

Project Development Phase

Model Performance Test

Team Members	POOVENTHAN D PRANESH S SABAREESHWARAN S SANDEEP P K
TEAM ID	PNT2022TMID04681
Project Name	Digital Naturalist - AI Enabled tool for Biodiversity Researchers
Maximum Marks	10 Marks

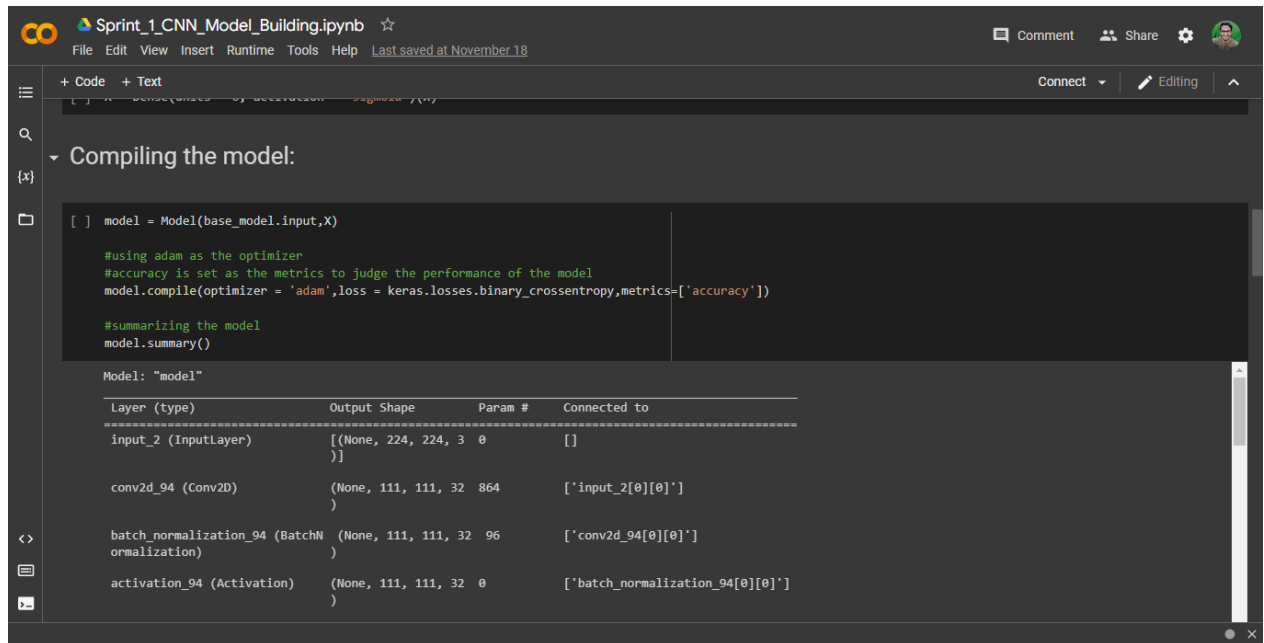
Model Performance Testing:

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

S. N o.	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 :



Sprint_1_CNN_Model_Building.ipynb ☆
File Edit View Insert Runtime Tools Help Last saved at November 18

+ Code + Text Connect Editing

Compiling the model:

