A Project Report

On

# IDENTIFYING BONE TUMOR USING X-RAY IMAGES

Project Submitted in partial fulfillment of requirements for the Award of the Degree of

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**


**Submitted by**

**JONNADA SRINU**

**20U91A0556**

Under the Esteemed guidance of

**R. BHUVANESWARI** **M. Tech**

Asst. Professor




**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**SRI MITTAPALLI COLLEGE OF EGINEERING**

(Approved by AICTE, New Delhi and Affiliated to JNTU Kakinada)

(An ISO 9001:2005 Certified Institution and Accredited by NAAC& NBA)

**TUMMALAPALEM, NH-5, GUNTUR, 522233, A.P.**


**(2020-2024)**

# SRI MITTAPALLI COLLEGE OF ENGINEERIN

**(Approved by AICTE, New Delhi and Affiliated to JNTU Kakinada)**

**(An ISO 9001:2005 Certified Institution and Accredited by NAAC & NBA)**

**TUMMALAPALEM, NH-5, GUNTUR, 522233, A.P.**

## CERTIFICATE

This is to certify that the project entitled **"IDENTIFYING BONE TUMOR USING X- RAY IMAGES"**is the Bonafede work of **JONNADA SRINU (20U91A0556)**, submitted in partial fulfillment of requirements for award degree of Bachelor of Technology in Computer Science and Engineering by **JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, KAKINADA** during the year 2023-2024.

**PROJECT GUIDE**
**R. BHUVANESWARI M. Tech**
**Asst. Professor**

**HEAD OF THE DEPARTMENT**
**Dr. S. GOPI KRISHNA M. Tech, Ph. D**
**Professor & HOD**

**EXTERNAL EXAMINER**

# ACKNOWLEDGEMENT

I would like to express my utmost gratitude to our Chairman Sri. **M.V. KOTESWARA RAO** and Secretary Sri. **M.B.V. SATYANARAYANA** for providing their support and stimulating environment for the development of our project.

I express our deep sense of reverence and profound graduate **Dr. S. GOPI KRISHNA** **M.Tech, Ph.D.,** Principal for providing me the great support for carrying out this project.

I extend my sincere thanks to **Dr. S. GOPI KRISHNA** **M.Tech, Ph.D,** Head of the Departmentof C.S.E for his co-operation and guidance to make this project successful.

I am also immensely thankful to my guide **Miss. R. BHUVANESWARI** **M. Tech,** for her moral support and guidance throughout the project. My sincere thanks also go to all the teaching and non-teaching staff for their constant support and advice.

I would like to thank my friends who have helped us in the successful completion of this project, either directly or indirectly.

**JONNADA SRINU**

**(20U91A0556)**

# DECLARATION

I J.Srinu declare that the contents of this project, in full or part, have not been submitted to any other university or institution for the awardof any degree or diploma.

I also declare that we have adhered to all principles of academic honesty and integrity and not mis represented or fabricated or falsified any idea/data/fact/source in our submission

I understand that any violation of the above will cause disciplinary action by the institution and can also evoke penal action for the sources which have not been properly cited or from whom proper permission has not been taken when needed.

**JONNADA SRINU**

**(20U91A0556)**

DATE :

PLACE :

# TABLE OF CONTENTS

# ABSTRACT

This paper is based on integration of the biomedical field and computer science. Paper contains the study of bone cancer and features to predict the type of the same. Related work to find cancer in human body using computer vision is discussed in this paper. Image segmentation technique like so bel, pre with, canny, K- means and Region Growing are described in this paper which can be stimulated for X-Ray and MRI image interpretation. Paper also shows the result of edge based and region-based image segmentation techniques applied on X-Ray image to detect osteosarcoma cancer present on bone using MATLAB. Finally, paper concluded by finding best suited segmentation method for grey scaled image with future aspects.

Bone tumors are abnormal growths of cells within bone tissue that can be benign or malignant, posing significant diagnostic challenges due to their diverse morphologies and locations. X-ray imaging remains one of the primary modalities for initial detection and assessment of bone tumors due to its widespread availability, cost-effectiveness, and ability to provide detailed structural information. This paper presents a comprehensive review of recent advancements in identifying bone tumors using X-ray images, focusing on various techniques, methodologies, and challenges encountered in clinical practice and research.

The review encompasses both traditional image processing approaches and modern deep learning- based methods for bone tumor detection, segmentation, classification, and characterization. It discusses the significance of feature extraction, image enhancement, and segmentation algorithms in preprocessing X-ray images to improve the accuracy and efficiency of tumor detection. Additionally, it explores the utilization of convolutional neural networks (CNNs) and other deep learning architectures for automated analysis of X-ray images, highlighting their strengths in learning discriminative features directly from raw pixel data.

# CHAPTER-1
# INTRODUCTION

# INTRODUCTION

X-ray image analysis provides one of the cheapest primary screening tools for the diagnosis of bone cancer. As reported in the medical literature a primary bone tumor usually appears with unsuspecting symptoms such as fracture of a bone, swelling around a bone,a new bone growth, or swelling in the soft tissues surrounding a bone. Oftentimes, an X-ray image of cancer-affected bone appears different from its surrounding healthy bones and flesh region. The X-ray absorption rate of bone cells in the cancer-affected region differs from that in healthy bone cells.

As a result, the image of cancer-affected bones appears in the form of a "ragged" surface(permeative bone destruction), tumor (geographic bone destruction), or holes (moth- eaten pattern of bone destruction). Grading of bone cancer and identification of the underlying bone-destruction pattern are two essential components needed for treatment of the disease. The stage and grade of bone cancer represent a measure of the severity of the disease. Progressive identification of destruction pattern in a cancer-affected bone also helps doctors to estimate the rapidity of growth of the disease, or prognosis of the treatment. Hence, automated classification of bone-cancer stage, grade, and destruction pattern will be useful to medical practitioners in order to plan for the course of treatment]

In the past few years, researchers have proposed different approaches for bone-tumor detection. Conventional image analysis techniques such as thresholding, region growing, classifiers, and Markov random field model have been used for the detection of tumor region in X-ray and MRI1 images. Firangi et al. have used multi-scale analysis of MRI perfusion images for bone-tumor segmentation.

They proposed a two-stage cascaded classifier for hierarchical classification of healthy and tumor tissues, and subsequently, to discriminate viable and non-viable tumors. Ping et al.

have proposed an approach based on intensity analysis and graph description for the detection and classification of bone tumor from clinical X-ray images.

The method analyzes a graph representation to locate the suspected tumor area. It can also classify the benign and malignant tumor depending on the number of pixels extracted from the analysis of brightness values. Bone CT images have also been widely used for fracture detection and disease diagnosis. A fusion between CT2 and SPECT3 images is proposed for the identification of cancerous regions in bone image. Yao et al. have designed an automated lytic bone metastasis detection system. The procedure uses adaptive thresholding, morphology, and region growth for the segmentation of spine region.

and a support vector machine (SVM) classifier is used for feature classification and to diagnose the affected lesions. Automated diagnosis of secondary bone cancer from a CT image of bone vertebrae was proposed by Huang et al

Their technique involves texture-based classifiers and an artificial neural network (ANN), which are used for the detection of abnormality.

Fuzzy-possibilistic classification and variational model are also utilized for multimodal bone cancer detection from CT and MRI images.

Most of the researchers have focused on the identification of bone tumor or cancer- affected region from CT or MRI. To the best of our knowledge, no automated method for determining the destruction pattern caused by bone cancer along with classification of its stage and grade is yet known. In this paper, we have proposed a computer-aided diagnostic method that can perform an automated analysis of bone X-ray images and identify the cancer-affected region. Our method can be used to localize the destruction pattern and to assess the severity of the disease based on its stage and grade.

The rest of the paper is organized as follows. “Methods” discusses various phases of the proposed method and the features used for bone cancer detection. An algorithm for localizing the cancer-affected zone is presented in “Localization of Cancer-Affected Region.” Procedures for classifying stage and grade of the disease are described in “Cancer Severity Analysis.” Results on test cases and performance of the proposed method are reported in “Identification of Bone-Destruction Pattern.” Discussions of test results and concluding remarks appear in “Results” and “Discussion,” respectively.

## 1.1. OBJECTIVE OF THE PROJECT:

Bone cancer originates from bone and rapidly spreads to the rest of the body affecting the patient. A quick and preliminary diagnosis of bone cancer begins with the analysis of bone-ray or MRI image. Compared to MRI, an *X*-ray image provides a low-cost diagnostic tool for diagnosis and visualization of bone cancer. In this paper, a novel technique for the assessment of cancer stage and grade in long bones based on *X*- ray image analysis has been proposed.

Cancer-affected bone images usually appear with a variation in bone texture in the affected region. A fusion of different methodologies is used for the purpose of our analysis. In the proposed approach, we extract certain features from bone *X*-ray images and use support vector machine (SVM) to discriminate healthy and cancerous bones.

# CHAPTER-2
# LITERATURE SURVEY

# LITERATURE SURVEY

## 1. "Long-bone fracture detection"

Introduced a unified technique for detecting and evaluating orthopedic fractures in long-bone digital X-ray images.

- Developed a user-friendly software tool for paramedics and doctors.
- Segmentation, contour generation, and fracture detection process explained.

## 2. "Automatic segmentation of bones in X-ray images"

- Proposed an efficient segmentation method for isolating bones from surrounding tissues in X-ray images.
- Utilized an entropy measure for automated segmentation, addressing challenges like low contrast and noise.
- Introduced metrics, ACDI and SEQA, to quantify segmentation quality, showing superiority over existing methods.

## 3. "Measurement of curvature in a quantized environment"

- Explored the practical application of curvature as a shape descriptor for pattern recognition and image processing.
- Investigated curvature's role in shape recognition and processing in a quantized environment.

## 4. "Multimodal bone cancer detection"

- Addressed challenges in precise bone cancer segmentation using fuzzy classification and variational models.
- Introduced a method involving registration, fuzzy-possibilistic classification, and variational modeling for accurate tumor detection.
- Preliminary results showed promising and accurate cancer region detection.

## 5. "Tutorial on support vector machines"

- Provided an overview of VC dimension, structural risk minimization, and linear Support Vector Machines (SVMs).
- Discussed SVM implementation, kernel mapping, and arguments supporting their observed high accuracy.
- Reviewed experiments inspired by these arguments.

## 6. "Decision forests for computer vision"

- Presented a unified framework involving random sub-windows and ensembles of extremely randomized trees.
- Applied the framework to various computer vision problems, including image classification and segmentation.
- Illustrated methods on biomedical applications.

### 7. "MR Imaging in staging of bone tumors"

- Investigated the effect of scapular dyskinesis on supraspinatus tendon healing after repair.
- Used a rat model and evaluated shoulder function, joint mechanics, and tendon properties.
- Identified scapular dyskinesis as a mechanism compromising supraspinatus healing after repair.

## 8. "Segmentation of bone tumor in MR perfusion images"

- Developed an approach for segmenting dynamic perfusion MR images into viable tumor, nonviable tumor, and healthy tissue.
- Used cascaded feedforward neural networks and multiscale pharmacokinetic features.
- Experiment results highlighted the discriminative nature of multiscale features.

## 9. "On the encoding of arbitrary geometric configurations"

- Described a method for encoding arbitrary geometric configurations for digital computer analysis and manipulation.
- Discussed ways of encoding geometric curves, focusing on the rectangular-array type of encoding.
- Explored the quantization of the slope function for manipulation.

## 10. "Machine learning analysis of gender differences"

- Explored the impact of gender differences on machine learning outcomes in visual inspection decision making.
- Presented a study with 50 female and 50 male subjects, revealing significant differences in induced decision trees.
- Highlighted the value of investigating hidden structures of cognitive gender differences in machine learning.

# CHAPTER-3
# SYSTEM ANALYSIS

# SYSTEM ANALYSIS

## 3.1 Existing System

X-ray analysis serves as a cost-effective primary screening tool for diagnosing bone cancer. Primary bone tumors often manifest with inconspicuous symptoms like bone fractures, swelling, new bone growth, or soft tissue swelling. Cancer-affected bones display distinct differences in X-ray absorption rates compared to healthy bones, resulting in recognizable visual patterns such as a "ragged" surface (permeative bone destruction), tumor (geographic bone destruction), or holes (moth-eaten pattern of bone destruction**).**

### 3.1.1 FAULTS:

- **Low Accuracy:**

Fault: The existing X-ray image analysis system is reported to have low accuracy in diagnosing bone cancer.

Implication: Low accuracy can lead to misdiagnosis and delayed treatment.

- **Potential Health Risks:**

Fault: Exposure to X-rays may increase the possibility of developing cancer later in life.

Implication: This poses a health risk to individuals undergoing X-ray screenings and may deter some from seeking necessary diagnostic procedures.

- **Tissue Effects:**

Fault: Tissue effects such as cataracts, skin reddening, and hair loss are mentioned, which occur at relatively high levels of radiation exposure.

Implication: These adverse effects can be concerning for patients and may outweigh the benefits of using X-ray for cancer diagnosis.

- **Limited Detection Capability:**

Fault: The description suggests that the existing system relies on visual analysis of X-ray images, which may not effectively distinguish cancer-affected bones from healthy ones.

Implication: This limitation may result in overlooking early signs of bone cancer, leading to delayed detection and treatment.

## 3.2 Proposed System

The retrospective study analyzed bone tumors on radiographs acquired prior to treatment from January 2000 to June 2020. Radiographs from 934 patients were evaluated including 667 benign bone tumors and 267 malignant bone tumors. Researchers used more than 600 cases to train the deep learning model and then 140 cases to validate and test it. The model achieved 80% accuracy and 88% specificity in the classification of bone tumors as malignant or benign. The models' accuracy in classifying tumors as malignant or benign was higher than that of two fourth year radiology residents and was comparable with that of two highly experienced musculoskeletal fellowship-trained radiologists**.**

- **High Accuracy in Classification:**

Proposed System Advantage: Achieves 80% accuracy and 88% specificity in classifying bone tumors as malignant or benign.

Implication: The proposed system addresses the low accuracy issue in the existing system.

- **Noninvasive and Painless Diagnosis:**

Proposed System Advantage: The proposed system offers noninvasive and painless methods for diagnosing diseases.

Implication: This is a significant improvement over the potential health risks associated with the existing X-ray system.

- **Supports Treatment Planning:**

Proposed System Advantage: Supports medical and surgical treatment planning.

Implication: The proposed system provides additional benefits in planning and implementing treatment strategies.

- **Expert-Level Performance:**

Proposed System Advantage: The model's accuracy is comparable to highly experienced radiologists.

