Identification of Lung Cancer Nodules from CT images using 2D Convolutional Neural Networks

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of

MASTER OF TECHNOLOGY

in

Computer Science & Engineering

Submitted by

SUTRISNA ANJOY

(Id: 2020csm008)

Under the supervision of

Prof. SEKHAR MANDAL

Department of Computer Science & Technology

Indian Institute of Engineering Science and Technology, Shibpur

Howrah 711103, June 2022

<u>CERTIFICATE</u>

It is certified that the work contained in the thesis titled **"Identification of Lung Cancer Nodules from CT images using 2D Convolutional Neural Networks"** prepared by **Sutrisna Anjoy** (Roll No: 2020csm008) has been carried out under my supervision in partial fulfillment of the requirements for the degree of Master of Technology and that to the best of my knowledge, this work has not been submitted elsewhere for a degree.

....

Prof. Sekhar Mandal
Department of Computer Science & Technology
Indian Institute of Engineering Science and Technology, Shibpur

Department of Computer Science & Technology
Indian Institute of Engineering Science and Technology, Shibpur
Howrah 711103 (West Bengal), India
June - 2022

CERTIFICATE OF APPROVAL

We hereby, have approved the thesis entitled **"Identification of Lung Cancer Nodules from CT images using 2D Convolutional Neural Networks"** this work carried out by **Sutrisna Anjoy** (Roll No: 2020csm008) under the guidance and supervision of **Prof. Sekhar Mandal** in partial fulfillment of the requirements for the degree of Master of Technology in Computer Science and Engineering.

Date………………….

Board of Examiners:

1……………………………………

Department of Computer Science & Technology
Indian Institute of Engineering Science and Technology, Shibpur
Howrah 711103 (West Bengal), India
June - 2022

ACKNOWLEDGEMENT

I would like to express my deep sense of respect and gratitude to my project guide, **Prof. Sekhar Mandal,** Department of Computer Science and Technology, **Indian Institute of Engineering science and Technology, Shibpur**. for his invaluable advice and resourceful guidance, inspiring instruction, active supervision and constant encouragement without which it would not be possible to give this thesis a shape. Also, I would like to express my gratitude towards all of my lab mates whose constant encouragement helps me to stay focused. I would like to express my admiration towards my family members who care about me constantly. Finally, I would like to give a special thanks to all my friends for their constant effort to make me happy.

I would also like to pay sincere thanks to all the faculty members for their cooperation during the preparation of this work.

Date……………………..

……………………………

Sutrisna Anjoy
Id: 2020csm008

Department of Computer Science & Technology
Indian Institute of Engineering Science and Technology, Shibpur
Howrah 711103 (West Bengal), India

Keywords

Computer Tomography, 2D Convolutional Neural Network, Computer Aided Diagnosis, Lung cancer, Lung Nodule, Lung Nodule Analysis Dataset(LUNA16), Classification, Segmentation, Deep Learning, Hounsfield Unit, Visual Geometry Group(VGG16), Malignant Nodules, Benign Nodule, Region of Interest.

Abstract

Detection of malignant nodules at early stages from computed tomography images is time-consuming and challenging for radiologists. An alternative approach is to introduce computer-aided diagnosis systems. Recently, deep learning approaches have outperformed other classification methods. In this paper, we use 2D convolutional neural networks to detect malignant nodules from CT scan images. We use modified VGG16 for the identification of lung cancer. LUNA 16 dataset is used to train and evaluate the proposed method, and experimental results show encouraging identification performance of the proposed method. We also compare the performance of the proposed method with the existing 2D CNN methods.

Lung cancer has one of the poorest survival outcomes of all cancers. A major number of deaths from cancer worldwide are happening because of it. The main reason behind this has been specified in a famous NCBI research paper[1] it shows us the actuality that two third of the cases are diagnosed at an advanced stage. The only way to improve this death rate is by early detection. Five-year survival rate at different stages of lung cancer is mentioned below in fig1. Moreover, CTOAM[2] proves that about 30% of CT scans are diagnosed wrong. We cannot entirely count on the radiation oncologist because in 30% of cases they are predicting wrong. In this research approach, we are trying to make the life of doctors more comfortable by detecting the nodules and as well as categorizing the nodules in one of the two classes benign and malignant. Deep learning algorithms are nonetheless more efficient, more accurate and less time-consuming, and easy to handle any type of image data. Here we are working with medical image data. A nodule is a collection of abnormal tissues in the lungs with an enormous growth rate. In the first stage often nodules get detected before there are barely any symptoms.

Contents

Abbreviations

CNN: Convolutional Neural Network
CT: Computed Tomography
HU: Hounsfield Unit
LUNA16: Lung Nodule Analysis 2016 Dataset
VGG16: Visual Geometry Group 16
CAD: Computer-Aided Diagnosis
ROI: Region of Interest
Attenuation coefficient: The attenuation coefficient is a measure of how easily a material can be penetrated by an incident of an energy beam.
DICOM: Digital Imaging and Communications in Medicine
MHD: Medical Image File
LIDC-IDRI: Lung Image Database Consortium image collection
TCIA: The Cancer Imaging Archive
ReLu: Rectified Linear Activation Unit

1. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5338577/
2. https://www.ctoam.com/precision-oncology/why-we-exist/standard-treatment/diagnostics/ct-scan/
3. https://theaisummer.com/medical-image-coordinates/

Chapter 1

Introduction

A lung nodule is the abnormal growth of cells that form in the lung tissues. A lung nodule can be benign or malignant. There is some specific morphological difference between these two types of nodules that radiologists can detect with up to 70% of the right predictions. In earlier stages of lung cancer, the nodule size is very small, furthermore difficult to classify if a nodule is malignant or benign.

In this research work, we try to make doctors' lives easier by pointing out the nodules and classifying them into one of the two categories. To accomplish this research work we use the Deep learning technique to train our model and we use a very deep neural network with 2D convolutional layers to build up our model. The main motive for this work is to more accurately classify a CT scan in the earlier stages of lung cancer where radiologists face issues and sometimes diagnose

1.1 Motivation for this work

At present, lung cancer is considered the primary cause of death worldwide. In 2018, 142670 people died in the USA [28]. According to the article [29], about 228,820 new lung cancer cases were detected in 2020, and about 135720 people have died. The early detection of the disease can only reduce the number of deaths.
Lung cancer is a disease of uncontrollable growth of abnormal lung cells. If we detect the malignant nodules in stage 1 then the survival rate is 83%-92% which is relatively high, whereas the survival rate is tremendously low in stage 4 (10% - 1%). The nodule is about 30 mm or less in size in stage 1; in stage 2, the nodule size is around 50 mm. CT scan is the most effective method to detect lung cancer nodules at early stages due to its high-resolution (3D) chest images. Manual detection of malignant lung nodules in the early stage from the CT Scanning images is a difficult task for radiologists. Hence, the alternative approach is computer-aided diagnosis (CAD) systems.

1.2 Literature Review

1. An enhanced multidimensional region-based fully convolutional network-based system is proposed in [2] for the detection and identification of lung nodules. The region of interest is selected using a median intensity projection to leverage 3D information from CT images and the deconvolutional layer. The sensitivity of The system is 98.1%.

2. A linear discriminant analysis-based CAD system is proposed in [1], resulting in a sensitivity of 70% applied on 187 nodules. A CAD system proposed in [8] uses deep features extracted from an autoencoder to classify lung nodules. Here, the LIDC dataset is used, and nodules are extracted from the 2D CT images using the annotations provided in the dataset. A 200-dimensional feature vector represents a nodule using a five-layered denoising autoencoder. This vector is fed into a binary decision tree for the nodule classification, and the method's sensitivity is 83.25%.

4. Shakeel et al. [3] propose a CAD system for lung cancer detection. After noise removal, the nodules are segmented from the CT scan images using a deep neural network, and features are also extracted. The effective features are selected with the help of hybrid spiral optimization intelligent-generalized rough set approach and an ensemble classifier is used for classification.

5. A lung cancer-detecting method is proposed in [4]. After image pre-processing, nodules are segmented using the watershed segmentation method. Different features (like perimeter, eccentricity, mean intensity, diameter, etc.) are extracted from each nodule. Then SVM classifier is used to detect the malignant nodules.
The accuracy of this system is 86.6%.

6. Akter et al. [5] propose a Fuzzy-based image segmentation scheme for the extraction of nodules from CT images. They use a neuro-fuzzy classifier to classify the lung nodules into malignant and benign classes, and the accuracy of their method is 90%.

7. Several deep learning-based CAD systems are presented in [9] for nodule detection. These solutions achieved a sensitivity of over 95%.

8. A Multi-crop Convolutional Neural Network (MC-CNN) is proposed in [10] for lung nodules classification, and the accuracy of this system is 87.14%.

9. A computer-aided decision support system for lung nodule detection based on a 3D Deep Convolutional Neural Network is proposed in [11]. The median intensity projection and multi-Region Proposal Network select potential region-of-interests. The training and validations are done using the LUNA16 dataset. The system has a sensitivity and accuracy of 98.4% and 98.51%, respectively.

10. Jiang et al. [12] propose a lung nodule detection scheme based on multigroup patches cut out from the lung images. A four-channel convolution neural networks model is designed for detecting nodules resulting in a sensitivity of 94%.

11. In this paper, we propose a CAD system that helps radiologists for identifying malignant nodules in CT images. The lung nodules are segmented using annotations provided in the dataset. We use a modified VGG16 2D convolutional neural network for the classification of nodules into benign and malignant classes.

12. Ding and Liao et al. used 3D Faster R-CNN for nodule detection to reduce false positive (FP) results of lung cancer diagnosis [16]. Faster R-CNN shows very good results for object detection. It was used with very deep modern CNN architecture, the dual path network (DPN), to learn the features of the nodules for classification [17].

13. Jiang Hongyang et al. designed group-based pulmonary nodule detection using multi patches scheme with a Frangi filter to boost the performance [14]. Images from the two groups were combined and a four-channel 3D CNN was proposed to learn the features marked by the radiologist. Their CAD system's results show a sensitivity of 80.06% with 4.7 FP for each scan and a sensitivity of 94% with an FP rate of 15.1.

14. Masood et al. proposed a deep fully convolutional neural network (DFCNet) for the detection and classification of pulmonary lung nodules in a CT image [18]. Initially, the nodule was classified as either benign or malignant; after that, the malignant nodule was further classified into four sub-classes on the basis of the CT image and metastasis information obtained from the medical IoT network.

15. Dr. Silvestri and his research team have proposed proteomic classifiers, along with nodule features, to differentiate between small-size benign and malignant lung nodules [19]. They have achieved very good results on 8–30 mm nodule sizes with a reduction of 40% in biopsies on benign nodules.

16. After the popularity of convolutional neural networks (CNNs) in image analysis, different types of connectivity patterns were proposed by researchers to increase the performance of deep CNNs. Up until now, in the deep CNNs, dense topology structures ResNet, DenseNet [20], and DPNs performance is superior as compared to other ones, but there is still room for connection improvements in these topologies [21]. The MixNet architecture has improved connection structures with better features of extraction and reduced parameter redundancy [22].

17. Gu Yu et al. proposed 3D deep CNN with multiscale prediction strategies for the detection of lung nodules from segmented images [16]. The 3D CNN performs much better with richer features than 2D CNN. In addition to 3D CNN, a multiscale lung nodule prediction strategy was applied for the small nodules with cube clustering techniques.

18. Zhao J et al. proposed a new method for lung segmentation and nodule detection by combining the features from CT and PET images [23]. They used a dynamic threshold-based segmentation method for lung parenchyma from CT scans and identified doubtful areas through PET scans. After that, they performed watershed-based segmentation techniques to find the suspected areas of nodules in the CT images. Later, a support vector machine was used to classify the nodules in the CT images through textual features and, lastly, PET images were used to validate the method.

19. In some of the works [24], MixNet was used for the first time for lung nodule detection and classification with GBM on publically available LUNA16 and LIDC-IDRI datasets and achieved very good results of detection (94%) and specificity (90%). In these datasets, only the nodules of sizes greater than 3 mm were annotated by the three to four expert radiologists. However, in the case of an individual radiologist's examination of a CT scan, nodules of sizes less than 6 mm are usually missed. CT scan analysis techniques are facing a lot of false positive results in the early stage of a lung cancer diagnosis. Therefore, a multi-strategy-based approach is needed for early-stage lung cancer detection.

Chapter 2

What is a CT scan?

The most suitable approach to investigate lung cancer is a CT scan. CT scan or computed tomography combines a series of X-ray images taken from divergent angles around your body and uses computer processing to create cross-sectional images (slices) of the bones, blood vessels, and soft tissues inside your body. CT scan images provide more detailed information than plain X-rays do, making it possible to diagnose and manage lung cancer earlier and more accurately. Our dataset contains low radiation-dose computer tomography images. A low radiation-dose CT scan is preferred for Lung cancer because it contains more detailed information than x-ray do and it has less radiation exposure than a standard radiation-dose CT scan. The CT scan is used to detect pulmonary lung nodules. CT scan uses X-rays to detect and record the amount of radiation observed by the different tissues. The CT scan set that we are using is following the RAI anatomical orientation (Right-Anterior-Inferior). Where right means passing the X-ray from right to left, Anterior means from front to back, and inferior means from bottom to top.

2.1 Contents of lung CT scan

A lung CT scan can contain different substances like bones, blood vessels, lung tissues, air, and water. Each substance has different radiodensity values. Hounsfield unit is a relative quantitative measurement of radiodensity used by radiologists in the interpretation of computer tomography images. Each substance has different HU or Hounsfield Unit values which depend on the absorption/attenuation coefficient of radiation within a substance used during CT reconstruction to produce a grayscale image. The linear attenuation coefficient of each material at the selected effective energy is converted to CT pixel numbers Hounsefiend unit using the standard equation:

$$HU_{material} = \frac{(\mu_{material} - \mu_{water})}{\mu_{water} \times 1000},$$

where μ_{water} is the attenuation coefficient of water and if you want to calculate the Hounsfield unit value for the lung tissues then:

$$HU_{lung\ tissue} = \frac{(\mu_{lung\ tissue} - \mu_{water})}{\mu_{water} \times 1000},$$

where $\mu_{lung\ tissue}$ is the attenuation coefficient of the lung tissue and $HU_{lung\ tissue}$ is the Hounsfield unit value of the lung tissue. Where this attenuation coefficient id depends upon the dosage of the CT scan, here in this project we are using a low-dose CT scan only.

