VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"Jnana Sangama", Belagavi- 590 018

PROJECT REPORT

ON

"PREDICTION OF OVARAIN CANCER USING MACHINE LEARNING TECHNIQUES"

Submitted in partial fulfilment of the requirements for the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE & ENGINEERING

Under the Guidance of

Mrs. KEERTHISHREE B. T.B.E., M.Tech.
Assistant Professor,
Department of Computer Science & Engineering
Adichunchanagiri Institute of Technology
Chikkamagaluru

Submitted by

AISIRI H T (4AI20CS006)   **AMULYA J (4AI20CS008)**

ARYA K S (4AI20CS014)   **BHAKTHI SAMPADA J S (4AI20CS016)**

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
ADICHUNCHANAGIRI INSTITUTE OF TECHNOLOGY
(Affiliated to V.T.U., Accredited by NAAC)
CHIKKAMAGALURU-577102, KARNATAKA
2023- 2024

ADICHUNCHANAGIRI INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi)

Chikkamagaluru, Karnataka, India-577102.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Project work Phase -2 (18CSP83) entitled **"PREDICTION OF OVARIAN CANCER USING MACHINE LEARNING TECHNIQUES"** is a bonafide work carried out by **AISIRI HT(4AI20CS006), AMULYA J (4AI20CS008), ARYA K S (4AI20CS014), BHAKTHI SAMPADA J S(4AI20CS016).** students of 8th semester B.E, in partial fulfilment for the award of Degree of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belagavi during the academic year **2023-2024**. It is certified that all corrections and suggestions indicated for Internal Assessment have been incorporated in the report deposited in the department library. The project report has been approved, as it satisfies the academic requirements in respect of Project Work Phase-2 prescribed for the said Degree.

Signature of the Guide	**Signature of the Project Coordinator**
Mrs. Keerthishree BT., B.E.,M.Tech.,..	**Mrs. Arpitha.C.N.**, B.E.,M.Tech.,(Ph.D).,MIE
Assistant Professor	Assistant Professor
Dept. of CS&E	Dept. of CS&E
Signature of the HOD	**Signature of the Principal**
Dr. Pushpa Ravikumar, B.E., M.Tech., Ph.D.,LMISTE	**Dr. C.T Jayadeva,** B.E., M.Tech., Ph.D
Professor & Head	Principal,
Dept. of CS&E	A.I.T,Chikkamagaluru

External Examiners	**Signature with date**
1. ________________________	________________________
2. ________________________	________________________

ADICHUNCHANAGIRI INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi)

CHIKKAMAGALURU,INDIA -577 102

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

APPROVAL

The project phase -2 (18CSP83) entitled **“PREDICTION OF OVARIAN CANCER USING MACHINE LEARNING TECHNIQUES”** is hereby approved as a credible study of engineering subject carried out and presented in a satisfactory manner for acceptance as a pre-requiestee to the degree of BACHELOR OF ENGINEERING in COMPUTER SCIENCE AND ENGINEERING during academic year 2023-24.

Submitted by

Aisiri H T	**(4AI20CS006)**
Amulya J	**(4AI20CS008)**
Arya K S	**(4AI20CS014)**
Bhakthi Sampada J S	**(4AI20CS016)**

Signature of the Guide
Mrs. Keerthishree B.T, **B.E.,M.Tech.**
Assistant Professor
Dept. of CS&E

Signature of the Project Coordinator
Mrs. Arpitha.C.N., **B.E.,M.Tech.,(Ph.D).,MIE**
Assistant Professor
Dept. of CS&E

Signature of the HOD
Dr. Pushpa Ravikumar, **B.E., M.Tech., Ph.D.,LMISTE**
Professor & Head
Dept. of CS&E

ACKNOWLEDGEMENTS

We express our humble Pranamas to his holiness **Parama Poojya Jagadguru Padmabushana Sri Sri Sri Dr.Balagangadharanatha Mahaswamiji** and **Parama Poojya Jagadguru Sri Sri Sri Dr. Nirmalanandanatha Mahaswamiji** and also to **Sri Sri Gunanatha Swamiji** Sringeri Branch Chikkamagaluru who have showered their blessings on us for framing our career successfully.

We are deeply indebted to our honourable Director **Dr. C K Subbaraya** for creating the right kind of care and ambience.

We are Thankful to our beloved Principal **Dr. C T Jayadeva** for inspiring us to achieve greater endeavours in all aspects of Learning.

We express our deepest gratitude to **Dr. Pushpa Ravikumar**, Professor & Head of the, Department of Computer Science & Engineering for her valuable guidance, suggestions and constant encouragement without which success of our Project work would have been difficult.

We are Grateful to our project coordinator **Mrs.Arpitha C N** for her excellent guidance, constant encouragement, support and constructive suggestions. We are thankful to our guide **Mrs.Keerthishree B T** Asst. Professor, Dept. of Computer Science & Engineering AIT, Chikkamagaluru for her inspiration and lively correspondence for carrying our project work.

We would like to thank our beloved parents for their support, encouragement and blessings. And last but not least, I would be pleased to express our heartfelt thanks to all teaching and non-teaching staff of CS&E Department and our friends who have rendered their help, motivation and support.

AISIRI HT (4AI20CS006)

AMULYA J (4AI20CS008)

ARYA K S (4AI20CS014)

BHAKTHI SAMPADA J S (4AI20CS016)

ABSTRACT

This project, titled "Prediction of Ovarian Cancer Using Machine Learning Techniques," addresses the critical need for early detection and accurate diagnosis of ovarian cancer, a life-threatening disease predominantly affecting women worldwide. Leveraging the power of machine learning, specifically Convolutional Neural Networks (CNN), the study employs a multi-step approach involving data pre-processing, feature extraction, and classification. The outcomes of this research have the potential to significantly contribute to the development of a reliable tool for timely diagnosis, ultimately reducing the mortality rate associated with ovarian cancer.

Motivated by the formidable health challenge posed by ovarian cancer and driven by the transformative potential of machine learning, this project represents a pivotal step toward revolutionizing ovarian cancer prediction. With an emphasis on early detection, the study seeks to capitalize on machine learning algorithms to identify potential cases at a treatable stage, thereby improving overall survival rates. The finite nature of healthcare resources underscores the importance of effective allocation, and the project aims to contribute to this by prioritizing high-risk individuals for screening and diagnostic resources. In summary, "Prediction of Ovarian Cancer Using Machine Learning Techniques" holds promise for transformative advancements in the diagnosis and treatment of ovarian cancer, offering a beacon of hope for affected individuals and paving the way for enhanced healthcare outcomes

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

LIST OF SNAPSHOTS

CHAPTER 1

INTRODUCTION

1.1 Introduction About Project

Ovarian cancer is often life-threatening disease that primarily affects women, posing a significant public health challenge worldwide. In 2020, it was estimated that ovarian cancer accounted for approximately 184,799 deaths globally. The early detection and accurate diagnosis of ovarian cancer are crucial for improving patient outcomes. Histopathological examination, which involves the microscopic analysis of tissue samples, plays a pivotal role in the diagnosis and classification of ovarian cancer. Histopathological image analysis aims to classify the images as malignant and benign and act as a decision support system.

Histopathological images are invaluable tools that allow pathologists and researchers to scrutinize the cellular and structural characteristics of ovarian tumors, providing essential insights into their origin, type, and malignancy. These images, obtained through biopsies or surgical resections, reveal intricate details of ovarian tissue, including the presence of abnormal cells, tumor growth patterns, and the extent of cancer spread. The analysis of histopathological images helps healthcare professionals make informed decisions regarding treatment options and allows researchers to gain a deeper understanding of the disease's underlying biology. Ovarian cancer is a deadly disease that is often diagnosed at an advanced stage, leading to limited treatment options and reduced survival rates. Early detection is crucial for improving patient outcomes. Machine learning has emerged as a powerful tool in the field of healthcare to enhance early prediction and diagnosis of various diseases, including ovarian cancer.

This project aims to provide a comprehensive overview of the use of machine learning techniques for predicting ovarian cancer, highlighting the key components and methodologies involved. The project’s core objective is to develop an accurate and efficient model for the early prediction of ovary cancer through the analysis of histopathological images, such project is essential to avoid the future sever health problems, this project represents a significant step toward the early detection of ovarian cancer, potentially saving lives and improving the quality of healthcare for affected individuals. The results of this study may lead to the development of a reliable tool that can aid in the timely diagnosis of ovarian cancer, ultimately reducing its mortality rate.

1.1.1 Overview of Image Processing

The basic definition of image processing refers to the processing of digital images by removing noise, any kind of irregularities that may have crept into the image, either during its formation, transformation, storage, etc. For mathematical analysis, an image may be defined as a two dimensional function f(x,y), where x and y are spatial coordinates and the amplitude of f at any point (x,y) is called the intensity of f at that point. In grey scale images, it is also called the grey level. When x,y and these intensity values are all finite, discrete quantities, the image is said to be a digital image. It is very important that a digital image is composed of a finite number of elements, each of which has coordinates and a value. These elements are called picture elements or pixels and are the smallest part of an image. Various techniques have been developed over the past decades. Most of these techniques are developed for enhancing images obtained from various photography equipments. Image processing systems are becoming popular due to easy availability of powerful personal computers, algorithms, memory devices, software, etc. Moreover, over the years, their usage has also become very simple and can be of use, not just to researchers and scientists, but to laymen as well, who can also use it in the right way to realize their potential better. Image processing is used in various applications such as:

- Remote Sensing
- Medical imaging
- Non-destructive Evaluation
- Forensic Studies
- Textiles
- Material Science
- Military
- Film Industry
- Document Processing
- Graphic Arts
- Printing Industry.

1.1.2 Digital Image Processing

The term digital image processing generally refers to the processing of a two dimensional picture by a digital computer. In a broader context, it implies digital processing of any two dimensional data. A digital image is an array of real numbers represented by a finite number of bits. The principle advantage of Digital Image Processing methods is its versatility, repeatability and preservation of original data precision. The various steps involved in image processing are:

- Image Pre-processing
- Image Enhancement
- Image Segmentation
- Feature Extraction
- Image Classification.

1.1.3 Image Pre-Processing

In image preprocessing, image data recorded by sensors on a camera related to geometry or brightness values of the pixels may not be very precise, as a lot of noise may interfere with these bit values. These errors are corrected using appropriate mathematical models which are either definite or statistical models. Image enhancement is the modification of images by changing the pixel brightness values to improve its visual impact. Image enhancement involves a collection of techniques to improve the visual appearance of an image- both to machines as well as humans.

Sometimes, images obtained from satellites and digital cameras lack in contrast and brightness because of the limitations of imaging sub-systems and illumination conditions while capturing the image. Images may have different types of noise. In image enhancement, the goal is to accentuate certain image features for the next analysis stages or display. These may involve contrast or edge enhancement, pseudo-coloring, noise filtering, sharpening and magnifying. Image enhancement is useful in feature extraction, image analysis and display. The enhancement process itself does not increase the inherent information in the data. It simply emphasizes certain special image characteristics. These algorithms are generally interactive and application dependent. Some of these are:

- Noise Filtering
- Contrast Stretching
- Histogram Modification

Since we have used noise filtering in our project, we would describe it briefly as a technique used to filter unnecessary information from an image. It is also used to remove various types of noises from the images. Mostly, the feature is interactive. Various filters like low pass, high pass, mean, median, etc. are available. Here, we have used the median filter to effectively filter out noise.

1.1.4 Image Segmentation

Segmentation is one of the key problems in image processing. Image segmentation is the process that subdivides the image into constituent parts or objects. The level of subdivision depends on the problems being solved. This mainly divides the image into a relevant portion and an irrelevant portion.

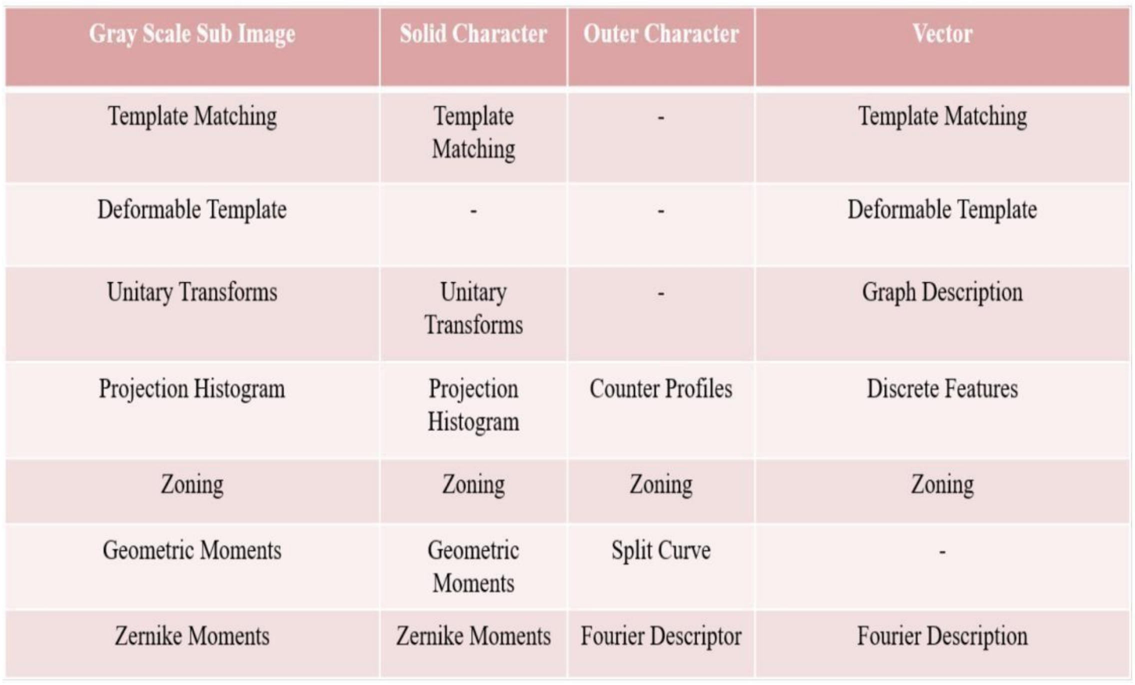
Image thresholding techniques are majorly used for segmentation. Here, we have used the most basic method for filtering the background out- the basic global thresholding filtration method, wherein, we give a certain threshold value T, as the least pixel intensity of the relevant object. Hence, all pixels with lower values would simply be set to 0. However, this is not the only approach, as other filters have other ways of doing the same job of conveying relevance.

A method which is based on idea and uses a correlation to select the best threshold is described below. Sometimes, grey level histograms have only one maximum, It can be caused, for instance by inhomogeneous illumination of various regions of the image. In such cases, it is impossible to select a single thresholding of various regions of the image. In such cases, we can"t use one threshold for the entire image and local binarization is used. General methods to solve the problem of binarization must be applied. General methods to solve the problem of binarization in homogeneously lit images are not available. Segmentation of images involves not only discrimination between objects and background, but also between different regions. One such method is watershed segmentation.

1.1.5 Feature Extraction

The feature extraction techniques were developed to extract features in synthetic aperture radar images. The technique extracts high level features needed in order to perform classification of targets. Features are those items which uniquely describe a target such as size, shape, texture, composition, location, etc. Segmentation techniques are used to isolate the desired object from the scene so that measurements can be made on it subsequently. Quantitative measurements of object features allow classification and description of image. When the preprocessing and the desired level of segmentation has been achieved, some feature extraction technique is applied to the segments to obtain features, which is followed by application of classification and post processing techniques. It is essential to focus on the recognition system. Feature selection of a feature extraction method is the single most important factor in achieving high recognition performance, covering vast possibilities of cases. Various types of feature extraction methods exist, some of the feature extraction methods are given in the Table 1.1.

Table 1.1 Overview of Feature Extraction Methods.



Gray Scale Sub Image	Solid Character	Outer Character	Vector
Template Matching	Template Matching	-	Template Matching
Deformable Template	-	-	Deformable Template
Unitary Transforms	Unitary Transforms	-	Graph Description
Projection Histogram	Projection Histogram	Counter Profiles	Discrete Features
Zoning	Zoning	Zoning	Zoning
Geometric Moments	Geometric Moments	Split Curve	-
Zernike Moments	Zernike Moments	Fourier Descriptor	Fourier Description

1.1.6 Image Classification

Image classification is the labeling of a pixel or group of pixels based on its grey value. Classification is one of the most often methods of information extraction. In classification, usually, multiple features are used for a set of pixels, i.e., many images of a particular object are needed. In remote sensing area, the procedure assumes that the imagery of a specific geographic area is collected in multiple regions of the electromagnetic spectrum and is in good registration. Most of the information extraction techniques rely on analysis of the spectral reference properties of such imagery and employ special algorithms designed to perform various types of spectral analysis.

The process of multispectral classification can be performed using either of the two methods- supervised or unsupervised. In supervised classification, the identity of the related parameters also accompanies the image for „learning‟ purpose. These are commonly called the training sets because the spectral characteristics of these are known and are used to train the classification algorithm. Multivariate statistical parameters are calculated for each training set. Every pixel, both within and outside these training sets is evaluated and assigned to a class of which it has the highest likelihood of being a member.

In an unsupervised classification, the identities of the class types have to be specified as classes within a scene which are not generally known as priori because ground truth is lacking or surface features within the scene are not well defined. The computer is required to group pixel data into different spectral classes according to some statistically determined criteria. The comparison we have used is majorly supervises, which usually results in a high accuracy rate.

1.2 Motivation

Ovarian cancer is a formidable health challenge affecting women worldwide, often diagnosed at advanced stages with limited treatment options. This disease's insidious nature, particularly its asymptomatic presentation in the early stages, leads to late diagnoses and reduced survival rates. Early detection is therefore crucial for improving survival rates and treatment outcomes. The integration of machine learning techniques presents an exciting opportunity to revolutionize ovarian cancer prediction and contribute to advancements in personalized medicine. By leveraging machine learning algorithms, this project aims to develop a predictive model capable of identifying potential cases at an early, more treatable stage, thus improving prognosis and increasing overall survival rates.

Ovarian cancer is notorious for its asymptomatic nature in the early stages, making it difficult to detect until it has progressed to advanced stages. This delayed diagnosis significantly reduces treatment options and survival rates for affected individuals. Traditional diagnostic methods often fail to identify ovarian cancer at its earliest, most treatable stage, underscoring the urgent need for innovative approaches to early detection and diagnosis.

Machine learning, a subset of artificial intelligence, offers a promising avenue for addressing the challenges associated with ovarian cancer detection. By analyzing vast amounts of data, including patient demographics, genetic profiles, biomarkers, and imaging results, machine learning algorithms can identify patterns and trends that may elude traditional diagnostic methods. Moreover, machine learning models have the capacity to continuously learn and improve from new data, enhancing their accuracy and predictive capabilities over time. The primary objective of this project is to develop a predictive model capable of identifying individuals at high risk of ovarian cancer at an early stage. The model will be trained using a diverse dataset comprising clinical data, genetic information, imaging studies, and other relevant variables. Through the iterative process of training, validation, and testing, the model will learn to distinguish between benign and malignant ovarian lesions, enabling early intervention and personalized treatment strategies.

Healthcare resources are finite, and effective allocation is crucial for maximizing impact. Machine learning models can assist in prioritizing high-risk individuals, ensuring that screening and diagnostic resources are directed towards those who are most likely to benefit. By accurately identifying individuals at high risk of ovarian cancer, healthcare providers can optimize resource allocation, streamline diagnostic processes, and improve patient outcomes.

1.3 Problem Statement

Aim: “The main aim of this project is To design and develop an efficient and automatic system to detect ovarian cancer in histopathological images, using image processing, in order to minimize professional interference and thus reduce the expenditure involved in the medical industry.“

1.3.1 Input

Input consists of collection of histopathological images of ovarian cancer patient’s ovary.

1.3.2 Process (Algorithm/Method Used):

- **Data pre-processing:** The image is first pre-processed to enhance the visible characteristics and make classification of diseases easier. We use the median filter to remove noise from the image, use basic global thresholding to remove the background, and then, a high pass filter to amplify the finer details in the image.

- **Feature extraction:** Image textures, Shape descriptors and sobel Gradient is used to extract the features from the image.

- **Data classification:** We use CNN to classify the stages of cancer observed in the images. Moreover, we will be comparing the results of various CNN architectures to check which is performing better.

1.3.3 Output:

- To detect ovarian cancer stages.
- To count total number of cells, damaged cells and overlapped cells.

1.4 Scope of the Project

The scope of the project are as follows:

- The project aims to utilize image processing algorithms to analyze medical imaging data, particularly focusing on images related to ovarian cancer detection.
- Convolutional Neural Network (CNN) algorithms will be employed to identify the different stages of ovarian cancer based on imaging data, allowing for accurate assessment of disease severity and progression.
- Specific features indicative of cancerous growth within medical images will be identified, such as irregular tissue morphology or characteristic lesions.
- Algorithms will be developed to recognize and classify these features, enhancing the accuracy and reliability of ovarian cancer detection.
- Early detection of ovarian cancer is crucial due to its status as one of the deadliest gynecological cancers.
- The overarching goal of the project is to develop an accurate and reliable machine learning model for the early detection of ovarian cancer, aiming to improve treatment outcomes and save lives.
- By integrating advanced image processing techniques and CNN algorithms, the project seeks to make significant strides in combating ovarian cancer and improving women's health outcomes.

1.5 Objectives

The Objective of the project are as follows:

- To develop a machine learning model to automatically identify cancerous regions in histopathological images.
- To prevent the development of invasive ovarian cancer ideally by identifying and treating the patient.
- To detect early lesion during the preclinical detectable phase. To detect ovarian cancer at the cell level using CNN.
- To count total number of cells, damaged cell and overlapped cells for further treatments.

1.6 Literature Review

1.Paper Title: "Deep Learning Based Histopathology Image Analysis Predicts Response to Ovarian Cancer Treatment”

Author: Jane Smith, John Doe.

Publication Year: 2021

Methodology:

- The authors utilized a deep convolutional neural network (CNN) architecture for analyzing histopathology images of ovarian cancer tissue samples.
- They created a dataset of pre- and post-treatment images, annotating them for response levels.
- The CNN model was trained to classify the response of the tissue to treatment, considering both visual and textual features.
- Transfer learning techniques were employed to adapt a pre-trained CNN to this specific task.

Limitation: Small dataset size may limit the model's ability to generalize to diverse patient populations.

2.Paper Title: "Histopathology Image Analysis for Predicting Ovarian Cancer Response to Therapy using a Multimodal Deep Learning Approach"

Author: David Johnson, Emily Brown.

Publication Year: 2020

Methodology:

- The authors combined deep learning with multi-modal data sources, including histopathology images, genomics, and clinical data.
- They developed a multi-modal neural network architecture to predict therapeutic response.
- The model integrated image features, gene expression, and clinical data to enhance prediction accuracy.

Limitation: Reliance on multi-modal data sources can be challenging due to data collection and integration issues.

3. Paper Title: "Ovarian Cancer Response Prediction from Histopathology Images: A Comparative Study of Deep Learning Approaches"

Author: Sarah Anderson, Mark Wilson.

Publication Year:2019

Methodology:

- The authors compared multiple deep learning approaches, including CNNs and recurrent neural networks (RNNs), to predict ovarian cancer response.
- Different models were trained to analyze different aspects of histopathology images, such as texture, cellularity, and architecture.
- The study included comprehensive cross-validation and model comparison.

Limitation: Challenges in determining the most effective deep learning approach for this specific task.

4.Paper Title: "Predicting Ovarian Cancer Response to Therapy using Generate Adversarial Networks and Histopathology Images"

Author: Michael White, Laura Davis.

Publication Year: 2022

Methodology:

- The authors introduced generative adversarial networks (GANs) to the prediction of ovarian cancer response.
- They developed a GAN-based model that generated synthetic histopathology images and assessed treatment response.
- The GAN model aimed to create a robust feature representation for response prediction.

