MEDICAL IMAGE SEGMENTATION USING DEEP LEARNING

A Project-II Report

Submitted in partial fulfillment of requirement of the

Degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE AND ENGINEERING

BY
Tanusha Shrivastava
(EN16CS301278)

Under the Guidance of
Prof. Sachin Solanki
Ms. Janhavi Deshpande

Department of Computer Science and Engineering
Faculty of Engineering
MEDI-CAPS UNIVERSITY, INDORE- 453331

May, 2020

MEDICAL IMAGE SEGMENTATION USING DEEP LEARNING

A Project-II Report

Submitted in partial fulfillment of requirement of the

Degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE AND ENGINEERING

BY

Tanusha Shrivastava
(EN16CS301278)

Under the Guidance of
Prof. Sachin Solanki
Ms. Janhavi Deshpande

Department of Computer Science and Engineering
Faculty of Engineering
MEDI-CAPS UNIVERSITY, INDORE- 453331

May, 2020

Report Approval

The project work **“Medical Image Segmentation (Using Deep Learning)”** is hereby approved as a creditable study of an engineering subject carried out and presented in a manner satisfactory to warrant its acceptance as prerequisite for the Degree for which it has been submitted.

It is to be understood that by this approval the undersigned do not endorse or approved any statement made, opinion expressed, or conclusion drawn there in; but approve the “Project Report” only for the purpose for which it has been submitted.

Internal Examiner
Name:
Designation:
Affiliation:

External Examiner
Name:
Designation:
Affiliation:

Declaration

I hereby declare that the project entitled **"Medical Image Segmentation (Using Deep Learning)"** submitted in partial fulfillment for the award of the degree of Bachelor of Technology in 'Computer Science and Engineering' completed under the supervision of **Prof. Sachin Solanki, Assistant Professor, Department of Computer Science and Engineering,** Faculty of Engineering, Medi-Caps University and **Ms. Janhavi Deshpande**, **Scientific Officer(C), Electronics Department**, BARC is an authentic work.

Further, I declare that the content of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for the award of any degree or diploma.

Date: ___________ **Tanusha Shrivastava (EN16CS301278)**

<u>Certificate</u>

We, **Prof. Sachin Solanki** and **Ms. Janhavi Deshpande** certify that the project entitled **"Medical Image Segmentation (Using Deep Learning)"**submitted in partial fulfillment for the award of the degree of Bachelor of Technology by **Tanusha Shrivastava** is the record carried out under my guidance and that the work has not formed the basis of award of any other degree elsewhere.

Prof. Sachin Solanki	Ms. Janhavi Deshpande
Assistant Professor,	Scientific Officer (C),
Computer Science Department	Electronics Department,
Medi-Caps University, Indore	Bhabha Atomic Research Center,Mumbai

Dr. Suresh Jain

Head of the Department

Computer Science and Engineering

Medi-Caps University, Indore

Acknowledgements

I would like to express my deepest gratitude to Honorable Chancellor, **Shri R C Mittal,** who has provided me with every facility to successfully carry out this project, and my profound indebtedness to **Prof. (Dr.) Sunil K Somani,** Vice Chancellor, Medi-Caps University, whose unfailing support and enthusiasm has always boosted up my morale. I also thank **Prof. (Dr.) D K Panda,** Dean, Faculty of Engineering, Medi-Caps University, for giving me a chance to work on this project. I would also like to thank my Head of the Department **Prof. (Dr.) Suresh Jain** for his continuous encouragement for betterment of the project. I am also thankful to **Prof. Sachin Solanki**, Assistant Professor, Department of Computer Science and Engineering, Medi-Caps University, for his guidance.

I take this opportunity to express my gratitude and deep regards to my External Guide, **Ms. Janhavi Deshpande,** Scientific Officer(C), Electronics Division, BARC (Mumbai) and **Smt. Padmini Sridharan**, Scientific officer (H+), Electronics Division,BARC (Mumbai) for giving me an interesting project to work on and for thier exemplary guidance, monitoring and constant encouragement throughout the project.

It is their help and support, due to which we became able to complete the design and technical report.

Without their support this report would not have been possible.

Tanusha Shrivastava (EN16CS301278)
B.Tech. IV Year
Department of Computer Science
Faculty of Engineering
Medi-Caps University, Indore

Abstract

Medical Image Segmentation is the process of automatic or semi-automatic detection of boundaries within a 2D or 3D image. A major difficulty of medical image segmentation is the high variability in medical images. First and foremost, the human anatomy itself shows major modes of variation. Furthermore, many different modalities (X-ray, CT, MRI, microscopy, PET, SPECT, Endoscopy, OCT, and many more) are used to create medical images. The result of the segmentation can then be used to obtain further diagnostic insights.

The project demonstrates the applicability of deep learning models viz. convolutional neural networks for segmentation of CT (Computed Tomography) scan images. Analyzing about image processing and segmentation, Convolutional neural network has been used as the project contain data as image. In this project, we have trained a deep learning model (U-Net) to segment bones and liver part of CT images. U-Net is a Fully Convolutional Network (FCN) that does image segmentation. Its goal is then to predict each pixel's class. The model uses varied and labelled dataset containing 10,000 number of images. Segmentation model has been trained using Python libraries TensorFlow and Keras. The model achieved accuracy of 95% within 100 number of epochs. Accuracy of segmentation is improved using transfer learning approach. Transfer learning (TL) is a research problem in machine learning (ML) that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. The model trained using transfer learning achieved accuracy of 97% within 10 epochs and only 25% training dataset. Thus, Transfer learning technique proven beneficiary for time context also. Moreover, transfer learning technique can work easily with smaller datasets.

The future work will focus on improving the accuracy further and considering more organs for segmentation.

Keyword: Python, Computed Tomography, TensorFlow, Deep Learning, Convolutional Neural Networks, U-Net architecture, Transfer learning.

Table of Contents

List of Figures

List of Tables

Abbreviations

Abbreviation	Description
BARC	Bhabha Atomic Research Centre
CT	Computed Tomography
GPU	Graphics Processing Unit
CNN	Convolutional Neural Network
DICOM	Digital Imaging and Communications in Medicine
RELU	Rectified Linear Unit

Chapter - 1

Introduction

1.1 Introduction About Organisation

The Bhabha Atomic Research Centre (BARC) is India's premier nuclear research facility, headquartered in Trombay, Mumbai, Maharashtra. BARC is a multi-disciplinary research centre with extensive infrastructure for advanced research and development covering the entire spectrum of nuclear science, engineering and related areas.

BARC's core mandate is to sustain peaceful applications of nuclear energy, primarily for power generation. It manages all facts of nuclear power generation, from theoretical design of reactors to, computerized modelling and simulation, risk analysis, development and testing of new reactor fuel materials, etc. It also conducts research in spent fuel processing and safe disposal of nuclear waste. Its other research focus areas are applications for isotopes in industries, medicine, agriculture, etc. BARC operates several research reactors across the country.

1.2 Introduction About Project Assigned

Medical image segmentation has proved an essential discovery of Deep learning and Machine Learning technologies. Medical imaging is the technique and process of creating visual representations of the interior of a body for clinical analysis and medical intervention, as well as visual representation of the function of some organs or tissues. Medical imaging seeks to reveal internal structures hidden by the skin and bones, as well as to diagnose and treat disease. Medical imaging also establishes a database of normal anatomy and physiology to make it possible to identify abnormalities. Although imaging of removed organs and tissues can be performed for medical reasons, such procedures are usually considered part of pathology instead of medical imaging.

Image segmentation is considered the most essential medical imaging process as it extracts the region of interest (ROI) through a semiautomatic or automatic process. It divides an image into areas based on a specified description, such as segmenting body organs/tissues in the medical applications for border detection, tumor detection/segmentation, and mass detection.

The influence of imaging technologies on the study of biomedical and neuroscience lead a new research area. The present-day automated imaging system is able to acquire and archive tons of image data. Scientists work on imaging not only need machines, but also must have an overlook of the properties of machines. Ideally, use a computer to analyze an image with little or no expertise. People can’t himself detect the various organs in the body, for this they would require some devices and those devices may further require a algorithm to work and distinguish body parts. Recent developments in imaging technology have rendered possible automated acquisition of high-quality data volume electron microscopy data. However, purely manual or purely automated strategies to distinguish images of body parts are likely to fail. In this project Segmentation algorithm partitions the image into different sets of body parts like bones, liver, stomach, bladder etc. A good segmentation algorithm will be able to answer the following:

1. Given two different segmentations of the same image, how can we differentiate the mismatches between them?
2. Given two different segmentations of two images, how can we minimize the disagreements with the ground truth?

In this project a segmentation method is shown which leads to an accurate segmentation result.

1.3 Objective

The main objective of this project is segmenting the body parts in a given image. For example, if an image is given of the abdominal region than it would segment the parts of abdominal region like bone, stomach, liver etc. For achieving this objective, the model must be trained with deep learning techniques like convolutional neural network, U-Net architecture by using various types of libraries of Python.

1.3.1 Segmentation

In computer vision, image segmentation is the process of partitioning a digital image into multiple segments. The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze. Image segmentation is typically used to locate objects and boundaries (lines, curves, etc.) in images. More precisely, image segmentation is the process of assigning a label to every pixel in an image such that pixels with the same label share certain characteristics. The result of image segmentation is a set of segments that collectively cover the entire image, or a set of contours extracted from the image. Each of the pixels in a region are similar with respect to some characteristic or computed property, such as color, intensity, or texture.

In case of medical image segmentation, the aim is to:

- Study anatomical structure
- Identify Region of Interest i.e. locate tumor, lesion and other abnormalities
- Measure tissue volume to measure growth of tumor (also decrease in size of tumor with treatment)

Various types of method can be used to segment medical images:

- Automatic Segmentation
- Edge based Segmentation
- Neural Network based Segmentation
- Region based Segmentation

This project is mainly concerned with Neural Network based Segmentation.

1.3.1.1 Neural Network Based Segmentation

Convolutional Neural Network consists of a set of layers. Each layer contains one or more planes. Input of each unit in a plane is accepted from a small neighborhood in the planes of the previous layer. Shared weight concept is applicable to each plane. Multiple planes in each layer detect multiple features. After detecting the features, the image is passed to a subsampling layer which is used to perform a local averaging of the weights. Shared weights help to reduce the number of parameters of the network.

1.4 Significance

Medical image segmentation, one of the significant aspects of image processing in the medical field, is a long-standing problem in the research area of computer vision and medical field. The main aim of segmentation is to extract the ROI (Region of Interest) for image analysis. The division of an image into meaningful structures, i.e., image segmentation, is often an essential step in image analysis, object representation, visualization, and many other image processing tasks. Segmentation plays an important role in image processing since separation of a large image into several parts makes further processing simpler. These several parts that are rejoined will cover the entire image. Segmentation may also depend on various features like color or texture that are contained in the image. Before denoising an image, it is segmented to recover the original image. The main aim of segmentation is to reduce the information and hence making easy analysis possible. There exist several image segmentation techniques, which partition the image into several parts based on certain image features like pixel intensity value,

color, texture, etc. and these techniques are categorized based on the segmentation method used. A great variety of segmentation methods has been proposed till now and they fall into the following major categories. In this project we are segmenting body parts to further detect cancer tumors so that patient can be cured at prior stages.

Chapter - 2

Project Analysis

2.1 Information Gathering

Requirements analysis is the tasks that an analyst performs to structure and organize requirements, specify and model requirements and designs, validate and verify information, identify solution options that meet business needs, and estimate the potential value that could be realized for a solution option.

The most important phase is to analyze what all are the requirements of the project. During analyzing I found that Segmentation of medical images is the major requirement of this project. For segmentation need to learn Deep learning, Neural network, Convolutional Neural Network and U-Net architecture.

2.1.1 Neural Network

A neural network is a computing model whose layered structure resembles the networked structure of neurons in the brain, with layers of connected nodes. A neural network can learn from data—so it can be trained to recognize patterns, classify data, and forecast future events.

A neural network breaks down the input into layers of abstraction. It can be trained over many examples to recognize patterns in speech or images, just as the human brain does. Its behavior is defined by the way its individual elements are connected and by the strength, or weights, of those connections. These weights are automatically adjusted during training according to a specified learning rule until the neural network performs the desired task correctly.

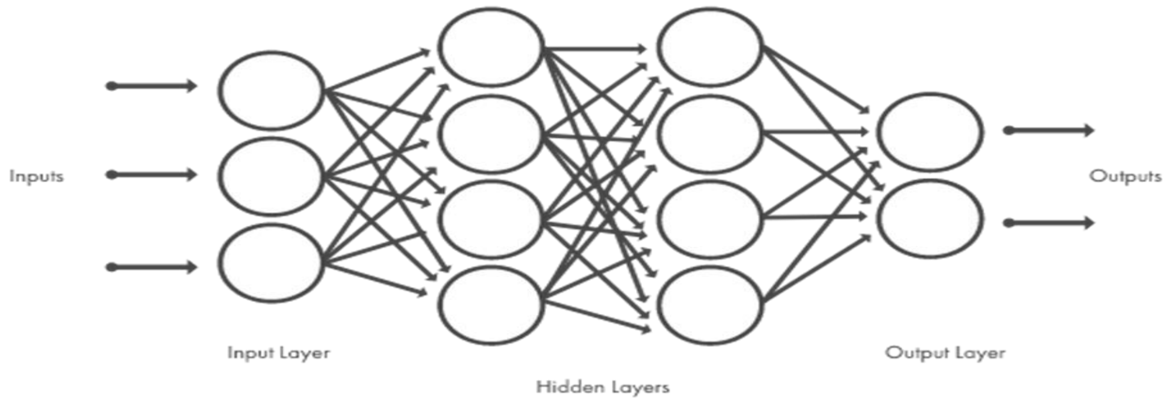


Fig.2.1 Neural Network Architecture

- **Backpropagation In Neural Network**

Single layer neural network, perceptron, consists of three basic elements,

- Connecting Links which are characterised by their weights, Wk0… Wkm
- Linear Combiner which adds the weighted input signals including bias,

$$Vk = \sum 〚W_{ki} * X_i〛 , \quad X_1 .. X_m \text{ are inputs with } X_0 \text{ as bias.}$$

- Activation function which operates on the combiner output and is used to limit the final output.

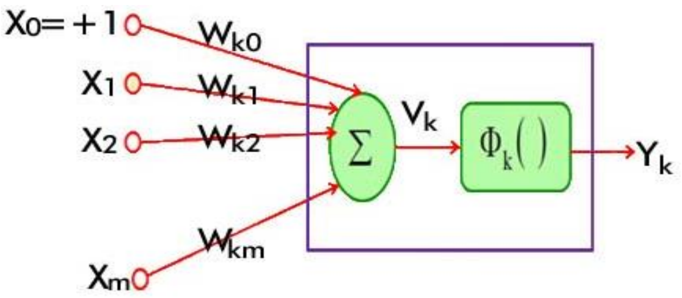


Fig. 2.2 Single Layer NN(Perceptron)

Process by which perceptron learns is called as **Backpropagation**. While training forward propagation predicts class for labelled input data. The error between predicted class and the actual label is backpropagated and the weights are updated to minimize this error. The learning in neural network consists of following steps:

- Computation of error between the prediction and labels
- Updating weights and bias to minimize the error.
- Forward propagation to predict on next input class using the updated weights.

