DIGITAL NATURALIST (FLORA AND FAUNA) DETECTION

by: SENTHIL MURUGAN S
KARTHICK M

Table of Contents

1.INTRODUCTION

1.1 OVERVIEW

Object detection is a very important task for different applications including autonomous driving,face detection ,video survilence.Current rates of species loss triggered numerous attempts to protect and conserve biodiversity. Species conservation, however, requires species identification skills, a competence obtained through intensive training and experience. Field researchers, land managers, educators, civil servants, and the interested public would greatly benefit from accessible, up-to-date tools automating the process of species identification. Currently, relevant technologies, such as digital cameras, mobile devices, and remote access to databases, are ubiquitously available, accompanied by significant advances in image processing and pattern recognition. The idea of automated species identification is approaching reality. We review the technical status quo on computer vision approaches for plant species identification, highlight the main research challenges to overcome in providing applicable tools, and conclude with a discussion of open and future research thrusts.In our project we are detecting the animal and plants and deploy that model in mobile app.

1.2 PURPOSE

Our visual system is optimized to detect, classify, and identify objects that we encounter in everyday life. Without this skill, we would not be able to recognize components of our visual field as, for instance, potentially useful or threatening.This project is very useful for a naturalist is someone who studies the patterns of nature, identify different kind of flora and fauna in the nature.

2.LITERATURE SURVEY

2.1 EXISTING PROBLEM

Published in: 2017 IEEE International Conference on Data Science and Advanced Analytics (DSAA)
Publisher: IEEE
Conference Location: Tokyo, Japan
Date of Conference: 19-21 Oct. 2017

2.2 PROPOSED SOLUTION

2.2.1. Convolutional neural network (CNN)

Convolutional Neural Network is an effective identification method, developed in recent years, that caused widespread attention. Now, CNN has become one of the most efficient methods in the field of pattern classification and recently, has been used more widely in the field of image processing .Athrough wide verification. CNN consists of one or more pairs of convolutional and max pooling layers. A convolutional layer applies a set of filters that process small local parts of the input where these filters are replicated along with the whole input space. A max-pooling layer generates a lower resolution version of the convolutional layer activations by taking the maximum filter activation from different positions within a specified window. This adds translation invariance and tolerance to minor differences of positions of objects parts. Higher layers use more broad filters that work on lower resolution inputs to process more complex parts of the input. Top fully connected layers finally combine inputs from all positions to do the classification of the overall inputs. This hierarchical organization generates good results in image processing tasks.

2.2.2. Convolutional neural network structure

- *Convolution layer,* the convolution operation extracts different features of the input. The first convolution layer extracts low-level features like edges, lines, and corners. Higher-level layers extract higher-level features.
- *Non-linear layers,* Neural networks in general and CNNs, in particular, rely on a non-linear 'trigger' function to signal distinct recognition of likely features on each hidden layer. CNNs may use a variety of specific functions such as rectified linear units (ReLUs) and continuous trigger (non-linear) functions to efficiently implement this non-linear triggering. A ReLU implements the function $y = \max(x, 0)$, so the input and output sizes of this layer are the same.
- The pooling/subsampling layer reduces the resolution of the features. It makes the features robust against noise and distortion.
- Fully connected layers are often used as the final layers of a CNN. These layers mathematically sum a weighting of the previous layer of features, indicating the precise mix of 'ingredients' to determine a specific target output result. In case of a fully connected layer, all the elements of all the features of the previous layer get used in the calculation of each element of each output feature.

2.2.3. Deep learning proposal

CNN architectures vary with type of images and especially when input image sizes are different. In this paper, the size of input images is considered to be *128 × 128* pixels. The proposed architecture is described in Table 1. After each Conv layer, ReLU activation function is used and for each pooling layer, MaxPooling approach is applied. The fully connected layers are defined as convolutional layers with the filter size of *1 × 1* as it is conventional in MatConvnet (Vedaldi & Lenc, 2015 Vedaldi, A., & Lenc, K. (2015). *Matconvnet: Convolutional neural networks for matlab*. Proceedings of the 23rd ACM international conference on Multimedia, ACM, 689–692. [Google Scholar]) The final layer has *n* units corresponding to *n* and category of leaf datasets. After all layers, a SoftMax loss is placed.