Implication: This addresses the issue of low accuracy and variability in performance observed in the existing system.

# CHAPTER-4

# SYSTEM REQUIRMENTS

# SYSTEM REQUIREMENT

## 4.1 Hardware Requirements:

- Speed - 1.1 GHz
- RAM - 4GB (min)
- Hard Disk - 500 GB
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
- Monitor - SVGA

## 4.2 Software Requirements:

- Operating System - Windows10(min)
- Programming Language - Python

## 4.3 System Study Feasibility Study

A Software Requirements Specification (SRS) – a requirements specification for a software system is a complete description of the behavior of a system to be developed. It includes a set of use cases that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements.

Nonfunctional requirements are requirements which impose constraints on the design or implementation (such as performance engineering requirements, quality standards, or design constraints).

System requirements specification: A structured collection of information that embodies the requirements of a system. A business analyst, sometimes titled system analyst, is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the systems development lifecycle domain, the BA typically performs a liaison

function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

## • ECONOMIC FEASIBILITY

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economic feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs. The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, there is nominal expenditure and economic feasibility for certain.

## • OPERATIONAL FEASIBILITY

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization's operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So, there is no question of resistance from the users that can undermine the possible application benefits. The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

## • TECHNICAL FEASIBILITY

Earlier no system existed to cater to the needs of 'Secure Infrastructure Implementation System'. The current system developed is technically feasible. It is a web-based user interface for audit workflow at NIC-CSD. Thus, it provides an easy access to. The users. The database's purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security.

## 4.4 INPUT AND OUTPUT DESIGN

### 4.4.1 INPUT DESIGN:

considering the requirements, procedures to collect the necessary input data in most efficiently designed. The input design has been done keeping in view that, the interaction of the user with the system being the most effective and simplified way.

Also, the measures are taken for the following

- Controlling the amount of input
- Avoid unauthorized access to the application.
- Eliminating extra steps
- Keeping the process simple

At this stage the input forms and screens are designed.

### 4.4.2 OUTPUT DESIGN:

All the screens of the system are designed with a view to provide the user with easy operations in simpler and efficient way, minimum key strokes possible. Instructions and important information are emphasized on the screen. Almost every screen is provided with no error and important messages and option selection facilitates. Emphasis is given for speedy processing and speedy transaction between the screens. Each screen assigned to make it as much user friendly as possible by using interactive procedures. So, to say user can operate the system without much help from the operating manual.

# CHAPTER-5
# UML DIAGRAM

# UML DIAGRAMS

## 5.1. CLASS DIAGRAM

The class diagram is the main building block of object-oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed. In the diagram, classes are represented with boxes which contain three parts**:**

- The upper part holds the name of the class
- The middle part contains the attributes of the class
- The bottom part gives the methods or operations the class can take or undertake

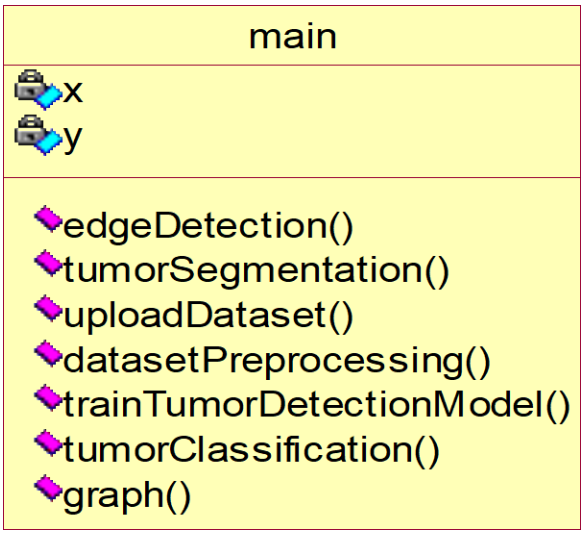



**Fig 1 Class diagram**

## 5.2. USE CASE DIAGRAM

A Use Case diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams.

A use case diagram is a graphical representation of the interactions between users (actors) and a system to achieve specific goals or tasks. In the context of a project focused on identifying bone tumors using X-ray images, a use case diagram can help illustrate the various functionalities and interactions involved in the system.

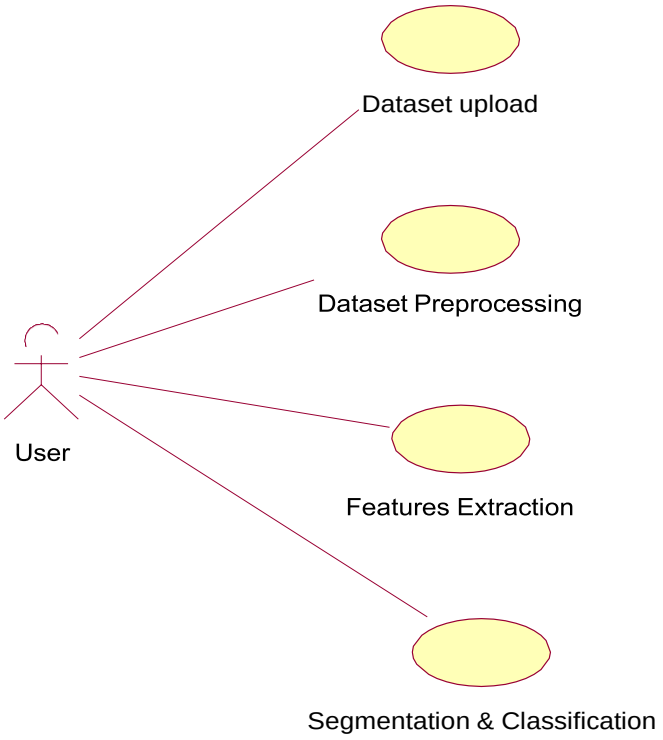



**Fig 2 Use Case Diagram**

# 5.3. SEQUENCE DIAGRAM

A **sequence diagram** is a kind of interaction diagram that shows how processes operate withgone another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations inthe Logical View of the system under development.

Sequence diagrams are sometimes called **event diagrams**, **event scenarios**, and timing diagrams.

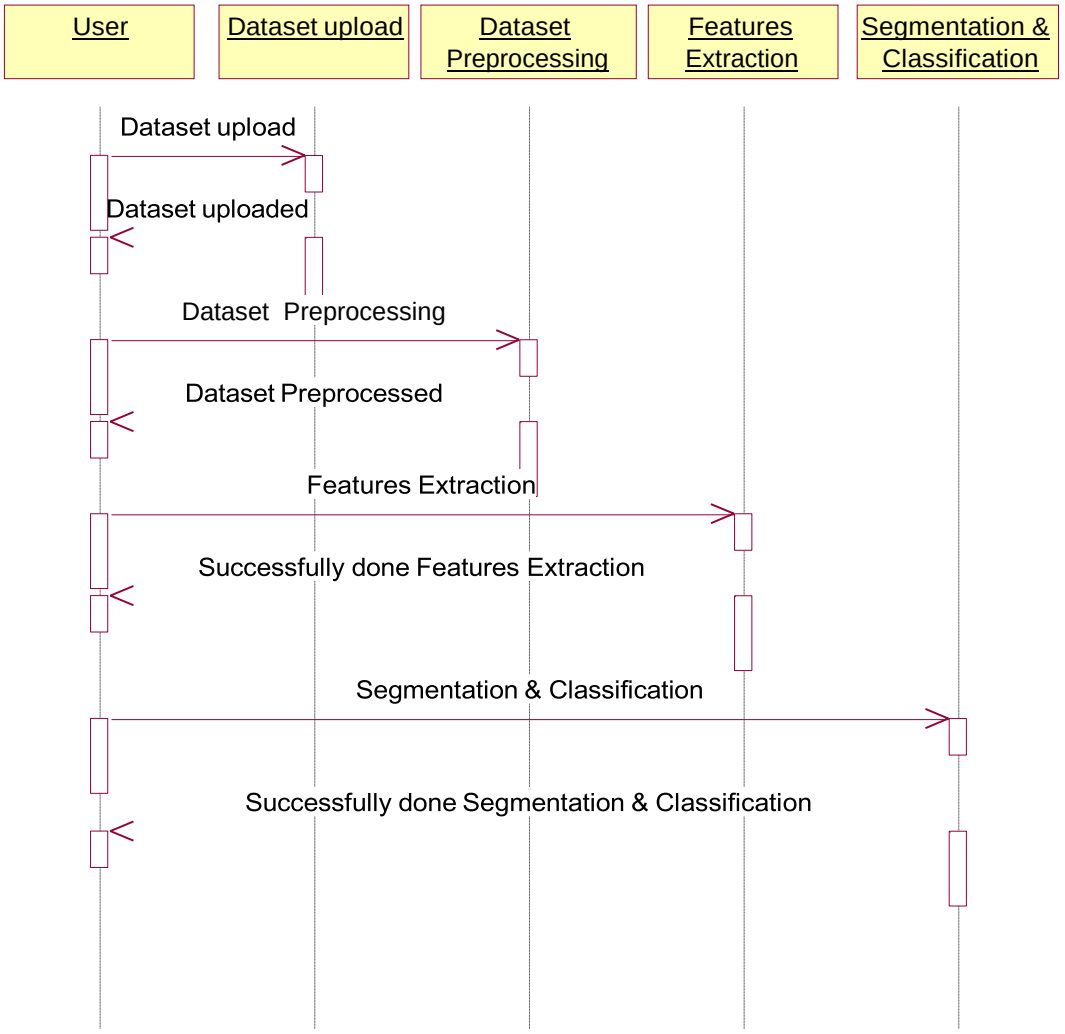



**Fig 3 sequence diagram**

## 5.4. COLLABORATION DIAGRAM

A collaboration diagram describes interactions among objects in terms of sequenced messages. Collaboration diagrams represent a combination of information taken from class, sequence, and use case diagrams describing both the static structure and dynamic behavior of a system.

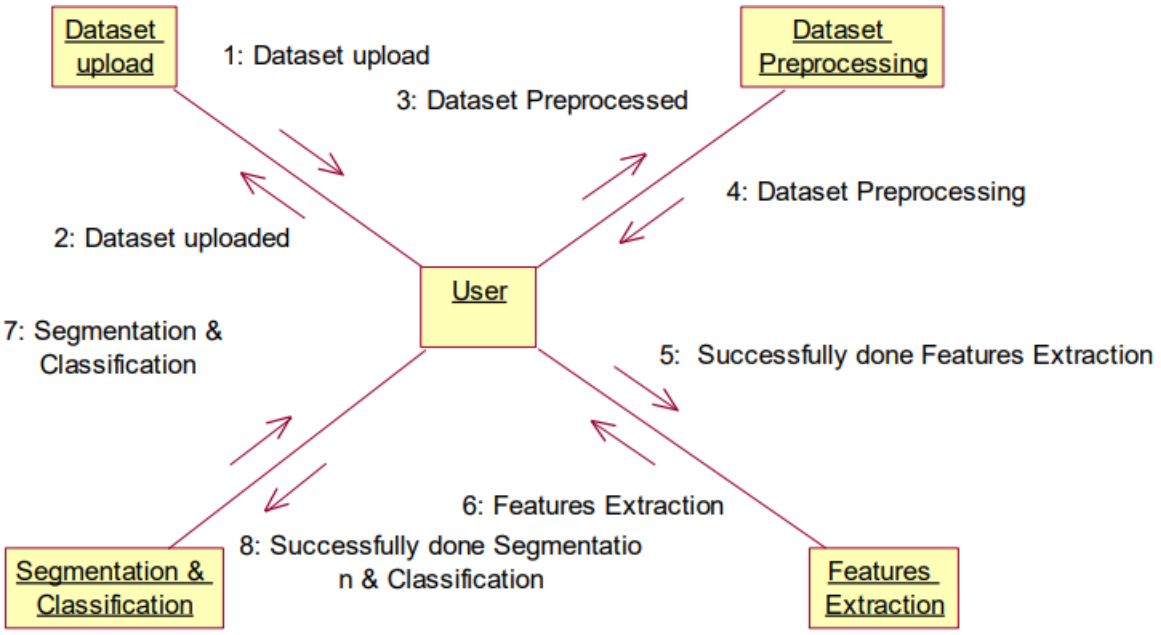



**Fig 4 Collaboration Diagram**

## 5.5. COMPONENT DIAGRAM

In the Unified Modelling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems.

Components are wired together by using an assembly connector to connect the required interface of one component with the provided interface of another component. This illustrates the service consumer - service provider relationship between the two components.

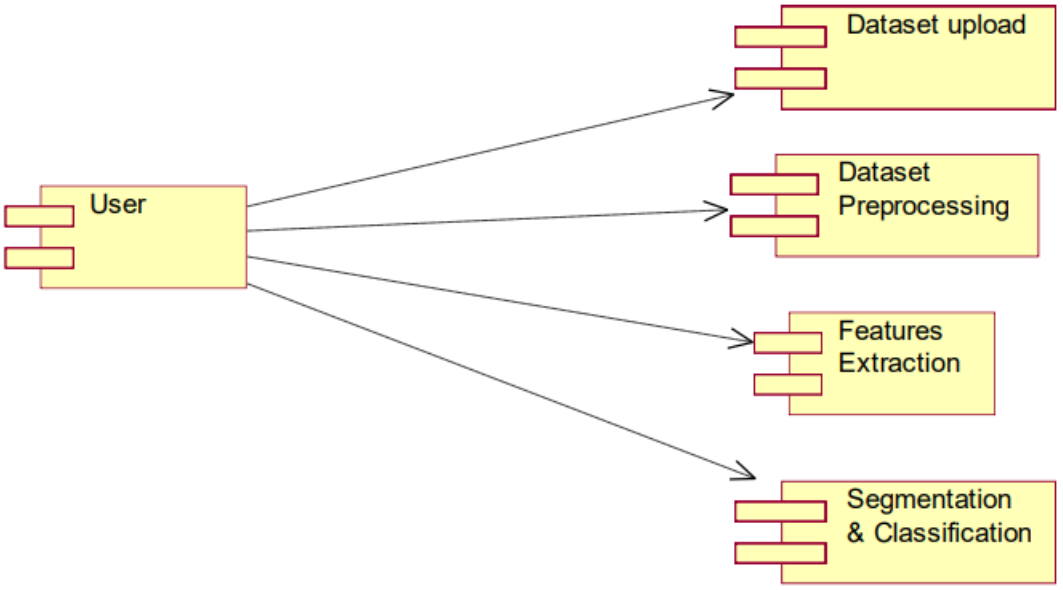



**Fig 5 Component Diagram**

## 5.6. DEPLOYMENT DIAGRAM:

A **deployment diagram** in the Unified Modeling Language models the *physical* deployment of artifacts on nodes. To describe a web site, for example, a deployment diagram would show what hardware components ("nodes") exist (e.g., a web server, an application server, and a database server), what software components ("artifacts") run on each node (e.g., web application, database), and how the different pieces are connected (e.g. JDBC, REST, RMI).