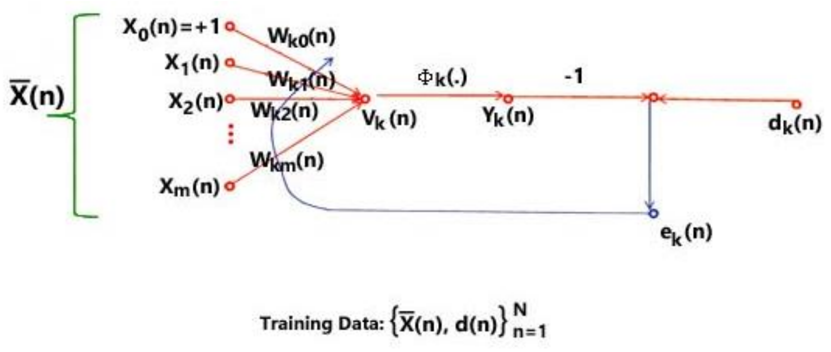


Fig. 2.3 Backpropagation in Neural Network

2.1.2 Convolutional Neural Network

A convolutional neural network (CNN or ConvNet) is one of the most popular algorithms for deep learning, a type of machine learning in which a model learns to perform classification tasks directly from images, video, text, or sound.

CNNs are particularly useful for finding patterns in images to recognize objects, faces, and scenes. They learn directly from image data, using patterns to classify images and eliminating the need for manual feature extraction.

CNN Architecture

Like other neural networks, a CNN is composed of an input layer, an output layer, and many hidden layers in between. These layers perform operations that alter the data with the intent of learning features specific to the data.

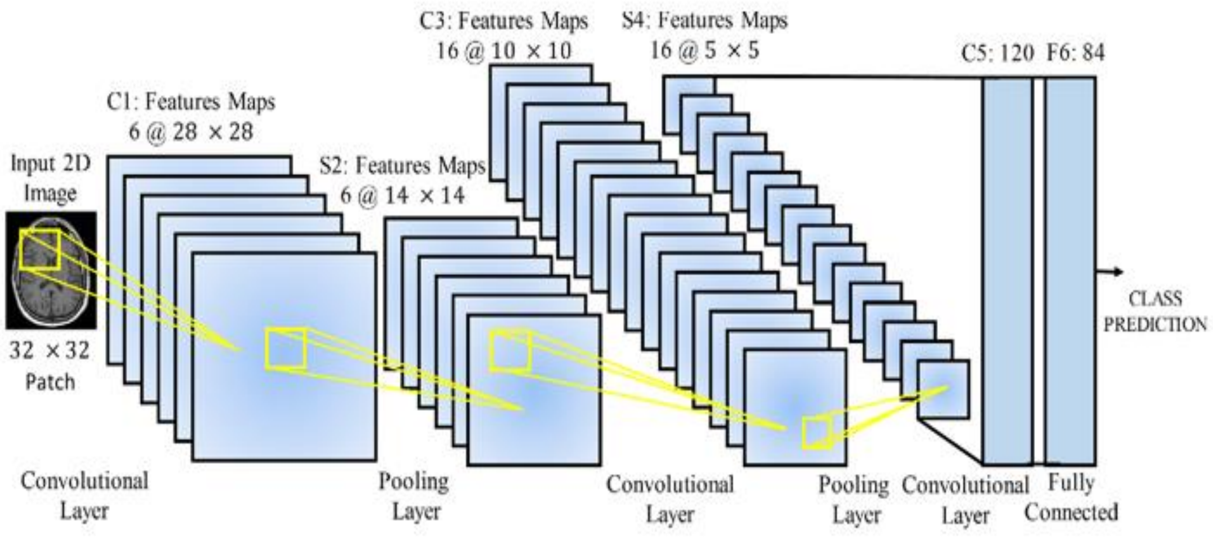


Fig.2.4 CNN Architecture

CNN Layer

- **Convolution layer:**

Convolution puts the input images through a set of convolutional filters, each of which activates certain features from the images. Convolution is the first layer to extract features from an input image. Convolution preserves the relationship between pixels by learning image features using small squares of input data. It is a mathematical operation that takes two inputs such as image matrix and a filter or kernel.

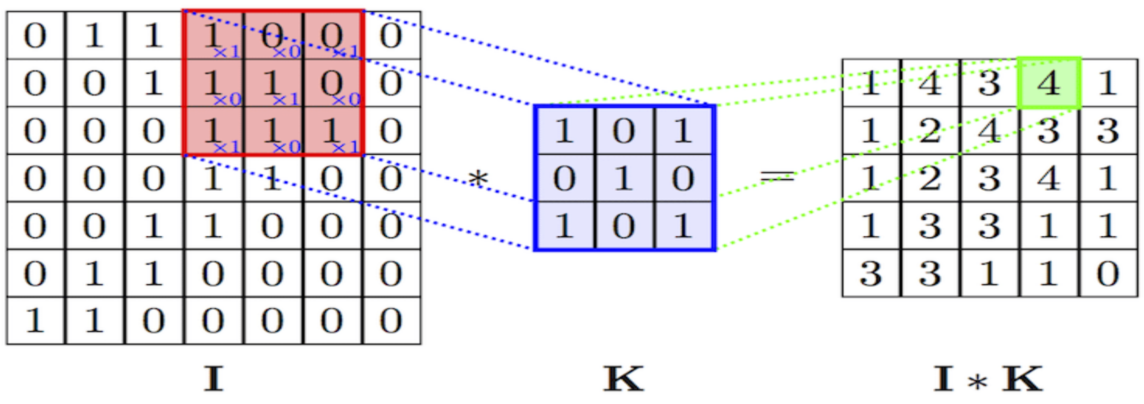


Fig.2.5 Convolutional Layer

- **ReLU layer:**

Rectified linear unit (ReLU) allows for faster and more effective training by mapping negative values to zero and maintaining positive values. This is sometimes referred to as activation, because only the activated features are carried forward into the next layer.

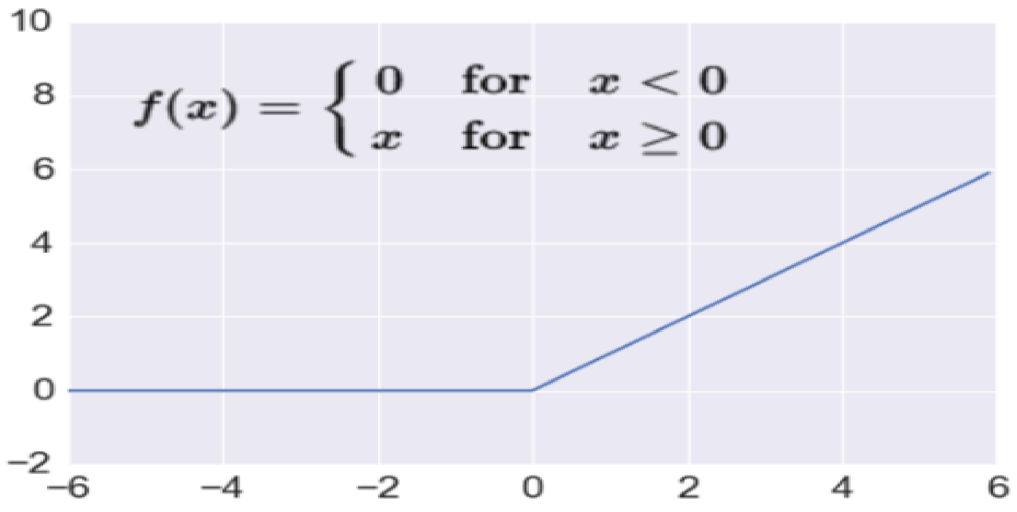


Fig.2.6 ReLU Layer

- **Pooling layer:**

Pooling simplifies the output by performing nonlinear downsampling, reducing the number of parameters that the network needs to learn. Pooling layers section would reduce the number of parameters when the images are too large. Spatial pooling also called subsampling or downsampling which reduces the dimensionality of each map but retains the important information.

- **Fully Connected layer:**

The layer we call as FC layer, we flattened our matrix into vector and feed it into a fully connected layer like neural network.

2.1.3 U-Net Architecture

U-Net is a convolutional neural network that was developed for biomedical image segmentation at the Computer Science Department of the University of Freiburg, Germany. The network is based on the fully convolutional network and its architecture was modified and extended to work with fewer training images and to yield more precise segmentations. Segmentation of a 512 × 512 image takes less than a second on a modern GPU.

The U-Net architecture stems from the so-called "fully convolutional network" first proposed by Long and Shelhamer. The main idea is to supplement a usual contracting network by successive layers, where pooling operations are replaced by up sampling operators. Hence these layers increase the resolution of the output.

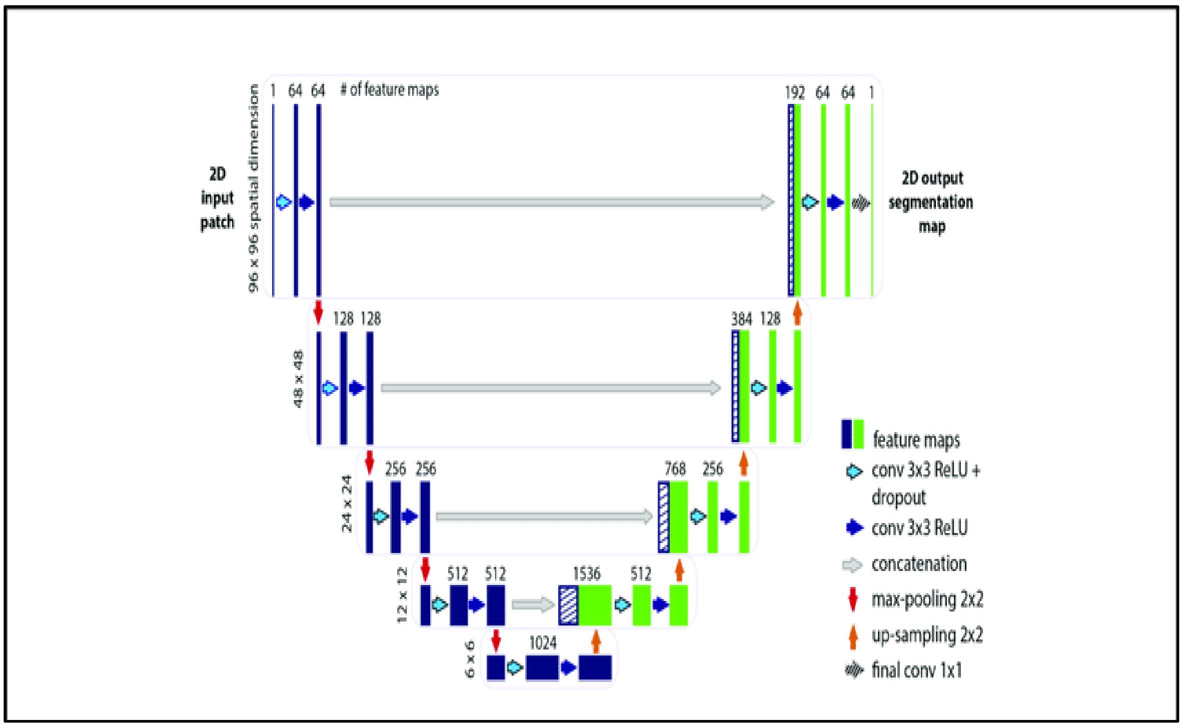


Fig.2.7 U-NET Architecture

2.2 System Feasibility

A feasibility study is an assessment of the practicality of a proposed project or system. A feasibility study aims to objectively and rationally uncover the strengths and weaknesses of an existing business or proposed venture, opportunities and threats present in the natural environment, the resources required to carry through, and ultimately the prospects for success. In its simplest terms, the two criteria to judge feasibility are cost required and value to be attained.

The feasibility of this project is further used in the medical field in the CT Scan images. The code would be further useful in detecting the cancer tumors.

2.2.1 Economical Feasibility

- This code does not have associated cost with it. The cost would only be associated with the CT Scanner for getting the images.

- The system will follow the freeware software standards.
- Bug fixes and maintaining tasks will have an associated cost.

2.2.2 Technical Feasibility

Medical Image Segmentation is a complete deep learning project using CNN and U-NET Architecture. The main technologies and tools that are associated with this project are:

- Python
- Python Libraries
- Dataset (DICOM Images)
- TensorFlow
- GPU Processor

2.2.3 Behavioral Feasibility

Being a project, which is used in the medical field which needs CT Scan images this project will contain significant amount of code lines. The file sizes and the complete project size will not exceed 500MB. For predicting the result, it takes 5ms in the in the CPU computing, if the GPU is used then time would be reduced which suggest that its giving the solution in real time.

From these it’s clear that this project is behavioral feasible.

2.3Experimental Setup

2.3.1 Hardware

- A system that supports latest version of IDE (Anaconda) and other technologies for smooth accessing of the project or a Internet access for using Google Colab.
- Integrated software system to build the project.
- RTX 2080 Ti GPU for training machine learning models.

2.3.2 SOFTWARE

- The project will require application of Machine Learning with various subtopics like convolutional neural network, U-Net architecture.
- Python programming language & libraries like TensorFlow and Keras.
- Anaconda's integrated development environment (IDE) or Google Colab.

GPU Processor	2 X Tesla GK210B
CUDA Cores	2496
Core clock	560MHz
Memory	24 GB

Table 2.1 GPU Characteristics

Chapter-3

System Analysis

3.1 Information Flow Diagram

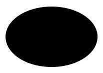
An information flow diagram or data flow diagram is a Unified Modeling Language (UML) representation that illustrates how information/data is being exchanged between system entities within a process. An information flow diagram assists user in understanding the entire flow of data from beginning to end.

3.1.1 Activity Flow Diagram

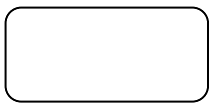
Activity diagram is defined as a UML diagram that focuses on the execution and flow of the behavior of a system instead of implementation. It is also called object-oriented flowchart. Activity diagrams consist of activities that are made up of actions which apply to behavioral modeling technology.

Activity Diagram Notations –

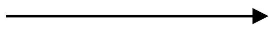
1. **Initial State –** The starting state before an activity takes place is depicted using the initial state.



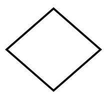
2. **Action or Activity State –** An activity represents execution of an action on objects or by objects. We represent an activity using a rectangle with rounded corners. Basically, any action or event that takes place is represented using an activity.



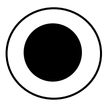
3. **Action Flow or Control flows –** Action flows or Control flows are also referred to as paths and edges. They are used to show the transition from one activity state to another.



4. **Decision node and Branching –** When we need to make a decision before deciding the flow of control, we use the decision node.



5. **Final State or End State** – The state which the system reaches when a particular process or activity ends is known as a Final State or End State.