3.THEORITICAL ANALYSIS

3.1 BLOCK DIAGRAM

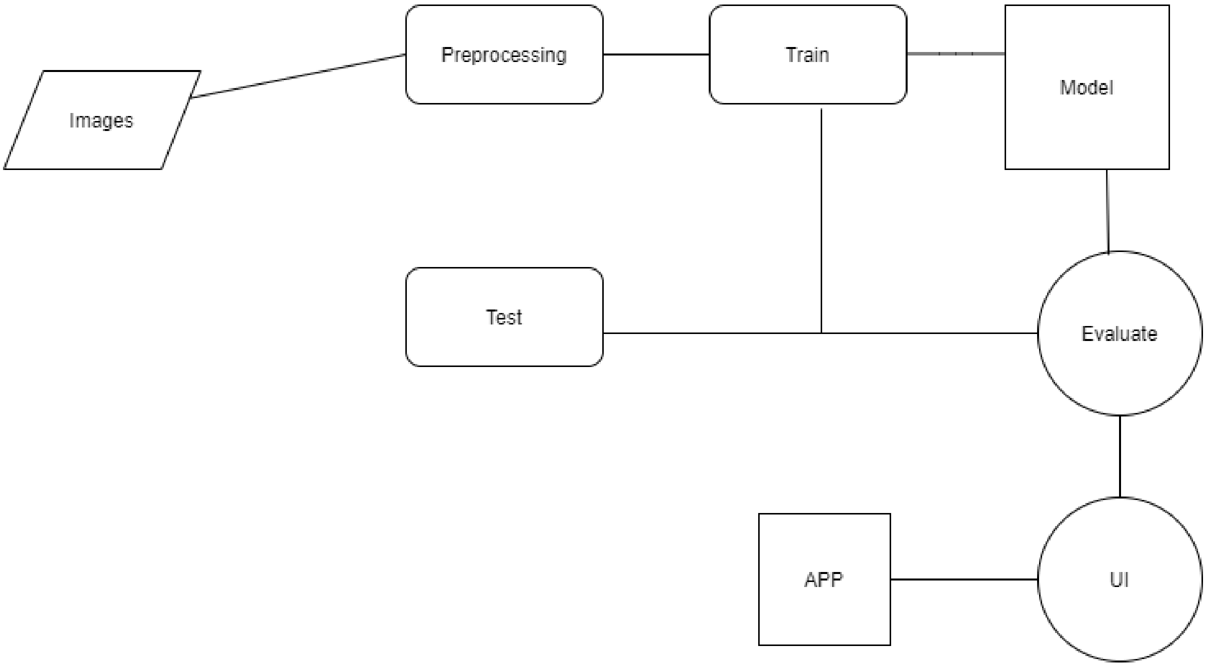


Figure 3.1 Overall Block diagram

3.2 HARDWARE REQUIREMENT

RAM : 1Gb or above
CAMERA : 2Mp
PROCESSOR : moderate processor
STORAGE : 1Gb

3.3 SOFTWARE REQUIREMENT

OS : Above 4.0 (Android)

4.EXPERIMENTAL INVESTIGATION

4.1 PROBLEMS WE FACED

- Lot of research was done on effective dataset for this model
- We had a problem to choose the effective batch size
- We did trail and error with the number of epoch to get the neccessary accuracy level
- To achieve necessary accuracy level we change the size of the dataset and also fluctual the number of epoch

