



The Chinese University of Hong Kong
Department of Computer Science and Engineering

LYU1802 - BotanWiki

Final Year Project
1st Term Report

Wong Tsz Hin 1155079510
Yan Chi Shing 1155078689

Supervised by: Prof. Lyu Rung Tsong Michael

Table of Contents

0. Abstract	4
1. Introduction	5
1.1. Motivation	5
1.2. Significance of the project	6
1.3. Goal	6
2. Technology Overview	7
2.1. TensorFlow	7
2.2. OpenCV and Keras	7
2.3. Android Studio	8
3. Preparation	10
3.1. Data Collection	10
3.2. Preprocessing	12
3.2.1. Data Specification	12
3.2.2. Data Augmentation.....	14
4. Mobile Application	16
4.1. Research on current market	16
4.1.1. Highlight 1 – Online Database	18
4.1.2. Highlight 2 – Assistance of Recognition	20
4.1.3. Highlight 3 – Community for user to share the plant photos	22
4.1.4. Highlight 4 – Development Issue	24
4.2. Design Specification	26
4.2.1. Architecture Design	26
4.2.2. Data Flow Diagram.....	28
4.2.3. Structural Diagram.....	30
4.2.4. UMLs	31
4.2.4.1. Use Case Diagram.....	31
4.2.4.1.1. Activity Diagram – Basic Recognition	32
4.2.4.1.2. Activity Diagram – Navigation	34
4.2.4.2. Functionality.....	36
4.2.5. User Interface	37
4.2.5.1. Screen Capture	37
4.2.6. Procedure of using Dr. Leaf	40
4.2.6.1. Basic Recognition.....	40
4.2.6.1.1. Step 1	40
4.2.6.1.2. Step 2	41
4.2.6.1.3. Step 3	42

4.2.6.1.3.1. Step 3.1	43
4.2.6.1.3.2. Step 3.2	44
4.2.6.2. View All plants and navigation	45
4.2.6.2.1. Step 1	45
4.2.6.2.1.1. Step 1.1	46
4.2.5.2.2. Step 2	47
5. Testing and Evaluation	48
5.1. Database	48
5.2. Logical Statement	48
5.3. UI Testing	49
6. AI Model	50
6.1. Background	50
6.2. Problem Defining and Solution	51
6.3. Neural Network	51
6.4. CNN	53
6.4.1. Concern	56
6.4.2. Portability	56
6.4.3. Speed	56
6.4.4. Develop Period	57
6.5. MobileNets	58
6.5.1. Depth-wise Separable Convolution	59
6.5.2. Hyperparameter	60
6.5.3. Performance	62
6.6. Workflow	63
6.6.1. Pretrained Model and Transfer Learning	63
6.6.2. Data	64
6.7. Experiment	65
7. Limitation and difficulties	66
7.1. Data Collection	66
7.2. Knowledge about plants	68
8. Conclusion	69
8.1. Mobile Application	69
8.1.1. Modularity	69
8.1.2. Evolvability	70
8.1.2.1. Database	70
8.1.2.2. Resource file system in Android Studio	72
8.1.3. User Friendliness	75

8.2. AI Model	76
9. Future Development.....	77
9.1. Mobile Application.....	77
9.1.1. Function.....	77
9.1.1.1. Recognition History.....	77
9.1.1.2. Advanced view all plants function	77
9.1.1.3. Offline data for the plants.....	78
9.1.2. UI/UX Design	79
9.1.3. Testing	80
9.2. AI Model	81
9.2.1. More Species	81
9.2.2. Visualization.....	82
10. Acknowledgement	83
11. References.....	84
12. Appendix.....	86
12.1. List of Figures.....	86
12.2. List of Tables.....	88

0. Abstract

Image recognition has an astonishing result in computer with the help of artificial Intelligence. The project has been kicked off and built an offline mobile application, a sustainable platform in mobile device to recognize plants by leaves in the Chinese University of Hong Kong, with a low error rate of recognizing the plants among the data we collected.

1. Introduction

1.1. Motivation

Being students studying computer science in CUHK, where is the most beautiful and scenes campus in Hong Kong, we always see different kind of trees and flowers shuttling in the school. Some of them have the peculiar appearance, some of them bloom the colorful flowers. Walking through the vivid campus. We might want to find out what kind of plant it is. Although some plants have already been labelled with the QR code, most of them do not have their own tag or lost the tag due to insufficient maintenance of the organization in charge.

AI technology nowadays have a wide range of research and usage, contributing to the society with influential impact. In the area of image recognition, neural network has the astonishing result of successfully recognizing object by training, for example a residual network has achieved recognizing the ImageNet test set, which has more than two hundred categories of object, only 3.57% error¹. Therefore, we would like to make an attempt to build a mobile application, equipped with artificial Intelligence to recognition the plants in CUHK in a mobile way.

¹ Result is presented in the article discussing a residual learning approach for recognizing images, see the article “Deep residual learning for image recognition”

1.2. Significance of the project

Plants has a vital role in the life of human as well as the ecosystem. Distinguishing the species of plants over a massive category is challenging and expensive as it requires a comprehensive study of the plants, including the growing habits, the habitat, the particular characteristic of the stem, leaves and so on.

This project would like to provide a quick and inexpensive way to recognize plants by the leaf. It may be possible to reduce the workload of botanists when they are working on research with a massive data.

People in urban area can access internet with their mobile device easily, whereas people in rural area may not access the internet with the robust connection. In this circumstance, the mobile application would like to deliver an offline solution for people without internet connection to recognize plants in countryside.

Although some projects have already been initiated to recognize trees and plants in CUHK. This project would try to utilize neural network to contribute the recognition of plants with an alternative method.

1.3. Goal

The project is aimed at building a mobile application with artificial Intelligence in image recognition to distinguish plants without relying on internet access.

2. Technology Overview

2.1. TensorFlow

TensorFlow is an open source machine learning library for developers to implement a wide range of machine learning techniques, from basic regression to classification. Google has also developed TensorFlow Lite, targeting at support machine learning in the mobile application development.

2.2. OpenCV and Keras

OpenCV is the open source computer vision library for handling pictorial data in computer. In this project, we use to get do some file transformation and data augmentation for the training data. The version we used is OpenCV3.0

```

1 import cv2
2 from glob import glob
3 from keras.preprocessing.image import ImageDataGenerator, array_to_img, img_to_array, load_img
4 import os
5
6 datagen = ImageDataGenerator(
7     zca_whitening=False,
8     rotation_range=40,
9     width_shift_range=0.2,
10    height_shift_range=0.2,
11    shear_range=0.5,
12    zoom_range=0.2,
13    horizontal_flip=True,
14    fill_mode='nearest')
15
16 pngs = glob('./**/*.png', recursive=True)
17 for j in pngs:
18
19
20     fileName = os.path.basename(j)
21     fileDirName = os.path.dirname(j)
22     print("Now at" + fileDirName)
23     print("Now in " + fileName)
24     print("----")
25
26     img = cv2.imread(j) # this is a PIL image
27     img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
28
29     #plt.imshow(img)
30     #print(img.shape)
31     img = img.reshape((1,) + img.shape)
32     #print(img.shape)
33     #x = img_to_array(img) # this is a Numpy array with shape (3, 150, 150)
34     #x = x.reshape((1,) + x.shape) # this is a Numpy array with shape (1, 3, 150, 150)
35
36
37     i = 0
38     for batch in datagen.flow(img, batch_size=10, save_to_dir=fileDirName, save_prefix=fileName[:2], save_format='jpg'):
39         i += 1
40         if i > 4:
41             break # otherwise the generator would loop indefinitely

```

Figure 1: Data Augmentation of shearing using OpenCV and Keras

2.3. Android Studio

We would like to initiate the mobile application on android platform, since the many people are now using android phone. According to the statistic from IDC on smartphone market share, Android phone has 84.8% share, and iPhone have only 15.1% share².

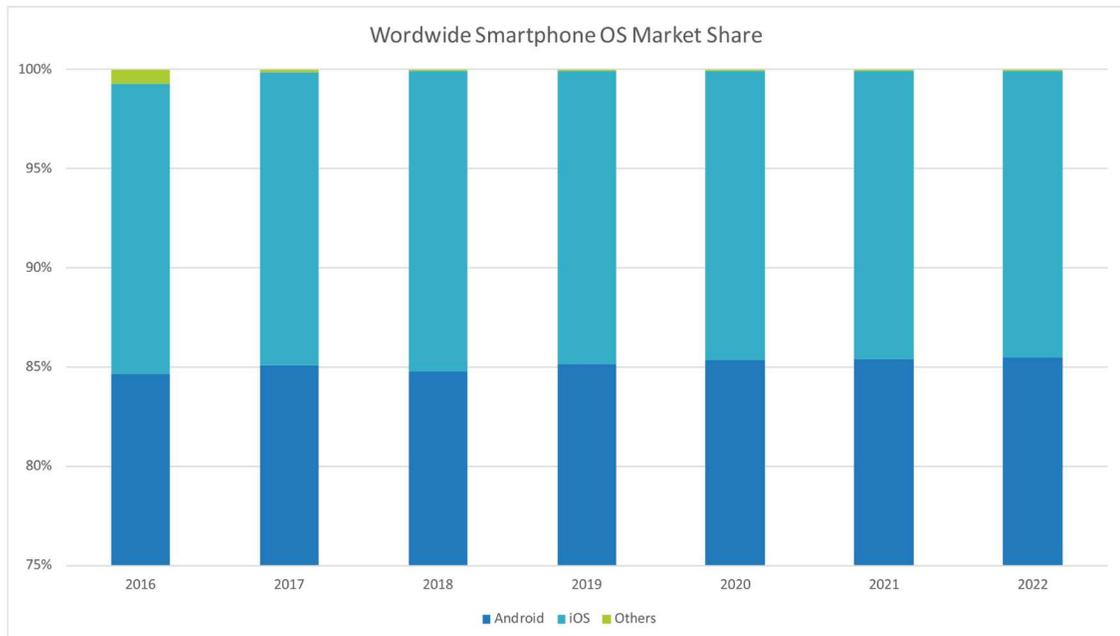


Figure 2: Statistic on Smartphone OS Market share from IDC

Moreover, getting support of android-related issue from online is easier than getting support of ios-related issue, from the question counting in stackoverflow. Therefore, we choose our project in the android platform.

² Statistical Data obtain from IDC researching on smartphone market share.

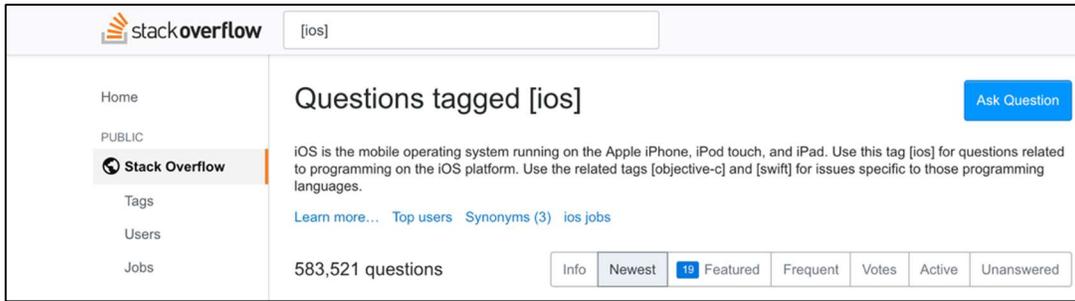


Figure 3: 583,521 questions have been asked about ios in stackoverflow

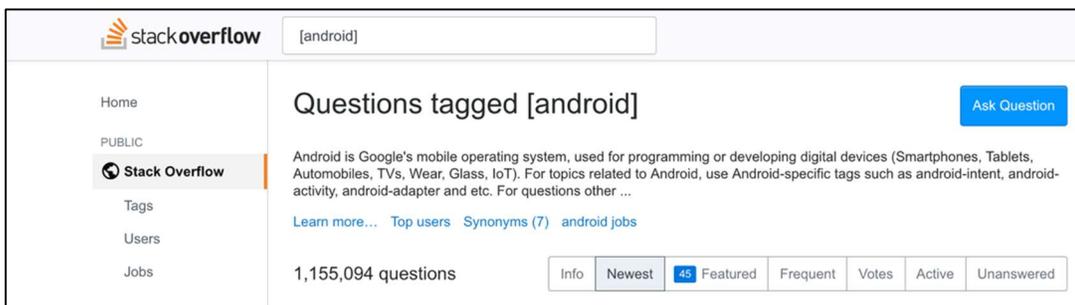


Figure 4: 1,155,094 questions have been asked about android in stackoverflow

Android Studio is the Android official Integrated Development Environment developed by Google. People who are interested in developing android application can take advantages from the IDE and build the app in a fast way.

3. Preparation

3.1. Data Collection

We have done some research about some database regarding leaves of trees, but we could not find a suitable data set for the project. As a result, we decided to collect the data by taking photos of the leaves by ourselves. With the guidance of the Curator of CUHK The Shiu-Ying Hu Herbarium, David Lau, we followed the online education website CUHK campus 100 plants³, and take the picture of 12 species of trees/plants. However, in the experiments we conducted and the development of the application, we only chose 10 species to train our model. Since “Araucaria heterophylla” and “podocarpus macrophyllus” are two plants that the leaves completely different from other plants’ leaves (see figure below for comparison), we would to keep our experiment in a more controllable way. Therefore, we kept the data collected away and trained the model with the rest of 10.