The nodes appear as boxes, and the artifacts allocated to each node appear as rectangles within the boxes. Nodes may have sub nodes, which appear as nested boxes. A single node in a deployment diagram may conceptually represent multiple physical nodes, such as a cluster of database servers.

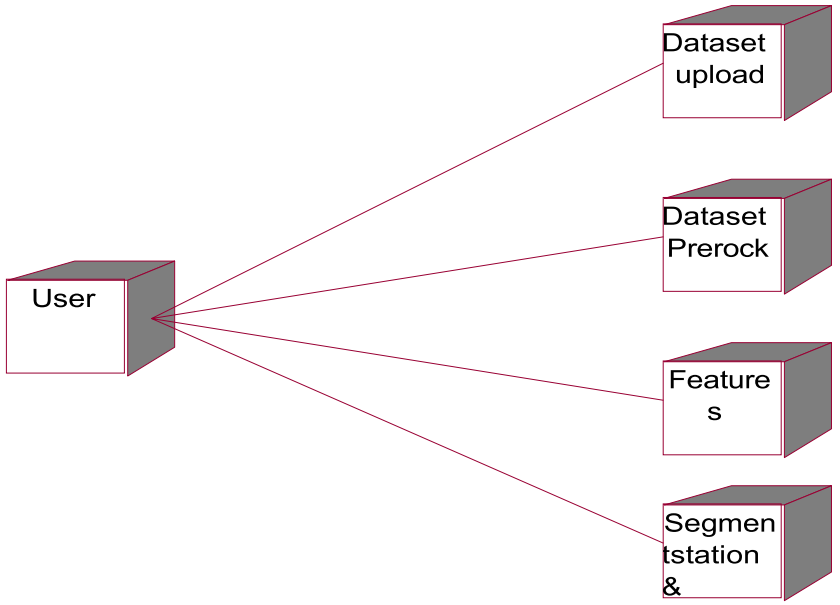



**Fig 6 Deployment Diagram**

## 5.7. ACTIVITY DIAGRAM:

**Activity diagram** is another important diagram in UML to describe dynamic aspects of the system. It is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system. So, the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent

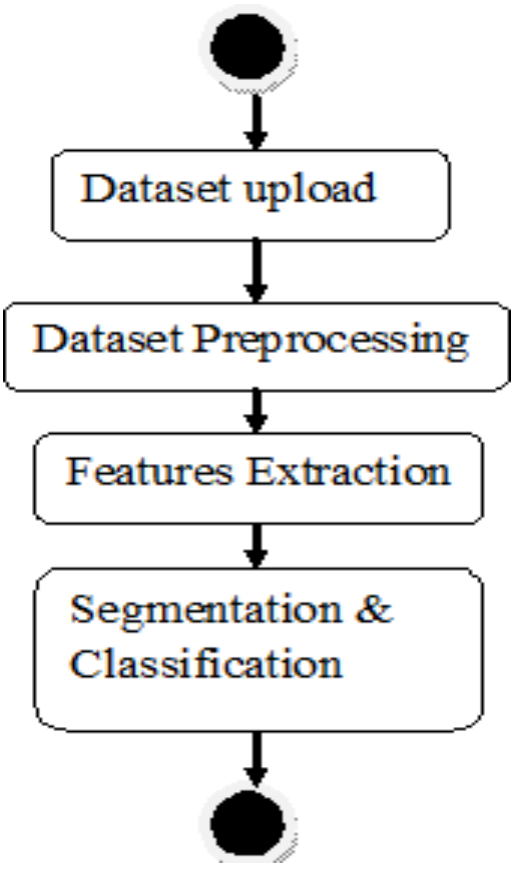



**Fig 7 Activity diagram**

## 5.8. DATA FLOW

Data flow diagrams illustrate how data is processed by a system in terms of inputs and outputs. Dataflow diagrams can be used to provide a clear representation of any business function. The technique starts with an overall picture of the business and continues by analyzing each of the functional areas of interest. This analysis can be carried out in precisely the level of detail required. The technique exploits a method called top-down expansion to conduct the analysis in a targeted way.

As the name suggests, Data Flow Diagram (DFD) is an illustration that explicates the passage of information in a process. A DFD can be easily drawn using simple symbols. Additionally, complicated processes can be easily automated by creating DFDs using easy-to-use, free downloadable diagramming tools. A DFD is a model for constructing and analyzing information processes. DFD illustrates the flow of information in a process depending upon the inputs and outputs. A DFD can also be referred to as a Process Model. A DFD demonstrates business or technical process with the support of the outside data saved, plus the data flowing from the process to another and the end results.

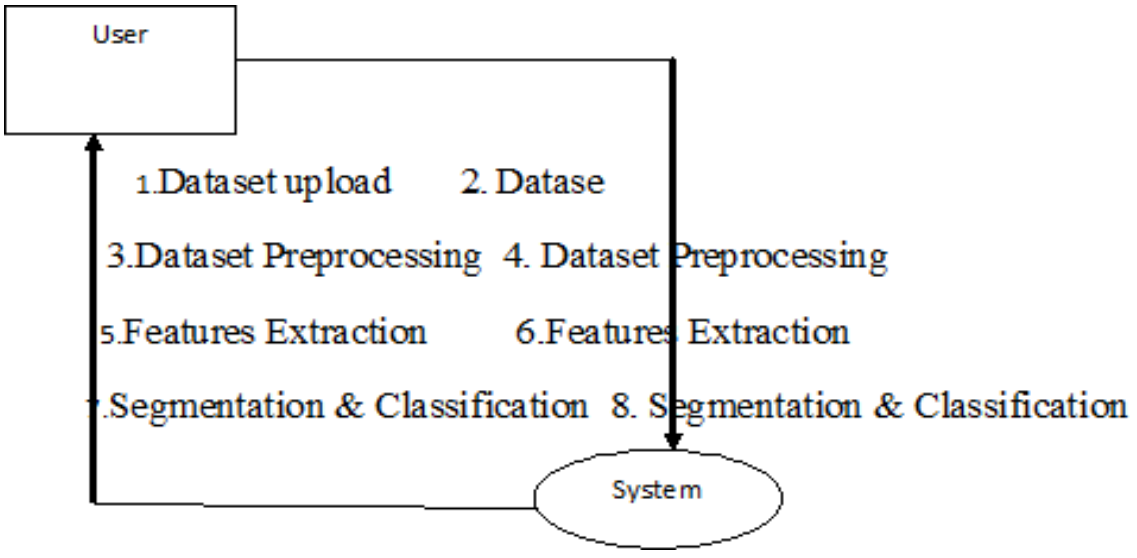



**Fig 8 Data Flow**

# CHAPTER-6
# TECHNOLOGY DESCRIPTION

# TECHNOLOGY DESCRIPTION

## 6.1 PYTHON

Below are some facts about Python.

- Python is currently the most widely used multi-purpose, high-level programming language.
- Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java.
- Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.
- Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc.

The biggest strength of Python is huge collection of standard libraries which can be used for the following –

- Machine Learning
- GUI Applications (like Kivy, Tkinter, PyQt etc.)
- Web frameworks like Django (used by YouTube, Instagram, Dropbox)
- Image processing (like OpenCV, Pillow)
- Web scraping (like Scrapy, Beautiful Soup, Selenium)
- Test frameworks
- Multimedia

### 6.1.1 ADVANTAGES OF PYTHON

Let’s see how Python dominates over other languages.

**1. Extensive Libraries**

Python downloads with an extensive library and it contain code for various purposes like regular expressions, documentation-generation, unit-testing, web browsers, threading, databases, CGI, email, image manipulation, and more. So, we don’t have to write the complete code for that manually.

**2. Extensible**

As we have seen earlier, Python can be extended to other languages. You can write some of your code in languages like C++ or C. This comes in handy, especially in projects.

**3. Embeddable**

Complimentary to extensibility, Python is embeddable as well. You can put your Python code in your source code of a different language, like C++. This lets us add scripting capabilities to our code in the other language.

**4. Improved Productivity**

The language's simplicity and extensive libraries render programmers more productive than languages like Java and C++ do. Also, the fact that you need to write less and get more things done.

**5. IOT Opportunities**

Since Python forms the basis of new platforms like Raspberry Pi, it finds the future bright for the Internet of Things. This is a way to connect the language with the real world.

**6. Simple and Easy**

When working with Java, you may have to create a class to print 'Hello World'. But in Python, just a print statement will do. It is also quite easy to learn, understand, and code. This is why when people pick up Python, they have a hard time adjusting to other more verbose languages like Java.

**7. Readable**

Because it is not such a verbose language, reading Python is much like reading English. This is the reason why it is so easy to learn, understand, and code. It also does not need curly braces to define blocks, and indentation is mandatory. These further aids the readability of the code.

**8. Object-Oriented**

This language supports both the procedural and object-oriented programming paradigms. While functions help us with code reusability, classes and objects let us model the real world. A class allows the encapsulation of data and functions into one.

**9. Free and Open-Source**

Like we said earlier, Python is freely available. But not only can you download Python for free, but you can also download its source code, make changes to it, and even distribute it. It downloads with an extensive collection of libraries to help you with your tasks.

**10. Portable**

When you code your project in a language like C++, you may need to make some changes to it if you want to run it on another platform. But it isn't the same with Python. Here, you need to code only once, and you can run it anywhere. This is called Write Once Run Anywhere (WORA). However, you need to be careful enough not to include any system-dependent features.

**11. Interpreted**

Lastly, we will say that it is an interpreted language. Since statements are executed one by one, debugging is easier than in compiled languages.

## 6.1.2 DISADVANTAGES OF PYTHON

So far, we've seen why Python is a great choice for your project. But if you choose it, you should be aware of its consequences as well. Let's now see the downsides of choosing Python over another language.

**1. Speed Limitations**

We have seen that Python code is executed line by line. But since Python is interpreted, it often results in slow execution. This, however, isn't a problem unless speed is a focal point for the project. In other words, unless high speed is a requirement, the benefits offered by Python are enough to distract us from its speed limitations.

**2. Weak in Mobile Computing and Browsers**

While it serves as an excellent server-side language, Python is much rarely seen on the client-side. Besides that, it is rarely ever used to implement smartphone-based applications. One such application is called Carbon Nelle.

The reason it is not so famous despite the existence of Bryton is that it isn't that secure.

**3. Design Restrictions**

As you know, Python is dynamically-typed. This means that you don't need to declare the type of variable while writing the code. It uses duck-typing. But wait, what's that? Well, it just means that if it looks like a duck, it must be a duck. While this is easy on the programmers during coding, it can raise run-time errors.

**4. Underdeveloped Database Access Layers**

Compared to more widely used technologies like JDBC (Java Database Connectivity) and ODBC (Open Database Connectivity), Python's database access layers are a bit underdeveloped. Consequently, it is less often applied in huge enterprises.

**5. Simple**

No, we're not kidding. Python's simplicity can indeed be a problem. Take my example. I don't do Java, I'm more of a Python person. To me, its syntax is so simple that the verbosity of Java code seems unnecessary.

This was all about the Advantages and Disadvantages of Python Programming Language.

## 6.1.3 HISTORY OF PYTHON

What do the alphabet and the programming language Python have in common? Right, both start with ABC. If we are talking about ABC in the Python context, it's clear that the programming language ABC is meant. ABC is a general-purpose programming language and programming environment, which had been developed in the Netherlands, Amsterdam, at the CWI (Centrum Wiskunde & Informatica).

The greatest achievement of ABC was to influence the design of Python was conceptualized in the late 1980s. Guido van Rossum worked that time in a project at the CWI, called Amoeba, a distributed operating system. In an interview with Bill Venners1, Guido van Rossum said: "In the early 1980s, I worked as an implementer on a team building a language called ABC at Centrum voor Wiskunde en Informatica (CWI). I don't know how well people know ABC's influence on Python.

I try to mention ABC's influence because I'm indebted to everything I learned during that project and to the people who worked on it."Later on in the same Interview, Guido van Rossum continued: "I remembered all my experience and some of my frustration with ABC. I decided to try to design a simple scripting language that possessed some of ABC's better properties, but without its problems.

So, I started typing. I created a simple virtual machine, a simple parser, and a simple runtime. I made my own version of the various ABC parts that I liked. I created a basic syntax, used indentation for statement grouping instead of curly braces or begin-end blocks, and developed a small number of powerful data types: a hash table (or dictionary, as we call it), a list, strings, and numbers."

## 6.2 MACHINE LEARNING

Before we take a look at the details of various machine learning methods, let's start by looking at what machine learning is, and what it isn't. Machine learning is often categorized as a subfield of artificial intelligence, but I find that categorization can often be misleading at first brush. The study of machine learning certainly arose from research in this context, but in the data science application of machine learning methods, it's more helpful to think of machine learning as a means of building modelsof data.

Fundamentally, machine learning involves building mathematical models to help understand data. "Learning" enters the fray when we give these models tunable parameters that can be adapted to observed data; in this way the program can be considered to be "learning" from the data. Once these models have been fit to previously seen data, they can be used to predict and understand aspects of newly observed data. I'll leave to the reader the more philosophical digression regarding the extent to which this type of mathematical, model-based "learning" is similar to the "learning" exhibited by the human brain. Understanding the problem setting in machine learning is essential to using these tools effectively, and so we will start with some broad categorizations of the types of approaches we'll discuss here.

### 6.2.1 CATEGORIES OF MACHINE LEANING:

At the most fundamental level, machine learning can be categorized into two main types: supervised learning and unsupervised learning.

Supervised learning involves somehow modeling the relationship between measured features of data and some label associated with the data; once this model is determined, it can be used to apply labels to new, unknown data. This is further subdivided into classification tasks and regression tasks: in classification, the labels are discrete categories, while in regression, the labels are continuous quantities. We will see examples of both types of supervised learning in the following section.

Unsupervised learning involves modeling the features of a dataset without reference to any label, and is often described as "letting the dataset speak for itself." These models include tasks such as clustering and dimensionality reduction. Clustering algorithms identify distinct groups of data, while

## 6.2.2 NEED FOR MACHINE LEARNING:

Human beings, at this moment, are the most intelligent and advanced species on earth because they can think, evaluate and solve complex problems. On the other side, AI is still in its initial stage and haven't surpassed human intelligence in many aspects. Then the question is that what is the need to make machine learn? The most suitable reason for doing this is, "to make decisions, based on data, with efficiency and scale".

