

A ROBUST DEEP LEARNING METHOD FOR RADIATION INDUCED PULMONARY FIBROSIS DISEASE CLASSIFICATION IN LUNG CT

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Technology in Biomedical Engineering.

by

NISHA S (39240027)

KEERTHANA R (39240019)



**DEPARTMENT OF BIOMEDICAL ENGINEERING
SCHOOL OF BIO AND CHEMICAL ENGINEERING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119**

APR – 2023



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF BIOMEDICAL ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of Nisha S (39240027), Keerthana R (39240019) who have done the project work as a team and carried out the project entitled "A Robust Deep Learning Method for Radiation Induced Pulmonary Fibrosis Disease Classification in Lung CT" under our supervision from October 2022 to April 2023.

Internal Guide
(Ms. SINDU DIVAKARAN)

Head of the Department
(Dr. T. SUDHAKAR)

HEAD OF THE DEPARTMENT
DEPARTMENT OF BIOMEDICAL ENGINEERING
SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Vandalur Nagar, Rajiv Gandhi Salai
Chennai - 600 077

Submitted for Viva-voce Examination held on 18.04.2023

Internal Examiner

External Examiner

DECLARATION

We, Nisha S (39240027) and Keerthana R (39240019) hereby declare that the Project Report entitled "A Robust Deep Learning Method for Radiation Induced Pulmonary Fibrosis Disease Classification in Lung CT" done by us under the guidance of Ms. Sindu Divakaran, Assistant Professor, Department of Biomedical Engineering at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology degree in Bio-Medical Engineering.

DATE: 18/04/2023

PLACE: Chennai

S. Nisha

1. Nisha S

Keerthana R

2. Keerthana R

SIGNATURE OF THE CANDIDATES

ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere thanks to the **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. We are grateful to them.

We convey our thanks to **Dr. T. Sudhakar M.Sc., Ph.D., Head of the Department, Department of Biomedical Engineering** for providing us necessary support and details at the right time during the progressive reviews.

We would like to express our sincere and deep sense of gratitude to our Project Guide **Ms. Sindu Divakaran, Assistant Professor, Department of Biomedical Engineering** for her valuable guidance, suggestions, and constant encouragement paving the way for the successful completion of our project work.

We wish to express our thanks to all **Teaching and Non-teaching** staff members of the **Department of Biomedical Engineering** who were helpful in many ways for the completion of the Project.

We would also like to express our gratitude towards our parents for supporting us in the successful completion of our project.

ABSTRACT

A chronic lung condition called pulmonary fibrosis, damages the tissues of the lungs by scarring, thickening, and hardening them. Detecting Pulmonary Fibrosis may be a significant task for medical professionals today. To overcome this problem, medical professionals have used various techniques and detection methods to inhibit the proliferation of Pulmonary Fibrosis. CT Scan is currently the simplest method for detecting Pulmonary Fibrosis. This uses CT images, which have the potential to be more effective than X-rays. This diagnostic method is extremely accurate because it can see organs in three dimensions. However, this method requires a radiologist to detect the disease and requires an extended time, which suggests it'll cut valuable time for medical practitioners if a patient is sick. As a result, as a quick alternative, it is necessary to implement a system to detect pulmonary fibrosis automatically. This is intended to help doctors detect computed tomography (CT) images of lungs infected with Pulmonary Fibrosis. The VGG19 transfer learning is used for the image classification process. Chest imaging plays a significant role in this phase where CT scans have proven to be effective in detecting Pulmonary Fibrosis within the lungs. The work is complicated by the lesions varying positions and sizes, uncertain boundaries, irregular shapes, and difficulty in obtaining a large collection of volumetric images with annotations for learning purposes. In this study, we propose a deep learning model to detect Pulmonary Fibrosis.

Keywords: Pulmonary fibrosis, Image Processing, Data Augmentation, Lung CT, Input Image, Pre-Processing, Model Training, Model Testing, Flask Framework.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO.
	ABSTRACT	i
	LIST OF ABBREVIATIONS	iv
	LIST OF FIGURES	vi
	LIST OF TABLES	viii
1	INTRODUCTION	01
	1.1.BASIC INTRODUCTION	01
	1.2.PULMONARY FIBROSIS	03
	1.3.SYMPTOMS	04
2	LITERATURE SURVEY	06
3	AIM AND SCOPE	08
	3.1. AIM	08
	3.2. SCOPE	08
4	MATERIALS AND METHODS	09
	4.1.SYSTEM REQUIREMENTS SPECIFICATION	09
	4.2.SOFTWARE SPECIFICATION	09
	4.2.1.PYTHON	09
	4.2.2. HYPERTEXT MARKUP LANGUAGE (HTML)	10
	4.2.3.CASCADING STYLE SHEETS (CSS)	10
	4.2.4.ANACONDA	11
	4.2.5.PYCHARM	12
	4.2.6.CONFUSION MATRIX	13
	4.3.BLOCK DIAGRAM OF PROPOSED MODEL	15
	4.3.1.COLLECTING DATASET	15

	4.3.2.PREPROSSING	16
	4.3.3.VGG - 19	17
	4.3.4.MODEL TRAINING AND TESTING	18
	4.3.5.FLASK FRAMEWORK AND WEB APPLICATION	19
5	RESULTS AND DISCUSSION	22
	5.1.COLLECTED DATASET IMAGES	22
	5.2.MODEL TRAINING AND TESTING IMAGES AND OUTPUT	23
	5.3.OUTPUT OF FLASK FRAMEWOEK AND WEB APPLICATION	29
	5.4.DISCUSSION	32
6	SUMMARY AND CONCLUSION	35
	6.1.SUMMARY	35
	6.2.CONCLUSION	36
	REFERENCES	37
	APPENDIX	40

LIST OF ABBREVIATIONS

ABBREVIATIONS	DESCRIPTION
CNN	CONVOLUTION NEURAL NETWORK
COVID	CORONA VIRUS DISEASE
CSS	CASCADING STYLE SHEETS
CT	COMPUTED TOMOGRAPHY
DL	DEEP LEARNING
FN	FALSE NEGATIVE
FP	FALSE POSITIVE
HTML	HYPER TEXT MARKUP LANGUAGE
IDE	INTEGRATED DEVELOPMENT ENVIRONMENT
ILD	INTERSTITIAL LUNG DISEASE
IPF	IDIOPATHIC PULMONARY FIBROSIS
JPEG	JOINT PHOTOGRAPHIC EXPERTS GROUP
MIME	MULTIPURPOSE INTERNET MAIL EXTENSIONS
MVC	MODEL VIEW CONTROLLER
PF	PULMONARY FIBROSIS
RGB	RED GREEN BLUE
SVC	SCALABLE VECTOR GRAPHICS
TN	TRUE NEGATIVE
TP	TRUE POSITIVE
URL	UNIFORM RESOURCE LOCATOR

VGG

VISUAL GEOMETRY GROUP

XML

EXTENSIBLE MARKUP LANGUAGE

XUL

XML USER INTERFACE LANGUAGE

LIST OF FIGURES

FIGURE	DESCRIPTION	PG.NO
4.1	Confusion Matrix	13
4.2	Block diagram of proposed model	15
4.3	Architecture of MVC	20
5.1	Positive Pulmonary Fibrosis CT Scan Image	22
5.2	Negative Pulmonary Fibrosis CT Scan Image	23
5.3	Model Trained Images	24
5.4	Model Testing Images	25
5.5	Confusion matrix without normalization and with normalization for epoch 5	26
5.6	Graph of confusion matrix illustrating accuracy and loss for epoch 5	26
5.7	Confusion matrix without normalization and with normalization for epoch 20	27
5.8	Graph of confusion matrix illustrating accuracy and loss for epoch 20	27
5.9	Link of Flask Web Application	29
5.10	Flask Web Application	29
5.11	Select the Image that has to be diagnosed	30
5.12	Upload the Image that has to be diagnosed	31

LIST OF FIGURES

FIGURE	DESCRIPTION	PG.NO
4.1	Confusion Matrix	13
4.2	Block diagram of proposed model	15
4.3	Architecture of MVC	20
5.1	Positive Pulmonary Fibrosis CT Scan Image	22
5.2	Negative Pulmonary Fibrosis CT Scan Image	23
5.3	Model Trained Images	24
5.4	Model Testing Images	25
5.5	Confusion matrix without normalization and with normalization for epoch 5	26
5.6	Graph of confusion matrix illustrating accuracy and loss for epoch 5	26
5.7	Confusion matrix without normalization and with normalization for epoch 20	27
5.8	Graph of confusion matrix illustrating accuracy and loss for epoch 20	27
5.9	Link of Flask Web Application	29
5.10	Flask Web Application	29
5.11	Select the Image that has to be diagnosed	30
5.12	Upload the Image that has to be diagnosed	31

5.13	Output of Non- Fibrosis Detected Image	31
5.14	Output of Fibrosis Detected Image	32

LIST OF TABLE

TABLE	DESCRIPTION	PG.NO
1.1	Types of Pulmonary Fibrosis	04
5.1	Tabulation of confusion matrix for epoch 5	28
5.2	Tabulation of confusion matrix for epoch 20	28

CHAPTER – 1

INTRODUCTION

1.1 BASIC INTRODUCTION:

A group of nearly 200 different pulmonary diseases is combined together to form a common group known as interstitial lung disease (ILD). Interstitial lung diseases are a diverse group of pulmonary parenchymal disorders caused by environmental insults, systemic diseases, and idiopathic conditions (Kropski J.A. et al., 2019). While being regarded as a rare disease, IPF has an incidence rate that ranges from 0.2 to 93.7 per 100,000 people annually, according to an analysis of 34 research on the disease's incidence and death (Rustam.F. et al., 2020). The radio-histopathological finding of the majority of diseases from this group suggests the progressive scarring of lung tissues, which would cause increased breathlessness in the patient. Automated lung detection from computed tomography (CT) images has the potential to significantly improve the current healthcare approach to tackling covid-19 (Zhou. Y. et al., 2020). It also disturbs the flow of oxygen into the bloodstream. The group shows similar clinical properties among them. It makes difficult to distinguish among them. It makes higher intra and inter variability of the observers. Even experienced medical experts find it difficult for differential diagnoses. Earlier diagnosis improves the chance of cure and recovery with less time. Computed tomography (CT) images (Afshar.X. et al., 2020) are identified as ideal imaging tools for studying patterns of ILD as it provides higher resolution as compared to CT images. Deep learning approach for the classification of pulmonary fibrosis plays an important role in medical domain (Isensee.F. et al., 2021). In particular for acute pulmonary embolism chest CT is crucial for corona virus disease 2019 patient therapy optimisation while also excluding alternative diagnosis or other pathologic diseases. (Grillet. F. et al., 2020)

Simon Walsh and colleagues employed a deep learning algorithm to categorize patients with idiopathic pulmonary fibrosis on CT in the Lancet Respiratory Medicine. Long-term pulmonary fibrosis raises the risk of developing lung cancer. Lung cancer is diagnosed using medical imaging, particularly Computer Tomography (CT)

images, and professionals find it challenging to understand and categorize cancer from the CT scan images (Jayaraj.D.et al.,2019). An aspect of the larger concept of deep learning is a subset of machine learning. Algorithms can produce labels for recently given unknown data using machine learning, which enables them to discover patterns from suitably classified (labelled) input data (usually images for radiology). Instead of being trained on specific rules, the system will recognize connections between input and output on its own. The categorization of the infection lesions from CT scans is crucial for the quantitative measurement of the disease progression in accurate diagnosis and follow-up assessment, where Computed Tomography (CT) imaging plays a critical role in the detection of manifestations in the lung associated with COVID-19 (Wang.G.et al.,2020). The application of machine learning to categorize images of pets as either being a dog or a cat would be a simplest example.

