# Chapter 1

# Introduction and Motivation

## 1.1. Motivation for Computer Aided Diagnosis (CAD) systems

Cancer is the deadliest disease worldwide. World Health Organization (WHO) estimated that cancer is the leading cause of morbidity and mortality, with approximately 14 million cases in 2012 [1]. The number of cases is expected to rise by about 70% in the next two years. Around 2.5 million people are estimated to live with the disease, over 7 lakh new cancer patients are registered every year, and 5 lakh die due to cancer in India [2]. Lung cancer has been the most common cancer in the world. The number of new cases estimated is 1.8 million in 2012 and it is the most common cancer in males and third most common cancer in females worldwide [3]. Lung cancer accounts for 11.3% of new cancers and 13.7% of cancer deaths in India [2]. Smoking accounts for 1 in 5 deaths among men and 1 in 20 deaths among women. Top five cancers in men and women account for 47.2 % of all cancers and cancers of oral cavity and lungs in males, and cervix and breast in females account for over 50% of all cancer deaths in India [2].

Lung cancer is rarely diagnosed in the early stages. Early diagnosis can significantly reduce the risk of death due to it. Cancer screening is currently being done using low-dose computed tomography (CT) scans which can detect the initial signs of lung cancer that is manifested as pulmonary nodules. Pulmonary nodules are abnormal structures which are approximately spherical in the shape and appear as bright spots on CT scans. A vast majority of pulmonary nodules are benign (non-cancerous) and do not require any treatment. Few nodules are malignant (cancerous) and their detection is the crucial step in diagnosing early lung cancer. After the detection of nodule, information about its features is responsible for the treatment.

A patient visits doctor with certain lung related symptoms. If doctor doubts it to be cancer, he will suggest the patient to take a CT scan of the chest region. Then, the chest CT scan is analyzed by an experienced radiologist, who determines the potential nodules present in the patient's lungs. A nodule can either be benign or malignant. If a nodule is malignant, doctor advise the patient to go for a biopsy test which exactly determines whether his lung contains cancerous

tumor. Going for biopsy test leads to deep stress and agony in patient, consumes money and time. Hence, reduction of false positives (i.e. benign nodules tagged as malignant) is crucial and has significant scope. The flow of lung cancer diagnosis is shown in Figure 1.1.

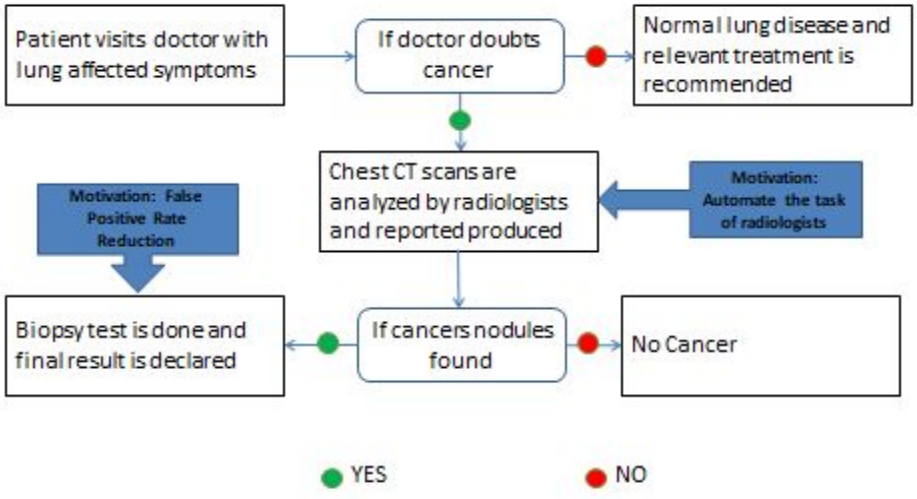


***Figure 1.1: Lung Cancer Diagnosis Flow Chart***

The arduous task of nodule detection is manually done by the radiologists after an extensive period of training. Past studies show that radiologists fail to detect nodules accurately due to fatigue and human-error [4] [5]. This has been the motivation for the development of Computer Aided Diagnosis (CAD) systems which are expected to assist radiologists in detecting nodules with high accuracy. Current CAD systems are developed using image processing techniques, classical machine learning algorithms, and Artificial Neural Networks (ANNs). They are good at detecting lung nodules. In the process, many false positives are generated which is a serious issue and needs to be answered.

With the advent of deep learning, Convolutional Neural Network (CNN) models and Re-Current Neural Network (RNN) models are being used for the better accuracy in nodule detection and false positive reduction tasks. These models are expected to enhance the efficiency of CAD systems which has been our motivation to develop deep learning architectures for better performance compared to the current state-of-art methods.

## 1.2. Objectives

Our primary objective was to detect cancerous nodules from the CT scans. Current state-of-art methods that are based on image processing techniques and classical machine learning techniques are able to do it with moderate accuracies. Methods that are based on CNN models have proved to be more accurate than the classical machine learning algorithms. We have implemented classical image processing algorithms and CNN models to accurately detect nodules.

Our secondary objective was to reduce the false positive rate. Current state-of-art methods suffer serious drawbacks due to high false positive rate. High false positive rate undermine our motivation to develop CAD systems as radiologists would prove to be better than these systems. CNN architecture designed by us has significantly reduced the false positive rate.

## 1.3. Applications

The system can drastically reduce the effort required by a radiologist to detect lung nodules. A radiologist requires an extensive training to detect nodules from CT scan images, though the rate of false positives is high. As mentioned earlier, the system can help to detect the lung cancer at an early stage, which will increase the patient's chance of surviving and deaths from lung cancer will go down, and also reduce the . The system will open a doorway of automating the health care.

## 1.4. Data Set

The data-set was obtained from the Lung Image Database Consortium and Infectious Disease Research Institute (LIDC/IDRI) database [6], largest publicly available lung CT scan data. This database consists of 1018 CT scans, which come with associated XML files combined with annotations done by four trained radiologists. These scans are accompanied with MetaImage (.mhd) images that are downloaded from LUNA16 website [7]. Each LIDC/IDRI scan was annotated by experienced radiologists independently who marked all suspicious lesions as: nodule >= 3mml; nodule<3mm; non-nodule. Although the database contained 1018 CT scans,

some were discarded which had inconsistent slice spacing or missing slices. After exclusion, the final database contained 888 scans.

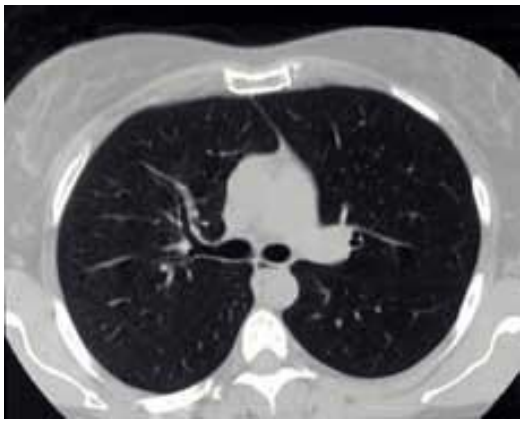

***Figure 1.2: View of a slice extracted from Lung CT scan [6]***

One CT scan is a set of 120 slices of 2D images. Each slice is a sectional view of a lung. We invested significant amount of time to extract the 2D images of potential nodules from the annotations done by the radiologists. These annotations are given in world co-ordinates which have to be converted into voxel in order to extract the potential nodules.

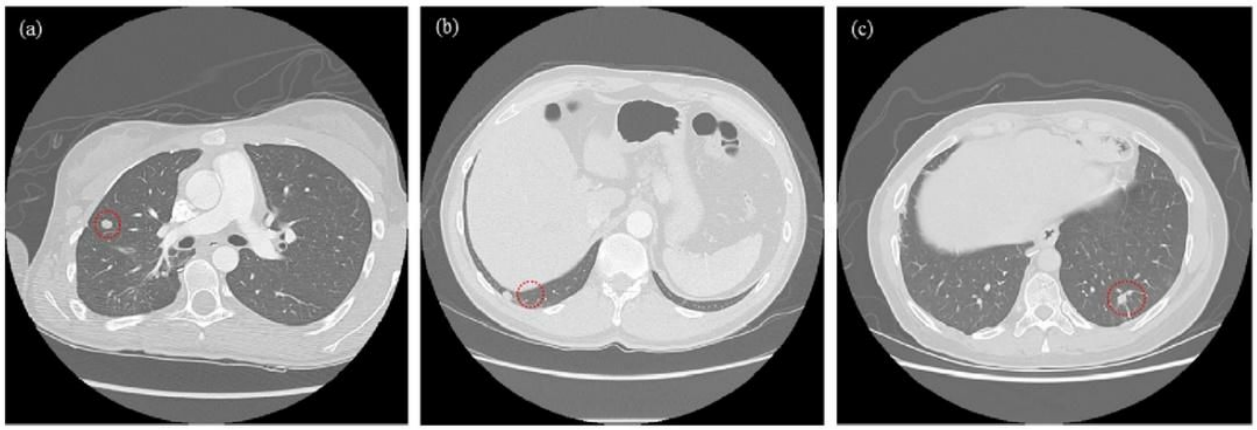


***Figure 1.3: Different sectional views of a CT scan slice, (a) represents front view, (b) represents horizontal cross sectional view, and (c) represents vertical cross sectional view [6]***

Using Simple insight Segmentation Tool kit (SimpleITK), we converted the raw scan into a 3D array. We ran a script that converted the co-ordinates from world to voxel. Voxel co-ordinates

are useful to correctly identify the region in the array containing potential nodule. The potential nodules are then extracted and a database of 64x64 PNG is created.

Figure 1.4 and 1.5 shows the final database obtained by us containing non-nodules and nodules respectively.

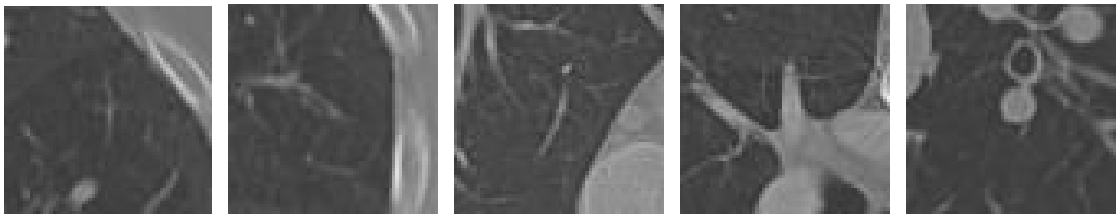

***Figure 1.4: Non-nodules***

***Figure 1.5: Nodules***

## 1.5. Organization of the Report

Problem statement, motivation and dataset are introduced in Chapter 1. Current state-of-art methods and literature works are detailed in Chapter 2. Chapter 3 describes our proposed. Chapter 4 contains results and related discussions. The report is concluded in Chapter 5 followed by references at the end.

# Chapter 2

# Literature Survey

In this chapter, detailed survey is carried out in various stages namely preprocessing, lung segmentation, potential nodule segmentation, feature extraction, and training machine learning models to classify potential nodules as nodules or non-nodules.

