

RESPIRATORY ANALYSIS DETECTION OF VARIOUS LUNG INFECTIONS USING COUGH SIGNAL

*A Mini Project Report submitted to JNTU Hyderabad in partial
fulfillment of the requirements for the award of the degree*

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING(NETWORKS)

Submitted by

Rakshitha Ramidi	20S11A7016
Palthya Sachin	20S11A7018
Etukuri Akshay	20S11A7003
Peraveni Vamshi	21S15A7005

Under the Guidance of

Mrs B.RASHMI

B. Tech, M. Tech.

Assistant Professor



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING(NW)

MALLA REDDY INSTITUTE OF TECHNOLOGY & SCIENCE

(Approved by AICTE New Delhi and Affiliated to JNTUH)

(Accredited by NBA & NAAC with “A” Grade)

An ISO 9001: 2015 Certified Institution

Maisammaguda, Medchal (M), Hyderabad-500100, T. S.

NOVEMBER 2023

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING(NW)

MALLA REDDY INSTITUTE OF TECHNOLOGY & SCIENCE

(Approved by AICTE New Delhi and Affiliated to JNTUH)

(Accredited by NBA & NAAC with “A” Grade)

An ISO 9001: 2015 Certified Institution

Maisammaguda, Medchal (M), Hyderabad-500100, T. S.

NOVEMBER 2023



CERTIFICATE

This is to certify that the mini project entitled “**Respiratory Analysis Detection of Various Lung Infections Using Cough Signal**” has been submitted by **Rakshitha ramidi(20S11A7016)**, **Palthya Sachin(20S11A7018)**, **Etukuri Akshay(20S11A7003)** and **Peraveni Vamshi(21S15A7005)** in partial fulfillment of the requirements for the award of **BACHELOR OF TECHNOLOGY** in **COMPUTER SCIENCE & ENGINEERING(NW)**. This record of bonafide work carried out by them under my guidance and supervision. **The result embodied in this mini project report has not been submitted to any other University or Institute for the award of any degree.**

Mrs B.Rashmi

Assistant Professor

Project Guide

Dr. Y.Madhusekhar

Associate professor

HEAD OF THE DEPARTMENT

External Examiner

ACKNOWLEDGEMENT

The Mini Project work carried out by our team in the Department of Computer Science and Engineering, Malla Reddy Institute of Technology and Science, Hyderabad. ***This work is original and has not been submitted in part or full for any degree or diploma of any other university.***

We wish to acknowledge our sincere thanks to our project **Mrs B.Rashmi**, Assistant Professor of Computer Science & Engineering for formulation of the problem, analysis, guidance and her continuous supervision during the course of work.

We acknowledge our sincere thanks to **Dr. Vaka Murali Mohan**, Principal and **Dr. Y.Madhusekhar**, Head of the Department and Coordinator, faculty members of CSE(cs) Department for their kind cooperation in making this Mini Project work a success.

We extend our gratitude to **Sri. Ch. Malla Reddy**, Founder Chairman MRGI and **Sri. Ch. Mahender Reddy**, Secretary MRGI, **Dr.Ch. Bhadra Reddy**, President MRGI, **Sri. Ch. Shalini Reddy**, Director MRGI, **Sri. P. Praveen Reddy**, Director MRGI, for their kind cooperation in providing the infrastructure for completion of our Mini Project.

We acknowledge our special thanks to the entire teaching faculty and non-teaching staff members of the Computer Science & Engineering Department for their support in making this project work a success.

Rakshitha Ramidi	20S11A7016
Palthya Sachin	20S11A7018
Etukuri Akshay	20S11S7003
Peraveni Vamshi	21S15A7005

INDEX

Chapter	Page No.
ABSTRACT	v
LIST OF FIGURES	vi
1. SYSTEM ANALYSIS	1
1.1 Existing System	2
1.1.1 Disadvantages of Existing System	2
1.2 Proposed System	2
1.2.1 Advantages of Proposed System	3
1.3 Introduction	3
2. LITERATURE SURVEY	4
3. SYSTEM DESIGN	7
3.1 System Architecture	8
3.1.2 Block Diagram	8
3.2 System Requirements	9
3.2.1 Hardware Requirements	9
3.2.2 Software Requirements	9
3.3 Modules	12
4. INPUT & OUTPUT DESIGN	16
4.1 Input Design	17
4.2 Output Design	18
5. SOFTWARE ENVIRONMENT	19
5.1 Python Technology	20
5.1.1 History of python	20
5.1.2 What can python do	21
5.1.3 Why python	21
5.1.4 Python syntax compared to other programming languages	23
5.1.5 Uses of Python	25
5.1.6 Python Features	26
5.2 Python installation	26
5.3 Installation of python	27

5.4 IDLE versions	32
5.5 Modules	33
6. SYSTEM STUDY	36
6.1 Economic Feasibility	37
6.2 Technical Feasibility	37
6.3 Social Feasibility	38
7. SYSTEM TESTING	39
7.1 Types of Tests	40
7.1.1 Unit testing	40
7.2 Integration testing	41
7.2.1 Functional testing	41
7.2.2 System testing	42
7.3 Acceptance testing	42
8. RESULTS	43
9. CONCLUSION & FUTURE ENHANCEMENT	54
10. BIBLIOGRAPHY	56
11. YUKTHI INNOVATION CERTIFICATE	59

ABSTRACT

Large number of people die every year of Pulmonary chronic lung diseases irrespective of their age. Lung sound analysis has been a key diagnostic aid to accurately detect pulmonary diseases. Earlier, manual detection was used which was not a dependable method to detect lung diseases due to various reasons like low audibility and difference in perceptions of different physicians for different sounds. Modern computerized analysis yield results with much higher accuracy and thus a better treatment can be given to patients suffering from various kinds of lung diseases.

These disorders include Asthma, Bronchitis, Emphysema, Tuberculosis and Pneumonia. Some of the symptoms are wheezing, shortness of breath, rhonchi and chronic cough. In this project we are using respiratory audio dataset to predict various diseases such as Asthma, Pneumonia, Bronchiectasis and many more. To implement this project, we have taken disease diagnosis dataset and respiratory audio dataset and then extract features from all audio dataset and then trained a convolution neural network (CNN) algorithm model. After training model we can upload any new test data to predict disease from it.

LIST OF FIGURES

Fig.No	Fig. Name	Page No
3.1	System architecture	8
3.2	Data flow diagram	11
3.3.1	UML diagram	13
3.3.2	Class diagram	14
3.3.3	Sequence diagram	14
3.3.4	Activity diagram	15
5.3	Installation of python	27
5.3.1	Latest version for windows	28
5.3.2	Different versions of python	28
5.3.3	Versions of python along with OS	29
5.3.4	To run python	30
5.3.5	Add python 3.6 to path	30
5.3.6	Set Up successful	31
5.3.7	Various python installations	31
5.3.8	Command prompt	32
5.3.9	Python IDLE version	32
5.3.10	Python IDLE for 64-bit	33

LIST OF TABLES

Table.no	Table name	Page no
3.2.1	Hardware requirements	9
3.2.2	Software requirements	9

CHAPTER-1

SYSTEM ANALYSIS

1.SYSTEM ANALYSIS

1.1Existing System

The lungs are important organs in the respiratory system and used for gas exchange (oxygen and carbon dioxide). When we breathe. Our lungs transfer oxygen from the air into the blood, and carbon dioxide from the blood into the air. Cough is the most common symptom of several respiratory diseases. Cough is a defense mechanism of the body which prevents the respiratory tract from inhaling foreign materials accidentally or those produced internally by infection, it is characterized as wet when the sounds carry features indicative of mucus; in the absence of perceivable wetness, it is called dry.

There are few Existing Systems like Spirometry, Lung Volume Test, Gas Diffusion Test.

I.1.1 Disadvantages of Existing System

1.Spirometry:

- Limitations: Spirometry may not always provide a complete assessment of lung function and can't pinpoint specific diseases or conditions.
- Inaccuracy: Results can be affected by factors like patient effort, equipment calibration, and variations in testing conditions.
- Inability to Diagnose Certain Conditions: Some lung diseases or conditions might not be fully captured by spirometry alone.

2.Lung Volume Test:

- Difficulty in Performing: Lung volume tests often require more specialized equipment and can be more complex to conduct compared to spirometry.
- Discomfort: Some people may find the process of performing lung volume tests, such as body plethysmography, uncomfortable or claustrophobic due to the need to sit in an enclosed space.

3.Gas Diffusion Test:

- Sensitivity to Variables: The results might be influenced by factors like age, hemoglobin levels, altitude, and other health conditions, which could affect the accuracy of the test.
- Interpretation Challenges: Interpreting gas diffusion test results requires a comprehensive understanding of various factors that can affect diffusion, which may pose challenges in diagnosis and assessment.

I.2 Proposed System

In this project we are using respiratory audio dataset to predict various diseases such as Asthma, Pneumonia, Bronchiectasis and many more. To implement this project we have taken disease diagnosis dataset and respiratory audio dataset and then extract features from all audio dataset and then trained a convolution neural network (CNN) algorithm model. After training model we can upload any new test data to predict disease from it.

Advantages of Proposed System

- Easy to Use
- Gives Accurate information
- Reliable

1.3 Introduction

Pulmonary disorder is the inability of a person to breathe normally. Manual analysis used in the past only gave an approximate idea of the disorder and hence a very rough treatment was given. This was working out well in the past. Drastic increase in pollution and non-healthy habits of people has given rise to more complex diseases and need a very accurate estimation of the extent of disease. This accuracy can only be bought by automation of the analysis. Researchers observed that the difference between sounds made by infected lungs and the normal healthy lungs could serve as a very good tool for the detailed study and detection of the disease. Recording the Lung sounds, filtering them from the Heart sounds and other noises and studying the waveform of the filtered Lung sound has been the de facto way of performing the analysis. Many methods are given for the filtering and processing of the Lung sounds. Brief review of the previous papers reveals several methods of filtering and examining the LS.

The most challenging task in the analysis is the separation of HS from the LS due to spectral and temporal overlap between the two sounds. Filtering techniques used are Modulation Domain filtering that filters the temporal trajectories of short term spectral components. Signal analysis is carried out by segmentation into consecutive overlapping frames and performing Fourier transform. Combination of adaptive-frequency domain filtering where a very simple method is described that involves subtracting heart sounds from combination of heart and lung sound. Respiratory analysis involves studying the process of respiration, where organisms exchange gases with their environment. This intricate system, vital for survival, includes breathing, gas transport, and cellular respiration. Understanding these mechanisms is crucial for comprehending overall physiological function.

CHAPTER-2
LITERATURE SURVEY

2. LITERATURE SURVEY

[1] Rutuja Mhetre and U.R. Bagal “Respiratory sound analysis for diagnostic information”

The most important concern in the medical domain is to consider the interpretation of data and perform accurate diagnosis. The bronchitis, pneumonia and many other pulmonary diseases causes respiratory disorders which affects respiratory systems. Diagnosis of these diseases is facilitated by pulmonary auscultation by using stethoscope. This method depends on individuals hearing capability, experience and ability to differentiate the sounds. The quantitative measurement and permanent record of the related parameters is difficult. The recording and analysis of the respiratory sounds may quantify the changes in abnormal respiratory sounds in respiratory disorder. The signal processing techniques may be used for diagnostic information.

[2] Bor-Shing Lin, Huey-Dong Wu and Sao-Jie Chen “Automatic Wheezing Detection Based on Signal Processing of Spectrogram and Back - Propagation Neural Network”

Wheezing is a common clinical symptom in patients with obstructive pulmonary diseases such as asthma. Automatic wheezing detection offers an objective and accurate means for identifying wheezing lung sounds, helping physicians in the diagnosis, long-term auscultation, and analysis of a patient with obstructive pulmonary disease. This paper describes the design of a fast and high-performance wheeze recognition system. A wheezing detection algorithm based on the order truncate average method and a back-propagation neural network (BPNN) is proposed. Some features are extracted from processed spectra to train a BPNN, and subsequently, test samples are analyzed by the trained BPNN to determine whether they are wheezing sounds. The respiratory sounds of 58 volunteers (32 asthmatic and 26 healthy adults) were recorded for training and testing. Experimental results of a qualitative analysis of wheeze recognition showed a high sensitivity of 0.946 and a high specificity of 1.0. Further, cough sound characteristics are correlated to the airflow parameters of spirometry. Our results show strong correlation of cough sound characteristics with airflow characteristics including FEV1, FVC and their ratios, which are important in identifying the type of lung diseases as either obstructive (obstruction in airway) or restrictive (restricts lung expansion).