Lately, organizations are investing heavily in newer technologies like Artificial Intelligence, Machine Learning and Deep Learning to get the key information from data to perform several real-world tasks and solve problems. We can call it data-driven decisions taken by machines, particularly to automate the process. These data-driven decisions can be used, instead of using programing logic, in the problems that cannot be programmed inherently. The fact is that we can't do without human intelligence, but other aspect is that we all need to solve real-world problems with efficiency at a huge scale. That is why the need for machine learning arises.

## 6.2.3 CHALLENGES IN MACHINES LEARNING:

While Machine Learning is rapidly evolving, making significant strides with cybersecurity and autonomous cars, this segment of AI as whole still has a long way to go. The reason behind is that ML has not been able to overcome number of challenges. The challenges that ML is facing currently are −

- **Quality of data** − Having good-quality data for ML algorithms is one of the biggest challenges. Use oflow-quality data leads to the problems related to data preprocessing and feature extraction.

- **Time-Consuming task** − Another challenge faced by ML models is the consumption of time especiallyfor data acquisition, feature extraction and retrieval.

- **Lack of specialist persons** − As ML technology is still in its infancy stage, availability of expert resources is a tough job.

- **No clear objective for formulating business problems** − Having no clear objective and well-defined goal for business problems is another key challenge for ML because this technology is not that mature yet.

- **Issue of overfitting & underfitting** − If the model is overfitting or underfitting, it cannot be representedwell for the problem.

- **Curse of dimensionality** − Another challenge ML model faces is too many features of data points. Thiscan be a real hindrance.

- **Difficulty in deployment** − Complexity of the ML model makes it quite difficult to be deployed in reallife.

## 6.2.4 APPLICATIONS OF MACHINES LEARNING:

Machine Learning is the most rapidly growing technology and according to researchers we are in the golden year of AI and ML. It is used to solve many real-world complex problems which cannot be solved with traditional approach. Following are some real-world applications of ML –

- Emotion analysis
- Sentiment analysis
- Error detection and prevention
- Weather forecasting and prediction
- Stock market analysis and forecasting
- Speech synthesis
- Speech recognition
- Customer segmentation
- Object recognition
- Fraud detection
- Fraud prevention
- Recommendation of products to customer in online shopping

## 6.2.5. NUMPY

NumPy, which stands for Numerical Python, is a fundamental package for numerical computing in Python. It provides support for multidimensional arrays (ND arrays), along with a collection of functions and tools for working with these arrays efficiently. Here's a brief overview of NumPy's key features-

- **Multidimensional Arrays (ND arrays):**

NumPy's ND array is a powerful data structure that represents arrays of any dimension.

ND arrays offer efficient storage and manipulation of large datasets, making them ideal for numericalcomputations.

- **Array Creation and Manipulation:**

NumPy provides various functions for creating arrays, such as NumPy. Array (), NumPy. Eros (), NumPy. Ones (), NumPy. Range (), NumPy. Lin space (), etc.

It offers extensive capabilities for manipulating arrays, including indexing, slicing, reshaping, concatenating, and splitting.

- **Element-Wise Operations:**

NumPy supports element-wise operations on arrays, enabling efficient computation without the need for explicit loops.

Arithmetic operations, mathematical functions, logical operations, and bitwise operations can be applied element-wise to arrays.

- **Broadcasting:**

Broadcasting is a powerful feature in NumPy that allows operations between arrays of different shapes. NumPy automatically broadcasts arrays to perform element-wise operations when their shapes are compatible.

- **Linear Algebra Operations:**

NumPy provides a comprehensive suite of linear algebra functions for performing operations such as matrix multiplication (numpy.dot ()), matrix inversion (numpy.linalg.inv ()), eigenvalue decomposition (numpy.linalg.eig ()), and more.

- **Random Number Generation:**

NumPy's NumPy. Random module offers functions for generating random numbers and random arrays with various distributions.

These functions are useful for tasks such as random sampling, permutation, and generating synthetic data for simulations.

- **Array Operations and Manipulations:**

NumPy includes functions for array manipulation, including sorting (NumPy. Sort ()), searching (NumPy. Where ()), unique elements (NumPy. Unique ()), stacking and splitting arrays, and more.

- **Performance and Efficiency:**

NumPy is implemented in C and Fortran, making it highly efficient and fast for numerical computations.

It leverages optimized algorithms and memory layout for improved performance, especially when working with large datasets.

## 6.3 DEEP LEARNING FRAMEWORKS

Deep learning is a subset of machine learning that focuses on training artificial neural networks with multiple layers to learn intricate patterns and representations from complex data. It aims to automatically discover hierarchical features through successive layers of abstraction, mimicking the structure and function of the human brain. Deep learning has revolutionized various fields such as computer vision, natural language processing, speech recognition, and reinforcement learning, achieving remarkable performance in tasks like image classification, object detection, language translation, and game playing. Key techniques in deep learning include Convolutional Neural Networks (CNNs) for

processing spatial data, Recurrent Neural Networks (RNNs) for handling sequential data, and Transformer models for capturing long-range dependencies in sequences. Deep learning has propelled advancements in artificial intelligence, enabling machines to perceive, understand, and generate data in ways previously thought impossible, and continues to drive innovation across diverse domains.

### 6.3.1 KERAS:

- Keras is a high-level neural networks API written in Python, capable of running on top of TensorFlow, Microsoft Cognitive Toolkit (CNTK), or Theano.
- It provides a user-friendly interface for building, training, and deploying deep learning models with minimal code.
- Keras allows easy experimentation with different architectures and hyperparameters, making it ideal for both beginners and experienced researchers.

### 6.3.2 TENSORFLOW:

- TensorFlow is an open-source machine learning framework developed by Google.
- It provides a comprehensive ecosystem of tools, libraries, and community resources for building and deploying machine learning models.
- TensorFlow offers flexibility, scalability, and performance for a wide range of tasks, including deep learning, reinforcement learning, and symbolic mathematics.

## 6.3. OPENCV (OPEN-SOURCE COMPUTER VISION LIBRARY):

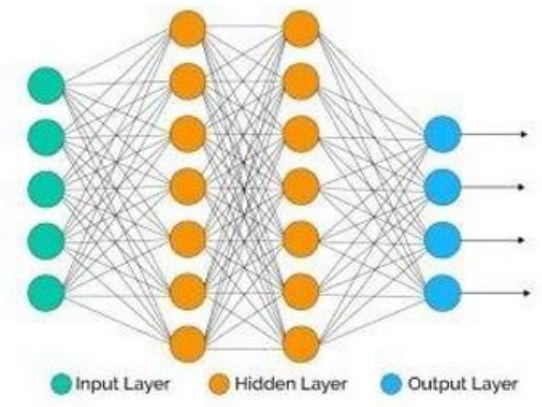
- OpenCV is a popular open-source computer vision and machine learning software library.
- It provides a wide range of functionalities for image and video processing, including feature detection, object recognition, image segmentation, and camera calibration.
- OpenCV is widely used in various applications such as robotics, augmented reality, face recognition, and medical image analysis.

### ➢ Feature Extraction and Representation:

The representation of an image as a 3D matrix having dimension as of height and width of the image and the value of each pixel as depth (1 in case of Grayscale and 3 in case of RGB). Further, these pixel values are used for extracting useful features using CNN.

- **Artificial Neural Networks:**

Artificial Neural Network is a connection of neurons, replicating the structure of human brain. Each connection of neuron transfers information to another neuron. Inputs are fed into first layer of neurons which processes it and transfers to another layer of neurons called as hidden layers. After processing of information through multiple layers of hidden layers, information is passed to final output layer.




There are capable of learning and they have to be trained. There are different learning strategies:

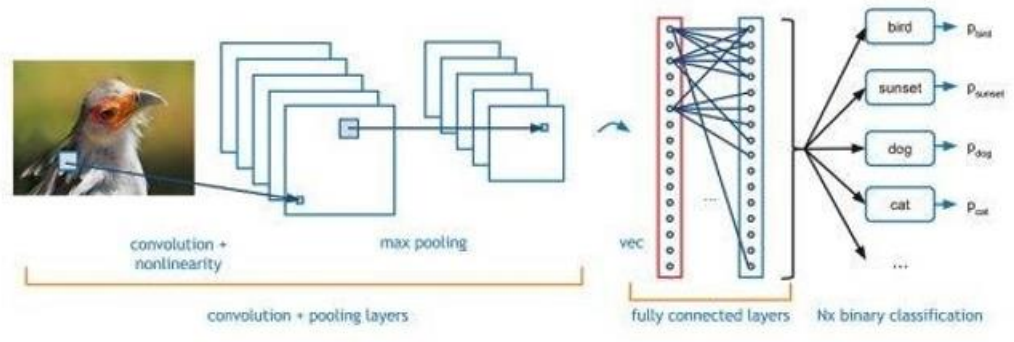
1. Unsupervised Learning

2.Supervised Learning 3.

Reinforcement Learning

## 6.3.4 CONVOLUTION NEURAL NETWORK:

Unlike regular Neural Networks, in the layers of CNN, the neurons are arranged in 3 dimensions: width, height, depth. The neurons in a layer will only be connected to a small region of the layer (window size) before it, instead of all of the neurons in a fully-connected manner. Moreover, the final output layer would have dimensions (number of classes), because by the end of the CNN architecture we will reduce the full image into a single vector of class scores.




## 1. Convolution Layer:

In convolution layer we take a small window size [typically of length 5*5] that extends to the depth of the input matrix. The layer consists of learnable filters of window size. During every iteration we slid the window by stride size [typically 1], and compute the dot product of filter entries and input values at a given position. As we continue this process well create a 2-Dimensional activation matrix that gives the response of that matrix at every spatial position. That is, the network will learn filters that activate when they see some type of visual feature such as an edge of some orientation or a blotch of some color.

## 2. Pooling Layer:

We use pooling layer to decrease the size of activation matrix and ultimately reduce the learnable parameters. There are two types of pooling,

### a) Max Pooling:

In max pooling we take a window size [for example window of size 2*2], and only

take the maximum of 4 values. Well lid this window and continue this process, so well finally get

anactivation matrix half of its original Size.

**b) Average Pooling:** In average pooling we take average of all values in a window.

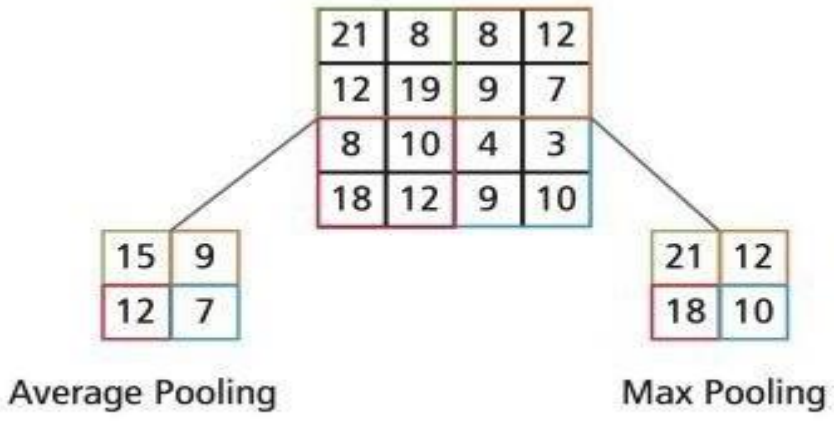



Figure 5.3: Types of pooling

**Fully Connected Layer:** In convolution layer neurons are connected only to a local region, while in a fully connected region, well connect the all the inputs to neurons

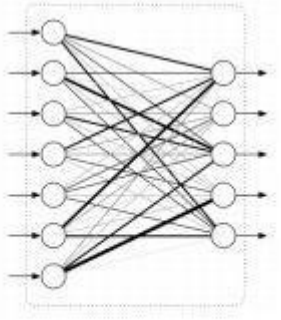


Figure 5.4: Fully Connected Layer

**Final Output Layer:** After getting values from fully connected layer, well connect them to final layer of neurons [having count equal to total number of classes], that will predict the probability of each image to be in different classes.

### 6.3.5 VS CODE

Visual Studio Code (VS Code) is a free, open-source source code editor developed by Microsoft. It has quickly gained popularity among developers due to its lightweight, fast, and highly customizable nature. Here's a brief overview of VS Code-

- **Cross-Platform Support:**

VS Code is available for Windows, macOS, and Linux, providing a consistent development experience across different operating systems.

- **Rich Feature Set:**

VS Code comes with a rich set of features, including syntax highlighting, code completion, linting, debugging, version control integration, and built-in terminal.

- **Extensions Ecosystem:**

One of the most powerful aspects of VS Code is its extensive extension ecosystem. Developers can enhance their workflow by installing extensions for additional languages, frameworks, tools, and themes.

VS Code supports various programming languages out of the box, including JavaScript, Python, Java, C++, and many more. Additionally, there are thousands of community-contributed extensions available in the Visual Studio Code Marketplace.

- **Intelligent Code Editing:**

VS Code provides intelligent code editing features such as IntelliSense (code completion), code refactoring, and automatic formatting, which help developers write code faster and with fewer errors.

- **Integrated Terminal:**

VS Code includes an integrated terminal that allows developers to run command-line tools and scripts without leaving the editor, streamlining the development workflow.

- **Built-in Git Integration:**

Git integration is built directly into VS Code, allowing developers to perform version control operations (such as committing, branching, and merging) without switching to a separate Git client.

- **Customizable and Configurable:**

VS Code is highly customizable and configurable, enabling developers to personalize their editor experience by adjusting settings, key bindings, and themes according to their preferences.

- **Active Development and Community Support:**

VS Code is actively developed by Microsoft, with regular updates and new features being released frequently. It also has a large and vibrant community of users and contributors who actively participate in discussions, issue reporting, and extension development.

# CHAPTER-7
# MODULE DESCRIPTION

# MODULE DESCRIPTION

Our provided code is designed to perform hand gesture recognition using a Convolutional Neural Network (CNN) with Keras and OpenCV. Below is the module design summarizing the functionalities of different parts of the code-

## 7.1 User Interface Module:

- This module is responsible for creating a graphical user interface (GUI) using the Tkinter library.
- It includes buttons for uploading the hand gesture dataset, training the CNN model, uploading test images for classification, and recognizing gestures from video input.
- The module displays text information and updates the GUI accordingly.

## 7.2 Data Processing Module:

- The data processing module involves functions for loading and preprocessing the hand gesture dataset.
- It includes functionality to resize, convert to grayscale, and normalize the images before feeding them into the CNN model.

