

Report 1

Computer vision project



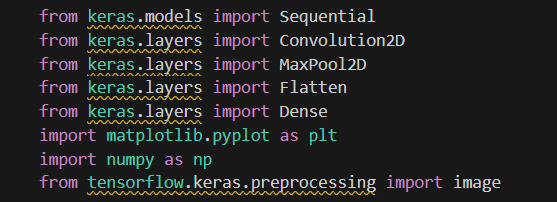
muhammad Sarosh Tahir

F2021376098

**Phase 1 Report**

**1. Importing Libraries**

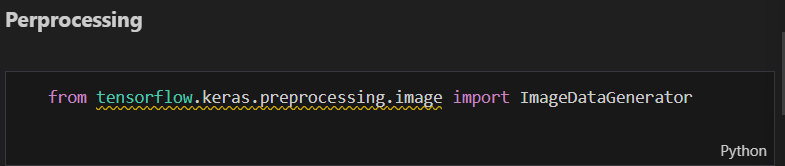
First, we install and import necessary libraries for building and training the model:

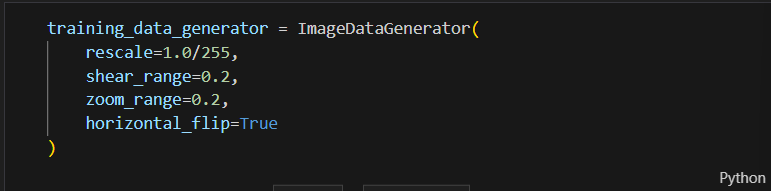


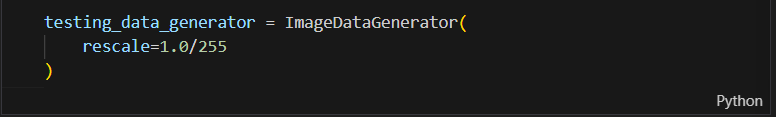
* **Keras and TensorFlow**: These libraries are used for building and training deep learning models.
* **Pillow**: A Python Imaging Library (PIL) fork used for opening, manipulating, and saving many different image file formats.
* **Matplotlib**: A plotting library used for visualizing data.
* **NumPy**: A library for numerical computations.

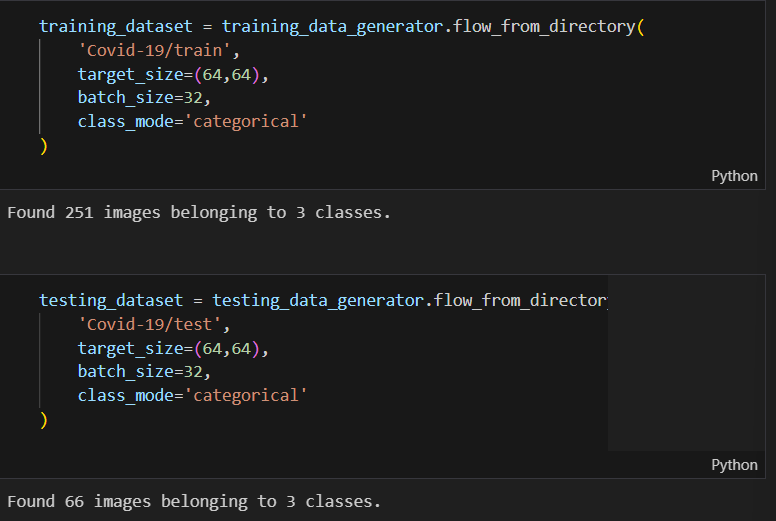
**2. Preprocessing**

Next, we preprocess the dataset using the ImageDataGenerator class from Keras to augment and normalize the images:





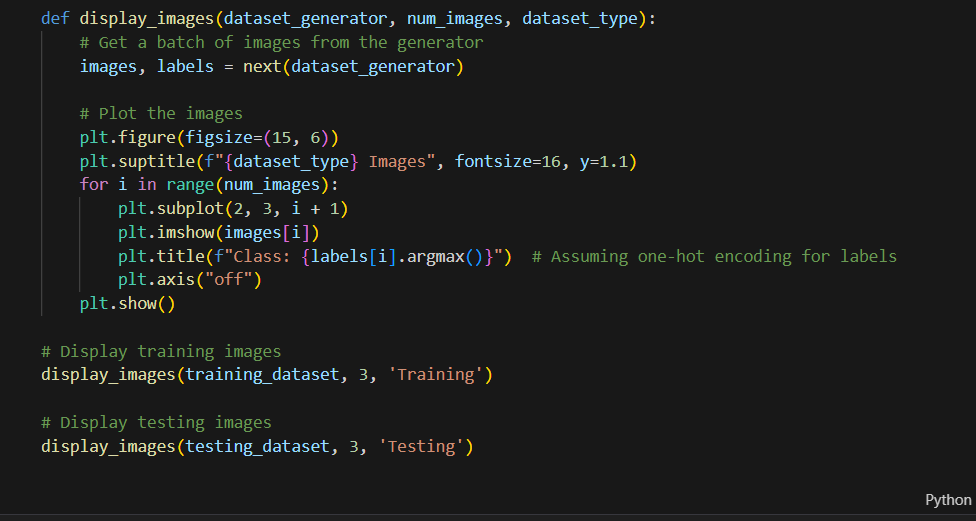


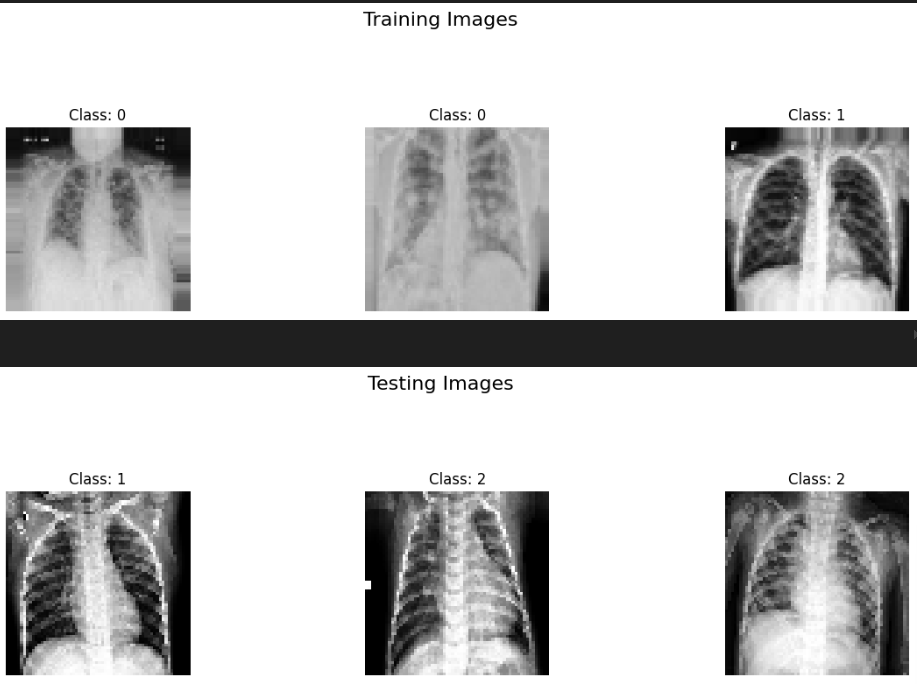


* **ImageDataGenerator**: This class allows for real-time data augmentation and normalization.
* **rescale**: Normalizes pixel values by scaling them.
* **shear\_range, zoom\_range, horizontal\_flip**: Augmentation parameters that help prevent overfitting by introducing variability in the training data.
* **flow\_from\_directory**: Loads images directly from the directory structure and prepares them for training/testing.

**3. Dataset Visualization**

To visualize some images from the dataset, we define a function and use it to display images:

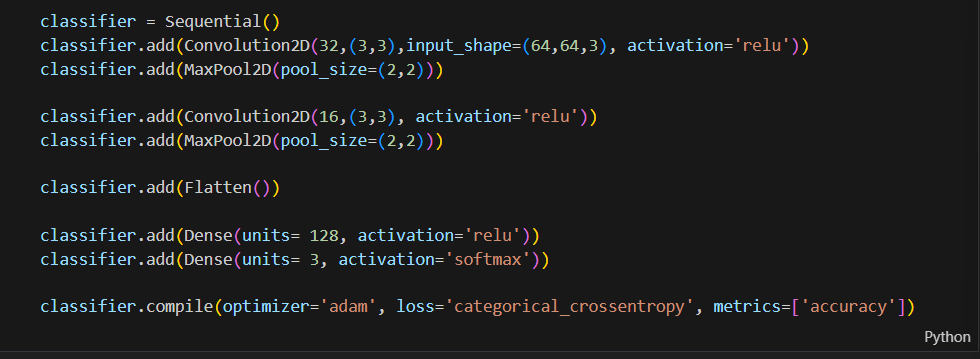




* **display\_images**: A function to display a batch of images and their corresponding labels.
* **next**: Retrieves the next batch of images from the generator.
* **plt.subplot, plt.imshow, plt.title, plt.axis, plt.show**: Functions from Matplotlib to create and display the plots

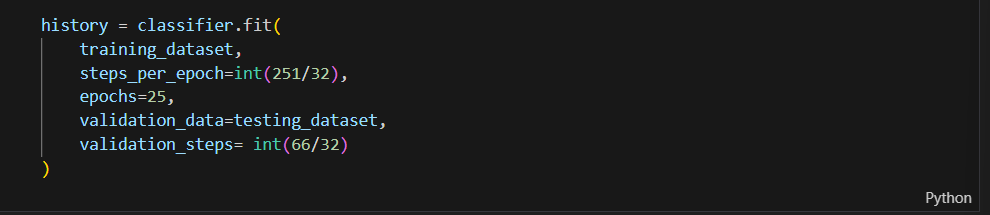
**4. CNN Model Creation**

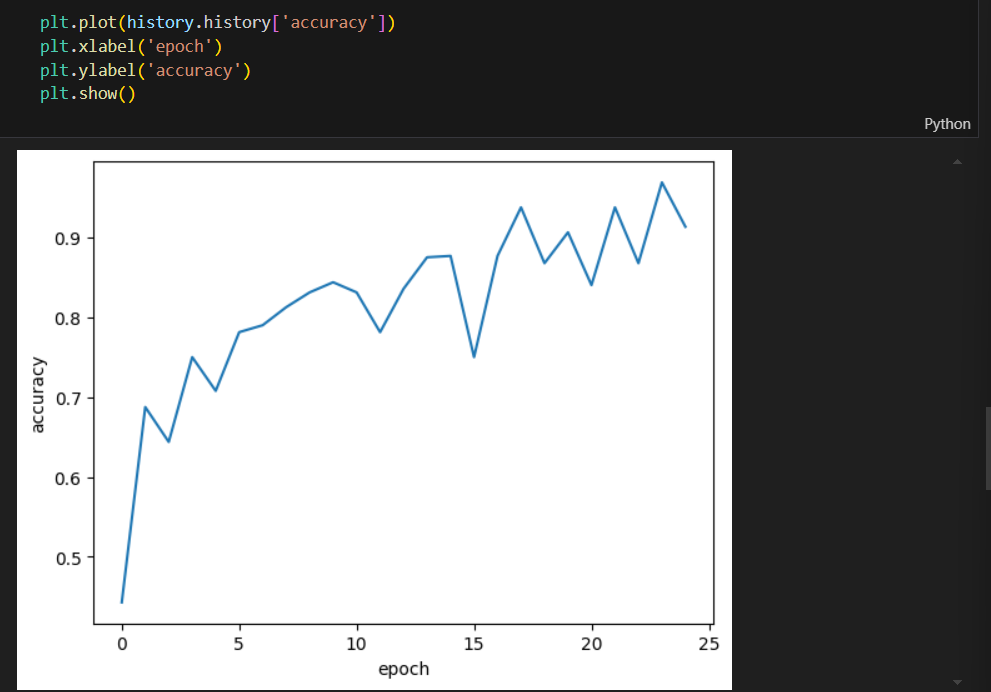
We then define and compile our Convolutional Neural Network (CNN) model:

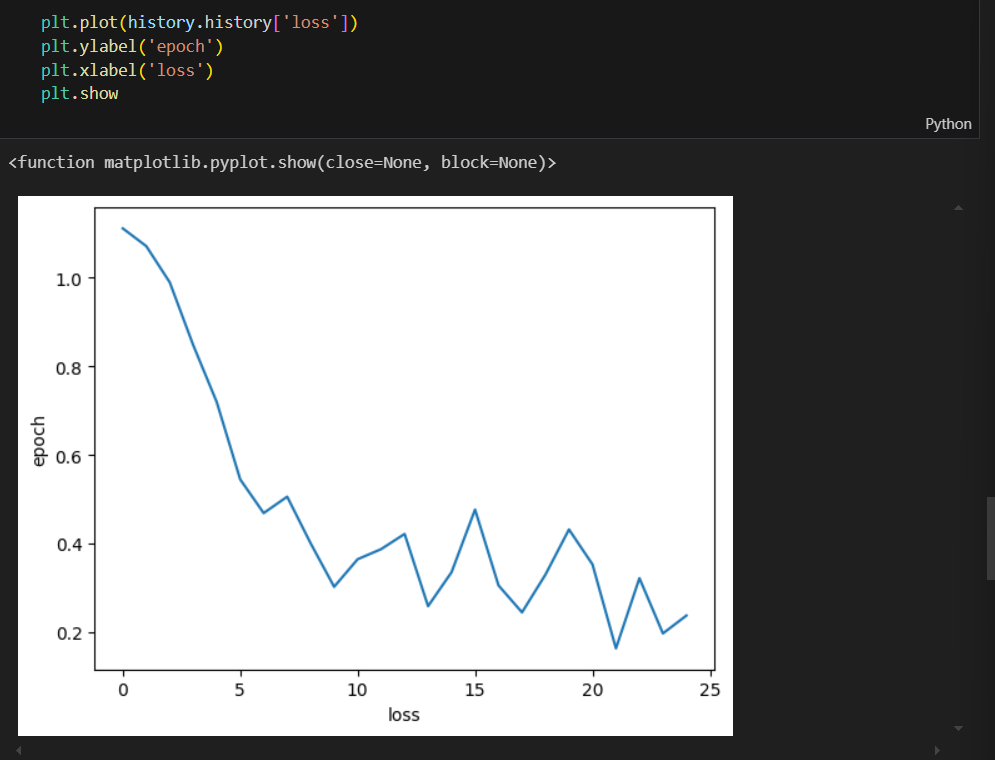


* **Sequential**: A linear stack of layers for our model.
* **Convolution2D**: Applies convolutional filters to the input image.
* **MaxPool2D**: Down-samples the input, reducing its dimensions.
* **Flatten**: Converts the 2D matrices to a 1D vector.
* **Dense**: Fully connected layers.
* **softmax**: Activation function for multi-class classification.
* **adam**: Optimizer.
* **categorical\_crossentropy**: Loss function for multi-class classification.

**5. Training**

****

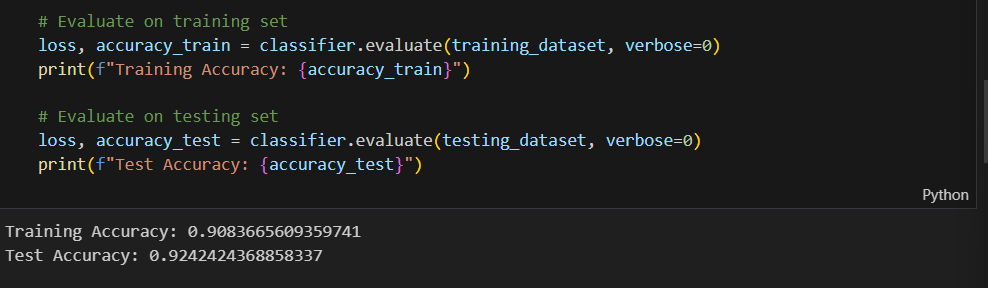




We train the model using the training dataset and validate it using the testing dataset:

* **fit**: Trains the model.
* **steps\_per\_epoch**: Number of batches of samples to use per epoch.
* **epochs**: Number of complete passes through the training dataset.
* **validation\_data**: Data to evaluate the loss and model metrics at the end of each epoch.
* **validation\_steps**: Number of batches of samples to use for validation.
* **plt.plot**: Plots the training accuracy and loss over epochs.

