MODELLING NON-OVERT FEATURES FOR FOOD ORIGIN RECOGNITION

BY

LIM JIA HONG

Supervised By

Dr. Aun Yichiet

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF INFORMATION SYSTEM (HONS)
BUSINESS INFORMATION SYSTEM
Faculty of Information and Communication Technology

(Kampar Campus)

JANUARY 2020

UNIVERSITI TUNKU ABDUL RAHMAN

REPORT STATUS DECLARATION FORM

Title: Modelling non overt features for food origin recognition

Academic Session: JAN 2020

I LIM JIA HONG

(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in

Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,

(Author's signature) (Supervisor's signature)\

Address:

1619 JALAN SATU,
13200 KEPALA BATAS,
PULAU PINANG

Aun YC

Supervisor's name

Date: 19.4.2020 **Date**: 23/4/2020

MODELLING NON-OVERT FEATURES FOR FOOD ORIGIN RECOGNITION

BY

LIM JIA HONG

Supervised By

Dr. Aun Yichiet

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF INFORMATION SYSTEM (HONS)
BUSINESS INFORMATION SYSTEM

Faculty of Information and Communication Technology

(Kampar Campus)

JANUARY 2020

DECLARATION OF ORIGINALITY

I declare that this report entitled "**MODELLING NON-OVERT FEATURES FOR FOOD ORIGIN RECOGNITION**" is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : 

Name : LIM JIA HONG

Date : 19 APRIL 2020

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisors, Dr.Aun Yichiet who has given me this bright opportunity to engage in an IT project. It is my first step to establish a career in object recognition area. A million thanks to you. Finally, I must say thanks to my parents and my family for their love, support and continuous encouragement throughout the course.

ABSTRACT

Good food is differentiated by taste, not by appearance. The rationale of increasing popularity of a food or food provider is inherited from the consumer trusts on their corresponding branding that guarantee great and familiar taste. Despite food are generally distinguishable by their presentation and plating; it is less obvious in the case of simple comfort food as their unique taste are often masked by their highly similar appearances. Two common motivations in identifying food origins are: (1) on the curiosity of where the food comes from and (2) on the trustworthy to confirm the origin of familiar food when generic packaging is used or due to absence of physical labels. This paper designs a food origin classification system using machine-learning to accurately classify local cuisines that is less discriminative with plating cues. The classifier model is constructed using CNN technique and extensive labeling to address the lack of limited discriminative features due to usage of simple and limited ingredients. These approaches are used for training the dataset in order to obtain high accuracy in tracing the food origin. Generally, the system process is divided into two phases, which are data collection and data processing. The experimental result shows that the model is highly accurate with correct detection up to 79% of true positive rate.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AWS	Amazon Web Services
CNN	Convolutional Neural Network
ANN	Artificial Neural Network
CL	Continual Learning

Chapter 1: Introduction

1.1 Problem Statement and Motivation

First of all, the one of the problems would be the there are no any existing food classification system is origin-aware, they just build a software that provide the function of object recognition, which mean it only able to detect what is the object name. In addition, the location metadata is dynamic, limited and unreliable. For example, when a food image that captured by someone do not mean the place he/she capture the food is the actual location that selling the food. Therefore, it increases the difficulty in checking the actual location of food, although we are able to check the location of the image captured in image properties.

Other than that, existing feature sets for food classification is not temporal-aware such that the behaviors of changes in terms of appearances, ingredients, serving size and plating design are not modelled. In fact, it is hard to automate food recognition with high accuracy when the same type of foods comes with different appearances, while the different food type might have the similarity in appearances. Generally, not only the recipe of food, the cooking method and the personal preferences of cooker will also affect the appearances of food. Other than that, the second challenge would be the unstructured ingredient distribution. For certain type of foods, the ingredients are distributed randomly across a plate, so it is hard to recognize the food even though those foods are same type. The third important challenge is the different scales of food image will affect the accuracy of food recognition. For instance, baked potato are looks similar with brown rice without considering their relative scale.

1.2 Project Scope

In project 1, the scope of this project is the proposed recognition system models can only work on the set of foods that are predetermined, this is because, instead of putting effort on recognize several objects, it is easier to success when focus on recognizing only one object in the early development. Moreover, the accuracy in detecting food are highly depending on the freshness of the food, the better of the freshness, the higher the accuracy, and vice versa. The reason is an expiring food will lose its saturation and getting worse on its appearance.

However, since all the datasets in Final Year Project 1 were manipulated by developer and the accuracy only tested in one type of food, so the system will not be able to train too much type of food. Therefore, in Final Year Project 2, the main project scope is to use the technique of crowdsourcing to collect datasets, which mean all the datasets will be provided by wide users. For example, any of the user can take a picture of the food and upload the food image with multiple input of the food information to a website provided by developer, then all the images will be downloaded and undergo a deep learning process.

Moreover, the recognition system will consider the cues on packaging labels if there are any overt cues. For example, the drinks of Coca-Cola and Pepsi are difficult to differentiate by people without packaging labels, therefore the packaging labels are playing an important role to help system recognize the food occasionally. Lastly, our model is still a location-based, it is weighted by locale for improving the recognition based on the principle of locality.

1.3 Project Objectives

The first objective of this proposed system is to automate the data annotation for modeling the non-overt features for food origins recognition using a deep learning method, which is convolutional neural network (CNN) without inferring to location metadata. This is because the location metadata is dynamic and not always will correct.

Secondly, our objective in Final Year Project 1 is to integrate the existing Amazon Web Services (AWS) with added localization using transfer learning in order to improve the accuracy on local cuisine detection. After this method has been implemented, it will reuse the previous results on the following task in order to improve the accuracy.

Thirdly, one of my objectives is to achieve high accuracy of detection with an average of 70% and above.

In addition, our main objective to model spatial and temporal behaviors of local cuisine using reinforcement learning for continuous recognition.

In Final Year Project 2, my objective is to build a website for user to upload food images for data collection purpose. Then, utilize all the images to undergo image pre-processing to build a CNN model. Next, deploy the trained model and implement it in to a web application by using Flask. Therefore, users are able to upload food image in this website (prediction) to obtain the food location. In conclusion, there will be two websites provided, the first one is for data collection, the second one is to perform prediction.

1.4 Impact, significance and contribution

This proposed system presents a food origin classification method. The innovation is to inventing a system that able to recognize the origin of the foods. For example, when I buy a bunch of Nasi Lemak from several stalls, I can find out which Nasi Lemak belongs to which stall by capturing a photo of Nasi Lemak and upload it on the website. After uploading the food image, system will detect the food origin and show the store name to user. Therefore, by having our system, users are able to track the origin of the foods.

Other than that, the second contribution is our project shows the novel technique for continuous recognition that compensate for temporal changes of cuisine appearances. Since any slight changes of the original appearance will require the machine learning model to learn the new pattern.

Lastly, our project proposed a fast-learning algorithm that leveraging on Keras Deep Learning for fine-grain food analytics using non-overt features.

1.5 Background Information

The project title that I have chosen to undertake, which called "Modelling non-overt features for food origin recognition " is to create a system that are able to do food recognition as well as detecting the food origin, which mean users able to know what is the name of the food and where is the location of the food by capturing photo of food.

Case in point, in this modern era, social media is growing rapidly throughout the world. People such as adult or teenagers are started to joining sites such as Facebook, Instagram, Snapchat and Twitter to interact with their family, friends or strangers. Admittedly, the introduction of social media has changed the world in many ways. The one of the issues is people are affected with the mania of sharing their life moments on social media. Apparently, people are often to post their food pictures on social media in order to share with their family, friends and strangers. At the meantime, if people are interested with the food image, they might tend to have a try on it. However, sometimes people might just post about the food image and forgot to state out the location of the food. So that people might miss out the chance on trying new foods. On the other hand, world is form by different country and religion. Different countries have its distinctive cultures such as festival, apparel, food and so on. When people travel to a country, they are totally unfamiliar with all those cultures. In this situation, people need to put effort on learning different culture and accept it. For example, when American travel to Malaysia, they would not know the name of our traditional food, such as Nasi Lemak, Roti Canai and so on.

Let's get to the point, throughout these two scenarios we might realize that our proposed system is able to help those people to solve their problem. At the current market, the existing software just able to help people to recognize the object name instead of the food origin. Therefore, the project of "Modelling non-overt features for food origin recognition" is a brand new and unique idea that helps to trace the food origin.

Chapter 2: Literature Review

2.1 VisualPal

As object recognition, Shagufta Md.Rafique Bagwan and Prof. L.J.Sankpal proposed a mobile phone application which named “VisualPal” [1]. The purpose of this application to be proposed is because the researchers are trying to help visually blind users to recognize all the surrounding objects. On top of that, this proposed application is able to detect the direction of maximum light as well as detects the colors [2].

By doing this proposed application, researchers are implementing a Hybrid algorithm into it. Bagwan & Sankpal(2015) said that the Hybrid algorithm is a combination of Euclidean Distance measures and the Artificial Neural Network(ANN) [3], which will strengthen the object recognition application. “The results are communicated to the blind user using verbal messages” (Bagwan & Sankpal, 2015, p.1).

Artificial Neural Network (ANN) is a framework which consists many different machine learning algorithms that will process complex data inputs [4]. For instance, in image recognition, ANN will learn to distinguish the images that contain dogs by analyzing the sample images that be labelled as “Dog” or “no dog”. After analyzing those sample images, ANN will identify any other dog images based on the analyzed results. In addition, Euclidean distance measures is a mathematical algorithm that help the system to calculates the error once the output value are not similar to the trained objects. Hence, the implementation of hybrid algorithm will reduce the rate of false positive results and lead to a high accuracy of recognition on the proposed application.

However, there are some disadvantages of using the ANN during the development. Firstly, ANN is requiring processors with the same level of processing power, which mean that all of the hardware is dependent to each other. Besides, ANN are always showing an unknown duration and it could not show optimum results on the value.

Figure below shows the procedure of using the proposed system. Figure 2.1.1 is showing the GUI of the proposed application. Figure 2.1.2 is the GUI for registration and training of the

object recognition. Figure 2.1.3 is showing the result of an object that has been recognized, which is a mouse. Lastly, figure 2.1.4 is showing the result of the color detection, which is yellow.

Fig. 2.1.1. GUI of Object Detection Module

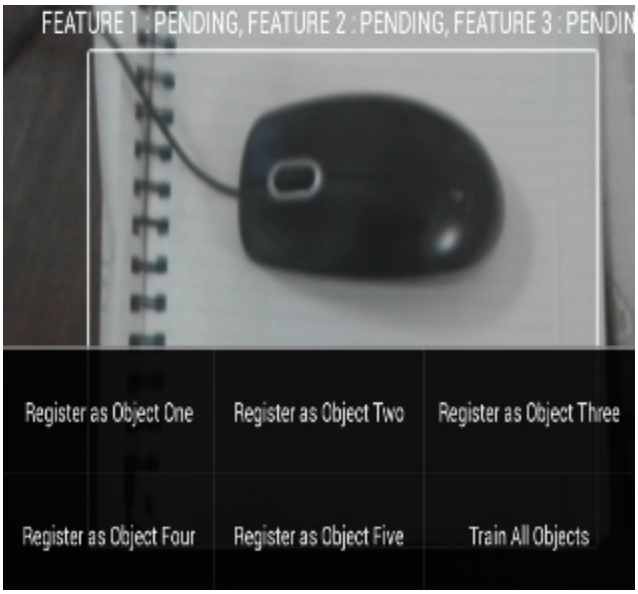


Fig. 2.1.2 Registration and Training of the Objects

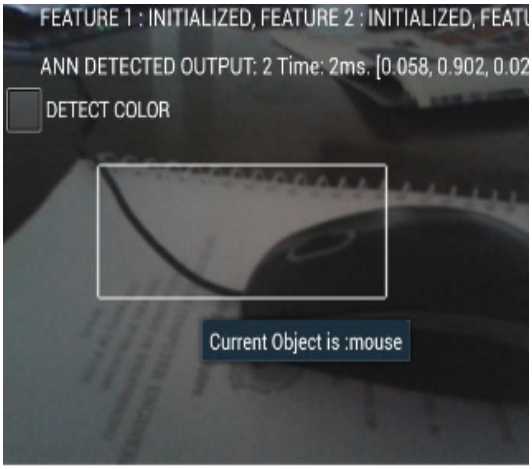
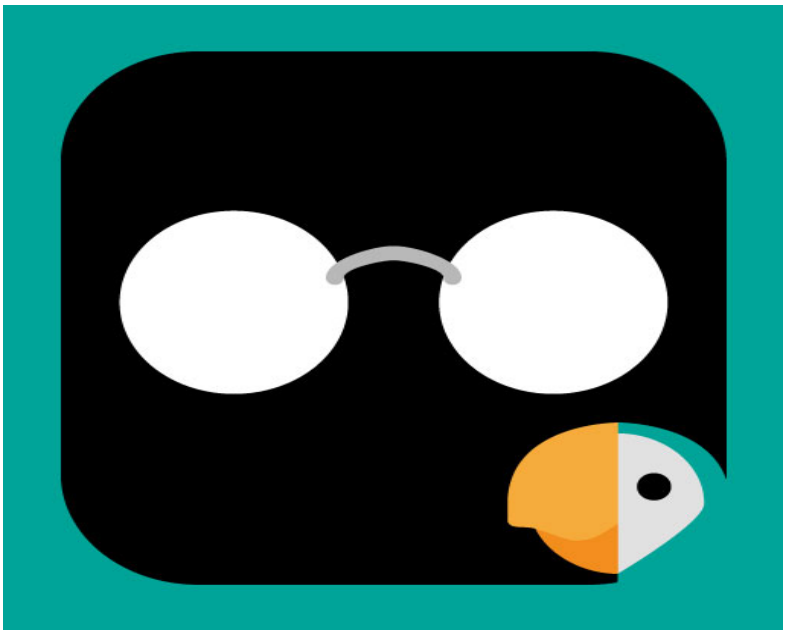


Fig. 2.1.3 Recognized object on a mouse.