[3] Gowrisree Rudraraju,ShubhaDeepti Palreddy “Cough sound analysis and objective correlation with spirometry and clinical diagnosis”

In India, there are 100 million people who suffer from various respiratory problems; globally it is about 1–1.2 billion. The main problem attributed to the prevalence of respiratory diseases is lack of cost-effective and lab-free methods for early diagnosis. Spirometry is the standard clinical test procedure for detection of respiratory problems, but it requires repetition, and is also expensive and not available in rural areas. Cough sounds carry vital information about the respiratory system and the pathologies involved. Through this study, we detail how a combination of standard signal processing features and domain-specific features play a key role in distinguishing cough patterns. We could establish a relationship between cough pattern and respiratory conditions including widened airway, narrowed airway, fluid filled air sacs, and stiff lungs. Further, cough sound characteristics are correlated to the airflow parameters of spirometry. Our results show strong correlation of cough sound characteristics with airflow characteristics including FEV1, FVC and their ratios, which are important in identifying the type of lung diseases as either obstructive (obstruction in airway) or restrictive (restricts lung expansion). We have constructed a machine learning model to predict obstructive versus restrictive pattern, and validated it using K-fold cross-validation based on ground truth data. With a pattern prediction accuracy of 91.97%, sensitivity of 87.2%, and specificity of 93.69%, our results are encouraging.

[4] Bruno Rocha,L. Mendes “Detection of Cough and Adventitious Respiratory Sounds in Audio Recordings by Internal Sound Analysis”

We present a multi-feature approach to the detection of cough and adventitious respiratory sounds. After the removal of near-silent segments, a vector of event boundaries is obtained and a proposed set of 126 features is extracted for each event. Evaluation was performed on a data set comprised of internal audio recordings from 18 patients. The best performance (F-measure = 0.69 ± 0.03 ; specificity = 0.90 ± 0.01) was achieved when merging wheezes and crackles into a single class of adventitious respiratory sounds. A literature survey on respiratory analysis encompasses a broad range of topics. Studies often explore respiratory diseases, diagnostic techniques, and advancements in technology. Key areas include pulmonary function tests, imaging modalities, and biomarkers for conditions like asthma or chronic obstructive pulmonary disease (COPD). Researchers also investigate the impact of environmental factors on respiratory health. Reviewing this diverse literature provides valuable insights into current trends and future directions in respiratory analysis.

CHAPTER-3

SYSTEM DESIGN

3.SYSTEM DESIGN

3.1 System Architecture

- **Input layers:** It is the layer in which we give input to our model. The number of neurons in this layer is equal to the total number of features in our data.
- **Hidden layer:** The input from the input layer is then feed into the hidden layer. There can be many hidden layers depending upon our model and data size. Each hidden layer can have different number of neurons which are generally greater than the number of features. The output from each layer is computed by matrix multiplication of output of previous layer with learnable biases followed by activation function which makes the network non-linear.
- **Output layer:** The output layer from the hidden layer is then fed into a logistic function like sigmoid or soft max which converts the output of each class into the probability score of each class.

See below image with layers

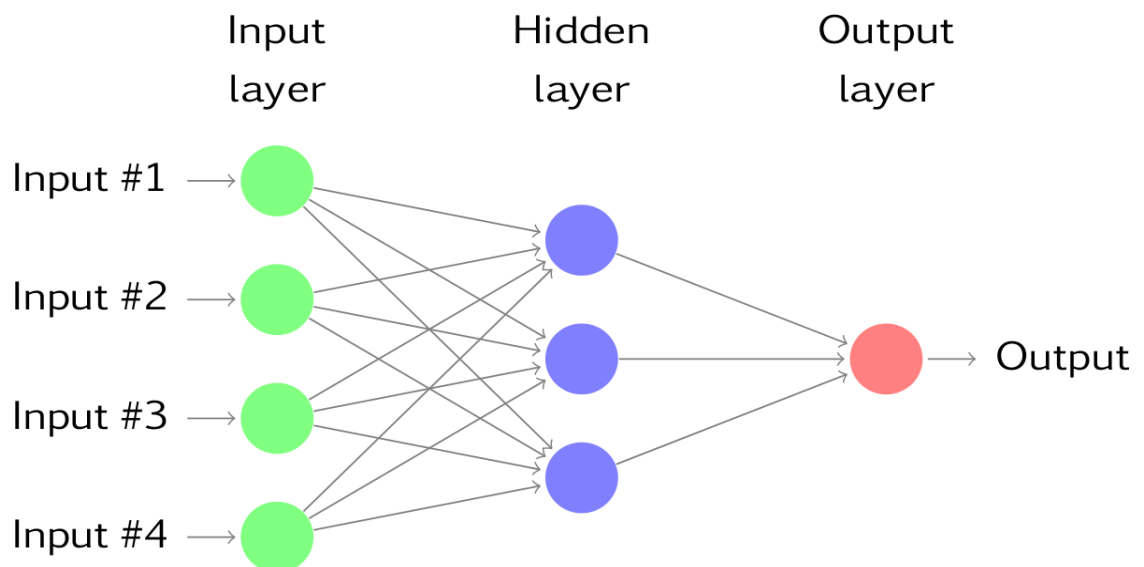


Figure:3.1: SYSTEM ARCHITECTURE

3.2 System Requirements

3.2.1 Hardware Requirements

System	Pentium Dual Core
Hard Disk	120 GB
Monitor	15'' LED
Input devices	Keyboard, Mouse
Ram	1 GB

Table 3.2.1:HARDWARE REQUIREMENTS

3.2.2 Software Requirements

Operating system	Windows 10
Coding Language	Python3.6.8
Tool	PyCharm
Data base	MYSQL
Server	Flask

Table 3.2.2: SOFTWARE REQUIREMENTS

3.3 Data Flow Diagram:

- The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
- The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
- DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
- DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

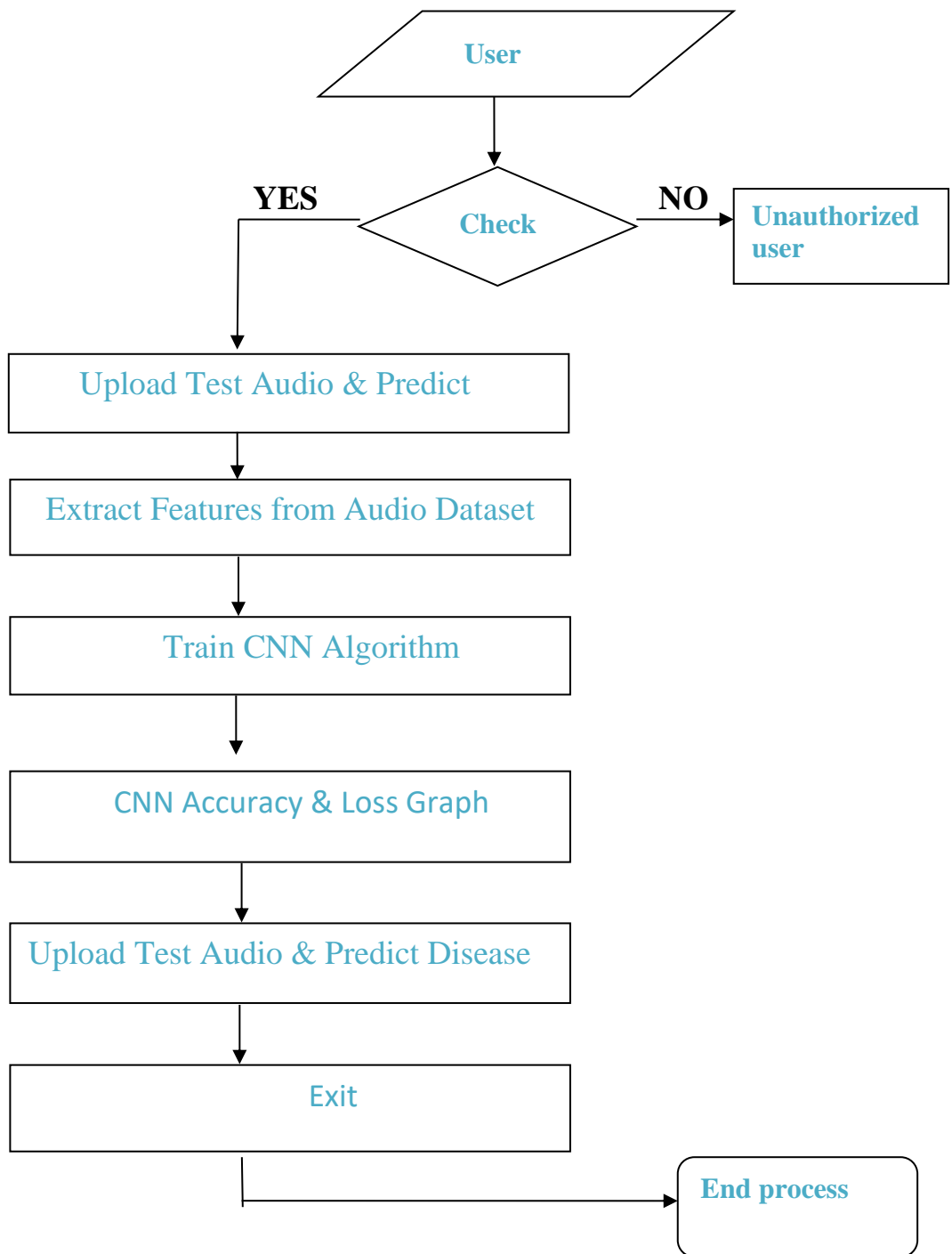


Figure:3.2. DATA FLOW DIAGRAM

3.4 UML Diagrams:

- UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.
- The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.
- The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.
- The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.
- The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

Goals:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Integrate best practices.