Below is the representation of Activity diagram my project (Medical Image Segmentation):

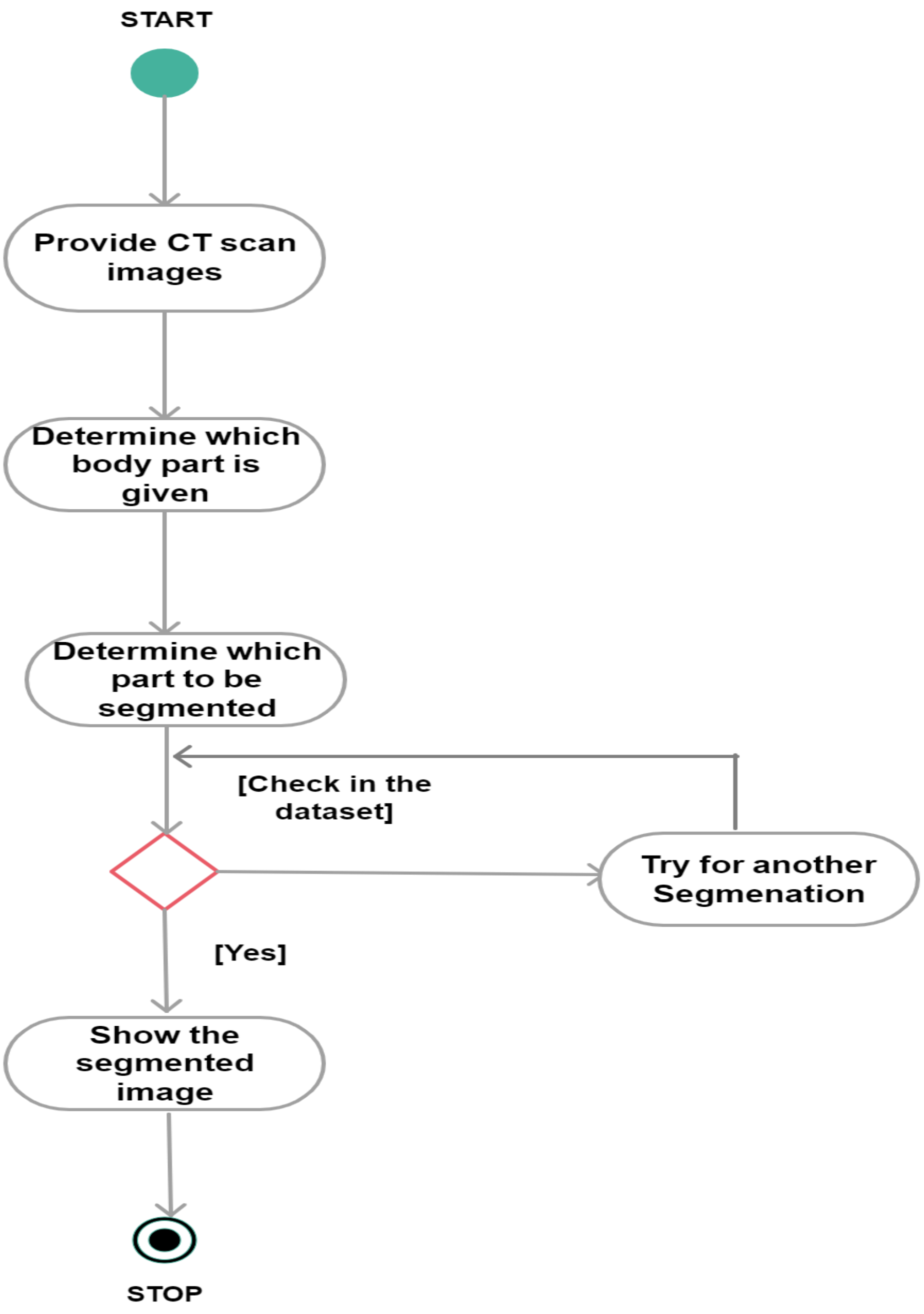

START
Provide CT scan images
Determine which body part is given
Determine which part to be segmented
[Check in the dataset]
Try for another Segmenation
[Yes]
Show the segmented image
STOP

Fig.3.1 Activity Diagram

Chapter-4

Design

4.1 Architectural Design

The software needs the architectural design to represent the design of software. IEEE defines architectural design as “the process of defining a collection of hardware and software components and their interfaces to establish the framework for the development of a computer system.” The software that is built for computer-based systems can exhibit one of these many architectural styles.

The Architecture design used in this project is the Convolutional Neural Network Architecture and the U-Net architecture which consists of some layers: Convolutional layer, ReLu layer, Pooling layer and Fully connected layer. U-Net uses the concept of CNN with the up-sampling part.

U-Net architecture consists of three sections: The contraction, The bottleneck, and the expansion section. The contraction section is made of many contraction blocks.

The first half of the network by minimizing a cost function related to the operation desired and at the second half it would be able to construct an image.

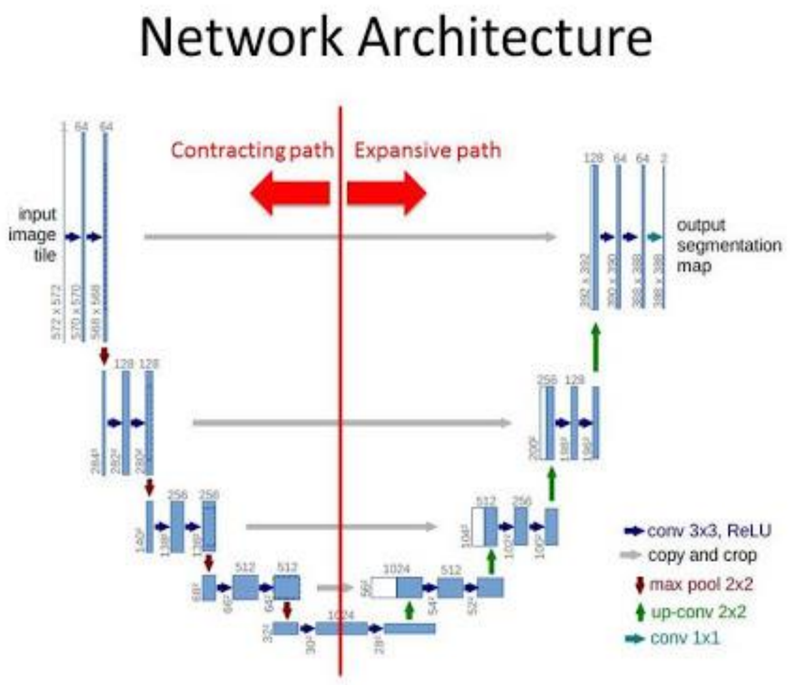
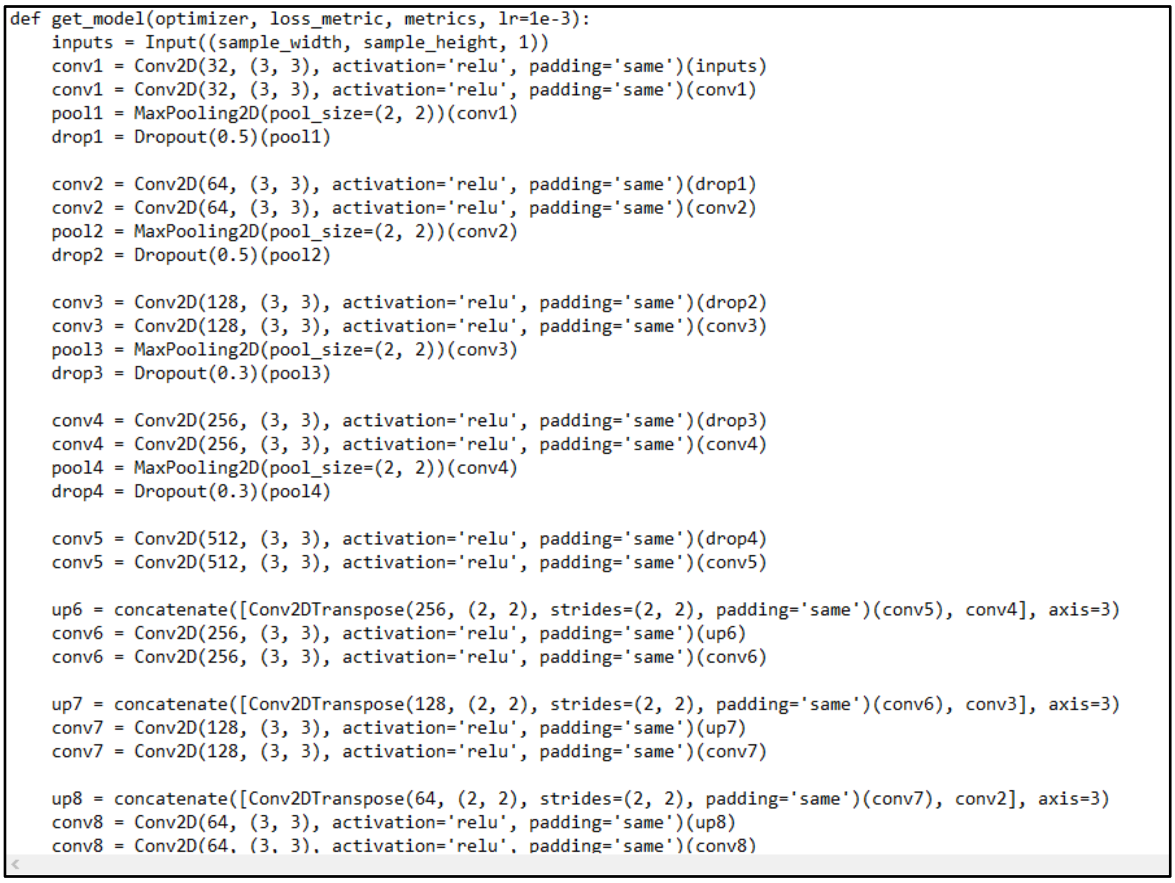


Fig.4.1 U-Net Architecture



```
def get_model(optimizer, loss_metric, metrics, lr=1e-3):
    inputs = Input((sample_width, sample_height, 1))
    conv1 = Conv2D(32, (3, 3), activation='relu', padding='same')(inputs)
    conv1 = Conv2D(32, (3, 3), activation='relu', padding='same')(conv1)
    pool1 = MaxPooling2D(pool_size=(2, 2))(conv1)
    drop1 = Dropout(0.5)(pool1)

    conv2 = Conv2D(64, (3, 3), activation='relu', padding='same')(drop1)
    conv2 = Conv2D(64, (3, 3), activation='relu', padding='same')(conv2)
    pool2 = MaxPooling2D(pool_size=(2, 2))(conv2)
    drop2 = Dropout(0.5)(pool2)

    conv3 = Conv2D(128, (3, 3), activation='relu', padding='same')(drop2)
    conv3 = Conv2D(128, (3, 3), activation='relu', padding='same')(conv3)
    pool3 = MaxPooling2D(pool_size=(2, 2))(conv3)
    drop3 = Dropout(0.3)(pool3)

    conv4 = Conv2D(256, (3, 3), activation='relu', padding='same')(drop3)
    conv4 = Conv2D(256, (3, 3), activation='relu', padding='same')(conv4)
    pool4 = MaxPooling2D(pool_size=(2, 2))(conv4)
    drop4 = Dropout(0.3)(pool4)

    conv5 = Conv2D(512, (3, 3), activation='relu', padding='same')(drop4)
    conv5 = Conv2D(512, (3, 3), activation='relu', padding='same')(conv5)

    up6 = concatenate([Conv2DTranspose(256, (2, 2), strides=(2, 2), padding='same')(conv5), conv4], axis=3)
    conv6 = Conv2D(256, (3, 3), activation='relu', padding='same')(up6)
    conv6 = Conv2D(256, (3, 3), activation='relu', padding='same')(conv6)

    up7 = concatenate([Conv2DTranspose(128, (2, 2), strides=(2, 2), padding='same')(conv6), conv3], axis=3)
    conv7 = Conv2D(128, (3, 3), activation='relu', padding='same')(up7)
    conv7 = Conv2D(128, (3, 3), activation='relu', padding='same')(conv7)

    up8 = concatenate([Conv2DTranspose(64, (2, 2), strides=(2, 2), padding='same')(conv7), conv2], axis=3)
    conv8 = Conv2D(64, (3, 3), activation='relu', padding='same')(up8)
    conv8 = Conv2D(64, (3, 3), activation='relu', padding='same')(conv8)
```

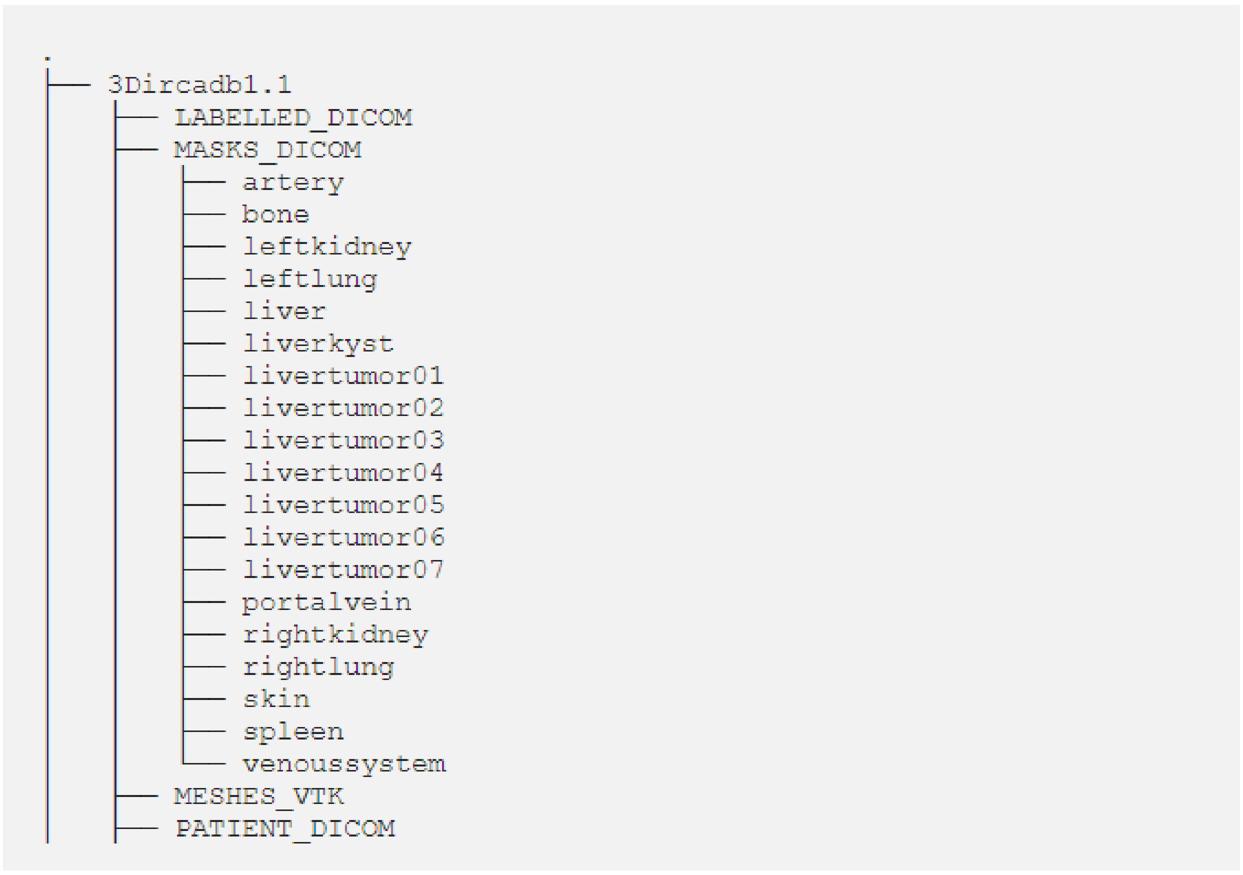
Fig.4.2 Code behind U-Net Architecture

4.2 Data Design

Data design is the first design activity, which results in less complex, modular and efficient program structure. The information domain model developed during analysis phase is transformed into data structures needed for implementing the software. The data objects, attributes, and relationships depicted in entity relationship diagrams and the information stored in data dictionary provide a base for data design activity. During the data design process, data types are specified along with the integrity rules required for the data.

