SUMMER INTERNSHIP PROJECT REPORT

AN EFFECTIVE APPROACH TO DETECT LUNG CANCER ON CT SCAN IMAGES USING SEGMENTATION AND CONVOLUTIONAL NEURAL NETWORK

UNDER THE GUIDANCE OF

DR. AMAN KUMAR (ASSISTANT PROFESSOR)

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING , NATIONAL INSTITUTE OF TECHNOLOGY,HAMIRPUR - 177005 (HP.)

SUBMITTED BY:-

HARIT YADAV (17MI418)

DIVYANSHU BHAIK (17MI446)

ACKNOWLEDGEMENT

This project has seen contributions from various individuals. It has been an honor to work under our guide, Dr Aman Kumar, Assistant Professor,Department of Electronics and Communication,NIT Hamirpur. We are extremely thankful to him for his support and mentorship throughout the project. This project would not have had better supervisors than him.

We would also like to thank Dr. Aman Kumar for giving us an opportunity to work under their guidance and blessing.

Lastly, I would like to thank my family and friends for their kind support. I feel grateful to Lord Almighty who has showered His graces upon me during this period.

Table of contents **Page No.**

1. INTRODUCTION

Lung cancer is one of the deadliest cancers worldwide. However, the early detection of lung cancer significantly improves survival rate. Cancerous (malignant) and noncancerous (benign) pulmonary nodules are the small growths of cells inside the lung. Detection of malignant lung nodules at an early stage is necessary for the crucial prognosis [1]. Early-stage cancerous lung nodules are very much similar to noncancerous nodules and need a differential diagnosis on the basis of slight morphological changes, locations, and clinical biomarkers[2] . The challenging task is to measure the probability of malignancy for the early cancerous lung nodules [3]. Various diagnostic procedures are used by physicians, in connection, for the early diagnosis of malignant lung nodules, such as clinical settings, computed tomography (CT) scan analysis (morphological assessment), positron emission tomography (PET) (metabolic assessments), and needle prick biopsy analysis [4]. However, mostly invasive methods such as biopsies or surgeries are used by healthcare practitioners to differentiate between benign and malignant lung nodules. For such a fragile and sensitive organ, invasive methods involve lots of risks and increase patients' anxieties.

The most suitable method used for the investigation of lung diseases is computed tomography (CT) imaging [5]. However, CT scan investigation has a high rate of false positive findings, with carcinogenic effects of radiations. Low-dose CT uses considerably lower power radiation contact than standard-dose CT. The results show that there is no significant difference in detection sensitivities between low-dose and standard-dose CT images. However, cancer-related deaths were significantly reduced in the selected population that were exposed to low-dose CT scans as compared to chest radiographs, which is depicted in the National Lung Screening Trial (NLST) database [6]. The detection sensitivity of lung nodules improves with sophisticated anatomical details (thinner slices) and better image registration techniques. However, this increases the datasets to a very large extent. Depending upon the slice thickness, up to 500 sections/slices are produced in one scan [7]. An experienced radiologist takes approximately 2–3.5 min to observe a single slice [8]. The workload of a radiologist increases significantly to screen a CT scan for the possible existence of a nodule. In addition to section thickness of the CT slices, detection sensitivity also depends on nodule features such as size, location, shape, adjacent structures, edges, and density.

Results show that only 68% of the time lung cancer nodules are correctly diagnosed when only one radiologist examines the scan, and are accurately detected up to 82% of the time with two radiologists. The detection of cancerous lung nodules at an early stage is a very difficult, tedious, and time-consuming task for radiologists. Screening a lot of scans with care requires plenty of time by the radiologist, meanwhile it is very much error-prone in the detection of small nodules [9].

To handle this situation our first approach will be to make segmentation of CT scan images after preprocessing. We are using a watershed algorithm to make segmentation. Watershed algorithms make a mask for cancer cells in lung images. reference of an Article ("Serge Beucher and Fernand Meyer. The morphological approach to segmentation: the watershed transformation. In Mathematical Morphology in Image Processing (Ed. E. R. Dougherty), pages 433–481 (1993).")

After successful making masks next step is to build a model on VGG16 NET transfer learning model for better accuracy. At last both the trained and optimized model will be used and combined to make a complete final model for the classification of lung cancer. Taking reference of a paper "Deep learning for lung Cancer" by A. Asuntha & Andy Srinivasan.

For the input layer, lung nodule CT images are used and are collected for various steps of the project. The source of the dataset is the LUNA16 dataset .

The LUNA16 dataset is a subset of LIDC-IDRI dataset, in which the heterogeneous scans are filtered by different criteria. Since pulmonary nodules can be very small, a thin slice should be chosen. Therefore scans with a slice thickness greater than 2.5 mm were discarded. Furthermore, scans with inconsistent slice spacing or missing slices were also excluded. This led to 888 CT scans, with a total of 36,378 annotations by radiologists. In this dataset, only the annotations categorized as nodules $\geq$ 3 mm are considered relevant, as the other annotations (nodules $\leq$ 3 mm and non-nodules) are not considered relevant for lung cancer screening protocols . Nodules found by different readers that were closer than the sum of their radii were merged. In this case, positions and diameters of these merged annotations were averaged. This results in a set of 2290, 1602, 1186 and 777 nodules annotated by at least 1, 2, 3 or 4 radiologists, respectively. In Figure 1, different slices from a LUNA16 CT scan.

2. RELATED WORK

Although the first computer-aided detection (CAD) system for lung nodule detection was designed in the late 1980s, these attempts were not appealing due to inadequate computational resources for advanced image analysis techniques at that time. After the invention of the graphical processing unit (GPU) and convolutional neural networks (CNN), the performance of computer-based image analysis and decision support systems got a high boost. A lot of deep learning-based medical image analysis models have been proposed by researchers, and a few of the most relevant lung nodule detection and classification methods are mentioned here [10].

Setio et al. proposed a 3D fully convolutional neural network for FP reduction in lung nodule classification [11]. A 3D network was used to analyze the 3D nature of the CT scans to reduce wrong diagnosis, and weighted sampling was used to improve results.

Ding and Liao et al. used 3D Faster R-CNN for nodule detection to reduce false positive (FP) results of lung cancer diagnosis [12]. 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 [13].

Jiang Hongyang et al. designed group-based pulmonary nodule detection using multi patches scheme with 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 sensitivity of 80.06% with 4.7 FP for each scan, and sensitivity of 94% with an FP rate of 15.1.

Zhu Wentao et al. proposed automated lung nodule detection and classification models using 3D DPN with 3D faster R-CNN and gradient boosting machine (GBM) by learning spatial features of lung nodules [12]. After preprocessing of the whole chest CT scan, the lung volume was segmented. The segmented lung volume was analyzed by 3D faster R-CNN with DPN and U-Net-like encoder–decoder architecture for nodule detection. After detecting the nodules, suspected nodule regions were cropped to learn the finer level features of the nodules to classify with DPN and gradient boosting machines. The model shows a detection accuracy of 87.5% with an error rate of 12.5%.

Masood et al. proposed a deep fully convolutional neural network (DFCNet) for the detection and classification of pulmonary lung nodules in a CT image [15]. 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.

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.

Zhao J et al. proposed a new method for lung segmentation and nodule detection by combining the features from CT and PET images [4]. 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.

Dr. Silvestri and his research team have proposed proteomic classifiers, along with nodule features, to differentiate between small size benign and malignant lung nodules [3]. They have achieved very good results on 8–30 mm nodule sizes with a reduction of 40% in biopsies on benign nodules.

In [17], the authors Lee H., Matin T., Gleeson F., Grau V. Medical, proposed 3D-CNN for the classification of the volumetric benign and malignant lung nodules to reduce the false positive results in an automated lung nodule detection setup in the CT scans.