## 2.1. CT Scan Details

Computed tomography (CT) scans, are special X-ray tests that produce cross-sectional images of the body using X-rays and a computer. CT scans are also referred to as computerized axial tomography. CT was developed independently by a British engineer named Sir Godfrey Hounsfield and Dr. Alan Cormack. It has become a mainstay for diagnosing medical diseases. CT scanners have vastly improved patient comfort because a scan can be done quickly. Improvements have led to higher-resolution images, which assist the doctor in making a diagnosis. For example, the CT scan can help doctors to visualize small nodules or tumors, which they cannot see with a plain film X-ray. The survival rate of lung cancer can be substantially improved if it is detected and treated in the early stage. Low dose computed tomography (CT) chest scans have been shown effective in screening lung cancer, however, reading the large CT volumes and detecting lung nodules accurately and repeatable demand enormous amount of radiologist's effort.

### CT scan facts

- CT scan images allow the doctor to look at the inside of the body just as one would look at the inside of a loaf of bread by slicing it. This type of special X-ray, in a sense, takes "pictures" of slices of the body so doctors can look right at the area of interest. CT scans are frequently used to evaluate the brain, lung, neck, spine, chest, abdomen, pelvis, and sinuses.

- CT has revolutionized medicine because it allows doctors to see diseases that, in the past, could often only be found at surgery or at autopsy. CT is non-invasive, safe, and well-tolerated. It provides a highly detailed look at many different parts of the body.
- The cross-sectional images generated during a CT scan can be reformatted in multiple planes, and can even generate three-dimensional images. These images can be viewed on a computer monitor, printed on film or transferred to a CD or DVD.
- People often have CT scans to further evaluate an abnormality seen on another test such as an X-ray or an ultrasound. They may also have a CT to check for specific symptoms such as pain or dizziness. People with cancer may have a CT to evaluate the spread of disease.
- A chest CT test is most commonly used to detect lung cancer by analyzing cancerous nodules present in the lungs.

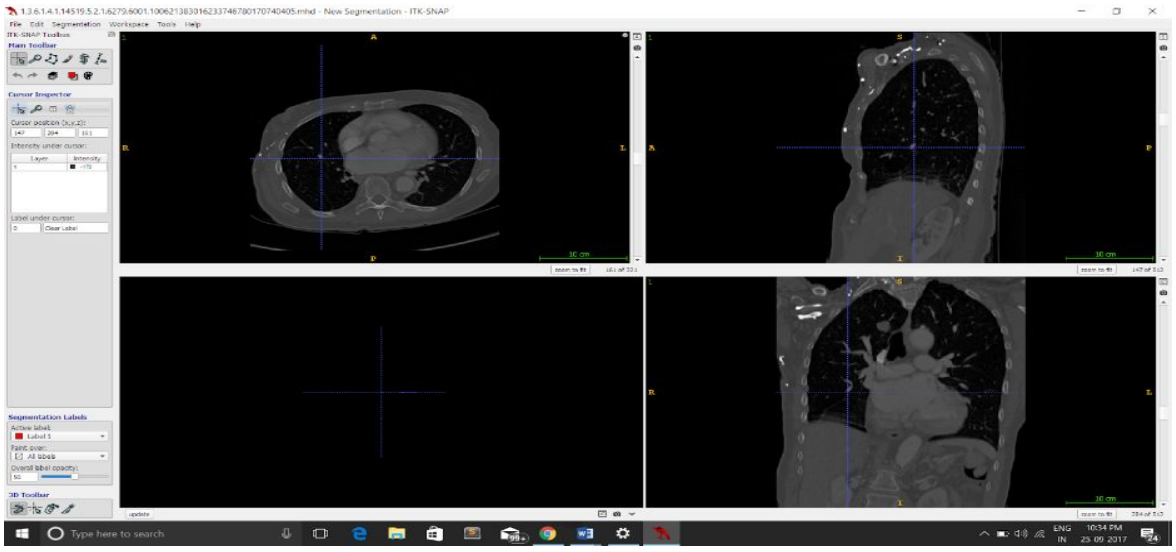

***Figure 2.1: A 3D visualization of CT scan***

## 2.2. Radiology Practices

Radiologists have previously relied on examining images from chest radiography and PET scans to detect lung cancer [8]. However, advancements in computed tomography (CT) in the 21st

century made it a more advantageous tool in both resolution and speed [9]. The manual detection of solid and sub solid pulmonary lesions in thoracic CT scans is quite error-prone, with a particularly high false-negative rate for detecting small nodules due to, for example, their size, density, location, and conspicuousness. To improve the hand annotation of nodules, medical experts expand beyond an axial scan mode and rely on other techniques such as maximum intensity projections and 3D volume renderings [10]. Maximum intensity projection (MIP) is a volume rendering technique for 3D images that projects voxels with maximum intensity of the parallel rays from a given viewpoint onto the plane [11]. This technique makes it easier to detect denser objects like nodules, since maximum projections will be concentrated in a particular area, whereas other structures like thin blood vessels will have maximum intensities more distributed throughout the lung/image.

## 2.3. Computer Aided Diagnosis Systems

Early CAD systems were based on classical image processing and machine learning algorithms. In the paper K. Murphy et al. [12], they used local image features to detect lung regions and potential nodules and K-Nearest Neighbors classification algorithm to classify the detected potential nodules as nodule or non-nodules. The system is trained and tested on three databases extracted from a large scale experimental screening study – Nelson Trial Database. Initial preprocessing involves subsampling of image data and segmentation of lung volume. They down sampled the image matrix size of 512x512 to 256x256 in order to improve the speed of the algorithm. Lung segmentation in this paper is carried out by the algorithm followed in the paper Sluimer et al. [13]. As the lung is essentially a bag of air in the body it shows up as a dark region in the CT scan. This contrast between the lung and the surrounding tissues forms the basis for their segmentation scheme. Apart from the segmentation of lung regions, airways and vessels are also segmented out in this paper. This ensures that nodule detection is performed within the lung volume only. Shape index and curvedness features are used to detect initial nodules [12]. It involves thresholding to define seed points within the lung volume which is obtained after the lung segmentation phase. Then seed points are grown to cluster by hysteresis thresholding. After discarding the smallest clusters, then they used KNN classification to classify the nodules.

Two levels of KNN are implemented with 18 features in first level and 135 features in final level. In the random selection of 813 scans from the screening study, a sensitivity of 80% with an average 4.2 false positive per scan is achieved.

In the paper Aoyama et al. [14], they classified nodules as cancerous/non-cancerous through a combination of LDA, heuristics, and a trained Artificial Neural Network.

M. Antonelli et al. [15] presented a more novel approach - they described a method for nodule segmentation and classification using a combination of image processing techniques and 3D geometric features. In this they used a database consisting of eight CT scans, each with about 300 slices. A slice is a 512× 512 pixels matrix with slice thickness of 1.25 mm. All the malignant nodules were correctly detected and recognized, even in the case in which nodules were next to blood vessels or the pleura, thus reducing false-positive number. First they removed the background (i.e., the pixels with the same grey level as the lungs but located outside the chest) from the image. After discarding few slices that do not contain lung images they classified remaining slices into three groups mainly upper, middle, and lower parts of the lung volume. They applied a different thresholding technique depending on the group the given slice belongs to and then morphological opening and closing operators are applied to improve the image and border definition. After reducing border size, the two longest border chains are chosen as pulmonary lobes. Then they reconstructed the pulmonary lobes borders so as to reinsert the nodules adjacent to the pleura previously suppressed by the thresholding operator. Finally, they applied a region filling operator to the pulmonary lobes chains in order to reintroduce the original values of grey levels inside the lungs. This result is used as input to the nodule detector. In order to detect the nodules they adopted a method that uses 3D shape information to identify spherical regions with a given grey level and Gaussian Filter.

In the paper Jan Hendrik Moltz et al [18], one interesting problem is answered which is the segmentation of Juxtapleural lung nodules. Pulmonary nodules that have extensive contact to the chest wall or other structures of similar density are a special challenge for potential nodule segmentation, which leads to increase in false positive rate. They proposed a ray casting approach to identify points at the visible boundary of the nodule and then approximate its shape

by an ellipsoid that is a least square fit of these points. The adjacent structures are cut off by morphological processing within a dilated version of this ellipsoid. Evaluation on a 333 juxtapleural nodules showed that their method yielded good results and it was integrated into a general segmentation algorithm for lung nodules with no substantial increase in computation time.

Proposed algorithm of Jan Hendrik Moltz et al consists of three parts with an aim to solve the problem of juxtapleural nodules. First, identify the point on the nodule boundary by region growing and subsequent ray casting from the seed point. Second, calculate the ellipsoid that approximates the shape of the nodule. Third, apply the convex hull operation. While in 71% of the cases the result of the original algorithm was classified as good, their extension increased this proportion to 89%.

## 2.4. Deep Learning in Medical Imaging

Recently, CNNs have become more popular in the general medical image processing community. Roth et al. [16] trained deep convolutional neural networks to be able to detect sclerotic spine metastases, lymph nodes, and colonic polyps and were able to increase sensitivity by 15-30% for each of the tasks. Furthermore, their results showed ConvNets generalize well to different medical imaging CAD applications.

In the paper Xiaojie et al [17], they have trained 3D ConvNet on LIDC/IDRI data-set. They generated potential nodules using a local geometric model based filter, classified candidates using 3D CNN. Their architecture contained 5 layers; 3 pooling layers (2x2 Kernel) and 2 fully connected layers. There are a very large number of non-nodule training examples naturally generated by the high-sensitivity candidate generation step. For each candidate clusters that are not true nodules, they applied a sampling grid with certain spacing to sample a sufficiently large set of non-nodule training examples that are also aligned to the local principal direction. The network is trained using a stochastic gradient descent algorithm with an adaptive learning rate scheme. Their model was able to get a sensitivity of 90% and 5 false positives per scan. While 2D CNN with multi-slice representation has been popular in medical image analysis

applications, it unavoidably leads to considerable loss of information. Here, they compared the proposed system to an alternative 2D CNN method with the popular tri-planar 2D representation. In this 2D CNN implementation, they kept everything the same as in the proposed method expect that a tri-planar 3-channel 2D representation was used. The 2D CNN architecture also had the same structure as the proposed 3D CNN architecture expect that 3D convolutional kernels were replaced with 2D convolutional kernels.

# Chapter 3

# Proposed Work

## 3.1. Preprocessing

Each slice of a CT scan is enhanced by preprocessing techniques with an aim to augment the image data for better lung segmentation and nodule extraction. We initially implemented Weiner Filter, Histogram Equalization, and Median Filter techniques on the CT scan slices.

### 3.1.1. Median Filter

The main idea of the median filter is to run through the signal entry by entry, replacing each entry with the median of neighboring entries. The pattern of neighbors is called the "window", which slides, entry by entry, over the entire signal [19]. Median filter is a nonlinear digital filtering technique which is used to remove noise from an image. It is one of the widely used preprocessing techniques. It preserves edges while removing the noise. Figure 3.1 shows the working of a medial filter.

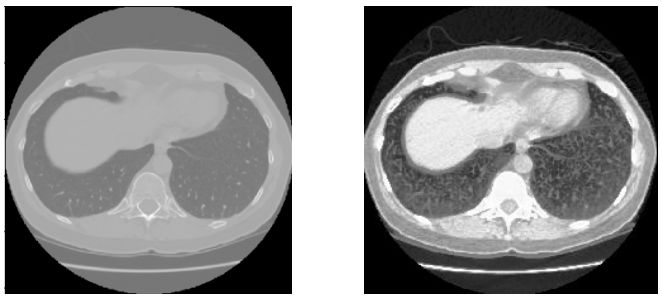

***Figure 3.1: Original CT slice (left), Slice after Median Filter (right)***

### 3.1.2. Weiner Filter

Weiner Filter aims to compute a statistical estimate of an unknown signal with the help of a related signal as an input and filtering it to produce the estimate [20]. It can be used to filter out

the noise from the original signal to provide an enhanced signal as output. We implemented Weiner Filter to remove noise from the original slice which would increase the accuracy of lung segmentation. The output slice of the Weiner Filter is shown in Figure 3.2.