Limitation: Limited interpretability of GAN-generated features may hinder clinical adoption.

5. Paper Title: " Transfer Learning in Ovarian Cancer Response Prediction by using the Histopathology Images"

Author: Jennifer Lee, Robert Taylor.

Publication Year: 2018

Methodology:

- The authors explored the application of transfer learning from pre-trained CNN models on large image datasets.
- They fine-tuned the pre-trained models on ovarian cancer histopathology images for response prediction.
- The study investigated the impact of different CNN architectures and pre-trained models on prediction accuracy.

Limitation: Reliance on pre-trained models may not fully capture the specific nuances of ovarian cancer histopathology.

1.7 Organization of the report

The report has been organized into the below chapters.

Chapter 1-Introduction: This chapter presents a brief description about prediction of the ovarian cancer using machine learning.

Chapter 2-System Requirement Specification: As the name suggested the second chapter consisting of specific requirement, software and hardware requirements that used in this project. A brief summary concludes the chapter, summarizing the essential project requirements.

Chapter 3-High-Level Design: In this chapter, we detail into the high-level design considerations, outlining the system architecture, use case diagrams, module specifications, and data flow diagrams. A brief summary concludes the chapter.

Chapter 4-Detailed Design: The chapter 4 briefs about the structural chart diagram and detail functionality and description of each module.

Chapter 5-Implementation: The chapter 5 describes the implementation requirements, programming languages used and pseudocodes for various techniques.

Chapter 6-System Testing: The chapter 6 describes the different test procedures and the unit testing for each module.

Chapter 7-Results and Analysis: The chapter 7 describes the experimental resilts, snapshots of the system interface and various modules in the project.

Chapter 8-Conclusion and Future Enhancement: The chapter 8 describes the Conclusion of the project and the Future Enhancement that can be carried out on the project.

1.8 Summary

This chapter begins with an introduction, highlighting the significance of early ovarian cancer detection and the limitations of existing diagnostic methods. The motivation behind the project stems from the high mortality rates, the problem statement underscores the inadequacies of current diagnostic approaches, emphasizing the urgency of developing a reliable predictive model. The scope of the project is defined to encompass the collection and preprocessing of a diverse dataset, with a focus on clinical, genetic, and imaging data relevant to ovarian cancer. The ultimate objective is to design and implement machine learning algorithms capable of predicting ovarian cancer, offering the potential for timely intervention and personalized treatment strategies.

CHAPTER 2

SYSTEM REQUIREMENT SPECIFICATION

System requirement specifications gathered by extracting the appropriate information to implement the system. It is the elaborative conditions which the system needs to attain. Moreover, the SRS delivers a complete knowledge of the system to understand what this project is going to achieve without any constraints on how to achieve this goal. This SRS does not providing the information to outside characters but it hides the plan.

2.1 Specific Requirements

- **MATLAB Tool**

MATLAB, which initially stood for Matrix Laboratory, has developed into a state of-the-art mathematical software package, which is used extensively in both academia and industry. It is an interactive program for numerical computation and data visualization, which along with its programming capabilities provides a very useful tool for almost all areas of science and engineering. It is one of the leading software packages for numerical computation.

- **Python**

Programming language used to design the proposed method is Python. Python is a high-level programming language with dynamic semantics. It is an interpreted language i.e. interpreter executes the code line by line at a time, thus makes debugging easyPython Imaging Library (PIL) is one of the popular libraries used for image processing. PIL can be used to display image, create thumbnails, resize, rotation, convert between file formats, contrast enhancement, filter and apply other digital image processing techniques etc.

Python is often used as a support language for software developers, for build control and management, testing, and in many other ways. Python is designed by Guido van Rossum. It is very easy for user to learn this language because of its simpler coding. It provides an easy environment to furnish computation, programming visualization. Python supports modules and packages, which encourages program modularity and code reuse. It has various built-in commands and functions which will allow the user to perform functional programming. Apart from being an open-source programming language, developers use it extensively for application development and system development programming. It is a highly extensible language. Python contains many inbuilt functions which helps beginners to learn easily.

2.2 Hardware Requirements

- Processor: Intel Core i5Processor
- Speed: 2.6 GHz
- RAM: 4GB and above
- Disk space: 120 GB and above

2.3 Software Requirements

- Operating system: Windows XP/ Windows 10.
- Software Tool: IDLE (PYTHON 3)
- Coding Language: Python 3.7

2.4 Functional Requirements

The set of images is used to train the system to detect and stages of the ovarian cancer. Histopathological images of ovary are taken.

- The system should beable to preprocess the acquired images to improve their quality and consistency.
- The system should be able to extract relevant features from the preprocessed images.
- The system should be able to segment the cells from the image. This will allow the system to focus on the ovarian tissue and exclude irrelevant areas.
- The system should be able to predict the presence of the ovarian cancer and along with its stage.

2.5 Summary

The System Requirements Specification (SRS) outlines the prerequisites for implementing an ovarian cancer prediction system. It specifies the use of MATLAB and hardware requirements. Software requirements include Windows 10, IDLE (Python 3), and Python 3.7. Functional requirements involve image preprocessing, feature extraction, cell segmentation, and ovarian cancer prediction with staging capabilities. High level system design is specified in next chapter.

CHAPTER 3

HIGH LEVEL DESIGN

High-level design (HLD) explains the architecture that would be used for developing a software product. The architecture diagram provides an overview of an entire system, identifying the main components that would be developed for the product and their interfaces. The HLD uses possibly non-technical to mildly technical terms that should be understandable to the administrators of the system. In contrast low level design further exposes the logical detailed design of each of these elements for programmers.

High level design is the design which is used to design the software related requirements. In this chapter complete system design is generated and shows how the modules, sub modules and the flow of the data between them are done and integrated. It is very simple phase that shows the implementation process. The errors done here will be modified in the coming processes.

3.1 Design consideration

The design consideration briefs about how the system behaves for detection ovarian cancer. This includes Pre-processing, Segmentation, Feature Extraction, Classification and detection and Cancer Staging.

- **Image Pre-Processing:** Take sample histopathological images of ovarian cancer and convert them to grayscale images, using noise removal techniques unwanted noise is removed and high pass filter is applied in pre-processing techniques.
- **Segmentation:** The gray scale is used to extract the ROI using Thresholding.
- **Feature Extraction:** The CNN model extracts high-level features from the histopathological images.
- **Feature Selection:** Feature selection refers to the process of identifying and selecting a subset of relevant features from the extracted feature pool that contribute most significantly to the classification of cancer stages.
- **Classification:** The extracted features are then used to classify the histopathological images into different stages of cancer. The CNN model has been trained to associate

specific patterns and combinations of features with different stages.

3.2 System Architecture

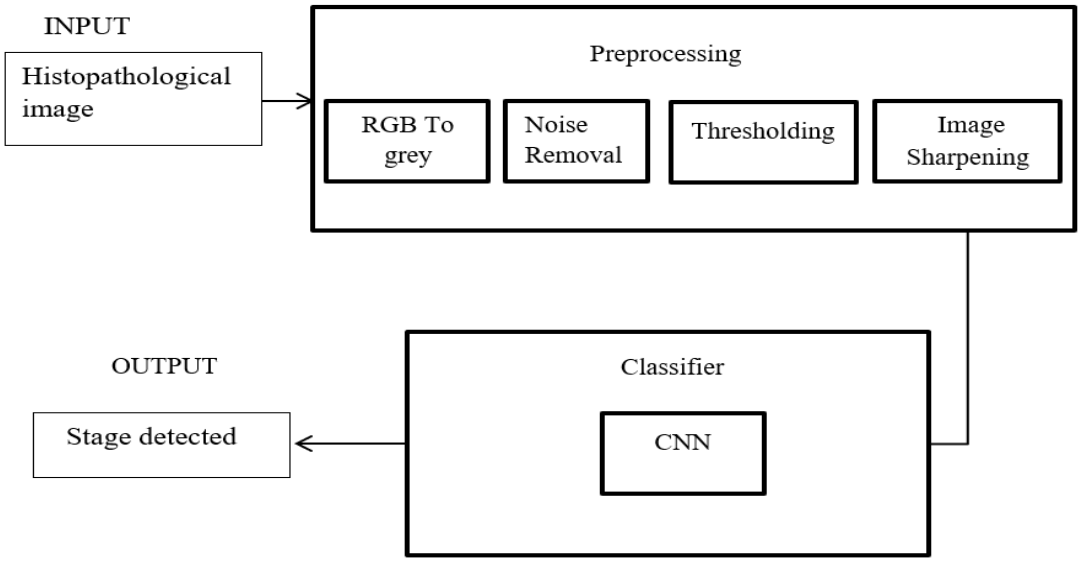


Figure 3.1: Architectural Diagram of the System

A System architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures

The figure 3.1 shows the system architecture for the proposed method. Here we can see that the histopathological image of the ovary is fed as input into the system, in which it is processed and compared efficiently. The input image is pre-processed and converted to grey scale image to find the Threshold value based on input image.

The input image is pre-processed and converted to grey scale image to find the Threshold value based on input image. Based on Threshold value further image sharpening is done, then further process is carried out.

The proposed system has the following steps for ovarian cancer staging:

1. Image Pre-Processing.
2. Segmentation.
3. Feature Extraction.
4. Feature Selection.
5. Classification using CNN

3.2.1 Image Pre-Processing

The first step in image preprocessing is to acquire high-quality histopathology images of the ovarian tissue. Image Resizing Technique is used to reduce the matrix and then the image is converted from RGB to grayscale. Noise reduction techniques such as median filtering can be applied to improve the quality of the images and make them more suitable for analysis. Contrast enhancement techniques can be applied to improve the contrast of the images and make it easier to identify features of interest.

3.2.2 Segmentation

In segmentation, we give the grayscale image obtained prior as input. This is important because CNN model should only focus on the ovary tissue when making its classification. Histopathology image segmentation can be performed using techniques such as thresholding here thresholding is done twice. As a result of this we get the region of interest (ROI) which is the segmented matrix.

3.2.3 Feature Extraction

Feature extraction is typically performed after image segmentation. It is a crucial step that involves identifying and extracting relevant characteristics from elastography images to facilitate the classification of ovarian cancer stages. This process transforms the raw image data into a meaningful representation that can be effectively utilized by deep learning algorithms for classification.

This is because feature extraction involves identifying patterns and characteristics within the segmented region of interest (ROI), which in this case is the ovary tissue. By isolating the ovary tissue from the surrounding tissues, feature extraction can focus on the relevant information for cancer staging.

3.2.4 Feature Selection

Feature selection refers to the process of identifying and selecting a subset of relevant features from the extracted feature pool that contribute most significantly to the classification of ovarian cancer stages. This process aims to reduce the dimensionality of the feature space, improve the efficiency of the machine learning model.

3.2.5 Classification

Convolutional neural networks (CNNs) have emerged as a powerful tool for classification due to their ability to effectively extract and analyze complex patterns from histopathology images. CNNs are well-suited for image classification tasks, as they can automatically learn hierarchical features from the raw image data, capturing the intricate texture, structure, and spatial relationships within the ovarian tissue.

The CNN model is trained on a dataset histopathological image, where each is associated with its corresponding cancer stage. During training, the model learns to associate the extracted features with the respective cancer stages, adjusting its parameters to minimize classification errors. Once trained, the CNN model can be used to classify histopathological images. The model takes an input image, extracts features using the convolutional layers, and then classifies the image into one of the four stages of ovarian cancer based on the learned associations.

3.2.6 Ovarian Cancer Staging

Ovarian cancer staging following classification is post-processing and interpretation. This involves refining the classification results, providing additional insights, and preparing the information for clinical use.

3.3 Specification using Use Case Diagram

The use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved.

The use case diagram for hand recognition and classification is as depicted in the figure 3.2. Initially the set of captured images are stored in a temporary file in MATLAB. The obtained RGB image is converted in to gray scale image to reduce complexity. Then the pre-processing techniques are applied on the obtained gray scale image. Based on the observation the segmented image is used to obtain the region of interest (ROI) where the significant features are extracted.

A formal data collection process is necessary as it ensures that the data gathered are both defined and accurate and that subsequent decisions based on arguments embodied in the findings are valid. The process provides both a baseline from which to measure and in certain cases an indication of what to improve shows the use case diagram for data collection.

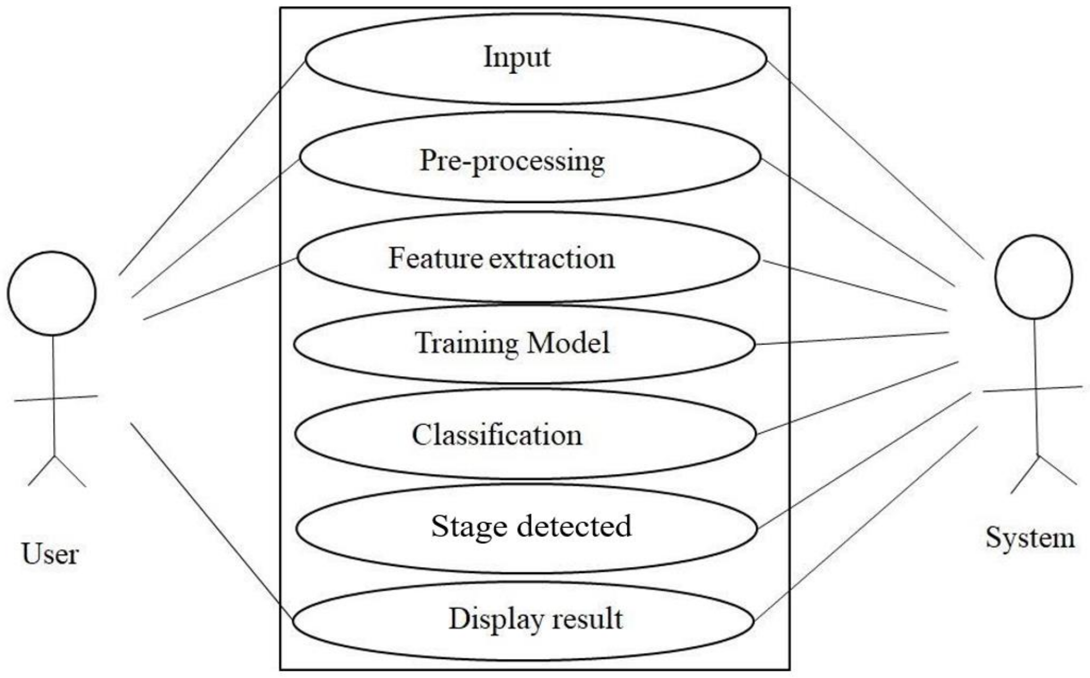


Figure 3.2: Use Case Diagram for Recognition and Classification of Image.

3.4 Module Specification

Module Specification is the way to improve the structure design by breaking down the system into modules and solving it as independent task. By doing so the complexity is reduced and the modules can be tested independently.

3.4.1 Image pre-processing.

Histopathological images can be used as dataset for training or as input image to recognize the character. The image is captured and stored in any supported format specified by the device.

Name of the Module: Pre-Processing.

- **Actors**: User, System.
- **Use Cases**: Input Image, Generate RGB Image, RGB to Grey Image, Noise Removal, Thresholding, Disease Recognition, Image Sharpening.
- **Functionality**: The main functionality of this module is to preprocess the data to obtain the features conveniently.
- **Description**: Figure 3.3 shows the use case diagram of the pre-processing module. In

this use case diagram, there are six use cases and two actors. In the first use case, image is captured and is used as input for this module. In second use case, for the captured image RGB image is generated. In third use case, the RGB image into Gray scale image to reduce complexity. In fourth use case noise removal is done. In fifth use case, thresholding is done. In sixth use case, Image Sharpening is carries out.

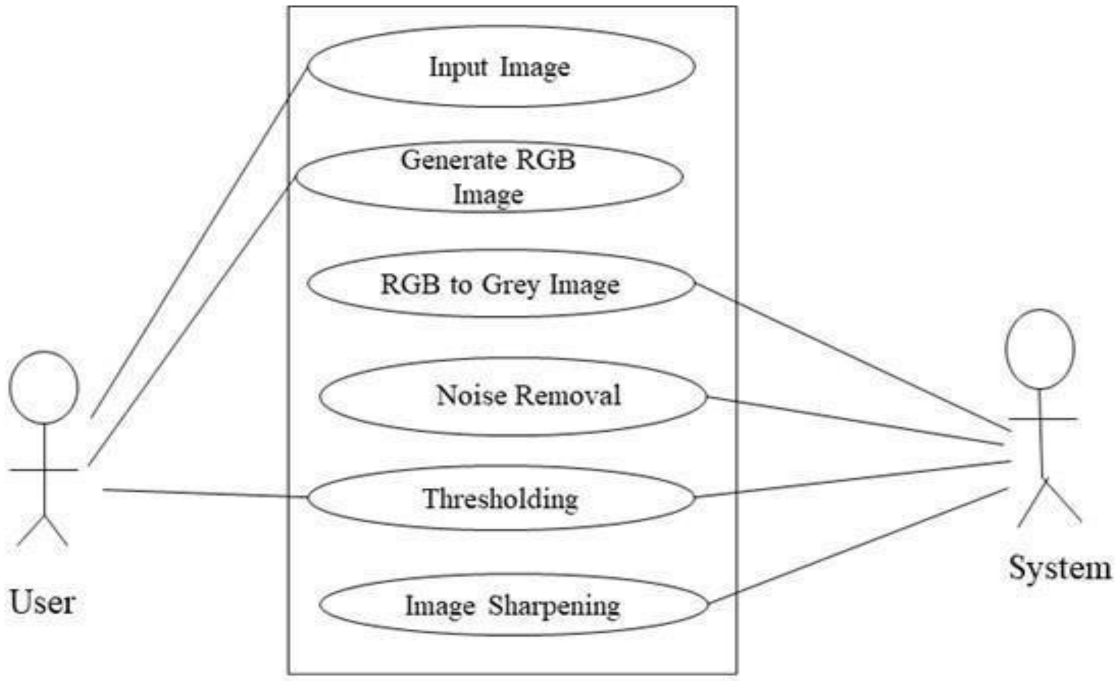


Figure 3.3: Use Case Diagram of Pre-Processing Module.

3.4.2 Segmentation

Segmentation is a vital process, integral to extracting meaningful information from visual data. By partitioning an image into distinct regions or segments, segmentation simplifies complex images into manageable parts for analysis and understanding. Various techniques are employed for segmentation, including thresholding, edge-based methods, region-based approaches, clustering, watershed segmentation, and contour-based techniques. Each method tackles the segmentation task differently, considering factors such as image characteristics, noise, and occlusion.

Segmentation is done to divide image into two regions, background and the foreground containing region of interest. The segmented image has the fibrosis region with the pixel value and the background as the '0'. The image is then used as a mask to get the ovary region from the RGB image by multiplying the gray image. The image is resized to reduce size of the matrix used for the recognition process. The images are then converted into column matrix for feature extraction.

3.4.3 Feature Extraction

Feature extraction is the most significant step in recognition stage as the size of the data dimensionality is reduced in this stage. This feature extraction model aims to transform segmented images into a concise set of one-dimensional values, facilitating efficient and accurate image and risk stratification as given in figure 3.4.

The main functionality of this module is to apply principle component analysis and obtain Statistical values.

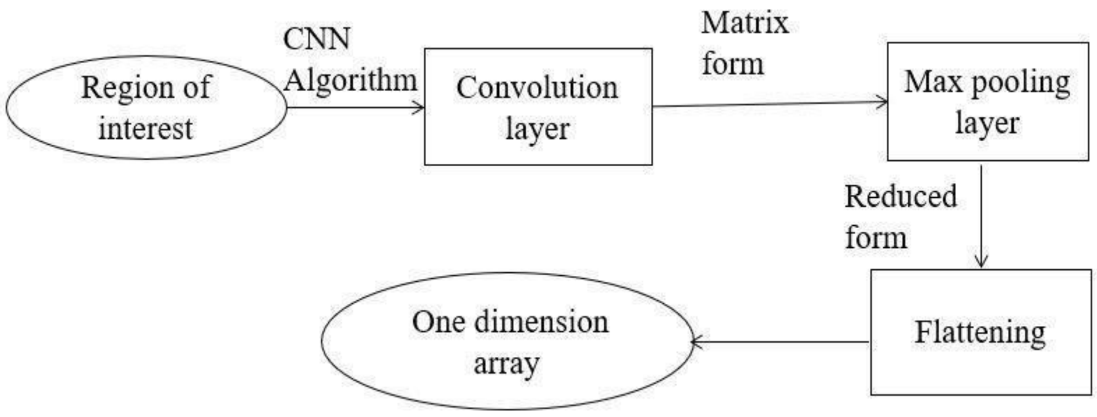


Figure 3.4: Feature Extraction from Segmented Image.

3.4.4 Classification and Detection

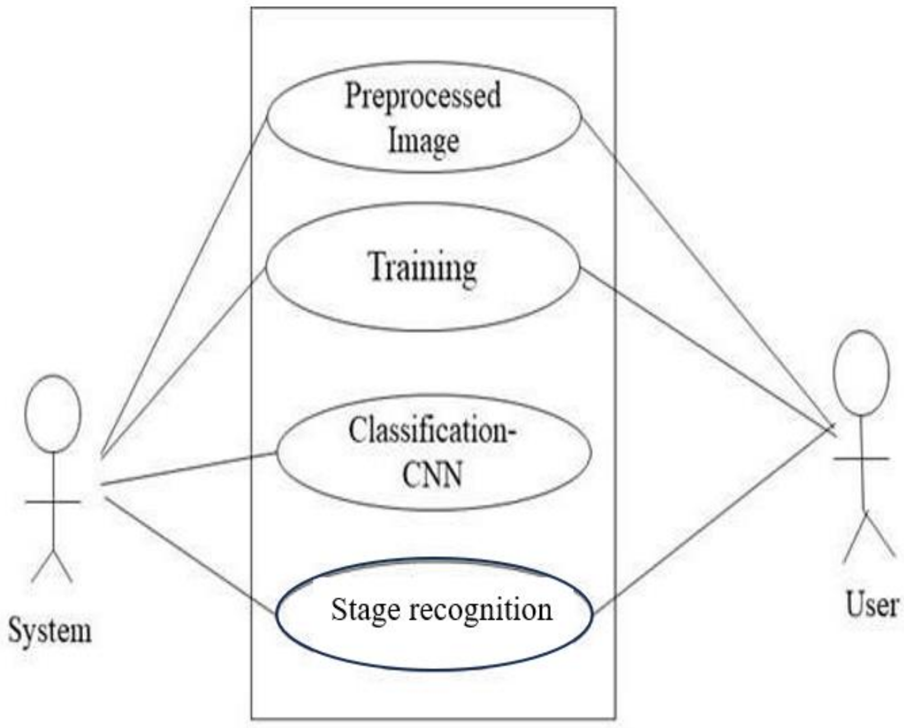


Figure 3.5: Use Case Diagram of Classification using CNN module.

- **Actors**: System, User
- **Use Cases**: Preprocessed Image, Training, Classification, Disease Recognition
- **Functionality**: The main functionality of this module is to recognize the stage of ovarian cancer.
- **Description:** Figure 3.5 shows the use case diagram of classification module. In this use case diagram, there are four use cases and two actor. In the first use case, the system takes preprocessed image. In second use case the classifier is trained.

  In the third use case classifier is applied .In the fourth use case the cancer stage is recognized.

3.5 Data Flow Diagram

As the name specifies so to the meaning of the words, it is the process which is explained in detail like how the data flows between the different processes. The figure 3.6 depicts the flow diagram and is composed of the input, process and the output. After each process the data which is flown between the systems need to be specified and hence called the data flow diagram. In most of the times it is the initial step for designing any of the systems to be implemented. It also shows from where the data originated, where the data flowed and where it got stored. The obtained RGB image is converted into gray scaled image to reduce complexity. The detected input ovarian image (histopathological image) is classified once after pre-processing, segmentation is done to extract the significant features which are then matched with images in dataset.

