PNEUMONIA LUNGS DISEASE PREDICTION WITH DEEP LEARNING

Srujana Konda - 700740479

Tejaswi Reddy Marri - 700742730

Revanth Bharadwaj Tarimela Roopavataram - 700743276

Sai Pavan Reddy Pogula - 700741739

Abstract:

This study focuses on developing a deep learning algorithm to predict Pneumonia, a respiratory disease that affects the lungs' alveoli. The research utilizes patient records with different attributes related to the disease and passes them into the algorithm for analysis. Pneumonia is typically caused by viral or bacterial infections, and the severity of the condition varies, with symptoms including coughing, chest pain, fever, and difficulty breathing. Risk factors for Pneumonia include conditions such as cystic fibrosis, asthma, and a weak immune system. The diagnosis of Pneumonia is usually based on symptoms and physical examination, with additional tests such as chest X-rays, blood tests, and sputum cultures used to confirm the diagnosis.

The study focuses on developing a deep learning algorithm to predict Pneumonia based on the patient's medical reports. The algorithm classifies and divides the data into segments, each containing information on whether the patient has Pneumonia or not. The algorithm is trained to analyze data related to other diseases related to Pneumonia to predict the output accurately. The research is crucial for hospitals, as the ability to predict Pneumonia accurately can help in diagnosing and treating patients more effectively.

In conclusion, this study emphasizes the importance of developing accurate and reliable algorithms for predicting Pneumonia. The research utilizes deep learning algorithms to analyze patient data and predict the presence or absence of the disease accurately. The results of this study can be instrumental in improving patient care and medical diagnosis in hospitals.

Keywords : 1]Pneumonia ,2]Respiratory disease ,3]Deep learning algorithm , 4]Chest X-ray ,5]Contamination ,6]Medical reports ,7]Predictive analysis

Introduction

Pneumonia is a respiratory disease that affects the human body, primarily the lungs. It is a condition where the alveoli, the small air sacs in the lungs, become inflamed. Symptoms usually include a dry or productive cough, chest pain, fever, and difficulty breathing. The severity of the condition can vary and can be caused by viral or bacterial infections, among other things.

Pneumonia is a serious health concern that can have significant consequences if left untreated. It can be caused by a variety of factors, including immune system diseases, cystic fibrosis, chronic obstructive pulmonary disease (COPD), asthma, diabetes, and cardiovascular failure. Other risk factors include a history of smoking, an inability to cough properly, and a weak immune system.

Diagnosing pneumonia typically involves a physical examination, as well as a chest X-ray, blood tests, and a culture of sputum. The diagnosis can also be made by determining where the infection was acquired. Treatment usually involves antibiotics or antiviral medication, as well as supportive care.

Given the seriousness of pneumonia, hospitals and healthcare providers are constantly looking for ways to improve diagnosis and treatment. In recent years, deep learning algorithms have emerged as a promising tool for detecting and diagnosing pneumonia. These algorithms use machine learning techniques to analyze large amounts of data and make predictions based on patterns and trends in the data.

The use of deep learning algorithms for pneumonia detection is particularly relevant given the large volume of patient data that hospitals and healthcare providers must manage. By analyzing this data using machine learning algorithms, healthcare providers can gain insights into patient health and make more informed decisions about treatment options.

In this work, we explore the use of deep learning algorithms for detecting and diagnosing pneumonia. We analyze patient records, including chest X-rays and other medical data, and pass this information through a machine learning algorithm. Our goal is to develop an accurate and reliable algorithm for detecting pneumonia, which can help healthcare providers make more informed decisions about treatment options.

To achieve our goal, we will be using Google Colab Python Tool, a cloud-based development environment that allows us to develop and run the application directly inside the cloud server. This eliminates the need for any specific software to be installed on the user's system, making it accessible to anyone with an internet connection. We will use the machine learning algorithm libraries built inside Colab to train and test our algorithm, and evaluate the accuracy of our results.

Motivation:

Pneumonia is a respiratory disease that poses a significant health threat to individuals worldwide. This potentially life-threatening lung infection affects millions of people each year, and early detection and accurate diagnosis are essential for effective treatment and improved patient outcomes. Unfortunately, the lack of experienced radiologists and the delay in reaching a conclusion often lead to treatment deferral, which can lead to severe consequences.

The mortality rate associated with lung illnesses can be reduced by early detection and prompt treatment. The severity of the condition can be predicted by identifying it in the early stages, allowing healthcare providers to plan treatment accordingly and improve patient outcomes. The importance of accurate and timely diagnosis in managing pneumonia cannot be overstated.

