Pneumonia Detection using Convolutional Neural Network

GOPIREDDY LAKSHMI KEERTHI (STUDENT ID: 11653876)	LAKSHMIKEERTHIGOPIREDDY@MY.UNT.EDU	MAJOR: ARTIFICIAL INTELLIGENCE
INDHU SRI KRISHNARAJ (STUDENT ID: 11661746)	INDHUSRIKRISHNARAJ@MY.UNT.EDU	MAJOR: ARTIFICIAL INTELLIGENCE
VISHAL RACHURI (STUDENT ID: 11658243)	VISHALRACHURI@MY.UNT.EDU	MAJOR: ARTIFICIAL INTELLIGENCE
JYOTHIKA VOLLIREDDY (STUDENT ID: 11652684)	JYOTHIKAVOLLIREDDY@MY.UNT.EDU	MAJOR: CYBER SECURITY
PAVANI JAYA KEERTHANA CHITTEDI (STUDENT ID: 11672192)	PAVANIJAYAKEERTHANCHITTEDI@MY.UNT.EDU	MAJOR: CYBER SECURITY

Introduction

Aim: Predict whether a person has Pneumonia or not from their chest X-ray

Motivation:

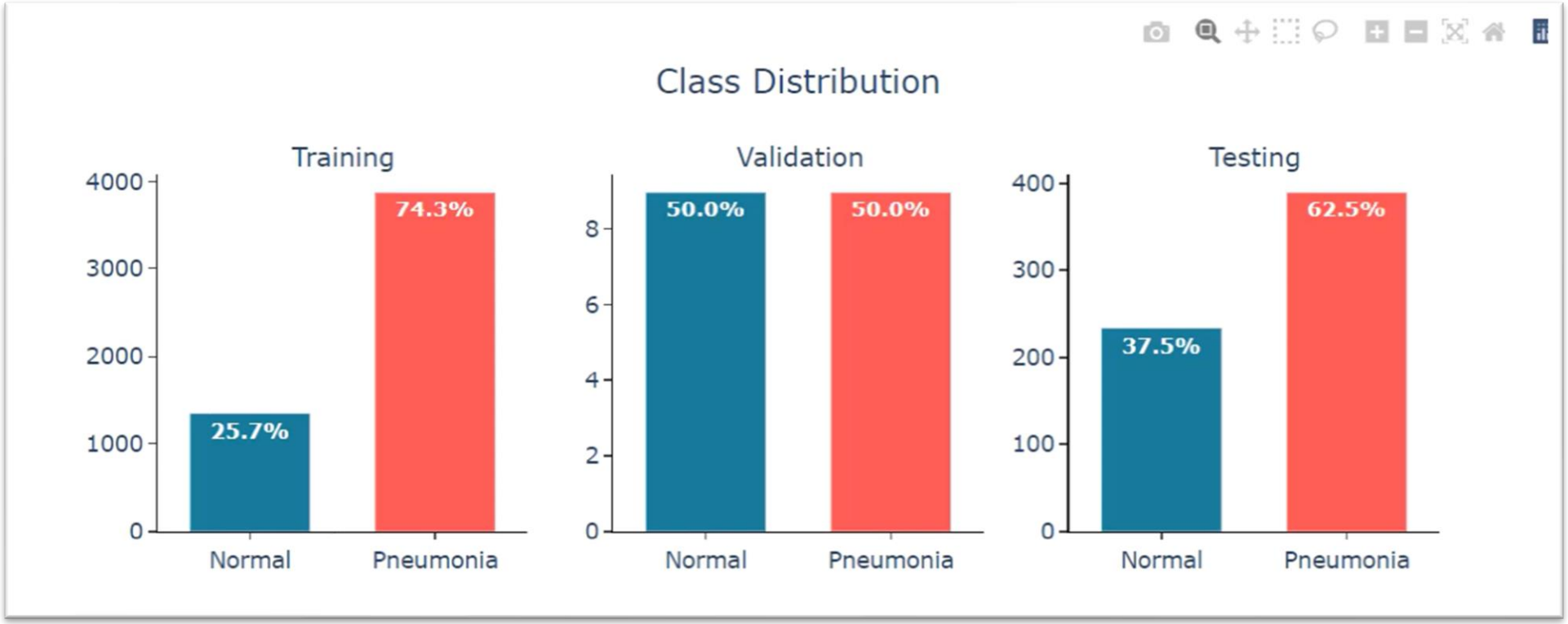
- We have heard about multiple respiratory problems during and post covid situations
- Pneumonia is one of the severe respiratory infection that can have serious consequences, especially for vulnerable populations such as children and elderly individuals
- Early and accurate detection of pneumonia is crucial for timely intervention and effective treatment
- Traditional methods for pneumonia diagnosis:
  - Using chest X-rays and clinical examinations,
  - Time-consuming and costly
- Leveraging Convolutional Neural Networks (CNNs), a type of machine learning model, for pneumonia detection from chest X-rays can help in faster diagnosis and is cost-effective
- Why CNN? CNNs is known to learn complex patterns from images