The algorithm is initially trained with expert-labelled images (validation or training set) and then tested using new, un-labelled images (testing set). Deep learning (DL) has demonstrated effectiveness in the field of medical imaging, and in the context of the recent COVID-19 epidemic, some research has begun to examine DL-based approaches for the aided diagnosis of lung disorders (Roy.S.et al.,2020).

Deep learning algorithms use an artificial neural network of layered connections to evaluate and process data from the raw input to the desired output classification. High-performing segmentation models using convolutional neural networks have made significant advancements in the medical imaging field (Tajbakhsh.N.et al.,2019). Early neural networks had few layers (typically less than five), but deep learning algorithms can have large numbers of layers, many of them specialized to augment or diminish specific input features. For the diagnosis, evaluation, and measurement of lung disorders, automatic, accurate lobe segmentation is essential. (Imran.A.et al.,2019).

The majority of research on deep learning focuses on patch-based classification (Agarwala.S.et al.,2020). Although the results show that deep learning methods can classify fibrotic lung disease with essentially equivalent performance to subspecialist radiologists, there are several limitations. Evaluation of lung capacity decrease is a crucial stage in the management of pulmonary fibrosis, and computed

tomography (CT) imaging is a particular method of evaluating the severity of lung damage caused by pulmonary fibrosis(Wang. L.et al.,2020). In general, the performance of a deep learning algorithm is improved with increasing number of images used for training and increasing accuracy of the expert labelling.

Instead of depending on explicit rule-based algorithms or learning from being's characteristics, deep learning enables computers to extract accurate and predictive features from unprocessed data(Gerard.S.E.et al.,2020). Convolution neural networks (CNN) have recently made advances that provide a data-driven method for more efficient feature extraction in an end-to-end optimization process. Numerous studies have used CNNs efficiently as part of their lobe classification framework (Xie.W.et al.,2020). For this study, the training set consisted of only 929 scans, which is very small compared to most machine learning tasks. The examination of pulmonary function decrease is a vital phase in the treatment and therapy of pulmonary fibrosis, and computed tomography (CT) imaging is a particularly useful tool for finding out the volume of lung damage pulmonary fibrosis has caused (Huang.C.et al.,2019). This issue was partly addressed by subdivision of each of the studies into roughly 500 individual four-image sets (so-called montages), each of which was analysed separately. Second, only a single radiologist did the expert labelling of the training set. This factor introduces bias into the algorithm training.

1.2 PULMONARY FIBROSIS:

A wider category of more than 200 interstitial lung diseases or ILDs, include pulmonary fibrosis(PF), which is defined by pneumonitis and/or damage. In ILDs, the lung's air sacs' walls, as well as the tissue and area surrounding them, are injured and damaged.As a result of a large number of fibroblasts continuing to proliferate, which causes lung inflammation and ultimately changes in lung tissue structure, pulmonary fibrosis, also known as interstitial lung disease, is a chronic, progressive interstitial lung disease(Da Li.et al.,2022).Pulmonary fibrosis is the medical term for an infectious respiratory disease that causes scar tissue in the lungs. Pulmonary fibrosis can be brought on by inhaling dangerous chemicals.

Additionally, some illnesses, medications, and genetics can contribute to PF. Other types of pulmonary fibrosis, such as hereditary pulmonary fibrosis, pulmonary fibrosis linked to dyskeratosis gene defects or Herman-sky Pudlak syndrome, may have a multi-factorial etiology. The cause is typically unknown Idiopathic pulmonary fibrosis is what causes this (IPF). It can be difficult to diagnose and establish IPF because of its highly unpredictable disease heritage (Glassberg.M.K.et al.,2019).

Table 1.1. Types of Pulmonary Fibrosis

TYPES	CAUSES
Drug-Induced	Prior or current use of amiodarone, nitrofurantoin, chemotherapy, methotrexate or other drugs known to affect the lungs
Radiation-Induced	Prior or current radiation treatment to the chest
Environmental (hypersensitivity pneumonitis)	Exposure to mold, animals or other triggers
Autoimmune (connective tissue disease related)	Joint inflammation, skin changes(particularly on the fingers and face),dry eyes or mouth
Occupational (pneumoconiosis)	Prior or current exposure to dusts, fibers, fumes or vapour that can cause PF (such as asbestos, coal, silica and others)
Idiopathic	When no cause can be identified

1.3 SYMPTOMS:

PF symptoms sometimes resemble those of other lung conditions. PF symptoms typically appear in patients between the ages of 50 and 70. breathing challenges (Dyspnea). prolonged cough that produces no results, breathing very quickly and shallowly (particularly when walking), gradual unintentional weight loss, increased tiredness/fatigue, and low-grade fevers Joint pain, myalgias in the muscles (arthralgias), the tip of the fingers or toes becoming clubbed (wider and rounded). An early lung disease diagnosis is crucial (An early lung disease diagnosis is crucial (Subrato Bharati.et al.,.2020).

The proposed research focuses on pulmonary fibrosis, which is one of the first diseases symptoms to manifest in a patient's lungs. Pre-processing, Model training, Model testing, Flask framework are the processes via which images of the lung affected by pulmonary fibrosis are processed.

CHAPTER – 2

LITERATURE SURVEY

"Lung transplantation for COPD/pulmonary emphysema" (2023) Geert M Verleden, Jens Gottlieb. One of the primary reasons for lung transplantation continues to be COPD and α -1 antitrypsin deficiency emphysema. Patients may be eligible for lung transplantation if all other treatment options, such as rehabilitation, oxygen therapy, noninvasive breathing, and lung volume reduction, have been tried and failed. Only a small percentage of patients with end-stage COPD will eventually receive a transplant due to an ongoing donor shortage. Depending on the expertise of the centre, the waiting list, the availability of donor lungs, and the patient's risk-benefit ratio, the operation may involve a single or double lung transplant.

"A pooled analysis of the risk prediction models for mortality in acute exacerbation of chronic obstructive pulmonary disease" (2023) Zile Ji, Xuanlin Li, Siyuan Lei, Jiaxin Xu, Yang Xie. Acute exacerbations of chronic obstructive pulmonary disease (AECOPD) have a poor prognosis, and severe AECOPD increases the chance of death. Differentiating between high- and low-risk categories is made easier by prediction models. Many prediction models have currently been developed and validated; however, they need to be carefully reviewed in order to identify better models that can be applied in clinical settings and provide data for upcoming studies.

"Etiologic Classification of Diffuse Parenchymal (Interstitial) Lung Diseases" (2022), Dr. von Haunersches Kinderspital, Many illnesses fall within the category of diffuse parenchymal lung diseases (DPLD), sometimes known as interstitial lung diseases (ILD). The characterization and classification of diseases enable sophisticated and individualized assessments of clinical disease, transmission risks due to genetic or environmental factors, and precision medicine therapies. Here, we give a succinct overview of the development of adult and juvenile pneumology ILD classification systems. Our focus is on an etiologic classification with four primary divisions: vascular diseases, systemic disease-related disorders, exposure-related disorders, and lung-

only (native parenchymal) disorders. Precision medicine and the discovery of novel entities depend on the division of diseases into molecularly defined entities.

"An Automated Diagnosis of Pulmonary Fibrosis Progression Prediction Using Honeycombing and Deep Learning" (2022). Raghu et al. characterized that IPF has an unpredictable deterioration in the patients' lung capacity, and it disturbs the aged crowd typically, mainly in the age group of 50 to 70 years. They also found that the median survival period was 3.8 years through the period 2001 to 2011. Raghu et al. did a study that demonstrates that smoking, environmental exposure, and microbial agents act as risk factors. They also studied the indications of this disease, primarily respiratory such as dry cough, fatigue, shortness of breath, reduced pulmonary function test results, and finally patterns of fibrosis in CT images of lungs.

F. Isensee, P. F. Jaeger, S. A. Kohl, J. Petersen, and K. H. Maier-Hein, "nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation," (2021). We created the deep learning-based segmentation technique nnU-Net, which adapts automatically to any new task, pre-processing, network architecture, training, and post-processing. A set of fixed parameters, interdependent rules, and empirical decisions are used to model the process's major design decisions. nnU-Net outperforms the majority of current techniques without the need for manual intervention, including highly specialised solutions on 23 open datasets used in international biomedical segmentation competitions.

Hollon, T. C. et al. "Near real-time intraoperative brain tumor diagnosis using stimulated Raman histology and deep neural networks" (2020). A parallel process that integrates deep convolutional neural networks (CNNs), label-free optical imaging, and stimulated Raman histology (SRH) to predict diagnoses at the bedside in almost real-time. In particular, our CNNs predict tumour diagnosis in the operating theatre in under 150 s, an order of magnitude faster than conventional techniques, thanks to training on over 2.5 million SRH pictures.

CHAPTER-3

AIM AND SCOPE

3.1 AIM

Our aim is to develop an automated system for classification of Pulmonary Fibrosis. The system can be used by healthcare specialists. The system incorporates image processing, pattern analysis, and computer vision techniques and is expected to improve the sensitivity, specificity, and efficiency of pulmonary fibrosis screening. The primary goal of medical imaging projects is to extract meaningful and accurate information from these images with the least error possible. The proper combination and parameterization of the phases enables the development of adjunct tools that can help on the early diagnosis or the monitoring of the fibrosis identification.

3.2 PROPOSED SYSTEM AND SCOPE

To develop Deep learning (DL) based automated diagnostic tools that can differentiate between pulmonary fibrosis (PF) and non - pulmonary fibrosis interstitial lung diseases (ILDs) in patients using high-resolution computed tomography (CT) scans. The primary motivation behind the Pulmonary fibrosis detection project is to identify the presence of a fibrosis. This would be especially beneficial in cases where it is necessary to determine if the fibrosis is positive or negative. By analyzing medical images and using computer-based procedures, the system can accurately detect fibrosis in lungs. The project focuses on utilizing the Convolution Neural Network Algorithm to analyze CT images from various patients, thereby allowing for efficient and reliable identification of fibrosis in lung. This system has the potential to be highly effective in aiding medical professionals in making informed decisions for the treatment and management of patients with pulmonary fibrosis. Little dependence on pre-processing, decreasing the needs of human effort developing its functionalities. It is easy to understand and fast to implement. It has the highest accuracy among all algorithms that predicts images.

CHAPTER-4

MATERIALS AND METHOD

4.1 SYSTEM REQUIREMENTS SPECIFICATION

4.1.1 Software Requirements

- Operating System → Windows 10
- Server-side Script → Python 3.6
- IDE → PyCharm
- Libraries Used → Pandas
- Framework → Flask

4.2 SOFTWARE SPECIFICATION:

4.2.1 Python

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library. Guido van Rossum began working on Python in the late 80s as a successor to the ABC programming language and first released it in 91 as Python 0.9.0. Python 2.0 was released in 2000 and introduced new features such as list comprehensions, cycle-detecting garbage collection, reference counting, and Unicode support. Python 3.0, released in 2008, was a major revision that is not completely backward-compatible with earlier versions. Python 2 was discontinued with version 2.7.18 in 2020.

Python consistently ranks as one of the most popular programming languages. Python is a multi-paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of its features support functional

programming and aspect-oriented programming (including meta programming and meta-objects [magic methods]. Many other paradigms are supported via extensions, including design by contract and logic programming.