Substance	Radiodensity(HU)
Air	-1000
Water and Blood	0
Lung tissue	-500
Bone	700

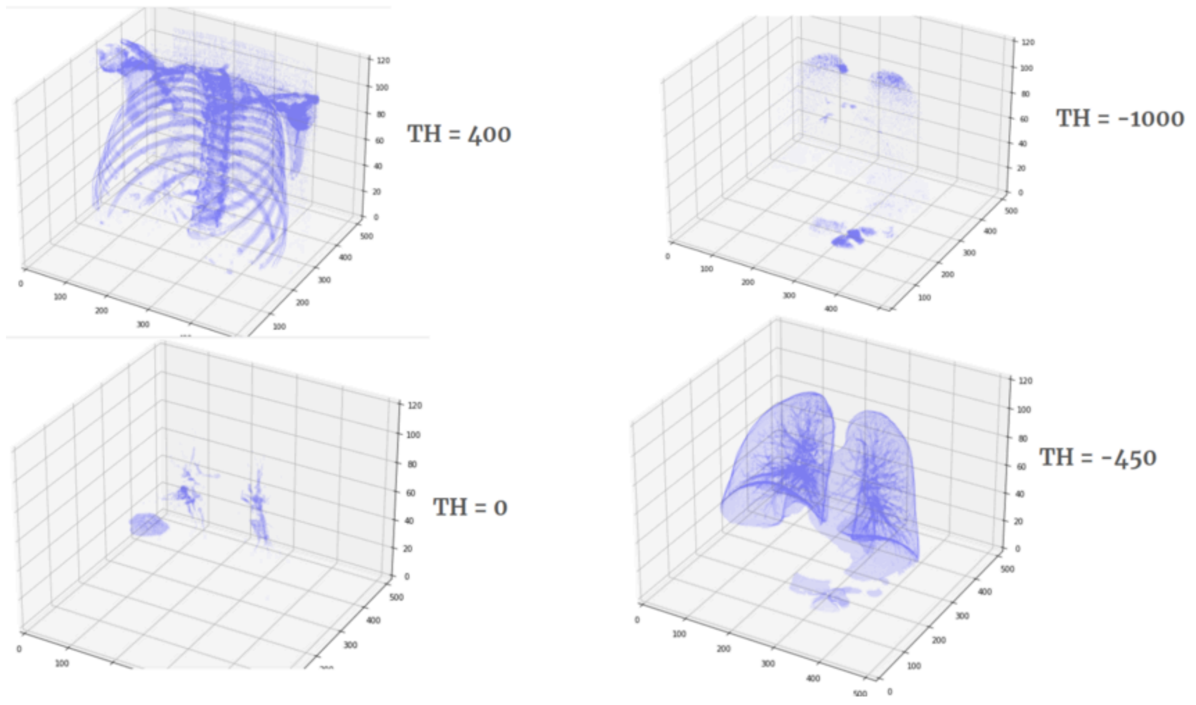


Fig. 1

Chapter 3

Dataset

There are various publicly available datasets for CT scan data in metal image format and DICOM format. In metal image format each CT scan contains two files one is the MHD file which contains metadata about the CT scan and the RAW file contains the real CT scan slice pixel data.

3.1 Available Datasets

Datasets that are available are described as follows:

LUNA16: This data was published for the LUNA grand challenge in 2016[26]. For the challenge, they use the publicly available LIDC/IDRI Dataset and exclude the CT scans with slice thickness greater than 2.5 mm. This data is in the format of the metal image(MHD/RAW) where the MHD file contains the metadata about the CT scan and the RAW file contains real data of the CT scan slices. This dataset contains a total of 888 CT scans all nodule size in this dataset is >=3.

LIDC-IDRI: This dataset is available on the official page of The Cancer Imaging Archive data as the name of LIDC-IDRI[27] data. This dataset contains a total of 1018 CT scans. In this dataset, the CT scans are available in the DICOM format. This dataset is generated and published by the National Cancer Institute for the development, training, and evaluation of the CAD system for lung cancer detection and diagnosis. This dataset also contains annotations for each nodule present in the CT scan in XML format; this data also contains the size of each nodule and the nodule count of each patient.

Kaggle Data Science Bowl 2017(DSB3): This dataset contains a total of 1397 CT scans. It has a CSV file that includes the id of each CT scan and their corresponding class that the CT scan has one or more malignant nodules or not or a person has cancer or not. Here CT scans are available in DICOM format

3.2 LUNA16

This dataset contains 888 CT scans in MHD and RAW format where the MHD file contains the metadata of each CT scan and the RAW file contains the original CT scan slice pixel data, also called metal image format. This dataset contains the following files and folders:

Subset: This folder contains a total of ten folders subset0 to subset9 which contain the MHD files and their corresponding RAW files for each 888 CT scan.

Annotations.csv: file contains annotation used as the reference standard for the nodule detection track.

Candidates.csv: file contains the coordinate of the lung nodules. A lung nodule can be benign or malignant. the original set of candidates used for the LUNA16 workshop at ISBI2016. This file is kept for completeness,

candidates_V2.csv: file that contains an extended set of candidate locations for the 'false positive reduction' track.

Images: The complete dataset is divided into ten subsets, all subsets are in compressed format. In each subset, the CT scans are available in Metalimage(mhd/raw) format. Each .mhd file is stored with .raw binary files for storing the pixel data for the CT scan.

Annotations: This file contains one finding per line. Each line of the csv file holds a series instance UID of the CT scan, the position is mentioned in x, y, and z coordinates in the world coordinate system, and the corresponding diameter of the nodule. This annotation file contains a total of 1186 nodules.

Candidates: each line of the candidate.csv file contains the Instance UID of the CT scan and the x, y, and z position of every candidate nodule in world coordinate format and the corresponding class of the nodule. There are 2 classes available: benign, which is also referred to as 0 in the candidate.csv file and another is malignant nodule class marked as 1 in the candidate.csv file. This list of candidates is provided for false positive reduction.

The position of the nodules is computed using three existing candidate nodule detection algorithms available. There are a total of 551065 nodules present in the candidate.csv file. In this computation 1120 out of 1186 were detected successfully.

After 2016 a new set of candidate.csv was generated, candidates_V2.csv. This new combination set substantially achieved more accuracy than the previous one(1166/1186 nodule).
Each CT scan .mhd file contains some CT scan attributes listed in the below table:
ObjectType = type of the file, LUNA16 dataset contains all Image files.

NDims = Specify the dimension of the original data, for each CT scan in this dataset number of dimensions is 3.

BinaryData = mention if the corresponding raw data is binary type or not. True specify that data is binary type.

BinaryDataByteOrderMSB = used to specify a particular byte ordering. False means corresponding data doesn't follow MSB byte ordering

CompressedData = indicates if the corresponding raw file is compressed or not. False means the corresponding raw file is not in a compressed format which is indeed true for all CT scans in LUNA16 data

Offset = Physical location (in millimeters and concerning to the machine coordinate system or the patient) of the first element in the image. The physical orientation of the object is defined as an NDims x NDims matrix that is serialized in a column-major format in MetaIO files. -120.049 9.480 -657 means in this coordinate the first element if the image is present it specifically tells us the origin of the CT scan.

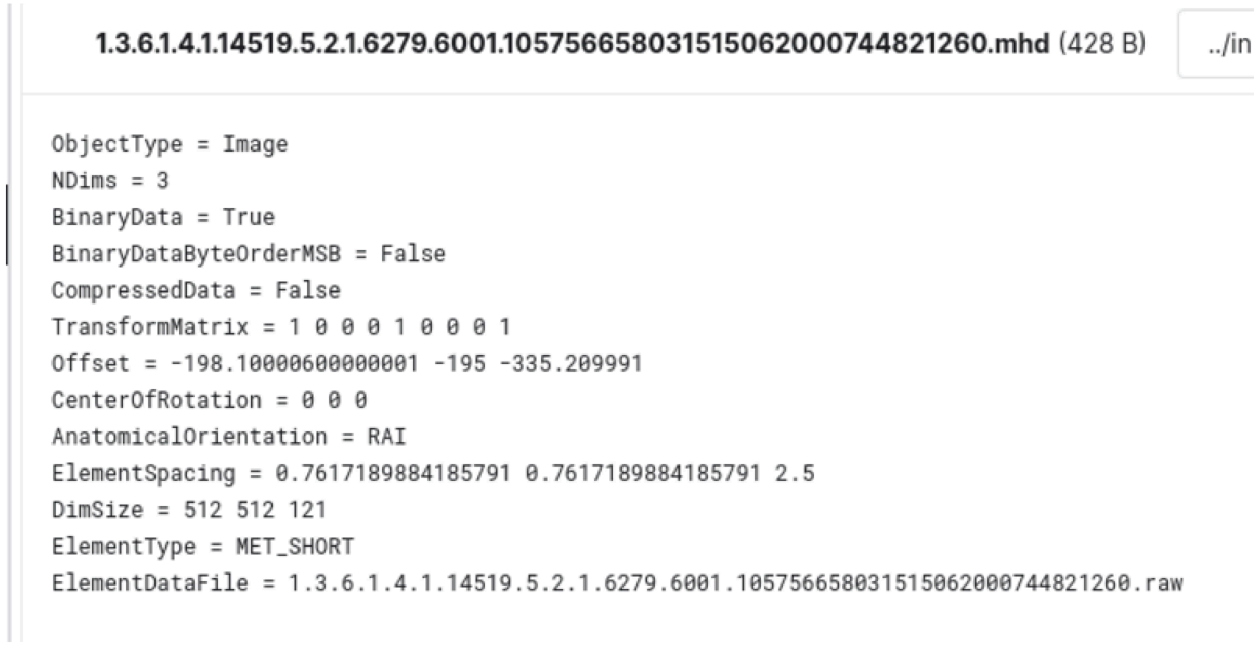
CenterOfRotation = It specifies which coordinate we have to rotate the image.0 0 0 means we don't have to rotate the image and 0 0 1 means we have to rotate the image with respect to the z coordinate.
AnatomicalOrientation = It refers to which anatomical orientation the CT scan is taken. Most of the CT scans in LUNA16 follow RAI anatomical orientation which means. CT scan is taken from right to left the anterior to posterior and then from interior to superior.

ElementSpacing = This attribute defines space between two coordinate points. 0.55 0.55 1 means space between two x coordinate points is 0.55, space between two y points is also 0.55, and space between two points in z coordinate is 1 the ElementSpacing in z coordinate refers to the space between each slice in a CT scan image.

DimSize = This object lets us know the dimension of the total CT scan. Here 512 512 321 means a total of 321 slices are present in the CT scan image and each slice is of size 512 512.

ElementType = It defines a metal image object tag. For most of the CT scans in LUNA16 data MET_SHORT is used as ElementType

ElementDataFile = It specifies the original raw file name corresponding to this mhd file. 1.3.6.1.4.1.14519.5.2.1.6279.6001.100621383016233746780170740405.raw this file name is the corresponding raw file to this mhd file.



1.3.6.1.4.1.14519.5.2.1.6279.6001.105756658031515062000744821260.mhd (428 B) ../in

```
ObjectType = Image
NDims = 3
BinaryData = True
BinaryDataByteOrderMSB = False
CompressedData = False
TransformMatrix = 1 0 0 0 1 0 0 0 1
Offset = -198.10000600000001 -195 -335.209991
CenterOfRotation = 0 0 0
AnatomicalOrientation = RAI
ElementSpacing = 0.7617189884185791 0.7617189884185791 2.5
DimSize = 512 512 121
ElementType = MET_SHORT
ElementDataFile = 1.3.6.1.4.1.14519.5.2.1.6279.6001.105756658031515062000744821260.raw
```

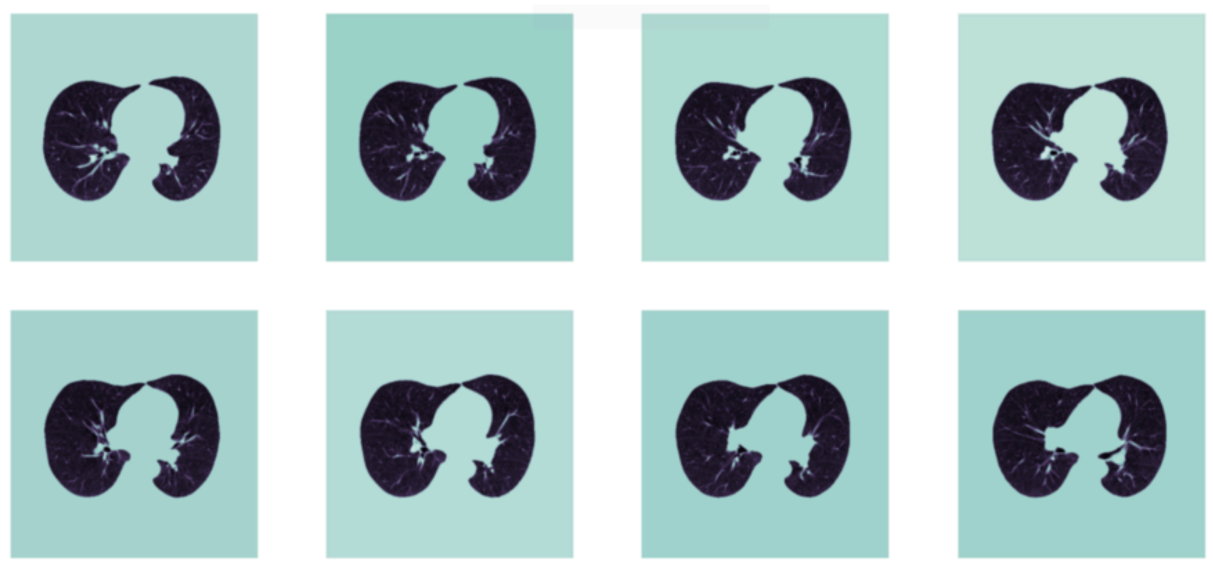
Fig. 2

3.3 Data Visualization:

Visualization of the dataset is an important part of training, it gives a better understanding of the dataset. But CT scan images are hard to visualize for a normal pc or any window browser. Therefore we use the pydicom library to solve this problem. The Pydicom library gives an image array and metadata information stored in CT images like patient's name, patient's id, patient's birth date, image position, image number, doctor's name, doctor's birth date, etc.

Luna16 dataset is a directory that contains many subdirectories named on patients' ids. A complete subdirectory is a 3d image of lungs which is stored in around 180 2d image slices according to their image number.

Thresholding: In this process, we select the pixel within a certain threshold value. A threshold value of (-400) is used to segment the lung tissues from scan images. We use this technique to visualize our data. lung CT scan slice after applying this technique is shown below

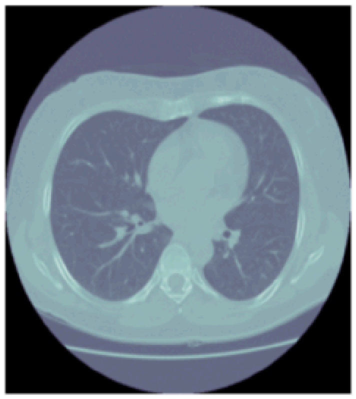


Thresholding

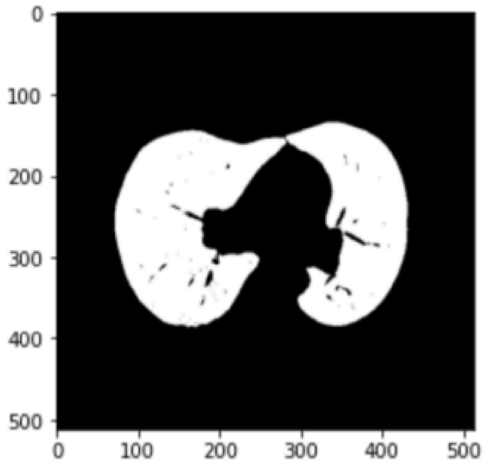
Fig. 1

3.4 Lung Segmentation: For the segmentation of the lungs, we use a watershed segmentation algorithm which is the most efficient pre-existed lung segmentation algorithm. Due to CT scan noise voxels at the edge of the lung, tended to fall outside the range of lung tissue radiodensity. To filter noise and include voxels from the edges, we use Marker-driven watershed segmentation. It produces much better segmentation than thresholding; all missing voxels are largely re-included (TH = -400).
we extract internal and external markers from CT scan images with the help of binary dilations and add them to a completely dark image using watershed methods. And it removes external noise from the image and gives a watershed marker of lungs and cancer cells. As we can see in the below figure watershed marker removes external noise and applies a binary mask on the image, black pixels in the lungs represent cancer cells.