5.FLOW CHART

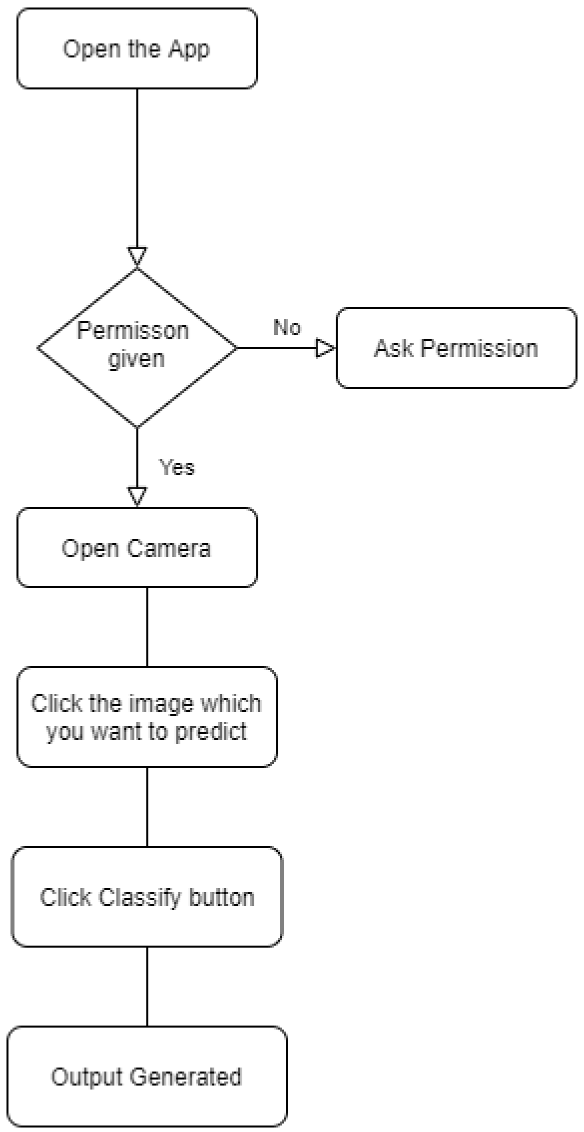
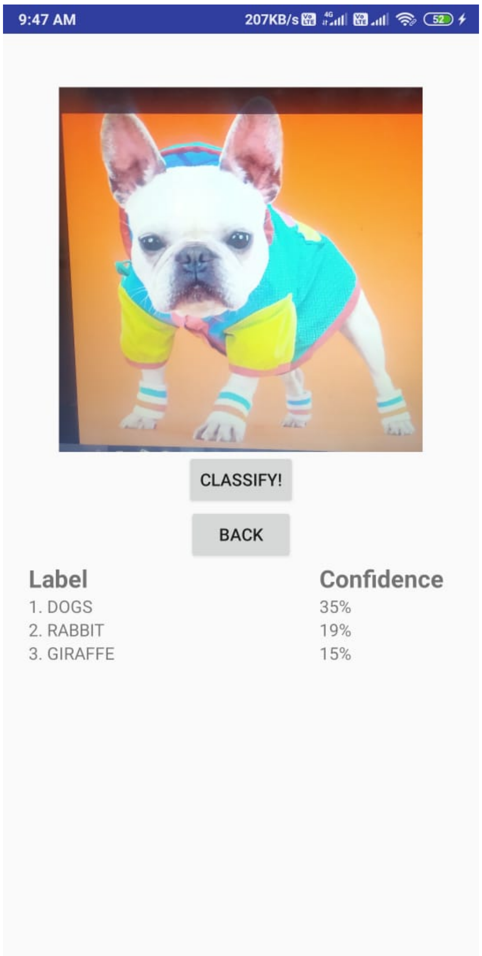
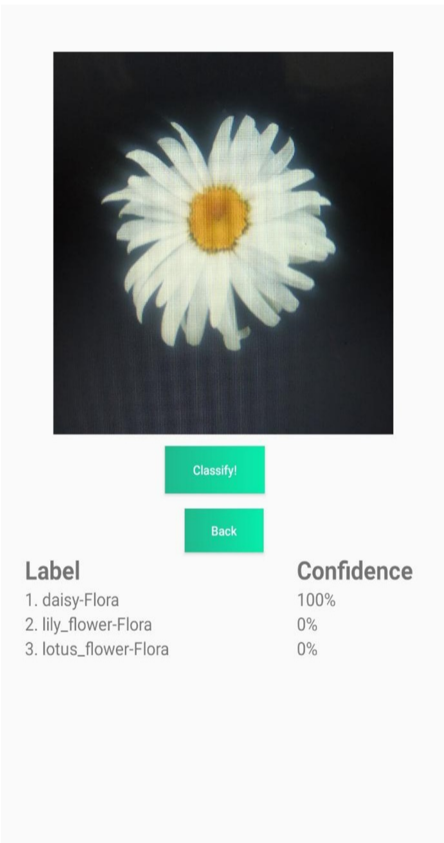


Figure 5.1 Flow diagram

6.RESULT


Welcome
Enter


9:47 AM
207KB/s
CLASSIFY!
BACK
Label
Confidence
1. DOGS
35%
2. RABBIT
19%
3. GIRAFFE
15%


Classify!
Back
Label
Confidence
1. daisy-Flora
100%
2. lily_flower-Flora
0%
3. lotus_flower-Flora
0%

7.ADVANTAGES AND DISADVANTAGES

ADVANTAGES:

- It gives scientists a first-hand look at social behavior.
- It allows researchers to study ideas that cannot be manipulated
- It allows for multiple methods of data collection

DISADVANTAGES:

- process will take some time
- It offers information that can have limited usefulness
- Consume more storage

8. APPLICATIONS

- BUSSINESS: The detection of flora and fauna is highly beneficial in Medical and Research field.The developement of Medicine field is gradually getting increased due to this detection
- AGGRICULTURAL: The diffenrent kinds of flora has been identified which will be useful for the farmers to cultivate flora in their lands.

9.CONCLUSION

The human being is the main responsible of the destruction of fauna and flora.so,people can do many efforts to respect the law of protection of fauna and flora. It is important,because we must live in a health environment and to conserve our animal and tree species .As a conclusion, fauna and flora constitute our environment. The human being is the main responsible of the destruction of fauna and flora.so,people can do many efforts to respect the law of protection of fauna and flora.It is important,because we must live in a health environment and to conserve our animal and tree species.