Python uses dynamic typing and a combination of reference counting and a cycle-detecting garbage collector for memory management. It uses dynamic name resolution (late binding), which binds method and variable names during program execution.

4.2.2 Hyper Text Markup Language(HTML)

The Hyper Text Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.

Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document. HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items.

HTML markup consists of several key components, including those called tags (and their attributes), character-based data types, character references and entity references. HTML tags most commonly come in pairs like `<h1>` and `</h1>`, although some represent empty elements and so are unpaired, for example `<img>`. The first tag in such a pair is the start tag, and the second is the end tag (they are also called opening tags and closing tags). Another important component is the HTML document type declaration, which triggers standards mode rendering.

4.2.3 Cascading Style Sheets (CSS)

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML or XML (including XML dialects such as SVG, MathML or XHTML). CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility; provide more flexibility and control in the specification of presentation characteristics; enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, which reduces complexity and repetition in the structural content; and enable the .css file to be cached to improve the page load speed between the pages that share the file and its formatting.

Separation of formatting and content also makes it feasible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader), and on Braille-based tactile devices. CSS also has rules for alternate formatting if the content is accessed on a mobile device.

The name cascading comes from the specified priority scheme to determine which style rule applies if more than one rule matches a particular element. This cascading priority scheme is predictable. The CSS specifications are maintained by the World Wide Web Consortium (W3C). Internet media type (MIME type) text/css is registered for use with CSS by RFC 2318 (March 98). The W3C operates a free CSS validation service for CSS documents.

In addition to HTML, other markup languages support the use of CSS including XHTML, plain XML, SVG, and XUL. CSS has a simple syntax and uses a number of English keywords to specify the names of various style properties.

4.2.4 Anaconda

More than 250 packages are pre-installed in the anaconda distribution, while more than 7,500 additional open-source packages, the conda package manager, and the virtual environment manager may all be downloaded from PyPI. Additionally, Anaconda Navigator, a graphical user interface, is provided as an alternative to the command line interface. (CLI). The management of package dependencies is a significant difficulty for Python data science and the main distinction between conda and the pip package manager. This is the basis for conda's existence. Any Python packages that depend on a package are immediately installed during pip installation without being compared to already installed packages. Regardless of the condition of the current installation, it will install a package along with any of its dependencies. As a result, a user with a functional. While conda determines how to install a compatible set of dependencies and displays a warning if this cannot be done, it analyses the current environment, including everything that is currently installed, along with any version limitations specified (for example, the user may wish to have Tensorflow version 2, 0 or higher).

Using the conda install command, users can install individual open source packages from the Anaconda repository, Anaconda Cloud (anaconda.org), or their own personal repository or mirror. Binaries for Windows 32/64-bit, Linux 64-bit, and Mac OS 64-bit are provided by Anaconda, Inc., which also creates and compiles the packages that are made accessible in the Anaconda repository itself. Pip can be used to install anything from PyPI into a conda environment, and conda will keep track of what it installs.

4.2.5 Pycharm

PyCharm is an integrated development environment (IDE) for professional Python developers that helps them be more productive, confident, and produce better code. It is available in two versions: PyCharm Pro and PyCharm Community, with the latter supporting the whole Python workflow out of the box, including web frameworks, frontend technologies, databases, and scientific instrumentation. PyCharm's community edition is completely free and open source. It fully supports Python,

providing code insight, debugging, testing, and other features. In the following areas, PyCharm provides its customers and developers with some of the best features:

- Completion and inspection of the code
- Support for web programming with frameworks like Django and Flask with advanced debugging.

4.2.6 Confusion matrix

A confusion matrix (fig 4.1) presents a table layout of the different outcomes of the prediction and results of a classification problem and helps visualize its outcomes. It plots a table of all the predicted and actual values of a classifier.

The diagram illustrates a confusion matrix for a two-class classification problem. It features a 2x2 grid of cells. Above the grid, a horizontal double-headed arrow is labeled 'ACTUAL'. To the left of the grid, a vertical double-headed arrow is labeled 'PREDICTED'. The columns represent the actual classes: 'Positive' and 'Negative'. The rows represent the predicted classes: 'Positive' and 'Negative'. The four cells contain the following labels: Top-Left (Actual Positive, Predicted Positive) is 'TRUE POSITIVE'; Top-Right (Actual Negative, Predicted Positive) is 'FALSE POSITIVE'; Bottom-Left (Actual Positive, Predicted Negative) is 'FALSE NEGATIVE'; Bottom-Right (Actual Negative, Predicted Negative) is 'TRUE NEGATIVE'.

		ACTUAL	
		Positive	Negative
PREDICTED	Positive	TRUE POSITIVE	FALSE POSITIVE
	Negative	FALSE NEGATIVE	TRUE NEGATIVE

Fig 4.1. Confusion Matrix

A two-class classification problem with the two outcomes being Positive and Negative can be considered. Given, a data point to predict, the models outcome will be any of these two.

By plotting the predicted values against the ground truth (actual) values, we get a matrix with the following representative elements:

True Positive (TP): These are the data points whose actual outcomes were positive and the algorithm correctly identified it as positive.

True Negatives (TN): These are the data points whose actual outcomes were negative and the algorithm correctly identified it as negative.

False Positives (FP): These are the data points whose actual outcomes were negative but the algorithm incorrectly identified it as positive.

False Negatives (FN): These are the data points whose actual outcomes were positive but the algorithm incorrectly identified it as negative.

The goal of evaluating a model using the confusion matrix is to maximize the values of TP and TN and minimize the values of FP and FN.

For an instance in prediction of pulmonary fibrosis disease, the outcomes would be a patient having pulmonary fibrosis disease or not. The confusion matrix here would be TP means that the patient actually has PF disease and the algorithm predicted it correctly. TN means the patient doesn't have PF disease and the algorithm predicted it correctly. So the goal should be to keep these values as high as possible.

To evaluate confusion matrix few factors are taken into consideration and are to be measured.

1. Accuracy: The number of samples correctly classified out of all the samples present in the test set.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

2. Precision (for the positive class): The number of samples actually belonging to the positive class out of all the samples that were *predicted* to be of the positive class by the model.

$$\text{Precision} = \frac{TP}{TP + FP}$$

3. Recall (for the positive class): The number of samples predicted correctly to be belonging to the positive class out of all the samples that *actually belong* to the positive class.

$$\text{Recall} = \text{TP} / \text{TP} + \text{FN}$$

4. F1-Score (for the positive class). The harmonic mean of the precision and recall scores obtained for the positive class.

$$\text{F1-Score} = 2 * \text{Precision} * \text{Recall} / \text{Precision} + \text{Recall}$$

5. Specificity: The number of samples predicted correctly to be in the negative class out of all the samples in the dataset that actually belong to the negative class.

$$\text{Specificity} = \text{TN} / \text{FP} + \text{TN}$$

4.3 BLOCK DIAGRAM OF PROPOSED MODEL

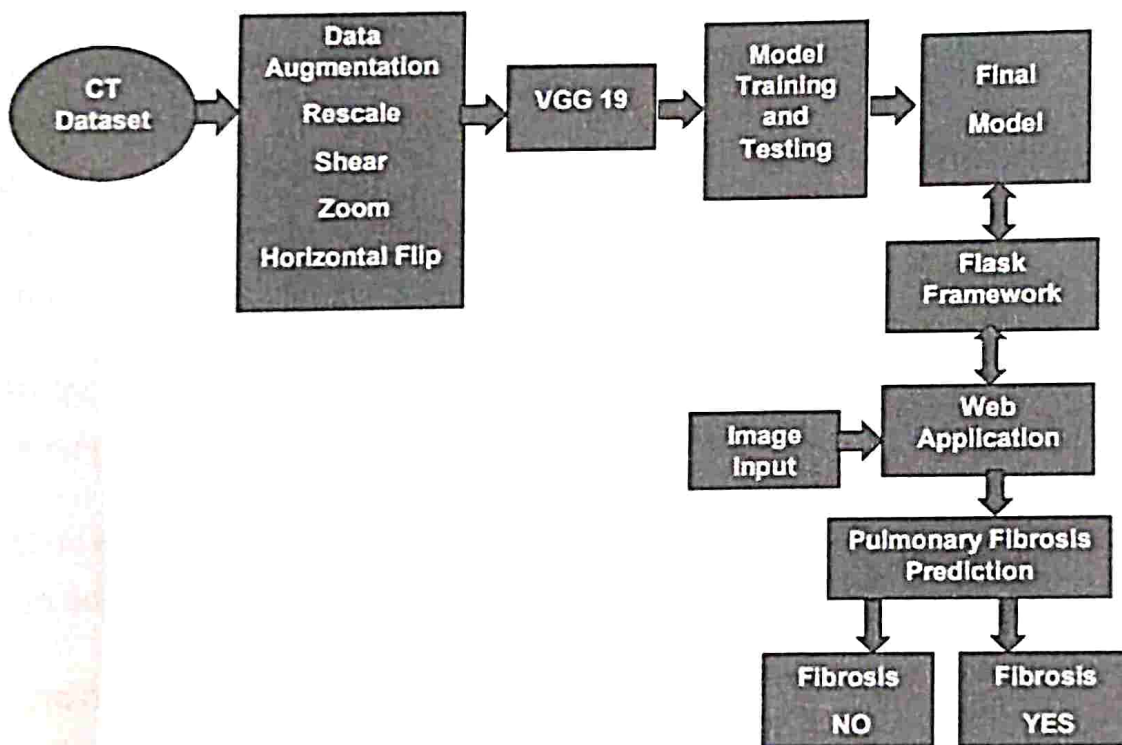


Fig 4.2. Block diagram of Proposed Model

4.3.1 Collecting dataset

The dataset was collected from kaggle website. The distribution of the CT and X-ray images among the three categories are as follows:

The training dataset folder named **“train”** consists of images to train the model. The validation dataset folder named **“val”** consists of images to validate the model in every epoch. They are used to obtain the training and validation accuracies and losses in every epoch while training the model. The images of the folder named **“test”** are completely unseen images of the hand gestures. Collecting data allows to capture a record of past events so that the data can be analysed to find recurring patterns. From those patterns, predictive models using deep learning algorithms can be built that look for trends and predict future changes.

4.3.2 Pre-processing

Data augmentation consists of gray scaling (RGB/BW to ranges of grey), Reflection (vertical / horizontal flip), Gaussian Blur (reduces image noise), Histogram equalization (increase global contrast), Rotation (may not preserve image size), Translation (moving the image along x or y axis), linear transformation such as random rotation (0-10 degrees), horizontal and vertical shifts, and horizontal and vertical flips. Data augmentation is done to teach the network desired invariance and robustness properties, when only few training samples are available.

An inception of v3 will be trained, ResNet50, VGG-19, Xception models on custom training dataset to classify among the two categories.

While resizing the images, `x_train`, `x_test`, `x_val`, `y_train`, `y_test` and `y_val` hand-in-hand can be derived. These terms can be described in a nutshell as follows:

`x_train`: Numpy arrays of the images of the training dataset

`y_train`: Labels of the training dataset

`x_test`: Numpy arrays of the images of the testing dataset

`y_test`: Labels of the testing dataset

`x_val`: Numpy arrays of the images of the validation dataset

y_val: Labels of the validation dataset.

The labels of the corresponding datasets using Image Data Generator was computed. This is used because the images are stored in folders. Inside the folders find out the corresponding labels of the images stored. Labels in this case are nothing but infected and uninfected.