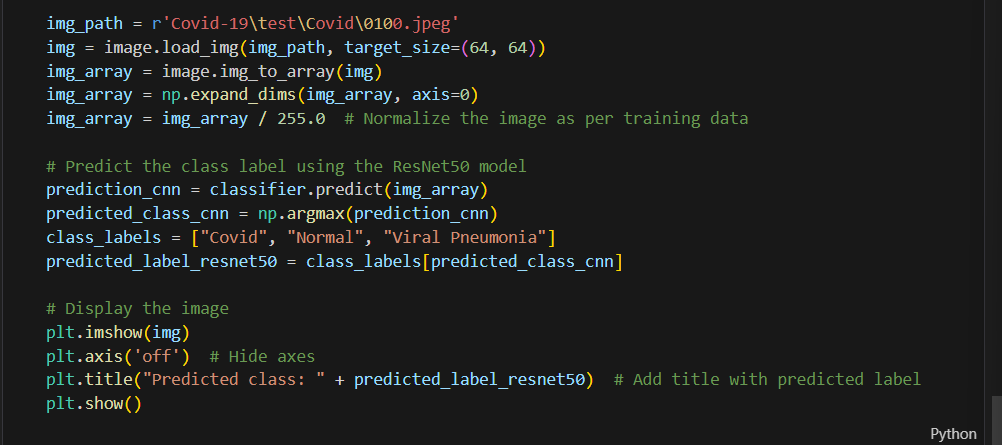
**6. Evaluation**

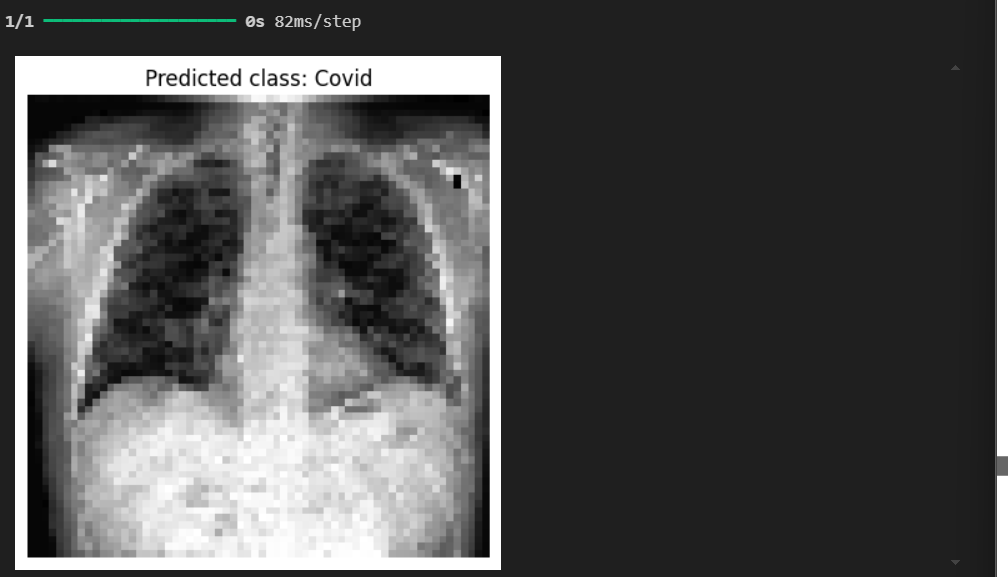
The model's performance is evaluated on both the training and testing datasets:  


* **evaluate**: Computes the loss and accuracy metrics for the dataset.

**7. Prediction**

We make predictions on a new image using the trained model:





* **image.load\_img**: Loads the image from the specified path.
* **image.img\_to\_array**: Converts the image to a NumPy array.
* **np.expand\_dims**: Adds an extra dimension to the array.
* **classifier.predict**: Predicts the class probabilities for the input image.
* **np.argmax**: Returns the index of the highest probability class.
* **plt.imshow, plt.axis, plt.title, plt.show**: Functions from Matplotlib to display the image with the predicted class label.

**8. Conclusion**

This code demonstrates the entire workflow from importing libraries and preprocessing data to building, training, and evaluating a CNN model for classifying chest X-ray images into COVID-19, Normal, and Viral Pneumonia categories.

### Next Steps

In the second phase of the project, we will fine-tune the model further and explore additional models such as ResNet, VGG16, and YOLO for image classification. These advanced models will help improve the accuracy and robustness of our classification system.