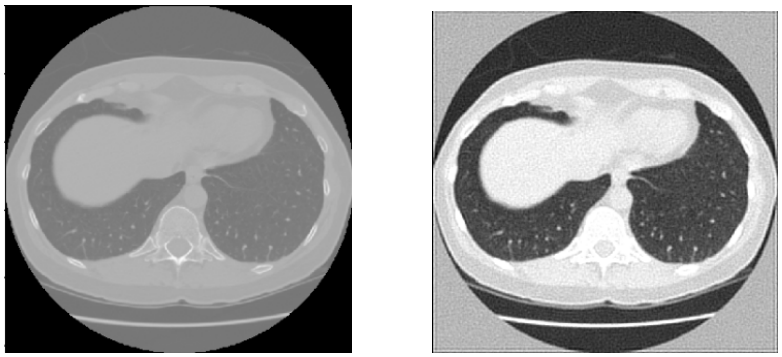

***Figure 3.2: Original CT slice (left), Slice after Weiner Filter (right)***

## 3.1.3. Histogram Equalization

Histogram Equalization aims to adjust the contrast through image histogram [21]. This adjustment can enable a better intensity distribution on the histogram. Histogram Equalization can accomplish it by efficiently spreading the most frequent intensity values. Our CT slices have lighter pixel intensities in both background and foreground. Histogram Equalization has effectively spread the pixels which enhanced the lung region. Figure 3.3 shows working of Histogram Equalization.

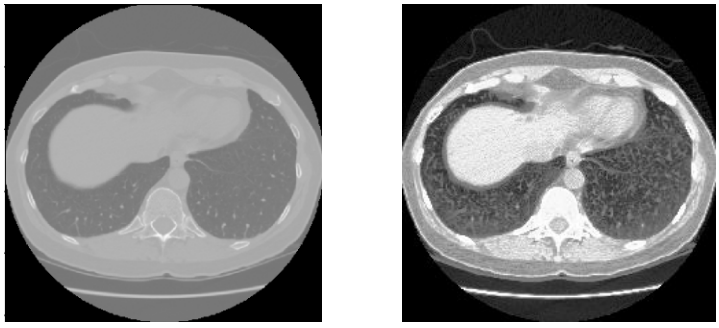

***Figure 3.3: Original CT slice (left), Slice after Histogram Equalization (right)***

## 3.2. Lung Segmentation

Lung regions are extracted from the preprocessed images through segmentation algorithms like Otsu segmentation, Chan Vese segmentation and K-means Clustering.

### 3.2.1. Otsu Segmentation

Otsu Segmentation is used to automatically perform clustering-based image thresholding, or, the reduction of a gray scale image to a binary image [22]. The algorithm assumes that the image contains two classes of pixels following bi-modal histogram (foreground pixels and background pixels), it then calculates the optimum threshold separating the two classes so that their combined spread is minimal, or equivalently, so that their inter-class variance is maximal. Figure 3.4 shows the resulting slice after lung segmentation through Otsu.

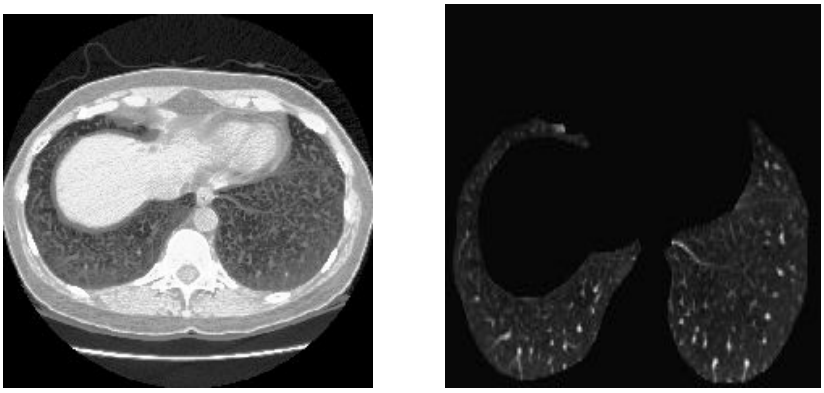

***Figure 3.4: Slice after Histogram Equalization (left), Slice after Otsu Segmentation (right)***

### 3.2.2. Chan Vese Segmentation

The Chan-Vese algorithm is an example of a geometric active contour model. Such models begin with a contour in the image plane defining an initial segmentation, and then we evolve this contour according to some evolution equation [23]. The goal is to evolve the contour in such a way that it stops on the boundaries of the foreground region. Figure 3.5 shows the result after Chan Vese Segmentation. As shown in figure 3.5, the output of Chan vese brightened the potential nodules which might be helpful in detecting it further.

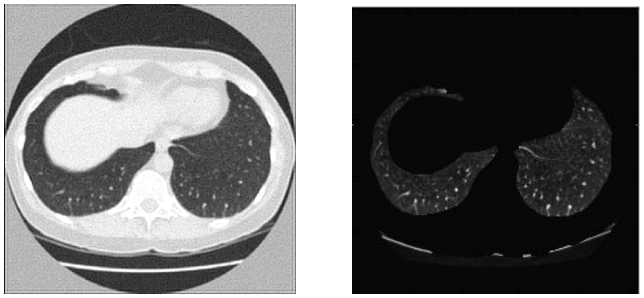

*Figure 3.5: Slice after Weiner Filter (left), Slice after Chan Vese Segmentation (right)*

### 3.2.3. K-Means Clustering

K-means is one of the simplest unsupervised learning algorithms that solve the well-known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) [24]. The main idea is to define k centers, one for each cluster. These centers should be placed in a cunning way because of different location causes different result. So, the better choice is to place them as much as possible far away from each other. The next step is to take each point belonging to a given data set and associate it to the nearest center. Figure 3.6 shows output after K-Means clustering.

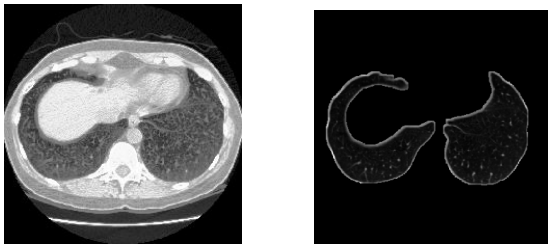

*Figure 3.6: Slice after Histogram Equalization (left), Slice after K-Means Clustering (right)*

Based on our observations, out of all the methods, combination of Histogram Equalization and Otsu Segmentation worked more effectively than the other combinations.

## 3.3. Potential Nodule Segmentation

After segmenting out the lung regions from the CT slices, we used Gaussian Filter to enhance the data for better nodule segmentation, which is done using connected components analysis method.

### 3.3.1. Gaussian Filter

Gaussian filter is a linear filter. It is usually used to blur the image or reduce noise. The Gaussian filter will blur edges and reduce the contrast [25]. This is helpful in proper edge detection and to identify the different nodule regions present in lung. A series of experiments are conducted on various blur radii, out of which we chose a radius of 0.5 mm as it gave satisfactory results the connected components detection. The output after Gaussian Filter can be seen in Figure 3.7.

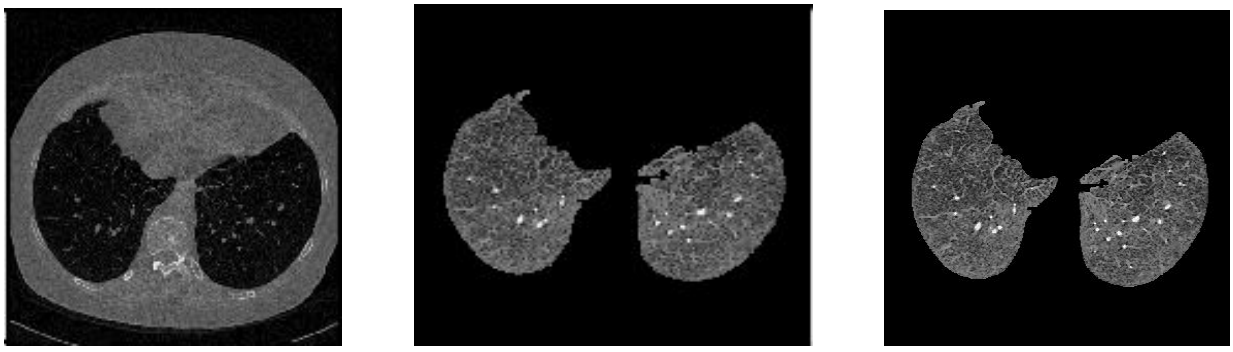

***Figure 3.7: Original Slice (left), Slice after lung segmentation (middle), Slice after Guassian Filter (right)***

### 3.3.2. Connected Components Analysis

The filtered output obtained after applying Gaussian Filter is made to go through Connected Components Analysis technique to get all the regions present inside the lung portion.

***Figure 3.8: Slice after Guassian Filter (left), Slice after segmenting nodules (middle), Slice after drawing boundary boxes around nodules (right)***

We implemented it using region props of a python library named Scikit-Image. Boundary boxes are drawn around the potential nodule regions to visualize them. Figure 3.8 shows the output of slices after connected components analysis technique is applied. Features are extracted from the segmented nodules. We segmented out the potential nodules as 64x64 PNG images with the help of the center co-ordinates of nodules, which we obtained using Scikit Learn - region props technique. We chose 64x64 because all the nodules are less than this size, and a reasonable amount of locality pixels are present. This ensured us a faster running time while extracting features, while training classical machine learning classifiers and deep learning models. Moreover, presence of locality pixels improved accuracy of our classifiers. Potential nodule is shown in Figure 3.9 which is used in the later stages for extracting features and training classifiers.

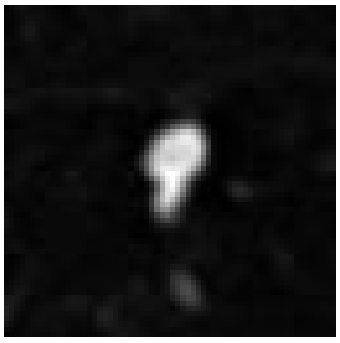

***Figure 3.9: The final 64x64 image showing a potential nodule***

### 3.3.3. Object Detection using deep learning models

Recently, there are many applications of object-detection in self-driving cars, face-detection, pedestrian detection, and video surveillance. The importance of nodule detection is to narrow down the search domain, so that nodule can be classified easily.

In our project, we want to detect potential nodules from CT scan images, it is a challenging task due to the following reasons:

- The mean area of nodules to be detected is around 7 square pixels, which is infinitesimal as compared to the size of the image.

- There are many chances of detecting false positives, which can be reduced by nodule classifier.
- Need to configure pre-trained Faster-RCNN [34] for detecting very small nodules. For that, we have used Tensorflow Object-detection API [35].