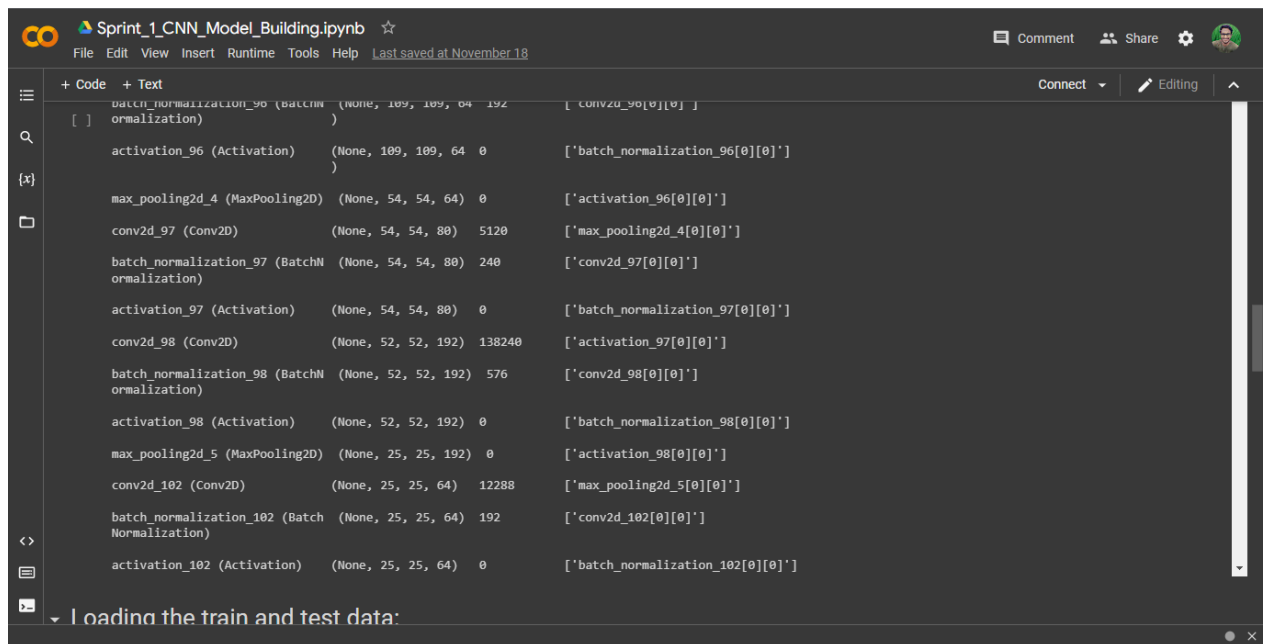
```
[ ] model = Model(base_model.input,X)

#using adam as the optimizer
#accuracy is set as the metrics to judge the performance of the model
model.compile(optimizer = 'adam',loss = keras.losses.binary_crossentropy,metrics=['accuracy'])

#summarizing the model
model.summary()
```

Model: "model"

Layer (type)	Output Shape	Param #	Connected to
input_2 (InputLayer)	[(None, 224, 224, 3)]	0	[]
conv2d_94 (Conv2D)	(None, 111, 111, 32)	864	['input_2[0][0]']
batch_normalization_94 (Batch Normalization)	(None, 111, 111, 32)	96	['conv2d_94[0][0]']
activation_94 (Activation)	(None, 111, 111, 32)	0	['batch_normalization_94[0][0]']



Sprint_1_CNN_Model_Building.ipynb ☆
File Edit View Insert Runtime Tools Help Last saved at November 18

