

CSEE5590/490: Python and Deep Learning Programming (2018 Fall)

Python Project Report

Submitted On:

04 December, 2018

Submitted By:

Fatema Hasta (Class ID: 10)

Vinay Jaibheem (Class ID: 11)

Farid Uddin Ahmed (Class ID: 02)

Zarin Tasnim Sandhie (Class ID: 26)



LIST OF THE DOCUMENTATIONS:

- i. Author
- ii. Objective
- iii. Features
- iv. Input/ Output Screenshots
- v. Implementation & Code Snippet
- vi. Evaluation
- vii. Conclusion
- viii. Links
- ix. References

AUTHORS

This report contains all the documents for the Group project. The assignment was done by Fatema Hasta (Class ID: 10), Vinay Jaibheem (Class ID: 11), Farid Uddin Ahmed (Class ID: 02) and Zarin Tasnim Sandhie (Class ID: 26), all of them are students for the course Python/Deep learning (CSEE5590/490, Fall 2018) at the University of Missouri-Kansas City (UMKC).

OBJECTIVE

The objectives of this Project is:

- It will take input in the form of an image.
- This image can be fed to the model that detects the type of food and classifies the image either into Cup Cake, Chicken wings, cheesecake etc.
- Based on this category, particular feature will be appeared to the user.

METHODOLOGY

- In order to detect the type of food, there will be a food dataset categorized into Cup Cake, Chicken wings, cheesecake etc. For our case,
- Convolutional Neural Networks (CNN) (Inception model) can be used to classify the new image of a food into one of these classes.
- By utilizing Edamam Food API, nutrition details of the food is identified.
- The datasets can later be extended into a more elaborate dataset.

CODE EXPLANATION

- For our project, we have taken 10 classes from “Food-101” dataset.
- The classes we have used are: 'apple_pie', 'baklava', 'bread_pudding', 'carrot_cake', 'cheesecake', 'chicken_curry', 'chicken_wings', 'chocolate_cake', 'club_sandwich' and 'cup_cakes'.
- Each class has 1000 total pictures.
- We have created 2 different folders named “train” and “validation” each of those contain 10 folders for each classes.
- The train folder is used for training dataset and each of the class folders inside train folder contains 7500 pictures.
- The validation folder is used for validating the dataset and each of the class folders inside the validation folder contain 2500 pictures.
- The test folder contain 20 random pictures from different classes which are later used for testing the model.
- All the images are augmented at first in which procedure the pictures are rotated, shifted, whitened etc. to increase the number of data in our dataset.
- Then inception is used as a pre-trained model.
- For the optimization, two different types of optimizers are used in two different pass.
- For the first pass, we used RMSprop and 5 epochs. The optimization is run for the whole model.

- For the second pass, SGD optimizer is used and 10 epochs is used. Here, expect the last two layers, all the other layer are frozen by making it false. The optimization is done only for the last two layers.
- Then the model is saved.
- This model can later be loaded and tested with test images.
- When we predict the name of an unknown food class from the model, this predicted name is saved in a .csv file.
- Then we pass this food name from the csv file to Edamam Api.
- Edamam Api has two features: one is to extract recipe and another is to extract calorie and nutrition chart from the recipe.
- At first, we extracted the recipe and then using that recipe, we extracted the calorie and nutrition.