Use Case Diagram:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

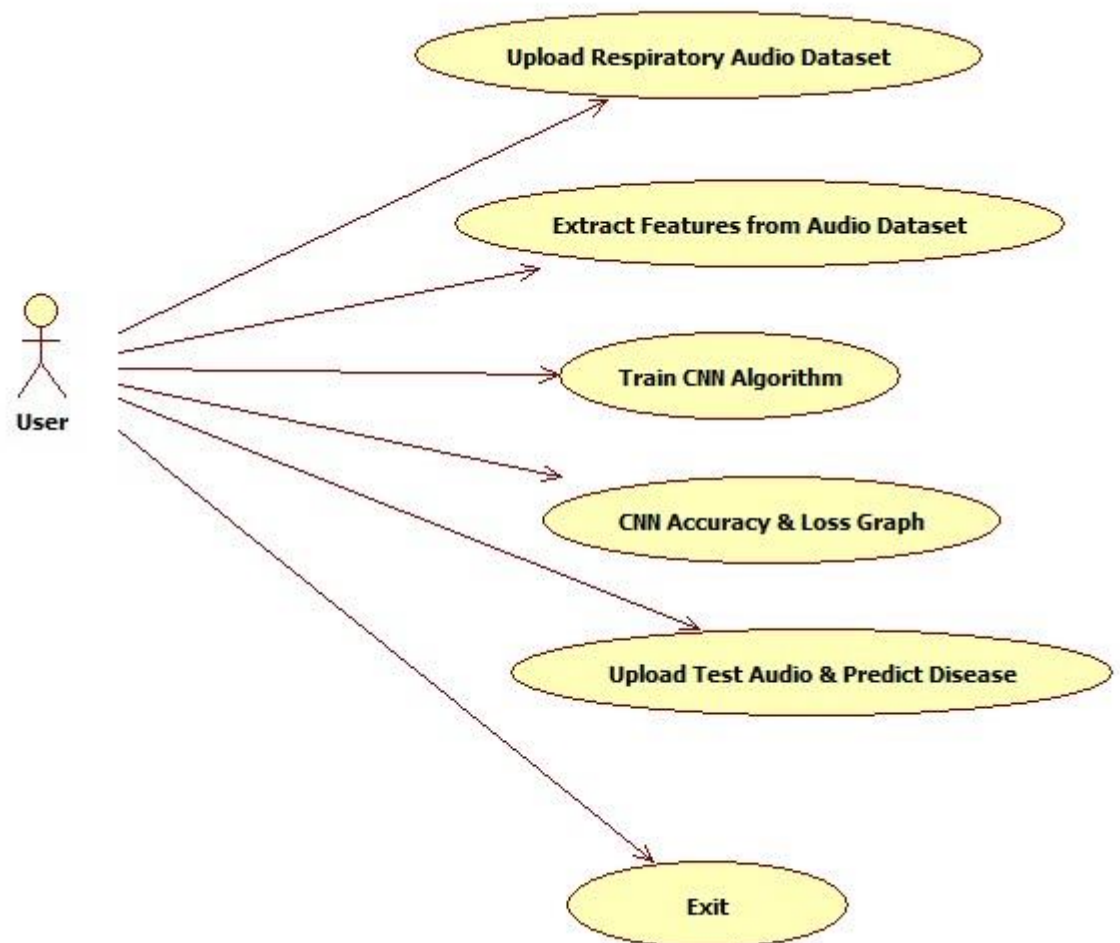


Figure: 3.3 1.UML DIAGRAMS

Class Diagram:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

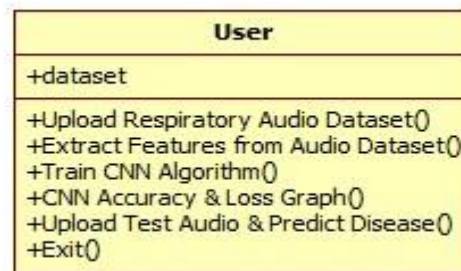


Figure:3.3.2. CLASS DIAGRAM

Sequence Diagram:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

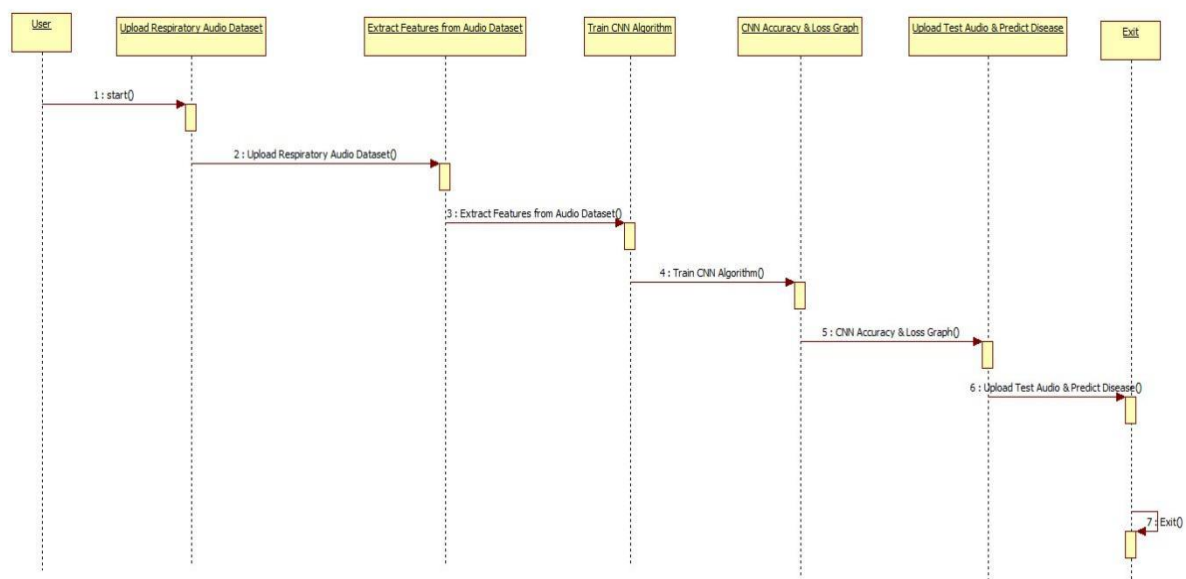


Figure:3.3.3. SEQUENCE DIAGRAM

Activity Diagram:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

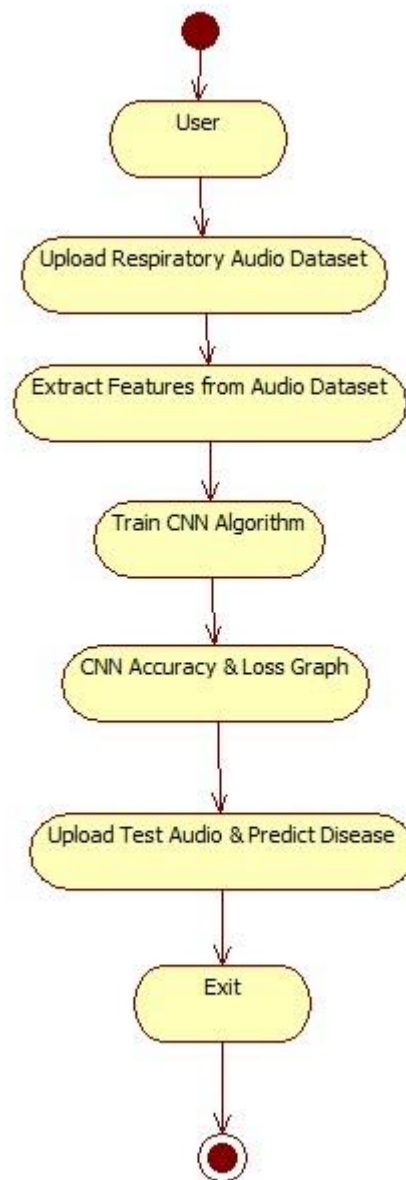


Figure:3.3.4. ACTIVITY DIAGRAM

CHAPTER-4
INPUT AND OUTPUT DESIGN

4.INPUT AND OUTPUT DESIGN

4.1Input Design

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

Objectives:

1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.
2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.
3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

4.2 Output Design

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2. Select methods for presenting information.

3. Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, current status or projections of the feature.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action and Confirm an action.

Objectives:

- **Accuracy and Sensitivity:**

Ensure the system accurately identifies cough signals associated with specific lung conditions, balancing sensitivity to detect subtle variations indicative of diseases.

- **Classification and Differentiation:**

Design outputs that categorize cough signals to distinguish between various lung diseases, aiding in precise diagnosis and treatment planning.

- **Real-time Monitoring:**

Enable real-time analysis and immediate feedback to healthcare professionals, facilitating prompt intervention and patient management.

- **User-Friendly Interface:**

Create an interface that is intuitive for healthcare practitioners, allowing easy interpretation of results and seamless integration into clinical workflows.

- **Interpretability:**

Provide clear and interpretable results, offering insights into the severity and progression of lung diseases based on cough signal patterns.

CHAPTER-5
SYSTEM ENVIRONMENT

5.SYSTEM ENVIRONMENT

5.1 Python Technology:

Python is a high-level, interpreted scripting language developed in the late 1980s by Guido van Rossum at the National Research Institute for Mathematics and Computer Science in the Netherlands. The initial version was published at the alt. Sources news group in 1991, and version 1.0 was released in 1994.

Python 2.0 was released in 2000, and the 2.x versions were the prevalent releases until December 2008. At that time, the development team made the decision to release version 3.0, which contained a few relatively small but significant changes that were not backward compatible with the 2.x versions. Python 2 and 3 are very similar, and some features of Python 3 have been back ported to Python 2. But in general, they remain not quite compatible.

Both Python 2 and 3 have continued to be maintained and developed, with periodic release updates for both. As of this writing, the most recent versions available are 2.7.15 and 3.6.5. However, an official End of Life date of January 1, 2020 has been established for Python 2, after which time it will no longer be maintained. If you are a newcomer to Python, it is recommended that you focus on Python 3, as this tutorial will do.

Python is still maintained by a core development team at the Institute, and Guido is still in charge, having been given the title of BDFL (Benevolent Dictator For Life) by the Python community. The name Python, by the way, derives not from the snake, but from the British comedy troupe Monty Python's Flying Circus, of which Guido was, and presumably still is, a fan. It is common to find references to Monty Python sketches and movies scattered throughout the Python documentation.

WHY CHOOSE PYTHON

If you're going to write programs, there are literally dozens of commonly used languages to choose from. Why choose Python? Here are some of the features that make Python an appealing choice.

Python is Popular

Python has been growing in popularity over the last few years. The 2018 Stack Overflow Developer Survey ranked Python as the 7th most popular and the number one most wanted technology of the year. World-class software development countries around the globe use Python every single day.

According to research by Dice Python is also one of the hottest skills to have and the most popular programming language in the world based on the Popularity of Programming Language Index.

Due to the popularity and widespread use of Python as a programming language, Python developers are sought after and paid well. If you'd like to dig deeper into [Python salary statistics and job opportunities, you can do so here.](#)

Python is interpreted

Many languages are compiled, meaning the source code you create needs to be translated into machine code, the language of your computer's processor, before it can be run. Programs written in an interpreted language are passed straight to an interpreter that runs them directly.

This makes for a quicker development cycle because you just type in your code and run it, without the intermediate compilation step.

One potential downside to interpreted languages is execution speed. Programs that are compiled into the native language of the computer processor tend to run more quickly than interpreted programs. For some applications that are particularly computationally intensive, like graphics processing or intense number crunching, this can be limiting.

In practice, however, for most programs, the difference in execution speed is measured in milliseconds, or seconds at most, and not appreciably noticeable to a human user. The expediency of coding in an interpreted language is typically worth it for most applications.

Python is Free

The Python interpreter is developed under an OSI-approved open-source license, making it free to install, use, and distribute, even for commercial purposes.

A version of the interpreter is available for virtually any platform there is, including all flavors of Unix, Windows, macOS, smart phones and tablets, and probably anything else you ever heard of. A version even exists for the half dozen people remaining who use OS/2.

Python is Portable

Because Python code is interpreted and not compiled into native machine instructions, code written for one platform will work on any other platform that has the Python interpreter installed. (This is true of any interpreted language, not just Python.)

Python is Simple

As programming languages go, Python is relatively uncluttered, and the developers have deliberately kept it that way.

A rough estimate of the complexity of a language can be gleaned from the number of keywords or reserved words in the language. These are words that are reserved for special meaning by

the compiler or interpreter because they designate specific built-in functionality of the language.

Python 3 has 33 keywords, and Python 2 has 31. By contrast, C++ has 62, Java has 53, and Visual Basic has more than 120, though these latter examples probably vary somewhat by implementation or dialect.

Python code has a simple and clean structure that is easy to learn and easy to read. In fact, as you will see, the language definition enforces code structure that is easy to read.

But It's Not That Simple For all its
syntactical simplicity, Python supports most constructs that would be expected in a very high-level language, including complex dynamic data types, structured and functional programming, and object-oriented programming.

Additionally, a very extensive library of classes and functions is available that provides capability well beyond what is built into the language, such as database manipulation or GUI programming.

Python accomplishes what many programming languages don't: the language itself is simply designed, but it is very versatile in terms of what you can accomplish with it.

This section gave an overview of the Python programming language, including:

- A brief history of the development of Python
- Some reasons why you might select Python as your language of choice

Python is a great option, whether you are a beginning programmer looking to learn the basics, an experienced programmer designing a large application, or anywhere in between. The basics of Python are easily grasped, and yet its capabilities are vast. Proceed to the next section to learn how to acquire and install Python on your computer.

Python is an open source programming language that was made to be easy-to-read and powerful. A Dutch programmer named Guido van Rossum made Python in 1991. He named it after the television show Monty Python's Flying Circus. Many Python examples and tutorials include jokes from the show.

Python is an interpreted language. Interpreted languages do not need to be compiled to run. A program called an interpreter runs Python code on almost any kind of computer. This means that a programmer can change the code and quickly see the results. This also means Python is slower than a compiled language like C, because it is not running machine code directly.

Python is a good programming language for beginners. It is a high-level language, which means a programmer can focus on what to do instead of how to do it. Writing programs in Python takes less time than in some other languages.

Python drew inspiration from other programming languages like C, C++, Java, Perl, and Lisp.

Python has a very easy-to-read syntax. Some of Python's syntax comes from C, because that is the language that Python was written in. But Python uses whitespace to delimit code: spaces

or tabs are used to organize code into groups. This is different from C. In C, there is a semicolon at the end of each line and curly braces ({}) are used to group code. Using whitespace to delimit code makes Python a very easy-to-read language.

Python is Popular

Python has been growing in popularity over the last few years. The 2018 Stack Overflow Developer Survey ranked Python as the 7th most popular and the number one most wanted technology of the year. World-class software development countries around the globe use Python every single day.

According to research by Dice Python is also one of the hottest skills to have and the most popular programming language in the world based on the Popularity of Programming Language Index.

Due to the popularity and widespread use of Python as a programming language, Python developers are sought after and paid well. If you'd like to dig deeper into Python salary statistics and job opportunities, you can do so [here](#).

Python is interpreted

Many languages are compiled, meaning the source code you create needs to be translated into machine code, the language of your computer's processor, before it can be run. Programs written in an interpreted language are passed straight to an interpreter that runs them directly.

This makes for a quicker development cycle because you just type in your code and run it, without the intermediate compilation step.

One potential downside to interpreted languages is execution speed. Programs that are compiled into the native language of the computer processor tend to run more quickly than interpreted programs. For some applications that are particularly computationally intensive, like graphics processing or intense number crunching, this can be limiting.