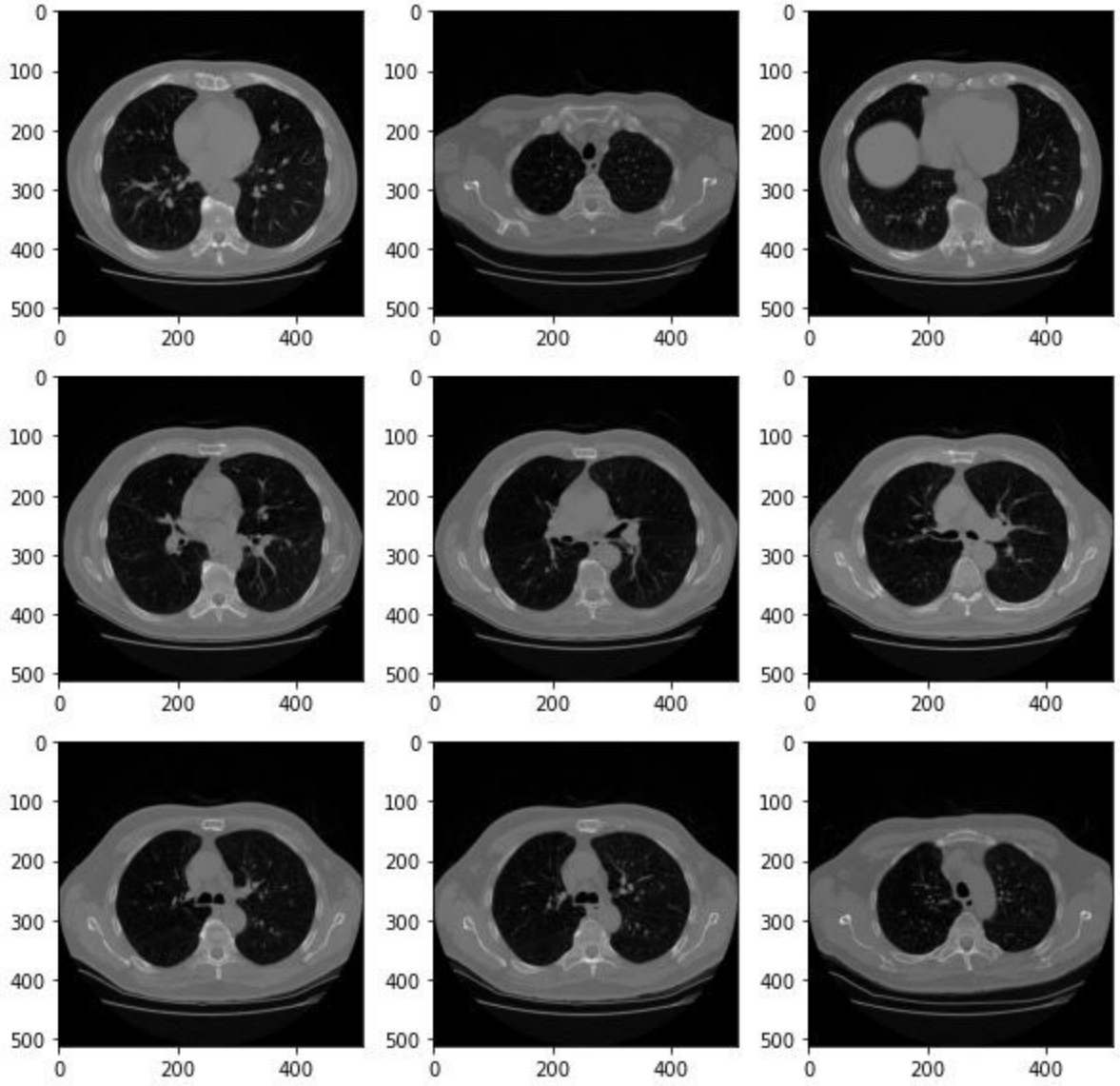
After the popularity of convolutional neural networks (CNNs) in image analysis, different types of connectivity patterns were proposed by researchers to increase the performance of deep CNNs. Up until now, in the deep CNNs, dense topology structures ResNet, DenseNet [18], and DPNs performance is superior as compared to other ones, but there is still room for connection improvements in these topologies [19]. The MixNet architecture has improved connection structures with better features of extraction and reduced parameter redundancy [20].

In some of the works [21], 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 lung cancers diagnosis. Therefore, a multi-strategy-based approach is needed for early-stage lung cancer detection.

3. VISUALIZATION OF DATASET

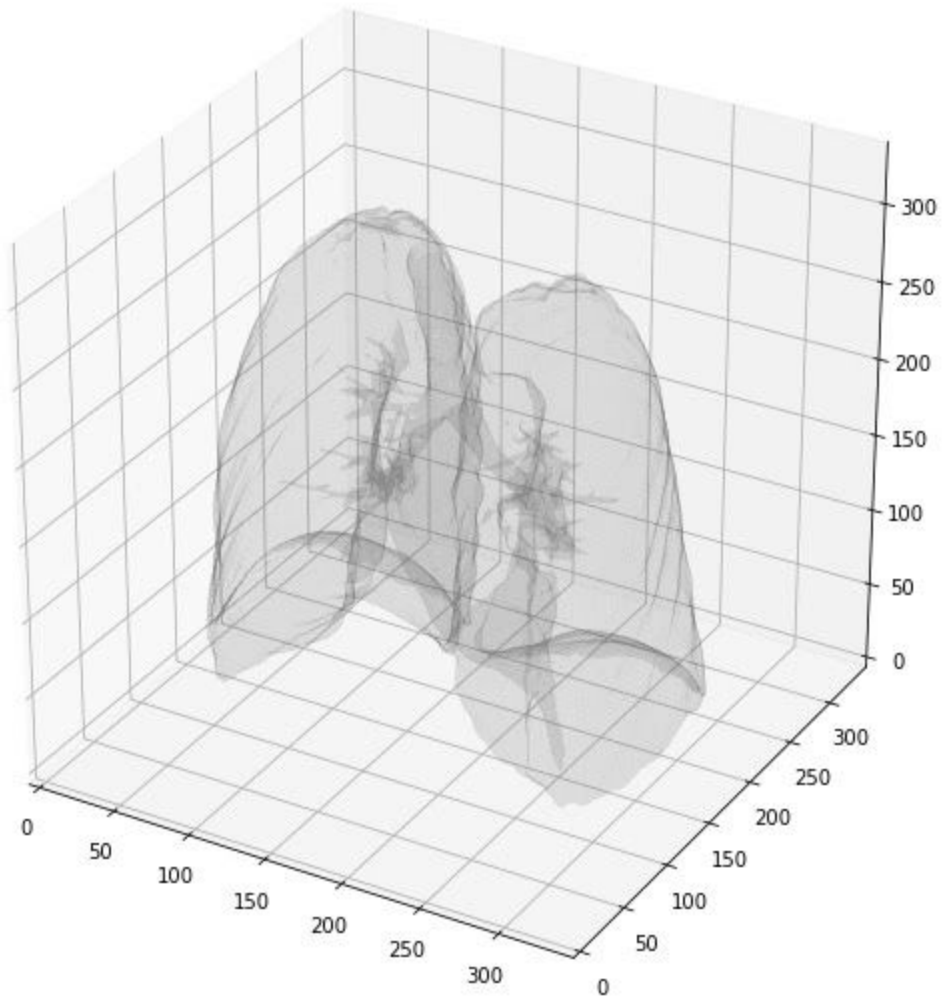
Visualization of dataset is an important part of training , it gives better understanding of dataset. But CT scan images are hard to visualize for a normal pc or any window browser. Therefore we use the pydicom library to solve this problem. The Pydicom library gives an image array and metadata information 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 which contains many subdirectories named on patient's ids. A complete subdirectory is 3d image of lungs which is stored in around 180 2d image slices according to their image number.



(fig 1 . original dicom slices from luna16 dataset)

After combining all images in a single subdirectory according to their image it represents a 3d image. 3d image of lungs gives a vast idea about lung cancer cells and other diseases in lungs .