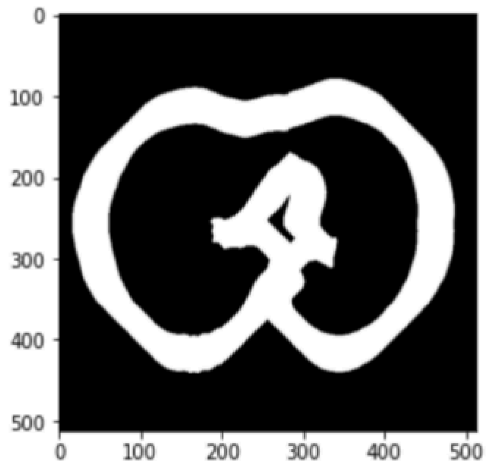
Without any processing



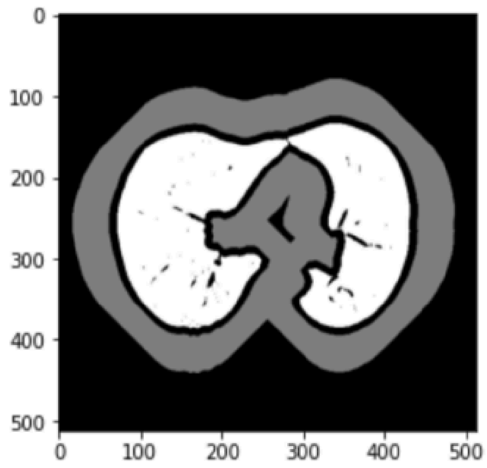
Internal Marker



External Marker



Watershed Marker



Chapter 4

Convolutional Neural Network

CNN or Convolutional Neural Network also known as ConvNet, is a class of neural networks specialized in processing the data with a grid-like topology, such as an image. A digital image contains pixels in a grid-like structure. To understand CNN we have first to understand how the human brain classifies and detects an object from real life or an image. The human brain first tries to identify some tiny features that are present in the image. If all the features that an object contains are present in an image in some specific order then we can say that the object is also present in the image.

The convolution layer is the core building block of CNN. In this work, we are working with 2D CNN because of this we only want to deliberate the information about 2D CNN. The main components of the two-dimensional convolutional layers are discussed below. We are only discussing the topic and the components that we are using in our work.

4.1 Different Components of CNN:

4.1.1 Conv2D:

Convolutional layer is usually abbreviated as the Conv2D layer. Format of Conv2D layer is as follows: Conv2D(Number of filters, Kernel or filter size, activation = "Name of activation function", kernel_initializer = "Name of kernel initializer", padding = "Specify the type of padding you want to use"). A kernel or filter in the Conv2D layer slides over the 2D image, performing an element-wise multiplication with the whole image and creating a feature map out of it; this process is also called a convolutional operation. This convolution operation is performed with the input and creates a feature map and this output feature map will pass to the next layers. For each filter, we will be having a feature map. All the features all together will make an output for the next layer.

Filters: Filters are also called feature detectors in the convolutional neural network. Filter represents a small feature that is present in the image. Feature detectors or filters help you to identify the features like edges, vertical lines, horizontal lines, and bends. We can decide the number of filters and the size of the filters present in the convolutional layer.

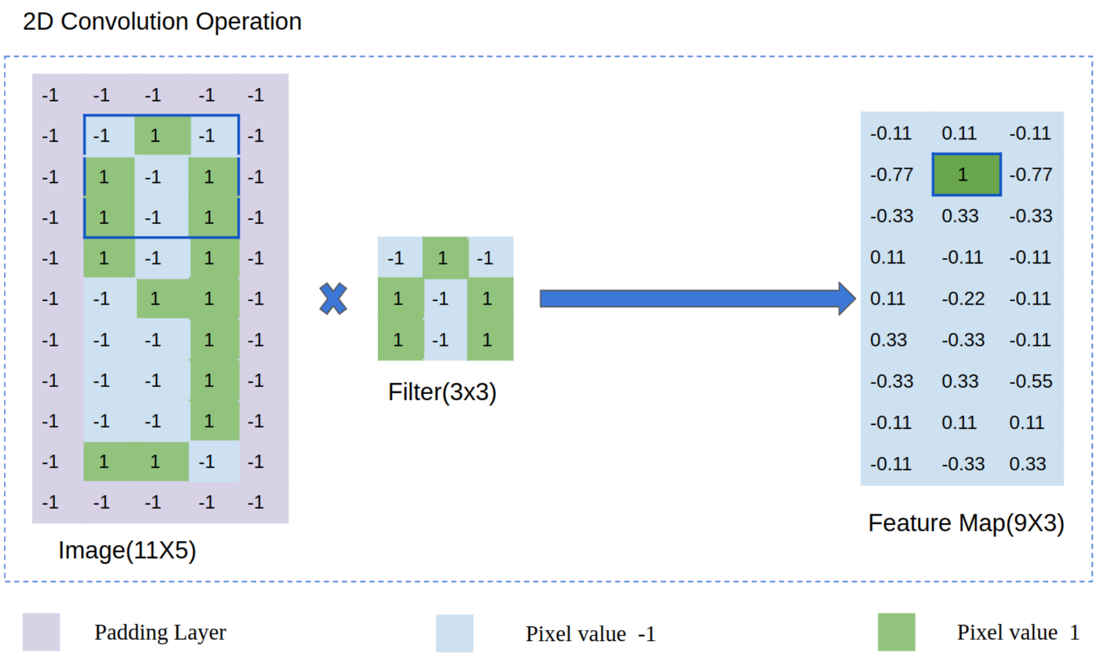


Fig. 3

ReLu: The activation function that we have used in our model is a rectified linear activation unit or ReLu. It's a non-linear activation function that is used in multilayer deep neural networks commonly in hidden layers. This function can be represented as

$$f(x) \ = \ max(x\,,\ 0) \ \ where\ x\ is\ input$$

or,

$$f(x) \ = \ x \quad if\ x \ > \ 0$$
$$f(x) \ = \ 0 \quad otherwise$$

ReLu is perhaps the most commonly used activation function for hidden layers. Because it's both easy to implement and effective in overcoming the limitations for other popular activation functions like sigmoid and tanh. Furthermore, we cannot use only the linear activation function; it will produce only a linear function which will not be suited for any complex image data. We need a non-linear transformation function for this type of complex data. ReLu speeds up our whole process by introducing a very simple gradient computing function. Relu doesn't suffer from vanishing gradient problems like sigmoid function.

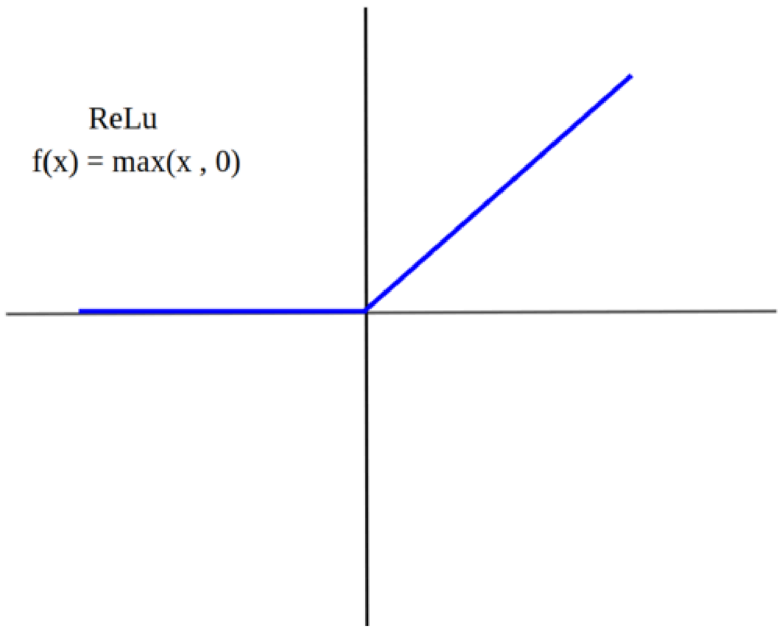
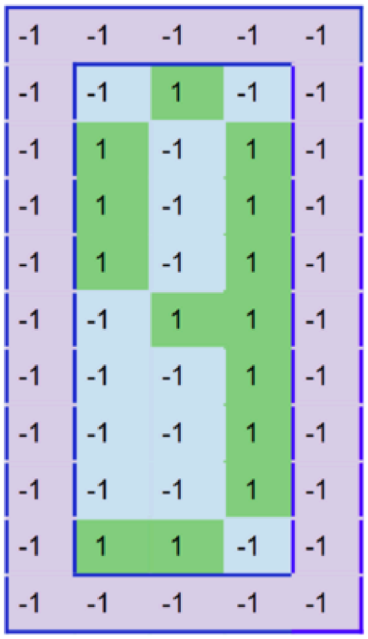


Fig. 4

Kernel Initializer: Weight initialization is used to define the initial values of the parameters and helps our model to converge in a better and faster way**.** Training a deep model is a sufficiently difficult task and most of the time algorithms are affected by the initial value of the parameter The initializer that we are using in all of our hidden is he_normal.

he_normal draws samples from a truncated normal distribution centered in zero with $stddev = sqrt\ (\frac{2}{fan-in})$ where fan-in is the number of input units in the weight tensor.

Padding: padding is simply a process of adding layers of zeros to our input image. For a grayscale image, if the image size id $(n \times m)$ and the filter size are $(f \times f)$ then after one convolution operation the output size will be $\left[\frac{n-f+2p}{s} + 1\right] \times \left[\frac{m-f+2p}{s} + 1\right]$ Where $(n \times m)$ is the image size, $(f \times f)$ is the filter size and s specify the stride and p defines padding. The image size will shrink every time a convolution operation is applied. The pixel in the corner or edges will get much less important than the pixel in the middle and will play a much less important role in output. To largely include the corner or the edge pixel and help them to play a great role in output we use padding where we want to treat each pixel as the same.



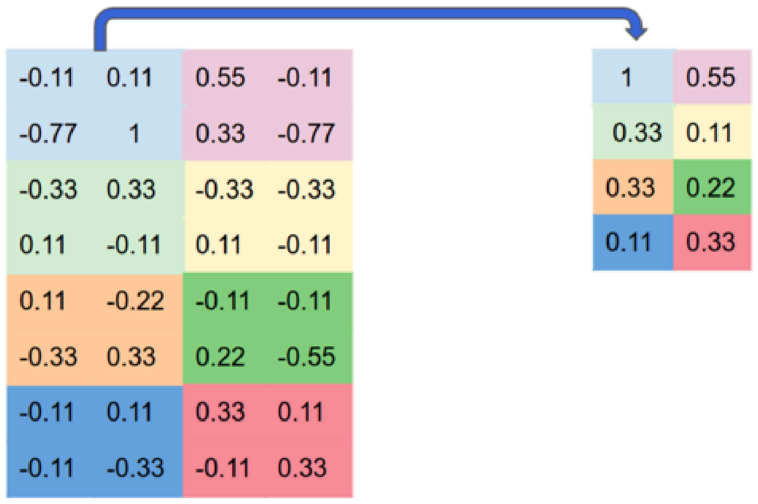
-1	-1	-1	-1	-1
-1	-1	1	-1	-1
-1	1	-1	1	-1
-1	1	-1	1	-1
-1	1	-1	1	-1
-1	-1	1	1	-1
-1	-1	-1	1	-1
-1	-1	-1	1	-1
-1	-1	-1	1	-1
-1	1	1	-1	-1
-1	-1	-1	-1	-1

Padding

Fig. 5

4.1.2 Max Pooling

Max Pooling is an operation that is used to downscale the image. Using the Max Pooling operation, we reduce the size of the image or input and thus reduce the computational cost. We use Max Pooling to extract the maximum value from the feature map. For example, if we use 2x2 Max Pooling then it would choose the maximum value from the 2x2 matrix. Picking up only the maximum value means extracting only the important features in a region. Max Pooling also adds translation invariance. Max Pooling adds shift invariance, rotation invariance, and scale invariance. Max Pooling is also used for making our model work more efficiently for the input image data with small distortion.



-0.11	0.11	0.55	-0.11
-0.77	1	0.33	-0.77
-0.33	0.33	-0.33	-0.33
0.11	-0.11	0.11	-0.11
0.11	-0.22	-0.11	-0.11
-0.33	0.33	0.22	-0.55
-0.11	0.11	0.33	0.11
-0.11	-0.33	-0.11	0.33

1	0.55
0.33	0.11
0.33	0.22
0.11	0.33

2x2 MaxPooling with Stride 2

Fig. 6

Stride: stride is a component of a Convolutional Neural Network that is used to compress the size of the data. It will decide the number of pixels by which the filter will move next. It controls the movement of the filter over an image or video. For example, if a neural network stride is set to one then the filter will move by one pixel at a time in the image. The size of the output image depends on the stride you decided and the size of the filter. If the image size is $(n \times m)$ and the filter size is $(f \times f)$ and the stride that you used is s then the resulting output image size will be:$\left[\frac{n-f+2p}{s} + 1\right] \times \left[\frac{m-f+2p}{s} + 1\right]$. Thus more strides will reduce the size of the output feature map. Where $(n \times m)$ is the image size, $(f \times f)$ is the filter size and s specify the stride and p defines padding.

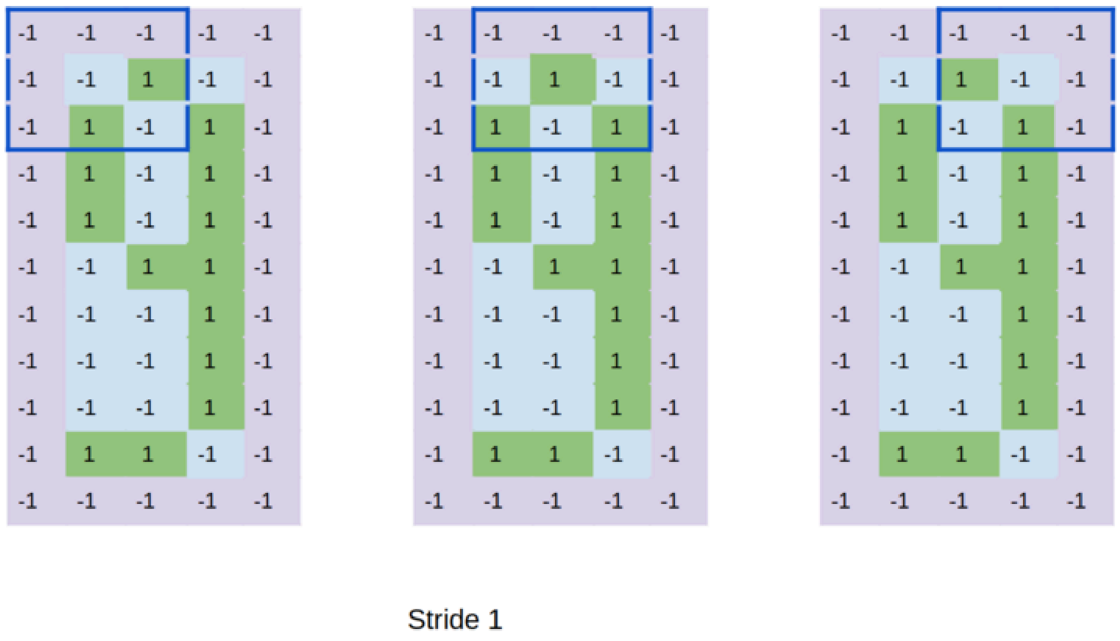


Fig. 7

4.1.3: Dropout

Dropout is a regularization technique that is used to prevent the overfitting of the model. Dropout is used to just drop some percent of the neurons from the network. dropout(0.1) means that you just randomly drop 10% of neurons from the network and then go to the next layer. Dropout is preferred after the dense layer, not after the convolution layer. Moreover, dropout makes models more robust by randomly removing some neurons. Moreover, it removes different neurons in the different passes that we go through. Dropout essentially creates a new network in every pass thus making our model works in a better way. If a large number of neurons are extracting the same feature in every pass then it adds more significance to these features and this leads to an overfitting problem. Furthermore, we would be wasting more computation power just to extract the same features in every pass. Dropout will simply be zeroing the output of a percentage of neurons.

