END-TO-END AUTOMATION ENABLES HIGH THROUGHPUT SPLEEN VOLUME ASSESSMENT FOR COMPUTED TOMOGRAPHY CLINICAL TRIALS

By

Hyeonsoo Moon

Thesis

Submitted to the Faculty of the

Graduate School of Vanderbilt University

for the degree of

MASTER OF SCIENCE

in

Electrical Engineering

May 11, 2018

Nashville, Tennessee

Approved by:

Bennett A Landman, Ph.D.

Richard Alan Peters Ph.D.

ACKNOWLEDGMENT

I have had wonderful support and encouragement from Dr. Landman and Dr. Peters during this work. I owe the success of this work to these two advisor.

Dr. Landman provided me the essential idea and motivation of this work. I had not experienced dealing with medical data, but Dr.Landman noticed me about the issues and ideas related to this field, so that I could settle down with the research. He always tried to advice me and give the answers and feedback to me as soon as possible. Thus I could spend my research hours so efficiently. Research environment was also great for me to focus on development and by using my own machine in the laboratory, I could feel sense of responsibilities and passion as well. Dr. Peters had been the kindest advisor for last two years for me. He always encouraged me and said good words that I am doing well. He listened my thought carefully, and sympathized with me. This made me not to give up my works so far. I am glad to meet Dr.Huo, who has been a perfect mentor for me not only about the research, but also about the living in lab, and being accustomed to the academic issues. I appreciate you Yuankai.

I am grateful MASI Lab, especially wonderful people in the lab, who are always make me smile and feel good, even solving my problems. Thank you guys.

I appreciate the time that I shared with my friends in Nashville. KSSA group people, and it was one of the best moment for me to meet you guys Hanui and Euihyun. Thank you for co-working and encouraging me as a good friend Shengchao.

Finally, I have my parents to thank for letting me in such a wonderful place to study and learn. Always love you my parents.

TABLE OF CONTENTS

Page

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

1.1 Motivation

Computed Tomography (CT) is being widely used for detecting anatomical issues inside of the human body, such as with the brain, spleen, pancreas, colon, and kidney. Compared to X-ray image, CT provides 3-D structural information that enable assessment of tissues within organs such as blood, the cerebrospinal fluid, tumor, and gray and white matter. Despite these many capabilities, some disadvantages of CT imaging still hamper research. **[3-4]** The issues are not only the problems of the CT image itself, but also the inconvenience and workflow that come from the process of knowledge generation from the imaging data. These problems are further highlighted in the context of a clinical trial. Herein, we focus on the example of the segmentation of organs. Specifically, we focus on segmented spleen tissue from abdominal CT image as it plays a role in detecting liver diseases and other infections such as brucellosis, hepatitis, and anemia by measuring volume and biomarker. **[1-2]** For these reasons, spleen segmentation is a central focus of study for better segmentation accuracy. These days, there exist various problems that arise in the whole segmentation processes which are not just about the accuracy of the segmentation itself. First, there is inconsistency in the transmission of the DICOM image captured by the CT scanner. CT data transfer is largely a transmission of recorded CDs, uploading to a specified shared directory, and direct uploading to a target server repository. In most cases, this is done by first or second methods. In these situations, the lack of information in the image header, the loss of a specific header, and other numerous issues hinder uploading images to the server properly.

In terms of workflow optimization from a traditional research perspective to a clinical

trial, preprocessing and post-processing take too much time compared to segmentation itself. With modern methods, most of segmentation is done automatically using deep learning, so the process is very fast. However, segmentation using deep learning requires preprocessing and post-processing of data. First of all, 3D CT images should be organized into 2D slices so that they can be input of segmentation pipeline because most deep neural network requires 2D input dataset to be trained and validated. **[5-6]** In preprocessing, slices also need to be resized and intensity-normalized based on image processing algorithms because input test data must have same dimension and same intensity range for being segmented using training data. Like preprocessing, segmented slices also have to be re-converted into 3D images whose dimensions are same as original input. Significantly, since subsidiary results derived from each sub-process need to be translated and dealt with different software and operating system, it makes researchers tired and spent redundant time. Those processes take relatively long time, but essential procedures for being parts of automatic segmentation.

1.2 Solution Summary

We propose containerizing tools for solving this problem. These days, tools for containerizing software packages and programs together have begun to be widely used, and one of the most famous one is Docker. Docker pursues 'Build, Ship, and Run any App, Anywhere'. We select Docker to synthesize whole pipeline process into one script file that does not require any other function calls or script processing except a single Docker script. The rest of paper will show detailed parts as follows: Data decompression, Data push and uploading, Preprocessing, Segmentation, and Post-processing. At first, methods being used in the pipeline, especially for each subsidiary part by part, will be introduced with algorithmic descriptions. An overall flowchart of pipeline will be explained. Then each

substitute part would be described with methods used within process, and intermediate results. Segmented spleen volume will be shown using pdf document so that not only clinicians but also patients easily can see it and utilize it for demonstration purposes. Since the whole processes are included and merged into one Docker container, containerization and cross-language system will be described.

1.2.1 System Pipeline

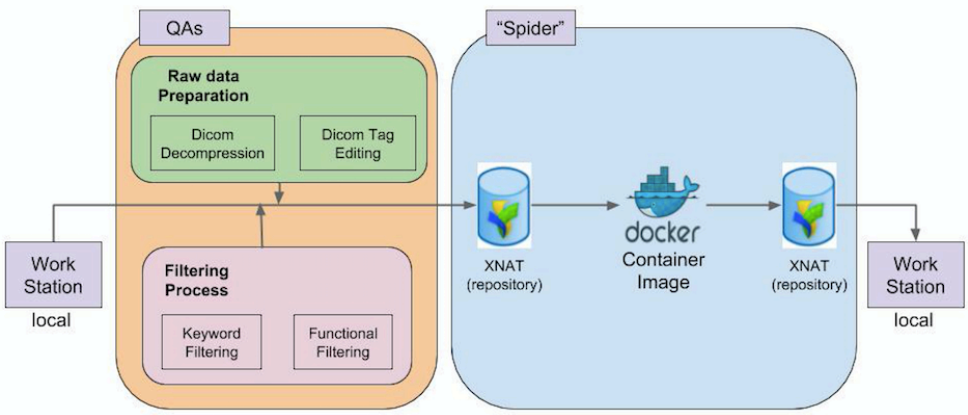


Figure 1.1: Overall system pipeline. It contains general processing steps from local data preprocessing to retrieving the labeled spleen volume back to the local machine.