	A	B	C	D	E
1	scientific name	abbreviations	Herbarium URL / Wiki Uri	Google Map	Lat,Lng
2	Camellia granthamiana	cg	http://syhuherbarium.sls.cuhk.edu.hk/collections/factsheet-pro/camellia-granthamiana/	https://goo.gl/maps/KbpuhDqAzRB2	22.415972,114.209083
3	canarium album	ca	http://syhuherbarium.sls.cuhk.edu.hk/collections/factsheet-pro/canarium-album/	https://goo.gl/maps/8thiw1dcJ22	22.415778,114.208917
4	androanthus chrysanthus	hc	http://syhuherbarium.sls.cuhk.edu.hk/collections/factsheet-pro/androanthus-chrysanthus/	https://goo.gl/maps/VdqFyqQ2ej	22.416278,114.209278
5	Hibiscus rosa-sinensis	hr	https://en.wikipedia.org/wiki/Hibiscus_rosa-sinensis	https://goo.gl/maps/SauDso9XLH2	22.415806,114.209333
6	camellia japonica	cj	http://syhuherbarium.sls.cuhk.edu.hk/collections/factsheet-pro/camellia-japonica/	https://goo.gl/maps/vnPmbl6wsPB2	22.416028,114.209139
7	cinnamomum burmannii	cb	http://syhuherbarium.sls.cuhk.edu.hk/collections/factsheet-pro/cinnamomum-burmannii/	https://goo.gl/maps/wfdnYDtzyuD2	22.415764,114.207528
8	baubinia variegata	bv	http://syhuherbarium.sls.cuhk.edu.hk/collections/factsheet-pro/baubinia-variegata/	https://goo.gl/maps/w62A16Avy8L2	22.416923,114.206799
9	sterculia lanceolata	sl	http://syhuherbarium.sls.cuhk.edu.hk/collections/factsheet-pro/sterculia-lanceolata/	https://goo.gl/maps/9eTToGZHFwv	22.417028,114.206833
10	machilus chekiangensis	mc	http://syhuherbarium.sls.cuhk.edu.hk/collections/factsheet-pro/machilus-chekiangensis/	https://goo.gl/maps/3KkgxvZJlMn	22.417583,114.206861
11	ficus altissima	fa	http://syhuherbarium.sls.cuhk.edu.hk/collections/factsheet/ficus-altissima/	https://goo.gl/maps/rSZJQqr4S62	22.417830,114.205124
12	podocarpus macrophyllus	pm	http://syhuherbarium.sls.cuhk.edu.hk/collections/factsheet/podocarpus-macrophyllus/	https://goo.gl/maps/zetowL8DC4Q2	22.418000,114.205083
13	Araucaria heterophylla	ah	http://www2.cuhk.edu.hk/cuhktree/TreeDetails.aspx?tid=9&loc=Franklin	https://goo.gl/maps/cReQuGxR9G52	22.418083,114.204806
14					
15					
16					
17					

Figure 5: Excel Table show the plants we collected

³ The website will show the map with plant marked in English name, <http://syhuherbarium.sls.cuhk.edu.hk/collections/courseware/campus-100-plants/>



Figure 6 : *Araucaria heterophylla*



Figure 7 : *podocarpus macrophyllus*



Figure 8 : Photo of "*machilus chekiangensis*"



Figure 9: Photo of "*ficus altissima*"



Figure 10 : Photo of "*cinnamomum burmannii*"

3.2. Preprocessing

3.2.1. Data Specification

With the limitation we mentioned later⁴, we collected the 12 species of plants and we preprocess the photo into two main categories, Single Leaf and Multiple leaves.



Figure 11: Example photo of single leaf



Figure 12: Example photo of Multiple Leaves

Name	Amount of photo
Araucaria heterophylla	N/a
bauhinia variegata	138
Camellia granthamiana	182
camellia japonica	261
canarium album	137
cinnamomum burmannii	139
ficus altissima	209

⁴ See 7. Limitations and difficulties for the elaboration.

handroanthus chrysanthus	254
Hibiscus rosa-sinensis	158
machilus chekiangensis	209
podocarpus macrophyllus	N/a
sterculia lanceolata	231

Table 1: Single leaf data set

Name	Amount of photo
Araucaria heterophylla	N/a
bauhinia variegata	150
Camellia granthamiana	164
camellia japonica	166
canarium album	166
cinnamomum burmannii	166
ficus altissima	203
handroanthus chrysanthus	201
Hibiscus rosa-sinensis	133
machilus chekiangensis	202
podocarpus macrophyllus	N/a
sterculia lanceolata	154

Table 2: Multiple leaves data set

3.2.2. Data Augmentation

In order to enhance the performance of AI by simulating different photo taking angles, we performed data augmentation to our data set, increase around 5 times larger than before. We use ImageDataGenerator from keras⁵, we augment the image 5 times with the condition set below,

- 1) Rotation in the range of 40 for a picture randomly
- 2) Width shifting in the range of 0.2 pixel randomly
- 3) Height shifting in the range of 0.2 pixel randomly
- 4) Shear in the range of 0.5 counter-clockwise direction in degree randomly
- 5) Zoom in the range of 0.2 times randomly
- 6) Horizontal flip randomly

After we did the augmentation, the amount of the data is shown as following,

Name	Amount of photo
Araucaria heterophylla	N/a
bauhinia variegata	798
Camellia granthamiana	1054
camellia japonica	1466
canarium album	793
cinnamomum burmannii	805
ficus altissima	1193
handroanthus chrysanthus	1436
Hibiscus rosa-sinensis	911
machilus chekiangensis	1203
podocarpus macrophyllus	N/a
sterculia lanceolata	1315

Table 3: Augmented Single leaf data set

⁵ Keras library provides multiple ways to process image data, see <https://keras.io/preprocessing/image/>

Name	Amount of photo
Araucaria heterophylla	N/a
bauhinia variegata	875
Camellia granthamiana	954
camellia japonica	965
canarium album	959
cinnamomum burmannii	964
ficus altissima	1175
handroanthus chrysanthus	1161
Hibiscus rosa-sinensis	776
machilus chekiangensis	1154
podocarpus macrophyllus	N/a
sterculia lanceolata	900

Table 4: Augmented multiple leaves data set

All in all, we have the following amount of to do the experiment and train the model,

	Original	Augmented
Single Leaf	1918	10974
Multiple leaves	1705	9883
Mix (Single leaf + multiple leaves)	3623	20857

Table 5: All data set

4. Mobile Application

4.1. Research on current market

Currently, there are some applications which are developed with the same purpose of performing the recognition on different plants with their photos. We have downloaded and tried out some of them, having the first-hand experience on the service provided by the mobile application. Also, we will evaluate, learn and take advantages from the application in the market.

Name of Application	Availability of Platform	Origin	Features	Installation Count
PlantSnap	<ul style="list-style-type: none"> • IOS • Android 	United State	<ul style="list-style-type: none"> • 585,000 species of plants which are available for classification in the database, with 90%⁶ of accuracy • Supported Identification include plant, mushroom and cactuses • User Login system for further interaction between users on the plant identification 	5,000,000+ in Google Play

⁶ Statistical data such as the size of database and accuracy of identification mentioned in official website, <https://www.plantsnap.com/>

PictureThis (形色 - 拍照識花識別植物)	<ul style="list-style-type: none"> • IOS • Android 	China	<ul style="list-style-type: none"> • 4,000 species of plants which are available for classification in the database, with 98%⁷ of accuracy • Response in 1 second for the recognition result • Online community for sharing recognition of plant 	500,000+ in Google Play
Pl@ntNet	<ul style="list-style-type: none"> • IOS • Android 	N/A	<ul style="list-style-type: none"> • 7657 species of plants in USA, 6700 species of plants in Western Europe⁸ • No accuracy provided • User can indicate which part of the plant are taken in the photo, to increase the performance of recognition 	5,000,000+ in Google Play

Table 6: Summary of mobile applications

⁷ Statistical data such as the size of database and accuracy of identification mentioned in official website, <http://www.xingseapp.com/>

⁸ Statistical data such as the size of database and accuracy of identification mentioned in official website, <https://identify.plantnet-project.org/>

4.1.1. Highlight 1 – Online Database

We discover that all the mobile application we tested rely on the online database to do the recognition, so that they are able to perform the recognition over a massive species of plants. Although the source is large enough to cover as many as kind of plants, slow network will cause the bad performance on the claimed “instant recognition” and deliver unpleasant user experience to people who want to know the plant instantly.

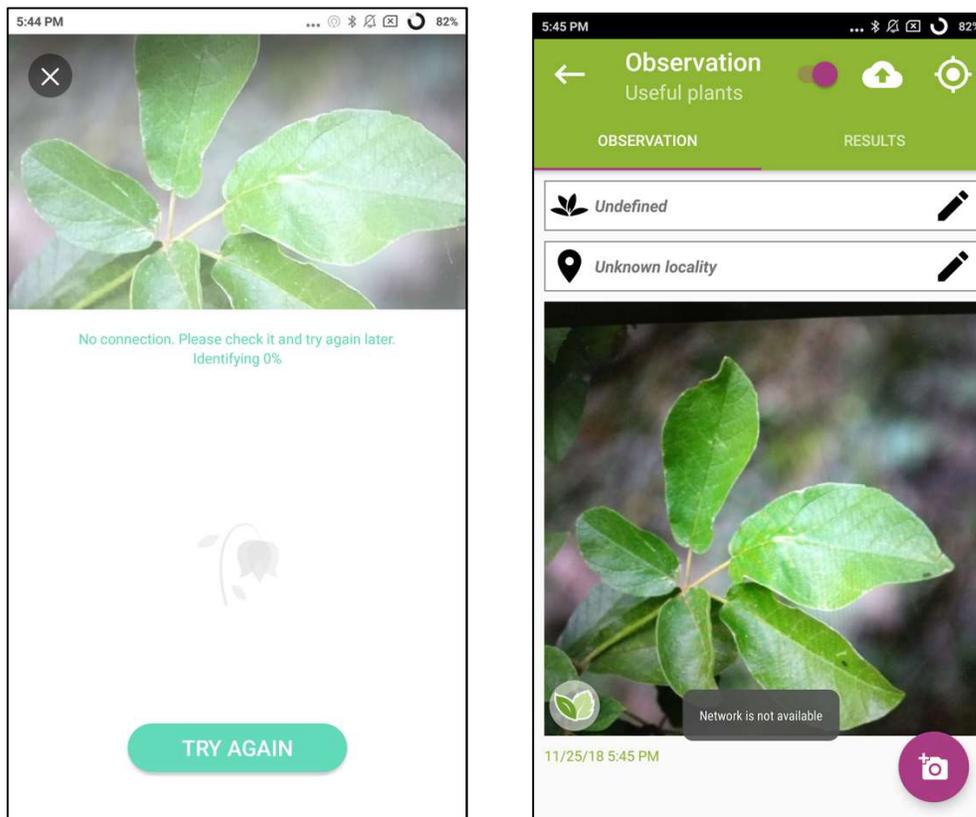


Figure 13 : Network Error in PictureThis and Pl@ntNet

As a result, we consider that user may always visit the countryside or some places where the network condition is unsatisfied, we will try to build the mobile application with offline database and do the recognition locally in the device.

4.1.2. Highlight 2 – Assistance of Recognition

In order to improve the recognition result, some of the application provide the extra feature tag or photo taking guidance for user. In pl@ntNet, it allows user to take picture of several parts of a plant, for example leaf, flower or fruit, and user have to indicate the part they are taking manually before doing the recognition. In PictureThis, user can shoot the picture with the provided focusing area, therefore the input picture to the modal will be standardized, having the similar format to the training data consisting of only leaves or being inside the valid area.

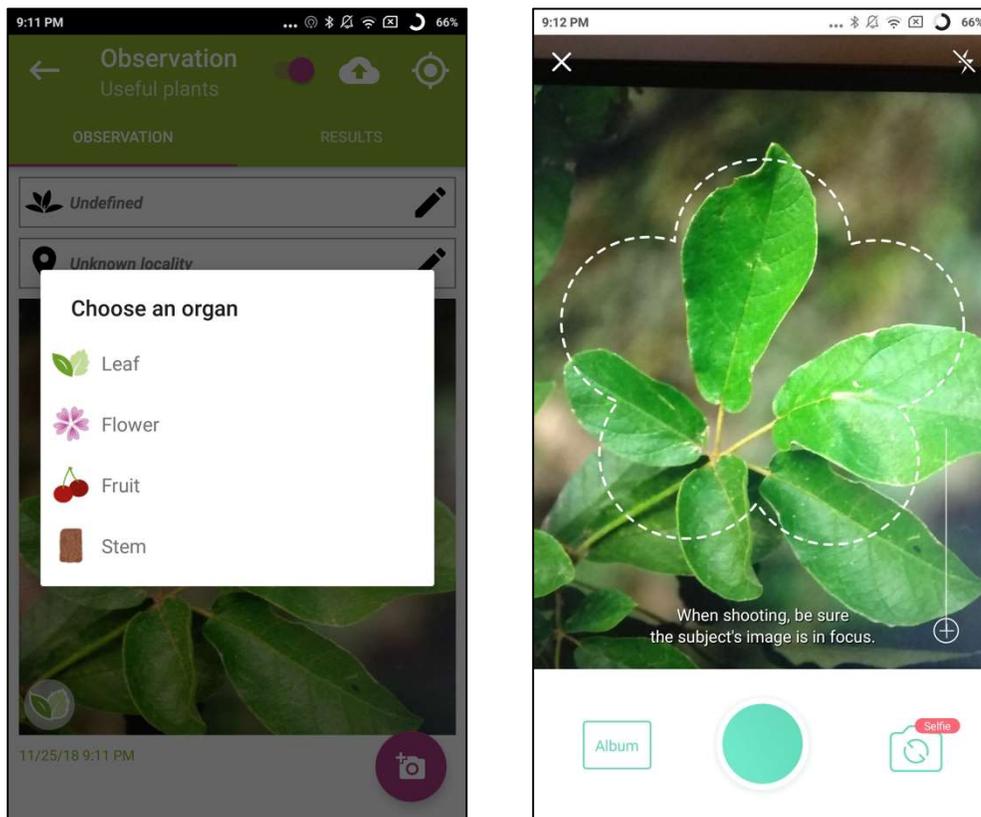


Figure 14 : Method using in pl@ntNet and PictureThis as assistance of recognition

It is a good way to control the input data with the guideline to the user, resulting in a more precise plant recognition. However, it may not be a user-friendly experience in the user perspective as they have to choose the option manually, in which will slow down the speed of getting the result from the application.