Fig. 2.1.4. Color Detection Result

2.2 Aipoly Vision



Aipoly Vision [5] [6] is a free application developed and offered by Aipoly Inc on 3rd January 2016. This application was using Machine Learning method to distinguish the type of the objects and name it when the phone's camera is pointing on the object. The goal of Aipoly Vision is to helping those peoples who are blind, color blind or visually impaired to recognize all the object around them.

The Aipoly Vision application was using the convolutional neural networks(CNN) to identify the items. It is a deep learning method that will be used in computer vision to recognize the objects that may not always have the same appearance, such as plants, food, color, people, animals, or any other common objects. As of now, Aipoly Vision application is able to recognizes over 1400 types of colors, identify over 2000 species of animals and plants, over 1000 type of foods item. Other than that, Aipoly Vision application are able to read the text in multiple languages as well as currency recognition. More importantly, it can maintain a rate of three times a second in objects recognition.

Among the existing systems, Aipoly Vision is using a different approach to running their system. Generally, users need to take a few pictures or upload pictures to the system and wait for the results, while Aipoly Vision are processing and recognizing the image on user's device without any internet connection needed. Since Aipoly Vision is running on user's smartphone, so that it can work on Airplane mode without any barriers. Which mean it is faster in recognition and save user's time on waiting the process time due to the slow internet connection.

However, due to the quick recognition on every single object, Aipoly Vision application is not as accurate as other existing application. This is because Aipoly Vision application is emphasizing on the efficiency of recognizing object, it does not have much time to analyze the object wisely, so the accuracy in recognition is lower than other existing application. In order to solve this problem, Aipoly Vision must optimize the algorithm used to be more effective or add in the latest technology to improve the accuracy. In addition, since Aipoly Vision application is using artificial intelligence, so it will keep learning about the new objects.

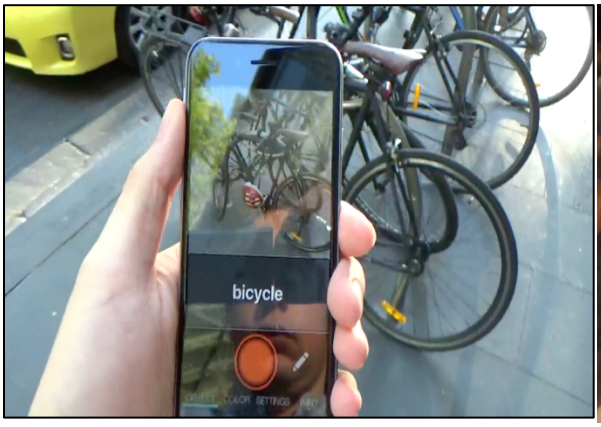


Fig. 2.2.1 Recognize an object as bicycle

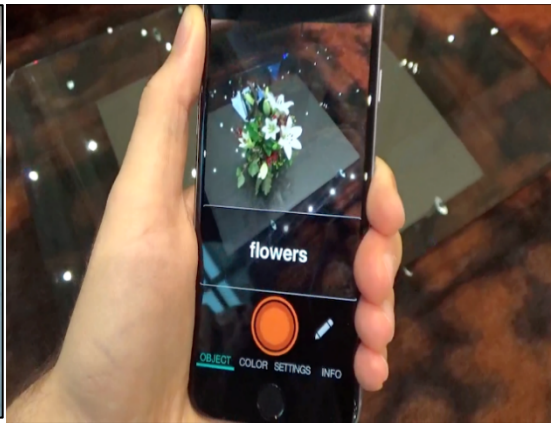
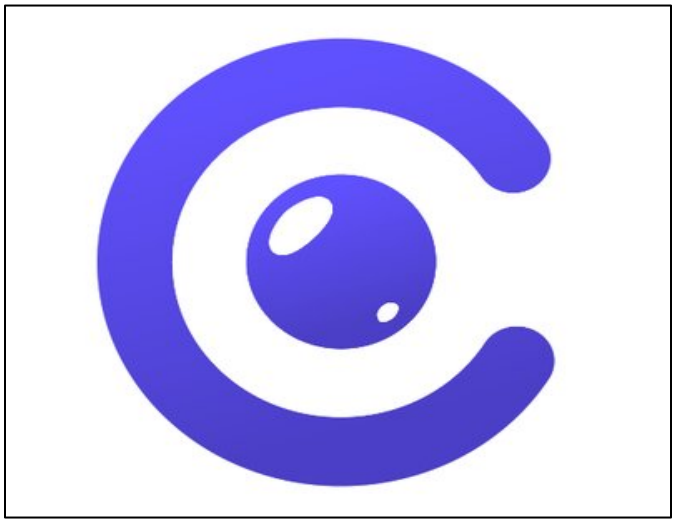


Fig. 2.2.2 Recognize an object as flowers

2.3 CamFind



CamFind [7][8] is a free application that is available to download in the Apple App Store and Google Play. It is developed by Image Searcher, Inc. in year 2013. CamFind is allow users to do visual search and image recognition by using the application. Users can identify any items, such as food, electronic devices, animal or any other common things by taking a picture with their smartphones. After the picture has been processed, it will provide a range of web-based information to users, such as the verbal messages, similar object's images, location information, price comparisons and so on [9]

. To ensure the application is progressively, Image Searcher, Inc. is using deep learning method to improve the quality of result as well as the accuracy of results. Moreover, Image Searcher, Inc. is implementing the visual search engine into this system. Once the images have been processed, it will search for the related information on the World Wide Web directly. In addition, the image will be recognizing based on its content, color, shape and texture, then the search engine will compare it in a database before delivering the results.

Apart from that, the GUI of CamFind is rather unattractive and non-user friendly. Therefore, developers should improve the GUI in order to meet user satisfaction.

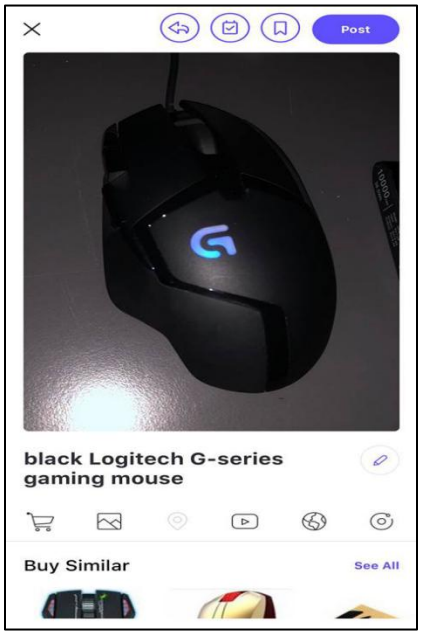


Figure .2.3.1 Recognize an object as Logitech G-series gaming mouse

The reason why CamFind can be the world's first successful mobile visual search engine is because it used to return high accuracy results to users. Other than that, instead of just showing the object's name, it shows a very detail and specific verbal message to user. Based on figure 2.3.1 above, we may realize that after recognize the object, it returns the result of "black Logitech G-series gaming mouse", instead of just a simply word "mouse". On top of that, CamFind ensure the duration of every single recognition are fall within 12 seconds, while the object that has been recognize at past will be detected faster.

2.4 Simultaneous Food Localization and Recognition

As an object recognition system, Marc Bolanos and Petia Radeva proposed a food localization and recognition system at year 2016 [10]. The objectives of this proposed system is to improve several medical condition, such as obesity, diabetes and so on by analyzing the nutrition habits of people. This phenomenon has been proved in [11], people are usually over underestimate the quantity of the food intake up to 33%.

Users need to capture a photo of the food with their smartphones, then the technique that researchers have been implemented, which is visual lifelogging [12] will analyze the life pattern of the users by capturing the pictures from the user point of view automatically. Other than that, researchers also using deep learning method, which are convolutional neural networks (CNNs) to run the system. By using convolutional neural networks, it will able to study the spatial pattern from images no matter how complex it is. Based on the convolutional neural networks, researchers have been proposed some novel approach for food recognition in conventional and egocentric vision pictures [13].

Firstly, Petia Radeva and Marc Bolanos proposed "the first food-related objects localization algorithm, which is specifically trained to distinguish images containing generic food and has the ability to propose several bounding boxes containing generic food in a single image" [14]. Then, they propose an algorithm that will reuse the food-relateImpld knowledge and then apply it to the food localization method [15]. Thirdly, they will "present the first egocentric dataset for food localization and recognition". Lastly, Marc Bolanos & Petia Radeva(2016) said that "we demonstrate that our methodology is useful for both conventional and egocentric pictures" [16].

Figure 2.4.1 Examples of convolutional food images (two on the left) and egocentric food images (two on the right)

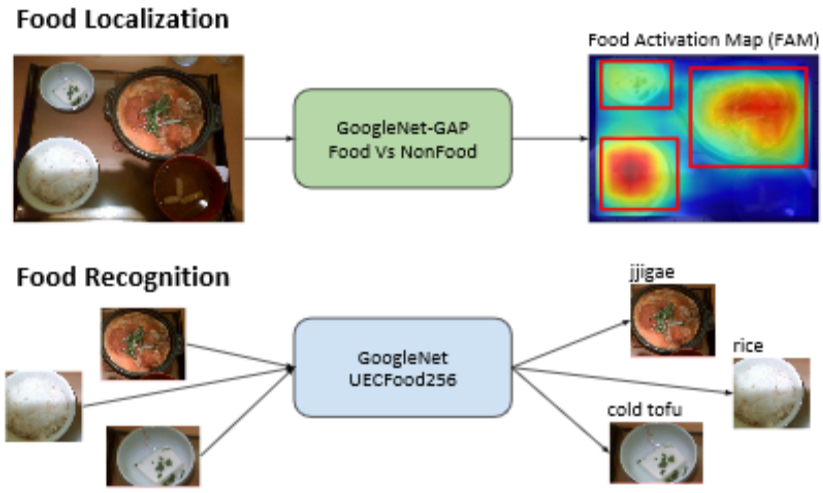


Figure 2.4.2 General scheme of the food localization and recognition proposal.

Conclusion

In conclusion, every system has its own pros and cons. After conducting the studies and research about my topic, I found that CamFind application, which developed by Image Searcher, Inc. is sophisticated and wholesome. Among the existing or proposed system, CamFind has the best performance in terms of object recognition accuracy. It ensures every single recognition has a very high accuracy. Otherwise, CamFind will return a very detail verbal message to users, and also shows the extra information which related to the images. Otherwise, the deep learning method will make sure CamFind is keep improving itself as the previous results will be learned and reuse it on the following task.

CHAPTER 3: System Design

3.1 Technologies Involved

Convolutional Neural Network (CNN)

The primary benefit of CNN for many such tasks is that the whole system is trained end to end, from raw pixels to ultimate categories, thereby alleviating the requirement to manually design a suitable feature extractor. Convolutional neural network can improve the accuracy of classification of all tasks. In brief, CNN is a deep learning method which reinforce itself continuously. It will keep learning new information to replace the outdate knowledge in order to maintain the high accuracy in recognition.