End-to-end pipeline starts from imaging data from a scanner. **Fig. 1.1** shows whole a broad overview of the process. Quality Assurance (QA) process (left square part) is composed with raw data preparation and filtering process. Clinical systems often use data compression, but research systems do not support it well. Thus, data decompression for removing compression from the files should be conducted beforehand. Herein, the **dcmtk** library function **dcmdjpeg** is used for this procedure. Decompressed data is header-edited with 'patient comments' tag, and filtered out with keyword and functional QAs. In **Fig. 1.1.**, right blue box shows main container that includes mainstream process of system described at **Fig. 1.2.**

1.2.2 Container Pipeline

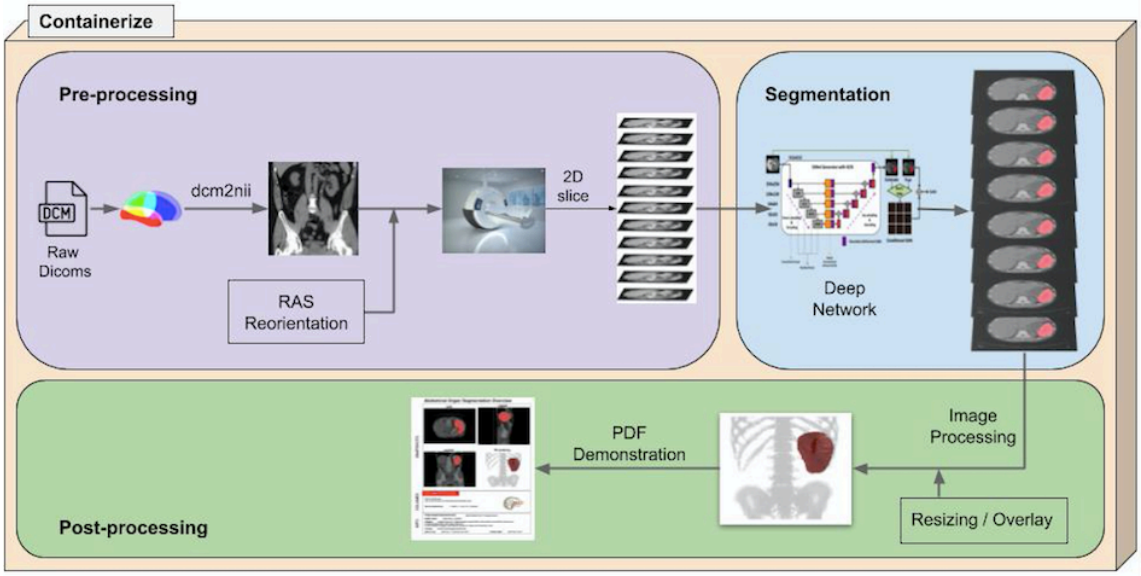


Figure 1.2: Container pipeline. Among the whole system pipeline, it shows mainstream about the data processing except the data translation through server.

Fig. 1.2. shows three main procedures. At preprocessing (purple box), **DICOMs** are converted into **NIfTI,** spatially orientation corrected, and sliced into set of 2D slices. Sliced images go into segmentation process which uses deep neural network SSNet (blue box). After having segmented slices, merging of slices is conducted to get 3-D volumetric labels. To be demonstrated properly and easy to be understand, the pipeline generates pdf file that shows auto-volumes, 3 views (axial, coronal, sagittal), and 3-D rendered image. These main part of the system is composed as **Docker** container.

Chapter 2

METHOD

This section introduces overall methods according to the processing sequences. Starting from data quality assurance and tag editing, the section also describes preprocessing, segmentation methods, and post-processing methods in order.

2.1 Data Compression

In this paper, for data loading process and for uploading through network, we use **DCMTK** toolkit from OFFIS. The **DICOM** research library can detect, but not read, compressed **DICOMs**. However, at most cases, **DICOM** files are being compressed as JPEG format to be less size. Thus, decompression is necessary before being done other process later on. 'dcmdjpeg' function detects compressed **DICOMs** and convert them into DICOM (".dcm") format files.

2.2 CT Header Commenting

The DICOM header information content depending on the scanners. Therefore, to upload CT data or send it to a certain repository, common tag should be fixed or edited by users. In this paper, target repository is set as **XNAT**, which is operated by VUIIS Center for Computational Imaging (VUIIS CCI) from Vanderbilt University. A Requirement for sending CT **DICOMs** to **XNAT** is tag (0010,4000), which is 'Patient Comments'. Among the widely used scanners, we test with two which are Siemens, Philips, and GE. Even if the data is driven from same scanner, there exists a case that some of them do not have tag (0010,4000) in it, and others have. So, we made function for editing (0010,4000) tag for

the **DICOMs** which have been passed 'Quality Assurance' before the process.

Table 2.1 shows part of **DICOM** header information that should be edited for being uploaded to the server. 'Patient Comment' is edited or added as it can be shown as Table 2.1. Red boxed tags are utilized for filtering data out, and blue boxed comment is used for uploading process. Note that the remaining fields have been anonymized for transmission to a clinical trial coordinating site and do not contain protected health information (PHI).

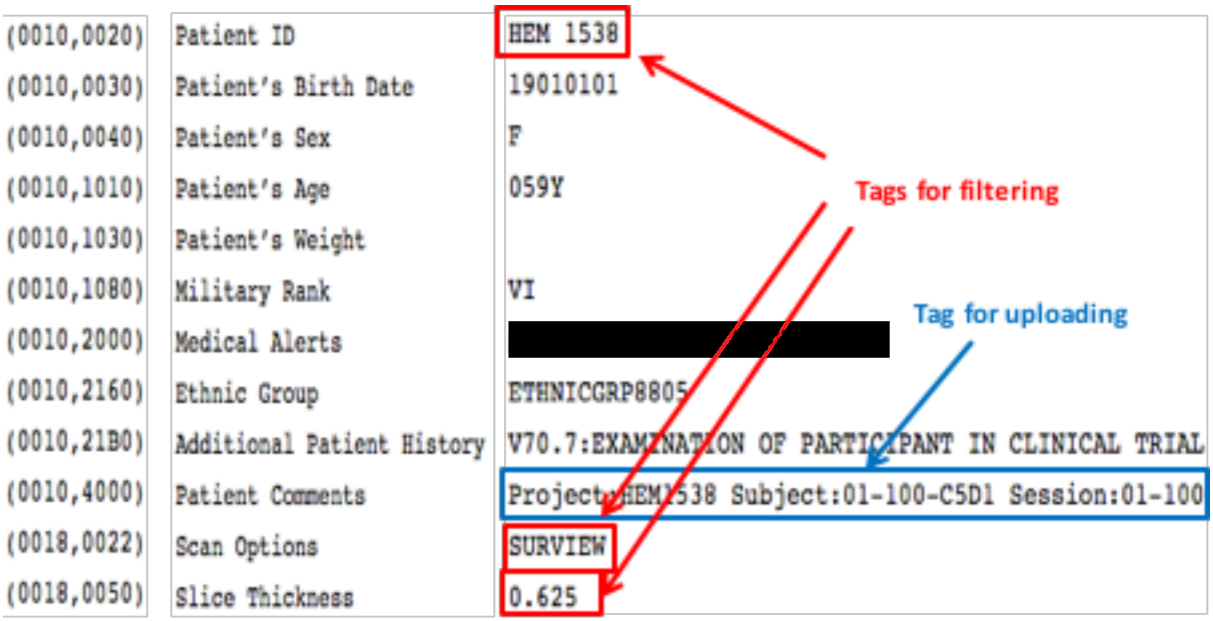


Table 2.1: Part of **DICOM** header tags. Header editing process modify 'Patient ID', and 'Session ID'. And both modified tags are used to make 'Patient Comments' which is server's sensing tag comment.

2.3 Quality Assurance

Ahead of tag editing procedure, bad images; not a real image, none of the three-view image (axial, coronal, sagittal), or scans that has too small or too large thickness compared to regular **DICOM** scans should be filtered so that the unintended data is not uploaded on the repository.

