EFFICIENT PNEUMONIA DETECTION IN CHEST XRAY IMAGES USING DEEP TRANSFER LEARNING

A PROJECT REPORT

submitted by

ASWATHY M S
LMES18MCA038

to

the APJ Abdul Kalam Technological University
in partial fulfillment of the requirements for the award of the Degree

of

Master of Computer Applications

Department of Computer Applications

MES College of Engineering
Kuttippuram, Malappuram - 679 582

June 2021

DECLARATION

I undersigned hereby declare that the project report **EFFICIENT PNEUMONIA DETECTION IN CHEST XRAY IMAGES USING DEEP TRANSFER LEARNING**, submitted for partial fulfillment of the requirements for the award of degree of Master of Computer Applications of the APJ Abdul Kalam Technological University , Kerala , is a bona fide work done by me under supervision of **Mr. Muhammad Jabir C,** Assistant Professor, Department of Computer Applications. This submission represents my ideas in my own words and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and/or the university and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other university.

Place:Kuttippuram ASWATHY M S

Date:30/06/2021 LMES18MCA038

DEPARTMENT OF COMPUTER APPLICATIONS
MES COLLEGE OF ENGINEERING, KUTTIPPURAM

CERTIFICATE

This is to certify that the report entitled **Efficient Pneumonia Detection in Chest Xray Images Using Deep Transfer Learning** submitted by **ASWATHY M S (LMES18MCA038)** to the APJ Abdul Kalam Technological University in partial fulfillment of the requirements for the award of the Master of Computer Applications is a bona fide record of the project work carried out by her under my guidance and supervision. This report in any form has not been submitted to any other university or institution for any purpose.

Internal Supervisor(s)	External Supervisor(s)
Project Coordinator	Head of the Department

Acknowledgement

My endeavor stands incomplete without dedicating my gratitude to a few people who have contributed towards the successful completion of my main project. I pay my gratitude to the Almighty for his invisible help and blessing for the fullfillment of this work. At the outset I express my heart full thanks to **Prof. Hyderali K**, Head of the department, for permitting me to Present this Main project. I take this opportunity to express my profound gratitude to **Mrs. Priya J D,** our Group Tutor, for her valuable support and help in doing this project. I express my sincere gratitude to my Faculty Guide **Mr. Muhammad Jabir C,** for his valuable supervision and guidance. I would like to express my thanks to **Mr. K P Balachandran**, project coordinator for his efforts in conducting my reviews in time properly coordinating my Project. I am also grateful to all our teaching and non teaching staff members for their encouragement, guidance and whole hearted support. Last but not least, I am gratefully indebted to my family and friends, who gave me a precious help in presenting my main project.

Sincerely,

ASWATHY M S

LMES18MCA038

Abstract

Pneumonia is one of the largest infectious diseases that cause death in children and elderly people across the globe. Pneumonia impacts all the elderly and young people's families and children everywhere. Chest X-rays are primarily used for the diagnosis of this disease. However, even for a trained radiologist, it is a challenging task to examine chest X-rays. There is a need to improve the diagnosis accuracy. In this work, an efficient model for the detection of pneumonia trained on digital chest X-ray images is proposed, which could aid the radiologists in their decision-making process. The final proposed weighted classifier model will be able to achieve a test accuracy of 80%+ on the unseen data from pneumonia dataset. Hence, the proposed model can be used for a quick diagnosis of pneumonia and can aid the radiologists in the diagnosis process.

Contents

Declaration **i**

Certificate **ii**

Acknowledgements **iii**

Abstract **iv**

Contents **v**

List of Figures **vii**

List of Tables **ix**

1 Introduction **1**
- 1.1 Background 1
  - 1.1.1 Motivation 1
- 1.2 Objective 1
- 1.3 Contribution 2
- 1.4 Report Organization 2

2 Literature Survey **3**

3 Methodology **5**
- 3.1 Introduction 5
- 3.2 Workflow 6
  - 3.2.1 Dataset 8
  - 3.2.2 Customized models 9
- 3.3 Implementation 11

4 Agile Methodology **12**
- 4.1 Introduction 12
- 4.2 User Story 14
- 4.3 Product Backlog 14
- 4.4 Project Plan 15

4.5 Sprint Backlog (Plan) 17
4.6 Sprint Backlog (Actual) 22
4.7 Product Backlog Review 27
4.8 Sprint Review 29
4.9 Testing and Validation 31
4.10 Git 32

5 Result and Discussions **33**
5.1 Dataset 33
5.2 Results 34
5.2.1 Import Data 34
5.2.2 Customized model 34
5.2.3 Data Augmentation 35
5.2.4 Fine-tuning 35
5.2.5 ResNet152V2 model training 36
5.2.6 Detection of Pneumonia 37

6 Conclusion **39**

References **40**

Appendix **42**
Source Code 42
Output 50
Git History 54

List of Figures

3.1 Architectural Diagram 7

3.2 Dataset 8

3.3 ResNet152V2 Architecture 10

3.4 Model Comparison 11

5.1 Dataset 33

5.2 import data 34

5.3 Customization 35

5.4 Augmentation 35

5.5 Fine-tuning 36

5.6 ResNet152V2 model training 37

5.7 Pneumonia detection 38

A. 1 User Interface 50

A. 2 File Select 50

A. 3 Pneumonia Detected 51

A. 4 Pneumonia not Detected 51

A. 5 Pneumonia Detected 52

A. 6 Pneumonia not Detected 52

A. 7 Loss over Epochs 53

A. 8 Validation over Epochs 53

A. 9 Git Commits 1 54

A. 10Git Commits 2 54

A. 11 Git Commits 3 55
A. 12 Git Commits 4 55
A. 13 Git Graph 1 56
A. 14 Git Graph 2 56
A. 15 Git Graph 3 57

List of Tables

4.1 User Story 14

4.2 Product Backlog 15

4.3 Project plan 16

4.4 Sprint Backlog (Plan) - Sprint 1 18

4.5 Sprint Backlog (Plan) - Sprint 2 19

4.6 Sprint Backlog (Plan) - Sprint 3 20

4.7 Sprint Backlog (Plan) - Sprint 4 21

4.8 Sprint Backlog (Actual) - Sprint 1 23

4.9 Sprint Backlog (Actual) - Sprint 2 24

4.10 Sprint Backlog (Actual) - Sprint 3 25

4.11 Sprint Backlog (Actual) - Sprint 4 26

4.12 Product Backlog Review (Sprint1) 27

4.13 Product Backlog Review (Sprint2) 27

4.14 Product Backlog Review (Sprint3) 28

4.15 Product Backlog Review (Sprint4) 28

4.16 Sprint Review (Sprint1) 29

4.17 Sprint Review (Sprint2) 30

4.18 Sprint Review (Sprint3) 30

4.19 Sprint Review (Sprint4) 30

4.20 Testing and Validation(Sprint1) 31

4.21 Testing and Validation(Sprint2) 31

4.22 Testing and Validation(Sprint3) 32

4.23 Testing and Validation(Sprint4) 32

Chapter 1

Introduction

1.1 Background

Pneumonia causes the death of around 700,000 children every year and affects 7% of the global population. Chest X-rays are primarily used for the diagnosis of this disease. However, even for a trained radiologist, it is a challenging task to examine chest X-rays. There is a need to improve the diagnosis accuracy. In this work, an efficient model for the detection of pneumonia trained on digital chest X-ray images is proposed, which could aid the radiologists in their decision-making process.

1.1.1 Motivation

The main motivation behind this research was to identify Pneumonia just by using the X-Ray images of the patients. As doctors must do a lot of certain tests to identify if the patient has Pneumonia or not. To solve the cumbersome problem, an ensemble of two deep learning models is developed, to make the work of the doctors simpler.

1.2 Objective

A novel approach based on a weighted classifier will be introduced, which will combine the weighted predictions from the state-of-the-art deep learning models such as ResNet152V2, Xception, InceptionV3, DenseNet121, and MobileNetV3 in an optimal way. This approach is a supervised learning approach in which the network predicts the result based on the quality of the dataset used.

1.3 Contribution

The major contributions in this project are:

1. Designed and developed a new system for detecting pneumonia from chest Xray.
2. How to minimise human intervention in detecting pneumonia.
3. How to effectively identify pneumonia in a timely manner.

1.4 Report Organization

The project report is divided into six sections. Section 2 describes literature survey. Section 3 describes the methodology and section 4 describes agile methodology used for implementing the project. Section 5 gives the results and discussions. Finally Section 6 gives the conclusion.

Chapter 2

Literature Survey

The risk of pneumonia is immense for many, especially in developing nations where billions face energy poverty and rely on polluting forms of energy. The WHO estimates that over 4 million premature deaths occur annually from household air pollution-related diseases including pneumonia. Over 150 million people get infected with pneumonia on an annual basis especially children under 5years old. In such regions, the problem can be further aggravated due to the dearth of medical resources and personnel. For example, in Africa's 57 nations, a gap of 2.3 million doctors and nurses exists. For these populations, accurate and fast diagnosis means everything. It can guarantee timely access to treatment and save much needed time and money for those already experiencing poverty.

Over the last decade, several machine learning based automated methods for identifying different types of pneumonia have been widely studied. Fiszman et al used a natural language processing (NLP) tool to identify acute bacterial pneumonia-related disease in chest X-ray. Performance of this type of resource intensive application is very much comparable to that of the human expert.

Chapman et al demonstrated three computerized methods using a rule base, a probabilistic Bayesian network, and a decision tree to diagnose the chest X-ray report associated with acute bacterial pneumonia. A study of feasibility of an NLP-based monitoring system is done to identify healthcare-associated pneumonia in neonates. However, practical clinical applications of these types of methods are limited due to the dependency on the information extracted from the narrative reports of the patients.

Parveen et al reports an unsupervised fuzzy c-means classification learning algorithm to detect pneumonia infected X-ray images. This approach improves classification accuracy as fuzzy c-means allocate weights to all the pixels of the input X-ray images.

Rajpurkar et al demonstrated ChexNet, a 121-layer deep convolutional neural network (CNN), that provides the probability of detecting or identifying pneumonia using a heatmap to localize the area of the infection.

Kermany et al introduced a transfer learning based DL framework to diagnose paediatric pneumonia using chest X-ray images. However, none of the methods are exploited to classify X-ray images with pneumonia for the CS framework to meet the need of remote end analysis.

Researchers are utilizing the results of machine learning predictions for solving problems of life science. Large volume of information can be extracted and used for future prevention of dangerous diseases. For extracting information from the medical images Python language is used.

High dimensional data consists of the medical images which has sizable amount of feature descriptors. To extract the numerous feature descriptors, feature extraction techniques are applied on good quality X-ray images. Deep learning neural networks are trained with the extracted data to create the prediction model. Research is additionally carried for prediction of pneumonia using machine learning classifiers.

Chapter 3

Methodology

3.1 Introduction

Deep learning is an artificial intelligence (AI) function that imitates the workings of the human brain in processing data and creating patterns for use in decision making. Deep learning is a subset of machine learning in artificial intelligence that has networks capable of learning unsupervised from data that is unstructured or unlabeled. Also known as deep neural learning or deep neural network. Transfer learning is a research problem in Deep learning (DL) that focuses on storing knowledge gained while training one model and applying it to another model. Convolutional neural networks are composed of multiple layers of artificial neurons. Artificial neurons, a rough imitation of their biological counterparts, are mathematical functions that calculate the weighted sum of multiple inputs and outputs an activation value.

Python used as the coding language for this project.It is best suited for this project,because,In python there is a large ecosystem of pre-built libraries for scientific computation. Libraries like NumPy, SciPy, and Pandas make doing scientific calculations easy and quick,In this project I used an open source web application called jupiter notebook that we can use to create and share documents that contain live code, equations, visualizations, and text.Jupyter Notebook is maintained by the people at Project Jupyter.Jupyter Notebooks are a spin-off project from the IPython project, which used to have an IPython Notebook project itself. The name, Jupyter, comes from the core supported programming languages that it supports: Julia, Python, and R. Jupyter ships with the IPython kernel, which allows us to write our programs in Python, but there are currently over 100 other kernels that we can also use.Jupyter Notebook was born out of IPython in 2014. It is a web application based on the server-client structure, and it allows us to create and manipulate notebook documents - or just “notebooks”.Jupyter Notebook provides us with an easy-to-use, interactive data science environment across many programming languages that doesn’t only work as an IDE, but also as a presentation or education tool. It’s perfect for those who are just starting out with data science.We can easily see and edit your code in order to create compelling presentations. For instance, we can use data visualization libraries like Matplotlib and Seaborn and show your graphs in the same document where your code is. Besides all of this, we can export our final work to PDF and HTML files, or we can just export it as a .py file. In addition, we can also create blogs and presentations from our notebooks.