4.1.3. Highlight 3 – Community for user to share the plant photos

A large amount of training data is advantageous to an AI modal to get a satisfied recognition result. The application we found also set up a platform for user to share the photo they took with the label. Not only can users enjoy sharing the photo with others through the platform, they are able to rate others photo if they think it have a correct label. Developers can record the user behavior of rating others' photos and enhance the database of plant with the help of massive users.

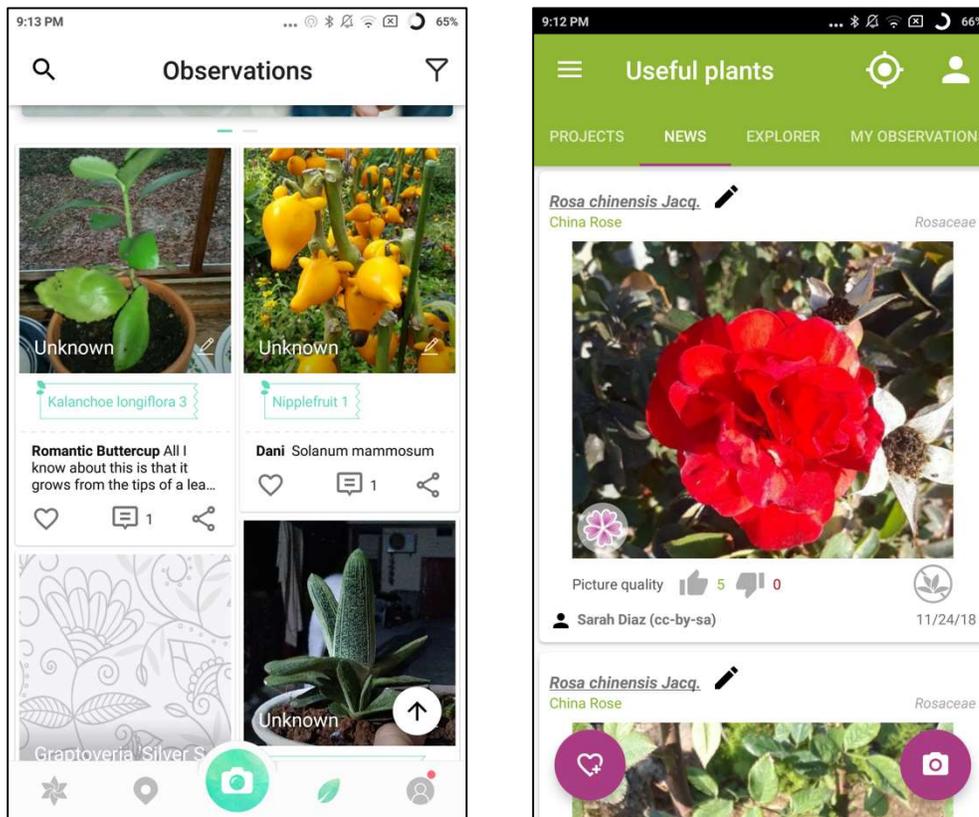


Figure 15 : Community for users in PictureThis and Pl@ntNet

This kind of community can result in win-win situation for both user and the developer, because the users can communicate and enjoy the interaction to others plant lovers who are using the same application. Handling massive leaf pictorial data is an arduous task for developer, this community can bring a great effort contributing to the mission of processing training data. The only concern of the community is that if the users do not have particular knowledge of the plant, they may make the wrong rating, causing the inappropriate training data for the model. Develop still cannot merely rely on the community to get the massive training data.

4.1.4. Highlight 4 – Development Issue

For the application “PlantSnap”, we made an attempt to try and test the function of it. However, we are not able to take a single picture with the application. It comes the mobile application development issue regarding testing in different devices. Portability of the application is the main concern of developer, since there are at least five important platform with enormous model of mobile phone respectively. Developers are not possible to test the application among all the device with several operation systems, and we are suggested to keep the basic and common feature and components that are available across in all platforms and build up the application upon those variants and components⁹.

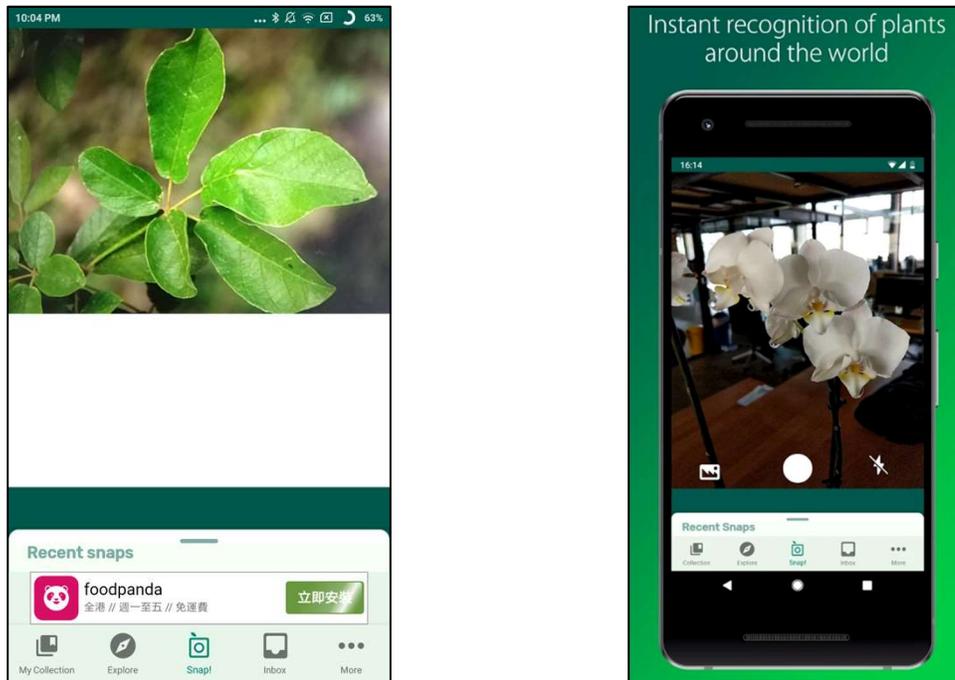


Figure 16 : Failure of capturing picture in PlantSnap and capture in Google Play of the UI

⁹ Issue discussed in article “Software engineering issues for mobile application development”, the five important are iPhone, Android, BlackBerry, Windows Phone, Symbian. In fact, IOS (15.1%) and Android (84.8%) should be the most significant platform for mobile application, according to the statistical data from IDC on worldwide Smartphone OS.

Reference: <https://www.idc.com/promo/smartphone-market-share/os>

In the application, we cannot use the photo capturing function as there is no button shown to do the task. The problem may be caused by the inadequate testing or not using the common components mentioned as above to compose the photo taking function. We may try to avoid it by calling the camera library to finish the photo taking step in our application.

4.2. Design Specification

The mobile application is developed as the offline AI application to recognize the specie of the plant by the photo of leaves. The following are the specification of the application.

4.2.1. Architecture Design

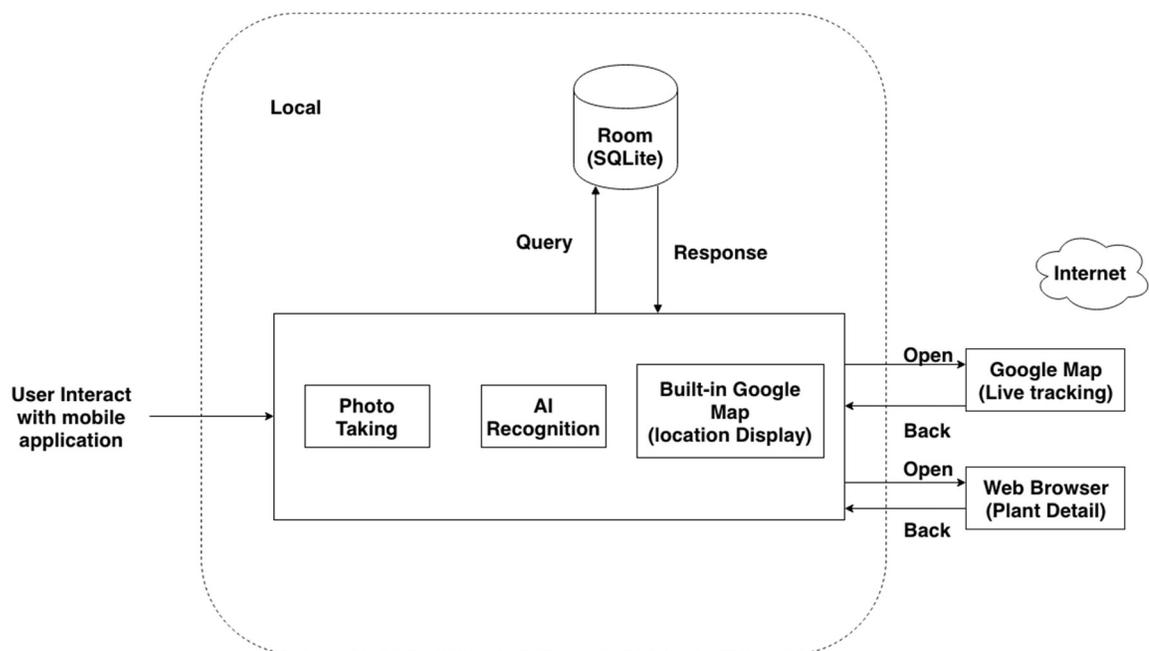


Figure 17 : Architecture Design Diagram

The mobile application is mainly composed by the functions, which can operate independently without internet access. Users will take photo of the leaf of the plant and pass the pictorial data as the input to application. After that, users will gain the plant result by the AI Recognition model locally in the mobile.

If the users want to get the details of the specific plant, they can click the detail button and the application will make an intent, requesting to open the browser and check out the detail of the plant in the website in CUHK Shiu-Ying Hu Herbarium.

If the users want to get the location of the plant in CUHK, they can click the location button and the application will make an intent, to open Google Map showing the particular plant location that we took data at.

The application also provides the built-in Google map that will not require mobile data, and mark all the plants on it that available for user to perform recognition.

4.2.2. Data Flow Diagram

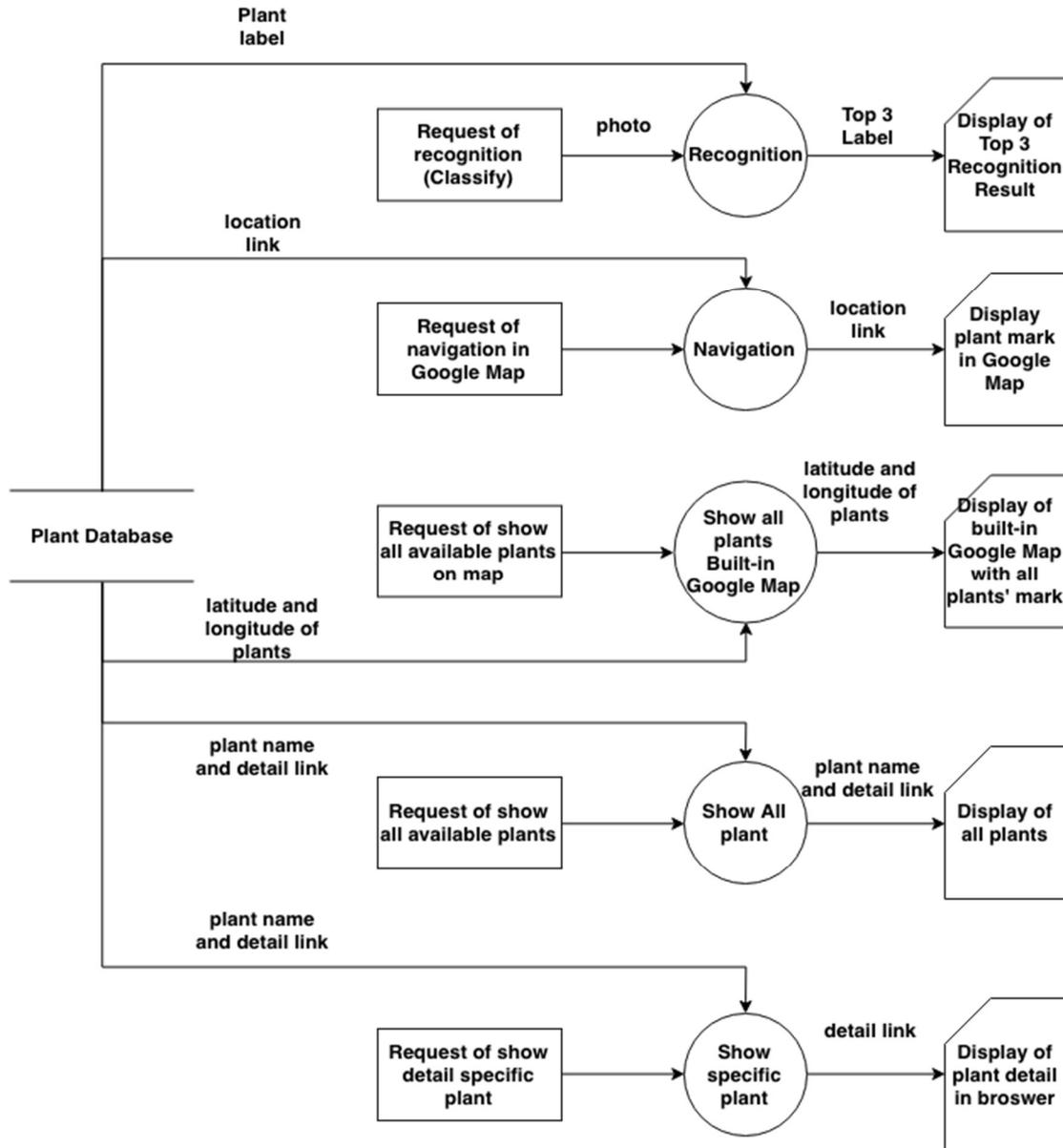


Figure 18 : Data Flow Diagram of mobile application

Five functions in the application share the same database to get their required data in order to implement their function.