In practice, however, for most programs, the difference in execution speed is measured in milliseconds, or seconds at most, and not appreciably noticeable to a human user. The expediency of coding in an interpreted language is typically worth it for most applications.

Python use [\[change / change source\]](#)

Python is used by hundreds of thousands of programmers and is used in many places. Sometimes only Python code is used for a program, but most of the time it is used to do simple jobs while another programming language is used to do more complicated tasks.

Its standard library is made up of many functions that come with Python when it is installed. On the Internet there are many other libraries available that make it possible for the Python language to do more things. These libraries make it a powerful language; it can do many different things.

Some things that Python is often used for are:

- Web development
- Scientific programming
- Desktop GUIs
- Network programming
- Game programming

5.1.1 Python Development Steps: -

Guido Van Rossum published the first version of Python code (version 0.9.0) at alt. Sources in February 1991. This release included already exception handling, functions, and the core data types of list, dict, str and others. It was also object oriented and had a module system. Python version 1.0 was released in January 1994. The major new features included in this release were the functional programming tools lambda, map, filter and reduce, which Guido Van Rossum never liked. Six and a half years later in October 2000, Python 2.0 was introduced. This release included list comprehensions, a full garbage collector and it was supporting unicode. Python flourished for another 8 years in the versions 2.x before the next major release as Python 3.0 (also known as "Python 3000" and "Py3K") was released. Python 3 is not backwards compatible with Python 2.x. The emphasis in Python 3 had been on the removal of duplicate programming constructs and modules, thus fulfilling or coming close to fulfilling the 13th law of the Zen of Python: "There should be one -- and preferably only one -- obvious way to do it." Some changes in Python 7.3:

- Print is now a function
- Views and iterators instead of lists
- The rules for ordering comparisons have been simplified. E.g. a heterogeneous list cannot be sorted, because all the elements of a list must be comparable to each other.
- There is only one integer type left, i.e. int. long is int as well.
- The division of two integers returns a float instead of an integer. "/" can be used to have the "old" behaviour.
- Text Vs. Data Instead Of Unicode Vs. 8-bit

Purpose: -

We demonstrated that our approach enables successful segmentation of intra-retinal layers—even with low-quality images containing speckle noise, low contrast, and different intensity ranges throughout—with the assistance of the ANIS feature.

5.1.2 Modules Used in Project: -

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 networks. 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 2.0 open-source license](#) on November 9, 2015.

Numpy

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows Numpy to seamlessly and speedily integrate with a wide variety of databases.

Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Matplotlib

Matplotlib is a Python 2D plotting library which 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](#) shells, the [Jupyter](#) Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object-oriented interface or via a set of functions familiar to MATLAB users.

Scikit – learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

- Python is Interpreted – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- Python is Interactive – you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

5.2 Python Installation:

Python a versatile programming language doesn't come pre-installed on your computer devices. Python was first released in the year 1991 and until today it is a very popular high-level programming language. Its style philosophy emphasizes code readability with its notable use of great whitespace.

The object-oriented approach and language construct provided by Python enables programmers to write both clear and logical code for projects. This software does not come pre-packaged with Windows.

5.2.1 How to Install Python on Windows and Mac:

There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version 3.7.4 or in other words, it is Python 3.

Note: The python version 3.7.4 cannot be used on Windows XP or earlier devices.

Before you start with the installation process of Python. First, you need to know about your **System Requirements**. Based on your system type i.e. operating system and based processor, you must download the python version. My system type is a **Windows 64-bit operating system**. So the steps below are to install python version 3.7.4 on Windows 7 device or to install Python 3. [Download the Python Cheatsheet here](#). The steps on how to install Python on Windows 10, 8 and 7 are **divided into 4 parts** to help understand better.

5.3 Download the Correct version into the system

Step 1: Go to the official site to download and install python using Google Chrome or any other web browser. OR Click on the following link: <https://www.python.org>



Figure: 5.3: Installation Of Python

Now, check for the latest and the correct version for your operating system.

Step2: Click on the Download tab.



Figure: 5.3.1: Latest Version For Windows

Step 3: You can either select the Download Python for windows 3.7.4 button in Yellow Color or you can scroll further down and click on download with respective to their version. Here, we are downloading the most recent python version for windows 3.7.4

Looking for a specific release?

Python releases by version number:

Release version	Release date		Click for more
Python 3.7.4	July 8, 2019	Download	Release Notes
Python 3.6.9	July 2, 2019	Download	Release Notes
Python 3.7.3	March 25, 2019	Download	Release Notes
Python 3.4.10	March 18, 2019	Download	Release Notes
Python 3.5.7	March 18, 2019	Download	Release Notes
Python 2.7.16	March 4, 2019	Download	Release Notes
Python 3.7.2	Dec. 24, 2018	Download	Release Notes

Figure: 5.3.2: Different Versions Of Python

Step 4: Scroll down the page until you find the Files option.

Step 5: Here you see a different version of python along with the operating system.

Files					
Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		68111671e5b2db4aef7b9ab01b7099be	23817663	SIG
XZ compressed source tarball	Source release		d33e4aa66097051c2eca45ee3604803	17131432	SIG
macOS 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.6 and later	6428b4fa7583da11a442c3a1ce08e6	34898416	SIG
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later	5dd605c38217a457738f5e4a936b241f	28082845	SIG
Windows help file	Windows		d63999573a29682ac56cade6b4f7cd2	8131761	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	9b003c3fcd8ec0b9a1e83184a40729a2	7504281	SIG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	a702b4b0ad70d8bdc3a43a583e563400	26880368	SIG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	28cb1c608b8d72a88e53a3bf351b4bd2	1362904	SIG
Windows x86 embeddable zip file	Windows		9fab1b818841879fa94133574139d8	6741626	SIG
Windows x86 executable installer	Windows		33cc802942a54446a3d8451476384789	25663848	SIG
Windows x86 web-based installer	Windows		1b670cfa5d317df82c30983ea371d87c	1324608	SIG

Figure: 5.3.3: Versions Of Python Along With Os

- To download Windows 32-bit python, you can select any one from the three options: Windows x86 embeddable zip file, Windows x86 executable installer or Windows x86 web-based installer.
- To download Windows 64-bit python, you can select any one from the three options: Windows x86-64 embeddable zip file, Windows x86-64 executable installer or Windows x86-64 web-based installer.

Here we will install Windows x86-64 web-based installer. Here your first part regarding which version of python is to be downloaded is completed. Now we move ahead with the second part in installing python i.e. Installation

Note: To know the changes or updates that are made in the version you can click on the Release Note Option.

Installation of Python

Step 1: Go to Download and Open the downloaded python version to carry out the installation process.

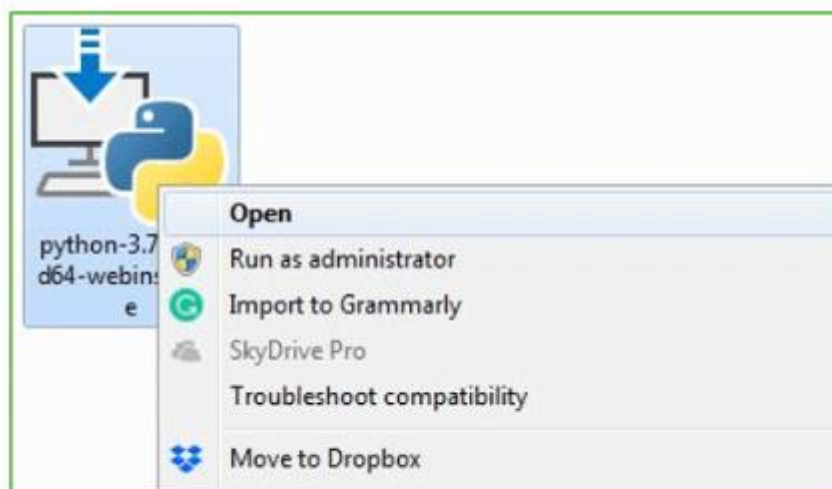


Figure: 5.3.4: To Run Python

Step 2: Before you click on Install Now, Make sure to put a tick on Add Python 3.7 to PATH.



Figure:

5.3.5: Add Python 3.6 To Path

Step 3: Click on Install NOW After the installation is successful. Click on Close.



Figure: 5.3.6: Set Up Successful

With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

Note: The installation process might take a couple of minutes.

Verify the Python Installation

Step 1: Click on Start

Step 2: In the Windows Run Command, type “cmd”.

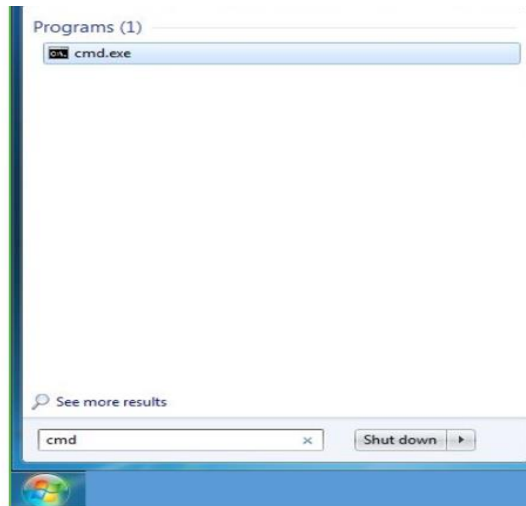


Figure: 5.3.7: Verifying Python Installation

Step 3: Open the Command prompt option.

Step 4: Let us test whether the python is correctly installed. Type **python -V** and press Enter.

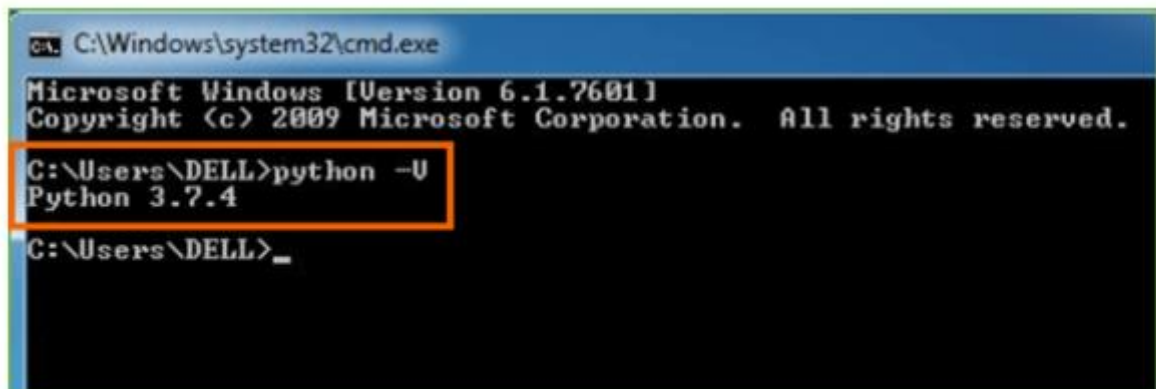


Figure: 5.3.8: Command Prompt

Step 5: You will get the answer as 3.7.4

Note: If you have any of the earlier versions of Python already installed. You must first uninstall the earlier version and then install the new one.

Check how the Python IDLE works

Step 1: Click on Start

Step 2: In the Windows Run command, type “python idle”.

