Project
Development Phase
Model Performance
Test

Date	19 November 2022
Team ID	**PNT2022TMID42852**
Project Name	Project - Digital Naturalist - AI Enabled tool for Biodiversity Researchers
Maximum Marks	10 Marks

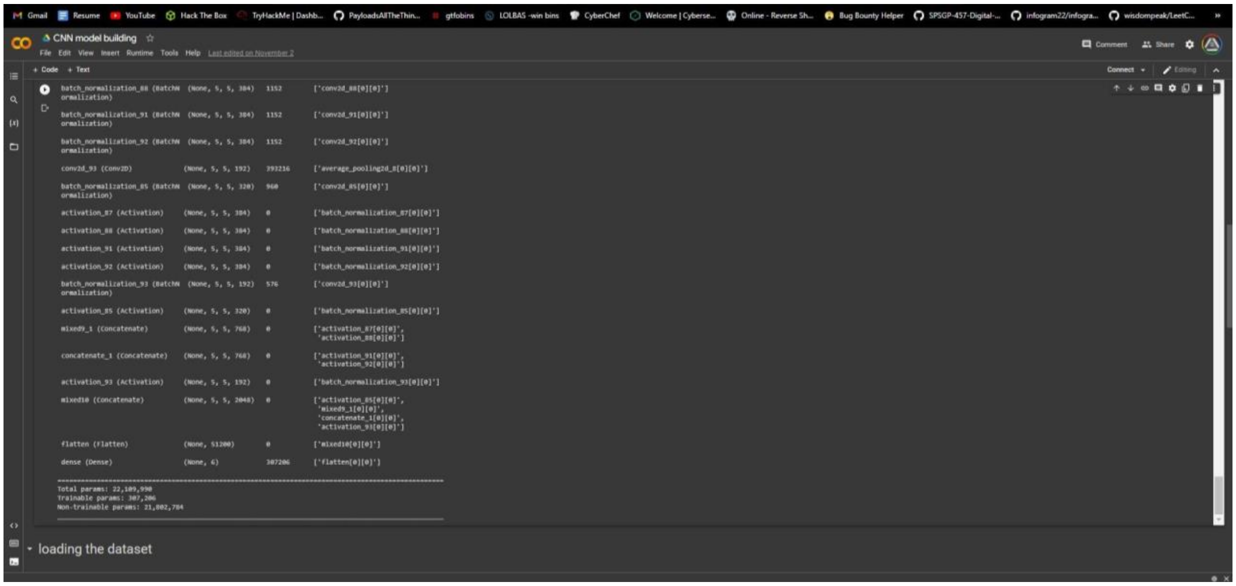
Model Performance Testing:

Project team shall fill the following information in model performance testing template.

S No.	Parameter	Values	Screenshot
1.	Model Summary	**Total params: 22,109,990** **Trainable params: 307,206** **Non-trainable params: 21,802,784**	Screenshot 1
2.	Accuracy	Training Accuracy - 92.8% Validation Accuracy - 85.6%	Screenshot 2

Screenshots - Please refer to the next page:

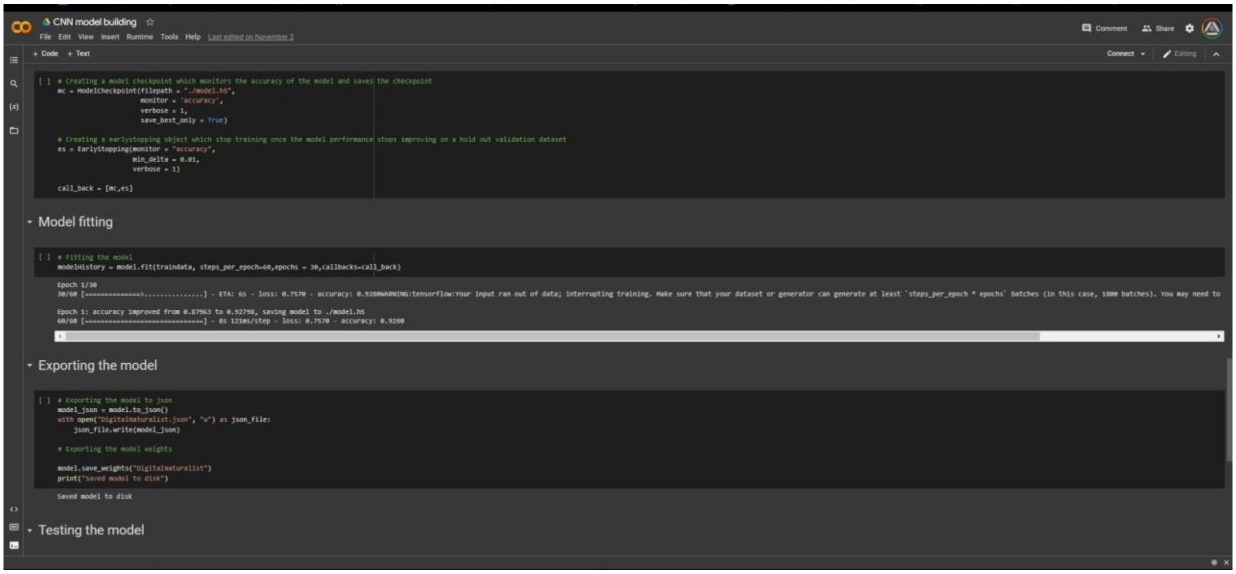
Screenshot 1 :



```
batch_normalization_88 (BatchN  (None, 5, 5, 384)   1152      ['conv2d_88[0][0]']
ormalization)

batch_normalization_91 (BatchN  (None, 5, 5, 384)   1152      ['conv2d_91[0][0]']
ormalization)

batch_normalization_92 (BatchN  (None, 5, 5, 384)   1152      ['conv2d_92[0][0]']
ormalization)

conv2d_93 (Conv2D)             (None, 5, 5, 192)   393216    ['average_pooling2d_8[0][0]']

batch_normalization_85 (BatchN  (None, 5, 5, 320)   960       ['conv2d_85[0][0]']
ormalization)

activation_87 (Activation)     (None, 5, 5, 384)   0         ['batch_normalization_87[0][0]']

activation_88 (Activation)     (None, 5, 5, 384)   0         ['batch_normalization_88[0][0]']

activation_91 (Activation)     (None, 5, 5, 384)   0         ['batch_normalization_91[0][0]']

activation_92 (Activation)     (None, 5, 5, 384)   0         ['batch_normalization_92[0][0]']

batch_normalization_93 (BatchN  (None, 5, 5, 192)   576       ['conv2d_93[0][0]']
ormalization)

activation_85 (Activation)     (None, 5, 5, 320)   0         ['batch_normalization_85[0][0]']

mixed9_1 (Concatenate)         (None, 5, 5, 768)   0         ['activation_87[0][0]',
                                                              'activation_88[0][0]']

concatenate_1 (Concatenate)    (None, 5, 5, 768)   0         ['activation_91[0][0]',
                                                              'activation_92[0][0]']

activation_93 (Activation)     (None, 5, 5, 192)   0         ['batch_normalization_93[0][0]']

mixed10 (Concatenate)          (None, 5, 5, 2048)  0         ['activation_85[0][0]',
                                                              'mixed9_1[0][0]',
                                                              'concatenate_1[0][0]',
                                                              'activation_93[0][0]']

flatten (Flatten)              (None, 51200)       0         ['mixed10[0][0]']

dense (Dense)                  (None, 6)           307206    ['flatten[0][0]']

==================================================================================================
Total params: 22,109,990
Trainable params: 307,206
Non-trainable params: 21,802,784
__________________________________________________________________________________________________
```

loading the dataset

Screenshot 2:


CNN model building
File Edit View Insert Runtime Tools Help Last edited on November 2
Comment
Share
+ Code + Text
Connect
Editing

```
# Creating a model checkpoint which monitors the accuracy of the model and saves the checkpoint
mc = ModelCheckpoint(filepath = "./model.h5",
                     monitor = 'accuracy',
                     verbose = 1,
                     save_best_only = True)

# Creating a earlystopping object which stop training once the model performance stops improving on a hold out validation dataset
es = EarlyStopping(monitor = "accuracy",
                   min_delta = 0.01,
                   verbose = 1)

call_back = [mc,es]
```

Model fitting

```
# Fitting the model
modelHistory = model.fit(traindata, steps_per_epoch=60,epochs = 30,callbacks=call_back)
```

```
Epoch 1/30
30/60 [==============>...............] - ETA: 6s - loss: 0.7570 - accuracy: 0.9280WARNING:tensorflow:Your input ran out of data; interrupting training. Make sure that your dataset or generator can generate at least `steps_per_epoch * epochs` batches (in this case, 1800 batches). You may need to

Epoch 1: accuracy improved from 0.87963 to 0.92798, saving model to ./model.h5
60/60 [==============================] - 8s 121ms/step - loss: 0.7570 - accuracy: 0.9280
```

Exporting the model

```
# Exporting the model to json
model_json = model.to_json()
with open("DigitalNaturalist.json", "w") as json_file:
    json_file.write(model_json)

# Exporting the model weights

model.save_weights("DigitalNaturalist")
print("Saved model to disk")
```

```
Saved model to disk
```

Testing the model