CODE SNIPPETS

Code for Image Classification with Inception Model on Food 101 dataset:

```

Editor - C:\Users\zsqgd\Documents\GitHub\CS5590-Python-deep-learning\Project\Python-DeepLearning-Project-master\Python_Project_Food\project_keras.py
1.twitter_airline_sentiment_CNN.py 2a.twitter_airline_sentiment_RNN.py 2b.twitter_airline_sentiment_LSTM.py 4.image_classification.py project_keras.py
1 from keras.applications.inception_v3 import InceptionV3
2 from keras.layers import Input
3 from keras.preprocessing.image import ImageDataGenerator
4 from keras.models import Sequential, Model
5 from keras.layers import Dense, Dropout, Activation, Flatten
6 from keras.layers import Convolution2D, MaxPooling2D, ZeroPadding2D, GlobalAveragePooling2D, AveragePooling2D
7 from keras.layers.normalization import BatchNormalization
8 from keras.callbacks import ModelCheckpoint, TensorBoard, CSVLogger
9 import keras.backend as K
10 from keras.optimizers import SGD, RMSprop, Adam
11 from time import time
12
13 train_path = '/home/fatema/Downloads/Python_Project_Food/train'
14 validation_path = '/home/fatema/Downloads/Python_Project_Food/validation'
15 test_path = '/home/fatema/Downloads/Python_Project_Food/test/'
16 classes = ['apple_pie', 'baklava', 'bread_pudding', 'carrot_cake', 'cheesecake', 'chicken_curry', 'chicken_wings', 'chocolate_cake', 'club_sandwich', 'cup_cak
17 ##### Set up Image Augmentation
18 print("Setting up ImageDataGenerator")
19 datagen = ImageDataGenerator(
20     featurewise_center=False, # set input mean to 0 over the dataset
21     samplewise_center=False, # set each sample mean to 0
22     featurewise_std_normalization=False, # divide inputs by std of the dataset
23     samplewise_std_normalization=False, # divide each input by its std
24     zca_whitening=False, # apply ZCA whitening
25     rotation_range=45, # randomly rotate images in the range (degrees, 0 to 180)
26     width_shift_range=0.125, # randomly shift images horizontally (fraction of total width)
27     height_shift_range=0.125, # randomly shift images vertically (fraction of total height)
28     horizontal_flip=True, # randomly flip images
29     vertical_flip=False, # randomly flip images
30     rescale=1./255,
31     fill_mode='nearest')
32 generator = datagen.flow_from_directory(train_path,target_size=(299,299),classes=classes, batch_size=10)
33 val_generator = datagen.flow_from_directory(validation_path,target_size=(299,299),classes=classes, batch_size=5)
34 test_generator = datagen.flow_from_directory(test_path,target_size=(299,299),class_mode=None,shuffle=False,batch_size=20)
35 K.clear_session()
36 base_model = InceptionV3(weights='imagenet', include_top=False, input_tensor=Input(shape=(299, 299, 3)))
37 x = base_model.output
38 x = GlobalAveragePooling2D()(x)
39 # x = Flatten()(x)
40 x = Dense(4096)(x)
41 x = BatchNormalization()(x)
42 x = Activation('relu')(x)

```

```

1.twitter_airline_sentiment_CNN.py x 2a.twitter_airline_sentiment_RNN.py x 2b.twitter_airline_sentiment_LSTM.py x 4.image_classification.py x project_keras.py x
40 x = Dense(4096)(x)
41 x = BatchNormalization()(x)
42 x = Activation('relu')(x)
43 x = Dropout(.5)(x)
44 predictions = Dense(10, activation='softmax')(x)
45
46
47 model = Model(input=base_model.input, output=predictions)
48
49 for layer in base_model.layers:
50     layer.trainable = False
51
52 model.compile(optimizer='RMSprop', loss='categorical_crossentropy',
53             metrics=['accuracy'])
54
55 print("First pass")
56 checkpointer = ModelCheckpoint(filepath='/home/fatema/Downloads/Re%3a_Regarding_Python%2fDeep_learning_Project/first.3.{epoch:02d}-{val_loss:.2f}.hdf5', verbose=1)
57 csv_logger = CSVLogger('first.3.log')
58 tensorboard=TensorBoard(log_dir="/home/fatema/Downloads/Re%3a_Regarding_Python%2fDeep_learning_Project/logs/{}".format(time())) #tensorboard declaration to vis
59 model.fit_generator(generator,
60                   validation_data=val_generator,
61                   validation_steps=500,
62                   steps_per_epoch=750,
63                   nb_epoch=5,
64                   verbose=1,
65                   callbacks=[csv_logger, checkpointer, tensorboard])
66 for layer in model.layers[:172]:
67     layer.trainable = False
68 for layer in model.layers[172:]:
69     layer.trainable = True
70 print("Second pass")
71 model.compile(optimizer=SGD(lr=0.0001, momentum=0.9), loss='categorical_crossentropy', metrics=['accuracy'])
72 checkpointer = ModelCheckpoint(filepath='/home/fatema/Downloads/Re%3a_Regarding_Python%2fDeep_learning_Project/second.3.{epoch:02d}-{val_loss:.2f}.hdf5', verbose=1)
73 csv_logger = CSVLogger('second.3.log')
74 model.fit_generator(generator,
75                   validation_data=val_generator,
76                   validation_steps=500, steps_per_epoch=750,
77                   nb_epoch=10,
78                   verbose=1,
79                   callbacks=[csv_logger, checkpointer, tensorboard]) #returns history object that has all the loss values.
80 model.save('model.h5')