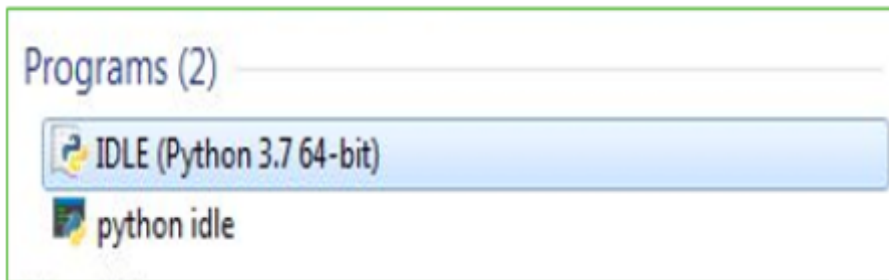


Figure: 5.3.9: Python IDLE Version

Step 3: Click on IDLE (Python 3.7 64-bit) and launch the program

Step 4: To go ahead with working in IDLE you must first save the file. **Click on File > Click on Save**

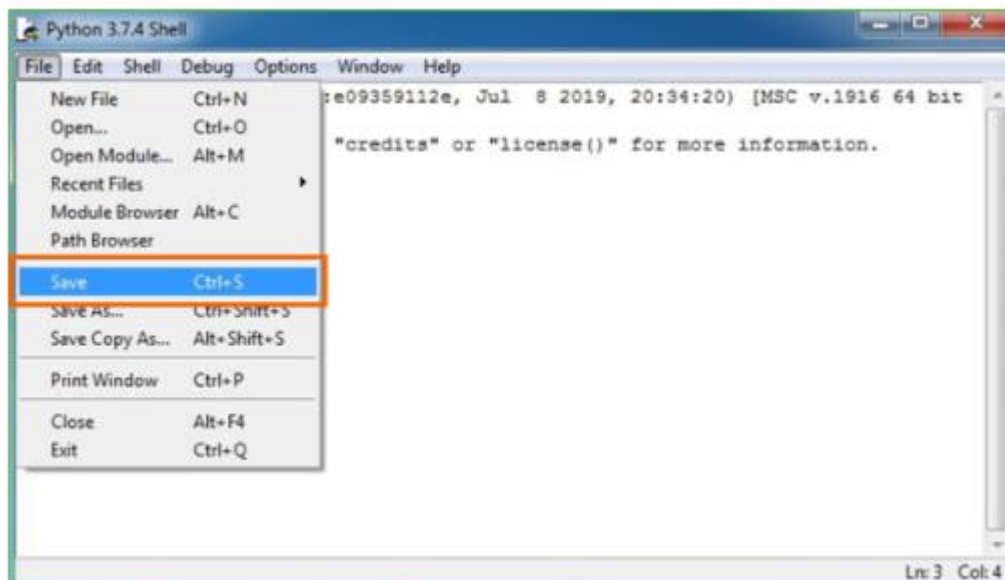


Figure: 5.3.10: Python IDLE For 64-Bit

Step 5: Name the file and save as type should be Python files. Click on SAVE. Here I have named the files as Hey World.

Step 6: Now for e.g., **enter print**

5.4 Modules:

- Gathering Data
- Data preparation
- Data Wrangling
- Analyze Data
- Train the model
- Test the model

Gathering data:

Data Gathering is the first step of the machine learning life cycle. The goal of this step is to identify and obtain all data-related problems.

In this step, we need to identify the different data sources, as data can be collected from various sources such as **files, database, internet**. It is one of the most important steps of the life cycle. The quantity and quality of the collected data will determine the efficiency of the output. The more will be the data, the more accurate will be the prediction.

Data preparation:

After collecting the data, we need to prepare it for further steps. Data preparation is a step where we put our data into a suitable place and prepare it to use in our machine learning training.

In this step, first, we put all data together, and then randomize the ordering of data.

This step can be further divided into two processes:

- **Data exploration:**

It is used to understand the nature of data that we have to work with. We need to understand the characteristics, format, and quality of data. A better understanding of data leads to an effective outcome. In this, we find Correlations, general trends, and outliers.

- **Data pre-processing:**

Now the next step is pre-processing of data for its analysis.

Data Wrangling:

Data wrangling is the process of cleaning and converting raw data into a useable format. It is the process of cleaning the data, selecting the variable to use, and transforming the data in a proper format to make it more suitable for analysis in the next step. It is one of the most important steps of the complete process. Cleaning of data is required to address the quality issues.

It is not necessary that data we have collected is always of our use as some of the data may not be useful. In real-world applications, collected data may have various issues, including:

- Missing Values
- Duplicate data
- Invalid data
- Noise

Data Analysis:

Now the cleaned and prepared data is passed on to the analysis step. This step involves:

- Selection of analytical techniques
- Building models
- Review the result

The aim of this step is to build a machine learning model to analyze the data using various analytical techniques and review the outcome. It starts with the determination of the type of the problems, where we select the machine learning techniques such as Classification, Regression, Cluster analysis, Association, etc. then build the model using prepared data, and evaluate the model.

Train Model:

Now the next step is to train the model, in this step we train our model to improve its performance for better outcome of the problem.

We use datasets to train the model using various machine learning algorithms. Training a model is required so that it can understand the various patterns, rules, and, features

Test Model:

Once our machine learning model has been trained on a given dataset, then we test the model. In this step, we check for the accuracy of our model by providing a test dataset to it.

Testing the model determines the percentage accuracy of the model as per the requirement of project or problem

CHAPTER-6
SYSTEM STUDY

6.SYSTEM STUDY

Feasibility Study:

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- Economic Feasibility
- Technical Feasibility
- Social Feasibility

4.1 Economical Feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased. Account for expenses related to ensuring ethical practices, patient privacy compliance, and adherence to regulatory standards. Invest in measures to address ethical considerations associated with data collection and usage. Assess potential cost savings in healthcare resulting from early detection, improved diagnostic efficiency, and better disease management. Consider factors such as reduced hospitalization, medication costs, and overall healthcare utilization.

4.2 Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system. Determine the feasibility of obtaining high-quality cough signal data using sensors, microphones, or other devices. Assess whether the collected data is representative of diverse populations and conditions. Evaluate the effectiveness of signal processing algorithms in accurately extracting relevant features from cough signals. Ensure that the chosen techniques can handle variations in cough patterns associated with different lung diseases.

4.3 Social Feasibility

Assessing the social feasibility of respiratory analysis for lung diseases using cough signals involves considering the acceptance, impact, and ethical implications within society. The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity.

The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system. Evaluate the level of awareness and understanding within the general public regarding respiratory analysis using cough signals. Assess how societal perceptions may influence acceptance and adoption.

CHAPTER-7

SYSTEM TESTING

7.SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPES OF TESTS

Unit testing:

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing:

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfactory, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test:

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

- Valid Input : identified classes of valid input must be accepted.
- Invalid Input : identified classes of invalid input must be rejected.
- Functions : identified functions must be exercised.
- Output : identified classes of application outputs must be exercised.
- Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test:

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing:

White Box Testing is a testing in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing:

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

7.1 Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

7.1.1 Test strategy and approach:

Field testing will be performed manually and functional tests will be written in detail.

Test objectives:

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

7.2 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

7.3 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

CHAPTER-8
RESULT

8.RESULT

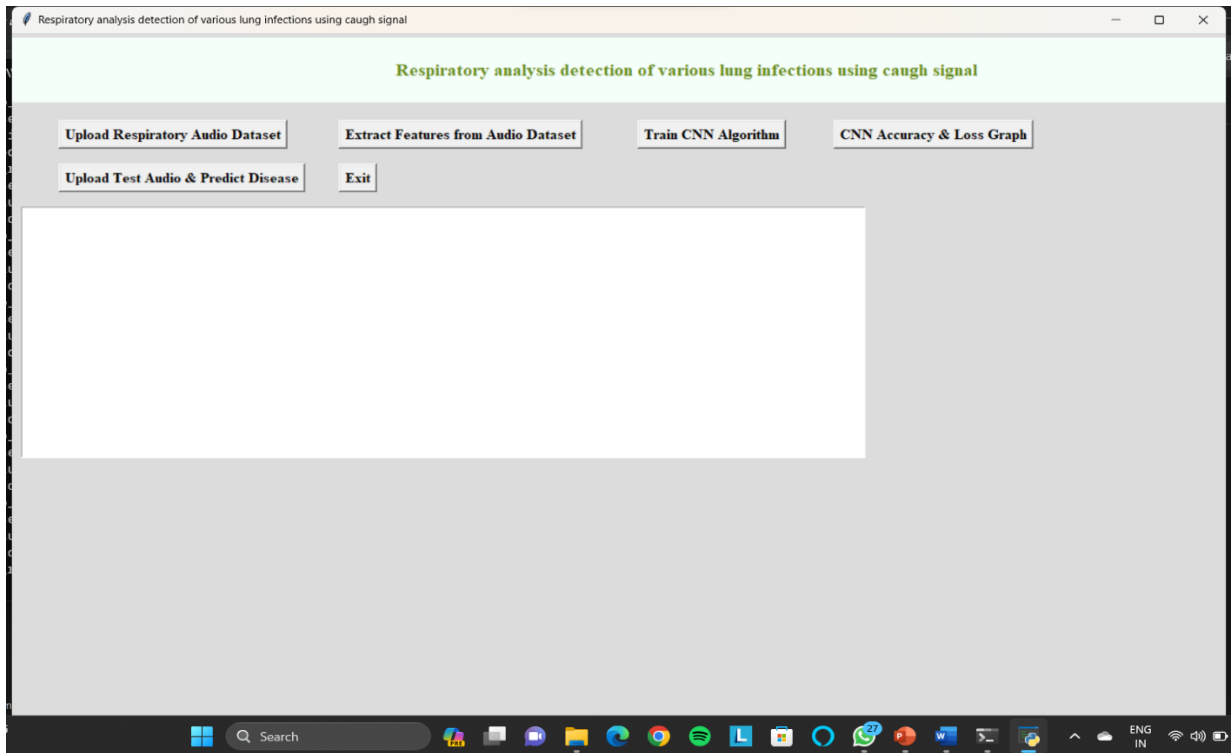
In this project we are using respiratory audio dataset to predict various diseases such as Asthma, Pneumonia, Bronchiectasis and many more. To implement this project we have taken disease diagnosis dataset and respiratory audio dataset and then extract features from all audio dataset and then trained a convolution neural network (CNN) algorithm model. After training model we can upload any new test data to predict disease from it.

To implement this project we have designed following modules

- 1) **Upload Respiratory Audio Dataset:** using this module we will upload disease diagnosis dataset and respiratory audio dataset
- 2) **Extract Features from Audio Dataset:** using this module we will extract features from both datasets and then build training dataset
- 3) **Train CNN Algorithm:** using above train dataset we will train CNN model and then build a trained model and this model can be used to predict disease from any new test audio files
- 4) **CNN Accuracy & Loss Graph:** using this module we will display comparison graph between accuracy and loss of CNN trained model
- 5) **Upload Test Audio & Predict Disease:** using this module we will upload test audio files and then apply CNN trained model on that test audio to predict disease

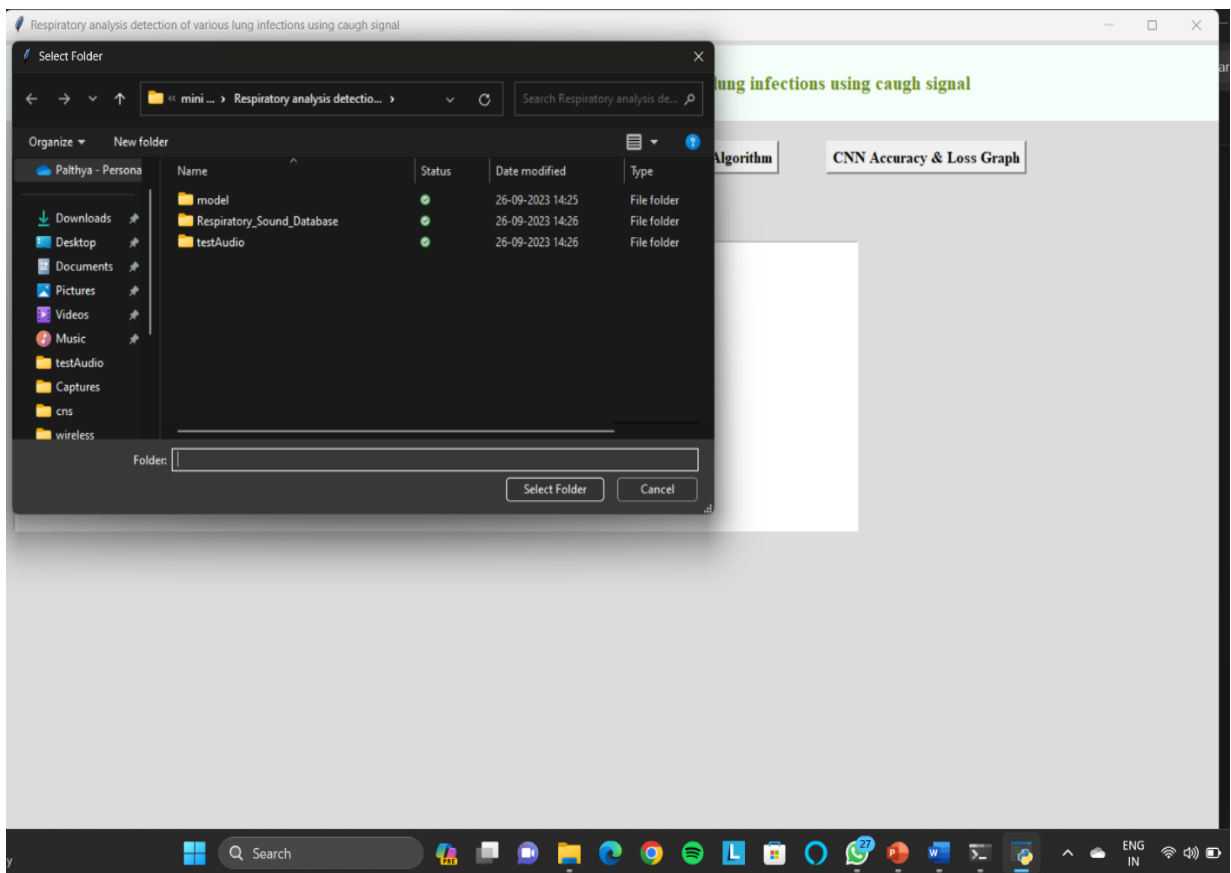
We have used below dataset to trained CNN model

