Performance Analysis of Deep Learning based approaches for detection and classification of Lung Cancer in humans

Project report submitted in partial fulfillment of the requirement for the degree of Bachelor of Technology

in

Computer Science and Engineering

by

Aman Srivastav (161328)

Under the supervision of

Dr. Pradeep Kumar Gupta (Associate Professor, Dept. of CSE & IT)

to

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal Pradesh

CERTIFICATE

Candidate's Declaration

I hereby declare that the work presented in this report entitled “Performance Analysis of Deep Learning based approaches for detection and classification of Lung Cancer in humans” in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering** submitted in the department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from August 2019 to May 2020 under the supervision of **Dr. Pradeep Kumar Gupta** (Associate Professor, Department of CSE & IT).

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Aman Srivastav (161328)

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Pgupta

Dr. Pradeep Kumar Gupta
Associate Professor
Department of Computer Science Engineering & Information Technology
Dated: **..../....**/2020

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of people whose ceaseless cooperation made it possible, whose constant guidance and encouragement crown all efforts with success**.** I am grateful to my project guide **Dr. Pradeep Kumar Gupta** for the guidance, inspiration and constructive suggestions that helped us in the preparation of the project. I am also grateful that I got to work with **Mr. Abhishek Bhardwaj (161312)** on this project, without his constant support and diligence I don't think I would have been able to complete this project.

I would also like to thank our colleagues who helped me in successful completion of the project by having constructive discussions with us on how to implement the project**.**

Finally a heartfelt thanks to my parents and my sister for constantly being the pillars of support. Thanks to **Aakash Suryavanshi** and **Sachin Gupta (161467)** for their support.

Date**: …./…./**2020 **Aman Srivastav**

(161328)

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

Abbrevations	Full Form
ANN	Artificial Neural Network
CNN	Convolutional Neural Networks
CT	Computed Tomography
DenseNet	Dense Convolutional Network
DL	Deep Learning
kNN	k-Nearest Neighbours
ML	Machine Learning
ReLU	Rectified Linear Unit
ResNet	Residual Network
SVM	Support Vector Machines
VGG	Visual Geometry Group

LIST OF FIGURES

LIST OF GRAPHS

LIST OF TABLES

ABSTRACT

LUNG CANCER

Cancer, a disease in which the body's cells develop out of balance. When it occurs in the lungs this condition is called lung cancer.

In the lungs, lung cancer may spread to lymph nodes or other organs in the body, such as the brain. Cancer from another organ in the body can also spread to the lungs. These are called metastases as cancer cells are spread from one organ to another.

Lung cancers are mainly of two main types called small cell and non**-**small cell**.** These types of lung cancer grow inversely and are treated differently**.** Non**-**small cell lung cancer is often found more in human than small cell lung cancer**.**

Diagnosis

A doctor on the lung cancer broadcast identifies a doubtful lesion, or a person can be feeling symptoms that could point to lung cancer, many different diagnostic tests are offered to confirm what the next steps should be**.**

Examples of these include:

Tissue sampling: Tissue sampling: If, on an mri scan, a doctor detects a suspicious lesion, they may recommend taking a sample of lung tissue to test for a possible cancer cell. There are various approaches to obtain a tissue sample, and the position of the lesion also relies on certain techniques.

One example is when a doctor performs a bronchoscopy, requiring the installation of a special small, lighted scope with a camera at the top. It lets the doctor see the lesion and take the samples afterward. Less accessible lung lesions can require more aggressive surgery to remove lung tissue, such as thoracoscopy or video-assisted chest surgery.

Lab testing: A doctor may also suggest blood tests or sputum tests to check for lung cancer. A doctor may use the gathered information to assess the type of lung cancer that could arise, and how severe the disease has become.

Imaging studies: Computed tomography (CT) and positron emission tomography (PET) scans may reveal areas of cancerous lung tissue. Bone scans can also be cancerous growth markers for gods. Doctors may also use these scans to track treatment development, or after a course of treatment, to ensure that cancer has not come back.

The importance of Early Diagnosis

Early diagnosis of lung cancer can be highly effective and can save lives. That is because lung cancer cells can migrate to and affect other parts of the body before a doctor identifies them in the lungs. If this spread or metastasis has occurred it makes it a very difficult task to treat the disease.

A doctor will sometimes suggest a person has screenings for lung cancer. Using a low-dose CT scanner, those are achieved. Not everyone is a contender for this screening, but in some individuals, it may help doctors identify the lung cancer earlier.

According to the American Lung Association, people who may be contenders for lung cancer screenings are those who**:**

- lie in the age group of 55 and 80 years**.**
- has a history of smoking of 30 pack**-** year, meaning that they have for 30 years smoked one pack per day or for 15 years have smoked two packs per day.
- is a person smoking currently or a smoker who has recently given up smoking but has

CHAPTER-1

INTRODUCTION

1.1 INTRODUCTION

Machine Learning

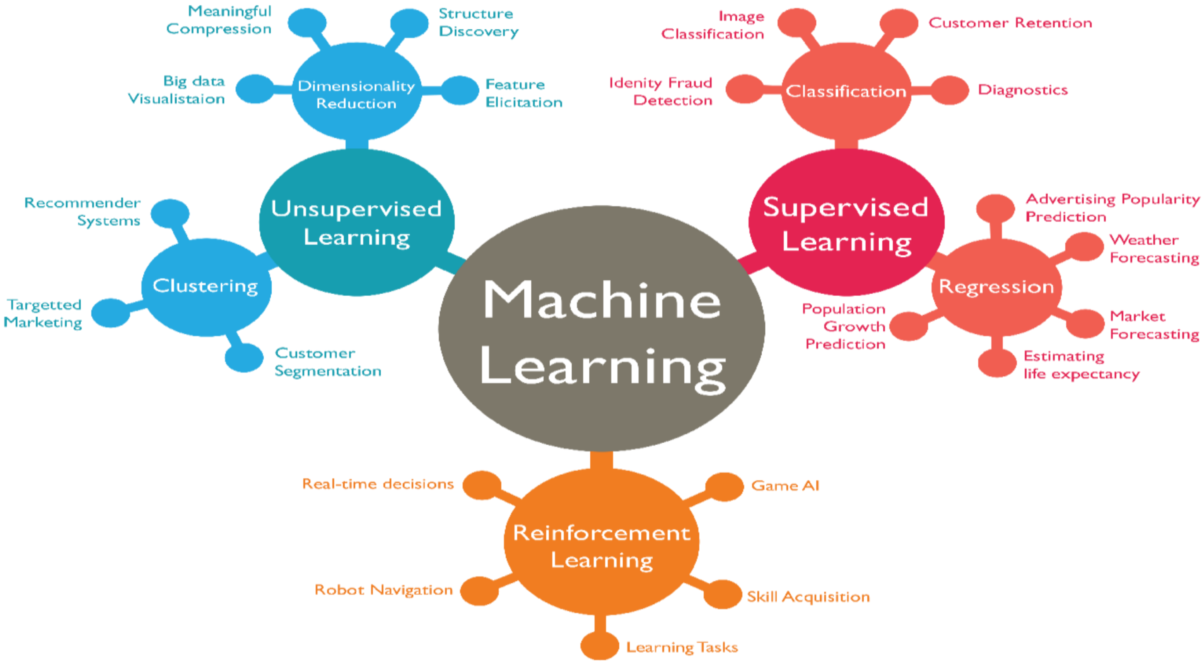


Figure 1 Machine Learning

Machine learning, subset of a bigger domain of study known as Artificial Intelligence that offers the machines the ability to study something with the help of suitable dataset by itself and develop from the experience of the dataset without being programmed directly. The primary focus of Machine learning is on those programs and can learn by navigating and using the data by themselves. The process of learning starts with observations, such as examples, experience directly incurred by humans or lessons, to search for and record patterns that are present in these data and to make better decisions in the coming future by leaning on the examples that we have provided to the model. The primary objective of this is to allow machines to automatically learn something without any kind of human assistance, and to let the machine adjust its actions according to examples provided in the past.

Methods of Machine Learning

Algorithms related to Machine learning are characterized into two key categories as supervised or unsupervised.

- Using named, those having labels, examples to predict on future happenings, **supervised algorithms** can be used where past learning is used for new data. We start with the study of a known dataset that is used to train only, the algorithm then learns and generates conditional function that help in making the output value predictions. After the training of the model from the dataset the program will provide expectations for any fresh input. The algorithm can then compare its output with the accurate, intended output that is provided in the training dataset, and find errors and compute accuracy to modify the model to give a high performance accordingly.

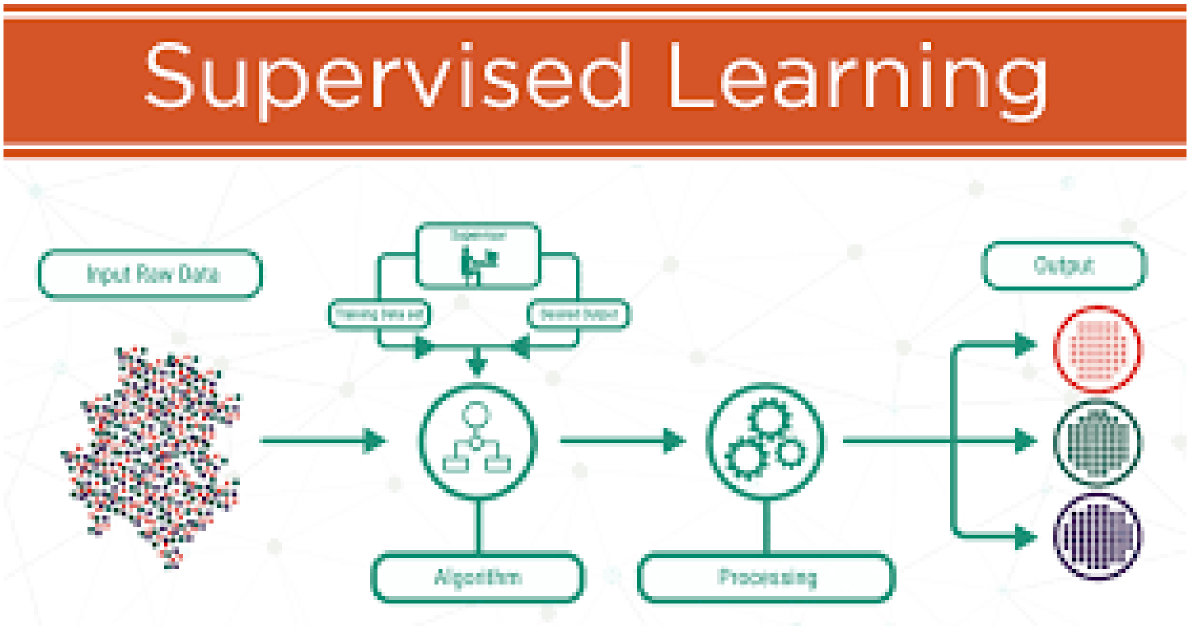


Figure 2 Supervised Learning

- Whereas, **unsupervised algorithms** are used in places where it is impossible to classify or label the information that is going to be used to train our model. Unsupervised learning explores how any systems can analyze unlabeled data to create a function to find a secret structure that is hiding somewhere in the data. The system won't be able to find out the correct output and in that manner might be termed as redundant, but the system travel arounds the data and draws matches from the datasets to explain structures that are hidden from unlabeled details that are present in the dataset.

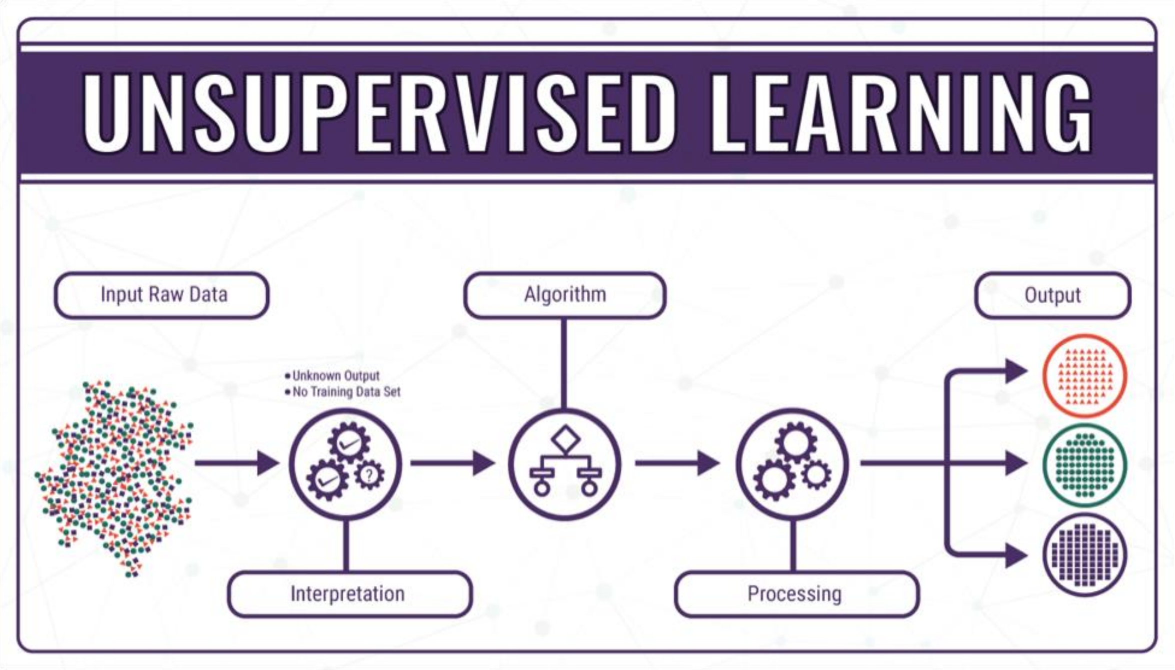


Figure 3 Unsupervised Learning

Machine learning algorithms helps us to analyze massive quantities of data. They usually providing faster, more trustworthy outcomes to identify lucrative opportunities or reduce hazardous risks, it can may take some extra time and money to train properly. Joining machine learning with AI and cognitive technologies can make the dispensation of large volumes of data present even quicker and more operational.