Pneumonia is a condition that can affect anyone, from young children to older adults. The symptoms of pneumonia typically include a combination of a productive or dry cough, chest pain, fever, and difficulty breathing. The severity of the symptoms varies depending on the underlying cause and the individual's overall health. Early recognition of symptoms and timely medical intervention can help prevent the progression of the disease and reduce the risk of complications.

Efficient diagnosis and treatment of pneumonia is crucial for minimizing morbidity and mortality rates associated with this respiratory disease. Radiology plays a critical role in the detection and diagnosis of pneumonia, as it allows for non-invasive visualization of the lungs. However, the shortage of experienced radiologists poses a significant challenge in providing timely and accurate diagnosis.

In conclusion, pneumonia is a severe and potentially life-threatening respiratory disease that requires timely and accurate diagnosis for effective treatment and improved patient outcomes. Early detection and accurate diagnosis can reduce the mortality rate associated with lung illnesses, and identifying it in the early stages can help predict the severity of the condition. The shortage of experienced radiologists highlights the need for developing efficient diagnostic tools to facilitate timely and accurate diagnosis of pneumonia.

Main Contribution and Objectives:

Project Execution Plan:

The objective of Pneumonia detection is to detect the Pneumonia news in the patient records. In this work, the dataset containing the Pneumonia chest X-ray images will be taken into consideration. The pre-processing will be applied to the dataset and the noisy and null value data will be removed from the dataset. After that, the data will be analyzed and visualized for further processing. The machine learning algorithm will be chosen to make the prediction.

Contribution:

The Pneumonia respiratory disease prediction will be the Python-based application that contributes to finding the Pneumonia in the patient image records. It will help predict Pneumonia in the early stage and to take the medicine at the proper time. [3][4]

Objective:

The objective of pneumonia lung disease prediction is to predict pneumonia from chest X-ray images. A set of x-ray images of patients with pneumonia is taken for the training and x-ray images of test images are also taken and finally, the results are predicted.

Evaluation:

The project evaluation can be tested with the deep learning algorithm prediction results. Since the deep learning algorithm will be used to predict Pneumonia, the accuracy of the algorithm result will be helpful to evaluate the results. The accuracy score of the algorithm in pneumonia detection helps to evaluate the dataset.

The application will be developed with Google Colab Python Tool as the project can be directly executed in any type of computer system with an internet connection. There is no need for any specific software to be installed in the user system. The Colab Tool helps to develop and run the application directly inside the cloud server where the Python library files are installed. The machine learning algorithm libraries are built inside the Colab. It helps the project use the deep learning algorithm to predict Pneumonia.

ALGORITHM:

DEEP LEARNING ALGORITHM: CONVOLUTIONAL NEURAL NETWORKS (CNNS)

Significance:

The main significance of the project is With the fast improvement of simulated intelligence methods, PC-helped Finding has drawn much consideration and has been effectively conveyed in numerous uses of medical services and clinical analysis. The amazing exhibition owes to the brilliant expressiveness and versatility of the brain organizations, albeit the models' instinct mostly can not be addressed unequivocally. Interpretability is, in any case, vital, even equivalent to the determination accuracy, for PC-supported analysis. [1][6]

Features:

Pneumonia is a respiratory contamination brought about by microbes, infections, or parasites, and it has been known as a very normal and possibly lethal illness in the past two centuries. Inspired by the finding system of human specialists, we join the clinical perception with clinical pictures. propose a requirement-based calculation that consolidates clinical information to fabricate a sensible convolutional neural network.[6][7]

Dataset:

The dataset contains lots of noisy information also. But with feature engineering, I will get better results. The first step is to import the libraries and load data. After that will take a basic understanding of the data like its shape, and sample if there are any NULL values present in the dataset. Understanding the data is an important step for prediction or any machine learning project. It is good that there are no NULL values.

The Pneumonia chest x-ray images of training and test images are taken as the dataset.

Detailed Design of Features:

This dataset contains the fields needed for analyzing the prediction of pneumonia disease in the lungs.

The exploratory examination is a cycle to investigate and comprehend the information and information related in a total profundity with the goal that it makes highlight designing and Deep Learning demonstrating steps smoothly and smoothed out for expectation. The exploratory examination assists with validating our presumptions or misleading.

The dataset is organized into 3 folders (train, test, value) and contains subfolders for each image category (Pneumonia/Normal). There are 5,863 X-Ray images (JPEG) a nd 2 categories (Pneumonia/Normal).

Chest X-ray images (anterior-posterior) were selected from retrospective cohorts of p ediatric patients
one to five years old from Guangzhou Women and Children's Medical Center, Guang zhou. All chest X-ray imaging was performed as part of the
patient's routine clinical care.

