VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belgaum-590018

A PROJECT REPORT **(15CSP85)** ON

"Breast Cancer Detection Using Deep Convolution Neural Network"

Submitted in Partial fulfillment of the Requirements for the Degree of

Bachelor of Engineering in Computer Science & Engineering

By

RAJANIPRIYA N (1CR16CS125)

SANDHYARANI (1CR16CS146)

SANNIDHI A SHETTY (1CR16CS150)

SWATHI PRIYA N (1CR16CS170)

Under the Guidance of,

Mrs.Manjima R L

Asst.Professor, Dept. of CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR INSTITUTE OF TECHNOLOGY

#132, AECS LAYOUT, IT PARK ROAD, KUNDALAHALLI, BANGALORE-560037

CMR INSTITUTE OF TECHNOLOGY

#132, AECS LAYOUT, IT PARK ROAD, KUNDALAHALLI, BANGALORE-560037

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the project work entitled **““Breast Cancer Detection Using Deep Convolution Neural Network”** carried out by **Ms. RAJANIPRIYA (1CR16CS125), Ms. SANDHYARANI (1CR16CS146)**, **Ms. SANNIDHI A SHETTY(1CR16CS150), Ms. SWATHI PRIYA N(1CR16CS170)** bonafide students of CMR Institute of Technology, in partial fulfillment for the award of **Bachelor of Engineering** in Computer Science and Engineering of the Visveswaraya Technological University, Belgaum during the year 2019-2020.It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the departmental library.

The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.

Mrs.Manjima R L	**Dr. Prem Kumar Ramesh**	**Dr. Sanjay Jain**
Asst.Professor	**Professor & Head**	**Principal**
Dept. of CSE, CMRIT	**Dept. of CSE, CMRIT**	**CMRIT**

External Viva

Name of the examiners	Signature with date
1.	________________
2.	________________

DECLARATION

We, the students of Computer Science and Engineering, CMR Institute of Technology, Bangalore declare that the work entitled "**Breast Cancer Detection Using Deep Convolution Neural Network**" has been successfully completed under the guidance of Mrs.Manjima R L, Computer Science and Engineering Department, CMR Institute of technology, Bangalore. This dissertation work is submitted in partial fulfillment of the requirements for the award of Degree of Bachelor of Engineering in Computer Science and Engineering during the academic year 2019 - 2020. Further the matter embodied in the project report has not been submitted previously by anybody for the award of any degree or diploma to any university.

Place:

Date:

Team members:

RAJANIPRIYA N (1CR16CS125) ___________________

SANDHYARANI (1CR16CS146) ___________________

SANNIDHI A SHETTY (1CR16CS150) ___________________

SWATHI PRIYA N (1CR16CS170) ___________________

ABSTRACT

For this project, we are using Deep Convolutional Neural Network (CNN) which is an advanced neural network. The dataset we use contain images (Histopathological images) from real life which are for both diseased and normal cases. CNN requires a large amount of data and days of training on high end systems to give optimum results. To avoid that, we use a process called transfer learning which make use of pertained model (trained on a dataset of 1 million images). This helps in getting closer to the optimum model. Further, there will be a training performed with our own dataset which helps us to create the optimum model for breast cancer detection.

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany a successful completion of any task would be incomplete without the mention of people who made it possible, success is the epitome of hard work and perseverance, but steadfast of all is encouraging guidance.

So, with gratitude We acknowledge all those whose guidance and encouragement served as beacon of light and crowned our effort with success**.**

We take this opportunity to express our sincere gratitude and respect to **CMR Institute of Technology, Bengaluru** for providing a platform to pursue our studies and carry out our final year project.

We have a great pleasure in expressing our deep sense of gratitude to **Dr. Sanjay Jain,** Principal, CMRIT, Bangalore, for his constant encouragement.

We would like to thank **Dr. Prem Kumar Ramesh,** Professor and Head, Department of Computer Science and Engineering, CMRIT, Bangalore, who has been a constant support and encouragement throughout the course of this project.

We consider it a privilege and honor to express our sincere gratitude to our guide **Mrs.Manjima R L, Asst.Professor,** Department of Computer Science and Engineering, for the valuable guidance throughout the tenure of this review.

We also extend our thanks to all the faculty of Computer Science and Engineering who directly or indirectly encouraged us.

Finally, we would like to thank our parents and friends for all their moral support they have given during the completion of this work.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

CAD-Computer Aided Design

DeCAF-Deep Convolutional Activation Features

FC-Fully Connected

GPU-Graphics Processing Unit

VGG-Visual Geometry Group

CHAPTER 1

INTRODUCTION

Cancer is currently a deadly disease rising across the globe. Among the several existing types of cancer, breast cancer (BC) presents two very concerning characteristics: It is the most common cancer among women worldwide. It presents a very high mortality rate when compared to other types of cancer.

Histopathological analysis remains the most widely used method for BC diagnosis and most of the diagnosis continues being done by pathologists. Applying visual inspection of histological samples under the microscope, automatic classification of histopathological images is a research topic that can make BC diagnosis faster and less prone to errors.

The detection of breast cancer is done by the analysis of either mammography or ultrasound imaging and regular check-ups. Pathologists check the microscopic elements and tissue structure for detailed analysis.

The approach in which we generally refer to as visual feature descriptors or hand-crafted features, follows a more "traditional" approach, where an evaluation of the combination of six different feature sets and four base classifiers is conducted, and the final system is defined by the combination that produces the best results in the validation set.

The reported results clearly show that the latter can achieve higher recognition rates. However, the development of such system requires longer training time, some tricks like random patches to improve performance, and still a lot of expertise from the developer to tweak the system.

1.1Relevance of the Project

Deep CNN diagnosis provides a second option for image diagnosis which can improve the reliability of experts' decision making. Advanced CNN technology has achieved great success in natural image classification and it has been used widely in bio-medical image processing. Digitized tissue histopathology has now become responsive to the application for computerized image analysis.

1.2 Scope of the Project:

- A Literature Review is a type of Review article. A literature review is a scholarly paper, which includes the current knowledge including Substantive findings, as well as theoretical and methodological contributions to a particular topic.
- Literature Reviews are often associated with academic-oriented literature, such reviews are found in academic journals and are not to be confused with book reviews that may also appear in the same publication
- Software requirement Specification is a fundamental document, which forms the foundation of the software development process. It not only lists the requirements of a system but also has a description of its major feature.
- Analysis is the process of finding the best solution to the problem. System analysis is the process by which we learn about the existing problems, define objects and requirements and evaluates the solutions.
- The chapter covers the implementation aspects of the project, giving details of the programming language and development environment used. It also gives an overview of the core modules of the project with their step by step flow.
- Result and conclusion chapter gives the result of the project and the snapshots of the project, its related information including graphs and other information of the output.
- Testing chapter gives an overview of the various types of testing incorporated during the entire duration of the project.
- The summary of the work carried, contributions if any, their utility along with the scope for further work.

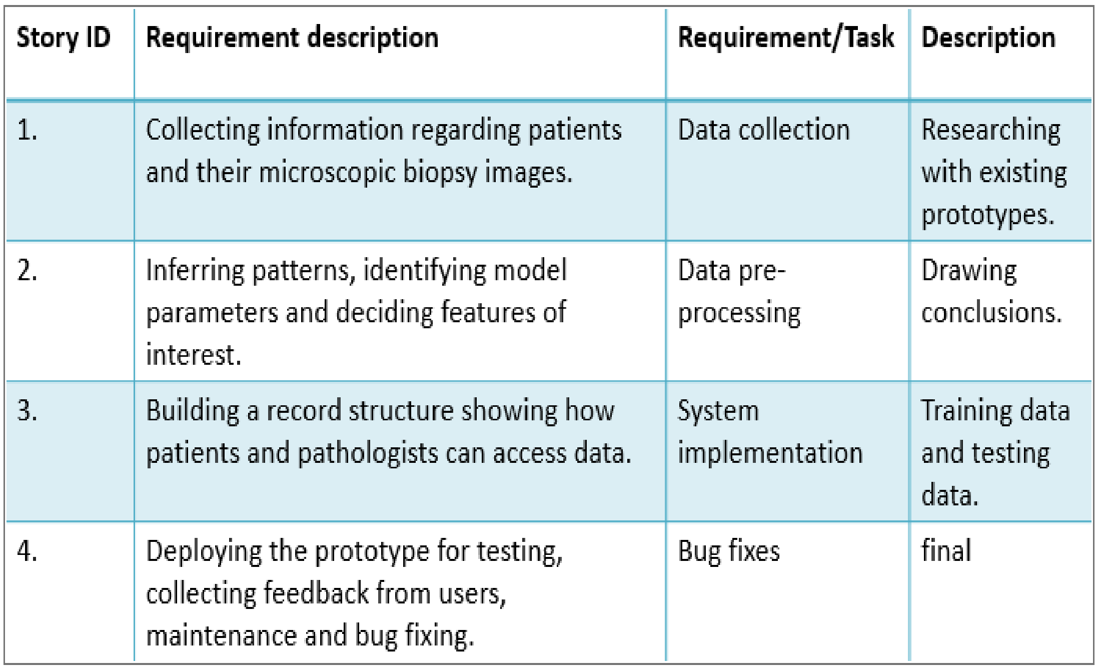
1.3 Objectives

- This project aims to allow using the high-resolution histopathological images as input to existing CNN, avoiding adaptation of models that can lead more complex and computationally costly architecture.
- Predicting if the cancer diagnosis is benign or malignant based on several observations/features.

1.4 Methodology

- Collecting information regarding patients and their microscopic biopsy images.
- Inferring patterns, identifying model parameters and deciding features of interest.
- Building a record structure showing how patients and pathologists can access data.
- Deploying the prototype for testing, collecting feedback from users, maintenance and bug fixing.

Agile Methodology



Story ID	Requirement description	Requirement/Task	Description
1.	Collecting information regarding patients and their microscopic biopsy images.	Data collection	Researching with existing prototypes.
2.	Inferring patterns, identifying model parameters and deciding features of interest.	Data pre-processing	Drawing conclusions.
3.	Building a record structure showing how patients and pathologists can access data.	System implementation	Training data and testing data.
4.	Deploying the prototype for testing, collecting feedback from users, maintenance and bug fixing.	Bug fixes	final

1.5 Problem Statement

- Goal: To build a robust model using machine-learned features and Convolutional Neural Networks for the detection of breast cancer from histopathological images.
- Most of these recent works related to BC classification are focused on Whole-Slide Imaging (WSI).
- The broad adoption of WSI and other forms of digital pathology has been facing obstacles such as the high cost of implementing and operating the technology, insufficient productivity for high-volume clinical routines, intrinsic technology-related concerns.
- Deep CNN is used which avoids handling large training data and large training time by taking output of CNN as input.

CHAPTER 2

LITERATURE SURVEY

2.1 PAPER 1:

Title:" Deep Features for Breast Cancer Histopathological Image Classification"

In this work we presented an investigation of the use of DeCAF features for breast cancer recognition using the BreaKHis dataset. The large size of the BreaKHis dataset has given us the opportunity to compare, on the same dataset, CNN trained from scratch with (DeCAF) features repurposed from another CNN trained on natural images, which often is not possible with medical image datasets since they are too small. From the results we can observe that these features are a viable alternative for a fast creation of image recognition systems using deep learning, and this system can perform better than systems using visual feature descriptors. Compared with a CNN trained from scratch, DeCAF features present comparable recognition rates. Note that training a CNN specifically for the problem requires more complex and slower training schemes.

This result is important for the design of future classification based systems in computer-aided diagnosis, since it shows that deep learned features, even if obtained with a CNN trained on other types of images, are valuable. With this study we make one more step towards transfer learning for medical image analysis and CAD/CADx systems, where CNN trained on ImageNet enable the detection of nodules in medical images.

As future work, one direction is to improve the recognition accuracy of DeCAF features using patches. Further investigation on the size of the patches, as well as overlapping patches, can be beneficial to increase the accuracies obtained with DeCAF features. Another investigation that can produce good results is the combination of these features with other visual descriptors and task-specific CNNs, to exploit the complementarity of these approaches. In addition, a better investigation on feature and classifier selection could also improve performance.

2.2 PAPER 2:

Title: "Breast Cancer Classification using Capsule Network with Pre-processed Histology Images"

The present work is based on the classification of breast cancer using capsule net architecture. From this work, it is clear that the performance of the conventional architectures can be improved by data pre-processing and parameter tuning. The results show that this method can be used as an automated tool to assist doctors in disease diagnosis, which may lead to higher concentration in the treatment at early stages rather than diagnosis and can increase the cancer survival rate.

The traditional approach consists of visual descriptors or handcrafted features which evaluates six different feature sets and base classifier for the system and outputs combination that produces best results in validation set. Where in CNN several patches of original images is taken for a CNN architecture. The reported results show the higher recognition rate compared to traditional method. The aim of present work is classify breast cancer using deep learning so as it can be used as an automated tool for the pathologist for diagnosis.

