I.Define the problem:

This dataset contains 1252 CT scans that are positive for SARS-CoV-2 infection (COVID-19) and 1230 CT scans for patients non-infected by SARS-CoV-2, 2482 CT scans in total. These data have been collected from real patients in hospitals from Sao Paulo, Brazil. The aim of this dataset is to encourage the research and development of artificial intelligent methods which are able to identify if a person is infected by SARS-CoV-2 through the analysis of his/her CT scans.

II.Gathering and preparing the data:

1. Resizing:

i)The CT Scan images are captured to be in different sizes, it has to be resized to a fixed size. Resizing the images to a consistent size is important, as most deep-learning models expect input images to be in a fixed size. There are several ways to resize a set of CT scan images to a fixed size before applying a deep learning model such as ResNet. One common method i have used here is Python Imaging Library (PIL) to resize the images.

2. Normalization:

Normalizing the images can help to improve the performance of the model by scaling the pixel values to a consistent range. This can be done by dividing the pixel values by 255. I have normalized the images by dividing the pixel values by 255 using numpy library.

3. Data augmentation:

Data augmentation can be used to artificially increase the size of the training dataset by applying random transformations to the images. This can help to make the model more robust to variations in the data. Common data augmentation techniques include random rotations, translations, and flips. There are many arguments that can be used to perform data augmentation on images. Here are a few commonly used arguments in the Keras library: rotation_range,width_shift_range,shear_range,zoom_range,horizontal_flip, fill_mode, brightness_range, and channel_shift_range. And I have performed data augmentation by using some of these arguments.

III.**Model Training:**
I have used CNN along with layer model like - ResNet-* .To train a model using ResNet, you can use one of the pre-trained models such as ResNet-18, ResNet-34, ResNet-50, ResNet-101, ResNet-110, ResNet-152, ResNet-164, or ResNet-1202.and I have used ResNet 50 on my model.And some of the other param i considered are: EPOCHS = 40, SIZE=64, and N_ch=3.

What is ResNet 50 model?ResNet-50 is a convolutional neural network that is 50 layers deep. You can load a pre-trained version of the network trained on more than a million images from the ImageNet database. The pre-trained network can classify images into 1000 object categories, such as keyboard, mouse, pencil, and many animals, ResNet is the short name for residual Network.

IV.PREDICTION AND EVALUATION:

There are several ways to evaluate its performance in terms of loss, accuracy, and predictions. This is done to avoid overfitting and to have an unbiased evaluation of the performance of the model.

I.Trained a deep learning model model using the fit_model method. The method trains the model on batches generated by the datagen.flow function, which performs real-time data augmentation on the input training data X_train and Y_train. The batch size used is specified by the batch_size argument, which is equal to the constant BATCH_SIZE. The number of iterations (epochs) over the entire training dataset is specified by the epochs argument, which is equal to the constant EPOCHS. The training progress is displayed with verbosity level verbose=1. Additionally, two callbacks are passed to the training process: annealer and checkpoint. These are used to modify the training process dynamically, such as reducing the learning rate over time (annealing) or saving a checkpoint of the model's weights. The trained model's performance is also evaluated on the validation dataset X_val and Y_val at the end of each epoch. The history of the training process, including the training and validation accuracy and loss, is returned and stored in the hist variable.

II.Pre-trained deep learning model called 'model' using the load_model function from the tensorflow or keras library. Then, it evaluates the performance of the model on a validation dataset X_val and Y_val by computing the final loss and final accuracy using the evaluate method of the model. The final loss and accuracy are then printed out with the format specified in the print statement.

III.Used the trained model to make predictions on the validation data X_val. The predicted output Y_pred will be an array of probability values representing the model's confidence in each class.

IV.the np.argmax function is used to convert the probability values into class labels by taking the index of the highest probability value for each sample.

V.confusion_matrix function to generate a confusion matrix, which is a 2-dimensional table that shows the number of correct and incorrect predictions made by the model.

VI.ON loading a pre-trained machine learning model called 'model' from a file named 'model.h5' in the '../output/kaggle/working/' directory. The model is evaluated on validation data (X_val and Y_val). And obtained final_loss and final_accuracy of :
Final Loss: 0.21825508773326874, Final Accuracy: 0.9074446558952332.