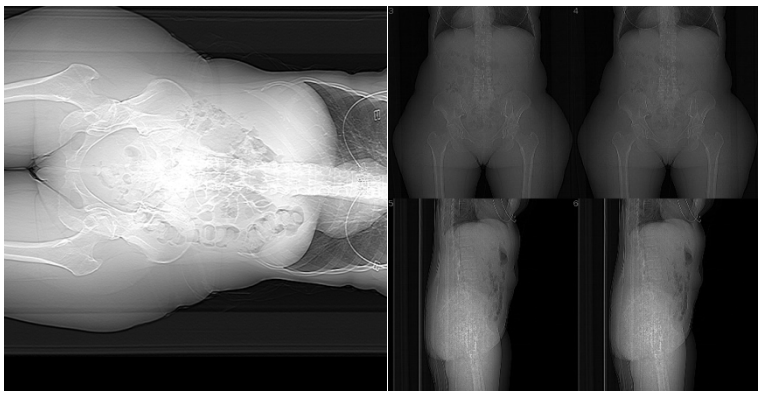


Figure 2.1: Image and data that are filtered through Quality assurance. (Left: image that include 'SCOUT' in ScanOptions. Right: image that has slice thickness which is larger than 15 mm.)

2.3.1 Functional QA

For the **DICOM** CT data, it is at first filtered before the 'commenting' process using 'Functional QA'. This is conducted by python function library '**dicom**'. If there exist a certain error when the **DICOM** image is tried to be opened with '**read_file**' function, system returns exception error related to functional issue.

2.3.2 Keyword QA

If the data has passed functional QA, next filtering is applied by keywords. Since **DICOM** image has specific headers such as 'ScanOptions' and 'SliceThinkness', the data can be filtered using these keywords. In **DICOM** images, there can be numerous kind of scan options and two representative options that have to be filtered out are 'DOSE' and 'SCOUT'. 'DOSE' means the data includes dose report in it so that it does not have real scan image. 'SCOUT' means that the data is 'scout view image'. A scout view is a preliminary image obtained prior to performing the major portion of a particular study. The scout image also has to be excluded. Slice thickness is relatively vaguer than scan options because thickness can be selected subjectively. However, it can also be good method for filtering irrelevant data out. **[7-8]** The designated slice thickness range for this QA is $0.5 <$ thickness < 15 mm. If the image has passed both functional and keyword quality assurance, the tag (0010,4000) will be edited. **Fig. 2.1** shows sample data that has to be not uploaded or filtered out through this process. Since numerous data still should not be used for segmentation algorithm, this keyword QA should follow functional QA.

2.4 Data Push

After header tag editing process, **DICOM** input data should be on the repository for being reported and utilized for later cases and experiments. In this paper, used repository, server, is **XNAT** which is operated by VUIIS Center for Computational Imaging (VUIIS CCI) from Vanderbilt University. **XNAT** is server that certain function or pipeline, so called 'Spider' can work automatically for the project user makes. Connection to server and pushing to server can be conducted with the python command, 'python storescu.py -aec AE server port' where AE is application entity title, and **DICOM** SCP server host with

port number. It will shows server connection status, and uploading result with the system message that is included in the ‘storescu()’ function.

2.5 Preprocessing

DICOM images are 3D volumetric images, so they should be sliced in to set of 2D images to be the model of the deep neural network, since the network training sets are composed as 2D. Before the resampling, the situation of the clinical trial data does not have the same orientation, which can cause fatal segmentation false, should be checked and dealt with the spatial correction function.

2.5.1 Image Resampling

NIfTI format image that is contained in header is 3-dimensional. 3D image itself can be utilized as an input data for deep neural network, but since most of the training data, and also the kernels(masks) that are being widely used in neural network are 2D, 3D medical image has to be converted into 2D slices. From original image, singleton dimension of the image is removed and merged into one 2D slice. After down-sampling, images also have to be resized into certain dimensions which depend on the networks being used. In the paper, the network requires 512x512 dimension image, so that it can be trained and validated in the process.

2.5.2 Spatial Correction

CT image has its own orientation. This orientation has to be same for all the input scans and for the training data that is used on deep learning process.

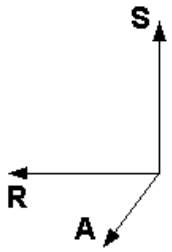


Figure 2.2: Neurological convention for axes, aka "**RAS**"

Fig. 2.2 shows orientation standard for letting all the input data and trained data to have corresponding direction of image. Used method for this is Right, Anterior, Superior (**RAS**) coordinate system. The reason we unify all the orientation of data into **RAS** formation is RAS is significant when performing matrix and vector math, where a right-hand coordinate system is customarily used (though a left-hand system can be used with appropriate adjustments). **[9]**

2.6 Convolutional Neural Network

For the spleen segmentation, we implement and validate deep learning using Convolutional Neural Network (**CNN**). Typically, a large amount of clinical trial spleen image is being segmented manually. The more slices CT image has, the more time it takes to manually label. To reduce time significantly, **CNN** is adopted. Especially, in this paper, end-to-end synthesis and segmentation network (**EssNet**) proposed at **[6, 10]** is selected to implement methods.

2.6.1 EssNet

The **EssNet** was trained by unpaired MRI and CT scans and only used manual labels from MRI scans. **EssNet** has been developed based on the **ResNet [11]**. Network is trained with already-stored training set, and make fake images that would be synthesized into

discriminator network from original images. This is called Generative Adversarial Networks(**GAN**), and it is improved towards **CycleGAN [12]**. Although this paper only uses CT clinical trial, there exists frequent cases that the input scans are not only the CT images, but also the MRI images. In these cases, both two images have to be dealt with paired vector of image set, or unpaired image. However, under most circumstances, paired-images do not exist so that researchers must make new image or data and synthesize it with the other one. **CycleGAN** uses unpaired image sets and furthermore, it offers 'translating back and forth' within images although they are not in group. **[12]**