3.2 Workflow

In this project, we will focus on Convolutional Neural Network (CNN), which is a class of deep neural network, most commonly applied to analyze visual imagery. We will use Keras in our project because Keras is an open-source software library that provides a Python interface for artificial neural networks. The implementation of our model will have 4 steps:

1.Data Collection

2.Data Augmentation

3.Development of transfer learning models

4.Train models

5.Evaluation

6.Choode the best model

7.UI Design and model Integration

8.Generate predicted results

The Architectural diagram of the system is given in Figure 3.1

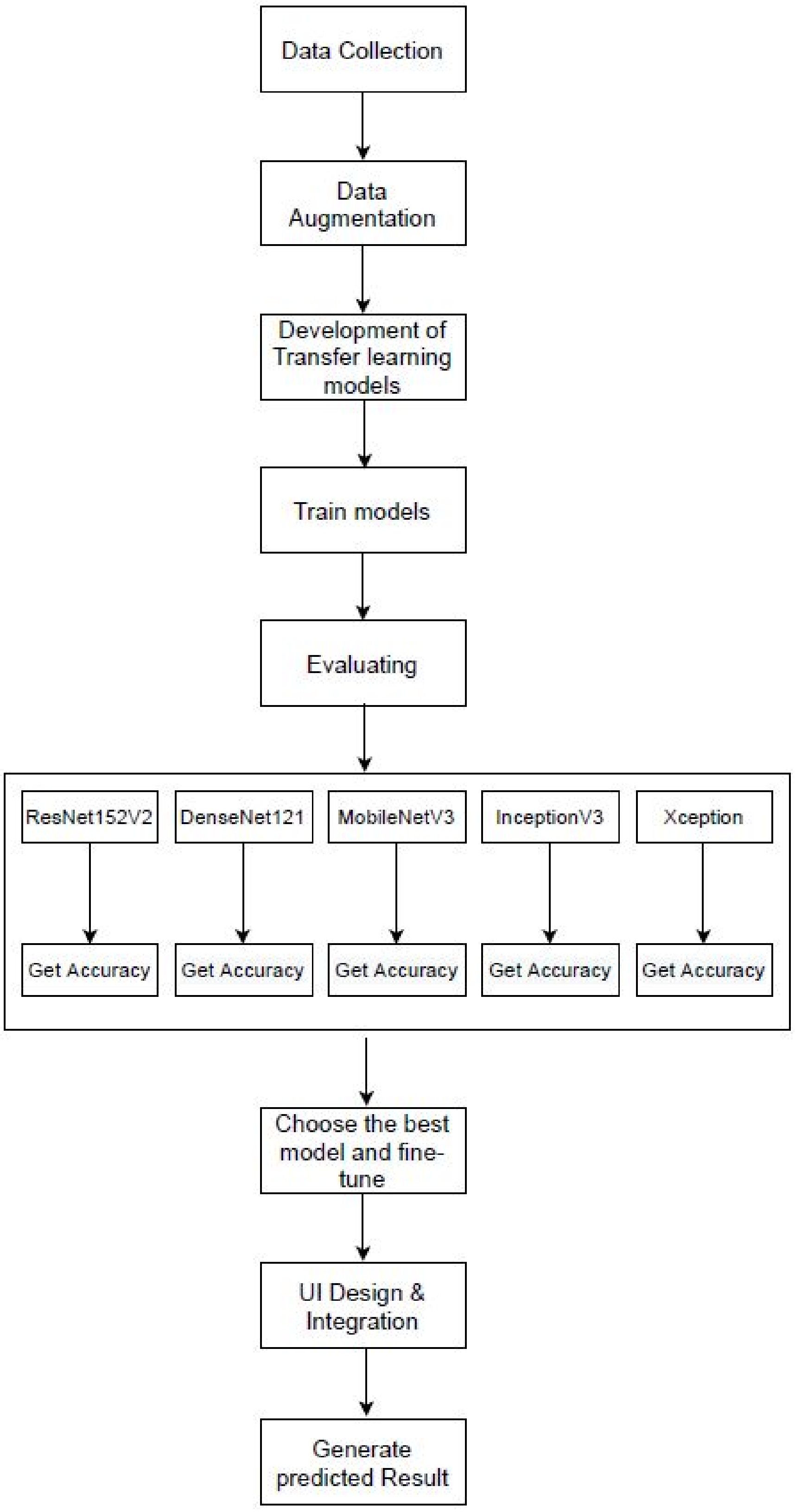


Figure 3.1: Architectural Diagram

3.2.1 Dataset

We have taken data set for the building of the model from the Kaggle. Kaggle, a subsidiary of Google LLC, is an online community of data scientists and machine learning practitioners. Kaggle allows users to find and publish datasets, explore and build models in a web-based data-science environment, work with other data scientists and machine learning engineers, and enter competitions to solve data science challenges.

The dataset contains pneumonia effected chest xray images and normal chest xray images.This dataset contains around 1200 images for training and around 360 images for validation in each case. Only 20% - 40% of training data will be used for validation.

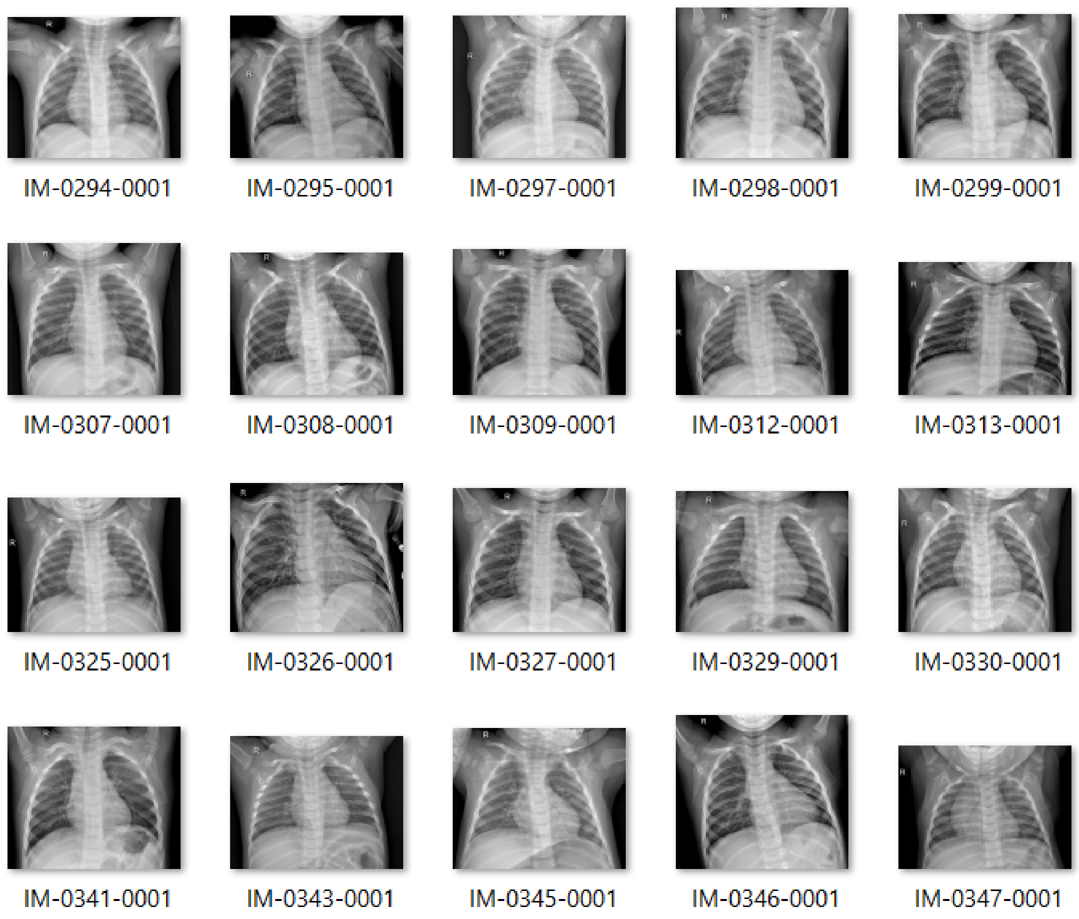


Figure 3.2: Dataset

3.2.2 Customized models

InceptionV3

Inceptionv3 is a convolutional neural network architecture from the Inception family that makes several improvements including using Label Smoothing, Factorized 7 x 7 convolutions, and the use of an auxiliary classifer to propagate label information lower down the network (along with the use of batch normalization for layers in the sidehead).Inception v3 TPU training runs match accuracy curves produced by GPU jobs of similar configuration. The model has been successfully trained on v2-8, v2-128, and v2-512 configurations. The model has attained greater than 78.1% accuracy in about 170 epochs on each of these.

Xception

Xception is a deep convolutional neural network architecture that involves Depthwise Separable Convolutions. It was developed by Google researchers. Xception is a convolutional neural network that is 71 layers deep. You can load a pretrained version of the network trained on more than a million images from the ImageNet database.

MobileNetV3

MobileNetV3 is the third version of the architecture powering the image analysis capabilities of many popular mobile applications. The architecture has also been incorporated in popular frameworks such as TensorFlow Lite. MobileNets need to carefully balance the advancements in computer vision and deep learning in general with the limitations of mobile environments. Google has regularly been releasing updates to the MobileNets architecture which incorporate some of the most novel ideas in the deep learning space.

DenseNet121

DenseNet is a convolutional neural network where each layer is connected to all other layers that are deeper in the network, that is, the first layer is connected to the 2nd, 3rd, 4th and so on, the second layer is connected to the 3rd, 4th, 5th and so on.

ResNet152V2

Residual Network (ResNet) is a CNN architecture with hundreds or thousands of convolutional layers. Previous CNN structures decreased the efficacy of additional layers. ResNet contains a huge number of layers, with strong performance. The primary difference between ResNetV2 and the original (V1) is that V2 uses batch normalization before each weight layer. In the field of image recognition and localization tasks, ResNet has strong performance that demonstrates the importance of many visual recognition tasks.

ResNet152V2 is used as a feature extraction model,instead of the VGG19-CNN model. The model has initial weights because it is a pre-trained model, which can help to gain acceptable accuracy faster than a traditional CNN. The model architecture consists of the ResNet152V2 model followed by a reshape layer, a flatten layer, a dense layer with 128 neurons, a dropout layer, and finally a dense layer with Softmax activation function to classify the image into its corresponding class.

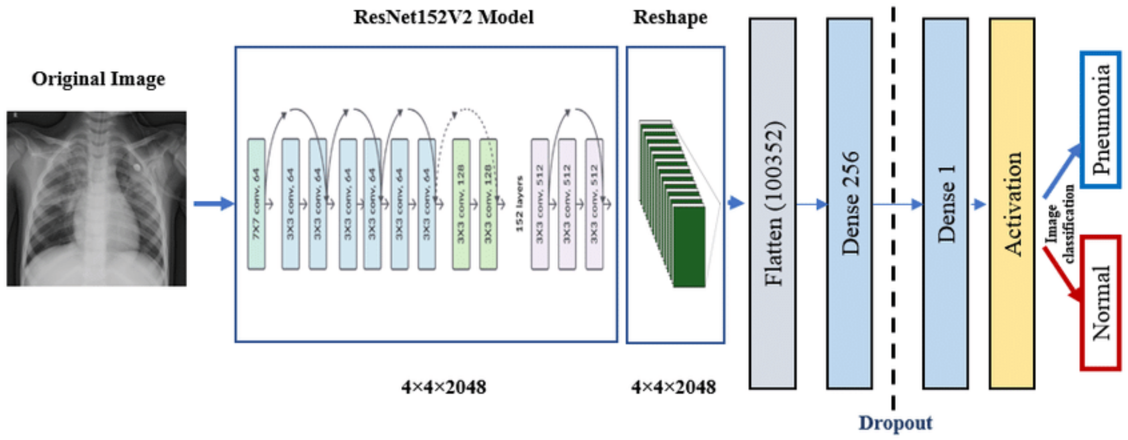
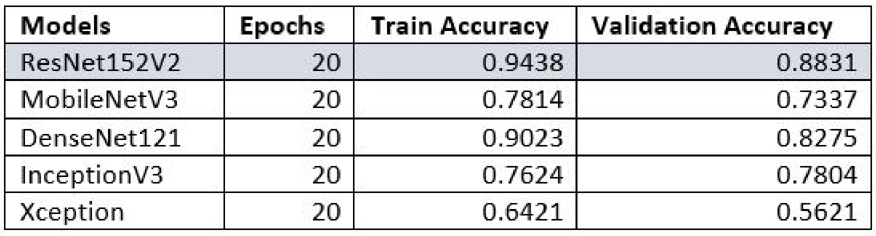


Figure 3.3: ResNet152V2 Architecture

ResNet152V2 has many advantages as a pre-trained model in accelerating training and converging to high accuracy rapidly. Therefore, adding it before deep learning models can build efficient and reliable models, which results in higher accuracy.

Deep learning is an artificial intelligence (AI) function that imitates the workings of the human brain in processing data and creating patterns for use in decision making. Deep learning is a subset of machine learning in artificial intelligence that has networks capable of learning unsupervised from data that is unstructured or unlabeled. Also known as deep neural learning or deep neural network.



Models	Epochs	Train Accuracy	Validation Accuracy
ResNet152V2	20	0.9438	0.8831
MobileNetV3	20	0.7814	0.7337
DenseNet121	20	0.9023	0.8275
InceptionV3	20	0.7624	0.7804
Xception	20	0.6421	0.5621

Figure 3.4: Model Comparison

3.3 Implementation

In this project, one of the main step is importing the data set. In this data set, pneumonia effected and normal chest Xray images are the data. After importing data set to the project,next data augmentation is done. After that the processes such as customization of model and training the model with the data are done. Once a model is trained, I can get to know the training and validation accuracy of the model. Also calculate the training and validation loss of the model. After building the models , compare the models and choose the model from that which have the highest accuracy. That model will be fine-tuned and saved. Design and develop the web user interface using the web framework Flask. The saved model will be integrated with the User Interface. In ResNet152V2 to detect pneumonia , pass the image through convolution layer . Then it will be process maxpooling and then flatten by using ReLu activation method. The output will be 0 or 1 as it is binary classification. It specifies that 1 for Pneumonia detected and 0 for Pneumonia not detected.

Chapter 4

Agile Methodology

4.1 Introduction

After the initial studies it is found that agile model of software development is suitable and is the best method for the development of this system. Agile methodology mainly focused on the client satisfaction through continuous delivery. Also it sets a minimum number of requirements and turns them in to a deliverable product. As this project has many individual requirements which can be delivered in parts and the user can gradually improve their work efficiency. Agile methodology has a family of methods of which scrum is selected for the development of this project. Scrum is process framework that has been used to manage complex product development. It is not a process or technique for building products rather it is a framework within which various processes can be employed. Also it is suitable method to support the development process. It focuses on lean software development and has in building better software effectively and efficiently.

Agile is one of the most widely used and recognized software development framework. The methodology those experts agreed upon was described as 'lightweight' and fast. Agile is also about being the adaptive and continuous improvement, as much as it is about constant feedback and speed of delivery.

Agile is a software development approach where a self-sufficient and cross-functional team works on making continuous deliveries through iterations and evolves throughout the process

by gathering feedback from the end users.
The major rules in scrum methodology are

1. **The product owner (PO)** : Who represents the stakeholder and the business.
2. **The scrum master** : Ensures the process followed, removes obstructions, and protects the development system
3. **Development team**: Cross functional, self organizing team who actually do the actual analysis, design implementation and testing process.

They work together in iterative time boxed durations called sprints. The first step is the creation of the product backlog by the PO. It's a to-do list of stuff to be done by the scrum team. Then the scrum team selects the top priority items and tries to finish them within the time box called a sprint. An easier way to remember all of this is to memorize the 3-3-5 framework. It means that a scrum project has 3 roles, 3 artifacts, and 5 events

These are:-

1. **Roles** : Product Owner, Scrum Master, and development team.
2. **Artifacts** : Product Backlog, Sprint Backlog and Product Increment.
3. **Events** : Sprint, Sprint planning, Daily Scrum, Sprint review and Sprint retrospective

The framework begins with a simple premise start with what can be seen or known. After that the progress is tracked and tweak as necessary. The three pillars of scrum are transparency, inspection and adaptation. In scrum everyone has a role.

The Git is used as the version control system for this project. Version control is a system that records changes to a file or set of files over time so that a specific versions can be recalled later. Version control systems are a category of software tools that help a software team for managing changes to source code over time. Version control software keeps track of every modification to the code in a special kind of database. If a mistake is made, developers can turn back the clock and compare earlier versions of the code to help fix the mistake while minimizing disruption to all team members.

4.2 User Story

A user story is a tool used in agile software development to capture a description of a software feature from an end-user perspective.The user story describes the type of user,what they want and why.A user story helps to create a simplified description of a requirement.