Data Description

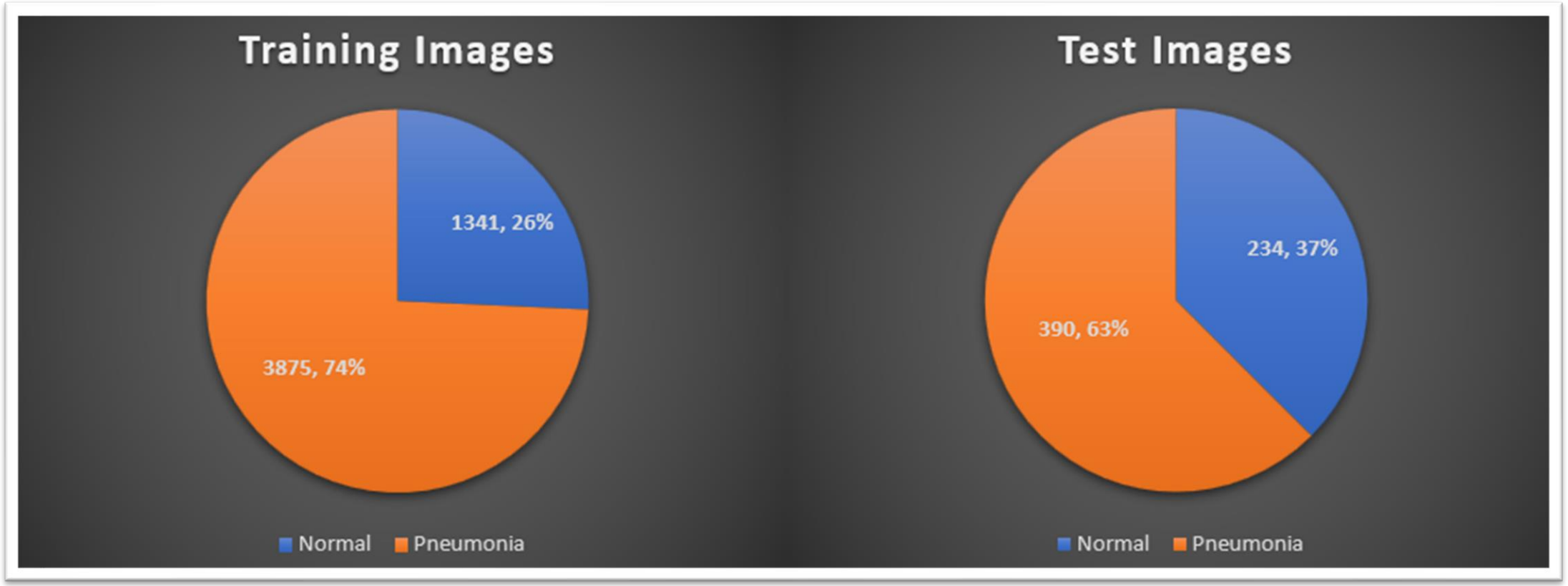
Data type: Images in jpeg format

- Data size: 1.16 GB
- Training Data:
  - Chest X-rays of normal patients: 1341
  - Chest X-rays of Pneumonia patients: 3875
- Test Data:
  - Chest X-rays of normal patients: 234
  - Chest X-rays of Pneumonia patients: 390
- Validation Data:
  - Only 16 images in original dataset
  - Using 15% of training data for validation