10.FUTURE SCOPE

Easily analysing the contest of observation utilizing and improvising latest machine learninig devolopement with an android application
R-CNN can be used for the future for faster prediction

11.BIBILOGRAPHY

Baldigo et al. (2017) Baldigo BP, Sporn LA, George SD, Ball JA. Efficacy of environmental DNA to detect and quantify brook trout populations in headwater streams of the Adirondack Mountains, New York. Transactions of the American Fisheries Society. 2017;146(1):99–111. doi: 10.1080/00028487.2016.1243578. [CrossRef] [Google Scholar]

Biggs et al. (2015) Biggs J, Ewald N, Valentini A, Gaboriaud C, Dejean T, Griffiths RA, Foster J, Wilkinson JW, Arnell A, Brotherton P, Williams P, Dunn F. Using eDNA to develop a national citizen science-based monitoring programme for the great crested newt (*Triturus cristatus*) Biological Conservation. 2015;183:19–28. doi: 10.1016/j.biocon.2014.11.029. [CrossRef] [Google Scholar]

Buxton et al. (2017) Buxton AS, Groombridge JJ, Zakaria NB, Griffiths RA. Seasonal variation in environmental DNA in relation to population size and environmental factors. Scientific Reports. 2017;7(1):1–9. doi: 10.1038/srep46294.[PMC free article] [PubMed] [CrossRef] [Google Scholar]

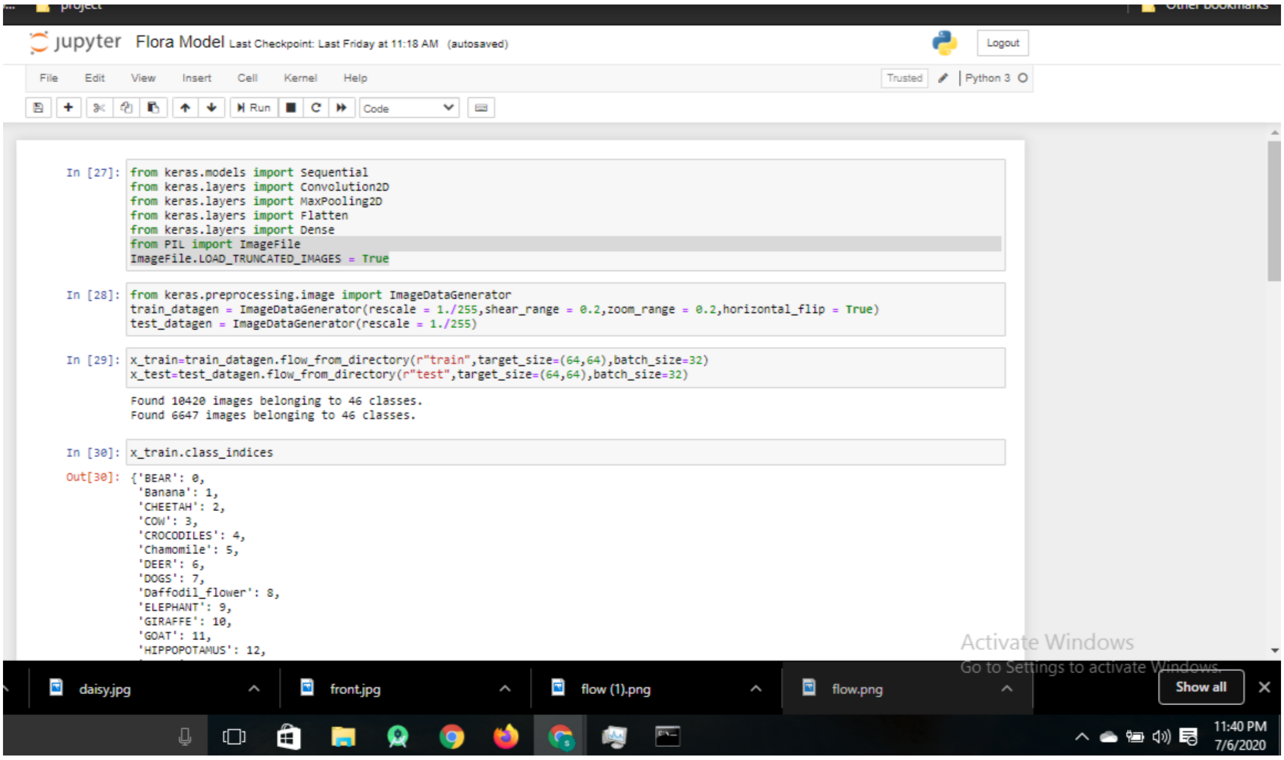
Clusa et al. (2017) Clusa L, Ardura A, Fernández S, Roca AA, García-Vázquez E. An extremely sensitive nested PCR-RFLP mitochondrial marker for detection and identification of salmonids in eDNA from water samples. PeerJ. 2017;5(1):e3045. doi: 10.7717/peerj.3045.[PMC free article] [PubMed] [CrossRef] [Google Scholar]