User story ID	As a <Type of user>	I want to perform <some task>	So that I can <achieve some goal>
1	User	Upload chest Xray Image	Check Pneumonia

Table 4.1: User Story

4.3 Product Backlog

A product backlog is a list of the new features, changes to existing features, bug fixes, infrastructure changes or other activities that a team may deliver in order to achieve a specific outcome.The product backlog is the single authoritative source for things that a team works on. That means that nothing gets done that isn't on the product backlog. Conversely, the presence of a product backlog item on a product backlog does not guarantee that it will be delivered. It represents an option the team has for delivering a specific outcome rather than a commitment.

It should be cheap and fast to add a product backlog item to the product backlog, and it should be equally as easy to remove a product backlog item that does not result in direct progress to achieving the desired outcome or enable progress toward the outcome. The Scrum Product Backlog is simply a list of all things that needs to be done within the project. It replaces the traditional requirements specification artifacts. These items can have a technical nature or can be user-centric e.g. in the form of user stories.The product backlog of the system is given in Table 4.2

PRODUCT BACKLOG			
ID	NAME	PRIORITY	ESTIMATE[Hrs]
1	Image processing and Data augmentation	1	60
1	Design and Development of DL architectures	2	70
1	Perfomance Evaluation of model	3	50
1	Development of web UI and Integration	4	60

Table 4.2: Product Backlog

4.4 Project Plan

A project plan that has a series of tasks laid out for the entire project, listing task durations, responsibility assignments, and dependencies. Plans are developed in this manner based on the assumption that the Project Manager, hopefully along with the team, can predict up front everything that will need to happen in the project, how long it will take, and who will be able to do it. Project plan is given in Table 4.3

User story ID	Task Name	Start date	End date	Days	Status (to be filled by Scrum Master)
Sprint 1		31-03-2021	13-04-2021	14	Completed
1	Data collection	31-03-2021	05-04-2021	6	Completed
1	Coding	06-04-2021	11/04/2021	6	Completed
1	Testing	12-04-2021	13-04-2021	2	Completed
Sprint 2		16-04-2021	29-04-2021	14	Completed
1	Coding	16-04-2021	27-04-2021	12	Completed
1	Testing	28-04-2021	29-04-2021	2	Completed
Sprint 3		10-05-2021	23-05-2021	14	Completed
1	Form Design	10-05-2021	13-05-2021	4	Completed
1	Integration of model	14-05-2021	20-05-2021	7	Completed
1	Testing	21-05-2021	23-05-2021	3	Completed
Sprint 4		01-06-2021	14-06-2021	14	Complete
1	Deployment	01-06-2021	07-06-2021	7	Completed
1	Testing and Validation	07-06-2021	14-06-2021	7	Completed

Table 4.3: Project plan

The Project has Four sprints:

1. **Sprint 1**

   Three tasks are planned in this sprint. First one is Problem definition , next is Data collection and Image preprocessing .

2. **Sprint 2**

   Three tasks are planned in this sprint. First one is design and development of Transfer Learning models and next one is Training model with dataset and evaluation of the model.

3. **Sprint 3**

   Two tasks are planned in this sprint. These are design and development of web user interface using flask and Integration of recommended deep learning model with web app.

4. **Sprint 4**

   In this sprint two tasks are planned to complete,one is Deployment of web app on cloud and second is testing and result discussion.

4.5 Sprint Backlog (Plan)

The sprint backlog is a list of tasks identified by the Scrum team to be completed during the Scrum sprint. During the sprint planning meeting, the team selects some number of product backlog items, usually in the form of user stories, and identifies the tasks necessary to complete each user story. Most teams also estimate how many hours each task will take someone on the team to complete.

Sprint 1:

Three tasks are planned in this sprint. First one is Problem definition , next is Data collection and Image preprocessing .Sprint backlog (planning) for sprint 1 is given in Table 4.4.

Sprint 2:

Three tasks are planned in this sprint. First one is design and development of Transfer Learning models and next one is Training model with dataset and evaluation of the model. Sprint backlog (planing) for sprint 2 is given in Table 4.5

Sprint 3:

Two tasks are planned in this sprint. These are design and development of web user interface using flask and Integration of recommended deep learning model with web app.The sprint backlog for sprint 3 is given in Table 4.6

Sprint 4:

In this sprint two tasks are planned to complete,one is Deployment of web app on cloud and second is testing and result discussion.The sprint backlog for sprint 4 is given in Table 4.7

Backlog item	Completion date	Original estimate in hours	Day 1 31/03/21	Day 2 01/04/21	Day 3 02/04/21	Day 4 03/04/21	Day 5 04/04/21	Day 6 05/04/21	Day 7 06/04/21	Day 8 07/04/21	Day 09 08/04/21	Day 10 09/04/21	Day 11 10/04/21	Day 12 11/04/21	Day 13 12/04/21	Day 14 13/04/21
User Story #1			Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours
Data collection	05/04/21	11	2	2	2	2	2	1	0	0	0	0	0	0	0	0
Coding	05/04/21	39	0	0	0	0	0	0	7	6	6	8	7	5	0	0
Testing	13/04/21	10	0	0	0	0	0	0	0	0	0	0	0	0	6	4
Total		60	2	2	2	2	2	1	7	6	6	8	7	5	6	4

Table 4.4: Sprint Backlog (Plan) - Sprint 1

Backlog item	Completion date	Original estimate in hours	Day 1 16/04/21	Day 2 17/04/21	Day 3 18/04/21	Day 4 19/04/210	Day 5 20/04/21	Day 6 21/04/21	Day 7 22/04/21	Day 8 23/04/21	Day 09 24/04/21	Day 10 25/04/21	Day 11 26/04/21	Day 12 27/04/21	Day 13 28/04/21	Day 14 29/04/21
User Story #1			Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours
Coding	27/04/21	55	7	6	6	7	5	6	6	7	5	0	0	0	0	0
Testing	29/04/21	15	0	0	0	0	0	0	0	0	0	2	3	3	4	3
Total		70	7	6	6	7	5	6	6	7	5	2	3	3	4	3

Table 4.5: Sprint Backlog (Plan) - Sprint 2

Backlog item	Completion date	Original estimate in hours	Day 1 10/05/21	Day 2 11/05/21	Day 3 12/05/21	Day 4 13/05/21	Day 5 14/05/21	Day 6 15/05/21	Day 7 16/05/21	Day 8 17/05/21	Day 09 18/05/21	Day 10 19/05/21	Day 11 20/05/21	Day 12 21/05/21	Day 13 22/05/21	Day 14 23/05/21
User Story #1			Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours
Form Design	13/05/21	15	4	2	3	1	3	2	0	0	0	0	0	0	0	0
Integration of model	20/05/21	25	0	0	0	0	0	0	6	5	4	2	4	5	0	0
Testing	23/05/21	10	0	0	0	0	0	0	0	0	0	0	0	0	6	4
Total		50	4	2	3	1	3	2	6	5	4	2	4	5	6	4

Table 4.6: Sprint Backlog (Plan) - Sprint 3

Backlog item	Completion date	Original estimate in hours	Day 1 01/06/21	Day 2 02/06/21	Day 3 03/06/21	Day 4 04/06/210	Day 5 05/06/21	Day 6 06/06/21	Day 7 07/06/21	Day 8 08/06/21	Day 09 09/06/21	Day 10 10/06/21	Day 11 11/06/21	Day 12 12/06/21	Day 13 13/06/21	Day 14 14/06/21
User Story #1			Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours
Deployment	07/06/21	35	7	8	7	6	4	2	3	0	0	0	0	0	0	0
Testing and Validation	14/06/21	25	0	0	0	0	0	0	0	5	6	2	4	4	3	1
Total		60	7	8	7	6	4	2	3	5	6	2	4	4	3	1

Table 4.7: Sprint Backlog (Plan) - Sprint 4

4.6 Sprint Backlog (Actual)

Actual sprint backlog is what adequate sprint planning is actually done by project team there may or may not be difference in planned sprint backlog. The detailed sprint backlog (Actual) is given below.

Backlog item	Completion date	Original estimate in hours	Day 1 31/03/21	Day 2 01/04/21	Day 3 02/04/21	Day 4 03/04/21	Day 5 04/04/21	Day 6 05/04/21	Day 7 06/04/21	Day 8 07/04/21	Day 09 08/04/21	Day 10 09/04/21	Day 11 10/04/21	Day 12 11/04/21	Day 13 12/04/21	Day 14 13/04/21
User Story #1			Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours
Data collection	05/04/21	11	2	2	2	2	2	1	0	0	0	0	0	0	0	0
Coding	05/04/21	39	0	0	0	0	0	0	7	6	6	8	7	5	0	0
Testing	13/04/21	10	0	0	0	0	0	0	0	0	0	0	0	0	6	4
Total		60	2	2	2	2	2	1	7	6	6	8	7	5	6	4

Table 4.8: Sprint Backlog (Actual) - Sprint 1

Backlog item	Completion date	Original estimate in hours	Day 1 16/04/21	Day 2 17/04/21	Day 3 18/04/21	Day 4 19/04/210	Day 5 20/04/21	Day 6 21/04/21	Day 7 22/04/21	Day 8 23/04/21	Day 09 24/04/21	Day 10 25/04/21	Day 11 26/04/21	Day 12 27/04/21	Day 13 28/04/21	Day 14 29/04/21
User Story #1			Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours
Coding	27/04/21	55	7	6	6	7	5	6	6	7	5	0	0	0	0	0
Testing	29/04/21	15	0	0	0	0	0	0	0	0	0	2	3	3	4	3
Total		70	7	6	6	7	5	6	6	7	5	2	3	3	4	3

Table 4.9: Sprint Backlog (Actual) - Sprint 2

Backlog item	Completion date	Original estimate in hours	Day 1 10/05/21	Day 2 11/05/21	Day 3 12/05/21	Day 4 13/05/21	Day 5 14/05/21	Day 6 15/05/21	Day 7 16/05/21	Day 8 17/05/21	Day 09 18/05/21	Day 10 19/05/21	Day 11 20/05/21	Day 12 21/05/21	Day 13 22/05/21	Day 14 23/05/21
User Story #1			Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours
Form Design	13/05/21	15	4	2	3	1	3	2	0	0	0	0	0	0	0	0
Integration of model	20/05/21	25	0	0	0	0	0	0	6	5	4	2	4	5	0	0
Testing	23/05/21	10	0	0	0	0	0	0	0	0	0	0	0	0	6	4
Total		50	4	2	3	1	3	2	6	5	4	2	4	5	6	4

Table 4.10: Sprint Backlog (Actual) - Sprint 3

Backlog item	Completion date	Original estimate in hours	Day 1 01/06/21	Day 2 02/06/21	Day 3 03/06/21	Day 4 04/06/210	Day 5 05/06/21	Day 6 06/06/21	Day 7 07/06/21	Day 8 08/06/21	Day 09 09/06/21	Day 10 10/06/21	Day 11 11/06/21	Day 12 12/06/21	Day 13 13/06/21	Day 14 14/06/21
User Story #1			Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours	Hours
Deployment	07/06/21	35	7	8	7	6	4	2	3	0	0	0	0	0	0	0
Testing and Validation	14/06/21	25	0	0	0	0	0	0	0	5	6	2	4	4	3	1
Total		60	7	8	7	6	4	2	3	5	6	2	4	4	3	1

Table 4.11: Sprint Backlog (Actual) - Sprint 4

4.7 Product Backlog Review

REVIEW FORM

SPRINT 1

Version:1.0 **Date:13/04/2021**

User story ID	Comments from Scrum master if any	Comments from Product Owner if any
1	Check data set thoroughly	Need to collect more data for better analysis
1	Successfully complete fetching of data	Satisfied

Table 4.12: Product Backlog Review (Sprint1)

SPRINT 2

Version:1.0 **Date:29/04/2021**

User story ID	Comments from Scrum master if any	Comments from Product Owner if any
1	Should preform data augmentation	Satisfied
1	Design and development of deep learning architecture	Should customize different models

Table 4.13: Product Backlog Review (Sprint2)

SPRINT 3

Version:1.0 **Date:23/05/2021**

User story ID	Comments from Scrum master if any	Comments from Product Owner if any
1	Perform evaluation of models	Check accuracy of all models
1	Integrating the model	Satisfied

Table 4.14: Product Backlog Review (Sprint3)

SPRINT 4

Version:1.0 **Date:14/06/2021**

User story ID	Comments from Scrum master if any	Comments from Product Owner if any
1	Deployment	Visualize final output.
1	Generate predicted result	Satisfied

Table 4.15: Product Backlog Review (Sprint4)

4.8 Sprint Review

At the end of each sprint a Sprint Review meeting is held. During this meeting the Scrum Team shows which Scrum Product Backlog items they completed (according to the Definition of Done) during the sprint. This might take place in the form of a demo of the new features. Backlog items that are not completed shall not be demonstrated. Otherwise this might suggest that these items are finished as well. Instead incomplete items/remaining activities shall be taken back into the Scrum Product Backlog, re-estimated and completed in one of the following sprints. The Sprint Review meeting should be kept very informal. No PowerPoint slides should be used and time for preparation and performing the meeting should be limited. During the meeting the Scrum Product Owner inspects the implemented backlog entries and accepts the solution or adds new stories to the Scrum Product Backlog to adapt the functionality. Participants in the sprint review typically include the Scrum Product Owner, the Scrum Team and the Scrum Master. Additionally management, customers, and developers from other projects might participate as well.

REVIEW FORM

SPRINT 1

Version:1.0 **Date:13/04/2021**

User story ID	Comments from Scrum master if any	Comments from Product Owner if any
1	Data Collected	Need to collect more data for better analysis
1	Data loaded	Satisfied with result

Table 4.16: Sprint Review (Sprint1)

SPRINT 2

Version:1.0 **Date:30/04/2021**

User story ID	Comments from Scrum master if any	Comments from Product Owner if any
1	Data augmentation completed	Satisfied
1	Model Training completed	Satisfied with result

Table 4.17: Sprint Review (Sprint2)

SPRINT 3

Version:1.0 **Date:24/05/2021**

User story ID	Comments from Scrum master if any	Comments from Product Owner if any
1	Form designing completed	Satisfied
1	Model Integration completed	Satisfied with result
1	Testing Completed	Satisfied

Table 4.18: Sprint Review (Sprint3)

SPRINT 4

Version:1.0 **Date:14/06/2021**

User story ID	Comments from Scrum master if any	Comments from Product Owner if any
7	Deployment completed	Satisfied
8	Output generated	Satisfied with result

Table 4.19: Sprint Review (Sprint4)

4.9 Testing and Validation

Sprint 1

Test #	Date	Action	Expected Result	Actual Result	Pass ? <Yes/ N o>
1	31/03/2021	Data Collection	Collect Large amount of Data	Collected	Yes
1	06/04/2021	Fetch dataset	Load images to program	Loaded	Yes

Table 4.20: Testing and Validation(Sprint1)

Sprint 2

Test #	Date	Action	Expected Result	Actual Result	Pass ? <Yes/ N o>
1	16/04/2021	Development of transfer learning models	Can choose the best models	Done	Yes
1	19/04/2021	Train models	Train the model with data	Done	Yes
1	25/04/2021	Evaluating models	Evaluate training and validation accuracy and loss	Done	Yes

Table 4.21: Testing and Validation(Sprint2)

Sprint 3

Test #	Date	Action	Expected Result	Actual Result	Pass ? <Yes/ N o>
1	10/05/2021	Development of web application	UI will be formed	Done	Yes
1	20/05/2021	Integration of recommended model	Get the actual result	Done	Yes

Table 4.22: Testing and Validation(Sprint3)

Sprint 4

Test #	Date	Action	Expected Result	Actual Result	Pass ? <Yes/ N o>
1	01/06/2021	Deployment	User can input various images for detection	Done	Yes

Table 4.23: Testing and Validation(Sprint4)

4.10 Git

Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency. To show the continuous development of the project the Gitlab histories are shown in Appendix from figure A.12 to A.15

Chapter 5

Result and Discussions

5.1 Dataset

We have taken dataset from Kaggle for building the model.The dataset contain around 1200 images in each cases.This dataset is class balanced.

