentation in 2015. U-Net has the connection between the encoder network and the decoder network. This connection transmits the features extracted in the encoder network and improves the segmentation accuracy.

In 2018, Chen et al. proposed DeepLab v3+, which was an encoder-decoder network with atrous separable convolution for semantic image segmentation. The encoder module encodes multi-scale contextual information by applying atrous convolution at multiple scales, while the simple yet effective decoder module refines the segmentation results along object boundaries [22].

¹Intersection over Union. That is one of the evaluation index for segmentation.

2.2.2 Typical Layers in Convolutional Neural Networks

This section describes the definition of typical layers in the CNNs based on the work by Dumoulin et al., 2018 [23].

Input Layer

Input layer is generally image data in classification tasks or segmentation tasks, and this is one of the important points of CNN compared to traditional neural networks. In other words, CNNs do not need manual feature extraction for learning because it can extract features automatically by optimizing the deep multiple layers' weights.

Convolution Layers

Convolution layers work as an image feature extraction layers. When I define the 4×4 input features and 3×3 convolution filter (or kernel) as Figures 2.1 (a) and (b), output features will be calculated by Equation 2.1 (Figure 2.2.) In the learning process, weights in the filter will be learned and optimized. Moreover, there are some other parameters for convolution such as padding size, stride size, kernel initializer, and so on. Padding is a process for adding some value (if 0 is given, it is called zero padding) on each side of the input features, and it affects the output feature size. Stride size means how many pixels are slid for filtering, for example, stride size of Figure 2.1 is 1. Kernel initializer is the method to initialize the filter (kernel) weights like He Normal Initialization [24].

$$0 \times 0 + 1 \times 0.5 + 3 \times 1 + 1 \times 0.5 + 4 \times 1 + 0 \times 1 + 1 \times 1 + 5 \times 0 + 0 \times 0 = 9 \quad (2.1)$$

Pooling Layers

Pooling layers work for aggregating the input features locally, by taking Max Pooling as example. Figure 2.3 shows an example of max pooling with pooling size (2, 2), without padding. In this process, the maximum value will be delivered to the output feature.

Fully Connected Layers

This layer works as a recognition layer for classification tasks. Fully connected layer connects all input and output features mutually like Figure 2.4, and all paths have weights. In the learning process, these weights will be learned. The output features in Figure 2.4 can be calculated with Equation 2.2. For instance, y_0 will be calculated with Equation 2.3.

0	1	3	2
1	4	0	2
1	5	0	0
2	3	1	2

0	0.5	1
0.5	1	1
1	0	0

(a) 4×4 Input Features (b) 3×3 Convolution Filter

Figure 2.1: Input Features and Convolution Filter.

0 <small>$\times 0$</small>	1 <small>$\times 0.5$</small>	3 <small>$\times 1$</small>	2
1 <small>$\times 0.5$</small>	4 <small>$\times 1$</small>	0 <small>$\times 1$</small>	2
1 <small>$\times 1$</small>	5 <small>$\times 0$</small>	0 <small>$\times 0$</small>	0
2	3	1	2

9	13.5
9.5	7.5

Figure 2.2: Convolution.

$$y_j = \sum_i w_{ij}x_i + w_{b_j}b \quad (2.2)$$

$$y_0 = w_{00}x_0 + w_{10}x_1 + w_{20}x_2 + w_{30}x_3 + w_{40}x_4 + w_{b_0}b \quad (2.3)$$

Upsampling Layers

In semantic segmentation tasks with deep learning, like FCN [25], SegNet [21], and U-Net [6], the output image has higher dimensions than the input features calculated in the former convolution layers. Therefore, the upsampling process is necessary to project feature maps to a higher dimensional space and it is generally given by transposed

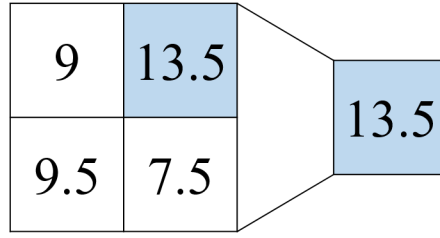


Figure 2.3: Max Pooling.

convolution. Here describes transposed convolution, which is also called as fractionally strided convolution, backwards strided convolution, or deconvolution.

Normal Convolution with Matrix Expression: When the input features, convolution filter, and output features are defined as shown in Figure 2.5, y_{00} will be calculated with Equation 2.4. Equation 2.4 can be converted as matrix multiplication shown in Equation 2.6, where input features X and filter weights \mathbf{w}_0 is defined by Equation 2.5. In these equations, input features X was originally two dimensional feature but in here it was expressed as a flattened matrix.

$$y_{00} = w_{00}x_{00} + w_{01}x_{01} + w_{02}x_{02} + w_{10}x_{10} + w_{11}x_{11} + w_{12}x_{12} + w_{20}x_{20} + w_{21}x_{21} + w_{22}x_{22} \quad (2.4)$$

$$X = \begin{pmatrix} x_{00} \\ x_{01} \\ x_{02} \\ x_{03} \\ x_{10} \\ \vdots \\ x_{32} \\ x_{33} \end{pmatrix}, \mathbf{w}_0 = \begin{pmatrix} w_{00} & w_{01} & w_{02} & 0 & w_{10} & w_{11} & w_{12} & 0 & w_{20} & w_{21} & w_{22} & 0 & 0 & 0 & 0 \end{pmatrix} \quad (2.5)$$

$$y_{00} = \mathbf{w}_0 X \quad (2.6)$$

Therefore, the relationship between input features X and output features Y of convolution is defined as Equation 2.8 where convolution filter weights W is defined as Equation 2.7.

$$W = \begin{pmatrix} w_{00} & w_{01} & w_{02} & 0 & w_{10} & w_{11} & w_{12} & 0 & w_{20} & w_{21} & w_{22} & 0 & 0 & 0 & 0 \\ 0 & w_{00} & w_{01} & w_{02} & 0 & w_{10} & w_{11} & w_{12} & 0 & w_{20} & w_{21} & w_{22} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & w_{00} & w_{01} & w_{02} & 0 & w_{10} & w_{11} & w_{12} & 0 & w_{20} & w_{21} & w_{22} \\ 0 & 0 & 0 & 0 & 0 & w_{00} & w_{01} & w_{02} & 0 & w_{10} & w_{11} & w_{12} & 0 & w_{20} & w_{21} & w_{22} \end{pmatrix} \quad (2.7)$$

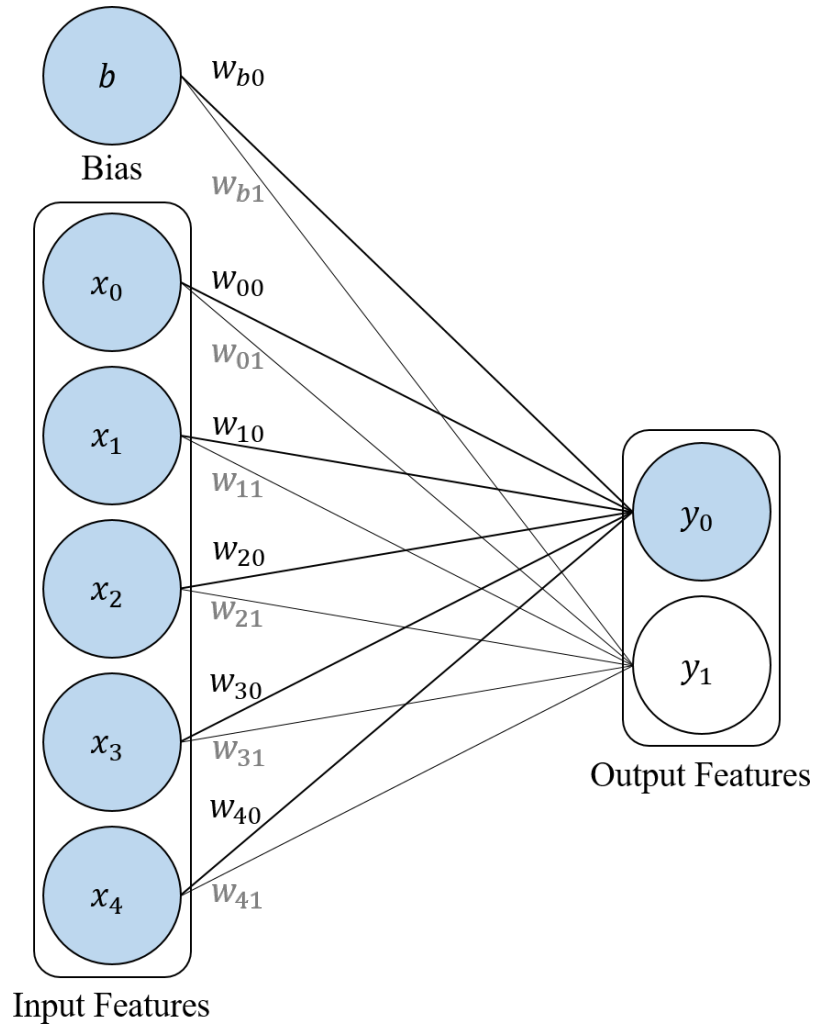


Figure 2.4: Fully Connected Layer

$$Y = WX \quad (2.8)$$

Transposed Convolution: Hence, Equation 2.9 shows an example of upsampling calculation where Y' defined as higher dimensional output features, W^T defined as a transposed matrix of W , and X' defined as lower dimensional input features. This transposed Equation 2.9 can be seen as a convolutional operation like Figure 2.7 for padded lower dimensional input features defined as Figure 2.6.

$$Y' = W^T X' \quad (2.9)$$

x_{00}	x_{01}	x_{02}	x_{03}
x_{10}	x_{11}	x_{12}	x_{13}
x_{20}	x_{21}	x_{22}	x_{23}
x_{30}	x_{31}	x_{32}	x_{33}

w_{00}	w_{01}	w_{02}
w_{10}	w_{11}	w_{12}
w_{20}	w_{21}	w_{22}

y_{00}	y_{01}
y_{10}	y_{11}

(a) Input Features (b) Convolution Filter (c) Output Features

Figure 2.5: Definition of Input Features, Convolution Filter, and Output Features

0	0	0	0	0	0
0	0	0	0	0	0
0	0	x'_{00}	x'_{01}	0	0
0	0	x'_{10}	x'_{11}	0	0
0	0	0	0	0	0
0	0	0	0	0	0

Figure 2.6: Padded Input Features of Transposed Convolution

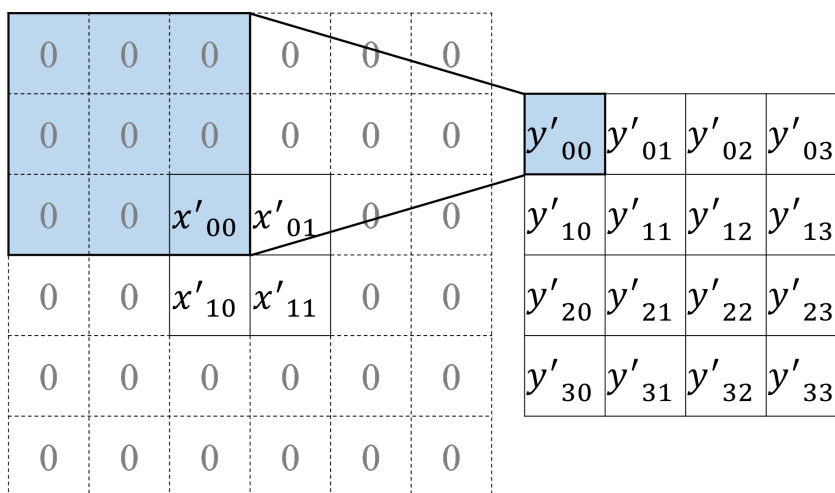


Figure 2.7: Transposed Convolution

Chapter 3

Experimental Materials

3.1 Lung Immunofluorescent Confocal Images

This study used lung immunofluorescent confocal images prepared by the LungMAP Consortium [U01HL122642¹] and downloaded from <https://lungmap.net/>, on October 1, 2019 [4] (Figure 3.1). The LungMAP consortium and the LungMAP Data Coordinating Center (1U01HL122638²) are funded by the National Heart, Lung, and Blood Institute (NHLBI). Figure 3.2 shows an example of lung immunofluorescent confocal image, and this image shows a sliced mouse lung tissue.

Lung immunofluorescence confocal images are stained by immunofluorescence method. This method stains several proteins at the same time as the colored image, and there are multiple combinations of stained proteins. This selective staining is realized by antibody reaction. For instance, Figure 3.2 is stained by antibodies to label specific proteins, e.g., “Sox9”, “Sftpc”, and “Acta2” (for more information, see Appendix A.) In addition, these stained proteins are the markers for specific types of cells. For instance, Sox9, which is green-stained protein, marks “chondrocyte”, “epithelial cell”, “unclassified fibroblast”, “pre-alveolar epithelial cell”. Furthermore, the dataset obtained by LungMAP contains the images of not only the human lung and also mouse lung for several time-series. Figure 3.2 is an example of a mouse (*Mus Musculus*) embryo 16.5 days lung tissue. This study focused on *Mus Musculus* embryo 16.5 days tissue as the first step.

3.2 Image Annotation

Image annotation describes where and what the region of interest is. This annotation is necessary for analyzing lung immunofluorescent confocal images because there are

¹Award number given by Tracking Accountability in Government Grants System (TAGGS)

²Grant number given by National Institute of Health (NIH)

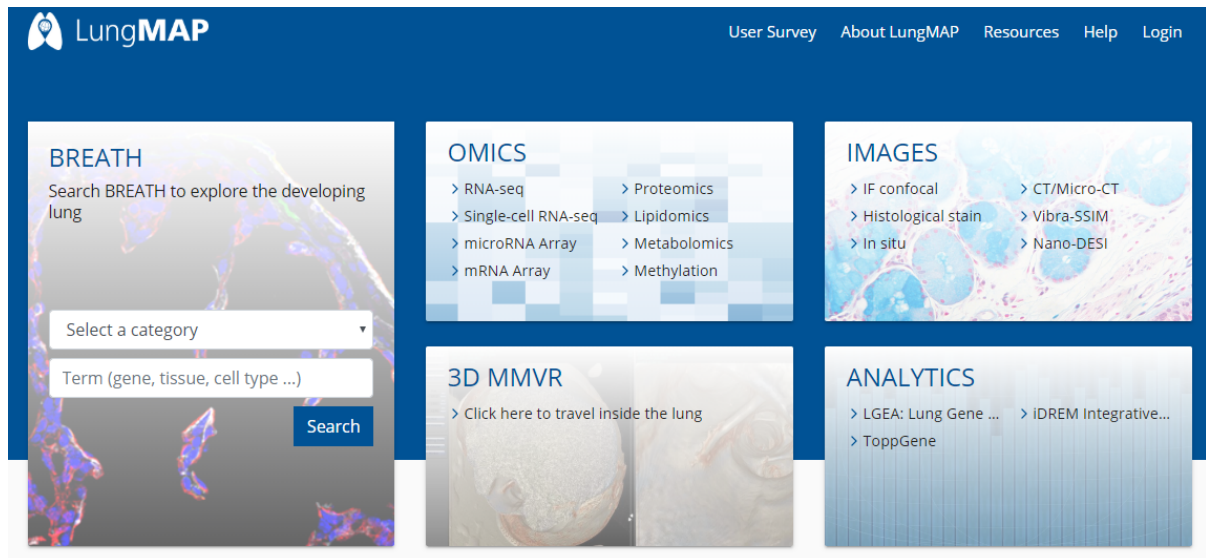


Figure 3.1: Website of LungMAP.

multiple object types in each image. Figure 3.3 shows examples of the objects in lung tissue. Moreover, these objects have complex shapes so that the segmentation label is required for analyzing them. Figure 3.4 (a) shows an example of segmentation label image, and (b) is the semitransparent segmentation label image over the lung tissue image.