We had extracted 512 x 512 CT scan slice containing nodules using annotations. The first task is to extract Region of Interest (ROI) i.e. lungs, then will train Faster-RCNN model with transfer learning, on pre-processed images to detect nodules.

### 3.3.3.1. Preprocessing and segmentation of ROI

For training of our faster-rcnn model, it is important to remove the noise and extract ROI. Following are the preprocessing techniques we followed.

**K-Means Clustering**

The original image is mean normalised and then k-means clustering with K = 2 is done on that so that we can cluster two parts of the lung so as to get only the ROI. K-means clustering [37] is a type of unsupervised learning, which is used when you have unlabeled data (i.e., data without defined categories or groups). The goal of this algorithm is to find groups in the data, with the number of groups represented by the variable K.

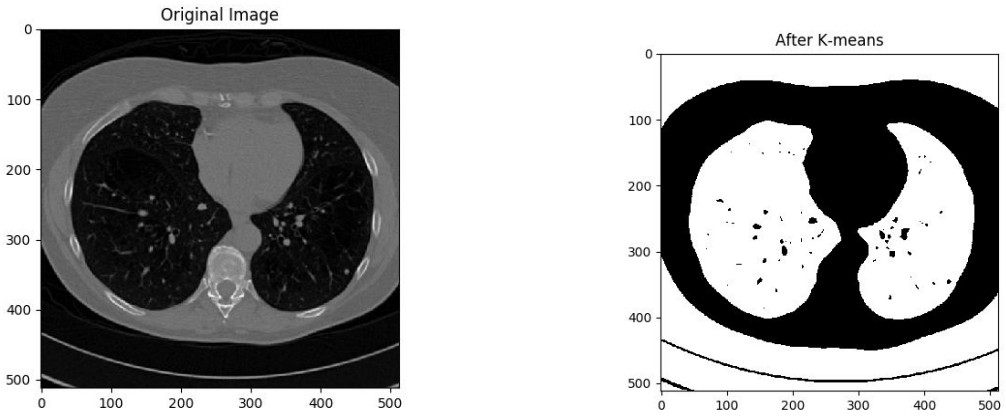


**Figure 3.10 : Original Image, Image after mean normalisation and k-means clustering.**

The algorithm works iteratively to assign each data point to one of K groups based on the features that are provided. We get the output as shown below.

**Erosion and Dilation**

Image obtained from the above step is eroded and then dilated. Morphological erosion [39] sets a pixel at (i, j) to the minimum over all pixels in the neighborhood centered at (i, j). The structuring element, selem, passed to erosion is a boolean array that describes this neighborhood. Whereas, morphological dilation [39] sets a pixel at (i, j) to the maximum over all pixels in the neighborhood centered at (i, j). Dilation enlarges bright regions and shrinks dark regions. We get the output as shown below.

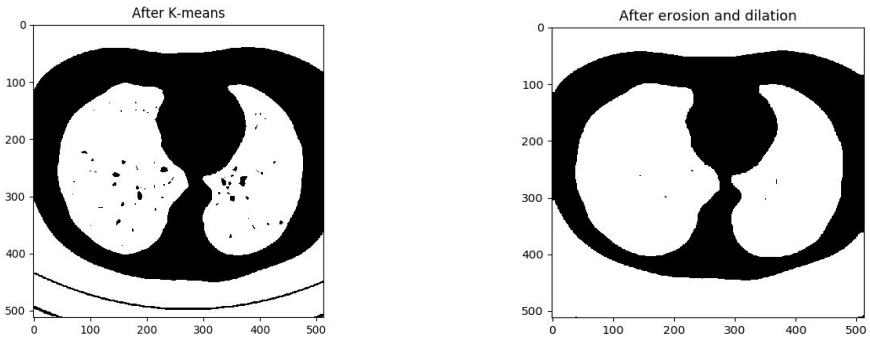


**Figure 3.11: Image obtained after k-means (left), image after applying erosion and dilation (right).**

**Connected Component Regions and segmenting ROI**

We have implemented the given approach to get all the connected regions present in the region the image to further detect the lung regions which are ROI.To get the mask of the lung regions, we applied connect component regions algorithm [40] to get all the regions connected and later we consider only the regions which determine the shape and size of the lung. We considered the largest two regions as lung regions and considered for making a mask. After obtaining the mask, we have multiplied the mask with the origin image, to get only the region of interest by removing

all the unwanted areas. This is necessary so as to train a model by providing only the region of interest so that the model would not be biased.

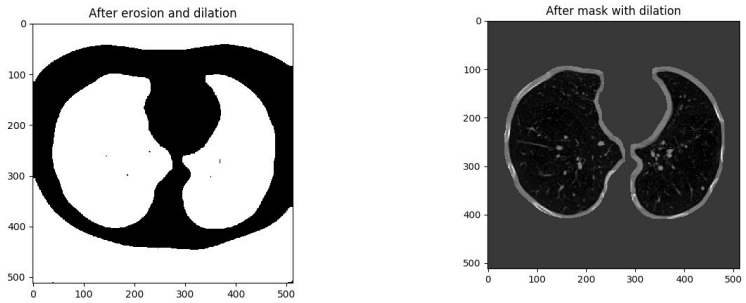


**Figure 3.12: eroded and dilated image (left), after multiplying mask with original image and applying dilation (right).**

### 3.3.3.2 Faster-RCNN

The transfer learning or to use weights of pre-trained CNN model for new-task has been successful, we have used pre-trained Faster R-CNN model [35] trained on Common Objects in Context (COCO) dataset. The challenge here is that this model can detect object not small than 44px [36], where Intersection over Union (IoU) threshold t = 0.5 and with anchor stride d = 16. In our dataset, the distribution of area of nodules, as shown in figure 3.13

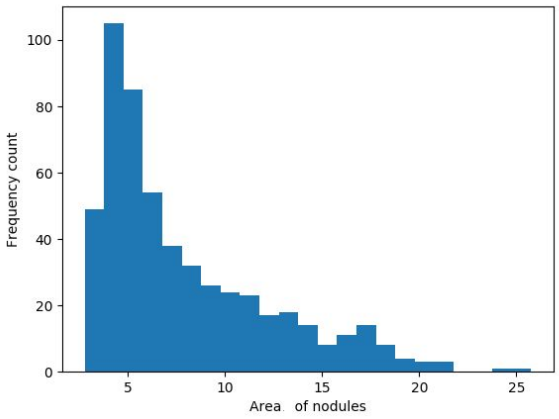


***Figure 3.13 : Distribution of size of nodules***

The mean area of nodules is 7px (approximately) which is very low than smallest object detected by pre-trained Faster-RCNN model. Thus, we need to change the anchor size of Faster R-CNN to detect nodules accurately.

**Anchors**

An anchor is a box which is very important in Faster-RCNN. The figure 4 illustrates the 9 anchors at (320,320) of image size of 600 x 800.

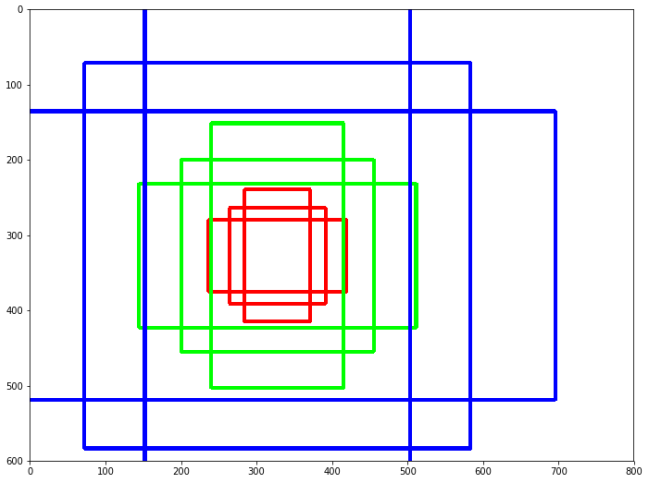


***Figure 3.14 : 9 anchors at (320,320)***

The anchors can be modified according to the application; there are 3 parameters: scales, aspect ratio, and stride to generate different type of anchors.

1. Scales are the area of the anchors to be generated. In the above figure, red, green and blue anchors of 128x128, 256x256, 512x512 respectively.
2. Aspect ratio is the ratio of width to height of anchors to be generated. In the figure above, for each scale, 3 boxes of width to height ratio 1:1, 1:2, and 2:1 are generated.
3. The stride is number of pixels to be skipped between two successive generation of anchors. Thus, for 512x512 image, there will be 1024 (32x32) positions for stride 16, and there will be total 9216 (1024x9) boxes. Region Proposal Network (RPN) of Faster R-CNN, ranks these anchors which has highest probability of Faster R-CNN.

As shown in figure 4, the anchors have better coverage than sliding window.

The relation between, $s_g$ the minimum size of the pixels to be detected, feature stride $d$ and $t$ is IOU threshold, can be mentioned as follows:

$$\frac{d(t+1) + d\sqrt{2t(t+1)}}{2-2t} \leq s_g \text{ [36]}$$

We rescaled 1.5 times the original image, which makes $s_g$ =11px, with stride d=8 and t=0.1, satisfies the above equation. Hence, we choose the scales of anchors A={8,12,16, 32,64,100}.

The aspect ratio of anchors are 1:1 and 1:2 and stride is 8. The above configuration of Faster R-CNN achieved 40% mean Average Precision (mAP) on test data set.

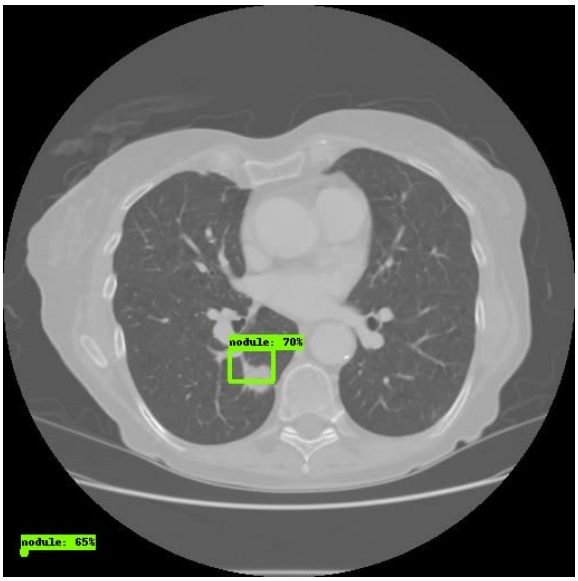


***Figure 3.15 : Sample image from test data-set.***

## 3.4. Feature Extraction

Shape characteristics analyze the spatial distribution of gray values, by computing local features at each point in the image. Shape feature is the most intuitive visual feature, which can be used to describe the main medical signs of CT image of pulmonary nodule ROI. The extracted components of the shape features mainly include area, equivalent diameter, perimeter,

circularity, rectangularity, elongation, eccentricity, Euler number, centroid of the region [26] [27].

1. **Area**

$$\sum_{x=1}^{N} \sum_{y=1}^{M} f(x,y) \text{ [26].....(1)}$$

Where f(x, y) is the pixels of the target and M and N are the length and width, respectively.

2. **Equivalent Diameter**

Equivalent diameter of a region is the diameter of a circle with the same area as of the region.

3. **Perimeter**

$$C = \sum_{i=1}^{M} \sum_{j=1}^{N} p(i,j) \text{ [26] ..... (2)}$$

Where p (i, j) is the pixels of the target edge and M and N are the length and width, respectively.