1. The “Recognition” function takes the photo as an input, and the model will indicate the top three similar results to the result, only if the score of prediction is greater or equal than 10%.
2. The “Navigation” function takes no input from user, as user click the marking in the built-in Google Map and application will direct users to the Google Map application in the device to do the navigation of the specific plant.
3. The “Show all plants in built-in Google Map” takes no input from user, and get the latitudes and longitudes information of plants, marking their position in the map. The markers are clickable, which means user can access further function in in the Google Map app if they click the mark of the plant.
4. The “Show all plants” function take no input from user and get all the name in Chinese and English of the plants and detail link (collected from CUHK Shiu-Ying Hu Herbarium), and show all plants as CardView¹⁰ to the users.
5. The “Show specific plant” function take no input from user, and it is trigger when the user click the detail button in the activity¹¹ of recognition result or in the activity of show all plants.

¹⁰ Terminology used in Android Development Document, describing a component showing information, <https://developer.android.com/reference/android/support/v7/widget/CardView>

¹¹ Terminology used in Android Development Document, describing a page or a building block in android app, <https://developer.android.com/guide/components/activities/>

4.2.3. Structural Diagram



Figure 19 : Table Plant

The application is applied with a simple database with the above table. The code is an unique identifier of the plant, taking the first letter of the scientific name of plant. If two plants have the same first letter of the scientific name, for example “Actaea pachypoda” and “Acer palmatum”, we will add numerical index behind the code according to the time of data collection, like AP1 and AP2.

The name of the plant mainly follows the information provided by the CUHK Shiu-Ying Hu Herbarium, as it is possible to have different naming convention in different place to the same plant.

The lat and lng are representing the latitude and the longitude of the plant we collected in CUHK, there is possible location error because we marked the location with our own mobile device at that moment.

The locationLink is generated by the lat and lng in Google Map, we need the link for requesting the intent of opening Google Map official application.

4.2.4. UMLs

We will try present the application by different UML diagrams, in order to conclude the usage in a more complete way.

4.2.4.1. Use Case Diagram

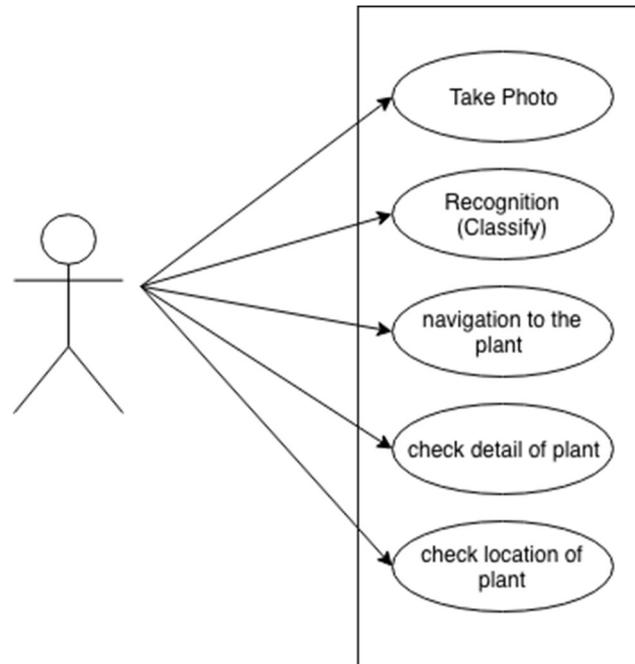


Figure 20 : Use Case Diagram of application

Users can use these five main functions in the mobile application. For “Take Photo” and “Recognition”, they have to implement in the sequential order to get the correct and expected output. They can be performed without internet connection and get the recognition result.

Checking location of plant do not need the internet connection as the marker will be shown in the built-in offline Google Map.

Besides, “navigation to the plant” and “check detail of plant” require the internet connection. If the users want to get the sophisticated information of the recognized plant, they can check it online by pressing the button provided.

4.2.4.1.1. Activity Diagram – Basic Recognition

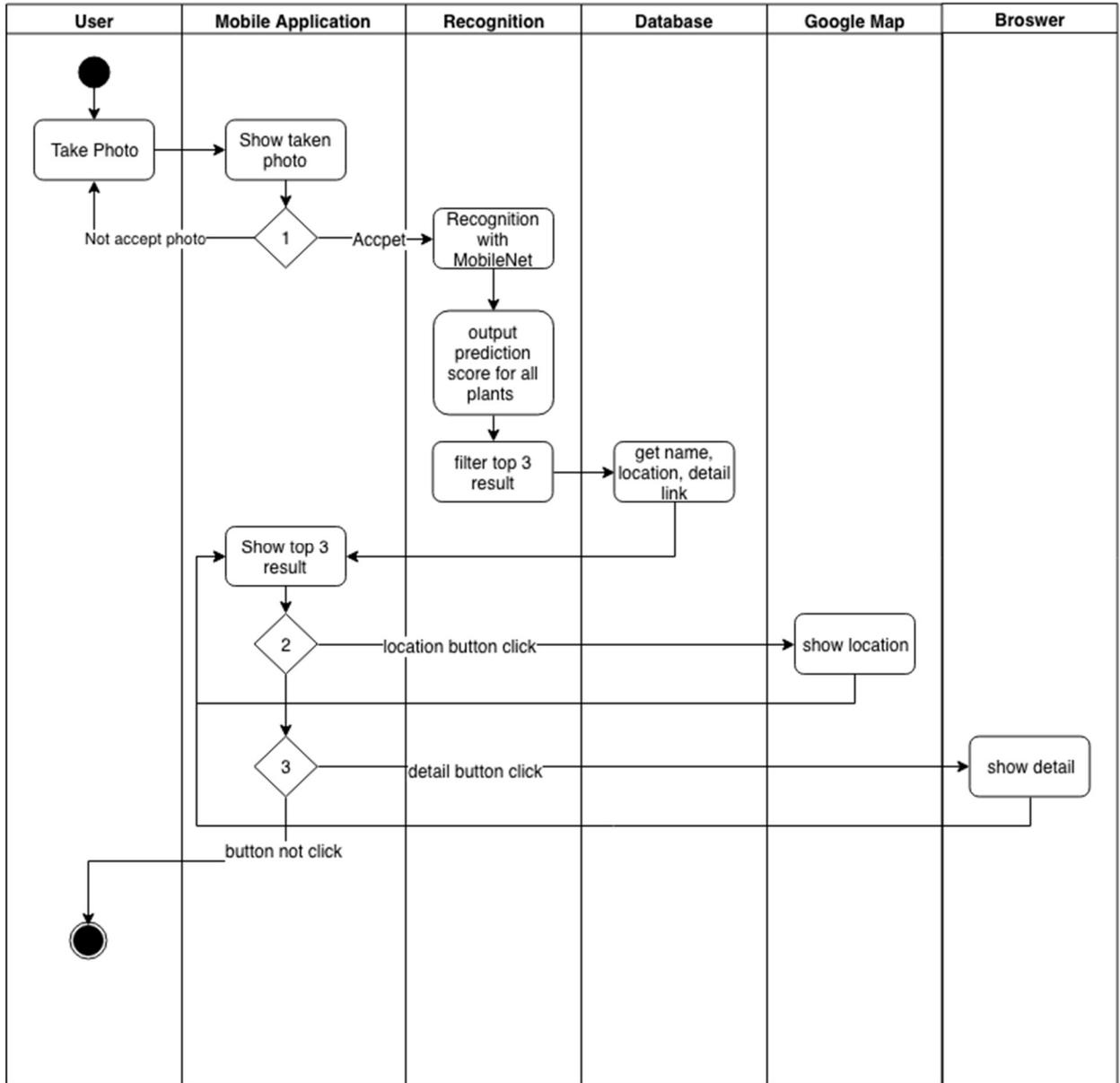


Figure 21 : Activity Diagram of Basic Recognition

Decision block 1: Users can retake the photo for recognition until they satisfy with it.

Decision block 2: It indicates the location button of the plant shown in result activity. If the users click the button, the mobile application will direct the user to Google Map application.

Decision block 3: It indicates the detail button of the plant shown in result activity. If the users click the button, the mobile application will direct the user to browser showing the page of herbarium of the specific plant.

This is the activity flow of a user when they normally use the application to the plant recognition. Up to the database, tasks will perform offline until user click the location button or the detail button of the plant. The activity diagram also shows the modularity of the mobile application, in which if we have to update to database or we would like to change the recognition modal, those amendments will not affect the application and cause bug in system.

4.2.4.1.2. Activity Diagram – Navigation

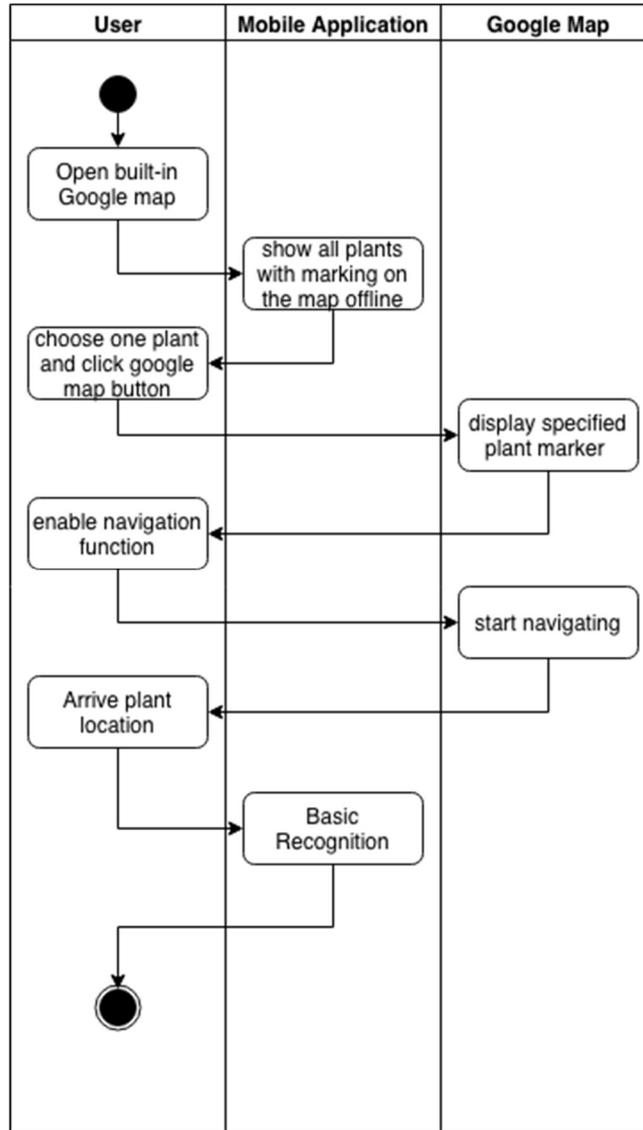


Figure 22 : Activity Diagram of Navigation

This is the activity flow of using the navigation function to reach the plant we record in CUHK. Users will see the location information and marking offline in the built-in Google Map. After user choose a kind of plant he is interested at, he can choose the marker,

and click the Google Map direct button to open Official Google Map application. Then user can follow the instruction of navigation to look for the plant. Once users finish navigation, he can go back the main application and try the photo recognition function.

4.2.4.2. *Functionality*

In summary, user can experience have several functions that the application brings along:

1. Get to know the species of plant by the leaf with photo recognition
2. Gain the detail of the recognized plant
3. Gain the location information of the recorded plant in CUHK
4. Locate and find the recorded plant in CUHK

We hope to deliver an instant plant recognition application without cornering the network issue to the users and help user to acquire the details of the plant if needed.

4.2.5. User Interface

In this semester, we have developed the mobile application of leaf recognition in Android platform, including the function mentioned before. The following will be some capture and procedure of using the mobile application.