Deep Learning

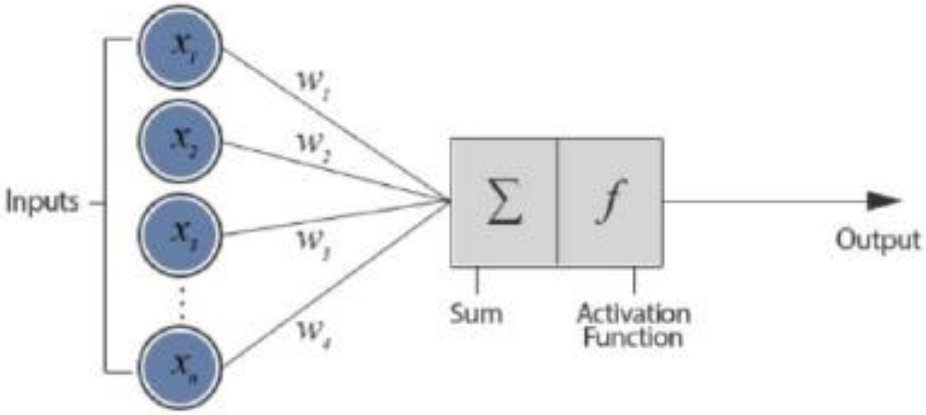


Figure 4 Deep Learning

Deep learning, a subset of machine learning, tries to replicates the functioning of the human intelligence in data handling and finding patterns which can help in the process of making insights from the data. It’s a vast study field and is capable of learning from any type of data-unstructured or unlabeled data without being unsupervised. Sometimes it is commonly referred as Deep Neural Network.

How Deep Learning Works

With the digital era, deep learning has grown and is still growing immensely, resulting in an outburst of data in different forms and from every part of the universe and sometimes universe. Referred as big data, this category of data is taken from sources such as the social media, online cinemas that show movies and internet search engines. This large amount of data is readily accessible and can be accessed via cloud computing applications like fetch.

The data, however, which is normally unstructured data in nature, is very vast and may take up decades for humans to comprehend and extract relevant ideas from it. Many companies realize the boundless potential that can be derived from this wealth of information that is lying scrambled on the internet, and increasingly adapting for support systems to AI systems.

Deep Learning vs. Machine Learning

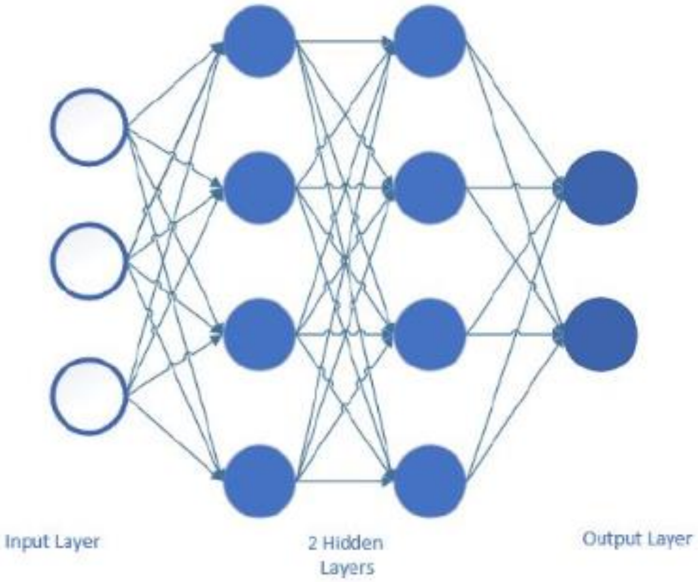


Figure 5 Neural Network

Machine learning is one of the utmost popular Artificial Intelligence techniques that is being used to handle data, a self-adaptive and self-learning algorithm which continues to develop on its experience or add more data.

To this end, if say a company working on digital payments wanted to distinguish the existence or possibility of fraudulent payments in its system, it might use machine learning technique. The computational algorithm that has been built into a computer system will be processing all the transactions that occur on the network, identify data set patterns that are present and point out any deviation that the pattern detects in the dataset.

Deep learning, a subset of machine learning, is designed via a hierarchical level of neural networks whose task is to perform machine learning processes. The artificial neural networks are built on the principles of human intelligence, with the nodes of the neuron linked like that in a web network. Whereas in the old-style programs the data analysis is constructed linearly, the hierarchical way that is used in deep learning systems work allows the machines to do the processing of the data with an approach that is nonlinear.

An old school approach to detecting fraudulent transaction or money filtering could rely on the amount of transactions that follows, while a deep learning technique which is not linear in nature would have included time, geographic locations, Internet Protocol address, and any other feature that would likely point to fraudulent activity being committed**.** The first layer of our neural network is used to process a data input which is raw like the quality of the transaction and transfer it as output to the next layer. The next layer processes the information from the layer coming before it by adding some additional information such as the Internet Protocol address of the user and then passes the result on to the next one.

The next layer then takes output information from the layer that came before it and uses the raw input data, such as geographic position, making the design of the system even better. These steps are then continued through all the stages of neuron network**.**

Convolutional Neural Network

CNNs are Deep Learning algorithms that can take an image as input and then assign location known as learnable weights and biases to numerous objects in the image and distinguish one from the other. Compared to other algorithms used for classification problems, the processing done before is not much needed in a simple CNN. Although the filters are hand-engineered in primitive methods, with appropriate training scale they has the ability to absorb these features.

The CNN architecture is alike to that of Neurons' connectivity pattern working in the Human Mind, and was motivated by the work of Visual Cortex. Individual neurons will only respond to provocations in a restricted region of the field of vision known as the Receptive Field. These fields then join together to cover the entire span of vision.

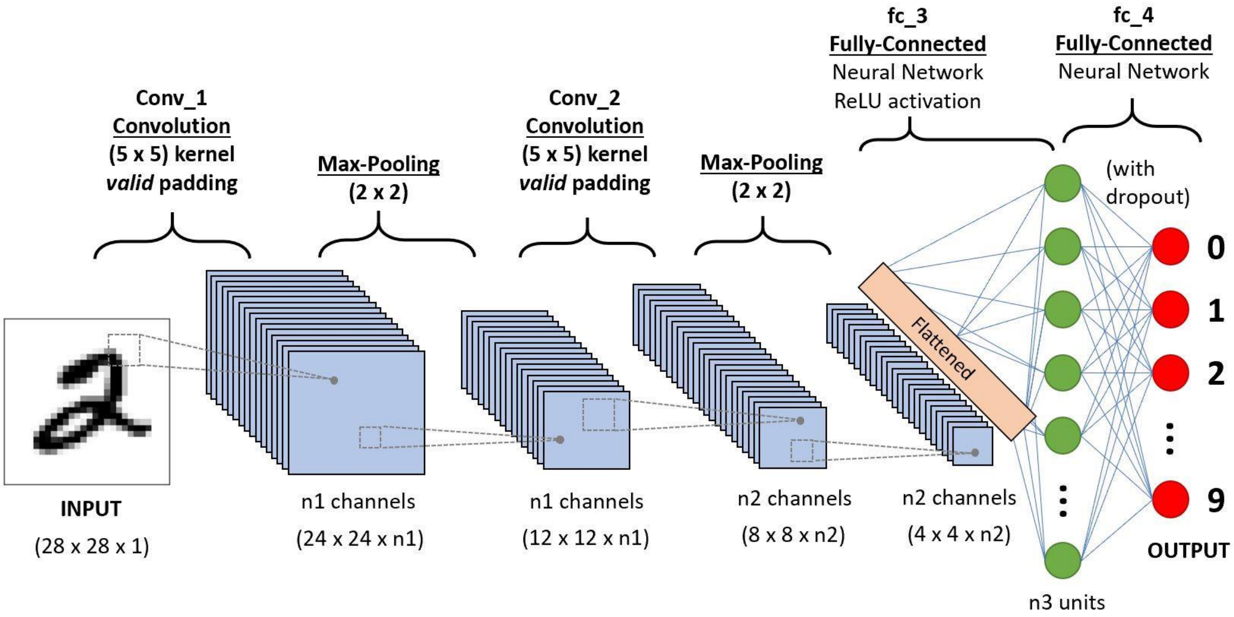


Figure 6 CNN Architecture

DenseNet

Recent research on CNN has revealed that training can be substantively deeper, more operational and more resourceful if they include small links between the layers in those that are close to the input and the output. In their paper, they have embraced this observation and introduced the Dense Convolutional Network commonly called DenseNet, which in a feed-forward fashion links each layer to every other layer. Although ordinary CNN layers with L have only L connections — one present between each layer and a succeeding layer — their network has ((L+1)*L)/2 direct way connections**.** The feature maps generated by all the previous layers are used as input for each layer, and they have with themselves some feature maps are used as input to all succeeding layers. DenseNets have many persuasive benefits: they mitigate the issue of the gradients that vanish, reinforce the circulation of features, enable reprocessing of features and greatly diminish the total number of parameters used.

The counter-intuitive outcome of this densely connected pattern is that the number of parameters needed is fewer than that of conventional CNN, because terminated feature maps need not be relearned. Traditional architectures that employ forward feeding can be seen as state algorithm which are transferred from one layer to another. Every layer reads the state from the layer before it and writes on the next layer. The state is changed now but it also transmits the information that needs to be stored. ResNets stores this statistics by transformations of the addition of outputs of layers. Now it has been shown that the contribution of many layers of ResNets is very minute to the training and can be dropped in a random way. This has made the ResNets state the same as an unrolled RNN, but the number of parameters in ResNets is considerably greater as every layer has some weights associated with them. The architecture that is suggested specifically varies between statistics applied to our network and statistics stored in the network. Layers in our model are slim, adding only a slight set of features to the network's "collective knowledge" and keeping the remaining maps ineffective — and the verdict of the final classifier is based on all of the network's feature maps.

In addition to improving the parameter performance, another great benefit of DenseNets is its transformed flow of statistics and gradients across the whole network, which allows us to train them easily. Increasing layers will have a direct admittance from the loss function to the gradients and the real input signals, resulting in implicit deep supervision. That helps train a deeper architecture of the network. We also find that densely connections have a standardizing effect which decreases overfitting on tasks with small training sets.

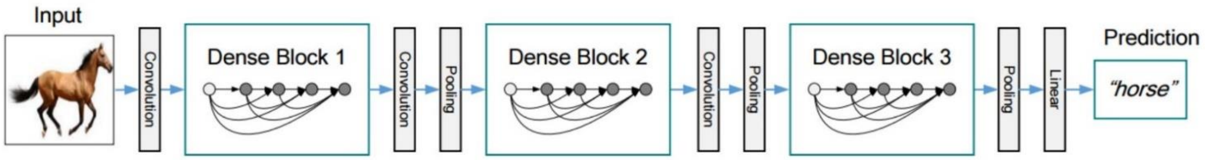


Figure 2. A deep DenseNet with three dense blocks. The layers between two adjacent blocks are referred to as transition layers and change feature map sizes via convolution and pooling.

Figure 7 DenseNet

The concatenation of feature maps learned from diverse layers surges variability in the input of succeeding layers and also improves the performance. That marks a significant difference between DenseNets and ResNets. DenseNets are simpler and more powerful compared to the Inception networks, which often concatenate feature maps from diverse layers**.**

ResNet

According to the theorem of universal approximation, we know that a forward feeding network with only a single layer is adequate to embody any mathematical function, given enough capacity. But alas, the layer can be enormous and the network would be inclined to overfit the data. It is also a growing thinking in the research community that the architecture of our network needs to go more deep**.**

The cutting-edge CNN architectures are going more and deeper since AlexNet. Although AlexNet had only five convolutional layers, there were 19 and 22 layers respectively for the VGG and GoogleNet, also known as InceptionV1.

But through the depth of the network doesn't work by simply piling layers together. Due to the infamous vanishing gradient problem, deep neural networks are tough to train — because the gradient has to be backpropagated to previous layers, repeating of multiplication will make the gradient indefinitely very small. And as the network deepens the output starts soaking or even begins to worsen fast**.**

The key idea behind using ResNet is to introduce a so-called "identity shortcut connection" that will skip more layers, as is shown in the figures below.

