PNEUMONIA LUNGS DISEASE PREDICTION WITH DEEP LEARNING

Srujana Konda - 700740479

Tejaswi Reddy Marri - 700742730

Revanth Bharadwaj Tarimela Roopavataram - 700743276

Sai Pavan Reddy Pogula - 700741739

Pneumonia is a respiratory disease found in the human body and it was named Pneumonia lung disease. In this work, we take the Pneumonia affected patient records and pass them into the deep learning algorithm. The different type of attributes based on the Pneumonia is taken and passed to the algorithm.[1]

Pneumonia is a fiery state of the lung influencing essentially the little air sacs known as alveoli. Side effects normally incorporate a mix of a useful or dry hack, chest torment, fever, and trouble relaxing. The seriousness of the condition is variable. Pneumonia is normally brought about by contamination with infections or microbes and less generally by different microorganisms, certain drugs or conditions, for example, immune system illnesses Hazard factors incorporate cystic fibrosis, ongoing obstructive pneumonia sickness, asthma, diabetes, cardiovascular breakdown, a background marked by smoking, an unfortunate capacity to hack like following a stroke and a feeble resistant framework. The conclusion is many times in light of side effects and actual assessment. Chest X-ray, blood tests, and culture of the sputum might assist with affirming the finding. The infection might be grouped by where it was gained. The data is been analyzed based on the other diseases that are related to Pneumonia using the deep learning algorithm and predicts the output.[1][2]

The problem is based on the sector, where the hospitals want to predict Pneumonia depending upon the medical reports of the patients using a deep learning algorithm, it classifies and divides the data into segments and each segment contains only one kind of data that is whether the person is suffering from Pneumonia or not.

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]

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)

Motivation:

The mortality from lung illnesses can be decreased using exact and convenient findings. conclusions and treatment are deferred due to the deficiency of experienced radiologists. The huge irregularity between the number of specialists and the number of inhabitants in a specific region likewise thwarts opportune finding. The decision-production of clinical staff can be enhanced by PC-supported analytic instruments, which join parts of PC vision and deep learning with radiological picture examination to perceive and remove designs.[5]

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]

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.

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.

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
```

Next, the dataset would be connected from the Google collab. Initially, the dataset is uploaded into the Google collab folder. Then the Python file should connect to the path from the Google collab folder.

Loading the Dataset

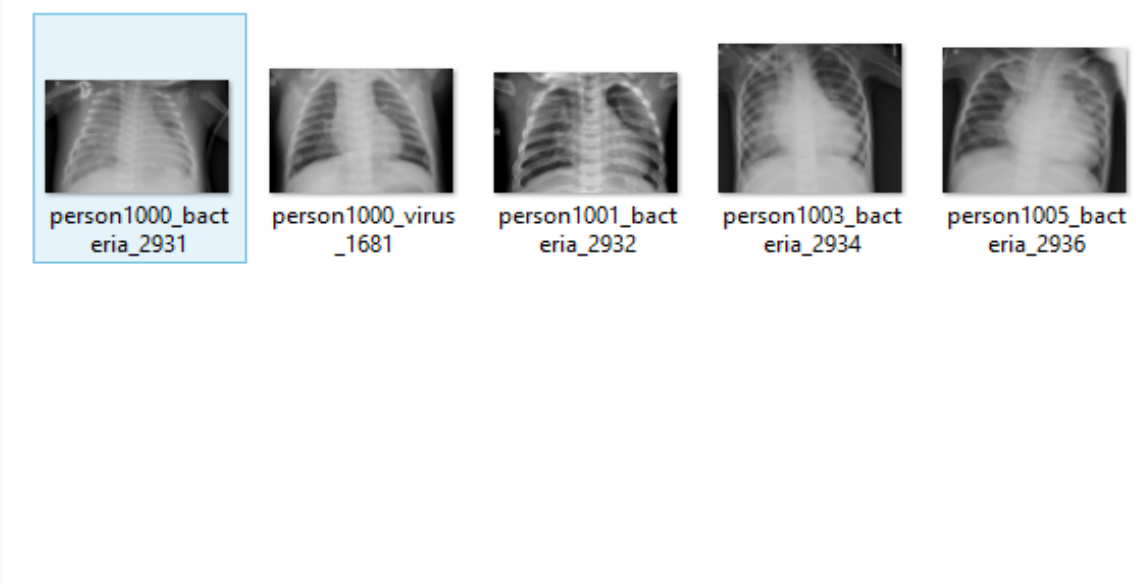
```
#1.Collecting the data
from google.colab import drive
drive.mount('/content/drive')