## 7.3 CNN Model Module:

- This module defines the architecture of the CNN model using Keras Sequential API.
- It includes convolutional layers, max-pooling layers, flattening layer, fully connected layers, and an output layer.
- The model is compiled with appropriate optimizer, loss function, and evaluation metrics.

## 7.4 Training Module:

- The training module is responsible for training the CNN model using the hand gesture dataset.
- It loads the preprocessed dataset, initializes the CNN model, and trains it using the fit method.
- The trained model and its weights are saved for future use.

## 7.5 Image Classification Module:

- This module provides functionality to upload test images and classify hand gestures using thetrained CNN model.
- It preprocesses the test images similar to the training images and feeds them into the trainedmodel for prediction.
- The predicted gesture labels are displayed on the GUI along with the original test images.

# CHAPTER-8
# CODE

# CODE

## Code for Bone Tumor:

```
from tkinter import message box

from tkinter import *

from tkinter import simple dialog

import tkinter

from tkinter import file dialog

import matplotlib.pyplot as plt

import NumPy as np

from tkinter.filedialog import askopenfilename

import os

import cv2

from sklearn.model_selection import train_test_split

from sklearn.metrics import accuracy_score

import imutils

from keras.utils.np_utils import to_categorical

from keras.layers import MaxPooling2D

from keras.layers import Dense, Dropout, Activation, Flatten

from keras.layers import Convolution2D

from keras.models import Sequential

from keras.models import model_from_json

import pickle

from sklearn import metrics

import ftplib
```

```
from tkinter import ttk


main = tkinter.Tk()
main.title("Identifying Bone Tumor using X-Ray Images") #designing main screen
main.geometry("1300x1200")


global filename
global accuracy
X = []
Y = []
global classifier
disease = ['No Tumor Detected','Tumor Detected']


with open('Model/segmented_model.json', "r") as json_file:
    loaded_model_json = json_file.read()
    segmented_model = model_from_json(loaded_model_json)
json_file.close()
segmented_model.load_weights("Model/segmented_weights.h5")
segmented_model._make_predict_function()


def edgeDetection():
    img = cv2.imread('myimg.png')
    orig = cv2.imread('test1.png')
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    thresh = cv2.threshold(gray, 30, 255, cv2.THRESH_BINARY)[1]
    contours = cv2.findContours(thresh, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
```

```
    contours = contours[0] if len(contours) == 2 else contours[1]
    min_area = 0.95*180*35
    max_area = 1.05*180*35
    result = orig.copy()
    for c in contours:
        area = cv2.contourArea(c)
        cv2.drawContours(result, [c], -1, (0, 0, 255), 10)
        if area > min_area and area < max_area:
            cv2.drawContours(result, [c], -1, (0, 255, 255), 10)
    return result


def tumorSegmentation(filename):
    global segmented_model
    img = cv2.imread(filename,0)
    img = cv2.resize(img,(64,64), interpolation = cv2.INTER_CUBIC)
    img = img.reshape(1,64,64,1)
    img = (img-127.0)/127.0
    preds = segmented_model.predict(img)
    preds = preds[0]
    print(preds.shape)
    orig = cv2.imread(filename,0)
    orig = cv2.resize(orig,(300,300),interpolation = cv2.INTER_CUBIC)
    cv2.imwrite("test1.png",orig)
    segmented_image = cv2.resize(preds,(300,300),interpolation = cv2.INTER_CUBIC)
    cv2.imwrite("myimg.png",segmented_image*255)
    edge_detection = edgeDetection()
```

```
    return segmented_image*255, edge_detection




def uploadDataset(): #function to upload dataset
    global filename
    filename = filedialog.askdirectory(initialdir=".")
    text.delete('1.0', END)
    text.insert(END,filename+" loaded\n");


def datasetPreprocessing():
    global X
    global Y
    X.clear()
    Y.clear()
    if os.path.exists('Model/myimg_data.txt.npy'):
        X = np.load('Model/myimg_data.txt.npy')
        Y = np.load('Model/myimg_label.txt.npy')
    else:
        for root, dirs, directory in os.walk(filename+"/no"):
            for i in range(len(directory)):
                name = directory[i]
                img = cv2.imread(filename+"/no/"+name,0) #reading images
                ret2,th2 = cv.threshold(img,0,255,cv.THRESH_BINARY+cv.THRESH_OTSU)
#processing and normalization images
                img = cv2.resize(img, (128,128)) #resizing images
                im2arr = np.array(img) #extract features from images
                im2arr = im2arr.reshape(128,128,1)
```

```
            X.append(im2arr)
            Y.append(0)
            print(filename+"/no/"+name)

    for root, dirs, directory in os.walk(filename+"/yes"):
        for i in range(len(directory)):
            name = directory[i]
            img = cv2.imread(filename+"/yes/"+name,0)
            ret2,th2 = cv.threshold(img,0,255,cv.THRESH_BINARY+cv.THRESH_OTSU)
            img = cv2.resize(img, (128,128))
            im2arr = np.array(img)
            im2arr = im2arr.reshape(128,128,1)
            X.append(im2arr)
            Y.append(1)
            print(filename+"/yes/"+name)

    X = np.asarray(X)
    Y = np.asarray(Y)
    np.save("Model/myimg_data.txt",X)
    np.save("Model/myimg_label.txt",Y)
print(X.shape)
print(Y.shape)
print(Y)
cv2.imshow('ss',X[20])
cv2.waitKey(0)
text.insert(END,"Total number of images found in dataset : "+str(len(X))+"\n")
```

```
    text.insert(END,"Total number of classes : "+str(len(set(Y)))+"\n\n")
    text.insert(END,"Class labels found in dataset : "+str(disease))



def trainTumorDetectionModel():
    global accuracy
    global classifier

    YY = to_categorical(Y)

    indices = np.arange(X.shape[0])
    np.random.shuffle(indices)

    x_train = X[indices]
    y_train = YY[indices]

    if os.path.exists('Model/model.json'):
        with open('Model/model.json', "r") as json_file:
            loaded_model_json = json_file.read()
            classifier = model_from_json(loaded_model_json)

        classifier.load_weights("Model/model_weights.h5")
        classifier._make_predict_function()
    else:
        X_trains, X_tests, y_trains, y_tests = train_test_split(x_train, y_train, test_size = 0.2,
random_state = 0)
        classifier = Sequential()
```

```
        classifier.add(Convolution2D(32, 3, 3, input_shape = (128, 128, 1), activation = 'relu'))

        classifier.add(MaxPooling2D(pool_size = (2, 2)))

        classifier.add(Convolution2D(32, 3, 3, activation = 'relu'))

        classifier.add(MaxPooling2D(pool_size = (2, 2)))

        classifier.add(Flatten())

        classifier.add(Dense(output_dim = 128, activation = 'relu'))

        classifier.add(Dense(output_dim = 2, activation = 'softmax'))

        print(classifier.summary())

        classifier.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics
 = ['accuracy'])

        hist = classifier.fit(x_train, y_train, batch_size=16, epochs=10,validation_split=0.2,
shuffle=True, verbose=2)

        classifier.save_weights('Model/model_weights.h5')

        model_json = classifier.to_json()

        with open("Model/model.json", "w") as json_file:

            json_file.write(model_json)

        f = open('Model/history.pckl', 'wb')

        pickle.dump(hist.history, f)

        f.close()

    f = open('Model/history.pckl', 'rb')

    data = pickle.load(f)

    f.close()

    acc = data['accuracy']

    accuracy = acc[4] * 100

    text.insert(END,'\n\nCNN Bone Tumor Model Generated. See black console to view layers of
CNN\n\n')

    text.insert(END,"CNN Bone Tumor Prediction Accuracy on Test Images : "+str(accuracy)+"\n")
```

```
def tumorClassification():
    filename = filedialog.askopenfilename(initialdir="testImages")
    img = cv2.imread(filename,0)
    img = cv2.resize(img, (128,128))
    im2arr = np.array(img)
    im2arr = im2arr.reshape(1,128,128,1)
    XX = np.asarray(im2arr)

    predicts = classifier.predict(XX)
    print(predicts)
    cls = np.argmax(predicts)
    print(cls)
    if cls == 0:
        img = cv2.imread(filename)
        img = cv2.resize(img, (800,500))
        cv2.putText(img, 'Classification Result : '+disease[cls], (10,
25),  cv2.FONT_HERSHEY_SIMPLEX,0.7, (0, 255, 255), 2)
        cv2.imshow('Classification Result : '+disease[cls], img)
        cv2.waitKey(0)
    if cls == 1:
        segmented_image, edge_image = tumorSegmentation(filename)
        img = cv2.imread(filename)
        img = cv2.resize(img, (800,500))
        cv2.putText(img, 'Classification Result : '+disease[cls], (10,
25),  cv2.FONT_HERSHEY_SIMPLEX,0.7, (0, 255, 255), 2)
        cv2.imshow('Classification Result : '+disease[cls], img)
```

```
        cv2.imshow("Tumor Segmented Image",segmented_image)


  cv2.imshow("Edge Detected Image",edge_image)
        cv2.waitKey(0)



def graph():
    f = open('model/history.pckl', 'rb')
    data = pickle.load(f)
    f.close()

    accuracy = data['accuracy']
    loss = data['loss']

    plt.figure(figsize=(10,6))
    plt.grid(True)
    plt.xlabel('Training Epoch')
    plt.ylabel('Accuracy/Loss')
    plt.plot(loss, 'ro-', color = 'red')
    plt.plot(accuracy, 'ro-', color = 'green')
    plt.legend(['Loss', 'Accuracy'], loc='upper left')
    plt.title('Bone Tumor CNN Model Training Accuracy & Loss Graph')
    plt.show()

font = ('times', 16, 'bold')
title = Label(main, text='Identifying Bone Tumor using X-Ray Images')
```

```
title.config(bg='darkviolet', fg='gold')

title.config(font=font)

title.config(height=3, width=120)

title.place(x=0,y=5)


font1 = ('times', 12, 'bold')

text=Text(main,height=20,width=130)

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=300,y=120)

text.config(font=font1)


font1 = ('times', 12, 'bold')

uploadButton = Button(main, text="Upload Tumor X-Ray Images Dataset",
command=uploadDataset)

uploadButton.place(x=300,y=605)

uploadButton.config(font=font1)


preprocessButton = Button(main, text="Dataset Preprocessing & Features Extraction",
command=datasetPreprocessing)

preprocessButton.place(x=725,y=605)

preprocessButton.config(font=font1)


cnnButton = Button(main, text="Trained CNN Bone Tumor Detection Model",
command=trainTumorDetectionModel)

cnnButton.place(x=1220,y=605)

cnnButton.config(font=font1)
```

```
classify Button = Button(main, text="Bone Tumor Segmentation &
Classification",command=tumorClassification)

classifyButton.place(x=300,y=685)

classifyButton.config(font=font1)



graph Button = Button(main, text="Training Accuracy Graph",

command=graph)graphButton.place(x=800,y=685)

graphButton.config(font=font1)



main.config(bg='turquoise')

main.mainloop()
```

# CHAPTER-9
# RESULTS

# RESULTS

## SCREENSHOTS

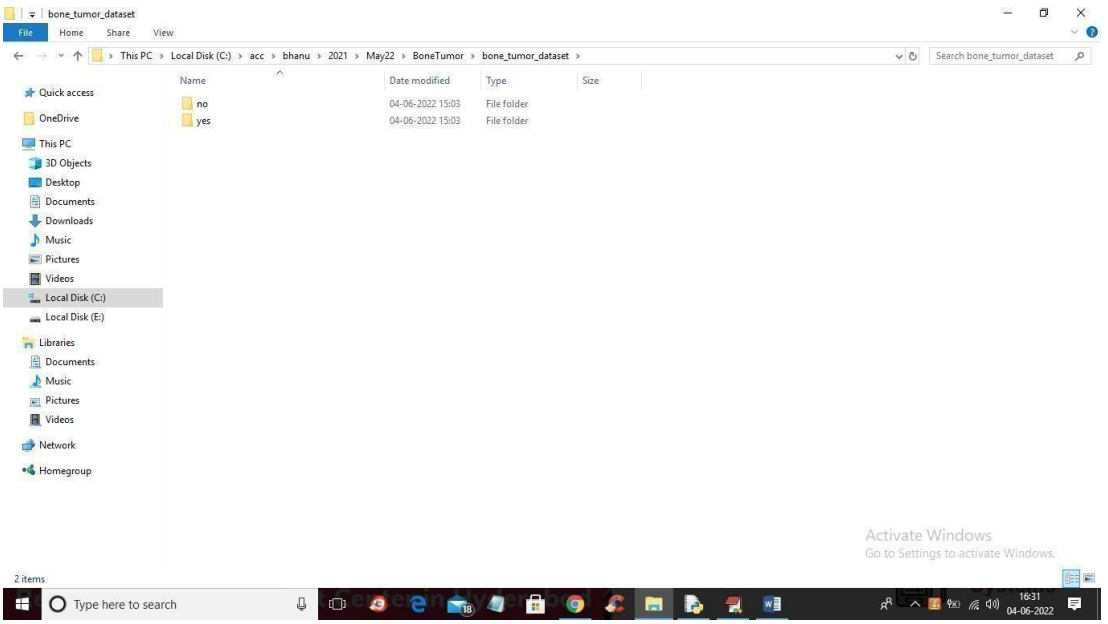
Identifying Bone Tumor using X-Ray Images

In this project we are implementing deep learning Convolution Neural Network (CNN) to predict bone tumor and to train this algorithm we have used bone images with and without tumor.