4.3.3 VGG – 19

VGG19 is a convolutional neural network (CNN) that has been widely used for image classification tasks. It is a deep learning architecture that is known for its accuracy in image recognition and classification. In the context of Pulmonary Fibrosis prediction, the VGG19 architecture can be used to classify CT scan images of lungs as either healthy or infected with Pulmonary Fibrosis. This involves training the model on a large dataset of CT scan images labelled as either healthy or infected with Pulmonary Fibrosis. During training, the VGG19 architecture learns to extract features from the input images and use them to make a prediction about the presence of Pulmonary Fibrosis. The trained model can then be used to predict the presence of Pulmonary Fibrosis in new CT scan images. One advantage of using the VGG19 architecture for Pulmonary Fibrosis prediction is its high accuracy. The architecture has been shown to achieve high accuracy rates in image classification tasks, which is crucial in medical applications such as disease diagnosis. Additionally, the VGG19 architecture is relatively easy to implement and can be trained using a variety of open-source deep learning libraries, such as Keras and Tensor Flow.

Steps involved in CNN:

Step 1: Choose a Dataset.

Step 2: Prepare Dataset for Training.

Step 3: Create Training Data.

Step 4: Shuffle the Dataset.

Step 5: Assigning Labels and Features.

Step 6: Normalizing X and converting labels to categorical data.

Step 7: Split X and Y for use in CNN.

4.3.4 Model Training and Testing

This step includes model building, model compilation, and finally fitting the model.

Model building: As mentioned earlier, the inception v3, ResNet50, VGG-19, Xception pre-trained model are used to classify infected and uninfected. Coming to the implementation, inception v3, ResNet50, VGG-19, Xception are imported.

Compiling the model: In this step, the labels of none of the datasets are one-hot encoded been observed. So, using sparse categorical cross-entropy as our loss function. The best optimizer called adam optimizer is used as it decides the best learning rate on its own.

Fitting the model: This step involves training the model. However, during the training it must be ensured that the model does not get over fit.

Overfitting is said to be the phenomenon due to which the model will work perfectly on trained dataset but will not work very well on new data. In that case, the model is also said to be over fitted to the particular dataset.

Thus, it is necessary to stop training the model at the earlier stage further if the validation loss suddenly starts increasing. The loss must decrease gradually as the model gets trained. If the model is trained for example 10 epochs. In case overfitting is observed anytime during the training, the above code waits for five more epochs (indicated by `patience=5` in the above code). A contiguous increase of validation loss for more than five epochs forcibly stops the model training.

The above snippet indicates for what purpose `x_train`, `x_val`, `y_train`, and `y_val` in compiling the model is obtained. It is better to set `shuffle` to `True` so that the model does not see the same image repeatedly in different batches.

Now, evaluation of the model is done by testing it on the test dataset. To obtain the deep learning model that function efficiently, not only observing its accuracy is enough but also it is equally essential to observe its classification report and confusion matrix. This is another reason behind computing `x_test` and `y_test` in pre-processing.

4.3.5 Flask Framework and Web application

Once the model is trained, it can be integrated into a Flask web application, allowing medical professionals to easily access and utilize the model for patient diagnosis. The Flask application can be built to accept input in the form of CT scan images of a patient's lungs. The input images can be processed and fed into the deep learning model for Pulmonary Fibrosis prediction. The model's output can then be displayed on the web application for medical professionals to review and make a diagnosis. Additionally, the Flask application can be designed to include other features, such as data visualization of the patient's CT scan images diagnosis results. This can make the application more comprehensive and user-friendly for medical professionals. Integrating the deep learning model for Pulmonary Fibrosis prediction into a Flask web application can provide a convenient and efficient tool for medical professionals in diagnosing the disease. The application can be customized to include additional features and can improve the accuracy and speed of diagnosis.

Steps in flask framework:

Step 1- Install Flask: The first step is to install Flask on the system. Flask can be installed using `pip`, which is the package installer for Python.

Step 2- Set up the project structure: Once Flask is installed, create a project directory with the required subdirectories and files, such as templates and static files.

Step 3- Define the routes: In Flask, routes define the URL endpoints that the application responds to. The routes should be defined in the Flask app's main Python file.

Step 4- Define the views: The views define the business logic of the application. They receive the data from the user, process it, and provide a response.

Step 5- Define the templates: The templates are the HTML files that define the layout and structure of the application.

Step 6- Serve static files: Flask allows serving static files such as CSS, JavaScript and images.

Step 7- Integrate the deep learning model: Integrate the deep learning model for Pulmonary Fibrosis prediction into the Flask application. This involves loading the model, processing the input data, and providing the output.

Step 8- Test and deploy: Once the Flask application is built, it needs to be tested to ensure that it is working correctly. After testing, the application can be deployed on a server so that it can be accessed by medical professionals.

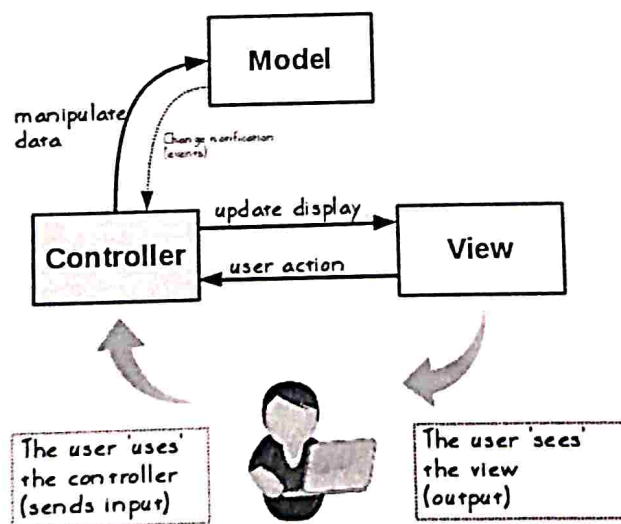


Fig 4.3. Architecture of MVC

Flask is a lightweight web framework for Python that allows you to build web applications quickly and easily. Flask works by Model-View-Controller (Fig 4.2) architecture pattern, where the model represents the data, the view represents the user interface, and the controller acts as an intermediary between the model and the view.

CHAPTER- 5

RESULTS AND DISCUSSION

5.1 COLLECTED DATASET IMAGES

Collecting data allows to capture a record of past events so that we can use data analysis to find recurring patterns.

From those patterns, you build predictive models using deep learning algorithms that look for trends and predict future changes.

Predictive models are only as good as the data from which they are built, so good data collection practices are crucial to developing high-performing models. CT images is collected for positive PF and negative PF.

About 300 images in positive and 300 images in negative are the collected dataset images.

Positive fib-19 CT Scan

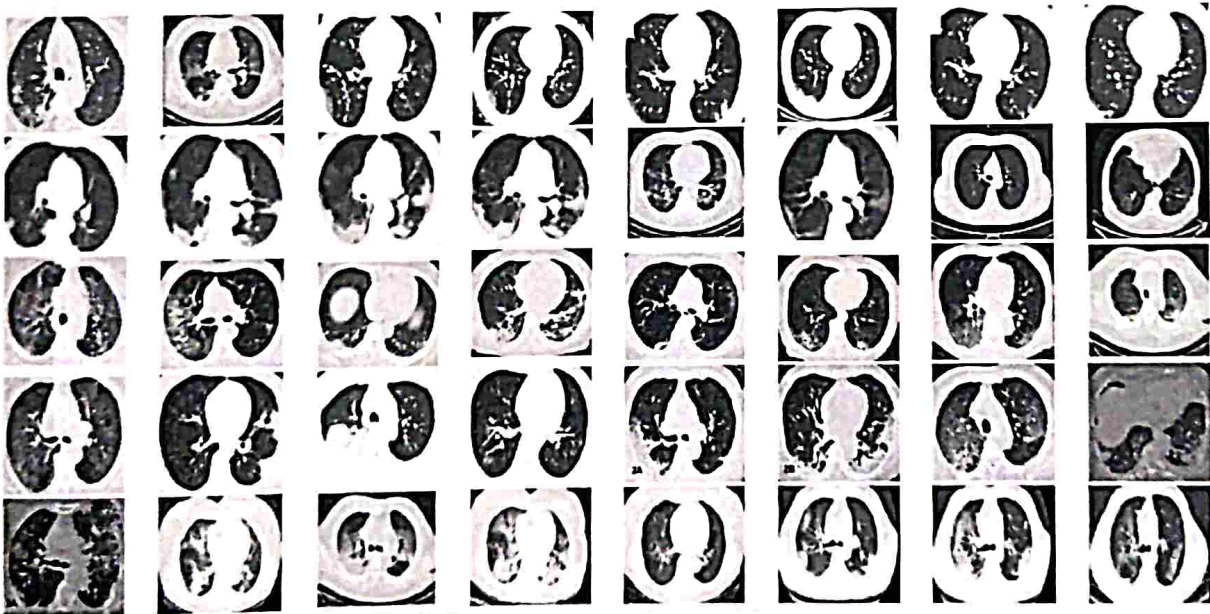


Fig 5.1. Positive Pulmonary Fibrosis CT Scan Images

Negative fib-19 CT Scan

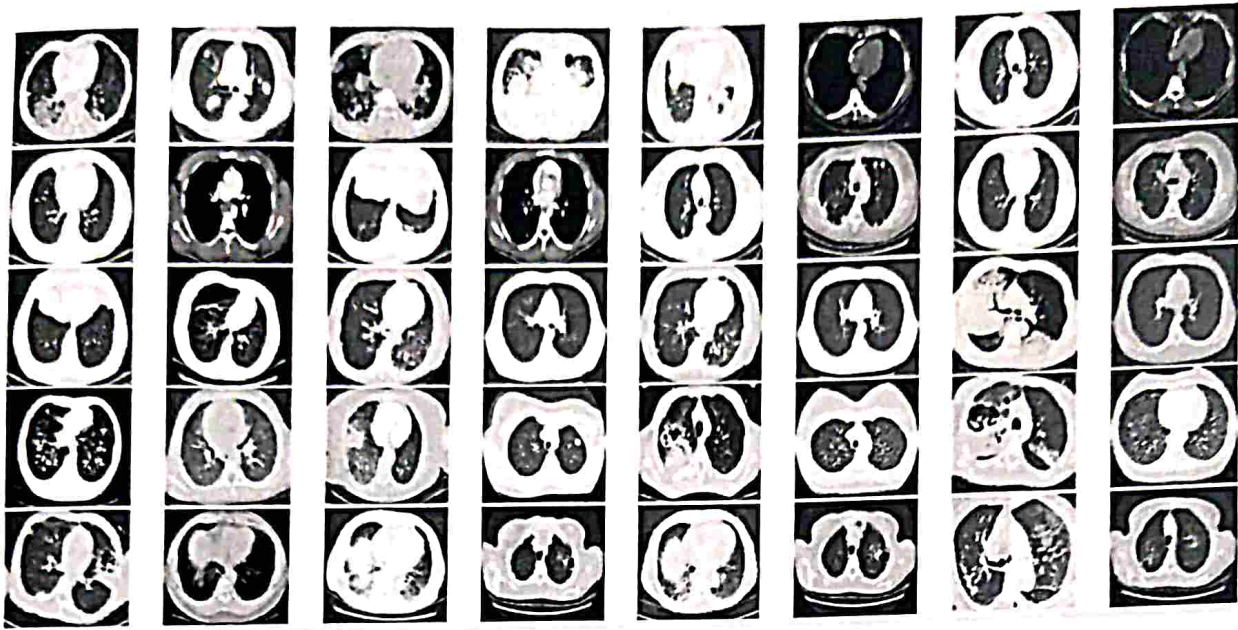


Fig 5.2. Negative Pulmonary Fibrosis CT Scan Images

5.2 MODEL TRAINING AND TESTING IMAGES AND OUTPUT

During training, the VGG19 architecture learns to extract features from the input images and use them to make a prediction about the presence of Pulmonary Fibrosis.