For the analysis of chest X-ray images, all chest radiographs were initially screened f or quality control by removing all low-quality or unreadable scans. The diagnoses for the images were then graded by two expert physicians before being cleared for traini ng the AI system. In order to account for any grading errors, the evaluation set was a lso checked by a third expert.

Analysis of pneumonia disease prediction:

It will begin from the principal segment and investigate every section and comprehend what influence it makes on the objective segment. At the necessary step, we will likewise perform preprocessing and include designing undertakings.

The point in acting top to bottom exploratory examination is to get ready and clean information for better Convolutional neural networks demonstrating to accomplish elite execution and summed up models. So it should begin with breaking down and setting up the dataset for expectation.

Modules:

1) Dataset collection
2) Data cleaning
3) Exploratory Image Analysis
4) Convolutional neural networks Modeling
5) Report

1) Dataset collection:

The information about the pneumonia disease with different types of chest x-ray images data are collected from different types of patients for the training and testing.

2) Data Cleaning:

The large dataset contains more noisy and improper data which have to be pre-processed to produce the quality dataset for further pruning. The dataset is cleaned and processed with the initial stage of removing the null values.

3) Exploratory Image Analysis

Exploratory analysis is a process to explore and understand the data and data relationship in a complete depth so that it makes feature engineering and machine learning modeling steps smooth and streamlined for prediction. It helps to prove our assumptions true or false. In other words, it helps to perform hypothesis testing.

4) Convolutional neural networks Modeling

Convolutional neural networks modeling helps to find the best algorithm with the best hyper parameters to achieve maximum accuracy. The dataset is split into 2 variants. 50% of image records are taken as training data and used to train the neural networks algorithm. The remaining 50% of images is applied to testing which helps to predict the process.

5) Report:

The Data is visualized based on the output of the convolutional neural networks algorithm and the data is mapped with different types of graphs to analyze and visualize the exact data to the user for the prediction. Marplot libraries are implemented to map the results based on the user requirements.

Load Packages:

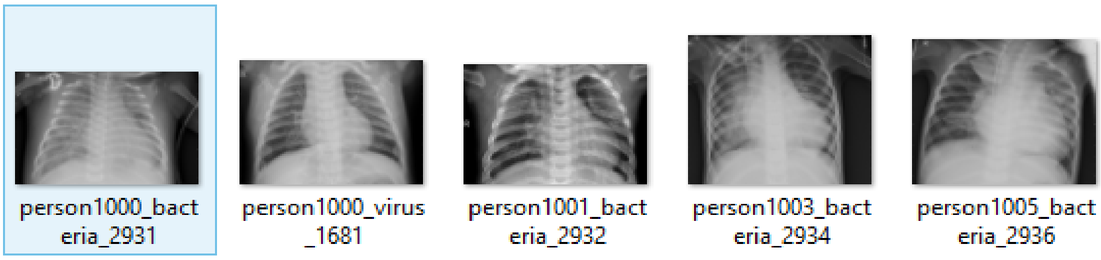
First step have to import the necessary packages to the application

IMPORT PANDAS AS PD

```
IMPORT NUMPY AS NP
IMPORT MATPLOTLIB.PYPLOT AS PLT
IMPORT SEABORN AS SNS
IMPORT KERAS
FROM KERAS.MODELS IMPORT SEQUENTIAL
FROM KERAS.LAYERS IMPORT DENSE, CONV2D , MAXPOOL2D , FLATTEN , D
ROPOUT , BATCH NORMALIZATION
FROM KERAS.PREPROCESSING.IMAGE IMPORT IMAGEDATAGENERATOR
FROM SKLEARN.MODEL_SELECTION IMPORT TRAIN_TEST_SPLIT
FROM SKLEARN.METRICS IMPORT CLASSIFICATION_REPORT,CONFUSION_M
ATRIX
FROM KERAS.CALLBACKS IMPORT REDUCELRONPLATEAU
IMPORT CV2
IMPORT OS
```

The information has an extremely straightforward design with elements. Each folder is related to test and train data.

Sample train images of pneumonia-affected patients:



Sample train images of normal patients:

Pre-processing:

The noisy data and empty values in the dataset are pre-processed. The images which are not needed for the evaluation of the model are also removed.

```
l = []
for i in train:
    if(i[1] == 0):
        l.append("Pneumonia")
    else:
        l.append("Normal")
sns.set_style('darkgrid')
```