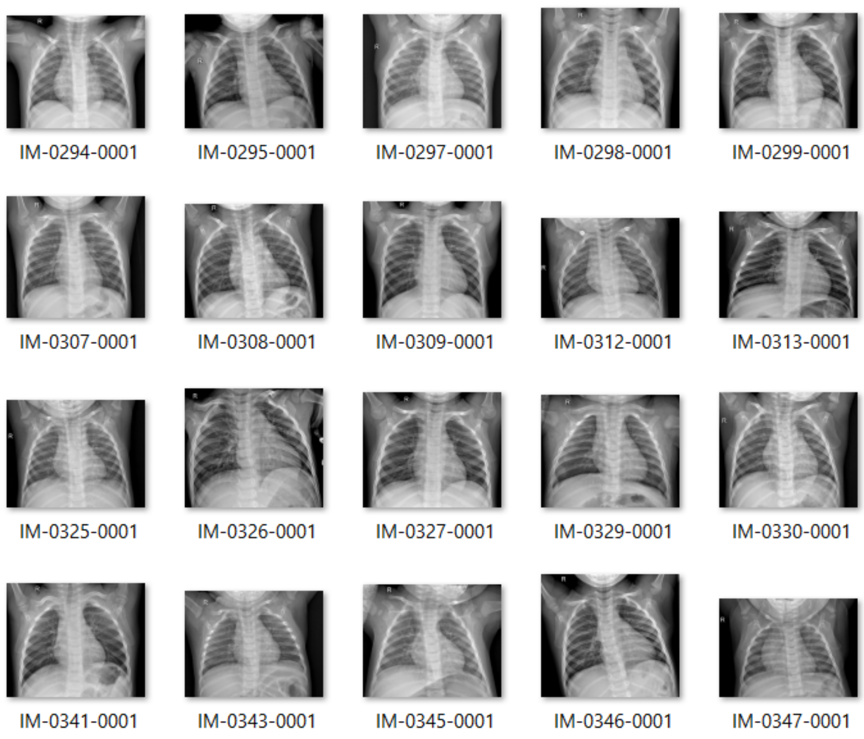


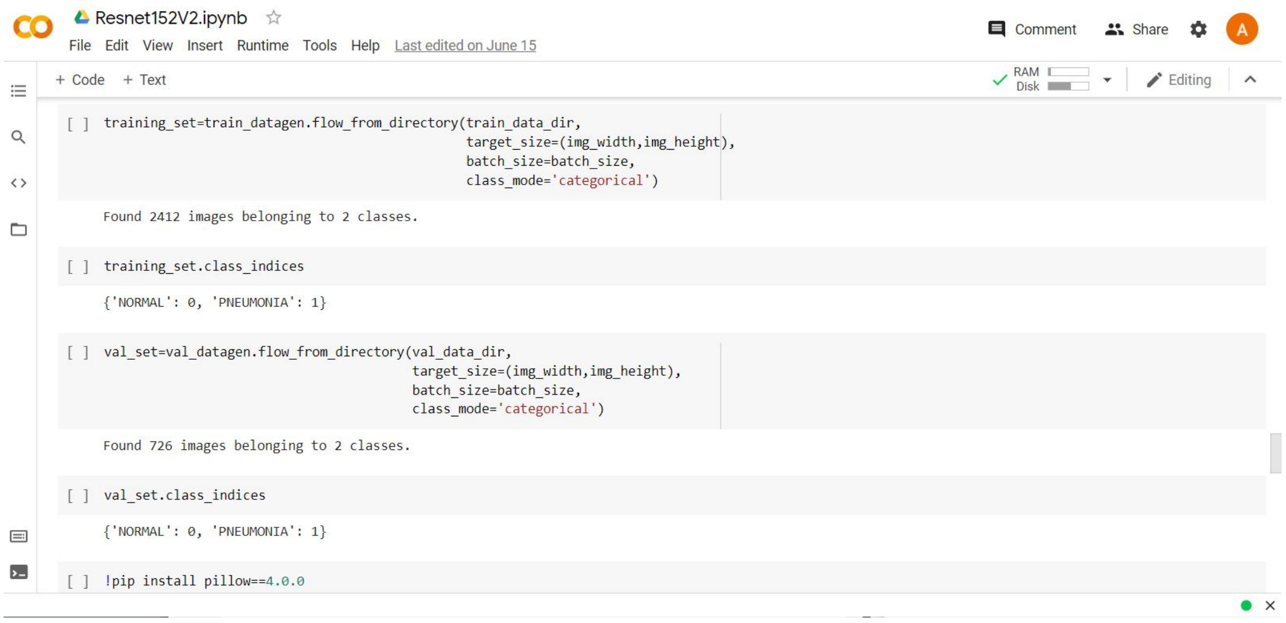
Figure 5.1: Dataset

5.2 Results

This section includes the implementation details of our project.

5.2.1 Import Data

In our project the main one of the step is import the dataset. The data set consists of chest Xray images.

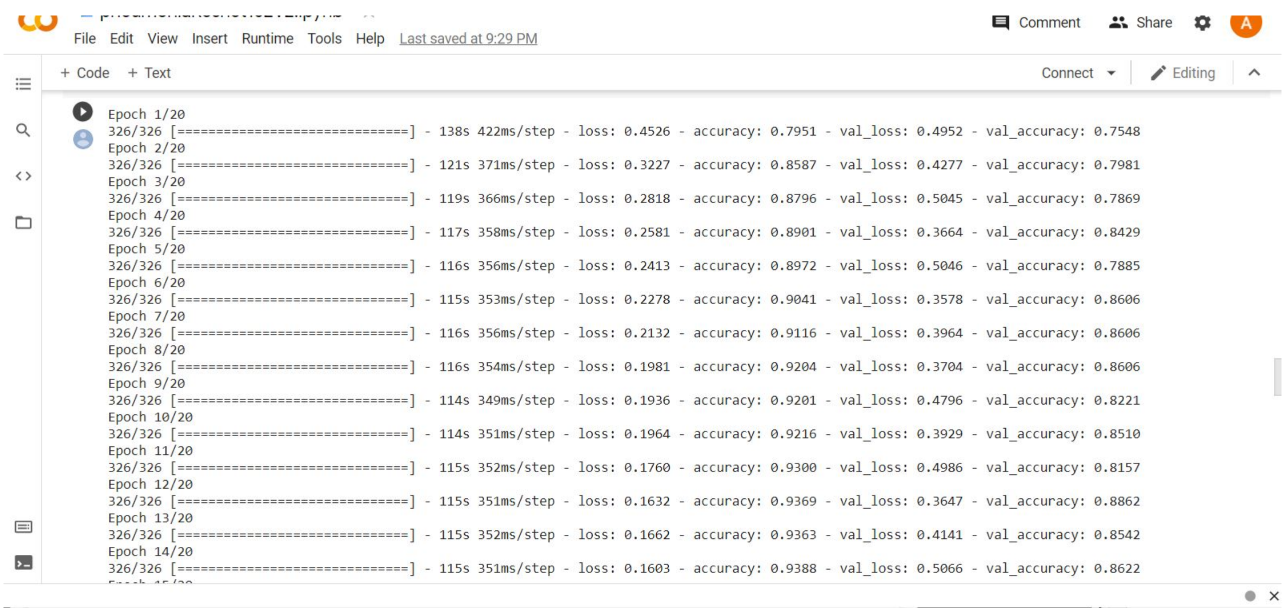

Resnet152V2.ipynb
File Edit View Insert Runtime Tools Help Last edited on June 15
Comment
Share
+ Code + Text
RAM
Disk
Editing

```
[ ] training_set=train_datagen.flow_from_directory(train_data_dir,
                                                 target_size=(img_width,img_height),
                                                 batch_size=batch_size,
                                                 class_mode='categorical')

    Found 2412 images belonging to 2 classes.

[ ] training_set.class_indices

    {'NORMAL': 0, 'PNEUMONIA': 1}

[ ] val_set=val_datagen.flow_from_directory(val_data_dir,
                                         target_size=(img_width,img_height),
                                         batch_size=batch_size,
                                         class_mode='categorical')

    Found 726 images belonging to 2 classes.

[ ] val_set.class_indices

    {'NORMAL': 0, 'PNEUMONIA': 1}

[ ] !pip install pillow==4.0.0
```

Figure 5.2: import data

5.2.2 Customized model

After importing data set to the project,next we are customizing deep learning models.Then we can compare the accuracy between each models.

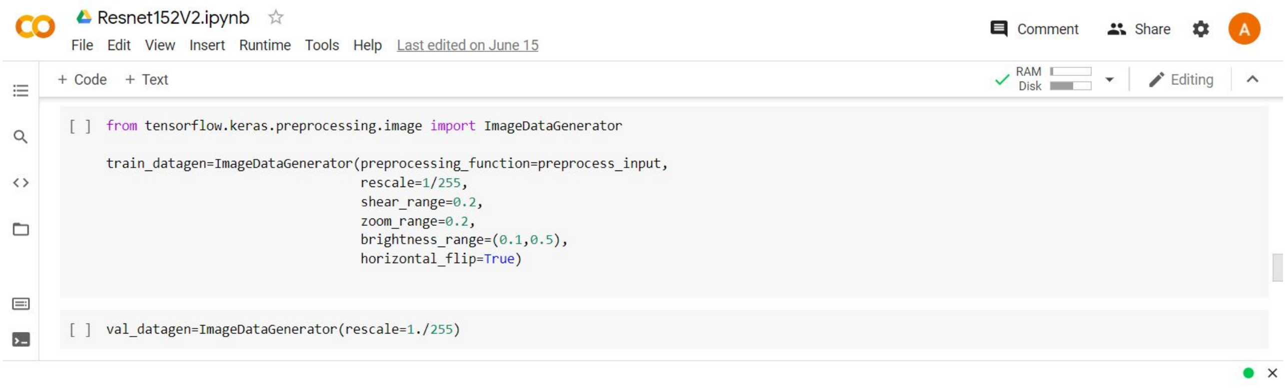

File Edit View Insert Runtime Tools Help Last saved at 9:29 PM
Comment
Share
+ Code + Text
Connect
Editing

```
Epoch 1/20
326/326 [==============================] - 138s 422ms/step - loss: 0.4526 - accuracy: 0.7951 - val_loss: 0.4952 - val_accuracy: 0.7548
Epoch 2/20
326/326 [==============================] - 121s 371ms/step - loss: 0.3227 - accuracy: 0.8587 - val_loss: 0.4277 - val_accuracy: 0.7981
Epoch 3/20
326/326 [==============================] - 119s 366ms/step - loss: 0.2818 - accuracy: 0.8796 - val_loss: 0.5045 - val_accuracy: 0.7869
Epoch 4/20
326/326 [==============================] - 117s 358ms/step - loss: 0.2581 - accuracy: 0.8901 - val_loss: 0.3664 - val_accuracy: 0.8429
Epoch 5/20
326/326 [==============================] - 116s 356ms/step - loss: 0.2413 - accuracy: 0.8972 - val_loss: 0.5046 - val_accuracy: 0.7885
Epoch 6/20
326/326 [==============================] - 115s 353ms/step - loss: 0.2278 - accuracy: 0.9041 - val_loss: 0.3578 - val_accuracy: 0.8606
Epoch 7/20
326/326 [==============================] - 116s 356ms/step - loss: 0.2132 - accuracy: 0.9116 - val_loss: 0.3964 - val_accuracy: 0.8606
Epoch 8/20
326/326 [==============================] - 116s 354ms/step - loss: 0.1981 - accuracy: 0.9204 - val_loss: 0.3704 - val_accuracy: 0.8606
Epoch 9/20
326/326 [==============================] - 114s 349ms/step - loss: 0.1936 - accuracy: 0.9201 - val_loss: 0.4796 - val_accuracy: 0.8221
Epoch 10/20
326/326 [==============================] - 114s 351ms/step - loss: 0.1964 - accuracy: 0.9216 - val_loss: 0.3929 - val_accuracy: 0.8510
Epoch 11/20
326/326 [==============================] - 115s 352ms/step - loss: 0.1760 - accuracy: 0.9300 - val_loss: 0.4986 - val_accuracy: 0.8157
Epoch 12/20
326/326 [==============================] - 115s 351ms/step - loss: 0.1632 - accuracy: 0.9369 - val_loss: 0.3647 - val_accuracy: 0.8862
Epoch 13/20
326/326 [==============================] - 115s 352ms/step - loss: 0.1662 - accuracy: 0.9363 - val_loss: 0.4141 - val_accuracy: 0.8542
Epoch 14/20
326/326 [==============================] - 115s 351ms/step - loss: 0.1603 - accuracy: 0.9388 - val_loss: 0.5066 - val_accuracy: 0.8622
```

Figure 5.3: Customization

5.2.3 Data Augmentation

Data augmentation is used to remove noise from the data.The techniques such as zooming, shearing, rescaling, horizontalflip ,so on are used.

