CT Scan Image Classification

Screenshots
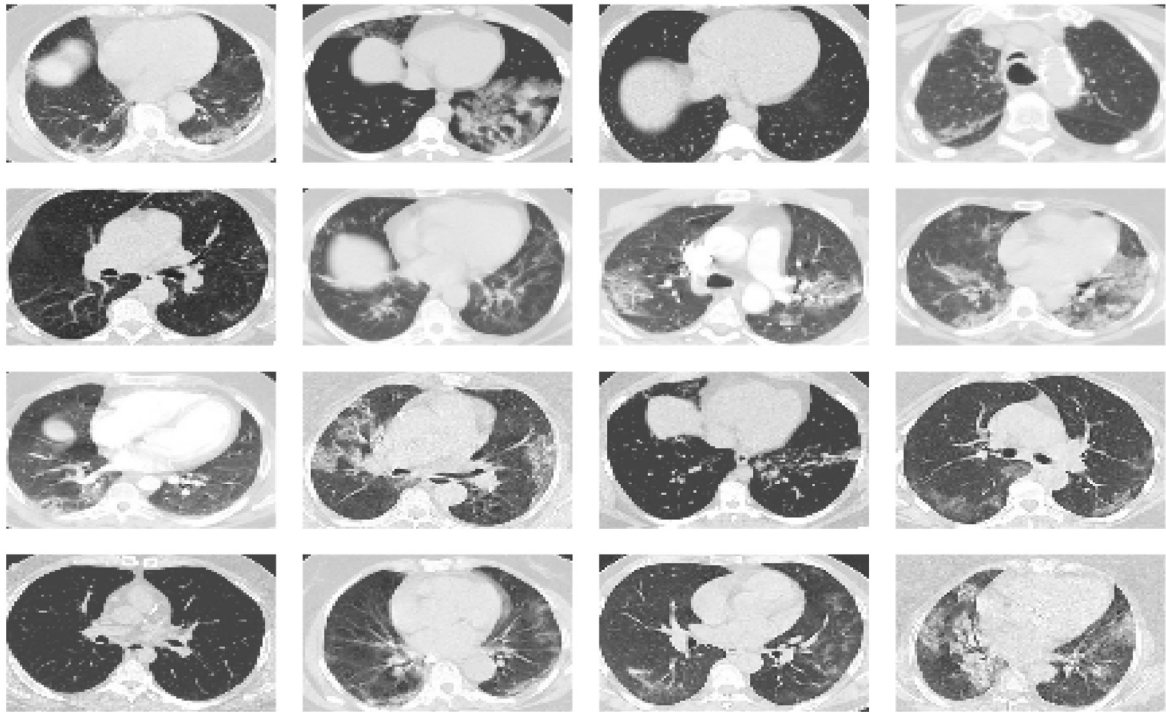

Fig.1 Original Images
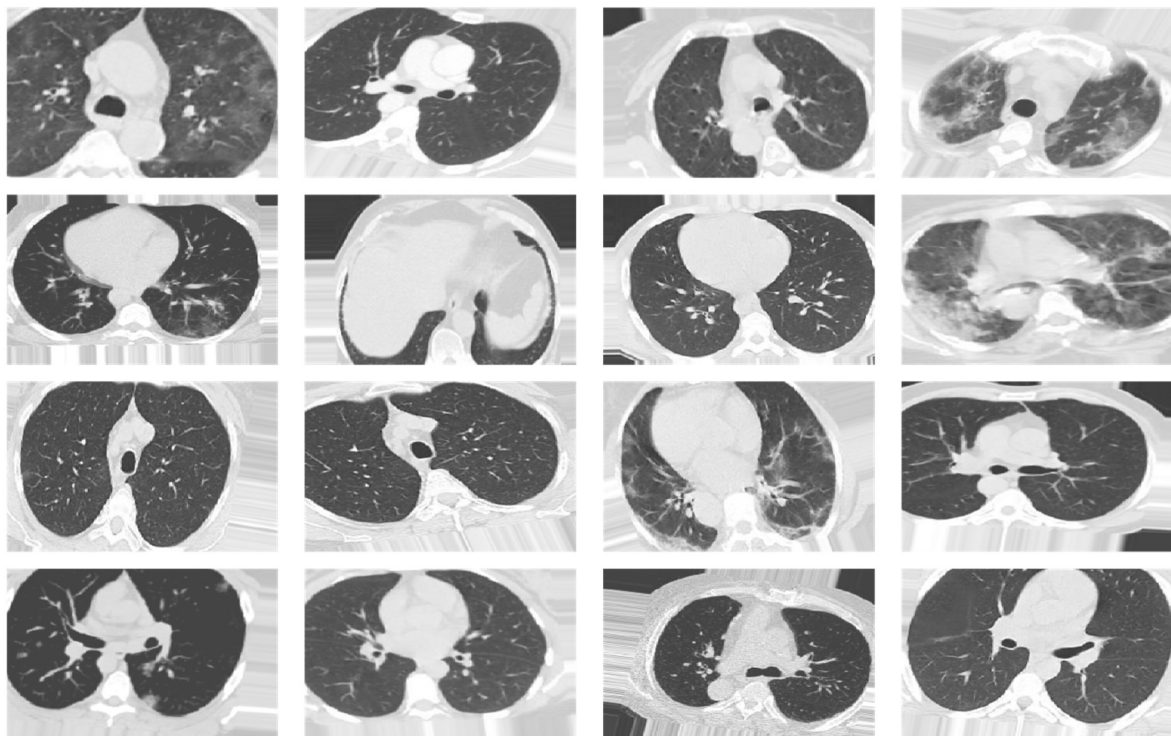

Fig 2. Augmented Images
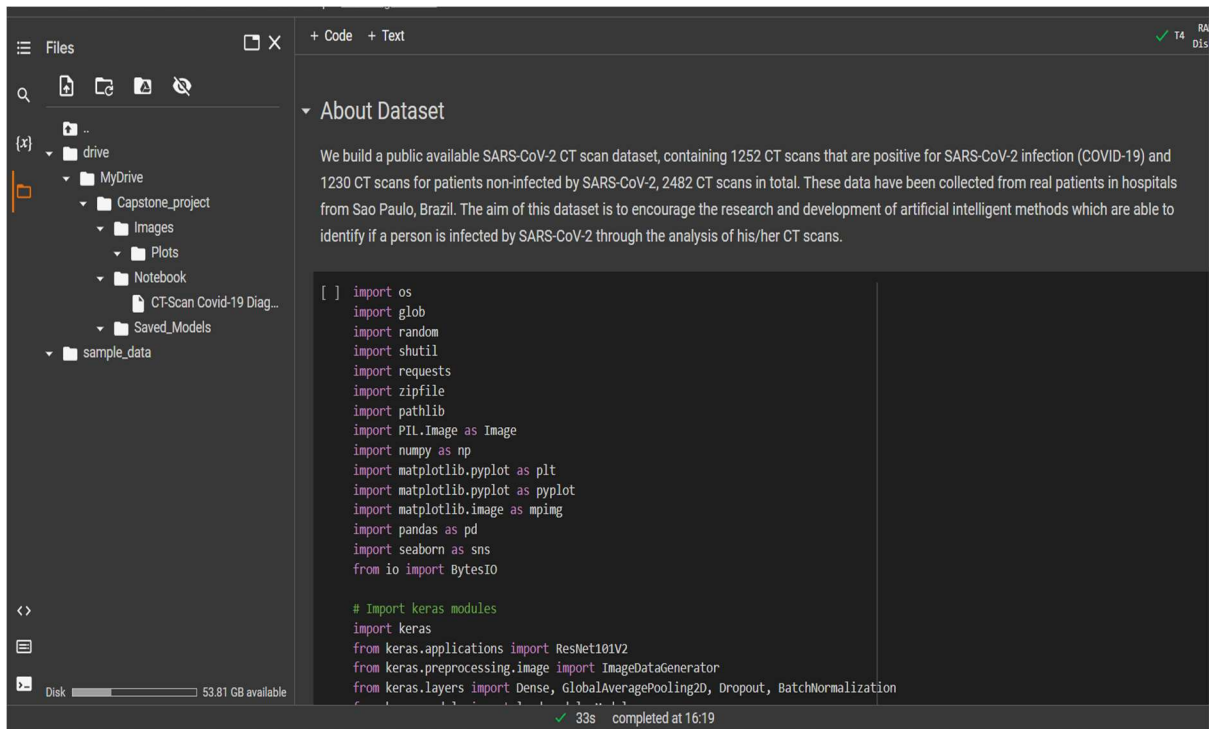
About Dataset

We build a public available SARS-CoV-2 CT scan dataset, containing 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.

```
import os
import glob
import random
import shutil
import requests
import zipfile
import pathlib
import PIL.Image as Image
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.pyplot as pyplot
import matplotlib.image as mpimg
import pandas as pd
import seaborn as sns
from io import BytesIO

# Import keras modules
import keras
from keras.applications import ResNet101V2
from keras.preprocessing.image import ImageDataGenerator
from keras.layers import Dense, GlobalAveragePooling2D, Dropout, BatchNormalization
```

Fig 3. Before Running the Project (Directories)