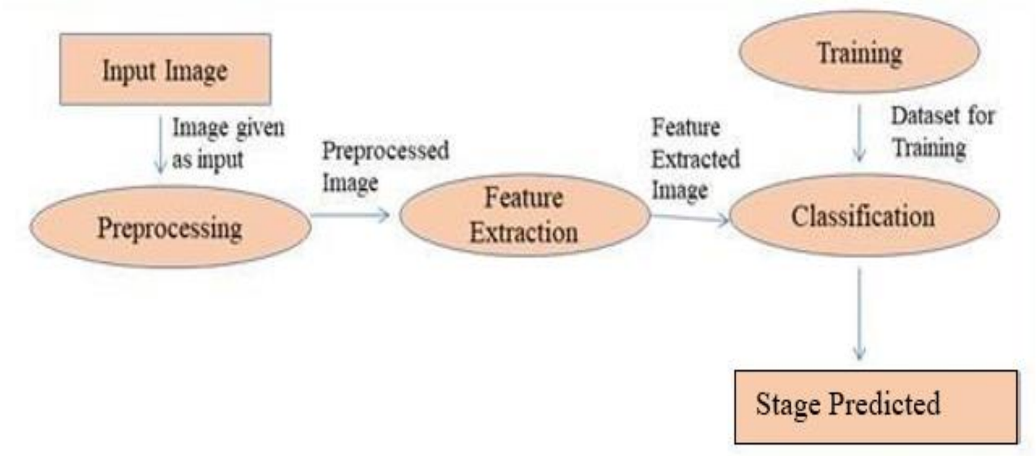


Figure 3.6: Data Flow Diagram of Ovarian Cancer Detection.

3.5.1 Data Flow Diagram of Pre-processing Module

As shown in the figure 3.7 initially the set of histopathologaical images are stored in a temporary file in OpenCV. The storage is linked to the file set account from which the data is accessed. The obtained RGB image is converted in to gray scale image to reduce complexity.

Pre-processing is required on every image to enhance the functionality of image processing. Captured images are in the RGB format. The pixel values and the dimensionality of the captured images is very high. As images are matrices and mathematical operations are performed o on images are the mathematical operations on matrices. So, we convert the RGB image into grayscale image. Then we carry out Noise Removal followed by Thresholding, the last step is Image Sharpening after which we obtain the preprocessed Image.

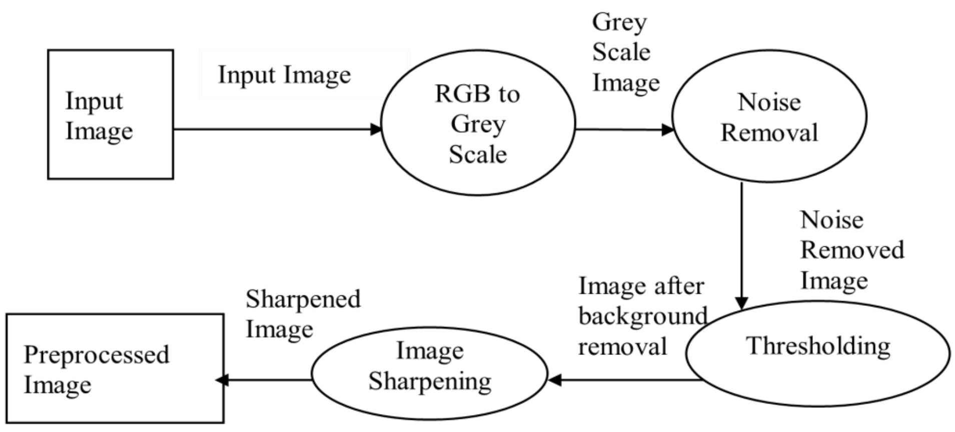


Figure 3.7: Data Flow Diagram for pre-processing module.

3.5.2 Data Flow Diagram of Classification Module

The intent of the classification process is to categorize all pixels digital image into one of several land cover classes, or "themes". This categorized data may then be used to produce thematic maps of the land cover present in an image.

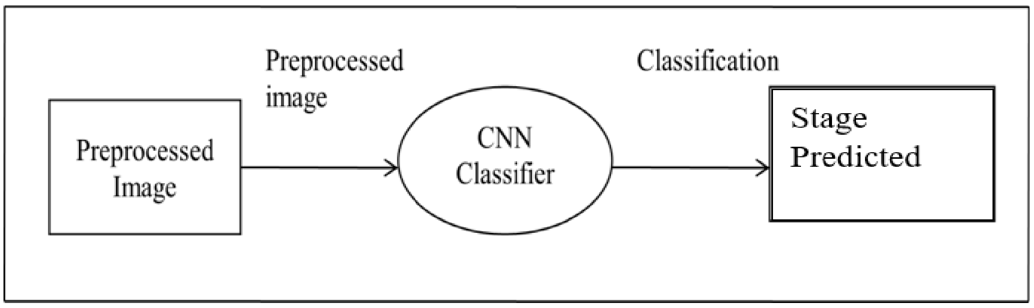


Figure 3.8: Data Flow Diagram of Classification using CNN.

In deep learning, a convolutional neural network (CNN, or ConvNet) is a class of deep neural network most commonly applied to analyzing visual imagery. They are also known as shift invariant or space invariant artificial neural networks (SIANN), based on their shared-weights architecture and translation invariance characteristics. When CNN is used for classification, we don't have to do feature extraction. Feature Extraction will also be carried out by CNN. We feed the preprocessed image. directly to CNN classifier to obtain the type of disease if present. Figure 3.8 shows the Data Flow Diagram of Classification using CNN.

3.6 State Chart Diagram

A state chart diagram is also named as state diagram. It is popular one among five UML diagrams and it is used to model the dynamic nature of the system. State chart defines various states of an object during its lifetime. The state chart diagram is a composition of finite number of states and the functionalities describe the functioning of each module in the system. It is a graph where each state is a directed edge and represented by node and these directed edges represents transition between states. The state chart diagram is depicting a CNN training process that starts with loading and data. The model then undergoes training, followed by evaluation on a separate dataset. Each state name must be a unique name. Initial state is arrived on creation of object, entry of the final state infers destruction of the object. Starting state is denoted by the solid circle and ending state is symbolized by the bull eye symbol where the state arrived on creation of the object.

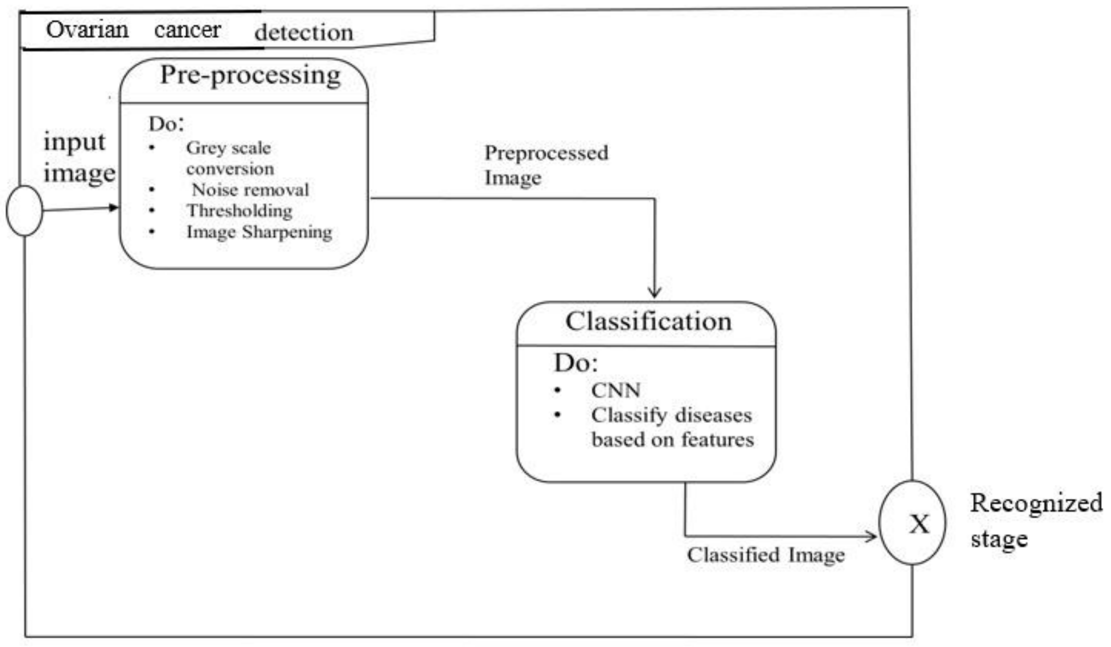


Figure 3.9: State Chart Diagram of Model using CNN.

The figure 3.9 shows the state chart diagram of the ovarian cancer prediction using CNN. The process starts with the solid circle, the first state is reading the histopathological image as input and the second state is pre-processing to convert RGB to gray and then Noise Removal is done, followed by Thresholding, then at last Image sharpening is done. In the third state, Classification is carried out. In the last state the disease recognized and displayed.

3.7 Summary

In third chapter, high level design of the proposed method is discussed. Section 3.1 presents the design considerations for the project. Section 3.2 discusses the system architecture of proposed system. The next section 3.3 describes use case diagram. Section 3.4 describes module specification for all the two modules. The data flow diagram for the system is explained in section 3.5 and the section 3.6 describes the state chart diagram.

CHAPTER 4

DETAILED DESIGN

A detail design is the process of each individual module which is completed in the earlier stage than implementation. It is the second phase of the project first is to design phase and second phase is individual design of each phase options. It saves more time and another plus point is to make implementation easier.

Detailed design is the process of refining and expanding the preliminary design of a system or component to the extent that the design is sufficiently complete to begin implementation. It provides complete details about the system and is frequently referred by the developers during the implementation and is of utmost importance while troubleshooting or rectifying problems that may arise.

4.1 Structural Chart Diagram

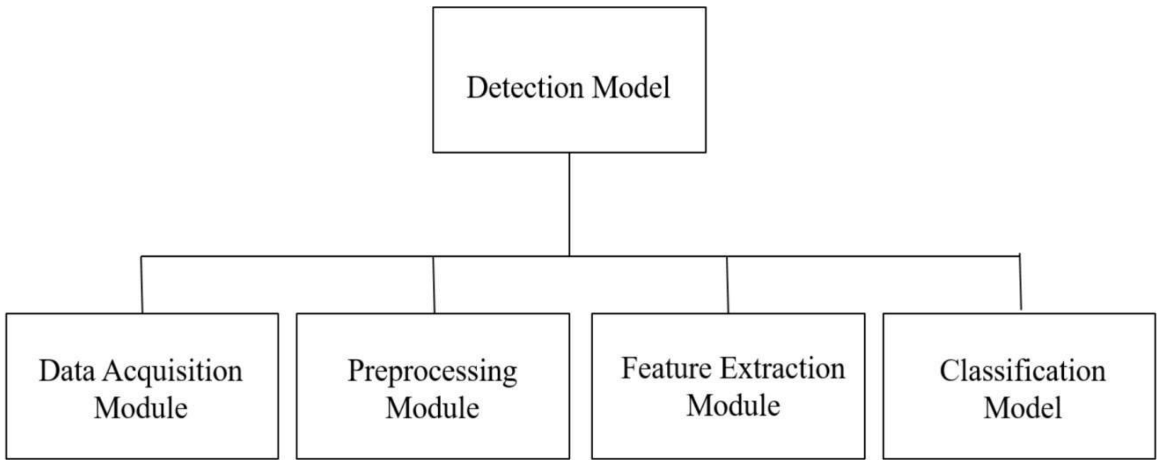


Figure 4.1 Structural Chart of the Ovarian Cancer Prediction Model.

The structural chart of the Ovarian cancer prediction model is depicted in the figure 4.1. The prediction model is composed of 4 modules, namely- the data acquisition module, the preprocessing module, the feature extraction module and then, the classification module. This constitutes the complete structure of the system, which specifies the modules that are to be considered during the implementation phase of the project.

4.2 Detail description of each module

This part of the report includes the flowchart of each individual module used to develop the model to predict the ovarian cancer

4.2.1 The flowchart for Data Acquisition

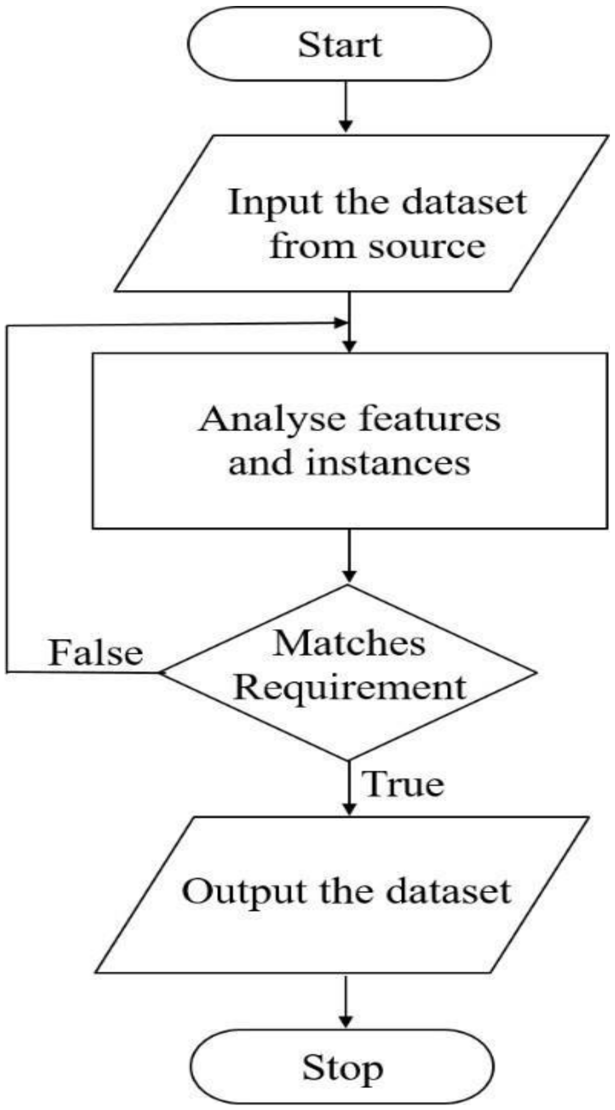


Figure 4.2: Flowchart for data acquisition.

The flowchart for collecting data is as depicted in the figure 4.2. The data set is collected from a source and a complete analysis is carried out. The image is selected to be used for training/testing purposes only if it matches our requirements and is not repeated.

4.2.2 Flowchart for Pre-Processing the Data Set

The figure 4.3 shows the flowchart for the pre-processing of the images received from the output of the previous step. This involves converting the image from the RGB format to greyscale to ease processing, the use of an averaging filter to filter out the noise, global basic thresholding to remove the background and consider consider only the image and a high-pass filter to sharpen the image by amplifying the finer details.

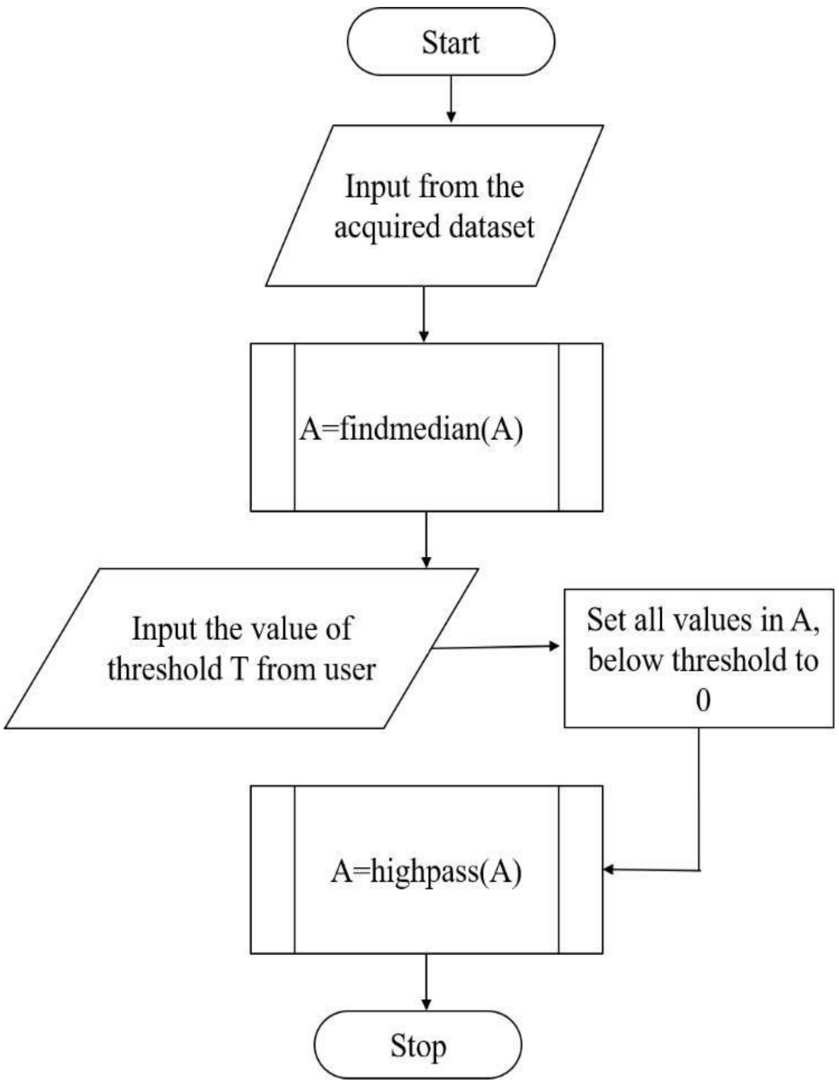
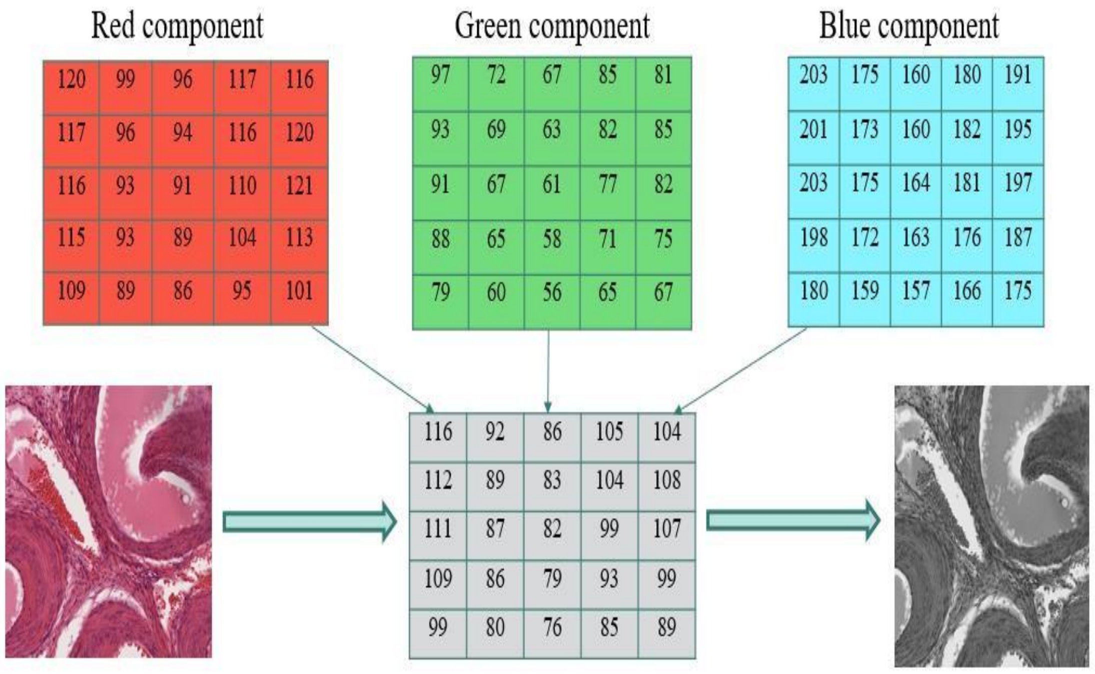

Start
Input from the acquired dataset
A=findmedian(A)
Input the value of threshold T from user
Set all values in A, below threshold to 0
A=highpass(A)
Stop

Figure 4.3: Flowchart for the preprocessing module.

Conversion from RGB to Greyscale

The first step in pre-processing is converting the image from RGB to Greyscale.It can be obtained by applying the below formula (4.1) to the RGB image.The figure 4.4 depicts the Conversion from RGB to grayscale. Converting an RGB (Red, Green, Blue) image to grayscale is a common preprocessing step in image processing and computer vision tasks. Grayscale images have only one channel, representing intensity or brightness, as opposed to the three channels (red, green, and blue) in RGB images. Converting an RGB image to grayscale is often done to simplify data representation and reduce computational complexity. Grayscale images, containing only intensity information, are memory-efficient and facilitate faster processing in various computer vision tasks. This conversion is particularly beneficial when color details are irrelevant, and tasks focus on overall brightness or structure, such as edge detection or facial recognition. By discarding color channels, grayscale images equalize the significance of all pixel values, contributing to algorithm robustness. While grayscale conversion is a common practice for simplifying analysis and resource efficiency, it is essential to consider the specific requirements of the task, as certain applications, like color based recognition, may necessitate preserving the RGB color information.

Formula: 0.2989*R+0.5870*G+0.1140*B ………………………………(4.1)



Red component

120	99	96	117	116
117	96	94	116	120
116	93	91	110	121
115	93	89	104	113
109	89	86	95	101

Green component

97	72	67	85	81
93	69	63	82	85
91	67	61	77	82
88	65	58	71	75
79	60	56	65	67

Blue component

203	175	160	180	191
201	173	160	182	195
203	175	164	181	197
198	172	163	176	187
180	159	157	166	175

116	92	86	105	104
112	89	83	104	108
111	87	82	99	107
109	86	79	93	99
99	80	76	85	89

Figure 4.4: Conversion of 5x5 RGB Matrix to 5x5 Gray matrix

Advantages of converting RGB color space to gray

- To store a single-color pixel of an RGB color image we will need 8*3 = 24 bits (8 bit for each color component).
- Only 8 bit is required to store a single pixel of the image. So we will need 33 % less memory to store grayscale image than to store an RGB image.
- Gray scale images are much easier to work within a variety of task like In many morphological operation and image segmentation problem, it is easier to work with single layered image (Gray image) than a three-layered image (RGB color image).
- It is also easier to distinguish features of an image when we deal with a single layered image.

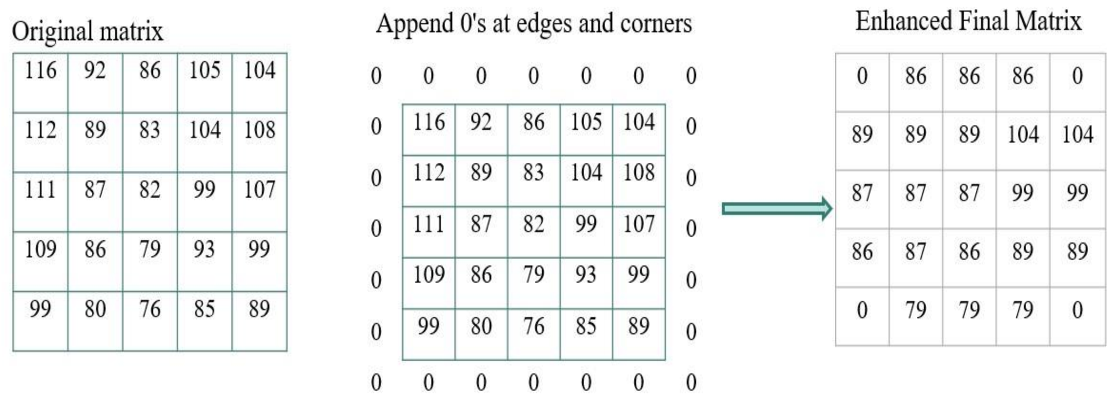
Noise Removal

Noise removal algorithm is the process of removing or reducing the noise from the image. The noise removal algorithms reduce or remove the visibility of noise by smoothing the entire image leaving areas near contrast boundaries. Noise removal is the second step in image pre-processing. Here the grayscale image which was obtained in the previous step is given as input. Here we are making use of Median Filter which is a Noise Removal Technique.