(fig 2. 3d image of lungs of a single patient)

Metadata stored in CT scan images can be extracted from images with help of pydicom library. Information stored in one image is shown in the below figure.

```
Dataset.file_meta -------------------------------
(0002, 0000) File Meta Information Group Length  UL: 192
(0002, 0001) File Meta Information Version       OB: b'\x00\x01'
(0002, 0002) Media Storage SOP Class UID         UI: CT Image Storage
(0002, 0003) Media Storage SOP Instance UID      UI: 1.2.840.113654.2.55.247817952625791837963403492891187883824
(0002, 0010) Transfer Syntax UID                 UI: Explicit VR Little Endian
(0002, 0012) Implementation Class UID            UI: 1.2.40.0.13.1.1.1
(0002, 0013) Implementation Version Name         SH: 'dcm4che-1.4.31'
-------------------------------------------------
```

```
-------------------------------------------------
(0008, 0005) Specific Character Set             CS: 'ISO_IR 100'
(0008, 0016) SOP Class UID                      UI: CT Image Storage
(0008, 0018) SOP Instance UID                   UI: 1.2.840.113654.2.55.247817952625791837963403492891187883824
(0008, 0060) Modality                           CS: 'CT'
(0008, 103e) Series Description                 LO: 'Axial'
(0010, 0010) Patient's Name                     PN: '00cba091fa4ad62cc3200a657aeb957e'
(0010, 0020) Patient ID                         LO: '00cba091fa4ad62cc3200a657aeb957e'
(0010, 0030) Patient's Birth Date               DA: '19000101'
(0018, 0060) KVP                                DS: None
(0020, 000d) Study Instance UID                 UI: 2.25.862087301405397123827718905017727342779506923977090073054 73
(0020, 000e) Series Instance UID                UI: 2.25.115758773296352289258085968002699747408935194517846260466 14
(0020, 0011) Series Number                      IS: "3"
(0020, 0012) Acquisition Number                 IS: "1"
(0020, 0013) Instance Number                    IS: "134"
(0020, 0020) Patient Orientation                CS: ''
(0020, 0032) Image Position (Patient)           DS: [-145.500000, -158.199997, -356.200012]
(0020, 0037) Image Orientation (Patient)        DS: [1.000000, 0.000000, 0.000000, 0.000000, 1.000000, 0.000000]
(0020, 0052) Frame of Reference UID             UI: 2.25.830335096344416863856520734629838018401219166784177196696 50
(0020, 1040) Position Reference Indicator       LO: 'SN'
(0020, 1041) Slice Location                     DS: "-356.200012"
(0028, 0002) Samples per Pixel                  US: 1
(0028, 0004) Photometric Interpretation         CS: 'MONOCHROME2'
(0028, 0010) Rows                               US: 512
(0028, 0011) Columns                            US: 512
(0028, 0030) Pixel Spacing                      DS: [0.597656, 0.597656]
(0028, 0100) Bits Allocated                     US: 16
(0028, 0101) Bits Stored                        US: 16
(0028, 0102) High Bit                           US: 15
(0028, 0103) Pixel Representation               US: 1
(0028, 0120) Pixel Padding Value                US: 63536
(0028, 1050) Window Center                      DS: "40.0"
(0028, 1051) Window Width                       DS: "400.0"
(0028, 1052) Rescale Intercept                  DS: "-1024.0"
(0028, 1053) Rescale Slope                      DS: "1.0"
(7fe0, 0010) Pixel Data                         OW: Array of 524288 elements
```

(fig 3. Metadata contain in a single dicom slice)

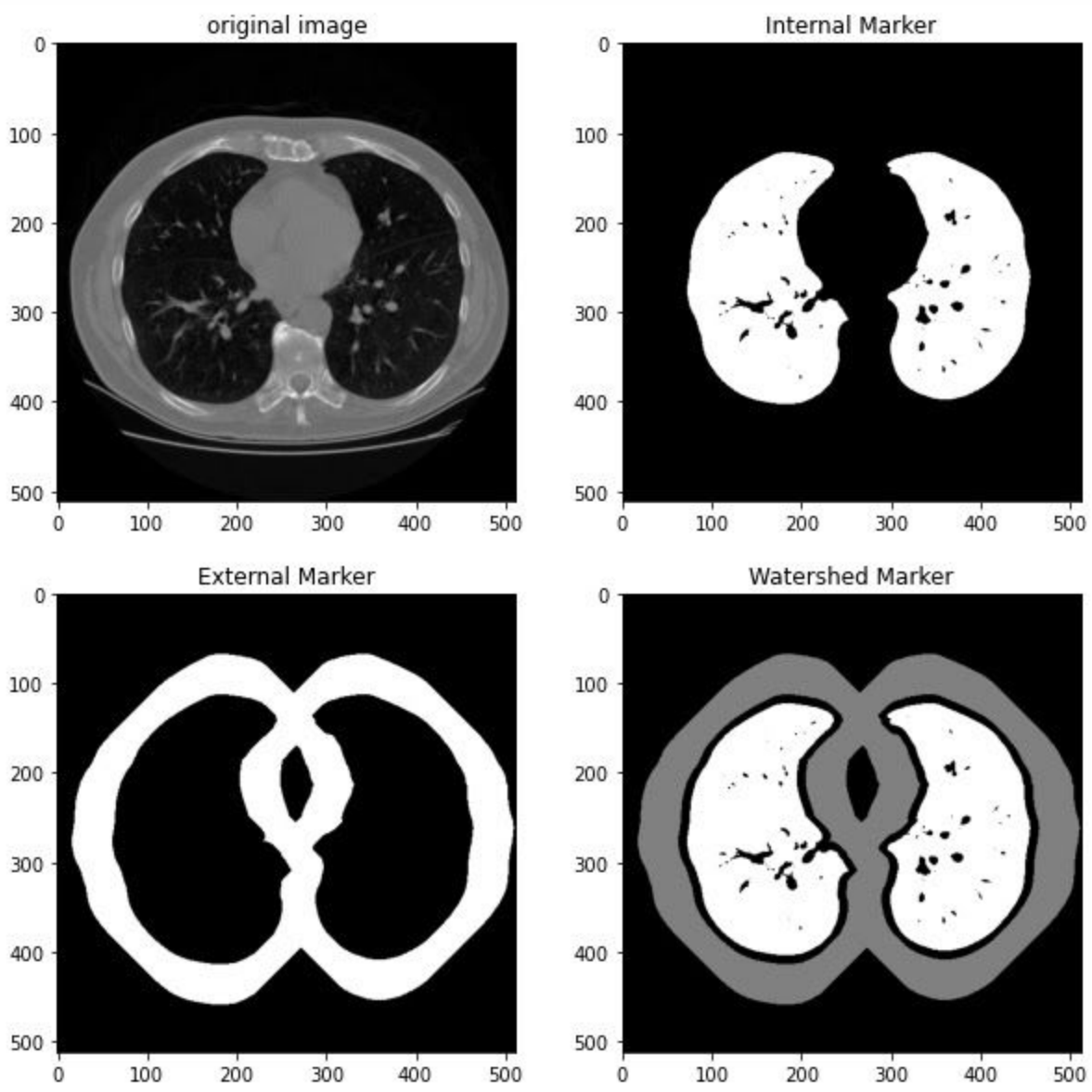
4. WATERSHED ALGORITHM

The watershed is a classical algorithm used for segmentation, that is, for separating different objects in an image.

a watershed is a transformation defined on a grayscale image. The name refers metaphorically to a geological watershed, or drainage divide, which separates adjacent drainage basins. The watershed transformation treats the image it operates upon like a topographic map, with the brightness of each point representing its height, and finds the lines that run along the tops of ridges.”The topological watershed” was introduced by M. Couprie and G. Bertrand in 1997

Starting from user-defined markers, the watershed algorithm treats pixels values as a local topography (elevation). The algorithm floods basins from the markers until basins attributed to different markers meet on watershed lines. In many cases, markers are chosen as local minima of the image, from which basins are flooded.