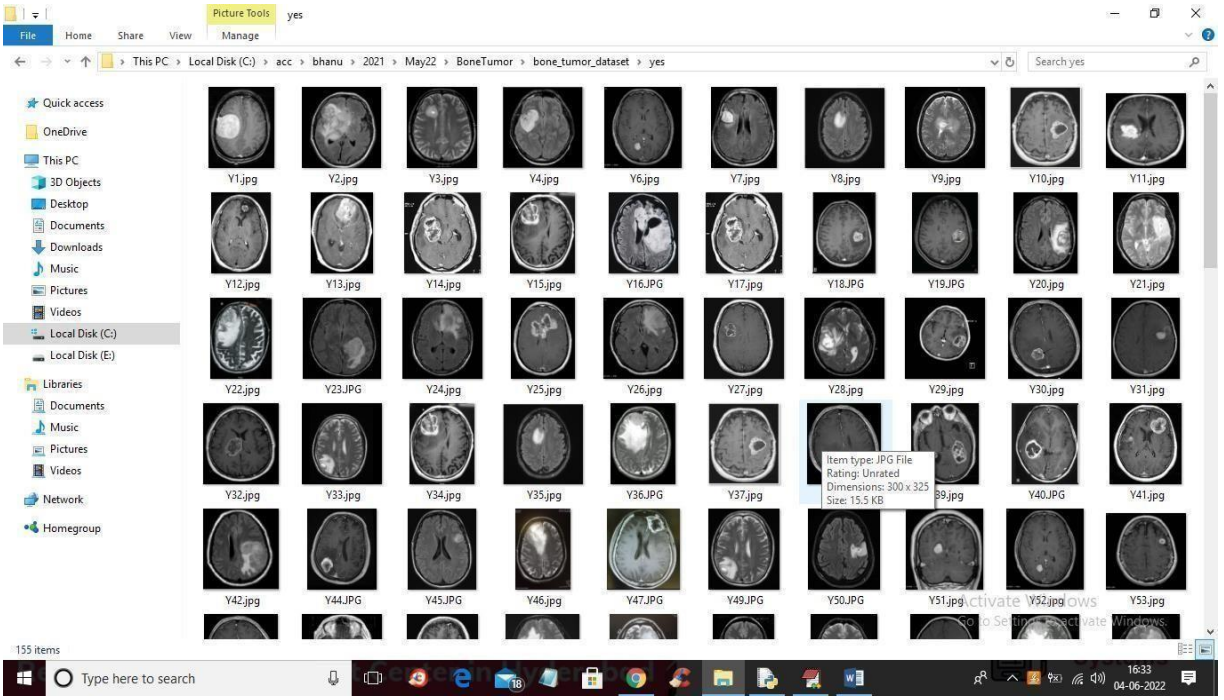
To implement this project, we have designed following modules

- Dataset upload: using this module we will upload dataset to application
- Dataset Preprocessing: using this module we will read all images and then convert them into GREY format and then resize all images to equal size and then normalize pixel values.
- Features Extraction: features or pixel values will be extracted from processed images and then input this feature to CNN to trained tumor prediction model
- Segmentation & Classification: using this module we will read test image and then apply segmentation to extract tumor part and then predict weather image is normal or contains any tumor. Edge detection technique will be applied to surround bounding box across tumor part

To implement this project, we have used below dataset

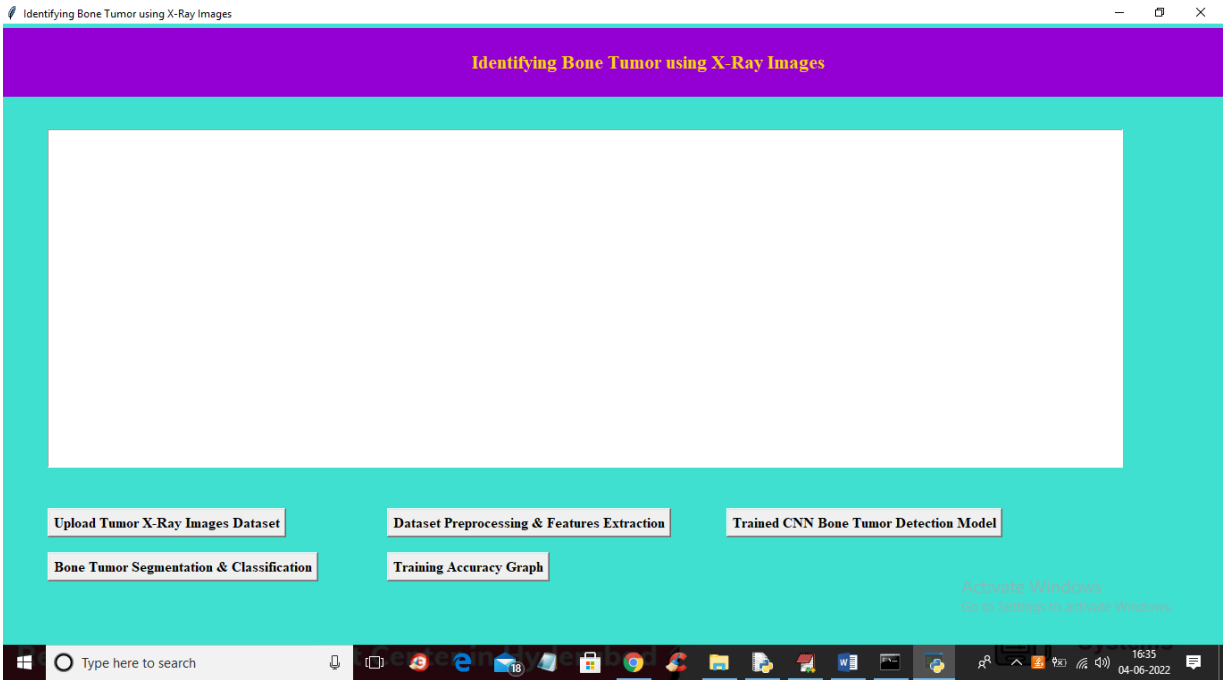



In above screen we have 2 folders called 'no and yes' where no folder contains normal bone images and 'yes' folder contains bone tumor images and just go inside any folder to view images like below screen

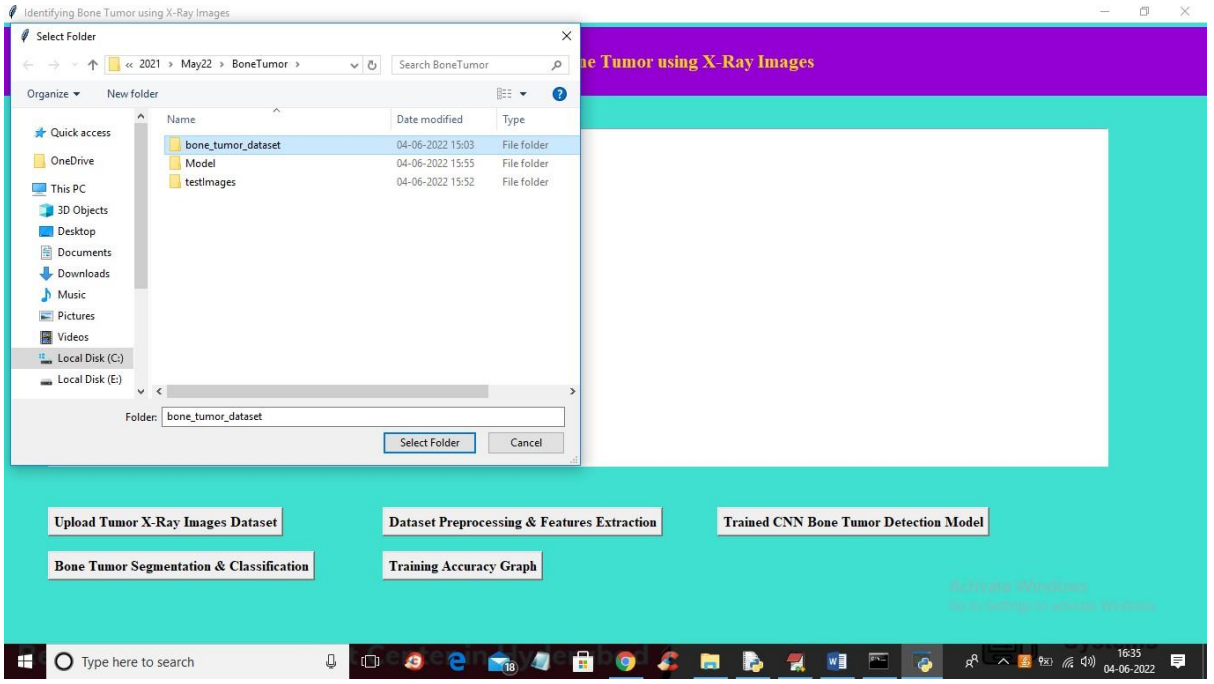



We are using above images to train CNN for tumor detection

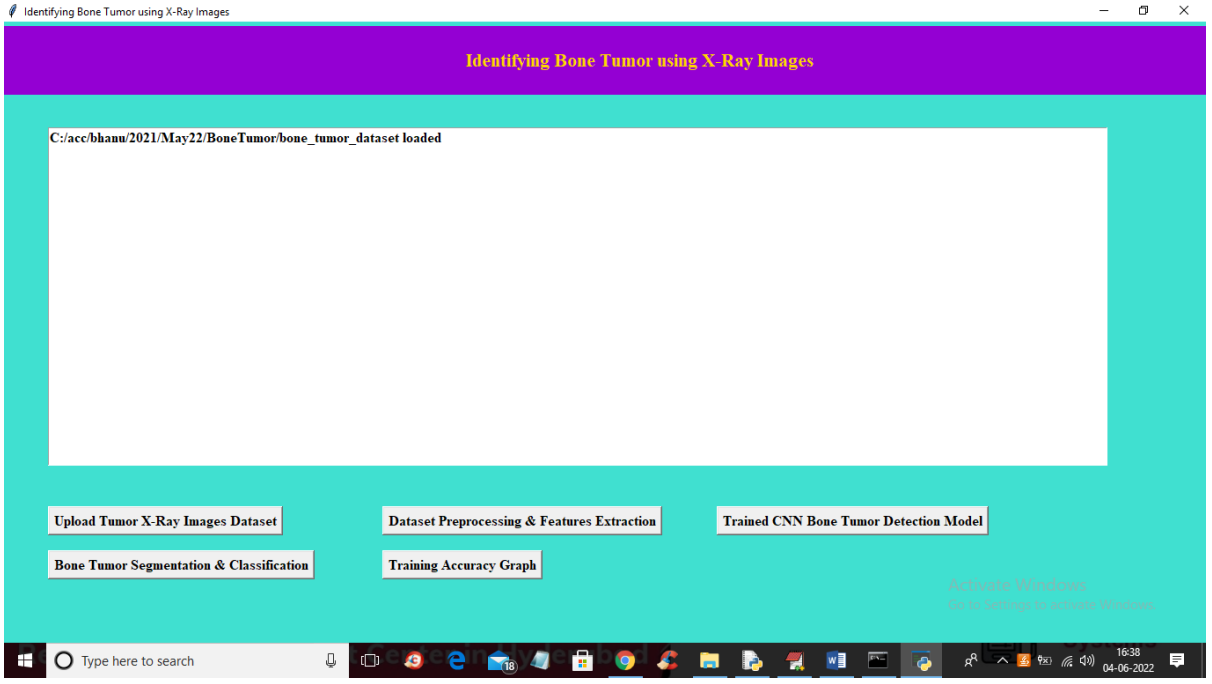
To run project double, click on run.bat file to get below screen



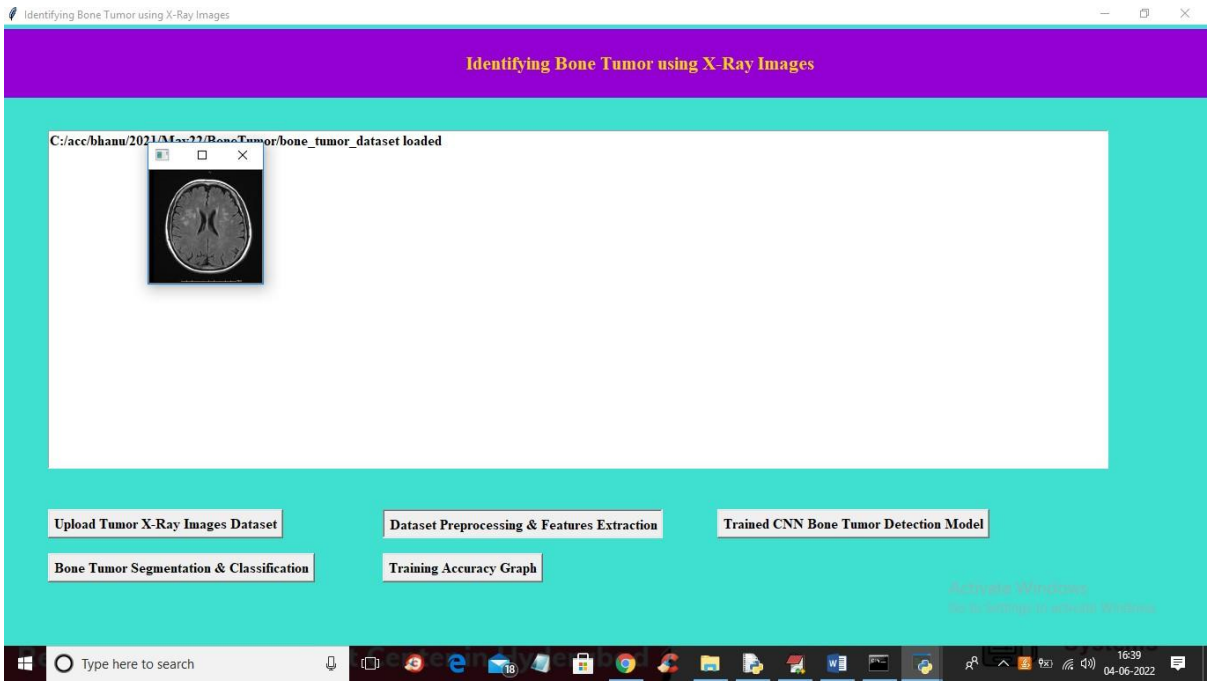

In above screen click on ‘Upload Tumor X-Ray Images Dataset’ button to upload X-Ray images dataset and get below output