4.2.5.1. Screen Capture



Figure 23 : Main page of Dr. Leaf



Figure 24: Main page after taking photo

Main page provides simple 2-step instruction for user, assisting them to do the recognition. We would like to keep the page as simple as possible, therefore only four buttons appear in the page

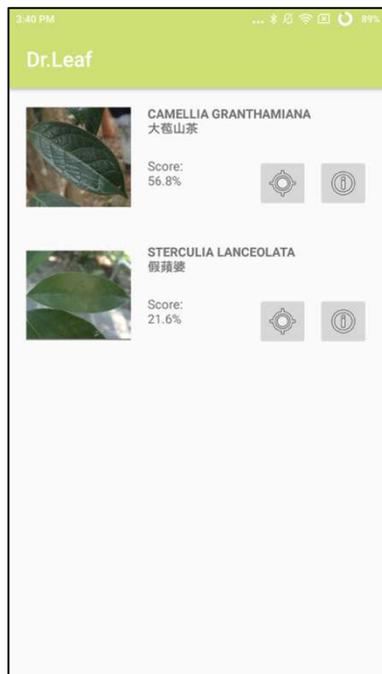


Figure 25: Top 3 Recognition Result



Figure 26: Browser Page of plant details



Figure 27: Location mark of the plant in Google Map

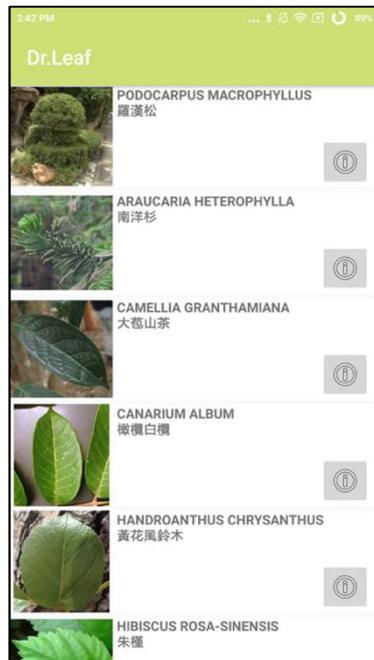


Figure 28 : View all plants in CardView

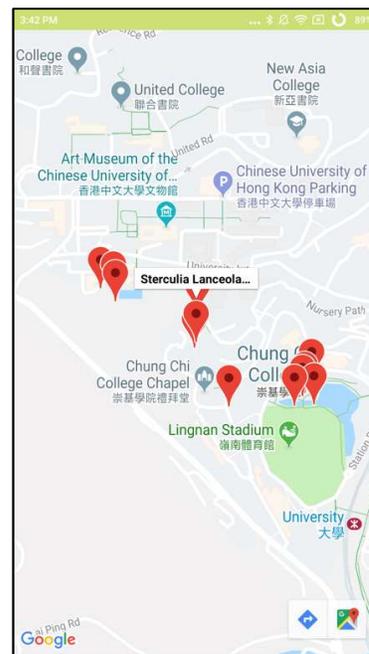


Figure 29: View all plants in Map

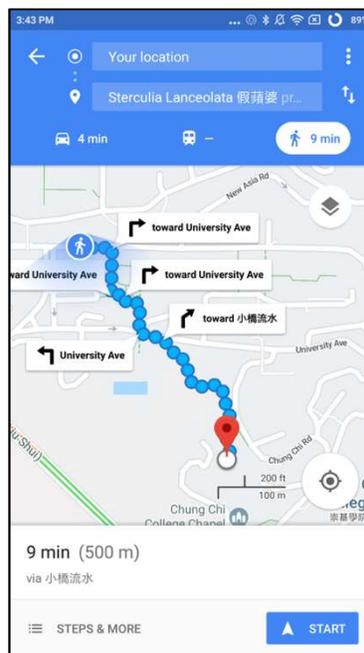


Figure 30 : Navigation in Google Map

4.2.6. Procedure of using Dr.Leaf

4.2.6.1. *Basic Recognition*

In this function, user can get the recognition result and choose the follow up option they want.

4.2.6.1.1. Step 1

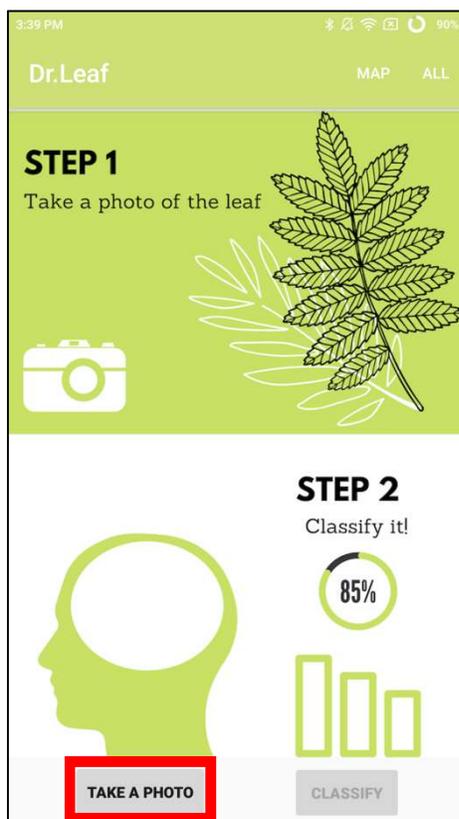


Figure 31 : Main Page of Dr.Leaf

Click the “take a photo button” and shoot the leaf of the plant.

4.2.6.1.2. Step 2



Figure 32 : Taken photo will be displayed

If you are satisfied with the photo, click “CLASSIFY” button to do recognition.

(Red box)

Otherwise you can take the photo again by clicking the “TAKE A PHOTO” Button

(Blue Box)

4.2.6.1.3. Step 3

Assume the “CLASSIFY” Button has been click with a satisfying leaf photo, Top 3 Result will be shown of the recognition if the score is greater than 10%.

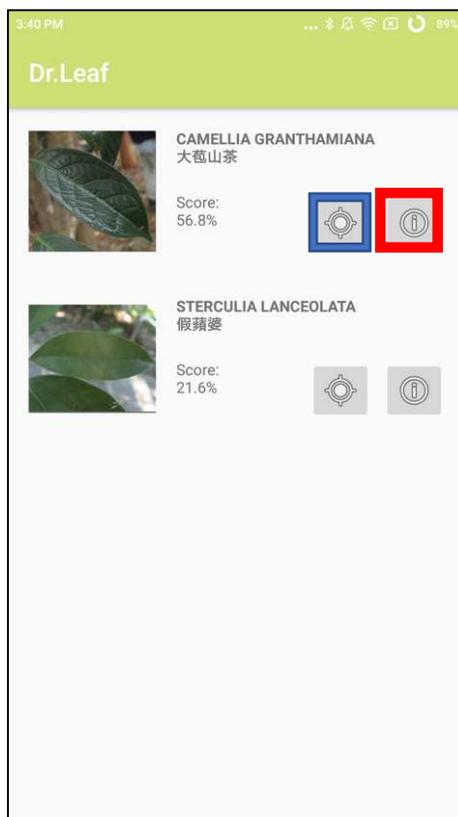


Figure 33 : Recognition Result

Users can choose to see the location of the plant by clicking the location button (Blue Box) or clicking the detail button (Red box) to check the detail of the plant provided by the herbarium.

4.2.6.1.3.1. Step 3.1

Assume user clicked the location button, Dr.Leaf will direct user to Google Map with the location of the selected plant.



Figure 34 : Google Map with dropped pin

4.2.6.1.3.2. Step 3.2

Assume user clicked the detail button, Dr.Leaf will direct user to the website of herbarium with the location of the selected plant.

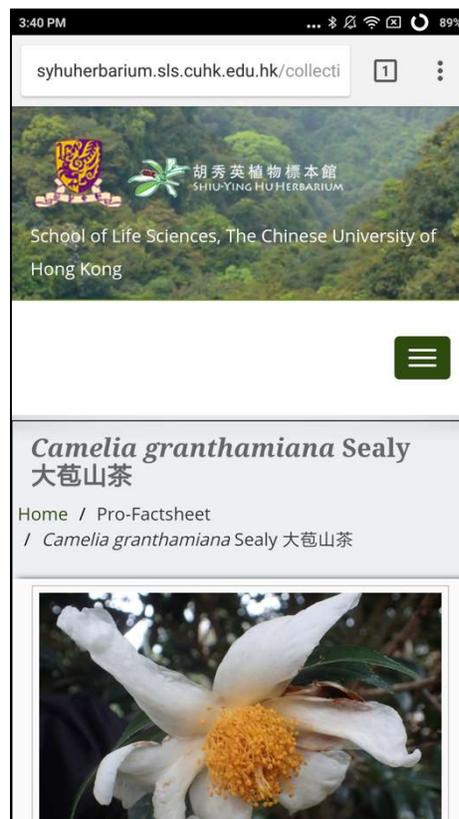


Figure 35 : Herbarium Website showing the Plant with tag "ca"

4.2.6.2. View All plants and navigation

User can check out all the recorded plant in application.

4.2.6.2.1. Step 1

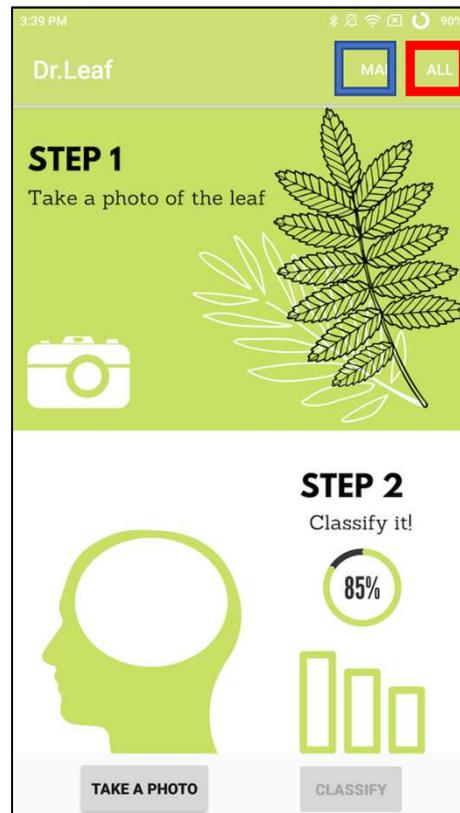


Figure 36 : Main Page of Dr. Leaf

There are two ways to view all the plants. User can choose to view all the plants in built-in Google Map (Blue Box) or view all the plants in list (Red Box)

4.2.6.2.1.1. Step 1.1

Assume user chose to view in list.

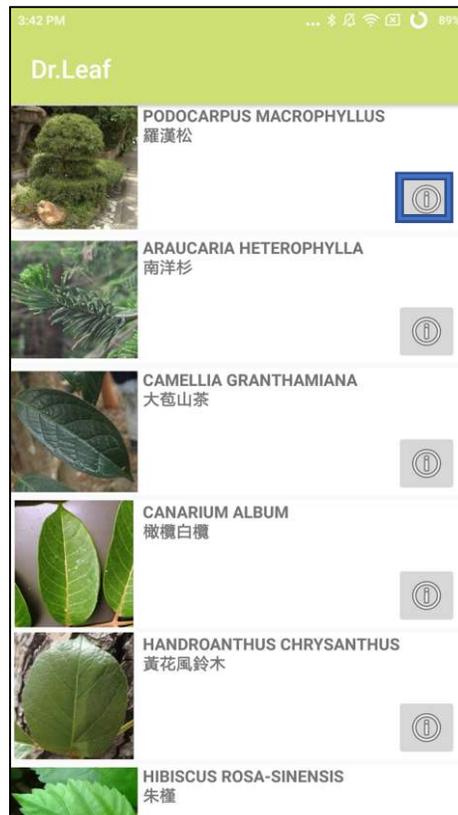


Figure 37 : Show all plants in list

User can click the detail button to get the detail of the plant (Blue box).

4.2.5.2.2. Step 2

Assume user chose to view in map.



Figure 38 : View all in Google Map

User can view all the plants as dropped pin with their name. If user click the navigation button (Blue box), Dr.Leaf will direct him to the Google Map and do so.

5. Testing and Evaluation

5.1. Database

Dr.Leaf has an offline database in the local device, which does not allow user to insert, update, or delete the data in the dataset. As we are using the API provide by Room¹², we can control every task or query to the database in the developer level and user has no right to access the database unless they do the recognition and query the plant indirectly.

5.2. Logical Statement

In the source code of the application, we seldom use logical statement to process data flowing in the application. There three two kind of data always flowing in the application.

1. Photo, pictorial data
2. Query result which is in the format of String
3. Intent¹³ request

The first two of data are directly being passed in the application, for example, passing the photo to the classifier and passing back the recognition in String. The third data is mainly concerned by the android system, regarding the process that has to be run for the next step of the application. It is also being passed in the application without logical decision. Therefore, testing on logical statement is not required in the application.

¹² Google develop and provide the Room as an abstraction layer of SQLite, we can easily handle the database by Room, <https://developer.android.com/training/data-storage/room/>

¹³ Terminology used in Android Development Document, describing an operation in android app, <https://developer.android.com/reference/android/content/Intent>

5.3. UI Testing

In the first prototype of the application, we discovered the application works perfectly fine in the emulator but fail to open in the actual mobile device of the pages that contain picture. After checking the coding error and potential mistake made by developer, we realized the picture, for instance, the icon of the plant and the instruction in the main page has a large size, consuming large amount of resource in an intent. When the memory is large enough and exceed the limitation of an intent¹⁴, therefore we reduce at least 40% of the size of the images and the application finally well fine in the mobile device.