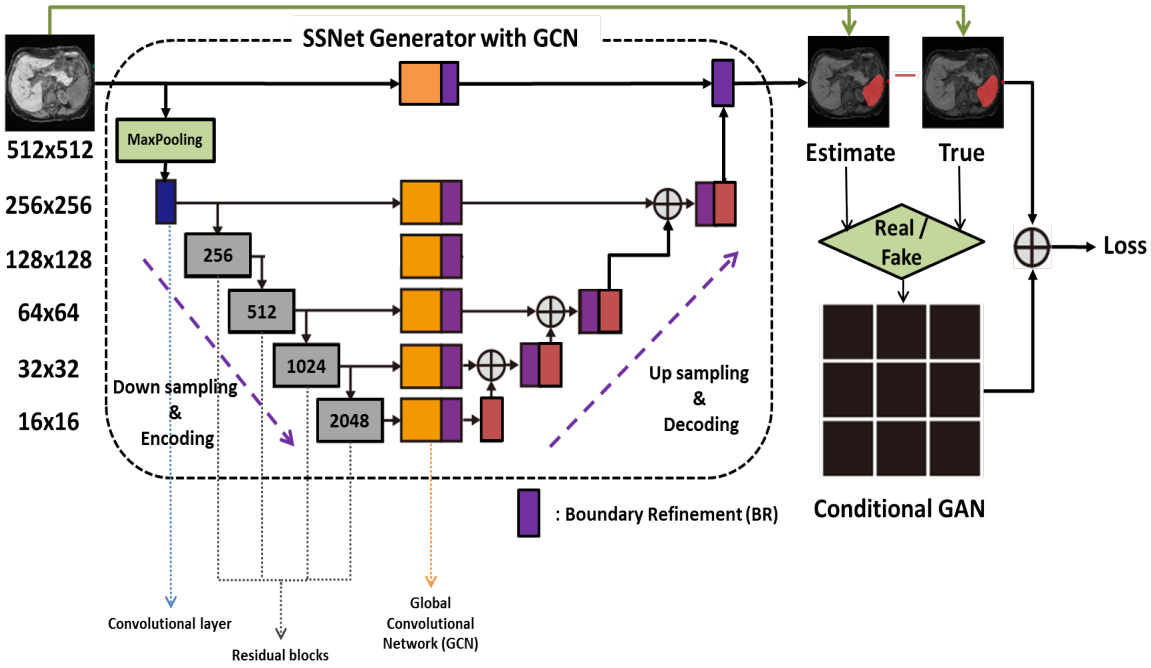


Figure 2.3: Adopted Splenomegaly Segmentation Network (**SSNet**) structure. Stride = 2, padding = 3. Blue box represents convolutional layer, grays for residual blocks, orange squires show Global Convolutional Network (**GCN**). Decoding and up-sampling process is conducted with Boundary Refinement (**BR**) and Deconvolutional layers.

Fig. 2.3 shows the **SSNet** being used for validate the input data to get the ideal labeled images. At encoding process, which is left part of **GAN** generator, four hierarchical residual blocks are used to extract feature from the input data. These encoding process plays a role

like principal component analysis, which is extracting specific part from the object. **[14]** Feature maps from each layer goes into Global Convolutional Network (**GCN**) for having new subsidiary feature arisen in the deep learning process. Those features are boundary refined with the method proposed in **[13]**. Conditional Global Adversarial Networks (**cGAN**) then are used for discriminate specific features from the extracted map. The loss rate is evaluated by the loss from the difference of estimate and ground truth, and **GAN** loss.

$$Loss_{SSNet} = Loss_{Dice} + \lambda \cdot Loss_{GAN} \quad (1)$$

$Loss_{Dice}$ means the Dice Similarity Coefficient (**DSC**) between the segmentation result from the network and the manual segmentation. The $Loss_{GAN}$ represents **GAN** loss that is Cross Entropy (**CE**) loss between **cGAN** estimations and ground truth model. λ is a constant value that indicates weights for adding two losses, and also was suggested from **[10]** as '100'. **Fig. 2.4** explains how main network runs in order. As it can be shown at **Fig. 2.4,** in Initializing part (**INIT**), the algorithm makes layers and blocks. Dimensions of the blocks depend on the residual blocks and are targeting toward the output dimension. 'Initializing 10 Boundary Refinement (**BR**) blocks' are used for the purple box processes in **Fig. 2.3.**

Algorithm 1 FCNGCN

```
procedure FCNGCN
def INIT:
    resnet ← models.restnet50
    conv1 ← nn.Conv2d(settings for kernel size, stride and padding)
    relu ← resnet.relu
    maxpool ← resnet.maxpool
    textThen
    testInit4Layers(layer1, layer2, layer3, layer4
    gcn1 ← GCN(2048,number of classes)
    gcn2 ← GCN(1024,number of classes)
    gcn3 ← GCN(512,number of classes)
    gcn4 ← GCN(64,number of classes)
    gcn5 ← GCN(64,number of classes)
    Then
    Init 10 Boundary Refinement(BR) blocks
    Then
    Init Deconvolutional Classifier blocks
    Then
    transformer ← nn.Conv2d(256,64,kernelsize=1)
def classifier:
    InitConv2d
    InitBatch Normalization
    InitReLU Activation Function
    InitDropout
    InitConv2D for Dropped out layer with half dimension
def forward:
    for Only First Layer do
        perform Maxpooling
        x ← input
        x ← conv1(x)
        x ← relu(x)
        conv_x ← x
        x ← maxpool(x)
        pool_x ← x
    fm1 ← layer1(x)
    fm2 ← layer2(fm1)
    fm3 ← layer3(fm2)
    fm4 ← layer4(fm3)
    Then
    gcfm1 ← gcn1(fm4)
    gcfm2 ← gcn2(fm3)
    gcfm3 ← gcn3(fm2)
    Then
    fs1 ← refine(upsample(gcfm1,fm3 + gcmf2))
    fs2 ← refine(upsample(fs1,fm2 + gcmf3))
    fs3 ← refine(upsample(fs2,pool_x + gcmf4))
    fs3 ← refine(upsample(fs3,conv_x + gcmf5))
    out ← refine(upsample(fs4,input))
     return out
```

Figure 2.4: Pseudo Code for fundamental **SSNet** algorithm.

2.6.2 Training Set

There should be training set of data in advance of evaluating and getting segmentation results. In this paper, 94 manually segmented spleen images are used to train the network. Those 94 training sets are from CT, and images have all the same dimension which is 512x512. Among 94 scans, 75 are from regular spleen, and 19 are from splenomegaly.

2.6.3 Network Function Parameters

For the training of the network, the paper chose the **Pytorch** library, which is using **Python** as a base language. With the function made by **Python**, training model for evaluation is made.

Variable	Description	Value
model_name	For spleen or whole body	'Spleen'
network	The network that is used	206 (=ResNet)
workers	Number of data loading workers	1
batchsize_lmk	Input batch size for lmk	4
lr	Learning rate	0.00001
augment	Whether use augmented or not	False
accreEval	Whether only evaluate accre result	False
viewName	For axial, coronal, sagittal view	view3
loss_fun	Dice | Dice_norm | Cross_entropy	cross_entropy
lmk_num	Number of output channels	7

Table 2.2: Parameter variables for the training model with network.

Table 2.2 shows list of the parameter for function call. There are more parameter settings inside such as 'epoch' for 'how many times to be iterated', 'cuda' for using GPU **CUDA** of **NVIDIA**, 'test_loader' for loading list of the test files, which are the input data made up to the text file list. With the parameters at **Table 2.2,** training begins. Selected model is Global Convolutional network (**GCN**) as mentioned above, and it is made by **Python** code.

Contained functions are **'GCN()'** for main training, **'Refine()'** that has boundary refinement role which is conducting **ReLU**, and Batch normalization as activation process. Overall processes are explained with **Fig. 2.3**.

2.7 Post Processing

After main segmentation procedure, data may still have problems even though it has been worked well with algorithm. One possible case that can happen is related to the image quality. Because of the lack of plentiful training set, diversities in input data set, and other issues during previous steps, there could be not perfect. For these reasons, we implemented image processing algorithms, especially methods for noise filtering. **[15-17]** After image processing steps, processed 2D images also have to be reconstructed into 3D volume to get the same dimension as original input.

2.7.1 Image Processing

Image processing algorithms are implemented in the process after segmentation. Since the segmentation can be filled imperfectly; includes holes in segmentation, uneven edges, rough labeling, image filtering is necessary. **Fig 2.5** shows the effectiveness of image processing steps as a post-processing, and result of the implementation. On the segmentation image, image opening, defined function get_largest_connection(), and closing are applied in a row. Those works remove most part of the noises and make contours of segmentation smoother with little loss of information.