In the proposed method, the histology images are fed as an input to the capsule network architecture and it consist of input layer, hidden layer and output layer. In fully connected network every neuron in one layer is connected to every other neuron to another layer which gives probability distribution. Present work is based on classification of breast cancer using capsule network and performance can be improved by improving pre-processing and parameter tuning.

2.3 PAPER 3:

Title: “Breast cancer diagnosis from biopsy images with highly reliable random subspace classifier ensembles”

In this paper, a reliable classification scheme based on serial fusion of Random Subspace ensembles has been proposed for the classification of microscopic biopsy images for breast cancer diagnosis. Rather than simply pursuing classification accuracy, we emphasized the importance of a reject option in order to minimize the cost of misclassifications so as to ensure high classification reliability. The proposed two-stage method used a serial approach where the second classifier ensemble is only responsible for the patterns rejected by the first classifier ensemble. The first stage ensemble consists of binary SVMs which were trained in parallel, while the second ensemble comprises MLPs. During classification, the cascade of classifier ensembles received randomly sampled subsets of features following the Random Subspace procedure. For both of the ensembles the rejection option was implemented by relating the consensus degree from majority voting to a confidence measure and abstaining to classify ambiguous samples if the consensus degree was lower than the threshold.

The effectiveness of the proposed cascade classification scheme was verified on a breast cancer biopsy image dataset. The combined feature representation from LBP texture description, Gray Level Co-occurrence Matrix and Curvelet Transform exploits the complementary strengths of different feature extractors; the combined feature was proved efficient with respect to the biopsy image classification task. The two-stage ensemble cascade classification scheme obtained a high classification accuracy (99.25 %) and simultaneously guaranteed a high classification reliability (97.65 %) with a small rejection rate (1.94 %). Moreover, the cascade architecture provides a mechanism to balance between classification accuracy and rejection rate. By adjusting the rejection threshold in each ensemble, the classification accuracy and reliability of the system can be modulated to a certain degree according to the specification of specific applications. For example, medical diagnosis tasks usually require high accuracy and reliability, therefore the rejection thresholds in each stage will be set to a high level in order to guarantee the correctness of the diagnosis.

Although the proposed system has shown promising results with respect to the biopsy image classification task, there are still some aspects that need to be further investigated. The benchmark images used in this work were cropped from the original biopsy scans and only cover the important areas of the scans. However, often it is difficult to find Regions of Interest (ROIs) that contain the most important tissues in biopsy scans, more efforts therefore need to be put into detecting ROIs from biopsy images. In this paper, the parameters for the cascade system, such as ensemble size and rejection threshold, were decided empirically; this may not have produced the most satisfactory performance with respect to all application contexts. Therefore, some self-adaptive rules or algorithms for automatically optimizing these parameters would be desirable.

2.4 PAPER 4:

Title: "Breast Cancer Histopathological Image Classification Using Convolutional Neural Networks"

In this paper, we have presented a set of experiments conducted on the BreaKHis dataset using a deep learning approach to avoid handcrafted features. We have shown that we could use an existing CNN architecture, in our case AlexNet, that has been designed for classifying color images of objects, and adapt it to the classification of BC histopathological images.

We have also proposed several strategies for training the CNN architectures, based on the extraction of patches obtained randomly or by sliding a window mechanism, that allow to deal with the high-resolution of these textured images without changing the CNN architectures designed for low resolution images. Our experimental results obtained on the BreaKHis dataset showed improved accuracy obtained by CNN when compared to traditional machine learning models trained on the same dataset but with state-of-the-art texture descriptors. Future work can explore different CNN architectures and the optimization of the hyperparameters. Also, strategies to select representative patches in order to improve the accuracy can be explored.

Classification of histopathology images into distinct histopathology patterns corresponding to the non-cancerous or cancerous condition of the analysed tissue is the primary goal. Main challenges of this system is dealing with inherent complexity of histopathological images. CNN has been widely used to achieve results in different pattern recognition problems. For images of microscopic and macroscopic textures CNN is able to surpass traditional textural descriptors.

Besides assessing different CNN architectures, we also use different methods to deal with high resolution texture images without changing CNN architecture used for low resolution images. The goal is to preserve the original tissue structures and molecular composition allowing to observe it in a light microscope. CNN has achieved success in image classification problem including medical image analysis. CNN consists of multiple trainable stages stacked on top of each other followed by supervised classifier and feature maps. Feature map is a 2D array storing a color channel of input image. Output consists of a set array where each feature map represents a particular feature extracted at location of the input.

2.5 PAPER 5:

Title: “Computer-aided diagnosis of breast cancer based on fine needle biopsy microscopic images”

In this study, four different clustering algorithms for nuclei segmentation were compared. The methods were applied in a medical decision support system and were tested in terms of the classification accuracy on real routine medical data acquired from the Regional Hospital in Zielona Góra. The main question to be answered was whether relatively simple but fast methods can be successfully applied in computer-aided diagnosis. The results reaching approximately 96–100% are very optimistic and provethe approach not only to be fast and relatively easy to implement but also to provide accurate medical information for a cytologist. More sophisticated segmentation methods might be disadvantageous considering slight or no resulting improvement along with the large increase in complexity and computational requirements. However, the presented methods often fail to appropriately identify overlapping nuclei. Additionally, if a given image is composed of very few nuclei, then the clustering might give incorrect results. Given the above, it is important to provide an adequate number of images (in the paper, the authors used 9 images per patient) to achieve a good detection accuracy.

The clustering algorithm selection impact on the final classification result was also verified. Although visual inspection showed some discrepancies in the quality of the segmentation results, the classification accuracy has not confirmed these differences.

In the future, this research is going to be directed toward a possible application of the presented approach to virtual slides (VS). VS are images that have extremely high resolution (9 gigapixels and more) and contain information on the whole slide. Moreover, there are plans to improve the accuracy for the three-class problem, where apart from benign and malignant cases, there is also a fibroadenoma case. Fibroadenoma is a benign tumor that can have some properties that are similar to a malignant tumor. However, such a task could require a more sophisticated classification approach.

2.6 PAPER 6:

Title: “Deep Features for Breast Cancer Histopathological Image Classification”

In this work we presented an investigation of the use of DeCAF features for breast cancer recognition using the BreaKHis dataset. The large size of the BreaKHis dataset has given us the opportunity to compare, on the same dataset, CNN trained from scratch with (DeCAF) features repurposed from another CNN trained on natural images, which often is not possible with medical image datasets since they are too small. From the results we can observe that these features are a viable alternative for a fast creation of image recognition systems using deep learning, and this system can perform better than systems using visual feature descriptors. Compared with a CNN trained from scratch, DeCAF features present comparable recognition rates. Note that training a CNN specifically for the problem requires more complex and slower training schemes.

This result is important for the design of future classification based systems in computer-aided diagnosis, since it shows that deep learned features, even if obtained with a CNN trained on other types of images, are valuable. With this study we make one more step towards transfer learning for medical image analysis and CAD/CADx systems, where CNN trained on ImageNet enable the detection of nodules in medical images.

As future work, one direction is to improve the recognition accuracy of DeCAF features using patches. Further investigation on the size of the patches, as well as overlapping patches, can be beneficial to increase the accuracies obtained with DeCAF features. Another investigation that can produce good results is the combination of these features with other visual descriptors and task-specific CNNs, to exploit the complementarity of these approaches. In addition, a better investigation on feature and classifier selection could also improve performance.

2.7 PAPER 7:

Title: “Deep Learning for Magnification Independent Breast Cancer Histopathology Image Classification”

In this work, we have proposed a general framework based on CNNs for learning breast cancer histopathology image features. The proposed framework is independent from microscopy magnification and faster than previous methods as it requires single training. Speed and magnification independence properties are achieved without sacrificing the state-of-the-art performance. Magnification independent models are scalable, new training images from any magnification level could be utilized and trained models could easily be tuned (fine-tuning) by introducing new samples.

In this work, we have also proposed a multi-task CNN architecture to predict both the image magnification level and its benign/malignancy property simultaneously. The proposed model allows combining image data from many more resolution levels than four discrete magnification levels. In fact, magnification level prediction could be formulated as a regression problem which is not limited to a discrete set of levels. Multi-task prediction requires essentially no additional computation over single-task prediction. Besides, experimental result shows that classification performance does not degrade in multi-task network.

For the future work, stain normalization, deeper architectures, and splitting the network before the last fully-connected layer could be investigated. It would be interesting to observe task-wise early stopping in multi-task architecture. More importantly, additional data with increased number of patients should be introduced. We believe CNNs are more promising in breast cancer histopathology image classification than handcrafted features and the data is the key issue to obtain more robust models.

CHAPTER 3

SYSTEM REQUIREMENTS SPECIFICATION

Software requirement Specification is a fundamental document, which forms the foundation of the software development process. It not only lists the requirements of a system but also has a description of its major feature. An SRS is basically an organization's understanding (in writing) of a customer or potential client's system requirements and dependencies at a particular point in time (usually) prior to any actual design or development work. It's a two-way insurance policy that assures that both the client and the organization understand the other's requirements from that perspective at a given point in time.

The SRS also functions as a blueprint for completing a project with as little cost growth as possible. The SRS is often referred to as the "parent" document because all subsequent project management documents, such as design specifications, statements of work, software architecture specifications, testing and validation plans, and documentation plans, are related to it. It is important to note that an SRS contains functional and non-functional requirements only; it doesn't offer design suggestions, possible solutions to technology or business issues, or any other information other than what the development team understands the customer's system requirements to be.

The SRS functions as a blueprint for completing a project. The goal of preparing the SRS document is to:

- Facilitate communication between the customer, analyst, system developers, maintainers.
- To form a foundation for the design phase.
- Support system testing facilities.
- Controlling the evolution of the system

3.1 Functional Requirement

Functional Requirement defines a function of software system and how system must behave when presented with specific inputs or conditions. These may include calculations, data, manipulation and processing and other specific functionality. In this system following are the functional requirement: -

- Training dataset must be loaded

3.2 Non-functional Requirement

Non-functional requirements are the requirements which are not directly concerned with the specific function delivered by the system. They specify the criteria that can be used to judge the operation of a system rather than specific behaviours. They may relate to emergent system properties. Non-functional requirements for this system are specified as follows:

- Responsiveness of the system needs to be appropriate since timely retrieval of sensitive health data is essential.
- The software utilised should be portable so that medical institutions can easily expand to their inter-connected hospitals, spread across locations.
- Privacy of sensitive data should always be maintained and must not be misused in any manner.
- Researchers requesting for data must be from authorized sources. Proof of consent is a responsibility that is borne by the medical institution where the health data is generated.

3.1 Hardware Requirements

- AWS p2. xlarge instance with configuration: 4 processor cores
- 60GB RAM
- Nvidia K80 (24GB) graphics card
- 20GB hard disk space for training
- Testing can be done on PC: 8GB RAM
- Intel i7 processor
- 10GB hard disk space

3.2 Software Requirement

3.2.1 Google Collab

Google Colab is a free cloud service and now it supports free GPU. Improves Python programming language coding skills. Not only is this a great tool for improving coding skills, but it also allows to develop deep learning applications using libraries. Collaboratory, or “Collab” for short, allows you to write and execute Python in your browser, with zero configuration required, free access to GPUs and easy sharing.

CHAPTER 4

SYSTEM ANALYSIS AND DESIGN

4.1 Overview

Analysis is the process of finding the best solution to the problem. System analysis is the process by which we learn about the existing problems, define objects and requirements and evaluates the solutions. It is the way of thinking about the organization and the problem it involves, a set of technologies that helps in solving these problems. Feasibility study plays an important role in system analysis which gives the target for design and development.

4.2 Feasibility Study

All systems are feasible when provided with unlimited resource and infinite time. But unfortunately, this condition does not prevail in practical world. So, it is both necessary and prudent to evaluate the feasibility of the system at the earliest possible time. Months or years of effort, thousands of rupees and untold professional embarrassment can be averted if an ill-conceived system is recognized early in the definition phase. Feasibility & risk analysis are related in many ways. If project risk is great, the feasibility of producing quality software is reduced.

4.2.1 Performance Analysis

Analyse Project Performance is the process of comparing actual project cost and schedule performance to the performance measurement baseline for the purposes of analysing the current status of a project.

For the complete functionality of the project work, performance analysis of various algorithms is included. The algorithms are evaluated in terms of its accuracy, error, and computational time over a wide range of datasets.

4.2.2 Technical Analysis

System is only beneficial only if it can be turned into information systems that will meet the organization's technical requirement. Simply stated this test of feasibility asks whether the system will work or not when developed & installed, whether there are any major barriers to implementation. Regarding all these issues in technical analysis there are several points to focus on: -

Changes to bring in the system: All changes should be in positive direction, there will be an increased level of efficiency and better customer service.

Required skills: Platforms & tools used in this project are widely used. So, the skilled manpower is readily available in the industry.

Acceptability: The structure of the system is kept feasible enough so that there should not be any problem from the user's point of view.