4. **Circularity**

$$Ro = \frac{C^2}{4\pi S} \text{ [26]..... (3)}$$

Circularity describes object shape that is close to the degree of circular, where $S$ is the area of the target region and $C$ is circumference of the target region. $0 < Ro < 1$ and $Ro$ value reflects the complexity of the measurement boundary; the shape is more complex and the $Ro$ value is smaller.

5. **Rectangularity**

$$R = \frac{S}{(W*H)} \text{ [27]..... (4)}$$

Where $S$ is the area of the target region and $H$ and, $W$ are the length and width, respectively.

6. **Elongation**

$$E = \frac{min(H, W)}{max(H, W)} \text{ [27]}\ldots\ldots. (5)$$

Elongation can distinguish different shapes of the images (such as circle, square, ellipse, thin and long, and short and wide), where *H* and *W* are the length and width, respectively.

**7. Eccentricity**

In mathematics, the eccentricity, denoted e, is a parameter associated with every conic section. It can be thought of as a measure of how much the conic section deviates from being circular. An ellipse is fitted to the ROI and the eccentricity is calculated. The eccentricity is the ratio of the distance between the foci of the ellipse and its major axis length.

$$E = \frac{c}{a} \text{ [27]}\ldots. (7)$$

Where c is the distance between center to either of the foci and a is length of semi-major axis.

**8. Euler Number**

Euler number is defined as the number of objects in the region minus the number of holes in those objects. We have used 8-connectivity to compute the Euler number measurement. This would be useful as a feature as we know that the nodule is solid in nature and it shouldn't consist of any holes in it.

$$E = N - H \text{ [27]} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (8)$$

Where N is number of regions and H is number of holes.

**9. Centroid**

We used centroid coordinates (x, y) of the region as two additional features. We also used Hu Moments. Hu Moments are helpful for describing, characterising and quantifying the shape of a region. Moment invariants provides information about image patterns regarding the image scaling, translation and rotation. In this paper we have considered

three invariants which are scale invariance, translation invariance and rotational invariance.

## 3.5. Classical Machine Learning Algorithms

Below are some of the machine learning algorithm we have used for training the model for the feature set we have extracted from the dataset.

### 3.5.1 Logistic Regression

For input $x$ and parameters $\theta$, the hypothesis $h_\theta(x)$ can be defined as

$$h_\theta(x) = sigmoid(\theta^T x)$$

$$sigmoid(z) = \frac{1}{1+e^{-z}} \text{ [28]} \quad \text{.....(9)}$$

The cost function of Logistic Regression $J(\theta)$ for output categorical variable y for m training examples can be defined as

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^i log\, h_\theta(x^i) + (1-y^i) log(1-h_\theta(x^i))\right] \text{ [29]} \quad \text{.....(10)}$$

The algorithm will choose the parameters $\theta$ which will minimise the cost function $J(\theta)$ and will predict the outcomes accordingly.

In our experiment the accuracy using logistic regression 27.3% and false positive rate of 1.1% .

### 3.5.2 Random Forests

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random forests are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. The generalization error for forests converges a.s. to a limit as the number of trees in the forest becomes large. The generalization error of a forest of tree

classifiers depends on the strength of the individual trees in the forest and the correlation between them [3].

In our experiment the accuracy using random forests true positive rate of 29.8% and a false positive rate of 2.1%

### 3.5.3 Support Vector Machines (SVM)

An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall [4].

The cost function of SVM is very similar to the cost function of logistic regression. The only difference is of $cost_1(\theta^T x^{(i)})$ and $cost_0(\theta^T x^{(i)})$. $cost_1(\theta^T x^{(i)})$ is the value when we substitute y=1 in the cost function of the logistic regression and $cost_0(\theta^T x^{(i)})$ is the value when we substitute y=0. On minimising this cost function we get the best hyper plane which try to maximise the margin between the classes. The SVM cost function is as shown below:

$$min\ \theta\ \sum_{i=1}^{m}[y^{(i)}cost_1(\theta^T x^{(i)})\ +\ (1-y^{(i)})cost_0(\theta^T x^{(i)})] + \tfrac{1}{2}\sum_{i=1}^{n}\theta_j^2\ [31] \quad \text{.....(11)}$$

The second term is L2 regularisation which will try to penalise the weights i.e it will try to have smaller weights. The first term is actual cost function on input $x$ with $\theta$ *parameters* and class variable y.

### 3.5.4 K-Nearest Neighbors (k-NN)

k-nearest neighbors algorithm (k-NN) is a non-parametric method used for classification and regression. In both cases, the input consists of the k closest training examples in the feature space. The output depends on whether k-NN is used for classification or regression. In k-NN classification, the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small) [5]. If k = 1, then the object is simply assigned

to the class of that single nearest neighbor. In k-nearest neighbors, a query point is assigned to the class which has the most representatives within the nearest k neighbors of the point.

The drawback is to choose the value of *k.* In our experiment we got the best results at K=2 and the accuracy using k-NN to detect nodules,with a true positive rate of 0.521 and false positive rate of 0.076

### 3.5.5 Naive-Bayes Classifier

Naive Bayes classifiers are a collection of classification algorithms based on Bayes' Theorem. To understand the classifier we need to understand the Bayes theorem which can be defined as below:

$$P(A \mid B) = \frac{P(B|A)\,P(A)}{P(B)} \quad [33] \qquad \text{.....(12)}$$

Now, with regards to Machine Learning Bayes theorem can be applied as below

$$P(y \mid X) = \frac{P(X|y)\,P(y)}{P(X)} \qquad \text{.....(13)}$$

where, y is class variable and X is a dependent feature vector (of size n).

If we put a naive assumption to Bayes' theorem which is independence among the features $x_0, x_1, x_2, .....x_n$. On imposing such a constraint we can have classifier model which can be shown as below:

$$y = argmax_y P(y) \prod_{i=1}^{n} P(x_i|y) \qquad \text{.....(14)}$$

## 3.6. Segmentation of Juxtapleural lung nodules

Pulmonary nodules that have extensive contact to the chest wall and other structures of similar density are a special challenge to automatic segmentation.

They increase false positive rate as the features of nodules get corrupted. It is required to remove the walls and extract only the nodules. For that we calculated an ellipsoid that approximates the shape of the nodule. On applying convex hull operation, only the nodule region is obtained. Then a nodule mask of this region is created. This mask is multiplied with the original image, which removed the wall and gave only nodule as the output. Below we show a series of technique used for removing the lung's wall region from the image. Figure 3.16 (a) shows the normalized image in which a nodule is present. First we perform connected components technique and get the region consisting of the nodule along with the wall as shown in Figure 3.16

(b). Figure 3.16 (c) shows the convex hull image of the Figure 3.16 (b). The convex hull is the set of pixels included in the smallest convex polygon that surround all white pixels in the input image. So to get the convex hull as our need we invert the pixel values in Figure 3.16 (b) and find a convex hull which looks like Figure 3.16 (c). Now the convex hull is used as a mask and is multiplied with Figure 3.16 (a) to get only the nodule without the wall region as shown in Figure 3.16 (d). Now this image is used for extracting the features and storing in a file.

***Figure 3.16: (a) Normalized Image, (b) Image containing both wall and nodule (white region) (c) Image after applying convex hell operator, (d) Final Image with only nodule***

# 3.7. Convolutional Neural Networks

## 3.7.1. Introduction to Convolutional Neural Networks

Convolutional Neural Networks (**ConvNets** or **CNNs**) are a category of Neural Networks that have proven very effective in areas such as image recognition and classification. ConvNets have been successful in identifying faces, objects and traffic signs apart from powering vision in robots and self-driving cars.

LeNet was one of the very first convolutional neural networks which helped propel the field of Deep Learning. This pioneering work by Yann LeCun was named LeNet5 after much previous successful iteration since the year 1988 [34]. At that time the LeNet architecture was used mainly for character recognition tasks such as reading zip codes, digits, etc.

ConvNets derive their name from the “convolution” operator.

The primary purpose of Convolution in case of a ConvNet is to extract features from the input image.

Convolution preserves the spatial relationship between pixels by learning image features using small squares of input data.

A filter slides over the input image (convolution operation) to produce a feature map. The convolution of another filter (with the green outline), over the same image gives a different feature map as shown. It is important to note that the Convolution operation captures the local dependencies in the original image. Also notice how these two different filters generate different feature maps from the same original image.

The size of the Feature Map (Convolved Feature) is controlled by three parameters that we need to decide before the convolution step is performed:

**Depth:** Depth corresponds to the number of filters we use for the convolution operation.
**Stride:** Stride is the number of pixels by which we slide our filter matrix over the input matrix. When the stride is 1 then we move the filters one pixel at a time.

**Zero-padding:** Sometimes, it is convenient to pad the input matrix with zeros around the border, so that we can apply the filter to bordering elements of our input image matrix.

An additional operation called ReLU has been used after every Convolution operation. ReLU stands for Rectified Linear Unit and is a non-linear operation. ReLU is an element wise operation (applied per pixel) and replaces all negative pixel values in the feature map by zero. The purpose of ReLU is to introduce non-linearity in our ConvNet; since most of the real-world data we would want our ConvNet to learn would be non-linear (Convolution is a linear operation – element wise matrix multiplication and addition, so we account for non-linearity by introducing a nonlinear function like ReLU). Spatial Pooling (also called subsampling or down sampling) reduces the dimensionality of each feature map but retains the most important information. Spatial Pooling can be of different types: Max, Average, Sum etc. In case of Max Pooling, we define a spatial neighborhood (for example, a 2×2 window) and take the largest element from the rectified feature map within that window. Instead of taking the

largest element we could also take the average (Average Pooling) or sum of all elements in that window. In practice, Max Pooling has been shown to work better.

The Fully Connected layer is a traditional Multi-Layer Perceptron that uses a softmax activation function in the output layer (other classifiers like SVM can also be used, but will stick to softmax in this post). The term "Fully Connected" implies that every neuron in the previous layer is connected to every neuron on the next layer.