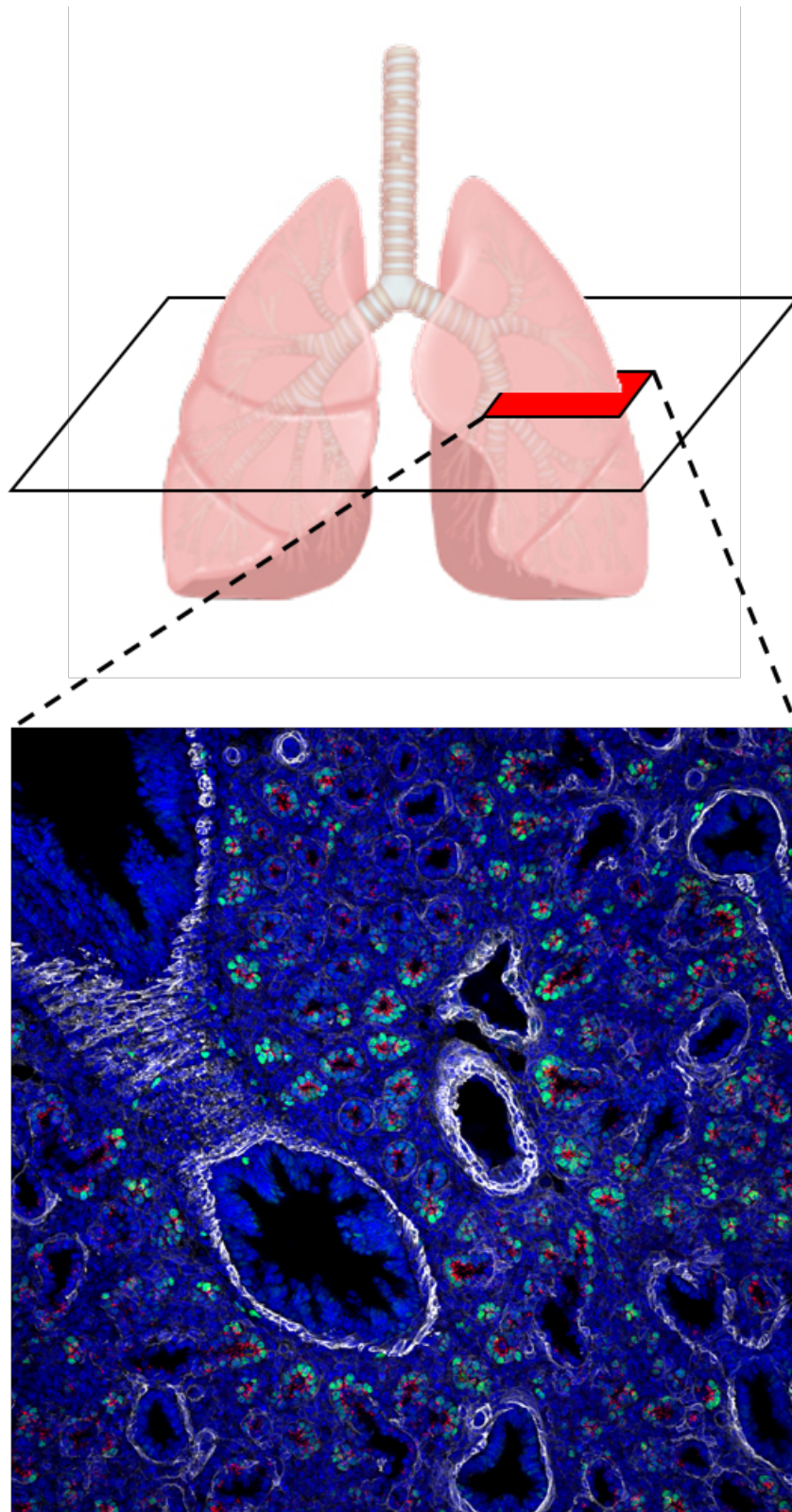
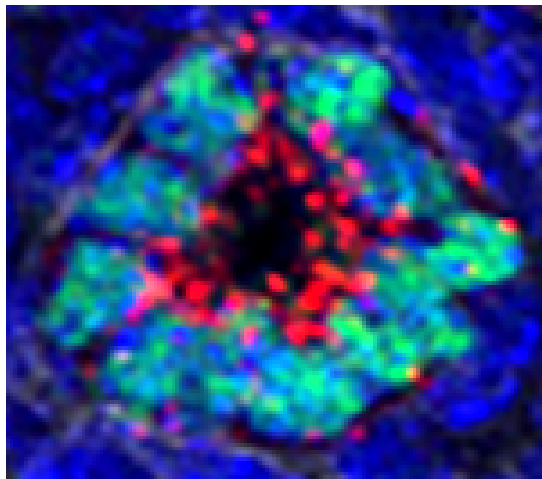
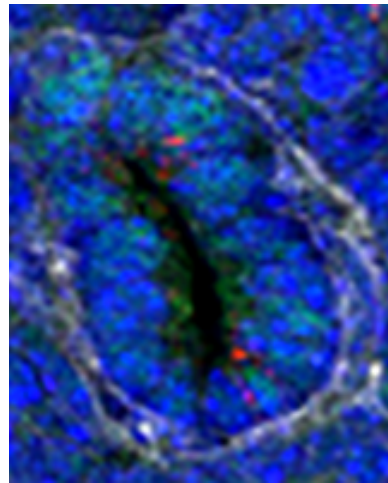


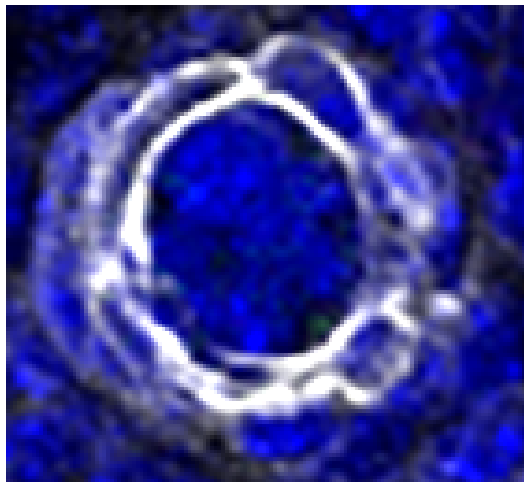
Figure 3.2: Example of Lung Immunofluorescent Confocal Image



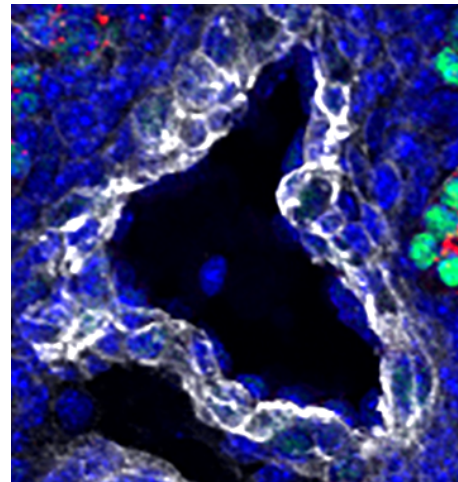
(a) Distal Acinar Tubule Bud



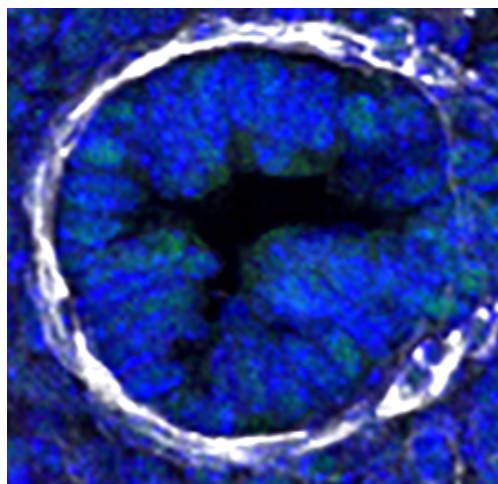
(b) Proximal Acinar Tubule



(c) Pulmonary Artery

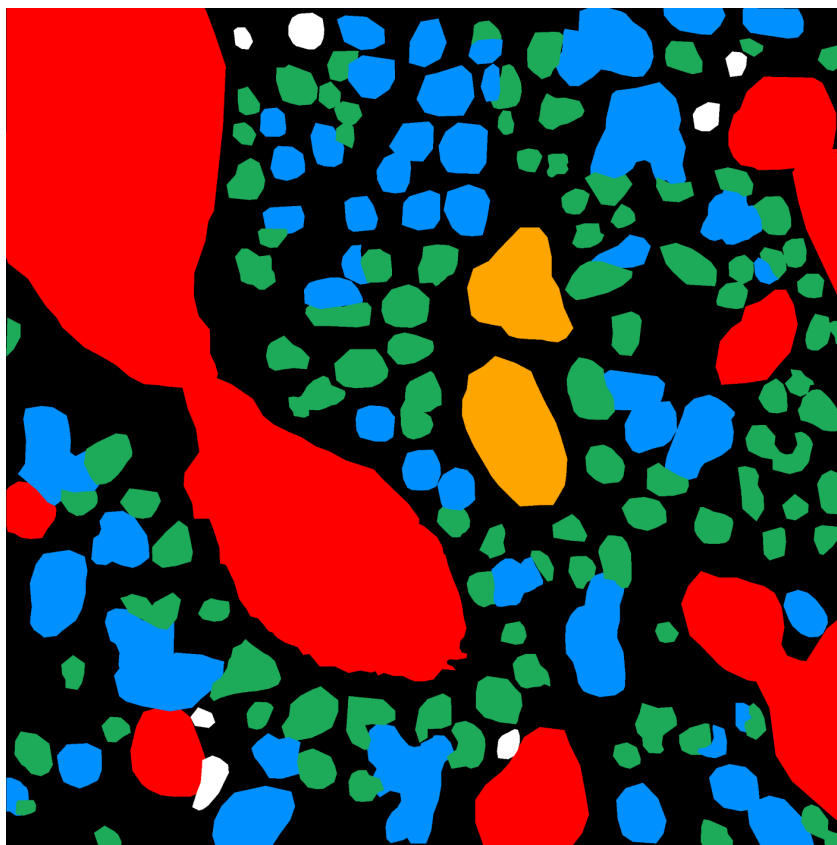


(d) Pulmonary Vein

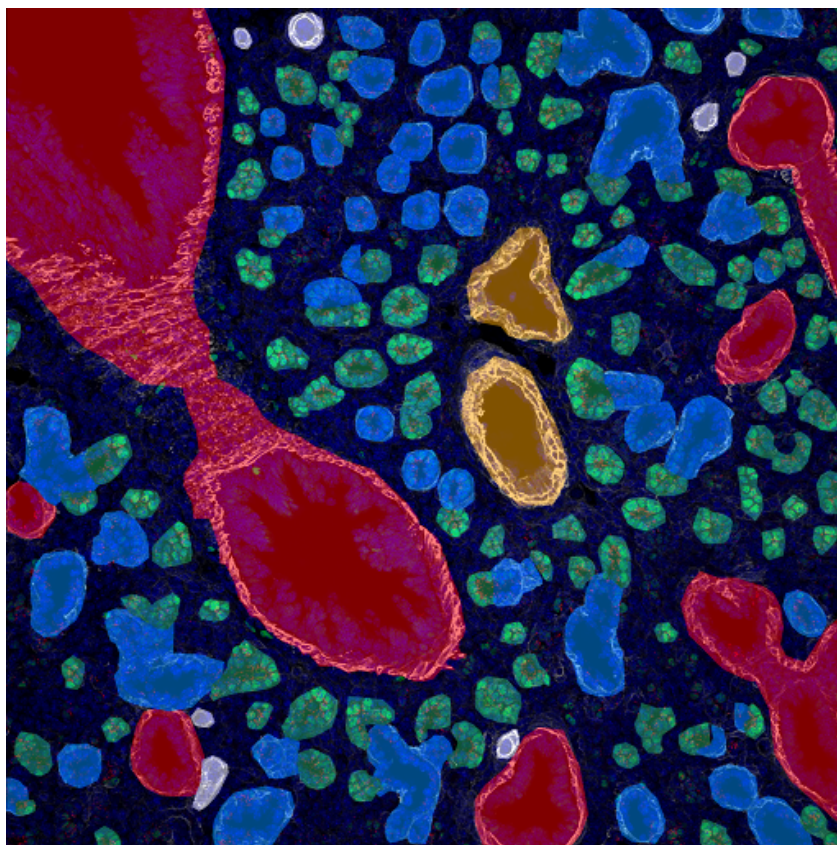


(e) Conductive Airway

Figure 3.3: Object Examples



(a) Segmentation Label Image



(b) Semitransparent Segmentation Label Image over the Tissue Image

Figure 3.4: Example of Segmentation Label Image and Segmented Image

Chapter 4

Methods

4.1 Development of Annotation System

4.1.1 Motivation

In the field of biomedical image analysis, annotations are necessary for analyzing those images or developing the automated system with machine learning techniques. These annotations are given in various formats, such as a label for Cancer versus Not Cancer, segmentation for regions of interest. For lung immunofluorescent image analysis, segmentation for multiple types of cells is required.

These annotation works require exclusive knowledge. However, there are few people who have such special knowledge, and thousands of images are required for analysis. Therefore, the annotation process gives a great deal of time to annotators. Thus, the demand for annotation support tools has been growing.

4.1.2 System Requirements

Most annotators like pathologists do not have enough time for annotating because of their daily works. Thus, the system should be ubiquitous; this means the developed system should not be subjected to any constraints on time and place. For these requirements, a web-based tool is one of the practical and meaningful approaches because the web-based tool will be easy to ask specialists and able to connect the database with the tool.

The developed system should have a function to search the images with specific conditions. The archived images have various information such as imaging condition, developmental stage (i.e., age) of the subjects, and the combination of antibodies for staining the lung tissue. This information is usually used to search the desired images for annotation. This function will make the annotators possible to access their desired image(s) easily.

In the system, polygon drawing function is also required. The desired cell should be extracted with polygon because the segmented shape is necessary for analysis or machine learning. This system should satisfy these requirements.

4.1.3 System Architecture

The system was developed by Java, Html, JavaScript, and CSS with Spring Framework. This system is now available at <http://galileo.ip.elec.mie-u.ac.jp/lung/tools/annotate/> through Apache Tomcat on Ubuntu server. Table 4.1 shows the detailed specifications of the webserver. Figure 4.1 is a screenshot of the annotation system. In the developed system, all of the image information and annotation information is stored in the database. Figure 4.2 shows an entity relationship diagram of the database.

Table 4.1: System Environment for Web-based Annotation Tool

Item	Specification
OS	Ubuntu 16.04.4 LTS (Xenial Xerus)
CPU	Intel Xeon E5-1410 4core/8thread 2.8GHz
RAM	24GB (4GBx6)
Storage 1	500GB (WDC WD5003ABYX-1)
Storage 2	500GB (WDC WD5003ABYX-1)
Java Version	1.8.0.171
Web Framework	Spring Framework 1.5.6
Web Server	Apache 2.4.18 + Tomcat 8.0.32
Database	MySQL 5.7.19 Oubuntu0.16.04.1

DOWNLOAD ALL ANNOTATION DATA as JSON FORMAT HELP MENU

SEARCH FORM

Species: Mus musculus
Age: E16.5
Protein: Red: Sftpc, Green: Sox9, White: Acta2
Magnification: 20X
SEARCH

RESULT TABLE

SHOW

2015-04-029_20X_C57Bl6_E16.5_LMM.14.24.4.46_SO...

ID	Species	Age	Magni
106	Mus musculus	E16.5	20X
107	Mus musculus	E16.5	20X
108	Mus musculus	E16.5	20X
109	Mus musculus	E16.5	20X

ANNOTATE FORM

Object: alveolar capillary bed

x1: - y1: - x2: - y2: -
ANNOTATE

Figure 4.1: Screenshot of web based annotation system.

4.1.4 Processing Flow

In the system, user can annotate by the following processes. For more detailed information, see Figure 4.3 related to 3 to 6, and Figure 4.4 related to 7 to 9.

1. Access to the system (<http://galileo.ip.elec.mie-u.ac.jp/lung/tools/annotate/>).
2. Log in to the system. (This process arrows to identify who gave the annotation.)
3. Search images from the database by filtering imaging condition.
4. Select image from the filtered image lists.
5. Draw rectangle of region of interest. (White rectangles in Figure 4.1 is an example of annotation rectangle.)
6. Select object class name and register annotation.
7. Select annotated rectangle.
8. Draw polygon based on the object's shape.
9. Register segmented object.