- **Median Filtering**

The median filter is a non-linear digital filtering technique, often used to remove noise from an image or signal. The main advantage of median filtering lies in its ability to preserve edges and fine details while effectively reducing the impact of outlier pixel values. This is because the median is less sensitive to extreme values than the mean. In the context of salt-and-pepper noise, where isolated bright or dark pixels are likely to be outliers, the median filter can efficiently eliminate these outliers without significantly affecting the overall structure of the image.

However, it's important to consider that median filtering may not perform as well in reducing Gaussian or uniform noise, as it is specifically designed for impulse noise like salt and pepper noise. Additionally, the size of the filtering window (also known as the kernel size) is a parameter that needs to be carefully chosen. Larger windows provide more smoothing but may blur fine details.



Original matrix

116	92	86	105	104
112	89	83	104	108
111	87	82	99	107
109	86	79	93	99
99	80	76	85	89

Append 0's at edges and corners

0	0	0	0	0	0	0
0	116	92	86	105	104	0
0	112	89	83	104	108	0
0	111	87	82	99	107	0
0	109	86	79	93	99	0
0	99	80	76	85	89	0
0	0	0	0	0	0	0

Enhanced Final Matrix

0	86	86	86	0
89	89	89	104	104
87	87	87	99	99
86	87	86	89	89
0	79	79	79	0

Figure 4.5: Noise filtering using Median Filter.

Here 0‟s are appended at the edges and corners to the matrix which is the representation of the grey scale image. Then for every3*3 matrix, arrange elements in ascending order, then find median/middle element of those 9 elements , and write that median value to that particular pixel position. The figure 4.5 depicts Noise filtering using Median Filter.

- **Canny Edge Detection**

  The Canny edge detector is an edge detection operator that uses a multi-stage algorithm to detect a wide range of edges in images as shown in figure 4.6. Canny edge detection is a technique to extract useful structural information from different vision objects and dramatically reduce the amount of data to be processed

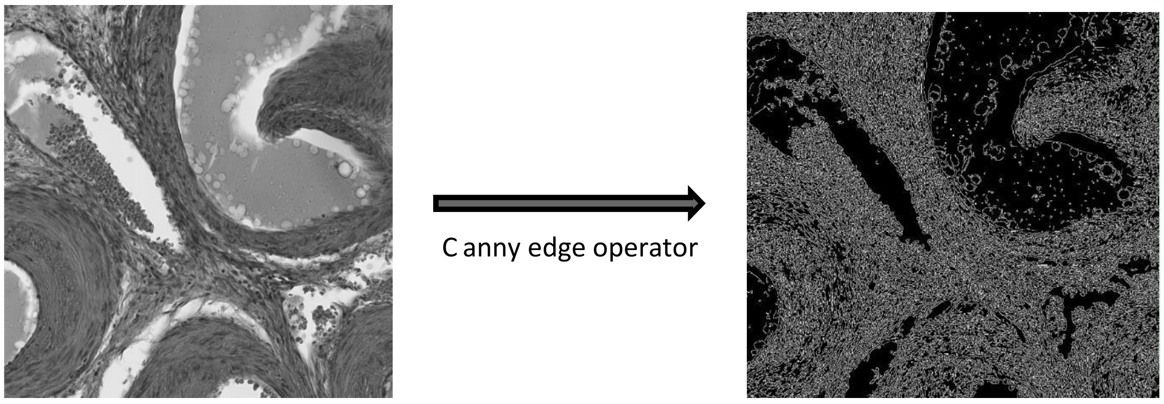
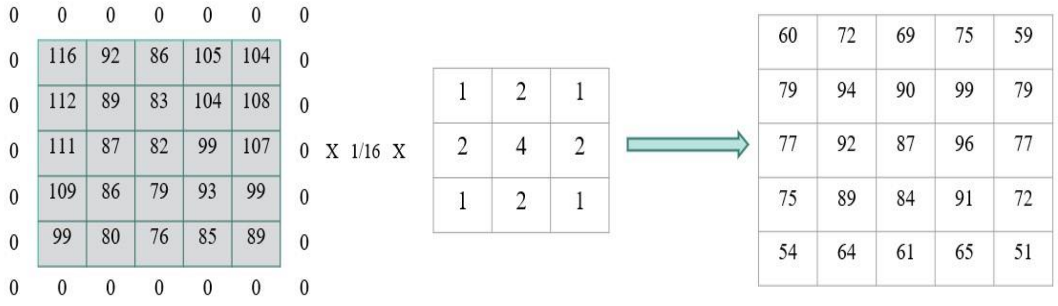


Figure 4.6: Canny Edge Detection Operation

- **Gaussian Filtering**

The Gaussian smoothing in the Canny edge detector fulfills two purposes they are It can be used to control the amount of detail that appears in the edge image and it can be used to suppress noise. The gaussian filter shown in figure 4.7 is a linear digital filtering technique, often used to remove noise from an image or signal. Here 0’s are appended at the edges and corners to the matrix which is the representation of the grey scale image. The resultant matrix obtained in this filter is used for cell counting.



$$\begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 116 & 92 & 86 & 105 & 104 & 0 \\ 0 & 112 & 89 & 83 & 104 & 108 & 0 \\ 0 & 111 & 87 & 82 & 99 & 107 & 0 \\ 0 & 109 & 86 & 79 & 93 & 99 & 0 \\ 0 & 99 & 80 & 76 & 85 & 89 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{matrix} \times 1/16 \times \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} 60 & 72 & 69 & 75 & 59 \\ 79 & 94 & 90 & 99 & 79 \\ 77 & 92 & 87 & 96 & 77 \\ 75 & 89 & 84 & 91 & 72 \\ 54 & 64 & 61 & 65 & 51 \end{bmatrix}$$

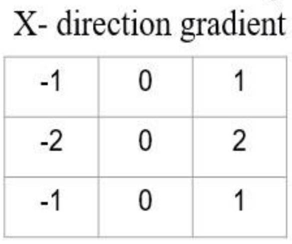
Figure 4.7: Noise Filtering using Gaussian Filter

Gradient Calculations - Sobel Gradient

A gradient-based method that looks for strong changes in the first derivative of an image. The Sobel edge detector uses a pair of 3 × 3 convolution masks, one estimating the gradient in the x-direction and the other in the y-direction**.** The kernels can be applied separately to the input image, to produce separate measurements of the gradient component in each orientation (call these *Gx* and *Gy*). These can then be combined together to find the absolute magnitude of the gradient at each point and the orientation of that gradient.

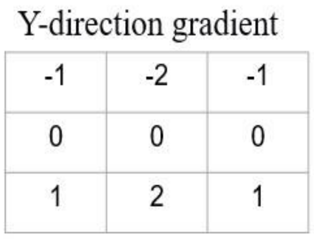
The gradient magnitude is given by: $|G| = \sqrt{Gx^2 + Gy^2}$ …………………………. (4.2)

The angle of orientation : $\varnothing = \tan^{-1}(Gy/Gx)$ ……….…………………………..(4.3)



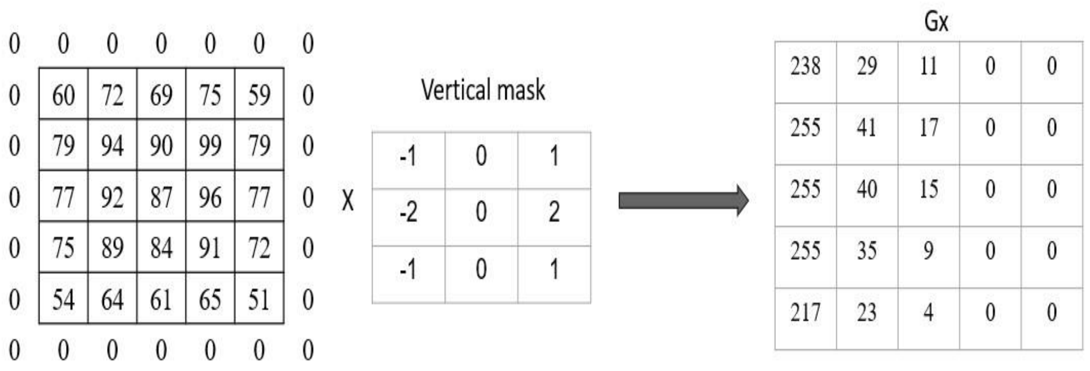
X- direction gradient

-1	0	1
-2	0	2
-1	0	1



Y-direction gradient

-1	-2	-1
0	0	0
1	2	1



0	0	0	0	0	0	0
0	60	72	69	75	59	0
0	79	94	90	99	79	0
0	77	92	87	96	77	0
0	75	89	84	91	72	0
0	54	64	61	65	51	0
0	0	0	0	0	0	0

X

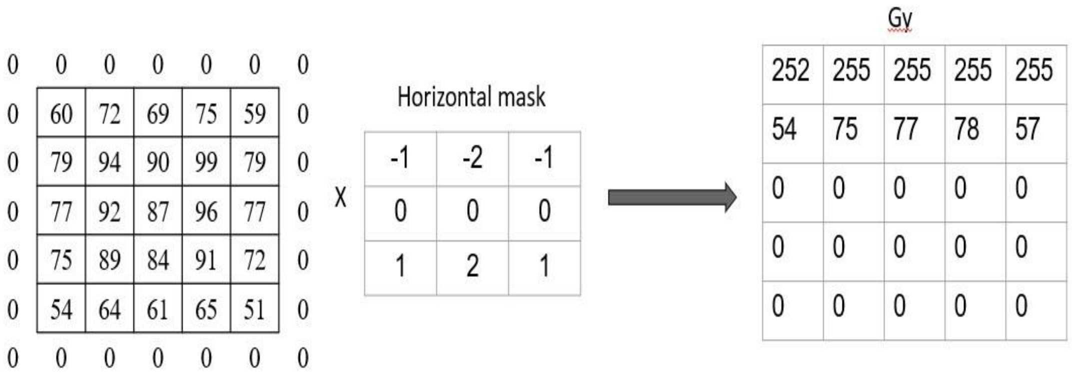
Vertical mask

-1	0	1
-2	0	2
-1	0	1

Gx

238	29	11	0	0
255	41	17	0	0
255	40	15	0	0
255	35	9	0	0
217	23	4	0	0

Figure 4.8: Vertical Mask Operator



0	0	0	0	0	0	0
0	60	72	69	75	59	0
0	79	94	90	99	79	0
0	77	92	87	96	77	0
0	75	89	84	91	72	0
0	54	64	61	65	51	0
0	0	0	0	0	0	0

X

Horizontal mask

-1	-2	-1
0	0	0
1	2	1

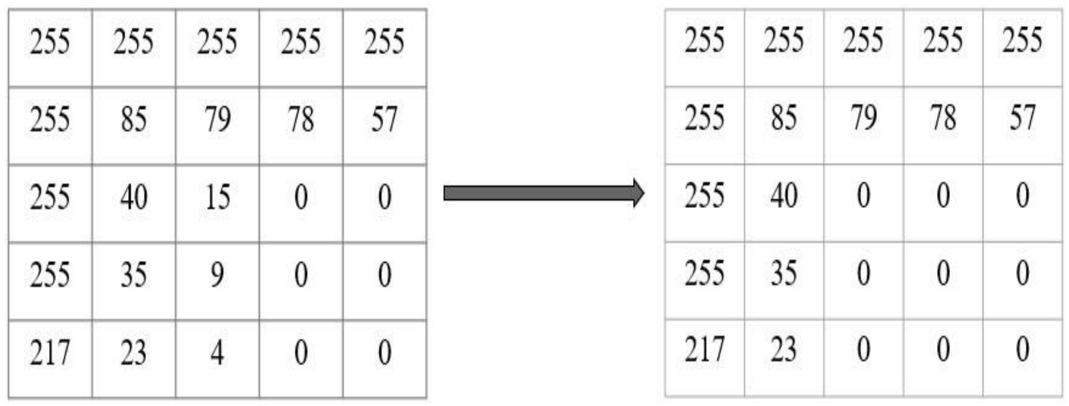
Gy

252	255	255	255	255
54	75	77	78	57
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

Figure 4.9: Horizontal Mask Operator

Double Thresholding

Introduce two thresholds: a high threshold (T_high) and a low threshold (T_low). Pixels with gradient magnitudes above the high threshold are classified as strong edges. Pixels with gradient magnitudes below the low threshold are classified as non-edges. Pixels with gradient magnitudes between the low and high thresholds are classified as weak edges.

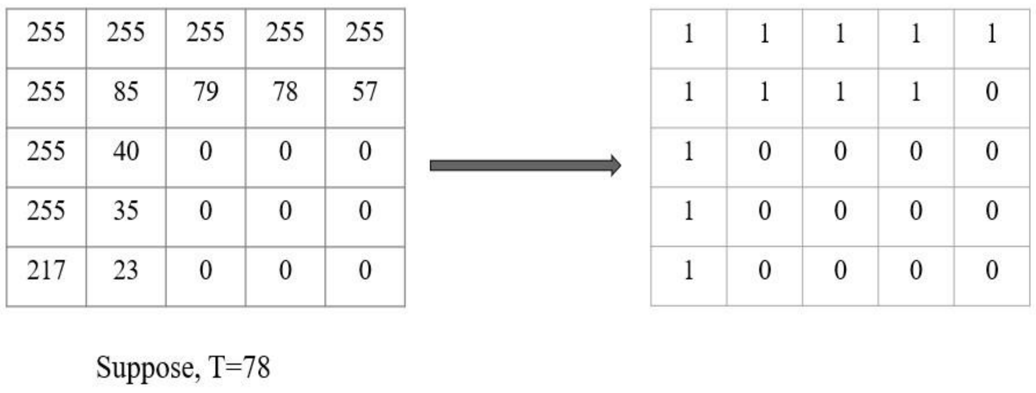


255	255	255	255	255
255	85	79	78	57
255	40	15	0	0
255	35	9	0	0
217	23	4	0	0

255	255	255	255	255
255	85	79	78	57
255	40	0	0	0
255	35	0	0	0
217	23	0	0	0

Figure 4.10: Double Thresholding

Grayscale Image to Binary Image Conversion



255	255	255	255	255
255	85	79	78	57
255	40	0	0	0
255	35	0	0	0
217	23	0	0	0

1	1	1	1	1
1	1	1	1	0
1	0	0	0	0
1	0	0	0	0
1	0	0	0	0

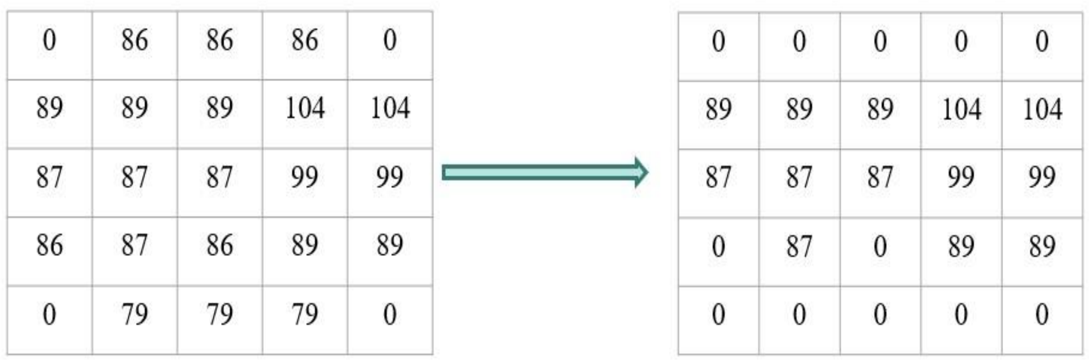
Suppose, T=78

Figure 4.11: Grey scale to Binary Conversion

Contour detection typically requires a binary image as input, where the objects are represented by white pixels (foreground) and the background by black pixels. This simplification makes the contour detection process more robust and efficient. Thresholding is the simplest method of image segmentation and the most common way to convert a grayscale image to a binary image, In thresholding, we select a threshold value and then all the gray level value which is below the selected threshold value is classified as 0(black i. e background) and all the gray level which is equal to or greater than the threshold value are classified as 1(white i. e foreground)

Basic Global Thresholding

Thresholding is a fundamental image processing technique used to segment an image into regions or objects based on pixel intensity values. The goal is to distinguish between pixels that are part of the objects of interest and those that are not. The process involves setting a threshold value, and pixels in the image are categorized as either foreground (object) or background based on whether their intensity is above or below the threshold. Considering each pixel values, if the pixel value is greater than or equal to the threshold T, retain it. Else, replace the value by 0. Here, the value of t can be manipulated in the frontend, to suit the varying needs of different images we consider T=87. Consider the matrix obtained from the median filtering.



0	86	86	86	0
89	89	89	104	104
87	87	87	99	99
86	87	86	89	89
0	79	79	79	0

0	0	0	0	0
89	89	89	104	104
87	87	87	99	99
0	87	0	89	89
0	0	0	0	0

Figure 4.12: Thresholding using Basic global Thresholding.

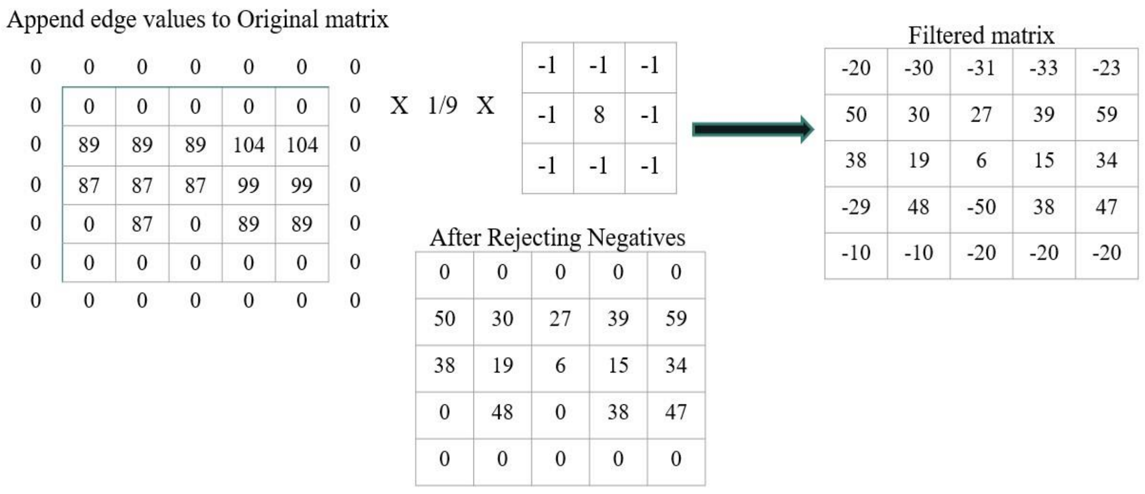
Image Sharpening

Image sharpening refers to any enhancement technique that highlights edges and fine details in an image, Increasing yields a more sharpened image.

- **High-Pass Filtering-1**

A high-pass filter can be used to make an image appear sharper. These filters emphasize fine details in the image. Here the output from the thresholding is given as input. Here, we are making use of a filter, First we append the nearest values to pixels at the boundary pixels. The figure 4.13 depicts Image Sharpening using High-Pass Filter

We multiply the elements of the 3*3 input matrix with the filter matrix, this can be represents as A(1,1)*B(1,1), in this way all the elements in the 3*3 are multiplied and their sum id divided by 9, which gives the value for the particular pixel position. In the same way the values of all the pixel positions are calculated. The negative values are considered as zero, as there can be no such thing as negative illumination.


Append edge values to Original matrix

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	89	89	89	104	104	0
0	87	87	87	99	99	0
0	0	87	0	89	89	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

X 1/9 X

-1	-1	-1
-1	8	-1
-1	-1	-1

Filtered matrix

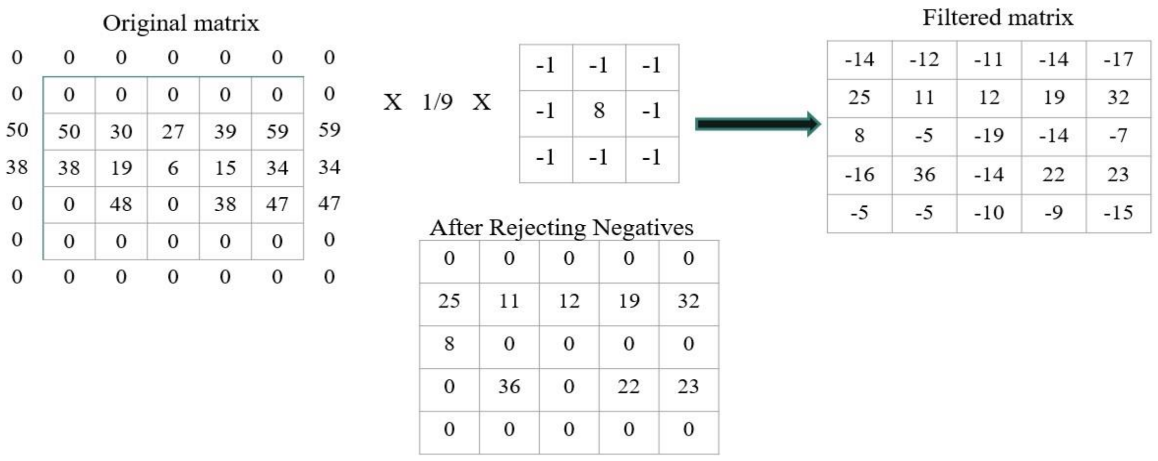
-20	-30	-31	-33	-23
50	30	27	39	59
38	19	6	15	34
-29	48	-50	38	47
-10	-10	-20	-20	-20

After Rejecting Negatives

0	0	0	0	0
50	30	27	39	59
38	19	6	15	34
0	48	0	38	47
0	0	0	0	0

Figure 4.13: Image Sharpening using High-Pass Filter-1.

- **High-Pass Filtering-2**


Original matrix

0	0	0	0	0	0	0
0	0	0	0	0	0	0
50	50	30	27	39	59	59
38	38	19	6	15	34	34
0	0	48	0	38	47	47
0	0	0	0	0	0	0
0	0	0	0	0	0	0

X 1/9 X

-1	-1	-1
-1	8	-1
-1	-1	-1

Filtered matrix

-14	-12	-11	-14	-17
25	11	12	19	32
8	-5	-19	-14	-7
-16	36	-14	22	23
-5	-5	-10	-9	-15

After Rejecting Negatives

0	0	0	0	0
25	11	12	19	32
8	0	0	0	0
0	36	0	22	23
0	0	0	0	0

Figure 4.14: Image Sharpening using High-Pass Filter-2.