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

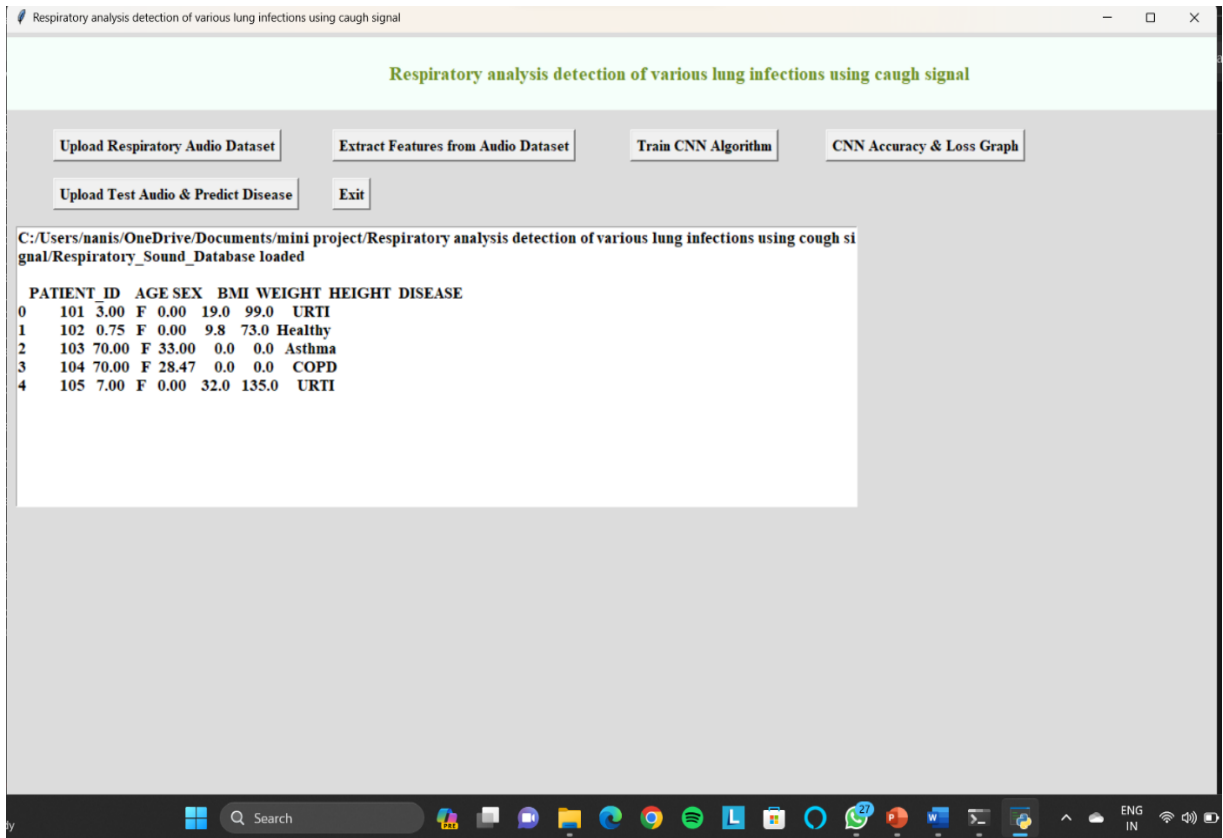


8.1 Screenshot

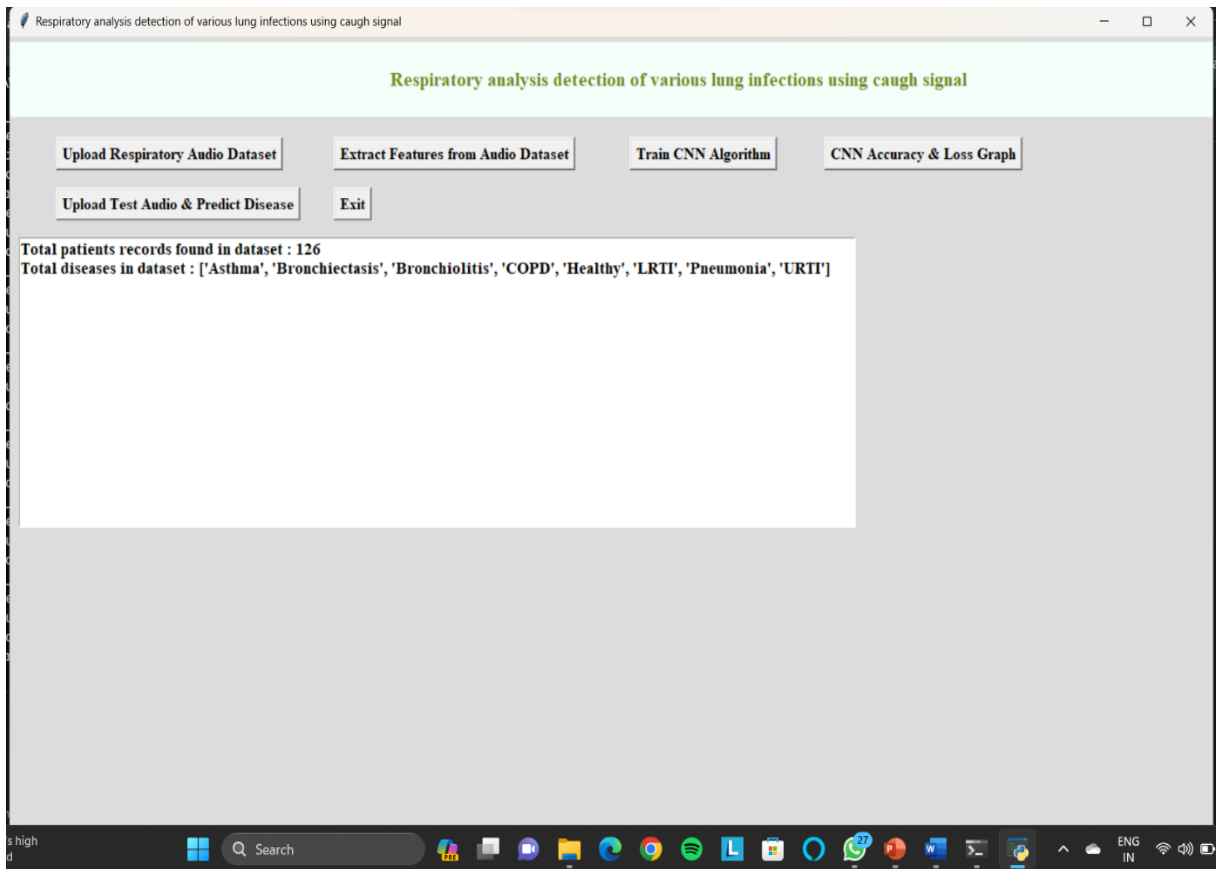
In below screen click on ‘Upload Respiratory Audio Dataset’ button to upload dataset



In above screen selecting and uploading entire respiratory sound folder and then click on ‘Select Folder’ button to load dataset and to get below screen



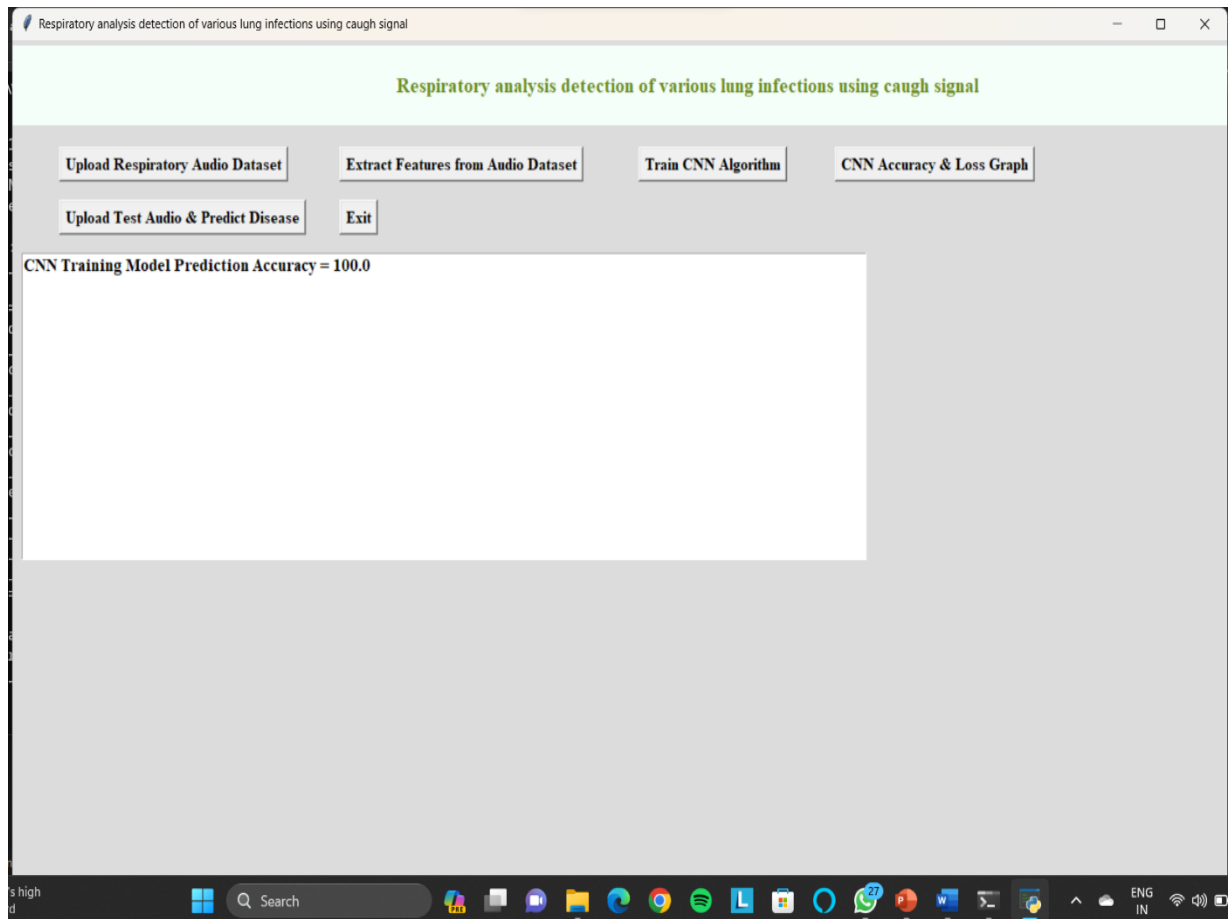
In above screen for each patient we can see associated with disease diagnose and above disease will be used as class label for each extracted audio features and now click on 'Extract Features from Audio Dataset' button to extract features from each audio files and then associate detected disease as class label to audio file.