2.7.1.1 Dilation

Image dilation is used for 'filling'. Even though the deep neural networks provide ideal result of segmentation, there usually exists certain hole inside of the segmented tissue. Through image dilation algorithm, those holes can be remarkably filled up.

$$X \oplus B = \{ (x + b) | \, x \in \mathrm{X}, b \in B \, \} \quad (2)$$

$$X \oplus B = \{ p \in E^2 | \, p = x + b, x \in X \; amd \; b \in B \} \quad (3)$$

where **X** is an original image input and **B** is structuring kernel. Output of the **Eq. (3)** can be Image **I**. Starting with all-zero image **I**, scan among the loop for all $\mathbf{b} \in \mathbf{B}$. In loop, $\mathbf{X_b}$, which is shifted image is used to update output image **I** such in formula of $\mathbf{I} = \mathbf{I} \vee \mathbf{X_b}$. Dilation is being utilized to fill small holes or bays in objects. For the result, the size of the object increases.

2.7.1.2 Erosion

Erosion is another technique for noise filtering. After filling the holes up with the image dilation method, remaining noise pixels should be removed for having only the target feature well. Erosion is used for this procedure.

$$X \ominus B = \{ x \in \mathrm{E}^2 | \, (x + b) \in \mathrm{X}, \; for \; every \; b \in B \quad (4)$$

where **X** is an original image, and **B** is structuring kernel element. Equivalently, it is verified for each image pixel **p**, whether the result fits to **X** for all possible **x+b**. If yes then the

outcome is 1, otherwise 0. As being shown in **Eq. (4)**, in the image, objects smaller than the kernel **B** vanish. Being opposite to Dilation, erosion has a role of simplifying structure. For the result, the size of the object decreases.

2.7.1.3 Get Largest Connection

In addition to fundamental image morphological algorithms above, function for getting largest connection between voxel (or pixel) is used. For 3D input image, first process conducted is finding connected voxels' lengths. For the voxel index that has less than **0.5***max (connected voxel length), it can be set as '0'.

Figure 2.5: Implementation of Image processing procedures. Left: before morphological image processing (opening – largest_connection – closing in order), Right: after morphological image processing.

2.8 Containerization

2.8.1 Docker

About the whole pipeline processes above, they need to be made up to large container that make these steps end-to-end. **Docker**, which is the container platform to provide application to user across the hybrid cloud. For running the pipeline, there exist a bunch of requirements and pre-installation process needed; preprocessing and post-processing functions are made by **Matlab** codes, segmentation by **Pytorch**. Besides, since the machine or Operating System (**OS**) of clinicians or users are various, the system environment should be identified. These required packages and **OS** often could be more than some gigabytes, so that it takes lot of time for user to search those packages and install them by themselves. **Docker** can reduce those time first of all. As a **'Dockerfile'** contains, all the packages, software and even Operating Systems can be pre-included into this file. those 'pre-installed on the **Docker** image' packages are stored on the **Dockerhub** based on Amazon Web Service(**AWS**). Once the user builds a **Docker** and run it using **Docker** command, they can get directly the result pdf demonstrations they need. This **Docker** container is mainly docked to user's machine and make the users be able to have new virtual machine environment. **Dockerfile**, which has whole system backgrounds from Operating System (**OS**) to tiny functions for running subsidiary pipeline process. **Fig. 2.6** shows brief components and software that are included in **Dockerfile**.

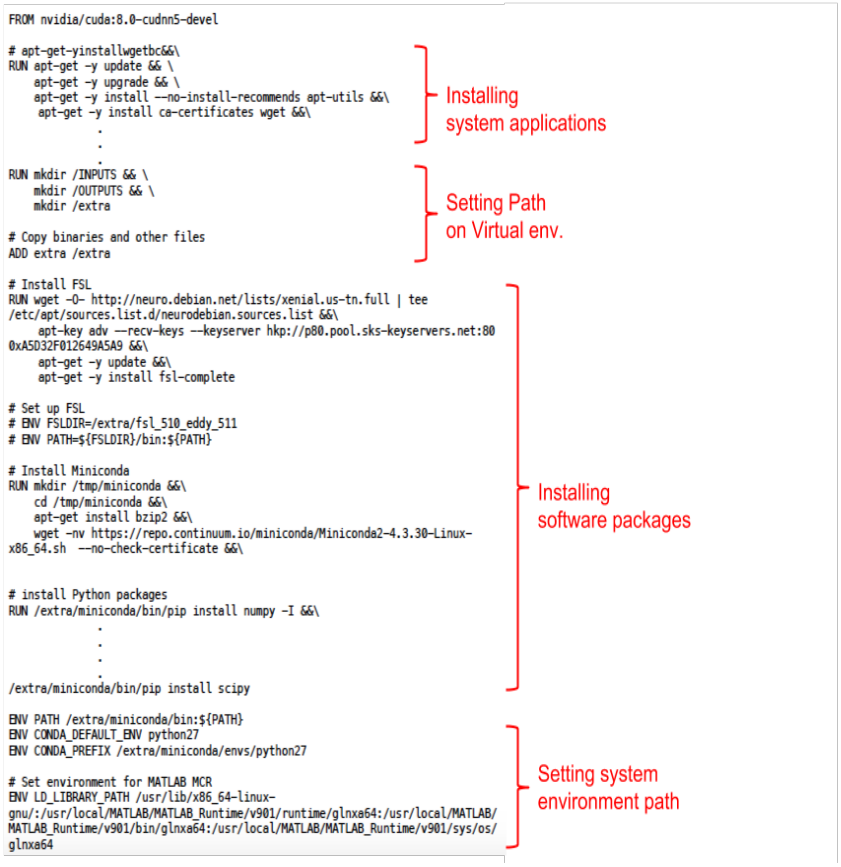


Figure 2.6: Dockerfile being included in **Docker** container. This allows container contains not only for the pipeline functions, but also the fundamental system applications and even Operating System (**OS**) in it.

EVALUATION

3.1 Dataset

Clinical Trial that is being used in the paper to test the pipeline is CT images from project protocol **HEM1538** whose title is 'A Phase 1 Dose Escalation and Expansion Study of *TGR-1202 + Ruxolitinib in Subjects with Primary Myelofibrosis (PMF), Post-Polycythemia Vera MF (PPV-MF), Post-Essential Thrombocythemia MF (PET-MF), MDS/MPN, or Polycythemia Vera Resistant to Hydroxyurea'*. Our subset of the **HEM1538** trial consists of 185 scans from 56 subjects in de-identified form. Before the QA process, the protocol has data that should not be uploaded and go into the pipeline process. **Fig. 3.1** shows three sample scans from **HEM1538** dataset which are all filtered-out scans.

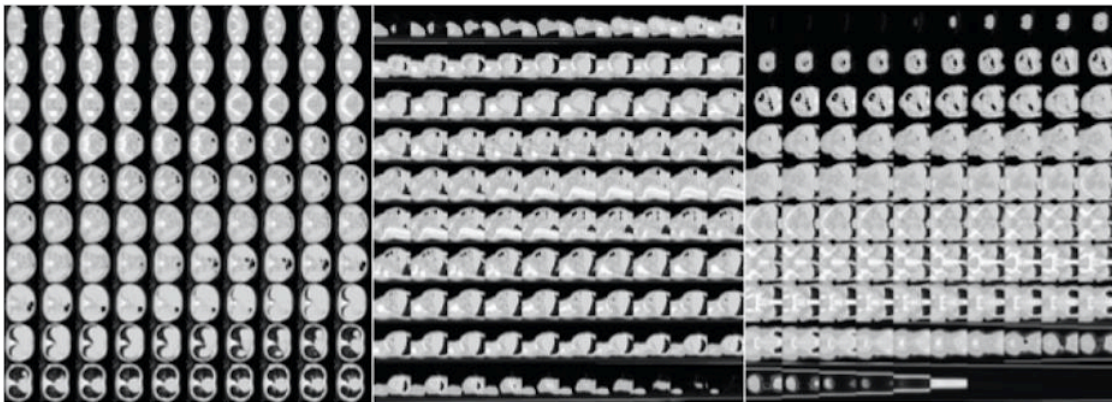


Figure 3.1: HEM1538 Clinical Trial dataset sample. Above three views are the filtered-out data.