The trained model can then be used to predict the presence of Pulmonary Fibrosis in new CT scan images.

X_train

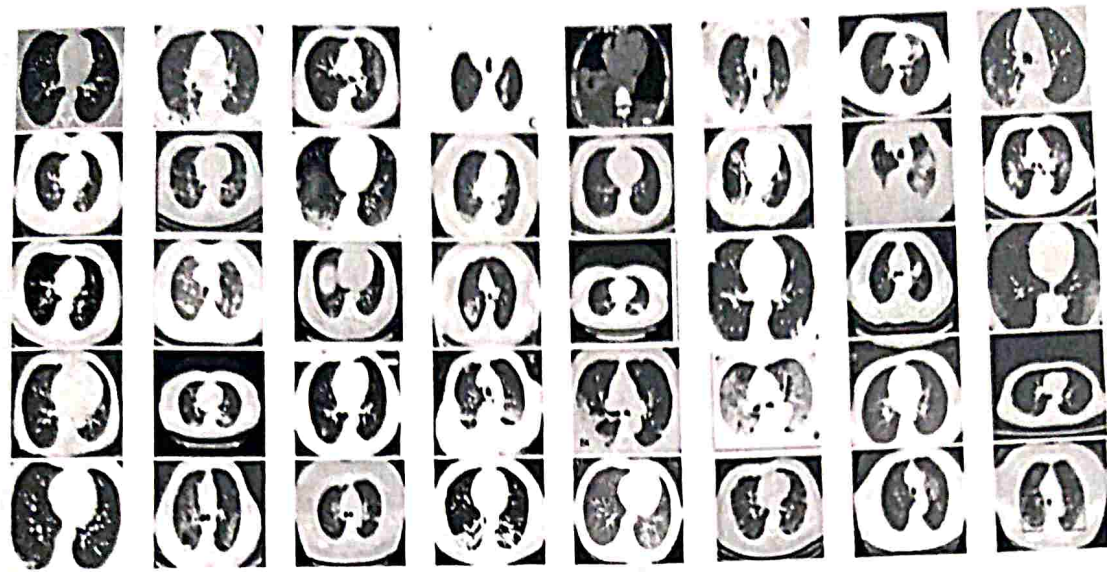


Fig 5.3. Model Trained Images

In this module, we test the trained deep-learning model using the test dataset, a type of test that makes detailed pictures of areas inside the body. Imaging tests use different forms of energy, such as x-rays (high-energy radiation), ultrasound (high-energy sound waves), radio waves, and radioactive substances. They may help diagnose disease, plan treatment, or determine how well treatment is working.

X_test

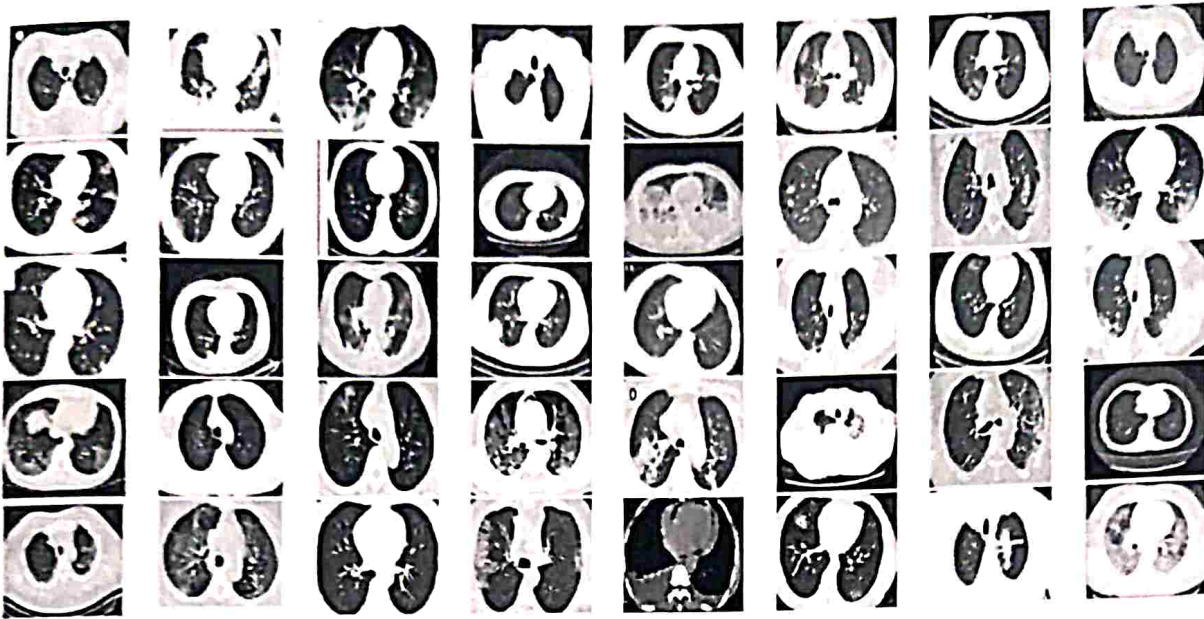


Fig 5.4. Model Testing Images

The Confusion Matrix created has four different quadrants:

- a. True Negative (Top-Left Quadrant)
- b. False Positive (Top-Right Quadrant)
- c. False Negative (Bottom-Left Quadrant)
- d. True Positive (Bottom-Right Quadrant)

True means that the values were accurately predicted, False means that there was an error or wrong prediction. A Confusion Matrix has been made to evaluate different measures to quantify the quality of the model.

An epoch is when all the training data is used at once and is defined as the total number of iterations of all the training data in one cycle for training the machine learning model. Another way to define an epoch is the number of passes a training dataset takes around an algorithm.

While comparing any of the two epochs the accuracy of the model will be increased. When epoch increases accuracy also increases.

For example, comparing values for epoch = 5 and epoch = 20

(fig 5.5 and fig 5.6) shows the Confusion matrix for epoch 5

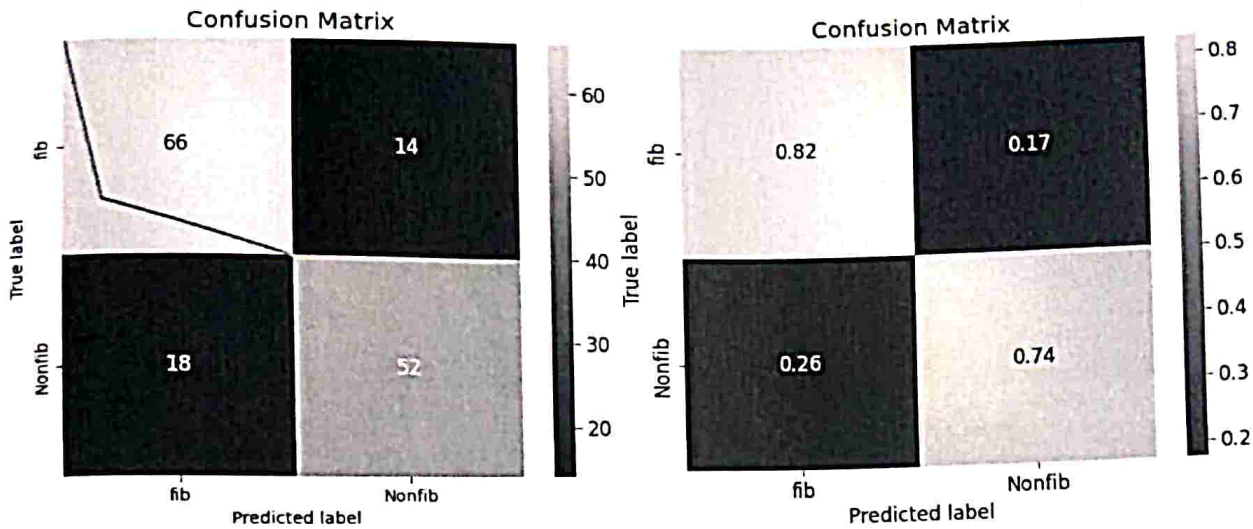


Fig. 5.5. Confusion matrix without normalization and with normalization for epoch 5

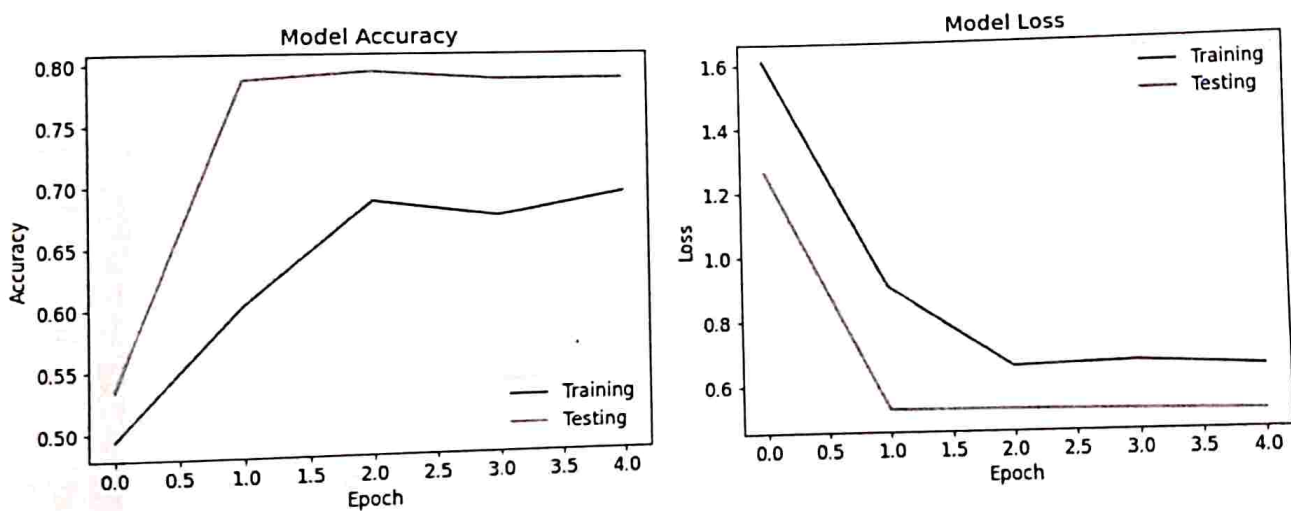


Fig. 5.6. Graph of confusion matrix illustrating accuracy and loss for epoch 5

The training and testing over time are visualized in the graph (fig 5.6). As expected accuracy for the testing set is slightly higher than that of the training. On contrary, loss for the training set is slightly higher than that of testing. Loss nearer to 0 indicates higher accuracy, whereas if the Loss is away from 0 then it indicates lower accuracy. Minimizing Loss gives greater accuracy for the classifier.