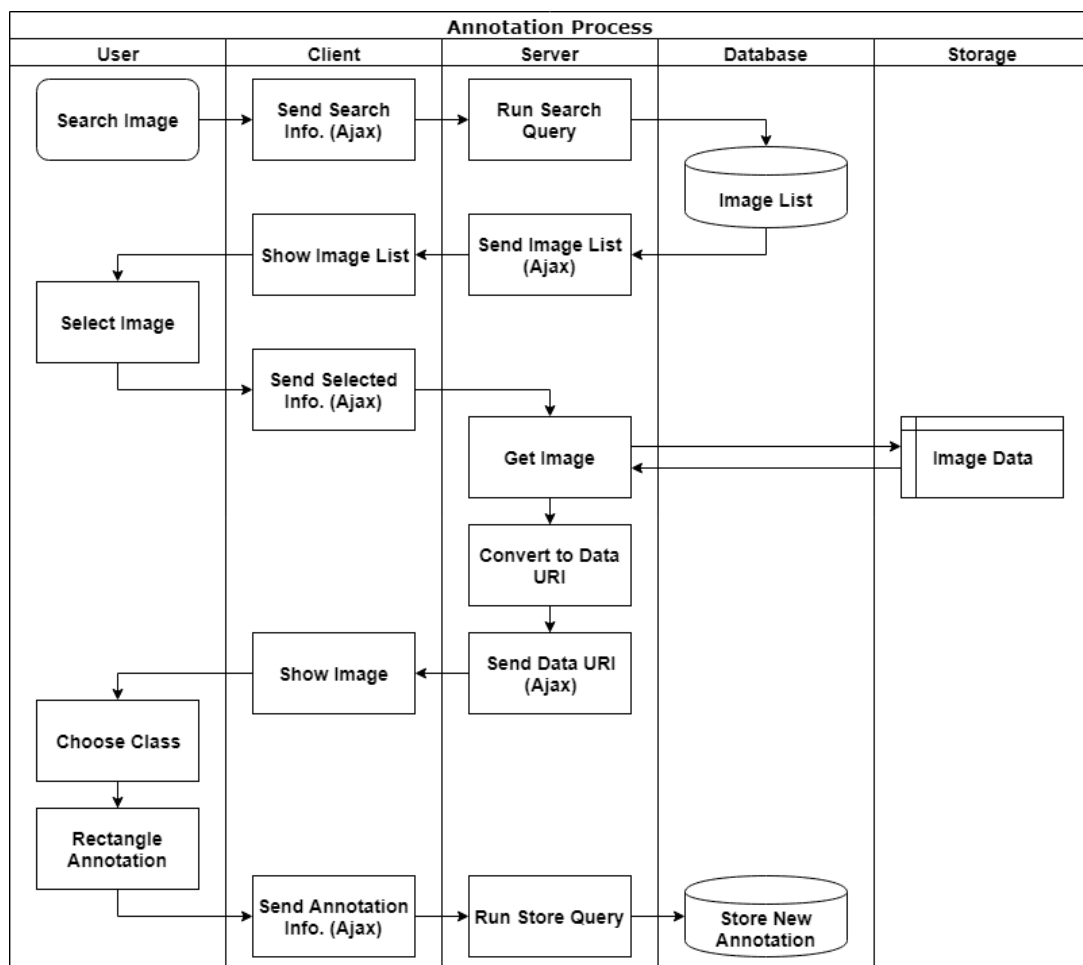


Figure 4.3: Dataflow diagram of web based annotation system.

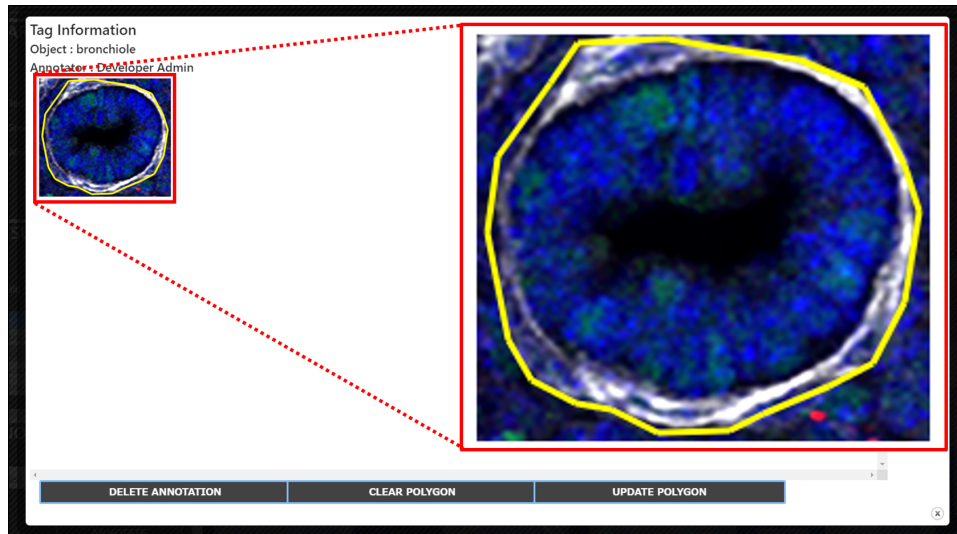


Figure 4.4: Drawing Polygon for Segmentation.

4.2 Construction of Dataset

4.2.1 Manual Image Annotation

In this study, the lung images were annotated under the experts' instructions. In the annotation process, 6 classes (Background, Conductive Airway, Distal Acinar Tubule Bud, Proximal Acinar, Pulmonary Artery, and Pulmonary Vein) for lung tissue of *Mus Musculus* Embryo 16.5 days were annotated. The used images were an indicator for proteins named Sftpc, Sox9, and Acta2, which were stained as red, green, and white, respectively.

4.2.2 Preprocessing

This thesis processed the image sets before training by the following methods proposed in [26]. This preprocessing method allows us to compute a big image with small processors and train image segmentation effectively.

Figure 4.5 shows the flow of this process. At first, the image size of the given image was changed from 2475×2475 to 1024×1024 . Subsequently, the 64-pixel mirror padding was applied to each side to reduce the inequality of cropping. In the cropping process, the padded images of 256×256 pixels were generated for every 128 pixels vertically and horizontally.

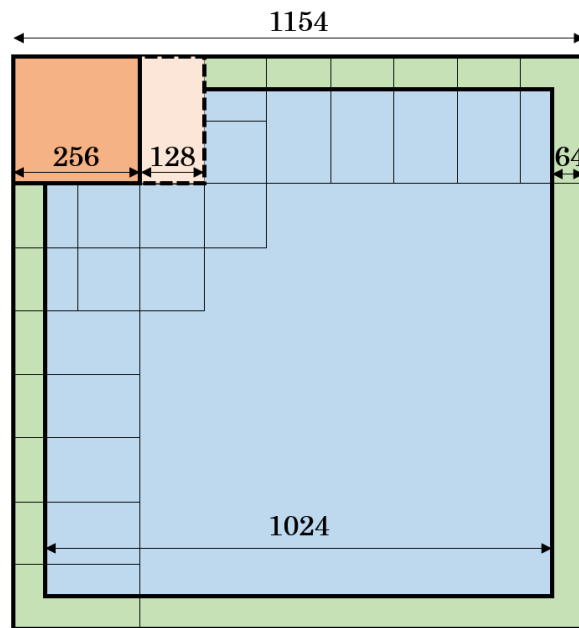


Figure 4.5: Conceptual diagram of padding and cropping.

As the next step, rotation (0, 90, 180, and 270 degrees) and horizontal flip were applied to the processed images (Figure 4.6). This process was only applied for training images.

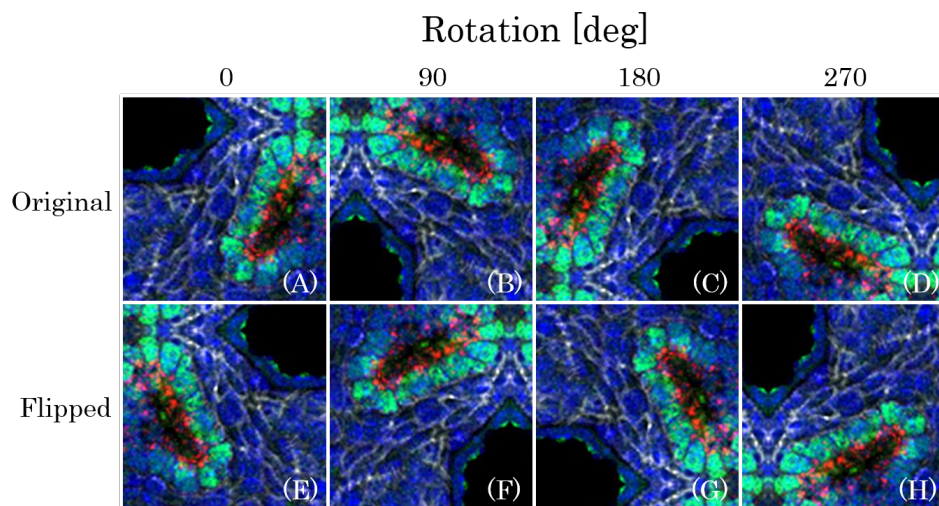


Figure 4.6: Augmentation example for cropped images.

Before the learning process, the images were divided into two groups, i.e., train and test sets by 4-fold cross validation. Finally, the number of training images was 1536 and the number of test images was 64 for each set.

4.3 Learning Process

4.3.1 Experimental Environment and Parameters

Table 4.2 shows an experimental environment in the learning process. This thesis compared state-of-the-art segmentation models based on CNNs, namely, FCN8s [18], U-Net [6], SegNet [21], and DeepLabv3+ [22]. For U-Net, this research expanded its output shape for multi-class segmentation. Table 4.3 shows models and their parameters, e.g., loss functions, learning rates, and optimizers used in this study. The trained models were used to predict the segmentation label for the test dataset.

Table 4.2: Experimental Environment for Segmentation

Item	Specification
OS	Ubuntu 16.04.5 LTS (Xenial Xerus)
CPU	Intel Core i7 6850K 6core/12thread 3.6GHz
GPU	NVIDIA GeForce GTX 1080Ti 11GB
RAM	32GB (8GBx4)
Storage 1	480GB (Intel SSDSC2BB48)
Storage 2	1TB (WDC WD10JPLX-00M)
Storage 3	1TB (WDC WD10JPLX-00M)
Deep Learning Framework	PyTorch 0.4.1
Language	Python3.5

Table 4.3: Parameters for training multi-class deep learning segmentation models.

	Model	Loss Function	Learning Rate	Optimizer
1	FCN8s	Mean Squared Error	0.001	Stochastic Gradient Descent
2	SegNet	Mean Squared Error	0.001	Stochastic Gradient Descent
3	DeepLab v3+	Mean Squared Error	0.001	Stochastic Gradient Descent
4	U-Net	Mean Squared Error	0.001	Stochastic Gradient Descent
5	U-Net	Mean Squared Error	0.005	Stochastic Gradient Descent
6	U-Net	Dice Loss	0.005	Stochastic Gradient Descent
7	U-Net	Generalized Dice Loss	0.005	Stochastic Gradient Descent

4.3.2 Network Architecture

This section describes the network architecture used in the experiment. Table 4.4 shows the meanings of symbols used in the following Tables (4.5 to 4.12) that describe network architectures.

FCN8s: Table 4.5 shows the architecture of FCN8s. X1 to X10 in Table 4.5 mean that the output feature given by the operation. For Element-wise Add shown in Table 4.5, the input features are described in notes.

SegNet: Table 4.6 describes the architecture of SegNet. In Max Unpooling Layer, index information was transported from the max pooling layer which had the same feature shape.

U-Net: Table 4.7 shows the architecture of U-Net. In the concatenate process, the output feature was generated by using the features shown in notes.

DeepLab v3+: Table 4.8, 4.9, 4.10, 4.11, 4.12, and Figure 4.7 describe the architecture of DeepLab v3+. Figure 4.7 shows the entire structure of deeplab including ResNet101 [27]. In ResNet, there are skip connections that connect the block and one previous block. Down sampling is used for skip connection in the first block to adjust the feature shape. The numbers after Loop Block in Table 4.9, 4.10, 4.11 mean the number of the same blocks.

Table 4.4: Meanings of Symbols in Network Architecture Table

Symbol	Meaning
Image	Input Image Data
Conv.	2D Convolution Layer
Max Pool.	Max Pooling Layer
Avg. Pool.	Average Pooling Layer
Max Unpol	Max Unpooling Layer
T. Conv.	2D Transposed Convolution Layer
Upsample	Upsampling Layer (same as Zooming in this thesis)
Element-wise Add	Mathematical Operation (Element-Wise Add for Features)
Concatenate	Mathematical Operation (Concatenation for Features)
ch	Number of Channels
k	Kernel Size (Filter Size) of Layer
p	Padding Size before Operation
s	Stride Size while Operation
output	Output Shape after Operation
ReLU	ReLU Activation Function
BN	Batch Normalization

Table 4.5: FCN8s

No.	Operation	ch	k	p	s	output	Activation	# of Params	Note
0	Image	3				(256, 256)			
1	Conv.	64	3	1	1	(256, 256)	ReLU	1792	
	Conv.	64	3	1	1	(256, 256)	ReLU	36928	
	Max Pool.	64	2	0	2	(128, 128)		0	X1
2	Conv.	128	3	1	1	(128, 128)	ReLU	73856	
	Conv.	128	3	1	1	(128, 128)	ReLU	147584	
	Max Pool.	128	2	0	2	(64, 64)		0	X2
3	Conv.	256	3	1	1	(64, 64)	ReLU	295168	
	Conv.	256	3	1	1	(64, 64)	ReLU	590080	
	Conv.	256	3	1	1	(64, 64)	ReLU	590080	
	Max Pool.	256	2	0	2	(32, 32)		0	X3
4	Conv.	512	3	1	1	(32, 32)	ReLU	1180160	
	Conv.	512	3	1	1	(32, 32)	ReLU	2359808	
	Conv.	512	3	1	1	(32, 32)	ReLU	2359808	
	Max Pool.	512	2	0	2	(16, 16)		0	X4
5	Conv.	512	3	1	1	(16, 16)	ReLU	2359808	
	Conv.	512	3	1	1	(16, 16)	ReLU	2359808	
	Conv.	512	3	1	1	(16, 16)	ReLU	2359808	
	Max Pool.	512	2	0	2	(8, 8)		0	X5
6	T. Conv.	512	3	1	2	(16, 16)	ReLU	2359808	X6
	Element-wise Add	512				(16, 16)	BN	1024	X4 + X6
7	T. Conv.	256	3	1	2	(32, 32)	ReLU	1179904	X7
	Element-wise Add	256				(32, 32)	BN	512	X3 + X7
8	T. Conv.	128	3	1	2	(64, 64)	ReLU + BN	295296	X8
9	T. Conv.	64	3	1	2	(128, 128)	ReLU + BN	73920	X9
10	T. Conv.	32	3	1	2	(256, 256)	ReLU + BN	18528	X10
11	Conv.	6	1	1	1	(256, 256)		198	