Dataset Distribution - Original Dataset as given in Kaggle

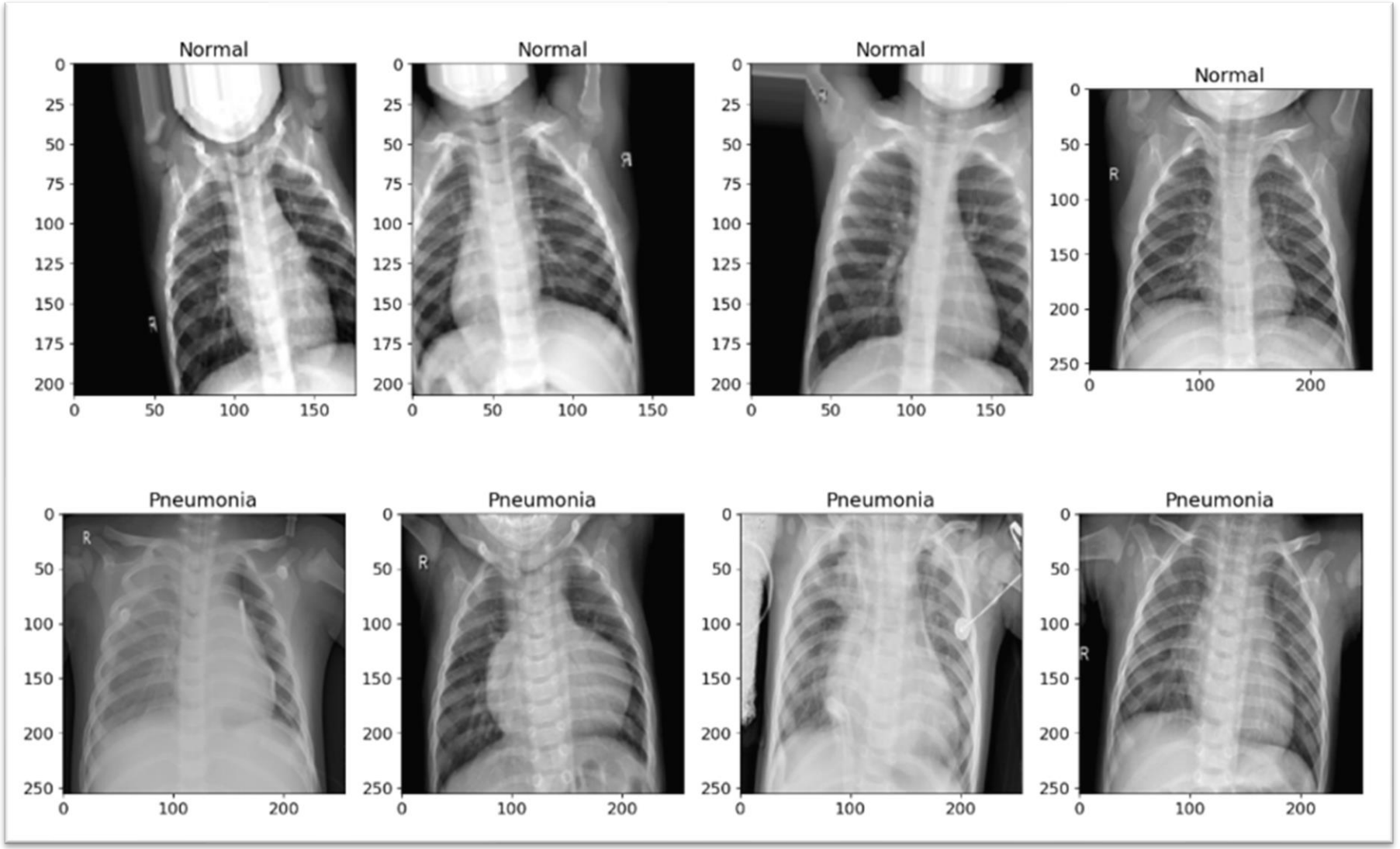


Dataset Distribution - Considering only training and testing images



*Validation data is 15% of the training data

Sample Images of Chest X-Rays



Model Training using Convolutional Neural Network

- Model was trained using Convolutional Neural Network
- 2D convolutional operation and ReLU activation function is used in each layer followed by max pooling for spatial reducing dimensions
- Number of filters in each layer: [16, 32, 64, 128, 128]

```
model = Sequential([
    Conv2D(16, (3, 3), activation='relu', input_shape=(CFG.img_height, CFG.img_width, 3)),
    MaxPooling2D(2, 2),

    Conv2D(32, (3, 3), activation='relu'),
    MaxPooling2D(2, 2),

    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D(2, 2),

    Conv2D(128, (3, 3), activation='relu'),
    MaxPooling2D(2, 2),

    Conv2D(128, (3, 3), activation='relu'),
    MaxPooling2D(2, 2),

    Flatten(),
    Dense(256, activation='relu'),
    Dropout(0.2),
    Dense(1, activation='sigmoid')
])
```

Model Training

```
34/34 [==============================] - ETA: 0s - loss: 0.2567 - acc: 0.8906 - precision: 0.9283 - recall: 0.9225
