SCHENIDER HACKATHON
Data Science

By Pablo Llobregat Ruiz

Note: All the code has been executed in a GoogleColab Notebook. `GPU 0: Tesla T4`

IMAGE CLASSIFICATION TASK

OBJECTIVE

To use a pre-trained VGG19 model to classify images into 3 clases (0,1,2).

VGG19

A pre-trained model is a model that has been already trained on a dataset and is available for use. The VGG19 model is a pre-trained model that has been trained on the ImageNet dataset. The ImageNet dataset is a large dataset of images that have been labeled with various object classes. The VGG19 model is a deep convolutional neural network that has 19 layers. The first 16 layers of the VGG19 model are the same as the 16 layers of the VGG16 model. The VGG19 model is a very powerful model that can be used for a variety of image classification tasks.

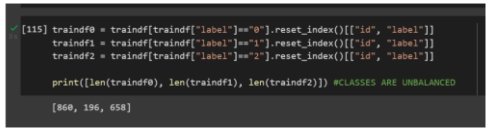
OUR DATASET

Our dataset consists in 2 csv files:
“train.csv” with the training images directory , label and information ; “test.csv”with the test images directory and information.

For processing the images I will create a `ImageDataGenerator` from keras module. To load images from csv to the generator I will use the function recently added to the keras module `flow_from_dataframe()`.

BALANCING CLASSES

Unbalanced class problem has been solved by adding more rotated images to the class 1 and 2 until having the same number.



```
[115] traindf0 = traindf[traindf["label"]=="0"].reset_index()[["id", "label"]]
      traindf1 = traindf[traindf["label"]=="1"].reset_index()[["id", "label"]]
      traindf2 = traindf[traindf["label"]=="2"].reset_index()[["id", "label"]]

      print([len(traindf0), len(traindf1), len(traindf2)]) #CLASSES ARE UNBALANCED

      [860, 196, 658]
```

IMAGE CLASSIFICATION TASK

DATA AUGMENTATION

Images have been randomly cropped and flipped using a self created function when passed to the ImageDataGenerator.

SCORE METRICS

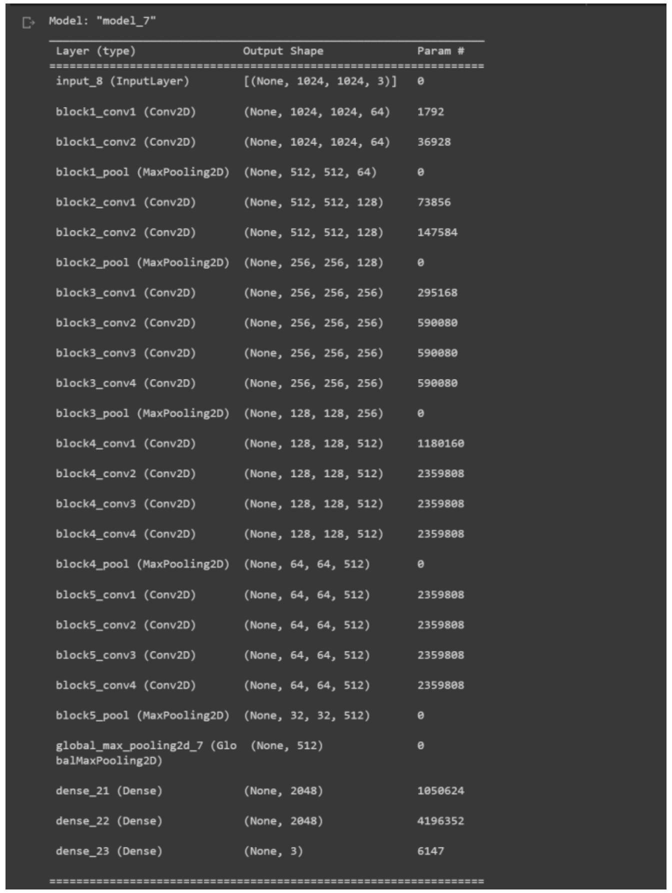
F-score and precisión funtions have been defined.

MODEL

Initial layer has been modified to our input size. 3 final Dense Layers have been added. The last one is a 3 nodes layer with SOFTMAX activation function in order to classify.

Weiths are initialized with VGG19 model.
All the layers weights have been unfrozen.

