Final Project Report

1. INTRODUCTION

1.1 PROJECT OVERVIEWS

The Covid-19 pandemic has placed unprecedented strain on global healthcare systems, necessitating innovative solutions for rapid and accurate diagnosis. This project, "Covid-19 Detection from Lung X-rays," leverages deep learning algorithms to analyze lung X-ray images, identifying signs of Covid-19 infection. By utilizing vast datasets and advanced image recognition technology, the project aims to facilitate early detection and improve patient outcomes through timely intervention.

1.2 OBJECTIVES

- Develop an AI-based system capable of accurately detecting Covid-19 from lung X-ray images.
- Reduce the diagnostic burden on healthcare professionals by providing a rapid and reliable screening tool.
- Enhance diagnostic capabilities in underserved and rural areas with limited access to expert radiologists.
- Aid public health authorities in monitoring Covid-19 trends and implementing targeted interventions.

2. PROJECT INITIALIZATION AND PLANNING PHASE

2.1 DEFINE PROBLEM STATEMENT

A hospital dealing with an influx of COVID-19 patients needs an efficient system to quickly and accurately analyze lung X-ray images,thereby, expediting reliable diagnosis to aid in triage and treatment decisions, despite varying image quality and patient demographics.

2.2 PROJECT PROPOSAL(PROPOSED PROPOSAL)

To address the defined problem, we propose the development of a deep learning-based AI system for the detection of Covid-19 from lung X-ray images. The proposed solution involves:

- Collecting a large dataset of labeled lung X-ray images, including Covid-19 positive,normal,viral pneumonia,lung opacity cases.
- Developing and training a deep learning model, specifically a convolutional neural network (CNN), to recognize patterns indicative of Covid-19 infection.

- Validating the model's accuracy and robustness through rigorous testing on a separate validation dataset.
- Integrating the AI system into clinical workflows to assist healthcare professionals in rapid diagnosis and decision-making.

2.3 INITIAL PROJECT PLANNING

The initial planning phase involved setting up a detailed project roadmap with defined milestones and deliverables:

Phase 1: Data collection and preprocessing (1 day)
Phase 2: Model development and initial training (2 days)
Phase 3: Model validation and evaluation (2 days)
Phase 4: Model optimization and tuning (2 days)
Phase 5: Deployment and integration (1 day)

3. DATA COLLECTION AND PREPROCESSING PHASE

3.1 DATA COLLECTION PLAN AND RAW DATA SOURCES IDENTIFIED

The data collection plan involved identifying and sourcing lung X-ray images from various publicly available datasets and hospital collaborations. The primary sources included:

Raw Data Sources Identified:

Covid-19 Images:

Padchest Dataset: 2,473 chest X-ray (CXR) images.
Germany Medical School: 183 CXR images.
SIRM, GitHub, Kaggle, and Twitter: 559 CXR images.
Additional GitHub Source: 400 CXR images.

Normal Images:

RSNA: 8,851 CXR images.
Kaggle: 1,341 CXR images.

Lung Opacity Images:

Radiological Society of North America (RSNA) CXR Dataset: 6,012 CXR images.

Viral Pneumonia Images:

Chest X-Ray Images (Pneumonia) Database: 1,345 CXR images.

All this diverse and extensive collection of lung X-ray images collectively in the dataset for ”CovidVision - Advanced COVID-19 Detection” is sourced from Kaggle categorized by COVID-19, pneumonia, lung opacity and normal conditions.

.

3.2 DATA QUALITY REPORT

A thorough data quality report was generated to assess the initial dataset's completeness, consistency, and accuracy:

- **Completeness:** Ensured all X-ray images were properly labeled with Covid-19 status.
- **Consistency:** Checked for uniformity in image resolution and format.
- **Accuracy:** Verified the correctness of labels.

3.3 DATA PREPROCESSING

Data preprocessing involved several steps to prepare the raw data for model training:

- **Image Resizing:** Standardized all X-ray images to a uniform size.
- **Normalization:** Applied normalization techniques to scale pixel values.
- **Augmentation:** Implemented data augmentation methods such as zoom to enhance model robustness.
- **Splitting:** Divided the dataset into training, validation, and test sets to ensure unbiased model evaluation.