We chose to use fine-tuning instead of the previous approach because it significantly improved the accuracy of the model. Fine-tuning allowed us to adjust the model's pre-trained layers to better fit the specific features of the new dataset. This adaptation was necessary to achieve higher accuracy, as the initial approach didn't provide satisfactory results. Fine-tuning demonstrated its effectiveness by producing improved performance on the task.

We used batch size is 4 because,

Certainly, by setting a batch size of 4 during fine-tuning, you are processing a smaller number of images in each training iteration. This can lead to better results and improved accuracy because:

1. **Reduced Overfitting**: Smaller batch sizes can introduce more noise into the learning process, acting as a form of regularization. This can prevent the model from overfitting to the training data and improve its generalization to unseen data.

2. **Better Convergence**: Smaller batch sizes often lead to quicker convergence during training. The model adapts faster to the data, as each batch provides a more frequent update to the model's weights based on the current training data.

However, it's worth noting that using a smaller batch size may also increase training time due to the more frequent weight updates. It's a trade-off between faster convergence and potentially increased computation time.

```
# Create a training data generator
train_generator = train_datagen.flow_from_directory(
    train_dir,             # Directory containing training images
    target_size=(W, H),    # Resize images to specified dimensions
    batch_size=4,          # Number of images per batch
    class_mode='categorical',  # Mode for class labels (categorical for one-hot encoded labels)
    shuffle=True           # No need to shuffle
)
```