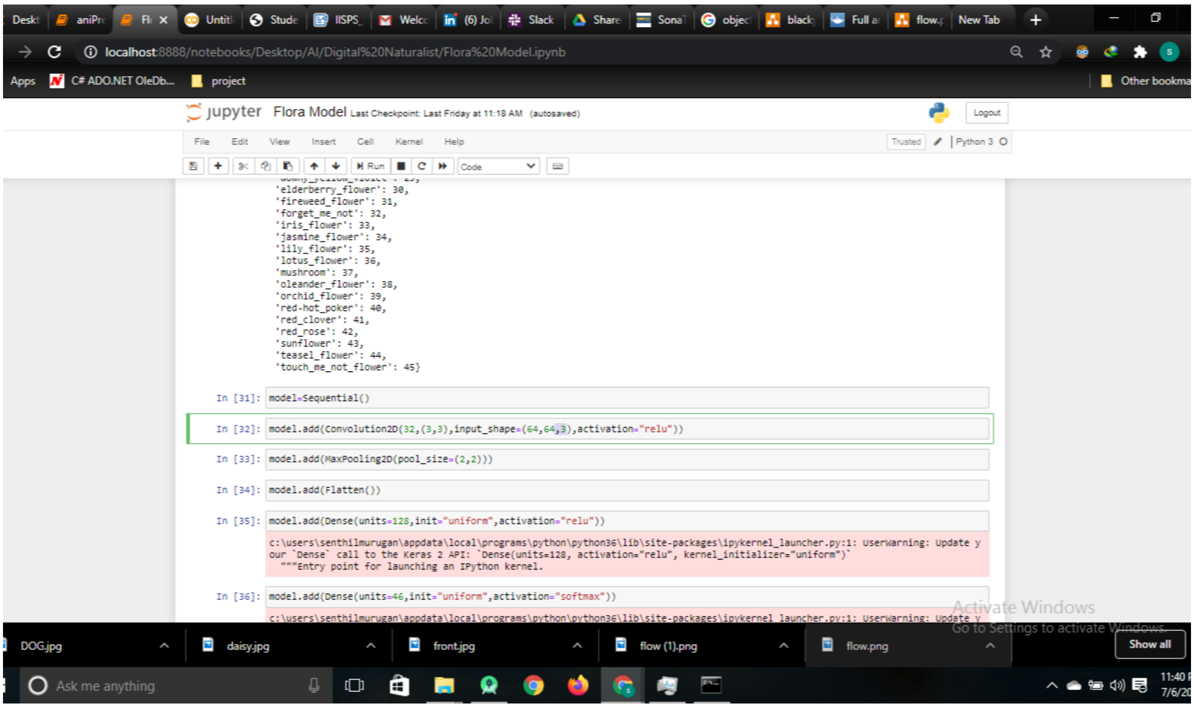
Goldberg, Strickler & Pilliod (2015) Goldberg CS, Strickler KM, Pilliod DS. Moving environmental DNA methods from concept to practice for monitoring aquatic macroorganisms. Biological Conservation. 2015;183:1–3. doi: 10.1016/j.biocon.2014.11.040. [CrossRef] [Google Scholar]

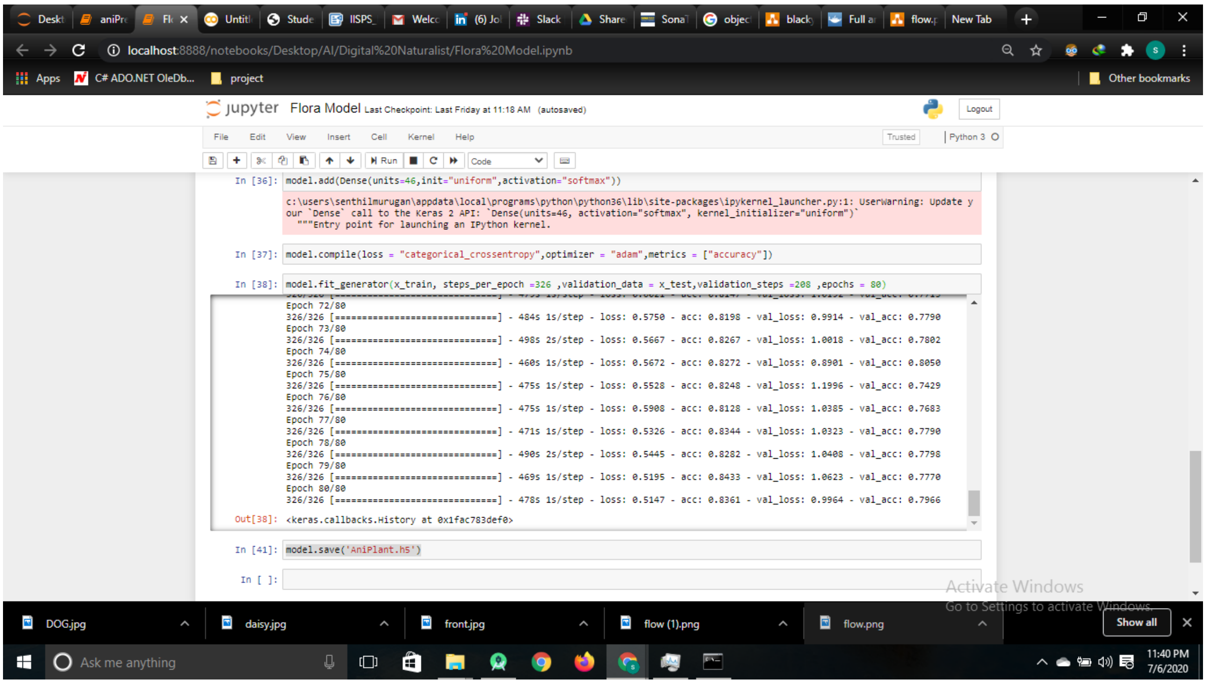
Gosner (1960) Gosner KL. A simplified table for staging anuran embryos larvae with notes on identification. Herpetologica. 1960;16:183–190.[Google Scholar]