Table 4.6: SegNet

No.	Operation	ch	k	p	s	output	Activation	# of Params
0	Image	3				(256, 256)		
1	Conv.	64	3	1	1	(256, 256)	BN + ReLU	1920
	Conv.	64	3	1	1	(256, 256)	BN + ReLU	37056
	Max Pool.	64	2	0	2	(128, 128)		0
2	Conv.	128	3	1	1	(128, 128)	BN + ReLU	74112
	Conv.	128	3	1	1	(128, 128)	BN + ReLU	147840
	Max Pool.	128	2	0	2	(64, 64)		0
3	Conv.	256	3	1	1	(64, 64)	BN + ReLU	295680
	Conv.	256	3	1	1	(64, 64)	BN + ReLU	590592
	Conv.	256	3	1	1	(64, 64)	BN + ReLU	590592
	Max Pool.	256	2	0	2	(32, 32)		0
4	Conv.	512	3	1	1	(32, 32)	BN + ReLU	1181184
	Conv.	512	3	1	1	(32, 32)	BN + ReLU	2360832
	Conv.	512	3	1	1	(32, 32)	BN + ReLU	2360832
	Max Pool.	512	2	0	2	(16, 16)		0
5	Conv.	512	3	1	1	(16, 16)	BN + ReLU	2360832
	Conv.	512	3	1	1	(16, 16)	BN + ReLU	2360832
	Conv.	512	3	1	1	(16, 16)	BN + ReLU	2360832
	Max Pool.	512	2	0	2	(8, 8)		0
6	Max Unpool.	512	2	0	2	(16, 16)		0
	Conv.	512	3	1	1	(16, 16)	BN + ReLU	2360832
	Conv.	512	3	1	1	(16, 16)	BN + ReLU	2360832
	Conv.	512	3	1	1	(16, 16)	BN + ReLU	2360832
7	Max Unpool.	512	2	0	2	(32, 32)		0
	Conv.	512	3	1	1	(32, 32)	BN + ReLU	2360832
	Conv.	512	3	1	1	(32, 32)	BN + ReLU	2360832
	Conv.	256	3	1	1	(32, 32)	BN + ReLU	1180416
8	Max Unpool.	256	2	0	2	(64, 64)		0
	Conv.	256	3	1	1	(64, 64)	BN + ReLU	590592
	Conv.	256	3	1	1	(64, 64)	BN + ReLU	590592
	Conv.	128	3	1	1	(64, 64)	BN + ReLU	295296
9	Max Unpool.	128	2	0	2	(128, 128)		0
	Conv.	128	3	1	1	(128, 128)	BN + ReLU	147840
	Conv.	64	3	1	1	(128, 128)	BN + ReLU	73920
10	Max Unpool.	64	2	0	2	(256, 256)		0
	Conv.	64	3	1	1	(256, 256)	BN + ReLU	37056
11	Conv.	6	3	1	1	(256, 256)		3462

Table 4.7: U-Net

No.	Operation	ch	k	p	s	output	Activation	# of Params	Note
0	Image	3				(256, 256)			
1	Conv.	64	3	1	1	(256, 256)	BN + ReLU	1920	X1
	Conv.	64	3	1	1	(256, 256)	BN + ReLU	37056	
2	Max Pool.	64	2	0	2	(128, 128)		0	X2
	Conv.	128	3	1	1	(128, 128)	BN + ReLU	74112	
	Conv.	128	3	1	1	(128, 128)	BN + ReLU	147840	
3	Max Pool.	128	2	0	2	(64, 64)		0	X3
	Conv.	256	3	1	1	(64, 64)	BN + ReLU	295680	
	Conv.	256	3	1	1	(64, 64)	BN + ReLU	590592	
4	Max Pool.	256	2	0	2	(32, 32)		0	X4
	Conv.	512	3	1	1	(32, 32)	BN + ReLU	1181184	
	Conv.	512	3	1	1	(32, 32)	BN + ReLU	2360832	
5	Max Pool.	512	2	0	2	(16, 16)		0	X5
	Conv.	512	3	1	1	(16, 16)	BN + ReLU	2360832	
	Conv.	512	3	1	1	(16, 16)	BN + ReLU	2360832	
6	Upsample	512				(32, 32)		0	X4, X6
	Concatenate	1024				(32, 32)		0	
	Conv.	256	3	1	1	(32, 32)	BN + ReLU	2360064	
	Conv.	256	3	1	1	(32, 32)	BN + ReLU	590592	
7	Upsample	256				(64, 64)		0	X3, X7
	Concatenate	512				(64, 64)		0	
	Conv.	128	3	1	1	(64, 64)	BN + ReLU	590208	
	Conv.	128	3	1	1	(64, 64)	BN + ReLU	147840	
8	Upsample	128				(128, 128)		0	X2, X8
	Concatenate	256				(128, 128)		0	
	Conv.	64	3	1	1	(128, 128)	BN + ReLU	147648	
	Conv.	64	3	1	1	(128, 128)	BN + ReLU	37056	
9	Upsample	64				(256, 256)		0	X1, X9
	Concatenate	128				(256, 256)		0	
	Conv.	64	3	1	1	(256, 256)	BN + ReLU	73920	
	Conv.	64	3	1	1	(256, 256)	BN + ReLU	37056	
10	Conv.	6	1	0	1	(256, 256)		390	

Table 4.8: DeepLab v3+

No.	Operation	ch	k	p	s	d	output	Activation	# of Params	Note
0	Image	3					(256, 256)			
1	Conv.	64	7	3	2	1	(128, 128)	BN + ReLU	9536	
	Max Pool.	64	3	1	2	1	(64, 64)		0	
2	Layer 1	256					(64, 64)		191232	X0
3	Layer 2	512					(32, 32)		1222584	
4	Layer 3	1024					(16, 16)		26090496	
5	Layer 4	2048					(16, 16)		13073736	X
6	Conv. (X)	256	1	0	1	1	(16, 16)	BN + ReLU	524800	X1
7	Conv. (X)	256	3	6	1	6	(16, 16)	BN + ReLU	4719704	X2
8	Conv. (X)	256	3	12	1	12	(16, 16)	BN + ReLU	4719704	X3
9	Conv. (X)	256	3	18	1	18	(16, 16)	BN + ReLU	4719704	X4
10	Avg. Pool. (X)	2048					(1, 1)	BN+ReLU	0	
	Conv.	256	1	0	1	1	(1, 1)		524800	
	Upsample	256					(16, 16)		0	X5
11	Concatenate	1280					(16, 16)	BN + ReLU	0	X1 ... X5
	Conv.	256	1	0	1	1	(16, 16)		328192	
	Upsample	256					(64, 64)		0	X6
12	Conv. (X0)	48	1	0	1	1	(64, 64)	BN + ReLU	12384	X7
	Concatenate	304					(64, 64)		0	X6, X7
13	Conv.	256	3	1	1	1	(64, 64)	BN + ReLU	700928	
	Conv.	256	3	1	1	1	(64, 64)	BN + ReLU	590336	
	Conv.	6	1	0	1	1	(64, 64)		1542	
	Upsample	6					(256, 256)		0	

Table 4.9: Layer 1

Part	Operation	ch	k	p	s	d	output	Activation	# of Params
Down Sampling	Conv.	256	1	0	1	1	(64, 64)	BN	16896
First Block	Conv.	64	1	0	1	1	(64, 64)	BN + ReLU	4224
	Conv.	64	3	1	1	1	(64, 64)	BN + ReLU	36992
	Conv.	256	1	0	1	1	(64, 64)	BN	16896
Loop Block (2)	Conv.	64	1	0	1	1	(64, 64)	BN + ReLU	4224
	Conv.	64	3	1	1	1	(64, 64)	BN + ReLU	36992
	Conv.	256	1	0	1	1	(64, 64)	BN	16896

Table 4.10: Layer 2

Part	Operation	ch	k	p	s	d	output	Activation	# of Params
Down Sampling	Conv.	512	1	0	2	1	(32, 32)	BN	132096
First Block	Conv.	128	1	0	1	1	(64, 64)	BN + ReLU	33024
	Conv.	128	3	1	2	1	(32, 32)	BN + ReLU	147712
	Conv.	512	1	0	1	1	(32, 32)	BN	66560
Loop Block (3)	Conv.	128	1	0	1	1	(32, 32)	BN + ReLU	66792
	Conv.	128	3	1	1	1	(32, 32)	BN + ReLU	147712
	Conv.	512	1	0	1	1	(32, 32)	BN	66560

Table 4.11: Layer 3

Part	Operation	ch	k	p	s	d	output	Activation	# of Params
Down Sampling	Conv.	1024	1	0	2	1	(64, 64)	BN	526336
First Block	Conv.	256	1	0	1	1	(64, 64)	BN + ReLU	131584
	Conv.	256	3	1	2	1	(64, 64)	BN + ReLU	590336
	Conv.	1024	1	0	1	1	(64, 64)	BN	264192
Loop Block (22)	Conv.	256	1	0	1	1	(64, 64)	BN + ReLU	262656
	Conv.	256	3	1	1	1	(64, 64)	BN + ReLU	590336
	Conv.	1024	1	0	1	1	(64, 64)	BN	264192

Table 4.12: Layer 4

Part	Operation	ch	k	p	s	d	output	Activation	# of Params
Down Sampling	Conv.	2048	1	0	1	1	(16, 16)	BN	2101248
First Block	Conv.	512	1	0	1	1	(16, 16)	BN + ReLU	525312
	Conv.	512	3	2	1	2	(16, 16)	BN + ReLU	2360320
	Conv.	2048	1	0	1	1	(16, 16)	BN	1052672
Second Block	Conv.	512	1	0	1	1	(16, 16)	BN + ReLU	1049600
	Conv.	512	3	4	1	4	(16, 16)	BN + ReLU	2360320
	Conv.	2048	1	0	1	1	(16, 16)	BN	1052672
Third Block	Conv.	512	1	0	1	1	(16, 16)	BN + ReLU	1049600
	Conv.	512	3	8	1	8	(16, 16)	BN + ReLU	2360320
	Conv.	2048	1	0	1	1	(16, 16)	BN	1052672

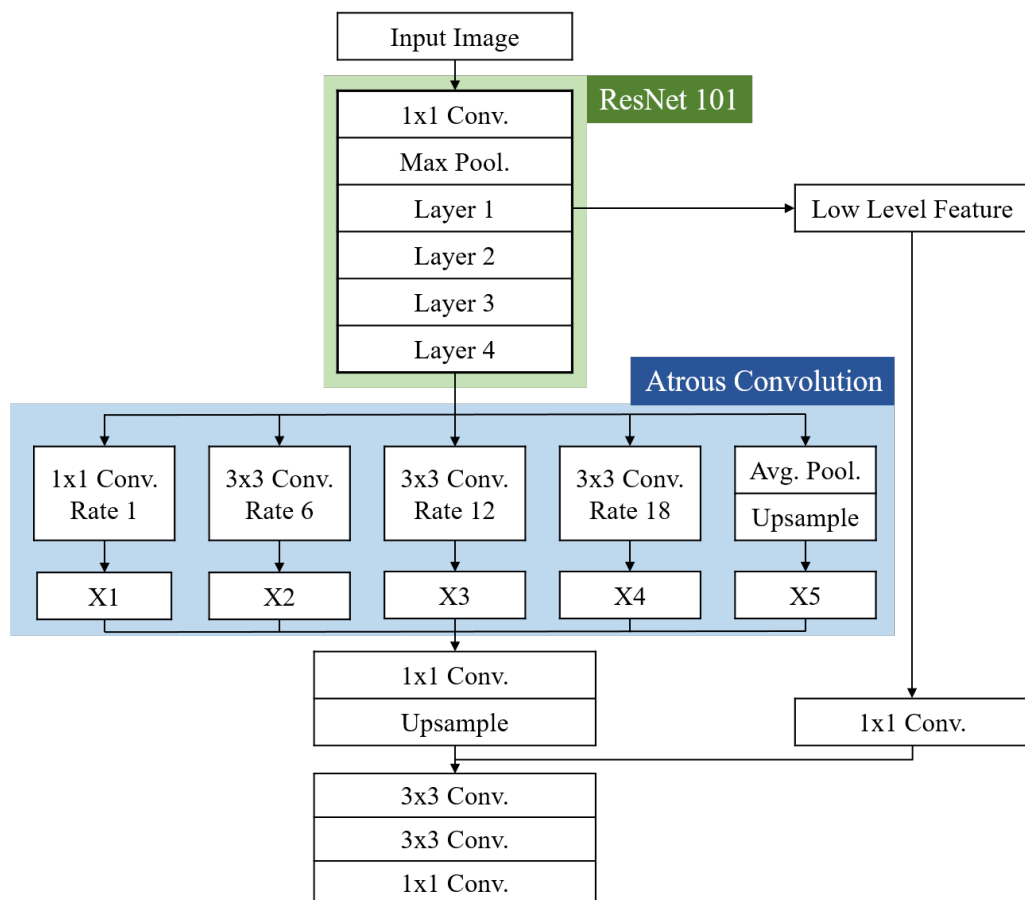


Figure 4.7: Network Structure of Deep Lab v3+

4.4 Evaluation Process

4.4.1 Postprocessing

In the postprocess, predicted images by the trained network were stitched into one image (1024×1024 pixels) by using the method in [26]. This paper stitched multiple images predicted by the trained network to generate the original size image before cropping by the method in [26]. There are over-wrapped pixels among each predicted image because of the cropping method. Therefore, each pixel should be evaluated as one specific class which is most possible by considering multiple predictions. When \mathbf{C}_i is given as predicted N-dimensional class probability vector which is defined by Equation 4.1 for the specific pixel of i^{th} predicted image, a summation of the predicted class probability vector $\mathbf{C}_{predict}$ can be defined by Equation 4.2. Furthermore, the final decision of predicted class for each pixel can be calculated by Equation 4.3 that takes the index of the most possible class.

$$\mathbf{C}_i = (p_0, p_1, \dots, p_N) \text{ where } N : \text{Number of Classes} \quad (4.1)$$

$$\mathbf{C}_{predict} = \sum_{i=1}^n \mathbf{C}_i \text{ where } n : \text{Number of Predictions} \quad (4.2)$$

$$C_{output} = i \text{ where } C_{predict,i} \text{ is } \max(\mathbf{C}_{predict}) \quad (4.3)$$

4.4.2 Evaluation Method

The segmentation accuracy is evaluated with Dice coefficient for each class and the entire image proposed in [26]. In Equations 4.1, 4.5, and 4.6, c means the class, N is the number of classes, \mathbf{U}_c means the one-hot vector for class c , $\mathbf{C}_{output_{xy}}$ is predicted class vector at any pixel (x, y) in the image which is calculated by equation 4.2 and 4.3 expressed as a one-hot vector, and $\mathbf{C}_{gt_{xy}}$ is ground truth class vector as a one-hot vector. Also, D_c means the dice coefficient for the specific class c , and D_{total} means the Dice coefficient for the entire image by judging the prediction result is correct or not.

$$\mathbf{U}_c = (U_1, \dots, U_N) \text{ where } U_i = \begin{cases} 0 & (i \neq c) \\ 1 & (i = c), \end{cases} \quad c = 1, 2, \dots, N. \quad (4.4)$$

$$D_c = \frac{2 \times \sum_x \sum_y |\mathbf{C}_{output_{xy}} \cdot \mathbf{C}_{gt_{xy}} \cdot \mathbf{U}_c|}{\sum_x \sum_y |\mathbf{C}_{output_{xy}} \cdot \mathbf{U}_c| + \sum_x \sum_y |\mathbf{C}_{gt_{xy}} \cdot \mathbf{U}_c|} \quad (4.5)$$

$$D_{total} = \frac{2 \times \sum_x \sum_y |\mathbf{C}_{output_{xy}} \cdot \mathbf{C}_{gt_{xy}}|}{\sum_x \sum_y |\mathbf{C}_{output_{xy}}| + \sum_x \sum_y |\mathbf{C}_{gt_{xy}}|} \quad (4.6)$$

Table 4.13: Parameters for a Deep Convolutional Generative Adversarial Network.

parameter	value
Epochs	50
Batch Size	10
Learning Rate	0.0002
zsize	100
ndf	128
ngf	128
Loss Function	Binary Cross Entoropy
Optimizer	Adam [28]

Table 4.14: Parameters for Segmentation Mask Generator by U-Net.

parameter	value
Epochs	100
Batch Size	8
Learning Rate	0.01
Loss Function	Dice Coefficient
Optimizer	Adam [28]

4.5 Synthetic Image Generation

In the synthetic image generation part, this thesis aimed to develop a synthetic image generator and segmentation mask generator using GAN. To achieve this, DCGAN [10] was applied as a synthetic image generator, and U-Net [6] was used as a segmentation mask generator.

DCGAN Training and Synthetic Image Generation

In the experiment, the network was trained with the parameters shown in Table 4.13, and the images which exist in reality were used for training the network. Consequently, 64 synthetic images were generated randomly using the trained generator. These synthetic images were used for the next step.

Segmentation Model Training and Segmentation Mask Generation

Subsequently, U-Net based segmentation model was trained with the parameters shown in Table 4.14. Finally, the trained network was applied to generate the segmentation mask for synthetic images made with the generator of DCGAN.