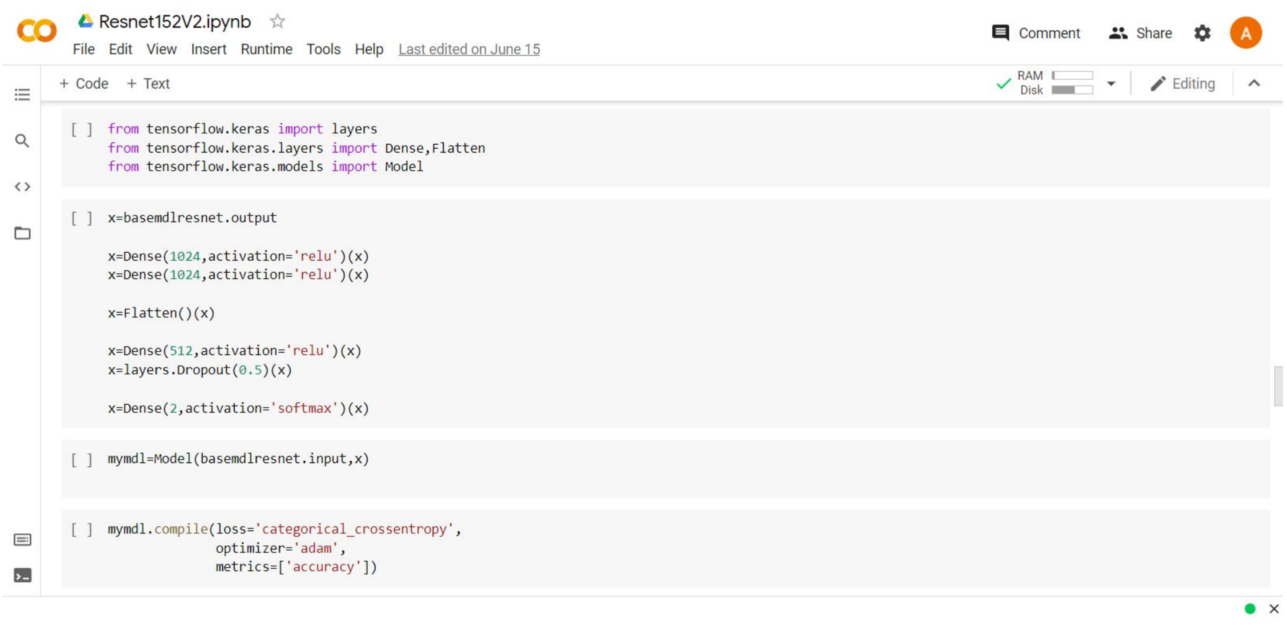

Resnet152V2.ipynb
File Edit View Insert Runtime Tools Help Last edited on June 15
Comment
Share
+ Code + Text
RAM
Disk
Editing

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

    train_datagen=ImageDataGenerator(preprocessing_function=preprocess_input,
                                     rescale=1/255,
                                     shear_range=0.2,
                                     zoom_range=0.2,
                                     brightness_range=(0.1,0.5),
                                     horizontal_flip=True)

[ ] val_datagen=ImageDataGenerator(rescale=1./255)
```

Figure 5.4: Augmentation

5.2.4 Fine-tuning

After Deep learning model training, fine-tuning will be done.


Resnet152V2.ipynb
File Edit View Insert Runtime Tools Help Last edited on June 15
Comment Share
+ Code + Text
RAM Disk
Editing

```
from tensorflow.keras import layers
from tensorflow.keras.layers import Dense,Flatten
from tensorflow.keras.models import Model
```

```
x=basemdlresnet.output

x=Dense(1024,activation='relu')(x)
x=Dense(1024,activation='relu')(x)

x=Flatten()(x)

x=Dense(512,activation='relu')(x)
x=layers.Dropout(0.5)(x)

x=Dense(2,activation='softmax')(x)
```

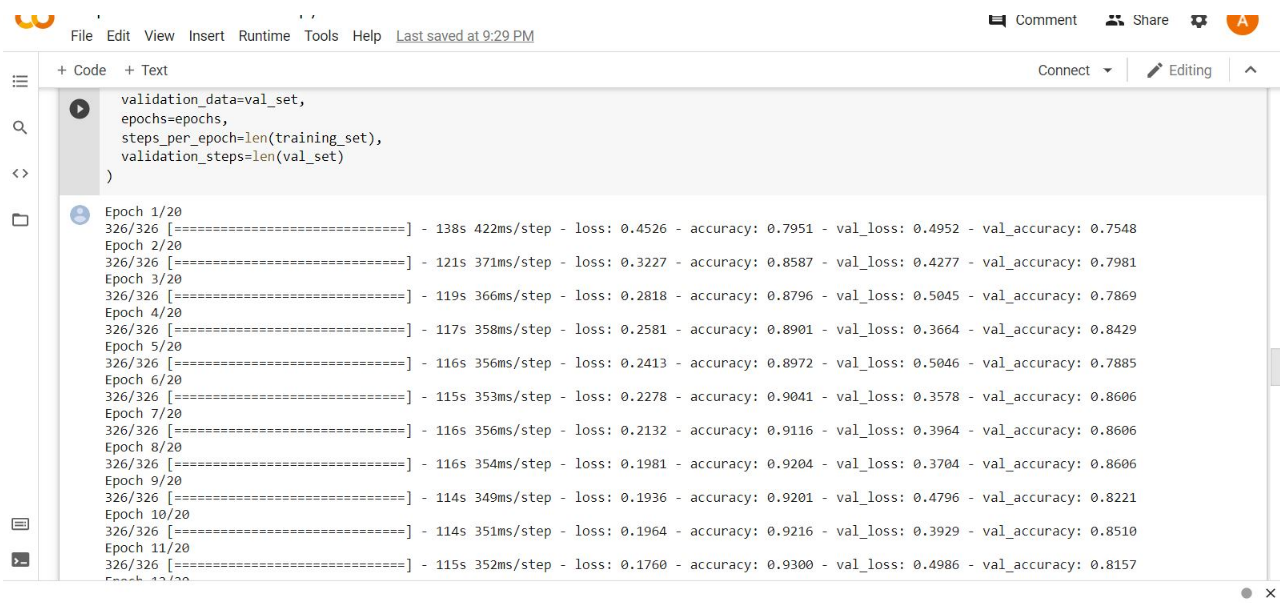
```
mymdl=Model(basemdlresnet.input,x)
```

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

Figure 5.5: Fine-tuning

5.2.5 ResNet152V2 model training

ResNet152V2 model training, For input and target, we are going to use entire set of pneumonia data, the model will be trained on the entire set. This will allow generating detection of pneumonia.



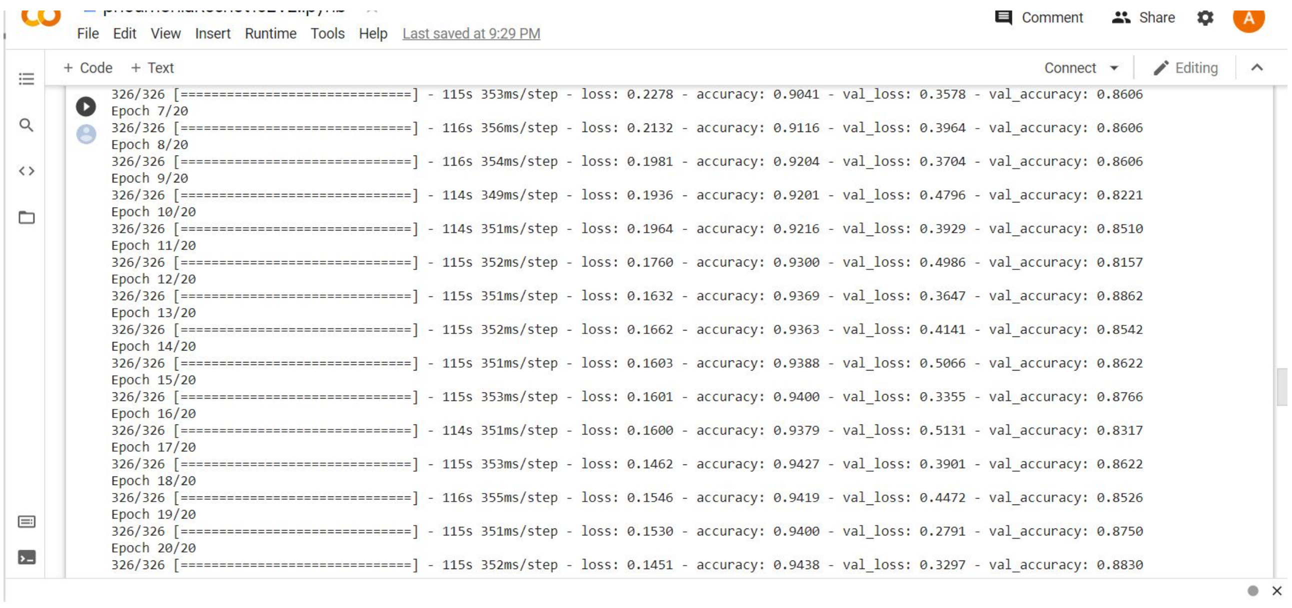
```
    validation_data=val_set,
    epochs=epochs,
    steps_per_epoch=len(training_set),
    validation_steps=len(val_set)
)
```

```
Epoch 1/20
326/326 [==============================] - 138s 422ms/step - loss: 0.4526 - accuracy: 0.7951 - val_loss: 0.4952 - val_accuracy: 0.7548
Epoch 2/20
326/326 [==============================] - 121s 371ms/step - loss: 0.3227 - accuracy: 0.8587 - val_loss: 0.4277 - val_accuracy: 0.7981
Epoch 3/20
326/326 [==============================] - 119s 366ms/step - loss: 0.2818 - accuracy: 0.8796 - val_loss: 0.5045 - val_accuracy: 0.7869
Epoch 4/20
326/326 [==============================] - 117s 358ms/step - loss: 0.2581 - accuracy: 0.8901 - val_loss: 0.3664 - val_accuracy: 0.8429
Epoch 5/20
326/326 [==============================] - 116s 356ms/step - loss: 0.2413 - accuracy: 0.8972 - val_loss: 0.5046 - val_accuracy: 0.7885
Epoch 6/20
326/326 [==============================] - 115s 353ms/step - loss: 0.2278 - accuracy: 0.9041 - val_loss: 0.3578 - val_accuracy: 0.8606
Epoch 7/20
326/326 [==============================] - 116s 356ms/step - loss: 0.2132 - accuracy: 0.9116 - val_loss: 0.3964 - val_accuracy: 0.8606
Epoch 8/20
326/326 [==============================] - 116s 354ms/step - loss: 0.1981 - accuracy: 0.9204 - val_loss: 0.3704 - val_accuracy: 0.8606
Epoch 9/20
326/326 [==============================] - 114s 349ms/step - loss: 0.1936 - accuracy: 0.9201 - val_loss: 0.4796 - val_accuracy: 0.8221
Epoch 10/20
326/326 [==============================] - 114s 351ms/step - loss: 0.1964 - accuracy: 0.9216 - val_loss: 0.3929 - val_accuracy: 0.8510
Epoch 11/20
326/326 [==============================] - 115s 352ms/step - loss: 0.1760 - accuracy: 0.9300 - val_loss: 0.4986 - val_accuracy: 0.8157
```

Figure 5.6: ResNet152V2 model training

5.2.6 Detection of Pneumonia

Once the model is trained, we can check the sample images for pneumonia detection. The idea here — detect for one future step, using the last 8 steps. Add new prediction to the array, remove the first entry from the same array and predict the next step with an updated array of 8 steps. As it is binary classification , it returns only 0 or 1 . 0 for pneumonia not detected and 1 for pneumonia detected.



```
326/326 [==============================] - 115s 353ms/step - loss: 0.2278 - accuracy: 0.9041 - val_loss: 0.3578 - val_accuracy: 0.8606
Epoch 7/20
326/326 [==============================] - 116s 356ms/step - loss: 0.2132 - accuracy: 0.9116 - val_loss: 0.3964 - val_accuracy: 0.8606
Epoch 8/20
326/326 [==============================] - 116s 354ms/step - loss: 0.1981 - accuracy: 0.9204 - val_loss: 0.3704 - val_accuracy: 0.8606
Epoch 9/20
326/326 [==============================] - 114s 349ms/step - loss: 0.1936 - accuracy: 0.9201 - val_loss: 0.4796 - val_accuracy: 0.8221
Epoch 10/20
326/326 [==============================] - 114s 351ms/step - loss: 0.1964 - accuracy: 0.9216 - val_loss: 0.3929 - val_accuracy: 0.8510
Epoch 11/20
326/326 [==============================] - 115s 352ms/step - loss: 0.1760 - accuracy: 0.9300 - val_loss: 0.4986 - val_accuracy: 0.8157
Epoch 12/20
326/326 [==============================] - 115s 351ms/step - loss: 0.1632 - accuracy: 0.9369 - val_loss: 0.3647 - val_accuracy: 0.8862
Epoch 13/20
326/326 [==============================] - 115s 352ms/step - loss: 0.1662 - accuracy: 0.9363 - val_loss: 0.4141 - val_accuracy: 0.8542
Epoch 14/20
326/326 [==============================] - 115s 351ms/step - loss: 0.1603 - accuracy: 0.9388 - val_loss: 0.5066 - val_accuracy: 0.8622
Epoch 15/20
326/326 [==============================] - 115s 353ms/step - loss: 0.1601 - accuracy: 0.9400 - val_loss: 0.3355 - val_accuracy: 0.8766
Epoch 16/20
326/326 [==============================] - 114s 351ms/step - loss: 0.1600 - accuracy: 0.9379 - val_loss: 0.5131 - val_accuracy: 0.8317
Epoch 17/20
326/326 [==============================] - 115s 353ms/step - loss: 0.1462 - accuracy: 0.9427 - val_loss: 0.3901 - val_accuracy: 0.8622
Epoch 18/20
326/326 [==============================] - 116s 355ms/step - loss: 0.1546 - accuracy: 0.9419 - val_loss: 0.4472 - val_accuracy: 0.8526
Epoch 19/20
326/326 [==============================] - 115s 351ms/step - loss: 0.1530 - accuracy: 0.9400 - val_loss: 0.2791 - val_accuracy: 0.8750
Epoch 20/20
326/326 [==============================] - 115s 352ms/step - loss: 0.1451 - accuracy: 0.9438 - val_loss: 0.3297 - val_accuracy: 0.8830
```

Figure 5.7: Pneumonia detection

Chapter 6

Conclusion

While this Pneumonia detector is not a fully deployable product that can change the world, it is clear that it is easy to get started with it. It is great to witness the growth and accuracy of deep learning in such real-world scenarios. This model is robust as it can work on any of the datasets that conform to the size of the image that is required for this model. We can observe that our model has given astounding results for the ” customized DenseNet121” model that has high precision and decent recall, but the other model, ”customized ResNet152V2” had given high recall but low precision.

References

[1] Ranjan, E. Paul, S. Kapoor, S. Kar, Aupendu, Sethuraman, R. Sheet and Debdoot, ”Jointly Learning Convolutional Representations to Compress Radiological Images and Classify Thoracic Diseases in the Compressed Domain,” in Indian Conference on Computer Vision, Graphics and Image Processing (ICVGIP), ACM, Hyderabad, 2018.

[2] B. Antin, J. Kravitz and E. Martayan, ”Detecting Pneumonia in Chest X-Rays with Supervised Learning,” 2017.

[3] O. Stephen, M. Sain, U. J. Maduh and D.-U. Jeong, ”An Efficient Deep Learning Approach to Pneumonia Classification in Healthcare,” Journal of HealthCare Engineering, vol. 2019, 2019.

[4] C. H. Sudre, W. Li, T. Vercauteren, S. Ourselin and M. J. Cardoso, ”Generalised Dice overlap as a deep learning loss function for highly unbalanced segmentations,” in Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support, 2017.

[5] Brox, O. Ronneberger, P. Fischer and Thomas, ”U-Net: Convolutional Networks for Biomedical Image Segmentation,” in International Conference on Medical image computing and computer-assisted intervention, 2015.

[6] M. Tan and Q. V. Le, ”EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks,” ICML, 2019.

[7] F. Chollet, ”Xception: Deep learning with depthwise separable convolutions,” in IEEE conference on computer vision and pattern recognition, 2017.

[8] K. He, X. Zhang, S. Ren and J. Sun, ”Deep Residual Learning for Image Recognition,” 2015.

[9] P. Rajpurkar, J. Irvin, K. Zhu, B. Yang, H. Mehta, T. Duan, D. Ding, A. Bagul, C. Langlotz, K. Shpanskaya, M. P. Lungren and A. Y. Ng, ”CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning,” 2017.

[10] T. Vijayakumar, ”NEURAL NETWORK ANALYSIS FOR TUMOR INVESTIGATION AND CANCER PREDICTION,” Journal of Electronics, pp. 89-98, 2019.

[11] D. T. Vijayakumar and M. R. Vinothkanna, ”Mellowness detection in dragon fruit using deep learning strategy,” Journal of Innovative Image Processing (JIIP) 2, pp. 35-43, 2020.

[12] X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri and R. M. Summers, ”ChestX-ray8: Hospital-scale Chest X-ray Database and Benchmarks on Weakly-Supervised Classification and Localization of Common Thorax Diseases,” in IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[13] T.-Y. Lin, P. Goyal, R. Girshick, K. He and P. Dollar, ”Focal loss for Dense Object Detection.,” in IEEE International Conference on Computer Vision (ICCV), 2017.

Appendix

Source Code