Summary

The main aim of this chapter is to find out whether the system is feasible enough or not. For these reasons' different kinds of analysis, such as performance analysis, technical analysis etc. is performed.

4.3 Use-Case Diagram

A use case diagram is usually simple. It does not show the detail of the use cases:

- It only summarizes some of the relationships between use cases, actors, and systems.
- It does not show the order in which steps are performed to achieve the goals of each use case.

The use-case diagram corresponding to the project is depicted in Fig 4.3

- ▶ There are five users

- Patient
- Pathologist
- Doctor
- Oncologist
- Breast cancer system

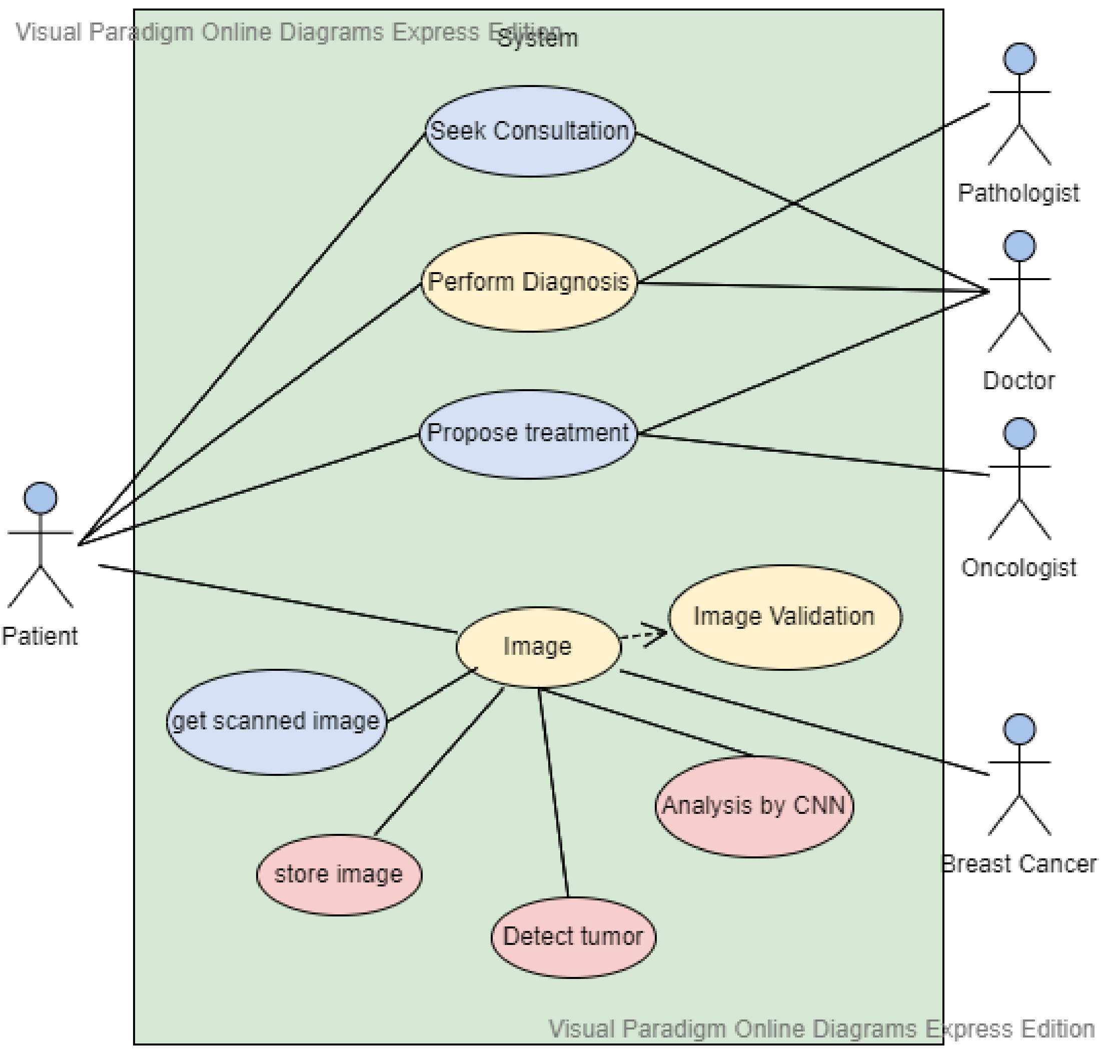


Fig 4.3 Use-Case Diagram

4.4 Class Diagram

A UML class diagram is made up of:

- A set of classes and
- A set of relationships between classes

The class diagram corresponding to the project is depicted in Fig 4.4.

The class diagram has the following classes:

- Pathologist: This is user interface class which has function for invoking train and predict.
- Feature extraction: This class has functions for extracting the features from the dataset and making it suitable for Random Forest.
- Random Forest: This function predicts whether it is Benign or Malignant by taking output of the feature extraction as its input.

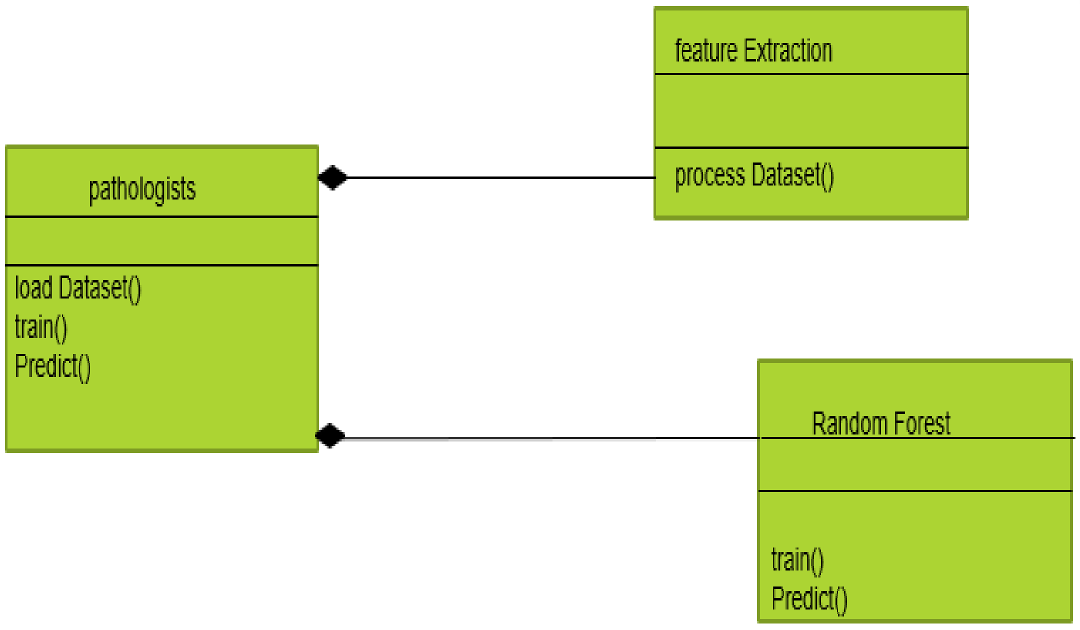


Fig 4.4 Class Diagram

4.5 Sequence Diagram

- As shown in the sequence diagram for training flow, Pathologist invokes train and trains a Random Forest model based on the training dataset.

- AS shown in the sequence diagram for Prediction, Random Forest invokes predict whether it is Benign or Malignant.

The sequence diagram corresponding to the project is depicted in Fig 4.5.

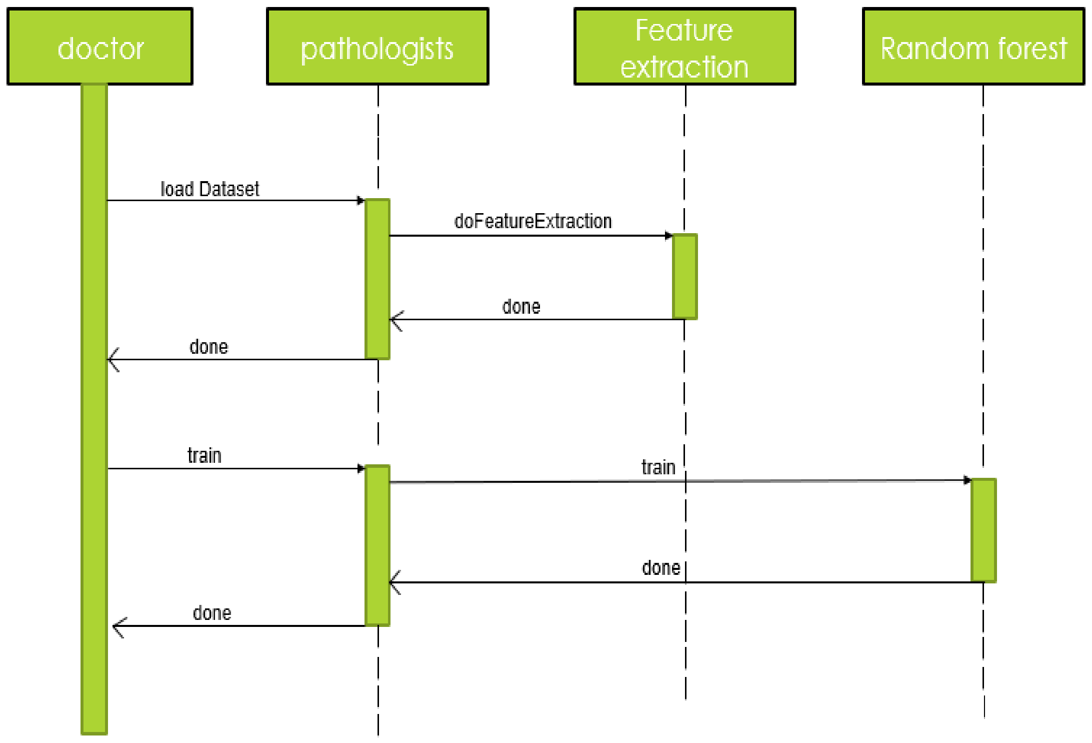


Fig 4.5 Sequence diagram for training flow

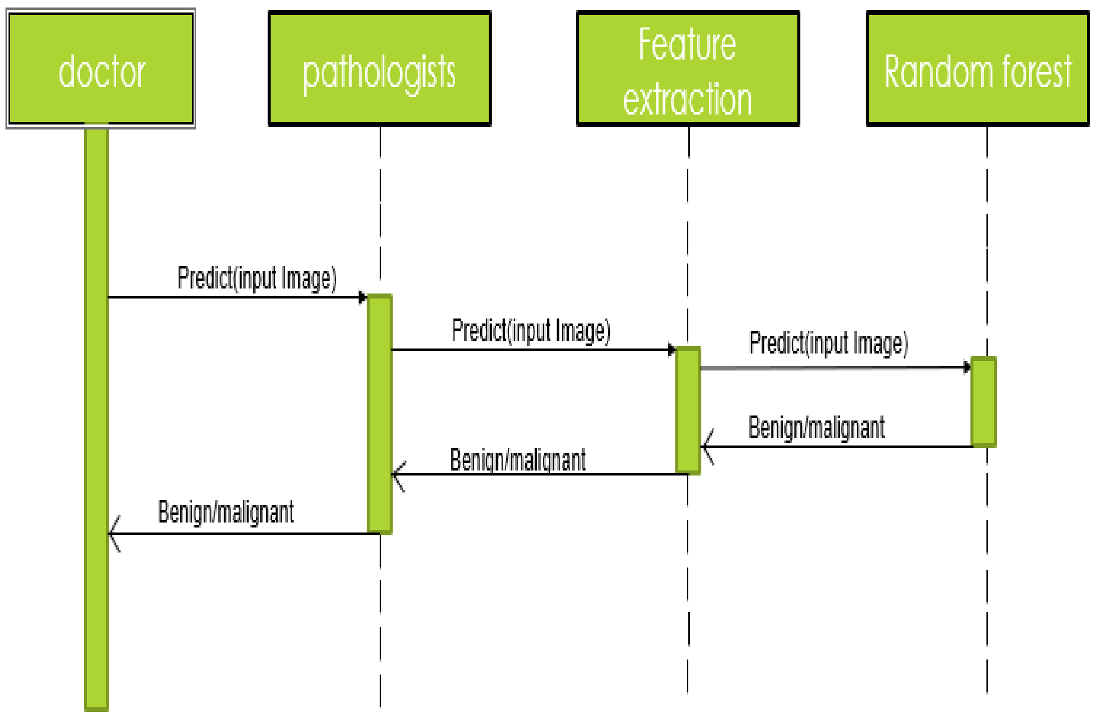


Fig 4.6 Sequence diagram for prediction

CHAPTER 5

IMPLEMENTATION

The implementation phase of the project is where the detailed design is actually transformed into working code. Aim of the phase is to translate the design into a best possible solution in a suitable programming language. This chapter covers the implementation aspects of the project, giving details of the programming language and development environment used. It also gives an overview of the core modules of the project with their step by step flow.

The implementation stage requires the following tasks.