```

Code for loading the saved model and testing with test files:

```

11
12 ##### Set up Image Augmentation
13 print("Setting up ImageDataGenerator")
14 datagen = ImageDataGenerator(
15     featurewise_center=False, # set input mean to 0 over the dataset
16     samplewise_center=False, # set each sample mean to 0
17     featurewise_std_normalization=False, # divide inputs by std of the dataset
18     samplewise_std_normalization=False, # divide each input by its std
19     zca_whitening=False, # apply ZCA whitening
20     rotation_range=45, # randomly rotate images in the range (degrees, 0 to 180)
21     width_shift_range=0.125, # randomly shift images horizontally (fraction of total width)=1/8
22     height_shift_range=0.125, # randomly shift images vertically (fraction of total height)=1/8
23     horizontal_flip=True, # randomly flip images
24     vertical_flip=False, # randomly flip images
25     rescale=1./255,
26     fill_mode='nearest')
27
28 generator = datagen.flow_from_directory(train_path,target_size=(299,299),classes=classes, batch_size=10)
29 val_generator = datagen.flow_from_directory(validation_path,target_size=(299,299),classes=classes, batch_size=5)
30 test_generator = datagen.flow_from_directory(test_path,target_size=(299,299),class_mode=None,shuffle=False,batch_size=20)
31 model = load_model('second.3.07-0.85.hdf5')
32
33 #score = model.evaluate_generator(val_generator,500)
34 #print(score)
35
36 test_generator.reset()
37 pred=model.predict_generator(test_generator,verbose=1,steps=1)
38 predicted_class_indices=np.argmax(pred,axis=1)
39
40 labels = (generator.class_indices)
41 print(labels)
42 labels = dict((v,k) for k,v in labels.items())
43 predictions = [labels[k] for k in predicted_class_indices]
44 print(predictions)
45 filenames=test_generator.filenames
46 results=pd.DataFrame({"Filename":filenames,
47                     "Predictions":predictions})
48 results.to_csv("results1.csv",index=False)

```

Code for Nutrition Api:

```
1 import requests
2 import json
3 import csv
4 import numpy as np
5 import pandas as pd
6
7 APP_ID = '4c5f3ddb'
8
9 API_KEY = '43600b1cc8bac1409d566f5613e43803'
10
11 API_ENDPOINT = 'https://api.edamam.com/api/nutrition-details?app_id=$APP_ID&app_key=$API_KEY'
12
13 file = pd.read_csv('C:\Vinay\Python_Deep_Learning\Project\Python_Project_Food\results.csv')
14 recipe_names = []
15 with open('C:\Vinay\Python_Deep_Learning\Project\Python_Project_Food\results.csv') as csv_file:
16     csv_reader = csv.DictReader(csv_file, delimiter=',')
17     line_count = 0
18     #print(csv_reader)
19     for row in csv_reader:
20         recipe_names.append(f'{row["Predictions"]}')
```

24 &from=0&to=3|

```
25
26 for i in recipe_names:
27     #f = open("nutrition_output.txt", 'w')
28     print("For", i)
29     #f.write(print("For", i))
30     url_one = "https://api.edamam.com/search?q="+i+"&app_id=dbc27b05&app_key=3a867d774216ecb55eba2387ecc3374d"
31     response = requests.get(url_one+"&app_id=dbc27b05&app_key=3a867d774216ecb55eba2387ecc3374d")
32     data = response.json()
33     #print(type(data))
34     #print(data)
35
36     hits = data["hits"]
37     #print(hits)
38     #print("\n")
39     recepie = hits[0]
40     #print(recepie)
41     recipe = recepie["recipe"]
42     #print(recipe)
```