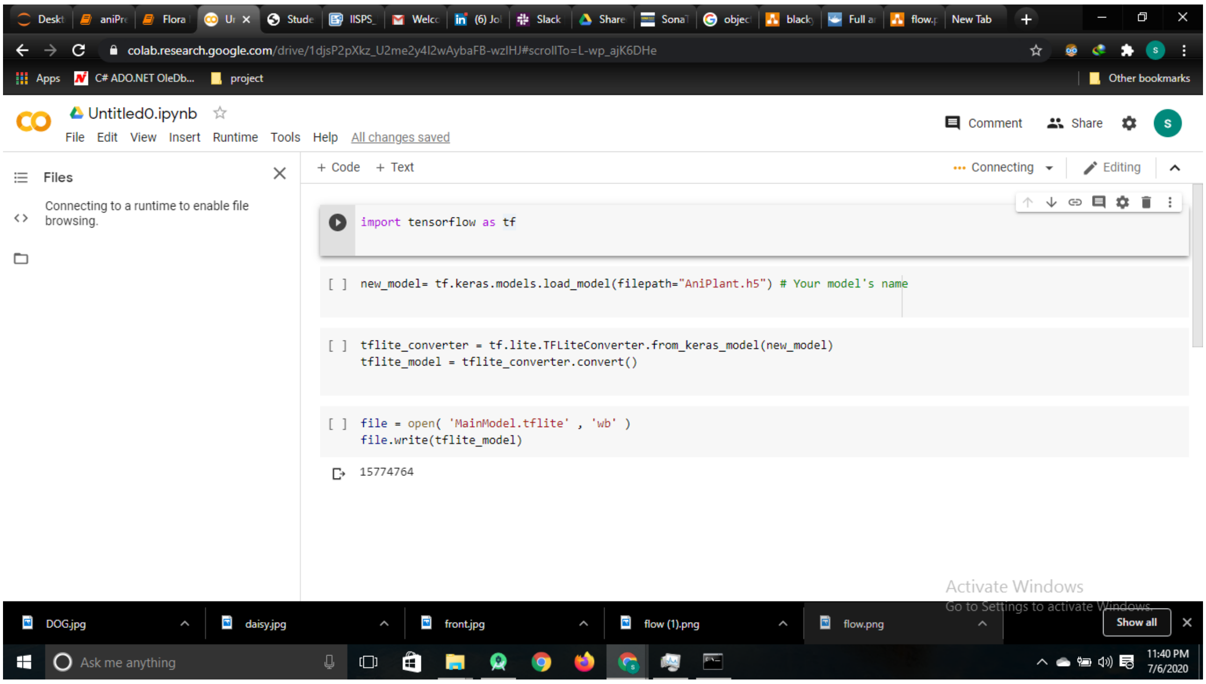
Igawa et al. (2015) Igawa T, Komaki S, Takahara T, Sumida M. Development and validation of PCR-RFLP assay to identify three Japanese brown frogs of the true frog genus *Rana*. Current Herpetology. 2015;34(1):89–94. doi: 10.5358/hsj.34.89. [CrossRef] [Google Scholar]