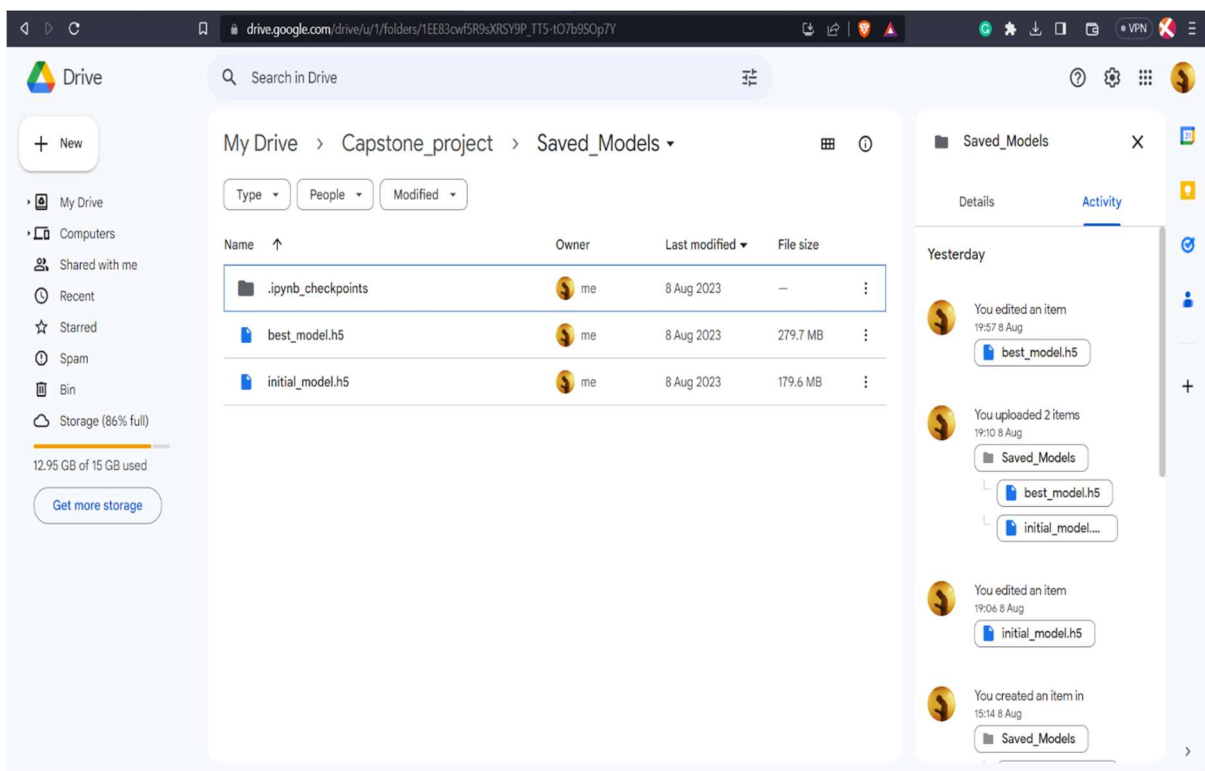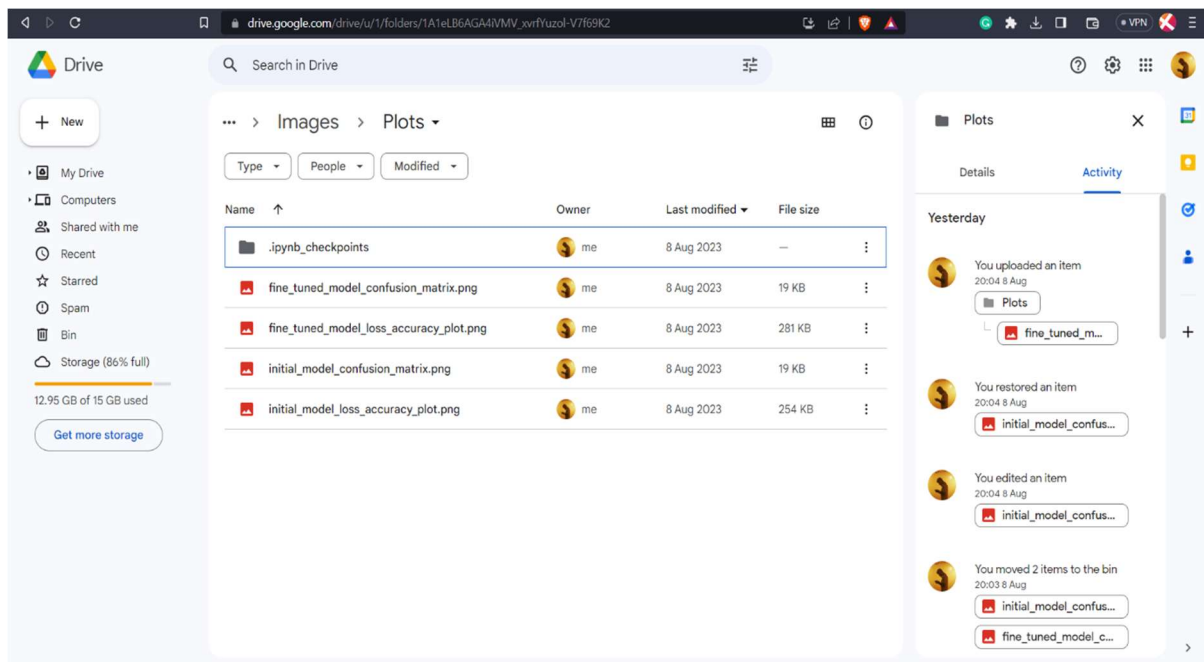Fig 4. After Running the Project (Directories)

Fig 5. Training the Model

Fig 6. Saved Models in Drive

Fig 7. Saved Images in Drive