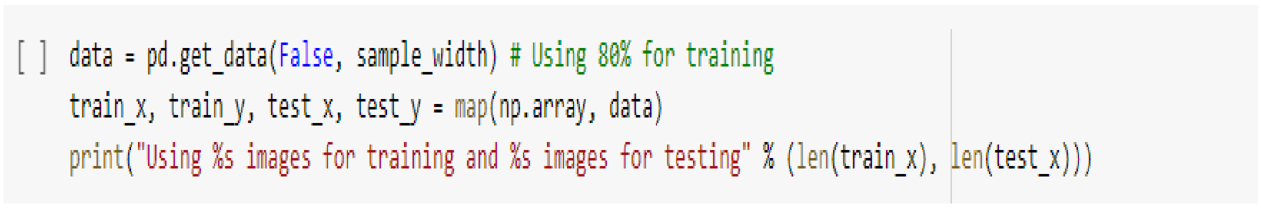
The Data Design of the project further can be analyzed as:

The database is composed of the 3D CT-scans of 10 women and 10 men with hepatic tumors in 75% of cases. The 20 folders correspond to 20 different patients. These folders are called "3D-IRCADb-1-number". Each folder contains 4 sub-folders called "PATIENT_DICOM", "LABELLED_DICOM", "MASKS_DICOM" and "MESHES_VTK". The images are provided by the authors in DICOM and VTK format in 512x512 pixels.



```
.
├── 3Dircadb1.1
│   ├── LABELLED_DICOM
│   ├── MASKS_DICOM
│   │   ├── artery
│   │   ├── bone
│   │   ├── leftkidney
│   │   ├── leftlung
│   │   ├── liver
│   │   ├── liverkyst
│   │   ├── livertumor01
│   │   ├── livertumor02
│   │   ├── livertumor03
│   │   ├── livertumor04
│   │   ├── livertumor05
│   │   ├── livertumor06
│   │   ├── livertumor07
│   │   ├── portalvein
│   │   ├── rightkidney
│   │   ├── rightlung
│   │   ├── skin
│   │   ├── spleen
│   │   └── venoussystem
│   ├── MESHES_VTK
│   ├── PATIENT_DICOM
```

Fig.4.3 Hierarchy of Data set



```
[ ]  data = pd.get_data(False, sample_width) # Using 80% for training
     train_x, train_y, test_x, test_y = map(np.array, data)
     print("Using %s images for training and %s images for testing" % (len(train_x), len(test_x)))
```

Fig.4.4 Code of Preprocessing of data

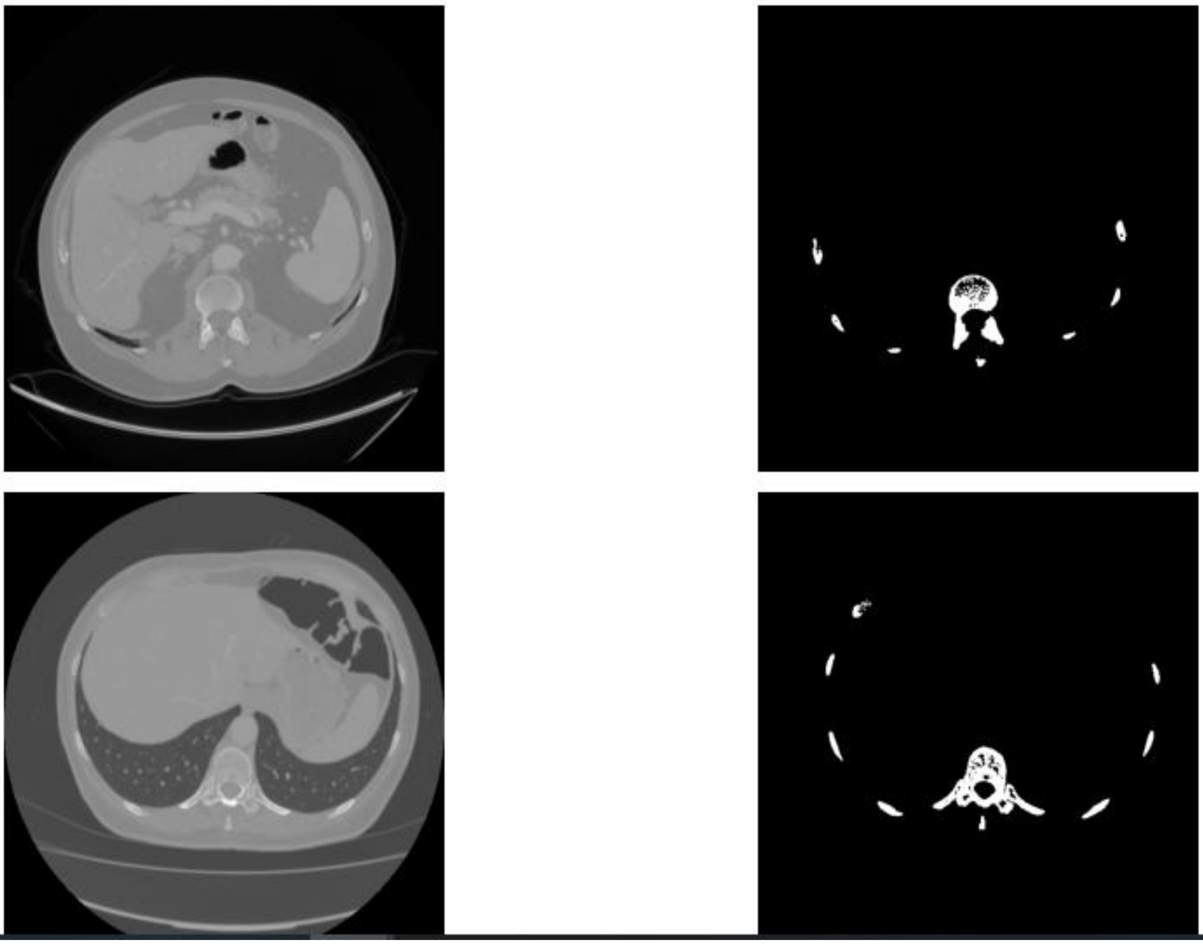


Fig.4.5 Images of the Training Set

(Left one is ground truth image and right is segmented image)

4.2.1 Loss Function Used in the Model Design

In this project, the design or the model defined is based on some rules which is accuracy and loss function. This code has its own loss function which is defined with the Dice Coefficient.

- **Dice Coefficient**

  Dice Coefficient is one of the most commonly used metrics in semantic segmentation. It is the area of overlap between the predicted segmentation and the ground truth divided by the area of union between the predicted segmentation and the ground truth.

  Dice Coefficient is 2 * the Area of Overlap divided by the total number of pixels in both images.

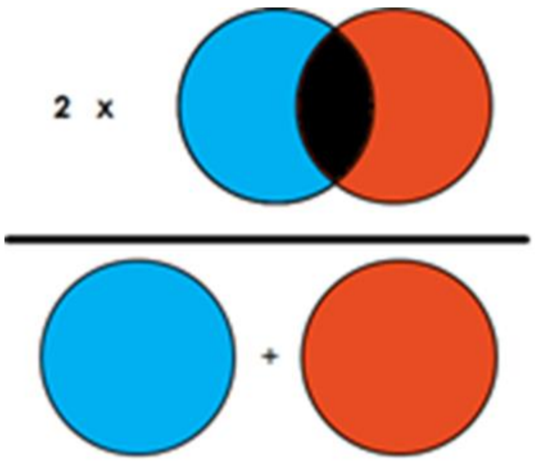
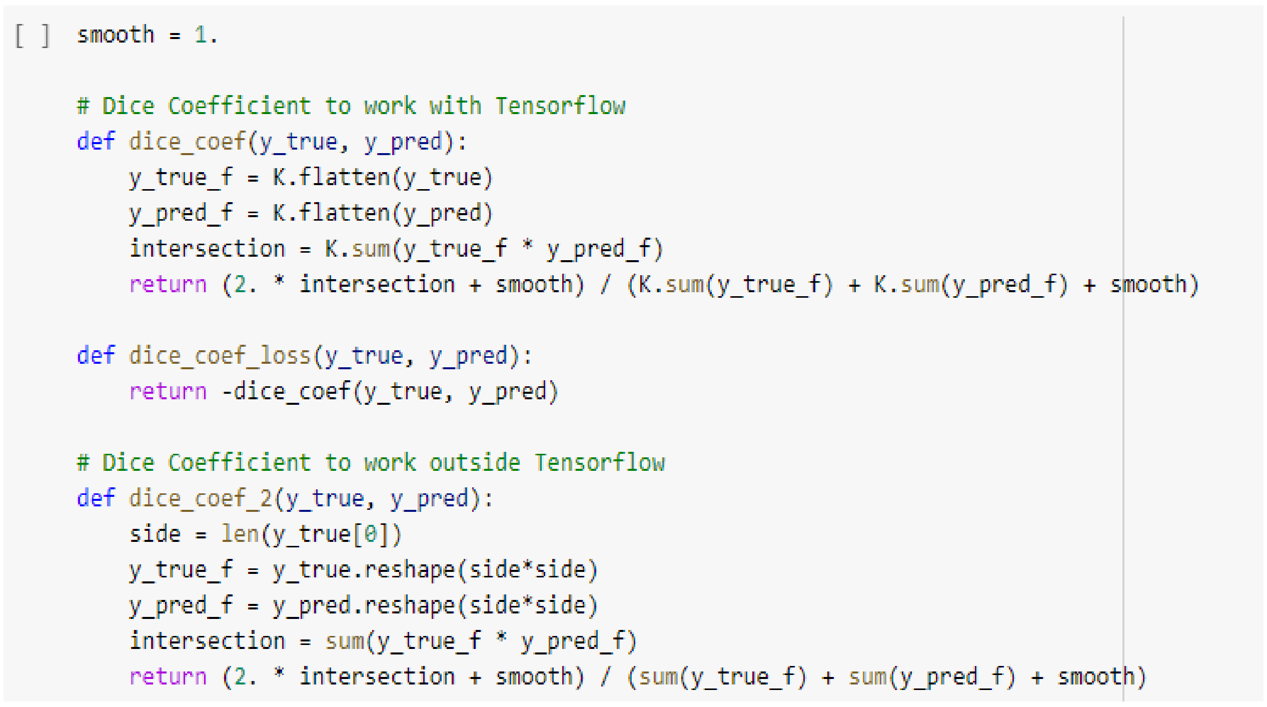


Fig.4.6 Dice coefficient



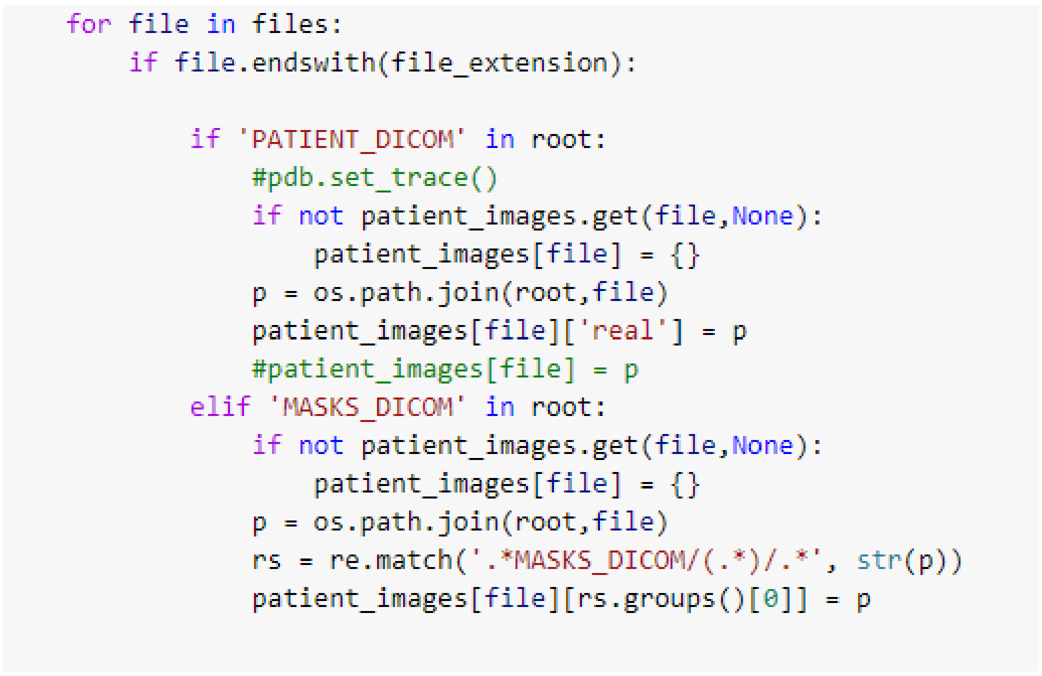
```
[ ]  smooth = 1.

     # Dice Coefficient to work with Tensorflow
     def dice_coef(y_true, y_pred):
         y_true_f = K.flatten(y_true)
         y_pred_f = K.flatten(y_pred)
         intersection = K.sum(y_true_f * y_pred_f)
         return (2. * intersection + smooth) / (K.sum(y_true_f) + K.sum(y_pred_f) + smooth)

     def dice_coef_loss(y_true, y_pred):
         return -dice_coef(y_true, y_pred)

     # Dice Coefficient to work outside Tensorflow
     def dice_coef_2(y_true, y_pred):
         side = len(y_true[0])
         y_true_f = y_true.reshape(side*side)
         y_pred_f = y_pred.reshape(side*side)
         intersection = sum(y_true_f * y_pred_f)
         return (2. * intersection + smooth) / (sum(y_true_f) + sum(y_pred_f) + smooth)
```

Fig.4.7 Code of Loss Function

4.2.2 Data Structures in the Model

Data structures which is used in the project:

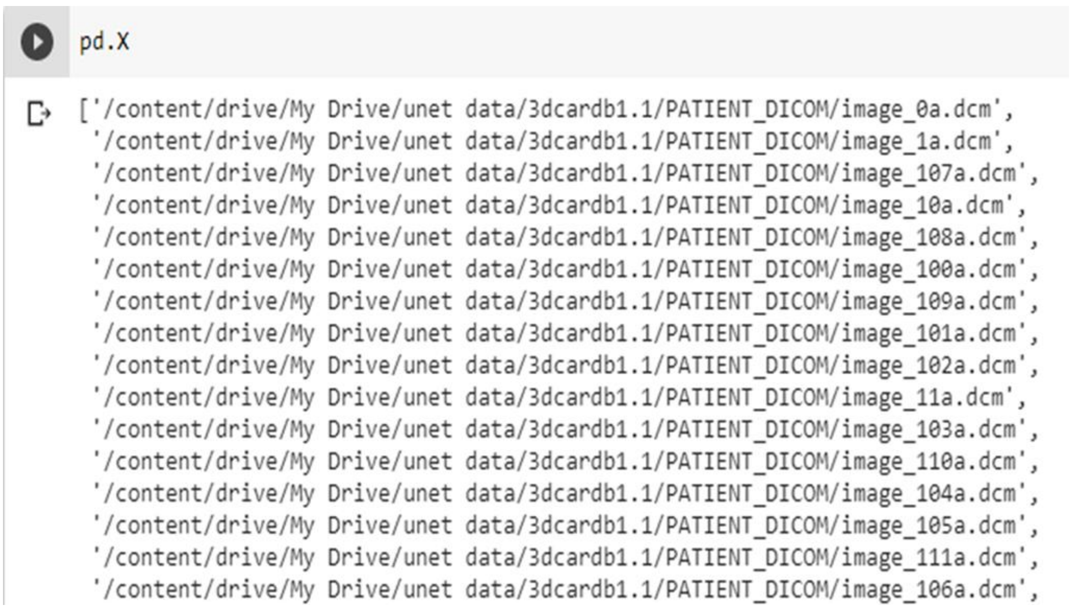
- Data Dictionary: A set of information describing the contents, format, and structure of a database and the relationship between its elements, used to control access to and manipulation of the database.



```
for file in files:
    if file.endswith(file_extension):

        if 'PATIENT_DICOM' in root:
            #pdb.set_trace()
            if not patient_images.get(file,None):
                patient_images[file] = {}
            p = os.path.join(root,file)
            patient_images[file]['real'] = p
            #patient_images[file] = p
        elif 'MASKS_DICOM' in root:
            if not patient_images.get(file,None):
                patient_images[file] = {}
            p = os.path.join(root,file)
            rs = re.match('.*MASKS_DICOM/(.*)/.*', str(p))
            patient_images[file][rs.groups()[0]] = p
```

Fig.4.8 Data Dictionary used in the model design

- List: A sequential set of elements to which you can add new elements and remove or change existing ones.

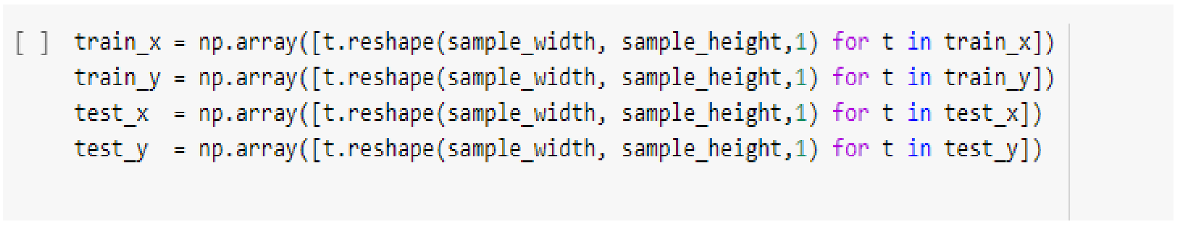


```
pd.X
```

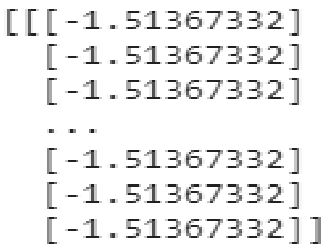
```
['/content/drive/My Drive/unet data/3dcardb1.1/PATIENT_DICOM/image_0a.dcm',
 '/content/drive/My Drive/unet data/3dcardb1.1/PATIENT_DICOM/image_1a.dcm',
 '/content/drive/My Drive/unet data/3dcardb1.1/PATIENT_DICOM/image_107a.dcm',
 '/content/drive/My Drive/unet data/3dcardb1.1/PATIENT_DICOM/image_10a.dcm',
 '/content/drive/My Drive/unet data/3dcardb1.1/PATIENT_DICOM/image_108a.dcm',
 '/content/drive/My Drive/unet data/3dcardb1.1/PATIENT_DICOM/image_100a.dcm',
 '/content/drive/My Drive/unet data/3dcardb1.1/PATIENT_DICOM/image_109a.dcm',
 '/content/drive/My Drive/unet data/3dcardb1.1/PATIENT_DICOM/image_101a.dcm',
 '/content/drive/My Drive/unet data/3dcardb1.1/PATIENT_DICOM/image_102a.dcm',
 '/content/drive/My Drive/unet data/3dcardb1.1/PATIENT_DICOM/image_11a.dcm',
 '/content/drive/My Drive/unet data/3dcardb1.1/PATIENT_DICOM/image_103a.dcm',
 '/content/drive/My Drive/unet data/3dcardb1.1/PATIENT_DICOM/image_110a.dcm',
 '/content/drive/My Drive/unet data/3dcardb1.1/PATIENT_DICOM/image_104a.dcm',
 '/content/drive/My Drive/unet data/3dcardb1.1/PATIENT_DICOM/image_105a.dcm',
 '/content/drive/My Drive/unet data/3dcardb1.1/PATIENT_DICOM/image_111a.dcm',
 '/content/drive/My Drive/unet data/3dcardb1.1/PATIENT_DICOM/image_106a.dcm',
```

Fig.4.9 List Data Structure Output

- Numpy array : Array in Numpy is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers.



```
[ ]  train_x = np.array([t.reshape(sample_width, sample_height,1) for t in train_x])
     train_y = np.array([t.reshape(sample_width, sample_height,1) for t in train_y])
     test_x  = np.array([t.reshape(sample_width, sample_height,1) for t in test_x])
     test_y  = np.array([t.reshape(sample_width, sample_height,1) for t in test_y])
```



```
[[[-1.51367332]
  [-1.51367332]
  [-1.51367332]
  ...
  [-1.51367332]
  [-1.51367332]
  [-1.51367332]]
```

Fig.4.10 Numpy Array and its output

Chapter-5

Testing and Results

A crucial part of any software development lifecycle is testing. This involves carrying out certain procedures and operations to understand the limitations of the software. It is evident that with testing the constraints of the application that particular bugs and errors are picked up and documented through test cases. This will improve the overall standard and quality of the chatbot and enhance the user experience.

Testing is the major quality control that can be used during software development. Its basic function is to detect the errors in the software. During requirement analysis and design, the output is a document that is usually textual and non-executable. After the coding phase, computer program is available that can be executed for testing purposes. This implies that testing not only has to uncover errors introduced during coding, but also errors introduced during previous phases. Thus the goal of the testing is to uncover requirement, design and coding errors in the program.

5.1 Testing Objective

- Testing is a process of executing a program with the intent of finding an error
- A good test case is one that has a high probability of finding an undiscovered error
- A successful test is one that uncovers an as-yet undiscovered error.

5.2 Testing Methods Used

Software Testing Strategies

A strategy for software testing integrates software test case design methods into a well-planned series of steps that result in the successful construction of software. As important, a software testing strategy provides a road map. Testing is a set of activities that can be planned and conducted systematically.

Various strategies are given below:

- Unit Testing
- Integration Testing
- Validation Testing
- User Acceptance Testing
- System Testing

Unit Testing

Unit testing focuses verification efforts on the smallest unit of software design of module. This is also known as “Module Testing”. Acceptance of package is used for computerization of module. Machine Utilization was prepared and approved by the project leader.

In this testing step, each module is found to be working satisfactory as regards to the expected output from the module. The suggested changes were incorporated into the system. Here each module in the Machine Utilization has been tested.

Integration Testing

After the package is integrated, the user test version of the software was released. This testing consists of testing with live data and various stress tests and result were noted down. Then the corrections were made based on the user’s feedback. Integration testing is systematic testing for constructing the program structure, while at the same time conducting tests to uncover errors associated within the interface. The objective is to take unit tested modules and build a program structure. All the modules are combined and tested as a whole. Here correction is difficult because the vast expenses of the entire program complicate the isolation of causes. Thus the integration testing step, all the errors uncovered are corrected for the next steps.

Validation Testing

At the culmination of integration testing, software is completely assembled as a package; interfacing errors have been uncovered and corrected, and a final series of software tests - Validation testing - may begin.

User Acceptance Testing

User acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective system users at time of development and making changes wherever required.

After performing all the tests, the system was found to be running successfully according to the user requirements i.e., (constraints).

System Testing

Software is only one element of a larger computer-based system. Ultimately, software is incorporated with other system elements and a series of system integration and validation tests are conducted.

5.3 Test Cases

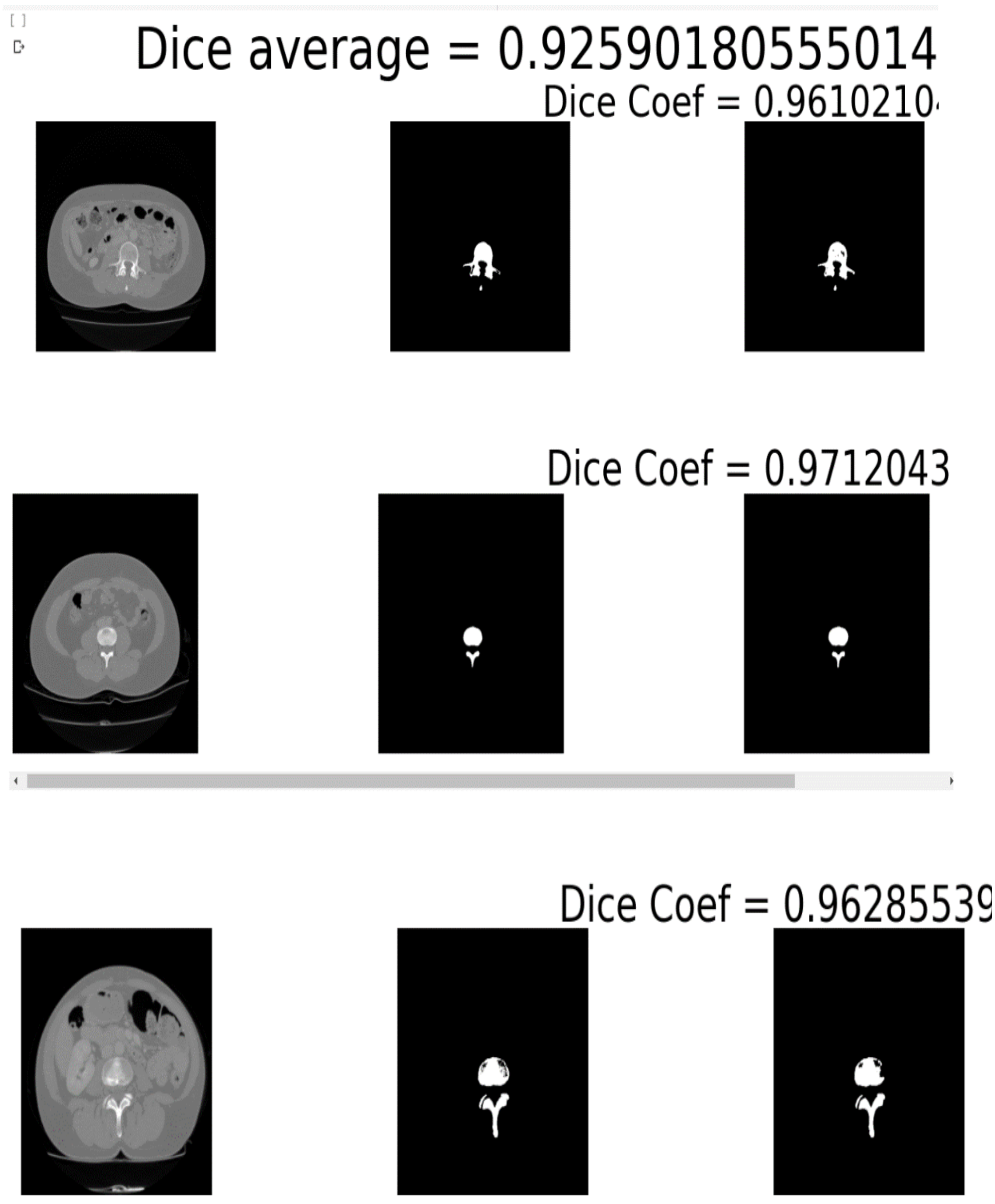
Based on the error and loss function used in the model test cases have been defined. The project was tested to measure the efficiency of the code which can clearly find out that which predicted image is accurate and which has some errors. The loss function defined in the code is Dice coefficient. It is one of the most commonly used metrics in semantic segmentation. It is the area of overlap between the predicted segmentation and the ground truth divided by the area of union between the predicted segmentation and the ground truth.

$$\frac{2 * |X \cap Y|}{|X| + |Y|}$$

Fig.5.1 Formula for Dice Coefficient

5.4 Sample Test Data and Results

The dataset in the code is divided into Training and Testing part. 80% of the data is used as the Training set and remaining 20% data is in the test set. The training is done with 100 epochs with a batch size of 16.



```
Dice average = 0.92590180555014
Dice Coef = 0.9610210
Dice Coef = 0.9712043
Dice Coef = 0.96285539
```

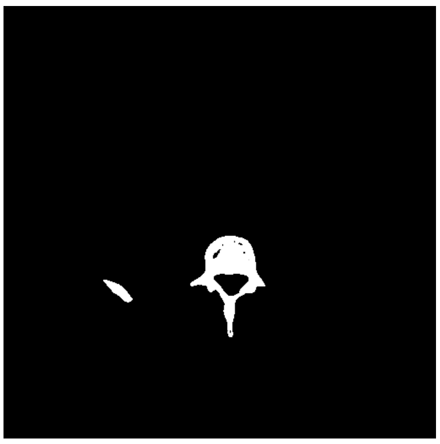
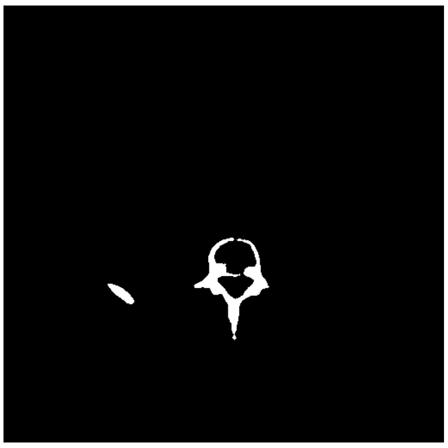
Fig.5.2 Segmented result with accuracy

(The leftmost column is the CT Scan image. The middle column is the real segmentation

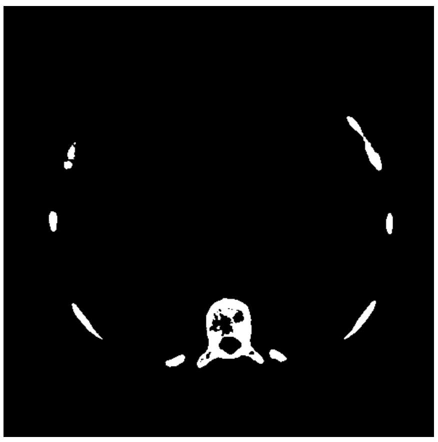
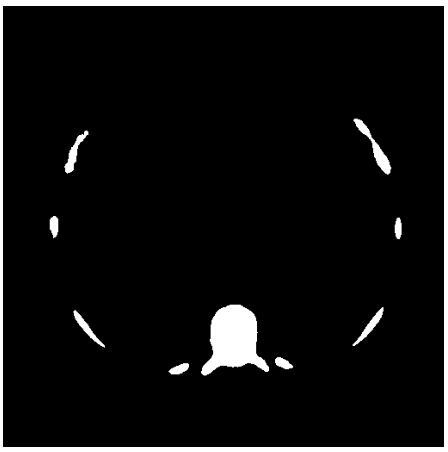
and the rightmost column is the segmentation generated by our 10-layer U-Net.)