Chapter 5

Results and Discussions

5.1 Segmentation

5.1.1 Experimental Results

Tables 5.1 and 5.2 show the segmentation results for lung IF confocal images. Table 5.3 is the comparison among these prediction results. Figure 5.3 shows the learning curve of FCN8s, SegNet, DeepLab v3+, and U-Net with mean squared error, learning rate = 0.001. According to Table 5.1 and Figure 5.3, U-Net marked better accuracy compared to other networks. Thus this thesis compared several loss functions and learning rates for U-Net. Figure 5.4 shows the learning curve of U-Net with mean squared error and generalized dice loss (learning rate = 0.005.) More detailed segmentation results are shown in Appendix B.

5.1.2 Discussion

Despite the overall good results obtained with U-Net model, Table 5.1 shows that the accuracy for C5 (pulmonary artery) is low in many cases. This study has identified this problem was caused by class imbalance, and thus the deep learning models cannot learn this structure well across the images available in our dataset. However, the obtained results showed promise in identifying the multiple classes corresponding to different proteins utilized in the imaging setup.

As future work, the training dataset will be expanded, and it will be adapted for the top-performing U-Net segmentation model to obtain better segmentation accuracy for every class in the dataset. As a potential clinical application, the accurate segmentation of the regions can help us measure the protein expression levels. Further, various robust loss functions will be applied to solve class imbalance problem, and its effect will be evaluated in the final segmentation accuracy.

Table 5.1: Test Results

Model	Loss Func.	lr	C1	C2	C3	C4	C5	C6	Avg.	Overall
FCN8s	MSE	0.001	86.5	78.8	81.0	65.5	0.0	30.4	57.0	80.6
SegNet	MSE	0.001	85.4	76.7	79.5	64.1	0.0	61.5	61.2	80.0
DeepLab	MSE	0.001	83.9	78.2	78.0	63.6	0.0	68.5	62.0	79.0
U-Net	MSE	0.001	87.0	81.2	80.4	68.5	6.5	69.2	65.5	82.1
U-Net	MSE	0.005	87.5	81.5	81.8	69.6	29.0	75.5	70.8	83.0
U-Net	Dice	0.005	88.3	82.6	82.2	70.3	0.0	40.2	60.6	83.0
U-Net	Gen. Dice	0.005	86.6	73.2	81.6	63.2	0.0	0.0	50.8	79.8

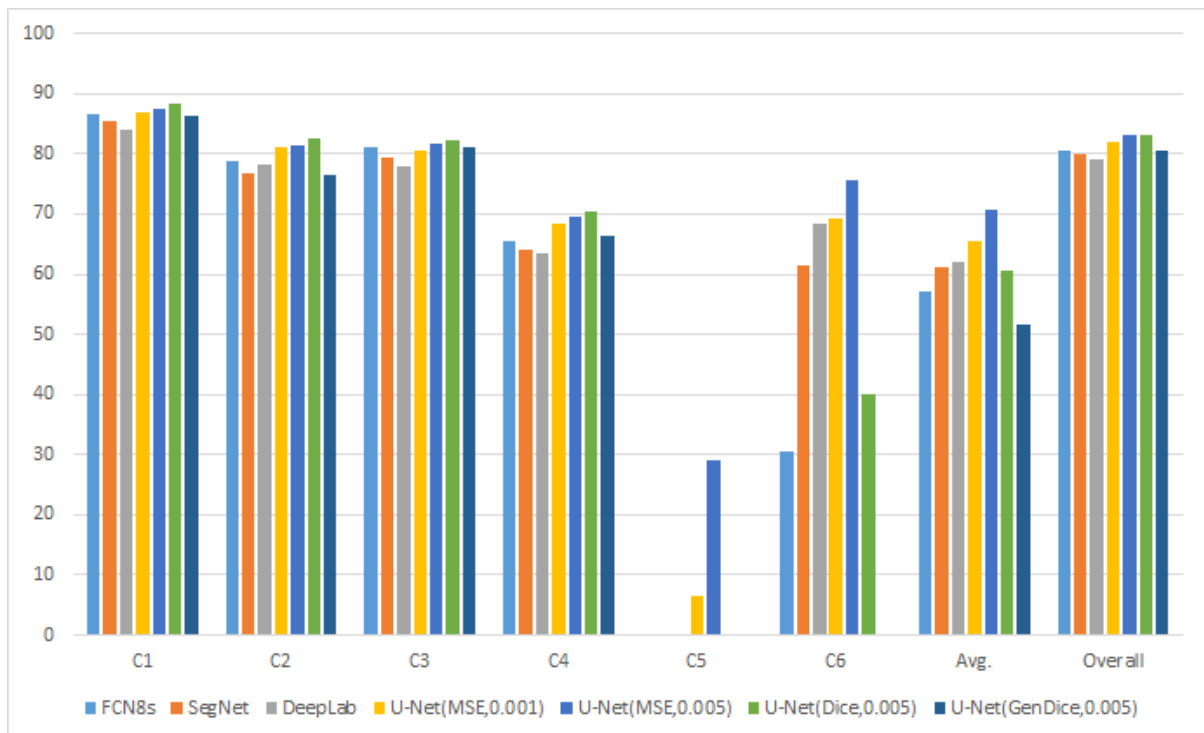


Figure 5.1: Bar Chart of Test Results

Table 5.2: Training Results

Model	Loss Func.	lr	C1	C2	C3	C4	C5	C6	Avg.	Overall
FCN8s	MSE	0.001	90.0	92.0	82.2	76.3	0.0	42.8	63.9	86.3
SegNet	MSE	0.001	96.3	97.6	93.8	95.0	0.0	92.9	79.3	95.7
DeepLab	MSE	0.001	89.1	87.8	81.5	73.5	0.0	78.9	68.1	85.6
U-Net	MSE	0.001	97.1	98.2	95.1	96.5	28.5	94.5	85.0	96.7
U-Net	MSE	0.005	98.2	98.9	96.9	97.7	93.3	98.6	97.3	98.1
U-Net	Dice	0.005	97.8	97.8	95.7	95.3	0.0	48.5	72.5	96.4
U-Net	Gen. Dice	0.005	92.8	83.0	88.9	87.6	0.0	0.0	58.7	88.7

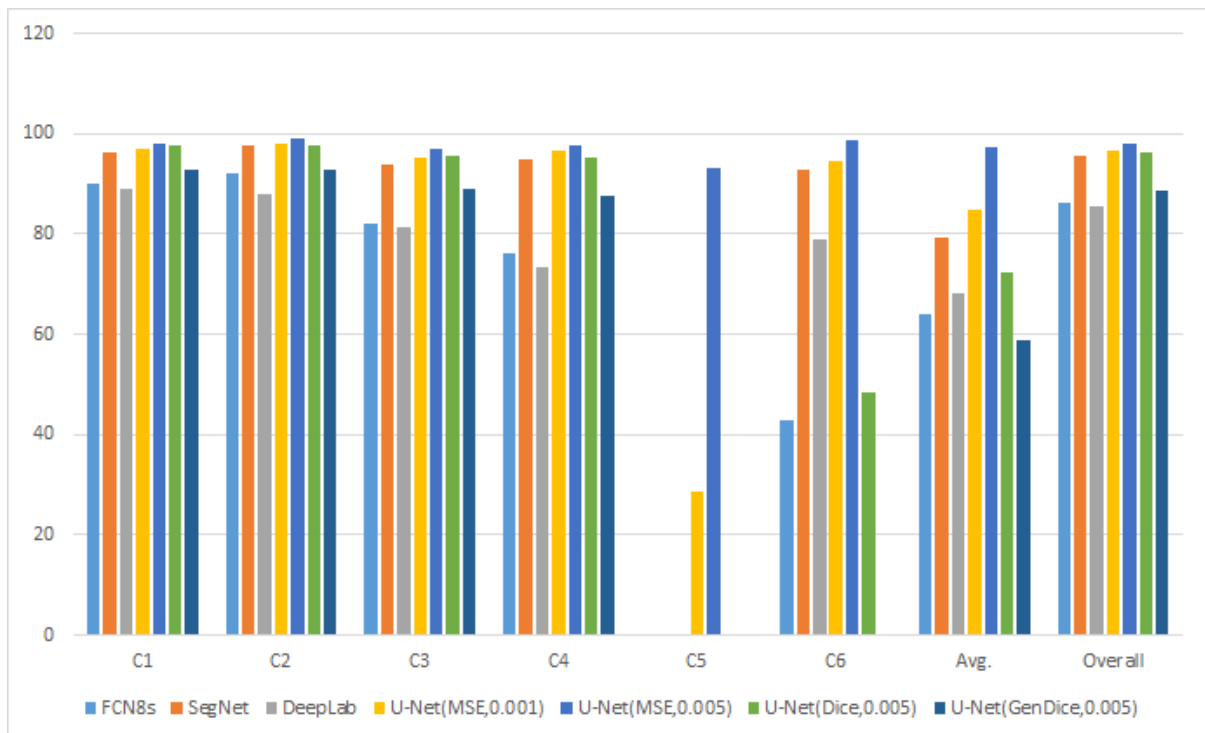
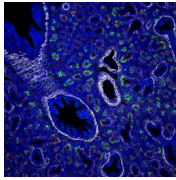
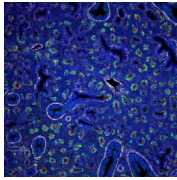
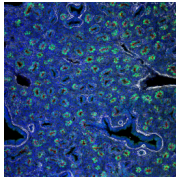
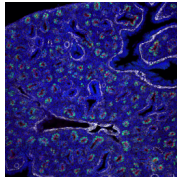
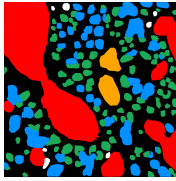
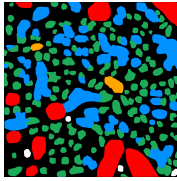
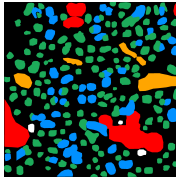
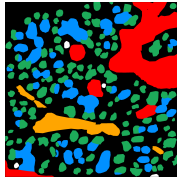
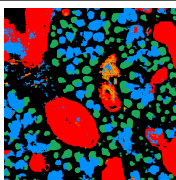
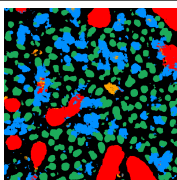
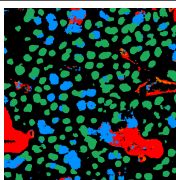
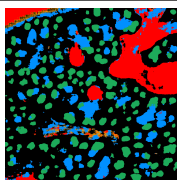
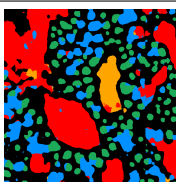
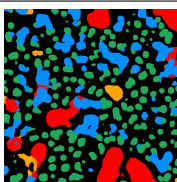
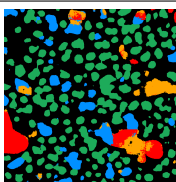
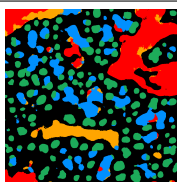
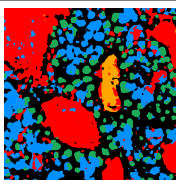
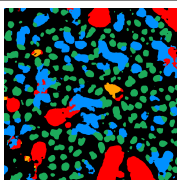
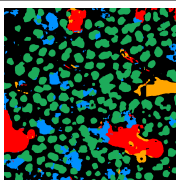
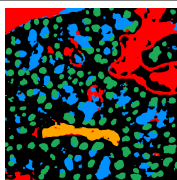
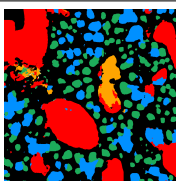
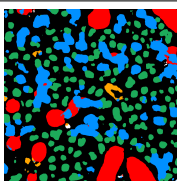
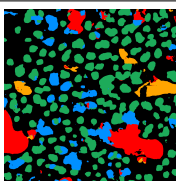
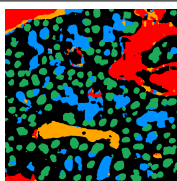
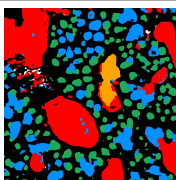
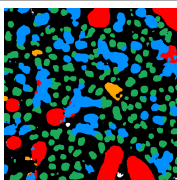
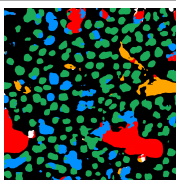
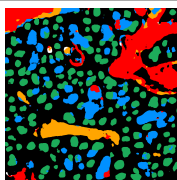
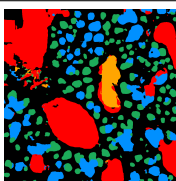
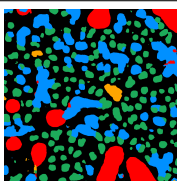
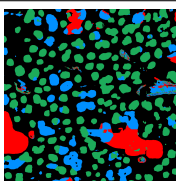
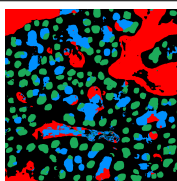
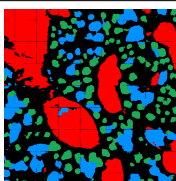
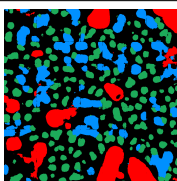
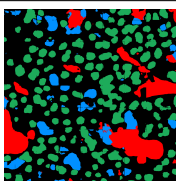
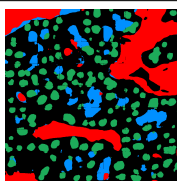


Figure 5.2: Bar Chart of Train Results

Table 5.3: Results

Model	Accuracy	Example 1	Example 2	Example 3	Example 4
Loss Func. learning rate					
Input Image					
Ground Truth					
FCN8s MSE 0.001	80.6				
SegNet MSE 0.001	80.0				
DeepLab MSE 0.001	79.0				
U-Net MSE 0.001	82.1				
U-Net MSE 0.005	83.0				
U-Net Dice 0.005	83.0				
U-Net Gen. Dice 0.005	79.8				