### 3.7.1. Proposed ConvNet Model for reducing false positives

**Input:** 64 × 64 CT Slice

**Layer1:** Conv Layer1 [64 × 64] - Channels = 64

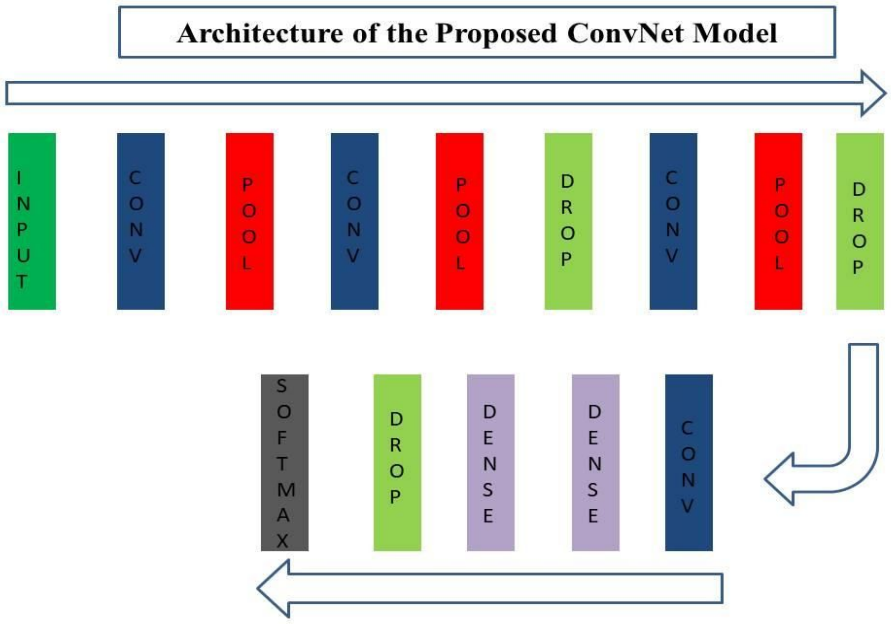


***Figure 3.17: Design of the proposed ConvNet architecture***

**Layer2:** Pool Layer1 [32 × 32] - Stride = 2

**Layer3:** Conv Layer2 [32 × 32] - Channels = 128

**Layer4:** Pool Layer2 [16 × 16] - Stride = 2

**Layer5:** Drop Out1 [Rate:0.2]

**Layer6:** Conv Layer3 [16 × 16] - Channels = 32

**Layer7:** Pool Layer3 [8 × 8] - Stride = 2

**Layer8:** Drop Out2 [Rate:0.4]

**Layer9:** Conv Layer4 [8 × 8] - Channels = 64

**Layer10:** Dense Layer1 [4096]

**Layer11:** Dense Layer2 [4096]

**Layer12:** Drop Out3 [Rate:0.4]

**Layer13:** Softmax

## 3.7. Oversampling of the Data Set

During the training of different machine learning algorithms, one of the challenges we faced is having less positive examples compared to that of negative examples. The positive and negative examples were in the ratio of 0.003: 1, which is why we weren't able to train the models better and not able to make use of all the negative data we had. As we know more the data better the model. To overcome this problem we have generated our own data with existing data known as data augmentation, which will increase the positive examples by 6 fold. Following are the different operation we performed on the positive data.

- **Rotation**
- **Flipping**

**Rotation:** The model has to recognize the nodule present in any orientation. So rotating the image at 90 degrees will not add any background noise in the image and would do our work of data augmentation. Three rotations have been performed, i.e 90, 180, 270 degrees respectively.

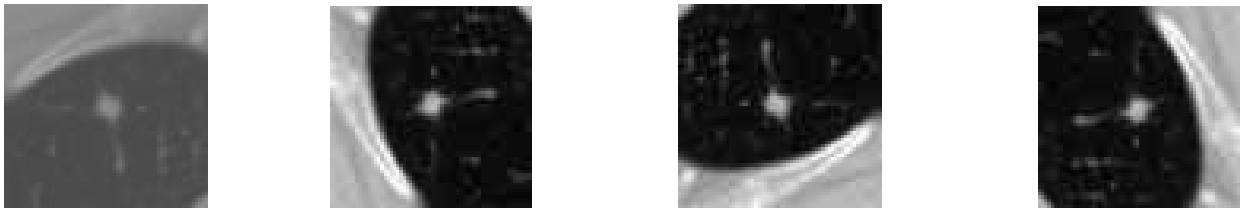

*Figure 3.18: (a) Original Image, (b) 90 degrees Rotated Image, (c) 180 degrees Rotated Image, (d) 270 degrees Rotated Image*

**Flipping:** Image flips are done additionally to augment the data. Once image is vertically flipped, once it is horizontally flipped and later it is flipped with respect to the origin which is known as transpose. The results are shown in the figure below.

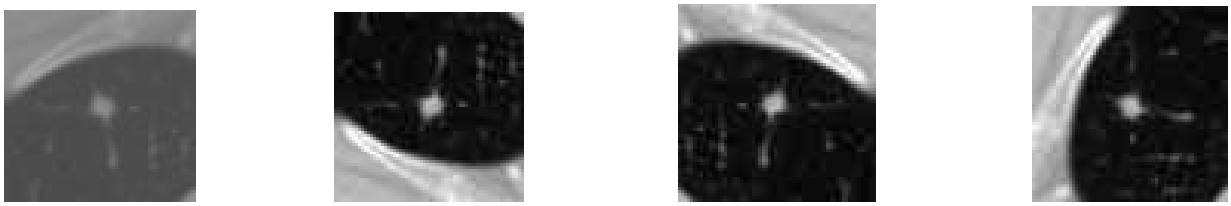

*Figure 3.19: (a) Original Image, (b) Vertical Flip Image, (c) Horizontal Flip Image, (d) Transposed Image*

# Chapter 4

# Simulation Results

Here we present the experiments performed and the results obtained for nodule detection for the database and analyze the performance of different algorithms we have used. Apart from the results obtained from the original dataset, we also got results from the oversampled data. In case of classical machine learning algorithms, experiments are performed with and without removing walls. However, ConvNet models are not implemented without walls, as substantial change in the accuracy is not observed. Our most successful model obtained a true positive rate of 94.43% and a false positive rate of 4.01% on an oversampled data.

## 4.1. Logistic Regression

| Pos:Neg | **80:20** | | | **70:30** | | | **60:40** | | | **50:50** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Train:Test | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 |
| TestAccuracy | 79.7 | 80.3 | 81.59 | 75.25 | 75.1 | 75.77 | 71.3 | 73.04 | 73.17 | **76.1** | 74.6 | 73.9 |
| TPR | 98.5 | 98.3 | 98.3 | 78.8 | 78.3 | 77.9 | 75.1 | 75 | 73.7 | **73** | 69 | 68 |
| Average TPR | 98.3 | | | 78.3 | | | 74.6 | | | 70 | | |
| FPR | 86.7 | 85.3 | 82.9 | 32 | 31.9 | 29.3 | 34.7 | 30 | 27.7 | **20.7** | 19.5 | 20 |
| Average FPR | 84.9 | | | 31 | | | 29.8 | | | 20 | | |

***Table 4.1: Results on Logistic Regression – Without Walls – Original Dataset***

| P:N | **80:20** | | | **70:30** | | | **60:40** | | | **50:50** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Train:Test | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 |
| TestAccuracy | 81.9 | 81.8 | 81.86 | 74.04 | 73.67 | 73.69 | 73.18 | 73.01 | 73.54 | **74.5** | 74.34 | 73.99 |
| TPR | 98 | 98 | 98.2 | 78.1 | 77.7 | 78 | 75.6 | 75.2 | 75.7 | **71.4** | 70.9 | 69.5 |
| Average TPR | 98 | | | 77.93 | | | 75.5 | | | 70.6 | | |
| FPR | 86.1 | 85.4 | 86.1 | 35.8 | 36.2 | 36.3 | 30.6 | 30 | 29.9 | **22.4** | 22.3 | 21.4 |
| Average FPR | 85.8 | | | 36.1 | | | 30.17 | | | 22.03 | | |

***Table 4.2: Results on Logistic Regression – Without Walls – Oversampled Dataset***

| P:N | **80:20** | | | **70:30** | | | **60:40** | | | **50:50** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Train:Test | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 |
| TestAccuracy | 79.4 | 79.5 | 80.85 | 75.25 | 75.4 | 75.1 | 68.8 | 70.39 | 70.41 | 72.4 | **73.25** | 72.76 |
| TPR | 99.2 | 98.5 | 98.5 | 82.3 | 81.4 | 80.1 | 73.6 | 73.8 | 73.4 | 69.8 | **70.7** | 69.5 |
| Average TPR | 98.7 | | | 81.2 | | | 73.6 | | | 70 | | |
| FPR | 90.7 | 89.9 | 87.1 | 39.1 | 37.3 | 36.7 | 38.6 | 35 | 34.3 | 24.8 | **24.1** | 23.9 |
| Average FPR | 89.2 | | | 37.7 | | | 35.97 | | | 24.27 | | |

***Table 4.3: Results on Logistic Regression – With Walls – Original Dataset***

| P:N | **80:20** | | | **70:30** | | | **60:40** | | | **50:50** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Train:Test | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 |
| TestAccuracy | 81 | 81 | 81 | 74.3 | 73.35 | 73.5 | 72.6 | 72.4 | 72.75 | **72.6** | 72.2 | 72 |
| TPR | 98.1 | 98.1 | 98.5 | 81.2 | 80 | 80.2 | 77.7 | 77.1 | 77.4 | **71.2** | 70.5 | 69.3 |
| Average TPR | 98.2 | | | 80.4 | | | 77.4 | | | 70.33 | | |
| FPR | 91.4 | 90.4 | 92 | 42.1 | 42.7 | 42.1 | 35.3 | 35 | 34.6 | **26** | 26 | 25.1 |
| Average FPR | 91.2 | | | 42.3 | | | 34.97 | | | 25.7 | | |

***Table 4.4: Results on Logistic Regression – With Walls – Oversampled Dataset***

## 4.2. Random Forest

| P:N | **80:20** | | | **70:30** | | | **60:40** | | | **50:50** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Train:Test | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 |
| TestAccuracy | 80.88 | 80.9 | 82.3 | 77.57 | 77.14 | 77.19 | 75.2 | 76.87 | 75.7 | **80.51** | 79.75 | 77.8 |
| TPR | 98.5 | 97.8 | 98 | 80.8 | 78.6 | 78.8 | 76.9 | 76.9 | 74.1 | **74.1** | 72.1 | 71.3 |
| Average TPR | 98.1 | | | 79.4 | | | 75.97 | | | 72.5 | | |
| FPR | 81.3 | 80.7 | 77.9 | 28.9 | 25.9 | 26.6 | 27.3 | 23.2 | 21.7 | **12.8** | 12.3 | 15.6 |
| Average FPR | 79.96 | | | 27.1 | | | 24.07 | | | 13.57 | | |

***Table 4.5: Results on Random Forest – Without Walls – Original Dataset***

| P:N | **80:20** | | | **70:30** | | | **60:40** | | | **50:50** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Train:Test | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 |
| TestAccuracy | 87 | 86.1 | 85.7 | 82.7 | 81.88 | 81 | 84.17 | 83.08 | 82.98 | **85.24** | 84.59 | 84.1 |
| TPR | 98.1 | 98.6 | 98.6 | 82.3 | 82 | 81.6 | 82.3 | 81.6 | 81.6 | **80.7** | 79.8 | 78.9 |
| Average TPR | 98.4 | | | 81.9 | | | 81.83 | | | 79.8 | | |
| FPR | 63.6 | 65.7 | 67.8 | 16 | 18.3 | 20.3 | 12.8 | 14.5 | 14.9 | **10.3** | 10.7 | 10.6 |
| Average FPR | 65.7 | | | 18.2 | | | 14.07 | | | 10.54 | | |

***Table 4.6: Results on Random Forest – Without Walls – Oversampled Dataset***

| P:N | **80:20** | | | **70:30** | | | **60:40** | | | **50:50** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Train:Test | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 |
| TestAccuracy | 80.88 | 81.5 | 82.2 | 77.57 | 75.6 | 77.44 | 74.4 | 74.66 | 74.4 | **79.4** | 79.26 | 77 |
| TPR | 99.6 | 99.3 | 98.7 | 83.1 | 79.3 | 80.1 | 74.5 | 74.3 | 74.5 | **73** | 72.6 | 70.7 |
| Average TPR | 99.2 | | | 808 | | | 74.43 | | | 72.1 | | |
| FPR | 85.3 | 83.5 | 81.4 | 33.6 | 32.4 | 28.8 | 25.7 | 24.7 | 25.7 | **13.9** | 13.8 | 16.7 |
| Average FPR | 83.4 | | | 31.6 | | | 25.37 | | | 14.8 | | |