The concatenation of feature maps learned from diverse layers surges variability in the input of succeeding layers and also improves the performance. That marks a significant difference between DenseNets and ResNets. DenseNets are simpler and more powerful compared to the Inception networks, which often concatenate feature maps from diverse layers.

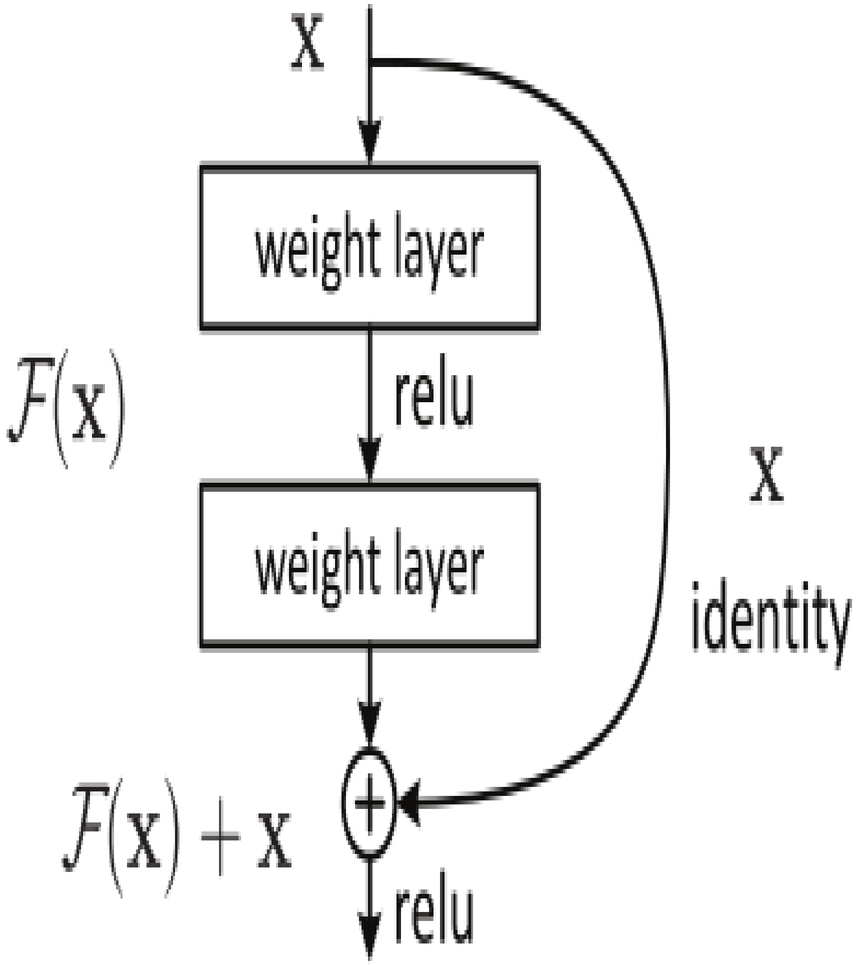


Figure 8 ResNet Skip Connection

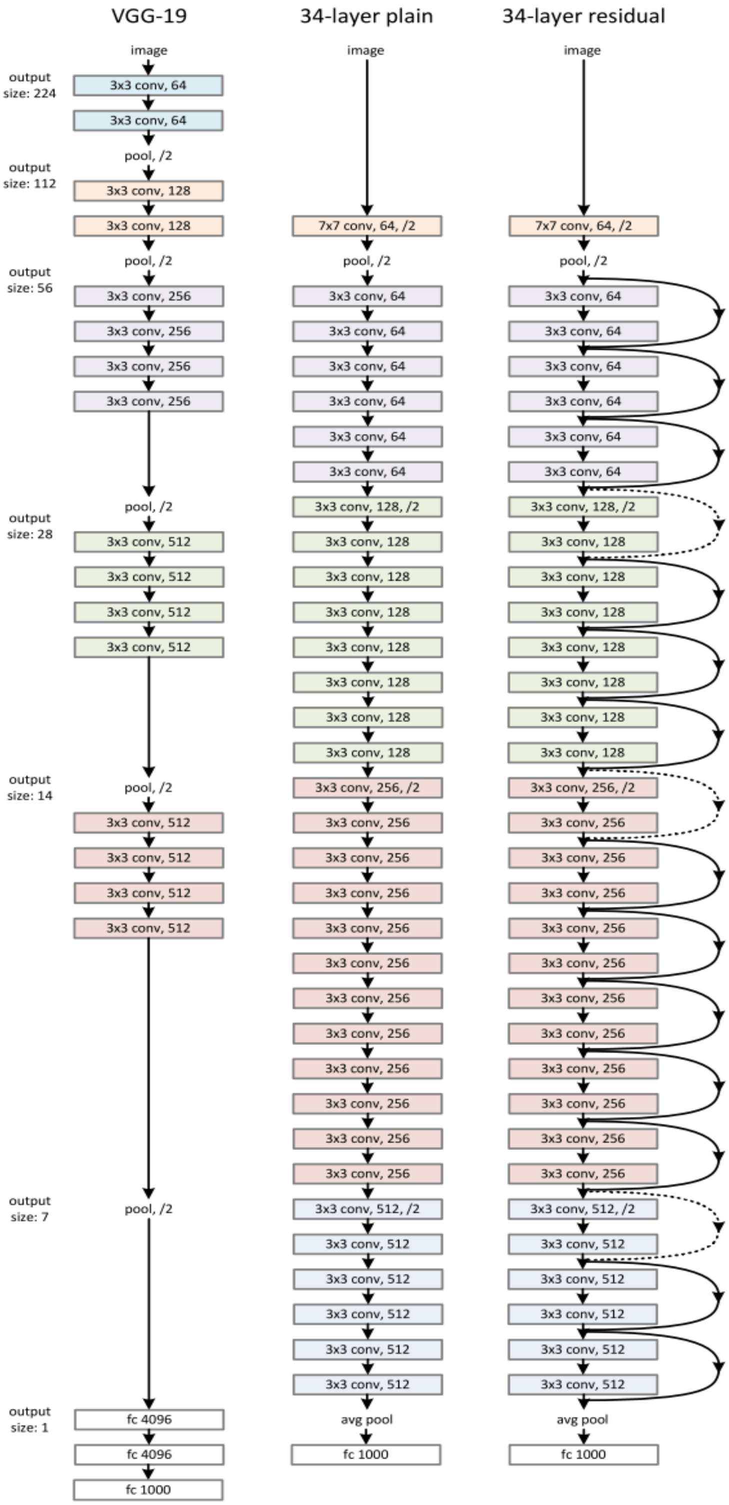


Figure 9 VGG & ResNet Full Model

The researchers contend that assembling layers will not degrade the performance of the network because we could simply stack and mapping identity (layer that does nothing) on the current network and the ensuing architecture produced will do the same thing. This shows that if the train error higher than the narrower counterparts should not occur with the deeper model. They hypothesize makes it easier for the stacked layers to match a remaining mapping and not to let them match the anticipated underlying mapping directly. And the remaining block above explicitly enables it to do this exactly.

VGG

While previous AlexNet derivatives concentrated in the first convolutional layer on a smaller window sizes and strides, VGG discusses another very critical feature of CNNs: width. Let's go and look at the architecture of the VGG**:**

- **Input:** The VGG takes an RGB image of 224 X 224 pixels. The researchers have cropped out center patch of 224 X 224 from all of the images for the ImageNet competition to keep the scale of the image input constant.

- **Conv Layers:** In VGG, the convolutional layers use a small accessible field (3 X 3, the smallest size yet capturing left or right and up or down). There are different 1 X 1 filters which function as linear input transformation, this is followed by the ReLU activation unit. The stage of convolution is set to a 1 pixel the effect of this is that after the convolution the three-dimensional resolution is conserved.

- **Fully-Connected (Dense) Neurons:** It has 3 FC layers: the first two of them have a total of 4096 channels each and the last one has only a thousand channels, 1 channel for each.

- **Hidden Layers:** All the hidden layers of VGG are using a ReLU (a major invention from AlexNet that is cutting training time). Generally speaking, VGG donot uses Local Response Normalization (LRN), since it surges the memory intake and time used to train without any increase in the precision.

The Difference

VGG is based on the AlexNet, has many alterations that separates it from other challenging models**:**

- Rather than using AlexNet's big receptive fields (11 X 11 with 4 step) it uses trivial receptive fields (3 X 3 with 1 step). There are now 3 units of ReLU instead of only one so the verdict function becomes extra inclusive. It also has lesser parameters, only 27 times the total N_C instead of 49 times the total N_C that AlexNet has).
- It provides 1 X 1 convolution layers in order to make the decision made more non-linear without altering the field size of receptives.
- Trivial-sized convection filters allow it to have a big number of weight layers; more layers is central to achieving better efficiency.

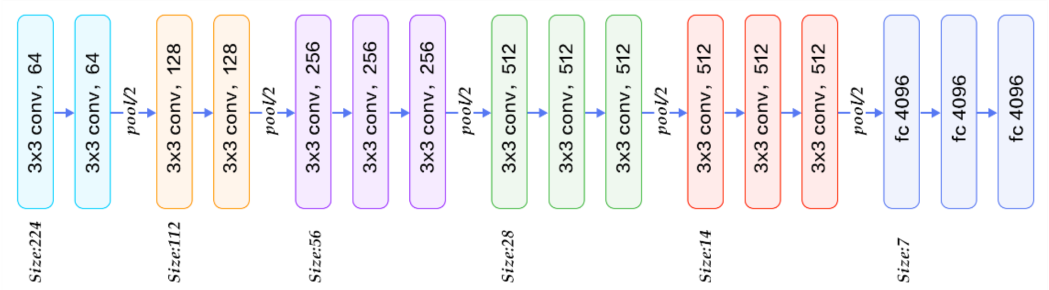


Figure 10 VGG_16

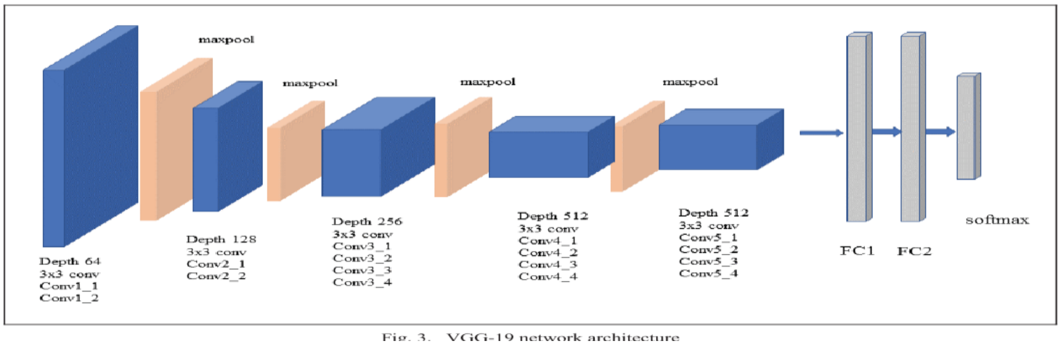


Figure 11 VGG_19

1.2 PROBLEM STATEMENT

Lung Cancer Classification and Detection

- Finding an efficient and effective way to carry out the Lung Cancer Detection (Classification of Nodule between Benign and Malignant Tumor) in 3D CT scan volumes.

- Comparing different ways in which Lung Cancer Detection (Classification of Nodule between Benign and Malignant Tumor) can be performed using different Deep Learning approaches in 3D CT scan volumes and suggesting the best one out of these.

- Comparing the results with that of the Machine Learning presented before and analyzing which one is giving better results.

1.3 OBJECTIVE

The project after it is completed should accomplish the following Objectives**:**

- **Education Oneself:** On Deep Learning A large part of this project contains a lot of self**-**education, initially I knew very little about deep learning, as part of this project I should have a good grasp on deep learning concepts**.**

- **User Research and Evaluation:** The project should have a user**-**centred design aspect**.** This means that the system should be designed to help certain users**.** In this project the user would be medical professionals who work in diagnosing lung cancer**.**

- **System: Upload CT Scans:** The system should be capable of getting CT Scans from Users that will be utilized by the Deep Learning Model**.**

- **System: Detection of Lung Cancer:** The system should be such that it can detect the lung cancer within the CT volumes that users have uploaded**.**

- **System: Display Results:** The system should provide relevant information that our user will understand appropriately and gain some insight out from it**.**

1.4 METHOLOGY

- Converting the images into numpy arrays from dicom format.

- Preprocessing the image,
    - Min-Max/Mean-Std Normalization.
    - Clipping and Cropping.
    - Reshaping and Resizing.

- Creating an appropriate CNN Architecture to process the dataset.

- Carry out various experiments,
    - With/Without Regularization.
    - Increasing/Decreasing Number of Dense Blocks.
    - Increasing/Decreasing Learning Rate.
    - Introducing Global Average Pooling 2D vs. Flat Layer.
    - Increasing/Decreasing Fully Connected Layers and its neurons.

- Settling on a final CNN architecture.

- Carrying out Testing for all the 60 volumes.

- Compiling results for many accuracy and efficiency measures,
    - Binary Accuracy.
    - Recall.
    - Time taken.

- Reporting best algorithm so far.