¹⁴ It is mentioned API 23 is 517716 bytes (~ 0.5mb) from <https://www.neotechsoftware.com/blog/android-intent-size-limit>

6. AI Model

6.1. Background

After AlphaGo, a computer program designed to play Go utilizing neural network, beat the best Go player in the world, more and more people heard the name of neural network.

Neural Network is a popular machine learning model with great power in finding the relationship between input and output. By finding the relationship between decisions on playing chess and winning probability, AlphaGo showed the power of neural network on playing chess. By adding convolutional layers, neural networks can also do good job in image recognition. We believe that this tool should be handful to our tree recognition projects.

6.2. Problem Defining and Solution

To recognize trees, we are solving a classification indeed. For some elements drawn from different sets, we want to find out which set they belong to. Usually same set of elements share some common features. We can compare the difference of features of different elements to find out which sets the elements belong to. For example, we can differentiate apple trees from banana trees with the color of their fruits.

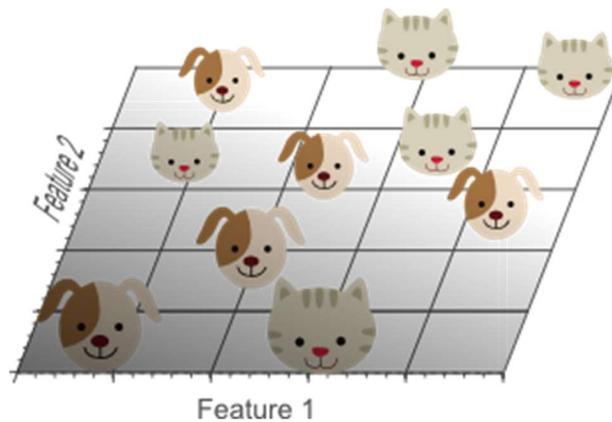


Figure 39: Classification between cats and dogs using 2 features.¹⁵

6.3. Neural Network

Traditionally, people select features by experience to solve classification problem. Eventually, the performance can be unstable. By simply feeding the features into the model, neural networks have the power on selecting useful features for classification. Therefore, we can avoid the unstable performance by make use of neural networks.

¹⁵ The image is obtained from <http://www.visiondummy.com/2014/04/curse-dimensionality-affect-classification/>

Literally, Neural Network is a network of processing units called neuron. Neuron is an activation function of the weighted sum of input. Inside the neural network, the weights are the parameter. We should train the model using optimization algorithm like gradient descent in order to get the suitable parameters.

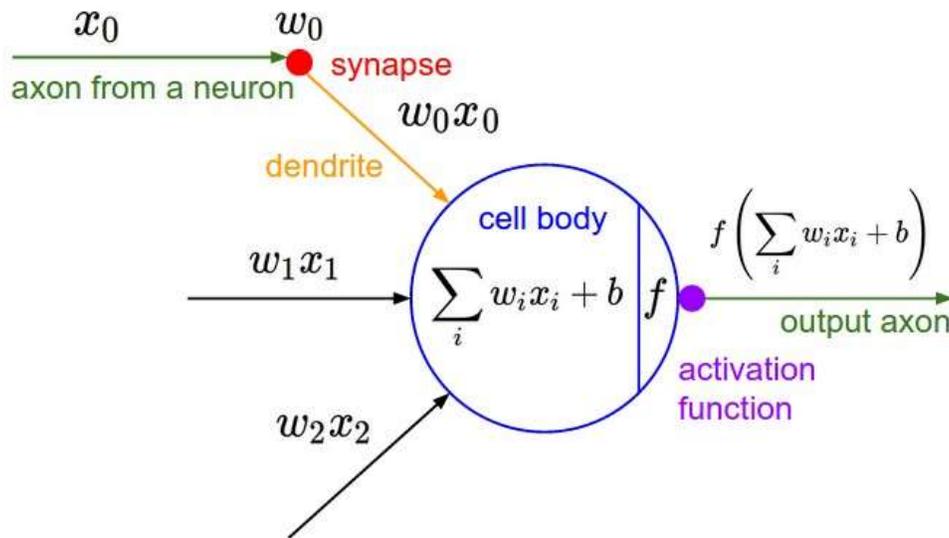


Figure 40: A neuron in neural network¹⁶

For different purpose, the neural network architectures can be different. The most commonly used ones are feed-forward network and recurrent network. For simplicity, people normally use feed-forward network. It has been mathematically proofed that standard multilayer feedforward networks can approximate any measurable function to any desired degree of accuracy.¹⁷ In reality, the performance of neural networks varies but we can we can overcome this problem with the increase in depth of neural network.¹⁸

¹⁶ The image is obtained from <https://stackoverflow.com/questions/38850538/single-neuron-in-neural-network-using-c>

¹⁷ The proof is in the article "Multilayer feedforward networks are universal approximators".

¹⁸ This is discussed in the article "Very deep convolutional networks for large-scale image recognition" and "Going deeper with convolutions".

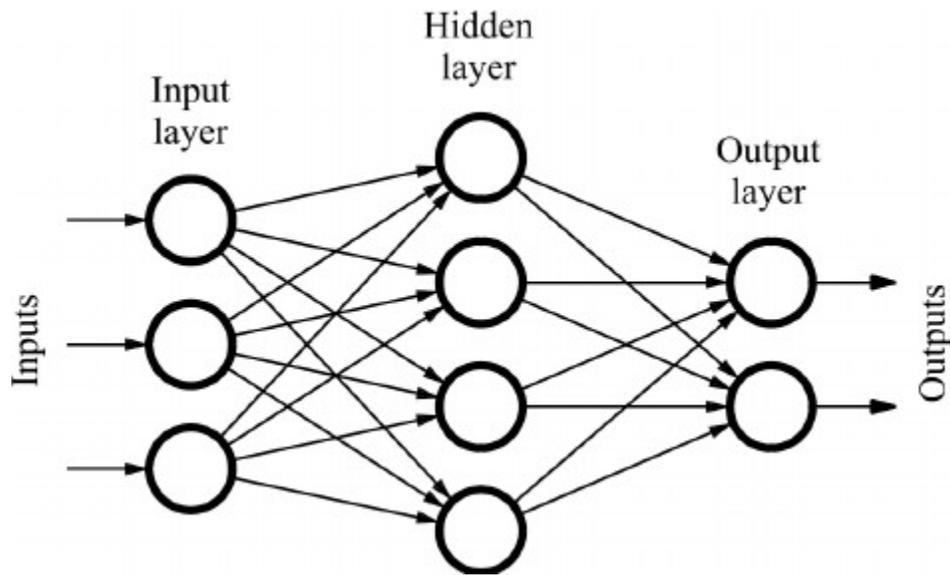


Figure 41: A feed forward neural network¹⁹

6.4. CNN

As mentioned before, the architecture of neural networks can vary depending on the problems and the needs.

Starting from 2012, the appearing of Alexnet, a Convolutional Neural Network (CNN), leads to a great success in image classification with the help of strong calculation power of modern graph processing unit. Using convolutional layer, Alexnet obtained top-1 accuracy of at least 60% on several ILSVRC dataset and became the champion of the ImageNet Large Scale Visual Recognition Challenge in 2012, which is a world-class competition.

CNN is a kind of neural network in which convolutional layers are added. In the convolutional layer, convolution is performed for the input and features can be

¹⁹ The image is obtained from https://www.researchgate.net/figure/Sample-of-a-feed-forward-neural-network_fig1_234055177

extracted from the images easily. To get useful features, we have to tune the parameters in the convolution kernel by training the model.

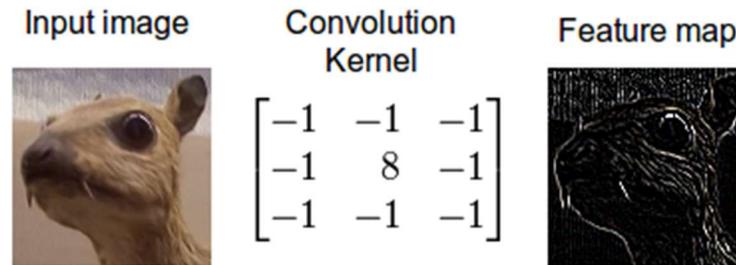


Figure 42: Output image(right) was obtained by performing convolution operation using the convolution kernel(middle) on the input image(left).²⁰

In early years, neural network is seen as a black box because it is difficult to understand what is happening inside the network. Recent years, researcher developed some visualization approaches to investigate the properties of neural networks. In figure 43, features extracted from different layers are shown. One can easily observed that, patterns extracted from the layers that is near the input are relatively simple, such as line and dot. And patterns extracted from the layers that is near the output are more complicated. We can inference that CNN first extracts different features, then combines the simple features to complicated features and finally using the abstract features to classify objects.

²⁰ The image is obtained from <https://developer.nvidia.com/discover/convolution>

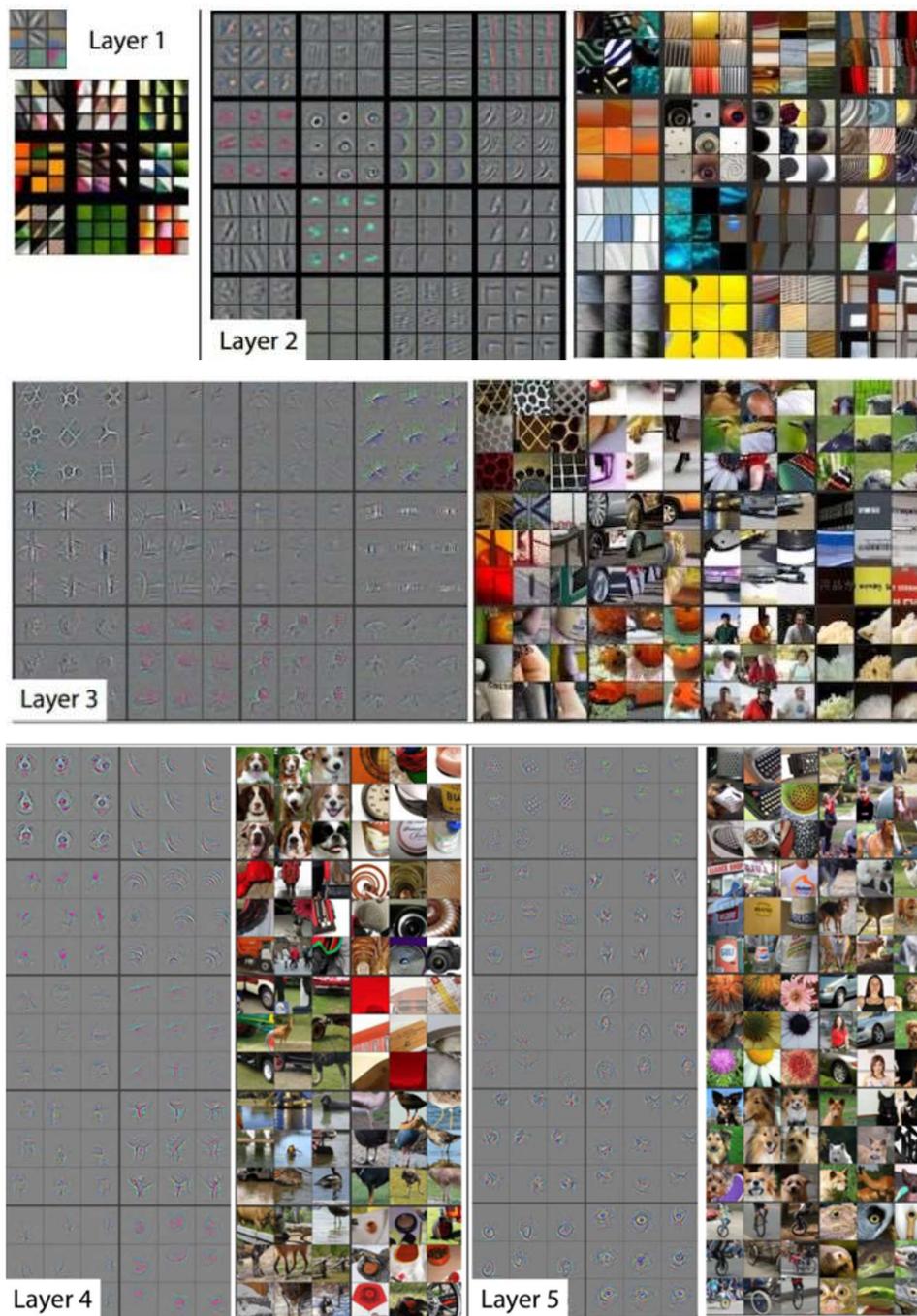


Figure 43: Visualization using deconvolutional approach. The figure shows features extracted in different layers.²¹

²¹ The image is obtained from Visualizing and Understanding Convolutional Networks

6.4.1. Concern

CNN is a powerful tool in classifying objects so we have no reason not choosing it. Since the success of Alexnet, CNN developed rapidly and a large amount of CNN variant were invented. Before choosing the suitable one, we should examine the requirements of our project.

6.4.2. Portability

The expected result of this project is to develop an offline mobile application. This means all the calculations would happen in the mobile device. The model should be small enough so that it can be stored in the device.

To persuade the customers to use the application, we should consider their needs. The size of the product should be reasonable. If the application occupies a large portion of storage, the user will not be willing to install it.

6.4.3. Speed

As an application, slow processing could downgrade the user experience seriously. Slow applications are impractical and could probably lose the market. The reasonable processing time should be within a few seconds. We should make a balance between speed and other aspects.