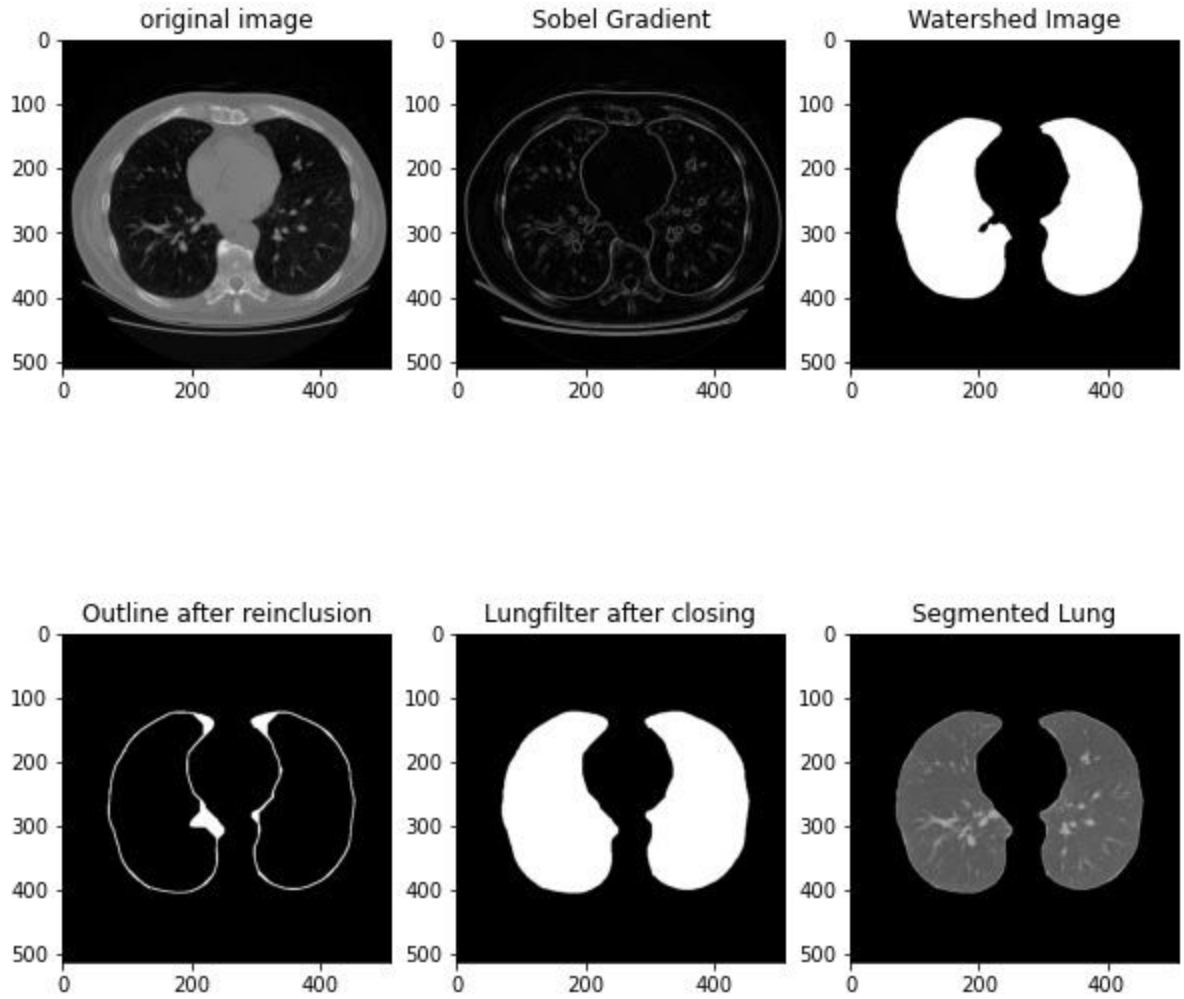
First , we extract internal and external markers from CT scan images with the help of binary dilations and add them with a complete dark image using watershed methods. And it removes external noise from the image and gives a watershed marker of lungs and cancer cells. As we can see in the below figure watershed marker removes external noise and applies a binary mask on the image , black pixels in lungs represent cancer cells.



(fig 4 . different markers extracted from CT scan image using watershed algorithm)

For better segmentation we integrate sobel filter with watershed algorithms .It removes the external layer of lungs. After removing the outer layer we use the internal marker and the Outline that was just created to generate the lungfilter using bitwise_or operations of numpy. It also removes the heart from CT scan images. Next step is to close off the lung filter with

morphological operations and morphological gradients. It provides better segmented lungs than the previous process. We can see this process in the figure below.



(fig 5. Image segmentation process visualization)

By doing this we in total generated 1002 images with related labels which includes almost 12 patients CT scan data in which there are almost the same number of cancer and non-cancer patients.

5. PROPOSED MODELS

The proposed model is a convolutional neural network approach based on lung segmentation on CT scan images. At first we preprocess the dataset of luna16. After preprocessing, the next step is to make lung segmentation with a watershed algorithm. Watershed algorithm highlights the lung part and makes binary masks for lungs using semantic segmentation approach.

We tried three different models of Convolutional Neural Networks, which are based on the comparative study of performance of each type model in different dataset and for different classification problems.

Our first model "Sequential_1" is the basic simple approach of using the convolution layers, flatten fully connected layers, max pooling and dropout in the middle layers, which performs significantly well on the number classification problem.

Summary of model is given below:

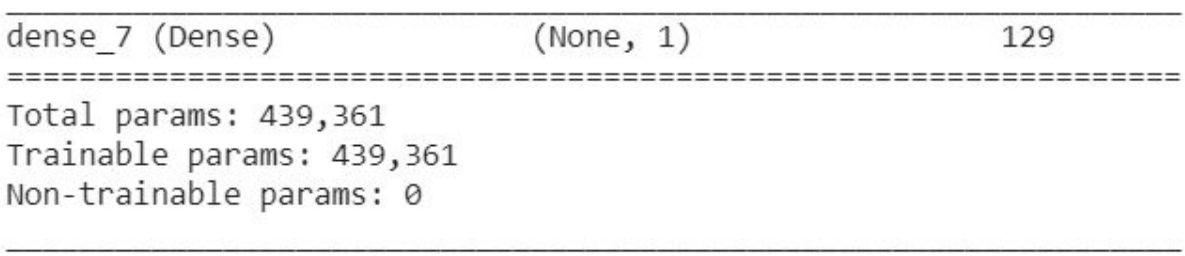
```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 510, 510, 32)      320
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 255, 255, 32)      0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 253, 253, 32)      9248
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 126, 126, 32)      0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 124, 124, 32)      9248
_________________________________________________________________
dropout_1 (Dropout)          (None, 124, 124, 32)      0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 122, 122, 32)      9248
_________________________________________________________________
flatten_1 (Flatten)          (None, 476288)            0
_________________________________________________________________
dense_1 (Dense)              (None, 128)               60964992
_________________________________________________________________
dropout_2 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_2 (Dense)              (None, 128)               16512
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 129
=================================================================
Total params: 61,009,697
Trainable params: 61,009,697
Non-trainable params: 0
_________________________________________________________________
```

(fig 6. Model summary of sequential_1)

Our second model “Sequential_2” is the Deep Convolutional Neural Network with Max Pooling and Fully connected layers in the end. This model with specified number of elements and layers performed best in many research papers with different datasets.

Summary of model is given below:

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_5 (Conv2D)            (None, 510, 510, 32)      320
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 255, 255, 32)      0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 253, 253, 32)      9248
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 126, 126, 32)      0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 124, 124, 32)      9248
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 62, 62, 32)        0
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 60, 60, 64)        18496
_________________________________________________________________
max_pooling2d_6 (MaxPooling2 (None, 30, 30, 64)        0
_________________________________________________________________
conv2d_9 (Conv2D)            (None, 28, 28, 64)        36928
_________________________________________________________________
max_pooling2d_7 (MaxPooling2 (None, 14, 14, 64)        0
_________________________________________________________________
conv2d_10 (Conv2D)           (None, 12, 12, 64)        36928
_________________________________________________________________
max_pooling2d_8 (MaxPooling2 (None, 6, 6, 64)          0
_________________________________________________________________
flatten_2 (Flatten)          (None, 2304)              0
_________________________________________________________________
dense_4 (Dense)              (None, 128)               295040
_________________________________________________________________
dense_5 (Dense)              (None, 128)               16512
_________________________________________________________________
dense_6 (Dense)              (None, 128)               16512
_________________________________________________________________
```



(fig 7. Model summary of sequential_2)

Our third and the last model approach was to use transfer learning on VGG-16 with some changes in the last three layers which are fully connected. This model gives appreciable results in object classification.

Summary of model is given below:

```
Model: "vgg16"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_3 (InputLayer)         (None, 224, 224, 3)       0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 224, 224, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 112, 112, 64)      0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 112, 112, 128)     73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 112, 112, 128)     147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 56, 56, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 56, 56, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 28, 28, 256)       0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 14, 14, 512)       0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
```

block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
flatten (Flatten)	(None, 25088)	0
fc1 (Dense)	(None, 4096)	102764544
fc2 (Dense)	(None, 4096)	16781312
fc3 (Dense)	(None, 128)	524416
fc4 (Dense)	(None, 128)	16512
output (Dense)	(None, 1)	129

Total params: 134,801,601
Trainable params: 17,322,369
Non-trainable params: 117,479,232

(fig 8. Model summary of Transfer learning model-VGG16)

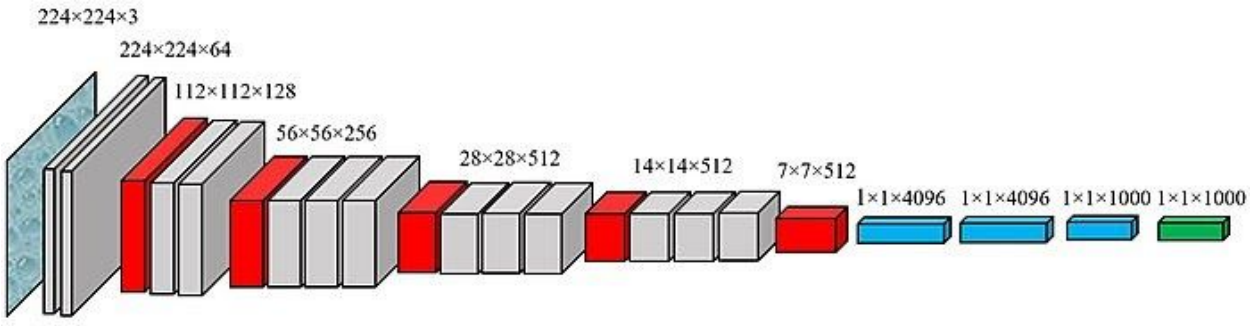
Important Information regarding the training model:

After making successful binary lung segmented masks, we train models on segmented lungs with a batch size of 32 for image data generator and using 100 images in each epoch for 30 epochs with exception of 500 images in each epoch for VGG-16. We are training images with the shape of (512,512,1) for the first two models and the shape of (512,512,3) for VGG-16. For a better result data augmentation is used to train models on different augmentation like shear range , zoom range , horizontal flip , rotation range , centre shift etc. For the end layer we used a single node for binary c;assification as we want to classify between cancer and non- cancer lungs.

Also we used the callbacks from tensorflow keras to save the best accuracy model so that we can run a complete 50 epoch training session to plot the comparison graphs.

6. TRANSFER LEARNING : VGG16-NET

VGG Net is the name of a pre-trained convolutional neural network (CNN) invented by Simonyan and Zisserman from Visual Geometry Group (VGG) at University of Oxford in 2014 and it was able to be the 1st runner-up of the ILSVRC (ImageNet Large Scale Visual Recognition Competition) 2014 in the classification task. VGG Net has been trained on ImageNet ILSVRC dataset which includes images of 1000 classes split into three sets of 1.3 million training images, 100,000 testing images and 50,000 validation images. The model obtained 92.7% test accuracy in ImageNet. VGG Net has been successful in many real world applications such as estimating the heart rate based on the body motion, and pavement distress detection .



(fig 9 . vgg16-net working architecture)

VGG Net has learned to extract the features (feature extractor) that can distinguish the objects and is used to classify unseen objects. VGG was invented with the purpose of enhancing classification accuracy by increasing the depth of the CNNs. VGG 16 and VGG 19, having 16 and 19 weight layers, respectively, have been used for object recognition. VGG Net takes input of 224×224 RGB images and passes them through a stack of convolutional layers with the fixed filter size of 3×3 and the stride of 1. There are five max pooling filters embedded between convolutional layers in order to down-sample the input representation (image, hidden-layer output matrix, etc.). The stack of convolutional layers are followed by 3 fully connected layers,

having 4096, 4096 and 1000 channels, respectively. The last layer is a soft-max layer . Below figure shows VGG network structure.

But in our approach we have images with the shape of (512,512) . so we build our own model using vgg16-net architecture. And compile the model with a powerful adam optimizer , learning rate is 0.0001 , entropy is binary_crossentropy and accuracy metrics. The below figure shows model summary , convolution layers, maxpooling layers and params.