(fig 5.7 and fig 5.8) shows the Confusion matrix for epoch 20

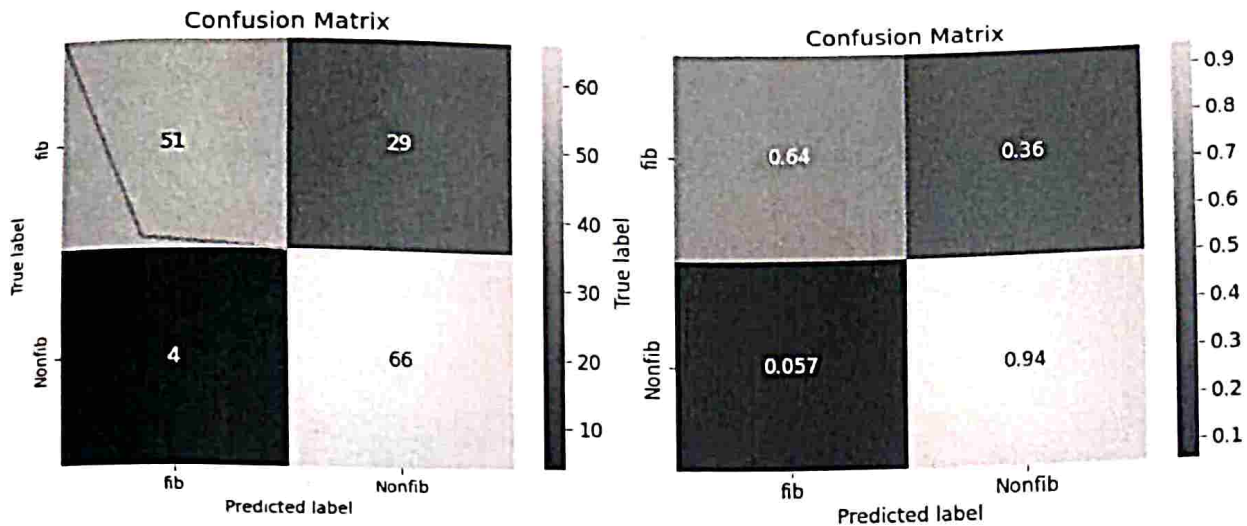


Fig. 5.7. Confusion matrix without normalization and with normalization for epoch 20

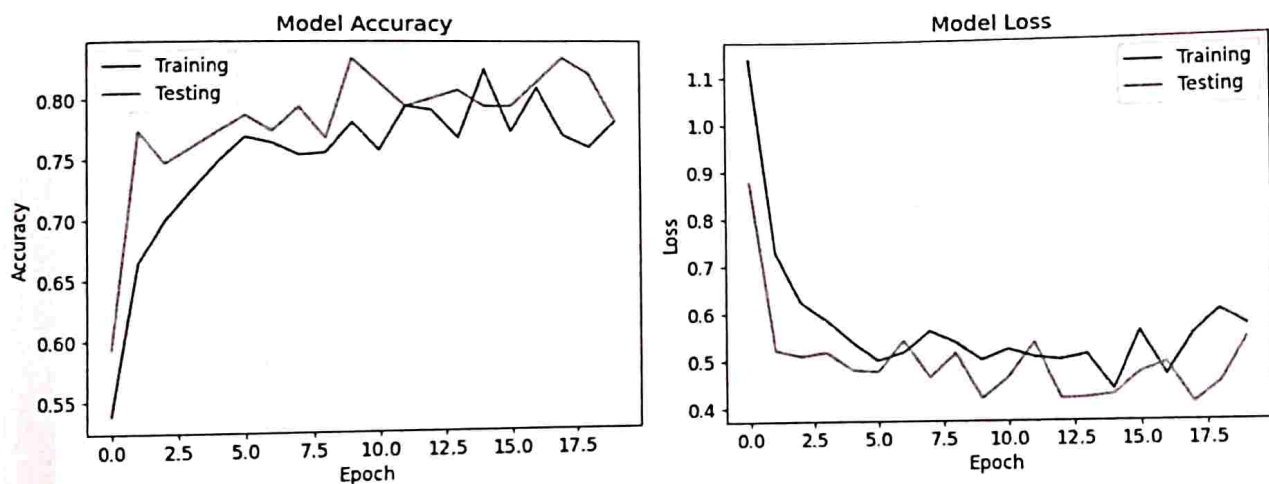


Fig. 5.8. Graph of confusion matrix illustrating accuracy and loss for epoch 20

The training and testing over time are visualized in the graph (fig 5.8). As expected accuracy for the testing set is slightly higher than that of the training. On contrary, loss for the training set is slightly higher than that of testing. Loss nearer to 0 indicates higher accuracy, whereas if the Loss is away from 0 then it indicates lower accuracy. Minimizing Loss gives greater accuracy for the classifier.

Macro averaging is perhaps the most straightforward among the numerous averaging methods. The macro-averaged F1 score is computed using the arithmetic mean of all

the per-class F1 scores. This method treats all classes equally regardless of their support values.

The weighted-average F1 score is calculated by taking the mean of all per-class F1 scores while considering each class's support.

Table 5.1. Tabulation of confusion matrix for epoch 5

	Precision	Recall	F1-score	Support
0	0.79	0.82	0.80	80
1	0.79	0.74	0.76	70
Accuracy			0.79	150
Macro average	0.79	0.78	0.78	150
Weighted average	0.79	0.79	0.79	150

Table 5.2. Tabulation of confusion matrix for epoch 20

	Precision	Recall	F1-score	Support
0	0.93	0.64	0.76	80
1	0.69	0.94	0.80	70
Accuracy			0.78	150
Macro average	0.81	0.79	0.78	150
Weighted average	0.82	0.78	0.78	150

5.3 OUTPUT OF FLASK FRAMEWORK AND WEB APPLICATION

Once the model is trained, it can be integrated into a Flask web application. The Flask application can be built to accept input in the form of CT scan images of a patient's lungs. The input images can be processed and fed into the deep learning model for Pulmonary Fibrosis prediction. The model's output can then be displayed on the web application for medical professionals for review and make a diagnosis.

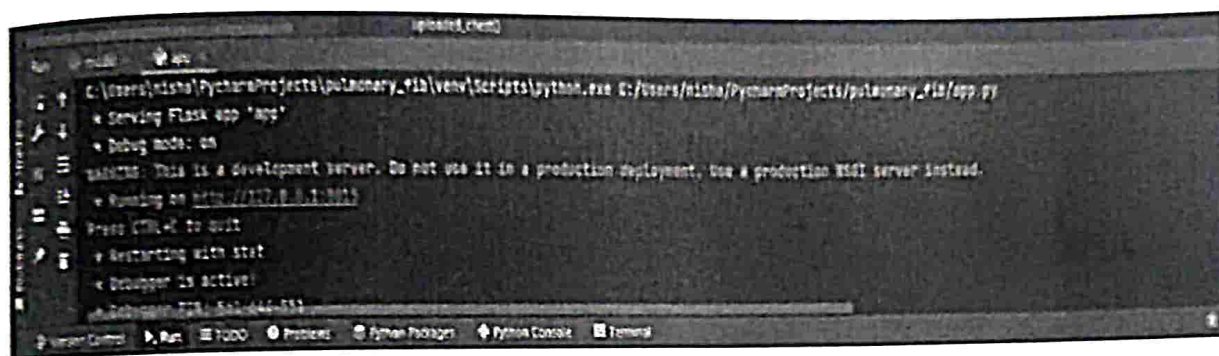


Fig 5.9. Link of Flask Web Application

The input images can be processed and fed into the deep learning model for Pulmonary Fibrosis prediction. The model's output can then be displayed on the web application for medical professionals for review and make a diagnosis.



Fig 5.10. Flask Web Application

Additionally, the Flask application can be designed to include other features, such as data visualization of the patient's CT scan images. This can make the application more comprehensive and user-friendly for medical professionals.

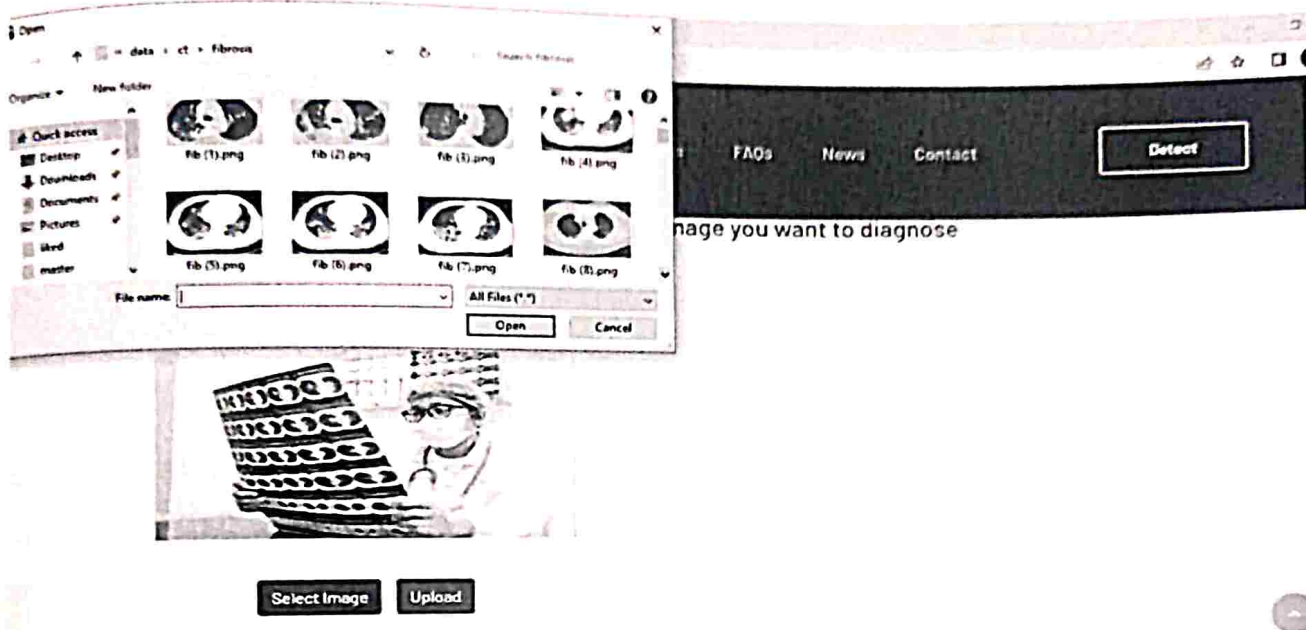


Fig 5.11. Select the Image that has to be diagnosed

Firstly, In the flask application page, the application displays either select the image or upload the image option. When clicking the select image option, dialog box appears displaying the images. Search for the appropriate image and then select the image.