- Careful planning.
- Investigation of the system and constraints.
- Design of methods to achieve the changeover.
- Evaluation of the changeover method.
- Correct decisions regarding selection of the platform
- Appropriate selection of the language for application development

5.1 VGG16 Architecture

- VGG-16 is a convolutional neural network architecture, its name VGG-16 comes from the fact that it has 16 layers.
- Its layers consist of Convolutional layers, Max Pooling layers, Fully connected layers.
- VGG-16 is used to extract the features from mammograms and selecting optimal features.
- Convolutional layers consists of filters (to extract significant features) and padding:
- Filters will be of 2 types: 1. horizontal filters 2. vertical filters.

- Padding: adding extra pixels to retain its quality.

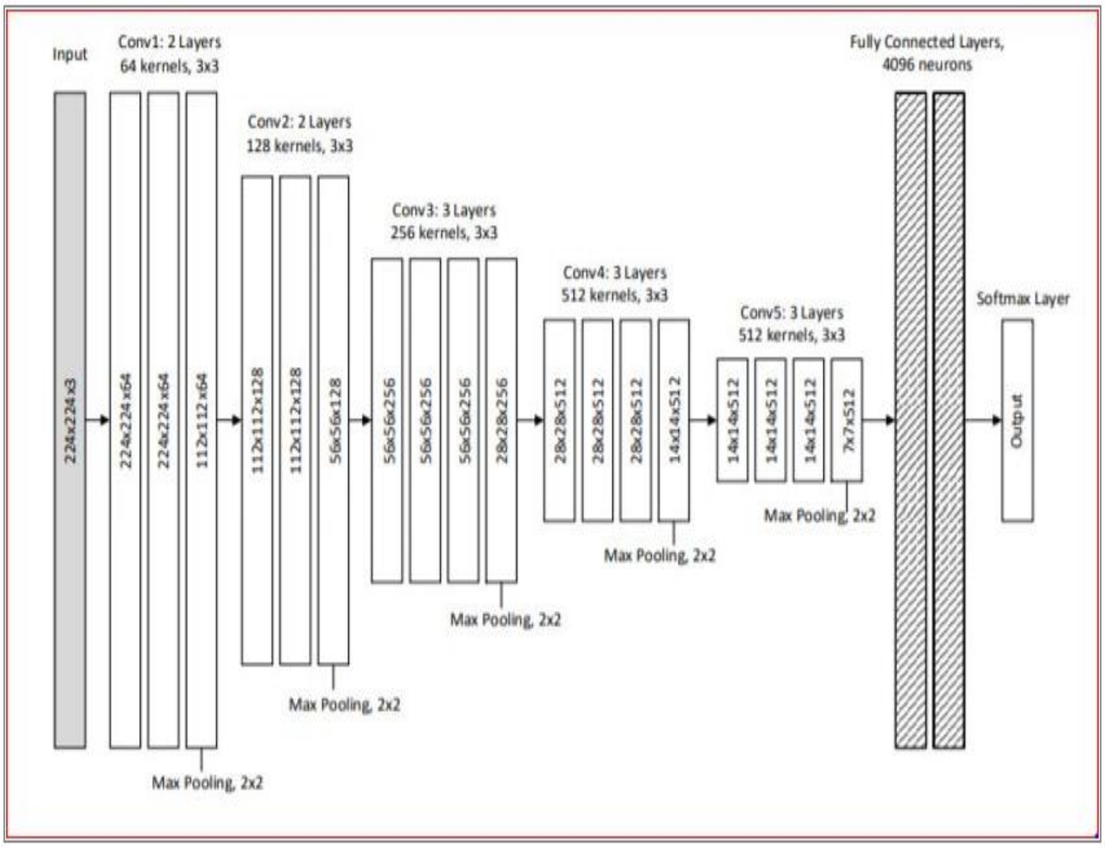


Fig 5.1: VGG16 Structure

VGG16 is a convolutional neural network model proposed by K. Simonyan and A. Zisserman from the University of Oxford in the paper “Very Deep Convolutional Networks for Large-Scale Image Recognition”. The model achieves 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes. It was one of the famous model submitted to ILSVRC-2014. It makes the improvement over AlexNet by replacing large kernel-sized filters (11 and 5 in the first and second convolutional layer, respectively) with multiple 3×3 kernel-sized filters one after another. VGG16 was trained for weeks and was using NVIDIA Titan Black GPU’s.

The input to cov1 layer is of fixed size 224 x 224 RGB image. The image is passed through a stack of convolutional (conv.) layers, where the filters were used with a very small receptive field: 3×3 (which is the smallest size to capture the notion of left/right, up/down, centre). In one of the configurations, it also utilizes 1×1 convolution filters, which can be seen as a linear transformation of the input channels (followed by non-linearity). The convolution stride is fixed to 1 pixel; the spatial padding of conv. layer input is such that the spatial resolution is preserved after convolution, i.e. the padding is 1-pixel for 3×3 conv. layers. Spatial pooling is carried out by five max-pooling layers, which follow some of the conv. layers (not all the conv. layers are followed by max-pooling). Max-pooling is performed over a 2×2-pixel window, with stride 2.

Three Fully-Connected (FC) layers follow a stack of convolutional layers (which has a different depth in different architectures): the first two have 4096 channels each, the third performs 1000-way ILSVRC classification and thus contains 1000 channels (one for each class). The final layer is the soft-max layer. The configuration of the fully connected layers is the same in all networks.

5.2 Convolutional Layer

The convolutional layer is the core building block of a CNN. Convolutional layers are the layers where filters are applied to the original image, or to other feature maps in a deep CNN. This is where most of the user-specified parameters are in the network. Convolutional Neural Networks (CNN or ConvNet) are complex feed forward neural networks. CNNs are used for image classification and recognition because of its high accuracy.

- The process is a 2D convolution on the inputs.
- The “dot products” between weights and inputs are “integrated” across “channels”.
- Filter weights are shared across receptive fields. The filter has same number of layers as input volume channels, and output volume has same “depth” as the number of filters.

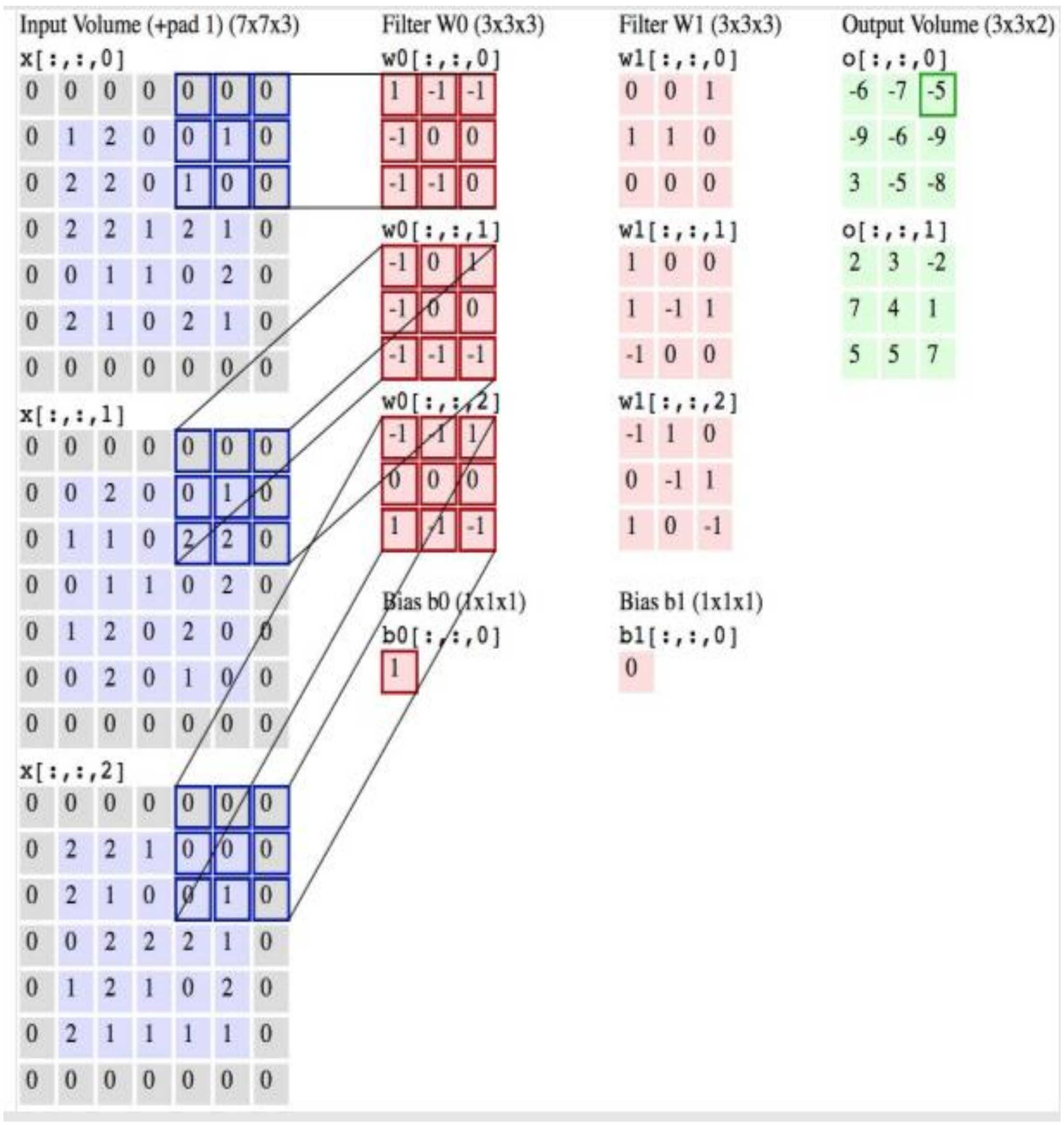


Fig 5.2: Convolutional Layer

5.3 Filters

A convolution is how the input is modified by a filter. In convolutional networks, multiple filters are taken to slice through the image and map them one by one and learn different portions of an input image. As a result, we see an image where only dark edges are emphasized.

As shown in the figure 5.3 and 5.4, the reading of the input matrix begins at the bottom right of image. Next the software selects a smaller matrix there, which is called a filter. Then the filter produces convolution that is moves along the input image. The filter's task is to multiply its value by the original pixel values. All these multiplications are summed up. One number is obtained in the end. After passing the filter across all positions, a matrix is obtained, but smaller than an input matrix.

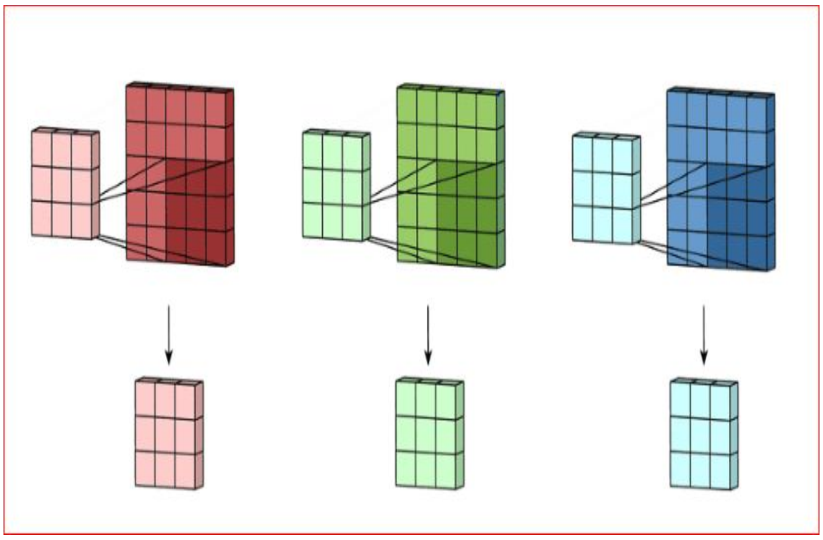
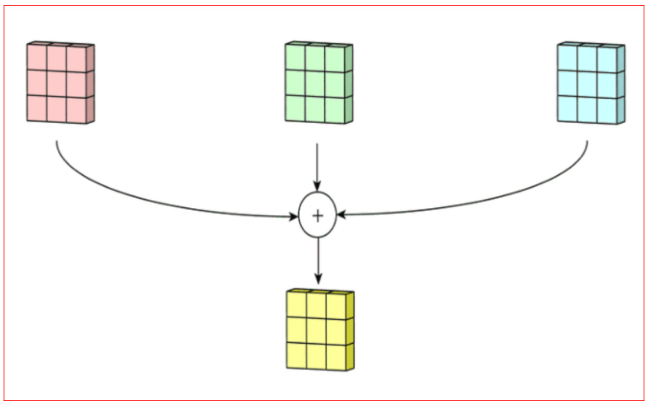



Fig 5.3: This is how filters work on RGB channels of a image

The network will consist of several convolutional network mixed with non linear and pooling layers. When the image passes through one convolution layer, the output of the first layer becomes the input for the second layer. And this happens with every further convolutional layer.