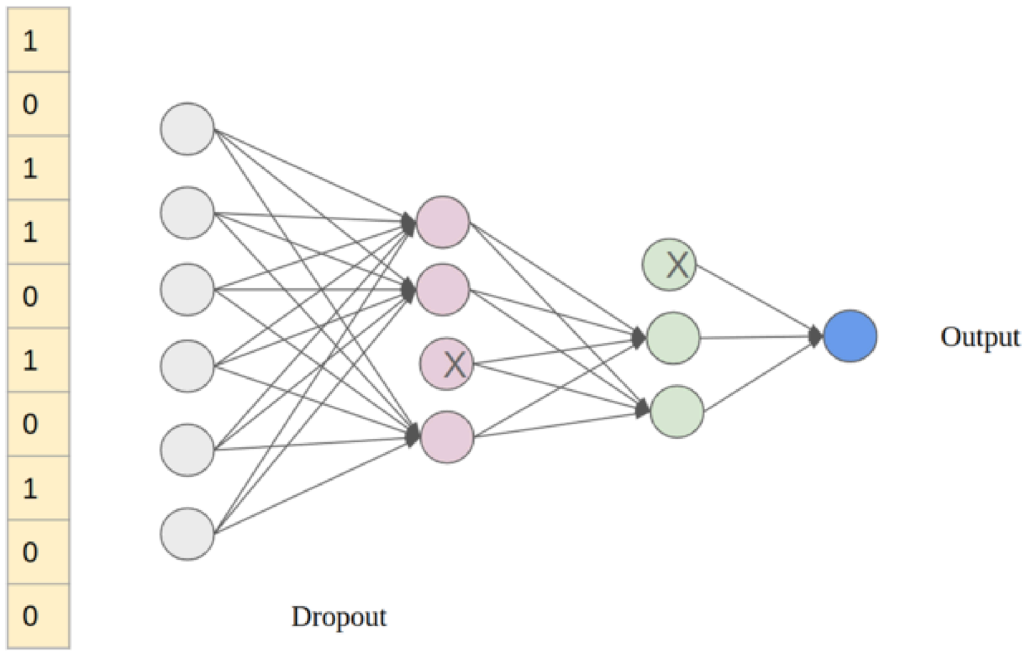


Fig. 7

4.1.4 Batch Normalization

BatchNormalization helps each layer of the network to learn independently. What it does is, it will normalize the output of the previous layer so that normalized data will be produced to the next layer. It helps the next layer to learn more efficiently and more accurately. It also works as a regularization tool and it will avoid the overfitting problem of the data. This layer is added to a sequential model to standardize the input and output. It is useful for complex data and for the imbalanced dataset that needs normalization before every convolution operation. It scales the output of the previous layer BatchNormalization has some kind of regularization effect. Using batch normalization means normalizing the input from the previous convolution layer. Moreover, batch normalization solves internal covariate shift, vanishing gradient, and overfitting problems. Internal covariate shift will occur if input statistical distribution keeps changing in every epoch. In the mini-batch gradient algorithm in one epoch the model goes through a batch of the image in our case we use batch size as 64. If the present batch image statistical distribution is very different from the statistical distribution of the previous batch it will lead to slow down our training process and it will take more time to converge. Batch normalization will normalize the input data in each hidden layer so that the statistical distribution will stay fairly constant thus helping our model to converge in a faster and better way.

4.1.5: Flatten Layer

This layer is used to convert the pool of feature maps to a single-dimensional array. A dense layer will only take a single-dimensional array as input, for any 2D or 3D data we need a flattened layer to pass the data to a dense layer.

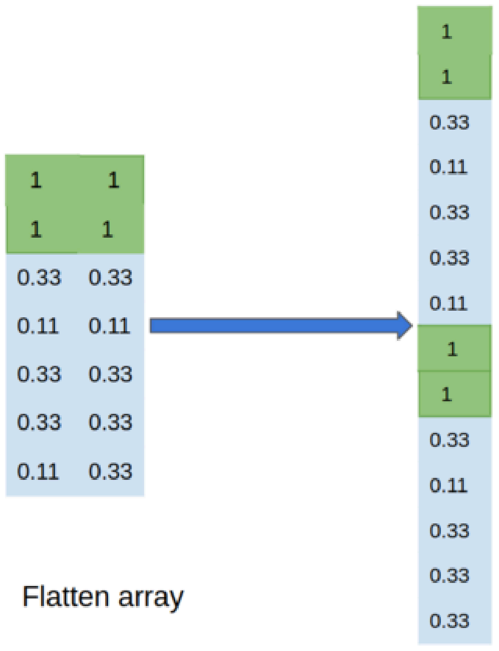


Fig. 9

4.1.6: Dense Layer

In a neural network a dense layer is a layer that is densely connected with its preceding layers. Densely connected means each neuron of this layer is connected with each neuron of the preceding layer. The dense layer will take all the neurons' output of the previous layer as input. Before giving an input to the dense layer we have to flatten the data.

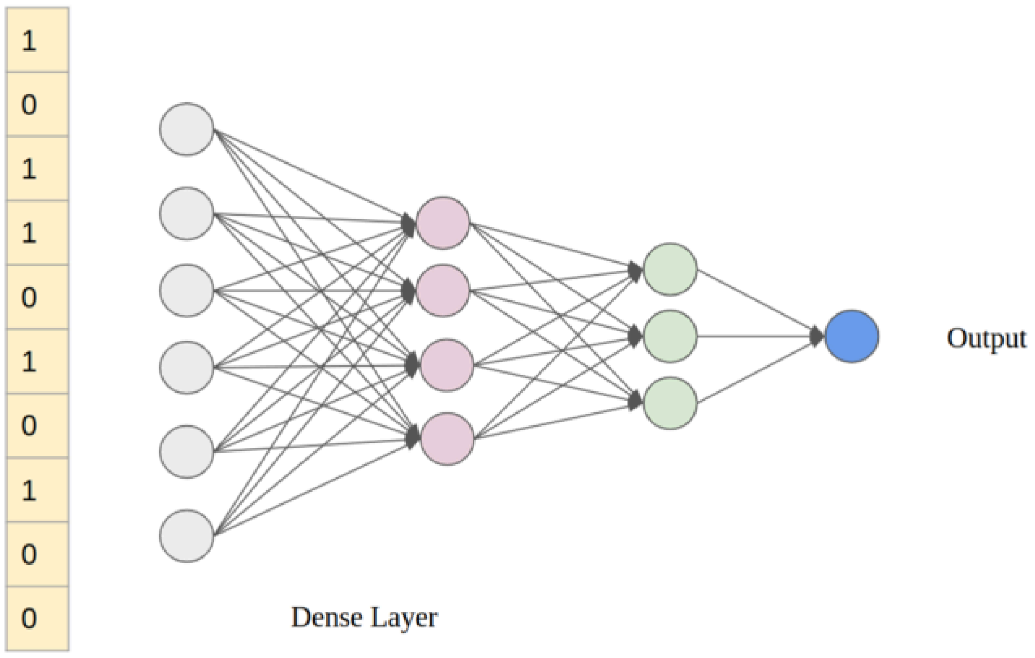


Fig. 10

4.2: Softmax function

The Softmax regression is a form of logistic regression that normalizes an input value into a vector of values that follows a probability distribution whose total sums up to 1. softmax is sometimes also referred to as multinomial regression. The component values of the softmax function is ranging between 0 and 1 it's also true for sigmoid but in softmax, all components will add up to 1, so that they can be interpreted as probabilities. the softmax function is used in the output layer only where the number of classes is greater than one. The standard softmax function is defined as

$$\sigma\left(z_i\right) = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_i}}$$

Where $z_i = w_0x_0 + w_1x_1 + w_2x_2 + \ldots\ldots + w_kx_k$

$$= \sum_{i=0}^{k} w_ix_i$$

$$= W^TX$$

and k is the number of classes or the model outputs the very first element of z is the bias and w is weight and x inputs.

Simply it applies the standard exponential function to the component you want to calculate the softmax value to all the output components of the model.

4.3: Categorical Cross-Entropy

Categorical cross-entropy is a loss function that calculates the loss of an example by computing the sum:

$$Loss = -\sum_{i=1}^{k} y_i . log(\hat{y}_i)$$

Where y_i is the true output value and $\hat{y}_i$ is the predicted output of the sample by our model and k is the total number of samples.
We can write the binary cross entropy function as

$$Loss = -\frac{1}{N}\left[\sum_{i=1}^{N} y_i . log(\hat{y}_i) + \sum_{i=1}^{N}(1 - y_i). log(1 - \hat{y}_i)\right]$$

Here N is the total number of samples.
Categorical cross-entropy is a well-suited loss function for the classification tasks. Softmax is the only activation function that is recommended to use with categorical cross-entropy loss or log loss function. It generally calculates the difference between two probability distributions. Calculation difference between two probability distributions is strongly related to KL divergence

$$KLD(y \mid \mid \hat{y}) = \left[- \sum_{i=0}^{k} y_i . log(\hat{y}_i) + \sum_{i=0}^{k} y_i . log(y_i)\right]$$

Where y_i is the true value of the sample and $\hat{y}_i$ is the prediction and k is the number of the total sample.

4.4: Adam Optimizer

Adaptive moment estimation optimizer or adam is a mini-batch gradient descent deep learning optimizer. Adam is a further extension of the stochastic gradient descent algorithm; it can be looked at as a combination of stochastic gradient descent and RMSprop. It takes the squared value of the gradient like RMSprop to update the learning rate additionally it takes advantage of the momentum and instead of taking the real value of the gradients, it takes the moving average value of the gradients. Now, what is moment? Nth moment of a random variable is defined as the expectation value of that random variable to the power of N. Moment is expressed as

$$m_n = E\left[X^n\right]$$

where X is the random variable and n is the time unit.
This optimizer works in an efficient way than other available optimizers for large problems with lots of data and a large number of parameters. Because of using the exponential moving average, the optimizer helps to converge the model in a faster way. It requires less memory and takes less time than other optimizers.

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1)\left[\frac{\delta L}{\delta W_t}\right]$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2)\left[\frac{\delta L}{\delta W_t}\right]^2$$

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t}$$

$$w_{t+1} = w_t - \frac{\alpha}{\sqrt{\hat{v}_t + \varepsilon}}$$

Where m_t = aggregate of gradients at time t with an initial value of m_t as 0

v_t = sum of the square of past gradients sum of $\left[\frac{\delta L}{\delta W_{t-1}}\right]^2$ values, with an initial value of 0.

β_1,β_2 = moving average parameter(default 0.9)

δL = derivative of loss function

δw_t = derivative of weight at time t

$\frac{\delta L}{\delta W_t}$ = gradient of the loss function with respect to the weight at time t, how loss is changing with a small change in weight.

$\hat{m}_t$ = bias corrected m_t

$\hat{v}_t$ = bias corrected v_t

α = learning rate

In this project we use Adam as an optimizer to reduce the time and to increase our accuracy.

Chapter 5

Proposed Method

In this chapter we want to discuss the approach to our project. We want to classify if an image contains a malignant nodule or not by observing the CT scan images. Our approach towards our project is to first extract the feature using which we want to classify a CT scan that contains a malignant nodule or not or whether a person is affected by lung cancer or not. A total lung CT scan area is not needed for our project; our region of interest is only the nodules from the CT scan. We will first discuss the Imbalancing handle part, then we discuss how to segment the nodules present in the CT scan.

5.1: Handle Imbalancing

Imbalanced data refers to those types of datasets where the target classes are unevenly distributed. In our case, our dataset is divided into two classes, which simply means the nodule is non-cancerous and the malignant class means the nodule contains a malignant or cancerous cell or the nodule is cancerous. The LUNA16 contains a candidate.csv file containing a total of 551065 nodules from which 549714 nodules are benign and the rest 1351 nodules are malignant. Only 0.2451% of nodules are cancerous. From Fig. 11 you can clearly observe that the data itself is hugely imbalanced. The number of malignant nodules is very less and not sufficient for model training. With this huge imbalance, we cannot train the data. If we train the data without handling this type of imbalance then it will classify all the nodules as non-malignant because 99.755% of the data are non-malignant.

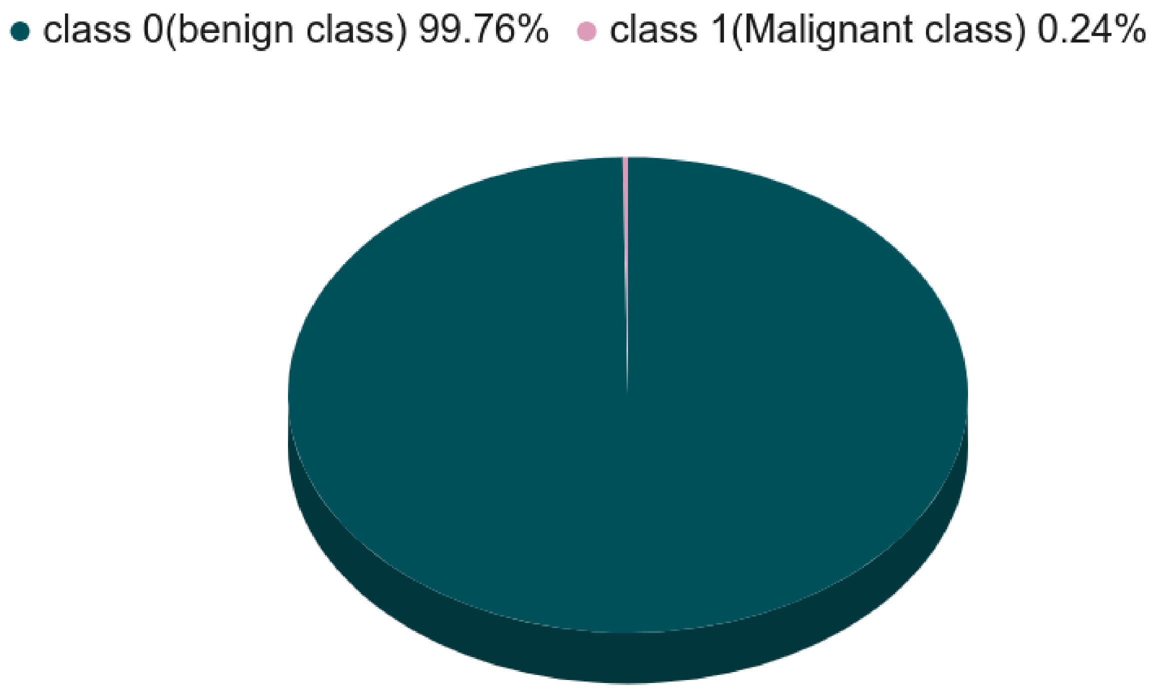


Fig. 11

To solve this problem there are some solutions available on the internet. Applying only one method will not work for our project though.

Undersampling of the majority class: To deal with this problem before splitting the data into training, testing, and validation data we randomly select only $5 \times 1351 = 6755$ nodule coordinates from the negative class in the candidate.csv file to somehow make the data trainable. This is also called under-sampling of the majority class data. We choose the parameter value of five to multiply with positive data because any value less than five will drastically reduce the size of the total trainable data. We also check the performance of our model by increasing and reducing the parameter value; it gives less accurate results than the value 5.