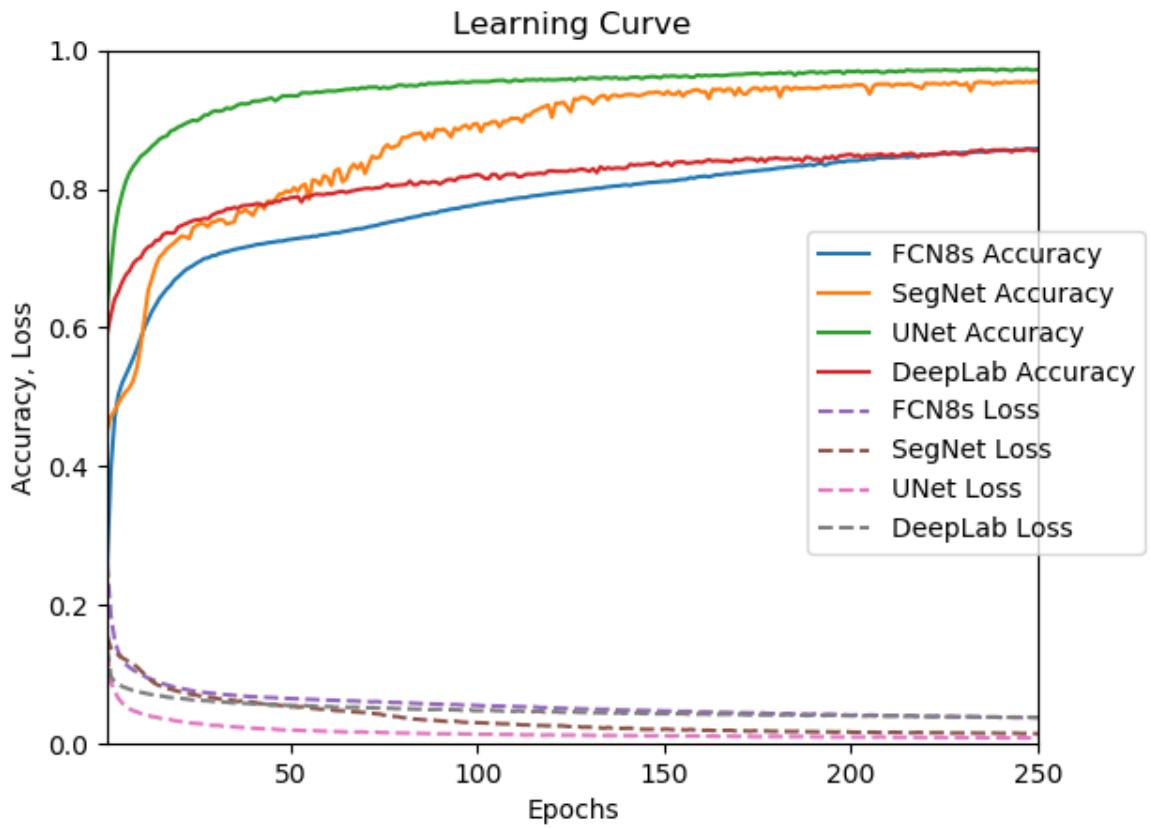
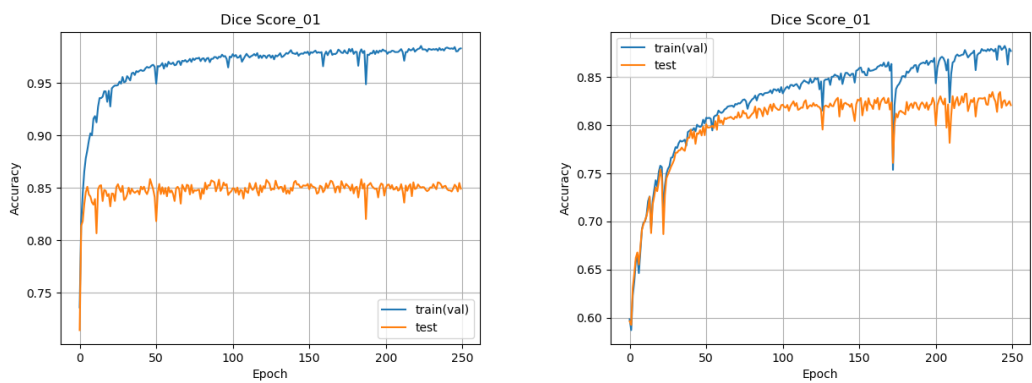


Figure 5.3: Learning Curve



(a) Mean Squared Error

(b) Mean Squared Error vs. Generalized Dice

Figure 5.4: Caption

5.2 Synthetic Image Generation

5.2.1 Experimental Results

Figure 5.5 shows samples of the train, test, and synthetic images from lung immunofluorescent confocal image sub regions. Each mask images were created by the trained U-Net, and synthetic images were created from noise through the generator of DCGAN model.

5.2.2 Discussion

As can be seen in Figure 5.5, some of the obtained fake images are similar to the original objects, but in some cases, the distorted objects are created by the generator. In such distorted cases, the trained segmentation network does not work well and obtained spurious objects.

Currently, this study focused on synthetically obtaining various distinct objects (corresponding to lung tissue classes), however, creating entire lung immunofluorescent confocal images should be discussed for improving the training images for obtaining improved multi-class segmentation with deep learning models. This thesis will evaluate the effect for segmentation accuracy when using fake images as the training dataset as well as with fused real images.

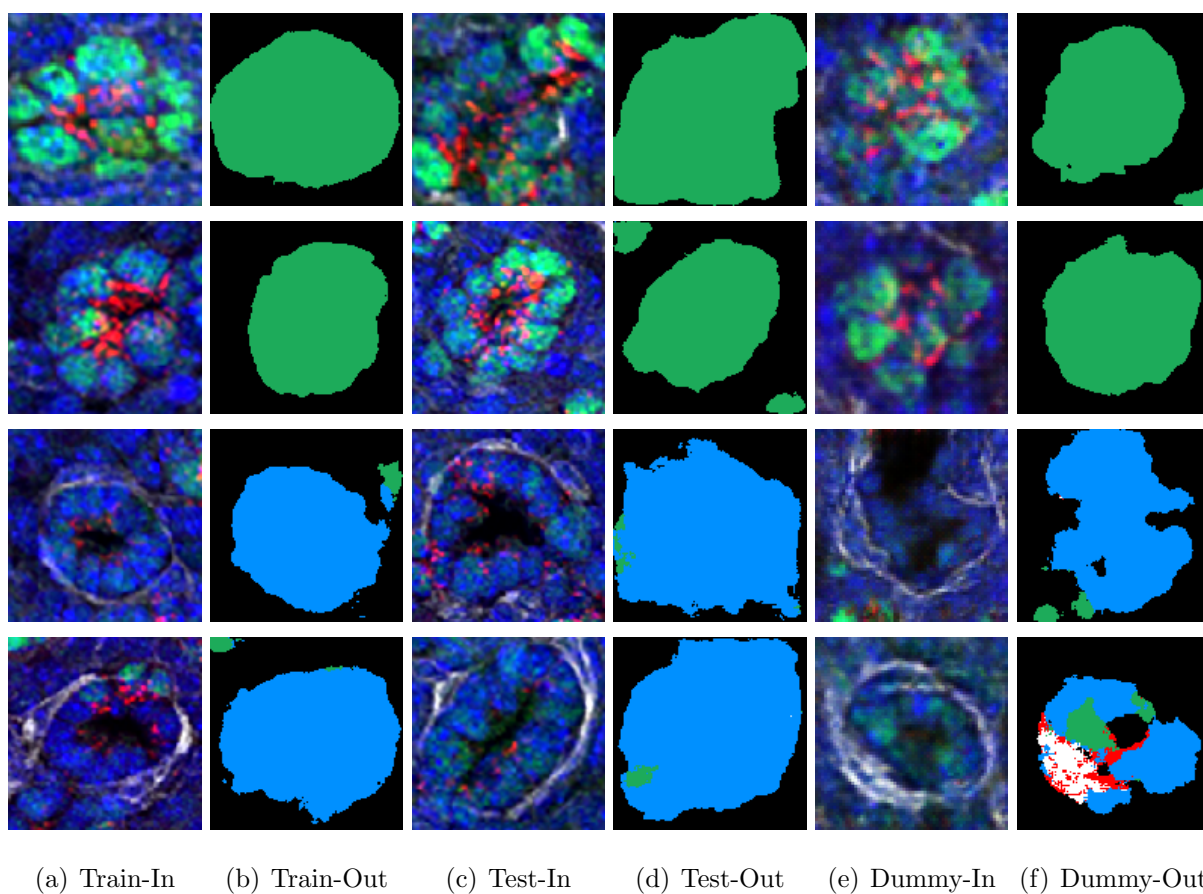


Figure 5.5: Synthetic Image Generation

Chapter 6

Concluding Remarks

6.1 Conclusion

The purpose of this paper is to propose the pipeline for lung immunofluorescent confocal image segmentation. These lung images were obtained from the online database operated by LungMAP, the research project on lung development analysis. Immunofluorescent confocal image is one kind of the provided dataset concluding RNA-sequence, Histological stain, CT images, and so on. Immunofluorescent confocal image is a full color image and it stains multiple specific proteins so that it is necessary to annotate them before analysis. However, lung tissue image has a complex structure and there is a big amount of images contrast to a few experts. Hence, this research aimed to develop an automatic annotation system with deep learning. For applying deep learning methods, the dataset is an important element to create a better network. Thus, a web-based annotation system was developed as the first step of the pipeline. Consequently, this thesis created the training dataset for deep learning-based segmentation network with the collected annotations by the developed system. In the experiment, 3 famous segmentation model, U-Net [6], SegNet [21], and DeepLabv3+ [22] were compared. As a result, U-Net marked the highest score, and it was 83.0% as the test result of overall segmentation accuracy.

6.2 Further Works

Firstly, the lung structure segmentation accuracy should be improved for practical use. To achieve this objective, we will consider the enhancement of the training dataset by two approaches as follows,

1. manual image annotation with developed web-based annotation tool,
2. and a pseudo dataset augmentation by synthetic image generation method.

The improvement of the segmentation networks will be discussed by adjusting network construction, loss function, learning rate, optimizer, and so on.

Secondly, the final goal of our research is to construct the pipeline for automated lung image segmentation and lung development analysis based on image information combined with the gene expression data. This paper mainly discussed the automated lung image segmentation on level 1 segmentation, biomedical structure level segmentation of lung tissue image. Therefore, the next work is to develop the following systems.

1. Individual segmentation part for level 2 segmentation identifying cell structure in level 1 segmented objects,
2. image feature extraction part for calculating the image features,
3. and analysis part for combining the obtained image information with the gene expression data.

Acknowledgement

First of all, I really would like to express my deepest gratitude to Prof. Shinji Tsuruoka, who is the Executive Vice-President of Mie University and Associate Prof. Hiroharu Kawanaka at Graduate School of Engineering, Mie University who offered continuing supports and constant encouragements.

I am also grateful to Prof. Bruce J. Aronow and Prof. V. B. Surya Prasath at Cincinnati Children's Hospital Medical Center, USA. They provided a lot of technical help and encouragement. Prof. Aronow accepted me as a short-term study abroad student at his laboratory in Cincinnati Children's Hospital Medical Center. Thanks to his help and encouragement, I was able to make good progress in my project. Prof. Surya gave me a lot of technical and language help in my research project and writing papers.

In addition, this study was partially supported by Mie University study abroad program. Thanks to this program, I could stay Cincinnati Children's Hospital Medical Center for a month to obtain great progress and discuss the project with experts. I would like to sincerely thank this scholarship program. And finally, I would like to appreciate other members related my research project.

Also, the results shown here are in whole based upon the data generated by the LungMAP Consortium [U01HL122642] and downloaded from <https://lungmap.net/>, on October 1, 2019. The LungMAP consortium and the LungMAP Data Coordinating Center (1U01HL122638) are funded by the National Heart, Lung, and Blood Institute (NHLBI).

Reference

- [1] M. E. Ardini-Poleske, R. F. Clark, C. Ansong, J. P. Carson, R. A. Corley, G. H. Deutsch, J. S. Hagood, N. Kaminski, T. J. Mariani, S. S. Potter, G. S. Pryhuber, D. Warburton, J. A. Whitsett, S. M. Palmer, and N. Ambalavanan, “Lungmap: The molecular atlas of lung development program,” *American Journal of Physiology-Lung Cellular and Molecular Physiology*, vol. 313, no. 5, pp. L733–L740, 2017.
- [2] M. E. Ardini-Poleske, T. J. Mariani, G. S. Pryhuber, R. S. Misra, L. Consortium *et al.*, “Initiating multiomics approach to understand neonatal chronic lung disease: the lungmap experience,” in *Updates on Neonatal Chronic Lung Disease*. Elsevier, 2020, pp. 45–59.
- [3] M. Guo, H. Wang, S. S. Potter, J. A. Whitsett, and Y. Xu, “Sincera: A pipeline for single-cell rna-seq profiling analysis,” *PLOS Computational Biology*, vol. 11, no. 11, pp. 1–28, 11 2015.
- [4] RTI International, “Lungmap official web site,” retrieved October 1, 2019, from <https://lungmap.net/>.
- [5] H. Pan, G. H. Deutsch, S. E. Wert, N. M. A. of Lung Development Program Consortium *et al.*, “Comprehensive anatomic ontologies for lung development: A comparison of alveolar formation and maturation within mouse and human lung,” *Journal of biomedical semantics*, vol. 10, no. 1, p. 18, 2019.
- [6] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.
- [7] A. Yonekura, H. Kawanaka, V. B. S. Prasath, B. J. Aronow, and H. Takase, “Automatic disease stage classification of glioblastoma multiforme histopathological images using deep convolutional neural network,” *Biomedical Engineering Letters*, vol. 8, no. 3, pp. 321–327, 2018.
- [8] A. Yonekura, H. Kawanaka, V. B. S. Prasath, B. J. Aronow, and H. Takase, “Improving the generalization of disease stage classification with deep CNN for glioma histopathological images,” in *IEEE International Conference on Bioinformatics and*

Biomedicine (BIBM), Kansas, MO, USA, November 2017, pp. 1222–1226, international Workshop on Deep Learning in Bioinformatics, Biomedicine, and Healthcare Informatics (DLB2H).

- [9] V. B. S. Prasath, “Deep learning based computer-aided diagnosis for neuroimaging data: focused review and future potential,” *Neuroimmunol Neuroinflammation*, vol. 5, p. 1, 2018.
- [10] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” *arXiv preprint arXiv:1511.06434*, 2015.
- [11] Y. Xu, T. Mizuno, A. Sridharan, Y. Du, M. Guo, J. Tang, K. A. Wikenheiser-Brokamp, A.-K. T. Perl, V. A. Funari, J. J. Gokey, B. R. Stripp, and J. A. Whitsett, “Single-cell rna sequencing identifies diverse roles of epithelial cells in idiopathic pulmonary fibrosis,” *JCI Insight*, vol. 1, no. 20, 3 2017.
- [12] G. Clair, P. D. Piehowski, T. Nicola, J. A. Kitzmiller, E. L. Huang, E. M. Zink, R. L. Sontag, D. J. Orton, R. J. Moore, J. P. Carson, R. D. Smith, J. A. Whitsett, R. A. Corley, N. Ambalavanan, and C. Ansong, “Spatially-Resolved Proteomics: Rapid Quantitative Analysis of Laser Capture Microdissected Alveolar Tissue Samples,” *Scientific Reports*, vol. 6, no. November, pp. 1–13, 2016.
- [13] L. Barbe, E. Lundberg, P. Oksvold, A. Stenius, E. Lewin, E. Björling, A. Asplund, F. Pontén, H. Brismar, M. Uhlén, and H. Andersson-Svahn, “Toward a confocal subcellular atlas of the human proteome,” vol. 7, no. 3, pp. 499–508, 2008.
- [14] “the human protein atlas official web site,” retrieved October 1, 2019, from <https://www.proteinatlas.org/>.
- [15] U. Sampathkumar, S. Prasath, S. Meena, and K. Palaniappan, “Assisted ground truth generation using interactive segmentation on a visualization and annotation tool,” 10 2016, pp. 1–7.
- [16] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov 1998.
- [17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. [Online]. Available: <http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf>

- [18] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 3431–3440.
- [19] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *CoRR*, vol. abs/1409.1556, 2014. [Online]. Available: <http://arxiv.org/abs/1409.1556>
- [20] C. Szegedy, Wei Liu, Yangqing Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 1–9.
- [21] V. Badrinarayanan, A. Kendall, and R. Cipolla, “Segnet: A deep convolutional encoder-decoder architecture for image segmentation,” *arXiv preprint arXiv:1511.00561*, 2015.
- [22] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, “Encoder-decoder with atrous separable convolution for semantic image segmentation,” in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 801–818.
- [23] V. Dumoulin and F. Visin, “A guide to convolution arithmetic for deep learning,” *arXiv preprint arXiv:1603.07285*, 2016.
- [24] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.
- [25] A. A. Novikov, D. Lenis, D. Major, J. Hladuvka, M. Wimmer, and K. Buhler, “Fully convolutional architectures for multi-class segmentation in chest radiographs,” *IEEE Transactions on Medical Imaging*, 2018.
- [26] S. Isaka, H. Kawanaka, V. B. S. Prasath, B. J. Aronow, and S. Tsuruoka, “Multi-class segmentation of lung immunofluorescence confocal images using deep learning,” in *3rd International Workshop on Deep Learning in Bioinformatics, Biomedicine, and Healthcare Informatics (DLB2H 2019) in conjunction with IEEE International Conference on Bioinformatics and Biomedicine (BIBM 2019)*, 2018, pp. 2362–2368.
- [27] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: <http://arxiv.org/abs/1512.03385>
- [28] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

Publication List

International Conferences

(1) Shu Isaka, Hiroharu Kawanaka, V. B. Surya Prasath, Bruce J. Aronow, and Shinji Tsuruoka, "Development of a Web-Based Image Annotation Tool for Lung Immunofluorescent Confocal Images," in the 4th International Symposium on Affective Science and Engineering, and the 29th Modern Artificial Intelligence and Cognitive Science Conference (ISASE-MAICS 2018), B2-1, pp. 1-5, 2018.