1.5 ORGANIZATION

The work that I'm doing is going to benefit many groups and organization around the world:

- **General Patients:** A software application where the patient can upload his/her CT scan volume of lung and can get an interim status report of his/her health will be very beneficial for them. It will be helpful for people who are busy with their day-to-day work and won't be able to find the time to get checked.

- **Hospitals:** On the event that there is non-availability of doctors or that the load of nos. of patients coming to get their CT scan checked is very high, some of these tasks can be delegated to the software application so as to cut out the response time drastically and let the doctors concentrate on high risk cases.

- **Health NGOs:** A large population of the world is still living is meagre conditions and are unable to get even the basic of medical attention without the intervention of Health NGOs like Red Cross. It becomes hard at times for these Health NGOs to find out doctors who would like to work under such conditions, in such a case a software application that can do the work automatically will cut the cost needed to be paid by the NGOs and will also be fast.

CHAPTER-2

LITERATURE SURVEY

2.1 IMAGE PROCESSING TECHNIQUES

Paper 3, Al-tarawneh: The results that they has obtained are similar to those obtained with the help of enhancement of image. They applied the bi-narization technique and for the purpose of extracting features they are using the masking approach and extracted following features **-** perimeter, area, eccentricity and average intensity - from the image**.**

Paper 6, Dwivedi: In their paper, they proposed a method for pre**-**processing of images known as contrast limited adaptive histogram equalization **(**CLAHE**).** They used gray**-**level co-occurrence matrix **(**GLCM**)** to carry out feature extraction from the images, which also provides the data pertaining to the position of the pixels having similar gray level values**.** Is's property is that it contains a wide assortment of statistical features that we can later extract for analysis purpose from the matrix**.** Automatic feature selection algorithms has been used by the authors of this paper for determining the greatest features**.**

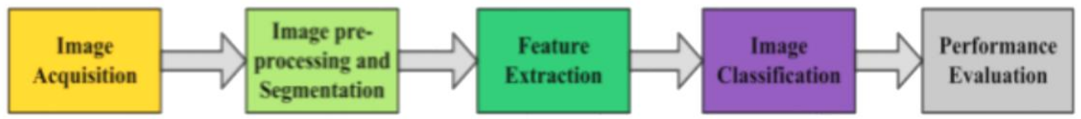
Paper 7, Sun: In this paper researchers shepherded a study to forecast the close**-**term risk of breast cancer by using the image dataset containing of only mammograms**.** The scheme that they proposed is calculated around the four modules of image processing like segmentation, classification, pre**-**processing of image, and feature extraction which helps in computing the asymmetry in image features**.**

Paper 8, Chaudhary: They are processing the image by using these techniques - pre-processing of image followed by extraction methods for features**.** The step, i.e., the first step comprises of two main subdivisions**:** segmentation and enhancement of image**.** For the enhancement of image, the authors have tested three algorithms – automatic enhancement, fast Fourier transform and Gabor filter. From their experimentations they established that the most apt technique is Gabor filter for the task of enhancement of image. And for the segmentation task, they identified thresholding technique as well as watershed's segmentation technique and established that the later out performs the former one.

Paper 9, Pratap: They have applied a similar watershed technique to perform the task of segmentation of image.

Paper 10, Bhusri: They used the laws pertaining to feature extraction for extracting the features from those regions that has some interest to us.

Paper 11, Kuruvilla: They have used a modest method for image segmentation, the Otsu's method, beside the morphological opening method that has episodic line for the arranging the element of fixed size. Both Intra-class and Inter-class variance can be minimized by this method. They found that the main downside of using this is that even after the binary image is obtained from the grayscale image through conversion, some openings are still there in the lung images that link to the arteries and the air that is existing inside the lung images and these may cause the models to forecast incorrect results as they look like a cancerous mass. They proposed that morphological operations need to be carried out to seal the areas left after applying the thresholding method to images.



. 1 Five basic steps for detection of lung cancer in humans

Figure 12 Image Pre-Processing Technique

2.2 PREVIOUS MACHINE LEARNING MODELS

Paper 6, Dwivedi: They decided to use multinomial multivariate Bayesian references in normal/non-cancerous and abnormal/cancerous images to classify images obtained after image processing stage discussed above.

Paper 12, Mitra and 13, Amato: Neural networks are the basically used in many approaches to machine learning. It can be seen as the classification of image into different classes, based upon input train dataset labels. There exist numerous machine learning algorithms that can be utilized for this tasks and can be categorized into following groups, known as unsupervised as well as supervised method. They have obtained the similar results**.**

Paper 14, Karabatak: To provide an effective computer-aided diagnostic method, they used many combinations of rules of association and the neural network. Use association rules reduces function vector dimensions, without significantly compromising the overall device accuracy.

Paper 15, Adi: They used GLCM and classified using naive Bayes, they built a method based on techniques of digital preprocessing of image for identifying cancer cells over the extraction of feature levels. They achieved 88.57 per cent accuracy in detecting lung cancer in their tests.

Paper 16, Joachims: He presented an improved training SVM on very large data sets and explained a highly successful way of implementing it via an SVM. He similarly used this technique during its optimization to reduce the problem size.

Paper 17, Tidke: They showcased a CAD system for lung cancer detection at early stage from CT volumes and for classification of whether the given image of tumor is a benign one or malignant. They showcased the model containing five-stages, and incorporated GLCM to extract textural features and classify images with SVM. Their result should an accuracy of 95 per cent using SVM after using only such a limited small size of dataset with only 25 JPEG images.

Paper 18, Touw: They recorded the key features of using the random forest algorithm and noted that this is one of the most commonly used in the field of life sciences for regression and task classification, such as patient-state disease prediction. It also enables additional relevant information to be extracted from the omics data.

Paper 19, Shi: They proposed a tumor profiling random forest clustering strategy based on data from the tissue microarray. We used this approach to diagnose and analyze a form of kidney cancer in adults that is the renal cell carcinoma.

Paper 20, Ramos-Gonzlez: We also developed a novel case-based inference method for diagnosing subtypes of lung cancer. To achieve high predictive accuracy they used feature selection tools of gradient boosted regression trees. They used support vector-machine methods, naive Bayes classifier, and kNN in their experiments.

2.3 PREVIOUS DEEP LEARNING MODELS

Paper 21, Sakamoto: They provided cascaded neural networks that are multi-staged with single-sided classificatory towards the incorrect positives in CT scan images of the lung nodule classification. They trained CNN with a well-adjusted dataset of a total of 888 images and in their experiments they achieved sensitivity of 92.4 and 94.5 per cent.

Paper 22, Demyanov: They offered the method that would automatically detect the dermoscopic patterns with profound CNN and other algorithms for image classification. They produced greater than 2000 samples in their experiments for all the classes of dermoscopic patterns with 83–88 per cent accurate rate of classification.

Paper 23, Lopez: They concentrated on the detection of skin cancer and introduced an approach based on deep learning for detecting it. They have conceived CNN architecture of VGGNet in their proposed solution.

Paper 24, Bewal: They analyzed various techniques of neural networks, such as CNN, ANN with a forward feeding network. For breast cancer recognition and established that NNs can critically assist healthcare professionals in giving additional view and critically lessen patient treatment times.

Paper 25, Havaei: They introduced a fully automated system of segmentation of brain tumor based on profound neural networks. Above all, they looked at the CNN architecture as can utilize both local as well as other general appropriate features.

Paper 26, Weng: They used deep learning methods for distinguishing between normal/non-cancerous and abnormal/cancerous lung automatically**.** They used the personal copy of Neural Computation and Applications 123 Author's CARS to diagnose lung cancer. Also utilizing the partial minimum square regression as well as SVM to extract as many as 35 features from the image to make predictions pertaining to lung cancer.

Paper 27, Sun: They developed and executed three highly organized algorithms that focused on a region that is of interest for multichannels. For this, they used CNN, DBN, and SDAE to include the features generated.

Paper 28, Mahbod: They used CNN for the detection and classification of objects over existing machine learning methods. They worked on skin cancer detection using an ensemble of pre-trained CNNs that learned from 2000 images of skin lesion. Such images are grouped into three groups-melanoma, seborrheic and benign keratosis.

Paper 30, Behrmann: They introduced deep CNN-based architecture and used deep learning for tumor classification. By this they integrated the classification step with the extraction of features into one model. Their study goal began research into advanced end-to - end methods of learning. The revised model showed improved results. Difficulty occurs when we use these techniques for segmentation to an extent so that the classifier's job may be easier downwards. Many of the techniques that are mentioned that identify the presence or absence of cancer but it is still a perplexing task to differentiate cancer into two benign and malignant forms and most of studies mentioned above failed in it. This will overcome the complexities of the methods being explored. It can sense the presence, lack of cancer mass and can even categorize mass into benign and malignant.

2.4 PREVIOUS METHODOLOGY

To get better results we have divided our task into following different stages known as the analysis of image and the classification of image.

Analyzing of Image: It is the process of improving the quality of an image so as to make it more readable for the humans. This process also removes any noise that might be present in the image and that hinder the performance of our model. It contains the following steps:

Pre-processing of Images: This is the process of conversion of image into greyscale and them removing the noise of the image through denoising. Three image denoising techniques are used – Gaussian, Median and Bilateral blur.

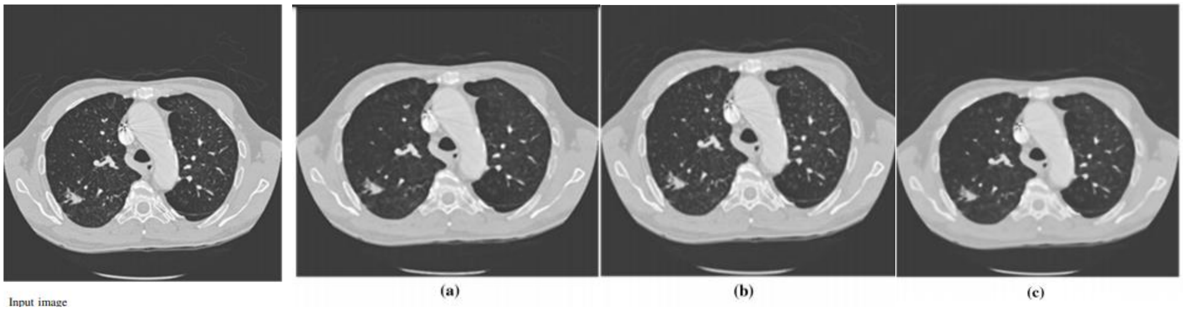


Figure 13 CT Image before and after applying blurs

In-depth analysis of results show that best results are obtained when we use Gaussian blur. After denoising we have to apply various methods of thresholding that will convert the greyscale image which contains values between ‘0’ to ‘255’ into a binary image which only consist of ‘0’ and ‘1’. While Otsu’s used adaptive mean thresholding, we have used global thresholding. Our method is found to work better than the former.

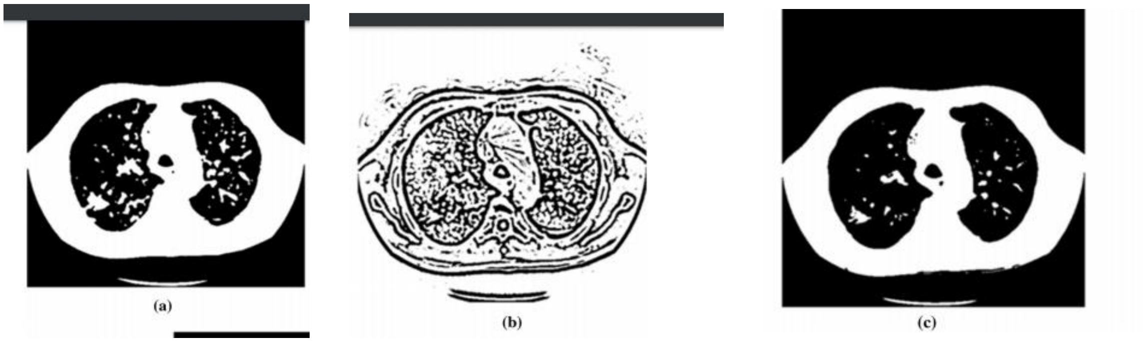


Figure 14 CT Images after applying thresholding methods

The above image shows us the result after thresholding is applied after the output of the Gaussian blur. Another problem we have to solve is to fill up the gaps that are sometimes left after thresholding, this can be done using morphological opening operation. They are very simple operations that can be performed on any image. These are of two types – erosion and dilation, both of them are generally used on binary image to perform these tasks. They can be implemented with the help of a structural element, called kernel. Like the soil erosion is responsible for the erosion of soil, erosion operation erodes the boundaries that are present in the foreground object. Let's say the kernel slides over an image then original image's pixel value will be considered '1' if any of the pixels that are present in the kernel's window is '1' else the value of the pixel will be assigned a '0'. Dilation can be seen as the opposite step of the erosion step, i.e., a value of the pixel is going to be considered '1' only if every value of the pixel encountered inside the kernel's window is a '1' else '0'. This grows the object size of our foreground.

After this we apply Feature Extraction which is defined as the taking out of information about the elementary parts of an image, these information includes the shape, density, arrangement, and the size of the object present in that image. Texture Feature Extraction can be defined as the collection of these feature using the process of texture analysis.