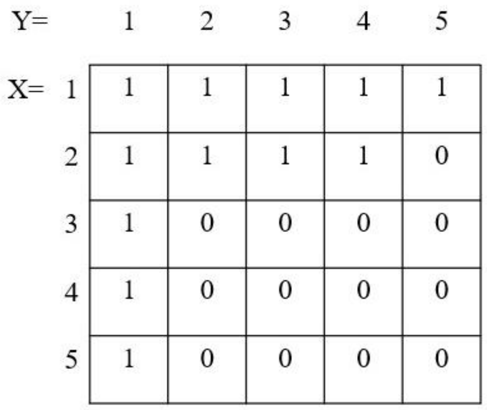
4.2.3 Overlapped Cell Count – Centroid Calculations

For binary images centroid calculations are often based on binary moments. Binary moments are used to compute various geometric properties of binary images, including the centroid. Given a binary image where each pixel is either 0 (background) or 1 (object), and each pixel has coordinates (x, y), the centroid (Cx, Cy) . can be calculated using the following formulas:

Cx=M10/M00 **Cy=M01/M00** **……………………………………**(4.4) where: M00 is the zero-order moment, representing the total area of the object,M10 is the firstorder moment in the x-direction, and M01is the first-order moment in the y-direction. The moments are calculated as follows:

$$\boldsymbol{M_{ij}=\sum_x\sum_y x^i * y^j . \text{binaryImage}(x,y)} \qquad \ldots\ldots\ldots\ldots.(4.5)$$

where: The sums are over all pixel coordinates (x, y) in the binary image, *i* and *j* are the orders of the moments.



Y=	1	2	3	4	5
X= 1	1	1	1	1	1
2	1	1	1	1	0
3	1	0	0	0	0
4	1	0	0	0	0
5	1	0	0	0	0

Figure 4.15: Binary Matrix for Centroid Calculation

- Find M00

1st iteration :1+1+1+1+1=5

2nd iteration:1+1+1+1+0=4

3rd iteration:1+0+0+0+0=1

4th iteration: 1+0+0+0+0=1

5th iteration: 1+0+0+0+0=1

M00= 12

- Find M10

1st iteration :1(1+1+1+1+1)=5

2nd iteration:2(1+1+1+1+0)=8

3rd iteration:3(1+0+0+0+0)=3

4th iteration: 4(1+0+0+0+0)=4

5th iteration: 5(1+0+0+0+0)=5

M10 = 25

- Find M01

1st iteration :1(1+1+1+1+1)=5

2nd iteration:2(1+1+0+0+0)=4

3rd iteration:3(1+1+0+0+0)=6

4th iteration: 4(1+1+0+0+0)=8

5th iteration:5(1+0+0+0+0)=5

M01 = 28

Cx = M10/M00 = 25 / 12 = 2.083

Cy = M01/M00 = 28 / 12 = 2.33

Distance Between Centroids

calculate the Euclidean distance between the centroids of the current pair of contours using the formula: **distance=** $\sqrt{(cxi-cxj)^2+(cyi-cyj)^2}$ ………………………….. (4.6)

If the calculated distance is below a certain threshold (20 in this case), it is considered that the corresponding cells are overlapping. It's a simple approach to identify instances where cells in the image are close to each other, and their centroids are within a specified distance.

$\sqrt{(cxi-cxj)^2+(cyi-cyj)^2} = \sqrt{2.083^2+ 2.33^2} = 3.125$

3.125 < 20 (therefore increment overlapping cell count.)

4.2.4 Flowchart for Classification

In the final phase of classification, Convolutional Neural Networks (CNNs) are employed to categorize leaf images into either healthy or diseased classes based on distinct disease types. CNNs operate by taking the output generated from a high-pass filter as input, bypassing the need for separate feature extraction. Within the CNN architecture, feature extraction is intrinsic, facilitated through convolution, rectification, and pooling operations performed iteratively. These sub-modules collectively process the input data to generate a comparison matrix. Subsequently, classification algorithms like the Softmax classifier are applied to interpret this matrix and assign the input image to one of the predefined classes based on the learned features and patterns extracted by the CNN model.

Classification using Convolutional Neural Networks

In deep learning, a convolutional neural network (CNN, or ConvNet) is a class of deep neural network most commonly applied to analyzing visual imagery. They are also known as shift invariant or space invariant artificial neural networks (SIANN),based on their shared-weights architecture and translation invariance characteristics

When CNN is used for classification we don‟t have to do feature extraction. Feature Extraction will also be carried out by CNN. We feed the preprocessed image directly to CNN classifier to obtain the stage of cancer if present. Flowchart for classification using CNN is shown in figure 4.16**.**

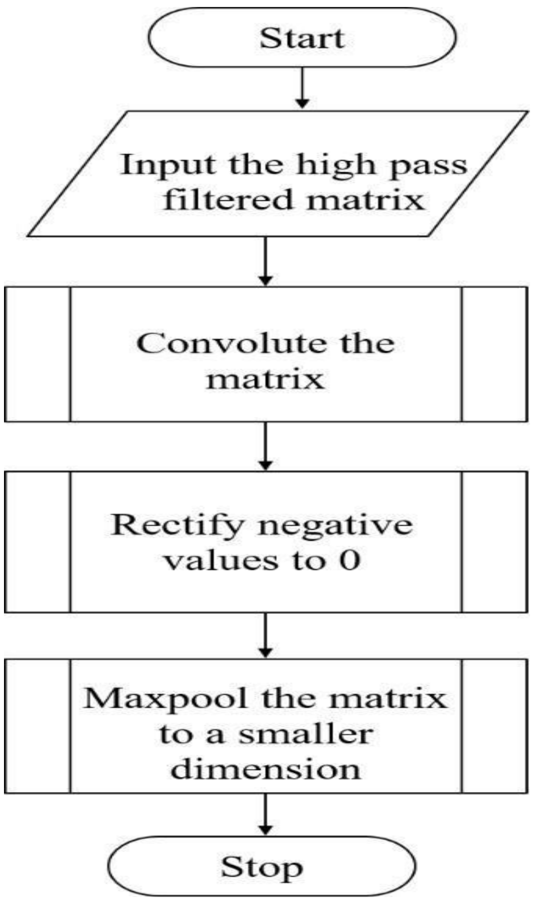

Start
Input the high pass filtered matrix
Convolute the matrix
Rectify negative values to 0
Maxpool the matrix to a smaller dimension
Stop

Figure 4.16: Flowchart for classification using CNN.

Typical CNN Architecture

CNN architecture is inspired by the organization and functionality of the visual cortex and designed to mimic the connectivity pattern of neurons within the human brain. The neurons within a CNN are split into a three-dimensional structure, with each set of neurons analyzing a small region or feature of the image.

In other words, each group of neurons specializes in identifying one part of the image. CNNs use the predictions from the layers to produce a final output that presents a vector of probability scores to represent the likelihood that a specific feature belongs to a certain class. Figure 4.17 shows the Typical CNN Architecture

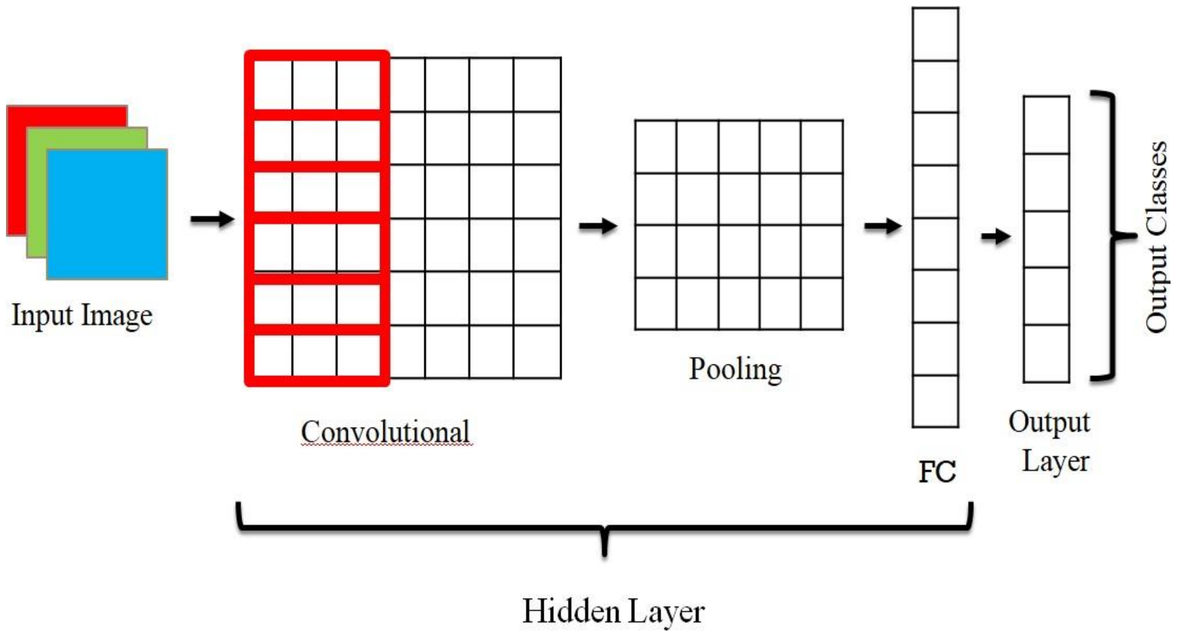
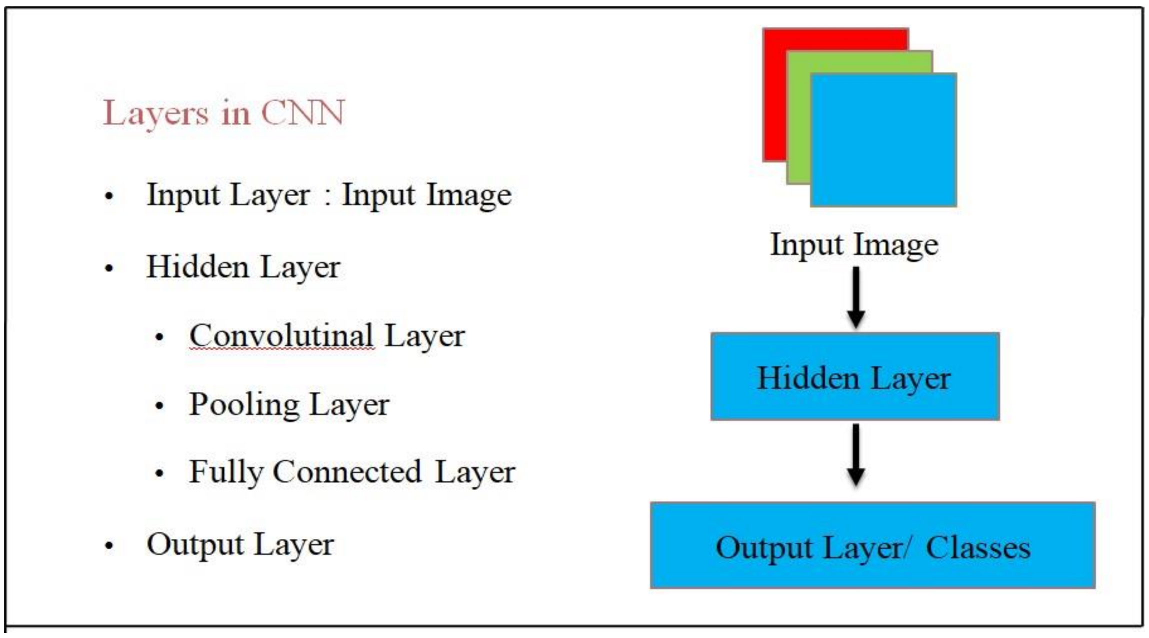


Figure 4.17: Typical CNN Architecture.

A CNN is composed of several kinds of layers:


Layers in CNN

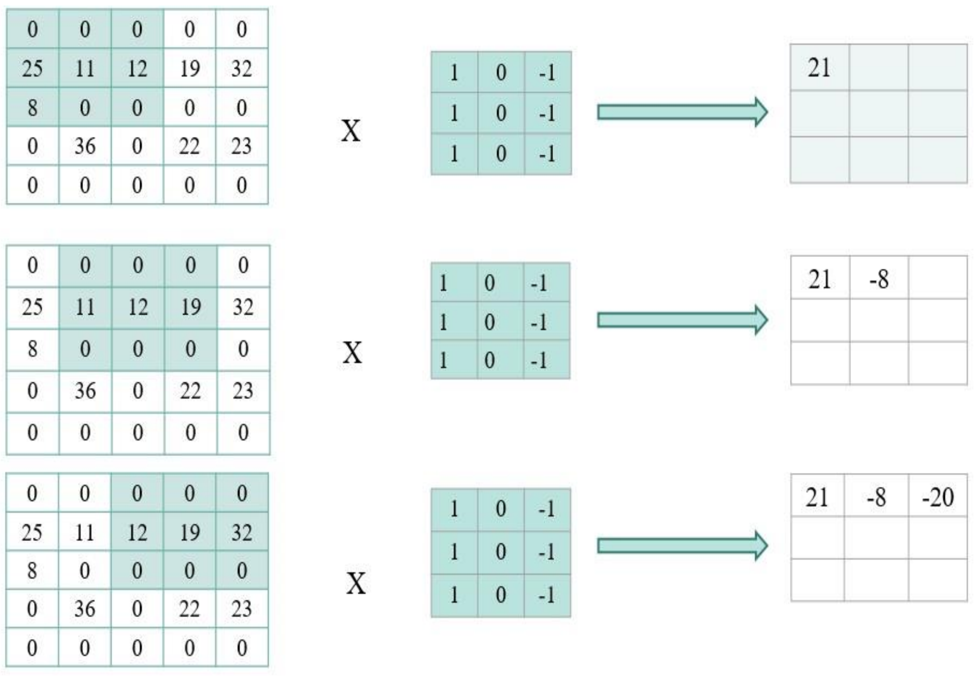
- Input Layer : Input Image
- Hidden Layer
  - Convolutinal Layer
  - Pooling Layer
  - Fully Connected Layer
- Output Layer

Figure 4.18: Layers in CNN.

- **Convolutional layer**-creates a feature map to predict the class probabilities for each feature by applying a filter that scans the whole image, few pixels at a time.
- **Pooling layer (downsampling)**-scales down the amount of information the convolutional layer generated for each feature and maintains the most essential information (the process of the convolutional and pooling layers usually repeats several times).
- **Fully connected layer**-"flattens" the outputs generated by previous layers to turn them into a single vector that can be used as an input for the next layer. Applies weights over the input generated by the feature analysis to predict an accurate label.
- **Output layer**-generates the final probabilities to determine a class for the image.

Convolutional Layer

Convolutional Layer is the first step in CNN, here 3*3 part of the given matrix which was obtained from High-pass filter is given as input. That 3*3 matrix is multiplied with the filter matrix for the corresponding position and their sum is written in the particular position. This is shown in the below figure. This output is given to pooling layer where the matrix is further reduced.Figure 4.19 shows the Convolutional Layer.



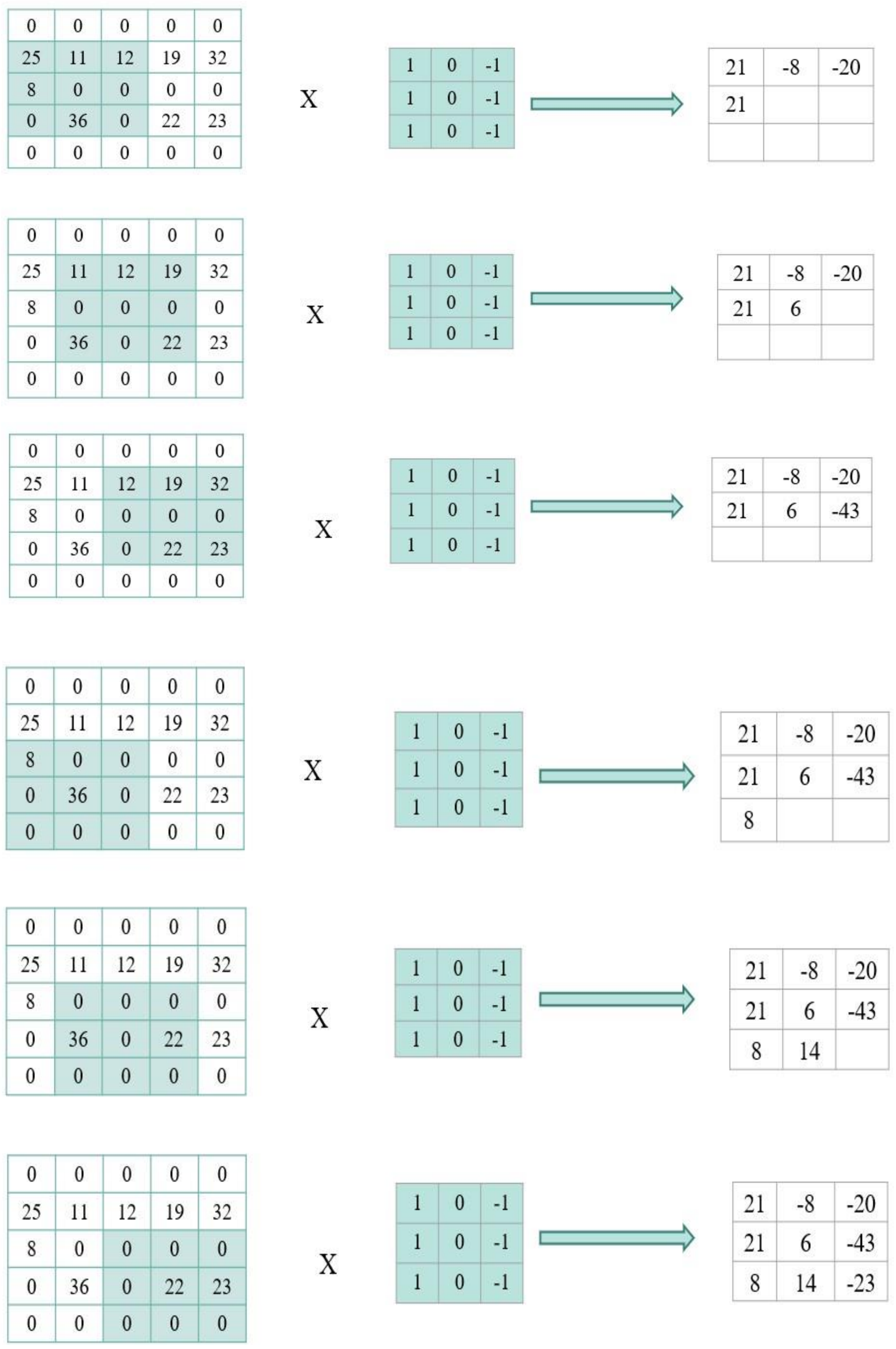


Figure 4.19: Convolutional Layer.

Convolution is followed by the rectification of negative values to 0s, before pooling. Here, it is not demonstratable, as all values are positive. In fact, multiple iterations of both are needed before pooling.

Pooling Layer

In Pooling layer 3*3 matrix is reduced to 2*2 matrix, this is done by selecting the maximum of the particular 2*2 matrix for the particular position. Figure 4.20 shows the Pooling Layer. Scales down the amount of information the convolutional layer generated for each feature and maintains the most essential information.

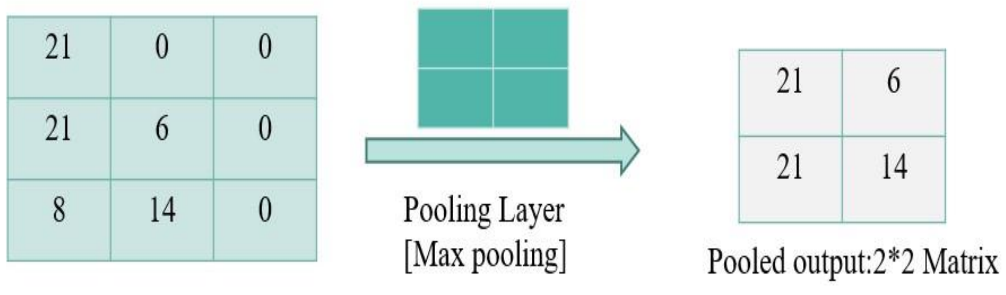
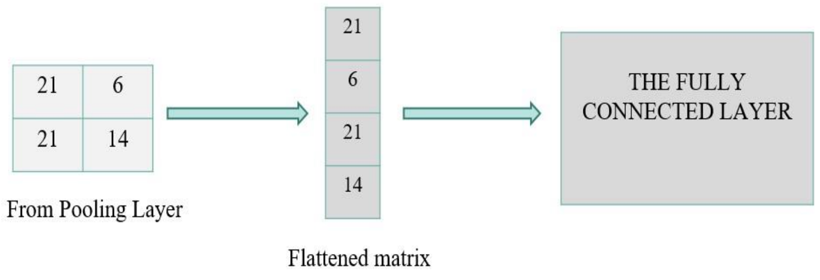


Figure 4.20: Pooling Layer.

Fully connected layer and Output Layer



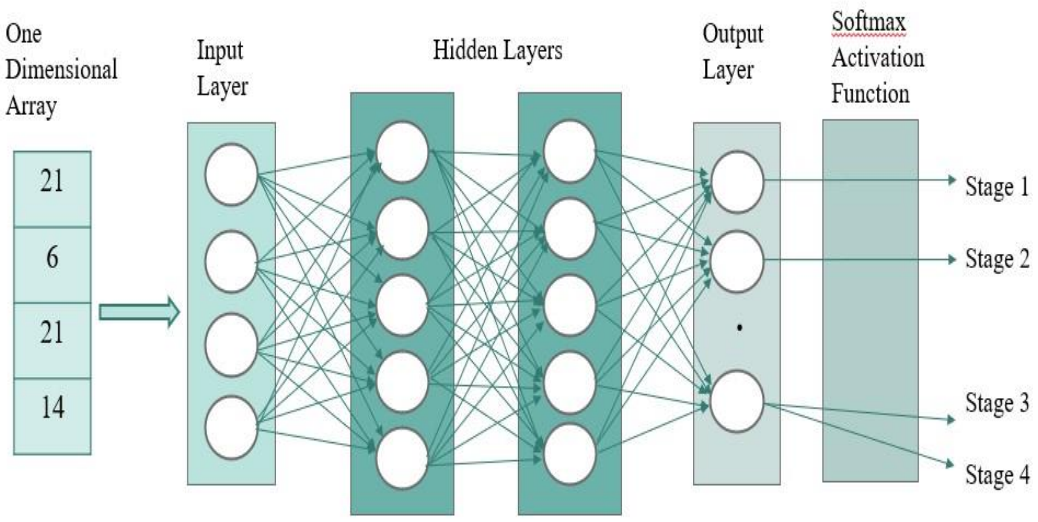


Figure 4.21: Fully connected layer and Output Layer.

The figure 4.21 is a diagram of a simple neural network with one hidden layer. The network in the image is a feedforward neural network, which means that information flows in one direction only, from the input layer to the output layer. The input layer consists of a single neuron, which takes a one-dimensional array of 21 numbers as input.The output layer consists of a single neuron, which outputs a single number between 0 and 1. This number can be interpreted as the probability of the input belonging to a particular class.

The output of the pooling layer is flattened and this flattened matrix is fed into the Fully Connected Layer. In the fully connected layer there are many layers, Input layer, Hidden layer and Output layers are parts of it. Then this output is fed into the classifier, in this case Softmax Activation Function is used to classify the image into healthy or a leaf with a particular disease if present. Figure 4.21 shows the Fully connected layer and Output Layer

SoftMax activation function

The SoftMax activation function is a nonlinear function that takes a vector of real numbers.Makes the function useful for tasks such as multi-class classification, where the output of the network needs to be a probability distribution over a set of possible classes.