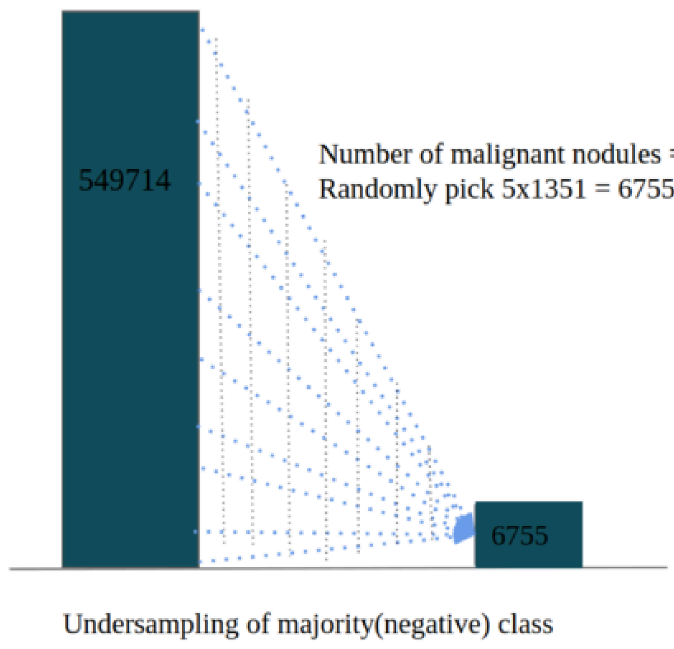

549714
Number of malignant nodules = 1351
Randomly pick 5x1351 = 6755 benign nodules
6755
Undersampling of majority(negative) class

Fig. 12

After sampling the positive class increased to 20%, after this random selection we split the data. 70% of the total data is selected as training data and 20% of the data is selected as test data and the rest data is considered as validation data. After applying the under-sampling, the data remain skewed towards the negative class. We cannot further under-sampling the majority class, which will reduce the data size drastically and lead to less accurate results.

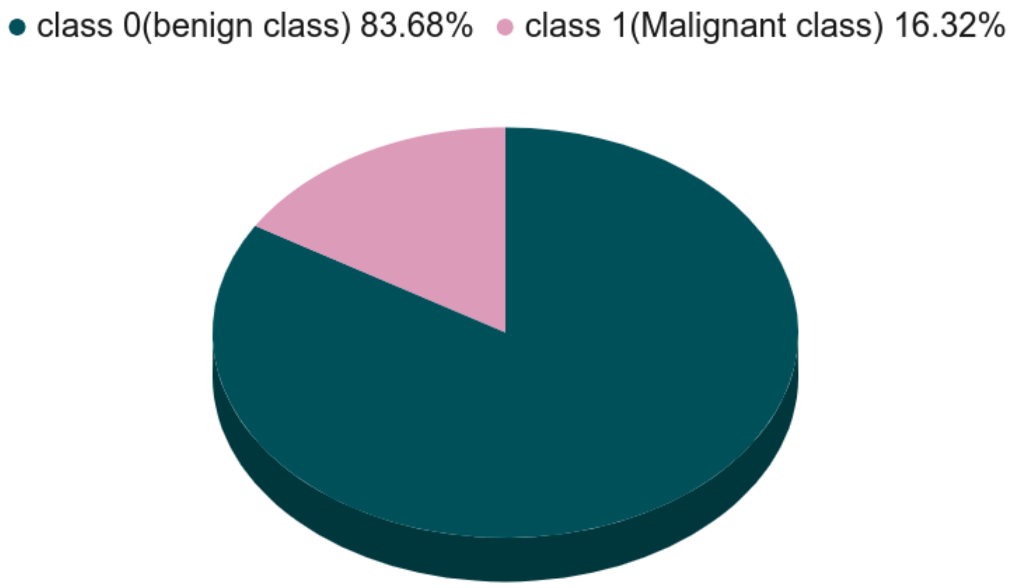


Fig. 13

To solve the further imbalance in the dataset we duplicate the instances of the positive cases which we can also refer to as oversampling of the minority classes. We duplicate the positive instances twice just by reindexing. The code snippet present in Fig. 13 describes the oversampling and reindexing procedure.

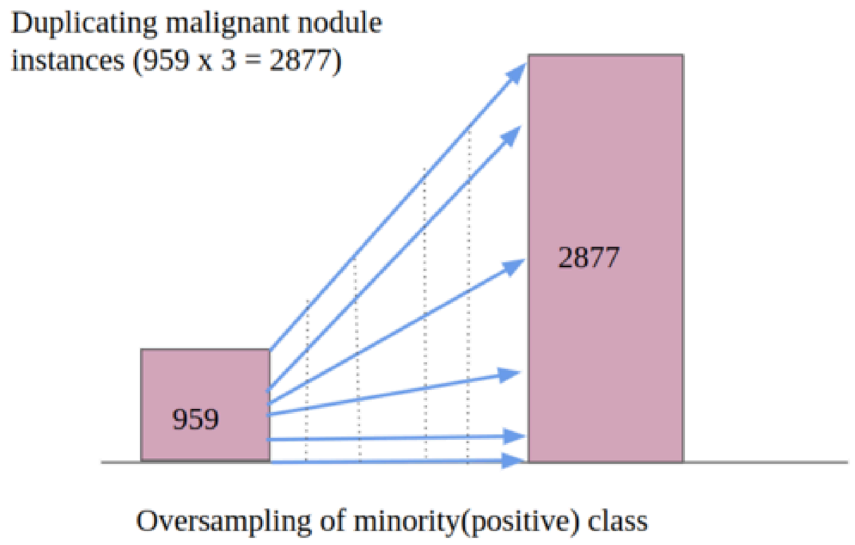

Duplicating malignant nodule instances (959 x 3 = 2877)
2877
959
Oversampling of minority(positive) class

Fig. 14

The Number of positive cases in the training set before oversampling is 959 where the total set size is 5836 around 16.324% of the total case is positive which is shown in Fig. 13. After oversampling, the number of positive cases increased to 2877 which is 37.10% of the total training set and the training data size has increased to 7754 images which are presented below in Fig. 14.

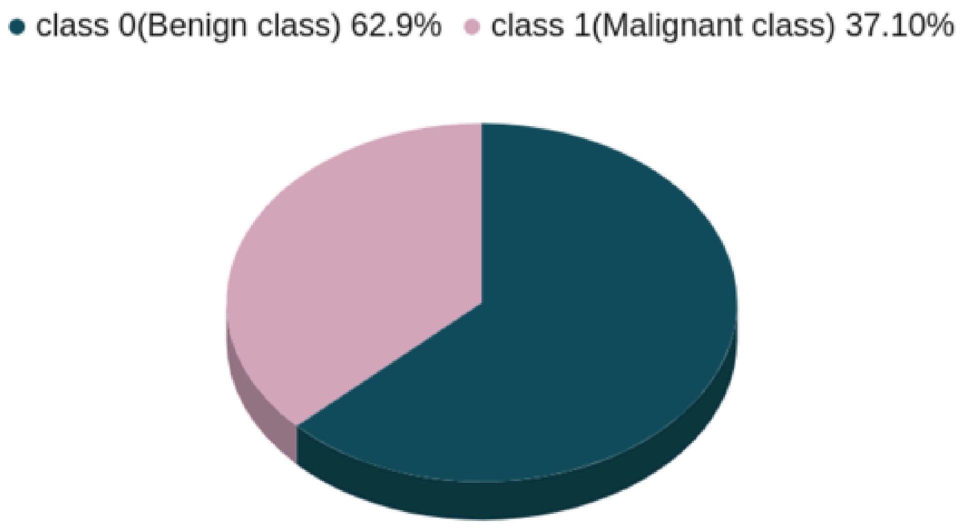


Fig. 14

5.2: Nodule segmentation

In the handle imbalancing section we pick some random negative cases and duplicate our positive cases, after picking up these specific points from the candidate.csv we have to crop out the nodule part from the CT scan images using those coordinates. To detect lung cancer from a CT scan image first we have to find the nodules. A nodule can be benign or malignant we have to classify or identify if a nodule is malignant or not if we find one or more than one nodules from a CT scan that are malignant then we can say that the person with this CT scan is affected by lung cancer.

We are trying to identify if a nodule is malignant or not for this purpose our region of interest is only the part of the CT Scan that contains the nodule part only.

5.2.1: Get the voxel coordinates: First, we have to read the image and then we have to get the voxel coordinates we cannot use the coordinate that is present in the candidate.csv because the coordinates given are real and not relative to the origin of the corresponding CT Scan.

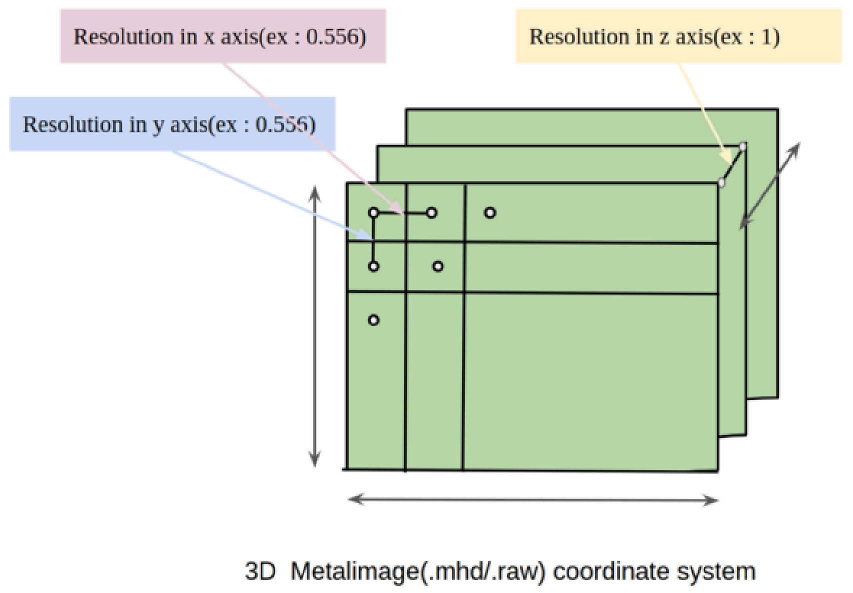

Resolution in x axis(ex : 0.556)
Resolution in z axis(ex : 1)
Resolution in y axis(ex : 0.556)
3D Metalimage(.mhd/.raw) coordinate system

Fig. 15

Here in Fig. 15 resolution refers to ElementSpacing in the z coordinate of the mhd file, and origin refers to the Offset object of the mhd file. To calculate the voxel coordinate we have to subtract the origin of the corresponding coordinate from the coordinate given in the candidate.csv.
Let's take a reference of one CT scan mhd file:
1.3.6.1.4.1.14519.5.2.1.6279.6001.100621383016233746780170740405.mhd which is present in the subset2.
origin offsets: $x = -120.049, \quad y = 9.48, \quad z = -657$

coordinates: $x = 89.32, \quad y = 190.84, \quad z = -516.82$

ElementSpacing ort resolution: $x = 0.556, \quad y = 0.556, \quad z = 1$

voxel_coord: $x = \frac{89.32 - (-120.049)}{0.556} = 376.56$

$$y = \frac{190.84 - 9.48}{0.556} = 326.18$$

$$z = \frac{-516.82 - (-657)}{1} = 140.18$$

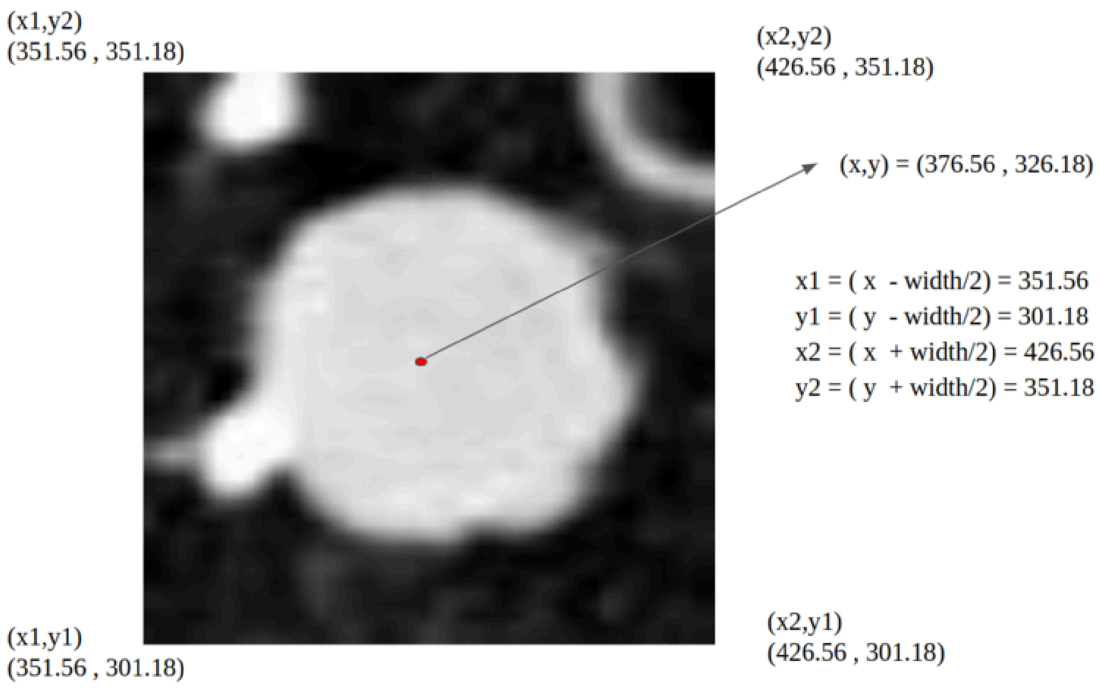
5.2.2: Crop the sub-image using the voxel coordinates: To get the sub-image from the raw file we have to first create the voxel coordinate, which we have already done in the previous paragraph. From the 3D raw image file, we have to first find the slice number from which the data will be cropped out. To find the slice number we have to know the value of the z coordinate from voxel_coord and we have to know the slice thickness of the CT scan data, which is present in the z coordinate of the ElementSpacing in the mhd files. Using this thickness we calculate the z coordinate in our test voxel_coord example where the z coordinate comes out to be 140.18, we can approximately say that this nodule is present in the 140th slice of the CT scan. Every CT scan image of our dataset contains more than 100 slices and the total number of slices can vary between 100 to 350. Now we already know the slice number we have to find the four coordinates from where we have to crop out our region of interest.
the x and y we have are the centers of our nodule or our region of interest, and the size of the image that we are cropping out is 50×50 Why we are taking this size is discussed in the next paragraph. To crop out the image we also need this image width. The vox_coord comes out to be as. $x = 376.56 \ and \ y = 326.18$. To crop out rectangle boxes, we need 4 coordinates.

$$x1 = (x - \frac{width}{2}) = 376.56 - \frac{50}{2} = 351.56$$

$$y1 = (y - \frac{width}{2}) = 326.18 - \frac{50}{2} = 301.18$$

$$x2 = (x + \frac{width}{2}) = 376.56 + \frac{50}{2} = 426.56$$

$$y2 = (y + \frac{width}{2}) = 326.18 + \frac{50}{2} = 351.18$$


(x1,y2)
(351.56 , 351.18)
(x2,y2)
(426.56 , 351.18)
(x,y) = (376.56 , 326.18)
x1 = (x - width/2) = 351.56
y1 = (y - width/2) = 301.18
x2 = (x + width/2) = 426.56
y2 = (y + width/2) = 351.18
(x1,y1)
(351.56 , 301.18)
(x2,y1)
(426.56 , 301.18)