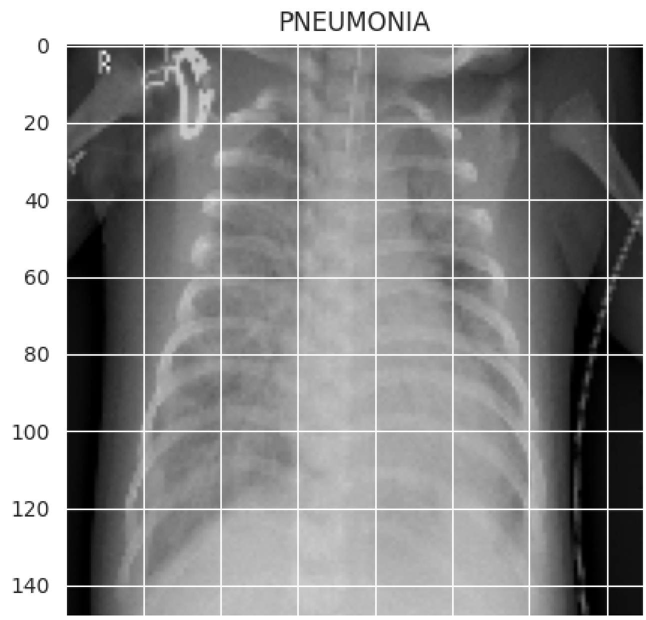
The count of images in each folder of pneumonia and normal patient records are taken.

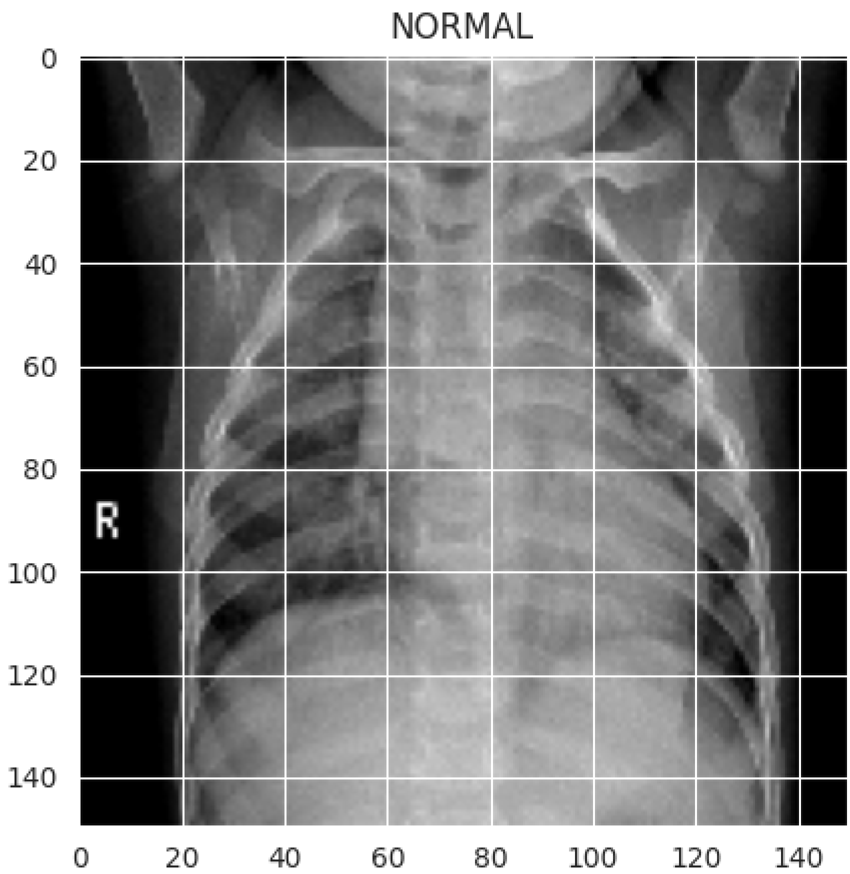
Previewing the images of Normal and affected classes:

```
plt.figure(figsize = (5,5))
plt.imshow(train[0][0], cmap='gray')
plt.title(labels[train[0][1]])

plt.figure(figsize = (5,5))
plt.imshow(train[-1][0], cmap='gray')
plt.title(labels[train[-1][1]])
```

Results:




NORMAL
R
0
20
40
60
80
100
120
140
0
20
40
60
80
100
120
140

Implementation And Status Report:

Name: Srujana Konda

Responsibility: Normalize the data

Status: Completed

Description: The first step in data preprocessing is to normalize the dataset. The normalization process involves converting the image data to a standard format that can be used by the deep learning algorithm. One way to normalize the data is to perform grayscale normalization. Grayscale normalization reduces the effect of illumination differences between images. It is also recommended to convert the image data to a [0..1] range as this can help the CNN converge faster during training.

Name: Tejaswi Reddy Marri

Responsibility: Resize data for deep learning

Status: Completed

Description: The next step in data preprocessing is to resize the images to a standard size. The resizing process involves changing the size of the image to a fixed dimension so that it can be used by the deep learning algorithm. The images should be resized to a size that is suitable for the model's architecture. The recommended size for the input images is 224 x 224 pixels. Resizing the images to a standard size can improve the performance of the deep learning model.