Epoch 18: val_loss did not improve from 0.28514
34/34 [==============================] - 33s 980ms/step - loss: 0.2567 - acc: 0.8906 - precision: 0.9283 - recall: 0.9225
- val_loss: 0.3441 - val_acc: 0.8427 - val_precision: 0.9771 - val_recall: 0.8072
Epoch 19/20
34/34 [==============================] - ETA: 0s - loss: 0.2904 - acc: 0.8805 - precision: 0.9222 - recall: 0.9153
Epoch 19: val_loss improved from 0.28514 to 0.26441, saving model to CNN_model.h5
34/34 [==============================] - 34s 989ms/step - loss: 0.2904 - acc: 0.8805 - precision: 0.9222 - recall: 0.9153
- val_loss: 0.2644 - val_acc: 0.8568 - val_precision: 0.8941 - val_recall: 0.9157
Epoch 20/20
34/34 [==============================] - ETA: 0s - loss: 0.2490 - acc: 0.9026 - precision: 0.9419 - recall: 0.9256
Epoch 20: val_loss did not improve from 0.26441
34/34 [==============================] - 35s 1s/step - loss: 0.2490 - acc: 0.9026 - precision: 0.9419 - recall: 0.9256 -
val_loss: 0.3091 - val_acc: 0.8619 - val_precision: 0.9216 - val_recall: 0.8898
624/624 [==============================] - 11s 17ms/step - loss: 0.4525 - acc: 0.8157 - precision: 0.7835 - recall: 0.974
4
Evaluate acc: 81.57%
Evaluate precision: 78.35%
Evaluate recall: 97.44%
```