(2) Shu Isaka, Hiroharu Kawanaka, V. B. Surya Prasath, Bruce J. Aronow, and Shinji Tsuruoka, "Region Classification for Lung Immunofluorescent Confocal Images," in Joint 2018 7th International Conference on Informatics, Electronics & Vision (ICIEV) & 2nd International Conference on Imaging, Vision & Pattern Recognition (icIVPR), pp. 41-45, 2018.

(3) Shu Isaka, Hiroharu Kawanaka, V. B. Surya Prasath, Bruce J. Aronow, and Shinji Tsuruoka, "Development and Application of Web-based Lung Image Annotation Tool for Automatic Annotation," in the 8th International Symposium for Sustainability by Engineering at Mie University (Research Area C), pp. 55-56, 2018.

(4) Shu Isaka, Hiroharu Kawanaka, V. B. Surya Prasath, Bruce J. Aronow, and Shinji Tsuruoka, "A Study on Classification Method for Automatic Lung Image Annotation," in International Workshop of Regional Innovation Studies (IWRIS 2018), pp. 9-12, 2018.

(5) Shu Isaka, Hiroharu Kawanaka, V. B. Surya Prasath, Bruce J. Aronow, and Shinji Tsuruoka, "Segmentation Pipeline for Lung Immunofluorescent Confocal Image Analysis," in the 9th International Symposium for Sustainability by Engineering at Mie University (Research Area C), pp. 3-4, 2019.

(6) Shu Isaka, Hiroharu Kawanaka, V. B. Surya Prasath, Bruce J. Aronow, "Multi-Class Segmentation of Lung Immunofluorescence Confocal Images Using Deep Learning," in 3rd International Workshop on Deep Learning in Bioinformatics, Biomedicine, and Healthcare Informatics (DLB2H 2019) in conjunction with IEEE International Conference on

Bioinformatics and Biomedicine (BIBM 2019), pp. 2362-2368, 2019.

Domestic Conferences

(1) 伊坂脩, 川中普晴, V. B. Surya Prasath, Bruce J. Aronow, 鶴岡信治, “敵対的生成ネットワークを用いた 疑似肺組織画像の生成に関する一試み (Pseudo Lung Tissue Image Generation Using Generative Adversarial Networks),” 令和元年度電気・電子・情報関係学会東海支部連合大会講演論文集, A1-7, 2019.

(2) 伊坂脩, 川中普晴, V. B. Surya Prasath, Bruce J. Aronow, 鶴岡信治, ”ディープラーニングを利用した肺組織免疫蛍光染色画像の多クラスセグメンテーション”, 第39回医療情報学連合大会, p. 310, 2019.

Appendix A

Detailed Immunofluorescence Method

Purpose

Purpose Immunofluorescence on slides of 7 μ m frozen sections of C57BL6 E16.5 lungs for SOX9, SFTPC, and ACTA2.

Abstract

Day 1

1. For Frozen tissue, rinse 2X in PBS, then 5 min in 4% PFA/PBS, then rinse 1X in PBS.
2. Briefly equilibrate slides in Antigen Retrieval Buffer.
3. Antigen retrieval, pH 6.0 (times will vary according to microwave).
4. 10 mM sodium citrate, pH 6.0, and heat in a microwave at 96°C.
5. Microwave according to instructions on microwave.
6. Cool on countertop, 15 min.
7. Rinse with dH₂O.
8. 1X PBS, 5 min.
9. Block in 4% Donkey serum/PBS - T, 2 hours at RT.
10. For Rabbit anti-SOX9 (AB-5535, Lot# 2167153, Millipore) dilute 1:100, For goat anti-SFTPC (SC-7706, Santa Cruz) dilute 1:100, for mouse anti-ACTA2 (α Smooth muscle actin, A5228, Sigma) dilute 1:2000 in blocking buffer. Spin down in μ fuge for 10 minutes and incubate on tissue overnight @ 4°C.

Day 2

1. Rinse slides in PBS - T 3X, 5 min.
2. Apply secondary antibody, Donkey Alexa Fluor 488 anti-rabbit IgG (A21206, Lot# 1608521, for anti-SOX9), Donkey Alexa Fluor 568 anti-goat IgG (A11057, Lot# 1485187, for anti-SFTPC) at 1:200, Donkey Alexa Fluor 647 anti-mouse IgG A31571, Lot# 1549801, ACTA2) at 1:200, in blocking buffer.

Spin down in μ fuge for 10 min, apply to tissue and incubated at room temperature for 1 hour.

3. Rinse in PBS-T 3X, 5 min.
4. Dilute DAPI 1:2000 and apply to slides for 10 min.
5. Wash in PBS-T 3X, 5 min.
6. Rinse slides in 0.1M PB, 3X, 5 min.
7. Add 1 drop of Prolong Gold anti-fade mounting medium (P36930).
8. Coverslip with Gold Seal Coverslip (Cat# 3422 Electron Microscopy Sciences, 22 X 22 mm).
9. Allow Prolong Gold to cure overnight at room temp in light sealed box.

Tissue Used

LMM.14.24.4.46, E16.5, C57BL6, Gender: Male, Crown-to-rump: 16.0mm, and Weight: 0.592g