4. MODEL DEVELOPMENT PHASE

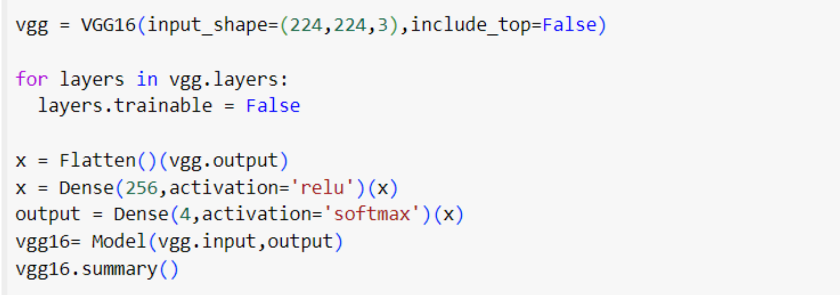
4.1 MODEL SELECTION REPORT

The model selection process for Covid-19 detection from lung X-ray images involved evaluating convolutional neural network (CNN) architectures known for their efficiency in image recognition tasks. The models considered included VGG16, ResNet50, Inception, and Xception.

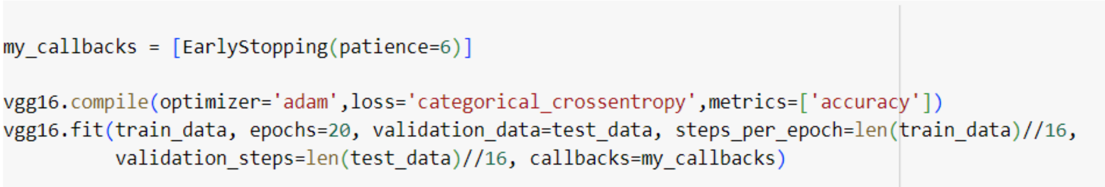
4.2 INITIAL MODEL TRAINING CODE, MODEL VALIDATION and EVALUATION REPORT

Initial Model Training Code:

- VGG16:

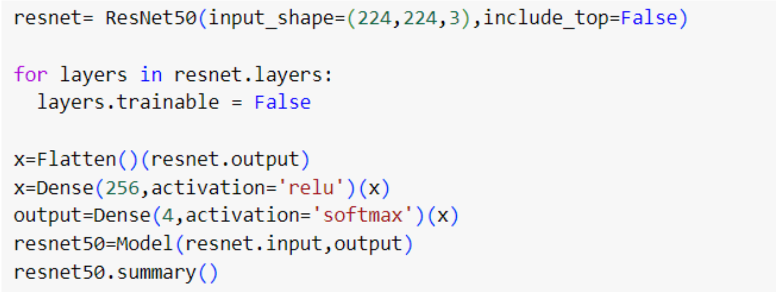


```python
vgg = VGG16(input_shape=(224,224,3),include_top=False)

for layers in vgg.layers:
  layers.trainable = False

x = Flatten()(vgg.output)
x = Dense(256,activation='relu')(x)
output = Dense(4,activation='softmax')(x)
vgg16= Model(vgg.input,output)
vgg16.summary()
```

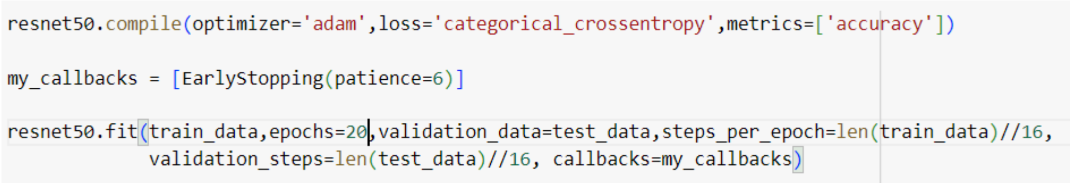


```python
my_callbacks = [EarlyStopping(patience=6)]

vgg16.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])
vgg16.fit(train_data, epochs=20, validation_data=test_data, steps_per_epoch=len(train_data)//16,
          validation_steps=len(test_data)//16, callbacks=my_callbacks)
```

- ResNet50:

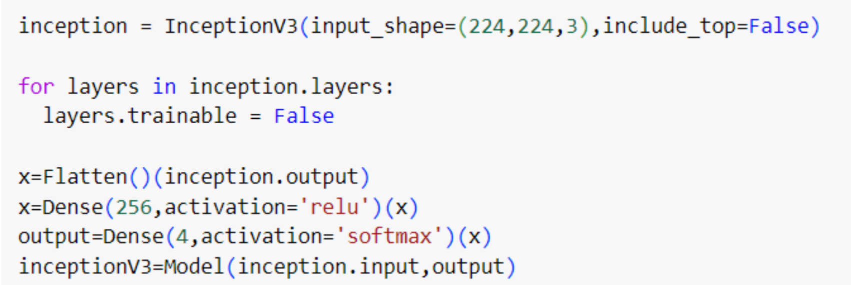


```python
resnet= ResNet50(input_shape=(224,224,3),include_top=False)

for layers in resnet.layers:
  layers.trainable = False

x=Flatten()(resnet.output)
x=Dense(256,activation='relu')(x)
output=Dense(4,activation='softmax')(x)
resnet50=Model(resnet.input,output)
resnet50.summary()
```

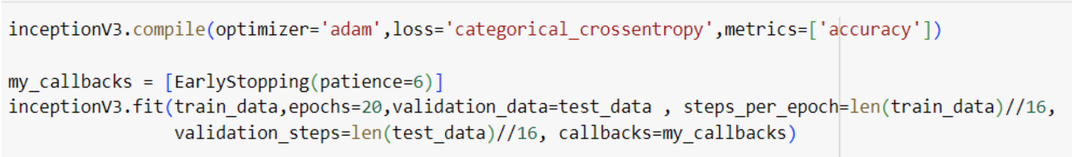


```python
resnet50.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])

my_callbacks = [EarlyStopping(patience=6)]

resnet50.fit(train_data,epochs=20,validation_data=test_data,steps_per_epoch=len(train_data)//16,
             validation_steps=len(test_data)//16, callbacks=my_callbacks)
```

- Inception:

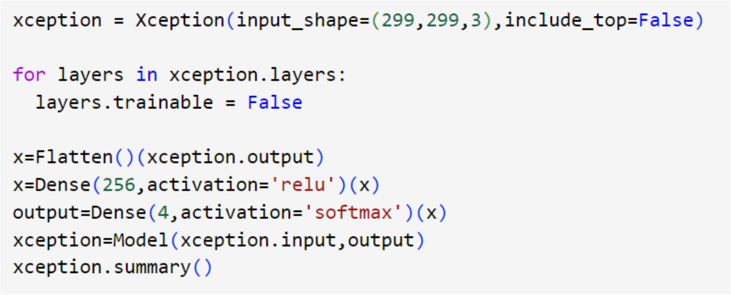


```python
inception = InceptionV3(input_shape=(224,224,3),include_top=False)

for layers in inception.layers:
  layers.trainable = False

x=Flatten()(inception.output)
x=Dense(256,activation='relu')(x)
output=Dense(4,activation='softmax')(x)
inceptionV3=Model(inception.input,output)
```

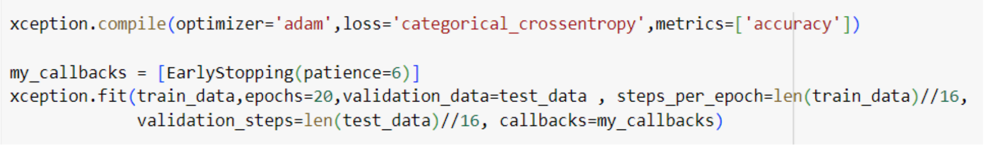


```python
inceptionV3.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])

my_callbacks = [EarlyStopping(patience=6)]
inceptionV3.fit(train_data,epochs=20,validation_data=test_data , steps_per_epoch=len(train_data)//16,
                validation_steps=len(test_data)//16, callbacks=my_callbacks)
```

- Xception:



```
xception = Xception(input_shape=(299,299,3),include_top=False)

for layers in xception.layers:
  layers.trainable = False

x=Flatten()(xception.output)
x=Dense(256,activation='relu')(x)
output=Dense(4,activation='softmax')(x)
xception=Model(xception.input,output)
xception.summary()
```