3.2 Pipeline Performance

As a result, the automated pipeline showed good performance in general. There exists an issue that partial regions of the spleens from some subjects are imperfect after being processed by the pipeline, so that additional manual labeling tasks are necessary. There could be various reasons for these symptoms, and potential reasons will be mentioned at **Chapter 4**. After conducting additional manual labeling, the segmentation result with overlaid volume was evaluated by a radiologist. Although we processed additional labeling works, the time expense for segmentation with the pipeline was still much faster than pure manual labeling, and the performance of the pipeline corresponds manual segmentation as well.

3.2.1 Segmentation Time

Spleen Segmentation for Clinical trial CT has been traditionally done manually. For the comparison between manual segmentation and auto-segmentation with the pipeline, Manual labeling speed should be verified first. One scan, which has 73 slices of 'spleen' part was manually labeled, and elapsed time was measured. Supposed there is no rest within labeling, elapsed time was 24 minutes. By this trial, average manual segmentation speed was set **as 20 second per slice**. For the whole dataset, which has 185 scans and 10552 slices that can be regarded as 'Spleen' part (slices that have spleen tissue in it), total manual segmentation time without any break is approximately **3,517 minutes**. And this is almost **59 hours** in total.

As for the pipeline process, we went through the whole pipeline procedure with the machine, that has specification, Intel(R) Xeon(R) CPU E5-1620 v3 @ 3.50GHz. 64bit with 256GB memory capacity, for the processes except deep neural network, and GPU with

specification of GEFORCE GTI 1080 Ti for deep neural network one. For the automated process without any additional manual labeling, it took **1,659 minutes** for total 185 scans of data. Since the results from this process contains some issues in the labels, manual editing process has been conducted. Manual editing process took **416 minutes**, for 36 scans selected for further manual process. In total, auto-pipeline with supplementary hand-operated editing took **2,075 minutes**, which is still faster than the traditional method. Considering that fully-manual labeling requires breaks and the hardware performance can be improved, this pipeline process could be advanced more.

3.2.2 Qualitative Result

For the demonstration purpose, the pipeline returns volumetric value with the pdf format views. **Fig. 3.2** shows one example of final overviews that clinicians and patients can directly get as 'Final result' of end-to-end pipeline.

As being shown at **Fig. 3.2**, pdf overview includes slice capture of overlaid scan so that users can easily assess and see visually. It also gives spleen volume calculated by counting labeled voxel in the segmented image. Since the pipeline return value can only be used for demonstration purpose, it should not be utilized for clinical usage.

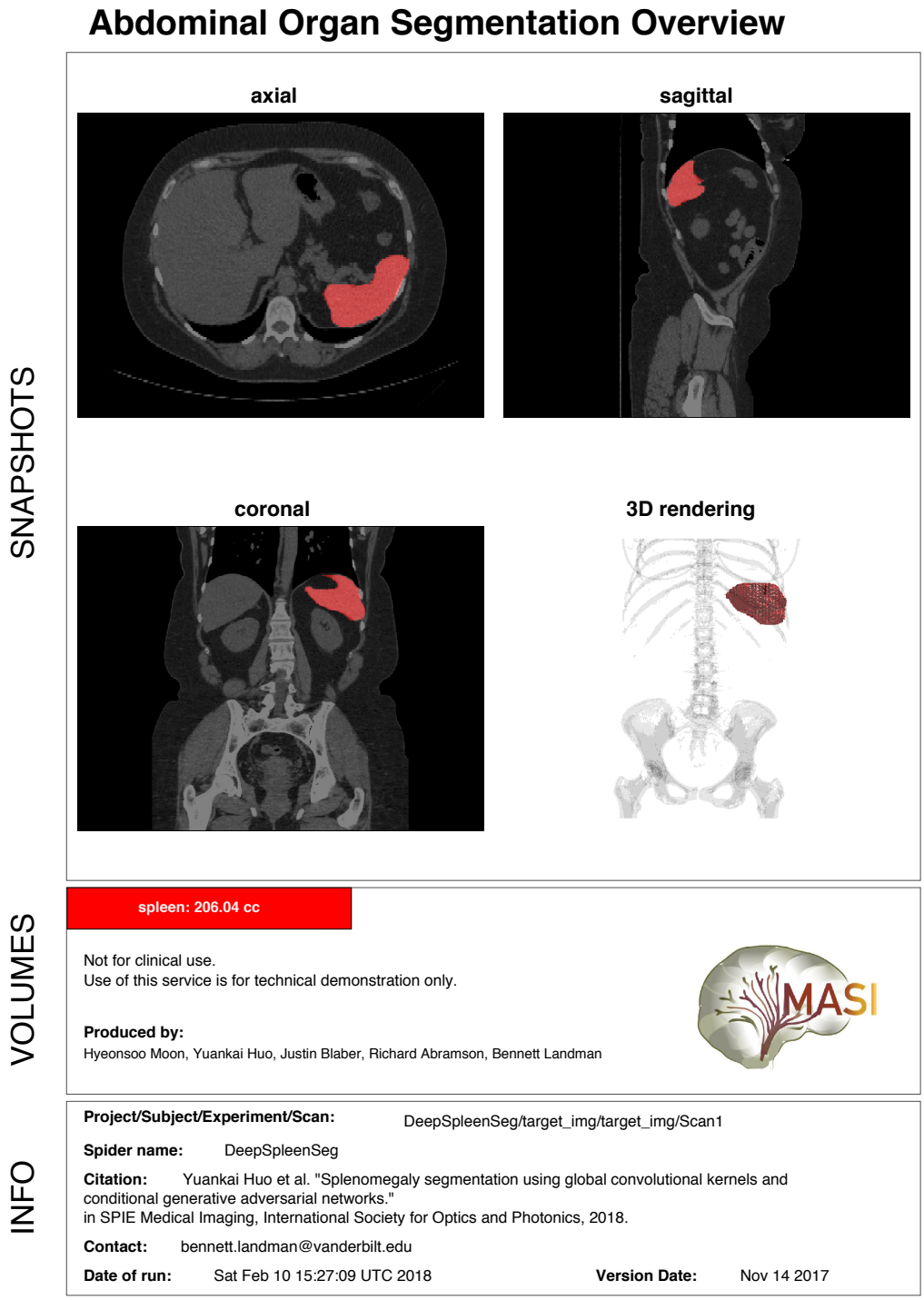
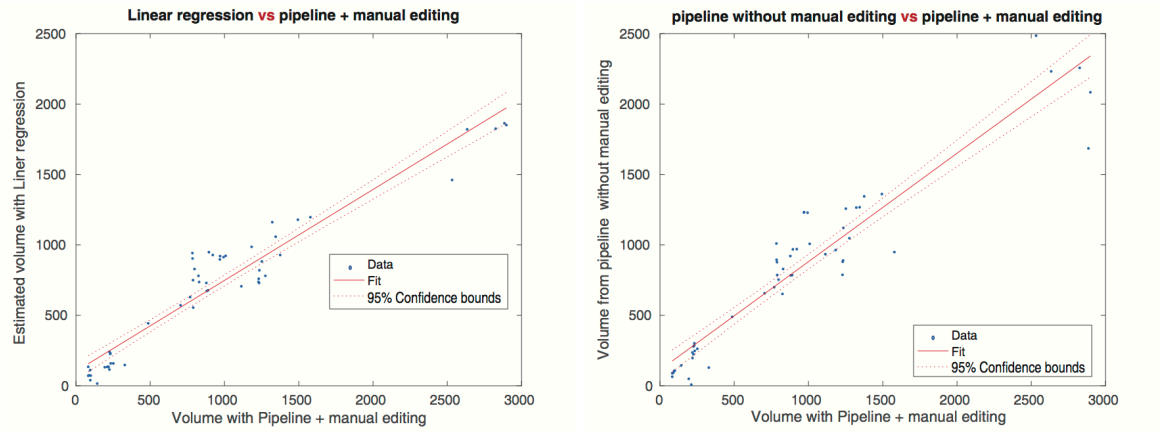