```
Model: "vgg16"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_3 (InputLayer)         (None, 224, 224, 3)       0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 224, 224, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 112, 112, 64)      0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 112, 112, 128)     73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 112, 112, 128)     147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 56, 56, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 56, 56, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 28, 28, 256)       0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 14, 14, 512)       0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
```

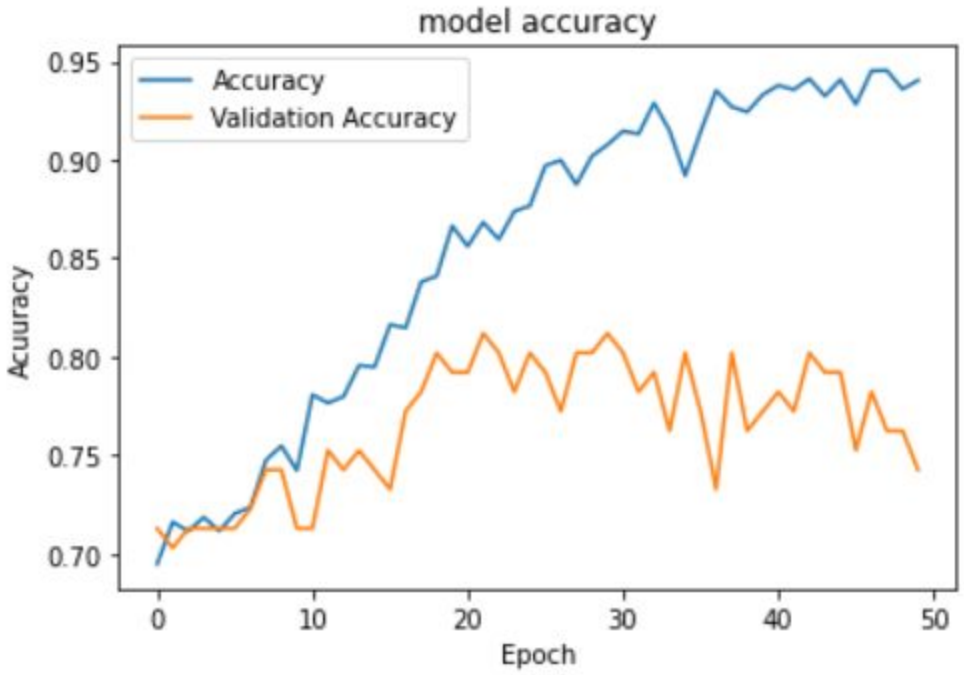
```
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 7, 7, 512)         0
_________________________________________________________________
flatten (Flatten)            (None, 25088)             0
_________________________________________________________________
fc1 (Dense)                  (None, 4096)              102764544
_________________________________________________________________
fc2 (Dense)                  (None, 4096)              16781312
_________________________________________________________________
predictions (Dense)          (None, 1000)              4097000
=================================================================
Total params: 138,357,544
Trainable params: 138,357,544
Non-trainable params: 0
_________________________________________________________________
```

(fig 10. Model summary of Transfer learning model-VGG16)

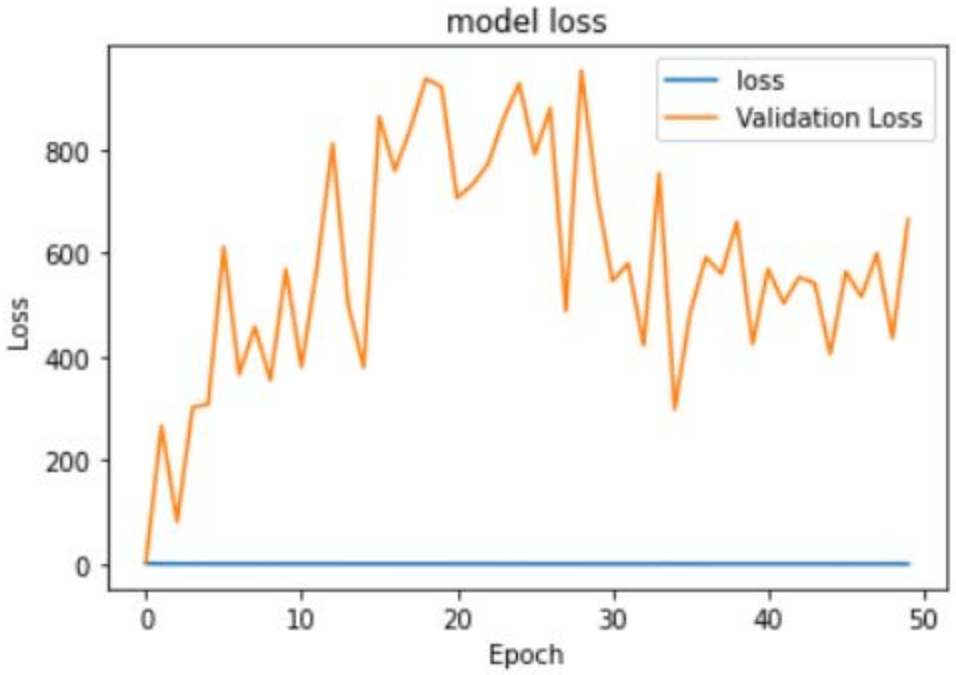
7. CONCLUSIONS AND RESULTS

Following are the graphs of accuracy and loss from the three models used for classification in our project.

Model: Sequential_1

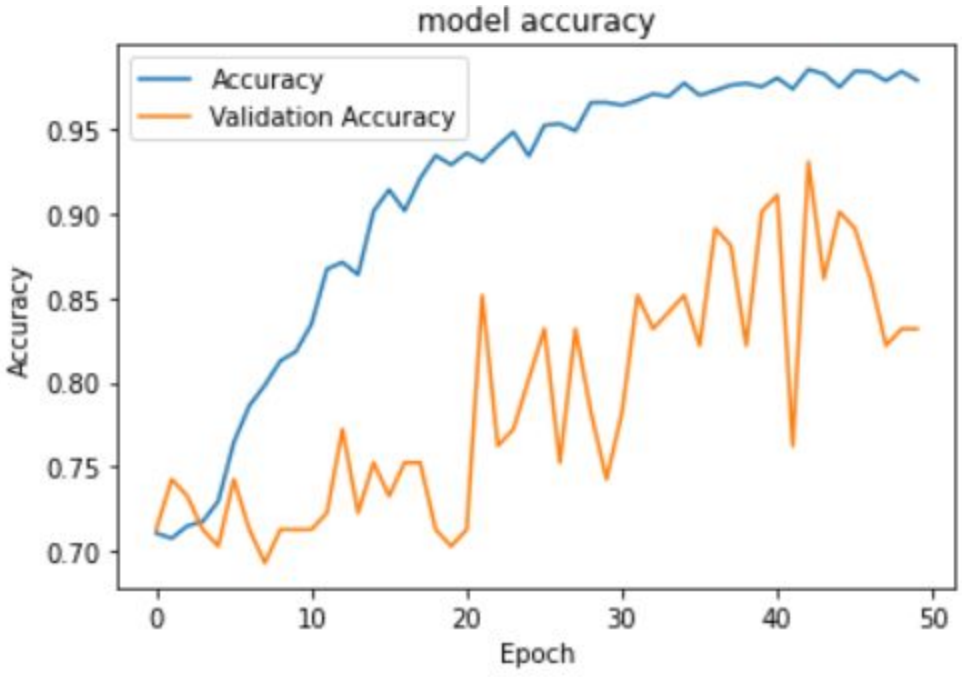


(fig 11 . Training and Validation Accuracy of model Sequential_1)

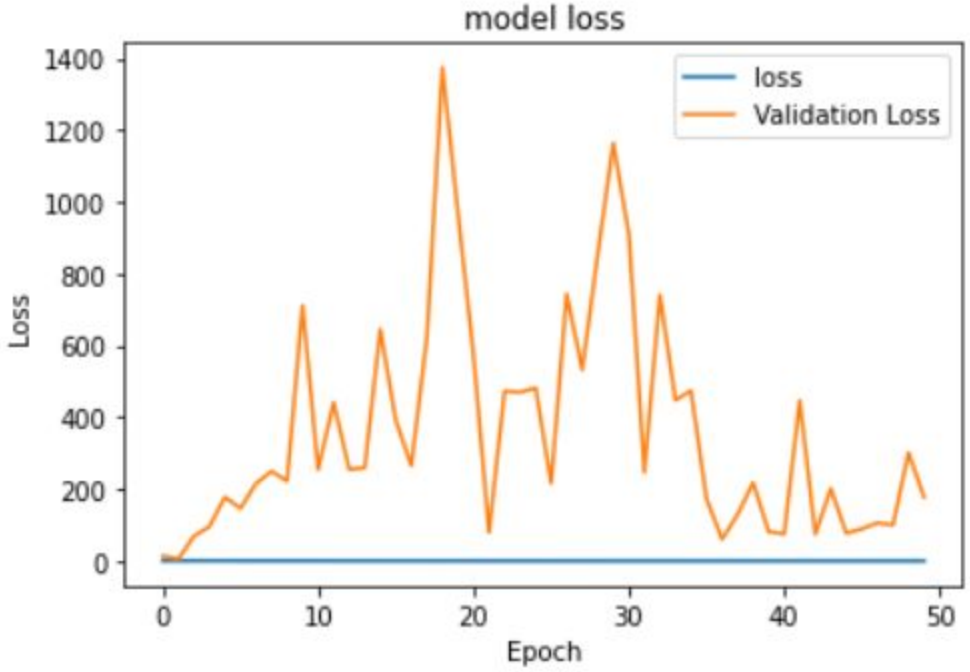


(fig 12. Training and Validation loss of model Sequential_1)

Model: Sequential_2

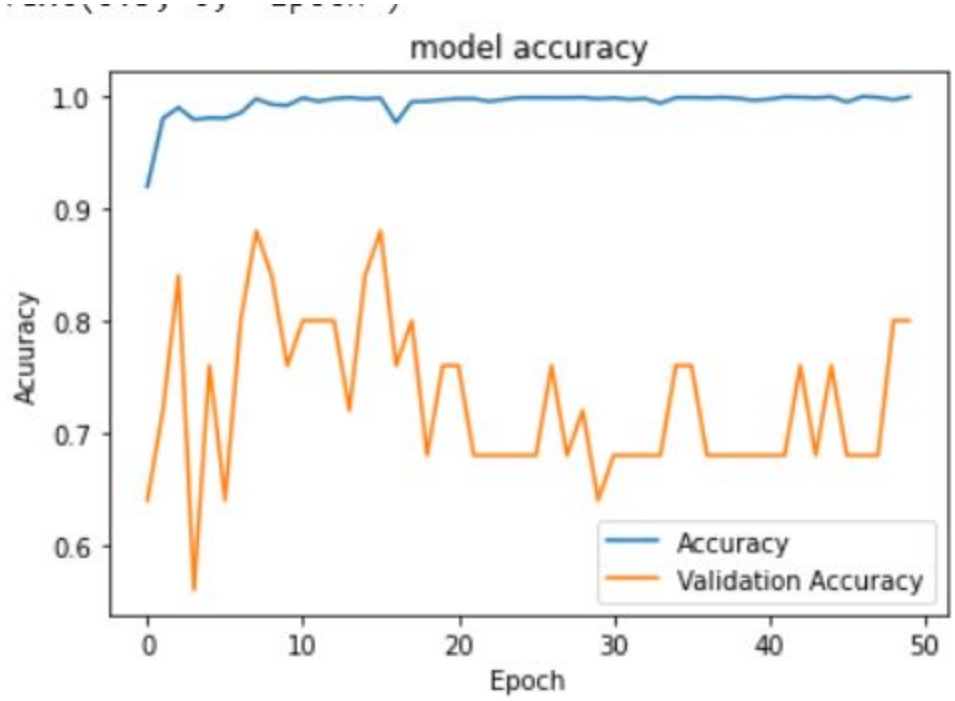


(fig 13 . Training and Validation Accuracy of model Sequential_2)

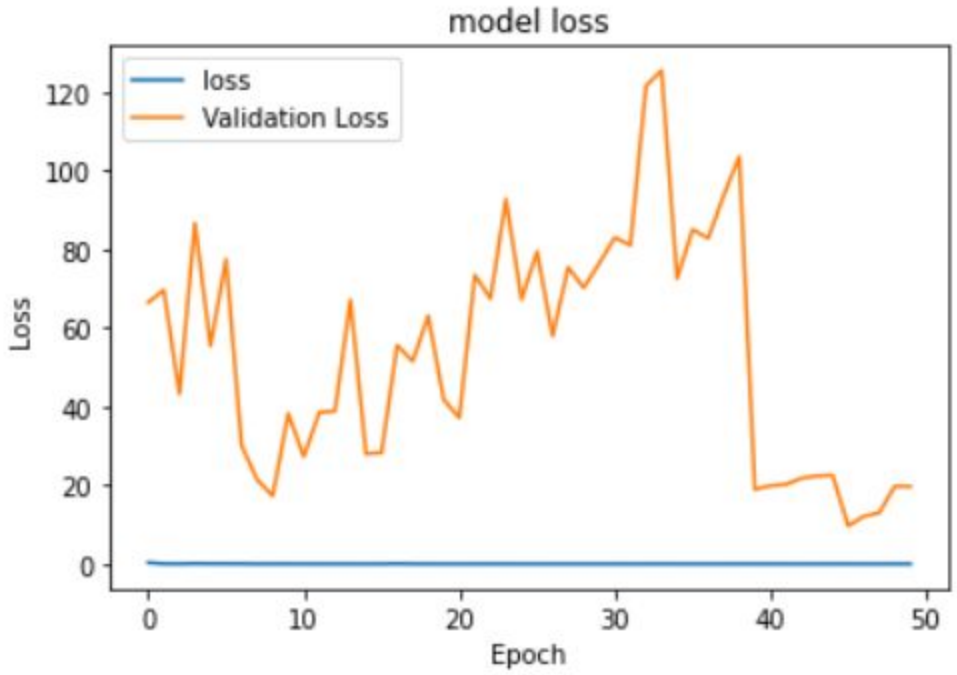


(fig 14. Training and Validation loss of model Sequential_2)

Model: VGG_16



(fig 15. Training and Validation Accuracy of model VGG_16)



(fig 16. Training and Validation loss of model VGG_16)

Tabular Comparison of model training accuracy and loss and model validation accuracy and loss:

Note: we are showing the best val. acc. related information in all models as the callbacks function will save only the best accuracy model.

Index	**Model**	**Train Acc.**	**Train loss**	**Val. Acc.**	**Val. loss**
1.	Sequential_1	90.77%	0.2242	81.19%	712.0875
2.	Sequential_2	98.53%	0.0442	93.07%	74.5244
3.	VGG_16	99.84%	0.0046	88.00%	28.2614

(Table 1. Accuracy and loss comparison of different models)

Results:

From the Results shown adobe in graphs and the comparison table we can conclude the following things:

1. The Sequential_1 model is performing the worst overall and even getting over trained without reaching the satisfactory amount of validation accuracy though the validation loss is least but this is a classification problem so it focuses more on accuracy.
2. Sequential_2 and VGG_16 are performing good when training the model and reaching appreciable levels of test accuracy and test loss but in validation Sequential is performing better in terms of accuracy as compared to VGG_16 but loss is less in VGG_16.
3. The poor performance of transfer learning can be due to 2 main reasons, which are less amount of training data in one epoch due to limited amount of resources and time and bad data input to the VGG_16. Both the topics are discussed in the Challenges Section.

Tabular comparison of the best accuracy model with some other models from from other research papers for binary classification:

Index	**Model**	**Accuracy**
1.	Sequential_2 (best model from project)	93.07%
2.	Deep Residual Network Link to paper	93.25%
3.	ACO_SVM Link to paper	93.2%
4.	ACO_ANN Link to paper	98.40%

(Table 2. Tabular comparison of proposed model with previous models)

8. CHALLENGES

There are some challenges which we faced during the making of the project which impacted the results and outputs of the project.

1. Dataset:

   The Dataset that we used was in .dmc format which is different from standard image processing format like .jpg ect., this format stores the whole information of the patient along with a 3D CT-Scan image of the patient which when converted into 2D numpy image results in the shape of (n, m, m, 1) format which is not supported by the models for transfer learning. And is also in float64 data type which is difficult to handle and process when it comes to CV2, skimage and matplotlib as the standard libraries.

   Moreover when the data is converted into 2D Grayscale image then it becomes difficult to train it with tensorflow and keras framework as they do not support such shapes (n, m, m) during model construction.

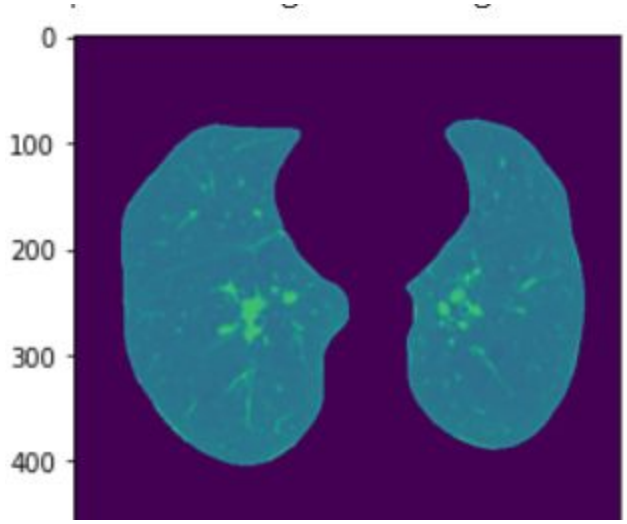
   Now when the images are converted back into RGB or BGR format using different libraries then the output images lose some or many features which are useful in the classification process.

   Some images output with different types of conversions is shown below and can be seen in the python notebook's last column.

   Note: Images are plotted using matplotlib.pyplot with different preprocessing done already which is mentioned along with the image.

   Image_shape: (512,512)

   Dtype: float64

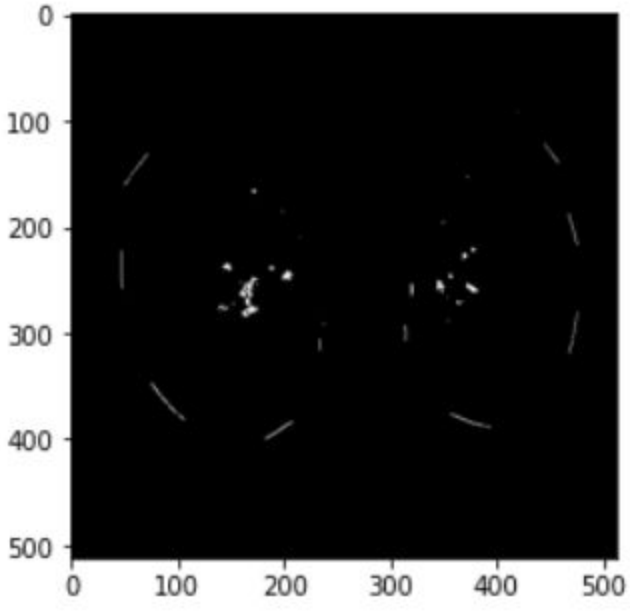


(fig 17. Float64 image of slice from original dataset)

Image_shape:(512,512,3). converted using skimage library.

Message while conversion by matplotlib:

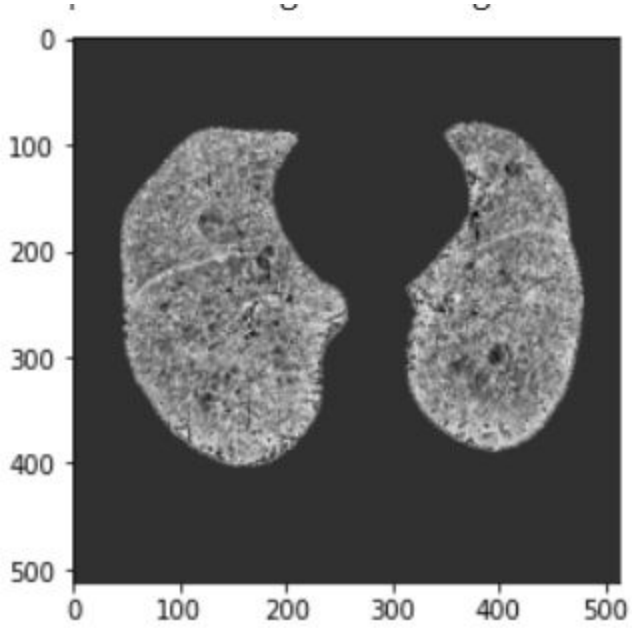
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).



(fig 18. Converted RGB image of a slice using skimage)

Image_shape: (512,512,3). Converted using cv2 library

Dtype: int8

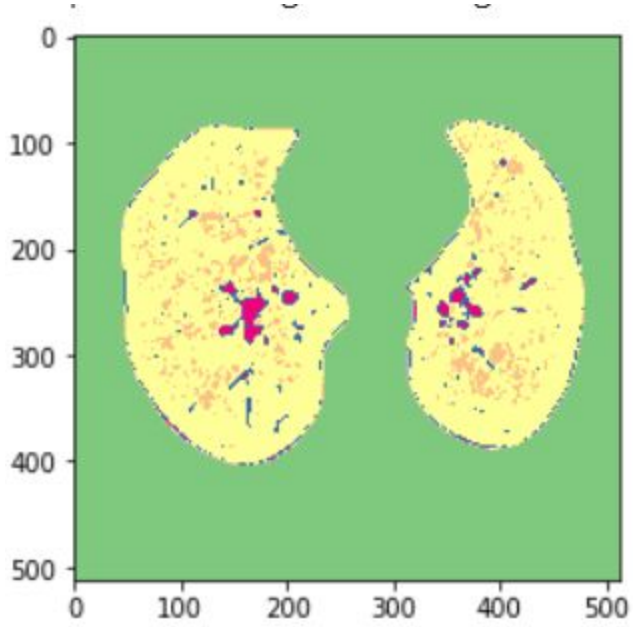


(fig 19. Converted RGB int8 image of a slice using opencv)

Some most suitable image displayed in float64 data type in shape (512,512) by matplotlib using cmap = ‘Accent’

Type: Cancer

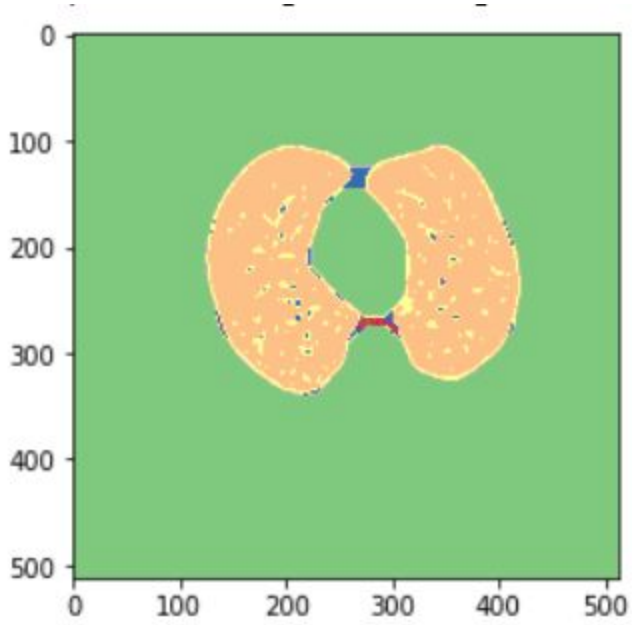
Image:



(fig 20. image of Lung cancer plotted with matplotlib)

Type: No_cancer

Image:



(fig 21. Image of healthy lungs plotted with matplotlib)

As we can visually see that many of the features of a lung's radiograph like shape, area, parameter, opacity etc. changes which can even be seen by different color representations and shades in the plot.

If the plotted images can be converted into the standard (512,512,3) format in int8 or int64 the model accuracy can even reach to 99% or above with much familiar linear trends in the plotted graphs.

2. Resources and Hardware Requirements:

   There is a limited amount of resources available for the processing of the dataset, first of all the dataset is a hard to find resource and after that the hardware for the processing is also limited.

   The whole project is made on Google Colab which provides the run time of 12 hour at max for one go and after that it needs a cooldown time of 9-12 hours which limits the progress speed along with the ram storage provided for the processing is 12BG but the when the data is compiled together in a numpy array then it becomes difficult to train small models for more than 1200 images in RGB or even grayscale mode.

   Also when the model is large then due to high number of parameters the terminal crashes as it’s RAM goes out of memory for regular 32 batch in one epoch for 1000 images so we need to decrease the number of batches which affects the model in more than one way (which is out of the context of this projects topic’s scope).