In the paper, they considered two methods for the extraction of texture feature:

GLCM: This can be useful for many analysis of texture feature and can be unitized to extract information related to the second order texture from an image. One measure for this intensity variation is provided at that pixel which is of our interest, calculations on this can be performed using GCLM. This consist of two parameters – relative distance present between the pair of pixel ‘d’ which is calculated in pixel number and also the relative orientation of pair of pixel ‘d’. In the direction of 0, 45, 90, 135 the value of H is quantized. Following GLCM features are used by us:

The measure of intensity contrast between any pixel and those present around it spanning the entire image is known as Contrast. It can be represented by the formula:

$$\sum_{n=0}^{G-1} n^2 \{\sum_{i=1}^{G} \sum_{j}^{G} p(i,j)\} \qquad i\text{-}j == n$$

The sum of all the pixel values p(i, j) which is is the absolute difference between the corresponding ith and jth value is called Dissimilarity. Its formula is:

$$\sum_{i=0}^{M} \sum_{j=0}^{N} p(i,j)(i-j)$$

The inverse difference moment or Homogeneity because of denominator. This causes the value of IDM to go high from images that are homogeneous and low for images that are not. It can be represented by the formula:

$$\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \frac{1}{(1+(i-j)^2)} p(i,j)$$

The measure of gray-level linear dependence at specified locations in between pixels seen relative to each other is called Correlation. It can be represented by the formula:

$$\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \frac{(i*j)*p(i,j)-\{u_x-u_y\}}{\sigma_x*\sigma_y}$$

The measure of homogeneity of any image is given by Angular second moment. There are very few gray levels in any homogeneous scene thus the values in the GCLM will be few but high in nature of p(i, j), this will make the sum of squares go high. It is expressed by the formula:

$$\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} p(i,j)^2$$

After applying GLCM the Energy of the image can be represented by the formula:

$$\text{Energy} = ASM^{1/2}$$

There are many statistical features that can be extracted from our ROI. We have extracted six of them, these are,

The total sum of all the values of pixel intensity divided by the total count of pixels is called Mean and can be expressed as the formula:

$$\mu = \frac{1}{(M*N)}\sum_{i=0}^{M}\sum_{j=0}^{N} p(i,j)^1$$

The approximation of average squared deviation of all the values of greyscale pixel p(i,j) from the mean is called Standard Deviation. It can be represented by the formula:

$$\sigma = \frac{1}{(M*N)}\sum_{i=0}^{M}\sum_{j=0}^{N}(p(i,j) - \mu)^2)^{1/2}$$

The degree to which the asymmetry of the distribution of pixel around the mean in a specified window is called Skewness. It is use to characterize the shape of the distribution as is normally a pure number. Its formula is:

$$\mathbf{S} = \frac{1}{(M*N)}\sum_{i=0}^{M}\sum_{j=0}^{N}[\frac{p(i,j)-\mu)^2}{\sigma}]^3$$

The measuring of the flatness present or the peakness present in any distribution relative to that of a normal distribution is known as Kurtosis. Its formula is:

$$K = (\frac{1}{(M*N)} \sum_{i=0}^{M} \sum_{j=0}^{N} [\frac{p(i,j) - \mu)^2}{\sigma}]^4$$

The moments of probability distribution that is done about the mean of a random variable and can be used to measure the value's deviation connected with the exact variable from average value of the distribution is given by the Fifth and Sixth central moment. Its formula is:

$$(\frac{1}{(M*N)} \sum_{i=0}^{M} \sum_{j=0}^{N} [\frac{p(i,j) - \mu)^2}{\sigma}]^5$$

$$(\frac{1}{(M*N)} \sum_{i=0}^{M} \sum_{j=0}^{N} [\frac{p(i,j) - \mu)^2}{\sigma}]^6$$

The measure of the error between the known and the predicted value of a pixel is called the Root Mean Squared Error**.** Its formula is:

$$\sqrt{\frac{p(i,j)^2}{M*N}}$$

For each image present in the dataset along with their respective name, the features listed above are extracted and all this is stored in a matrix. Classification of the image step is achieved after the analysis of the image step is completed with the aid of various machine learning approaches. There are various machine learning algorithms that can be incorporated at this step. The classification of image, i.e., to classify an image as benign ('0') or malignant ('1'). At this stage, the dataset is divided into two parts-one for training purposes and the other for testing purposes. After fine-tuning parameters on the test set that is held, we can train on the training dataset and in between test continuously. We will also need to shuffle the dataset every time it is passed as without this both testing and training dataset might get imprinted on the memory of the model which can lead to overfitting on the dataset which will lead to poor results at the time of testing on actual CT volumes. We will also need to look closely at the ratio in which test and train set are divided so as to keep suitable amount of data for both the purposes.

These issues will be resolved by splitting our dataset into three subsets – one that will be used for training, the other for cross validation at the end of each epoch and the last one for testing on models. The task of the train set is to assist the algorithm in finding features by which it can identify and decide which class the image belongs to, whereas with the help of the test set we can determine which of the approaches we have used is best performed and should be used to test in real time. The cross validation set is used to fine-tune the values of various parameters and hyper-parameters in order to maximize the accuracy of our approaches, this set can also be used for the evaluation of the final performance of our system. This helps in reducing overfitting of our model, as it can so happen that the train set might start bleed into the test set even after we are shuffling continuously.

Thus the problem that our model might memorize the whole dataset is solved now and now we can use it to predict the labels of testing class. We can now also bring the cross validation set that the model hasn't seen yet and would therefore only be able to perform well on it if it has generalized well enough. Our spilt is the one of 70 per cent used for the train set, and we give 15 per cent each for both the validation and test collection. Many algorithms like k-NN, SVM, naïve Bayes, the classifier of stochastic gradient descent, decision trees, multilayer perceptron and decision trees are used in this project.

CHAPTER-3
SYSTEM DEVELOPMENT

3.1 TECHNOLOGY DECISIONS

Overview

In this section, I give details about the technologies that I have used for this project. Although there are many tools that exist out there in the market, I've found that these tools perform well for the problem that needs to be solved.

Google Colaboratory

It was developed by google can be seen as a research project with the intention of helping to disseminate research and education related to machine learning. The environment is that of Jupyter notebook that doesn't require any setting before using and can run entirely on cloud. The model is being developed using Google Colab as I don't have the required GPU and memory storage available with me offline.

Anaconda

It is a popular data science platform from where you can create data science projects and machine learning. It consist of many different libraries such as NumPy, SciPy, Pandas, Math, Matplotlib, Keras, Tensorflow etc come with the Anaconda and IDE's such as Jupyter Notebook, Spyder and etc.

Jupyter Notebooks

It is a software IDE that is open source which helps the developers in creation and sharing of documents that contain live code and many more things. We are using multiple jupyter notebooks -for training as well as testing. In the later stages these will be unified to make the software application.

Python

It is interpreted language and high level in nature which can be used for programming general purpose programs. Widely used for scientific computing, it can be utilized for a extensive variety of programming jobs like machine learning, data mining, web application development etc. It is the main language that is used in the creation of this project. Majority of our work is to done using python and libraries available with it.

Numpy

Numpy is a library in Python that is used for efficient numerical computing in Python. This library is highly enhanced to do mathematical tasks. In the project workflow Numpy is heavily used in data pre-processing and preparation. One of the main features about Numpy is it's highly efficient n-dimensional array (ndarray). Compared to a list in Python a Numpy array can be n-dimensions and has more features associated with the ndarray. Numpy can also perform way more efficient mathematical operations compared to the math library in Python.

Pandas

Pandas is a library in Python, like numpy is also used for data pre-processing and preparation. One of the main features about pandas is the DataFrame and Series data structure. These data structures are optimized and contain fancy indexing that allow a variety of features such as reshaping, slicing, merging, joining and etc to be available. Pandas and Numpy both are extremely powerful when used together for manipulating data.

Matplotlib

It is a plotting library in the Pytho programming language that allows the programmers to create graphs and visualizations that can be used for wide variety of tasks with ease of use. Its greatest feature is that it integrates very well with Jupyter Notebook and creating visualizations is simplified. Matplotlib also works very well with pandas and numpy. Matplotlib is extensively used in the project for the compilation of results.

OpenCV (Open Source Computer Vision)

It is a python computer vision library. It was mainly written in the C/C++ language and is abstracted to interface with Java, C++, and Python. It is a myriad of different powerful tools that can be used for working with images, extraction of features, data manipulation of image etc.

Pydicom

It is a python package that is designed to work with DICOM files. It's made for two tasks - inspection and modification of DICOM data and doing that in a laidback way. These changes can be again written into a new file. Being a pure package, it can be executed anywhere where python can be executed without anyother necessities, although you might need NumPy for manipulation of pixel data.

Tensorflow

It is a deep learning library by developed and made open source by Google. It was originally developed by engineers at Google who were working on the famous Google Brain and has been used for research on machine learning and deep learning. Tensorflow at its core is about computations of multidimensional arrays called tensors but what makes Tensorflow great is its ability to be flexible to deploy computations on different devices such as CPU's and GPU's.

Keras

Keras is also a Deep Learning Framework that summaries much of the code in the other Frameworks like Tensorflow and Theano. Compared with the other frameworks Keras is more minimalist. Keras is easy to use and the availability of multiple type of layers makes using it very time saving for any project.

3.2 MODEL DEVELOPMENT

Dataset Information

The dataset that we are using is provided by the National Cancer Institute (NCI) as well as American Association of Physicists in Medicine (AAPM) for the LUNGx challenge conducted as a part of 2015 SPIE Medical Imaging Conference to carry out the task of Lung Nodule Classification between Benign and Malignant.

The dataset contains 10 CT scan volumes for the purpose of training and 60 CT scan volumes for the purpose of testing. However, deep learning being a data-centric approach, it needs a lot of data to garner important features that can help it in achieving this task. Keeping this in mind we have divided the whole dataset in three sets –train set, validation set, and test set. The ratio of these sets are as follows: 56 volumes (80% of dataset) for training set, 4 volumes(5.72% of dataset) for validation set, 10 volumes(14.28% of dataset) for testing.

The above ratio is not a hard coded ratio, any other researcher may find any other ratios suitable and can carry out working with the same. Depending the availability of more CT volumes the researcher can also that to the dataset to increase the dataset size or perform various operations like flipping, zooming, distorting to increase the size of the dataset.

CNN Architectures Implemented

Until now we have made two CNN Architectures and heavily looked into the conditions for choosing suitable parameters as well as hyper-parameters so that we can achieve the maximum performance those architecture are capable of showing. These two architectures are – VGG and DenseNet. We will discuss the specifications like as layer (CNN and dense) specifications, optimizer specifications, initializers and regularizers specifications of both of these in details below.

VGG Architecture

- CNN Layers - Four blocks each with
  - Three Conv-BatchNorm-Relu.
  - Conv starting with 64 filters and going up to 512 filters in the final block.
  - Same padding.
  - Filter size in the first block is 5 X 5 and in all others is 3 X 3.
  - Max pooling with 2 X 2 acting as a bridge between blocks.

- Flatten layer act as a bridge between CNN and Fully-Connected Layers.

- Fully-Connected Layers – Three FC layers with
  - 256, 64 and 1 neurons respectively.
  - Final layer has sigmoid function as activation.

- Optimizer – Adam Optimizer with
    - Default values of parameters ß1, ß2 and decay.
    - Learning rate: 10^{-7}.
    - Kernel Initializer – glorot_normal.

- No regularizer (L1 or L2) is used in this architecture.

- Batch size used is 24.

- Information on Parameters
  - Total params: 48, 364, 545.
  - Trainable params: 48, 352, 641.
  - Non-trainable params: 11, 904.

VGG Model Images

Full Model

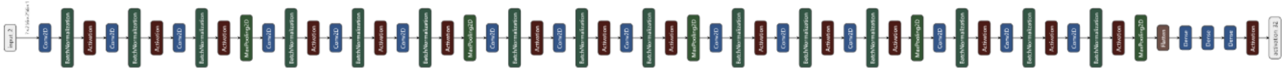


Figure 15 Full VGG Model

CNN Layers Block

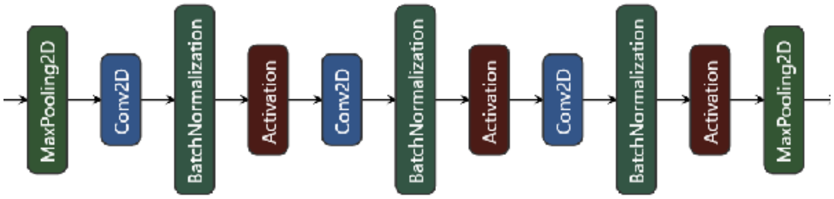


Figure 16 One VGG Block

Convolution to Fully Connected Layer

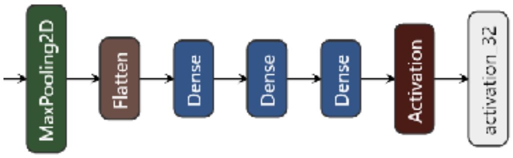


Figure 17 VGG Fully Connected Layers

DenseNet Architecture

- Dense CNN Layers – Seven Dense blocks each with
    - 3 Conv-BatchNorm-Relu layers.
    - Growth rate of 4 after each C-BN-R layer.
    - In between dense blocks one Conv-BatchNorm-Relu and 2 X 2 Max Pooling.
    - Conv starting with 8 filters and going up to 1024 filters in the final block.
    - Same padding and Filter size in all CNN layers is 3 X 3.