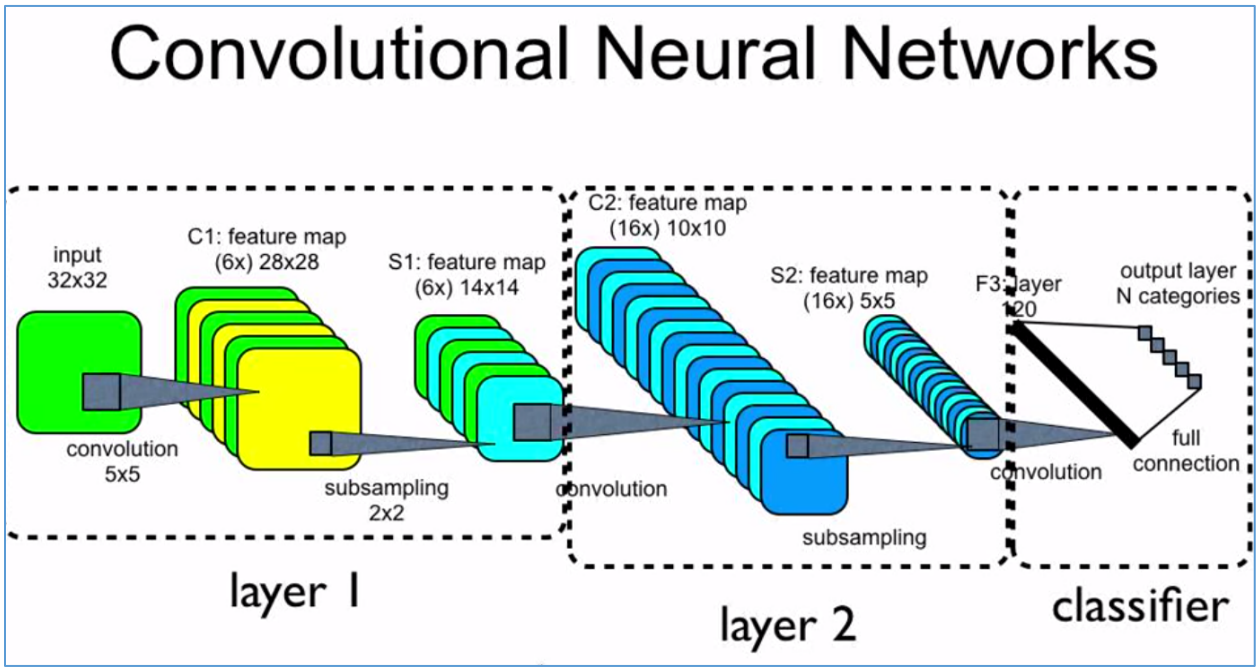


Figure 3.1.1 Diagram of Convolutional Neural Networks

HTML, PHP and MySQL

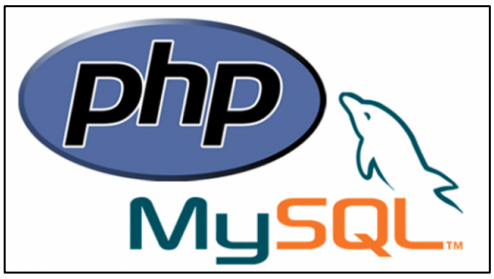


Figure 3.1.2 PHP and MySQL

In my Final Year Project 2, HTML language was applied to design a webpage in order to provide an interface for user to upload the food images. Next, I used PHP, which is a server-side language to collect the user's input data of the food information such as food image, location name as well as the food name. After that, I used MySQL to download the images uploaded to my local computer to proceed to the step of image processing.

Google Places API – Place Search

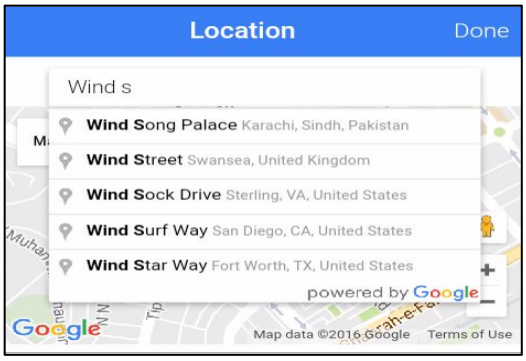


Figure 3.1.3 Google Places API- Place Search

In current Final Year Project, I integrate the google places API into my webpage. Since Places API allows me to search for a place information by using a number of categories, such as geographic location or a prominent point of interest. By using this Google API feature, users are able to input location name, location address with a string format, then the place search will pop up the related result for user to select. After user select the particular location, my system can get the location name, longitude and latitude automatically.

Hostinger

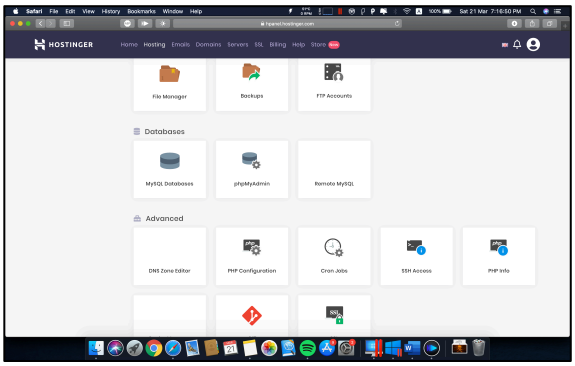


Figure 3.1.4 Hostinger's interface

It is a web hosting service provider that allows me to publish my web application on the internet, so users are able to reach my webpage by searching the domain name on any web browser. On top of that, all the form data that input by user will be stored in the Hostinger's database. Meanwhile, all the food images will also store in the file manager of Hostinger. Therefore, I can download all the uploaded images to my local computer.

Anaconda – Spyder

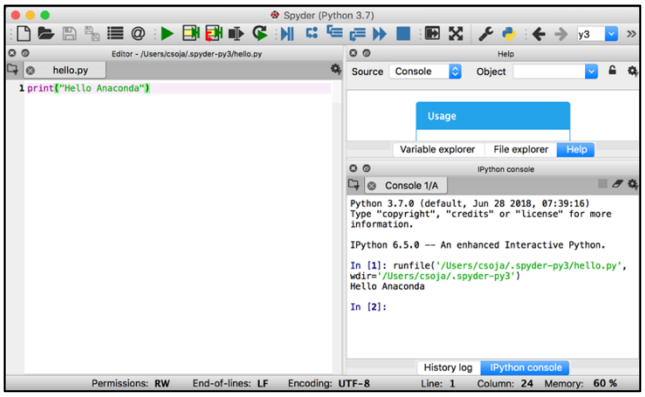


Figure 3.1.5 Anaconda Spyder's Interface

Spyder is an integrated development environment (IDE) for scientific programming in the Python language. It is also a pre-installed software that is included on the Anaconda Navigator. In my Final Year Project 2, it allows me to build a Convolutional Neural Network to train a model with my datasets.

Flask

Figure 3.1.6 Flask

Flask is a featured rich popular web framework that written in python, which mean it's providing a third-party Python library that allow user to build a web application, such as webpages, commercial website or a blog.

3.2 Block Diagram

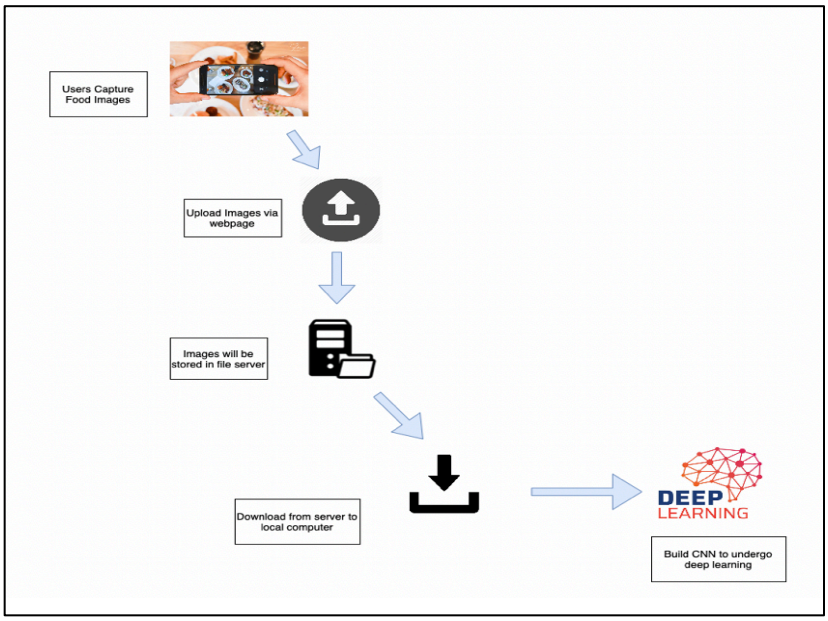


Figure 3.2.1 Block Diagram of the training flow

Figure 3.2.1 is a block diagram that display the overall procedure of food recognition. In the first step, users will capture the image of their foods, then upload those images with variety of inputs, such as location name, location's longitude, location's latitude and food name. After users upload their images successfully, those images will be stored in server's file manager. Meanwhile, those inputs will be inserted into database with a unique name.

Next, developer has to download all the images from the server to local computer. Then, developer will use Anaconda – Spyder to run python script to build a Convolutional Neural Network to train all those images.

3.3 Module 1 - Data Collection

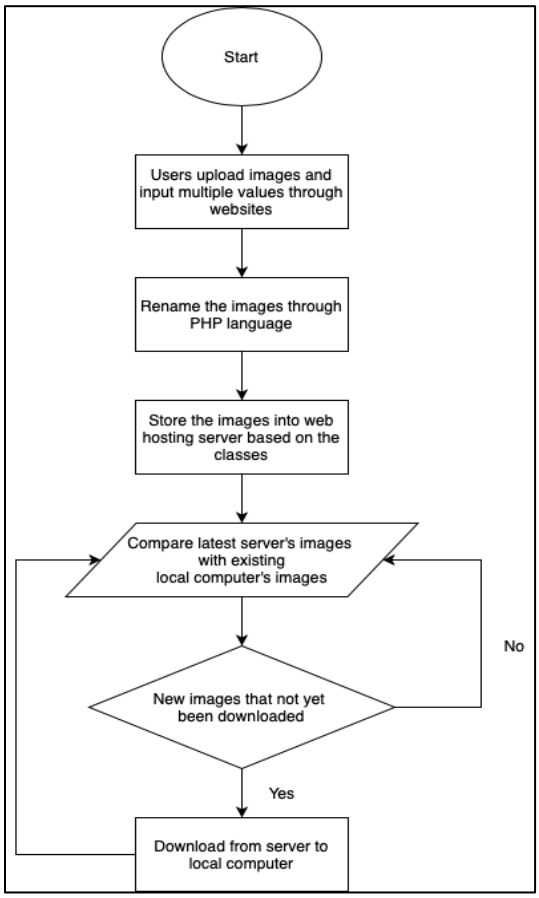


Figure 3.3.1 Flow chart of data collection

As opposed to Final Year Project 1, which prepared all the datasets by my own, Final Year Project 2 implement the concept of crowdsourcing to collect all the datasets from outsiders. In order to collect the datasets, developer provide a hosting website for users to upload images and insert some required inputs.

After a certain period, developer need to download all the uploaded images from server by using a webpage, which involved PHP language that perform function to download all the images. Before download the images, developer had created an algorithm to detect which images already exist in the local computer, then eliminate all the existing image from the list. After confirming the non-exists images, then developer will click “Download” button to download all the remaining images into local computer.

3.4 Module 2 - Data Processing

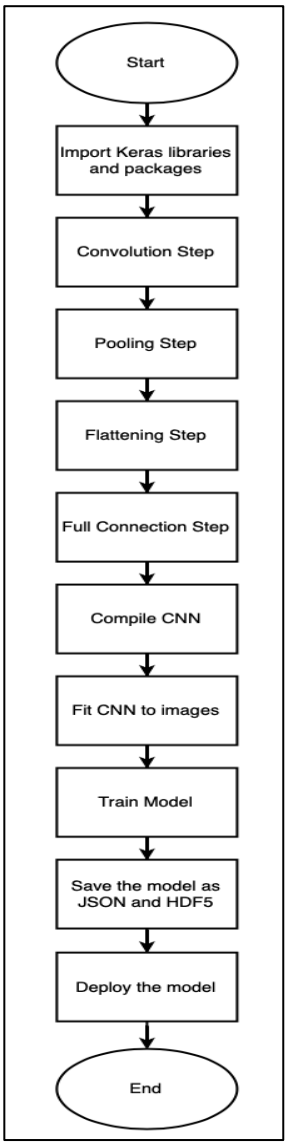


Figure 3.4.1 Flow chart of data processing

After download all the images from server and stored the images into specific folder, the following step is to build a Convolutional Neural Network (CNN) and undergo tuning process to input all the parameters to train a model. Next, save the model as JSON format as well as HDF5 format. Lastly, deploy the saved model and implement it into a web application by using Flask. Therefore, users are able to upload food images through the website, then a predicted result of a location will be displayed on the interface.

Chapter 4: Implementation

4.1 Data collection

In Final Year Project 2, all the images needed are still remain as food images. However, the main difference compare with Final Year Project 1 is all of the images are provided by users instead of preparing by developer himself. Therefore, developer had built a website that allow users to upload food image and also input the location name and food name.