Crop out the subimage

Fig. 16

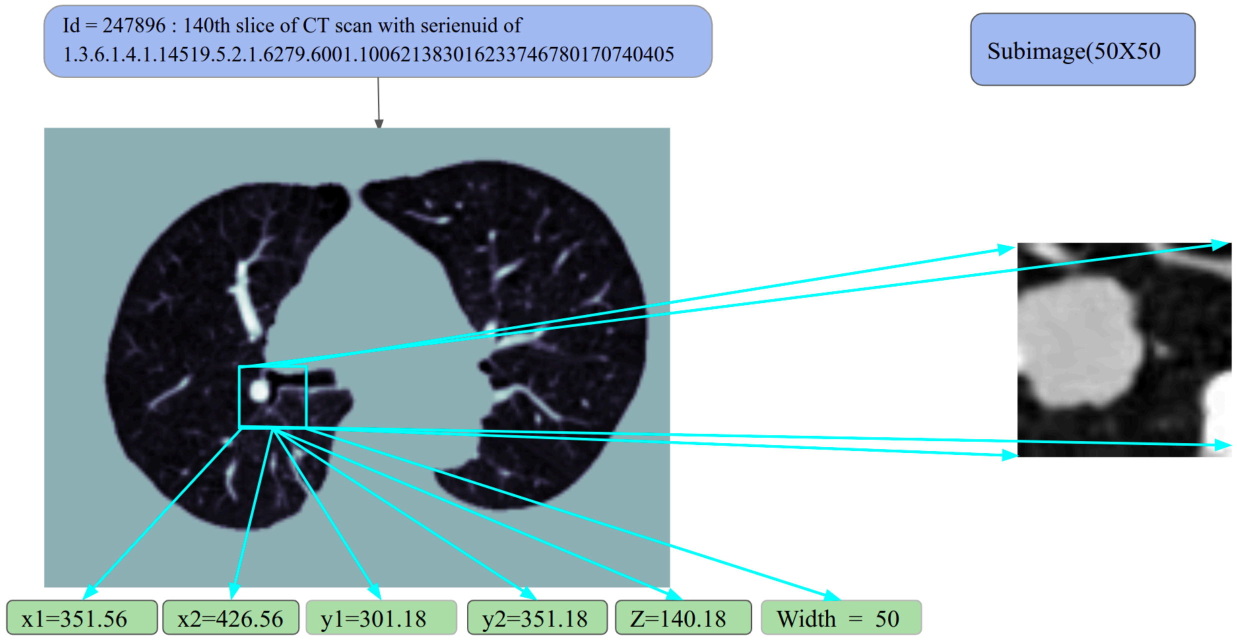


Fig. 17

5.2.3: Normalization: The sub-image that we have cropped from the CT scan RAW data stores the Hounsfield unit value as pixels, each containing the Hounsfield value of that place. We discussed the Hounsfield unit value in detail in chapter2. To process an image we need its RGB value or a value between 0 to 255 but the Hounsfield unit value ranges from -1000 to 700. To process the image we have to normalize the pixel value. By default, the CT scan images come as grayscale images. Furthermore, a number of elements can be present except the lung tissue in a CT scan like bone, air, and water which is also discussed in chapter 2. Our region of interest is lung nodules which will grow in lung tissues so, we need only lung tissue to accomplish this work.
Lung tissue has a Hounsfield unit value of -500, we want the pixel with a -500 value to get more attention. Furthermore, most of the pixels lay in this range.
We apply a simple normalization rule before cropping the image.

$$maxHU = 400$$

$$minHU = -1000$$

$$Pixel_{value} = \frac{Pixel_{value} - minHU}{maxHU - minHU}$$

$$if\ Pixel_{value} > 1\ \ then\ \ Pixel_{value} = 1$$

$$if\ Pixel_{value} < 0\ then\ Pixel_{value} = 0$$

$$Finally\ Pixel_{value} = Pixel_{value} \times 255$$

if we calculate the value of the pixel for some specific Hounsfield unit values:
Pixel_value = -500(lung tissue)

$$Pixel_{value} = \frac{-500 - (-1000)}{400 - (-1000)} => 0.3571 => 0.3571 \times 255 => 91.0714$$

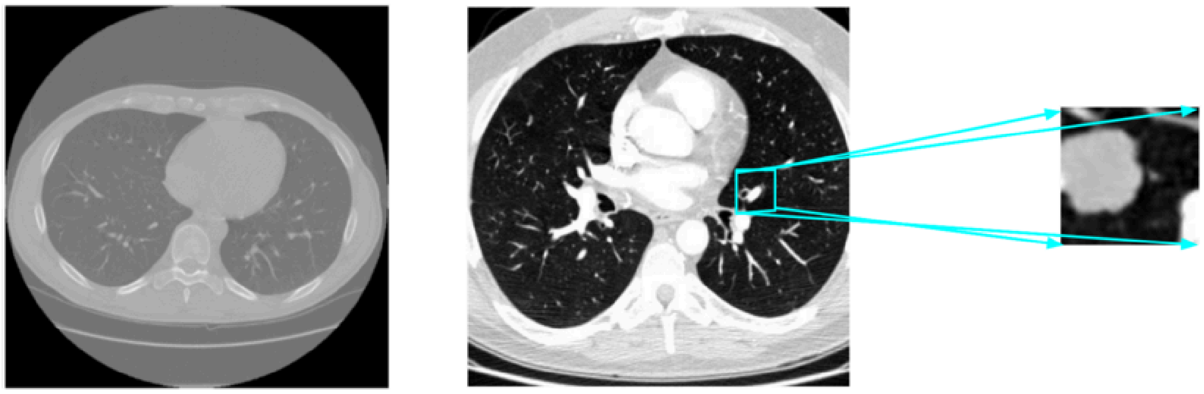
Pixel_value = -1000(air)

$$Pixel_{value} = \frac{-1000 - (-1000)}{400 - (-1000)} => 0 => 0\text{(black pixel)}$$

Pixel_value = 700(bone

$$Pixel_{value} = \frac{700 - (-1000)}{400 - (-1000)} => 1.214 => 1\ (white\ pixel)$$

This normalization technique converts all the pixels which contain lung tissue to gray color; any pixel which contains air or bone material simply converts it to black and white.



CT scan slice Without Normalization   CT scan slice after Normalization   Subimage(50X50)

Fig. 18

Why 50 X 50?

The below pictures give you some insights into the pixel and mm relation. A standard lung CT scan slice size is 360 X 360 mm. Total lung CT scan slice size in 512 X 512-pixel with this total image nearly 352 X 352-pixel data contain lung data and the rest 80 pixels on both sides have no information. In this project, we want to detect the malignant nodules in the early stages, the first stage or the second stage itself. In the first stage, the lung nodule size will always be less than 30 mm.

Different stages of lung cancer	mm
stage IA1	5
stage IA2	15
stage IA3	25
stage IB	30
stage IIA	40

A 50 X 50 cropped image will cover a 35.15 X 35.15 area which exactly covers all the nodules of the stage1 and we have the center of the nodule if any bogger nodule appears then also we can detect it easily if there is a nodule of 5 mm and we take the sub-image size larger than 50 then it will be the case that there will be a lot more noise other than nodule then our model will not work properly work for the very early stage which we want to achieve. Early detection is the only way to diagnose lung cancer in a better way. For more prominent nodules, some parts will be missing but malignant nodules will be present in the sub-image, and anyway, a nodule with a size of 35mm or greater will be easy to detect and easy to classify.

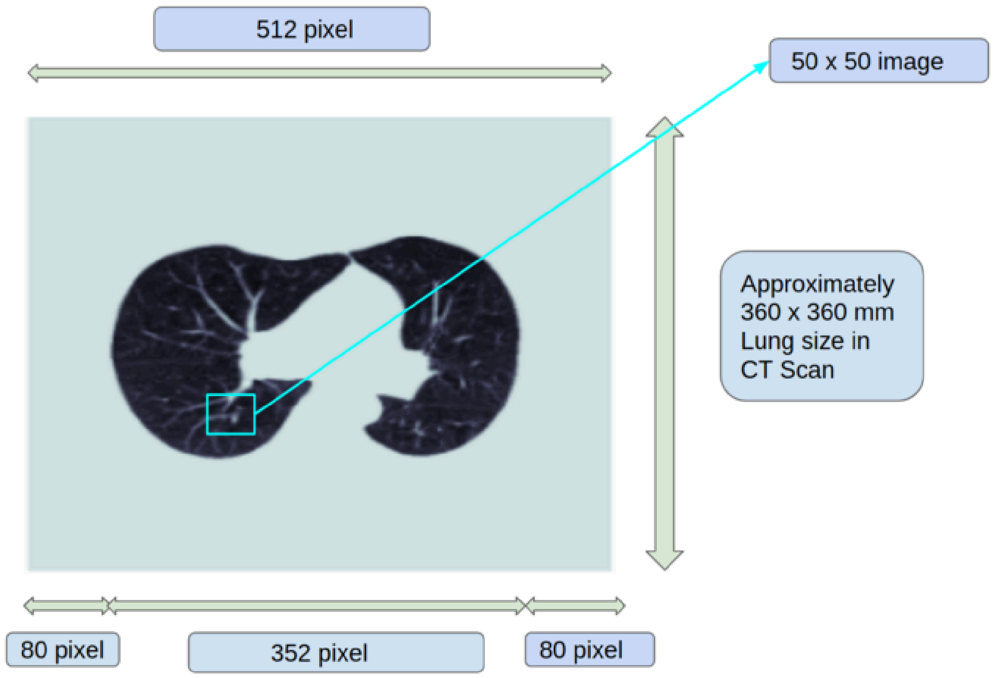


Fig. 19

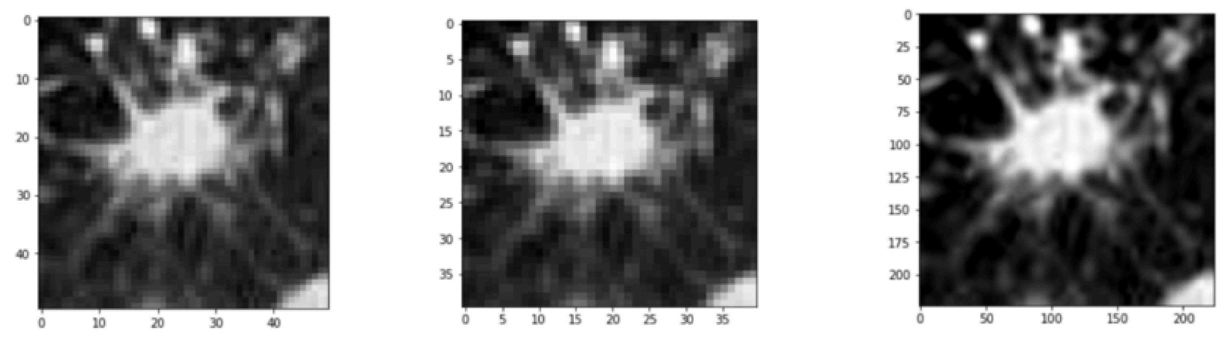
Preprocessing: As it's a CT scan data and it produced the grayscale image it has less noise and after normalization, it didn't need much-preprocessing steps as a normal RGB image does. We use simple preprocessing steps as follows:

Subtract mean: In this preprocessing step we take each pixel value which is between 0 to 255 that we have already done in the normalization step, we divide the pixel value by 255 and subtract the mean value which is 0.25. Most of the project papers that we observe on this topic take 0.25 as the mean value in the preprocessing step. According to medical physics lung, a CT scan has a mean pixel value of 0.25 or most of the normalized pixel values lie in the nearest area of value 0.25.

$$Pixel_{value} = (\frac{Pixel_{vale}}{255} \star 0.25) \star 255$$

Downsample data: We downsample data to the size of 40 × 40 using INTER_AREA interpolation. INTER_AREA pixel area relation for resampling. It is the best-suited method for reducing the size of the image or shrinking the image. here we reduce our image size from 50 × 50 to. 40 × 40.

Upsample data: After downsampling, we upsample the data. For upsampling, we use INTER_CUBIC interpolation which uses bicubic interpolation over a 4X4 pixel neighborhood; this combination of downsampling and then upsample will help remove the small distortion that is present in the CT scan. The presence of a very small component is not important for our training.



Subtract mean (50 X 50)   Downsample data (40 X 40)   Upsample data (224 X 224)

Fig. 20

Network architecture and training phase:

To accomplish this problem we try several pre-existing neural networks with a great level of modification with that and we will present only the network which achieves the highest accuracy. The network that achieves the highest accuracy is a great modification above the transfer learning VGG16 model. Below we presented a diagram of our network. We use a deep neural network to predict the class label of the nodules. The proposed model has 16 layers that have weights.

The original VGG16 model also has 16 weighted layers; hence we call the model a modified VGG16. In our case, the image size is 50 × 50 × 1, whereas, in the case of VGG16, the image size is 224 × 224 × 3. We use batch normalization to speed up the training process and remove minor distortion in the image to avoid overfitting.

The LUNA 16 dataset has only 0.24% malignant candidate nodules, which may lead to the overfitting of the network. We introduce dropouts to prevent overfitting. The architecture of the networks is summarized in Fig. 21.

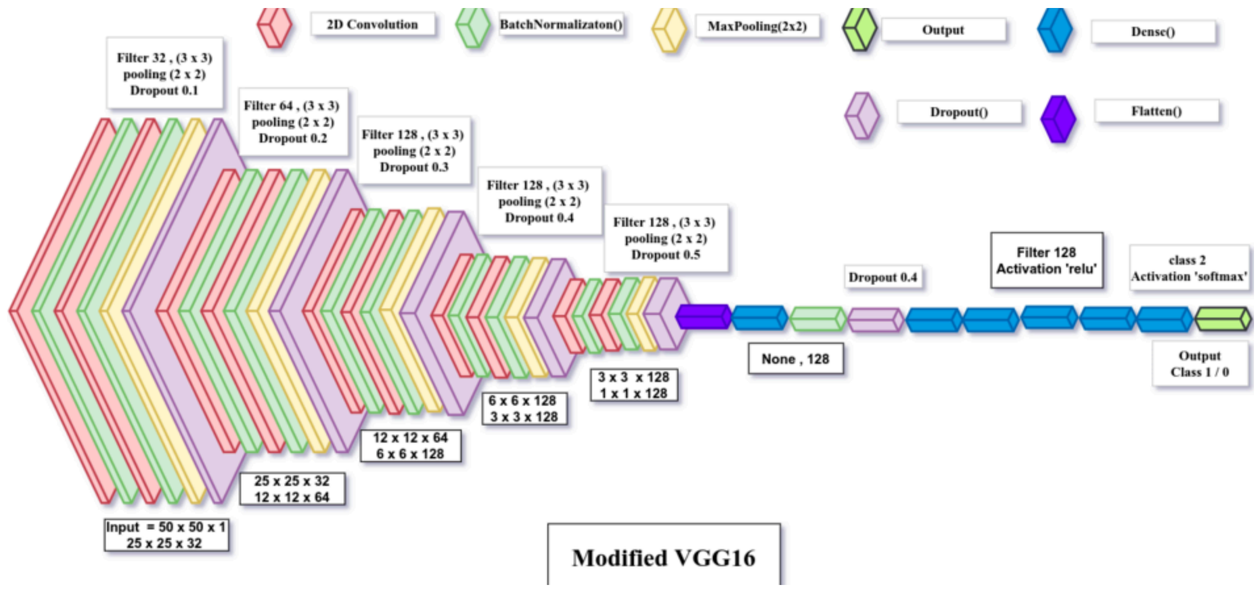
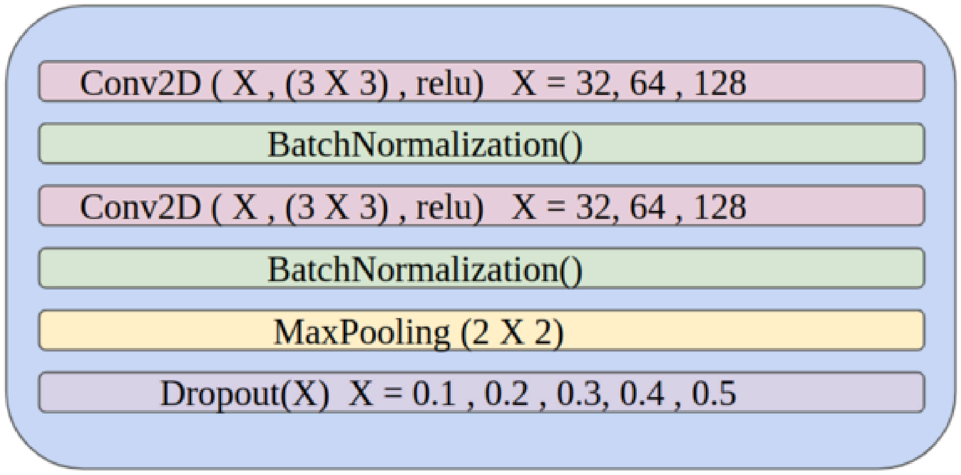

2D Convolution
BatchNormalizaton()
MaxPooling(2x2)
Output
Dense()
Dropout()
Flatten()
Filter 32 , (3 x 3) pooling (2 x 2) Dropout 0.1
Filter 64 , (3 x 3) pooling (2 x 2) Dropout 0.2
Filter 128 , (3 x 3) pooling (2 x 2) Dropout 0.3
Filter 128 , (3 x 3) pooling (2 x 2) Dropout 0.4
Filter 128 , (3 x 3) pooling (2 x 2) Dropout 0.5
Dropout 0.4
Filter 128 Activation 'relu'
class 2 Activation 'softmax'
None , 128
Output Class 1 / 0
3 x 3 x 128 1 x 1 x 128
6 x 6 x 128 3 x 3 x 128
12 x 12 x 64 6 x 6 x 128
25 x 25 x 32 12 x 12 x 64
Input = 50 x 50 x 1 25 x 25 x 32
Modified VGG16

Fig. 21

Model: if we describe the model we use 5 blocks of the same kind with different hyperparameters in each of the five blocks we have the same structure shown as below.