5.4.1 Variation of Some Images with the Actual

Dice Coefficient: 0.776348956

Dice Coefficient: 0.8603815214

Dice Coefficient: 0.890761564

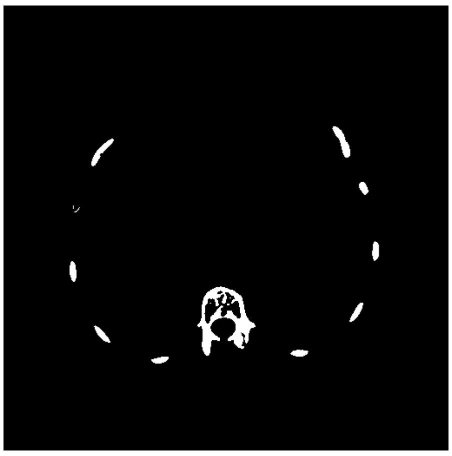
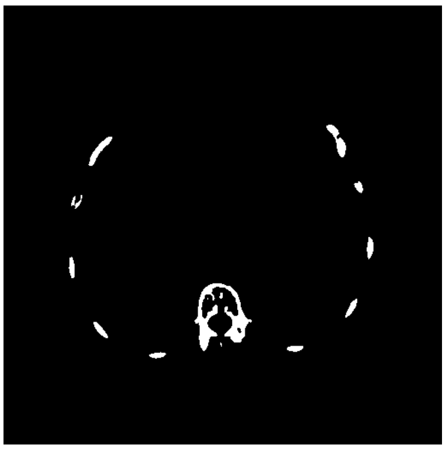



Fig.5.3 Variation of segmented image

The left image is the predicted image and the right image is the actual image

5.4.2 Selection of Epoch and Batch Size

The number of epochs is a hyperparameter that defines the number times that the learning algorithm will work through the entire training dataset.

While training this code I have selected number of epochs as 100 and batch size as 16. There is not as such hard and fast rule on selecting no. of epoch. Epoch selection is done by analyzing the result of the model. We need to analyze at which point of time the relation between the error and epochs gets constant or in other words when our model shows the saturation point.

A training dataset can be divided into one or more batches. When all training samples are used to create one batch, the learning algorithm is called batch gradient descent. When the batch is the size of one sample, the learning algorithm is called stochastic gradient descent. When the batch size is more than one sample and less than the size of the training dataset, the learning algorithm is called mini-batch gradient descent.

- **Batch Gradient Descent**. Batch Size = Size of Training Set
- **Stochastic Gradient Descent**. Batch Size = 1
- **Mini-Batch Gradient Descent**. 1 < Batch Size < Size of Training Set

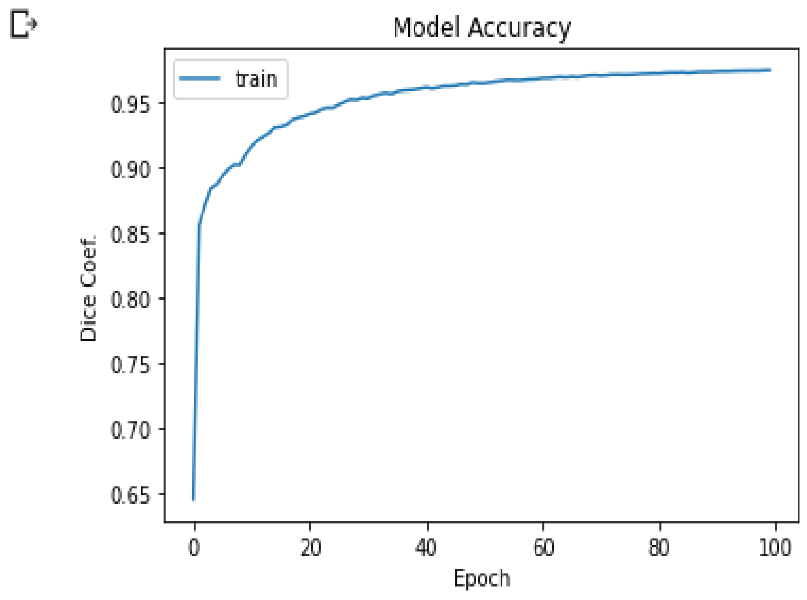


Fig.5.4 Trained Model Accuracy Graph

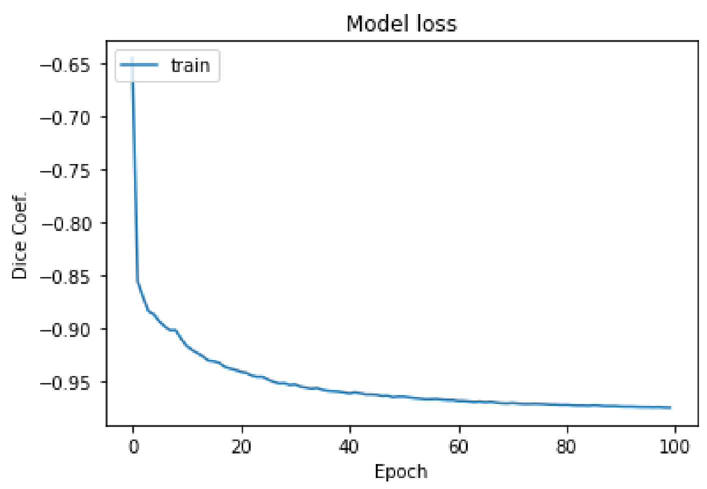


Fig.5.5 Trained Model Loss Graph

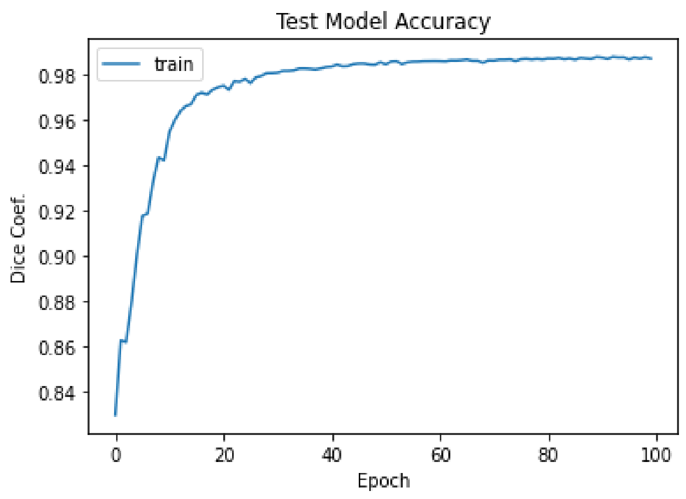


Fig.5.6 Test Model Accuracy Graph

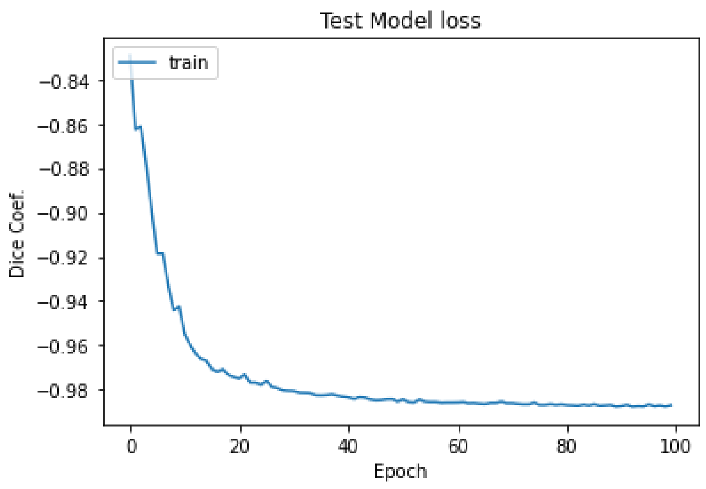


Fig.5.7 Test Model Loss Graph

5.5 Improving Accuracy and Results

Many challenges have occurred while creating this project with deep learning algorithm. These challenges may include:

Overfitting

A statistical model is said to be overfitted, when we train it with a lot of data. When a model gets trained with so much of data, it starts learning from the noise and inaccurate data entries in our data set. Then the model does not categorize the data correctly, because of too much of details and noise.

To overcome this problem, I have used the concept called "**Dropout**" during the training process. Simply put, dropout refers to ignoring units (i.e. neurons) during the training phase of certain set of neurons which is chosen at random. More technically, At each training stage, individual nodes are either dropped out of the net with probability *1-p* or kept with probability *p*, so that a reduced network is left; incoming and outgoing edges to a dropped-out node are also removed.

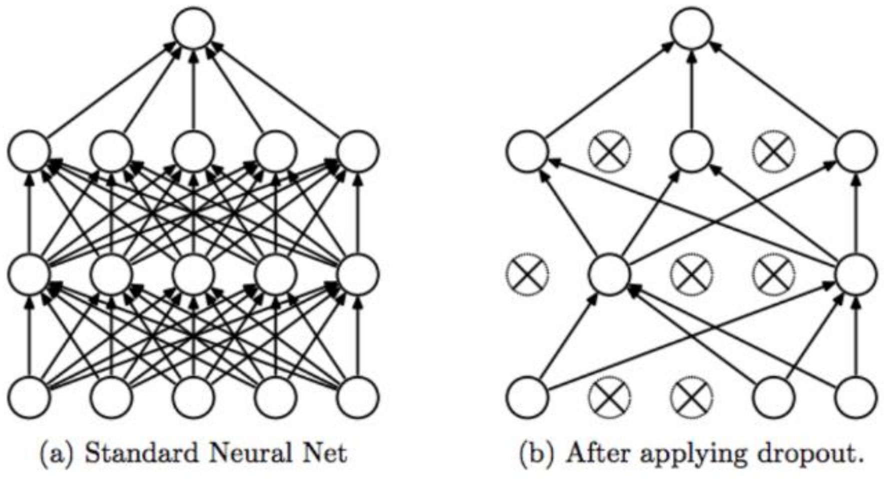


Fig.5.8 Reducing overfitting by Dropout

Training Time

Reducing the training time and having faster convergence is a core topic of many studies. It is the time taken by model to train itself for a dataset.

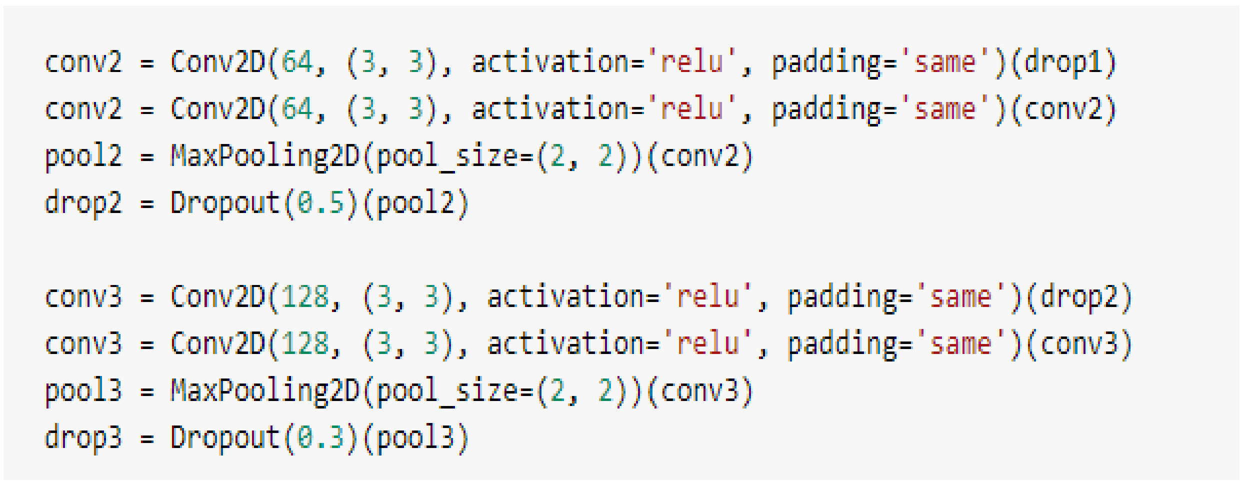
To overcome this challenge, I have used **Pooling layers** in the Convolutional Neural Network. Its function is to progressively reduce the spatial size of the representation to reduce the number of parameters and computation in the network. Pooling layer operates on each feature map independently.

Another method I have used to reduce the training time is **Batch Normalization.** It is a technique for training very deep neural networks that standardizes the inputs to a layer for each mini batch. This has the effect of stabilizing the learning process and dramatically reducing the number of training epochs required to train deep networks.

Gradient Vanishing

Deeper networks are proven to have better performance, yet they are struggling with the issue of exploding or completely vanishing of propagated signal (gradient), in other words, the final loss cannot be effectively back propagated to shallow layers.

This issue in this project is solved by applying **Rectified Linear units**. ReLU is the most commonly used activation function in neural networks, especially in CNNs. ReLU is the max function(x,0) with input x e.g. matrix from a convolved image. ReLU then sets all negative values in the matrix x to zero and all other values are kept constant. ReLU is computed after the convolution and therefore a nonlinear activation function like tanh or sigmoid.



```
conv2 = Conv2D(64, (3, 3), activation='relu', padding='same')(drop1)
conv2 = Conv2D(64, (3, 3), activation='relu', padding='same')(conv2)
pool2 = MaxPooling2D(pool_size=(2, 2))(conv2)
drop2 = Dropout(0.5)(pool2)

conv3 = Conv2D(128, (3, 3), activation='relu', padding='same')(drop2)
conv3 = Conv2D(128, (3, 3), activation='relu', padding='same')(conv3)
pool3 = MaxPooling2D(pool_size=(2, 2))(conv3)
drop3 = Dropout(0.3)(pool3)
```

Fig.5.9 Code of applying Dropout, Pooling and ReLU in the model

Organ Appearance

The heterogeneous appearance of the target organ is one of the big challenges in medical image segmentation. The target organ or lesion may vary hugely in size, shape, and location from patient to patient.

To overcome this challenge, I have researched various datasets available online. I found many corrupted data where I found poor test results. I also analyzed various site's data like MNIST dataset, Kaggle's dataset, etc. After searching lot, I found the good quality data and that can be verified in my project through test results.