```
app.py

from flask import Flask, flash, request, redirect, url_for,
   render_template
from werkzeug.utils import secure_filename
from tensorflow.keras.models import load_model
import numpy as np
import os
import cv2

# Creating a Flask Instance
app = Flask(__name__)

IMAGE_SIZE = (150, 150)
UPLOAD_FOLDER = 'static/uploads'
ALLOWED_EXTENSIONS = set(['png', 'jpg', 'jpeg'])
app.config.from_object(__name__)
app.config['SECRET_KEY'] = '7d441f27d441f27567d441f2b6176a'
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER

print("Loading Pre-trained Model ...")
```

```
model = load_model('model.h5')


def image_preprocessor(path):
    '''
    Function to pre-process the image before feeding to model.
    '''
    print('Processing Image ...')
    currImg_BGR = cv2.imread(path)
    b, g, r = cv2.split(currImg_BGR)
    currImg_RGB = cv2.merge([r, g, b])
    currImg = cv2.resize(currImg_RGB, IMAGE_SIZE)
    currImg = currImg/255.0
    currImg = np.reshape(currImg, (1, 150, 150, 3))
    return currImg


def model_pred(image):
    '''
    Perfroms predictions based on input image
    '''
    print("Image_shape", image.shape)
    print("Image_dimension", image.ndim)
    # Returns Probability:
    # prediction = model.predict(image)[0]
    # Returns class:
    prediction = model.predict_classes(image)[0]
    '''  if prediction == 1:
        return "Pneumonia"
    else:
        return "Normal"'''
    return (prediction)



def allowed_file(filename):
    return '.' in filename and \
           filename.rsplit('.', 1)[1].lower() in
              ALLOWED_EXTENSIONS



@app.route('/', methods=['GET', 'POST'])
def home():
    return render_template('index.html')



@app.route('/upload', methods=['GET', 'POST'])
def upload_file():
    # Checks if post request was submitted
    if request.method == 'POST':
        '''
        - request.url - http://127.0.0.1:5000/
        - request.files - Dictionaary of HTML elem "name"
           attribute and corrospondiong file details eg.
        "imageFile" : <FileStorage: 'Profile_Pic.jpg'
            ('image/jpeg')>
        '''
        # check if the post request has the file part

        if 'imageFile' not in request.files:
            flash('No file part')
            return redirect(request.url)
        # check if filename is an empty string
        file = request.files['imageFile']
        if file.filename == '':
            flash('No selected file')
            return redirect(request.url)

        # if file is uploaded
        if file and allowed_file(file.filename):
            filename = secure_filename(file.filename)
            imgPath = os.path.join(app.config['UPLOAD_FOLDER'],
               filename)
            file.save(imgPath)
            print(f"Image saved at {imgPath}")
            # Preprocessing Image
            image = image_preprocessor(imgPath)
            # Perfroming Prediction
            pred = model_pred(image)
            return render_template('upload.html', name=filename,
               result=pred)
    return redirect(url_for('home'))



if __name__ == '__main__':
    app.run(debug=True)
    #app.run(port=5002, threaded=False)
```

```
ResNet152V2 model building


!pip install tensorflow-gpu
# dimensions of our images.
img_width =224
img_height = 224
```

```
train_data_dir =
   '/content/drive/MyDrive/Datasets/pneumonia/Train'
validation_data_dir =
   '/content/drive/MyDrive/Datasets/pneumonia/Val'

# epochs - number times that the algorithm will work(iterate)
   through the entire training dataset.
epochs = 20

#batch_size -- the number of training examples utilized in
   one iteration.
batch_size = 16

from keras.applications.resnet_v2 import
   ResNet152V2,preprocess_input
basemdlrsnet152=ResNet152V2()
basemdlrsnet152.summary()

basemdlrsnet152=ResNet152V2(weights='imagenet',input_shape=(img_width,i

for lyr in basemdlrsnet152.layers:
  lyr.trainable=False

from tensorflow.keras import layers
from tensorflow.keras.layers import Dense, Flatten
from tensorflow.keras.models import Model

x=basemdlrsnet152.output
x=Dense(1024,activation='relu')(x)

x=Dense(1024,activation='relu')(x)
```

```
# Add a fully connected layer with 512 hidden units and ReLU
   activation
x=Dense(512,activation='relu')(x)

x=Dense(256,activation='relu')(x)

# Add a dropout rate of 0.5 # to avoid overfitting
x = layers.Dropout(0.3)(x)
x=Flatten()(x)
# Add a final softmax layer for classification -- output
x=Dense(1,activation='sigmoid')(x)

mymdl = Model(basemdlrsnet152.input, x)

mymdl.compile(
  loss='binary_crossentropy',
  optimizer='adam',
  metrics=['accuracy']
)

from tensorflow.keras.preprocessing.image import
   ImageDataGenerator
train_datagen =
   ImageDataGenerator(preprocessing_function=preprocess_input,rescale
   = 1/255, shear_range = 0.2, zoom_range = 0.2,
                            brightness_range = (0.1, 0.5),
                               horizontal_flip=True)

# this is the augmentation configuration we will use for
   testing:only rescaling
```

```
val_datagen = ImageDataGenerator(rescale=1. / 255)

training_set =
   train_datagen.flow_from_directory(train_data_dir,
                                     target_size=(img_width,
                                        img_height),
                                     batch_size = batch_size,
                                     class_mode = 'binary')

val_set = val_datagen.flow_from_directory(validation_data_dir,
                                 target_size = (img_width,
                                    img_height),
                                 batch_size = batch_size,
                                 class_mode = 'binary')

hist = mymdl.fit(training_set,
  validation_data=val_set,
  epochs=epochs,
  steps_per_epoch=len(training_set),
  validation_steps=len(val_set)
)

# Save the model
mymdl.save("mymodel.h5")

plt.style.use("classic")
plt.figure(figsize=(16, 9))
plt.plot(hist.history['loss'], label="Train Loss")
plt.plot(hist.history['val_loss'], label="Valid Loss")
plt.legend()
plt.xlabel("Epochs")
plt.ylabel("Loss")
plt.title("Loss over the Epochs")
plt.show()