***Table 4.7: Results on Random Forest – With Walls – Original Dataset***

| P:N | **80:20** | | | **70:30** | | | **60:40** | | | **50:50** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Train:Test | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 |
| TestAccuracy | 87.2 | 86.6 | 86.3 | 83.2 | 82 | 81.3 | 84.46 | 83.8 | 83.4 | **85.6** | 85.2 | 84.35 |
| TPR | 99.2 | 99 | 99 | 83.4 | 82.6 | 82.2 | 83.5 | 83 | 82.6 | **81.7** | 81.2 | 79.9 |
| Average TPR | 99 | | | 82.7 | | | 83.03 | | | 80.93 | | |
| FPR | 64 | 65 | 66.6 | 17.3 | 19.4 | 20.6 | 14.1 | 14.9 | 15.4 | **10.4** | 10.9 | 11.1 |
| Average FPR | 65.2 | | | 19.1 | | | 14.8 | | | 10.8 | | |

***Table 4.8: Results on Random Forest – With Walls – Oversampled Dataset***

## 4.3. K-Nearest Neighbor

| P:N | **80:20** | | | **70:30** | | | **60:40** | | | **50:50** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Train:Test | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 |
| TestAccuracy | 80 | 76.4 | 77.6 | 75.5 | 73.6 | 72.55 | 72.4 | 72.3 | 71.3 | **76.1** | 76 | 73.6 |
| TPR | 98.9 | 91 | 91.8 | 77 | 74.1 | 73.1 | 72.9 | 71.4 | 69.8 | **71.6** | 68.8 | 65.8 |
| Average TPR | 93.9 | | | 74.7 | | | 71.37 | | | 68.73 | | |
| FPR | 86.7 | 77.1 | 77 | 27.3 | 28.1 | 28.8 | 28.4 | 26.2 | 26.3 | **19.2** | 16.5 | 18.5 |
| Average FPR | 80.2 | | | 28.1 | | | 26.97 | | | 18.07 | | |

***Table 4.9: Results on KNN – Without Walls – Original Dataset***

| P:N | **80:20** | | | **70:30** | | | **60:40** | | | **50:50** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Train:Test | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 |
| TestAccuracy | 86.7 | 86 | 85.4 | 81.8 | 81 | 80.43 | 82.9 | 82.18 | 81.7 | **83.84** | 83.61 | 82.8 |
| TPR | 98.3 | 98 | 97.5 | 81.2 | 80.6 | 80.7 | 81.5 | 80.6 | 80.2 | **80.1** | 79.8 | 78.8 |
| Average TPR | 97.9 | | | 80.83 | | | 80.77 | | | 79.57 | | |
| FPR | 62.5 | 63.9 | 64.8 | 18.2 | 17.9 | 20.1 | 14.7 | 15.4 | 15.7 | **12.5** | 12.6 | 13.1 |
| Average FPR | 63.7 | | | 18.7 | | | 15.27 | | | 12.73 | | |

***Table 4.10: Results on KNN – Without Walls – Oversampled Dataset***

| P:N | **80:20** | | | **70:30** | | | **60:40** | | | **50:50** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Train:Test | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 |
| TestAccuracy | 76.7 | 76.22 | 77.9 | 74.4 | 73.5 | 72.1 | 72.8 | 73.8 | 72.4 | **76.4** | 76.4 | 73.6 |
| TPR | 93.6 | 92 | 93.2 | 76.5 | 73.6 | 72.6 | 73.6 | 72.8 | 70.9 | **72.3** | 70.2 | 66 |
| Average TPR | 92.9 | | | 74.2 | | | 72.43 | | | 69.5 | | |
| FPR | 82.7 | 81.7 | 81.4 | 29.7 | 26.5 | 28.8 | 28.4 | 24.7 | 25.1 | **19.2** | 17 | 18.5 |
| Average FPR | 81.9 | | | 28.3 | | | 26.07 | | | 18.23 | | |

***Table 4.11: Results on KNN – With Walls – Original Dataset***

| P:N | **80:20** | | | **70:30** | | | **60:40** | | | **50:50** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Train:Test | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 |
| TestAccuracy | 86.4 | 85.7 | 85.3 | 80.55 | 79.7 | 79.7 | 81.5 | 80.9 | 80.9 | **83** | 82.59 | 82 |
| TPR | 98.1 | 98 | 97.6 | 80.9 | 79.9 | 80 | 80.4 | 79.7 | 79.1 | **79.2** | 78.7 | 78 |
| Average TPR | 97.9 | | | 80.3 | | | 7973 | | | 78.63 | | |
| FPR | 63.1 | 64.9 | 65.5 | 20.3 | 20 | 21 | 16.6 | 17.1 | 16.2 | **13** | 13.5 | 14.1 |
| Average FPR | 64.2 | | | 20.4 | | | 16.63 | | | 13.53 | | |

***Table 4.12: Results on KNN – With Walls – Oversampled Dataset***

## 4.4. Naïve Bayes

| P:N | **80:20** | | | **70:30** | | | **60:40** | | | **50:50** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Train:Test | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 |
| TestAccuracy | 78.8 | 76 | 75.4 | 73.7 | **74.7** | 75.4 | 68.87 | 70.9 | 70.86 | 66.5 | 68.6 | 66.42 |
| TPR | 87.9 | 85 | 84 | 83.5 | **84.1** | 84.5 | 83.8 | 83.4 | 82.6 | 80 | 80 | 79.5 |
| Average TPR | 85.6 | | | 84 | | | 83.27 | | | 79.83 | | |
| FPR | 53.3 | 56.9 | 57.9 | 46.1 | **45.5** | 46.3 | 54.5 | 48.7 | 47.7 | 47.4 | 44.1 | 46.9 |
| Average FPR | 56 | | | 45.9 | | | 50.3 | | | 46.13 | | |

***Table 4.13: Results on Naïve Bayes – Without Walls – Original Dataset***

| P:N | **80:20** | | | **70:30** | | | **60:40** | | | **50:50** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Train:Test | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 |
| TestAccuracy | 76.6 | 76 | 75.4 | 74.61 | 71.21 | 71.42 | 69.33 | 68.96 | **69.22** | 66.17 | 66.29 | 66.9 |
| TPR | 83.4 | 82.9 | 82.8 | 81.5 | 81.1 | 81.7 | 82.8 | 82.1 | **82.2** | 81.9 | 81.9 | 81.2 |
| Average TPR | 83 | | | 81.4 | | | 882.36 | | | 81.67 | | |
| FPR | 52.1 | 52.6 | 54.9 | 52.2 | 52.7 | 52.4 | 51.9 | 51.6 | **50.8** | 49.4 | 49.2 | 47.5 |
| Average FPR | 53.2 | | | 52.4 | | | 51.43 | | | 48.7 | | |

***Table 4.14: Results on Naïve Bayes – Without Walls – Oversampled Dataset***

| P:N | **80:20** | | | **70:30** | | | **60:40** | | | **50:50** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Train:Test | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 |
| TestAccuracy | **78.5** | 76 | 75.4 | 71.9 | 72.85 | 71.13 | 68.65 | 69.9 | 69.4 | 64.7 | 65.8 | 64.2 |
| TPR | **87.9** | 85 | 84.4 | 83.5 | 83.6 | 80.3 | 84.5 | 85.1 | 83.5 | 78.4 | 80 | 78.4 |
| Average TPR | 85.7 | | | 82.4 | | | 84.37 | | | 78.93 | | |
| FPR | **54.7** | 56.9 | 59.3 | 51.6 | 50.3 | 50.7 | 56.3 | 54 | 52.9 | 49.6 | 49 | 50.2 |
| Average FPR | 56.9 | | | 50.8 | | | 54.4 | | | 49.6 | | |

***Table 4.15: Results on Naïve Bayes – With Walls – Original Dataset***

| P:N | **80:20** | | | **70:30** | | | **60:40** | | | **50:50** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Train:Test | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 | 80:20 | 70:30 | 60:40 |
| TestAccuracy | **75.67** | 75.4 | 75.1 | 72 | 71.14 | 71.3 | 68.1 | 69.97 | 68.24 | 64.23 | 65.1 | 65.7 |
| TPR | **80.2** | 80.3 | 80.6 | 84 | 83.3 | 83.8 | 84 | 83.2 | 83.5 | 81.8 | 80 | 79.6 |
| Average TPR | 80.3 | | | 83.7 | | | 83.57 | | | 80.47 | | |
| FPR | **43.7** | 44.8 | 47.6 | 56.9 | 58.1 | 57.6 | 56.6 | 55.9 | 55.5 | 53.1 | 49.6 | 48.3 |
| Average FPR | 45.36 | | | 57.5 | | | 56 | | | 50.33 | | |

***Table 4.16: Results on Naïve Bayes – With Walls – Oversampled Dataset***

## 4.5. ConvNet

| P:N | **80:20** | **70:30** | **60:40** | **50:50** |
|---|---|---|---|---|
| TrainAccuracy | 97.71 | 99.95 | 99.88 | **99.63** |
| TestAccuracy | 67.77 | 72.22 | 80.74 | **82.97** |
| TPR | 93.33 | 90.37 | 88.15 | **86.67** |
| FPR | 57.78 | 45.93 | 26.67 | **20.74** |

***Table 4.17: Results on ConvNet – Original Dataset***

| P:N | **80:20** | **70:30** | **60:40** | **50:50** |
|---|---|---|---|---|
| TrainAccuracy | 97.97 | 98.9 | **99.6** | 99.57 |
| TestAccuracy | 83.33 | 82.6 | **84.07** | 84.45 |
| TPR | 83.41 | 80 | **77.78** | 83.7 |
| FPR | 20.74 | 14.81 | **9.63** | 14.81 |

***Table 4.18: Results on ConvNet – Oversampled Dataset***

We analyzed true positives, false positives, true negatives, and false negatives. To get insight how well our model is classifying and if there is any room for improvements.

Here, we present some images of true positive, false positives, false negatives, and true negatives predicted by our proposed model on oversampled data with positive to negative ratio 50:50 and train test split is 70:30. This model achieved 94.43% TPR and 4.09% FPR.

### 4.5.1. True Positives

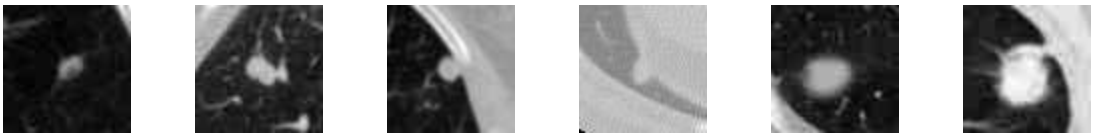

***Figure 4.1: (a) Simple nodule, (b ) Nodule with walls, (c) Nodule attached to walls, (d) Nodule surrounded by walls with both the sides, (e) Big nodule, (f) Big nodule attached to wall***

True positives are nodules which are correctly classified as nodules and are as shown in figure 4.1; our model is doing a good job in classifying nodules in various shapes. Our model is able to classify majority of nodules attached with walls, nodules surrounded by walls with both sides, bigger nodules, and bigger nodules attached with walls. That's the reason why we think our TPR is high. We analyzed here why our model has high TPR, but to further increase TPR, we need to analyze false negatives.