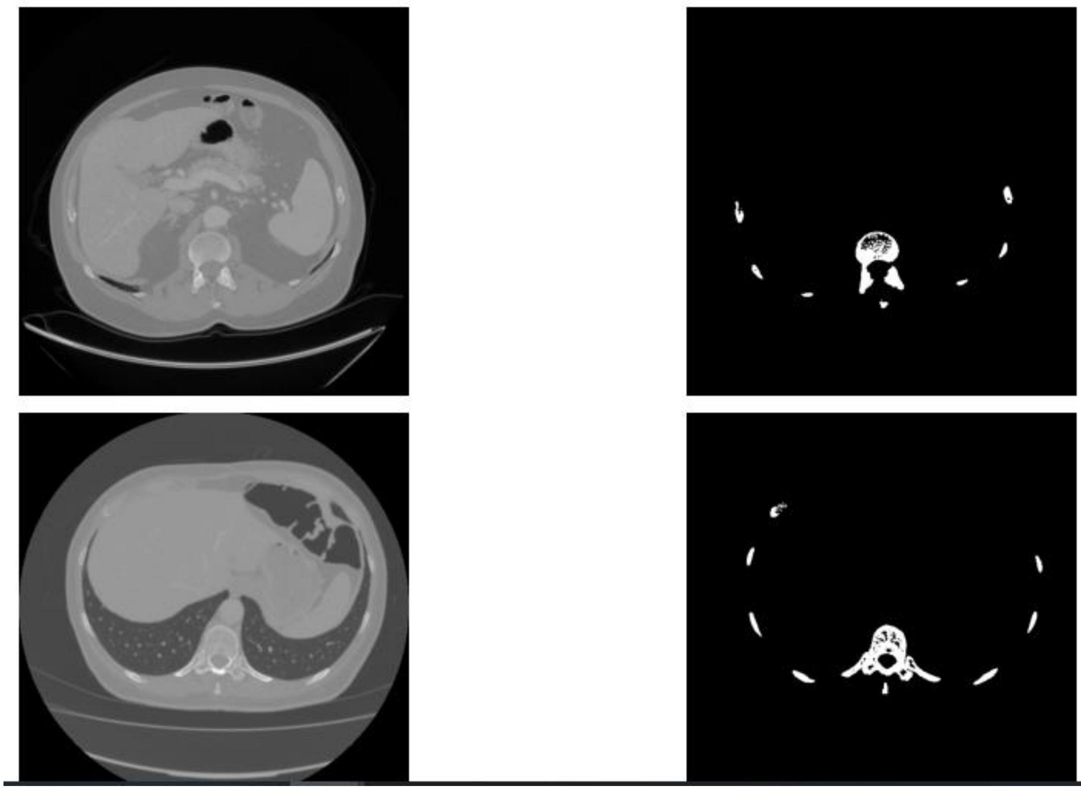


Fig.5.10 Dataset in project

Overcoming all the difficulties above defined, this model is built with great efficiency and it would show result with correct output.

5.5.1 Transfer Learning

To improve the accuracy of the model another technique called Transfer Learning has been applied. Transfer learning make use of the knowledge gained while solving one problem and applying it to a different but related problem. For example, knowledge gained while learning to recognize cars can be used to some extent to recognize trucks.

With transfer learning, instead of starting the learning process from scratch, you start from patterns that have been learned when solving a different problem. This way you leverage previous learnings and avoid starting from scratch.

Pre-Training

When we train the network on a large dataset (for example: ImageNet) , we train all the parameters of the neural network and therefore the model is learned. It may take hours on your GPU.

Fine Tuning

We can give the new dataset to fine tune the pre-trained CNN. Consider that the new dataset is almost similar to the original dataset used for pre-training. Since the new dataset is similar, the same weights can be used for extracting the features from the new dataset.

1. If the new dataset is very small, it's better to train only the final layers of the network to avoid overfitting, keeping all other layers fixed. So remove the final layers of the pre-trained network. Add new layers. Retrain only the new layers.
2. If the new dataset is very much large, retrain the whole network with initial weights from the pretrained model.

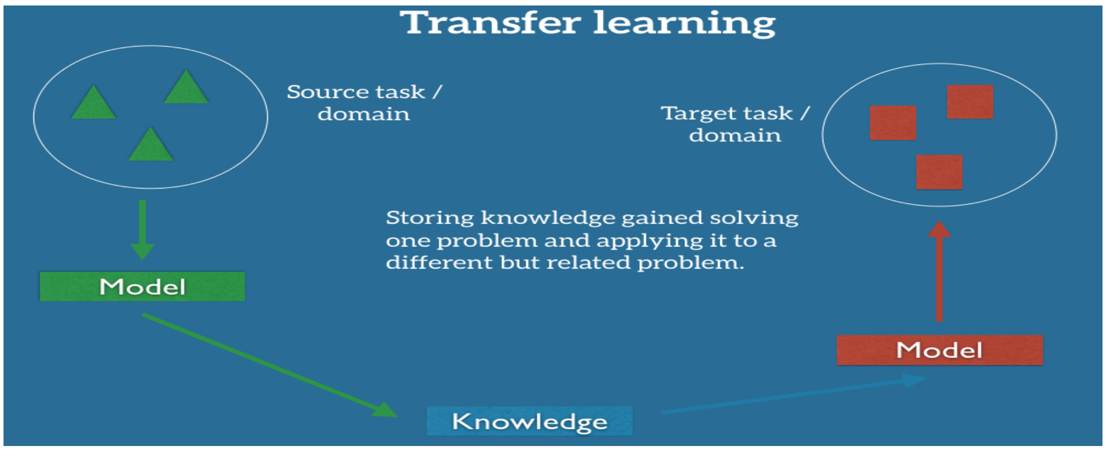
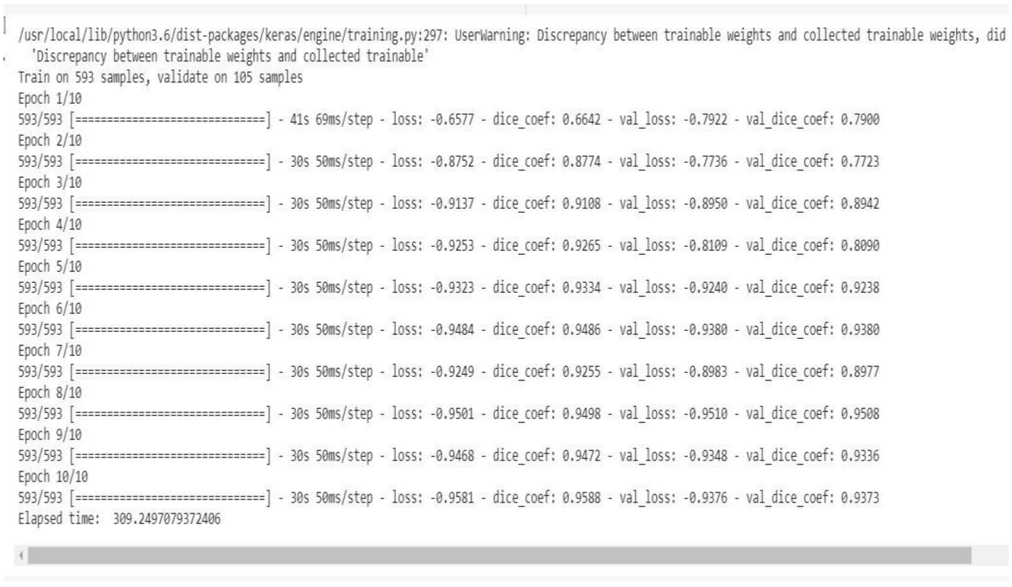

Transfer learning
Source task / domain
Target task / domain
Storing knowledge gained solving one problem and applying it to a different but related problem.
Model
Model
Knowledge

Fig.5.11 Transfer Learning

In this project Transfer Learning is being used to analyze the accuracy of model trained from scratch versus accuracy of the model trained using transfer learning. For this I have used the model previously trained on ‘bones’ data and used it to another model to predict ‘liver’ data.

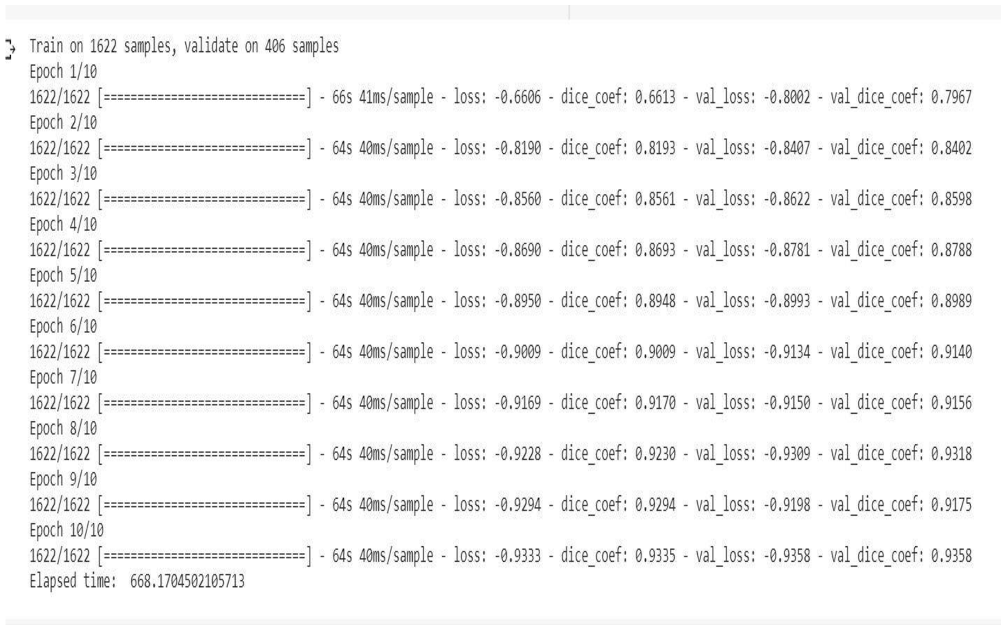
The comparison between these two models can be done based on these results:

1. On the basis of Model Training



```
/usr/local/lib/python3.6/dist-packages/keras/engine/training.py:297: UserWarning: Discrepancy between trainable weights and collected trainable weights, did
  'Discrepancy between trainable weights and collected trainable'
Train on 593 samples, validate on 105 samples
Epoch 1/10
593/593 [==============================] - 41s 69ms/step - loss: -0.6577 - dice_coef: 0.6642 - val_loss: -0.7922 - val_dice_coef: 0.7900
Epoch 2/10
593/593 [==============================] - 30s 50ms/step - loss: -0.8752 - dice_coef: 0.8774 - val_loss: -0.7736 - val_dice_coef: 0.7723
Epoch 3/10
593/593 [==============================] - 30s 50ms/step - loss: -0.9137 - dice_coef: 0.9108 - val_loss: -0.8950 - val_dice_coef: 0.8942
Epoch 4/10
593/593 [==============================] - 30s 50ms/step - loss: -0.9253 - dice_coef: 0.9265 - val_loss: -0.8109 - val_dice_coef: 0.8090
Epoch 5/10
593/593 [==============================] - 30s 50ms/step - loss: -0.9323 - dice_coef: 0.9334 - val_loss: -0.9240 - val_dice_coef: 0.9238
Epoch 6/10
593/593 [==============================] - 30s 50ms/step - loss: -0.9484 - dice_coef: 0.9486 - val_loss: -0.9380 - val_dice_coef: 0.9380
Epoch 7/10
593/593 [==============================] - 30s 50ms/step - loss: -0.9249 - dice_coef: 0.9255 - val_loss: -0.8983 - val_dice_coef: 0.8977
Epoch 8/10
593/593 [==============================] - 30s 50ms/step - loss: -0.9501 - dice_coef: 0.9498 - val_loss: -0.9510 - val_dice_coef: 0.9508
Epoch 9/10
593/593 [==============================] - 30s 50ms/step - loss: -0.9468 - dice_coef: 0.9472 - val_loss: -0.9348 - val_dice_coef: 0.9336
Epoch 10/10
593/593 [==============================] - 30s 50ms/step - loss: -0.9581 - dice_coef: 0.9588 - val_loss: -0.9376 - val_dice_coef: 0.9373
Elapsed time:  309.2497079372406
```

Fig.5.12 Epoch of Transfer Learning Model



```
Train on 1622 samples, validate on 406 samples
Epoch 1/10
1622/1622 [==============================] - 66s 41ms/sample - loss: -0.6606 - dice_coef: 0.6613 - val_loss: -0.8002 - val_dice_coef: 0.7967
Epoch 2/10
1622/1622 [==============================] - 64s 40ms/sample - loss: -0.8190 - dice_coef: 0.8193 - val_loss: -0.8407 - val_dice_coef: 0.8402
Epoch 3/10
1622/1622 [==============================] - 64s 40ms/sample - loss: -0.8560 - dice_coef: 0.8561 - val_loss: -0.8622 - val_dice_coef: 0.8598
Epoch 4/10
1622/1622 [==============================] - 64s 40ms/sample - loss: -0.8690 - dice_coef: 0.8693 - val_loss: -0.8781 - val_dice_coef: 0.8788
Epoch 5/10
1622/1622 [==============================] - 64s 40ms/sample - loss: -0.8950 - dice_coef: 0.8948 - val_loss: -0.8993 - val_dice_coef: 0.8989
Epoch 6/10
1622/1622 [==============================] - 64s 40ms/sample - loss: -0.9009 - dice_coef: 0.9009 - val_loss: -0.9134 - val_dice_coef: 0.9140
Epoch 7/10
1622/1622 [==============================] - 64s 40ms/sample - loss: -0.9169 - dice_coef: 0.9170 - val_loss: -0.9150 - val_dice_coef: 0.9156
Epoch 8/10
1622/1622 [==============================] - 64s 40ms/sample - loss: -0.9228 - dice_coef: 0.9230 - val_loss: -0.9309 - val_dice_coef: 0.9318
Epoch 9/10
1622/1622 [==============================] - 64s 40ms/sample - loss: -0.9294 - dice_coef: 0.9294 - val_loss: -0.9198 - val_dice_coef: 0.9175
Epoch 10/10
1622/1622 [==============================] - 64s 40ms/sample - loss: -0.9333 - dice_coef: 0.9335 - val_loss: -0.9358 - val_dice_coef: 0.9358
Elapsed time:  668.1704502105713
```