In order to build this website, it involves multiple languages and tools which cover the front-end and the back-end. First of all, HTML and Bootstrap were being used to design a webpage with multiple input fields. After that, implement Google Places API for user to input the location. For example, when user write "KFC Kampar", it will display the related result which contain location name and location address as a dropdown for user to pick the correct one. Then, JavaScript will help developer to retrieve the location name, longitude and latitude that will be stored into database later.

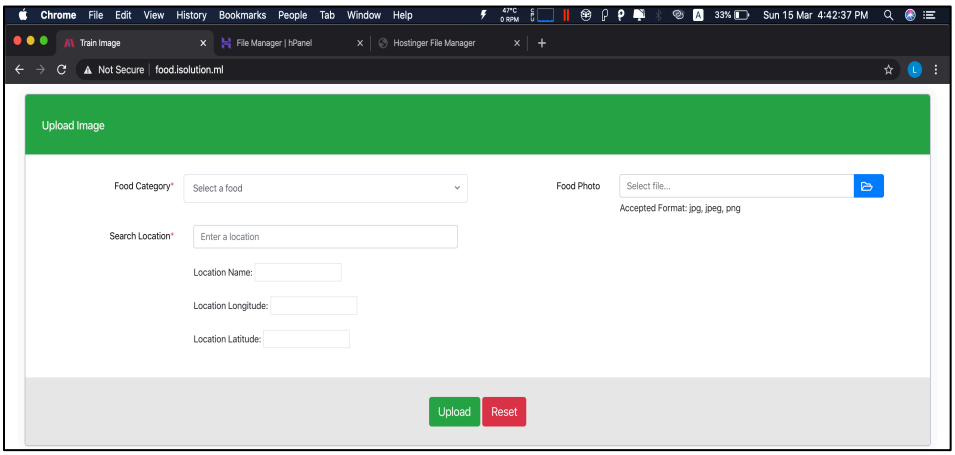


Figure 4.1.1 Interface for user to upload images

After user input all the required fields and click the submit button, the jQuery AJAX will automatically send all the input values to a controller, which contains PHP language and MySQL. In that controller, PHP language act as a role that handle all of the input values and the uploaded image. On top of that, it will also define the image name in a standard way. For instance, if a user uploads a food image with the food name of "Fried Chicken" and location name of "KFC Kampar",

then the back-end logic will check whether this input values is existing in the database. If YES and the existing record is 1, then the image will be renamed as "KFC_Kampar~Fried_Chicken-2". Else if there is no existing record in the database, then the image will be renamed as "KFC_Kampar~Fried_Chicken-1".

Guideline of the special character used in image name:

1) "_" **Underscore** is the special character that concatenate the name of the value that within same input field, this is because space is not allowed for the image name.
2) "~" **Tilde** is the special character that concatenate two different input fields, which are the location name and the food name.
3) "-" **Hyphens** is the special character that concatenate the input value along with the counter of file name, which represent number of file existing.

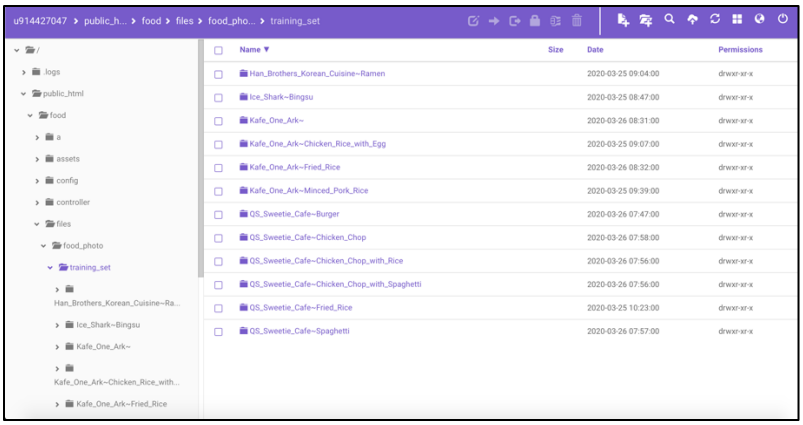


Figure 4.1.2 Image stored in Hostinger's server

After go through the standardization process, the input values and images will be inserted into web hosting database and file manager (Figure 4.1.2). When developer want to download all the images from server, just browse the localhost webpage in Figure 4.1.3. First of all, developer need to click "Sync Before Click Download" in order to retrieve an array of image name that stored in server's database. Then, the download button will be enabled at this moment, so developer can click the "Download" button to compare the array of server's images with the array of local computer's images, after remove all of the matched image name, the remaining images name will

be stored in a new array then perform file_put_contents and file_get_contents method (PHP function) to download the images from server.

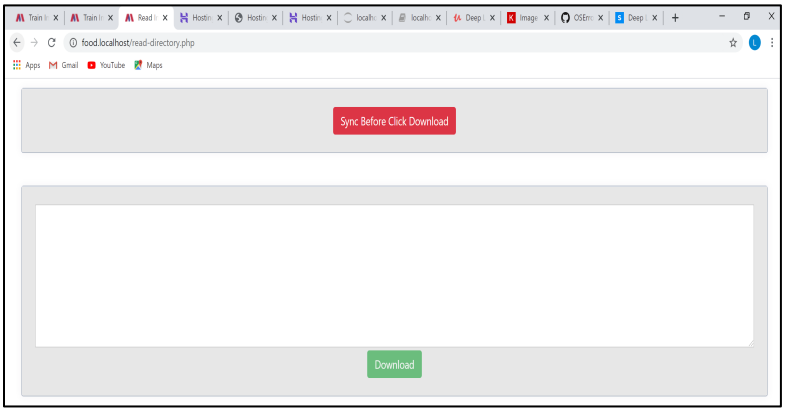


Figure 4.1.3 Interface to download all the images from server

4.2 Data Processing

4.2.1 Build CNN

When the datasets has been prepared, the following step will be train all these images. However, before the training begin, developer has to install Anaconda Spyder, which is an integrated development environment for writing python language. Meanwhile, developer need to create a python file and put it at a correct working directory.

```
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 keras.layers import Dropout
```

Figure 4.2.1.1 Keras packages

In order to build a CNN, the mandatory step is to import Keras libraries and packages (Figure 4.2.1) for CNN. The first thing is Sequential, which will be used to initialise neural network as sequence of layer. The second package is Convolution2D, it is the package that use to add convolutional layer, since the datasets are in two dimensions instead of three dimensions. Other than that, MaxPooling2D is required for adding pooling layers. Next, flatten will be imported to convert all pooled feature maps that created through convolution and MaxPooling into this large feature vector that is then becoming the input of our fully connected layers. The last package is Dense, which will add fully connected layer in this neural network.

```
#Initialising the CNN
classifier = Sequential()
```

Figure 4.2.1.2 Initialize CNN

After import Keras libraries, the first step is to initialising the CNN by creating an object of the sequential class and named it as classifier (Figure 4.2.1.2).

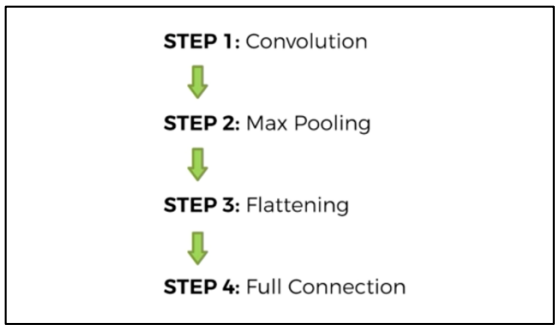


Figure 4.2.1.3 Procedure of building a CNN

Next, figure 4.2.1.3 clearly indicate the procedure of building a CNN, which consists of 4 steps, Convolution, Max Pooling, Flattening and Full Connection.

Step 1: Convolution

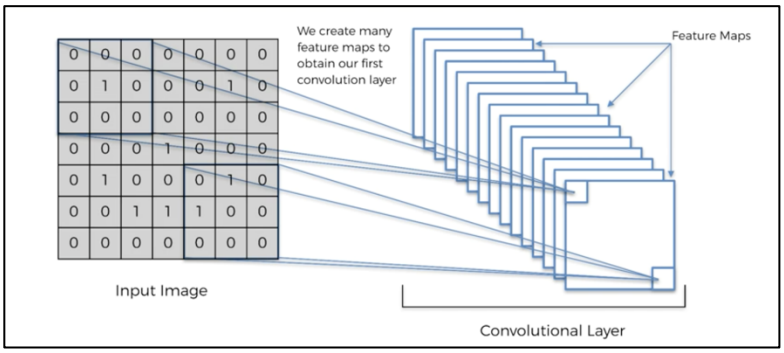


Figure 4.2.1.4 Converting input image into a table of pixel values (1 and 0)

```
# Step 1 - Convolution
classifier.add(Convolution2D(32, (3, 3), input_shape = (64, 64,  3 ), activation = 'relu'))
```

Figure 4.2.1.5 Convolution Step

The main purpose of this step is to adding a convolutional layer by converting the images into a table which in pixel values format (1 and 0). It will apply multiple feature detectors on the input image. Firstly, it will add the parameters of convolutional layers, which is Convolution2D that import in the beginning. After that, the value of 32 that came after the Convolution2D is the number of filters that represent the number of feature detector that will be applied on the input image to create the same number of feature maps. Then, the (3, 3) means it will create a feature

detector that have dimension of 3 by 3. Moreover, the input_shape is the shape of the input image, in this project the input shape of the image will be 64 x 64 pixels. In addition, since the datasets is colour images, so all of the images will be converted into 3D arrays during the image pre-processing part. Therefore, the value of 3 in the input shape is representing 3D arrays that composed of three channels, each channel corresponding to one color, RGB, and each channel correspond to one of the arrays that contains the pixels of our images.

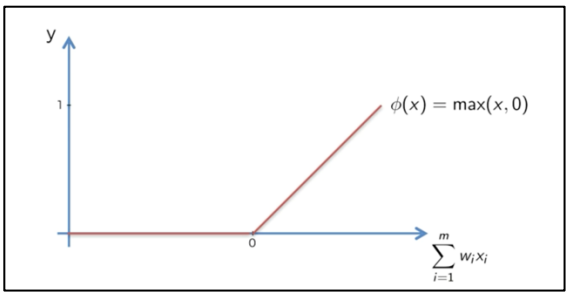


Figure 4.2.1.6 Non-Linear Graph

Lastly, the activation function is a rectifier function (relu) that ensure don't have any negative pixel values in the feature maps in order to get non-linearity (Figure 4.2.1.6) in the CNN.

Step 2: Pooling

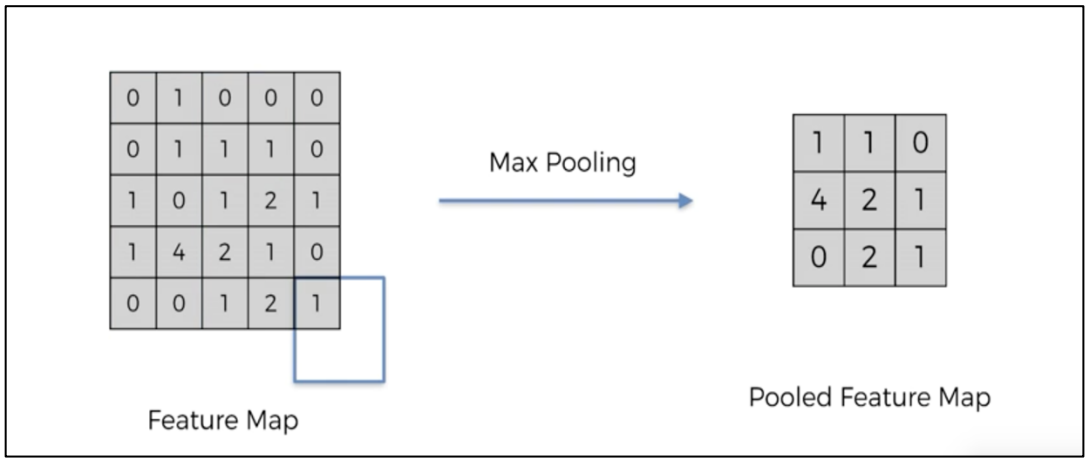


Figure 4.2.1.7 Reducing the size of feature map

Step 2 is to reduce the size of feature map without losing its performance. For instance, the original feature map in Figure 4.2.7 is 5 by 5, but after apply max pooling will get a reduced size of 3 by 3. The primary aim of max pooling is to reduce the nodes in the future fully connected layer in order to reduce the time execution with a great performance which lead to less compute-intensive.

```
# Step 2 - Max Pooling
classifier.add(MaxPooling2D(pool_size = (2, 2)))
```

Figure 4.2.1.8 Pooling Step

Figure 4.2.8 is the code to complete the step of max pooling. The size parameter is pool_size, 2 by 2 is the value that will retain the high numbers in the feature maps that the text and specific features of the input image. For this case, the feature map will be divided by two. Hence, the size of feature map will be reduced without cut down the performance.