I choose very low learning rate, thus it will take more epochs.



```
Model: "model_7"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_8 (InputLayer)        [(None, 1024, 1024, 3)]   0

 block1_conv1 (Conv2D)       (None, 1024, 1024, 64)    1792

 block1_conv2 (Conv2D)       (None, 1024, 1024, 64)    36928

 block1_pool (MaxPooling2D)  (None, 512, 512, 64)      0

 block2_conv1 (Conv2D)       (None, 512, 512, 128)     73856

 block2_conv2 (Conv2D)       (None, 512, 512, 128)     147584

 block2_pool (MaxPooling2D)  (None, 256, 256, 128)     0

 block3_conv1 (Conv2D)       (None, 256, 256, 256)     295168

 block3_conv2 (Conv2D)       (None, 256, 256, 256)     590080

 block3_conv3 (Conv2D)       (None, 256, 256, 256)     590080

 block3_conv4 (Conv2D)       (None, 256, 256, 256)     590080

 block3_pool (MaxPooling2D)  (None, 128, 128, 256)     0

 block4_conv1 (Conv2D)       (None, 128, 128, 512)     1180160

 block4_conv2 (Conv2D)       (None, 128, 128, 512)     2359808

 block4_conv3 (Conv2D)       (None, 128, 128, 512)     2359808

 block4_conv4 (Conv2D)       (None, 128, 128, 512)     2359808

 block4_pool (MaxPooling2D)  (None, 64, 64, 512)       0

 block5_conv1 (Conv2D)       (None, 64, 64, 512)       2359808

 block5_conv2 (Conv2D)       (None, 64, 64, 512)       2359808

 block5_conv3 (Conv2D)       (None, 64, 64, 512)       2359808

 block5_conv4 (Conv2D)       (None, 64, 64, 512)       2359808

 block5_pool (MaxPooling2D)  (None, 32, 32, 512)       0

 global_max_pooling2d_7 (Glo  (None, 512)              0
 balMaxPooling2D)

 dense_21 (Dense)            (None, 2048)              1050624

 dense_22 (Dense)            (None, 2048)              4196352

 dense_23 (Dense)            (None, 3)                 6147

=================================================================
```

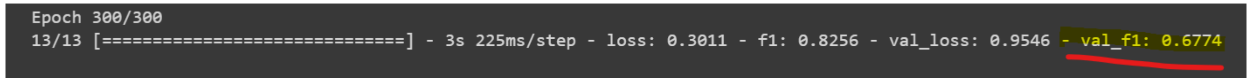
IMAGE CLASSIFICATION TASK

FITTING THE MODEL

300 epochs , batch size of 20 images, 13 steps per epoch

RESULTS

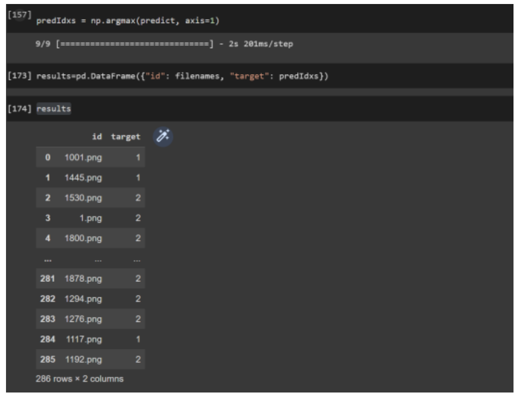
The model has been fitted with 3370 images, and evaluated with 34 images. The final f1 score with this small evaluation dataset was 0,67.



```
Epoch 300/300
13/13 [==============================] - 3s 225ms/step - loss: 0.3011 - f1: 0.8256 - val_loss: 0.9546 - val_f1: 0.6774
```

GETTING PREDICTIONS

With this model we Will make the predictions for the test dataset.



```
[157] predIdxs = np.argmax(predict, axis=1)

      9/9 [==============================] - 2s 201ms/step

[173] results=pd.DataFrame({"id": filenames, "target": predIdxs})

[174] results
```

	id	target
0	1001.png	1
1	1445.png	1
2	1530.png	2
3	1.png	2
4	1800.png	2
...	...	...
281	1878.png	2
282	1294.png	2
283	1276.png	2
284	1117.png	1
285	1192.png	2

286 rows × 2 columns