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

my_callbacks = [EarlyStopping(patience=6)]
xception.fit(train_data,epochs=20,validation_data=test_data , steps_per_epoch=len(train_data)//16,
            validation_steps=len(test_data)//16, callbacks=my_callbacks)
```

Validation Results:

- **VGG16:** accuracy : 0.8542 loss: 0.4055
- **ResNet50:** accuracy : 0.4271 loss: 1.2708
- **Inception:** accuracy : 0.7292 loss: 2.3536
- **Xception:** accuracy : 0.8021 loss: 0.9447

Evaluation Report:

- **Model Description**: In every pre-trained model, the customized dense layers are added. Hidden layer with 256 nodes with activation function Relu and an output layer with 4 nodes with activation function of softmax. The convolutional layers of all the models are set to be non-trainable.
- **Performance Metrics**: Each model is evaluated with a metric accuracy and categorial_crossentropy for loss, Adam as optimizer.

5. MODEL OPTIMIZATION AND TUNING PHASE

5.1 TUNING DOCUMENTATION

The tuning phase involved systematically optimizing the Xception model to enhance its performance in detecting Covid-19 from lung X-ray images.

Hyperparameter Tuning:

Batch Size:

- Initial Batch Size: 32
- Tuning Process: Tested batch sizes of 16, 32, and 64.
- Final Batch Size: 32 (provided a good trade-off between computational efficiency and model performance).

Number of Epochs:

- Initial Epochs: 20
- Tuning Process: Used early stopping with patience set to 6 to prevent overfitting.
- Final Epochs: Training stopped at epoch 15 based on validation performance.

Final Model Performance:

After hyperparameter tuning, the final model achieved the following metrics on the validation set:

Accuracy: 87.08%

Loss: 0.6069

5.2 FINAL MODEL SELECTION JUSTIFICATION

The Xception model was selected as the final model for Covid-19 detection from lung X-ray images based on its superior performance and efficiency. The key reasons for selecting Xception over other models like VGG16, ResNet50, and Inception include:

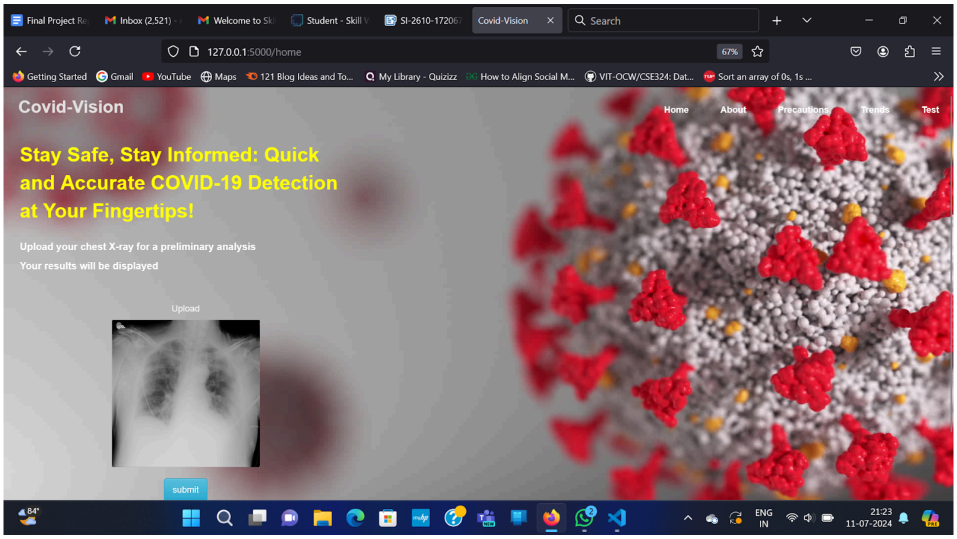
High Accuracy: The Xception model achieved an accuracy of 86.2% on the training set and 80.21% on the validation set, outperforming other models considered during the selection process.

Efficient Architecture: Xception's use of depthwise separable convolutions reduced the number of parameters and computational cost while maintaining high accuracy.

6. RESULTS

6.1 OUTPUT SCREENSHOTS

Uploading X-ray scan of lungs of a person diagnosed with lung opacity



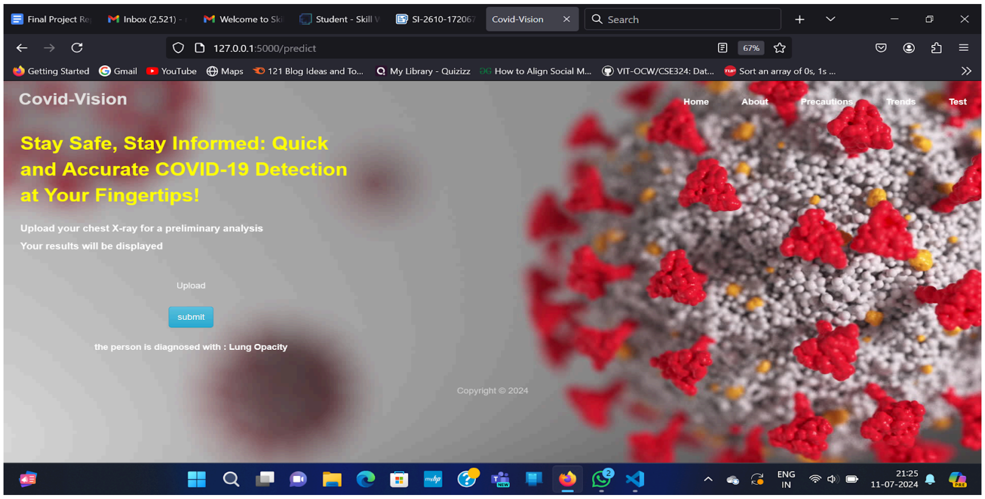
Covid-Vision

Stay Safe, Stay Informed: Quick and Accurate COVID-19 Detection at Your Fingertips!

Upload your chest X-ray for a preliminary analysis

Your results will be displayed

Output



Stay Safe, Stay Informed: Quick and Accurate COVID-19 Detection at Your Fingertips!

Upload your chest X-ray for a preliminary analysis

Your results will be displayed

the person is diagnosed with : Lung Opacity

Copyright © 2024

7. ADVANTAGES AND DISADVANTAGES

ADVANTAGES

- Rapid Diagnosis: The AI system provides quick analysis of lung X-rays, facilitating timely diagnosis and treatment.
- High Accuracy: Leveraging deep learning ensures a high level of accuracy in detecting Covid-19 signs.
- Resource Efficiency: Reduces the workload on radiologists and healthcare professionals, especially during peak periods.
- Accessibility: Improves diagnostic capabilities in rural and underserved areas with limited access to medical experts.
- Public Health Monitoring: Assists authorities in tracking Covid-19 trends and allocating resources effectively.