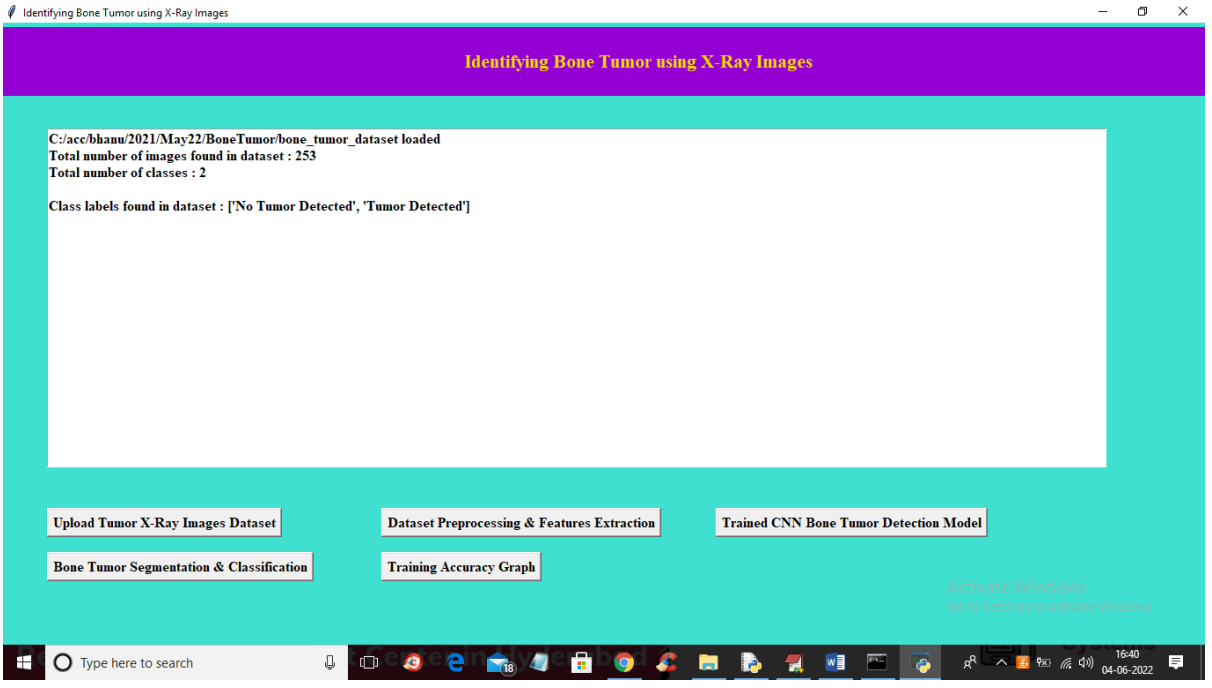

In above screen selecting and uploading brain tumor dataset and then click on 'Select Folder' button to load dataset and then get below output



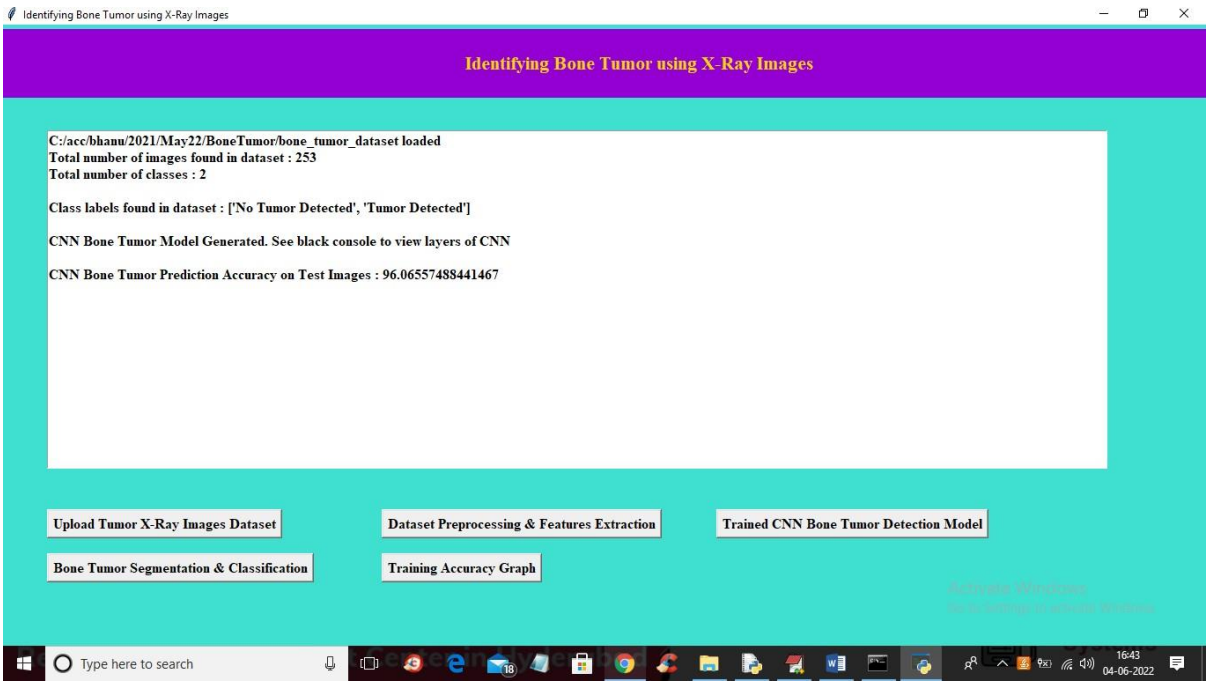

In above screen dataset loaded and now click on 'Dataset Preprocessing & Features Extraction' button to read all images and then process and extract features to train with CNN



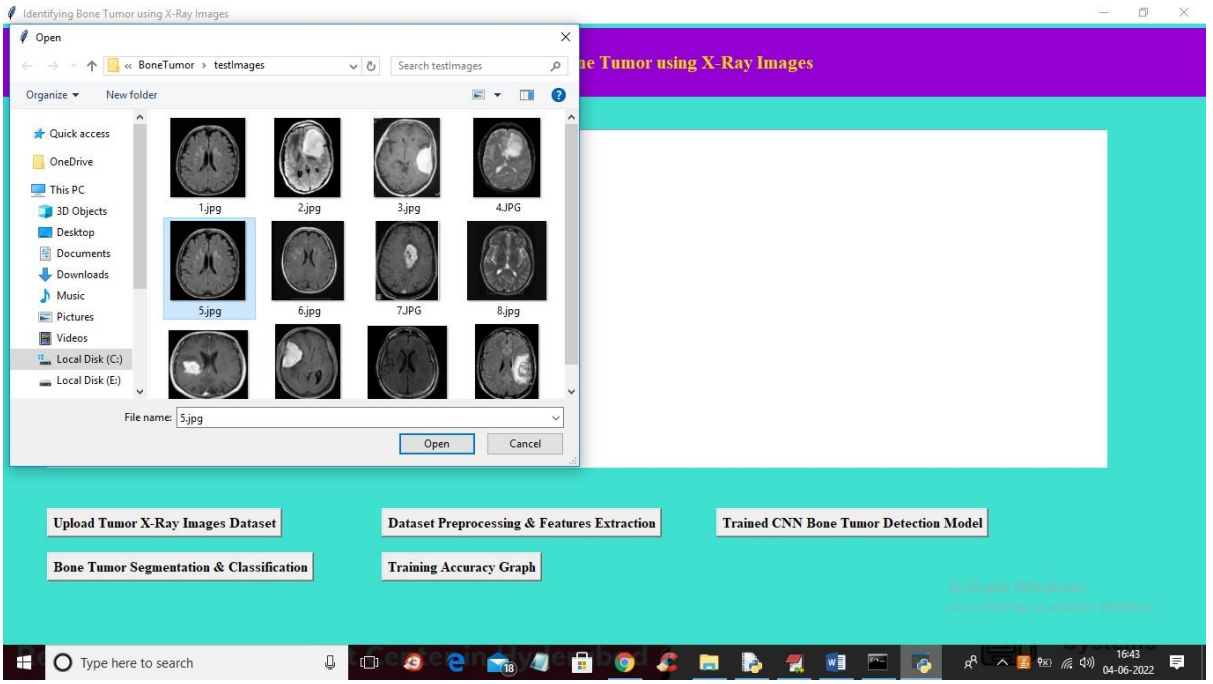

In above screen all images are processed and to check images are loaded properly so I am displaying one sample processed image and now close that image to get below output



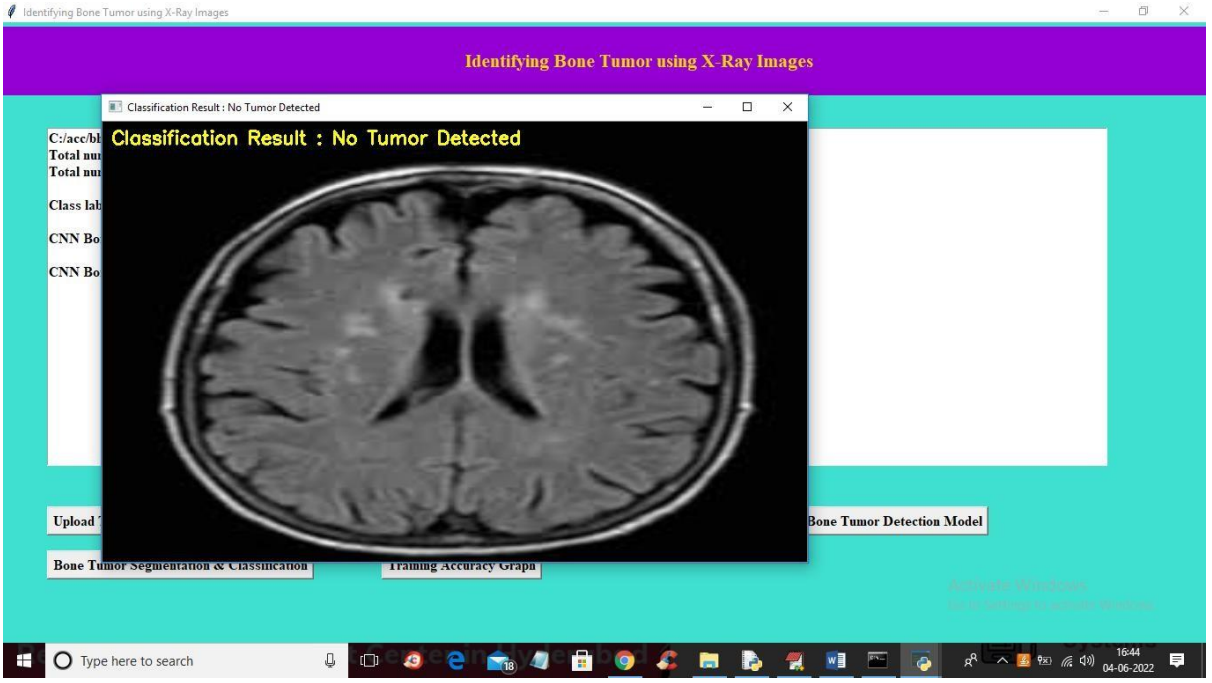

In above screen we can see dataset contains 253 images with and without tumor class label and now click on 'Trained CNN Bone Tumor Detection Model' button to train CNN with above extracted features and get below output



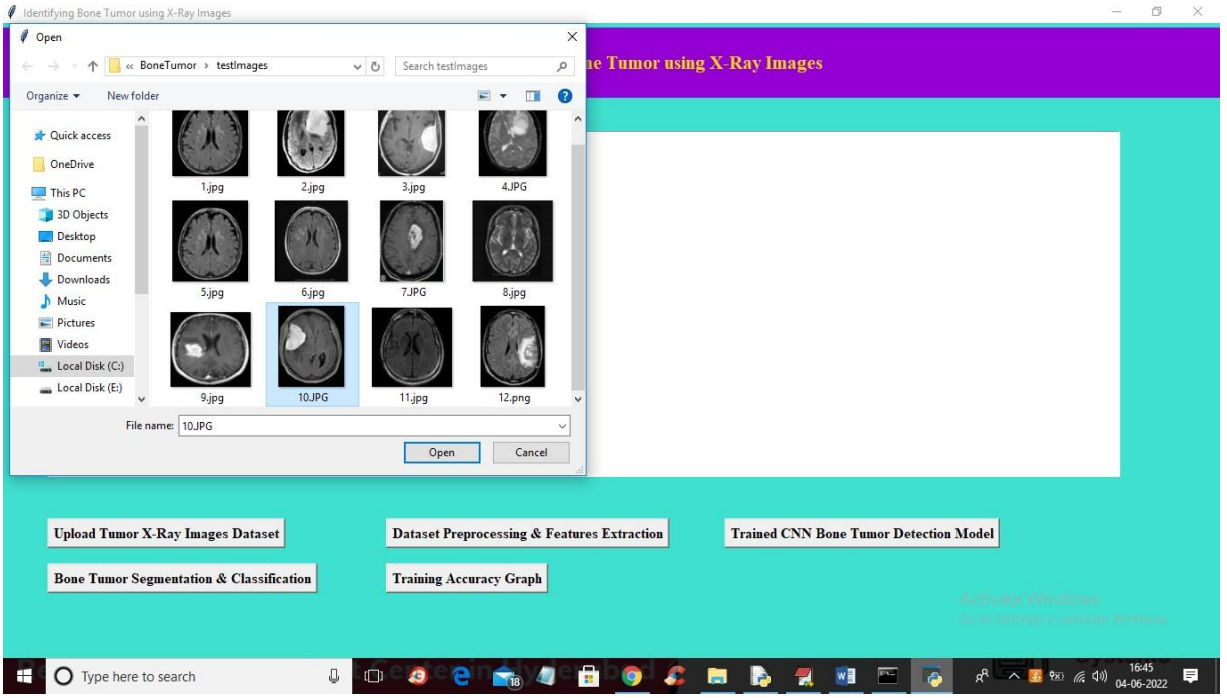

In above screen CNN training completed and we got it accuracy as 96% and now click on 'Bone Tumor Segmentation & Classification' button to upload test image and get below output



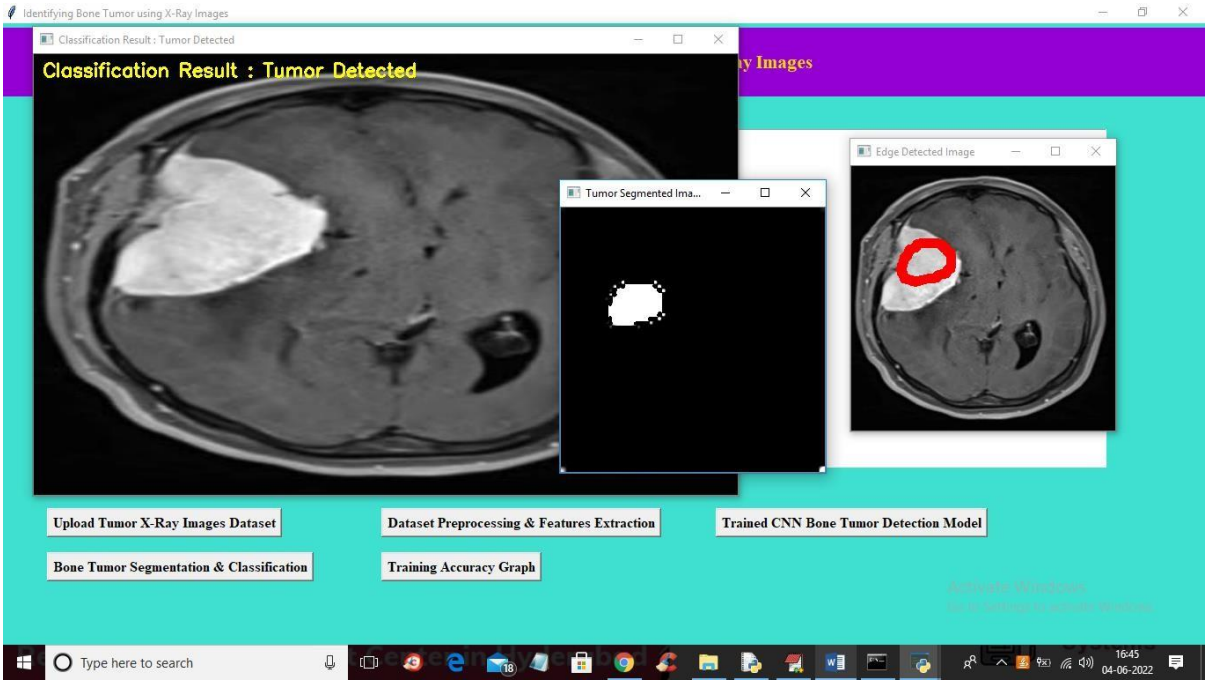

In above screen selecting and uploading 5.jpg file and then click on 'Open' button to get below output