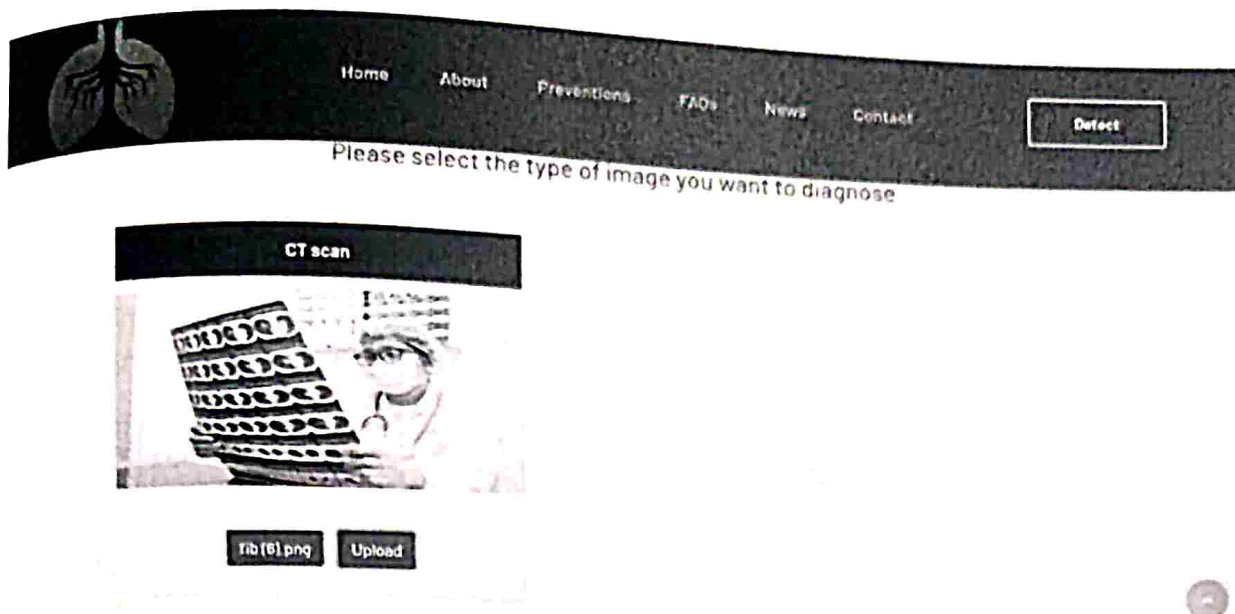


Fig 5.12. Upload the Image that has to be diagnosed

Secondly, after selecting the image click on the upload option to upload the image and make sure that the image is under the maximum allowed size. The image uploaded should be only in JPEG format.



Fig 5.13. Output of Non – fibrosis detected image

Third step, Once the image being uploaded it displays the output. The output will display either a fibrosis or a non- fibrosis image depending upon the image that has been uploaded. For instance, if the nature of the image uploaded is not affected by lung disease it will display the output as No Fibrosis image.

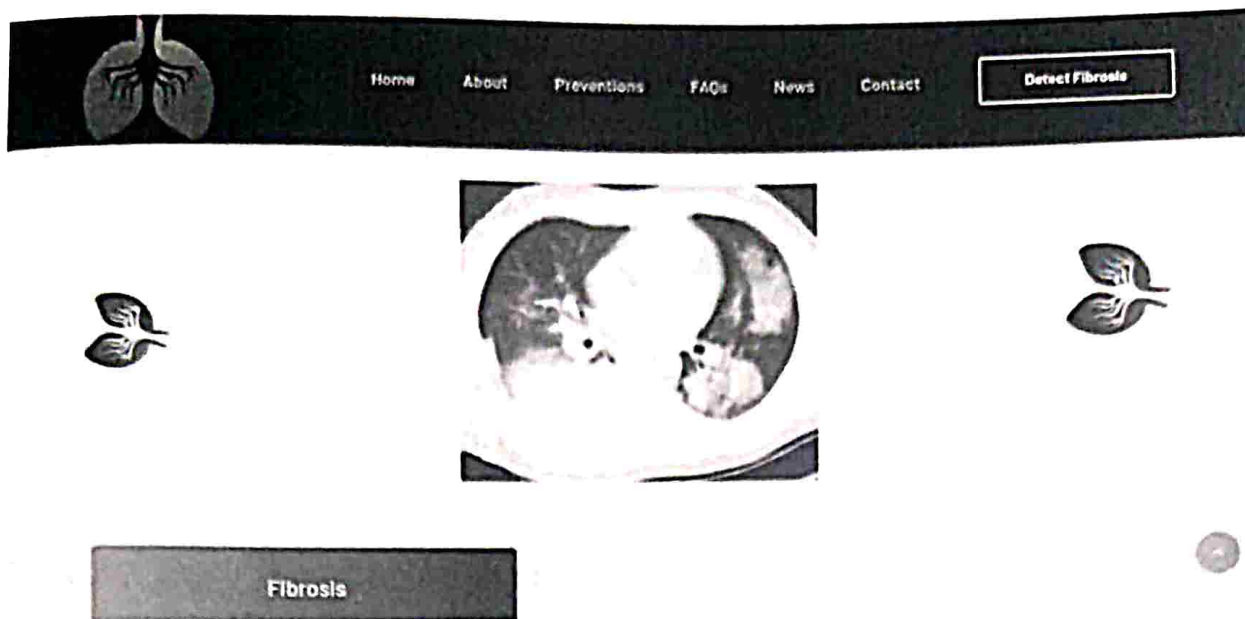


Fig 5.14. Output of Fibrosis detected image

If the nature of the image uploaded is affected by lung disease it will display the output as Fibrosis image.

5.4 DISCUSSION

This study showed that Deep learning with VGG19 can identify predictive CT patterns associated with radiological disease progression in PF. These patterns predict how patients might appear in the future. A set of possible indicators was created by identifying frequently occurring visual patterns in a patient test. Proposals related to the evolution of radiological disease were chosen after training a computer to recognize the temporal sequence of scan pairings based on these attributes. They generate an indicator of radiological disease progression. Even though the clustering was blinded for survival data, it produced clusters with noticeably different outcomes when patients were grouped simply based on how similar these signatures were. On an external validation cohort gathered at a separate location in a different culture, same result proved repeatable. This result was replicable using data from an external validation cohort gathered at a different site in a different nation. Finally, exploratory analysis of lung imaging pattern transition pathways revealed routes from healthy to sick tissue that may serve as the basis for theories about the underlying disease mechanisms based on the imaging data.

The ability of deep learning algorithms to analyse various data types and find similarities is one of their greatest assets. It would make sense that the inclusion of non-radiological data from the laboratory and pulmonary function evaluations, clinical examination results, and demographic characteristics might further strengthen our capacity to make predictions about outcomes in patients with interstitial lung disease, such as response to particular medications and mortality, which could significantly improve patient management. Over the past few years, a wide range of medical concerns have been addressed using machine learning approaches; imaging applications seem especially prominent given the diversity and complexity of imaging data.

Typically, Deep learning algorithms suggest ways to automatically find and measure well-known features in images using CNN. The viability of quantitative image analysis, pattern recognition, and machine-aided detection in IPF/ILD has been demonstrated in related recent work (Wu. X et al.,2019).

In this study, we identify lung pattern types associated with radiological disease progression in addition to those known and named such as ground glass lesion. This is significant since established patterns have a limited ability to accurately diagnose lung disorders that cause fibrosis (Jonigk. D et al.,2019). As a result, using data-driven methods to increase our understanding of disease markers may help imaging become more diagnostic and predictive. Our findings demonstrate that it is possible to expand the marker vocabulary in IPF to include additional patterns that are consistently related with the course of the disease.

Finally, we can visualize shared component transition paths that correspond to regularly occurring lung tissue changes during the evolution of the radiological disease. This may make it possible to convert imaging features chosen by Deep learning models into physiologically significant predictions about the underlying radiological disease process. The analysis shows that by using follow-up exams, these transition networks can be identified in patients.

The feature allows for the evaluation of how lung tissue changes from one pattern to another over time throughout the evolution of the radiological disease. This provides

transition pathways to marker patterns, which may help us understand the underlying responsible for validating and their mark in imaging.

This research has several number of limitations, even though our results from the Deep learning algorithms demonstrate comparable performance to the expert radiologists. Compared to other Deep learning results, our cohort is small. Also, the imaging-based diagnosis was the benchmark for PF, which might have biased the model. The replication cohort's imaging results were heterogeneous since they included both contrast-enhanced and unenhanced images.

CHAPTER – 6

SUMMARY AND CONCLUSION

6.1 SUMMARY

In summary, integrating the deep learning model for Pulmonary Fibrosis prediction into a Flask web application can provide a convenient and efficient tool for medical professionals in diagnosing the disease. Interstitial lung disease (ILD) is a rare group of pulmonary parenchymal disorders caused by environmental insults, systemic diseases, and idiopathic conditions. It has an incidence rate of 0.2-93.7 per 100,000 people annually and radio-histopathological findings suggest progressive scarring of lung tissues. Pulmonary fibrosis is a chronic lung condition that damages the tissues of the lungs by scarring, thickening, and hardening them.

To detect it, medical professionals have used various techniques and detection methods. CT Scan is the simplest method, but requires a radiologist to detect the disease and requires an extended time. A system to detect pulmonary fibrosis automatically is needed to help doctors detect CT images of lungs infected with the disease. Automated lung detection from computed tomography (CT) images has the potential to significantly improve the current healthcare approach to tackling covid-19. CT images provide higher resolution and deep learning approach for classification of pulmonary fibrosis. The application can be customized to include additional features and can improve the accuracy and speed of diagnosis. Pass all these regions (images) to the CNN and classify them into various classes.

This study proposed a deep learning model using VGG19 transfer learning for the image classification process of lungs infected with Pulmonary Fibrosis. Chest imaging plays a significant role in this process as it has proven to be effective in detecting the disease within the lungs. The proposed model can save valuable time for medical practitioners and improve the accuracy and speed of diagnosis.

6.2 CONCLUSION

In conclusion, detecting Pulmonary Fibrosis is a crucial task for medical professionals, but current methods such as CT scans can be time-consuming and require a radiologist to detect the disease. To overcome this problem, implementing a system to automatically detect Pulmonary Fibrosis using deep learning can be a valuable alternative. In this study, we proposed a deep learning model using VGG19 transfer learning for the image classification process. The model was trained using CT scan images of lungs infected with Pulmonary Fibrosis. Chest imaging plays a significant role in this process as it has proven to be effective in detecting the disease within the lungs. Automating the detection of Pulmonary Fibrosis can save valuable time for medical practitioners and improve the accuracy and speed of diagnosis. Our proposed model can be a promising tool for medical professionals in detecting Pulmonary Fibrosis and improving patient outcomes.

REFERENCES

- [1] Afshar.X, Heidarian.S, Enshaei.N.F ,Naderkhani,R Moezedin. J, Oikonomou.A, Fard.F, Samimi.K, Plataniotis, and Mohammadi. A, "Pulmonary fibrosis computed tomography (ct) scan dataset applicable in machine learning and deep learning," 2020.
- [2] Agarwala.S, Kale.M, Kumar.D, Swaroop.R, Kumar.R, Kumar Dhara A, Basu Thakur.S, Sadhu.A, Nandi.D, "Deep learning for screening of interstitial lung disease patterns in high-resolution CT images " *Clinical Radiology*,Vol.75, Issue 6, 2020, PP.1-481.
- [3] Da Li, Zhuo Liu, Lin Luo, Siyu Tian, and Jingyuan Zhao," Prediction of Pulmonary Fibrosis Based on X-Rays by Deep Neural Network", *Journal of Healthcare Engineering*, 2022, pp.1-13, *Journal of Healthcare Engineering*, 2022, pp.113. <https://doi.org/10.1155/2022/3845008>
- [4] Dong. E, H. Du, and Gardner. L, "An interactive web based dashboard to track Pulmonary fibrosis in real time," *The Lancet infectious diseases*, vol. 20, no. 5 ,2020, pp. 533–534.
- [5] Fan D.-P, Zhou. T, Ji G.-P, Zhou. Y, Chen. G, Fu. H, Shen. J and Shao. L, "Inf-Net: Automatic COVID-19 Lung Infection Segmentation from CT Scans," *IEEE Transactions on Medical Imaging*, vol. 39, no. 8, 2020, pp. 2626–2637.
- [6] Gerard SE, Herrmann J, Kaczka DW, Musch G, Fernandez-Bustamante A, and Reinhardt JM, "Multi-resolution convolutional neural networks for fully automated segmentation of acutely injured lungs in multiple species," *Medical Image Analysis*, vol. 60, 2020, pp.101592.
- [7] Glassberg.M.K., "Overview of idiopathic pulmonary fibrosis evidence-based guidelines and recent developments in the treatment landscape," *American Journal of Managed Care*, vol. 25, 2019, pp.195-203.
- [8] Grillet. F, Behr. J, Calame. P, Aubry.S and Delabrousse, E, "Acute pulmonary embolism associated with Pulmonary fibrosis pneumonia detected with pulmonary ct angiography," *Radiological Society of North America*, vol.296, 2020.