Name: Sai Pavan Reddy Pogula

Responsibility: Data Augmentation

Status: Completed

Description: Data augmentation is an important step in deep learning as it allows us to expand our dataset artificially. This helps to avoid overfitting and can improve the generalization performance of the model. The goal of data augmentation is to create new examples of the training data by applying different transformations to the original images. Some common transformations include rotation, zooming, shifting, and flipping. By applying these transformations, we can create new examples of the training data and make our model more robust.

Name:Revanth Bharadwaj Tarimela Roopavataram

Responsibility: Training and implementation of the model

Status: Completed

Description: The final step in the process is to train the deep learning model. The model should be implemented using a suitable architecture such as Convolutional Neural Networks (CNNs). The model should be designed to handle image classification tasks. It is recommended to use SAME padding to ensure that the input image is fully covered by the filter and specified stride. After training, the model should be evaluated using the testing dataset to check its accuracy and performance. If the performance is not satisfactory, the model should be fine-tuned by adjusting the hyperparameters and re-training.

Implementation of Machine Learning models:

The dataset is divided into training and testing data. The split dataset is passed into the different deep learning algorithm models and the accuracy levels were found.

Normalize the data:

Perform a grayscale normalization to reduce the effect of illumination differences. Moreover, the CNN converges faster on [0..1] data than on [0..255]

Resize data for deep learning:

The train and test dataset is resized with the pre-processing of the images for the evolution of the deep learning model.

Data Augmentation:

To avoid the overfitting problem, we need to artificially expand our dataset. We can make your existing dataset even larger. The idea is to alter the training data with small transformations to reproduce the variations.

Approaches that alter the training data in ways that change the array representation while keeping the label the same are known as data augmentation techniques. Some popular augmentations people use are grayscales, horizontal flips, vertical flips, random crops, color jitters, translations, rotations, and much more.

By applying just a couple of these transformations to our training data, we can easily double or triple the number of training examples and create a very robust model.

For the data augmentation:

1. Randomly rotate some training images by 30 degrees
2. Randomly zoom by 20% some training images
3. Randomly shift images horizontally by 10% of the width
4. Randomly shift images vertically by 10% of the height
5. Randomly flip images horizontally.

Once the model is made, next fit the dataset to training.

Training and implementation of the model:

SAME Padding: it applies padding to the input image so that the input image gets fully covered by the filter and specified stride. It is called SAME because, for stride 1, the output will be the same as the input.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 150, 150, 32)      320

 batch_normalization (BatchN  (None, 150, 150, 32)     128
 ormalization)

 max_pooling2d (MaxPooling2D  (None, 75, 75, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 75, 75, 64)        18496

 dropout (Dropout)           (None, 75, 75, 64)        0

 batch_normalization_1 (Batc  (None, 75, 75, 64)       256
 hNormalization)

 max_pooling2d_1 (MaxPooling  (None, 38, 38, 64)       0
 2D)
```

The above model trains and tests the x-ray chest images with the help of convolutional neural networks.

```
[11] learning_rate_reduction = ReduceLROnPlateau(monitor='val_accuracy', patience = 2, verbose=1,factor=0.3, min_lr=0.000001)
```

```
[12] history = model.fit(datagen.flow(x_train,y_train, batch_size = 32) ,epochs = 12 , validation_data = datagen.flow(x_val, y_val) ,callbacks = [learning_rate_reduct
```

Results:

```
Epoch 1/12
1/1 [==============================] - 4s 4s/step - loss: 0.8185 - accuracy: 0.6000 - val_loss: 0.6934 - val_accuracy: 0.5000 - lr: 0.0010
Epoch 2/12
1/1 [==============================] - 1s 1s/step - loss: 12.8000 - accuracy: 0.5000 - val_loss: 0.7177 - val_accuracy: 0.5000 - lr: 0.0010
Epoch 3/12
1/1 [==============================] - ETA: 0s - loss: 14.5727 - accuracy: 0.5000
Epoch 3: ReduceLROnPlateau reducing learning rate to 0.0003000000142492354.
1/1 [==============================] - 1s 1s/step - loss: 14.5727 - accuracy: 0.5000 - val_loss: 0.7555 - val_accuracy: 0.5000 - lr: 0.0010
Epoch 4/12
1/1 [==============================] - 1s 916ms/step - loss: 1.2016 - accuracy: 0.8000 - val_loss: 0.7142 - val_accuracy: 0.5000 - lr: 3.0000e-04
Epoch 5/12
1/1 [==============================] - ETA: 0s - loss: 0.9349 - accuracy: 0.9000
Epoch 5: ReduceLROnPlateau reducing learning rate to 9.000000427477062e-05.
1/1 [==============================] - 1s 992ms/step - loss: 0.9349 - accuracy: 0.9000 - val_loss: 0.6812 - val_accuracy: 0.5000 - lr: 3.0000e-04
Epoch 6/12
1/1 [==============================] - 1s 935ms/step - loss: 0.7690 - accuracy: 0.8000 - val_loss: 0.6850 - val_accuracy: 0.6000 - lr: 9.0000e-05
Epoch 7/12
1/1 [==============================] - 2s 2s/step - loss: 0.0935 - accuracy: 1.0000 - val_loss: 0.6353 - val_accuracy: 0.8000 - lr: 9.0000e-05
Epoch 8/12
1/1 [==============================] - 2s 2s/step - loss: 0.6091 - accuracy: 0.7000 - val_loss: 0.6743 - val_accuracy: 0.5000 - lr: 9.0000e-05
Epoch 9/12
1/1 [==============================] - ETA: 0s - loss: 0.5968 - accuracy: 0.9000
Epoch 9: ReduceLROnPlateau reducing learning rate to 2.700000040931627e-05.
1/1 [==============================] - 1s 843ms/step - loss: 0.5968 - accuracy: 0.9000 - val_loss: 0.7725 - val_accuracy: 0.5000 - lr: 9.0000e-05
Epoch 10/12
1/1 [==============================] - 1s 878ms/step - loss: 0.8542 - accuracy: 0.8000 - val_loss: 0.9045 - val_accuracy: 0.5000 - lr: 2.7000e-05
Epoch 11/12
```

Finding the accuracy of the model:

```
[13] print("Loss of the model is - " , model.evaluate(x_test,y_test)[0])
     print("Accuracy of the model is - " , model.evaluate(x_test,y_test)[1]*100 , "%")
```

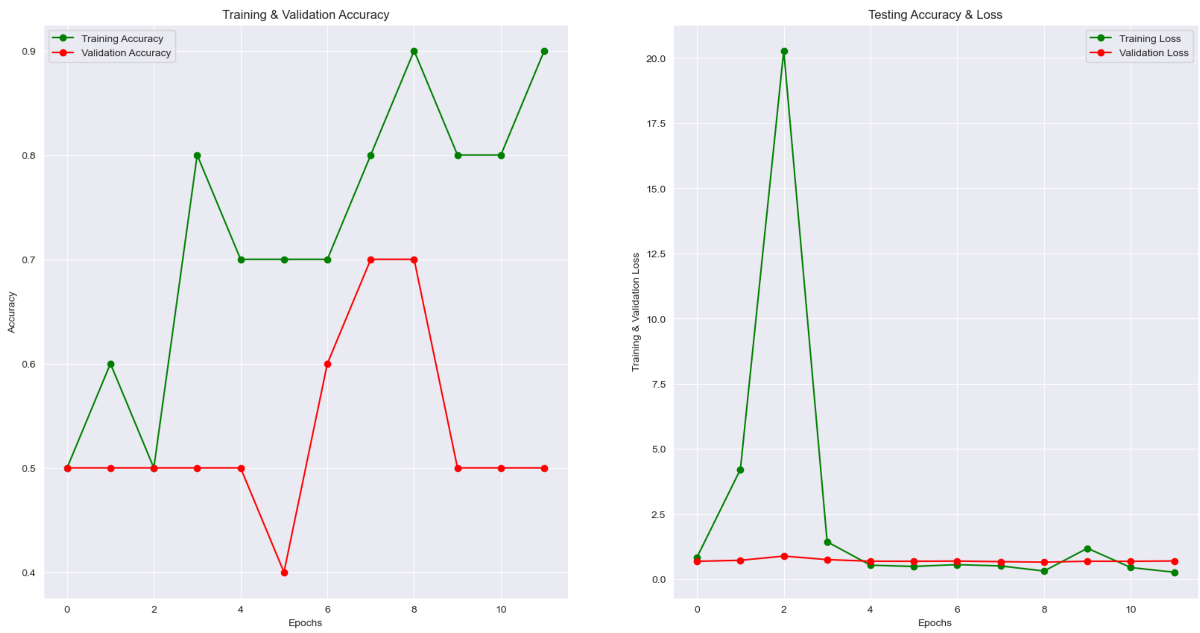
Results of the accuracy:

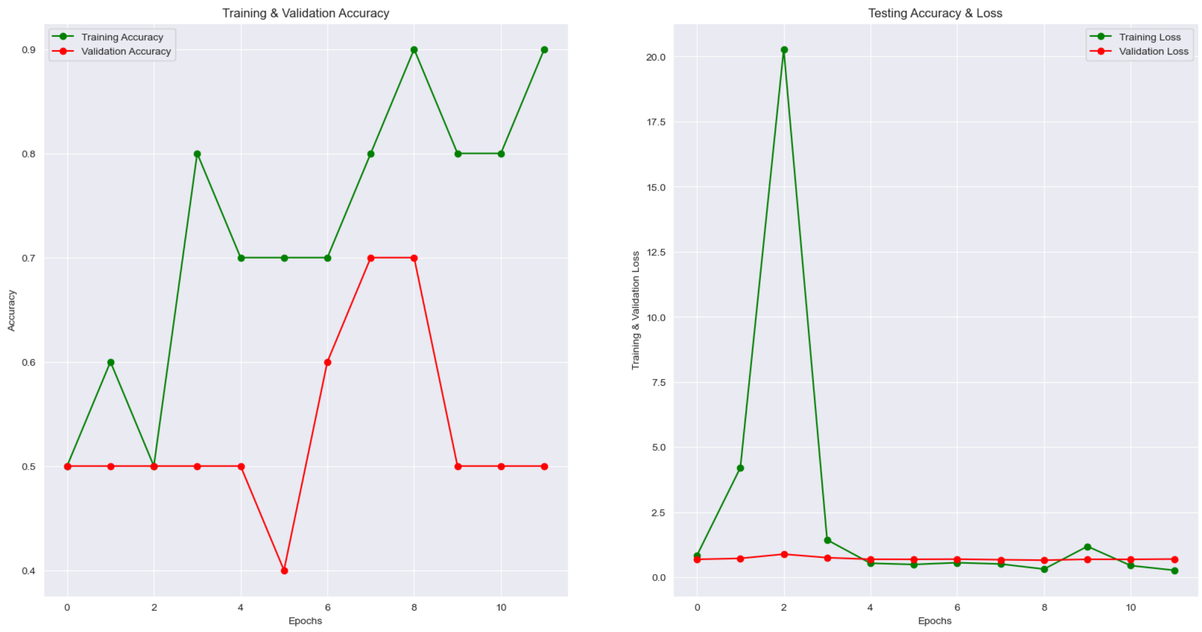
```
1/1 [==============================] - 0s 297ms/step - loss: 1.1859 - accuracy: 0.5000
Loss of the model is -  1.1859419345855713
1/1 [==============================] - 0s 286ms/step - loss: 1.1859 - accuracy: 0.5000
Accuracy of the model is -  50.0 %
```

Analysis after Model Training