Fig.5.13 Epoch of Model Trained from scratch

2. On the basis of Accuracy

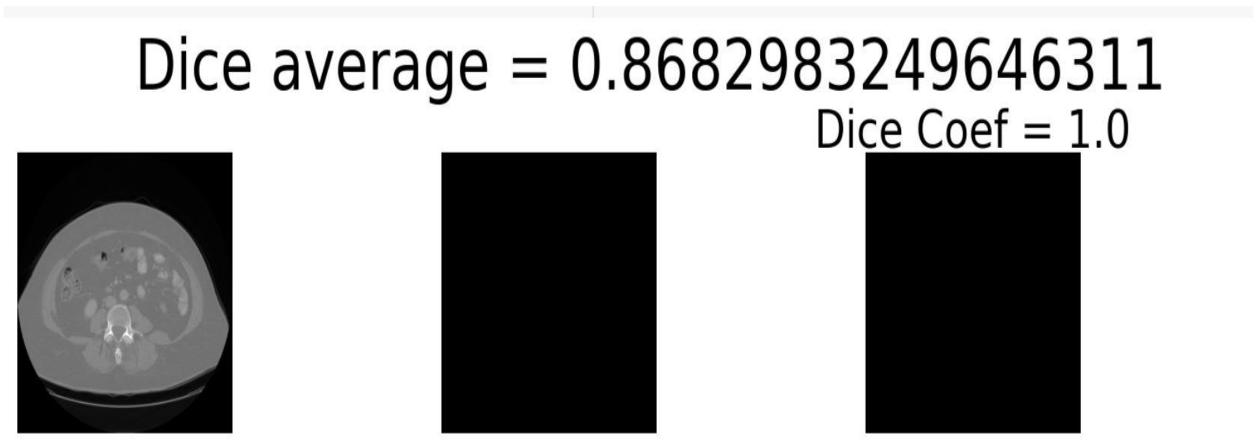


Fig.5.14 Average Dice Coefficient of Transfer Learning Model

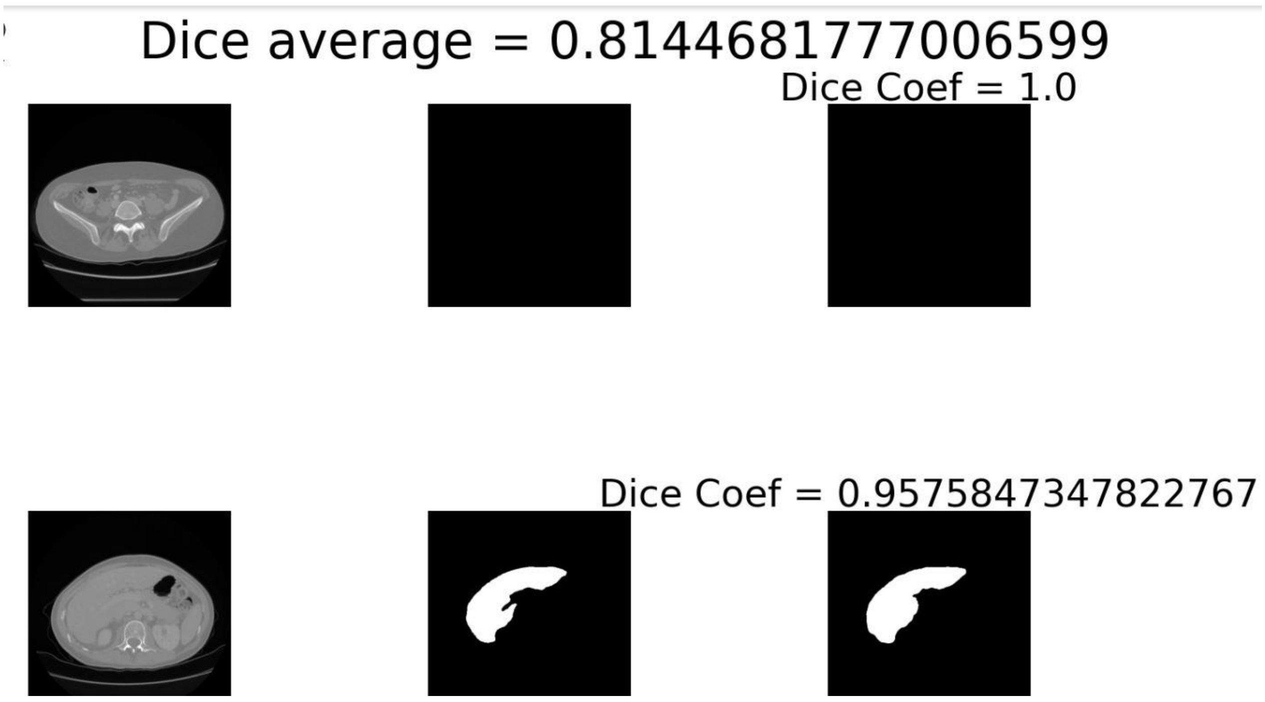

Dice average = 0.8144681777006599
Dice Coef = 1.0
Dice Coef = 0.9575847347822767

Fig.5.15 Average Dice Coefficient of Model trained from scratch

3. On the basis of graph of the Model

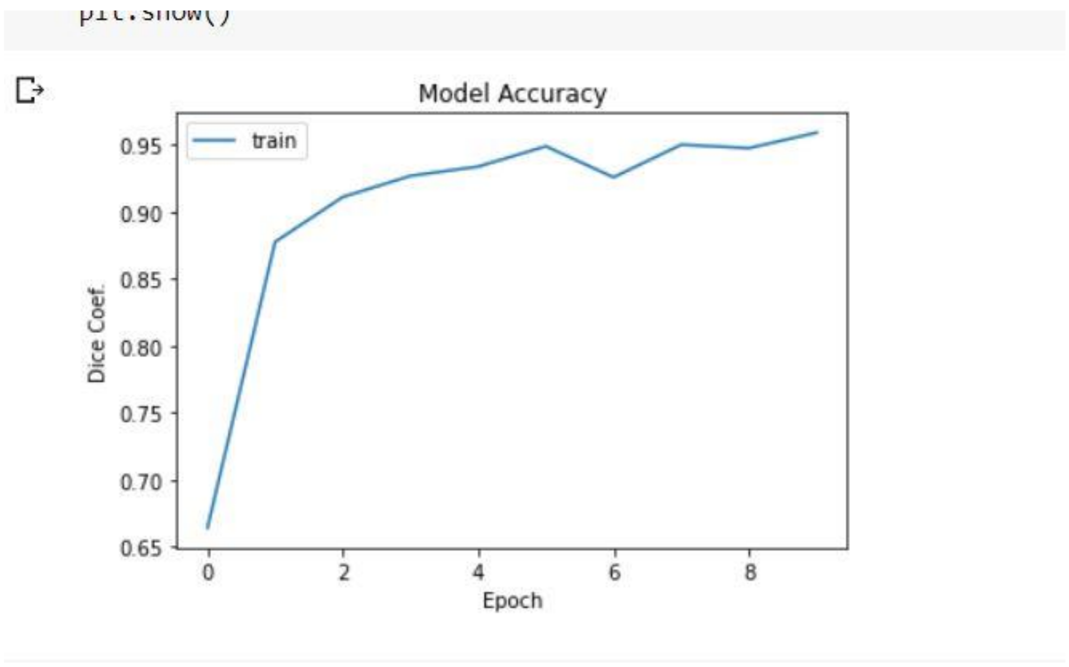

```
plt.show()
```

Fig.5.16 Accuracy graph of Transfer Learning Model

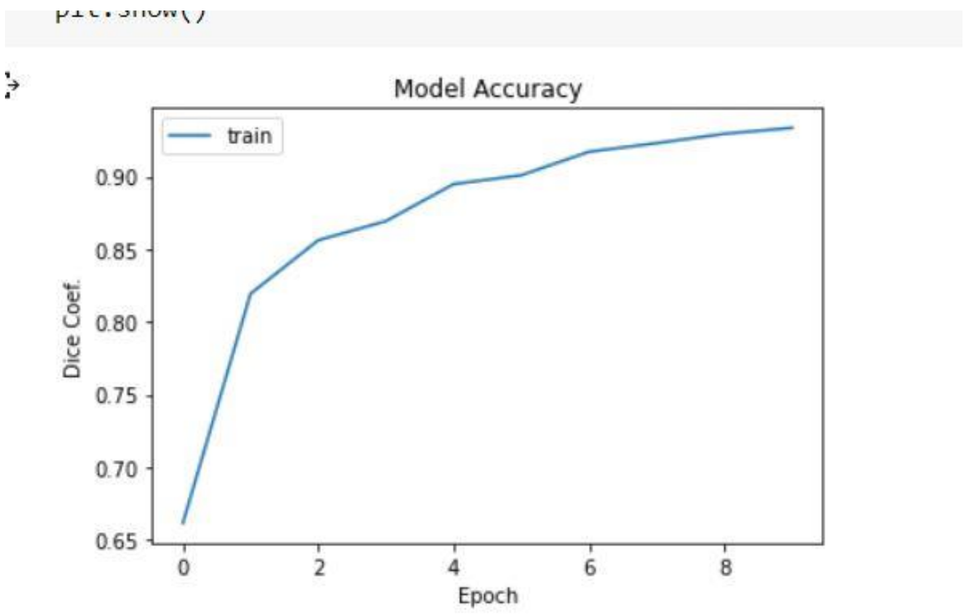

```
plt.show()
```

Fig.5.17 Accuracy graph of Model trained from scratch

From the above comparison we can see, model trained from scratch achieved accuracy of 95% in 10 epochs with 1622 samples whereas the 97% accuracy was achieved with transfer learning approach within 10 epochs and with using only 25% of samples.

The reason behind this can be, as we are using a network which is already trained to detect bones from CT images, this transferred model network has learned to identify basic image components like lines, corners, intensity variations from CT images. This cuts off the epochs required to train the random model for the basic structures and this also overcomes overfitting due to trained model initialization.

Thus, transfer learning approach greatly reduced the training time and proved its usefulness in cases where lesser amount of labelled data is available.

Chapter-6

Limitations

The provided solutions, while possessing obvious advantages, have some important limitations in terms of functionalities and use cases. We will discuss them below:

- This project is mainly designed to use in the medical field for the CT scan system, so it will need technical staffs to handle it.

- The dataset, used to train the machine learning model, is limited as some of the images downloaded were corrupt.

- The project is purely dependent on the datasets. Good datasets would lead to better result.

- This code will not segment whole parts of the images, only specific parts which is trained upon is segmented.

- Since the data used can be corrupt as it is not self-designed data so images segmented may show little variation.

Chapter-7

Future Scope

Future scopes of medical segmentation project:

- **Adding more datasets**: This code is trained only on particular image i.e. part of abdominal region. So, adding more datasets to it and training on those datasets would help in getting vast results.

- **Creating more interactive output**: Labelling every part in an image may give clear identification of the output image.

- **Segmentation of more parts**: Since the model is trained on some specific body parts, the project would be better if it can segment a greater number of parts.

- **Learning new techniques:** New techniques like Transfer Learning is needed so that model can further be used predict various output.

- **Improving Accuracy**: We have to improve dice coefficients for better results.

- **Improving algorithm:** We have to improve our algorithm to achieve far better results on a far better dataset.

Chapter-8

Learning After Training

The project “Medical Image Segmentation” is the research project based on deep learning approach. It has been a great privilege to complete this project at a renowned organization **Bhabha Atomic Research Centre (BARC).** Through this internship period I came to know about the organization and their working domain. This project is completed under the guidance **Ms. Janhavi Deshpande,** Scientific officer(C), Electronics Division, BARC(Mumbai).

Through this internship I learned various models of Machine Learning which gave me more knowledge about this project. I also learned about various techniques like Segmentation, Convolutional Neural Network, U-Net architecture, Transfer Learning. I came to know how to work with machine learning models, how segmentation is carried out, after segmenting the data further efforts were made to improve the accuracy and results. To accomplish this purpose epoch and batch selection were made, dropout layer was used to reduce overfitting, how training time can be reduced, problem of gradient vanishing problem was reduced with ReLU layer, problem of appearance of organ was resolved.

This internship has been a valuable experience that helped me in gaining insights to how a project is developed in industry. It inculcated professional etiquettes in me that will prove to be useful in professional life that lies ahead. This training happened at a very crucial time, right before the final year of my degree that boosted my confidence for professional life. Additionally, I felt I was able to contribute to the company by assisting and working on project.

Overall my training at Bhabha Atomic Research Centre has been a success. I was able to gain practical skills, work in professional environment and learn networking to gain useful connections. I am grateful for such an experience.

Chapter-9

Conclusion

With a lot of research on this project, the project is being developed with various technologies of Data Learning and Machine Learning. As this project requires segmenting the data so for this various segmentation techniques have been studied and I found that neural network-based segmentation was proven best. Developing intelligent/advanced methods for medical image segmentation has become a hotspot, leading to hybrid approaches for efficient segmentation.

Analyzing about image processing and segmentation, Convolutional neural network has been used as the project contain data as image. Further researching on different types of neural network I found that U-Net architecture can be used for image segmentation purpose. U-Net architecture has contracting part and expansive path. Contracting helps in feature detection whereas expansive path helps in concatenation of feature detection with the spatial representation of feature. With the help of these two paths it became easy to segment images. After segmenting the data further efforts were made to improve the accuracy and results. To accomplish this purpose epoch and batch selection were made, dropout layer was used to reduce overfitting, how training time can be reduced, problem of gradient vanishing problem was reduced with ReLU layer, problem of appearance of organ was resolved. On further research another technique called Transfer Learning came into lime light. Transfer learning technique uses previously trained to further models which have less number of data. This technique successfully accomplished and I got better results compared with model that was trained from scratch. This technique is useful because the transferred model network has learned to identify basic image components like lines, corners, intensity variations from CT images. This cuts off the epochs required to train the random model for the basic structures and this also overcomes overfitting due to trained model initialization. Further improving this project on algorithm and datasets will help this project to become feasible.

This project has gone through testing process for error detection as it will be helpful in medical field for further analysis of tumor detection, detecting any anomaly present in the bodies so that patients can be cure at prior stages.

Chapter-10

Bibliography and References

[1] Hesamian, M., Jia, W., He, X. and Kennedy, P., 2020. Deep Learning Techniques For Medical Image Segmentation: Achievements And Challenges.

[2] Medium. 2020. Applied Deep Learning - Part 4: Convolutional Neural Networks. [online] Available at: <https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2> [Accessed 4 April 2020].

[3] Sharma, N., Ray, A., Shukla, K., Sharma, S., Pradhan, S., Srivastva, A. and Aggarwal, L., 2020. Automated Medical Image Segmentation Techniques.

[4] En.wikipedia.org. 2020. Medical Image Computing. [online] Available at: <https://en.wikipedia.org/wiki/Medical_image_computing> [Accessed 4 April 2020].

[5] IRCAD France. 2020. 3D-Ircadb 01 | IRCAD France. [online] Available at: <https://www.ircad.fr/research/3d-ircadb-01/> [Accessed 4 April 2020].

[6] Medium. 2020. U-NET Convnet For CT-Scan Segmentation. [online] Available at: <https://medium.com/@fabio.sancinetti/u-net-convnet-for-ct-scan-segmentation-6cc0d465eed3> [Accessed 4 April 2020].

[7] MissingLink.ai. 2020. Image Segmentation In Deep Learning: Methods And Applications Missinglink.Ai. [online] Available at: <https://missinglink.ai/guides/computer-vision/image-segmentation-deep-learning-methods-applications/> [Accessed 4 April 2020].

[8] Mathworks.com. 2020. What Is Deep Learning? | How It Works, Techniques & Applications. [online] Available at: <https://www.mathworks.com/discovery/deep-learning.html> [Accessed 4 April 2020].

[9] Santos, L., 2020. Image Segmentation · Artificial Inteligence. [online] Leonardoaraujosantos.gitbooks.io. Available at: <https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/image_segmentation.html> [Accessed 4 April 2020].