- Flatten layer act as a bridge between Dense and Fully-Connected Layers.

- Fully-Connected Layers – Three FC layers with
    - 512, 128 and 1 neurons respectively.
    - Final layer has sigmoid function as activation, others have Relu.

- Optimizer – Adam Optimizer with
    - Default values of parameters ß1, ß2 and decay.
    - Learning rate: 10^{-7}.
    - Kernel Initializer – glorot_normal.

- Regularizer L2(0.01) is used in this architecture.

- Batch size used is 24.

- Information on Parameters
    - Total params: 40, 257, 269.
    - Trainable params: 40, 251, 741.
    - Non-trainable params: 5, 528.

DenseNet Model Images

Full Model

Figure 18 Full DenseNet Model

Single Dense Block

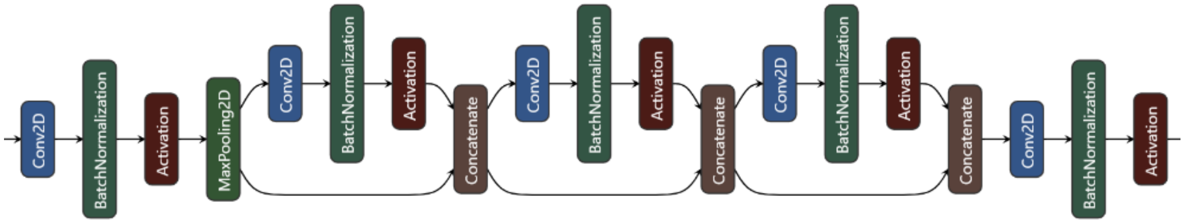


Figure 19 One Dense Block

Conv to Fully Connected Layer

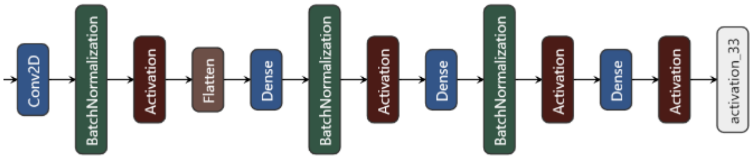


Figure 20 DenseNet Fully Connected Layers

3.3 ALGORITHMS

Create_Dataset Function

```
1. Input os paths of all the training and testing volumes.
2. For all the training and testing volumes, repeat
      a. Load all the files in that volume.
      b. Convert those dicom files into numpy arrays.
      c. Resave them in another place.
3. End.
```

Create_CSV_Image_Label Function

```
1. Input os paths of all the training and testing volumes.
2. Create a dataframe of the output labels of each volumes from the csv file provided.
3. For all the training and testing volumes, repeat
      a. Load all the os paths of all the files in that volume
      b. Save the os paths and the output label as a tuple.
      c. Append this whole into a dataframe.
4. Now split the dataframe into two parts – one for training set and one for validation sets.
5. Convert the dataframe into a csv file.
6. End.
```

Counter_train & Counter_valtest (Generator Class)

```
1. Take an appropriate batch size.
2. Input the train or val csv file as applicable.
3. Convert csv into numpy array.
4. Take a subset of this array of batch size.
```

5. For each file, repeat
    a. Apply clipping and cropping.
    b. Apply reshaping and resizing.
    c. Create the x-input & y-label numpy array.
6. Check for any NANs in both the numpy arrays.
7. Return the numpy array or else show error.
8. End.

Random_CSV Function

1. Input both the csv image & label files of Training and Validation.
2. Convert them into numpy array.
3. Call random function on both.
4. Resave as csv files.
5. End.

3.4 OTHER ANALYTICAL APPROACHES TAKEN

1. **Early Stopping:** This is employed so that the model can decide for itself when overfitting is starting and stop training to stop this.

2. **Model Checkpoint:** Weights of all the epochs are saved so that if any time the program stopped in the middle it can be recovered from the last epoch that was successfully completed.

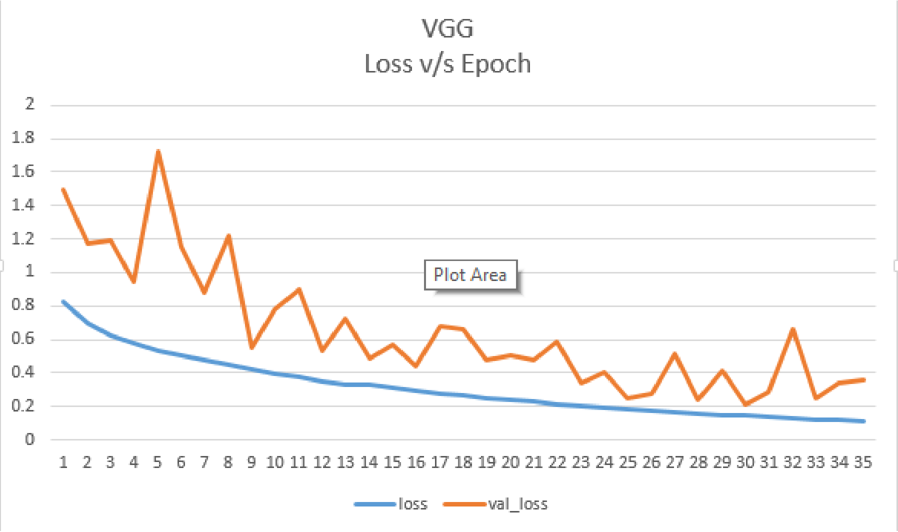
3. **CSV Logger:** Logging the values of accuracy measures and loss functions after each epoch so that they can be later plotted and information can be shown in a better manner.

CHAPTER-4

PERFORMANCE ANALYSIS

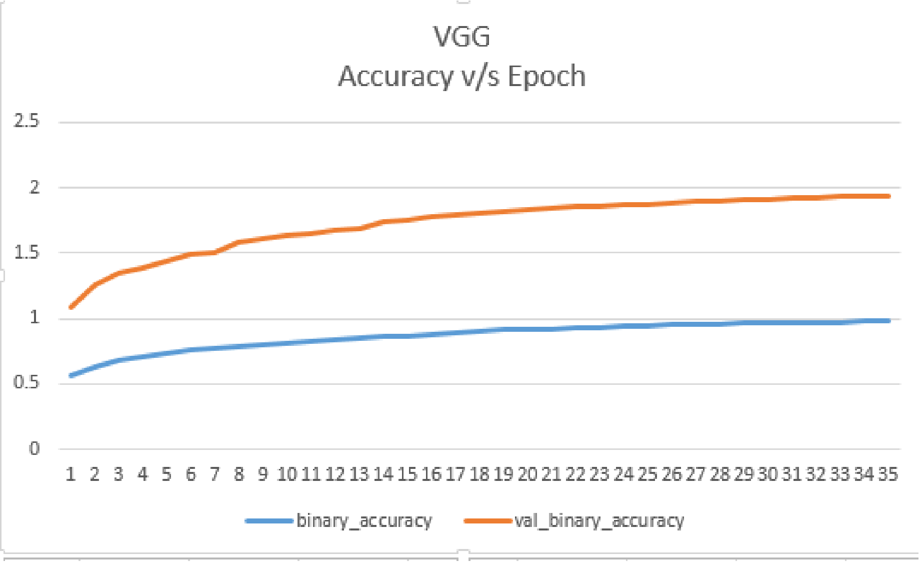
4.1 RESULTS FOR VGG

Loss v/s Epochs



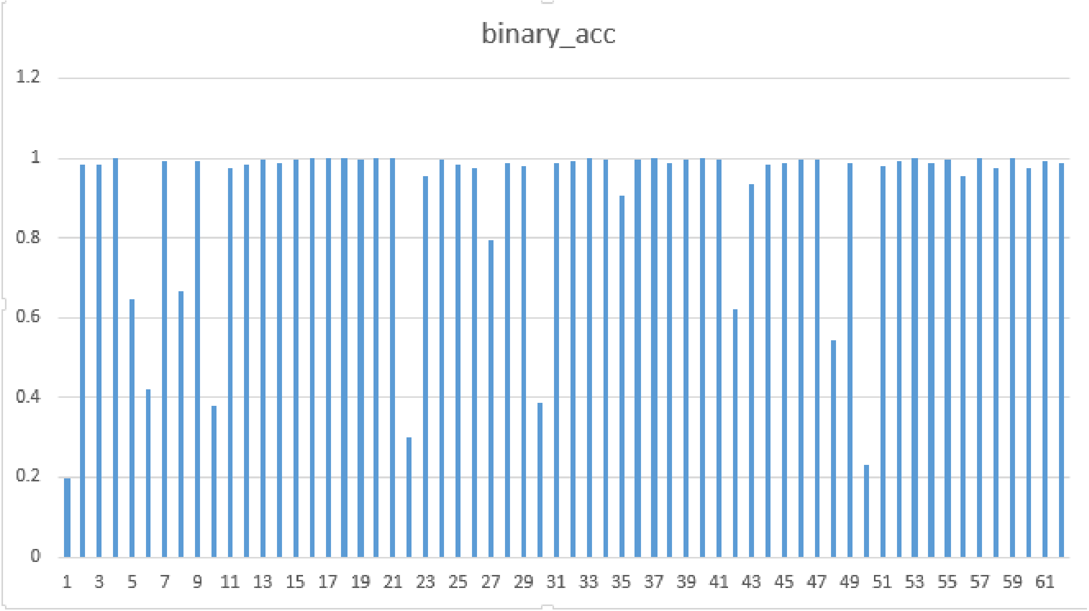
Graph 1

Accuracy v/s Epochs



Graph 2

Volume-wise Testing Results



Graph 3

Final Training Results

epoch	binary_accuracy	loss	val_binary_accuracy	val_loss
0	0.565313	0.825575	0.524637	0.663891
1	0.625188	0.700722	0.62945	0.472846
2	0.675875	0.625371	0.673312	0.562034
3	0.706	0.580393	0.686414	0.364412
4	0.72925	0.536034	0.713472	1.189756
5	0.752563	0.502988	0.735973	0.654694
6	0.770063	0.473952	0.737397	0.405791
8	0.7865	0.450464	0.798918	0.768678
9	0.799375	0.425596	0.808886	0.125569
10	0.816625	0.398345	0.812589	0.379836
11	0.828688	0.378668	0.820564	0.52276
12	0.843625	0.352693	0.831387	0.177103
13	0.856313	0.33357	0.836229	0.386214
14	0.857563	0.334254	0.88123	0.148403
15	0.869063	0.310157	0.888636	0.255123
16	0.879625	0.297107	0.902877	0.147678
17	0.891875	0.278986	0.902592	0.403382

18	0.899375	0.267744	0.908288	0.39491
19	0.909063	0.251869	0.910852	0.222262
20	0.91625	0.238537	0.916833	0.262994
21	0.917938	0.230295	0.919681	0.246195
22	0.925875	0.214965	0.927371	0.37025
23	0.934813	0.206684	0.925377	0.129309
24	0.94025	0.197283	0.930504	0.205424
25	0.941875	0.18818	0.932498	0.058822

Table 1

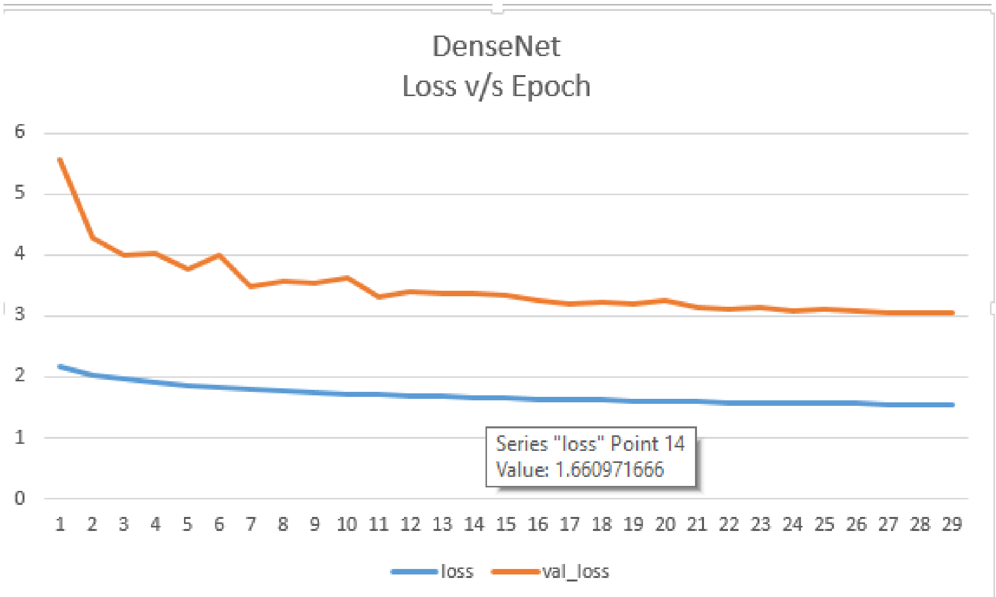
Mean Testing Results

	volume	**binary_acc**	**recall**	**time**
mean	All	0.907478	0.941405	22.97629
std	All	0.196217	0.118662	3.619902