Step 3: Flattening

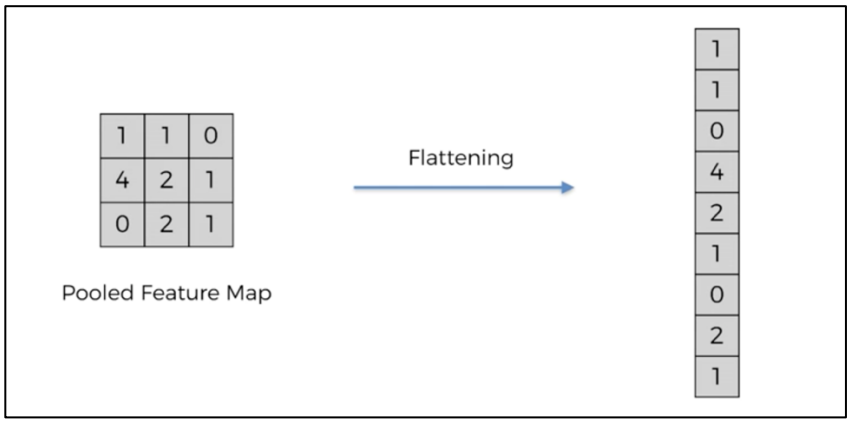


Figure 4.2.1.9 Flatten the pooled feature map

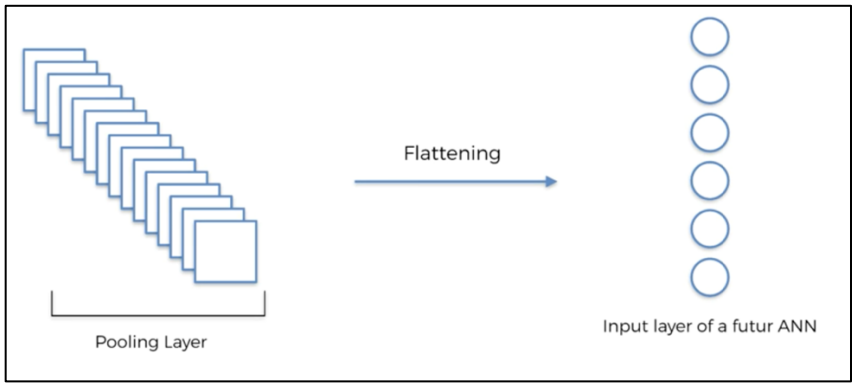


Figure 4.2.1.10 Flatten the pooled feature map

The function of flattening is to gather all the pooled feature maps and assign all of them into a single vector. Then, after apply the flattening step will get huge number of single vectors that contains all different cells of all the different feature maps (Figure 4.2.1.10).

```
# Step 3 - Flattening
classifier.add(Flatten())
```

Figure 4.2.1.11 Flattening Step

In figure 4.2.11, all of the feature maps will be flatten in the pooling layer into huge number of single vectors.

Step 4: Full Connection

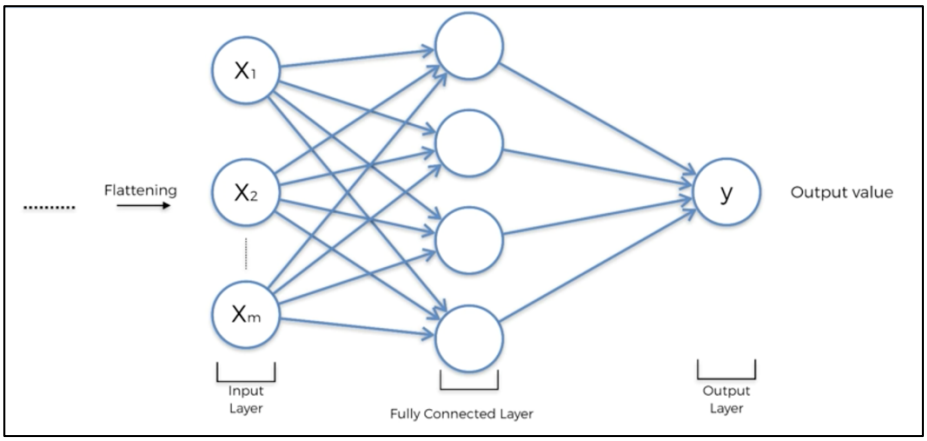


Figure 4.2.1.12 Composing a fully connected layer

In step 4 is to compose some fully connected layer. In other words, it is to transform the image into one-dimensional vector contain the information of the spatial structure or some pixel patterns in the image.

```
# Step 4 - Full Connection
classifier.add(Dense(activation="relu", units=128))
classifier.add(Dense(activation="softmax", units=11))
```

Figure 4.2.1.13 Full Connection Step

This code is to create hidden layer, which also known as fully connected layer. First of all, dense function is added in order to add fully connected layer. The output_dim refers to the amount of nodes which contained in the hidden layer, while the units of 128 represent the amount of hidden nodes in the hidden layer. Otherwise, the activation function is using rectifier activation function which is “relu”. Then, copy the code and paste it below and change the value of output_dim as well as the activation function. The activation function in the second code is using softmax due to the multiple class is involved. Then, the units of 11 represent the number of current classes.

Step 5: Compile CNN

```
# Compiling the CNN
classifier.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics = ['accuracy'])
```

Figure 4.2.1.14 CNN compilation step

In this code, the optimizer is the parameter that will be used to choose the stochastic gradient descent algorithm, which is Adam algorithm. On top of that, the loss function is categorical_crossentropy, since we have multiple classes to be trained. Lastly, the metrics parameter is choosing the accuracy metric.

4.2.2 Train model

```
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)

training_set = train_datagen.flow_from_directory('food_photo/training_set',
                                                 target_size=(64, 64),
                                                 batch_size=20,
                                                 class_mode='categorical')

test_set = test_datagen.flow_from_directory('food_photo/test_set',
                                            target_size=(64, 64),
                                            batch_size=20,
                                            class_mode='categorical')

classifier.fit_generator(training_set,
                         steps_per_epoch=109, # number of train images
                         epochs=15,
                         validation_data=test_set,
                         validation_steps=20) # number of test images
```

Figure 4.2.2.1 Train model

After compile the CNN, the next step is to fit the CNN to our datasets. Moreover, the one of the main aims in this step is to use image augmentation process that basically consists of pre-processing the test image to prevent overfitting. First and foremost, import the Image Data Generator from Keras preprocessing.image. After that, keep all the default value that suggested by Keras at train_datagen and test_datagen.

Next, declare variable of training_set and test_set in order to extract the images. The first parameter is the directory of the datasets, while the target size is the size of the images that is expected in the CNN model, which is 64 by 64 dimensions. In addition, assign a value of 20 as the size of the batches in which some random sample of the images will be included and that contains the number of images that will go through the CNN, while the class mode is specified as categorical due to the multiple classes involved.

Lastly, apply the model.fit_generator to fit the CNN to the training set and test the performance of the CNN model on the test set. Moreover, the steps_per_epoch is 109, which refers to the number of training set and there will be 15 epochs. On the other hand, validation data correspond to the test set on which we want to evaluate the performance of our CNN and the validation_step is the number of test set images, which have 20 of them. Then, the function of fit generator will fit our CNN to our training set and test its performance on the test set at the same time, and this fit generator method is applied onto the CNN model, which is "classifier".

4.2.3 Save model

```
# Save the model
model_json = classifier.to_json()
with open("model.json", "w") as json_file:
    json_file.write(model_json)
classifier.save_weights("model.h5")
```

Figure 4.2.3.1 Save the training model

During this phase, the mandatory step is to serialize the training model to JSON with a file name of "model.json". Meanwhile, serialize the weights of the CNN to HDF5 file as well, which called "model.h5". These files will be used later for prediction purpose.

4.2.4 Deploy the model

After complete the training step and export it as json and h5 file, the next part is to deploy the deep learning model with the by using Flask, which is a web framework that use for build a web application and implement the deep learning model into the web application.

```
import numpy as np
import os
import tensorflow.keras.models
from tensorflow.keras.models import model_from_json
import tensorflow as tf

def init():
    json_file = open('model\model.json','r')
    loaded_model_json = json_file.read()
    json_file.close()
    loaded_model = model_from_json(loaded_model_json)
    #load woeights into new model
    loaded_model.load_weights("model\model.h5")
    print("Loaded Model from disk")

    #compile and evaluate loaded model
    loaded_model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])

    return loaded_model
```

Figure 4.2.4.1 Load the json and h5 file

Primarily, before starting to build the web application, the first thing to do is to create a python file called “load.py” and define a function called “init” and load the training model file, which are “model.json” and “model.h5” (Figure 4.2.4.1). Then, create another python file which named “app.py” to build a flask app.

```
from __future__ import division, print_function
# coding=utf-8
import sys
import os
import glob
import re
import numpy as np
import json
import pickle

# Keras
from keras.applications.imagenet_utils import preprocess_input
from tensorflow.keras.models import load_model
from keras.preprocessing import image

# Flask utils
from flask import Flask, redirect, url_for, request, render_template
from werkzeug.utils import secure_filename
from gevent.pywsgi import WSGIServer

sys.path.append(os.path.abspath("./model"))
from load import *
# Define a flask app
app = Flask(__name__)

global model
#initialize these variables
model = init()
# Load your trained model
model._make_predict_function()
```

Figure 4.2.4.2 Import all the required packages

In the file of "app.py", import all the required packages for the future use (Figure 4.2.4.2). Moreover, load the trained model by calling the function of init() from the file of "load.py" (Figure 4.2.4.1). Then, initialize the Flask app and also initialize a declared variable called "model", and load the trained model into the variable.

```
def model_predict(img_path, model):


    img = image.load_img(img_path, target_size=(64, 64))

    # Preprocessing the image
    x = image.img_to_array(img)
    # x = np.true_divide(x, 255)
    x = np.expand_dims(x, axis=0)

    x = preprocess_input(x, mode='caffe')

    preds = model.predict(x)
    return preds
```

Figure 4.2.4.3 Image pre-processing function

Define a function called "model_predict", which include two parameters, image path and the trained model (Figure 4.2.4.3). Next, create a new variable called "img" and import two arguments into the load_img function, which is image path and a size of 64 by 64 dimensions of images. Then, add a new dimension due to colour image has 3 dimensions, img_to_array will create the third dimension. After that, add another dimension by using expand_dims which taken from numpy, because this dimension is corresponding to the batch. In other words, the function of neural networks like the predict function cannot accept a single input by itself, it only accepts the inputs in a batch, even if the batch contains only one input. Therefore, import the test image and then specify axis equals zero in the expand_dims function because axis equal zero means that the index of this new dimension we are adding will have the first index, which is index zero.

```python
@app.route('/', methods=['GET'])
def index():
    # Main page
    return render_template('index.html')


@app.route('/predict', methods=['GET', 'POST'])
def upload():
    if request.method == 'POST':
        # Get the file from post request
        f = request.files['file']

        # Save the file to ./uploads
        base = 'C:\wamp64\www\food\files'
        basepath = os.path.dirname(base)
        file_path = os.path.join(
            basepath, '\wamp64\www\\food\\files\\food_photo\prediction', secure_filename(f.filename))
        f.save(file_path)

        # Make prediction
        preds = model_predict(file_path, model)


        # Process your result for human
        # pred_class = preds.argmax(axis=-1)            # Simple argmax
        #pred_class = preds.argmax(axis=1)
        print('preds: ',preds)
        #result = str(preds) # Convert to string

        if preds[0][0] == 1:
            prediction = 'Ramen from Han Brothers Korean Cuisine'
        elif preds[0][1] == 1:
            prediction = 'Bingsu from Ice Shark'
        elif preds[0][2] == 1:
            prediction = 'Chicken Rice with Egg from Kafe One Ark'
        elif preds[0][3] == 1:
            prediction = 'Fried Rice from Kafe One Ark'
        elif preds[0][4] == 1:
            prediction = 'Minced Pork Rice from Kafe One Ark'
        elif preds[0][5] == 1:
            prediction = 'Burger from QS Sweetie Cafe'
        elif preds[0][6] == 1:
            prediction = 'Chicken Chop from QS Sweetie Cafe'
        elif preds[0][7] == 1:
            prediction = 'Chicken Chop Spaghetti from QS Sweetie Cafe'
        elif preds[0][8] == 1:
            prediction = 'Chicken Chop Rice from QS Sweetie Cafe'
        elif preds[0][9] == 1:
            prediction = 'Fried Rice from QS Sweetie Cafe'
        elif preds[0][10] == 1:
            prediction = 'Spaghetti from Buddies'
        print('prediction: ', prediction)
        return prediction
    return None
```

Figure 4.2.4.4 Receive uploaded image from rendered template and make prediction

Moreover, render a written HTML template, which called index.html. Then, define an app route called "/predict" with a method of GET and POST. When user upload an image and the submit button is onclick, then the /predict will be called, and then the uploaded image will be stored in a preset path's folder which called "prediction". Once the image has been stored successfully, the function of model_predict will be triggered, then it will load the image path into the model and return a result. Lastly, the result will be displayed at the HTML template.

```python
if __name__ == "__main__":
    #decide what port to run the app in
    port = int(os.environ.get('PORT', 5000))
    #run the app locally on the givn port
    app.run(host='0.0.0.0', port=port)
    #optional if we want to run in debugging mode
    app.run(debug=True)
```

Figure 4.2.4.5 Assign port number to flask app

Choose port 5000 to run the Flask app, then use app.run function to run the app locally.