Validation and test metrics are available as above

Hyperparameter Tuning and Activated Early Stopping

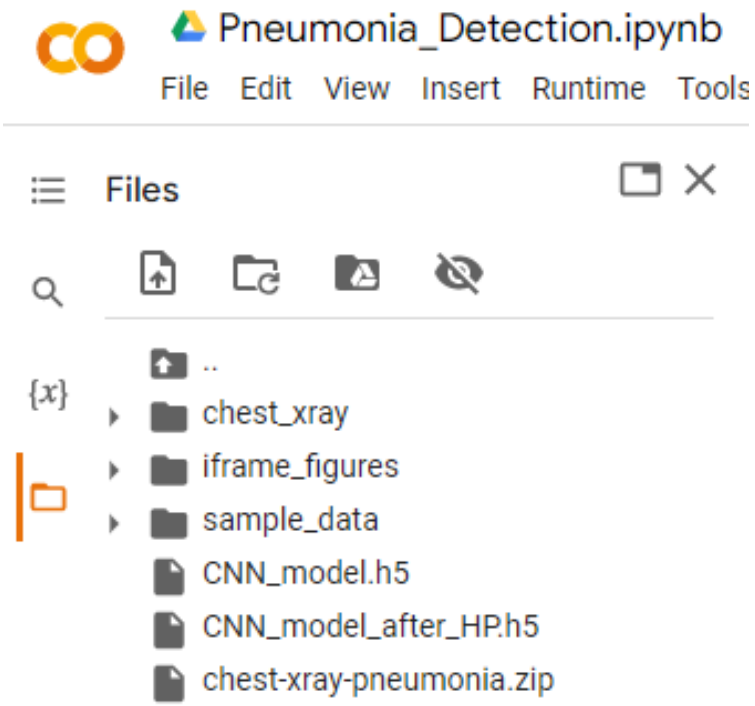
```
# Best Parameters
params = {'n_layers': 3,
          'n_filters_0': 73,
          'kernel_size_0': 5,
          'pooling_0': 'max',
          'n_filters_1': 114,
          'kernel_size_1': 4,
          'pooling_1': 'max',
          'n_filters_2': 114,
          'kernel_size_2': 5,
          'pooling_2': 'avg',
          'n_neurons_fc1': 239,
          'dropout_rate': 0.11514078475381184,
          'learning_rate': 0.00010254312064434285
}
```

```
Epoch 13/20
34/34 [==============================] - ETA: 0s - loss: 0.3064 - acc: 0.8778 - precision: 0.9176 - recall: 0.9209
Epoch 13: val_loss did not improve from 0.28386
34/34 [==============================] - 156s 5s/step - loss: 0.3064 - acc: 0.8778 - precision: 0.9176 - recall: 0.9209 - val
_loss: 0.3062 - val_acc: 0.8657 - val_precision: 0.9457 - val_recall: 0.8692
Epoch 13: early stopping
624/624 [==============================] - 36s 57ms/step - loss: 0.3626 - acc: 0.8590 - precision: 0.8432 - recall: 0.9513
Evaluate acc: 85.90%
Evaluate precision: 84.32%
Evaluate recall: 95.13%
```

Validation and test metrics are available as above

Model Saving

- Model is saved with .h5 extension
- .h5 extension indicates **Hierarchical Data Format 5 File**
- We have saved two models,
  - CNN model without hyperparameter tuning
  - CNN model without hyperparameter tuning and early stop