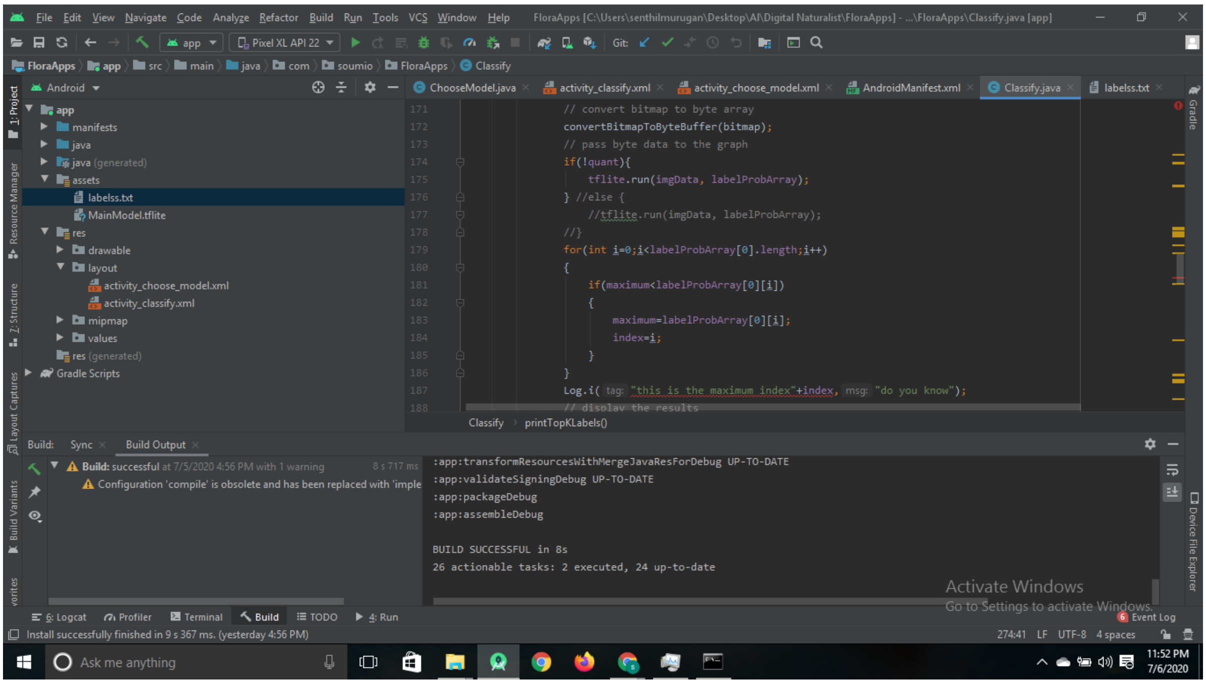
APPENDIX


jupyter Flora Model Last Checkpoint: Last Friday at 11:18 AM (autosaved)
In [27]: from keras.models import Sequential
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense
from PIL import ImageFile
ImageFile.LOAD_TRUNCATED_IMAGES = True
In [28]: from keras.preprocessing.image import ImageDataGenerator
train_datagen = ImageDataGenerator(rescale = 1./255,shear_range = 0.2,zoom_range = 0.2,horizontal_flip = True)
test_datagen = ImageDataGenerator(rescale = 1./255)
In [29]: x_train=train_datagen.flow_from_directory(r"train",target_size=(64,64),batch_size=32)
x_test=test_datagen.flow_from_directory(r"test",target_size=(64,64),batch_size=32)
Found 10420 images belonging to 46 classes.
Found 6647 images belonging to 46 classes.
In [30]: x_train.class_indices
Out[30]: {'BEAR': 0,
'Banana': 1,
'CHEETAH': 2,
'COW': 3,
'CROCODILES': 4,
'Chamomile': 5,
'DEER': 6,
'DOGS': 7,
'Daffodil_flower': 8,
'ELEPHANT': 9,
'GIRAFFE': 10,
'GOAT': 11,
'HIPPOPOTAMUS': 12,
Activate Windows
Go to Settings to activate Windows.
daisy.jpg
front.jpg
flow (1).png
flow.png
Show all
11:40 PM
7/6/2020


localhost:8888/notebooks/Desktop/AI/Digital%20Naturalist/Flora%20Model.ipynb
jupyter Flora Model Last Checkpoint: Last Friday at 11:18 AM (autosaved)
'elderberry_flower': 30,
'fireweed_flower': 31,
'forget_me_not': 32,
'iris_flower': 33,
'jasmine_flower': 34,
'lily_flower': 35,
'lotus_flower': 36,
'mushroom': 37,
'oleander_flower': 38,
'orchid_flower': 39,
'red-hot_poker': 40,
'red_clover': 41,
'red_rose': 42,
'sunflower': 43,
'teasel_flower': 44,
'touch_me_not_flower': 45}
In [31]: model=Sequential()
In [32]: model.add(Convolution2D(32,(3,3),input_shape=(64,64,3),activation="relu"))
In [33]: model.add(MaxPooling2D(pool_size=(2,2)))
In [34]: model.add(Flatten())
In [35]: model.add(Dense(units=128,init="uniform",activation="relu"))
c:\users\senthilmurugan\appdata\local\programs\python\python36\lib\site-packages\ipykernel_launcher.py:1: UserWarning: Update your `Dense` call to the Keras 2 API: `Dense(units=128, activation="relu", kernel_initializer="uniform")`
"""Entry point for launching an IPython kernel.
In [36]: model.add(Dense(units=46,init="uniform",activation="softmax"))
c:\users\senthilmurugan\appdata\local\programs\python\python36\lib\site-packages\ipykernel_launcher.py:1: UserWarning: Update y
DOG.jpg
daisy.jpg
front.jpg
flow (1).png
flow.png
Show all