Figure 3.2: PDF Output for demonstration purposes. The pdf format includes three sections mainly. Snapshots section describe axial view, sagittal view, coronal view of mean slice, and rendering of 3D volume. Volume section shows brief description and actual spleen auto-volume from the pipeline. Info section has scan information and assessment data.

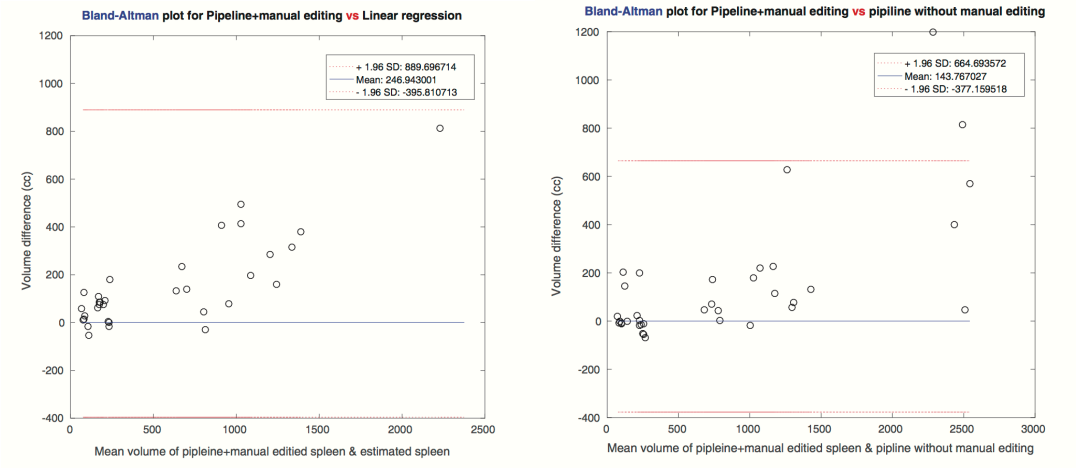
3.2.2 Quantitative Analysis

The pipeline performance can also be evaluated quantitatively using the method that comparing correlation of the spleen length of after-pipeline process and manual spleen length. Spleen volume is calculated by the maximum craniocaudal length, maximum size in axial plane(width), and thickness on axial scan. The equation for spleen volume size is introduced at **[18,19]. [18]** describes that spleen length can be a standalone feature to determine spleen issues related to the volume like splenomegaly because it correlates well with the splenic CT volume. **[19]** shows that 'Volume [cc] = 30 + 0.58 x L (cm) x D (cm) x T (cm)', so this also indicates that spleen volume measurement can be assessed quantitatively with a single parameter which is length of spleen. In this paper, we calculate volume estimation with the length of the spleen using linear regression **[18]** and draw correlation map of estimated spleen volume comparison for both auto-pipeline method and manual to evaluate the performance of the pipeline. **[18]** introduces that the spleen volume can be estimated using singular indexes (Length, Width, Thickness). Our analysis is first focused on evaluating (1) correlation between segmentation from pipeline with manual editing and from linear regression method, and (2) correlation between segmentation from pipeline with manual editing and before the manual labeling. The evaluation is done only for the 36 scans that have to be manually edited. **Fig. 3.3-(a)** shows compared plot of correlation for case (1), and (2). For this test, the estimation using linear regression shows better performance than our pipeline using deep neural network. The reason why the x-axis, which is the standard ground truth model for this experiment is since we manually edited imperfect scans and had confirmation from a radiologist, processed scans can be said as 'ground truth'. While the regression estimated volume showed correlation score of 0.9814, the pipeline without manual labeling returns correlation score as 0.9704. This can be described with two factors.

The one is the fact that the number of training set used in the pipeline for labeling is too small and specifically, even among the small amount of training set, scans for splenomegaly were only 19, that is too small to validate 185 scans of data. The other reason can be 94 trained scans themselves for more than twice amount of test set are not enough for ideal segmentation **[20].** Especially, these insufficiency of performance is significantly shown for the splenomegaly whose spleen size is bigger than 1500cc. Even though the pipeline processed volume shows lower correlation coefficient value than the estimated volume set, the pipeline still gives reasonable results with very limited number of training dataset. Besides, considering that this result is before additional manual editing, the performance of the pipeline has much more possibilities to be improved. Furthermore, even though the linear regression estimation of spleen volume has shown bit better performance for various types of scan inputs, the overall differences and errors of the linear regression methods are greater than the pipeline method. This difference plot can be shown with the **Fig. 3.3-(b),** which is using Bland-Altman plot **[21].**



(a)



(b)

Figure. 3.3: Quantitative analyze result plots. (a) represents correlation map for left: spleen volume from linear regression estimation and spleen volume from the pipeline with manual editing(GT), and right: spleen volume from the pipeline and GT. It is remarkable that for the large spleens, the pipeline method shows degraded performance. (b) introduces Bland-Altman plots for both two cases. It can be verified that the difference gap of linear regression estimated volumes is bigger (wider) than the pipeline method.

DISCUSSION

4.1 Volumetric issue

Could be mentioned with outlier in the correlation map, and also with the numerical value that can be generated through pipeline.

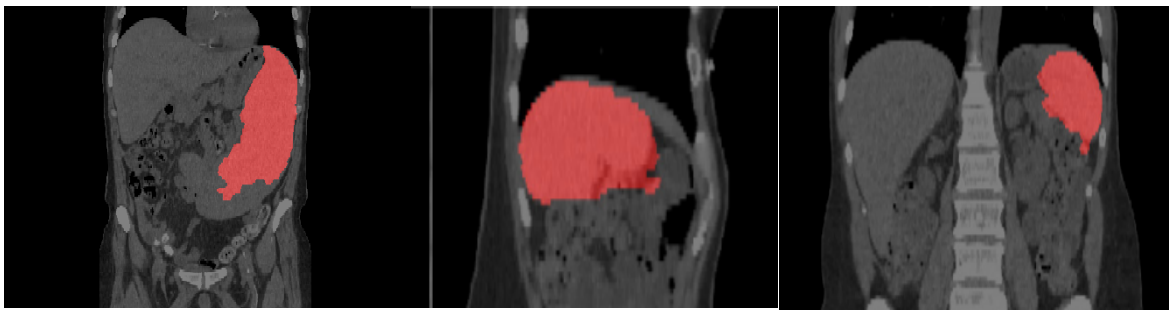


Figure 4.1: Imperfect segmentation generated by the deep neural network process. (Left: spleen volume beneath lost, Middle: there exists over-labeled part of spleen. Right: there exists 'bleeding' label from spleen.)