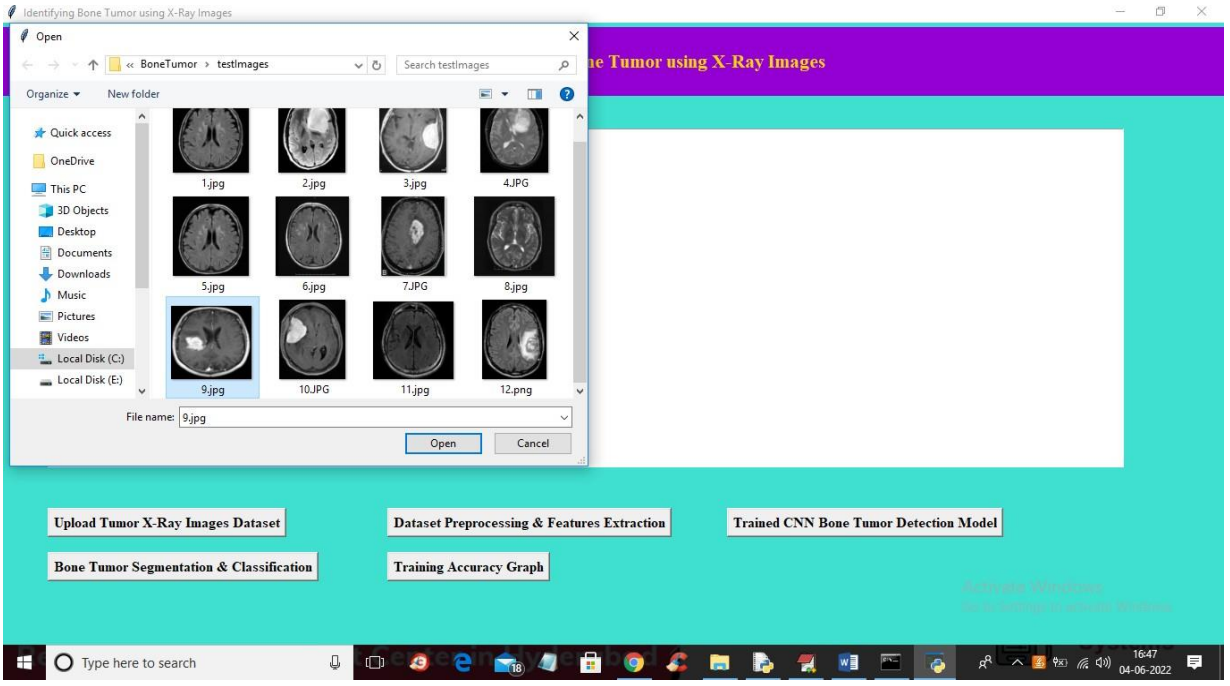

In above image ‘No Tumor Detected’ and now try another image.



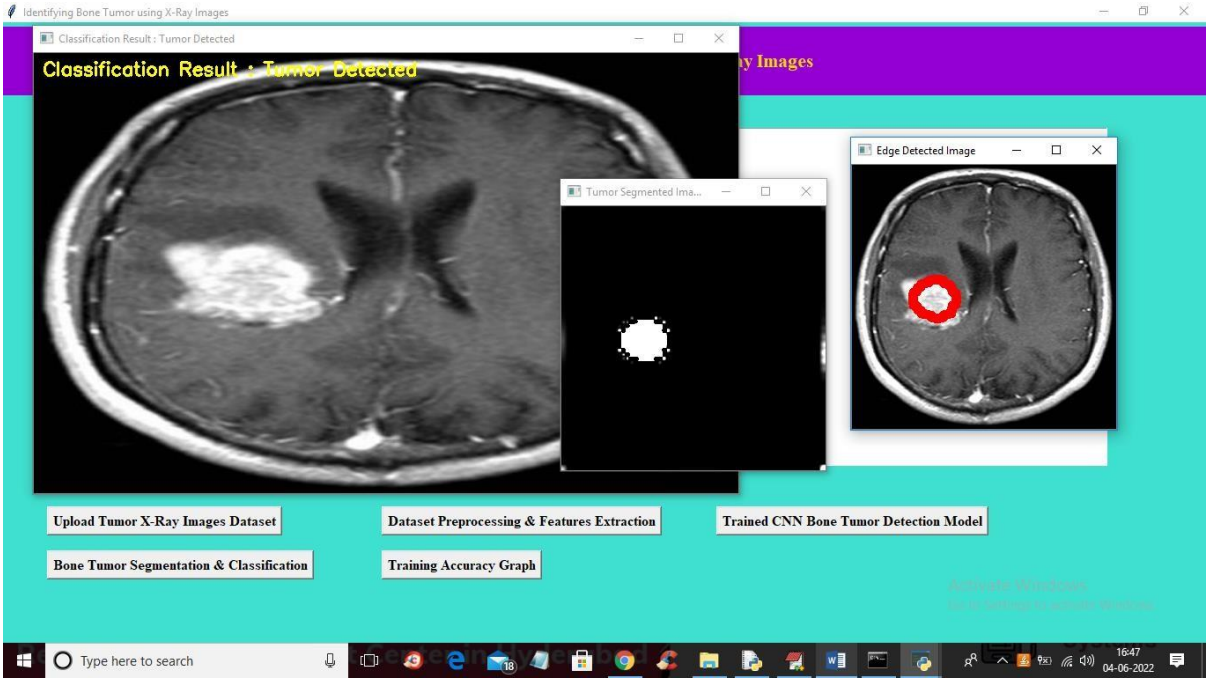

In above screen selecting and uploading’10.jpg’ and then click on ‘Open’ button to get below output



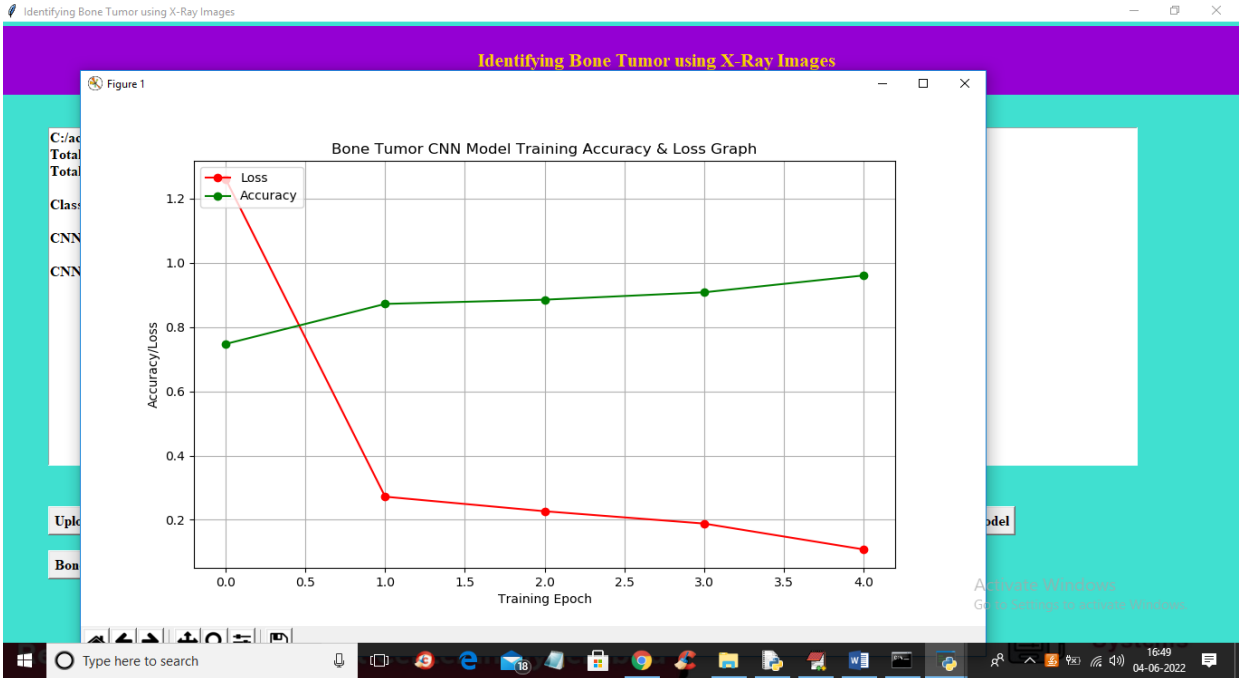

In above screen first image is the original image which classified as tumor detected and second image is tumor segmented image and 3rd image is the tumor edge detected image and see another image is below screen.




In above screen uploading 9.jpg image and click open button to get below output




In above screen we can see tumor detected with segmented out tumor image and with tumor edge detected. Similarly, you can upload other images and test and now click on 'Training Accuracy Graph' button to get below graph




In above graph x-axis represents training EPOCH and y-axis represents training accuracy and loss values and green line representing accuracy and red line represents LOSS and in above graph we can see with each increasing epoch accuracy got increase and loss got decrease.

# CHAPTER-10
# CONCLUSION

# CONCLUSION

In this work, we have proposed a technique for automated long-bone cancer diagnosis for thefirst time that is solely based on the analysis of an input *X*-ray image. The proposed method integrates several interdisciplinary concepts such as statistical runs-test, local entropy- and standard deviation- based tools, digital-geometric analysis, SVM classification, and decision tree. The notion of ortho- convex cover of a cluster of marked pixels is used for convenient visualization and diagnosis of the disease and for grading the severity of cancer-affected regions.

The use of digital-geometric tools leads to a fast estimation of the area of ROI as the requiredcomputation needs only integer-domain operations. Experimental results on a medical database of healthy and cancer-affected *X*-ray images reveal that the proposed method is fairly accurate as AUC for bone cancer detection is more than 0.85. Further, in 85% of cases, the bone-destruction pattern, stage, and grade of cancer predicted by the automated tool correctly match with the actual findings adjudged by the doctors and medical professionals.

# CHAPTER-11
# REFERENCES

# REFERENCES

1. American Joint Committee on Cancer (AJCC) AJCC cancer staging manual vol 17(6). Springer-Verlag (2010)

2. Alelyani S, Tang J, Liu H: Feature selection for clustering: a review. In: Data clustering: Algorithms and applications, vol 29, pp 110–121. CRC Press, 2013

3. Bandyopadhyay O, Biswas A, Bhattacharya BB: Long-bone fracture detection in digital X-ray images based on digital-geometric techniques. Comput Methods Programs Biomed 123:2–14, 2016 [PubMed]

4. Bandyopadhyay O, Chanda B, Bhattacharya BB: Automatic segmentation of bones in X-ray images based on entropy measure. Int J Image Graph 16(1):1650,001–32, 2016

5. Bennett JR, Donald JSM: On the measurement of curvature in a quantized environment. IEEE Trans Comput 24(8):803–820, 1975

6. Bourouis S, Chennoufi I, Hamrouni K: Multimodal bone cancer detection using fuzzy classification and variational model. In: Proceedings, IAPR, vol LNCS 8258, pp 174–181, 2013

7. Brant WE, Helms CA. Fundamentals of diagnostic radiology. Philadelphia: Wolters Kluwer; 2007. [Google Scholar]

8. Burges CJC: A tutorial on support vector machines for pattern recognition. Data Min Knowl Disc 2:121–167, 1998

9. Criminisiand A, Shotton J. Decision forests for computer vision and medical image analysis. London: Springer; 2013. [Google Scholar]

10. Dillencourt MB, Samet H: A general approach to connected component labeling for arbitrary image representations. J ACM 39:253–280, 1992

11. Ehara S: MR Imaging in staging of bone tumors. J Cancer Imaging 6:158–162, 2006 [PMC free article] [PubMed]

12. Frangi A, Egmont-Petersen M, Niessen W, Reiber J, Viergever M: Segmentation of bone tumor in MR perfusion images using neural networks and multiscale pharmacokinetic features. Image Vis Comput 19:679–690, 2001

13. Freeman H: On the encoding of arbitrary geometric configurations. IRE Trans Electron Comput 10:260–268, 1961

14. Hastie T, Tibshirani R, Friedman J. The elements of statistical learning. London: Springer; 2009. [Google Scholar]

15. Heidl W, Thumfart S, Lughofer E, Eitzinger C, Klement EP: Machine learning based analysis of gender differences in visual inspection decision making. Inf Sci 224:62–76, 2013

16. Heymann D. Bone cancer. Cambridge: Academic Press; 2015. [Google Scholar]

17. Hsu C, Chang C, Lin C (2003) Practical guide to support vector classification. http://www.csie.ntu.edu.tw/cjlin/papers.html

18. Huang S, Chiang K: Automatic detection of bone metastasis in vertebrae by using CT images. In: Proceedings, world congress on engineering, vol 2, pp 1166–1171, 2012

19. Leven RI, Rubin DS, Rastogi S, Siddiqui MS. Statistics for management. India: Pearson; 2012. [Google Scholar]

20. Maulik U, Chakraborty D: Fuzzy preference-based feature selection and semisupervised SVM for cancer classification. IEEE Trans NanoBiosciences 13(2):152–160, 2014 [PubMed]

21. Nandy SC, Mukhopadhyay K, Bhattacharya BB: Recognition of largest empty orthoconvex polygon in a point set. Inf Process Lett 110(17):746–752, 2010

22. Nisthula P, Yadhu RB: A novel method to detect bone cancer using image fusion and edge detection. Int J Eng Comput Sci 2(6):2012–2018, 2013

23. Ping YY, Yin CW, Kok LP: Computer aided bone tumor detection and classification using X-ray images. In: Proceedings, international federation for medical and biological engineering, pp 544–557, 2008