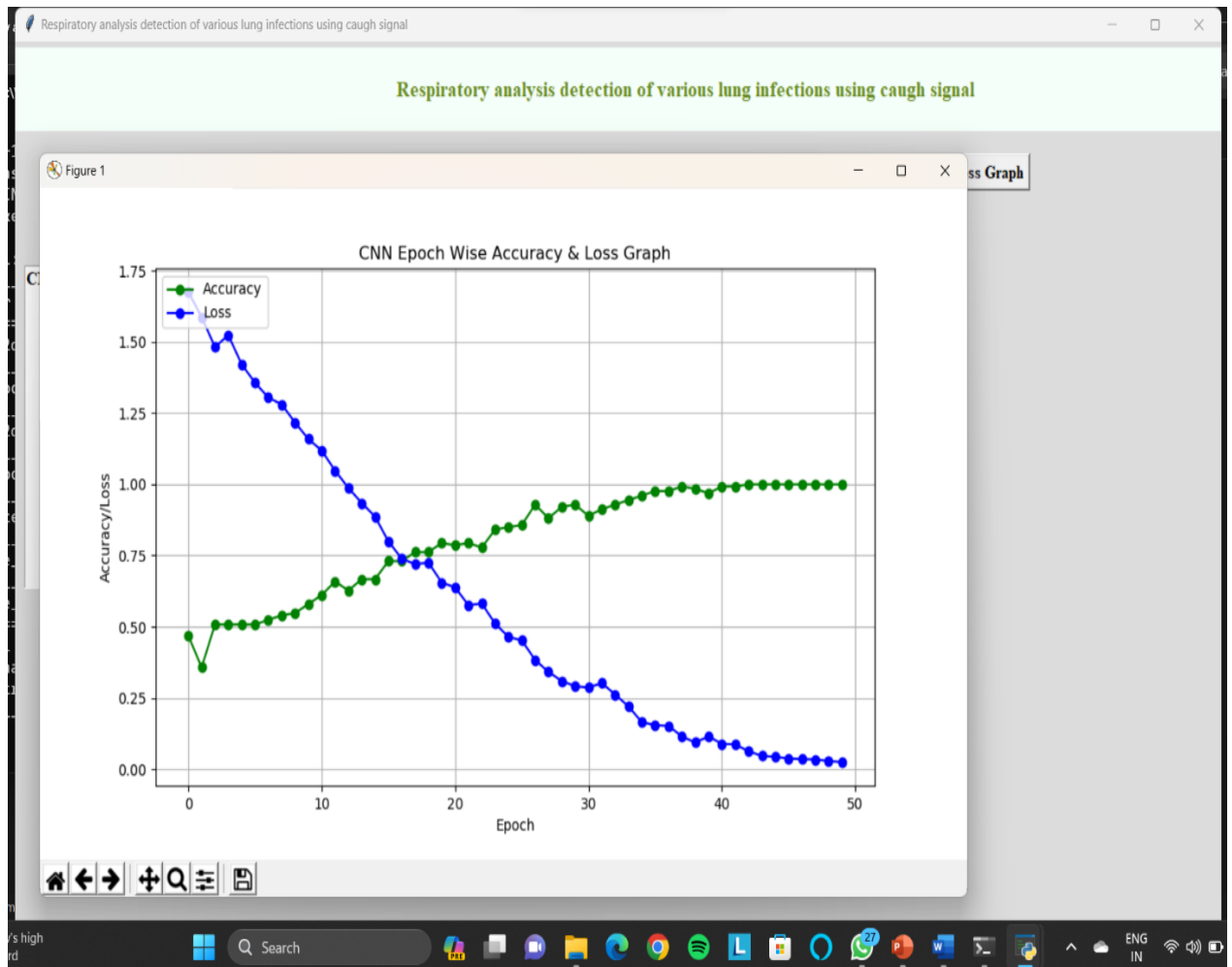
8.2 Screenshot

In above screen application found 126 patients audio files and this audio dataset contains 8 different diseases and now dataset is ready and now click on 'Train CNN Algorithm' button to train CNN with above dataset and then calculate CNN prediction accuracy

In below screen CNN trained on audio features and got 100% accuracy and now click on ‘CNN Accuracy & Loss Graph’ button to get below graph



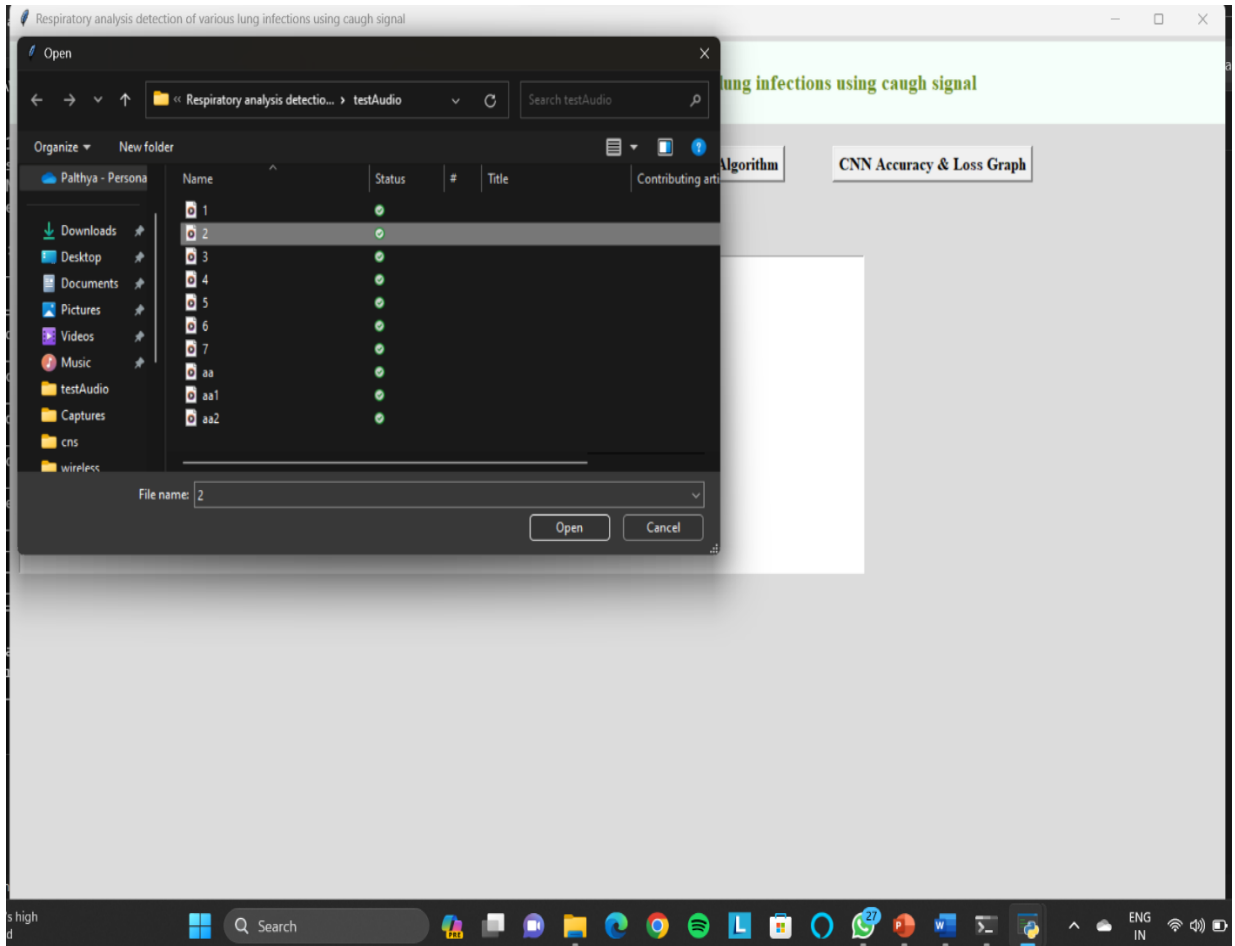
In below graph x-axis represents EPOCH/ITERATIONS and Y-axis represents accuracy and loss values and green line represents accuracy and blue line represents LOSS and we used 50 EPOCHS to train CNN model and we can see with each increasing epoch accuracy get increased and loss value got decrease to 0 and accuracy increased to 100%.



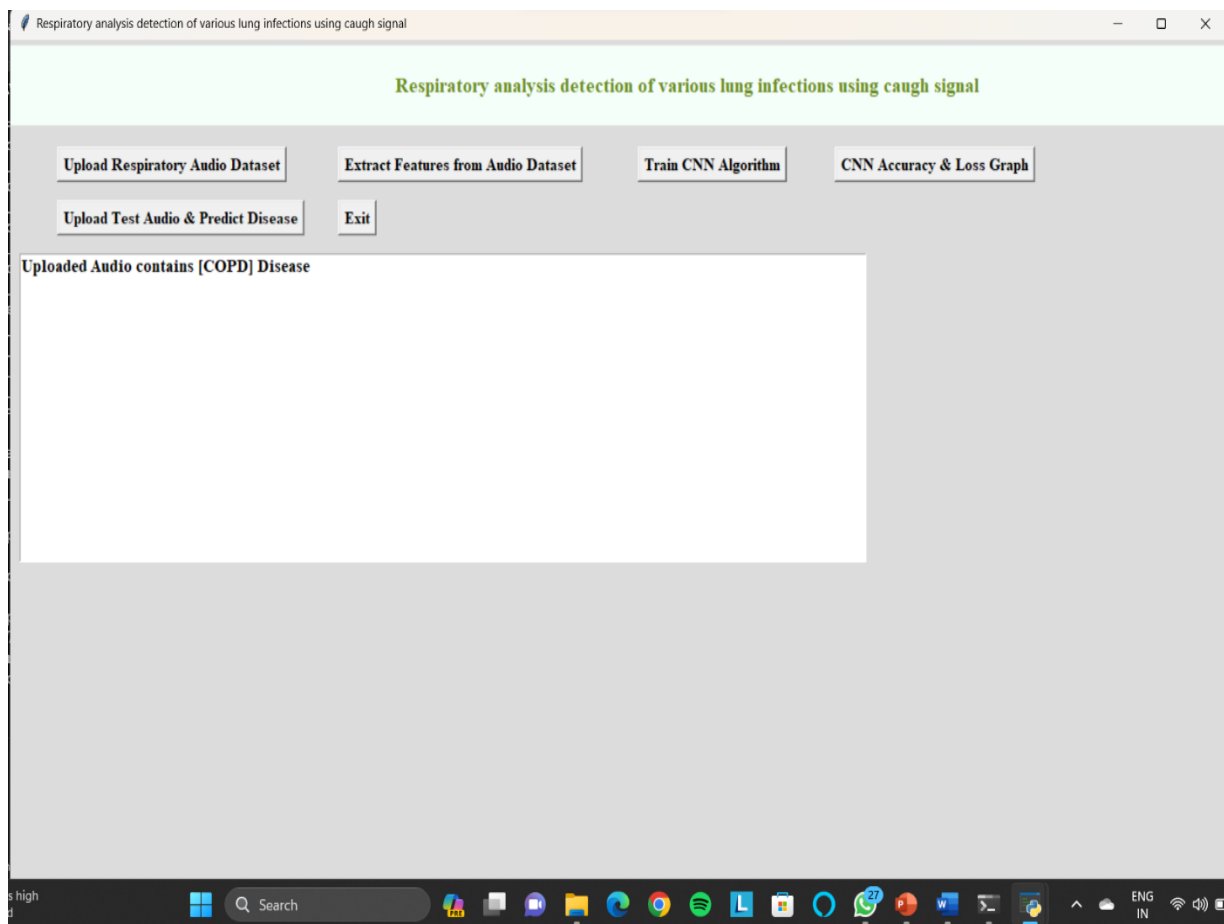
8.3 Screenshot

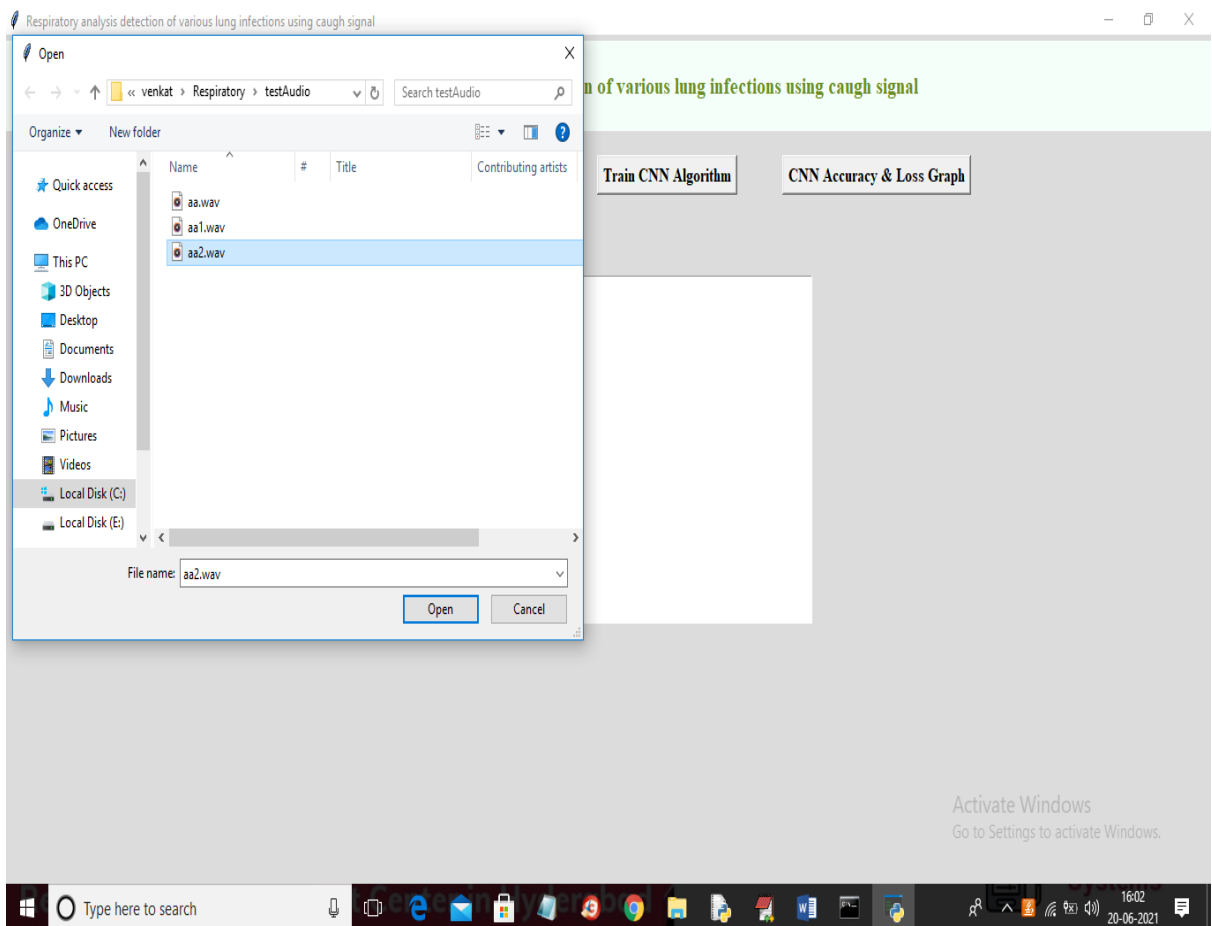
Now click on 'Upload Test Audio & Predict Disease' button to upload test audio file