plt.style.use("ggplot")
plt.figure(figsize=(16, 9))
plt.plot(hist.history['accuracy'], label="Train Accuracy")
plt.plot(hist.history['val_accuracy'], label="Valid Accuracy")
plt.legend()
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.title("Accuracy over the Epochs")
plt.show()
```

Output

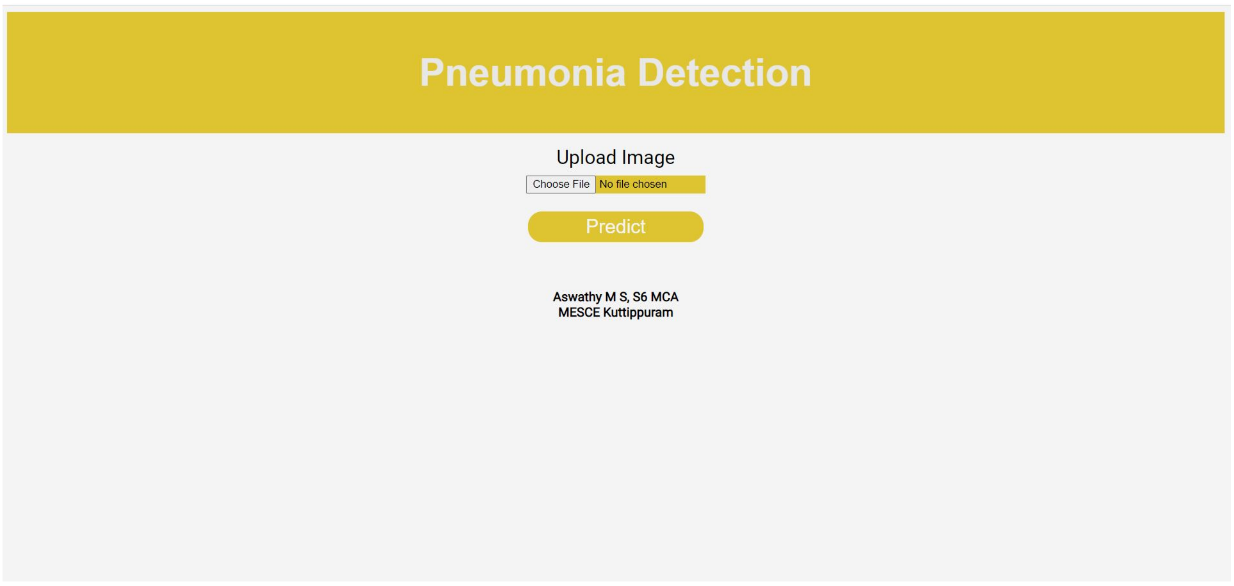


Figure A. 1: User Interface

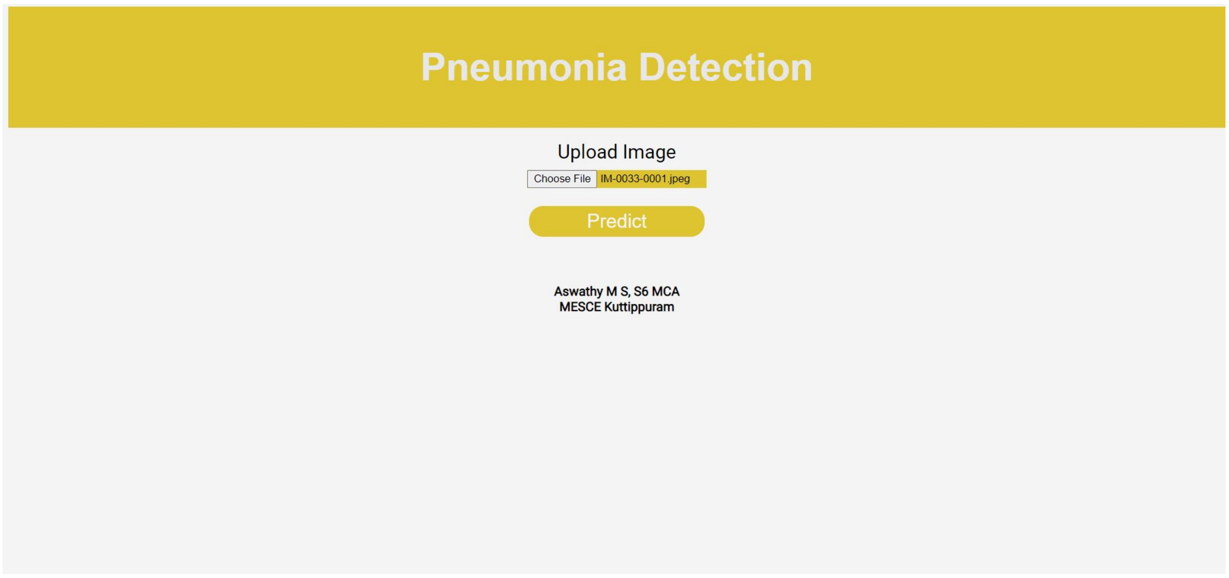
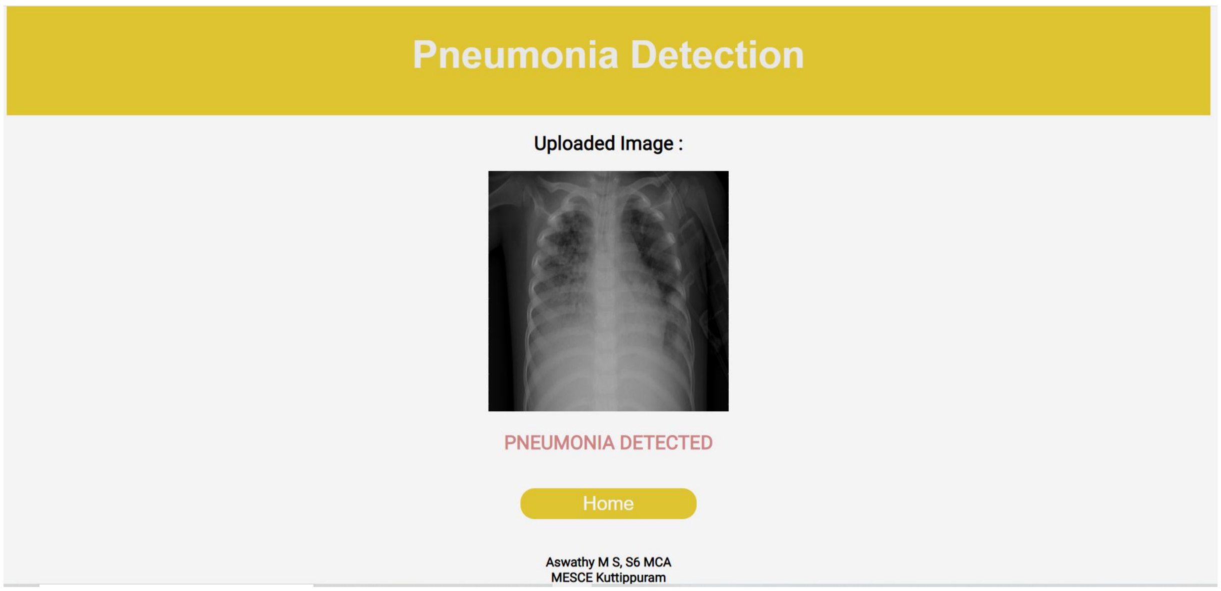


Figure A. 2: File Select



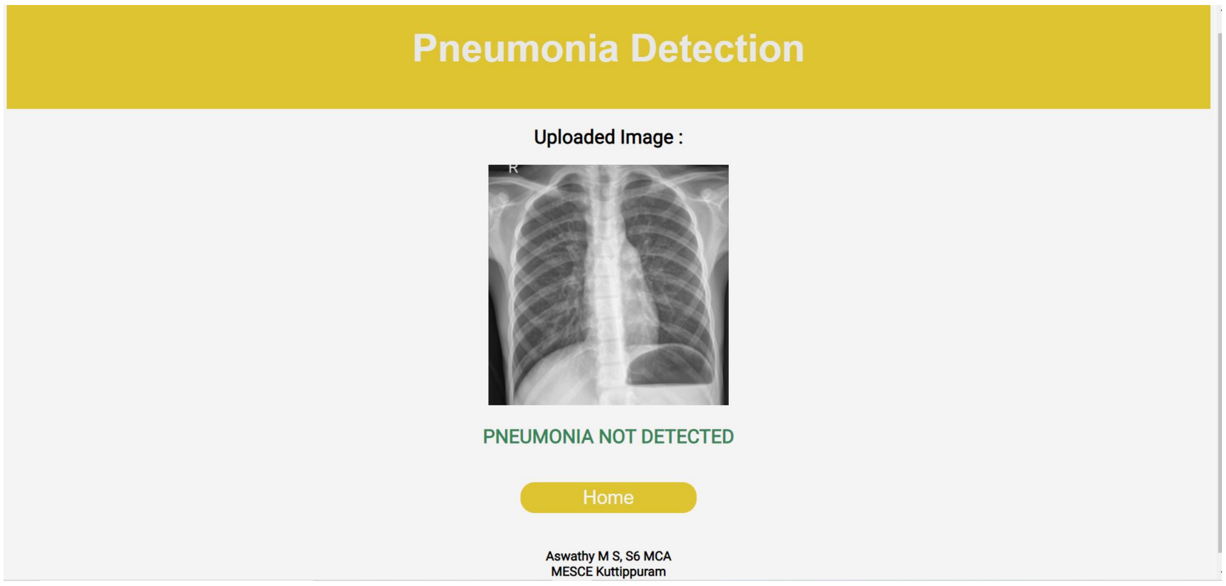
Pneumonia Detection

Uploaded Image :

PNEUMONIA DETECTED

Aswathy M S, S6 MCA
MESCE Kuttippuram

Figure A. 3: Pneumonia Detected



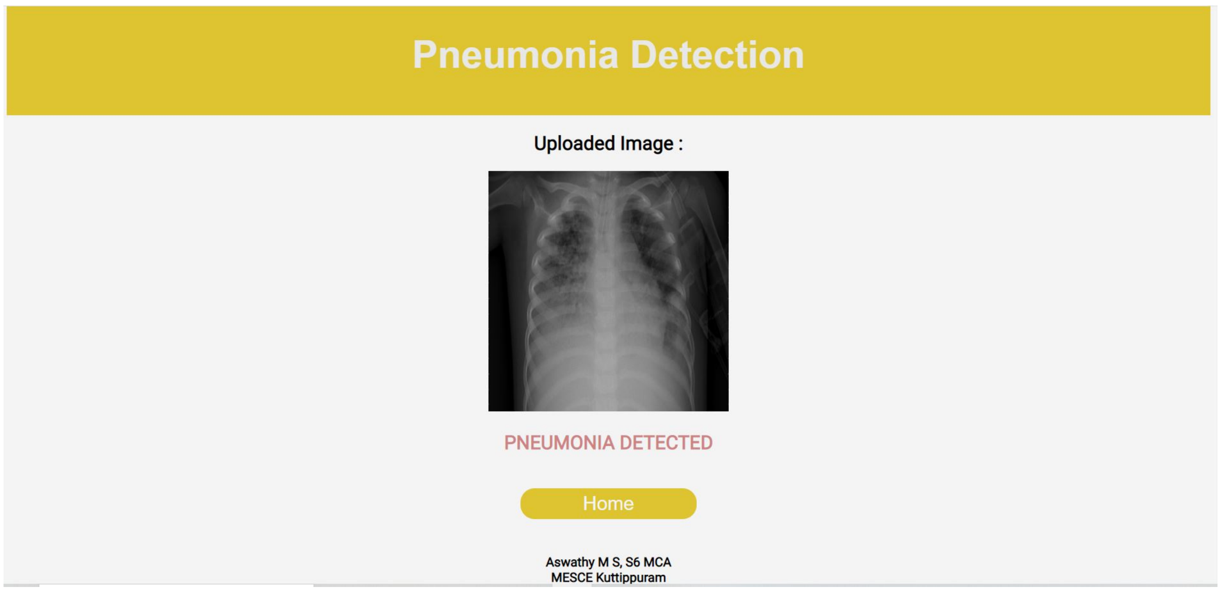
Pneumonia Detection

Uploaded Image :

PNEUMONIA NOT DETECTED

Aswathy M S, S6 MCA
MESCE Kuttippuram

Figure A. 4: Pneumonia not Detected

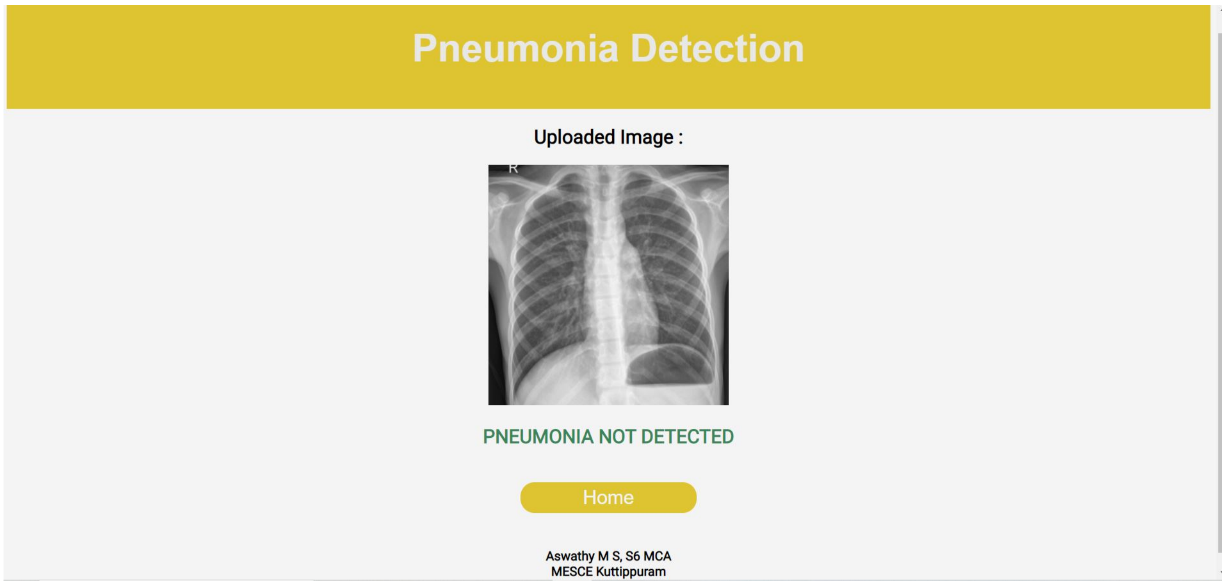

Pneumonia Detection

Uploaded Image :

PNEUMONIA DETECTED

Aswathy M S, S6 MCA
MESCE Kuttippuram

Figure A. 5: Pneumonia Detected


Pneumonia Detection

Uploaded Image :

PNEUMONIA NOT DETECTED

Aswathy M S, S6 MCA
MESCE Kuttippuram

Figure A. 6: Pneumonia not Detected

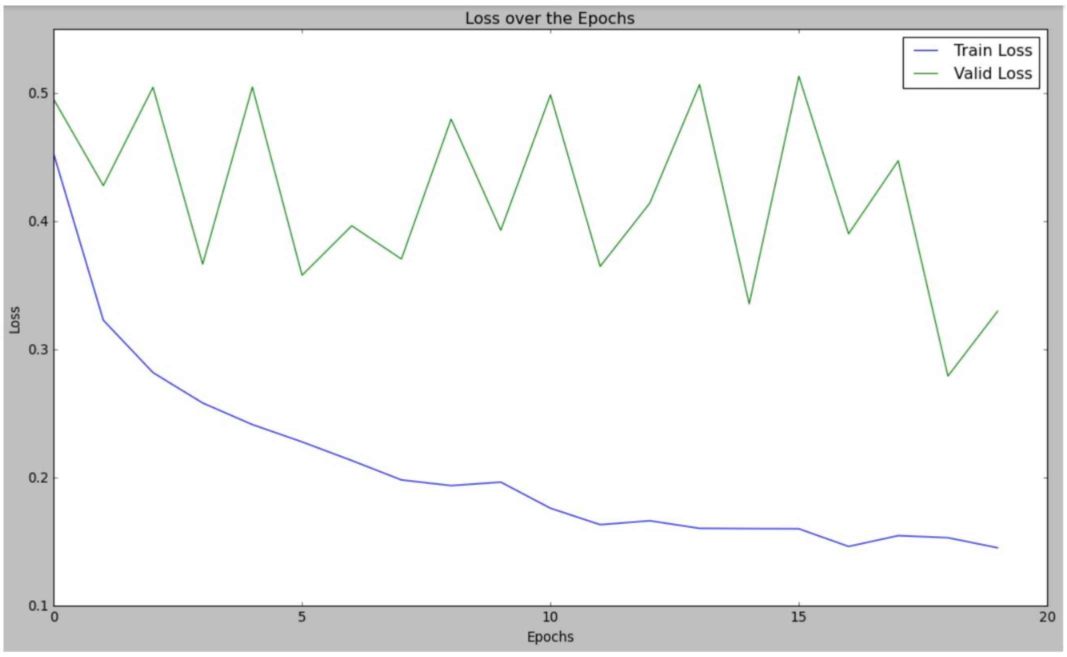


Figure A. 7: Loss over Epochs

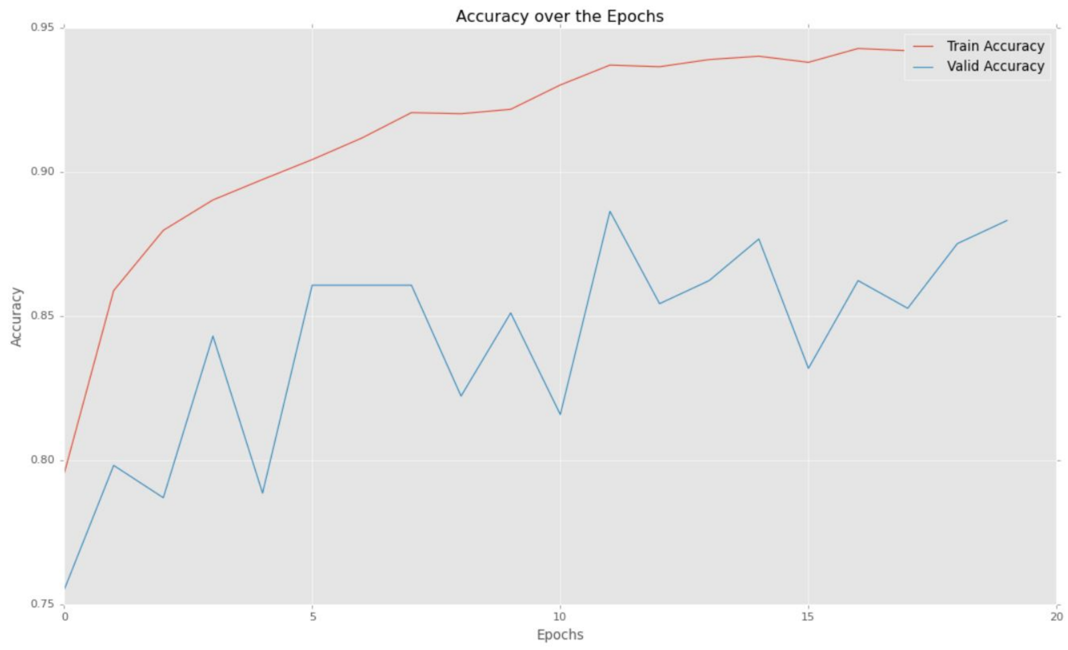
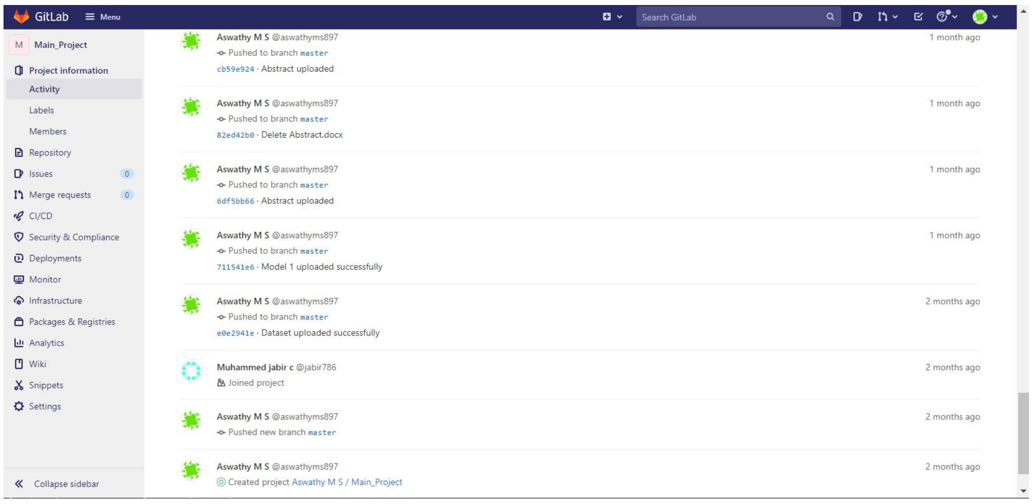


Figure A. 8: Validation over Epochs

Git History

Commits



Aswathy M S @aswathyms897 1 month ago
Pushed to branch master
cb59e924 · Abstract uploaded

Aswathy M S @aswathyms897 1 month ago
Pushed to branch master
82ed42b0 · Delete Abstract.docx

Aswathy M S @aswathyms897 1 month ago
Pushed to branch master
6df5bb66 · Abstract uploaded

Aswathy M S @aswathyms897 1 month ago
Pushed to branch master
711541e6 · Model 1 uploaded successfully

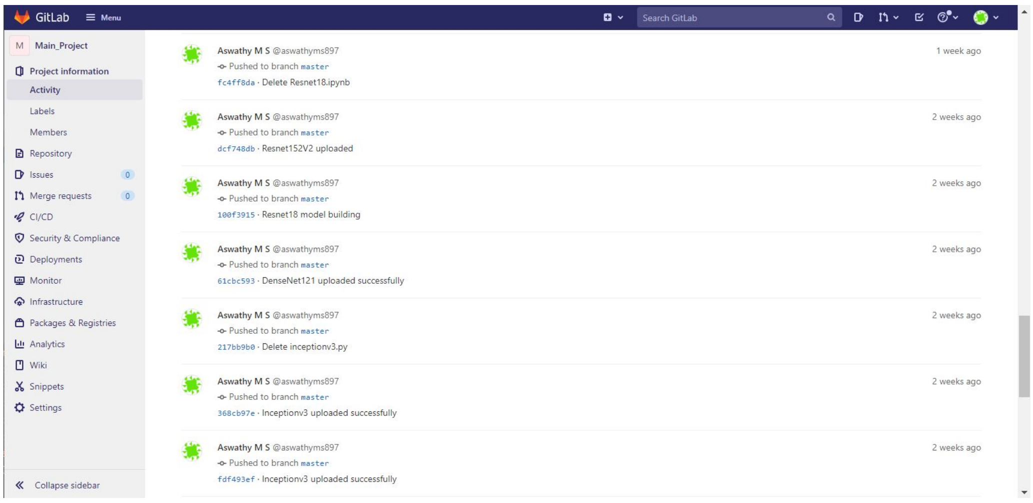
Aswathy M S @aswathyms897 2 months ago
Pushed to branch master
e0e2941e · Dataset uploaded successfully

Muhammed jabir c @jabir786 2 months ago
Joined project

Aswathy M S @aswathyms897 2 months ago
Pushed new branch master

Aswathy M S @aswathyms897 2 months ago
Created project Aswathy M S / Main_Project

Figure A. 9: Git Commits 1



Aswathy M S @aswathyms897 1 week ago
Pushed to branch master
fc4ff8da · Delete Resnet18.ipynb

Aswathy M S @aswathyms897 2 weeks ago
Pushed to branch master
dcf748db · Resnet152V2 uploaded

Aswathy M S @aswathyms897 2 weeks ago
Pushed to branch master
100f3915 · Resnet18 model building

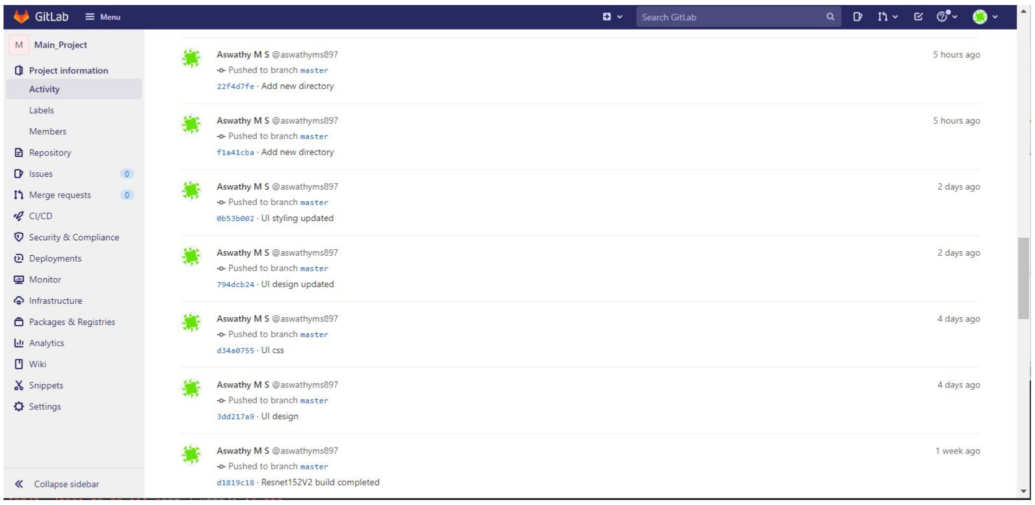
Aswathy M S @aswathyms897 2 weeks ago
Pushed to branch master
61cbc593 · DenseNet121 uploaded successfully

Aswathy M S @aswathyms897 2 weeks ago
Pushed to branch master
217bb9b0 · Delete inceptionv3.py

Aswathy M S @aswathyms897 2 weeks ago
Pushed to branch master
368cb97e · Inceptionv3 uploaded successfully

Aswathy M S @aswathyms897 2 weeks ago
Pushed to branch master
fdf493ef · Inceptionv3 uploaded successfully