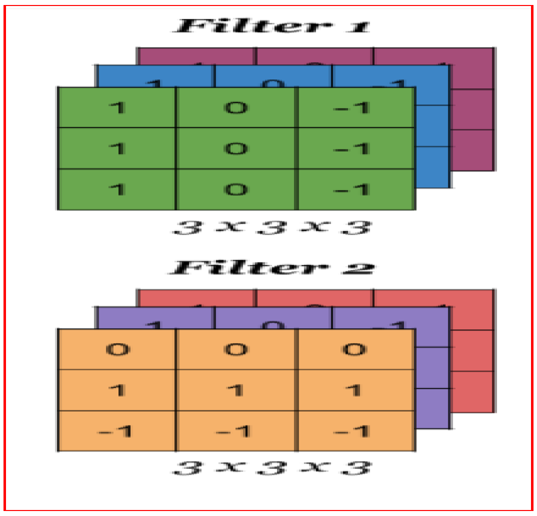


Fig 5.4: Shows filter used in convolutional layer

The non linear is added after each convolution operation. It has an activation function, which brings non linear property. Without this property a network would not be intense and will not be able to model the class label.

The pooling layer follows the non-linear layer. It works with width and height of the image and perform operation on them. As a result, the image volume is reduced. After completion of series of convolutional, non-linear and pooling layers, it is fully connected layer. The layer takes the output information from convolutional networks to the end of the network results in an N dimensional vector.

5.4 Pooling Layer

pooling layer is another building block of a CNN. Its function is to progressively reduce the spatial size of the representation to reduce the number of parameters and computation in the network. Pooling layer operates on each feature map independently. The most common approach used in pooling is max pooling.

Pooling layers provide an approach to down sampling feature maps by summarizing the presence of features in patches of the feature map. Two common pooling methods are average pooling and max pooling that summarize the average presence of a feature and the most activated presence of a feature respectively.

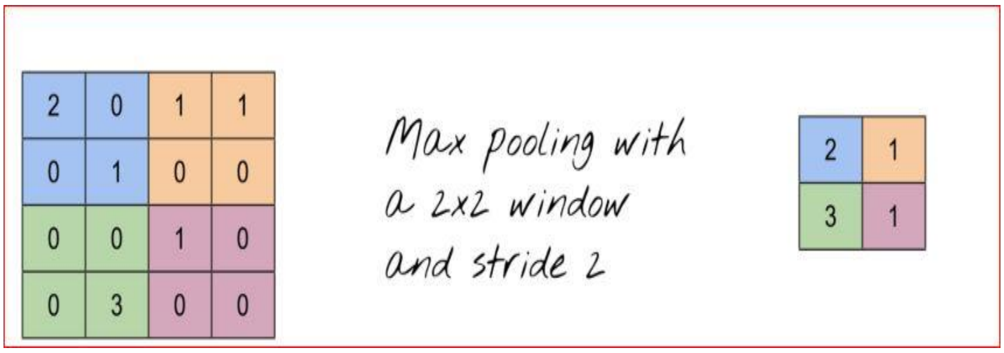


Fig 5.5: Shows filter used in convolutional layer

As shown in the figure 5.5, consider the max pooling with 2x2 window and stride 2, maximum value in that stride is 2, in the next stride, the maximum value is 1. Then the maximum value 3 is followed by 1.

By combining these maximum values, a new matrix is formed, which has reduced in the size of the presentation to reduce the number of parameters and computation in the network.

- Convolutional layers provide activation maps.
- Pooling layer applies non-linear down sampling on activation maps.
- Pooling is aggressive (discard info); the trend is to use smaller filter size and abandon pooling.

The addition of a pooling layer after the convolutional layer is a common pattern used for ordering layers within a convolutional neural network that may be repeated one or more times in a given model.

The pooling layer operates upon each feature map separately to create a new set of the same number of pooled feature maps.

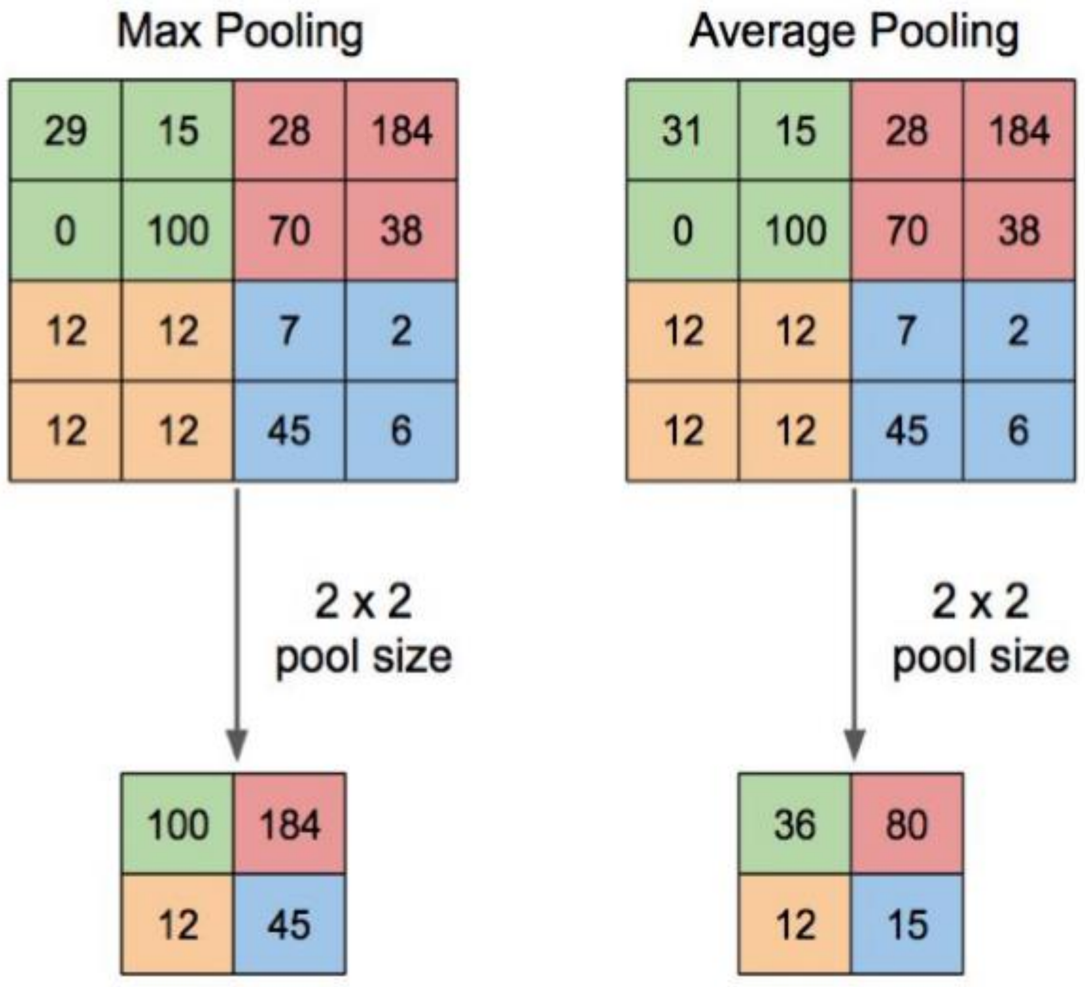


Fig 5.6: Pooling Layer

5.5 Feature Extraction

Feature extraction a type of dimensionality reduction that efficiently represents interesting parts of an image as a compact feature vector. This approach is useful when image sizes are large and a reduced feature representation is required to quickly complete tasks such as image matching and retrieval. Feature extraction describes the relevant shape information contained in a pattern so that the task of classifying the pattern is made easy by a formal procedure. In pattern recognition and in image processing, feature extraction is a special form of dimensionality reduction.

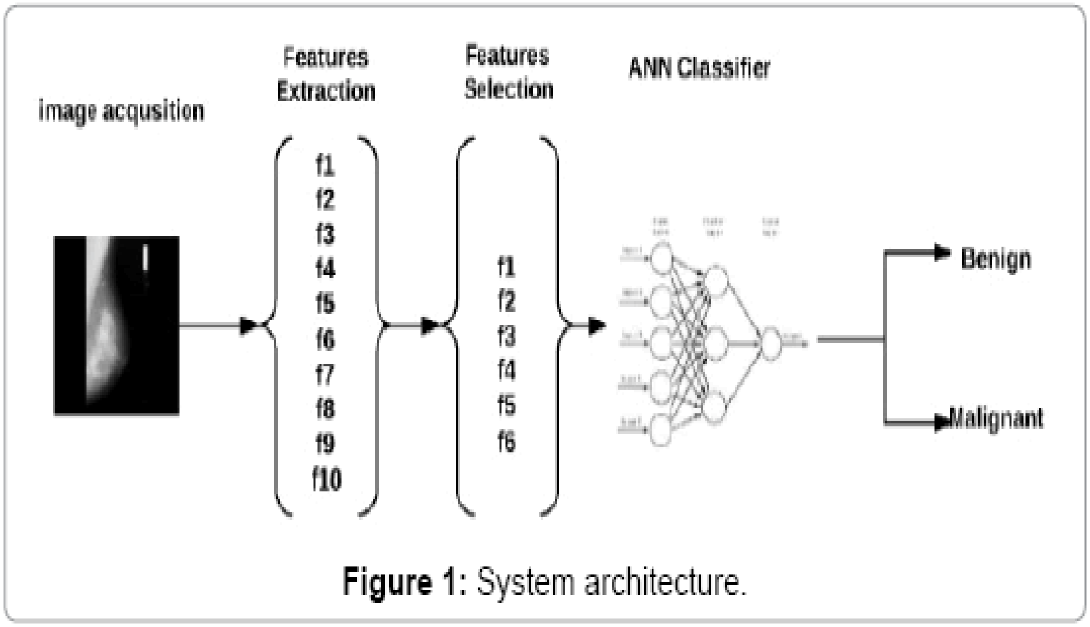


Fig 5.7 Feature Extraction

When performing deep learning feature extraction, we treat the pre-trained network as an arbitrary feature extractor, allowing the input image to propagate forward, stopping at pre-specified layer, and taking the outputs of that layer as our features.

As shown in the figure 5.7, when an image is selected, the features are extracted from the image. Then the information features are selected using which we can classify whether the tumour is benign or malignant.

5.8 Random Forest

Random forests, otherwise known as the random forest model, is a method for classification and other tasks. It operates from decision trees and outputs classification of the individual trees. The random forest is a classification algorithm consisting of many decisions' trees. It uses bagging and feature randomness when building each individual tree to try to create an uncorrelated forest of trees whose prediction by committee is more accurate than that of any individual tree.

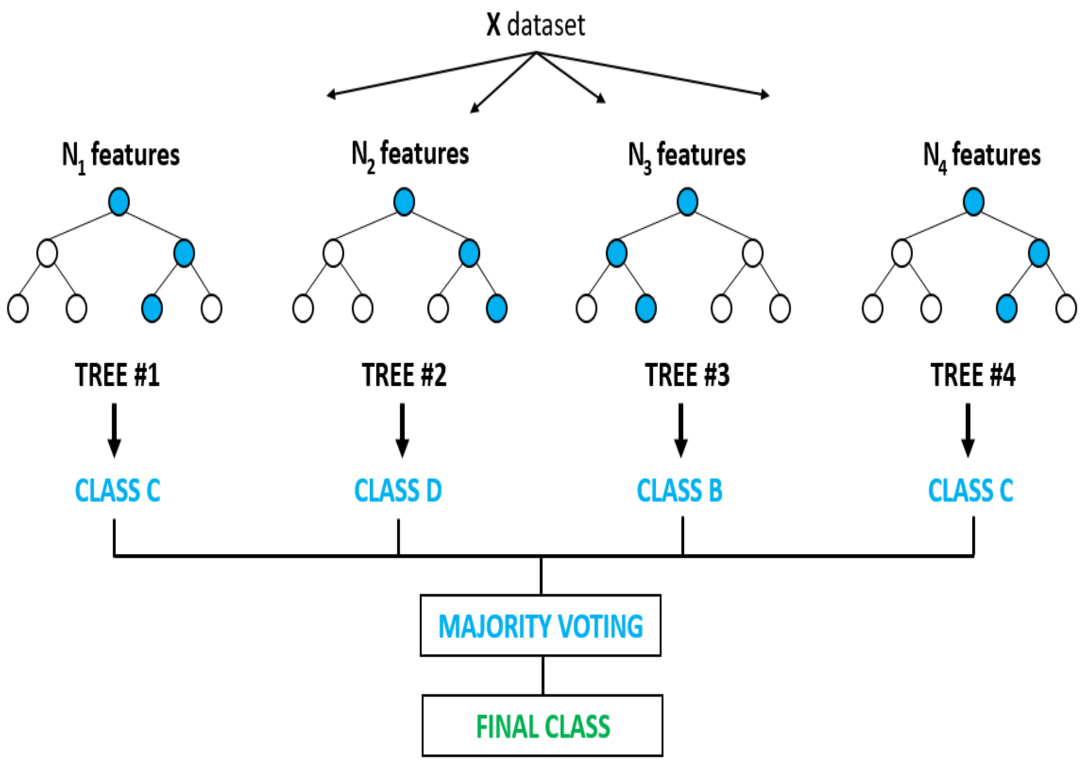


Fig 5.8: Random Forest

5.8.1 Features of Random Forests

- It is unexcelled in accuracy among current algorithms.
- It runs efficiently on large data bases.
- It can handle thousands of input variables without variable deletion.
- It gives estimates of what variables are important in the classification.
- It generates an internal unbiased estimate of the generalization error as the forest building progresses.
- It has an effective method for estimating missing data and maintains accuracy when a large proportion of the data are missing.
- It has methods for balancing error in class population unbalanced data sets.