Conv2D (X , (3 X 3) , relu) X = 32, 64 , 128
BatchNormalization()
Conv2D (X , (3 X 3) , relu) X = 32, 64 , 128
BatchNormalization()
MaxPooling (2 X 2)
Dropout(X) X = 0.1 , 0.2 , 0.3, 0.4 , 0.5

Each block structure

Fig. 22

Each of the five-block has six layers each 2D convolutional has a batch normalization below it and in each block, at last, we have one dropout which has different hyperparameters in each block one MaxPooling(2×2) layer. 2D convolutional layer in each block have filter size of 3×3 and we use ReLu as activation function in the convolution step and we also use he_uniform as kernel initializer.

After these five blocks, we have got a feature map of the size $1 \times 1 \times 128$ We flatten this feature map and then apply one round of the Dense layer and then the BatchNormalization layer. Then we apply five consecutive Dense layers and then finally the output layer. Here we use a total of 11 Batch Normalization layers which helps us with internal covariate shift and normalize image data after each convolution layer and makes our model work efficiently with imbalanced data and also helps us avoid overfitting and makes our model more flexible toward new data and it also speeds up the training process. We also use the dropout layer to a great extent to avoid the overfitting problem. We built our model in such a way that it did not cause any overfitting as our data itself is prone to it, because the data is imbalanced from the very first point. We also focussed on making our model take less time. Each of the layers is discussed in great detail from the point of view of this project in chapter 4.

Augmentation and training:

Before training, we augment the training data because after handling imbalanced of the data size reduced too much we use some general augmentation techniques which are shown below:

```
width_shift_range=0.1,
height_shift_range=0.1,
horizontal_flip=True,
zoom_range=0.1,
vertical_flip =True,
rotation_range = 20,
featurewise_center = False
```

Here we use small changes in range because the image is small and by doing these things we may miss classifying some of the early nodules or comparatively smaller nodules.
In the output layer we use, $Adam\ optimizer\ with\ 0.0005\ learning\ rate$ we use $categorical\ cross - entropy$ as loss function, we use $number\ of\ epochs\ = \ 300$ and . $batch\ size\ = \ 64$. Where Adam is the combination of RMSProp and stochastic gradient descent algorithms which make our model converge in a better and faster way

this topic is thoroughly discussed in chapter4. The categorical cross-entropy function has also been described in detail in chapter 4. Epoch indicate the number of passes through your total batch, in one epoch total batch of image passthrough the model and will produce some modification to the existing weights and after each epoch, it calculates the loss for the total training dataset and it also calculates the loss for the 20% validation set. Modifying the weights depending on the training loss can be inefficient and lead to the overfitting of the model. overfitting means your data works properly with the training data but for the test set and validation set it didn't work well.

Overfitting will occur only when you modify your weights depending on the loss happening on the training set only, the weights getting modified in this way that it only works well for the training or data for any new data or test data or validation data it will not show the as good result as the training set. To avoid this problem we add a part of data to the validation set and after each epoch, our model will calculate the validation loss, and depending on that loss it will make changes in weights. This modification will also work well with any new data.

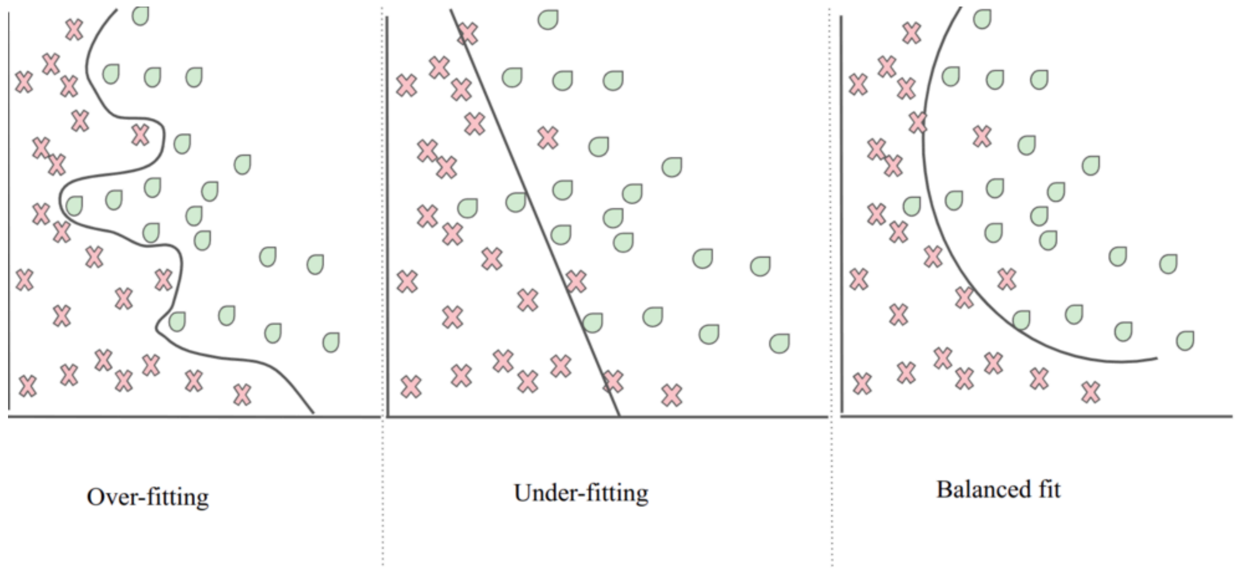


Fig.

Fig. shows you how overfitting modifies the weights and makes a really difficult polynomial curve to somehow fit the training set in it, it won't work well with the new datas. Here we take a simple polynomial case to describe the overfitting problem.

The below figure shows you how validation and training accuracy changes with each epoch. you see a certain gap between them as the epoch increases the weights will be more dependent on the training set increasing the epoch will give you less accuracy we find 300 as a threshold for it.

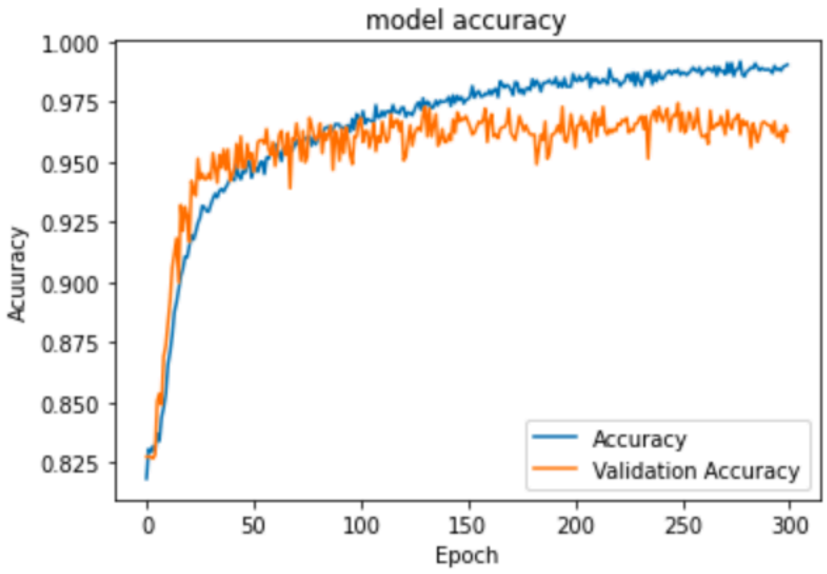
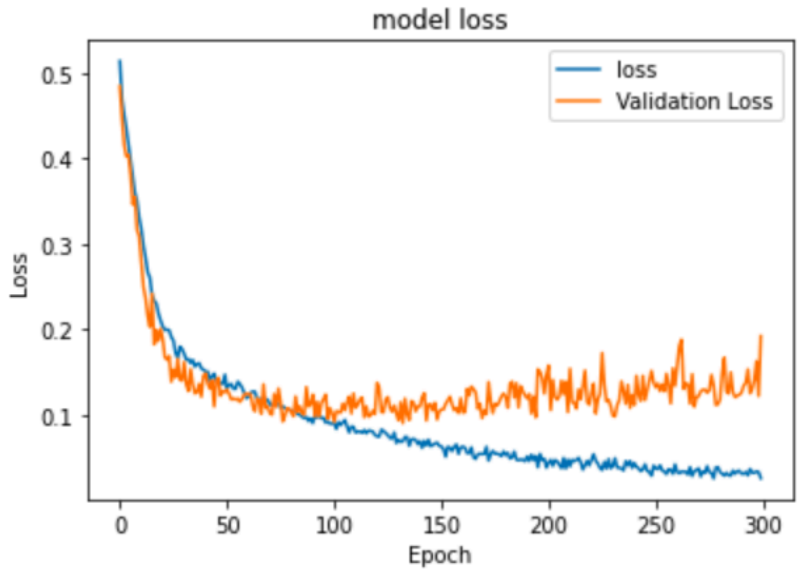


Fig.

Also check the validation loss and training loss difference in below graph:



you can observe that validation loss fluctuates while the training loss value is lower and stable also. We can conclude that there is small overfitting that is present and without

Chapter 6
Experimental results and conclusion

In this chapter, we want to discuss the results produced by our model. We will further discuss the precision, and recall of our classes and also mention the sensitivity and specificity of the model, and also calculate the accuracy of the model.

Precision and recall: These two are also called performance matrices which are used to evaluate the performance of the classification task.

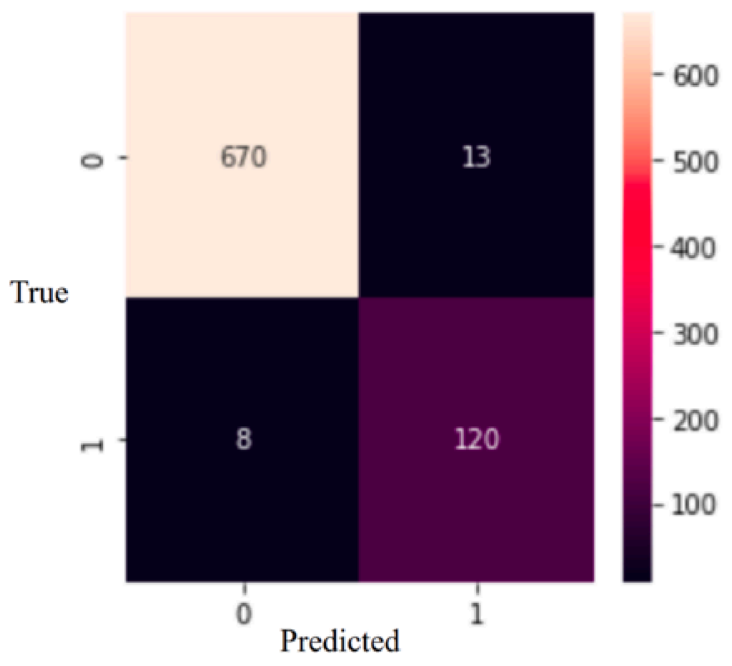
True positive: This combination tells us how many times a positive class is correctly predicted as positive by our model.

True negative: This combination tells us how many times a negative class is correctly predicted as negative by our model.

False positive: This combination tells us how many times a negative class is incorrectly predicted as positive by our model.

False negative: This combination tells us how many times a positive class is incorrectly predicted as negative by our model.

Our main goal here is to reduce the false negative class or the cases where positive classes are misclassified as negative classes. In our data, the number of cases of negative classes is higher than the number of positive cases. A positive class is prone to misclassifying. To reduce the false negative class we add too many BatchNormalization and Dropout layers in our model. Below in Fig., you can see the confusion matrix of our model. The total number of samples in the 10% test set is 811 among them 683 cases are negative and 128 cases are truly positive. Nearly 18% of the total samples are positive cases. in the confusion matrix, 1 means the positive class or malignant nodule and 0 means the benign class or non-malignant class.



Confusion Matrix

Fig.