Figure A. 10: Git Commits 2



Aswathy M S @aswathyms897 — 5 hours ago
Pushed to branch master
22f4d7fe · Add new directory

Aswathy M S @aswathyms897 — 5 hours ago
Pushed to branch master
f1a41cba · Add new directory

Aswathy M S @aswathyms897 — 2 days ago
Pushed to branch master
0b53b002 · UI styling updated

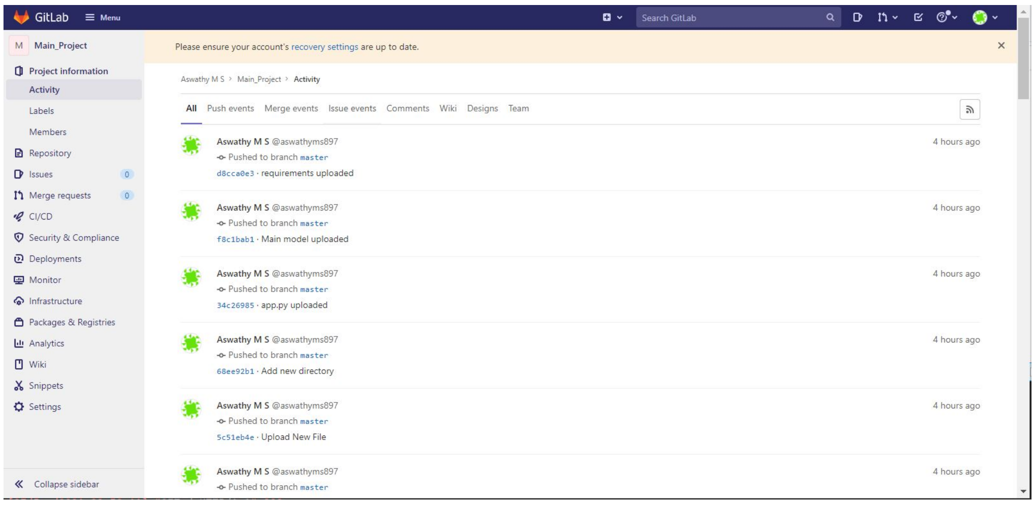
Aswathy M S @aswathyms897 — 2 days ago
Pushed to branch master
794dcb24 · UI design updated

Aswathy M S @aswathyms897 — 4 days ago
Pushed to branch master
d34a0755 · UI css

Aswathy M S @aswathyms897 — 4 days ago
Pushed to branch master
3dd217a9 · UI design

Aswathy M S @aswathyms897 — 1 week ago
Pushed to branch master
d1819c18 · Resnet152V2 build completed

Figure A. 11: Git Commits 3



Please ensure your account's recovery settings are up to date.

Aswathy M S @aswathyms897 — 4 hours ago
Pushed to branch master
d8cca0e3 · requirements uploaded

Aswathy M S @aswathyms897 — 4 hours ago
Pushed to branch master
f8c1bab1 · Main model uploaded

Aswathy M S @aswathyms897 — 4 hours ago
Pushed to branch master
34c26985 · app.py uploaded

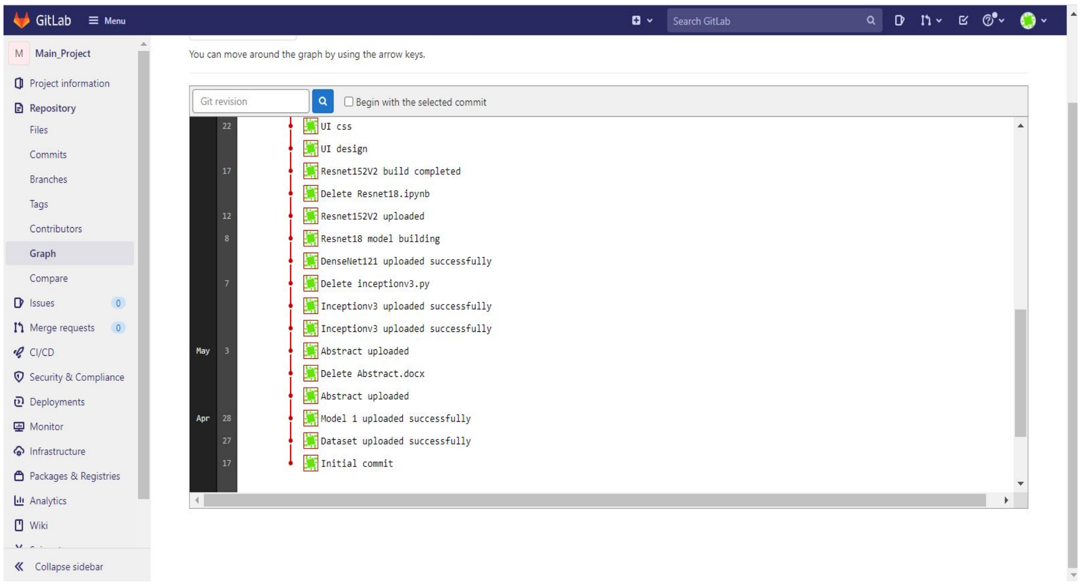
Aswathy M S @aswathyms897 — 4 hours ago
Pushed to branch master
68ee92b1 · Add new directory

Aswathy M S @aswathyms897 — 4 hours ago
Pushed to branch master
5c51eb4e · Upload New File

Aswathy M S @aswathyms897 — 4 hours ago
Pushed to branch master

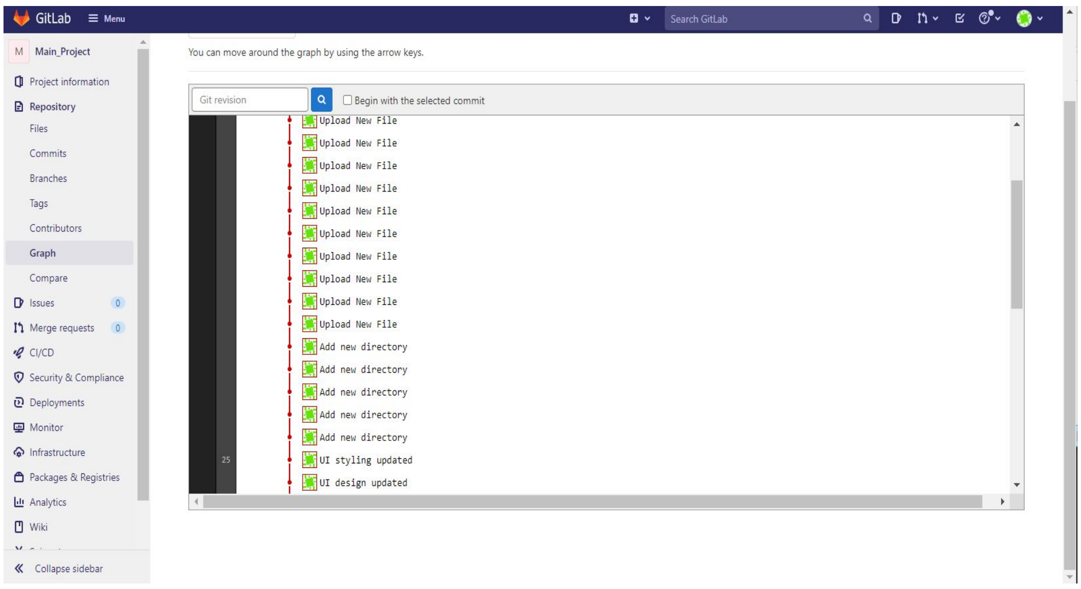
Figure A. 12: Git Commits 4

Graph



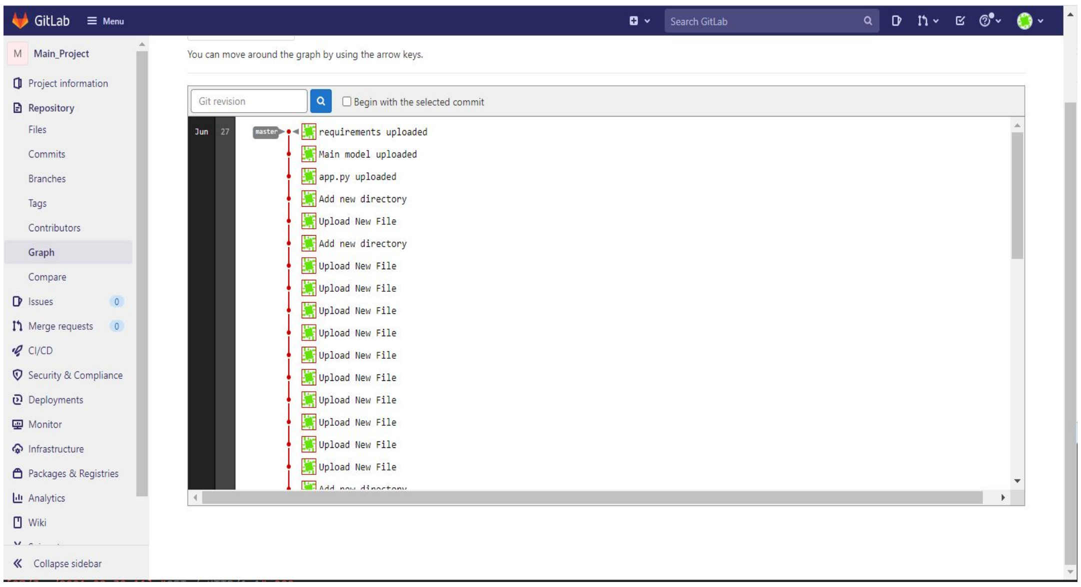
```
     22  UI css
         UI design
     17  Resnet152V2 build completed
         Delete Resnet18.ipynb
     12  Resnet152V2 uploaded
      8  Resnet18 model building
         DenseNet121 uploaded successfully
      7  Delete inceptionv3.py
         Inceptionv3 uploaded successfully
         Inceptionv3 uploaded successfully
May   3  Abstract uploaded
         Delete Abstract.docx
         Abstract uploaded
Apr  28  Model 1 uploaded successfully
     27  Dataset uploaded successfully
     17  Initial commit
```

Figure A. 13: Git Graph 1



```
     Upload New File
     Upload New File
     Upload New File
     Upload New File
     Upload New File
     Upload New File
     Upload New File
     Upload New File
     Upload New File
     Upload New File
     Add new directory
     Add new directory
     Add new directory
     Add new directory
     Add new directory
 25  UI styling updated
     UI design updated
```

Figure A. 14: Git Graph 2



You can move around the graph by using the arrow keys.

```
requirements uploaded
Main model uploaded
app.py uploaded
Add new directory
Upload New File
Add new directory
Upload New File
Upload New File
Upload New File
Upload New File
Upload New File
Upload New File
Upload New File
Upload New File
Upload New File
Upload New File
Add new directory
```

Figure A. 15: Git Graph 3