train = get_training_data('/content/drive/My Drive/Colab Notebooks/pneumonia/train')
test = get_training_data('/content/drive/My Drive/Colab Notebooks/pneumonia/test')
val = get_training_data('/content/drive/My Drive/Colab Notebooks/pneumonia/val')
```

```
Mounted at /content/drive
<ipython-input-2-b2613b36a4a4>:15: VisibleDeprecationWarning: Creating an ndarray from ragged nested sequences (which is a list-or-tuple of lists-or-tup
  return np.array(data)
```

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')
```

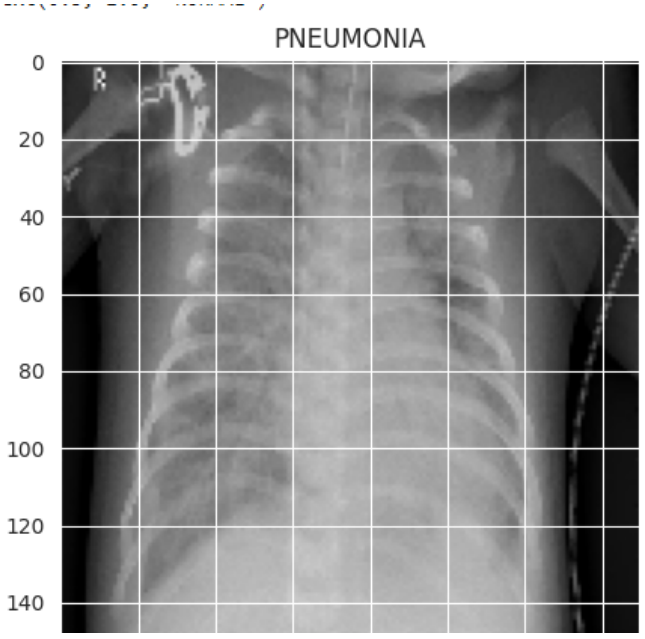
The count of images in each folder of pneumonia and normal patient records are taken.

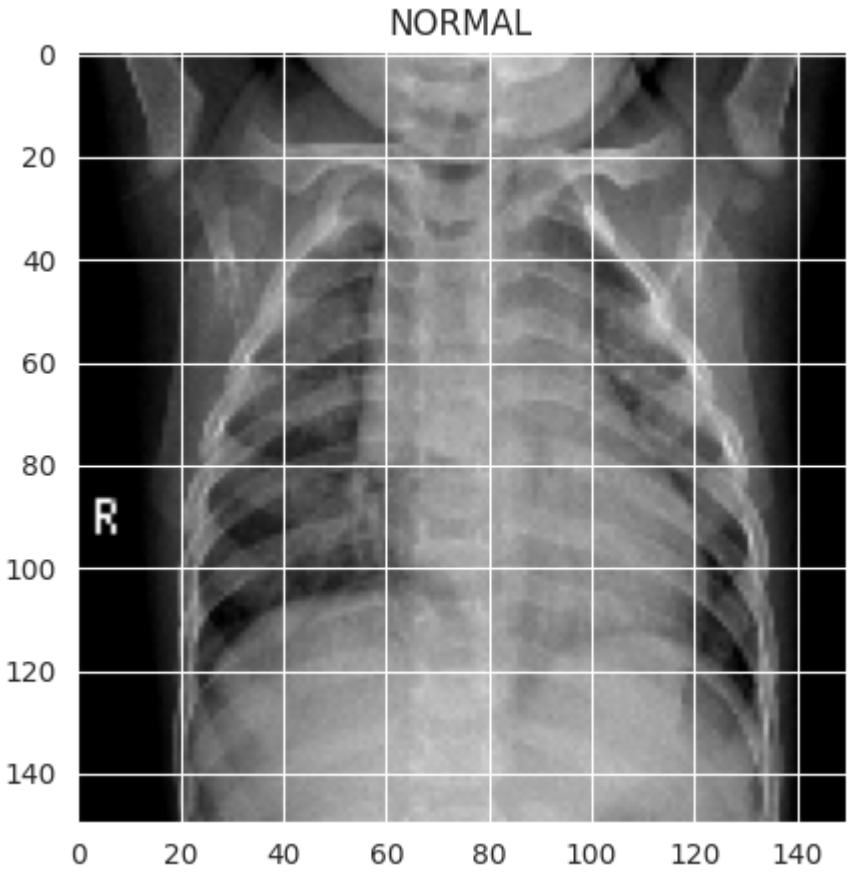
Previewing the images of Normal and affected classes:

```
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:





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: On Going

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: On Going

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.