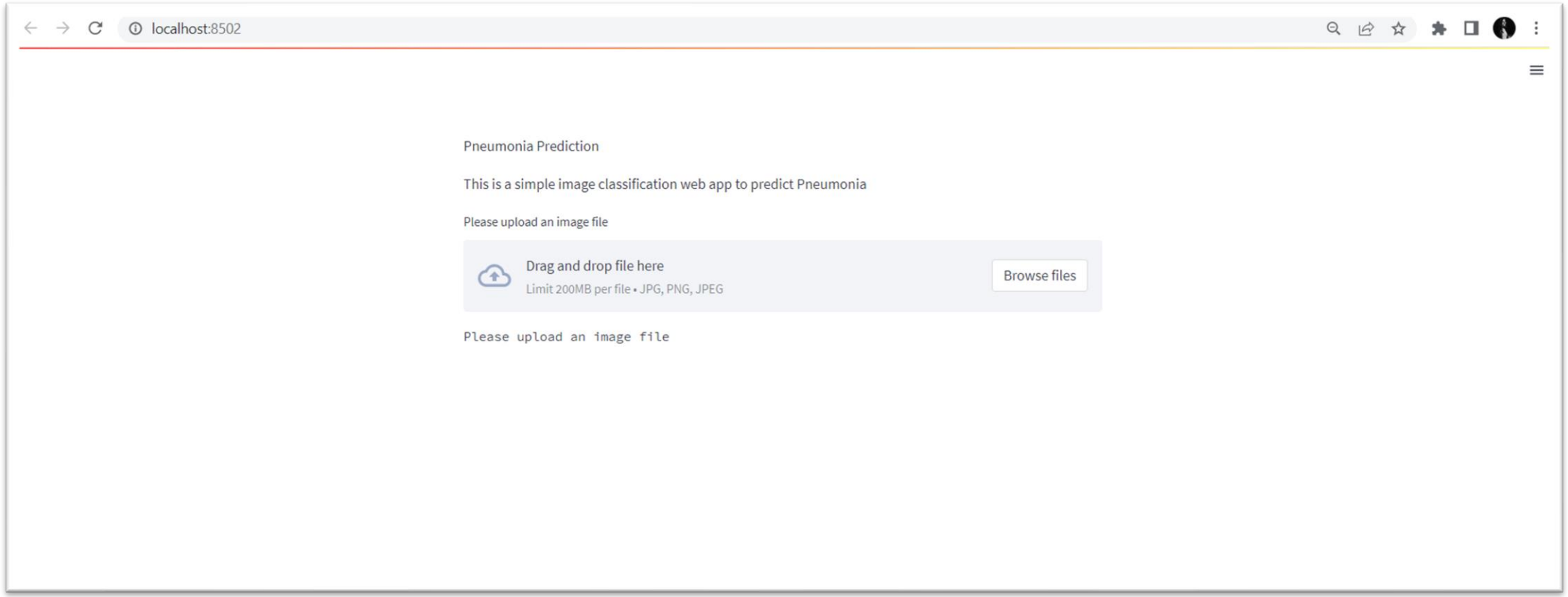


Model Deployment

- Using the model after hyperparameter tuning and enabling early stop as the final model
- Deploying to a local webpage using Streamlit
- Streamlit is a python-based library for creating web apps