Precision: precision is the ratio of correctly classified samples of a specific class to the total number of samples that are predicted to be in the corresponding class either correctly or incorrectly classified. In simple terms for a class, the precision of that class is the ratio of correctly classified samples to all those samples predicted to be in the class. for positive class, precision will be calculated as:

$$precision_{positive} = \frac{True\ positive}{True\ positive\ +\ False\ positive} = \frac{True\ positive}{Total\ number\ of\ samples\ predicted\ to\ be\ positive}$$

$$precision_{positive} = \frac{120}{120\ +13} = 0.9022 \approx 90\%$$

precision for negative class calculated as

$$precision_{negative} = \frac{True\ negative}{True\ negative +\ False\ negative} = \frac{True\ negative}{Total\ number\ of\ samples\ predicted\ to\ be\ negative}$$

$$precision_{negative} = \frac{670}{670+8} = 0.9882 \approx 99\%$$

Recall: recall is the ratio of correctly classified samples of a specific class to the total number of samples that are true to be in the corresponding class. In simple terms for a class, the recall of that class is the ratio of correctly classified samples to all those samples truly to be in that class. for positive class, the recall will be calculated as

$$recall_{positive} = \frac{True\ positive}{True\ positive\ +\ False\ negative} = \frac{True\ positive}{Total\ number\ of\ samples\ predicted\ to\ be\ positive}$$

$$recall_{positive} = \frac{120}{120\ +8} = 0.9375 \approx 94\%$$

precision for negative class calculated as

$$recall_{negative} = \frac{True\ negative}{True\ negative +\ False\ positive} = \frac{True\ negative}{Total\ number\ of\ samples\ predicted\ to\ be\ negative}$$

$$recall_{negative} = \frac{670}{670\ +13} = 0.9809 \approx 98\%$$

Sensitivity and specificity: These two terms are also used as medical terms: sensitivity means how many positive cases or disease cases are correctly predicted among all positive cases and specificity means how many negative cases or healthy cases are correctly predicted among all negative cases.

sensitivity: we can also describe it as a recall for positive class. It is the ratio of correctly predicted positive class samples to all truly positive classes. My model sensitivity can be calculated as

$$Sensitivity = \frac{True\ positive}{True\ positive\ +\ False\ negative} = \frac{True\ positive}{Total\ number\ of\ samples\ predicted\ to\ be\ positive}$$

$$Sensitivity = \frac{120}{120\ +8} = 0.9375 \approx 94\%$$

specificity: we can also describe it as a recall for a negative class. It is the ratio of correctly predicted negative class samples to all truly negative classes. My model specificity can be calculated as

$$Specificity = \frac{True\ negative}{True\ negative +\ False\ positive} = \frac{True\ negative}{Total\ number\ of\ samples\ predicted\ to\ be\ negative}$$

$$Specificity = \frac{670}{670\ +13} = 0.9809 \approx 98\%$$

Accuracy: overall accuracy of our model would be calculated using the weighted average of F1 score of each class. where F1 score calculated as:

$$F1\ score_{\ positive} = \frac{2 \times (\ precisionn_{positive} \times\ recall_{positive})}{(precision_{positive} + recall_{positive})} = \frac{2 \times (0.9022 \times\ 0.9375)}{(0.9022 + 0.9375)} = 0.9195 \approx 92\%$$

$$F1\ score_{negative} = \frac{2 \times (\ precision_{negative} \times\ recall_{\ negative})}{(precision_{negative} + recall_{negative})} = \frac{2 \times (0.9882 \times\ 0.9809)}{(0.9882 + 0.9809)} = 0.9845 \approx 98\%$$

$$Accuracy = weighted\ average\ of\ F1\ scores$$

$$Total\ support = 811$$

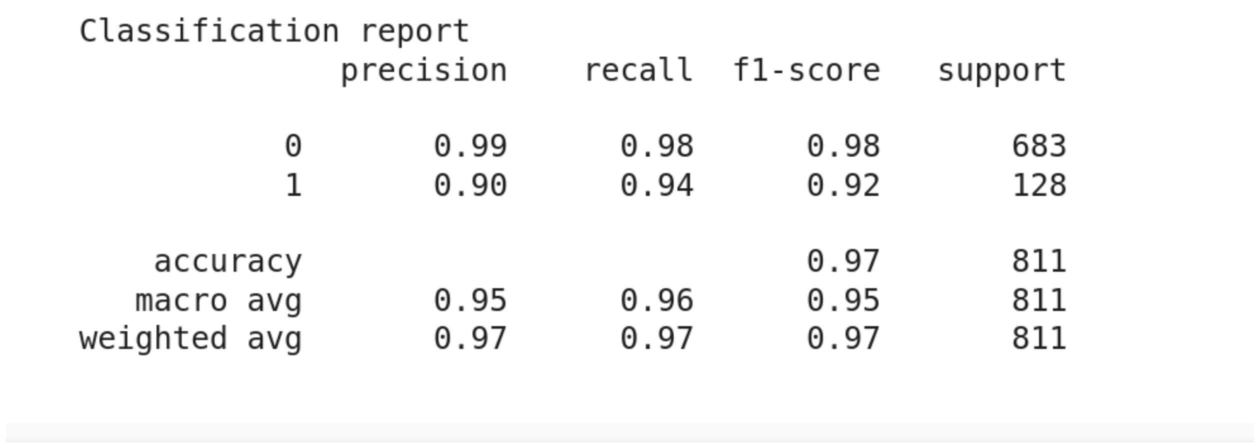
$$Support\ in\ negative\ class = 683$$

$$Support\ in\ positive\ class = 128$$

$$Accuracy = \frac{(\ Support\ in\ negative\ class \times F1\ scoree_{negative} + Support\ in\ positive\ class \times Fi\ score_{positive})}{Total\ support}$$

$$Accuracy = \frac{(683 \times 0.9845 + 128 \times 0.9195}{811} = 0.9742 \approx 97\%$$

In fig. we attached the classification report:



```
Classification report
              precision    recall  f1-score   support

           0       0.99      0.98      0.98       683
           1       0.90      0.94      0.92       128

    accuracy                           0.97       811
   macro avg       0.95      0.96      0.95       811
weighted avg       0.97      0.97      0.97       811
```

Fig.

In this report, we explain all the things in the upper part except the macro average which simply means the score of all classes.

Results: We have got an overall accuracy of 97% which is quite decent and in the final epoch the loss on the training set is 0.0319 and the validation loss is 0.1281. The highest accuracy that we got in the training set is nearly 99% and the accuracy on the validation set that we got is nearly 97%. Accuracy and loss changes in training and validation set has shown below:

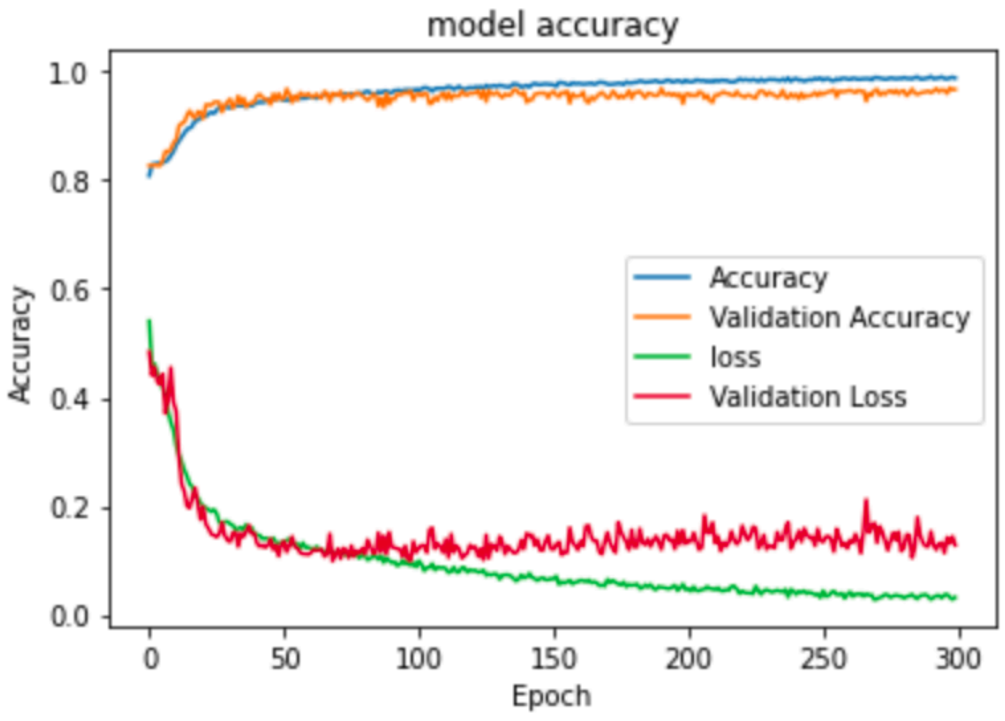


Fig.

Test data Performance analysis table:

Class	Precision	Recall	F1 score	Accuracy	Specificity	Sensitivity
Benign-Nodules	**99%**	**98%**	**98%**	**97%**	**98%**	**94%**
Malignant-Nodule	**90%**	**94%**	**92%**			

Comparision with other existing methods:

Method	Dataset	Accuracy	Sensitivity
Hamdalla method[13]	IQ-OTH/NCCD	93.45%	95.15%
Setio method[14]	LIDC-IDRI	93.5%	90.1%
Xie method[15]	LUNA16	95.0%	86.42%
Kumar method[8]	LIDC-IDRI	75.01%	83.35%
W.Shen method[10]	LIDC-IDRI	93%	77%
Proposed Method	LUNA16	97%	94%

Here in the comparison table we try to compare some 2D and 3D existing model with our proposed method. There is a very less number of existing 2DCNN methods which is applied in the LUNA16 dataset. You can notice we make certain improvements in accuracy as well as sensitivity also.

Chapter 7
Conclusion and Future work

In conclusion, we can say that we build a custom model using a 2D convolutional neural network named modified VGG16 to train our LUNA16 dataset we classify a nodule present on the lung tissue as benign or malignant of a nodule comes out to be malignant then the person with that corresponding CT scan has been affected with lung cancer. Our model is faster than any 3D CNN method as the 3D model takes a huge amount of data also to achieve a good result with a 3D model we need a CT scan with a decent number of slices. Where our model can make predictions based on only one slice. In India-like countries, the average number of slices present in a CT scan is around 16, and to work with a 3D model which is trained on LUNA16 or LIDC-IDRI dataset will need a CT scan with a number of slices not less than 100.
We gained an accuracy of our model of 97% and the model sensitivity is 94% on test data. We are also able to reduce the computation time for the training process by modifying our model. We are also able to reduce the time taken by the training process. Also, we have gained a higher specificity of 98% which will reduce the number of biopsies referred by specialists.

Here we use the candidate nodules and the coordination mentioned in the candidate.csv. If a data has no such annotation we will have to face a lot of problems solving that issue. In our future work, we want to work with the nodule detection a for that we use the annotation .csv and candidate.csv to generate the mask images. We have already prepared the mask image by dividing the data into 10 different parts for RAM constraints and the size of the whole data is also too big to handle. The training part is left which will again take a lot of computational power. To train this object detection we will be going to use a 2D convolutional UNet model. We will not cover the object detection part in this thesis because it includes a lot of details and we will not be able to cover all the things in one thesis in a nicer way.

Chapter 8
Bibliography

[1] S. G. Armato, III, M. L. Giger, C. J. Moran, J. T. Blackburn, K. Doi, and H. MacMahon, Computerized detection of pulmonary nodules on CT scans, vol. 19, no. 5, pp. 1303 - 1311, 1999

[2] A. Masood et al., Automated Decision Support System for Lung Cancer Detection and Classification via Enhanced RFCN With Multilayer Fusion RPN, in IEEE

[3] Transactions on Industrial Informatics, vol. 16, no. 12, pp. 7791-7801, Dec. 2020.3. Shakeel, P.M., Burhanuddin, M.A. & Desa, M.I. Automatic lung cancer detection from CT image using improved deep neural network and ensemble classifier, Neural Comput & Applic (2020).

[4] Suren Makaju, P.W.C. Prasad, Abeer Alsadoon, A.K. Singh, A. Elchouemi, Lung Cancer Detection using CT Scan Images, Procedia Computer Science, Volume 125, 2018, pp. 107-114.

[5] Akter, O., Moni, M.A., Islam, M.M. et al. Lung cancer detection using enhanced segmentation accuracy, Appl Intell 51, pp. 3391-3404 (2021).

[6] R. L. Siegel, K. D. Miller, and A. Jemal, Cancer statistics, 2019, CA, Cancer J. Clinicians, vol. 69, no. 1, pp. . 7-34, 2019.

[7]Cancer Facts and Figures 2020. Atlanta: American Cancer Society. [Cited 2020 May 02]. Available from: https://www.cancer.org/research/cancer-facts-statistics/all-cancer-facts-figures/cancer-facts-figures-2020.html

[8] Kumar, D.; Wong, A.; Clausi, D.A., Lung Nodule Classification Using Deep Features in CT Images, In Proceedings of the 2015 12th Conference on Computer and Robot Vision, Halifax, NS, Canada, 3-5 June 2015; pp. 133-138.

[9] A. A. A. Setio et al., Validation, comparison, and combination of algorithms for automatic detection of pulmonary nodules in computed tomography images: The

LUNA16 challenge, Medical Image Analysis, vol. 42, pp. 1-13, 2017.

[10] W. Shen et al., Multi-crop convolutional neural networks for lung nodule malignancy suspiciousness classification 10, Pattern Recognit., vol. 61, pp. 663-673, 2017.

[11] A.Masood et al., Cloud-based automated clinical decision support system for de-detection and diagnosis of lung cancer in chest CT, IEEE Journal of Translational Engineering in Health and Medicine, vol. 8, 2020, Art. no. 4300113.

[12] H. Jiang, H. Ma, W. Qian, M. Gao, and Y. Li, An automatic detection system of lung nodule based on multigroup patch-based deep learning network, IEEE Journal of Biomedical and Health Informatics, vol. 22, no. 4, pp. 227-1237, 2018.

[13] Fadil, Hamdalla and Al-Huseiny, Muay and Mohsen, Furat and Khalil, Enam and Hassan, Zainab, Diagnosis of Lung Cancer Based on CT Scans Using CNN, In Proceedings of the Conference Series: Materials Science and Engineering, pp 022-035, 2020.

[14] A. A. A. Setio et al, Pulmonary Nodule Detection in CT Images: False Positive Reduction Using Multi-View Convolutional Networks, IEEE Transactions on Medical Imaging, vol. 35, no. 5, pp. 1160-1169, 2016

[15] H. Xie et al., Automated pulmonary nodule detection in CT images using deep convolutional neural networks, Pattern Recognition, vol. 85, pp. 109-119, 2019.

[16] Zhu W., Liu C., Fan W., Xie X. DeepLung: Deep 3D dual path nets for automated pulmonary nodule detection and classification; Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV); Lake Tahoe, NV, USA. 12–15 March 2018; pp. 673–681. [Google Scholar]

[17] Ren S., He K., Girshick R., Sun J. Faster R-CNN: Towards real-time object detection with region proposal networks. IEEE Trans. Pattern Anal. Mach. Intell. 2017;39:1137–1149. DOI: 10.1109/TPAMI.2016.2577031. [PubMed] [Google Scholar]

[18] Masood A., Sheng B., Li P., Hou X., Wei X., Qin J., Feng D. Computer-Assisted Decision Support System in Pulmonary Cancer detection and stage classification on CT

images. J. Biomed. Inform. 2018;79:117–128. DOI: 10.1016/j.jbi.2018.01.005. [PubMed] [Google Scholar]

[19] Silvestri G.A., Tanner N.T., Kearney P., Vachani A., Mission P.P., Porter A., Springmeyer S.C., Fang K.C., Midthun D., Mazzone P.J. Assessment of plasma proteomics biomarker's ability to distinguish benign from malignant lung nodules: Results of the PANOPTIC (Pulmonary Nodule Plasma Proteomic Classifier) trial. Chest. 2018;154:491–500. DOI: 10.1016/j.chest.2018.02.012. [PMC free article] [PubMed] [Google Scholar]

[20] Huang G., Liu Z., Van Der Maaten L., Weinberger K.Q. Densely connected convolutional networks; Proceedings of the IEEE conference on computer vision and pattern recognition; Honolulu, HI, USA. 21–26 July 2017; pp. 2261–2269. [Google Scholar]

[21] Cho., Li J., Xiao H., Jin X., Yan S., Feng J. Advances in Neural Information Processing Systems. NIPS; San Diego, CA, USA: 2017. Dual path networks; pp. 4467–4475. [Google Scholar]

[22] Wang W., Li X., Lu T., Yang J. Mixed link networks. aiXiv. 20181802.01808 [Google Scholar]

[23] Shi Z., Zhao J., Han X., Pei B., Ji G., Qiang Y. A new method of detecting pulmonary nodules with PET/CT based on an improved watershed algorithm. PLoS ONE. 2015;10:e0123694. [PMC free article] [PubMed] [Google Scholar]

[24] Nasrullah N., Sang J., Alam M.S., Xiang H. Pattern Recognition and Tracking XXX. International Society for Optics and Photonics; Bellingham, WA, USA: 2019. Automated detection and classification for early stage lung cancer on CT images using deep learning; p. 27. [Google Scholar]