As **Fig. 4.1** shows, the pipeline generates imperfect labels for some scans. (In here, 36 scans). The biggest reasons that can impact the segmentation performance itself is lack of training dataset, and variety of spleen size among human. The latter factor can also be resolved when the former issue gets improved. In most cases of studies in field of machine learning and recognition, the size of training dataset used to at least same as test dataset. The most widely used ratio is 70% for training set, and 30% for test set. Since the dataset being used as training set in the paper has a size of 94 (75 normal spleens, 19 splenomegaly) in despite of the test set has 185 scans. That can cause exceptions and imperfect segmentation result in the process.

4.2 Future Work

First of all, as it mentioned at volumetric issue, there are 36 slices that have undergone additional manual editing process, and that can be caused by lack of training dataset and diversity in human spleen size. What can be done in the future is, we can use the segmented volumes, which are generated by the pipeline and additional manual editing process. Those improved segmentation images can be the part of the new bigger training set, so that the pipeline can be improved as long as it processes the data. One issue that should be considered in this process is, there exist relation between learning rate in the neural network and the size of the training set **[20]**. Therefore, learning rate should also be modified and analyzed in accordance with the size of the training set.

Additionally, we evaluated the performance of the pipeline with the spleen volume estimation by linear regression method alone. As **[18]** asserts, with the measured spleen indices, which are not only the length, but also the width (perpendicular to hilum, maximum on any section), and thickness (perpendicular to width, maximum on any section), the volume estimation can be conducted. As a later work, more evaluation could be done by comparison of estimated spleen volumes and the spleen volumes derived by the pipeline in the future. By using different methods from the suggest linear regression method **[18]**, such as equation 'Volume [cc] = 30 + 0.58 x L (cm) x D (cm) x T (cm)' proposed in **[19],** or by using other index information like width and thickness, estimation measurement for the spleen volume can be also compared to the auto-pipeline paired by manual editing process

CONCLUSION

The main motivation of this thesis work is originated the inconveniences in the middle of clinical trial segmentation process. A number of problems exist at numerous parts and subsidiary process. Traditional method for clinical trial segmentation which is manual labeling used to require so much redundant time and funding to do so. Semi-automated process for segmentation has been developed, but still difficult for clinicians to utilize it without additional tasks and studying. So, we focused on lessening the processing time, with maintaining corresponding quality of the segmentation performance. We stress that the pipeline reduces additional manual processing time rather than the auto-pipeline's speed itself, because the auto-pipeline proposed in the paper could be faster and improved depending on the hardware specification and system environments. To develop an end-to-end system, we combined methods from pre-processing (**DICOM** header editing, Quality Assurance, Slicing 3D volumes into set of 2D slices), segmentation procedure (based on SSNet using resnet network and GAN), and post-processing (Image processing – Closing, Opening, Get_largest_connection). For the demonstration purposes, final result that includes labeled scans and spleen volume data is shown as **PDF** format. Whole subsidiary and intermediate outputs such as 2D slices, Resized dimensionality images, Label images before the overlay are saved under the sub-directories, but the final output only includes **PDF** file and Volume size. Docker container has been an ideal method to carry our idea on, and to release the pipeline system to users. For more improvement, our future goal would be (1) evaluating with rest two singular standalone indexes (width and thickness), (2) to have more analyzing method for evaluating and proving the performance of pipeline using volumetric correlation map using the estimated volume by different methods **[19]** and true volume value,

and (3) constructing more dataset based on the segmentation result derived by the pipeline to have more training set. This process will make segmentation neural network more concrete and has more reliable performances.

REFERENCES

[1] Paley, M. R. and P. R. Ros (2002). Imaging of spleen disorders. The Complete Spleen, Springer**:** 259-280.

[2] Redmond, H., et al. (1989). "Surgical anatomy of the human spleen." British journal of surgery **76**(2): 198-201.

[3] Kang, W.-X., et al. (2009). The comparative research on image segmentation algorithms. Education Technology and Computer Science, 2009. ETCS'09. First International Workshop on, IEEE.

[4] Sharma, N. and L. M. Aggarwal (2010). "Automated medical image segmentation techniques." Journal of medical physics/Association of Medical Physicists of India **35**(1): 3.

[5] Ji, S., et al. (2013). "3D convolutional neural networks for human action recognition." IEEE Transactions on Pattern Analysis and Machine Intelligence **35**(1): 221-231.

[6] Huo, Y., et al. (2017). "Simultaneous total intracranial volume and posterior fossa volume estimation using multi-atlas label fusion." Human brain mapping **38**(2): 599-616.

[7] Alshipli, M. and N. A. Kabir (2017). "Effect of slice thickness on image noise and diagnostic content of single-source-dual energy computed tomography." Journal of Physics: Conference Series **851**.

[8] Seet, K., et al. (2010). "Optimal slice thickness for cone-beam CT with on-board imager." Biomedical imaging and intervention journal **6**(3): e31.

[9] Yung, K. T., et al. (2011). "Atlas-based automated positioning of outer volume suppression slices in short-echo time 3D MR spectroscopic imaging of the human brain." Magnetic resonance in medicine **66**(4): 911-922.

[10] Huo, Y., et al. (2018). Splenomegaly segmentation using global convolutional kernels and conditional generative adversarial networks. Medical Imaging 2018: Image Processing, International Society for Optics and Photonics.

[11] Szegedy, C., et al. (2017). Inception-v4, inception-resnet and the impact of residual connections on learning. AAAI.

[12] Zhu, J.-Y., et al. (2017). "Unpaired image-to-image translation using cycle-consistent adversarial networks." arXiv preprint arXiv:1703.10593.

[13] Peng, C., et al. (2017). "Large Kernel Matters--Improve Semantic Segmentation by Global Convolutional Network." arXiv preprint arXiv:1703.02719.

[14] Mahmood, A., et al. (2017). Resfeats: Residual network based features for image classification. Image Processing (ICIP), 2017 IEEE International Conference on, IEEE.

[15] Koyuncu, H., et al. (2017). "An Efficient Pipeline for Abdomen Segmentation in CT Images." Journal of digital imaging: 1-13.

[16] Serra, J. (1983). Image analysis and mathematical morphology, Academic Press, Inc.

[17] Gil, J. Y. and R. Kimmel (2002). "Efficient dilation, erosion, opening, and closing algorithms." IEEE Transactions on Pattern Analysis and Machine Intelligence 24(12): 1606-1617.

[18] Bezerra, A. S., et al. (2005). "Determination of splenomegaly by CT: is there a place for a single measurement?" American Journal of Roentgenology 184(5): 1510-1513.

[19] Estimate of spleen volume on CT or MRI, splenic index – calculatorhttp:// radclass.mudr.org/estimate-spleen-volume-ct-or-mri-splenic-index-calculator

[20] Eaton, H. A. and T. L. Olivier (1992). "Learning coefficient dependence on training set size." Neural Networks 5(2): 283-288.

[21] Xu, Z. (2016). Automatic Segmentation of the Human Abdomen on Clinically Acquired CT, Vanderbilt University.