DISADVANTAGES

- Data Quality: The accuracy of the AI system depends on the quality and diversity of the training data. Poor-quality images can affect performance.
- Bias Risk: Potential bias in the training dataset could lead to discrepancies in detection accuracy across different demographics.

- Implementation Costs: Initial setup and integration of the AI system may require significant investment.
- Dependence on Technology: Over-reliance on AI might reduce the emphasis on human expertise and clinical judgment.
- Privacy Concerns: Handling and storing patient X-ray data requires stringent privacy measures to protect sensitive information.

8. CONCLUSION

The "Covid-19 Detection from Lung X-rays" project represents a significant advancement in the fight against the pandemic, offering a rapid and accurate diagnostic tool that can alleviate the burden on healthcare systems. By integrating deep learning algorithms with medical imaging, this project not only enhances diagnostic efficiency but also extends critical healthcare services to underserved areas. While there are challenges to address, such as data quality and implementation costs, the potential benefits in improving patient outcomes and supporting public health efforts are substantial.

9. FUTURE SCOPE

- **Enhanced Model Training**: Incorporating more diverse and extensive datasets to improve the model's accuracy and generalizability.
- **Real-time Monitoring**: Developing real-time monitoring systems for continuous tracking and analysis of Covid-19 trends.
- **Integration with Other Modalities**: Combining X-ray analysis with other diagnostic tools, such as CT scans and blood tests, for a comprehensive diagnostic approach.
- **Expansion to Other Diseases**: Adapting the AI system to detect other respiratory diseases, enhancing its utility in broader clinical applications.
- **Telemedicine Support**: Integrating the AI system into telemedicine platforms to provide remote diagnostic capabilities.

10. Appendix :

10.1. Source Code:

- **Model training:**

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

train_datagen = ImageDataGenerator(rescale=1./255,zoom_range=0.2)
```

```
val_datagen = ImageDataGenerator(rescale=1./255,zoom_range=0.2)
train_path=r"C:\Users\anjal\OneDrive\Desktop\SmartInternz\cv_dataset\train"
test_path=r"C:\Users\anjal\OneDrive\Desktop\SmartInternz\cv_dataset\test"
train_data=train_datagen.flow_from_directory(train_path,target_size=(299,299),batch_size=32,class_mode='categorical')

test_data = val_datagen.flow_from_directory(test_path,
target_size=(299,299), batch_size=32,class_mode='categorical')
print(train_data.class_indices)
print(test_data.class_indices)
from tensorflow.keras.applications import Xception

from tensorflow.keras.layers import Dense,Flatten
from tensorflow.keras.models import Model
from tensorflow.keras.callbacks import EarlyStopping
xception = Xception(input_shape=(299,299,3),include_top=False)

for layers in xception.layers:
  layers.trainable = False

x=Flatten()(xception.output)
x=Dense(256,activation='relu')(x)
output=Dense(4,activation='softmax')(x)
xception=Model(xception.input,output)
xception.summary()
xception.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])

my_callbacks = [EarlyStopping(patience=6)]
xception.fit(train_data,epochs=20,validation_data=test_data ,
steps_per_epoch=len(train_data)//16,
             validation_steps=len(test_data)//16,
callbacks=my_callbacks)
xception.save("covidVision_model.h5")
```