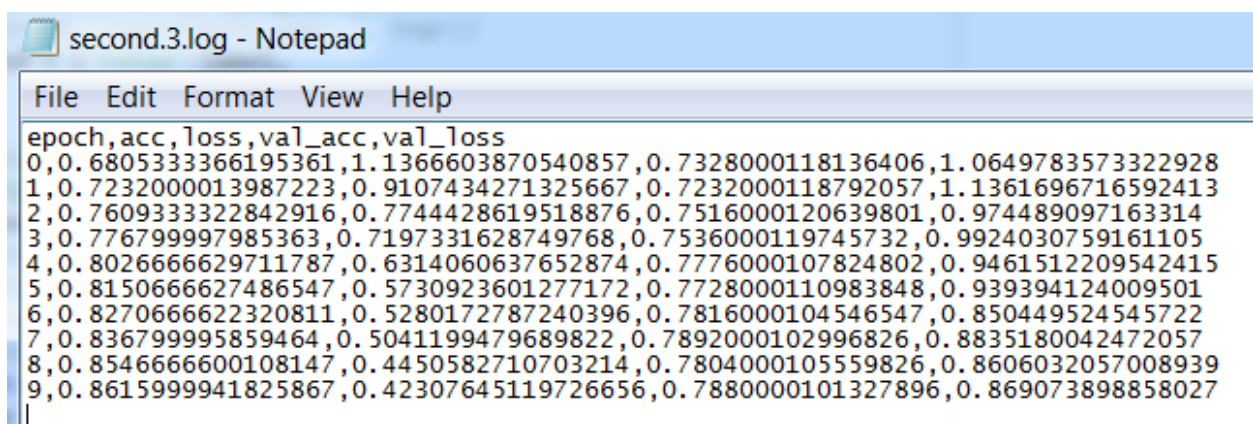
```

37 #print(hits)
38 #print("\n")
39     recepie = hits[0]
40 #print(recepie)
41     recipe = recepie["recipe"]
42 #print(recipe)
43     ingredients = recipe["ingredientLines"]
44 #print(json.dumps(ingredients))
45 #print(type(str))
46 #print(type(ingr))
47     label = recipe["label"]
48
49     print("Recipe is: ",label)
50     #f.write(print("Recipe is: ", label))
51
52 #print("\n",ingredients, "\n")
53 #print(label)
54 #ingr = json.dumps(ingredients)
55     ingr = ingredients
56     print("Ingredients are: ",ingr)
57     #f.write(print("Ingredients are: ", ingr))
58     data_second = {'title':label,
59                   'ingr':ingr}
60
61 #print(json.dumps(data_second, indent=4))
62     url = "https://api.edamam.com/api/nutrition-data?app_id=4c5f3ddb&app_key=43600b1cc8bac1409d566f5613e43803"
63     req = requests.get(url,data_second)
64 #print(req.status_code)
65 #print(req.text)
66     inventory = json.loads(req.text)
67 #print(inventory)
68     for key in inventory:
69         if key == 'calories':
70             print(key,inventory[key])
71             #f.write(print(key, inventory[key]))
72
73         if key == 'totalNutrients':
74             print(key,inventory[key])
75             #f.write(print(key, inventory[key]))
76         #for key_two in inventory[key]
77
78     print("-----")

```

OUTPUT SNIPPET

Log file for the second pass with 10 epochs:



```

second.3.log - Notepad
File Edit Format View Help
epoch,acc,loss,val_acc,val_loss
0,0.6805333366195361,1.1366603870540857,0.7328000118136406,1.0649783573322928
1,0.7232000013987223,0.9107434271325667,0.7232000118792057,1.1361696716592413
2,0.7609333322842916,0.7744428619518876,0.7516000120639801,0.974489097163314
3,0.776799997985363,0.7197331628749768,0.7536000119745732,0.9924030759161105
4,0.8026666629711787,0.6314060637652874,0.7776000107824802,0.9461512209542415
5,0.8150666627486547,0.5730923601277172,0.7728000110983848,0.939394124009501
6,0.8270666622320811,0.5280172787240396,0.7816000104546547,0.850449524545722
7,0.836799995859464,0.5041199479689822,0.7892000102996826,0.8835180042472057
8,0.8546666600108147,0.4450582710703214,0.7804000105559826,0.8606032057008939
9,0.8615999941825867,0.42307645119726656,0.7880000101327896,0.869073898858027

```

Predicated result for the test files:

	A	B
1	Filename	Predictions
2	test_folder/test1.jpg	baklava
3	test_folder/test10.jpg	apple_pie
4	test_folder/test11.jpg	apple_pie
5	test_folder/test12.jpg	apple_pie
6	test_folder/test13.jpg	chocolate_cake
7	test_folder/test14.jpg	chocolate_cake
8	test_folder/test15.jpg	chicken_wings
9	test_folder/test16.jpg	chicken_wings
10	test_folder/test17.jpg	apple_pie
11	test_folder/test18.jpg	baklava
12	test_folder/test19.jpg	baklava
13	test_folder/test2.jpg	cheesecake
14	test_folder/test20.jpg	carrot_cake
15	test_folder/test3.jpg	chocolate_cake
16	test_folder/test4.jpg	chicken_wings
17	test_folder/test5.jpg	cup_cakes
18	test_folder/test6.jpg	carrot_cake
19	test_folder/test7.jpg	apple_pie
20	test_folder/test8.png	club_sandwich
21	test_folder/test9.jpg	apple_pie

Output from the nutrition Api:

The screenshot shows the PyCharm IDE with the following components:

- Project View:** Shows a project structure with folders like 'logs', 'test', 'train', 'validation', and files like 'first3.01-2.40.hdf5', 'load_model.py', 'nutrition_output.txt', 'nutritionapi.py', 'project_keras.py', and 'results.csv'.
- Code Editor:** Displays the contents of 'nutritionapi.py', which includes imports for requests, json, csv, numpy, and pandas. It defines APP_ID, API_KEY, and API_ENDPOINT, and contains logic to read a CSV file and process recipe names.
- Run Console:** Shows the output of the script. It identifies 'Baklava' and 'cheesecake' recipes, lists their ingredients, and provides total nutrient information (Energy, Fat, FASAT) for each.

```

1  import requests
2  import json
3  import csv
4  import numpy as np
5  import pandas as pd
6
7  APP_ID = '4c5f3ddb'
8
9  API_KEY = '43600b1ec8ba01409d566f5613e43803'
10
11 API_ENDPOINT = 'https://api.ednam.com/api/nutrition-details?app_id=$APP_ID&app_key=$API_KEY'
12
13 file = pd.read_csv('C:\Vinay\Python_Deep_Learning\Project\Python_Project_Food\results.csv')
14 recipe_names = []
15
16 for i in recipe_names:
17     for key in inventory:
18         if key == 'calories':

```

Run Console Output:

```

C:\Users\vinay\Anaconda3\python.exe C:/Vinay/Python_Deep_Learning/Project/Python_Project_Food/nutritionapi.py
For Baklava
Recipe is: Rolled Baklava recipes
Ingredients are: ['1/2 cup (1 stick) unsalted butter, melted', '1 1/2 cups coarsely ground almonds (7 ounces)', '1 1/2 cups coarsely ground walnuts (7 ounces)',
calories 813
totalNutrients {'ENERG_KCAL': {'label': 'Energy', 'quantity': 813.795, 'unit': 'kcal'}, 'FAT': {'label': 'Fat', 'quantity': 92.05985, 'unit': 'g'}, 'FASAT': {'lab
-----
For cheesecake
Recipe is: Grilled Cheesecake
Ingredients are: ['1 tablespoon butter', '2 slices pound cake (any flavor), each piece buttered on one side', '1 small slice cheesecake (any flavor), cut into ve
calories 101
totalNutrients {'ENERG_KCAL': {'label': 'Energy', 'quantity': 101.814, 'unit': 'kcal'}, 'FAT': {'label': 'Fat', 'quantity': 11.517619999999999, 'unit': 'g'}, 'FAS
-----

```

EVALUATION

- The accuracy for the code is quite good. With a total of 15 epochs, we reached a training accuracy of 86% and validation accuracy of 79%. With more time and more epochs, the accuracy can be increased more.
- We used two different kinds of optimizer which had a very positive result on our code.
- When implemented on 20 random test files, most of the predictions were good. Only 3 predictions were wrong.
- The nutrition of the predicted food was calculated accurately with the nutrition api.

CONCLUSION

During this project, we learned about the different kinds of models like AlexNet, ResNet, LeNet, VGG16, VGG19, inception etc. We also learned the used of different optimizers in same model. Due to time shortage, we only worked on 10 classes from food dataset. With proper time and appropriate resources, it will be possible to implement our project with all the classes in the food dataset.

LINKS

Github Link:

<https://github.com/fatemahasta/Python-DeepLearning-Project>

REFERENCES

1. <https://github.com/stratospark/food-101-keras>
2. https://github.com/matterport/Mask_RCNN
3. <http://blog.stratospark.com/deep-learning-applied-food-classification-deep-learning-keras.html>