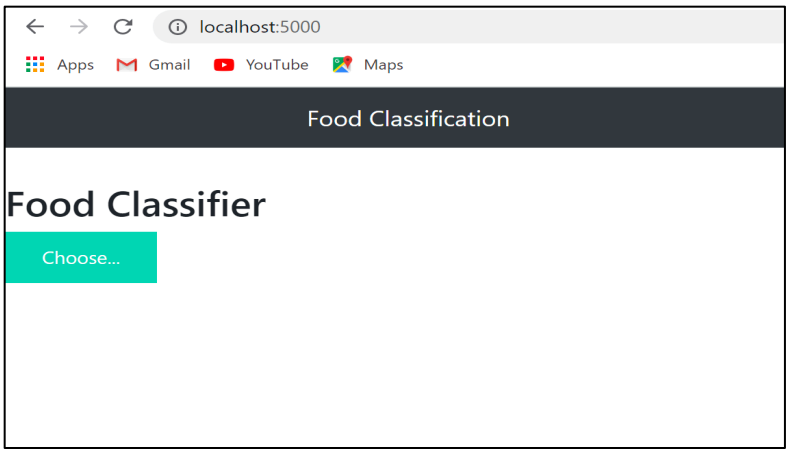


Figure 4.2.4.6 HTML template for user upload image (Phase 1)

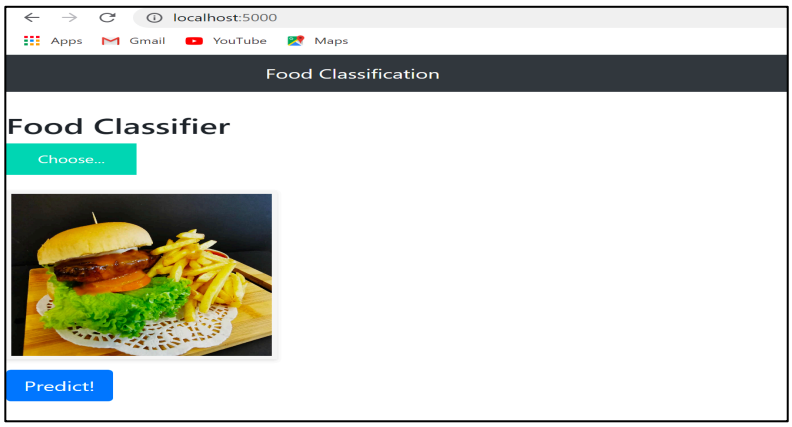


Figure 4.2.4.7 HTML template for user upload image (Phase 2)

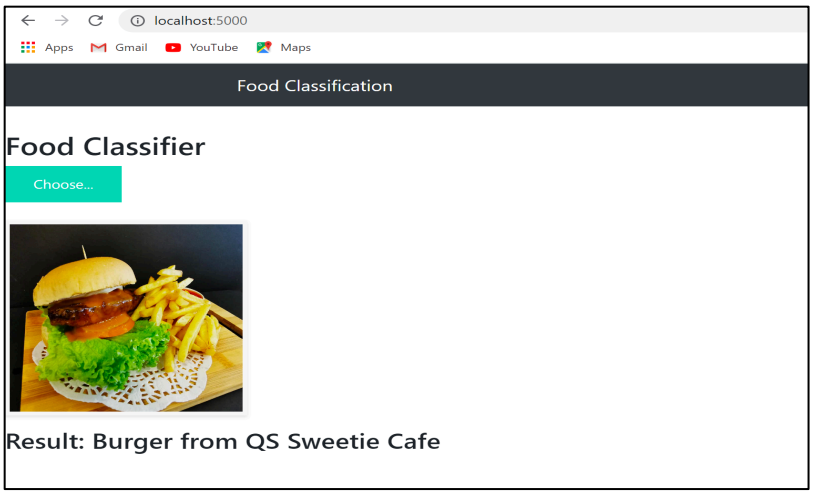


Figure 4.2.4.8 HTML template for user upload image (Phase 3)

Figure 4.2.4.6, Figure 4.2.4.7 and Figure 4.2.4.8 are the web application that allow user to upload food image in order to do prediction. In phase 1, users can click the “choose” button to select a food image. In phase 2, after user select food image, the selected image will be displayed on the interface and a “predict” button will appear automatically. Then, once user click the “predict” button, the image will be stored in a pre-set file path and undergo the process that explained in figure 4.2.4.4.

CHAPTER 5: Testing and Evaluating Results

5.1 Accuracy for training set and test set

During the image pre-processing, the configuration of epoch is 10, which mean 10 times of the training process will be completed consequently.

```
Epoch 1/15
86/86 [==============================] - 82s 958ms/step - loss: 1.4105 - accuracy: 0.5162 - val_loss: 0.8195 - val_accuracy: 0.7113
Epoch 2/15
86/86 [==============================] - 79s 913ms/step - loss: 0.4589 - accuracy: 0.8386 - val_loss: 0.3306 - val_accuracy: 0.7910
Epoch 3/15
86/86 [==============================] - 83s 966ms/step - loss: 0.2553 - accuracy: 0.9180 - val_loss: 0.6733 - val_accuracy: 0.7077
Epoch 4/15
86/86 [==============================] - 80s 928ms/step - loss: 0.1388 - accuracy: 0.9522 - val_loss: 1.1970 - val_accuracy: 0.7127
Epoch 5/15
86/86 [==============================] - 81s 942ms/step - loss: 0.1061 - accuracy: 0.9693 - val_loss: 1.1801 - val_accuracy: 0.6655
Epoch 6/15
86/86 [==============================] - 78s 910ms/step - loss: 0.0708 - accuracy: 0.9771 - val_loss: 2.2082 - val_accuracy: 0.7090
Epoch 7/15
86/86 [==============================] - 80s 936ms/step - loss: 0.0848 - accuracy: 0.9755 - val_loss: 1.5785 - val_accuracy: 0.6620
Epoch 8/15
86/86 [==============================] - 80s 932ms/step - loss: 0.0814 - accuracy: 0.9731 - val_loss: 0.3247 - val_accuracy: 0.6716
Epoch 9/15
86/86 [==============================] - 80s 934ms/step - loss: 0.0624 - accuracy: 0.9827 - val_loss: 1.5430 - val_accuracy: 0.7465
Epoch 10/15
86/86 [==============================] - 81s 947ms/step - loss: 0.0428 - accuracy: 0.9873 - val_loss: 0.0013 - val_accuracy: 0.6679
Epoch 11/15
86/86 [==============================] - 79s 919ms/step - loss: 0.0220 - accuracy: 0.9926 - val_loss: 0.9214 - val_accuracy: 0.6690
Epoch 12/15
86/86 [==============================] - 82s 957ms/step - loss: 0.0412 - accuracy: 0.9888 - val_loss: 1.1314e-04 - val_accuracy: 0.7500
Epoch 13/15
86/86 [==============================] - 80s 931ms/step - loss: 0.0358 - accuracy: 0.9887 - val_loss: 1.6326 - val_accuracy: 0.7923
Epoch 14/15
86/86 [==============================] - 82s 948ms/step - loss: 0.0298 - accuracy: 0.9886 - val_loss: 0.0079 - val_accuracy: 0.7948
Epoch 15/15
86/86 [==============================] - 82s 950ms/step - loss: 0.0273 - accuracy: 0.9900 - val_loss: 1.2377 - val_accuracy: 0.7042
```

Figure 5.1.1 Accuracy of training set and test set with 15 epochs

Sequence of epochs	Training Accuracy	Testing Accuracy
1	0.5162	0.7113
2	0.8386	0.7910
3	0.9180	0.7077
4	0.9522	0.7127
5	0.9693	0.6655
6	0.9771	0.7090
7	0.9755	0.6620
8	0.9731	0.6716
9	0.9827	0.7465
10	0.9873	0.6679
11	0.9926	0.6690
12	0.9888	0.7500
13	0.9887	0.7923
14	0.9886	0.7948
15	0.9900	0.7042

Table 5.1.1 Accuracy of training set and test set with 15 epochs

Average training accuracy = 0.9359 = 93.59%

Average testing accuracy = 0.7170 = 71.70%

Based on the result shown in Table 5.1.1, the accuracy for training was improving gradually, this is because the deep learning method will reinforce itself, so the accuracy will be obtained increasingly.

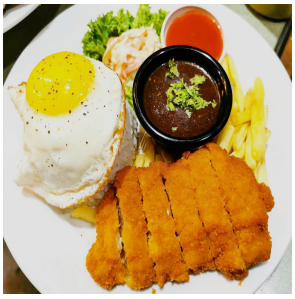
5.2 Prediction with test images

In order experiment on the actual performance of the Keras deep learning model, 12 random captured food images are selected, which consist of 3 Spaghetti, 3 Chicken Chop Spaghetti, 3 Burger and 3 Chicken Chop Rice to undergo prediction. Each categories of food consist of different ingredient and different appearance. Moreover, all of the food images will be uploaded through a webpage (in Figure 4.2.4.6), then the images will be sent to the training model to undergo prediction. Next, the webpage will print out the food name and location name. The result of food images will be collected in table 5.2.1 as shown below.

Test Images	Results Shown	True / False
Test1.jpg 	Spaghetti from Buddies	✓
Test2.jpg 	Spaghetti from Buddies	✓
Test3.jpg 	Spaghetti from Buddies	✓

Test4.jpg 	Chicken Chop Spaghetti from QS Sweetie Cafe	✓
Test5.jpg 	Spaghetti from Buddies	X
Test6.jpg 	Chicken Chop Spaghetti from QS Sweetie Cafe	✓

Test7.jpg 	Burger from QS Sweetie Cafe	✓
Test8.jpg 	Spaghetti from Buddies	X
Test9.jpg 	Burger from QS Sweetie Cafe	✓

Test10.jpg 	Chicken Chop Rice from QS Sweetie Cafe	✓
Test11.jpg 	Chicken Chop Spaghetti from QS Sweetie Cafe	X
Test12.jpg 	Chicken Chop Rice from QS Sweetie Cafe	✓

Test13.jpg 	Minced Pork Rice from Kafe One Ark	✗
Test14.jpg 	Minced Pork Rice from Kafe One Ark	✗
Test15.jpg 	Minced Pork Rice from Kafe One Ark	✓

Table 5.2.1 Test results of test images

5.3 Confusion matrix

In this project, a confusion matrix will be applied to measure the performance of the Keras deep learning model. This table will contain different combinations of predicted and actual values. It includes four types of terms, which are true positives, true negatives, false positives and false negatives. True positives are the cases that predicted as yes and the actual result is true. Next, true negatives are the cases that predicted as no but the actual result is true. In addition, false positives are the cases predicted as yes but the actual result is false. Last but not least, false negatives are the cases predicted as no but the actual result is true.