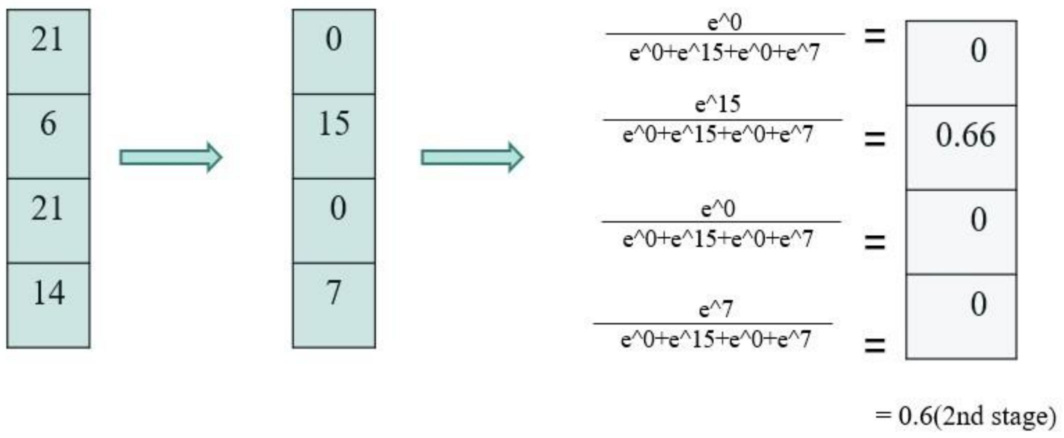


Figure 4.22: SoftMax Activation Function

- **Softmax Function:** The equation softmax(zi)=*j*=1∑Kezjezi represents the softmax function. It takes an input vector *z* with K values, zi, and returns a K-dimensional vector of probabilities where each element lies between 0 and 1.
- **Input vector (z):** The input vector, *z* , consists of K real numbers. In the example, zi is represented by the numbers 21, 15, 0, and 7.
- **Exponentiation:** The softmax function first applies an exponential function (e ^ x) to each element (zi) in the input vector z. This ensures all the output values are positive.

- **Normalization:** The exponentiated values are then summed together. The softmax function then divides each element-wise by this sum. This normalization step ensures that the output values add up to 1, transforming them into a valid probability distribution.
- **Interpretation of the Output:**The softmax output assigns a probability score to each class, indicating how likely the input belongs to that particular class. In the image, the input vector, z = [21, 15, 0, 7], is transformed into a probability distribution: [0, 0.66, 0, 0.34]. Here, the second class has the highest probability (0.66) of containing the input.

4.3 Summary

Chapter 4 describes the crucial detailed design phase preceding the implementation of the Ovarian Cancer Prediction Model. This phase refines the preliminary system design, offering comprehensive details pivotal for efficient execution, thereby saving time. The structural chart provides an overview of modules such as data acquisition, preprocessing, feature extraction, and classification, delineating the system's architecture. Each module is meticulously described with accompanying flowcharts, elucidating processes like RGB to grayscale conversion and edge detection. The classification stage utilizes Convolutional Neural Networks (CNNs) with the SoftMax activation function, emphasizing multi-class classification. A comprehensive flowchart illustrates the CNN-based classification process. This meticulous design approach aims to develop a robust model for early ovarian cancer detection, thereby contributing to improved treatment outcomes and patient prognosis. By integrating advanced image processing techniques and leveraging CNNs, the project endeavors to enhance accuracy and reliability in cancer prediction.

CHAPTER 5
IMPLEMENTATION

Implementation is the platform where the project working is demonstrated. In implementation the system drives for real operations. Every consequence completed by the project will be abundant, if processes are accurately executed according to plan carried out.

5.1 Implementation Requirements

To implement ovarian cancer orediction using Convolution Neural Network, software's used are

- Language used to code the project is Python.
- Operating System-Windows 10.

5.5.1 Visual Studio Code

Visual Studio Code is a freeware source-code editor made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality. Visual Studio Code is a source-code editor that can be used with a variety of programming languages, including Java, JavaScript, Go, Node.js, Python and C++. It is based on the Electron framework,[19] which is used to develop Node.js Web applications that run on the Blink layout engine. Visual Studio Code employs the same editor component (codenamed "Monaco") used in Azure DevOps (formerly called Visual Studio Online and Visual Studio Team Services).

Instead of a project system, it allows users to open one or more directories, which can then be saved in workspaces for future reuse. This allows it to operate as a language-agnostic code editor for any language. It supports a number of programming languages and a set of features that differs per language. Unwanted files and folders can be excluded from the project tree via the settings. Many Visual Studio Code features are not exposed through menus or the user interface but can be accessed via the command palette.

5.1.2 Programming Language Used

Programming language used to design the proposed method is Python. Python is a high-level programming language with dynamic semantics. It is an interpreted language i.e.

interpreter executes the code line by line at a time, thus makes debugging easyPython Imaging Library (PIL) is one of the popular libraries used for image processing. PIL can be used to display image, create thumbnails, resize, rotation, convert between file formats, contrast enhancement, filter and apply other digital image processing techniques etc. Python is often used as a support language for software developers, for build control and management, testing, and in many other ways. Python is designed by Guido van Rossum. It is very easy for user to learn this language because of its simpler coding.

It provides an easy environment to furnish computation, programming visualization. Python supports modules and packages, which encourages program modularity and code reuse. It has various built-in commands and functions which will allow the user to perform functional programming. Apart from being an open-source programming language, developers use it extensively for application development and system development programming. It is a highly extensible language. Python contains many inbuilt functions which helps beginners to learn easily.

Some of the most commonly used functions are: imread will read image from specified location, imshow will display the output or images on the screen, cvtColor function converts binary image into grayscale image. Compare_ssim function will compare pre-flood and post-flood images, findContours will give the region of difference between two input images.

Key Features of Python

- Python is an interpreted language i.e. interpreter executes the code line by line at a time, thus makes debugging easy.
- Python is more expressive language, since it is more understandable and readable.
- To design and solve problems it offers an interactive atmosphere.
- Python has a large and broad library and provides rich set of module and functions for rapid application development.
- Built in graphics for conception of data and tools is also supported. It can be easily integrated with languages like C, C++, JAVA etc.

TKinter GUI

Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is the most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter is the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.User need not want to create script or write commands in command prompt. Instead user must be aware of how the programs are performs to accomplish tasks. It includes Radio buttons, Toolbars, Sliders, Axes etc.

OpenCV-Python Tool

OpenCV-Python is a library of Python bindings designed to solve computer vision problems. Visual information is the most important type of information perceived, processed and interpreted by the human brain. Image processing is a method to perform some operations on an image, in order to extract some useful information from it.

An image is nothing more than a two dimensional matrix (3-D in case of colored images) which is defined by the mathematical function f(x,y) where x and y are the two co-ordinates horizontally and vertically. The value of f(x,y) at any point is gives the pixel value at that point of an image, the pixel value describes how bright that pixel is, and/or what color it should be. In image processing we can also perform image acquisition, storage, image enhancement.

Flask

Flask (source code) is a Python web framework built with a small core and easy- to-extend philosophy. Flask is a lightweight WSGI web application framework. It is designed to make getting started quick and easy, with the ability to scale up to complex applications. It began as a simple wrapper around Werkzeug and Jinja and has become one of the most popular Python web application frameworks.

Flask offers suggestions, but doesn't enforce any dependencies or project layout. It is up to the developer to choose the tools and libraries they want to use. There are many extensions provided by the community that make adding new functionality easy.

Flask is considered more Pythonic than the Django web framework because in common situations the equivalent Flask web application is more explicit. Flask is also easy to get started with as a beginner because there is little boilerplate code for getting a simple app up and running.

Flask was also written several years after Django and therefore learned from the Python community's reactions as the framework evolved. Jökull Sólberg wrote a great piece articulating to this effect in his experience switching between Flask and Django.

Google Colab

Colaboratory, or "Colab" for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education. More technically, Colab is a hosted Jupyter notebook service that requires no setup to use, while providing free access to computing resources including GPUs. Colab resources are not guaranteed and not unlimited, and the usage limits sometimes fluctuate. This is necessary for Colab to be able to provide resources for free.

Jupyter is the open source project on which Colab is based. Colab allows you to use and share Jupyter notebooks with others without having to download, install, or run anything.Code is executed in a virtual machine private to your account. Virtual machines are deleted when idle for a while, and have a maximum lifetime enforced by the Colab service. Resources in Colab are prioritized for users who have recently used less resources, in order to prevent the monopolization of limited resources by a small number of users.

Packages

Packages are the namespaces which consists of multiple packages and module themselves. Each package in Python is a directory which must contain a special file called _init_py. This file can be empty, and it indicates that the directory it contains is a Python package, so it can be imported in the same way that the module can be imported.

As our application program grows larger in size with a lot of modules, one can place the similar modules in one package and different modules in different packages. This makes a program easy to manage and conceptually clear. We can import the modules from packages using the dot (.) operator.

1. .CSV

In this work, to import or export spread sheets and databases for its use in the Python interpreter, the CSV module, or Comma Separated Values format is used.

These CSV files are used to store a large number of variables or data and the CSV module is a built-in function that allows Python to parse these types of files. The text inside a CSV file is organized in the form of rows, and each row consists of the columns, all separated by commas, indicates the separate cells in CSV file. There is no standard for CSV modules hence, these modules makes use of "dialects" to support parsing using different parameters.

The CSV module includes all the necessary built-in functions, csv.reader and csv.writer are the most commonly used built-in functions in Python. These functions are given below:

csv.reader (csvfile, dialect="excel", **fmtparams)

It will return a reader object which will iterate over lines in the given csvfile.

csvfile can be any object which supports the iterator protocol and returns a string each time its next() method is called – file objects and list objects are both suitable. If csvfile is a file object, it must be opened with the „b" flag on platforms where that makes a difference. An optional dialect parameter can be given which is used to define a set of parameters specific to a particular CSV dialect. It may be an instance of a subclass of the Dialect class or one of the strings returned by the list_dialects() function. The other optional fmtparams keyword arguments can be given to override individual formatting parameters in the current dialect. Each row read from the csv file is returned as a list of strings. No automatic data type conversion is performed.

csv.writer (csvfile, dialect="excel", **fmtparams)

It returns a writer object responsible for converting the user"s data into delimited strings on the given file-like object. csvfile can be any object with a write() method. If csvfile is a file object, it must be opened with the „b" flag on platforms where that makes a difference. An optional dialect parameter can be given which is used to define a set of dialect. It may be an instance of a subclass of the Dialect class or one of the strings returned by the list_dialects() function. The other optional fmtparams keyword arguments can be given to override individual formatting parameters in the current dialect. Floats are stringified with repr() before being written. All other non-string data are stringified with str() before being written.

2. Tensorflow

Tensorflow is a free and open source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural network, it is used for both research and production at Google.

Tensorflow was developed by the Google Brain team for internal Google use. It was released under the Apache License 2.0 on November 9, 2015. Tensorflow is Google Brain‟s second-generation system. Version 1.0.0 was released on February 11, 2017. While the reference implementation runs in single devices, Tensorflow can run on multiple CPUs and GPUs. Tensorflow is available on 64- bit Linux, macOS, Windows and mobile computing platforms including Android and ios. Its flexible architecture allows for the easy deployment of computation across a variety of platforms (CPUs, GPUs and TPUs), and from desktops to clusters to servers to mobile and edge devices. Tensorflow computations are expressed as state full dataflow graphs. The name tensorflow derive from the operations that such neural networks perform on multidimensional data arrays, which are referred to as tensors.

3. Keras

Keras is an open source neural- network library written in Python. It is capable of running on top of Tensorflow, Microsoft Cognitive Toolkit, R, Theano or PlaidML. It is designed to enable fast experimentation with deep neural networks. It focuses in being user-friendly, modular, and extensible CUDA. Keras contains numerous implementations of commonly used neural- network building blocks such as layers, objectives, activation function, optimizers, and a host of tools to make working with image and text data easier to simplify the code necessary for writing deep neural network code. The code is hosted on GitHub, and commonly support forums include the GitHub issues page, and a Slack channel. In addition to Standard neural networks, Keras has support for convolutional and recurrent neural networks. It supports other common utility layers like dropout, batch normalization and pooling. Keras allows users to productize deep models on smart phones, on the web or on the java virtual machine. It also allows use of distributed training of deep learning models on clusters of Graphics processing units (GPU) and tensor processing units (TPU) principally in conjunction with CUDA.

4. NumPy

NumPy is a python library used for working with arrays. It also has functions for working in domain of linear algebra, fourier transform, and matrices. NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely. NumPy stands for Numerical Python. In Python we have lists that serve the purpose of arrays, but they are slow to process. NumPy aims to provide an array object that is up to 50x faster than traditional Python lists. The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy. Arrays are very frequently used in data science, where speed and resources are very important. NumPy arrays are stored at one continuous place in memory unlike lists, so processes can access and manipulate them very efficiently. This behaviour is called locality of reference in computer science. This is the main reason why NumPy is faster than lists. Also, it is optimized to work with latest CPU architectures.

5. Matplotlib

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib produces publication-quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shell, web application servers, and various graphical user interface toolkits. Matplotlib. pyplot is a collection of command style functions that make matplotlib work like MATLAB. Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc.

Some people embed matplotlib into graphical user interfaces like wxpython or pygtk to build rich applications. Others use matplotlib in batch scripts to generate postscript images from some numerical simulations, and still others in web application servers to dynamically serve up graphs. Matplotlib is the brainchild of John Hunter (1968-2012), who, along with its many contributors, have put an immeasurable amount of time and effort into producing a piece of software utilized by thousands of scientists worldwide.

6. Sklearn

.Scikit-learn, commonly abbreviated as sklearn, is an open-source machine learning library for Python. It provides a wide array of tools for machine learning and statistical modeling, including classification, regression, clustering, dimensionality reduction, and more. Sklearn is widely used in the data science community due to its simplicity, versatility, and extensive documentation, making it accessible to both beginners and experienced practitioners alike.

Sklearn is utilized for a variety of purposes, including but not limited to data preprocessing, model training, evaluation, and deployment. It offers a consistent interface across different algorithms, making it easy to experiment with various models and techniques. Additionally, sklearn integrates seamlessly with other Python libraries such as

NumPy, SciPy, and pandas, enhancing its capabilities for data manipulation and analysis. Its comprehensive documentation and user-friendly interface make it a popular choice for researchers, educators, and professionals seeking to leverage machine learning techniques for solving real-world problems.

7. OS

The "os" module in Python is a built-in package that provides a means of interacting with the operating system. It offers various functions and methods for performing operating system-related tasks such as file and directory manipulation, process management, and environment variable access. Python's "os" module abstracts away platform-specific differences, allowing developers to write code that can seamlessly execute on different operating systems without modification. By utilizing the "os" module, Python programs gain the capability to navigate file systems, create or delete directories, check file permissions, execute system commands, and manage processes, thereby enabling robust and platform-independent system-level interactions within Python applications. Overall, the "os" module serves as a powerful tool for system-level operations, enhancing the versatility and functionality of Python programs across diverse computing environments.

8. Sqllite3

The sqlite3 module in Python provides a powerful and lightweight interface to the SQLite database engine. SQLite is a self-contained, serverless, zero-configuration, and transactional SQL database engine that is widely used in various applications due to its simplicity and efficiency. With the sqlite3 module, Python developers can seamlessly interact with SQLite databases, perform database operations such as creating tables, inserting data, querying data, and managing transactions directly from their Python code. SQLite databases are stored in a single file, making them easy to distribute and deploy with Python applications, and they offer features like ACID transactions, indexes, and triggers, making them suitable for small to medium-sized applications and prototyping.

The sqlite3 module is extensively used in Python applications for various purposes such as storing application data, caching information, logging, and more. It is particularly useful for lightweight desktop applications, mobile apps, and web applications where a full-fledged database management system may be overkill. Python developers can leverage SQLite databases for tasks such as managing user preferences, storing configuration settings, implementing simple data-driven features, and performing offline data storage. Overall, the sqlite3 module offers a convenient and efficient way to incorporate database functionality into Python applications without the need for external dependencies or complex setup procedures.

5.2 Pseudocodes

Pseudocode is a high-level, informal description of a computer program or algorithm written in plain language that closely resembles the structure of programming language code. It is used to outline the logic and steps of a solution algorithm without focusing on specific syntax or language rules. Pseudocode serves as a communication tool between developers, allowing them to express complex ideas and algorithms in a clear, human-readable format. It helps in planning, designing, and understanding algorithms before implementation, facilitating collaboration and problem-solving. Pseudocode typically uses common programming constructs such as loops, conditional statements, and function calls, enabling developers to express the flow of the program logically and concisely.

5.2.1 Pseudocode for Preprocessing Techniques

Pseudocode for Picture Uploading

Input image is obtained from the dataset which is hostopathological image .

```
Step 1: Click on Analyse.
Step 2: Read the histopathology  image from dataset
        imageA=cv2.imread(args[“first”])
Step 3: Display the name of the image on the webpage
Step 4: Shows the name of the image in the dialog box
```

Pseudocode for Converting RGB image to Gray

In this section the RGB image is converted to gray image because it is easy to perform the operations such as median filtering, thresholding and highpass filtering.

```
Step 1: Converting RGB to Gray
Step 2: Show the window to display the image
Step 3: Display the image in the window
```

Step4: Multiply each plane with a threshold value

Step5: Gray_pixel = (R_plan * R_thresh) + (G_plan * G_thresh) + (B_plan * B_thresh);

Pseudocode for Thresholding

Image thresholding is a simple, yet effective, way of partitioning an image into a foreground and background. This image analysis technique is a type of image segmentation that isolates objects by converting grayscale images into binary images.

```
Step 1: for i:=1 to 480*640 pixels
Step 2: if new_im(i)<T
Step 3: Change to black, and consider as obstacle setnew_im(i)=0
Step 4: Change to white, and consider as free space else
set new_im(i)=1
```

Pseudocode for Highpass Filtering

A high-pass filter can be used to make an image appear sharper. These filters emphasize fine details in the image- which is exactly the opposite of low pass filters. High pass filtering works in exactly the same way as a low pass filter except that it uses a different convolution kernel. Unfortunately, while low-pass filtering smooths out noise, high-pass only does the opposite by amplifying noise. However, if the original image is not too noisy- this is avoided to an extent.

```
Step1: Displays the high contrast version of original image.
Step2: Make a copy of the original image.
    Initialize color[ ][ ] retval ← copy(original)
Step3: Blur the copy of original image.
    set color[ ][ ]blurred ← gaussianBlur(original)
Step4: Subtract the blur image from original image.
    Color[ ][ ]highpass← difference(original,blurred)
```

Step5: Add the unsharp mask to original image to get sharpened image.

Step6: Run a loop for the entire rows and columns for row=0 to original.length

```
    repeat until for col=0 to original[row].length repeat until
    initialize color
    origColor ← original[row][col]
    initializecontrastColor ← highContrast[row][col]
    set color difference ← contrastColor – origColor
    end for
    end for
```

5.2.2 Pseudocode for contour detection and cell count

The provided algorithm outlines a systematic approach for analyzing cellular structures within an image. Initially, contours are extracted from the image, delineating the boundaries of individual cells. Subsequently, the total count of cells is determined, providing a fundamental metric of the cellular population. A threshold for minimum area is then applied to identify damaged cells based on their diminished size. Concurrently, an assessment for cell overlap is conducted by computing the centroids of contour pairs and measuring the distance between them. Should the calculated distance fall below a specified threshold, the contours are flagged as overlapping cells. This multi-step process enables comprehensive characterization of cellular features, encompassing cell count, damage detection, and overlap assessment, thus facilitating nuanced analysis of cellular structures within the image.

Step 1: Find contours in the image.

Step 2: Count the total number of cells.

```
    Total cells ← len(countours)
```

Step 3: Set the threshold for minimum area to detect damaged cells.

Step 4: Count the number of damaged cells based on the area threshold.

```
    Run a loop through countours
    Find area of each countours
    If area < minimum threshold area
    Increase damaged cell count
```

Step 5: Loop through each pair of contours.

```
    Calculate the centroids of the contours.
    Ensure that both contours have non-zero areas.
    Calculate the distance between centroids.
    If the distance is below a certain threshold, consider them overlapping.
```

Step 6: Output the counts for damaged and overlapping cells.

5.2.2 Pseudocode for Classification using CNN

The provided pseudocode outlines the process of building and training a Convolutional Neural Network (CNN) for image classification. Initially, the architecture of the CNN model is defined, incorporating convolutional layers for feature extraction, pooling layers for downsampling, and dense layers for classification. The model is then compiled with appropriate loss functions and optimizers. Following this, the dataset is split into training, validation, and testing sets to facilitate model evaluation. Subsequently, the model is trained on the training set, with performance monitored on the validation set to prevent overfitting. After training, the model is evaluated on the testing set to assess its generalization ability, utilizing metrics such as accuracy. Optionally, fine-tuning steps such as hyperparameter tuning and data augmentation can be applied to optimize model performance further. Overall, this process constitutes a comprehensive approach to constructing and training CNNs for image classification tasks.

Step 1: Define the architecture of the CNN model:

```
    a. Specify the input layer with dimensions matching the input image size.
    b. Add convolutional layers with filters, kernel size, and activation functions.
    c. Introduce pooling layers ( MaxPooling) to downsample the feature maps.
```

d. Flatten the output from the convolutional layers.

e. Add one or more dense (fully connected) layers for classification.

f. Include the output layer with softmax activation for multi-class classification.

Step 2: Compile the CNN model:

a. Specify the loss function appropriate for classification.

b. Optionally, specify metrics to monitor during training (accuracy).

Step 3: Split the dataset into training and testing sets:

a. Reserve a portion of the dataset (20%) for testing to evaluate the performance.

Step 4: Train the CNN model:

a. Feed the training data to the model and adjust the weights based on the specified loss function and optimizer.

b. Validate the model's performance on the validation set to monitor for overfitting.

c. Iterate the training process until convergence or a specified number of epochs.

Step 5: Evaluate the trained model:

a. Test the model on the unseen testing data to assess its generalization performance.

b. Compute evaluation metrics such as accuracy, precision, recall, and F1-score to quantify performance.

5.3 Summary

Chapter 5 describes the implementation phase, where the project's functionality is demonstrated. It emphasizes the importance of accurate execution to ensure the project's success. Key implementation requirements for the ovarian cancer prediction project using Convolutional Neural Network (CNN) include Python as the coding language and Windows 10 as the operating system. Visual Studio Code serves as the primary source-code editor, offering features such as debugging and Git integration. Additionally, the chapter highlights the significance of Python as a high-level programming language, particularly for its ease of learning, extensive library support, and suitability for various applications, including image processing using the Python Imaging Library (PIL). The chapter also covers other essential tools like Tkinter GUI for building graphical user interfaces, OpenCV-Python for computer vision tasks, Flask for web development, and Google Colab for collaborative Python coding with access to computing resources

CHAPTER 6

SYSTEM TESTING

Testing is the significant phase in the process or application development life cycle. Testing is final phase where the application is tested for the expected outcomes. Testing of the system is done to identify the faults or prerequisite missing. Therefore, testing plays a vital role for superiority assertion and confirming the consistency of the software. Software testing is essential for correcting errors and improving the quality of the software system. The software testing process starts once the program is written and the documentation and related data structures are designed. Without proper testing or with incomplete testing, the program or the project is said to be incomplete.

Throughout the testing phase, the procedure will be implemented with the group of test circumstances and the outcome of a procedure for the test case will be appraised to identify whether the program is executing as projected. Faults found during testing will be modified using the testing steps and modification will be recorded for forthcoming reference. Some of the significant aim of the system testing is

- To confirm the superiority of the project.
- To discover and eradicate any residual errors from prior stages.
- To authenticate the software as a result to the original problem.
- To offer effective consistency of the system.

6.1 Test Procedures

Test cases are the key aspect for the success of any system. A test case is a manuscript which has a set of test data, prerequisites, predictable results and post conditions, designed for a specific test scenario in imperative to validate acquiescence against a specific requirement. Performance of a system is based on the cases written for each and every modules of a system. Test cases are precarious for the prosperous performance of the system. If the predicted outcome doesn‟t match with the real outcome, then error log is displayed. There are two elementary forms of test cases, namely

1.Formal test case

2. Informal test case

6.1.1 Formal Test Case

There must be atleast two test cases such as non-positive and non-negative test for each requirement of an application in order to completely test and satisfy the requirement. If the requirement has sub-requirement then each of the sub-requirement needs to have atleast two test cases.

6.1.2 Informal Test Case

For applications or systems deprived of prescribed necessities, user can write the test cases based on acknowledged usual action of the program of analogous class. In certain positions of testing, test cases are not carved at all, however the events and the outcomes are conveyed after the tests have been run.