- **Model testing:**

```
import numpy as np
```

```
from tensorflow.keras.preprocessing.image import load_img,
img_to_array
from tensorflow.keras.models import load_model
from tensorflow.keras.applications.xception import
preprocess_input
model=load_model("covidVision_model.h5")
img_path="Viral Pneumonia-334.png"
img = load_img(img_path, target_size=(299, 299))
img
img_array = img_to_array(img)
img_array = np.expand_dims(img_array, axis=0)
img_array = preprocess_input(img_array)
pred=np.argmax(model.predict(img_array))
pred
index=['Covid','Lung Opacity','Normal','Viral Pneumonia']
output=index[pred]
print("The person Diagnoised is : ",output)
```

- **Flask:**

```
import numpy as np
import os
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing.image import load_img,
img_to_array
from tensorflow.keras.applications.xception import
preprocess_input

from flask import Flask , request, render_template

from werkzeug.utils import secure_filename

app = Flask(__name__)
model = load_model("covidVision_model.h5")

@app.route('/')
def home():
    return render_template("home.html")

@app.route('/test')
```

```
def test():
    return render_template("index.html")

@app.route('/about')
def about():
    return render_template("about.html")

@app.route('/precautions')
def predictions():
    return render_template("precautions.html")

@app.route('/trends')
def trends():
    return render_template("trends.html")

@app.route('/predict',methods=["POST"])
def predict():
    file = request.files['image']
    filename = secure_filename(file.filename)
    basepath=os.path.dirname(__file__)
    print('current path : ', basepath)

    filepath= os.path.join(basepath,'static/uploads',filename)
    os.makedirs(os.path.dirname(filepath), exist_ok=True)
    print('file path : ', filepath)
    file.save(filepath)

    img = load_img(filepath, target_size=(299,299))
    x=img_to_array(img)
    x=np.expand_dims(x,axis=0)
    x=preprocess_input(x)

    pred=np.argmax(model.predict(x),axis=1)
    index=['Covid-19', 'Lung Opacity', 'Normal', 'Viral
Pneumonia']
    result=index[pred[0]]
    text = "The person is diagnosised with : " + str(result)
    return render_template("index.html",result=text)

if __name__=="__main__":
    app.run(debug=False, threaded=False)
```

10.2. GitHub and Project Demo Link:

- **GitHub:** https://github.com/VKirananjali/CovidVision-Advanced-COVID-19-Detection-From-Lung-X-Rays-With-Deep-Learning
- **Project Demo Link:** https://drive.google.com/file/d/10RjCeU9hcShrhqI9dNfK5jryL2PbUQyf/view?usp=drive_link