- [9] Huang .C, Han .H, Yao.Q, Zhu.S, and Zhou.S.K, "3D U2-Net: A 3D universal U-Net for multi-domain medical image segmentation," In Medical Image Computing and Computer-Assisted Intervention, vol. 2, 2019, pp. 291-299.
- [10] Imran A, Hatamizadeh A, Ananth SP, Ding X, Tajbakhsh N, and Terzopoulos D, "Fast and automatic segmentation of pulmonary lobes from chest CT using a progressive dense V-network," Computer Methods in Biomechanics Biomedical Engineering: Imaging and Visualization, pp. 1–10, 2019.
- [11] Jayaraj, D. and Sathiamoorthy, S., "Random Forest based Classification Model for Lung Cancer Prediction on Computer Tomography Images". International Conference on Smart Systems and Inventive Technology (ICSSIT), 2019, (pp.100-104).
- [12] Jonigk D, Stark H, Braubach P et al., "Morphological and molecular motifs of fibrosing pulmonary injury patterns". Journal of Pathology Clinical Research, 2019, pp.256–271.
- [13] Kropski.J.A and Blackwell. T.S, "Progress in Understanding and Treating Idiopathic Pulmonary Fibrosis," Annual Review of Medicine, ISSN: 0066-4219, vol. 70, no. 1, 2019, pp. 211–224.
- [14] Roy.S et al., "Deep Learning for Classification and Localization of PULMONARY FIBROSIS Markers in Point-of-Care Lung Ultrasound", IEEE Transactions on Medical Imaging, vol. 39, 2020, pp .2676-2687.
- [15] Rustam.F et al., "PULMONARY FIBROSIS Future Forecasting Using Supervised Machine Learning Models", IEEE Access, vol. 8,2020, pp. 101489-101499.
- [16] Subrato Bharati, Prajoy Podder, M. Rubaiyat Hossain Mondal," Hybrid deep learning for detecting lung diseases from X-ray images", Informatics in Medicine Unlocked, Vol.20, 2020, pp.100-391.
- [17] Tajbakhsh.N, Jeyaseelan.L, Li.Q, Chiang.J,Wu .Z, and Ding .X,"Embracing imperfect datasets: A review of deep learning solutions for medical image segmentation," Medical Image Analysis, vol. 63, 2019, pp.101-693.

- [18] Wang. G, Liu. X, Li. C, Xu. Z, Ruan J, Zhu. H, Meng. T, Li. K, Huang. N, and Zhang. S, "A noise-robust framework for automatic segmentation of COVID-19 pneumonia lesions from CT images," *IEEE Transactions on Medical Imaging*, vol. 39, no. 8, 2020, pp.2653–2663.
- [19] Wu X, Kim GH, Salisbury ML et al," Computed tomographic biomarkers in idiopathic pulmonary fibrosis", *American journal of Respiratory and Critical Care Medicine*, vol. 199, issue 1, 2019, pp. 12 – 21.
- [20] Xie. W, Jacobs. C, Charbonnier J.-P, and van Ginneken B, "Relational modeling for robust and efficient pulmonary lobe segmentation in CT scans," *IEEE Transactions on Medical Imaging*, vol. 39, no. 8, 2020, pp. 2664–2675.

APPENDIX

```
from builtins import range, input
```

```
from tensorflow.keras.layers import Input, Lambda, Dense, Flatten,  
GlobalAveragePooling2D, Dropout
```

```
from tensorflow.keras.models import Model, load_model
```

```
from tensorflow.keras.applications import VGG19
```

```
from tensorflow.keras.applications.vgg16 import preprocess_input
```

```
from tensorflow.keras.preprocessing import image
```

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

```
from sklearn.metrics import confusion_matrix, roc_curve
```

```
import seaborn as sns
```

```
import numpy as np
```

```
import matplotlib.pyplot as plt
```

```
from glob import glob
```

```
import pandas as pd
```

```
import cv2
```

```
from sklearn.model_selection import train_test_split
```

```
from sklearn.preprocessing import LabelBinarizer
```

```
from tensorflow.keras.utils import to_categorical
```

```
#define size to which images are to be resized
```

```

IMAGE_SIZE = [224, 224]

# training config:

epochs = 2

batch_size = 32


#define paths

fib_path = 'data/ct/CT_fib'
nonfib_path = 'data/ct/CT_Nonfib'


# Use glob to grab images from path .jpg or jpeg
fib_files = glob(fib_path + '/*')
nonfib_files = glob(nonfib_path + '/*')

# Visualize file variable contents
print("First 5 fib Files: ",fib_files[0:5])
print("Total Count: ",len(fib_files))
print("First 5 Nonfib Files: ",nonfib_files[0:5])
print("Total Count: ",len(nonfib_files))

# Fetch Images and Class Labels from Files
fib_labels = []
nonfib_labels = []


fib_images=[]
nonfib_images=[]

```

```

for i in range(len(fib_files)):
    image = cv2.imread(fib_files[i]) # read file
    image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB) # arrange format as per keras
    image = cv2.resize(image, (224, 224)) # resize as per model
    fib_images.append(image) # append image
    fib_labels.append('CT_fib') # append class label
for i in range(len(nonfib_files)):
    image = cv2.imread(nonfib_files[i])
    image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    image = cv2.resize(image, (224, 224))
    nonfib_images.append(image)
    nonfib_labels.append('CT_Nonfib')
# look at a random image for fun
def plot_images(images, title):
    nrows, ncols = 5, 8
    figsize = [10, 6]

    fig, ax = plt.subplots(nrows=nrows, ncols=ncols, figsize=figsize, facecolor=(1, 1, 1))

    for i, axi in enumerate(ax.flat):
        axi.imshow(images[i])
        axi.set_axis_off()
    plt.suptitle(title, fontsize=24)

```

```

plt.tight_layout(pad=0.2, rect=[0, 0, 1, 0.9])
plt.show()

plot_images(fib_images, 'Positive fib-19 CT Scan')
plot_images(nonfib_images, 'Negative fib-19 CT Scan')
# Convert to array and Normalize to interval of [0,1]
fib_images = np.array(fib_images) / 255
nonfib_images = np.array(nonfib_images) / 255
# Split into training and testing sets for both types of images
fib_x_train, fib_x_test, fib_y_train, fib_y_test = train_test_split(
    fib_images, fib_labels, test_size=0.2)
nonfib_x_train, nonfib_x_test, nonfib_y_train, nonfib_y_test = train_test_split(
    nonfib_images, nonfib_labels, test_size=0.2)

# Merge sets for both types of images
X_train = np.concatenate((nonfib_x_train, fib_x_train), axis=0)
X_test = np.concatenate((nonfib_x_test, fib_x_test), axis=0)
y_train = np.concatenate((nonfib_y_train, fib_y_train), axis=0)
y_test = np.concatenate((nonfib_y_test, fib_y_test), axis=0)

# Make labels into categories - either 0 or 1, for our model
y_train = LabelBinarizer().fit_transform(y_train)
y_train = to_categorical(y_train)
y_test = LabelBinarizer().fit_transform(y_test)

```

```

y_test = to_categorical(y_test)
plot_images(fib_x_train, 'X_train')
plot_images(fib_x_test, 'X_test')

# y_train and y_test contain class labels 0 and 1 representing fib and Nonfib for
X_train and X_test

# Building Model

vggModel = VGG19(weights="imagenet", include_top=False,
    input_tensor=Input(shape=(224, 224, 3)))

outputs = vggModel.output
outputs = Flatten(name="flatten")(outputs)
outputs = Dropout(0.5)(outputs)
outputs = Dense(2, activation="softmax")(outputs)

model = Model(inputs=vggModel.input, outputs=outputs)

for layer in vggModel.layers:
    layer.trainable = False

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

```

```

train_aug = ImageDataGenerator(
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    horizontal_flip=True
)

# Visualize Model
model.summary()

train_aug = ImageDataGenerator(
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    horizontal_flip=True
)

history = model.fit(train_aug.flow(X_train, y_train, batch_size=batch_size),
                    validation_data=(X_test, y_test),
                    validation_steps=len(X_test) / batch_size,
                    steps_per_epoch=len(X_train) / batch_size,
                    epochs=epochs)

# Save Model and Weights
model.save('vgg_ct.h5')
model.save_weights('vgg_weights_ct.hdf5')

# Load saved model

```

```

model = load_model('vgg_ct.h5')
y_pred = model.predict(X_test, batch_size=batch_size)
prediction=y_pred[0:10]
for index, probability in enumerate(prediction):
    if probability[1] > 0.5:
        plt.title('%0.2f % (probability[1]*100) + '% fib')
    else:
        plt.title('%0.2f % ((1-probability[1])*100) + '% Nonfib')
plt.imshow(X_test[index])
# Convert to Binary classes
y_pred_bin = np.argmax(y_pred, axis=1)
y_test_bin = np.argmax(y_test, axis=1)
fpr, tpr, thresholds = roc_curve(y_test_bin, y_pred_bin)
plt.plot(fpr, tpr)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.0])
plt.rcParams['font.size'] = 12
plt.title('ROC curve for our model')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.grid(True)

```

```
def plot_confusion_matrix(normalize):  
    classes = ['fib', 'Nonfib']  
    tick_marks = [0.5, 1.5]  
    cn = confusion_matrix(y_test_bin, y_pred_bin, normalize=normalize)  
    sns.heatmap(cn, cmap='plasma', annot=True)  
    plt.xticks(tick_marks, classes)  
    plt.yticks(tick_marks, classes)  
    plt.title('Confusion Matrix')  
    plt.ylabel('True label')  
    plt.xlabel('Predicted label')  
    plt.show()
```

```
print('Confusion Matrix without Normalization')
```

```
plot_confusion_matrix(normalize=None)
```

```
print('Confusion Matrix with Normalized Values')
```

```
plot_confusion_matrix(normalize='true')
```

```
from sklearn.metrics import classification_report
```

```
print(classification_report(y_test_bin, y_pred_bin))
```

```
plt.plot(history.history['accuracy'])
```

```
plt.plot(history.history['val_accuracy'])
```

```
plt.title('Model Accuracy')
```

```
plt.ylabel('Accuracy')
```

```
plt.xlabel('Epoch')
```

```
plt.legend(['Training', 'Testing'])
```

```
plt.savefig('vgg_ct_accuracy.png')
```

```
plt.show()
```

```
plt.plot(history.history['loss'])
```

```
plt.plot(history.history['val_loss'])
```

```
plt.title('Model Loss')
```

```
plt.ylabel('Loss')
```

```
plt.xlabel('Epoch')
```

```
plt.legend(['Training', 'Testing'])
```

```
plt.savefig('vgg_ct_loss.png')
```

```
plt.show()
```