Table 2

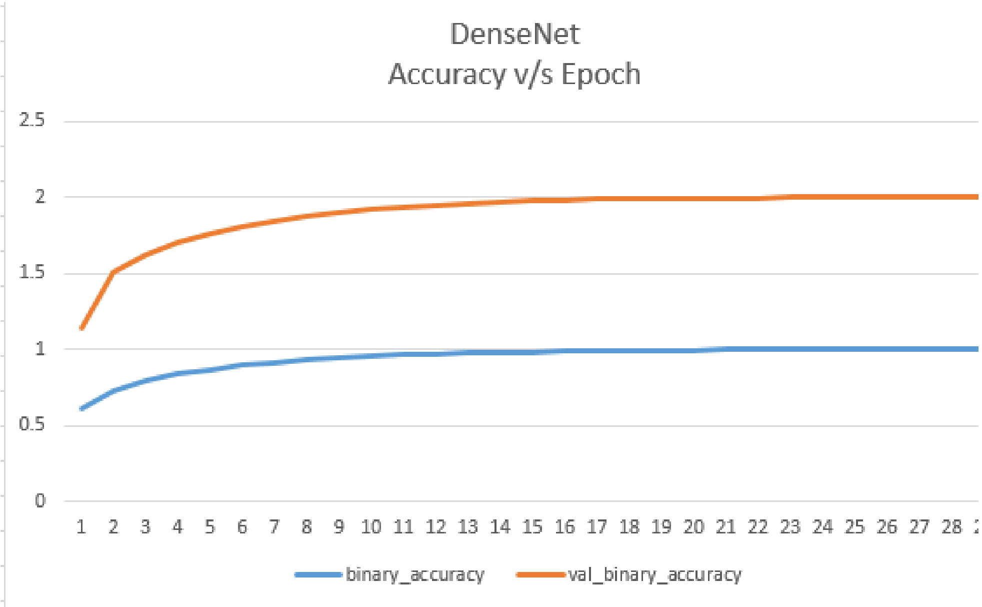
4.2 RESULTS FOR DENSENET

Loss v/s Epochs



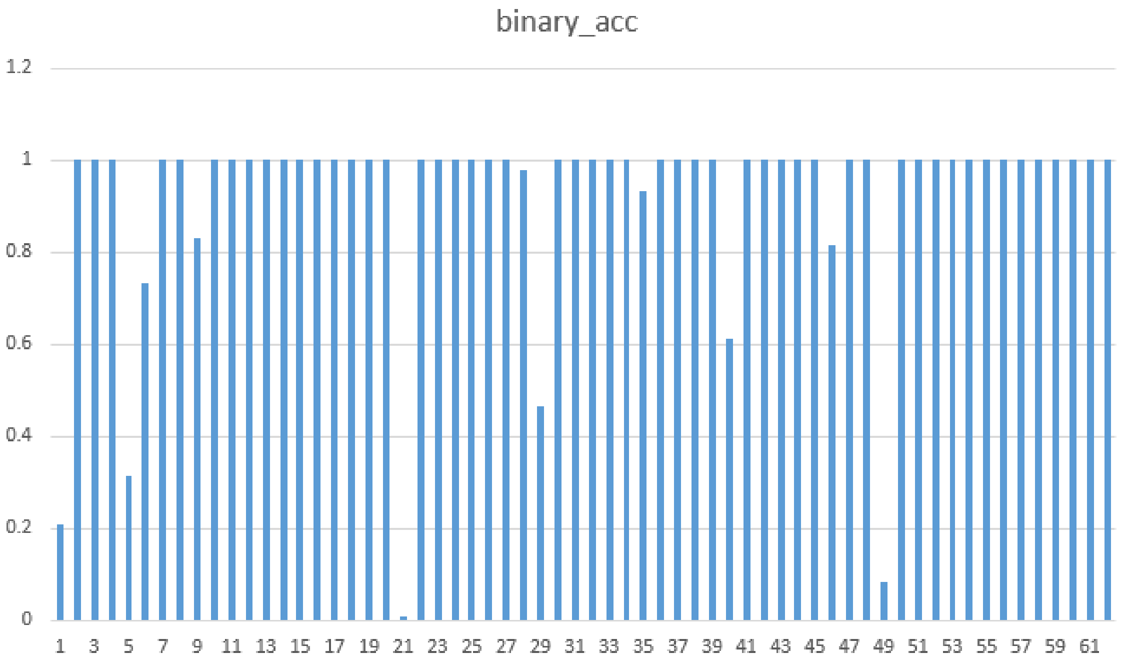
Graph 4

Accuracy v/s Epochs



Graph 5

Volume-wise Testing Results



Graph 6

Final Training Results

epoch	binary_accuracy	loss	val_binary_accuracy	val_loss
0	0.611	2.175982	0.52495	3.376263
1	0.730563	2.038971	0.781865	2.248262
2	0.792188	1.966137	0.826632	2.02013
3	0.836875	1.910252	0.871115	2.11265
4	0.867125	1.867116	0.892216	1.912991
5	0.893563	1.828668	0.918164	2.158024
6	0.910875	1.798796	0.932421	1.689144
7	0.931125	1.77153	0.944967	1.801228
8	0.943	1.745639	0.956943	1.803495
9	0.952875	1.724803	0.967208	1.910019
10	0.961438	1.706659	0.9712	1.592941
11	0.969375	1.690209	0.980325	1.70887
12	0.975688	1.673098	0.984887	1.684247
13	0.97975	1.660972	0.990305	1.710631
14	0.984063	1.645219	0.991731	1.69995

15	0.987375	1.635752	0.994297	1.604241
16	0.992	1.623406	0.996578	1.568368
0	0.989688	1.619305	0.998574	1.594779
1	0.9925	1.60709	0.998289	1.584654
2	0.993625	1.596661	0.999145	1.664391
3	0.9965	1.588375	0.99943	1.548215
4	0.996125	1.581519	0.99943	1.529015
5	0.99725	1.574111	0.99943	1.567194
6	0.997438	1.56857	0.999715	1.522747
7	0.997563	1.562633	1	1.558361
8	0.997438	1.558177	1	1.524769
9	0.998188	1.552676	1	1.511056
10	0.998563	1.546248	1	1.508842
11	0.998875	1.542286	1	1.50771

Table 3

Mean Testing Results

	volume	**binary_acc**	**recall**	**time**
mean	All	0.930612	0.953663	11.21085
std	All	0.207309	0.141193	1.988164

Table 4

CHAPTER-5

CONCLUSIONS

5.1 CONCLUSION

We used the dataset provided by the National Cancer Institute **(**NCI**)** as well as American Association of Physicists in Medicine **(**AAPM**)** for the LUNGx challenge conducted as a part of 2015 Conference on SPIE Medical Imaging to carry the task of Lung Nodule Classification between Benign and Malignant**.** Making this project provided us with a unique opportunity to create a state of art algorithm and to compare the same with students of other academia, industry, and government**.**

We preprocessed the data to change it from dicom files to numpy arrays so that the computations on the dataset will become easy**.** After this the volumes are cropped based on a square box created with a diameter of 128 pixels from their Nodules Center provided in the excel sheet of dataset**.**

The two algorithms that we have implemented till now are:
VGG using which our final train accuracy is **94.18%** and test accuracy is **90.74%.**
DenseNet using which our final train accuracy **is 99.88%** and test accuracy is **93.06%.**

It is concluded that DenseNet performs better than VGG, it also requires less time and parameters and is thus efficient and effective.

Aman Srivastav (161328)

5.2 FUTURE SCOPE

- Researching better Data Preprocessing Techniques to enhance the results,
  - Standard Normalization.
  - Row-wise/Column-wise Normalization.
  - Clipping and Cropping.
  - Resizing and Reshaping.

- Researching about various other Deep Learning CNN Architectures,
  - UNet.
  - Dense-UNet.
  - ResNet.
  - InceptionNet.
  - AlexNet.

- Further enhancing the implementation of following functions,
  - Loading Dataset.
  - Create dataset.
  - Crease CSV for Images and Labels.
  - Predict for Volumes.

- Further Hyper tuning of following parameters for better models,
  - Learning Rate.
  - No. of Hidden Layers.
  - Train-Dev-Test Sets Split.
  - Mini-batch Size.
  - Number of Epochs.

REFERENCES

1. Cancer Research UK (2017) Cancer mortality for common cancers. http://www.cancerresearchuk.org/health-professional/can cer-statistics/mortality/common-cancers-compared. Accessed May 2017

2. Dimililer K, Ugur B, Ever YK (2017) Tumor detection on CT lung images using image enhancement. Online J Sci Technol 7(1):133–138

3. Al-tarawneh MS (2012) Lung cancer detection using image processing techniques. Leonardo Electron J Pract Technol 20:147–58

4. Armato III SG, Hadjiiski L, Tourassi GD, Drukker K, Giger ML, Li F, Redmond G, Farahani K, Kirby JS, Clarke LP (2015) SPIEAAPM-NCI Lung nodule classification challenge dataset. The Cancer Imaging Arch. https://doi.org/10.7937/K9/TCIA.2015. UZLSU3FL

5. Gonzalez RC, Woods RE (2002) Digital image processing. Prentice Hall, Upper Saddle River, NJ, pp 797–800

6. Dwivedi MS, Borse MR, Yametkar MA (2014) Lung cancer detection and classification by using machine learning and multinomial Bayesian. IOSR J Electron Commun Eng (IOSRJECE) 9(1):69–75

7. Sun W, Zheng B, Lure F, Wu T, Zhang J, Wang BY, Saltzstein EC, Qian W (2014) Prediction of near-term risk of developing breast cancer using computerized features from bilateral mammograms. Comput Med Imaging Graph 38(5):348–357

8. Chaudhary A, Singh SS (2012) Lung cancer detection on CT images by using image processing. In: Proceedings of 2012 IEEE international conference on computing sciences (ICCS). pp 142–146

9. Pratap GP, Chauhan RP (2016) Detection of Lung cancer cells using image processing techniques. In: Proceedings of IEEE international conference on power electronics, intelligent control and energy systems (ICPEICES). pp. 1–6

10. Bhusri S, Jain S, Virmani J (2016) Classification of breast lesions based on laws' feature extraction techniques. In: Proceedings of 3rd international conference on computing for sustainable global development (INDIACom). pp. 1700–1704

11. Kuruvilla J, Gunavathi K (2014) Lung cancer classification using neural networks for CT images. Comput Methods Programs Biomed 113(1):202–209

12. Mitra S, Pal SK (1995) Fuzzy multi-layer perceptron, inferencing and rule generation. IEEE Trans Neural Netw 6(1):51–63

13. Amato F, Lpez A, Pea-Mndez EM, Vahara P, Hampl A, Havel J (2013) Artificial neural networks in medical diagnosis. J Appl Biomed 11:47–58

14. Karabatak M, Ince MC (2009) An expert system for detection of breast cancer based on association rules and neural network. Expert Syst Appl 36(2):3465–3469

15. Adi K, Widodo CE, Widodo AP, Gernowo R, Pamungkas A, Syifa RA (2017) Nave Bayes algorithm for lung cancer diagnosis using image processing techniques. Adv Sci Lett 23(3):2296–2298

16. Joachims T (1998) Making large-scale SVM learning practical (No. 1998, 28). In: Technical Report, SFB 475: Komplexittsreduktion in Multivariaten Datenstrukturen, Universitt Dortmund, pp 1–18

17. Tidke SP, Chakkarwar VA (2012) Classification of lung tumor using sVM. Int J Comput Eng Res 2(5):1254–1257

18. Touw WG, Bayjanov JR, Overmars L, Backus L, Boekhorst J, Wels M, van Hijum SA (2012) Data mining in the life sciences with random forest: A walk in the park or lost in the jungle? Brief. Bioinform. 14(3):315–326

19. Shi T, Seligson D, Belldegrun AS, Palotie A, Horvath S (2005) Tumor classification by tissue microarray profiling: random forest clustering applied to renal cell carcinoma. Mod Pathol 18(4):547–557

20. Ramos-Gonzlez J, Lpez-Snchez D, Castellanos-Garzn JA, de Paz JF, Corchado JM (2017) A CBR framework with gradient boosting based feature selection for lung cancer subtype classification. Comput Biol Med 86:98–106

21. Sakamoto M, Nakano H, Zhao K, Sekiyama T (2017) Multi-stage neural networks with single-sided classifiers for false positive reduction and its evaluation using Lung X-ray CT images. ArXiv preprint arXiv:1703.00311, pp 1–11

22. Demyanov S, Chakravorty R, Abedini M, Halpern A, Garnavi R (2016) Classification of dermoscopy patterns using deep convolutional neural networks. In: Proceedings of 13th international symposium on biomedical imaging (ISBI). pp 364–368