	Predicted					
Actual	**Foods images**	Spaghetti - Buddies	Chicken Chop Spaghetti - QS Sweetie Cafe	Burger - QS Sweetie Cafe	Chicken Chop Rice - QS Sweetie Cafe	Minced Pork Rice - Kafe One Ark
	Spaghetti - Buddies	3	0	0	0	0
	Chicken Chop Spaghetti - QS Sweetie Cafe	1	2	0	0	0
	Burger - QS Sweetie Cafe	1	0	2	0	0
	Chicken Chop Rice - QS Sweetie Cafe	0	1	0	2	0
	Minced Pork Rice - Kafe One Ark	0	1	0	1	1

Table 5.3.1 Confusion matrix of Keras Deep Learning Model

Table 5.3.1 shows the confusion matrix of the predicted results. Based on the results, there are 3 correct results on the prediction of Spaghetti from Buddies. Moreover, there are 2 correct results on the prediction of Chicken Chop Spaghetti from QS Sweetie Cafe. For Burger from QS Sweetie Cafe and Chicken Chop Rice from QS Sweetie Cafe, both of them also obtain 2 correct results from the prediction. Moreover, the Keras Deep learning model has predicted only 1 correct result on Minced Pork Rice from Kafe One Ark. Afterwards, the precision values and recall values will be calculated based on the table results above. Generally, a high precision model will produce fewer false positive while a high recall model will produce fewer false negative. Lastly, figure 5.3.1 is the formula to calculate the precision and recall values.

$$Precision_i = \frac{M_{ii}}{\sum_j M_{ji}}$$

$$Recall_i = \frac{M_{ii}}{\sum_j M_{ij}}$$

Figure 5.3.1 Formula of precision and recall values

Foods images	**Precision values**
Spaghetti from Buddies	3 / (3+1+1+0+0) = 0.60
Chicken Chop Spaghetti from QS Sweetie Cafe	2 / (0+2+0+1+1) = 0.5
Burger from QS Sweetie Cafe	2 / (0+0+2+0+0) = 1.00
Chicken Chop Rice from QS Sweetie Cafe	2 / (0+0+0+2+1) = 0.67
Minced Pork Rice from Kafe One Ark	1/ (0+0+0+0+1) = 1.00

Table 5.3.2 Precision values of Food images

Foods images	**Recall values**
Spaghetti from Buddies	3 / (3+0+0+0+0) = 1.00
Chicken Chop Spaghetti from QS Sweetie Cafe	2/ (1+2+0+0+0) = 0.67
Burger from QS Sweetie Cafe	2 / (1+0+2+0+0) = 0.67
Chicken Chop Rice from QS Sweetie Cafe	2 / (0+1+0+2+0) = 0.67
Minced Pork Rice from Cafe One Ark	1/ (0+1+0+1+1) = 0.33

Table 5.3.3 Recall values of Food images

The above table shows the precision values and recall values of each food images. The precision values of Spaghetti from Buddies is 0.60, while Chicken Chop Spaghetti from QS Sweetie Cafe is 0.50. Other than that, there are 1.00 of precision values on Burger from QS Sweetie Cafe as well as Minced Pork Rice from Kafe One. Lastly, Chicken Chop Rice from QS Sweetie Cafe obtained a value of 0.67 for precision values.

For the recall values, there is a value of 1.00 on Spaghetti from Buddies, 0.67 on Chicken Chop Spaghetti from QS Sweetie Café, 0.67 on Burger from QS Sweetie Cafe and 0.67 for the Chicken Chop Rice from QS Sweetie Cafe. Lastly, Minced Pork Rice from Kafe One Ark obtained a value of 0.33 for recall values.

The overall accuracy is the sum of true positives divide by all other values, accuracy = (3+2+2+2+1) / (3+0+0+0+0+1+2+0+0+0+1+0+2+0+0+0+1+0+2+0+0+1+0+1+1) = 0.67. Hence, the overall accuracy of the prediction is 0.67, or 67%.

In my opinion, there are some of the food images obtain a low accuracy during the prediction, this is because this project is on early stage, and the datasets were provided by outside users, so the number of datasets might not sufficient enough for image pre-processing. Therefore, the accuracy of the prediction cannot reach the expectation level. However, as time goes on, there will be more and more datasets provided for each food category, then the accuracy of the prediction definitely will improve gradually. Moreover, building a CNN model in Anaconda Spyder provide a high degree of freedom for us to customize the CNN. It allows us to tune every single possible parameter to improve the performance and accuracy of the deep learning model.

In conclusion, this is a crowdsourcing and timeliness project which depends on incessancy of food images upload by outside users. The greater the number of datasets uploaded, the higher the accuracy can be obtained by image pre-processing. Meanwhile, the experiment on tuning the parameter also the must to improve the accuracy.

CHAPTER 6: Conclusion

Machine learning is an application of Artificial Intelligence (AI) that has strong ability to study and make use of the new knowledge to improve its experience. For instance, examples or data provided by us are used to discover similar patterns or traits to allow the system to make better decisions or choice in the future. The more data provided to the system, the higher the probability of being accurate. The purpose of machine learning is to allow computers to be able to learn by themselves and improve upon mistakes without human instruction.

In order to automate the data annotation for modeling a non-overt features for food origins recognition, machine learning is the best way to be implemented into my proposed system, this is because it allows my system to learn the pattern of the foods, such as appearances, ingredients, serving size as well as the cues on the packaging labels. While Keras is the ideal and powerful python library that helps me to developing and evaluating a deep learning model which able to trace the food origin without referring to the location metadata, but based on the pattern of food image itself.

6.1 Contribution

The reason of proposed this project is because nowadays the social media have high popularity around the world, and most of the people used to snap their foods and posting it on social media, such as Facebook, Instagram, Snapchat and so on. However, those people who may interested on the food but do not know where did the foods been selling. Therefore, the existent of my proposed system will help resolve this common issue.

Other than that, this is a unique and epoch-making project among the deep learning fields. It not only allows users to perform food recognition by uploading food image, it also provides a platform for users to participating in the data collection procedure, so users can contribute one's bit to helping this food tracer system grow in strength. The main reason to allows the participation of outsiders is because there are myriad of foods exist in this world, and the number of datasets and the type of foods can be provided by developer is limited. However, if the users from all over the worlds were uploading the food images and specified the food name and location name constantly, there must be countless type of foods can be predicted in this system.

6.2 Issues and Challenges Encounter

Throughout this project, there was some issue and challenge that encountered in this project. The main stumbling block is the low accuracy in model validation. The main reason can be imputed to the insufficiency of the datasets. This is because this project is in initial stages, so the popularity of this system is very low, and only few users upload images through the system.

The second reason of low accuracy in model validation is the inexperience in the tuning process. This is because the parameters need to be finely tuned in order to build a better CNN model with high validation accuracy. If the values of any required parameters are slightly change, it will affect the performance and the validation accuracy of the CNN model.

Next, the one of the issues of this project is the further development of this system will be difficult due to the nature of my system design. This is because the datasets uploaded by users will be stored at a hosting server. Then, developer need to download all the images after a cycle period, then build a CNN model again and undergo another image pre-processing. Under this system design, it requires developer to maintain the system constantly, otherwise the accuracy of the model cannot be improved.

6.3 Future Work

For the future work of this project, in order to improve the accuracy of the CNN model, a multiple experiment needs to be conducted to obtain the best and proper values for every single parameter that will maximize the accuracy and performance. Secondly, the other mandatory work is to promote this system globally, so there will be more and more users can upload food images through my system.

Lastly, the most important work that need to be done is to implement the continual learning (CL) into my system. By using continual learning, my training model will able to reinforce itself continually from a stream of data. In other words, it will keep retrain the model and adapt in production as new data comes in.

BIBLIOGRAPHY

[1] Bagwan, S. M., & Sankpal, L. J. (2015). VisualPal: A mobile app for object recognition for the visually impaired. *2015 International Conference on Computer, Communication and Control (IC4)*. doi:10.1109/ic4.2015.7375665

[2] Bagwan, Sankpal, "detects the direction of maximum light, detects colors and most importantly recognizes an object", Sinhgad Academy of Engineering Kondhwa),2015.

[3] Bagwan, Sankpal, "Hybrid algorithm that uses Artificial Neural Network and Euclidean Distance measures in combination that makes the proposed system a strong object recognition mobile application", Sinhgad Academy of Engineering Kondhwa),2015.

[4] "Build with AI | DeepAI". *DeepAI*. Retrieved 2018-10-06.

[5] M. (2017, June 28). Aipoly Vision: Sight for Blind & Visually Impaired. Retrieved from https://www.closingthegap.com/aipoly-vision-sight-for-blind-visually-impaired/

[6] A. (n.d.). Retrieved from https://www.tecnoaccesible.net/en/content/aipoly-vision

[7] Image Searcher, Inc. (7 April 2013). "CamFind - Search, QR Reader, Price Comparison & Barcode Scanner". iTunes.

[8] Image Searcher, Inc. "CamFind - Visual Search Engine". Google Play.

[9] Jackie Dove. "CamFind launches CloudSight API for multiplatform visual search functionality". The Next Web.

[10] Bolanos, M., & Radeva, P. (2016). Simultaneous food localization and recognition. *2016 23rd International Conference on Pattern Recognition (ICPR)*. doi:10.1109/icpr.2016.7900117

[11] Steven W Lichtman, Krystyna Pisarska, Ellen Raynes Berman, MichelePestone, Hillary Dowling, Esther Offenbacher, Hope Weisel, StanleyHeshka, Dwight E Matthews, and Steven B Heymsfield. Discrepancybetween self-reported and actual caloric intake and exercise in obesesubjects.New England Journal of Medicine, 327(27):1893–1898, 1992.

[12] Marc Bolanos, Mariella Dimiccoli, and Petia Radeva.Towardsstorytelling from visual lifelogging: An overview.arXiv preprintarXiv:1507.06120, 2015.

[13] Marc Bolanos, Petia Radeva, “In this paper, we propose a novel and fast approach based on CNNs for detecting and recognizing food in both conventional and egocentric vision pictures”, Department of Mathematics and InformaticsUniversitat de Barcelona, Barcelona, Spain, 2016.

[14] (Marc Bolanos, Petia Radeva, 2016, p.3140), “the first food-related objects localization algorithm, which is specifically trained to distinguish images containing generic food and has the ability to propose several bounding boxes containing generic food in a single image”, Department of Mathematics and Informatics Universitat de Barcelona, Barcelona, Spain, 2016.

[15] Marc Bolanos, Petia Radeva, “propose a food recognition algorithm, which learns by re-using food-related knowledge and can be applied on the top of the food localization method
”, Department of Mathematics and Informatics Universitat de Barcelona, Barcelona, Spain, 2016.

[16] (Marc Bolanos, Petia Radeva, 2016, p.3140), “we demonstrate that our methodology is useful for both conventional and egocentric pictures”, Department of Mathematics and Informatics Universitat de Barcelona, Barcelona, Spain, 2016.

APPENDICES

Appendix A: Biweekly Reports

FINAL YEAR PROJECT WEEKLY REPORT

(Project I / Project II)

Trimester, Year: S3, Y3	**Study week no.: 1**
Student Name & ID: LIM JIA HONG, 16ACB03441	
Supervisor: Dr. Aun Yichiet	
Project Title: Modelling non-overt features for food origin recognition	

1. WORK DONE NO
2. WORK TO BE DONE Think the new direction of FYP 2
3. PROBLEMS ENCOUNTERED No problem.
4. SELF EVALUATION OF THE PROGRESS Ok.

Supervisor's signature

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project I / Project II)

Trimester, Year: S3, Y3	**Study week no.: 3**
Student Name & ID: LIM JIA HONG, 16ACB03441	
Supervisor: Dr. Aun Yichiet	
Project Title: Modelling non-overt features for food origin recognition	

1. WORK DONE

Confirm the direction of FYP 2

2. WORK TO BE DONE

Build a website for users to upload food images

3. PROBLEMS ENCOUNTERED

No problem.

4. SELF EVALUATION OF THE PROGRESS

Ok.

Supervisor's signature

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project I / Project II)

Trimester, Year: S3, Y3	**Study week no.: 5**
Student Name & ID: LIM JIA HONG, 16ACB03441	
Supervisor: Dr. Aun Yichiet	
Project Title: Modelling non-overt features for food origin recognition	

1. WORK DONE

Successfully build a website and deploy it on server

2. WORK TO BE DONE

Build a CNN model

3. PROBLEMS ENCOUNTERED

No problem.

4. SELF EVALUATION OF THE PROGRESS

Ok.

Supervisor's signature

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project I / Project II)

Trimester, Year: S3, Y3	**Study week no.: 7**
Student Name & ID: LIM JIA HONG, 16ACB03441	
Supervisor: Dr. Aun Yichiet	
Project Title: Modelling non-overt features for food origin recognition	

1. WORK DONE

Half way of building CNN

2. WORK TO BE DONE

Build CNN model and undergo image pre-processing

3. PROBLEMS ENCOUNTERED

Face some unknown issues, and discuss it with supervisor

4. SELF EVALUATION OF THE PROGRESS

Not satisfied.

Supervisor's signature

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project I / Project II)

Trimester, Year: S3, Y3	**Study week no.: 9**
Student Name & ID: LIM JIA HONG, 16ACB03441	
Supervisor: Dr. Aun Yichiet	
Project Title: Modelling non-overt features for food origin recognition	

1. WORK DONE

Able to train the model

2. WORK TO BE DONE

Deploy the deep learning model into production and allow users to undergo prediction

3. PROBLEMS ENCOUNTERED

No problem.

4. SELF EVALUATION OF THE PROGRESS

Ok.

Supervisor's signature

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project I / Project II)

Trimester, Year: S3, Y3	**Study week no.: 11**
Student Name & ID: LIM JIA HONG, 16ACB03441	
Supervisor: Dr. Aun Yichiet	
Project Title: Modelling non-overt features for food origin recognition	

1. WORK DONE

Almost done for deploying deep learning model

2. WORK TO BE DONE

Deploy the deep learning model, complete the documentation

3. PROBLEMS ENCOUNTERED

Face some issues on deploy process

4. SELF EVALUATION OF THE PROGRESS

So far ok

Supervisor's signature

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project I / Project II)

Trimester, Year: S3, Y3	**Study week no.: 13**
Student Name & ID: LIM JIA HONG, 16ACB03441	
Supervisor: Dr. Aun Yichiet	
Project Title: Modelling non-overt features for food origin recognition	

1. WORK DONE

Done deploying the training model and documentation

2. WORK TO BE DONE

Review the overall system performance and documentation. Record video for live demo

3. PROBLEMS ENCOUNTERED

No problem.