CODE:

Using VGG16 to diagnose breast cancer detection

```
import pandas as pd
import numpy as np
import os
from google.colab import drive
import pickle
//importing python libraries that will be used later in the program

print(os.getcwd())
drive.mount('/content/gdrive')
root_path = '/content/gdrive/My Drive/project/Brest Cancer/'
```

Instantiate VGG16

```
import tensorflow as tf
# Import the VGG16 network architecture
//we use VGG16 network architecture for extracting features from the image

from keras.applications import VGG16;
//we are importing VGG16 from Keras application
```

VGG16 is a pre-trained model, it already has a network trained with some object detection.

```
IMG_WIDTH = 50
IMG_HEIGHT = 50
IMG_DEPTH = 3
BATCH_SIZE = 16
//we initialise dimension of image as 50x50x3 with batch of 16 images

# Instantiate the model with the pre-trained weights (no top)
//then we initialise  the model in conv_base
```

```
conv_base = VGG16(weights='./vgg16_weights_tf_dim_ordering_tf_kernels_notop.h
5',
                  include_top=False,
                  input_shape=(IMG_HEIGHT, IMG_WIDTH, IMG_DEPTH))

# Show the architecture
conv_base.summary()
//here we can see all the layers present in VGG16
```

Classification using Random Forest on extracted features

```
from sklearn.ensemble import RandomForestClassifier

tr = train_features.reshape(train_features.shape[0], -1)
clf = RandomForestClassifier(max_depth=5, n_estimators=5)
//storing RandomForest classifier in clf and setting the depthof forest as 5.

print(tr.shape)
//shape of train features

clf.fit(tr, y_train)
//training the training image features that we are extracting along wuth these
corresponding labels.

(8998, 512)
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                criterion='gini', max_depth=5, max_features='auto',
                  max_leaf_nodes=None, max_samples=None,
                  min_impurity_decrease=0.0, min_impurity_split=None,
                  min_samples_leaf=1, min_samples_split=2,
                    min_weight_fraction_leaf=0.0, n_estimators=5,
                    n_jobs=None, oob_score=False, random_state=None,
                    verbose=0, warm_start=False)
val = val_features.reshape(val_features.shape[0], -1)
// reshaping val_features and storing it in val.

clf.score(val, y_val)
//to check the accuracy of our previous training, i.e. how accurately this model will
predict if a person has cancer or not.
```

Classification using neural network

```
import tensorflow as tf

labs = list(set(y))
//storing all the labels in y as a list in labs.

alt_num_labs_train = np.asarray([labs.index(i) for i in y_train])

alt_num_labs_val = np.asarray([labs.index(i) for i in y_val])

model = tf.keras.models.Sequential([
  tf.keras.layers.Flatten(),
  tf.keras.layers.Dense(512, activation=tf.nn.relu),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.Dense(2, activation=tf.nn.softmax)
])

adm = tf.keras.optimizers.Adam(lr=0.001, beta_1=0.9, beta_2=0.999, decay=0.9, ams
grad=False)
//we are storing Adam optimizer in adm.

model.compile(optimizer=adm,
            loss='sparse_categorical_crossentropy',
            metrics=['accuracy'])
model.fit(train_features, alt_num_labs_train, epochs=15)
// training the model with model.fit
//we are taking epochs as 15, so our model will train for 15 times and tries to reach
the best prediction accuracy.

model.evaluate(val_features, alt_num_labs_val)

import tensorflow as tf

# Define the densely connected classifier
NB_TRAIN_SAMPLES = tr_flat.shape[0]
NB_VALIDATION_SAMPLES = tst_flat.shape[0]
NB_EPOCHS = 50
//we are storing all the flattened features of train features in tr_flat and all flatten
features from val_features in tst_flat.

model = models.Sequential()
model.add(layers.Dense(512, activation='relu', input_dim=(1*1*512)))
model.add(layers.Dense(2, activation=tf.nn.softmax))

model.compile(
   loss='binary_crossentropy',
    optimizer=optimizers.RMSprop(lr=2e-4),
    metrics=['acc'])
reduce_learning = callbacks.ReduceLROnPlateau(
    monitor='val_loss',
    factor=0.2,
    patience=2,
    verbose=1,
    mode='auto',
    epsilon=0.0001,
    cooldown=2,
    min_lr=0)

eary_stopping = callbacks.EarlyStopping(
    monitor='val_loss',
    min_delta=0,
    patience=7,
    verbose=1,
    mode='auto')

callbacks = [reduce_learning, eary_stopping]

# Train the model
history = model.fit(
    tr_flat,
    num_labs_train,
    epochs=NB_EPOCHS,
    validation_data=(tst_flat, num_labs_val),
//training the model

    callbacks=callbacks
)
import matplotlib.pyplot as plt

%matplotlib inline
acc = history.history['acc']
val_acc = history.history['val_acc']
loss = history.history['loss']
val_loss = history.history['val_loss']
epochs = range(1, len(acc) + 1)
```

```
//here we are graphically representing the prediction accuracy during training and
validation.

plt.title('Training and validation accuracy')
plt.plot(epochs, acc, 'red', label='Training acc')
plt.plot(epochs, val_acc, 'blue', label='Validation acc')
plt.legend()

plt.figure()

//here we are graphically representing the prediction loss during training and
validation.

plt.title('Training and validation loss')
plt.plot(epochs, loss, 'red', label='Training loss')
plt.plot(epochs, val_loss, 'blue', label='Validation loss')

plt.legend()

plt.show()
```

Classification on a specific image

```
#path = r'./Dataset/dataset_IDC/8863/0/8863_idx5_x1201_y1251_class0.png'
path = r'./8863_idx5_x1151_y1101_class1.png'
import cv2
image = cv2.imread(path)
//reading the image

image = image.reshape(1, image.shape[0], image.shape[1], image.shape[2])
_sp_features = conv_base.predict(np.array(image), batch_size=BATCH_SIZE, verbo
se=1)
```

```
class_lab = ["No Cancer", "Cancer"]
sp_features = _sp_features.reshape(-1, 512)
```

//extracting features using conv_base i.e. VGG16 architecture and storing it in sp_features.

```
lab = model.predict(sp_features)
```

//defining labels as no cancer or cancer

```
pred = [np.where(i == max(i))[0][0] for i in lab]
```

//we can see from the given input image that the tumor is benign or malignant

```
print("Diagnosis (predicted):", class_lab[pred[0]])
```

//printing the prediction, whether the tumor is cancerous or not.

CHAPTER 6

RESULTS AND DISCUSSION

- The overall accuracy for breast cancer diagnosis achieved equal to 86%.
- The dataset showed improved accuracy obtained by CNN when compared to traditional learning techniques.
- Deep CNN diagnosis provides a second option for image diagnosis which can improve the reliability of expert decision making.
- The following snapshots and graphs define the results or outputs that we will get after step by step execution of each proposed protocol.

SCREENSHOTS

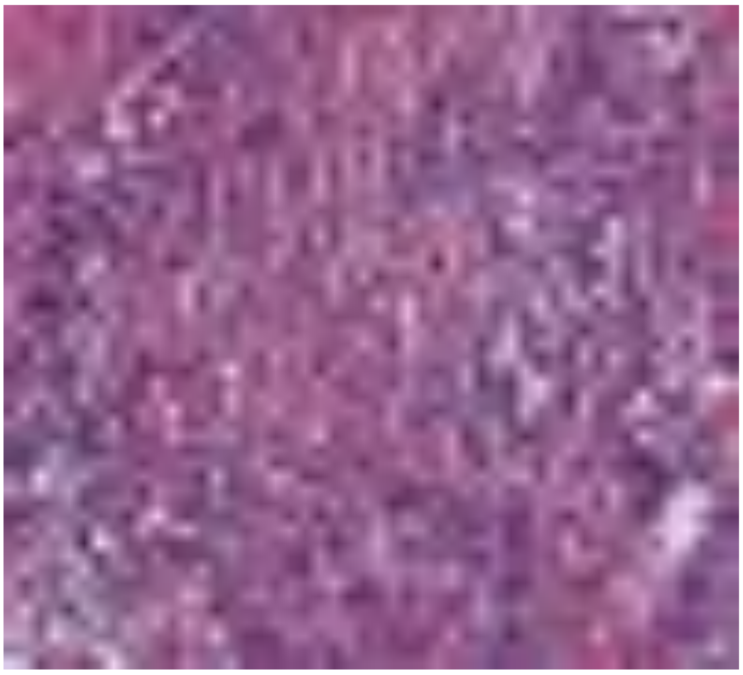
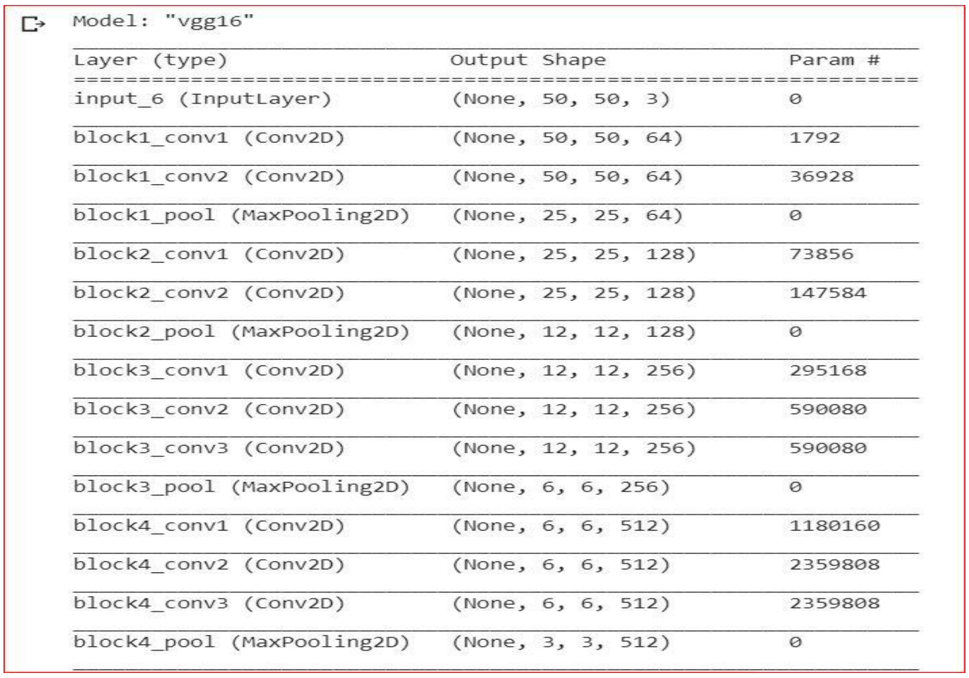


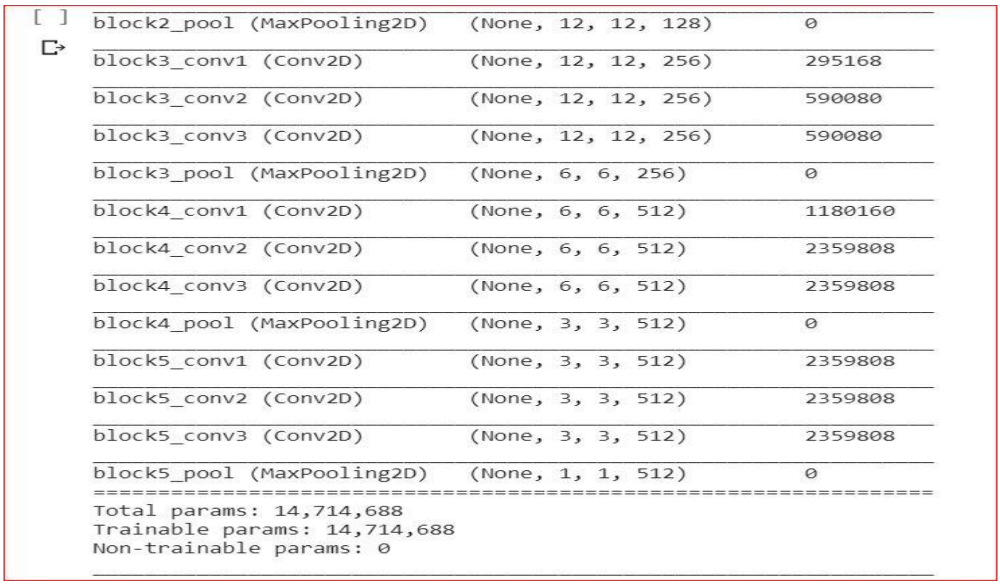
Fig 6.1: Input image

- Taking a 50X50 size image as an input image
- This image is the most affected part/patch of the original scanned image

- 

```
Model: "vgg16"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_6 (InputLayer)         (None, 50, 50, 3)         0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 50, 50, 64)        1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 50, 50, 64)        36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 25, 25, 64)        0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 25, 25, 128)       73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 25, 25, 128)       147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 12, 12, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 12, 12, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 12, 12, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 12, 12, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 6, 6, 256)         0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 6, 6, 512)         1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 6, 6, 512)         2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 6, 6, 512)         2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 3, 3, 512)         0
_________________________________________________________________
```