```
[14] epochs = [i for i in range(12)]
     fig , ax = plt.subplots(1,2)
     train_acc = history.history['accuracy']
     train_loss = history.history['loss']
     val_acc = history.history['val_accuracy']
     val_loss = history.history['val_loss']
     fig.set_size_inches(20,10)

     ax[0].plot(epochs , train_acc , 'go-' , label = 'Training Accuracy')
     ax[0].plot(epochs , val_acc , 'ro-' , label = 'Validation Accuracy')
     ax[0].set_title('Training & Validation Accuracy')
     ax[0].legend()
     ax[0].set_xlabel("Epochs")
     ax[0].set_ylabel("Accuracy")
```





The above graph depicts the training accuracy with the validation accuracy of the convolutional neural network.

Predictions

```
predict_x=model.predict(x_test)
predictions=np.argmax(predict_x,axis=1)


predictions = predictions.reshape(1,-1)[0]
predictions[:15]
```

The predictions of the model are evaluated by bypassing the testing and training images of the dataset.

The results of prediction results are displayed to show the precision, recall, f1-score, and support.

```
[18] print(classification_report(y_test, predictions, target_names = ['Pneumonia (Class 0)','Normal (Class 1)']))

                     precision    recall  f1-score   support

Pneumonia (Class 0)       0.50      1.00      0.67         5
   Normal (Class 1)       0.00      0.00      0.00         5