In below screen selecting and uploading 'aa.wav' file and then click on 'Open' button to get below prediction result

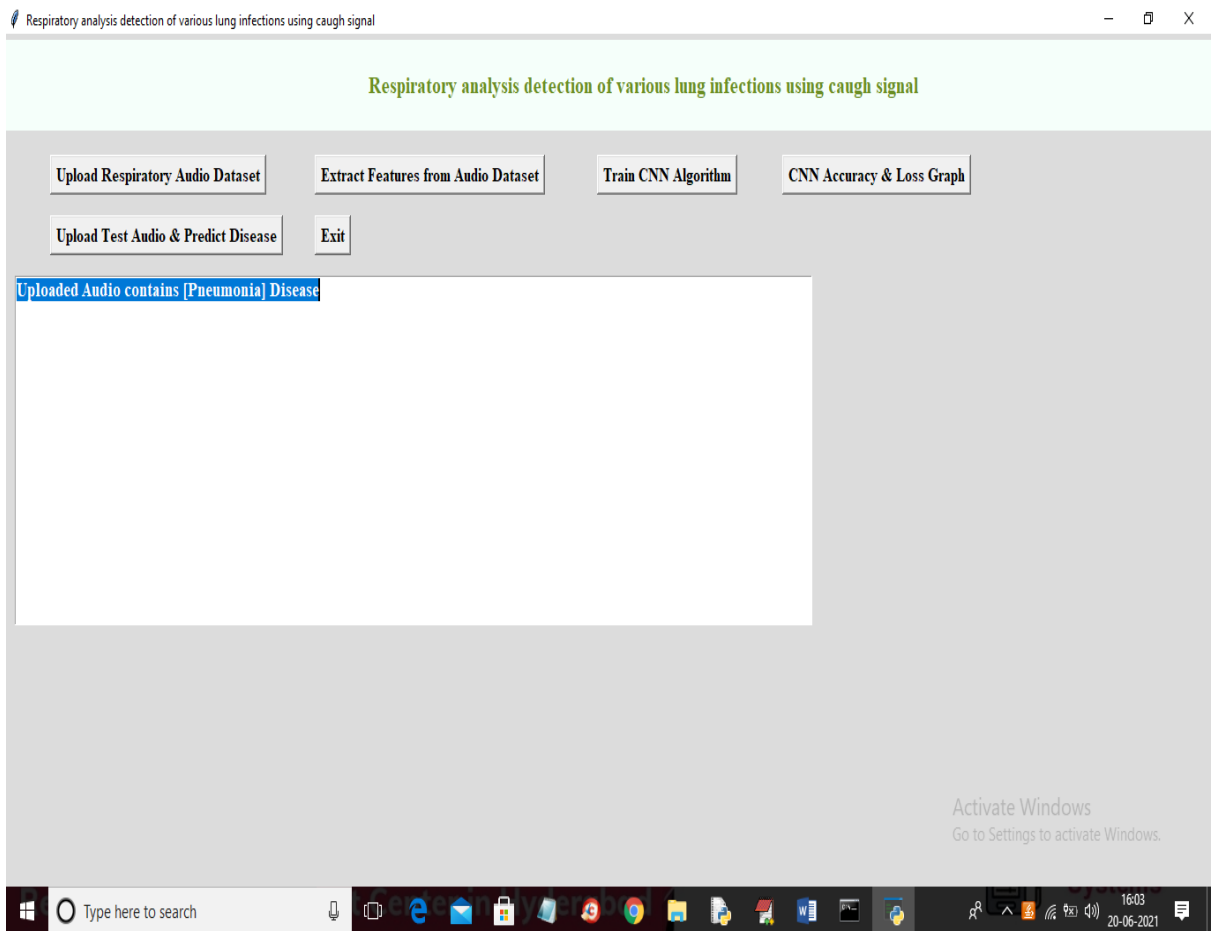


In below screen in blue colour text we can see disease predicted as “ASTHMA” form uploaded audio file and test with other file also





For above selected audio below is the result



8.4 Screenshot

In above screen uploaded audio disease predicted as 'Pneumonia' and similarly you can upload other files and predict disease

CHAPTER-9
CONCLUSION & FUTURE ENHANCEMENT

9.CONCLUSION & FUTURE ENHANCEMENT

Conclusion:

The lungs are important organs in the respiratory system and used for gas exchange (oxygen and carbon dioxide). When we breathe. Our lungs transfer oxygen from the air into the blood, and carbon dioxide from the blood into the air. To implement this project we have taken disease diagnosis dataset and respiratory audio dataset and then extract features from all audio dataset and then trained a convolution neural network (CNN) algorithm model. After training model we can upload any new test data to predict disease from it.

Future Enhancement

The future enhancement of respiratory analysis using cough signals holds significant potential in diagnosing and monitoring lung diseases. Incorporating cough signals into respiratory analysis could provide several benefits:

Early Detection: Analyzing cough signals could aid in early disease detection. Changes in cough patterns or characteristics might indicate the onset or progression of certain lung conditions.

Objective Assessment: Using cough signals for analysis can offer a more objective measure for assessing respiratory health, complementing other diagnostic methods. This can be particularly useful in diseases where subjective reporting may vary.

Monitoring and Management: Cough signal analysis might help in monitoring disease progression, treatment effectiveness, and overall management of respiratory conditions by providing continuous, real-time data.

Differentiation and Classification: It could assist in distinguishing between various lung diseases based on distinctive cough patterns or features, aiding in more accurate diagnosis and targeted treatment.

Telemedicine and Remote Monitoring: Incorporating cough signal analysis into telemedicine tools could enable remote monitoring of patients' respiratory health, allowing for early intervention and reducing the need for in-person visits. However, challenges like standardization of cough signal analysis, noise interference, and the need for comprehensive databases for various respiratory conditions to train algorithms.

CHAPTER 10
BIBLIOGRAPHY



10.BIBLIOGRAPHY

- [1] Hannah Bast, Mattias Hertel, Mostafa M. Mohammed “Tokenization repair in the presence of spelling errors.” Publisher: Association for Computational Linguistics, Volume: Proceedings of 25th conference on computational natural language learning, pages: 279-289, Issue: November 2021.
- [2] Yifei Hu, Xiaonan Jing, Youlim Ko “Misspelling Correction with Pre-trained Contextual Language Model”, arXiv: 2101.03204v1 [cs.CL] 8 Jan 2021.
- [3] Xiangci Li, Hairong liu, Liang Huang “Context aware stand-alone Neural spelling correction.” Findings of the Association for Computational Linguistics: EMNLP 2020, pages 407–414 November 16 - 2020.
- [4] Ahmed Yunus, Md Masum “A context Free Spell Correction Method using Supervised Machine learning Algorithms”, International Journal of Computer Applications, ISSN: 0975-8887, Volume 176, Pg No. 27, and June 2020.
- [5] Daniel Hladek, Jan Stas and Matus Pleva “Survey of Automatic Spelling Correction”, Issue Human Computer Interaction for Intelligent Systems, Published: 13 October 2020.
- [6] Pravallika Etoori, Manoj Chinnakotla, Radhika Mamidi “Automatic Spelling Correction for Resource-Scare Languages using Deep Learning” Proceedings of ACL 2018, Student Research Workshop, Issue: July 15 - 20, 2018.
- [7] Christanti, M.V.; Naga, D.S., “Fast and accurate spelling correction using trie and Damerau-levenshtein distance bigram”, Telkomnika (Telecommun. Comput. Electron. Control.) 2018, 16, 827–833.
- [8] S. M. El Atawy, A. Abd ElGhany “Automatic Spelling Correction based on n-Gram Model” International Journal of Computer Application, Volume: 182, Number: 11, Issue: 2018.
- [9] Tao Ge, Furu Wei, Ming Zhou “Reaching Human-level Performance in Automatic Grammatical Error Correction: An Empirical Study”, Cornell University article, Submitted on 3 Jul 2018, Version v5.
- [10] Keisuke sakaguchi, Kevin Duh, Matt post, Benjamin Van Durme “Robust word recognition via semi-character recurrent neural network.” Association for the Advancement of Artificial Intelligence, volume: 2, issue: & Feb 2017.
- [11] Miikka Silfverberg, Pekka Kaupppinen, Krister Linden “Data-Driven Spelling Correction using Weighted Finite-State Methods” Proceedings of the ACL Workshop on Statistical NLP and Weighted Automata, Issue: August, 2016.
- [12] Daniel Hladek, Jan Stas, Jozef Juhar, “Learning string distance with smoothing for OCR spelling correction”, Multimedia Tools and Applications, Published: 07 December 2016.

CHAPTER 11
YUKTHI INNOVATION CERTIFICATE

CHAPTER 11

YUKTHI INNOVATION CERTIFICATE

 <p>MoE's INNOVATION CELL (GOVERNMENT OF INDIA)</p>	<p>INSTITUTION'S INNOVATION COUNCIL</p> <p>MOE'S INNOVATION CELL</p>	 <p>INSTITUTION'S INNOVATION COUNCIL (Ministry of Education Initiative)</p>
Institute Name: Malla Reddy Institute of Technology & Science		
Title of the Innovation/Prototype: RESPIRATORY ANALYSIS DETECTION OF VARIOUS LUNG INFECTIONS USING COUGH SIGNAL		
Team Lead Name: Rakshitha Ramidi	Team Lead Email: ramidirakshitha@gmail.com	Team Lead Phone: 6281331076
Team Lead Gender: Female		
FY of Development: 2023-24	Developed as part of: Academic Requirement/Study Project	Innovation Type: Product, Process
		TRL LEVEL: 3
Theme: Healthcare & Biomedical devices..		
Define the problem and its relevance to today's market / society / industry need: Large number of people die every year of Pulmonary chronic lung diseases irrespective of their age. Lung sound analysis has been a key diagnostic aid to accurately detect pulmonary diseases.		
Describe the Solution / Proposed / Developed: In this project we are using respiratory audio dataset to predict various diseases such as Asthma, Pneumonia, Bronchiectasis and many more. To implement this project we have taken disease diagnosis dataset and respiratory audio dataset and extract features from all audio dataset and then trained a convolution neural network (CNN) algorithm model. After training model we can upload any new test data to predict disease from it.		
Explain the uniqueness and distinctive features of the (product / process / service) solution: In this project we are using respiratory audio dataset to predict various diseases such as Asthma, Pneumonia, Bronchiectasis and many more. To implement this project we have taken disease diagnosis dataset and respiratory audio dataset and extract features from all audio dataset and then trained a convolution neural network (CNN) algorithm model. After training model we can upload any new test data to predict disease from it.		
How your proposed / developed (product / process / service) solution is different from similar kind of product by the competitors if any: In this project we are using respiratory audio dataset to predict various diseases such as Asthma, Pneumonia, Bronchiectasis and many more. To implement this project we have taken disease diagnosis dataset and respiratory audio dataset and extract features from all audio dataset and then trained a convolution neural network (CNN) algorithm model. After training model we can upload any new test data to predict disease from it.		
Is there any IP or Patentable Component associated with the Solution?: No		