```
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:

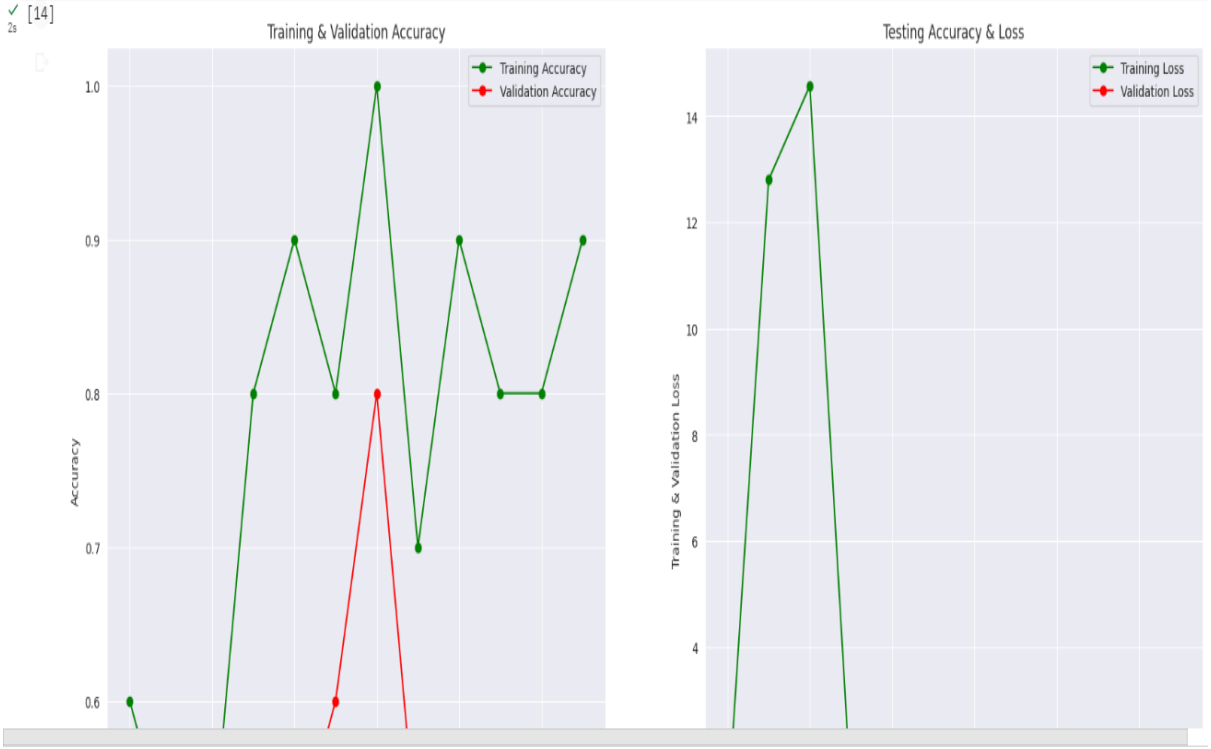
```
1/1 [==============================] - 0s 297ms/step - loss: 1.1859 - accuracy: 0.5000
Loss of the model is -  1.1859419345855713
1/1 [==============================] - 0s 286ms/step - loss: 1.1859 - accuracy: 0.5000
Accuracy of the model is -  50.0 %
```

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")
```

[14]



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

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/