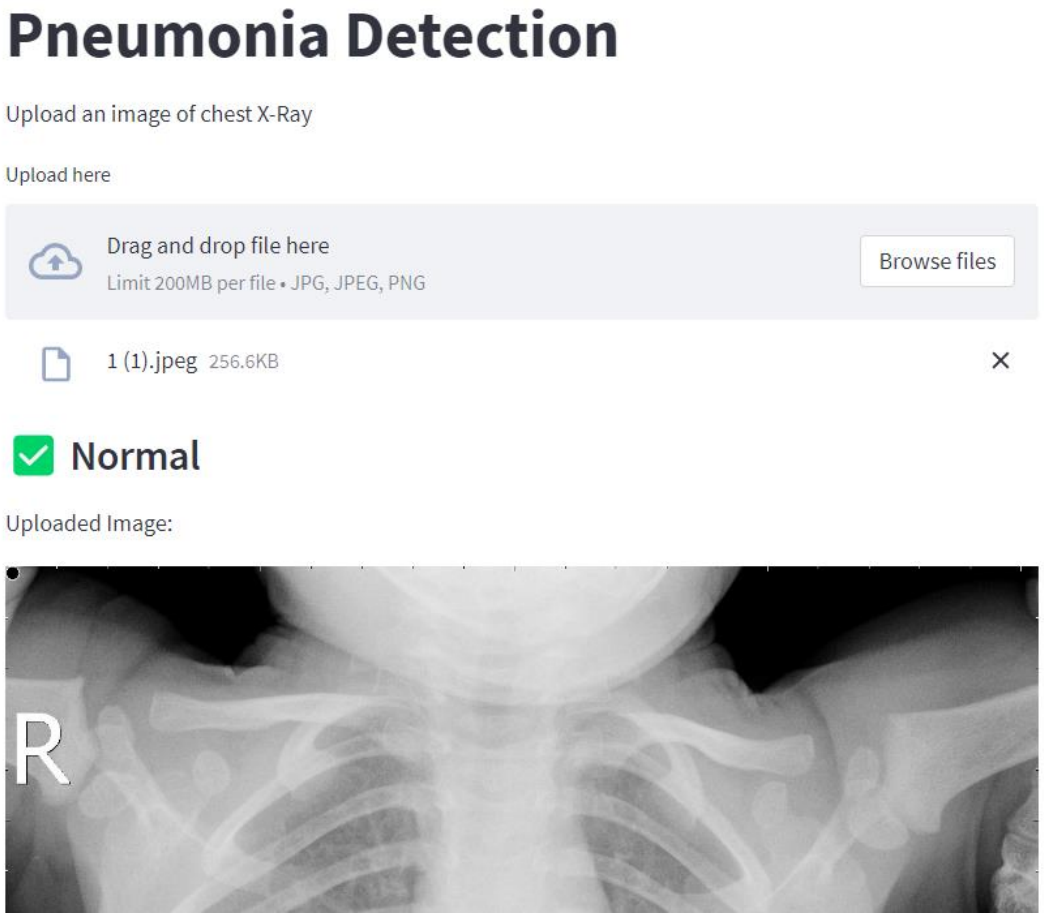
Model Deployment: Input Page

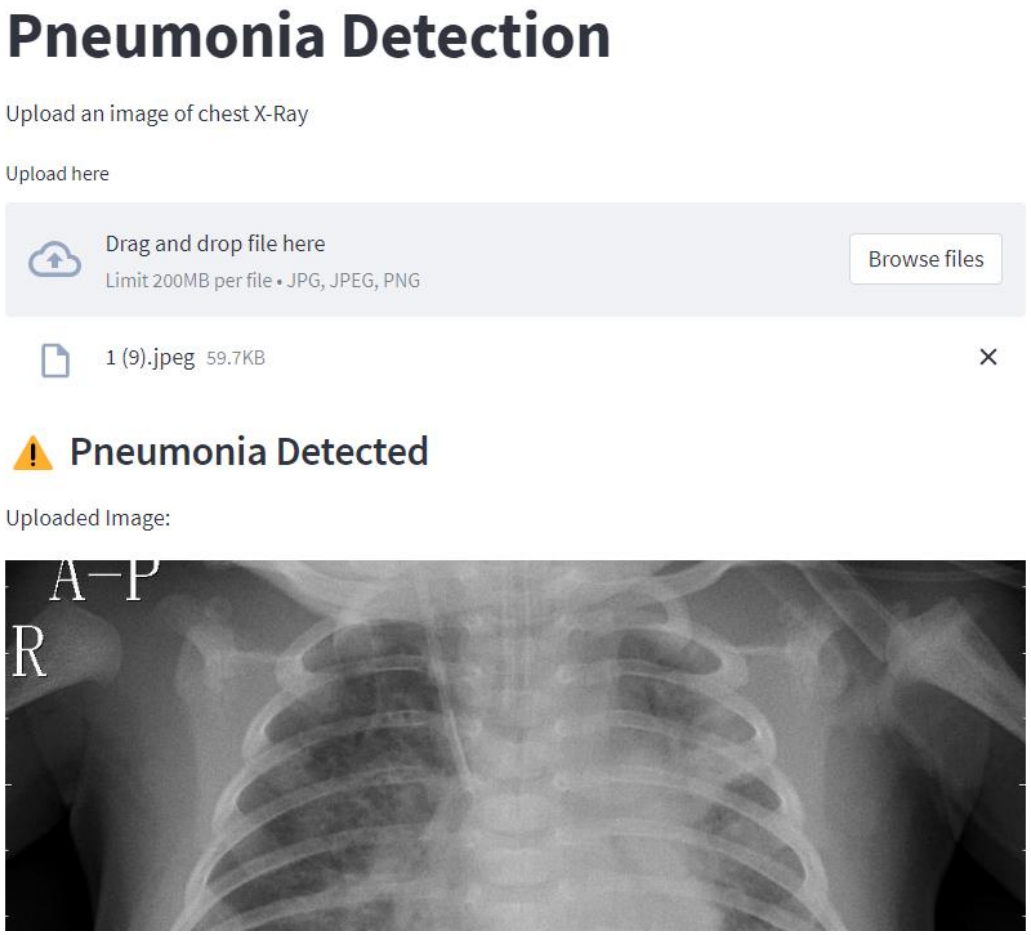
- Input should be an image (Chest X-Ray) in .jpg or .png or .jpeg formats



Model Deployment: Output Page

- Output i.e., prediction made by the model is displayed above the image





THANK YOU!