Appendix B

Segmentation Results

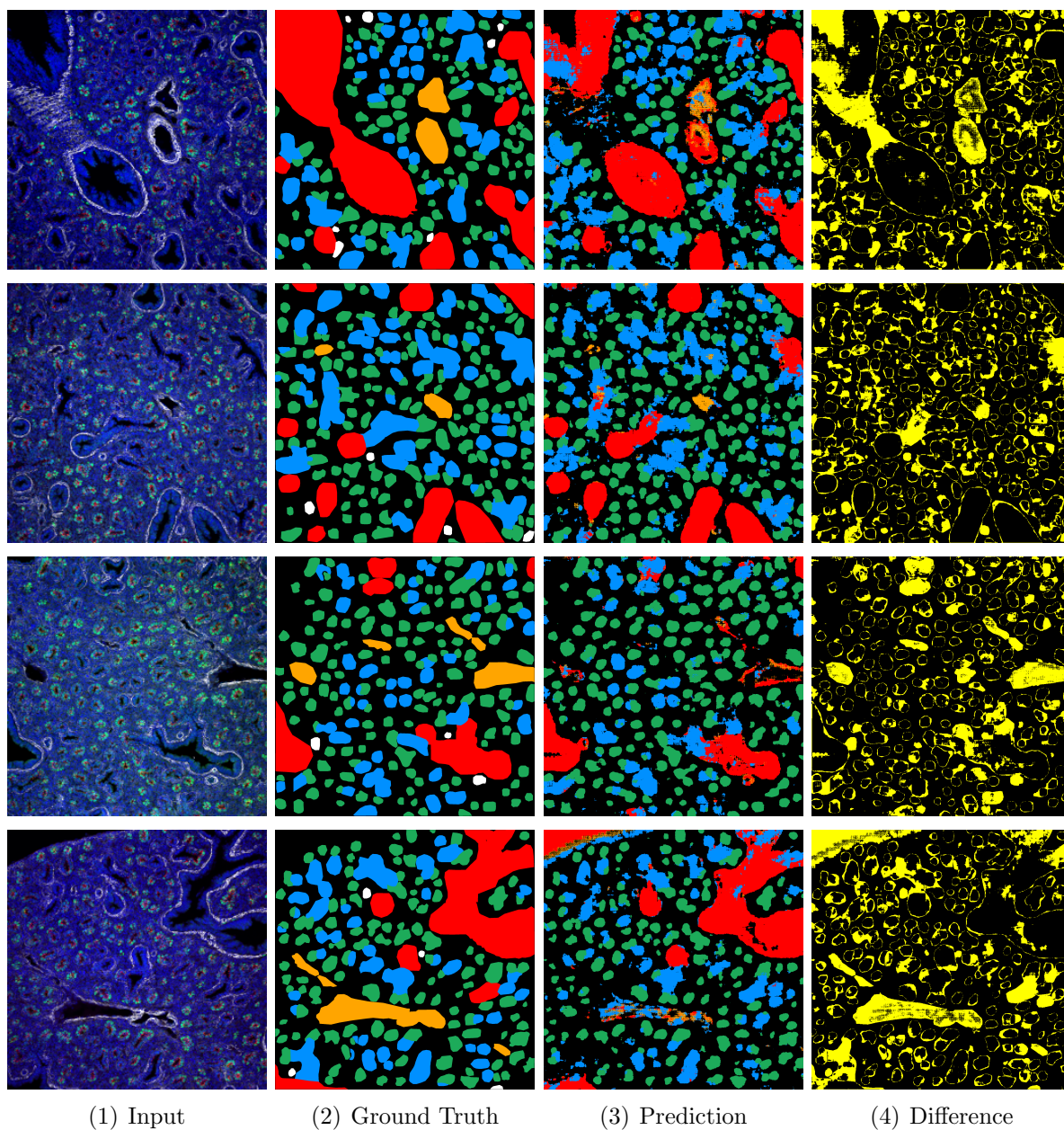


Figure B.1: FCN8s

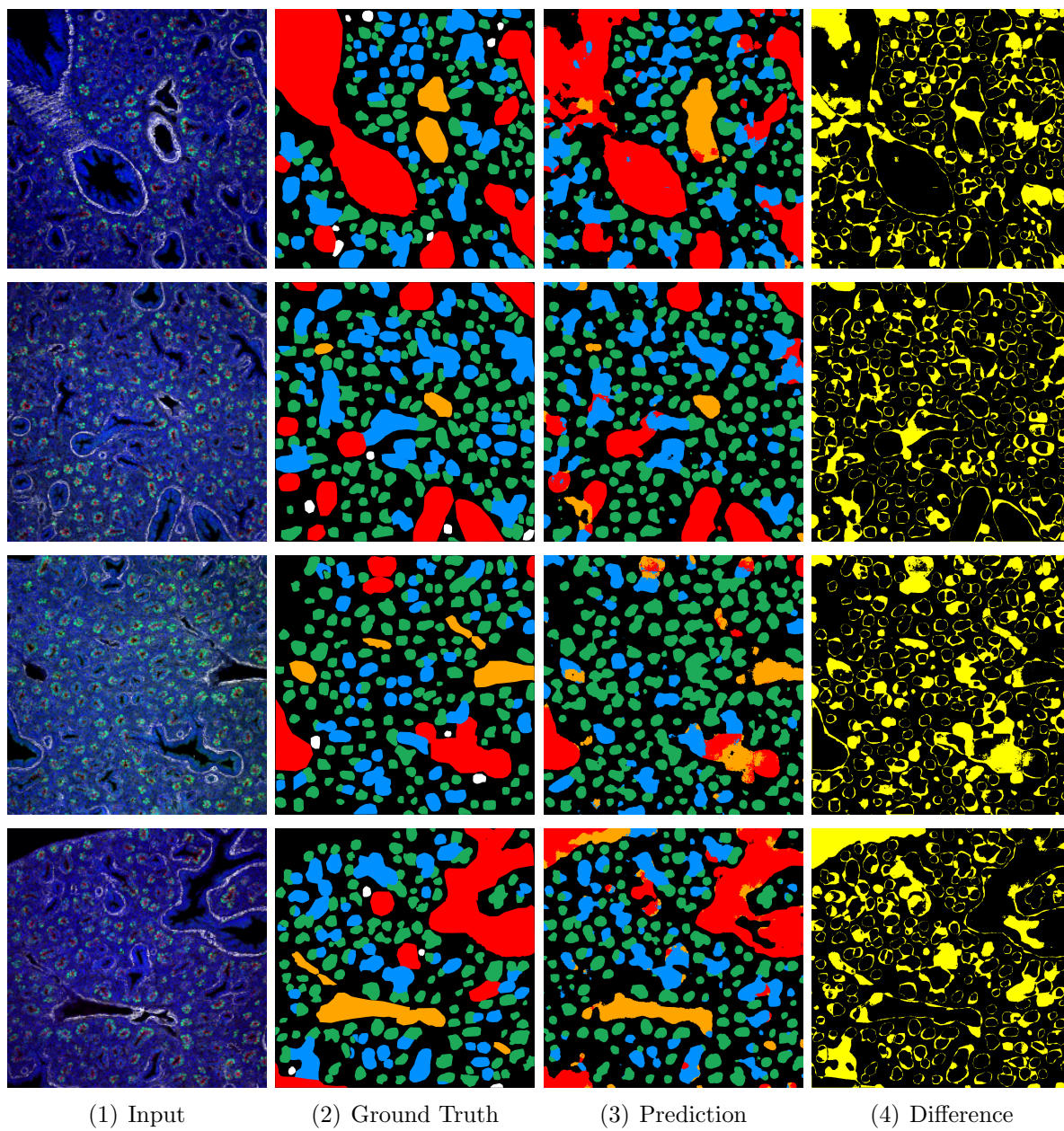


Figure B.2: SegNet

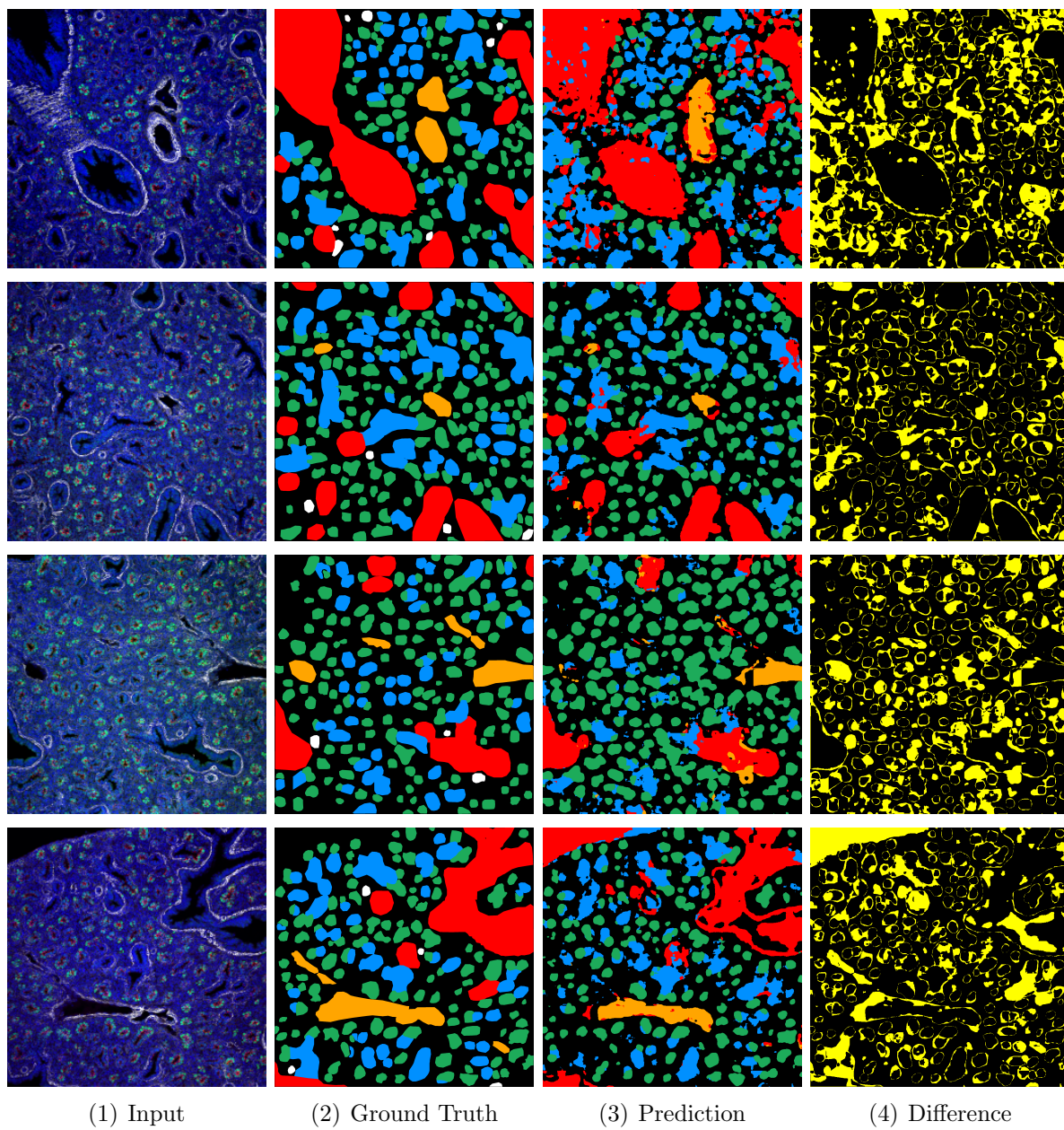


Figure B.3: Deep Lab v3+

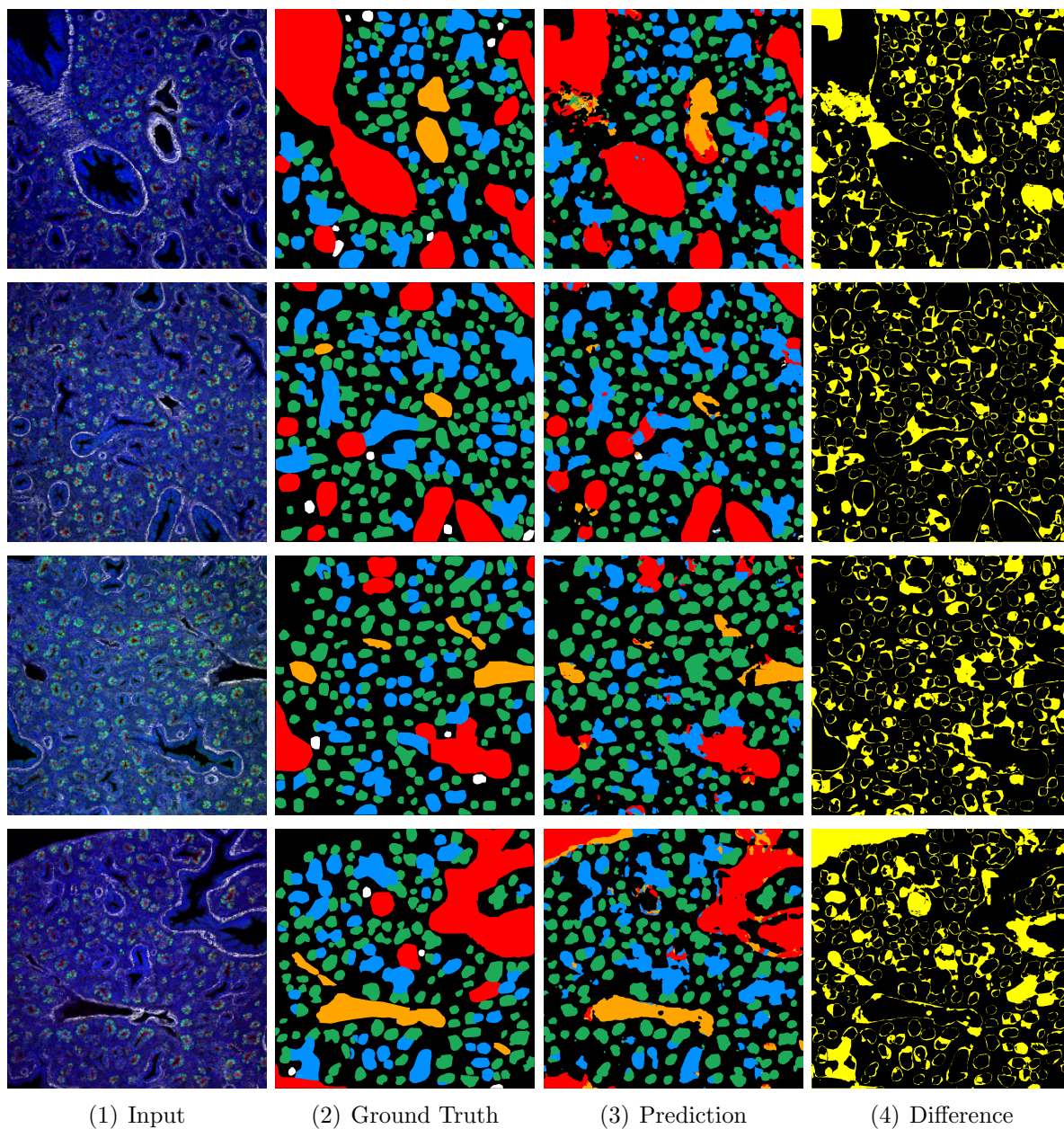


Figure B.4: U-Net (lr=0.001, loss func.=MSE)

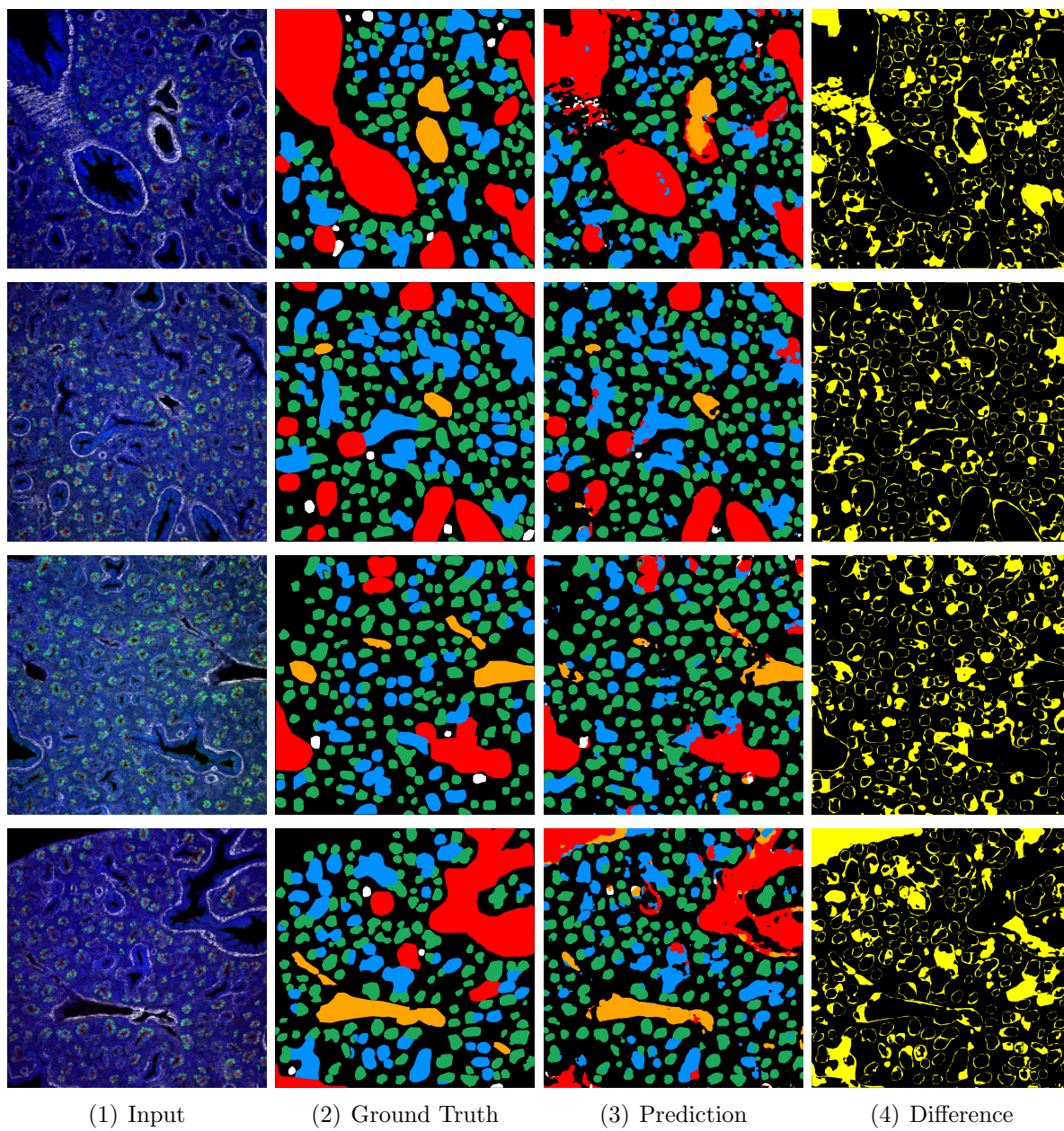


Figure B.5: U-Net (lr=0.005, loss func.=MSE)

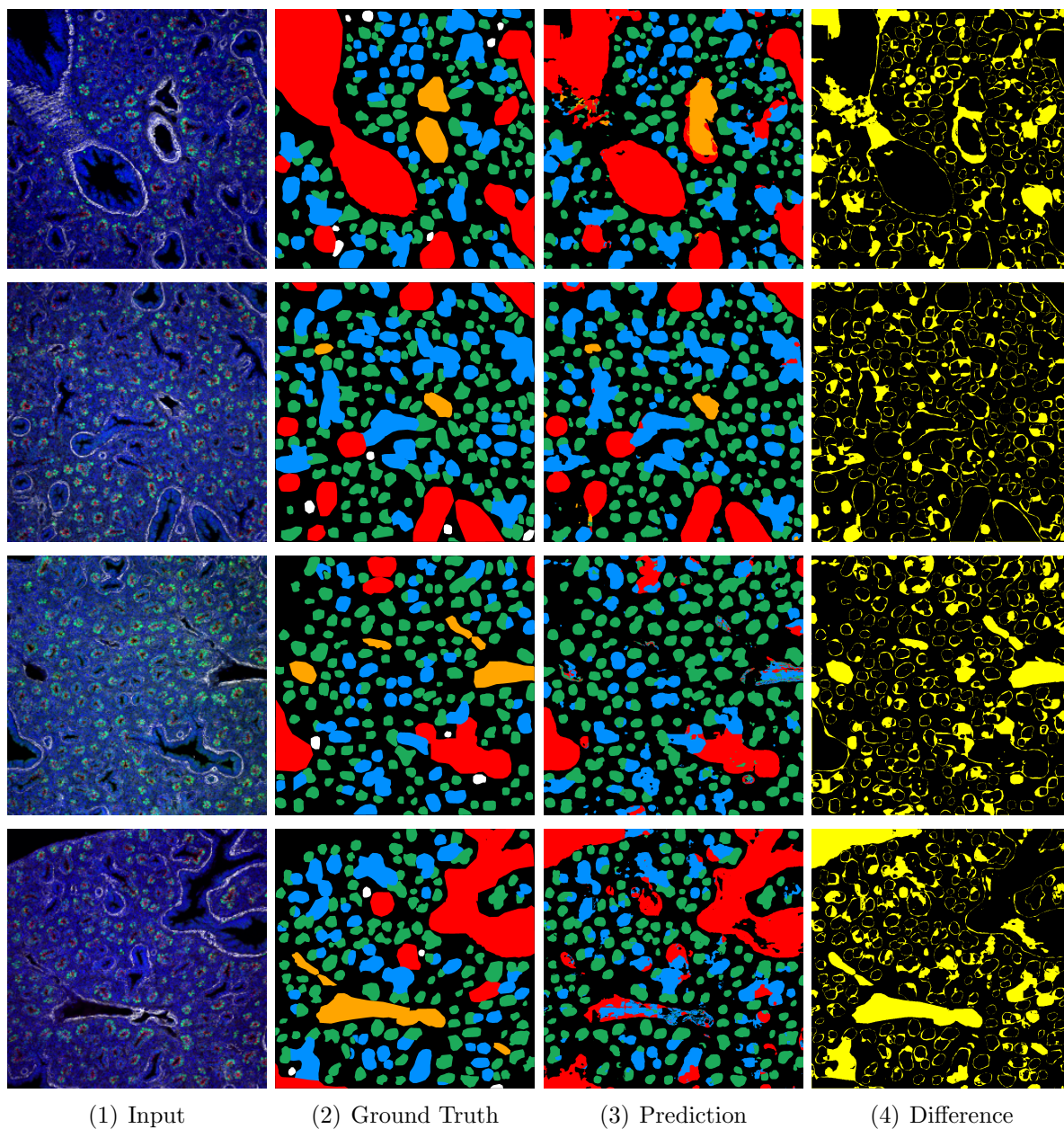


Figure B.6: U-Net (lr=0.005, loss func.=Dice Loss)

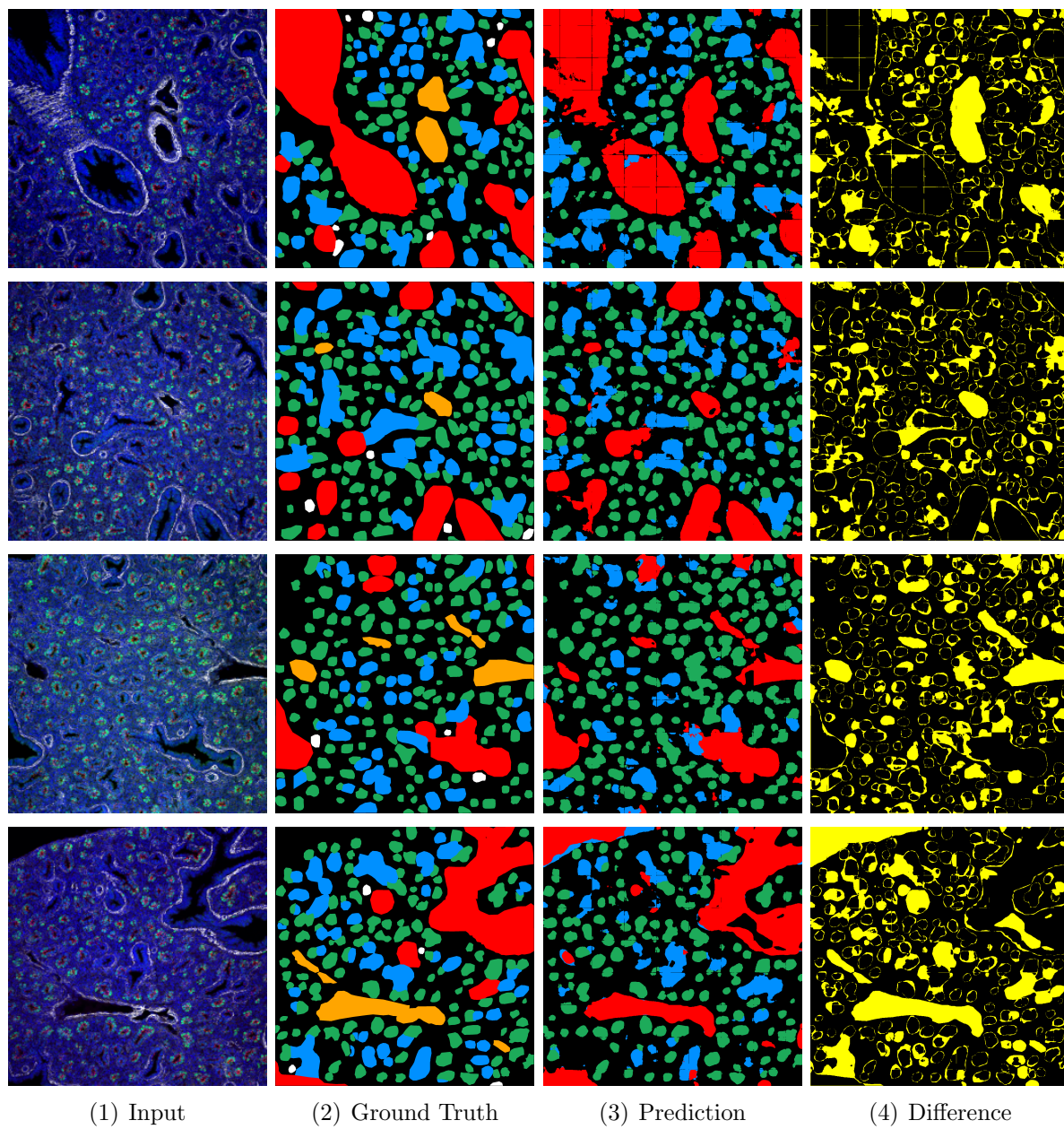


Figure B.7: U-Net (lr=0.005, loss func.=Gen. Dice Loss)