National College of Ireland

Diet Food Recipe Recommendation by Ingredients using Image Processing and Object Detection

MSc Research Project
Data Analytics

Sindhuja
Student ID: x21133409

School of Computing
National College of Ireland

Supervisor: Mohammed Hasanuzzman

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Sindhuja
Student ID:	X21133409
Programme:	Data Analytics
Year:	2022
Module:	MSc Research Project
Supervisor:	Mohammed Hasanuzzman
Submission Due Date:	15/12/2022
Project Title:	Diet Food Recipe Recommendation by Ingredients using Image Processing and Object Detection
Word Count:	900
Page Count:	6

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	15/12/2022

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Diet Food Recipe Recommendation by Ingredients using Image Processing and Object Detection

Sindhuja

x18138977

1 Introduction

The manual for the configuration includes a detailed and step-by-step overview of the full process of putting the configuration into effect. The hardware and software specifications are outlined for the research project on the topic of "Predicting the winner of a tennis match using machine learning techniques." The subject matter of the project is "Predicting the winner of a tennis match using machine learning techniques." Recipe Suggestions for Healthy Consumption Arranged in Accordance with the Ingredients They Require Resulting from the Processing of Images and the Identification of Objects Utilizing a Model of Deep Learning This project's objective is to compile recipe ideas for nutritious eating that are categorised according to the ingredients used in the dishes. CNN is leading this effort. Our web application's capabilities are about to be increased, and one of the upcoming additions is the capacity to recognise images.

2 System Specification

Colab is a catch-all term for the Google Collaboratory cloud platform, which was utilised in the process of carrying out the implementation of this project. The colab is equipped to accommodate both graphical processing units (GPUs) and tensor processing units (TPUs). Bisong (2019

2.1 Hardware

The GPU instance at Google Colab had 250 GB of memory,
while the RAM capacity was 13 GB.
There were two virtual CPUs at Google Colab, each operating at 2.2 GHz.
There was a total of 32 gigabytes of free space on the drive.

2.2 Software

Python was the programming language that was used, including for cleaning, in order to get the project up and running. Python was used to carry out all of the activities necessary for the encoding, implementation of dimension reduction, and evaluation stages of the pre-processing step

3 Importing Libraries

The cloud platform already has pre-defined versions of a number of the necessary libraries installed when it is initially provisioned. Importing the essential libraries for the other programmes was done only when it was absolutely necessary. This was done every time it was deemed essential to do so. At this stage in the process, the import of the necessary libraries ought to be the primary focus of attention.

```
[ ]  import pandas as pd

     import numpy as np
     import tensorflow as tf
     from tensorflow import keras
     from keras import layers
     from keras.preprocessing.image import ImageDataGenerator
     # train_generator = train_datagen.flow_from_directory(
     #     r'C:\\Users\\rashi\\OneDrive\\Desktop\New folder 1\\Launchcode\\Tensorflow\\chest_x-ray_
     #     target_size=(300, 300),
     #     batch_size=128,
     #     class_mode='binary'
     # )
```

Figure 1: Importing Libraries

4 Data Extraction

4.1 Importing Files

During this stage of the process, the data set will be mounted on Google Drive, and after that, the file will be imported from Google Drive.

```
[ ]  '''
     This cell provides the link to the directory of the Kaggle API token file
     '''
     os.getcwd()
     os.environ['KAGGLE_CONFIG_DIR'] = 'drive/MyDrive/Kaggle'

[ ]  '''
     Here we download and unzip the dataset downloaded from Kaggle
     '''

     !kaggle datasets download -d kritikseth/fruit-and-vegetable-image-recognition
     !unzip -q fruit-and-vegetable-image-recognition.zip -d data
     !rm fruit-and-vegetable-image-recognition.zip

     Downloading fruit-and-vegetable-image-recognition.zip to /content
     100% 1.98G/1.98G [01:35<00:00, 24.4MB/s]
     100% 1.98G/1.98G [01:35<00:00, 22.2MB/s]
```

Figure 2: Importing Data

4.2 Set Path

At this point, the directory where the data set is located is specified, and then the data itself is read.

```
import pandas as pd
```

```
import numpy as np
import tensorflow as tf
from tensorflow import keras
from keras import layers
from keras.preprocessing.image import ImageDataGenerator
# train_generator = train_datagen.flow_from_directory(
#     r'C:\\Users\\rashi\\OneDrive\\Desktop\New folder 1\\Launchcode\\Tensorflow\\c
#     target_size=(300, 300),
#     batch_size=128,
#     class_mode='binary'
# )
```

Figure 3: Working directory path

4.3 Reading the data

```
Here we download and unzip the dataset downloaded from Kaggle
'''

!kaggle datasets download -d kritikseth/fruit-and-vegetable-image-recognition
!unzip -q fruit-and-vegetable-image-recognition.zip -d data
!rm fruit-and-vegetable-image-recognition.zip
```

```
Downloading fruit-and-vegetable-image-recognition.zip to /content
100% 1.98G/1.98G [01:35<00:00, 24.4MB/s]
100% 1.98G/1.98G [01:35<00:00, 22.2MB/s]
```

Figure 4: Reading Data

5 Exploratory Data Analysis (EDA)

Python profiling was utilized throughout the process of carrying out the exploratory data analysis. The python profiling feature is a single line of code that, when executed, provides a more complete comprehension of the data's insights. It does an analysis of the data and then generates a report in HTML format detailing all the missing values, outliers, class balance, correlations, and other fundamental information on the dataset, etc.