   So, these are the two major Challenges we have to face during the making of the project.

9. REFERENCES

1. Bjerager M., Palshof T., Dahl R., Vedsted P., Olesen F. Delay in diagnosis of lung cancer in general practice. Br. J. Gen. Pract. 2006;56:863–868. [PMC free article] [PubMed] [Google Scholar]

2. Nair M., Sandhu S.S., Sharma A.K. Cancer molecular markers: A guide to cancer detection and management. Semin. Cancer Biol. 2018;52:39–55. doi: 10.1016/j.semcancer.2018.02.002. [PubMed] [Google Scholar]

3. Silvestri G.A., Tanner N.T., Kearney P., Vachani A., Massion 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]

4. 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]

5. Lee K.S., Mayo J.R., Mehta A.C., Powell C.A., Rubin G.D., Prokop C.M.S., Travis W.D. Incidental Pulmonary Nodules Detected on CT Images: Fleischner 2017. Radiology. 2017;284:228–243. [PubMed] [Google Scholar]

6. Diederich S., Heindel W., Beyer F., Ludwig K., Wormanns D. Detection of pulmonary nodules at multirow-detector CT: Effectiveness of double reading to improve sensitivity at standard-dose and low-dose chest CT. Eur. Radiol. 2004;15:14–22. [PubMed] [Google Scholar]

7. Demir Ö., Çamurcu A.Y. Computer-aided detection of lung nodules using outer surface features. Bio-Med. Mater. Eng. 2015;26:S1213–S1222. doi: 10.3233/BME-151418. [PubMed] [Google Scholar]

8. Bogoni L., Ko J.P., Alpert J., Anand V., Fantauzzi J., Florin C.H., Koo C.W., Mason D., Rom W., Shiau M., et al. Impact of a computer-aided detection (CAD) system integrated into a picture

archiving and communication system (PACS) on reader sensitivity and efficiency for the detection of lung nodules in thoracic CT exams. J. Digit. Imaging. 2012;25:771–781. doi: 10.1007/s10278-012-9496-0. [PMC free article] [PubMed] [Google Scholar]

9. Al Mohammad B., Brennan P.C., Mello-Thoms C. A review of lung cancer screening and the role of computer-aided detection. Clin. Radiol. 2017;72:433–442. doi: 10.1016/j.crad.2017.01.002. [PubMed] [Google Scholar]

10. Automated Lung Nodule Detection and Classification Using Deep Learning Combined with Multiple Strategies. Nasraullah Nasrullah, Jun Sang, Mohammad S. Alam, Muhammad Mateen, Bin Cai and Haibo Hu. [PMC]

11. Setio A.A.A., Traverso A., de Bel T., Berens M.S.N., van den Bogaard C., Cerello P., Chen H., Dou Q., Fantacci M.E., Geurts B., et al. Validation, comparison, and combination of algorithms for automatic detection of pulmonary nodules in computed tomography images: The LUNA16 challenge. Med. Image Anal. 2017;42:1–13. doi: 10.1016/j.media.2017.06.015. [PubMed] [Google Scholar]

12. 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]

13. 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]

14. Jiang H., Ma H., Qian W., Gao M., Li Y. An automatic detection system of lung nodules based on a multigroup patch-based deep learning network. IEEE J. Biomed. Heal. Inform. 2018;22:1227–1237. doi: 10.1109/JBHI.2017.2725903. [PubMed] [Google Scholar]

15. 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]

16. Gu Y., Lu X., Yang L., Zhang B., Yu D., Zhao Y., Gao L., Wu L., Zhou T. Automatic lung nodule detection using a 3D deep convolutional neural network combined with a multi-scale prediction strategy in chest CTs. Comput. Biol. Med. 2018;103:220–231. doi: 10.1016/j.compbiomed.2018.10.011. [PubMed] [CrossRef] [Google Scholar]

17. Yu L., Dou Q., Chen H., Heng P.-A., Qin J. Multilevel contextual 3-D CNNs for false positive reduction in pulmonary nodule detection. IEEE Trans. Biomed. Eng. 2016;64:1558–1567. [PubMed] [Google Scholar]

18. 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]

19. Chen Y., 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]

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

21. 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]

Link to the project folder on Google Drive:

Project_lung_cancer_binary_classification

Note: link is shareable only with NITH related google drives.

Thankyou