6.4.4. Develop Period

As a starting point, the application can only classify ten species of trees in this phase. In the future, we expect we would increase the number of trees the application can recognize. If the training time is too long, it is difficult to do testing and update the application immediately.

6.5. MobileNets

After considering the above concern, MobileNets seems to be a suitable candidate to our project. MobileNets is a CNN architecture developed by Google and features in small size and high speed. The more exciting thing is that the decrease in model size and the increase in running speed would not cause a huge decline in accuracy.

Table 1. MobileNet Body Architecture

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
5× Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

Figure 44: MobileNet architecture²²

²² The table of the architecture is obtained from the article “Mobilenets: Efficient convolutional neural networks for mobile vision applications”.

6.5.1. Depth-wise Separable Convolution

The difference between MobileNet and normal CNN is the adding of depth-wise separable convolution. Depth-wise Separable Convolution is a factorized convolution that has extremely small size with similar accuracy when comparing with some popular models.

Traditional convolution layer has the function of extracting features and combining features. Depth-wise Separable Convolution separates them and consists of two parts: the depth-wise convolutional layer which is responsible for the feature extraction, and the pointwise convolutional layer which is responsible for combining features.

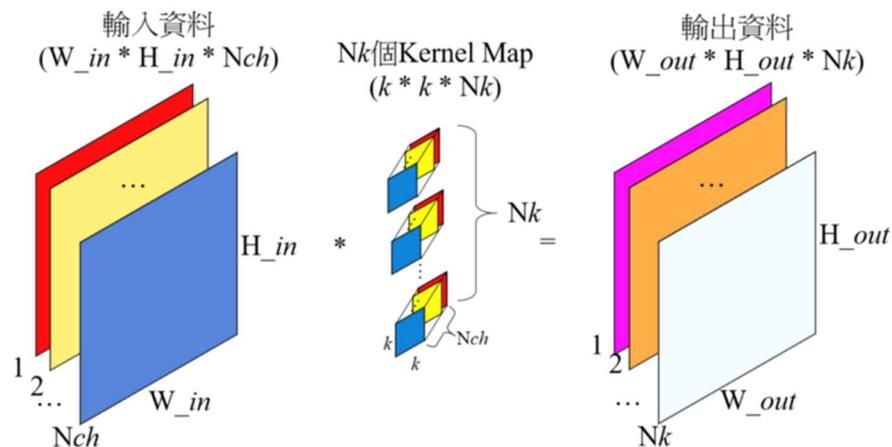


Figure 45: Normal convolutional layer(The symbol '*' is the convolution operation)²³

²³ Picture source:

<https://medium.com/@chih.sheng.huang821/%E6%B7%B1%E5%BA%A6%E5%AD%B8%E7%BF%92-mobilenet-depthwise-separable-convolution-f1ed016b3467>

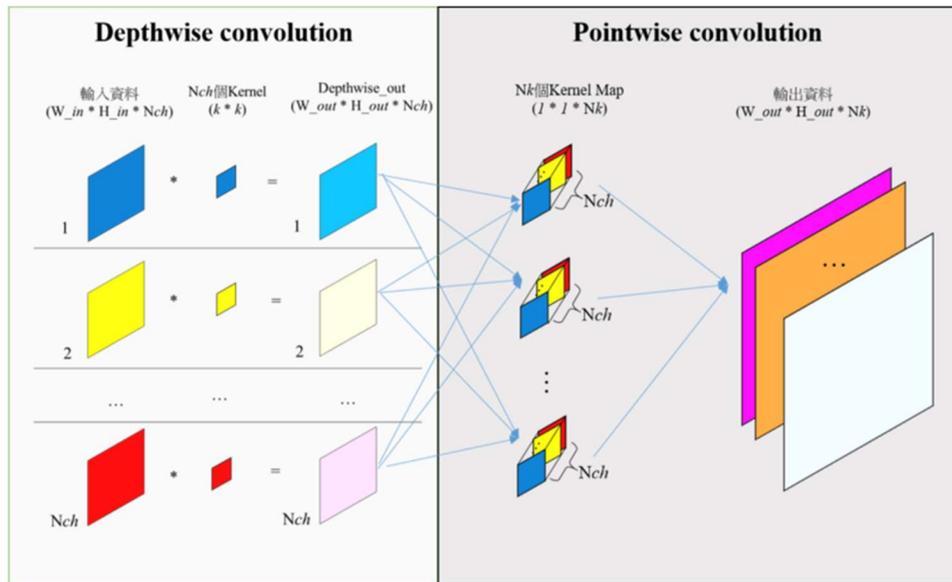


Figure 46: Depth-wise Separable Convolution(The symbol ‘*’ is the convolution operation)²⁴

6.5.2. Hyperparameter

Beside the special design of convolutional layer, MobileNet provides two hyperparameters, which are the resolution multiplier and the width multiplier, for controlling model size and computation speed.

Resolution multiplier control the input image resolution. Lower resolution of input image means less pixels, the amount of calculation can be reduced and therefore the training time and testing time can be reduced.

Width multiplier control the size of convolutional kernels. A small width multiplier can reduce the size of the model. Also, the decrease in number of parameters means less calculation and the speed would increase.

²⁴ picture source:

<https://medium.com/@chih.sheng.huang821/%E6%B7%B1%E5%BA%A6%E5%AD%B8%E7%BF%92-mobilenet-depthwise-separable-convolution-f1ed016b3467>

However, reduction in resolution would lead to the loss of features. Therefore, the model might not be able to extract enough features and the accuracy would be lower. And smaller kernel size would downgrade the ability of the model.

There is no magic in this two hyperparameter but MobileNet gives us a choice to make a trade-off between accuracy, and size and speed. Figure 47 and 48 shows the trade-off between accuracy, speed and size.

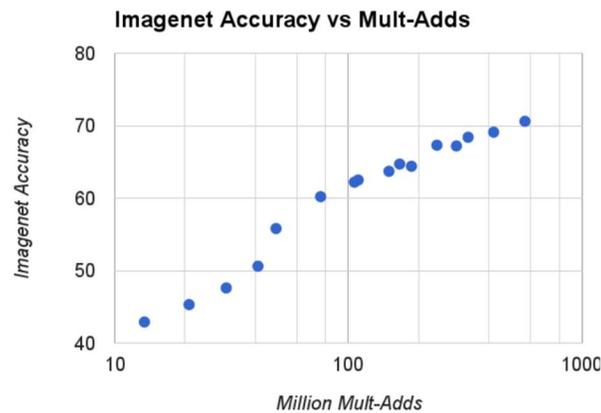


Figure 47: Trade-off between accuracy and speed in terms of number of multi-add operation²⁵

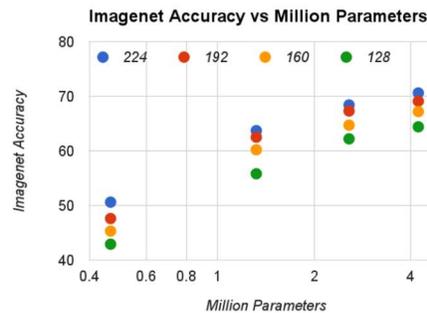


Figure 48: Trade-off between accuracy and size in terms of number of parameters²⁶

²⁵ The figure is obtained from the article “Mobilenets: Efficient convolutional neural networks for mobile vision applications”.

²⁶ The figure is obtained from the article “Mobilenets: Efficient convolutional neural networks for mobile vision applications”.

6.5.3. Performance

From figure 49, if a 3*3 convolution kernel is used, the depth-wise separable convolution uses about 9 times less calculation than standard convolutions since output length N in hidden layers are small and can be omitted.²⁷

From figure 50, with a special design, the number of parameters of depth-wise separable convolution is less than other popular CNN architectures. Although depth-wise separable convolution and standard convolution is not equivalent, depth-wise separable convolution gives similar accuracy in practice when comparing with other popular CNN architecture.

$$\frac{D_K \cdot D_K \cdot M \cdot D_F \cdot D_F + M \cdot N \cdot D_F \cdot D_F}{D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F} = \frac{1}{N} + \frac{1}{D_K^2}$$

Figure 49: Ratio of calculation amount between depth-wise separable convolution and standard convolution. Numerator: Amount of calculation of depth-wise separable convolution. Denominator: Amount of calculation of standard convolution. D_K : Length of convolution kernel. M : Number of channels of input image. N : Number of channels of output image. D_F : Length of input image. ²⁸

²⁷ The discussion of the amount of calculation is in the article “Mobilenets: Efficient convolutional neural networks for mobile vision applications”.

²⁸ The formula is obtained from the article “Mobilenets: Efficient convolutional neural networks for mobile vision applications”.

Model	ImageNet Accuracy	Million Mult-Adds	Million Parameters
1.0 MobileNet-224	70.6%	569	4.2
GoogleNet	69.8%	1550	6.8
VGG 16	71.5%	15300	138

Figure 50: Comparison on performance between standard MobileNet and some popular CNN architectures. First column shows the accuracy of different architecture. Second column compares the speed in terms of million multi-adds. Third column compares the size of differ.²⁹

6.6. Workflow

6.6.1. Pretrained Model and Transfer Learning

Google pretrained a MobileNet model using Imagenet dataset, in which there are 1000 different classes. By retrain the last layer of the pretrained model using our own dataset, we can get benefit from transfer learning. Both training speed and accuracy would be increase.

Besides, recognizing trees using CNN requires a large amount of labelled image data. When using transfer learning, the number of image data could be smaller while keeping similar accuracy.

The purpose of our project is to differentiate the species of trees in CUHK. After doing research, we cannot find large amount of image data about trees in CUHK. Utilizing the pretrained model can reduce our effort on collecting the image data and labelling. Therefore, we would like to make use of the pretrained MobileNet model in our tree recognizing application.

²⁹ The table is obtained from the article “Mobilenets: Efficient convolutional neural networks for mobile vision applications”.

6.6.2. Data

When using CNN, we should prepare a large amount of image data for model training. There are a lot of features of trees, like leaves, fruit, flower and so on. As a starting point, we want to use only one feature to classify trees. Intuitively, people think of leaves when thinking of trees. Also, leaves can be easily obtained and observed so we decide to prepare a dataset of leaves images of different species for model training.

6.7. Experiment

To classify the species of the trees, users should take a picture of leaves. Initially, we expect the user to take a photo of a single clear leaf. In some cases, the trees are tall and people cannot easily take a high-quality photo of single leaf. Therefore, we might have to prepare a model which can recognize trees through photos of multiple leaves very well.

To allow the model to recognize trees using photos of multiple leaves, we should train the model with multiple leaves images. However, it may lead to the downgrade in accuracy of model. Therefore, we prepared a dataset of single leaf images, a dataset of multiple leaves image and a dataset of both image data. We want to compare the quality of models trained by different dataset so that we can decide which model we should use and the requirement of the input image to the user.

To increase the training data and avoid overfitting problem, we also performed augmentation on images. Therefore, we have 6 sets of datasets, including: original single leaf dataset, original multiple leaves dataset, original mixed leaves dataset, augmented single leaf dataset, augmented multiple leaves dataset and augmented mixed leaves dataset.

	Original	Augmented
Single leaf	95.6%	96.8%
Multiple leaves	97.9%	98.5%
Mixed leaves	97.3%	97.2%

Table 7: Accuracy of pretrained MobileNet re-trained by different datasets

After training the model with different datasets and performing validation, we obtained the validation score on the different models. Surprisingly, it is very high. If

there is no other unobservable error like overfitting on dataset we prepared, this tree recognizing application can be put into application.

7. Limitation and difficulties

7.1. Data Collection

To obtain a large amount of labelled data, finding existed datasets and crawling in the internet are the ideal methods. However, since we only want to investigate the trees in CUHK, we cannot find the tree dataset we needed. For crawling images in the internet, we can obtain labelled images by typing keyword on Google Image. However, we cannot get high-quality image data without filtering and the label on the images might be incorrect. Without professional knowledge on trees, we cannot identify the trees easily. By feeding the images labelled incorrectly, we could possibly obtain bad results.

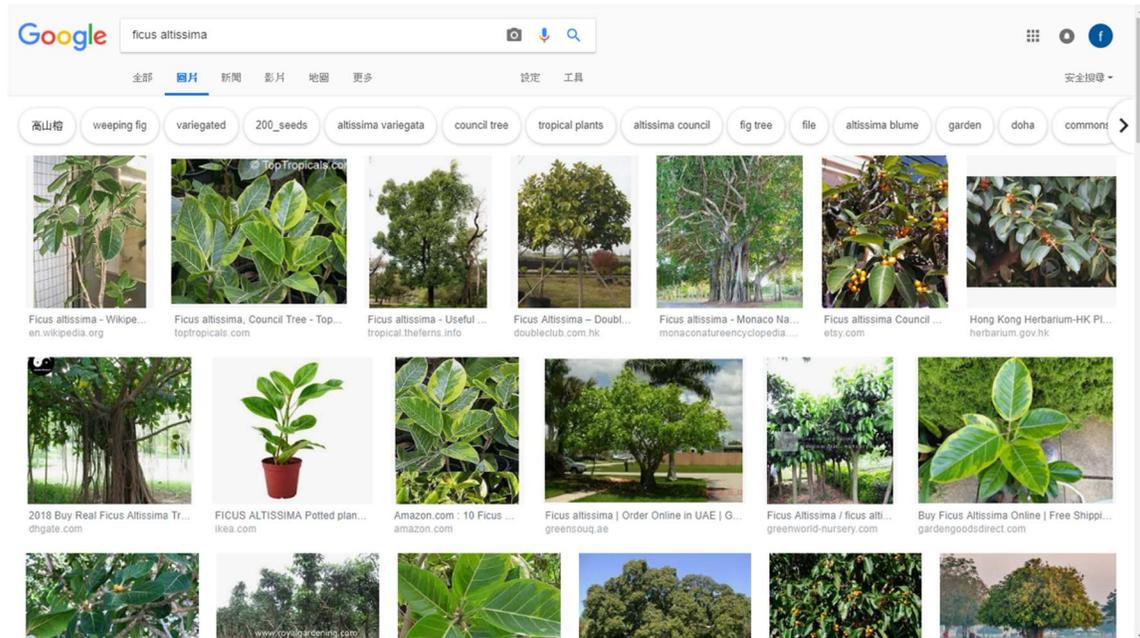


Figure 51: Search results of *Ficus altissima* on Google Image

Some leaves are too high, we cannot pick and take photo with it, therefore it will be the limitation of the variety of plant in dataset if we only use the self-taking photo.