jupyter Flora Model Last Checkpoint: Last Friday at 11:18 AM (autosaved)
In [36]: model.add(Dense(units=46,init="uniform",activation="softmax"))
c:\users\senthilmurugan\appdata\local\programs\python\python36\lib\site-packages\ipykernel_launcher.py:1: UserWarning: Update your `Dense` call to the Keras 2 API: `Dense(units=46, activation="softmax", kernel_initializer="uniform")`
"""Entry point for launching an IPython kernel.
In [37]: model.compile(loss = "categorical_crossentropy",optimizer = "adam",metrics = ["accuracy"])
In [38]: model.fit_generator(x_train, steps_per_epoch =326 ,validation_data = x_test,validation_steps =208 ,epochs = 80)
Epoch 72/80
326/326 [==============================] - 484s 1s/step - loss: 0.5750 - acc: 0.8198 - val_loss: 0.9914 - val_acc: 0.7790
Epoch 73/80
326/326 [==============================] - 498s 2s/step - loss: 0.5667 - acc: 0.8267 - val_loss: 1.0018 - val_acc: 0.7802
Epoch 74/80
326/326 [==============================] - 460s 1s/step - loss: 0.5672 - acc: 0.8272 - val_loss: 0.8901 - val_acc: 0.8050
Epoch 75/80
326/326 [==============================] - 475s 1s/step - loss: 0.5528 - acc: 0.8248 - val_loss: 1.1996 - val_acc: 0.7429
Epoch 76/80
326/326 [==============================] - 475s 1s/step - loss: 0.5908 - acc: 0.8128 - val_loss: 1.0385 - val_acc: 0.7683
Epoch 77/80
326/326 [==============================] - 471s 1s/step - loss: 0.5326 - acc: 0.8344 - val_loss: 1.0323 - val_acc: 0.7790
Epoch 78/80
326/326 [==============================] - 490s 2s/step - loss: 0.5445 - acc: 0.8282 - val_loss: 1.0408 - val_acc: 0.7798
Epoch 79/80
326/326 [==============================] - 469s 1s/step - loss: 0.5195 - acc: 0.8433 - val_loss: 1.0623 - val_acc: 0.7770
Epoch 80/80
326/326 [==============================] - 478s 1s/step - loss: 0.5147 - acc: 0.8361 - val_loss: 0.9964 - val_acc: 0.7966
Out[38]: <keras.callbacks.History at 0x1fac783def0>
In [41]: model.save('AniPlant.h5')


Untitled0.ipynb
import tensorflow as tf
new_model= tf.keras.models.load_model(filepath="AniPlant.h5") # Your model's name
tflite_converter = tf.lite.TFLiteConverter.from_keras_model(new_model)
tflite_model = tflite_converter.convert()
file = open('MainModel.tflite' , 'wb')
file.write(tflite_model)
15774764


FloraApps [C:\Users\senthilmurugan\Desktop\AI\Digital Naturalist\FloraApps] - ...\FloraApps\Classify.java [app]
ChooseModel.java
activity_classify.xml
activity_choose_model.xml
AndroidManifest.xml
Classify.java
labelss.txt
app
manifests
java
java (generated)
assets
labelss.txt
MainModel.tflite
res
drawable
layout
activity_choose_model.xml
activity_classify.xml
mipmap
values
res (generated)
Gradle Scripts
// convert bitmap to byte array
convertBitmapToByteBuffer(bitmap);
// pass byte data to the graph
if(!quant){
tflite.run(imgData, labelProbArray);
} //else {
//tflite.run(imgData, labelProbArray);
//}
for(int i=0;i<labelProbArray[0].length;i++)
{
if(maximum<labelProbArray[0][i])
{
maximum=labelProbArray[0][i];
index=i;
}
}
Log.i(tag: "this is the maximum index"+index, msg: "do you know");
// display the results
Build: successful at 7/5/2020 4:56 PM with 1 warning
Configuration 'compile' is obsolete and has been replaced with 'imple
:app:transformResourcesWithMergeJavaResForDebug UP-TO-DATE
:app:validateSigningDebug UP-TO-DATE
:app:packageDebug
:app:assembleDebug
BUILD SUCCESSFUL in 8s
26 actionable tasks: 2 executed, 24 up-to-date
Activate Windows
Go to Settings to activate Windows.
Install successfully finished in 9 s 367 ms. (yesterday 4:56 PM)
Ask me anything
11:52 PM
7/6/2020