### 4.5.2. False Negatives

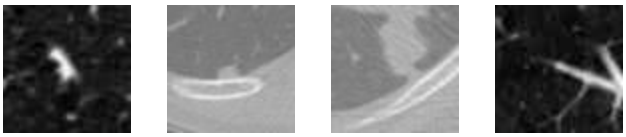

***Figure 4.2: (a) Simple nodule, (b ) Nodule with walls, (c) Big Nodule attached to walls, (d) Random Noise***

False negatives are nodule classified wrongly as non-nodules. To increase TPR, we need to analyze the false negatives and modify our architecture accordingly based on inferences. Our model is not able to classify all nodules correctly due to noise as shown in figure 4.2, if we can decrease the ratio (although very small currently) of simple nodule as false negatives, our TPR

will increase. Due to noise, model seldom failed to classify nodules attached with walls and bigger nodule attached to walls. As we analyzed the ways to increase TRP, it's time to analyze true negatives to answer the question of less FPR.

### 4.5.3. True Negatives

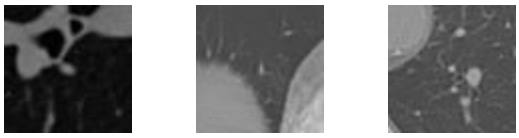

***Figure 4.3: (a) Random structures which are not nodules, (b ) Only walls, (c) Looks like nodule but they are small veins***

True negatives are non-nodules classified correctly as non-nodules. To have low FPR, we should have higher true negatives, as shown in figure 4.3 our model is able to classify well random structures which do not resemble with nodules, but it is tough to classify non-nodules which are similar to nodules.

### 4.5.4. False Positives

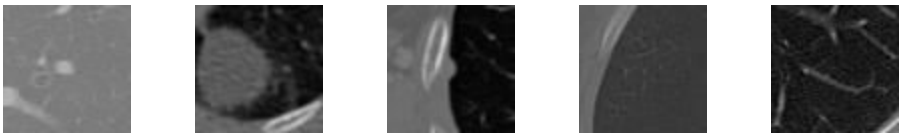

***Figure 4.4: (a) Similar to nodule, (b ) Similar to bigger nodules, (c) Similar to nodule attached to walls, (d )Only wall, (e) Random noise***

False positives are non-nodules wrongly classified as nodules. As shown in figure 4.4, our model is classifying well some of lookalikes of nodules, but even failed to classify simple walls and noise as non-nodules. We can decrease FPR further, if our model will be able to decrease the ratio of random noise and only walls as false positives.

## 4.6 Comparison of algorithms

| Algorithm | With Walls | | Without Walls | |
|---|---|---|---|---|
| | TPR | FPR | TPR | FPR |
| LogisticRegression | 70.7 | 24.1 | 73 | 20.7 |
| Random Forest | 73 | 13.9 | 74 | 12.8 |
| K-NN | 72.3 | 19.2 | 71.6 | 19.2 |
| Naive Bayes | 87.9 | 54.7 | 84.1 | 45.5 |
| ConvNet | 87.5 | 18.38 | - | - |

***Table 4.19: Comparison of algorithms (Original Dataset)***

| Algorithm | With Walls | | Without Walls | |
|---|---|---|---|---|
| | TPR | FPR | TPR | FPR |
| LogisticRegression | 71.2 | 26 | 71.4 | 22.4 |
| Random Forest | 81.7 | 10.4 | 80.7 | 10.3 |
| K-NN | 79.2 | 13 | 80.1 | 12.5 |
| Naive Bayes | 80.2 | 43.7 | 82.2 | 50.8 |
| ConvNet | 94.43 | 3.91 | - | - |

***Table 4.20: Comparison of algorithms (Oversampled Dataset)***

# Chapter 5

# Conclusion

We have successfully fulfilled our defined objectives. Automating a human task has always been hard, so is the task of radiologist. Detecting pulmonary nodules through CT scans is an arduous task for radiologists and is prone to human errors. Our primary objective which was to detect nodules automatically has performed well on our dataset. However, it suffers False Positive Rate (FPR). To reduce that, we worked to design a classifier which has significantly decreased the FPR. Although the long term goal is to completely replace radiologists, it is not really feasible because computers lack the intelligence we possess. Instead, these systems are expected to assist radiologists on a large scale. They can make use of this software as their assistant and ensure that their predictions are true.

With the completion of this project, we have explored work of other researchers and various state-of-art techniques which aim to solve the same problem. Moreover we implemented the CAD system using cutting-edge technologies to improve the performance of our system. The same approach with a few alterations can be helpful to predict other cancers and even other diseases. Overall, we are proud that we worked to improve the quality of human lives and contribute to scientific community concerned about health care.

# Chapter 8

# References


[1] World Health Organization – Cancer Statistics, 2012

[2] Nandakumar, A. "National cancer registry programme." Indian Council of Medical Research, Consolidated report of the population based cancer registries, New Delhi, India 96 (1990): 202-208.

[3] National Cancer Institute Cairo, Egypt Statistics, 2015

[4] Swensen, Stephen J., et al. "Screening for lung cancer with low-dose spiral computed tomography." American journal of respiratory and critical care medicine 165.4 (2002): 508-513.

[5] Gurney, Jud W. "Missed lung cancer at CT: imaging findings in nine patients." Radiology 199.1 (1996): 117-122.

[6] Armato III, Samuel G., et al. "Lung image database consortium: developing a resource for the medical imaging research community." Radiology 232.3 (2004): 739-748.

[7] Lung Nodule Analysis (LUNA) challenge data-set, https://luna16.grand-challenge.org, 2016

[8] Soda, Hiroshi, et al. "Limitation of annual screening chest radiography for the diagnosis of lung cancer. A retrospective study." Cancer 72.8 (1993): 2341-2346.

[9] Sone, S., et al. "Characteristics of small lung cancers invisible on conventional chest radiography and detected by population based screening using spiral CT." The British journal of radiology 73.866 (2000): 137-145. APA

[10] Diederich, Stefan, et al. "Screening for early lung cancer with low-dose spiral CT: prevalence in 817 asymptomatic smokers." Radiology 222.3 (2002): 773-781.

[11] S. Napel and et al. Ct angiography with spiral CT and maximum intensity projection. Radiology, 185:607–610.

[12] K. Murphy and et al. A large scale evaluation of automatic pulmonary nodule detection in chest CT using local image features and k-nearest-neighbor classification. Medical Image Analysis, 13:757–770, 2009.

[13] Sluimer, I.C., Prokop, M., van Ginneken, B., 2005. Towards automated segmentation of the pathological lung in CT. IEEE Transactions on Medical Imaging 24 (8), 1025–1038

[14] Aoyama, Masahito, et al. "Automated computerized scheme for distinction between benign and malignant solitary pulmonary nodules on chest images." Medical Physics 29.5 (2002): 701-708.

[15] Antonelli, M., Frosini, G., Lazzerini, B., & Marcelloni, F. (2004). Lung Nodule Detection in CT Scans. In International Conference on Computational Intelligence (Vol. 1, pp. 365-368).

[16] Roth, Holger R., et al. "Improving computer-aided detection using convolutional neural networks and random view aggregation." IEEE transactions on medical imaging 35.5 (2016): 1170-1181.

[17] Xiojie and et al. Computer-aided detection using 3D convolutional neural networks. IEEE Trans. on Medical Imaging, 152-155, 2015.

[18] Moltz, J.H., Kuhnigk, J.M., Bornemann, L. and Peitgen, H., 2008. Segmentation of juxtapleural lung nodules in ct scan based on ellipsoid approximation. In Proceedings of First International Workshop on Pulmonary Image Processing. New York (pp. 25-32).

[19] T. Huang, G. Yang, and G. Tang, "A fast two-dimensional median filtering algorithm", IEEE Trans. Acoust., Speech, Signal Processing, vol. 27, no. 1, pp. 13–18, 1979.

[20] Wiener, Norbert. Extrapolation, interpolation, and smoothing of stationary time series. Vol. 7. Cambridge, MA: MIT press, 1949.

[21] Hum, Yan Chai, Khin Wee Lai, and Maheza Irna Mohamad Salim. "Multiobjectives bihistogram equalization for image contrast enhancement." Complexity 20.2 (2014): 22-36.

[22] Otsu, Nobuyuki. "A threshold selection method from gray-level histograms." IEEE transactions on systems, man, and cybernetics 9.1 (1979): 62-66.

[23] Chan T, Vese L. An active contour model without edges. Scale-Space Theories in Computer Vision. 1999:141-51.

[24] Hartigan, John A., and Manchek A. Wong. "Algorithm AS 136: A k-means clustering algorithm." Journal of the Royal Statistical Society. Series C (Applied Statistics) 28.1 (1979): 100-108

[25] Masreliez, C. "Approximate non-Gaussian filtering with linear state and observation relations." IEEE Transactions on Automatic Control 20.1 (1975): 107-110

[26] Zhou, Tao et al. "Pulmonary Nodule Detection Model Based on SVM and CT Image Feature-Level Fusion with Rough Sets." BioMed Research International2016 (2016): 8052436. PMC. Web. 4 Dec.2017: 121-125.

[27] Burger, W., Burge, M. J., Burge, M. J., & Burge, M. J. (2009). Principles of digital image processing (p. 221). London: Springer, 342-351.

[28]. Han, Jun, and Claudio Moraga. "The influence of the sigmoid function parameters on the speed of backpropagation learning." From Natural to Artificial Neural Computation (1995): 195-201.

[29]. Cox, David R. "The regression analysis of binary sequences." Journal of the Royal Statistical Society. Series B (Methodological) (1958): 215-242.

[30]. Breiman, Leo. "Random forests." Machine learning 45.1 (2001): 5-32

[31]. Cortes, Corinna, and Vladimir Vapnik. "Support-vector networks." Machine learning 20.3 (1995): 273-297.

[32]. Altman, Naomi S. "An introduction to kernel and nearest-neighbor nonparametric regression." The American Statistician 46.3 (1992): 175-185.

[33]. Kendall, M. G., Stuart, A., & Ord, J. K. (1948). The advanced theory of statistics (Vol. 1, pp. 42-46). London: C. Griffins.

[34]. Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: towards real-time object detection with region proposal networks. In Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1 (NIPS'15), C. Cortes, D. D. Lee, M. Sugiyama, and R. Garnett (Eds.), Vol. 1. MIT Press, Cambridge, MA, USA, 91-99.

[35]. "Speed/accuracy trade-offs for modern convolutional object detectors." Huang J, Rathod V, Sun C, Zhu M, Korattikara A, Fathi A, Fischer I, Wojna Z, Song Y, Guadarrama S, Murphy K, CVPR 2017.

[36]. Christian Eggert, Dan Zecha, Stephan Brehm, and Rainer Lienhart. 2017. Improving Small Object Proposals for Company Logo Detection. In Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval (ICMR '17). ACM, New York, NY, USA, 167-174. DOI: https://doi.org/10.1145/3078971.3078990

[37] Hartigan, J.A. and Wong, M.A., 1979. Algorithm AS 136: A k-means clustering algorithm. Journal of the Royal Statistical Society. Series C (Applied Statistics), 28(1), pp.100-108.

[38] Soille, P., 2013. Morphological image analysis: principles and applications. Springer Science & Business Media.

[39] Burger, W., Burge, M.J., Burge, M.J. and Burge, M.J., 2009. Principles of digital image processing (p. 221). London: Springer.