23. Lopez AR, Giro-i-Nieto X, Burdick J, Marques O (2017) Skin lesion classification from dermoscopic images using deep learning techniques. In: Proceedings of 13th IASTED international conference on biomedical engineering (BioMed). pp 49–54

24. Bewal R, Ghosh A, Chaudhary A (2015) Detection of breast cancer using neural networks a review. J Clin Biomed Sci 5(4):143–148

25. Havaei M, Davy A, Warde-Farley D, Biard A, Courville A, Bengio Y, Pal C, Jodoin PM, Larochelle H (2017) Brain tumor segmentation with deep neural networks. Med image Anal 35:18–31

26. Weng S, Xu X, Li J, Wong ST (2017) Combining deep learning and coherent anti-Stokes Raman scattering imaging for automated differential diagnosis of lung cancer. J Biomed Opt 22(10):106017

27. Sun W, Zheng B, Qian W (2017) Automatic feature learning using multichannel ROI based on deep structured algorithms for computerized lung cancer diagnosis. Comput Biol Med 89:530–539

28. Mahbod A, Ecker R, Ellinger I (2017) Skin lesion classification using hybrid deep neural networks. arXiv preprint arXiv:1702.08434, pp 1–5

29. Esteva A, Kuprel B, Novoa RA, Ko J, Swetter SM, Blau HM, Thrun S (2017) Dermatologist-level classification of skin cancer with deep neural networks. Nature 542(7639):115–118

30. Behrmann J, Etmann C, Boskamp T, Casadonte R, Kriegsmann J, Maass P (2017) Deep learning for tumor classification in imaging mass spectrometry. Bioinformatics 1:1–10

APPENDICES

7.1 TRAINING CODE

Create Dataset Function

```
def create_dataset():
  train_volume_path = 'dataset/train/'
  test_volume_path = 'dataset/test/'

  new_train_path = 'dataset_new/train/'
  new_test_path = 'dataset_new/test/'

  train_volumes = np.array(sorted(os.listdir(train_volume_path)))
  test_volumes = np.array(sorted(os.listdir(test_volume_path)))

  #for train set
  for i in train_volumes:
    print(i)

    X = []

    for j in sorted(os.listdir(train_volume_path + i + '/')):
      X.append(train_volume_path + i + '/' + j)

    #loop through all the DICOM files
    j = 0
    for filenameDCM in X:
      #read the file
      ds = dicom.read_file(filenameDCM)

      #store the raw image data
      np.save(new_train_path + i + '/' +  '%04d' % j, ds.pixel_array)
      j+=1

  #for test set
  for i in test_volumes:
    print(i)

    X = []
    for j in sorted(os.listdir(test_volume_path + i + '/')):
      X.append(test_volume_path + i + '/' + j)

    #loop through all the DICOM files
    j = 0
    for filenameDCM in X:
      #read the file
      ds = dicom.read_file(filenameDCM)

      #store the raw image data
      np.save(new_test_path + i + '/' +  '%04d' % j, ds.pixel_array)
      j+=1

create_dataset()
```

Create CSV for images and labels

```
def create_csv_image_label():
  new_train_path = 'dataset_new/train/'
  new_test_path = 'dataset_new/test/'

  df_data = np.array(pd.read_excel('dataset_new/TestSet_NoduleData_PublicRelease_wTruth.xlsx', nrows=73,  usecols='A,E', squeeze=True, dtype=str))
  for x in df_data:
    if x[1][0] == 'B':
      x[1] = 0.
    else:
      x[1] = 1.

  df = {x:y for x, y in df_data}
  df['LUNGx-CT005'] = 1.
  df['LUNGx-CT054'] = 0.
  df['LUNGx-CT056'] = 0.

  data_train = []

  data_valtest = []
  train_volume_paths = np.array(sorted(os.listdir(new_train_path)))
  test_volume_paths = np.array(sorted(os.listdir(new_test_path)))
  #for training data
  for i in train_volume_paths:
      for j in sorted(os.listdir(new_train_path + i + '/')):
          if i[0] == 'B':
            data_train.append([i + '/' + j, 0.])
          else:
            data_train.append([i + '/' + j, 1.])

  #also taking 50 volumes from testing data
  for i in test_volume_paths:
      for j in sorted(os.listdir(new_test_path + i + '/')):
        if int(i[8:]) <= 50:
          data_train.append([i + '/' + j, df[i]])

  np.random.shuffle(data_train)

  data_valtest = data_train[16000:]
  data_train = data_train[:16000]

  df_train = pd.DataFrame(data_train)
  df_train.to_csv(r'df_train.csv')

  df_valtest = pd.DataFrame(data_valtest)
  df_valtest.to_csv(r'df_valtest.csv')

create_csv_image_label()
```

Generator Class

```
class Counter_train:
    def __init__(self, low, high):
        self.current = low
        self.high = high

    def __iter__(self):
        return self

    def __next__(self):
        if self.current > self.high:
          #create_csv_image_label()
          random_csv()
          self.current = 0
        else:
          batch_size = 24
          df_new = pd.read_csv('df_train.csv', index_col=0, squeeze=T
rue, dtype=str)
          df_new = np.array(df_new)

          paths = df_new[self.current:self.current + batch_size]

          image_final_shape = (paths.shape[0], 256, 256, 1)
          label_final_shape = (paths.shape[0], 1)
          clip_len_min = 0.0
          clip_len_max = +4000.0

          X = []
          y = []

          for path in paths:
            if path[0][:2] == 'LU':
              new_volume_X = np.load('dataset_new/test/' + path[0]).a
stype('float32')
            else:
              new_volume_X = np.load('dataset_new/train/' + path[0]).
astype('float32')

            new_volume_X = new_volume_X[128:384, 128:384]

            #Normalizing X
            X_norm = (new_volume_X - clip_len_min)/(clip_len_max - cl
ip_len_min)

            #Append
            X.append(X_norm)
            y.append(path[1])
          X = np.array(X)
          y = np.array(y)

          #Reshaping X and y
          X_final = X.reshape(image_final_shape)
          y_final = y.reshape(label_final_shape).astype('float32')

          assert (not np.any(np.isnan(X_final)))
          assert (not np.any(np.isnan(y_final)))

          self.current += batch_size

          return X_final, y_final
```

Training Model

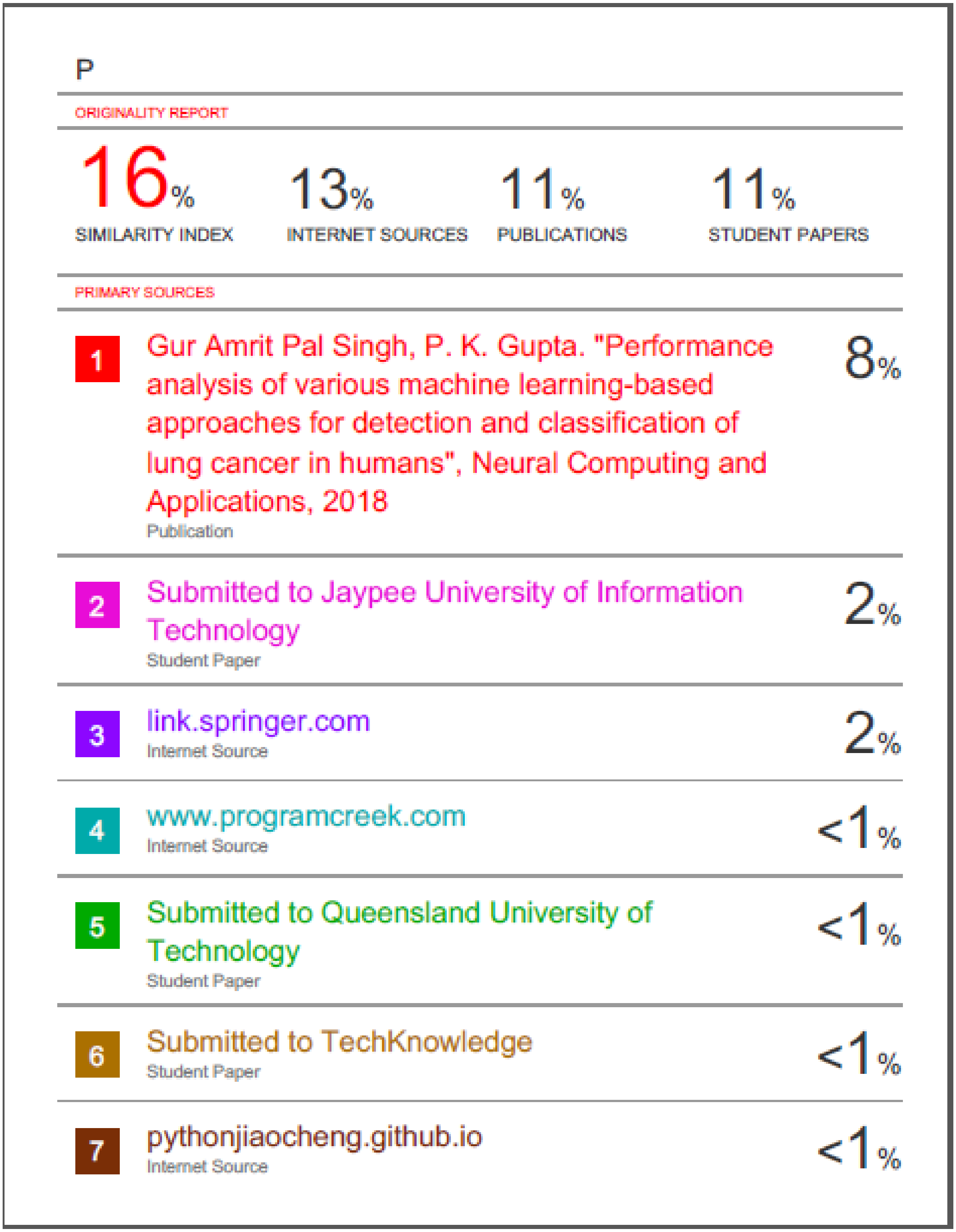
```
batch_size = 24
num_of_epochs = 150

df_train = pd.read_csv('df_train.csv', index_col=0)
final_train_size = df_train.shape[0]
train_steps = int(np.ceil(final_train_size / batch_size))

df_valtest = pd.read_csv('df_valtest.csv', index_col=0)
final_valtest_size = df_valtest.shape[0]
valtest_steps = int(np.ceil(final_valtest_size / batch_size))

weight_path = '../gdrive/My Drive/nuns4/weights_{epoch:02d}-{val_binary_accuracy:.2f}.hdf5'
cp = ModelCheckpoint(weight_path, monitor='val_binary_accuracy', save_best_only=False, save_weights_only=True, mode='max')
es = EarlyStopping(monitor='val_binary_accuracy', min_delta=0, patience=4, mode='max')
csv = CSVLogger('../gdrive/My Drive/Pro_2020.csv', separator=',', append=True)
callbacks_list = [es, cp, csv]

path = '../gdrive/My Drive/nuns4/weights_14-0.84.hdf5'
model.load_weights(path)
model.fit_generator(Counter_train(0, final_train_size), steps_per_epoch=train_steps, epochs=num_of_epochs, validation_data=Counter_valtest(0, final_valtest_size), validation_steps=valtest_steps, callbacks=callbacks_list, initial_epoch=14)
```



P

ORIGINALITY REPORT

16%	13%	11%	11%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Gur Amrit Pal Singh, P. K. Gupta. "Performance analysis of various machine learning-based approaches for detection and classification of lung cancer in humans", Neural Computing and Applications, 2018 Publication	8%
2	Submitted to Jaypee University of Information Technology Student Paper	2%
3	link.springer.com Internet Source	2%
4	www.programcreek.com Internet Source	<1%
5	Submitted to Queensland University of Technology Student Paper	<1%
6	Submitted to TechKnowledge Student Paper	<1%
7	pythonjiaocheng.github.io Internet Source	<1%

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: ..21th July,2020...

Type of Document (Tick): PhD Thesis | M.Tech Dissertation/ Report | [✓ B.Tech Project Report] | Paper

Name: AMAN SRIVASTAV **Department:** CSE **Enrolment No** 161328

Contact No. 6394921447 **E-mail.** amansri2997@gmail.com

Name of the Supervisor: DR. PRADEEP KUMAR GUPTA

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): PERFORMANCE ANALYSIS OF DEEP LEARNING BASED APPROACHES FOR DETECTION AND CLASSIFICATION OF LUNG CANCER IN HUMANS.

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

Complete Thesis/Report Pages Detail:

- Total No. of Pages = 66
- Total No. of Preliminary pages = 10
- Total No. of pages accommodate bibliography/references = 9
  (Pg. no. 48 – 56)

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at .... (%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

Pgupta

(Signature of Guide/Supervisor)

Pgupta

Signature of HOD

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

Copy Received on	**Excluded**	**Similarity Index (%)**	**Generated Plagiarism Report Details** (Title, Abstract & Chapters)	
	• All Preliminary Pages • Bibliography/Images/Quotes • 14 Words String		Word Counts	
Report Generated on			Character Counts	
		Submission ID	Total Pages Scanned	
			File Size	

Checked by

Name & Signature **Librarian**

....

Please send your complete thesis/report in (PDF) with Title Page, Abstract and Chapters in (Word File) through the supervisor at plagcheck.juit@gmail.com