**DAY 03**

**21.03.2023**

**Dataset provided:**

An examination dataset was provided to us(interns) in google drive.

It contained three folders namely train, validation and validation2.

Each folder further had 5 sub folders(multi-class) which were normal, cheat, phone, paperseeing and paperexchange.

Train set: train

Test set: validation

Validation set: validation2

**Online Class Summary:**

Introduction to Jupyter.

Downloading and Installing Jupyter.

Explanation of datasets.

**Google Drive**

https://drive.google.com/drive/folders/1v0nUtgz8wMnn7JJWqw4MHPaVJZZ1GqOa?usp=drive\_link

**Google Colab**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import plotly.express as px

import os

import tensorflow as tf

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from sklearn.preprocessing import LabelBinarizer

from sklearn.metrics import roc\_curve, auc, roc\_auc\_score

# from IPython.display import clear\_output

import warnings

warnings.filterwarnings('ignore')

# 1. Train Set

train\_gen = ImageDataGenerator(rescale = 1.0/255.0) # Normalise the data

train\_image\_generator = train\_gen.flow\_from\_directory(

                                            train\_path,

                                            target\_size=(150, 150),

                                            batch\_size=32,

                                            class\_mode='categorical')

# 2. Validation Set

val\_gen = ImageDataGenerator(rescale = 1.0/255.0) # Normalise the data

val\_image\_generator = train\_gen.flow\_from\_directory(

                                            validation\_path,

                                            target\_size=(150, 150),

                                            batch\_size=32,

                                            class\_mode='categorical')

# 3. Test Set

test\_gen = ImageDataGenerator(rescale = 1.0/255.0) # Normalise the data

test\_image\_generator = train\_gen.flow\_from\_directory(

                                            test\_path,

                                            target\_size=(150, 150),

                                            batch\_size=32,

                                            class\_mode='categorical')

# Build a custom sequential CNN model

model = Sequential() # model object

# Add Layers

model.add(Conv2D(filters=32, kernel\_size=3, strides=1, padding='same', activation='relu', input\_shape=[150, 150, 3]))

model.add(MaxPooling2D(2, ))

model.add(Conv2D(filters=64, kernel\_size=3, strides=1, padding='same', activation='relu'))

model.add(MaxPooling2D(2))

# Flatten the feature map

model.add(Flatten())

# Add the fully connected layers

model.add(Dense(128, activation='relu'))

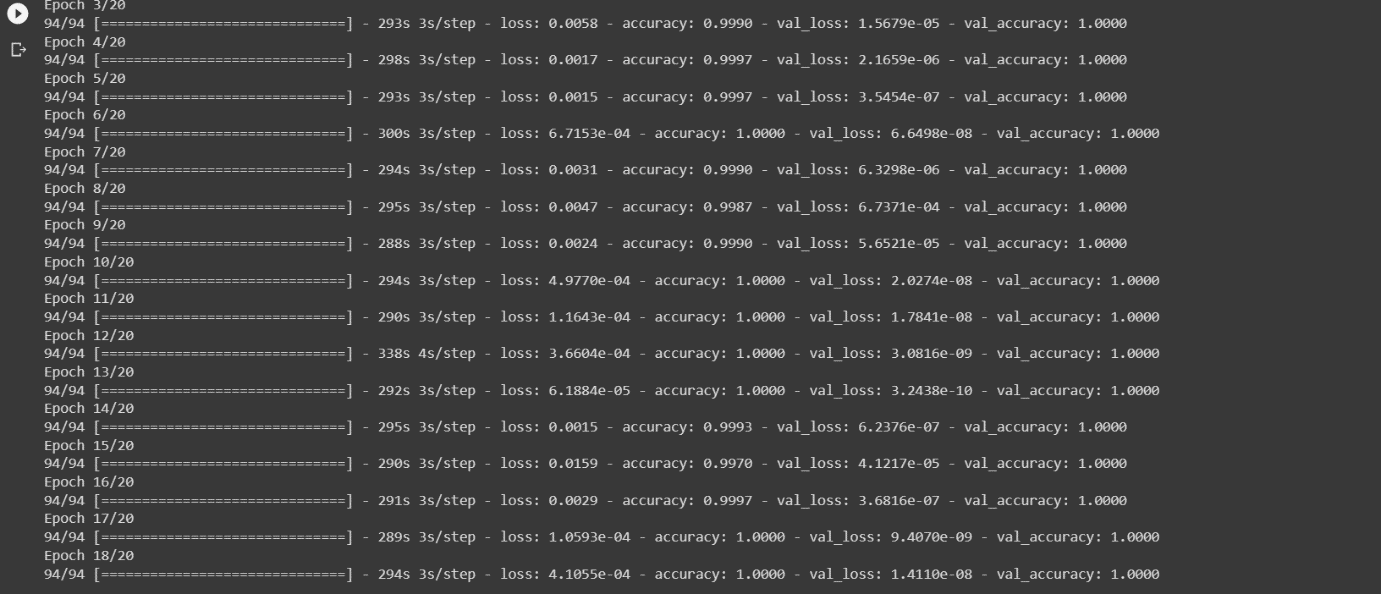
model.add(Dropout(0.25))

model.add(Dense(128, activation='relu'))