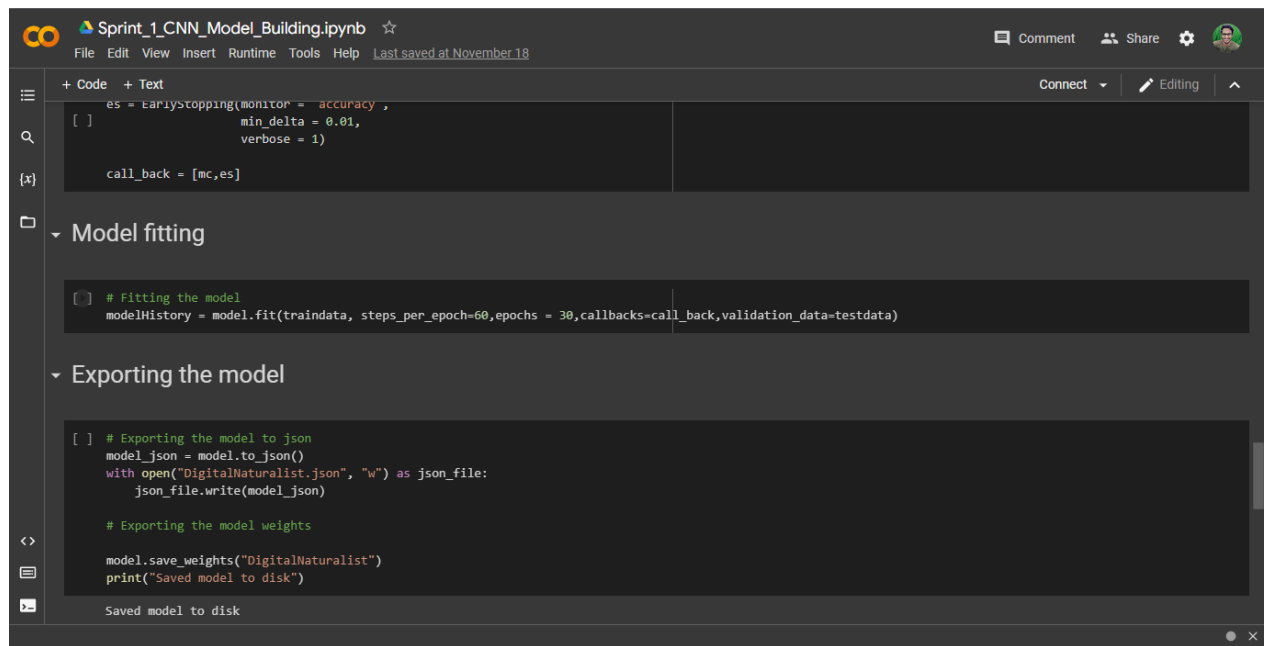
+ Code + Text Connect Editing

```
[ ] batch_normalization_96 (Batch Normalization)
activation_96 (Activation)
max_pooling2d_4 (MaxPooling2D)
conv2d_97 (Conv2D)
batch_normalization_97 (Batch Normalization)
activation_97 (Activation)
conv2d_98 (Conv2D)
batch_normalization_98 (Batch Normalization)
activation_98 (Activation)
max_pooling2d_5 (MaxPooling2D)
conv2d_102 (Conv2D)
batch_normalization_102 (Batch Normalization)
activation_102 (Activation)
```

batch_normalization_96 (Batch Normalization)	(None, 109, 109, 64)	192	['conv2d_96[0][0]']
activation_96 (Activation)	(None, 109, 109, 64)	0	['batch_normalization_96[0][0]']
max_pooling2d_4 (MaxPooling2D)	(None, 54, 54, 64)	0	['activation_96[0][0]']
conv2d_97 (Conv2D)	(None, 54, 54, 80)	5120	['max_pooling2d_4[0][0]']
batch_normalization_97 (Batch Normalization)	(None, 54, 54, 80)	240	['conv2d_97[0][0]']
activation_97 (Activation)	(None, 54, 54, 80)	0	['batch_normalization_97[0][0]']
conv2d_98 (Conv2D)	(None, 52, 52, 192)	138240	['activation_97[0][0]']
batch_normalization_98 (Batch Normalization)	(None, 52, 52, 192)	576	['conv2d_98[0][0]']
activation_98 (Activation)	(None, 52, 52, 192)	0	['batch_normalization_98[0][0]']
max_pooling2d_5 (MaxPooling2D)	(None, 25, 25, 192)	0	['activation_98[0][0]']
conv2d_102 (Conv2D)	(None, 25, 25, 64)	12288	['max_pooling2d_5[0][0]']
batch_normalization_102 (Batch Normalization)	(None, 25, 25, 64)	192	['conv2d_102[0][0]']
activation_102 (Activation)	(None, 25, 25, 64)	0	['batch_normalization_102[0][0]']

Loading the train and test data:

Screenshot 2:



The screenshot displays a Jupyter Notebook titled "Sprint_1_CNN_Model_Building.ipynb". The interface includes a top menu bar with options like File, Edit, View, Insert, Runtime, Tools, and Help. On the left, there is a sidebar with icons for file management and a search bar. The main area contains three code cells. The first cell defines an EarlyStopping callback. The second cell, under the heading "Model fitting", fits the model. The third cell, under the heading "Exporting the model", exports the model to a JSON file and saves the weights. The output of the third cell is "Saved model to disk".

```
+ Code + Text
[ ] 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, validation_data=testdata)

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
```