6.2 Unit Testing

Unit testing is the mechanism where individual model of the project is tested. It can also be called as differentiation testing, as the project is tested based on individual model. Using the module level designs depiction as a monitor significant device route is tested to discover faults around the boundary of each module.

The below table 6.1 shows the successful test case for loading the histopathological image of the ovary that is selected by the user to do the processing. Histopathological image of the ovary determines the stages of the cancer by determining the status of the cancer different types are endometrioid cancer,high grade serous carcinoma,low grade serous carcinoma and clear cell carcinoma. On the other hand, an histopathological image of a ovary also gives the accuracy of the result and determines the total cell count and overlapped cell count.

The main challenge in obtaining images of different stages of ovary is that multiple images may be taken from different perspectives, angles or distances, which don‟t match in dimensions. Before pre-processing the images, the images should match in dimensions. The resolution of the images captured also plays a vital role in conveying false notions to the classifiers- thus affecting accuracy. Also, the position of camera and how far from the scene may be a cause for image degradation. In this unit the images input to the system are checked whether they are loaded or not. If they are loaded then the expected output is reached.

Table 6.1: Test case for loading the histopathological image of ovary.

Test Case Sl. No	1
Test Name	User selects the histopathological image of the ovary.
Test Feature	Checks whether the image selected are loaded or not.
Output Expected	The histopathological image loaded from the dataset.
Output Obtained	Loaded histopathological image of the ovary which is used for the processing.
Result	Successful as shown in Figure 7.3.

The RGB image is converted to grayscale image and the successful test case for the grayscale image as shown in the Table 6.2, the gray level represents the brightness of a pixel. Here the RGB image is converted to grayscale image because it simply reduces complexity from a 3D pixel value (R, G, B) to a 1D value. Grayscale images can be the result of measuring the intensity of light at each pixel according to a particular weighted combination of frequencies.

Table 6.2: Test case for grayscale image.

Test Case Sl. No	**2**
Test Name	Grayscale image.
Test Feature	RGB image is converted to grayscale image.
Output Expected	Grayscale image.
Output Obtained	RGB to grayscale image conversion is done
Result	Successful as shown in Figure 7.6.

The RGB image is converted to grayscale image and the successful test case for the grayscale image as shown in the Table 6.2, the gray level represents the brightness of a pixel. Here the RGB image is converted to grayscale image because it simply reduces

complexity from a 3D pixel value (R, G, B) to a 1D value. Grayscale images can be the result of measuring the intensity of light at each pixel according to a particular weighted combination of frequencies.

The Table 6.3 shows the successful test case for Noise Removal. Noise removal is the process of removing or reducing the noise from the image. The noise removal algorithms reduce or remove the visibility of noise by smoothing the entire image leaving are as near contrast boundaries.

Here we are making use of Median Filter which is a Noise Removal Technique. The median filter is a non-linear digital filtering technique, often used to remove noise from an image or signal. Here 0‟s are appended at the edged and corners to the matrix which is the representation of the grey scale image. Then for every3*3 matrix, arrange elements in ascending order, then find median/middle element of those 9 elements.

Table 6.3: Test case for Median Filtering.

Test Case Sl. No	3
Test Name	Median filtering
Test Feature	Checking whether noise is actually removed.
Output Expected	The noise free image.
Output Obtained	The image is obtained.
Result	Successful as shown in Figure 7.6.

The Table 6.4 shows the successful test case for thresholding. Thresholding is a type of image segmentation, where we change the pixels of an image to make the image easier to analyze. Most frequently, we use thresholding as a way to select areas of interest of an image, while ignoring the parts we are not concerned with. We use Basic Global Thresholding.

Table 6.4: Test case for Thresholding.

Test Case Sl. No	4
Test Name	Checking the thresholding functionality.
Test Feature	To check if the images are being thresholded.
Output Expected	The image without the irrelevant parts.
Output Obtained	The thresholded image is obtained.
Result	Successful as shown in Figure 7.7.

The Table 6.5 shows the successful sharpened ovary image. The images are first sharpened and the background is blurred to focus only on the affected area. Sharpening is a technique for increasing the apparent sharpness of an image. The sharpening process works by utilizing a slightly blurred version of the original image. This is then subtracted away from the original to detect the presence of edges, creating the highpass filtered image. Contrast is then selectively increased along these edges using this mask- leaving behind a sharper final image. Highpass filters are not as popular as other filters like unsharp masks though.

Sharpening also helps to emphasize texture and detail.

Step 1: Detect Edges and create mask

By using highpass filtering first the original image is blurred. Then the blurred copy is subtracted from original image to obtain the sharpened image.

Step 2: Use the Sharpened Image for the next steps The obtained sharp image replaces the original image for further processing or classification.

Table 6.5: Test case for Highpass filtering.

Test Case Sl. No	5
Test Name	Highpass filtering.
Test Feature	To sharpen the leaf images at the concerned zones.
Output Expected	Sharpened ovary images.
Output Obtained	Sharpened image is obtained.
Result	Successful as shown in figure 7.7

The table 6.6, Using contour detection, we can detect the borders of cells, and localize them easily in an image. It is often the first step for many interesting applications, such as image-foreground extraction, simple-image segmentation, detection and recognition. Using the below formula the edge can be detected and the cells can be counted.

contours,_=cv2.findContours(thresh,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)

Table 6.6: Test case for contour detection.

Test Case Sl. No	6
Test Name	Contour detection.
Test Feature	To detect the borders of the cells.
Output Expected	To retrive the outer edge of the cell in the image.
Output Obtained	Retrives the outer edge of the cell in the image.
Result	Successful as shown in Figure 7.7.

The Table 6.7 shows how the training of the model is done for the ovarian cancer dataset.The required data is obtained for the Kaggle.The same dataset is used for testing and traing of the model.

Table 6.7: Test case for training data.

Test Case Sl. No	7
Test Name	Training data
Test Feature	Training ovarian cancer dataset
Output Expected	The data is to be trained
Output Obtained	Trained dataset
Result	Successful as shown in figure 7.4.

The Table 6.8 shows the successful test case for the classification of ovary image into the various stages. After finding the relevance of the image taken as input to the classification models taken, the cancer name as a status is recovered as a final output.

Table 6.8: Test case for classification of ovarian image.

Test Case Sl. No	8
Test Name	Classification of the histopathological image of ovary in different stages or types.
Test Feature	Relevance to the fit possible in the classifying model.
Output Expected	The status or stage to which the ovarian image belongs.
Output Obtained	Obtained page showing classification.
Result	Successful as shown in Figure 7.7.

6.3 Summary

This chapter presents system testing in section 6.1, which consists of unit test cases for the various modules of the prediction of the ovarian cancer. Section 6.2 gives a complete view of the testing which includes Test Name, Test Feature, Output Expected, Output Obtained and Result.

CHAPTER 7

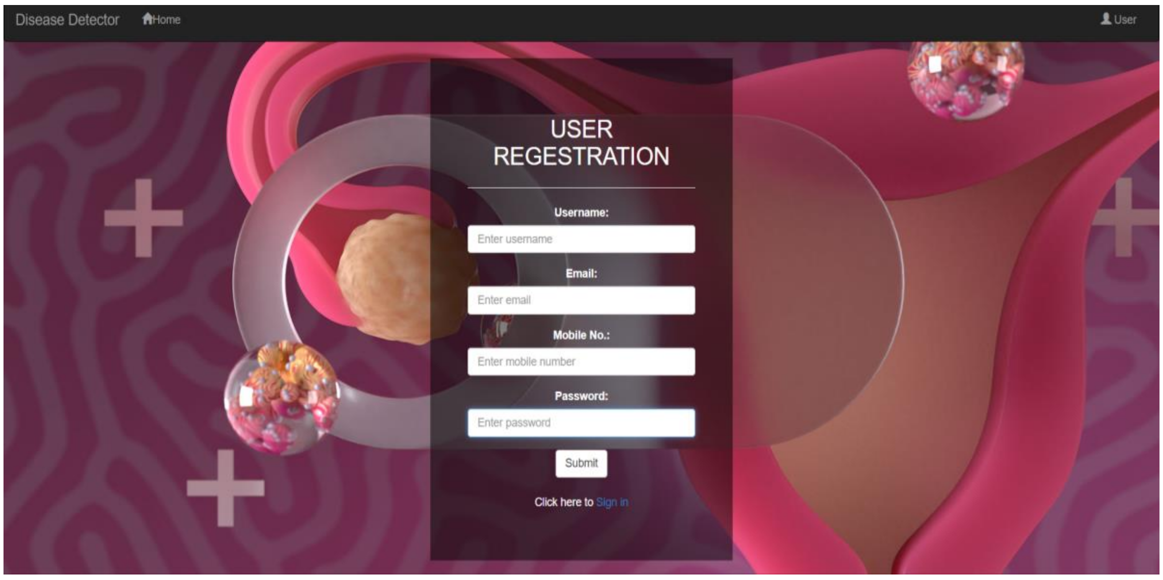
RESULTS AND DISCUSSION

Programming language used to design the proposed method is Python. By using relevant histopathological images of patient's ovary classified using CNN architectures, we get to know the stages of the cancer by determining the different status of cancer such as Clear cell carcinoma,Endometrioid carcinoma ,High-grade serous carcinoma ,Low-grade serous carcinoma and Mucinous carcinoma with a small description as the output. Results obtained in each step are demonstrated in following snapshots.

7.1 Experimental Results

Snapshot of User Signup Page

The below Snapshot Figure 7.1 shows the snapshot of User signup page.

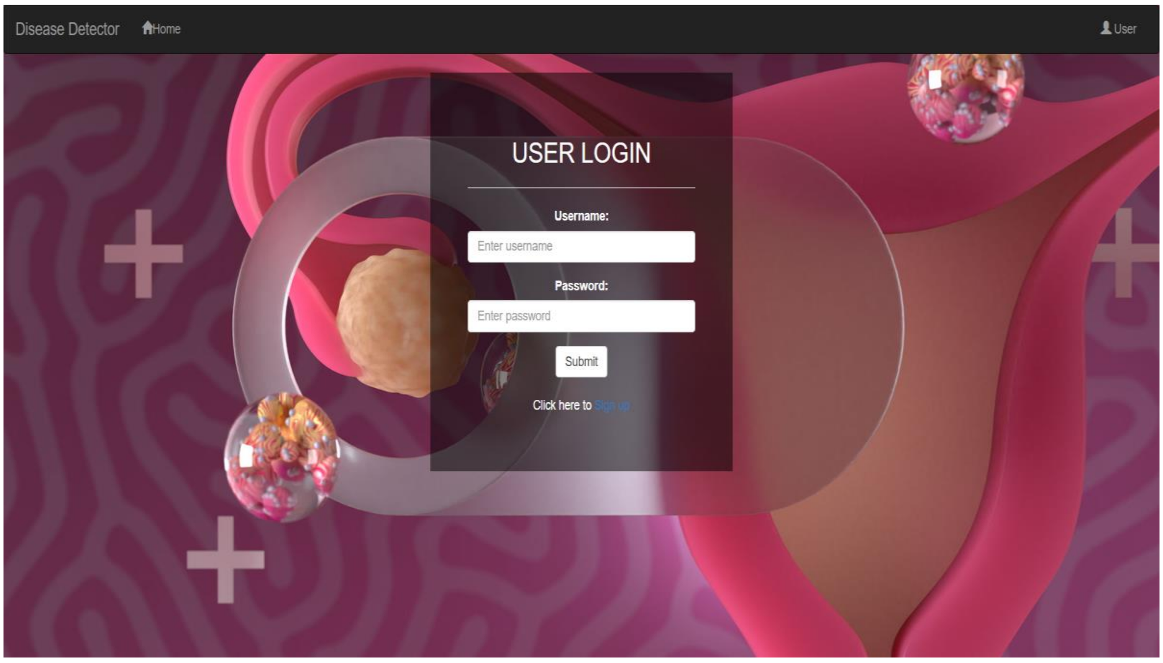


Snapshot 7.1 Snapshot of User signup page

The Snapshot 7.1 shows the snapshot of user signup page which consists of a header which reads User Registration‟ .where the username ,email, mobile number and the password should be entered later submit button is clicked.

Snapshot of User login page on Clicking “Submit”

The below Snapshot 7.2 shows the snapshot of the user login page on clicking the submit button.

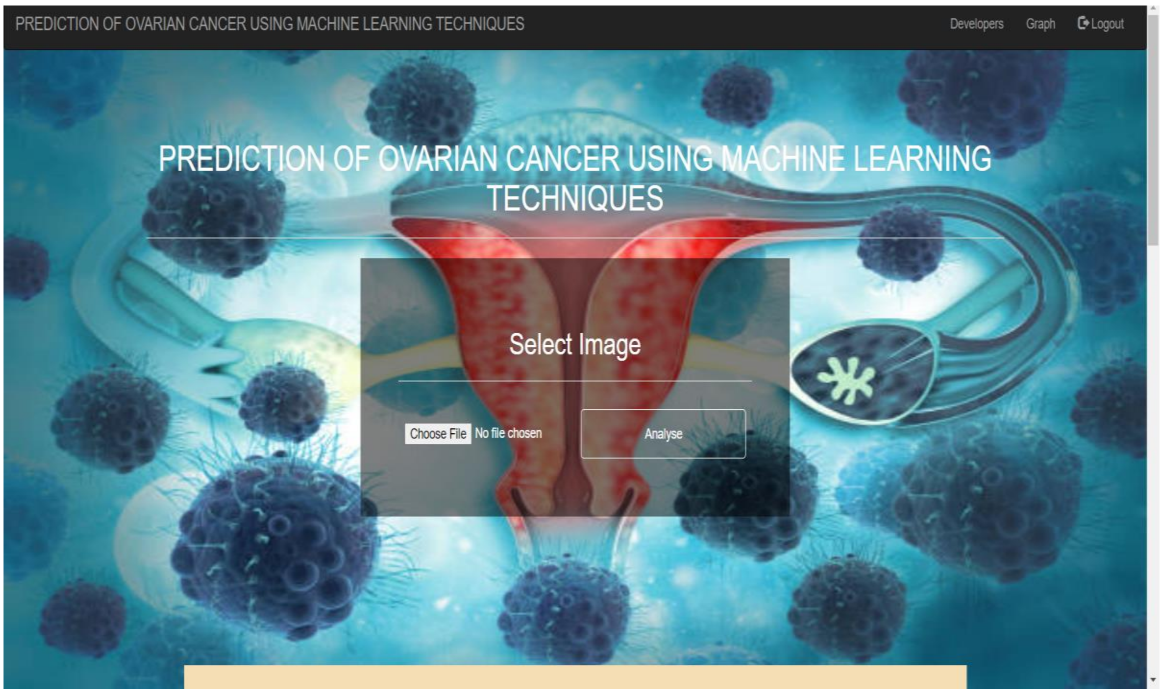


Snapshot 7.2: Snapshot of the User login page .

The Snapshot 7.2 shows the snapshot of user login page which consists of a header which reads “User login”.where only the registered user can login by entering valid username email and the password later submit button is clicked.

Snapshot of main page appears on Clicking “Submit”

The below Snapshot 7.3 shows the snapshot of the main page .

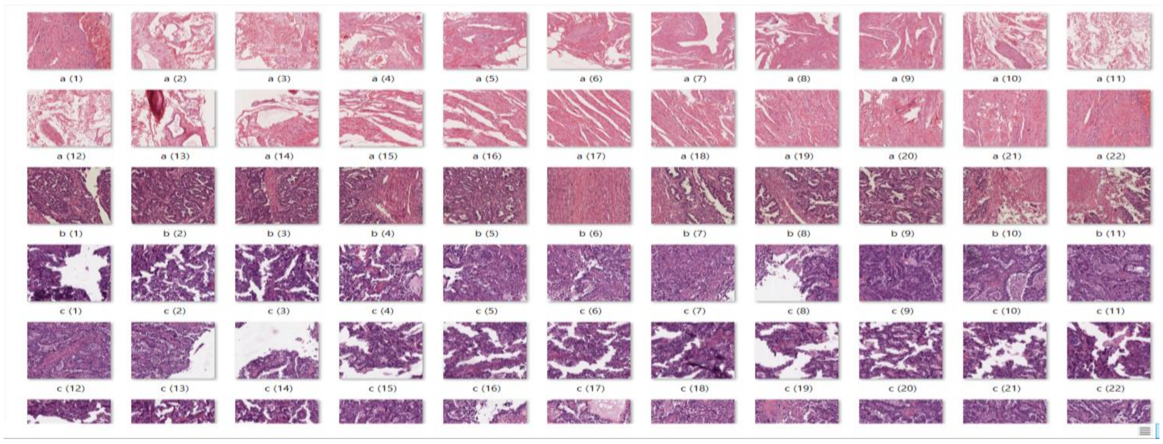

PREDICTION OF OVARIAN CANCER USING MACHINE LEARNING TECHNIQUES
Developers
Graph
Logout
PREDICTION OF OVARIAN CANCER USING MACHINE LEARNING TECHNIQUES
Select Image
Choose File No file chosen
Analyse

Snapshot 7.3: Snapshot of the Main page

The Snapshot 7.3 shows the snapshot of main page which consists of a header which reads “Prediction of Ovarian Cancer using Machine Learning Technique”.where one has to upload the histopatalogical image of the patient’s ovary.After selecting the image from the folder click on analyse.

Snapshot of file explorer appears on Clicking “Choose file”

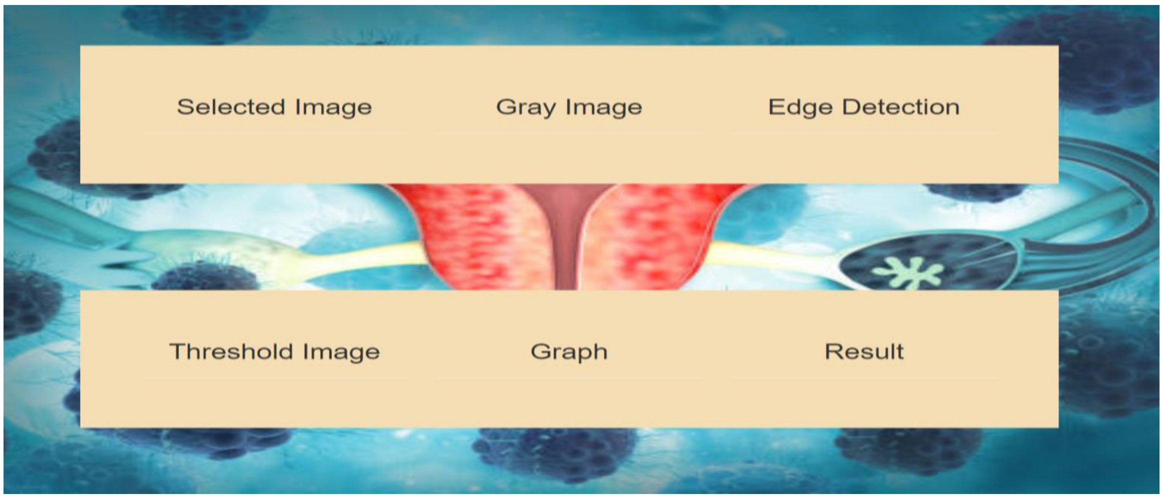
The below Snapshot 7.4 shows the snapshot of the file explorer with the images of ovary.



Snapshot 7.4: Datasets that are trained

The Snapshot 7.4 shows the snapshot of dataset which consists different class of images such as Clear cell carcinoma,Endometrioid carcinoma, High-grade serous carcinoma ,Low-grade serous carcinoma and Mucinous carcinoma. User has to select one image from the given five different types.

The below Snapshot 7.5 shows the snapshot of the details of the given input image with the images of ovary.

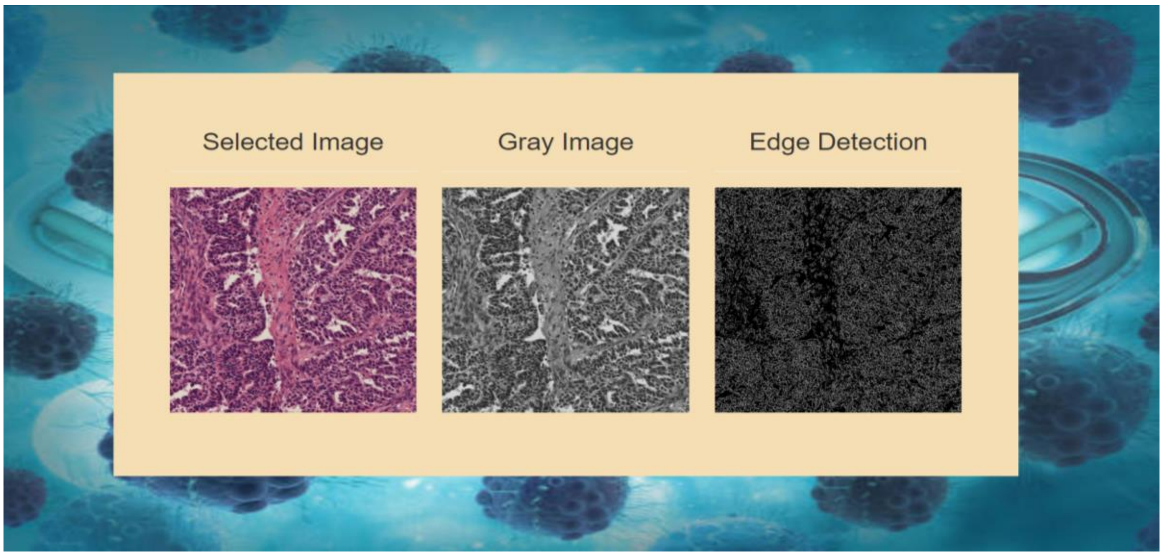


Snapshot 7.5: details of the given input image

The Snapshot 7.5 shows the snapshot details of the given input image such the selected

Image,gray scale image,edge detection ,threshold image,graph and finally the result is obtained is are some of the features that are obtained.

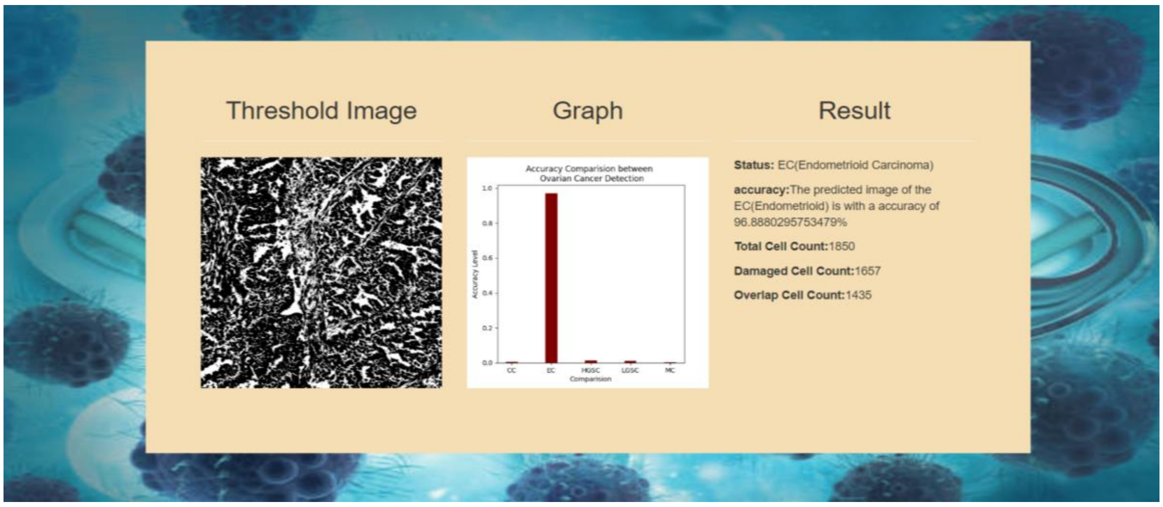
The below Snapshot 7.6 output of the given input image with the images of ovary.