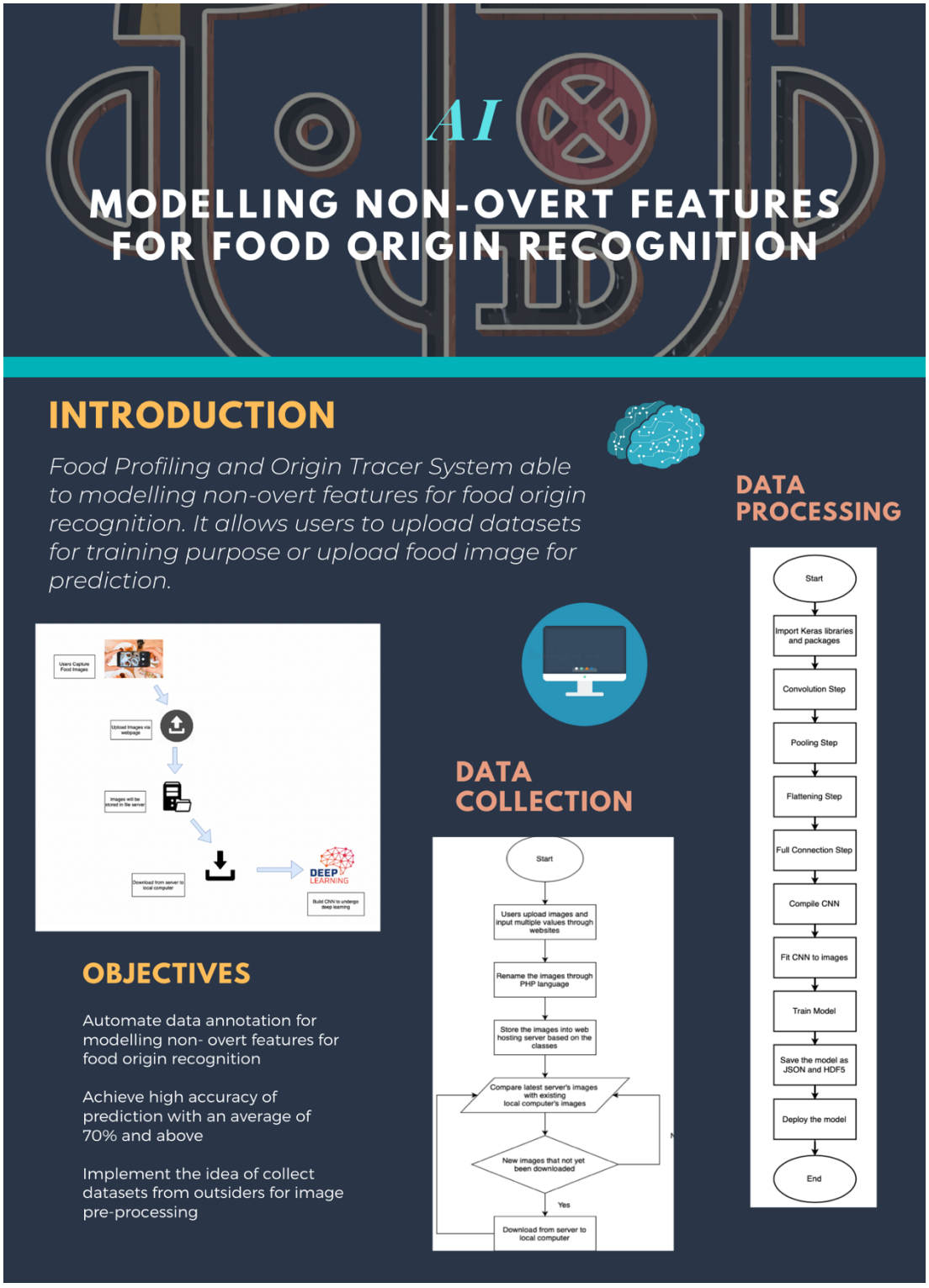
4. SELF EVALUATION OF THE PROGRESS

Ok.

Supervisor's signature

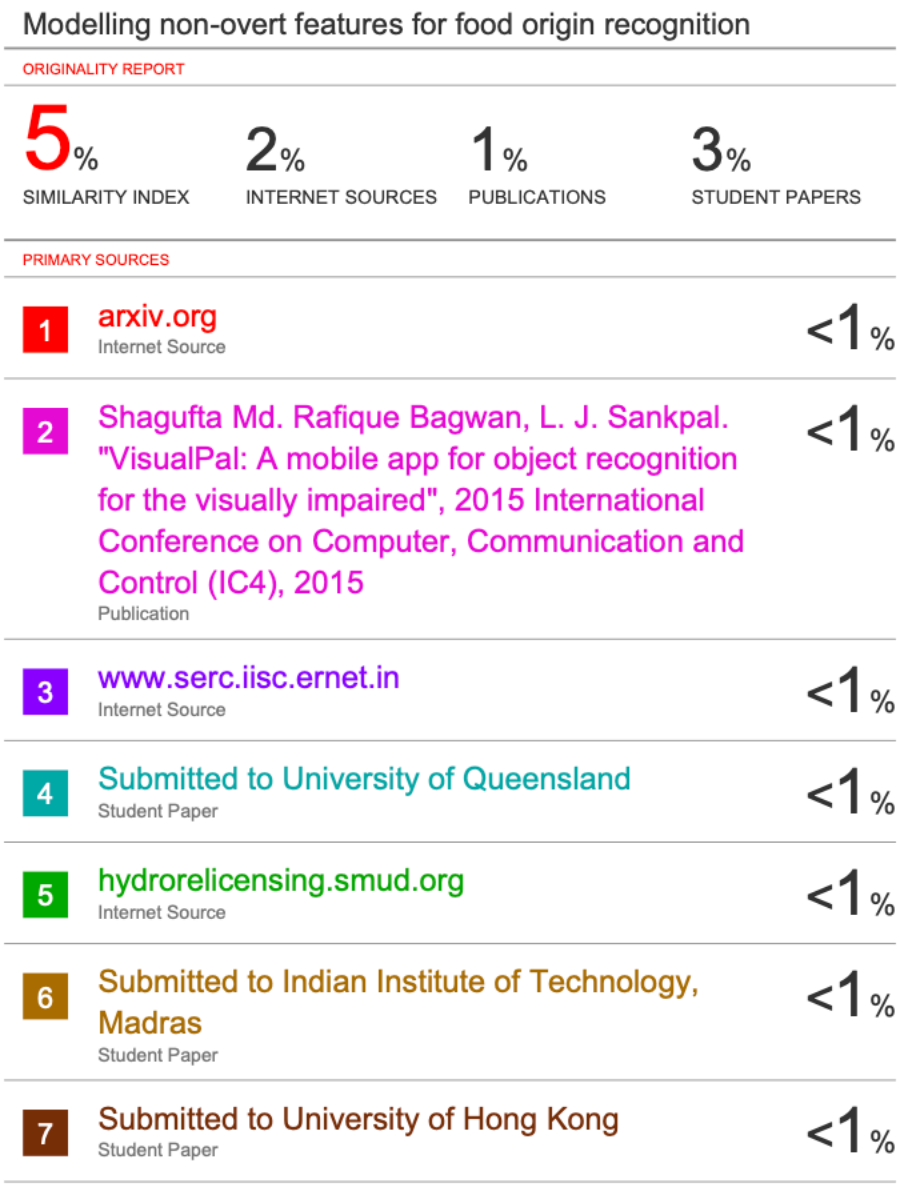
Student's signature

Appendix B: Poster



PLAGIARISM CHECK RESULT

Appendix C: Plagiarism Check Summary



Modelling non-overt features for food origin recognition

ORIGINALITY REPORT

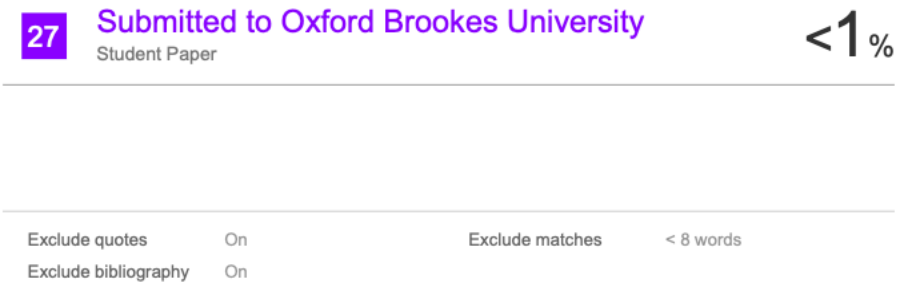
5%	2%	1%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	arxiv.org Internet Source	<1%
2	Shagufta Md. Rafique Bagwan, L. J. Sankpal. "VisualPal: A mobile app for object recognition for the visually impaired", 2015 International Conference on Computer, Communication and Control (IC4), 2015 Publication	<1%
3	www.serc.iisc.ernet.in Internet Source	<1%
4	Submitted to University of Queensland Student Paper	<1%
5	hydrorelicensing.smud.org Internet Source	<1%
6	Submitted to Indian Institute of Technology, Madras Student Paper	<1%
7	Submitted to University of Hong Kong Student Paper	<1%

8	Submitted to Grant Union High School Student Paper	<1%
9	Submitted to RMIT University Student Paper	<1%
10	Submitted to University of Northumbria at Newcastle Student Paper	<1%
11	Submitted to University of Nigeria Student Paper	<1%
12	studylib.net Internet Source	<1%
13	Submitted to University of Northampton Student Paper	<1%
14	Submitted to University of Florida Student Paper	<1%
15	Qing Yu, Masashi Anzawa, Sosuke Amano, Makoto Ogawa, Kiyoharu Aizawa. "Food Image Recognition by Personalized Classifier", 2018 25th IEEE International Conference on Image Processing (ICIP), 2018 Publication	<1%
16	Shubham Melvin Felix, Sumer Kumar, A. Veeramuthu. "A Smart Personal AI Assistant for Visually Impaired People", 2018 2nd International Conference on Trends in	<1%

	Electronics and Informatics (ICOEI), 2018 Publication	
17	Submitted to National Institute Of Technology, Tiruchirappalli Student Paper	<1%
18	cliftondevelopment.com Internet Source	<1%
19	Submitted to University of Wales, Bangor Student Paper	<1%
20	deepai.org Internet Source	<1%
21	Submitted to Varsity College Student Paper	<1%
22	Submitted to Udayana University Student Paper	<1%
23	Submitted to University of Stellenbosch, South Africa Student Paper	<1%
24	Submitted to University of Sydney Student Paper	<1%
25	Submitted to International Islamic University Malaysia Student Paper	<1%
26	Submitted to Rutgers University, New Brunswick Student Paper	<1%


27
Submitted to Oxford Brookes University
Student Paper
<1%
Exclude quotes On
Exclude bibliography On
Exclude matches < 8 words

Appendix D: FM-IAD-005

Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1of 1

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	LIM JIA HONG
ID Number(s)	16ACB03441
Programme / Course	Bachelor Of Information Systems (Hons) Business Information Systems
Title of Final Year Project	Modelling non-overt features for food origin recognition

Similarity	**Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)**
Overall similarity index: 5 % **Similarity by source** Internet Sources: 2 % Publications: 1 % Student Papers: 3 %	
Number of individual sources listed of more than 3% similarity: 0	

Parameters of originality required and limits approved by UTAR are as Follows:

(i) Overall similarity index is 20% and below, and

(ii) Matching of individual sources listed must be less than 3% each, and

(iii) Matching texts in continuous block must not exceed 8 words

Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.

Note: Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

Signature of Supervisor
Name: Aun YC
Date: 23/4/2020

Signature of Co-Supervisor
Name: _____
Date: _____

UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY
(KAMPAR CAMPUS)

CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	16ACB03441
Student Name	LIM JIA HONG
Supervisor Name	Dr. Aun Yichiet

TICK (√)	DOCUMENT ITEMS Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
√	Front Cover
√	Signed Report Status Declaration Form
√	Title Page
√	Signed form of the Declaration of Originality
√	Acknowledgement
√	Abstract
√	Table of Contents
√	List of Figures (if applicable)
√	List of Tables (if applicable)
	List of Symbols (if applicable)
√	List of Abbreviations (if applicable)
√	Chapters / Content
√	Bibliography (or References)
√	All references in bibliography are cited in the thesis, especially in the chapter of
√	Appendices (if applicable)
√	Poster
√	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)

*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report. (Signature of Student) Date: 19 April 2020	Supervisor verification. Report with incorrect format can get 5 mark (1 grade) reduction. (Signature of Supervisor) Date: 23/4/2020