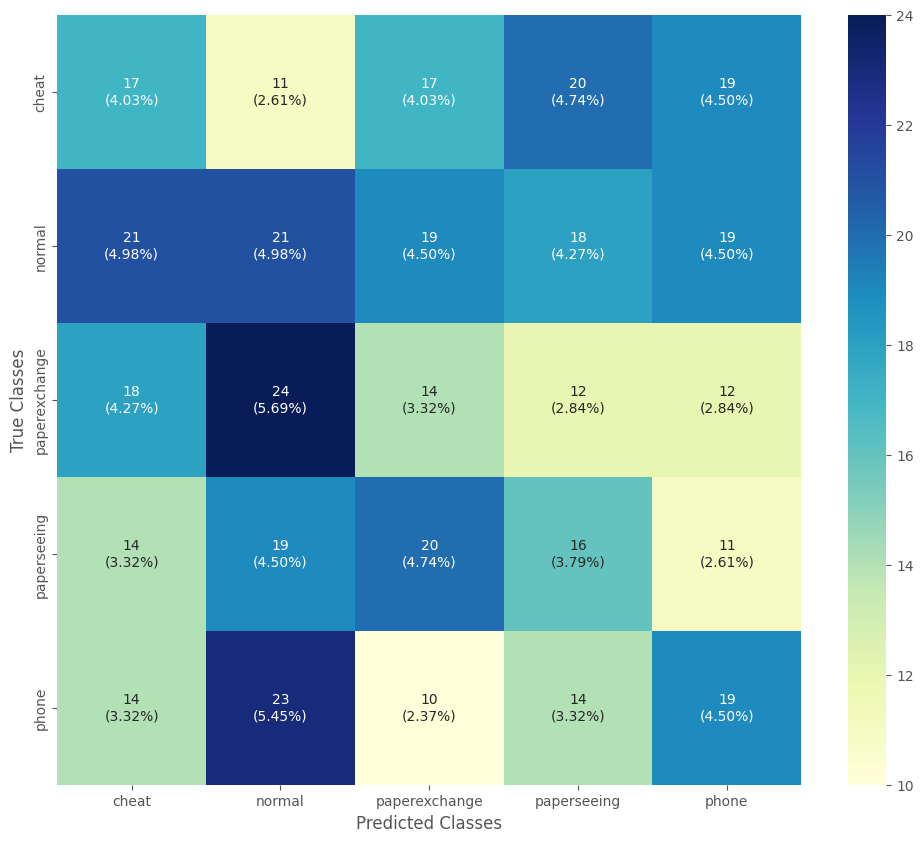
model.add(Dense(5, activation='softmax'))

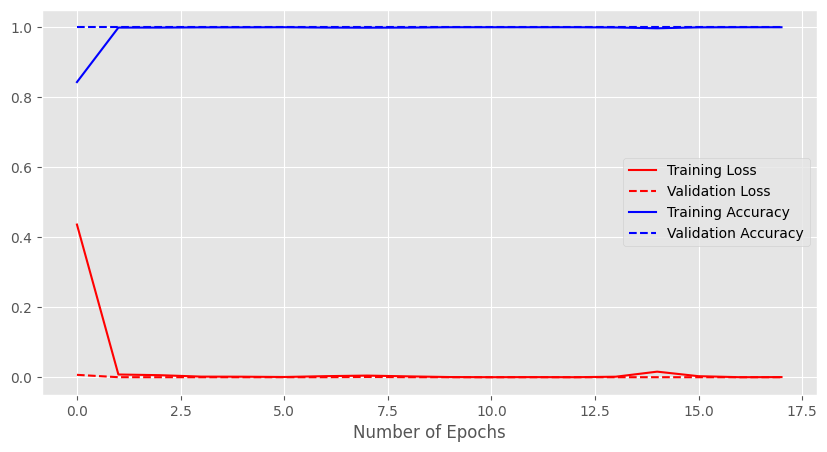
# print the model summary

model.summary()



**Graphs:**

****

****