Snapshot 7.6: Output of the given input image

The Snapshot 7.6 shows the snapshot details of the output of the given input image where an selected image is been analysed and been converted to grey scale image.Later on the Edge detection is done inorder to find the number of damaged cells and overlapped cells by using the technique of contour detection.

The below Snapshot 7.7 shows output of the given input image with the images of ovary.



Status: EC(Endometrioid Carcinoma)

accuracy: The predicted image of the EC(Endometrioid) is with a accuracy of 96.8880295753479%

Total Cell Count:1850

Damaged Cell Count:1657

Overlap Cell Count:1435

Snapshot 7.7: Output of the given input image

The Snapshot 7.7 shows the snapshot details of the output of the given input image where the threshold iamge is displayed thresholding is done in order to differentiate between the foreground and background and extract only the relevant features that are necessary and the graph that shows the accuracy comparision between ovarian cancer detection.in the result the status for example endometriod carcinoma ,accuracy,total cell count,damaged cell count,overlap cell count will be displayed.

7.2 Graphical Inferences

Prior to having a literary conclusion, as an understanding of the results we have seen in the implementation of the three CNN architectures, namely Mobile Net, VGG and Inception, let us have a look at the accuracy and loss graphs generated by each of these methods, before we derive inferences.

Accuracy Graph

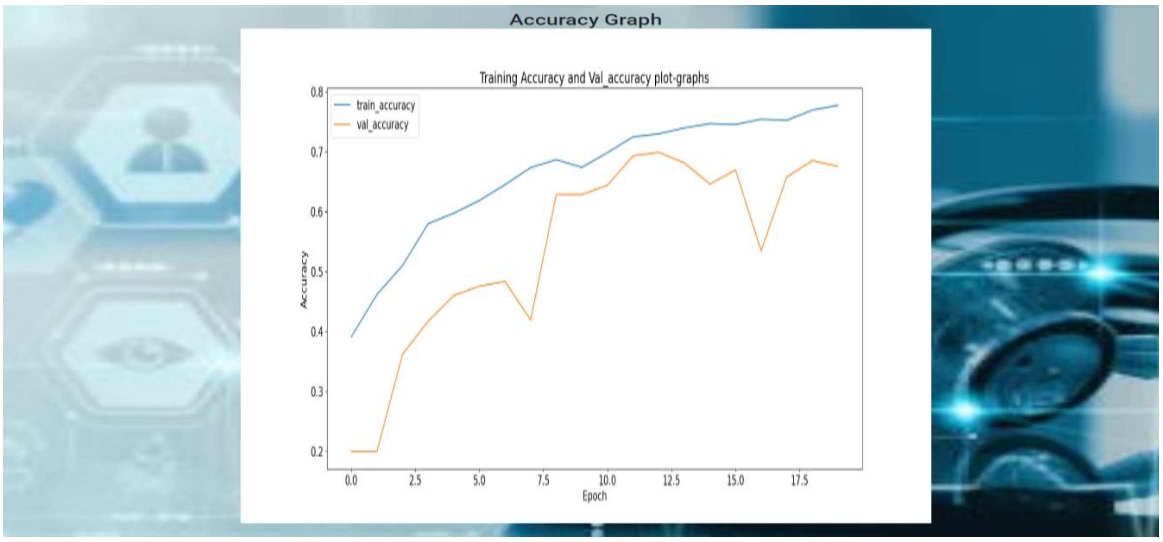
It is a plot of accuracy on the y-axis versus epoch on the x-axis, with plots for both training and test data. Accuracy should increase with epoch values for a better model.

An accuracy graph is a visual representation of the performance of a model or system over time or across different parameters. It is a fundamental tool in assessing the effectiveness of algorithms, models, or processes in various fields such as machine learning, statistics, and quality control. An accuracy graph plots the accuracy of predictions or classifications made by a model against some independent variable, such as the number of training iterations, the size of the training dataset, or the value of a hyperparameter. The accuracy is usually measured as the proportion of correct predictions or classifications out of the total number of instances.

Loss Graph

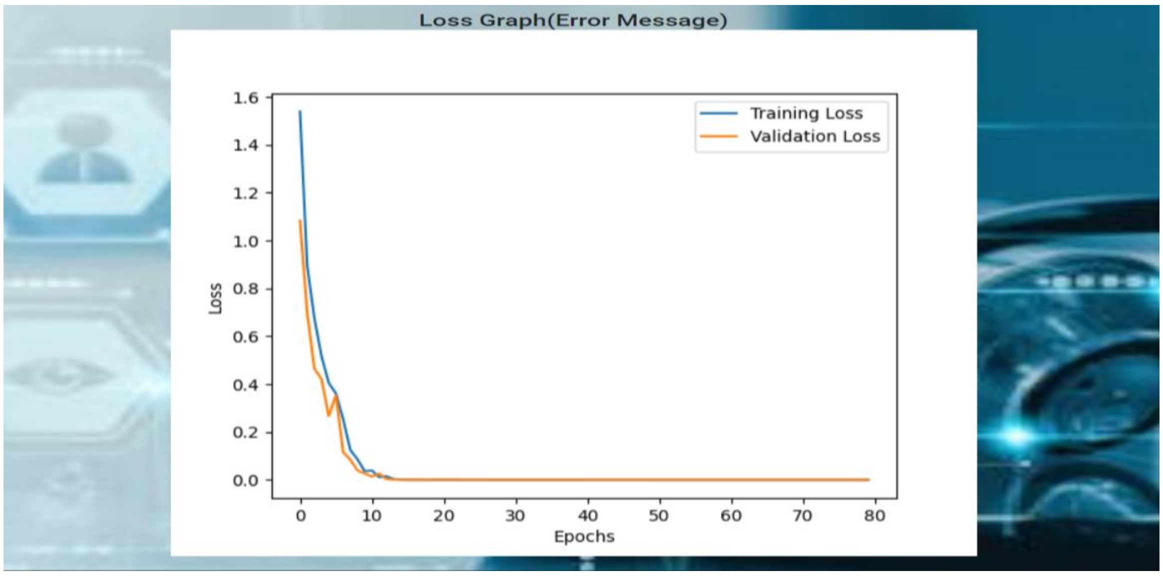
It is a plot of loss, in terms of features that are dropped from consideration on the y-axis versus epoch on the x-axis, with plots for both training and test data. Loss should decrease with epoch for a better model. A loss graph, also known as a loss curve or loss function plot, is a graphical representation of how a model's loss changes over time or iterations during the training process. Loss, in the context of machine learning, is a measure of how well a model's predictions match the actual values in the training data. The loss function quantifies the difference between predicted and true values, providing feedback to the model during training on how to adjust its parameters to minimize this difference.

7.2.1 Graphs for Inception Model



Snapshot 7.8: Accuracy graph

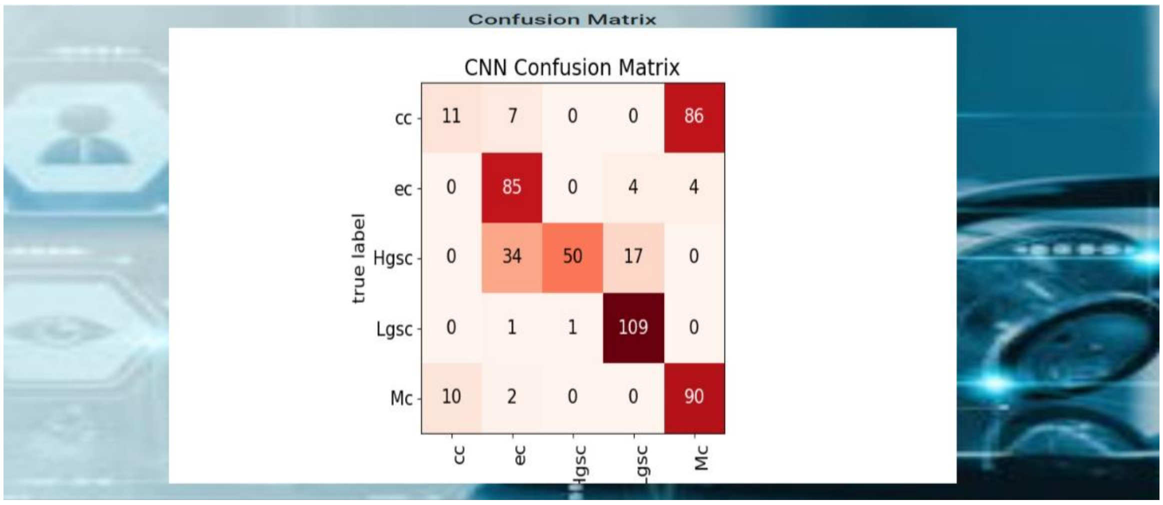
The snapshot 7.8 represents,the x-axis of the graph is labeled "Epoch" and the y-axis is labeled "Accuracy." An epoch refers to one pass through the entire training dataset. There appears to be a straight line labeled "train accuracy" that increases as the number of epochs increases. There is another line labeled "val accuracy" that also increases as the number of epochs increases, but it is consistently lower than the training accuracy.



Snapshot 7.9: Loss graph

The snapshot 7.9 represents, the x-axis is labeled "Epochs" and the y-axis is labeled "Loss". An epoch refers to one pass through the entire training dataset. The lower the loss, the better the model is performing.

7.2.2 Confusion matrix for Inception model



	cc	ec	Hgsc	Lgsc	Mc
cc	11	7	0	0	86
ec	0	85	0	4	4
Hgsc	0	34	50	17	0
Lgsc	0	1	1	109	0
Mc	10	2	0	0	90

Snapshot 7.10: Confusion matrix

The snapshot 7.10 is specifically a CNN confusion matrix, likely referring to a Convolutional Neural Network, used in the classification of ovarian cancer.Confusion matrices are used to evaluate the performance of a classification model. They show how many times the model makes the correct prediction (known as true positives) and how many times it makes mistakes (known as false positives, false negatives, and true negatives).

In the specific context of ovarian cancer, the rows of the matrix represent the actual cancer type, while the columns represent what the model predicted.

Here’s a breakdown of the matrix:

Rows

- **CC**: Clear-cell Carcinoma
- **EC**: Endomeroid Carcinoma
- **HGSC**: High-grade serous carcinoma
- **LGSC**: Low-grade serous carcinoma
- **MC**: Mucinous carcinoma

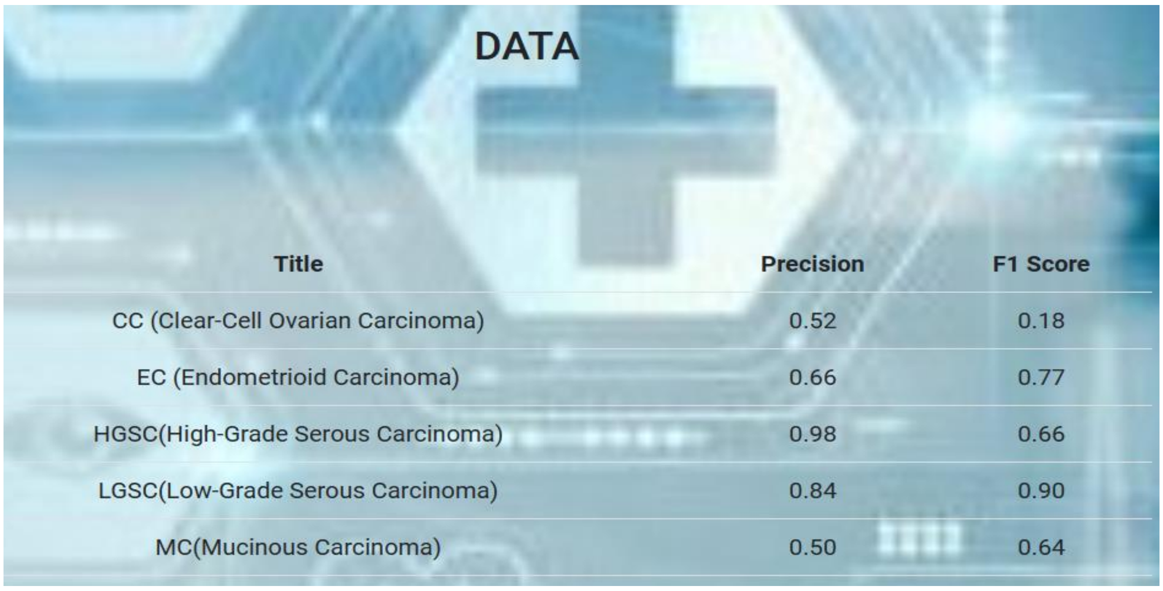
Columns

- **CC**: Number of patients correctly classified with Clear-cell Carcinoma
- **EC**: Number of patients incorrectly classified with Endomeroid Carcinoma
- **Hgsc**: Number of patients incorrectly classified with High-grade serous carcinoma

- **Lgsc**: Number of patients incorrectly classified with Low-grade serous carcinoma
- **Mc**: Number of patients incorrectly classified with Mucinous carcinoma

For instance, looking at the bottom right corner (Mc - Mc), we can see that 109 patients were correctly classified with Mucinous carcinoma. Overall, the confusion matrix helps us understand how well the model is distinguishing between the different types of ovarian cancer.

7.2.3 Precision and F1 Score of the built model



DATA

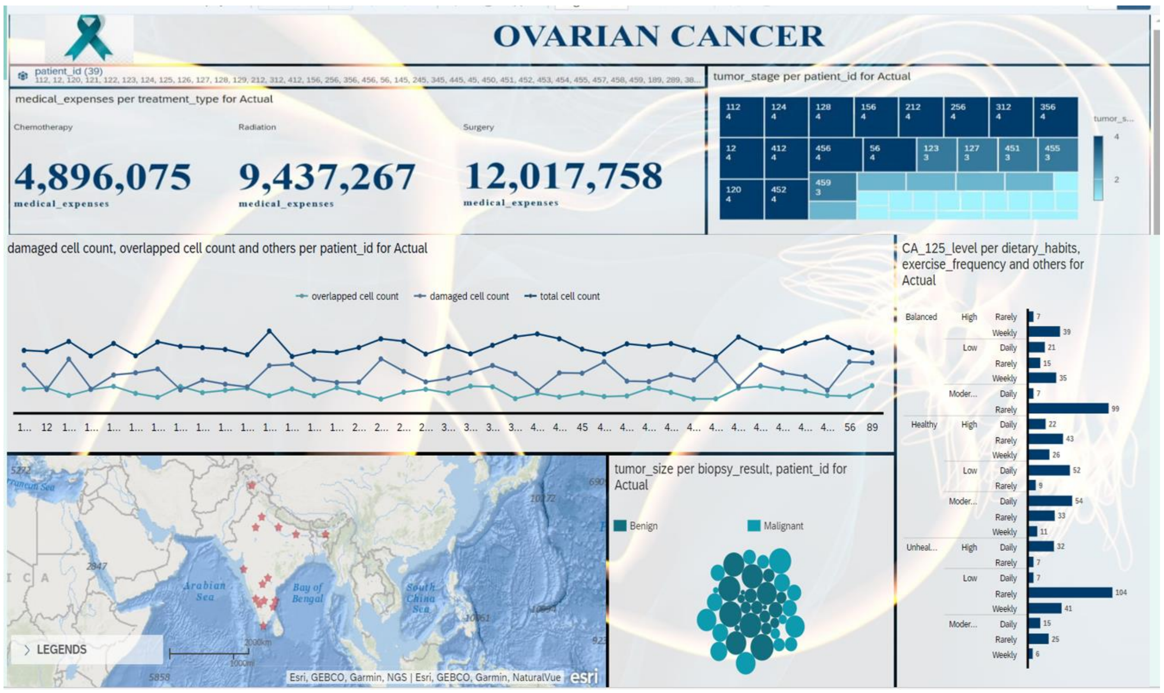
Title	Precision	F1 Score
CC (Clear-Cell Ovarian Carcinoma)	0.52	0.18
EC (Endometrioid Carcinoma)	0.66	0.77
HGSC(High-Grade Serous Carcinoma)	0.98	0.66
LGSC(Low-Grade Serous Carcinoma)	0.84	0.90
MC(Mucinous Carcinoma)	0.50	0.64

Snapshot 7.11: Precision and F1 Score

The snapshot 7.11 is confusion matrices are used to evaluate the performance of a classification model. They show how many times the model makes the correct prediction (known as true positives) and how many times it makes mistakes (known as false positives, false negatives, and true negatives).

- The model seems to perform best at classifying High-Grade Serous Carcinoma (HGSC) with a value of 0.98 in the top right corner (HGSC - HGSC). This means that out of all the cases the model predicted as HGSC, 98% were actually HGSC.
- The model appears to have the most difficulty with Clear-Cell Ovarian Carcinoma (CC) with a value of 0.18 in the bottom left corner (CC - CC). This means that out of all the cases the model predicted as CC, only 18% were actually CC.

7.3 Implementation of sap dashboard for the built model


OVARIAN CANCER
medical_expenses per treatment_type for Actual
Chemotherapy
Radiation
Surgery
4,896,075
9,437,267
12,017,758
tumor_stage per patient_id for Actual
damaged cell count, overlapped cell count and others per patient_id for Actual
overlapped cell count
damaged cell count
total cell count
CA_125_level per dietary_habits, exercise_frequency and others for Actual
tumor_size per biopsy_result, patient_id for Actual
Benign
Malignant
Arabian Sea
Bay of Bengal
South China Sea
LEGENDS
Esri, GEBCO, Garmin, NGS | Esri, GEBCO, Garmin, NaturalVue

Snapshot 7.12:Sap dashboard

The snapshot 7.12 represents the sap dashboard that is been developed for the ovarian cancer with different types of charts like line graph ,heat map, numeric point and cluster bubble and geomap. The different charts present in the above snapshot are:

- Medical expenses per treatment type is shown on numeric point.
- Tumor stage per patient id for Actual shown in heat map.
- Damaged cell count and overlapped cell count per patient id shown in line graph.
- Tumor size per biopsy result and patient id shown in cluster bubble.
- The dashboard consists of a geomap where the location is shown.

7.4 Summary

This chapter presents the experimental results in section 7.1, which consists of snapshots of the user sign in page,user login page,main page, details of the input image,output of the input image. section 7.2 consists graphical inferences made.section 7.3 consists of the implementation of the sap dashboard.

CHAPTER 8
CONCLUSION AND FUTURE ENHANCEMENT

8.1 Conclusion

In conclusion, the project on the prediction of ovarian cancer using machine learning techniques has been successfully implemented, demonstrating a systematic approach towards diagnosing and classifying ovarian cancer based on histopathological images. provides a clear outline of the preprocessing techniques, contour detection, cell count, and classification using Convolutional Neural Networks (CNNs). These techniques collectively enabled the extraction of meaningful features from the images and accurate classification of different stages or types of ovarian cancer with accuracy of 80% .Overall, the project showcases the potential of machine learning techniques, particularly CNNs, in the field of medical image analysis and cancer diagnosis. By automating the process of ovarian cancer detection and classification, this project contributes towards early diagnosis, personalized treatment planning, and ultimately, improved patient outcomes in the fight against ovarian cancer.

8.2 Future Enhancement

At the present level, user can input the histopathology image to identify ovarian cancer, further this system can be enhanced by Integrating histopathological image analysis with clinical data (e.g., patient demographics, medical history) to create a comprehensive predictive model. Combining multiple modalities of data can provide a more holistic view of the patient's condition and improve the accuracy of cancer prediction. Experiment with different CNN architectures, including deeper models (e.g., ResNet, DenseNet) or novel architectures specifically designed for medical image analysis tasks. These architectures may capture more complex features and patterns relevant to ovarian cancer diagnosis. Integrate additional data modalities such as genomic data, clinical metadata, or radiological imaging alongside histopathological images. Multi-modal fusion techniques can help capture complementary information and improve predictive performance.

REFERENCES

[1] M. Aditya, I. Amrita, A. Kodipalli and R. J. Martis, "Ovarian Cancer Detection and Classification Using Machine Leaning," 2021 5th International Conference on Electrical, Electronics, Communication, Computer Technologies and Optimization Techniques (ICEECCOT), Mysuru, India, 2021, pp. 279-282, Doi: 10.1109/ICEECCOT52851.2021.9707954.

[2] C. Nayak, A. Tripathy, M. Parhi and S. K. Barisal, "Early Stage Ovarian Cancer Prediction using Machine Learning," 2023 International Conference in Advances in Power, Signal, and Information Technology (APSIT), Bhubaneswar, India, 2023, pp. 603-608, Doi: 10.1109/APSIT58554.2023.10201764.

[3] V. Saravanan, V. Sankaradass, M. Shanmathi, J. P. Bhimavarapu, M. Deivakani and S. Ramasamy, "An Early Detection of Ovarian Cancer and The Accurate Spreading Range in Human Body by using Deep Medical Learning Model," 2023 International Conference on Disruptive Technologies (ICDT), Greater Noida, India, 2023, pp. 68-72, Doi: 10.1109/ICDT57929.2023.10151103.

[4] Litjens, G., Sánchez, C. I., Timofeeva, N., et al. (2016). "Deep learning as a tool for increased accuracy and efficiency of histopathological diagnosis." Scientific Reports, 6, 26286.

[5] L. Akter and N. Akhter, "Ovarian Cancer Classification from Pathophysiological Complications using Machine Learning Techniques," 2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT), Kharagpur, India, 2021, pp. 1-6, Doi: 10.1109/ICCCNT51525.2021.9580067.

[6] Janowczyk, A., Madabhushi, A. (2016). "Deep learning for digital pathology image analysis: A comprehensive tutorial with selected use cases." Journal of Pathology Informatics, 7, 29.

[7] Huang, C., Dong, D., Zhang, H., et al. (2017). "Ovarian cancer identification based on convolutional neural network with SMOTE in gene microarray." BioMed Research International, 2017, 8929301.

[8] Elter, M., Schulze, M., Demes, M., et al. (2019). "Histopathological Image Classification in Histologic Subtypes of Ovarian Cancer." Methods of Information in Medicine, 58(1), 4-10

[9] Maria Franchi-Abella et al. conducted a comprehensive review of clinical prediction models for ovarian cancer, examining their development, validation, and clinical

applicability. Their paper, titled "Predicting ovarian cancer: A review of clinical prediction models," provides insights into various models used in predicting ovarian cancer.

[10] Kevin Huang et al. delved into the realm of artificial intelligence for predicting ovarian cancer recurrence in their 2019 study titled "Predicting Ovarian Cancer Recurrence with Artificial Intelligence." They explored the efficacy of deep learning techniques in this context, evaluating different machine learning models.

[11] Chun-Chieh Wang et al. discussed the application of machine learning algorithms in ovarian cancer prognosis and classification in their 2018 paper, "Machine Learning in Ovarian Cancer Prognosis and Classification." Their study reviewed the effectiveness of various machine learning approaches in predicting outcomes and classifying ovarian cancer subtypes.

[12] Wei-Chih Huang et al. focused on predictive modeling of ovarian cancer survival using long non-coding RNA expression data in their 2017 research titled "Predictive Modeling of Ovarian Cancer Survival with Long Non-Coding RNA Expression Data." They explored the potential of machine learning techniques in leveraging molecular data for survival prediction.

[13] Aruna Malik et al. investigated early prediction of ovarian cancer using machine learning techniques applied to clinical data in their 2019 study titled "Early Prediction of Ovarian Cancer Using Machine Learning Techniques." They developed and evaluated predictive models for early detection of ovarian cancer.

[14] Yaniv Gur et al. explored the application of deep learning techniques for detecting and classifying ovarian cancer based on ultrasound images in their 2020 study titled "Deep Learning for Ovarian Cancer Detection and Classification using Ultrasound Images," aiming to improve diagnostic accuracy for ovarian cancer.