           accuracy                           0.50        10
          macro avg       0.25      0.50      0.33        10
       weighted avg       0.25      0.50      0.33        10
```

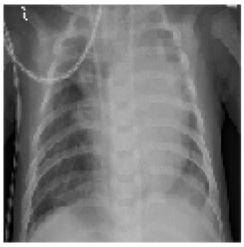
Results of the predicted class:

The predicted class and the actual class x-ray images are displayed.

The results also show the incorrectly predicted classes.

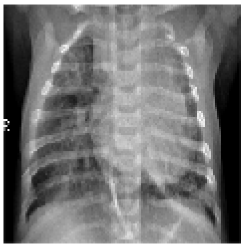
Predicted Class 0,Actual Class 0

Predicted Class 0,Actual Class 0

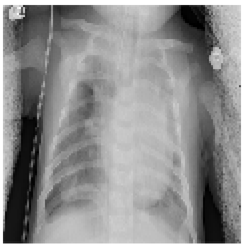


Predicted Class 0,Actual Class 0

Predicted Class 0,Actual Class 0



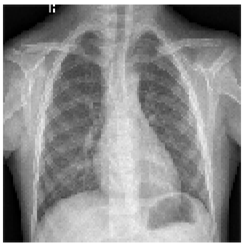
Predicted Class 0,Actual Class 0



The predicted class and the actual class x-ray images are displayed.

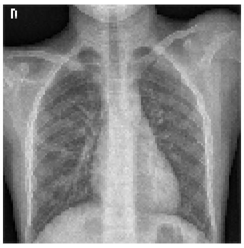
The results also show the incorrectly predicted classes.

Predicted Class 0,Actual Class 1



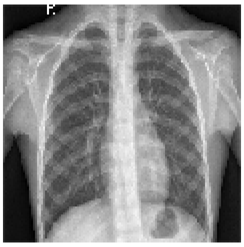
Predicted Class 0,Actual Class 1

Predicted Class 0,Actual Class 1



Predicted Class 0,Actual Class 1

Predicted Class 0,Actual Class 1



References/Bibliography:

[1]. Hao ren2,1∗ aslan b. wong1∗, wanmin lian3∗, weibin cheng2, ying zhang1, jianwei he1, qingfeng liu4, jiasheng yang4, chen zhang5, (member, ieee), kaishun wu1, (member, ieee), haodi zhang1," DOI: 10.1109/ACCESS.2021.3090215, IEEE Access.

[2]. Xuelin Qian, Student Member, IEEE, Huazhu Fu, Senior Member, IEEE, Weiya Shi, Tao Chen, Member, IEEE, Yanwei Fu∗, Member, IEEE, Fei Shan∗, and Xiangyang Xue∗, Member, IEEE "© IEEE 2020.

[3] NCIRD, "https://www.cdc.gov/coronavirus/2019-ncov/lab/guidelinesclinical-specimens.html," 2020.

[4] T. Ai, Z. Yang, H. Hou et al., "Correlation of chest ct and rt-pcr testing in coronavirus disease 2019 (covid-19) in china: a report of 1014 cases," Radiology, p. 200642, 2020.

[5] M. Chung, A. Bernheim, X. Mei et al., "CT imaging features of 2019 novel coronavirus (2019-nCoV)," Radiology, vol. 295, no. 1, pp. 202– 207, 2020.

[6] Y. Pan and H. Guan, "Imaging changes in patients with 2019-nCov," European Radiology, feb 2020.

[7] Y. Fang, H. Zhang, J. Xie, M. Lin, L. Ying, P. Pang, and W. Ji, "Sensitivity of chest ct for covid-19: comparison to rt-pcr," Radiology, p. 200432, 2020.

[8] J. P. Kanne, B. P. Little, J. H. Chung, B. M. Elicker, and L. H. Ketai, "Essentials for radiologists on covid-19: an update radiology scientific expert panel," 2020.

[9] M. Liu, W. Zeng, Y. Wen, Y. Zheng, F. Lv, and K. Xiao, "COVID19 pneumonia: CT findings of 122 patients and differentiation from influenza pneumonia," European Radiology, p. 1, 2020.

[10] W. Chen, X. Xiong, B. Xie et al., "Pulmonary invasive fungal disease and bacterial pneumonia: a comparative study with high-resolution CT," American journal of translational research, vol. 11, no. 7, p. 4542, 2019.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in Advances in neural information processing systems, 2012, pp. 1097–1105.

[12] X. Wang, R. Girshick, A. Gupta, and K. He, “Non-local neural networks,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 7794–7803.

[13] X. Qian, Y. Fu, T. Xiang, Y.-G. Jiang, and X. Xue, “Leader-based multi-scaleattentiondeeparchitectureforpersonre-identification,”IEEE transactions on pattern analysis and machine intelligence, vol. 42, no. 2, pp. 371–385, 2019.

[14] W. Wang, Y. Fu, X. Qian, Y.-G. Jiang, Q. Tian, and X. Xue, “Fm2unet: Face morphological multi-branch network for makeup-invariant face verification,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 5730–5740.

[15] K. Chen, J. Pang, J. Wang et al., “Hybrid task cascade for instance segmentation,” in IEEE conference on computer vision and pattern recognition, 2019, pp. 4974–4983.

[16] https://colab.research.google.com/

[17] https://www.tutorialspoint.com/google_colab/what_is_google_colab.htm

[18] https://www.codingforentrepreneurs.com/courses/python-google-colab-sheets-drive/