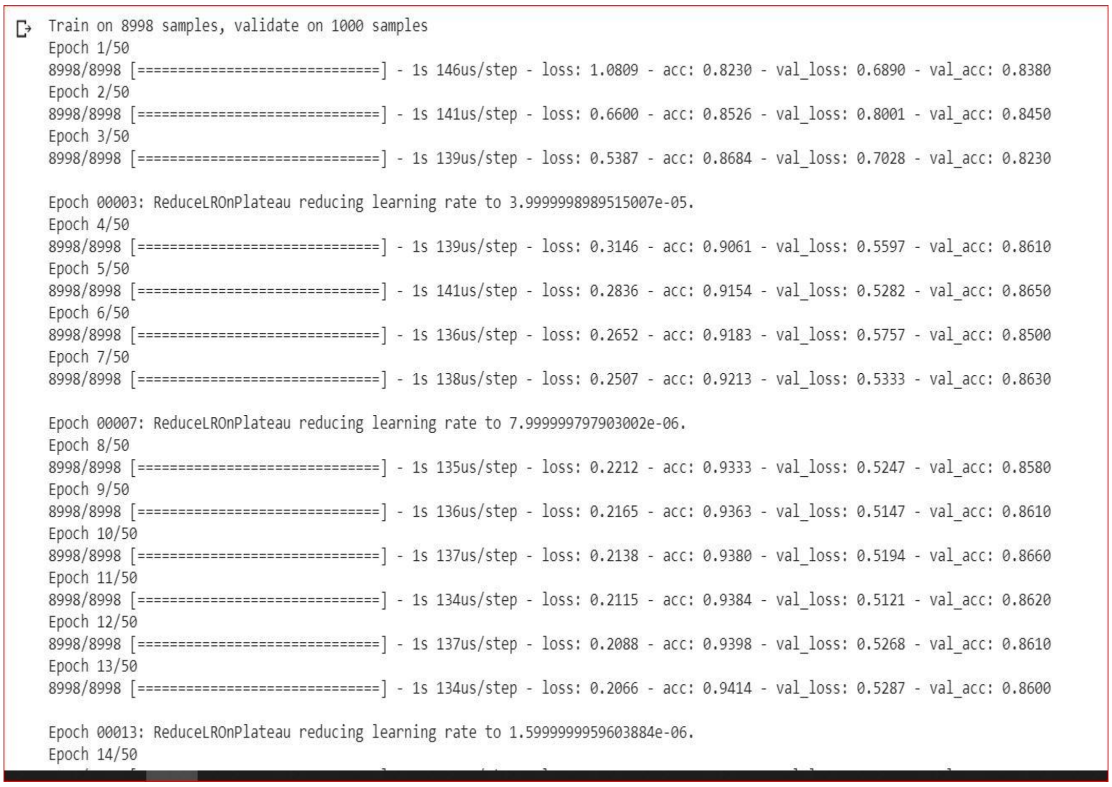
Fig 6.2: Output of vgg16 model



```
block2_pool (MaxPooling2D)   (None, 12, 12, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 12, 12, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 12, 12, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 12, 12, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 6, 6, 256)         0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 6, 6, 512)         1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 6, 6, 512)         2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 6, 6, 512)         2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 3, 3, 512)         0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 3, 3, 512)         2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 3, 3, 512)         2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 3, 3, 512)         2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 1, 1, 512)         0
=================================================================
Total params: 14,714,688
Trainable params: 14,714,688
Non-trainable params: 0
_________________________________________________________________
```

Fig 6.3: Output of vgg16 model (cont..)

- This image consists of various layers inside vgg16 architecture
- It shows how the image size is decreasing in each layer i.e., input layer, Convo2D layer, maxpooling layer



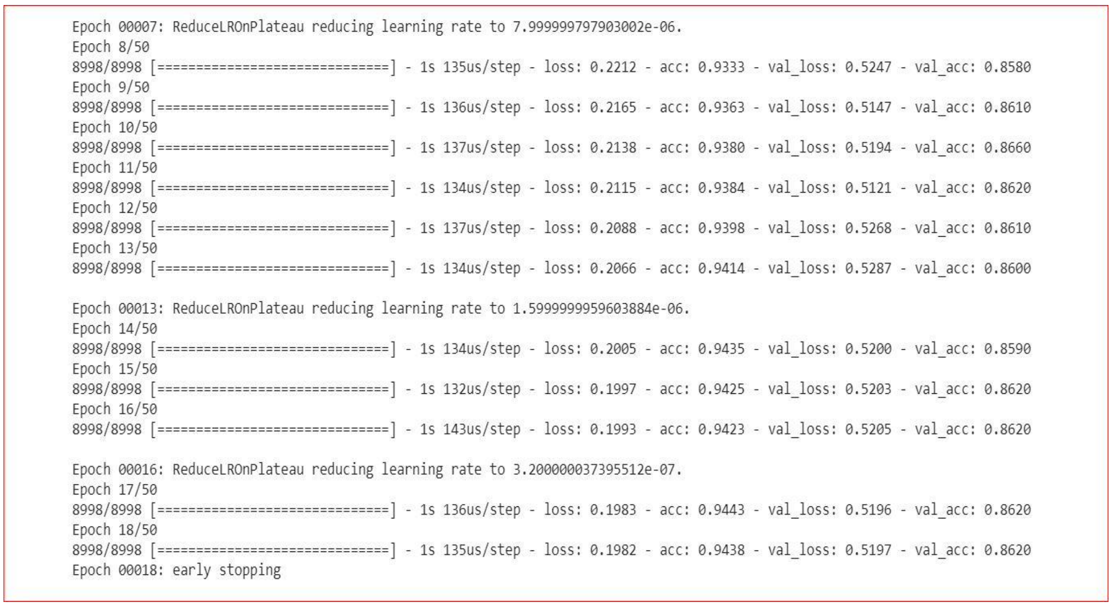
```
Train on 8998 samples, validate on 1000 samples
Epoch 1/50
8998/8998 [==============================] - 1s 146us/step - loss: 1.0809 - acc: 0.8230 - val_loss: 0.6890 - val_acc: 0.8380
Epoch 2/50
8998/8998 [==============================] - 1s 141us/step - loss: 0.6600 - acc: 0.8526 - val_loss: 0.8001 - val_acc: 0.8450
Epoch 3/50
8998/8998 [==============================] - 1s 139us/step - loss: 0.5387 - acc: 0.8684 - val_loss: 0.7028 - val_acc: 0.8230

Epoch 00003: ReduceLROnPlateau reducing learning rate to 3.9999998989515007e-05.
Epoch 4/50
8998/8998 [==============================] - 1s 139us/step - loss: 0.3146 - acc: 0.9061 - val_loss: 0.5597 - val_acc: 0.8610
Epoch 5/50
8998/8998 [==============================] - 1s 141us/step - loss: 0.2836 - acc: 0.9154 - val_loss: 0.5282 - val_acc: 0.8650
Epoch 6/50
8998/8998 [==============================] - 1s 136us/step - loss: 0.2652 - acc: 0.9183 - val_loss: 0.5757 - val_acc: 0.8500
Epoch 7/50
8998/8998 [==============================] - 1s 138us/step - loss: 0.2507 - acc: 0.9213 - val_loss: 0.5333 - val_acc: 0.8630

Epoch 00007: ReduceLROnPlateau reducing learning rate to 7.999999797903002e-06.
Epoch 8/50
8998/8998 [==============================] - 1s 135us/step - loss: 0.2212 - acc: 0.9333 - val_loss: 0.5247 - val_acc: 0.8580
Epoch 9/50
8998/8998 [==============================] - 1s 136us/step - loss: 0.2165 - acc: 0.9363 - val_loss: 0.5147 - val_acc: 0.8610
Epoch 10/50
8998/8998 [==============================] - 1s 137us/step - loss: 0.2138 - acc: 0.9380 - val_loss: 0.5194 - val_acc: 0.8660
Epoch 11/50
8998/8998 [==============================] - 1s 134us/step - loss: 0.2115 - acc: 0.9384 - val_loss: 0.5121 - val_acc: 0.8620
Epoch 12/50
8998/8998 [==============================] - 1s 137us/step - loss: 0.2088 - acc: 0.9398 - val_loss: 0.5268 - val_acc: 0.8610
Epoch 13/50
8998/8998 [==============================] - 1s 134us/step - loss: 0.2066 - acc: 0.9414 - val_loss: 0.5287 - val_acc: 0.8600

Epoch 00013: ReduceLROnPlateau reducing learning rate to 1.5999999959603884e-06.
Epoch 14/50
```

Fig 6.4: Accuracy achieved



```
Epoch 00007: ReduceLROnPlateau reducing learning rate to 7.999999797903002e-06.
Epoch 8/50
8998/8998 [==============================] - 1s 135us/step - loss: 0.2212 - acc: 0.9333 - val_loss: 0.5247 - val_acc: 0.8580
Epoch 9/50
8998/8998 [==============================] - 1s 136us/step - loss: 0.2165 - acc: 0.9363 - val_loss: 0.5147 - val_acc: 0.8610
Epoch 10/50
8998/8998 [==============================] - 1s 137us/step - loss: 0.2138 - acc: 0.9380 - val_loss: 0.5194 - val_acc: 0.8660
Epoch 11/50
8998/8998 [==============================] - 1s 134us/step - loss: 0.2115 - acc: 0.9384 - val_loss: 0.5121 - val_acc: 0.8620
Epoch 12/50
8998/8998 [==============================] - 1s 137us/step - loss: 0.2088 - acc: 0.9398 - val_loss: 0.5268 - val_acc: 0.8610
Epoch 13/50
8998/8998 [==============================] - 1s 134us/step - loss: 0.2066 - acc: 0.9414 - val_loss: 0.5287 - val_acc: 0.8600

Epoch 00013: ReduceLROnPlateau reducing learning rate to 1.5999999959603884e-06.
Epoch 14/50
8998/8998 [==============================] - 1s 134us/step - loss: 0.2005 - acc: 0.9435 - val_loss: 0.5200 - val_acc: 0.8590
Epoch 15/50
8998/8998 [==============================] - 1s 132us/step - loss: 0.1997 - acc: 0.9425 - val_loss: 0.5203 - val_acc: 0.8620
Epoch 16/50
8998/8998 [==============================] - 1s 143us/step - loss: 0.1993 - acc: 0.9423 - val_loss: 0.5205 - val_acc: 0.8620

Epoch 00016: ReduceLROnPlateau reducing learning rate to 3.200000037395512e-07.
Epoch 17/50
8998/8998 [==============================] - 1s 136us/step - loss: 0.1983 - acc: 0.9443 - val_loss: 0.5196 - val_acc: 0.8620
Epoch 18/50
8998/8998 [==============================] - 1s 135us/step - loss: 0.1982 - acc: 0.9438 - val_loss: 0.5197 - val_acc: 0.8620
Epoch 00018: early stopping
```

Fig 6.5: Accuracy achieved (cont...)