5.1 Checking Class Imbalance

The class should be equally balanced to get efficient results, hence the process of under sampling or over sampling takes place depending on the data. Here in this dataset, the class was equally balanced.

```
# this is the augmentation configuration we will use for training
train_datagen = ImageDataGenerator(
        rescale=1./255,
        shear_range=0.2,
        zoom_range=0.2,
        horizontal_flip=True)

# this is the augmentation configuration we will use for testing:
# only rescaling
test_datagen = ImageDataGenerator(rescale=1./255)

# this is a generator that will read pictures found in
# subfolers of 'data/train', and indefinitely generate
# batches of augmented image data
train_generator = train_datagen.flow_from_directory(
        'data/train',  # this is the target directory
        target_size=(150, 150),  # all images will be resized to 150x150
        batch_size=batch_size,
        class_mode='categorical')  # since we use binary_crossentropy loss, we ne
```

```
Found 3115 images belonging to 36 classes.
Found 351 images belonging to 36 classes.
Found 359 images belonging to 36 classes.
```

Figure 5: Class Imbalance check

5.2 Defines and compiles the model

```
[ ]  NUM_CLASSES = len(os.listdir("data/train"))

[ ]  # Defines & compiles the model
     model = tf.keras.models.Sequential([
         tf.keras.layers.Conv2D(16, (3,3), activation='relu', input_shape=(150, 150, 3)),
     tf.keras.layers.MaxPooling2D(2, 2),
     keras.layers.Dropout(rate=0.15), #adding dropout regularization throughout the model to deal with overfitting
     # The second convolution
     tf.keras.layers.Conv2D(32, (3,3), activation='relu'),
     tf.keras.layers.MaxPooling2D(2,2),
     keras.layers.Dropout(rate=0.1),
     # The third convolution
     tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
     tf.keras.layers.MaxPooling2D(2,2),
     keras.layers.Dropout(rate=0.10),
     # Flatten the results to feed into a DNN
     tf.keras.layers.Flatten(),
     # 512 neuron hidden layer
     tf.keras.layers.Dense(512, activation='relu'),
```

Figure 6: Defines and compiles the model

The Model defines and compiles adding dropout regularization throughout the model to deal with overfitting. A loss function as well as an optimizer must be specified by us. By utilizing the compile attribute, we can compile a model. Before we use it, let us first have a look at the parameters it has.

6. Training and testing dataset

In this stage, the data set was split into training and testing in 70:30 ratio.

```
from tensorflow.keras.utils import load_img, img_to_array
# from keras.applications.vgg16 import preprocess_input



#load the image
my_image = load_img('data/test/capsicum/Image_1.jpg', target_size=(150, 150))

#preprocess the image
my_image = img_to_array(my_image)
my_image = my_image.reshape((1, my_image.shape[0], my_image.shape[1], my_image.shape[2]))
# my_image = preprocess_input(my_image)

#make the prediction
prediction = model.predict(my_image)
prediction
```

```
1/1 [==============================] - 0s 14ms/step
array([[0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0.]], dtype=float32)
```

Figure 7: Training and testing dataset

7. Deep Learning Model- Convolutional Neural Network or CNN

```
[ ]  # Defines & compiles the model
    model = tf.keras.models.Sequential([
        tf.keras.layers.Conv2D(16, (3,3), activation='relu', input_shape=(150, 150,
    tf.keras.layers.MaxPooling2D(2, 2),
    keras.layers.Dropout(rate=0.15), #adding dropout regularization throughout the m
    # The second convolution
    tf.keras.layers.Conv2D(32, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    keras.layers.Dropout(rate=0.1),
    # The third convolution
    tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    keras.layers.Dropout(rate=0.10),
    # Flatten the results to feed into a DNN
    tf.keras.layers.Flatten(),
    # 512 neuron hidden layer
    tf.keras.layers.Dense(512, activation='relu'),

    # 3 output neuron for the 3 classes of Animal Images
    tf.keras.layers.Dense(NUM_CLASSES, activation='softmax')
    ])
```

Figure 8: Convolutional Network model Implementation

Figure 8 says defines the compilation of the model that has been used to create a network design for deep learning that acquires knowledge directly from data. CNNs are very helpful for recognizing objects, classifications, and categories in photos by discovering patterns in the images themselves.

Reference

Google colaboratory (no date) *Google Colab*. Google. Available at: https://colab.research.google.com/drive/1f8IQXiVi5myVbF3FrNkIpNE0RWM0jBxP#scrollTo=d2_JYOsJZ5iv (Accessed: December 14, 2022).