- The above figure consists of accuracy achieved for each of the epoch value

- For different epoch value, the accuracy achieved is different
- For epoch=18, the maximum accuracy achieved is 94%

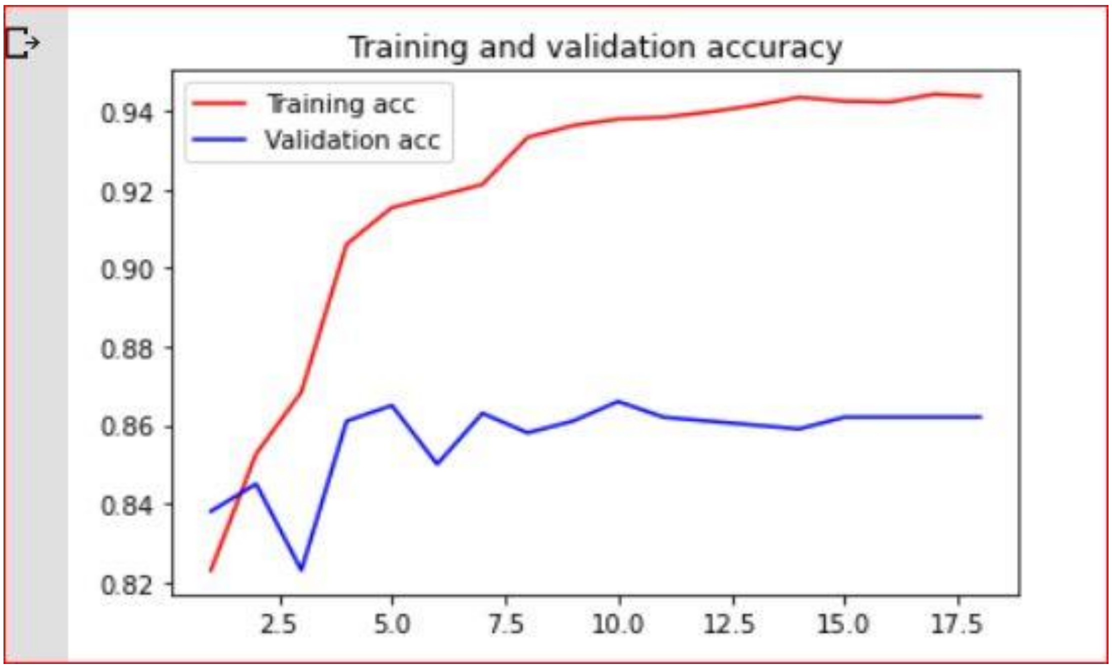


Fig 6.6: Accuracy graph

- The above graph is the accuracy graph obtained from the achieved accuracies
- As we can see, as the training accuracy at its peak, which shows the maximum result
- Once it becomes constant, the maximum result is taken

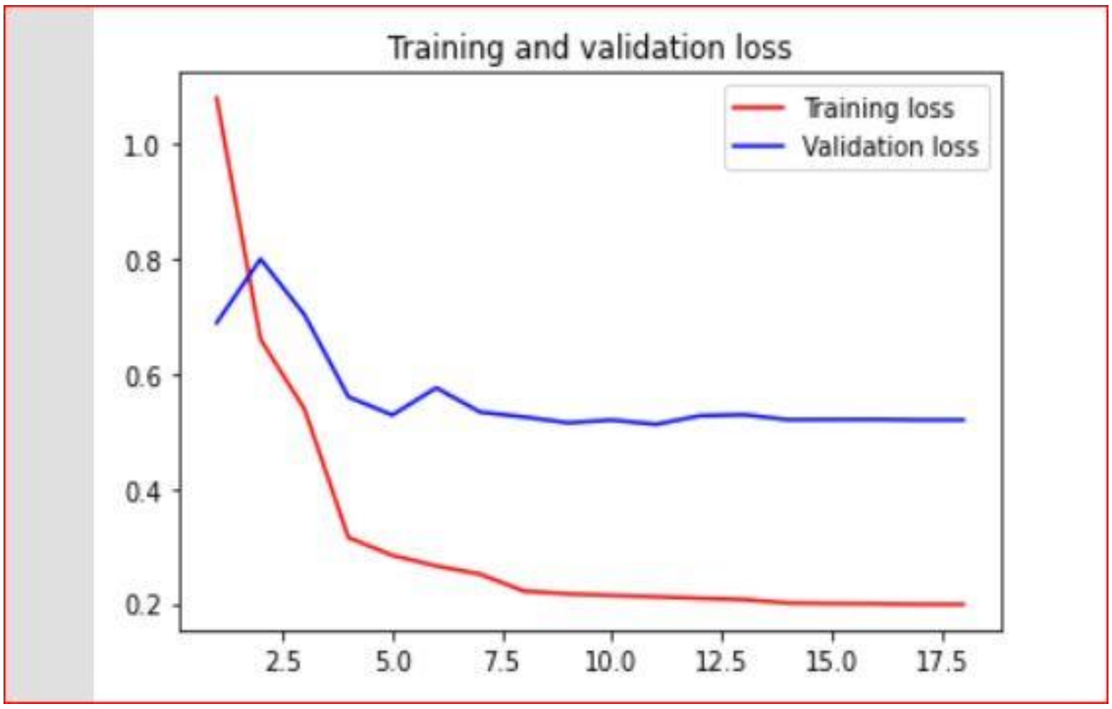
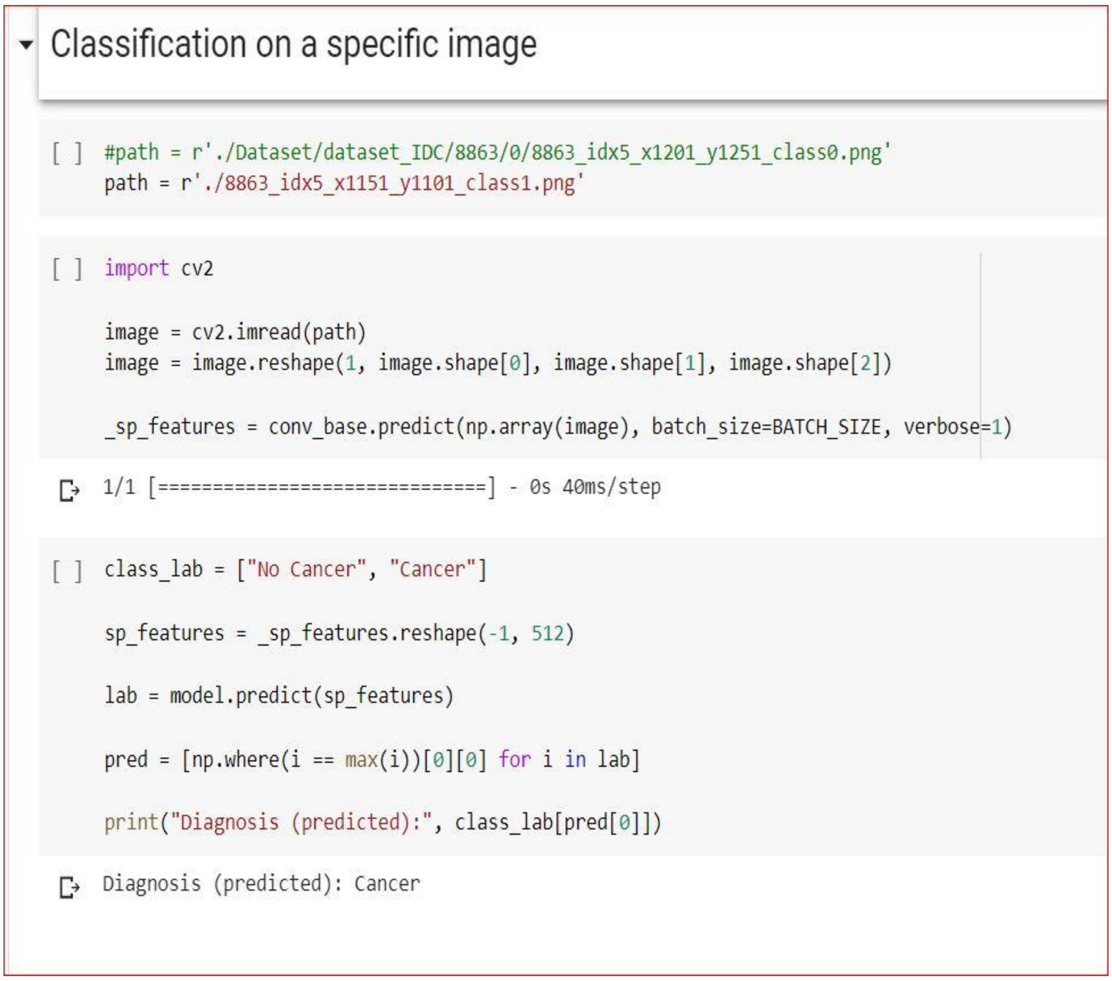


Fig 6.7: Loss graph

- The above graph shows loss in training and validation



Classification on a specific image

```
[ ] #path = r'./Dataset/dataset_IDC/8863/0/8863_idx5_x1201_y1251_class0.png'
    path = r'./8863_idx5_x1151_y1101_class1.png'
```

```
[ ] import cv2

    image = cv2.imread(path)
    image = image.reshape(1, image.shape[0], image.shape[1], image.shape[2])

    _sp_features = conv_base.predict(np.array(image), batch_size=BATCH_SIZE, verbose=1)

    1/1 [==============================] - 0s 40ms/step
```

```
[ ] class_lab = ["No Cancer", "Cancer"]

    sp_features = _sp_features.reshape(-1, 512)

    lab = model.predict(sp_features)

    pred = [np.where(i == max(i))[0][0] for i in lab]

    print("Diagnosis (predicted):", class_lab[pred[0]])

    Diagnosis (predicted): Cancer
```

Fig 6.8: Output shown for a specific image

- This figure shows output obtained for a given input image
- Based on the model prediction, it predict whether the image is cancerous or not
- For the input image of size 50X50, the model predicts the diagnosis as Cancer

CHAPTER 7

TESTING

This chapter gives an overview of the various types of testing incorporated during the entire duration of the project.

7.1 Unit Testing

Testing of an individual software component or module is termed as Unit Testing. It is typically done by the programmer and not by testers, as it requires detailed knowledge of the internal program design and code. It may also require developing test driver modules or test harnesses.

7.2 Component Testing

Component Testing is mostly performed by developers after the completion of unit testing. Component Testing involves testing of multiple functionalities as a single code and its objective is to identify if any defect exists after connecting those multiple functionalities with each other.

7.3 Integration Testing

Testing of all integrated modules to verify the combined functionality after integration is termed as Integration Testing. Modules are typically code modules, individual applications, client and server applications on a network, etc. This type of testing is especially relevant to client/server and distributed systems.

7.4 System Testing

Under System Testing technique, the entire system is tested as per the requirements. It is a Black-box type Testing that is based on overall requirement specifications and covers all the combined parts of a system.

7.5 Interface Testing

The objective of this Interface Testing is to validate the interface as per the business requirement. The expected interface of the application is mentioned in the detailed design document and interface mock-up screens. Checks if the application correctly connects to the server.

7.6 Compatibility Testing

Compatibility Testing checks whether the application is compatible with the specified software and hardware requirements and functions efficiently as expected.

7.7 Performance Testing

Performance testing is used to check for appropriate and efficient performance is shown by the system as per the requirements. The connection requirements are to be maintained to ensure efficient performance evaluation.

7.8 Usability Testing

Under Usability Testing, User-friendliness check is done. The application flow is tested to know if a new user can understand the application easily or not, proper help is documented if a user gets stuck at any point. Basically, system navigation is checked in this testing.

CHAPTER 8

CONCLUSTION AND FUTURE WORK

8.1 CONCLUSION

The present work is based on the classification of breast cancer using capsule net architecture. From this work, it is clear that the performance of the conventional architectures can be improved by data pre-processing and parameter tuning.

The results show that this method can be used as an automated tool to assist doctors in disease diagnosis, which may lead to higher concentration in the treatment at early stages rather than diagnosis and can increase the cancer survival rate.

As it is difficult to detect the breast cancer in early stages, doctors can use CNN as second opinion. CNN provides accuracy and quality compared to other methods used to predict the breast cancer tumours as Benign and Malignant. CNN can be used as a second opinion by the doctors to diagnose the patients.

8.2 FUTURE WORK

- One direction is to improve the recognition accuracy of features using patches, size and the overlapping of patches.
- stain normalization, deeper architectures, and splitting the network before the last fully-connected layer could be investigated.
- Additional data with increased number of patients should be introduced.
- A better investigation on feature and classifier selection could also improve performance.

REFERENCES

- P. Boyle and B. Levin. (2008) "**World cancer report**" 2008. [Online]. Available:http://www.iarc.fr/en/publications/pdfs-online/wcr/2008/wcr 2008.pdf

- S. R. Lakhani, E. I.O., S. Schnitt, P. Tan, and M. van de Vijver, **"WHO classification of tumours of the breast**", 4th ed. Lyon: WHO Press, 2012.

- F. Spanhol, L. S. Oliveira, C. Petitjean, and L. Heutte, "**A dataset for breast cancer histopathological image classification**," IEEE Transactions on Biomedical Engineering (TBME), vol. 63, pp. 1455–1462, 2015.

- "**Breast cancer histopathological image classification using convolutional neural networks**," in 2016 International Joint Conference on Neural Networks (IJCNN). IEEE, jul 2016, pp. 2560–2567.

- N. Bayramoglu, J. Kannala, and J. Heikkil¨a**, "Deep learning for magnification independent breast cancer histopathology image classification,"** in 23rd International Conference on Pattern Recognition, vol. 1, December 2016.

- Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Fei-Fei, L.: "**Largescale video classification with convolutional neural networks"**. In: Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on. pp. 1725–1732. IEEE (2014)

- Krizhevsky,A.,Sutskever,I.,Hinton,G.E.:"**Imagenet classification with deep convolutional neural networks"**. In: Advances in neural information processing systems. pp. 1097–1105 (2012)