7.2. Knowledge about plants

To develop an advanced tree recognizing application, we should not only focus on the AI technology but also the characteristics of trees.

Before the whole development process, we have make an inquiry to Dr. David Lau, the Curator of Shiu-Ying Hu Herbarium CUHK. Unexpectedly, relying on leaves only is not enough to differentiate the species of trees in professional point of view. For scientific study and forensic evidence, more that 15 features of trees would be used normally. Similar to human being, the features of one species and those of their relatives are very similar. With only a few features, the classification could possibly incorrect.



Figure 52: Flowers of *Bauhinia variegata*(left) and Hong Kong Orchid Tree(right).³⁰

Besides, even for the same species of tree, their features could be not the same if they grow in different environment. With different intensity of sun light or different climate change, their features could vary. Since our data is collected in CUHK only, there could be “overfitting” problem and the application could not be able to recognize the same kind of tree growing in USA.

³⁰ Picture source: <https://zh.wikipedia.org/wiki/%E6%B4%8B%E7%B4%AB%E8%8D%8A>,
<https://zh.wikipedia.org/wiki/%E5%AE%AE%E7%B2%89%E7%BE%8A%E8%B9%84%E7%94%B2>

8. Conclusion

8.1. Mobile Application

We will conclude the application with software engineering principles and make evaluate for the reference in future development.

8.1.1. Modularity

Recall the architecture design Diagram,

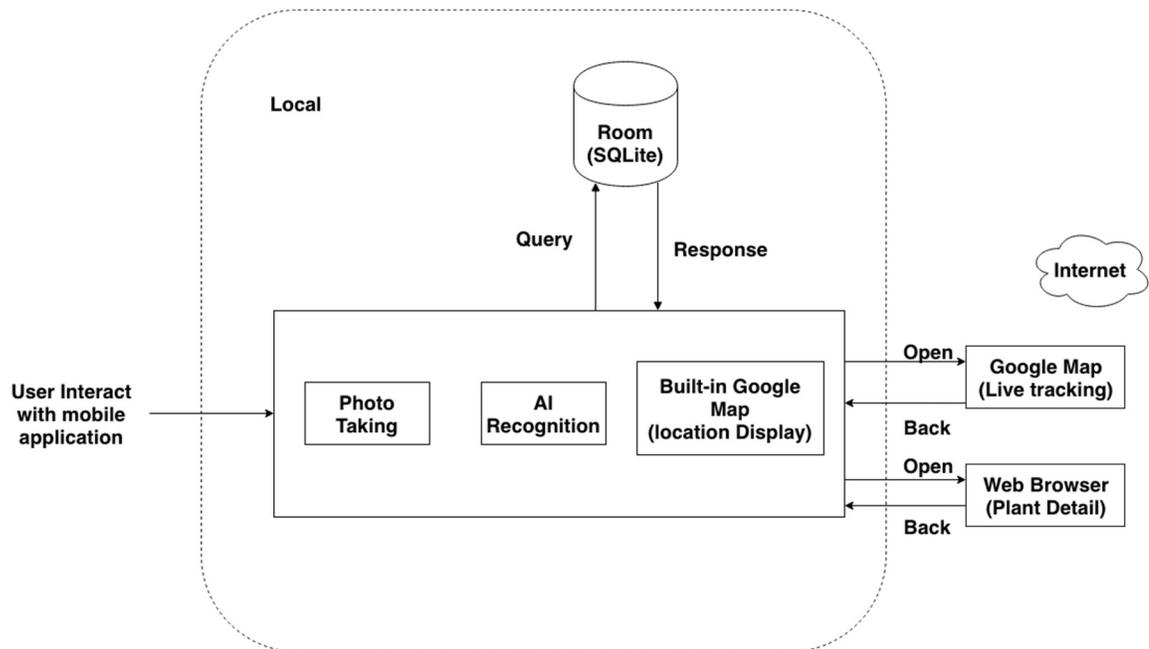


Figure 53: Architecture Design Diagram

Dr. Leaf composed by different modules,

- (i) Photo taking in native camera application in the mobile
- (ii) AI Recognition with TensorFlow Lite
- (iii) Room Database
- (iv) Inter access operation (Live tracking and Plant Detail)

With the separation of different modules, it is easier for developers to perform debugging and locate the error in different modules. Moreover, if there is application update in the specific module, for example the update in Google Map, it may require different data input as the API might be different from the older version of Google Map, developers can still stick on the way of building application and give the correct information to the module without redesign the mobile application to suit the update. The modularity in Dr.Leaf will be benefit for development in the future and make the application in the road of sustainable development.

8.1.2. Evolvability

The mobile application is developed with good evolvability in terms of database and source file system in Android Studio.

8.1.2.1. Database

For every time developer change the structure of database, a new schema will be generated and the older one will be saved for reference in the future. Hence, if there is any amendment of the database, future developer will be able to check the past record for reference and update the desired structure they want.

```
1 {
2   "formatVersion": 1,
3   "database": {
4     "version": 3,
5     "identityHash": "ff414d2037da9108e2fa40b6be767a95",
6     "entities": [
7       {
8         "tableName": "plant_table",
9         "createSql": "CREATE TABLE IF NOT EXISTS `${TABLE_NAME}` (`code` TEXT NOT NULL, `nameEng` TEXT, `nameChi` TEXT, `detailLink`
10        "fields": [
11          {
12            "fieldPath": "plantCode",
13            "columnName": "code",
14            "affinity": "TEXT",
15            "notNull": true
16          },
17          {
18            "fieldPath": "nameEng",
19            "columnName": "nameEng",
20            "affinity": "TEXT",
21            "notNull": false
22          },
23          {
24            "fieldPath": "nameChi",
25            "columnName": "nameChi",
26            "affinity": "TEXT",
27            "notNull": false
28          },
29          {
30            "fieldPath": "detailLink",
31            "columnName": "detailLink",
32            "affinity": "TEXT",
33            "notNull": false
34          },
35          {
36            "fieldPath": "locationLink",
37            "columnName": "locationLink",
38            "affinity": "TEXT",
39            "notNull": false
40          },
41        ]
42      }
43    ]
44  }
45 }
```

Figure 54: Json file of schemas of database in current version

8.1.2.2. Resource file system in Android Studio

The application built upon a well-structured resource file system, organizing every component in the application such as String will be used in showing instruction to user, color that will be presented of the button and or in the CardView. Once the developer wants to update the String, for instance showing on the button, he can change it inside the resources control file instead of find the single button among the massive codes. Also, it is beneficial for developer to add and control multiple language with xml tag in the file.

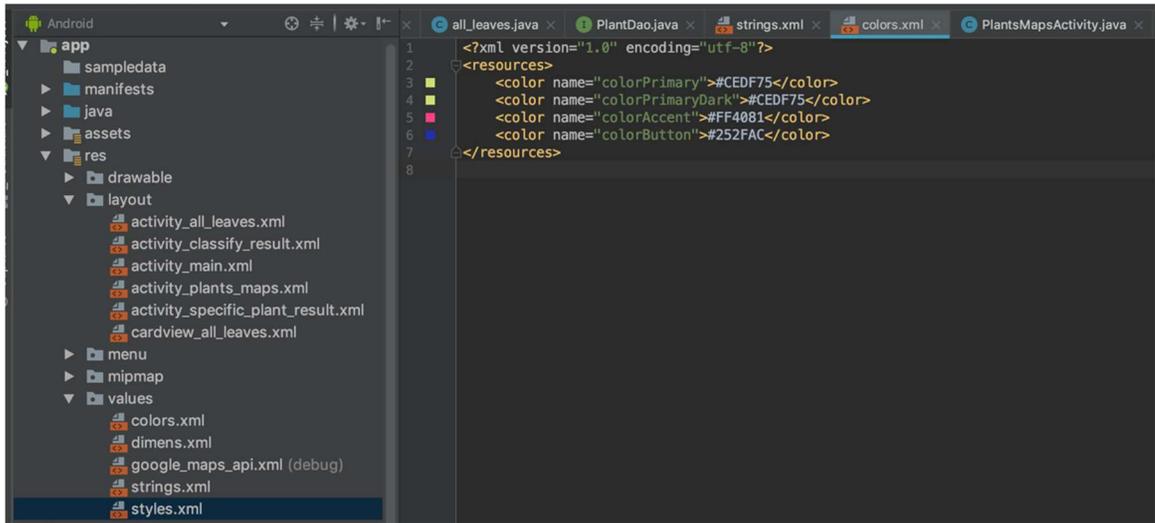


Figure 55 : Example of showing file color control xml in the application

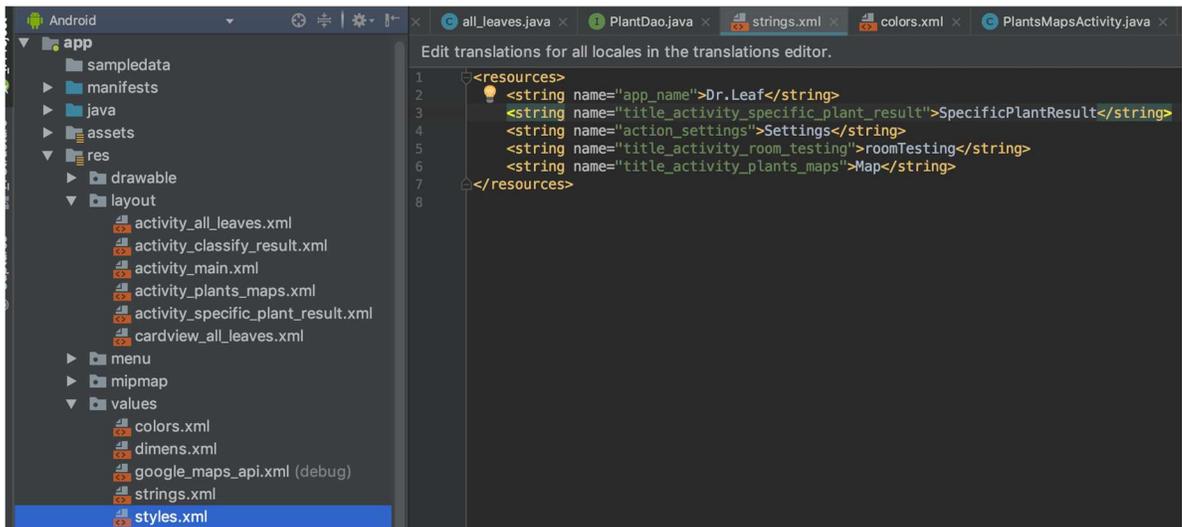


Figure 56 : Example of showing String control in the application

▼ app	25 Nov 2018 at 4:18 PM	--	Folder
app.iml	25 Nov 2018 at 4:18 PM	17 KB	Document
▶ build	Today at 8:09 PM	--	Folder
build.gradle	22 Nov 2018 at 10:50 PM	2 KB	Document
▶ libs	1 Nov 2018 at 3:54 PM	--	Folder
proguard-rules.pro	1 Nov 2018 at 3:54 PM	751 bytes	Document
▶ sampledata	21 Nov 2018 at 12:04 AM	--	Folder
▼ schemas	20 Nov 2018 at 5:57 PM	--	Folder
▼ com.leafrecognition.leafrecognition.db.PlantRoomDatabase	24 Nov 2018 at 5:09 PM	--	Folder
1.json	21 Nov 2018 at 5:33 PM	2 KB	JSON
2.json	24 Nov 2018 at 5:07 PM	2 KB	JSON
3.json	24 Nov 2018 at 5:09 PM	2 KB	JSON
▼ src	24 Nov 2018 at 4:19 PM	--	Folder
▶ androidTest	1 Nov 2018 at 3:54 PM	--	Folder
▶ debug	24 Nov 2018 at 4:19 PM	--	Folder
▼ main	25 Nov 2018 at 4:01 PM	--	Folder
AndroidManifest.xml	25 Nov 2018 at 4:01 PM	3 KB	XML Document
▶ assets	13 Nov 2018 at 4:47 PM	--	Folder
▶ java	19 Nov 2018 at 2:06 PM	--	Folder
▼ res	20 Nov 2018 at 11:39 PM	--	Folder
▶ db	19 Nov 2018 at 2:05 PM	--	Folder
▶ drawable	25 Nov 2018 at 4:20 PM	--	Folder
▶ drawable-v24	1 Nov 2018 at 3:54 PM	--	Folder
▶ layout	Today at 7:55 PM	--	Folder
▶ menu	24 Nov 2018 at 4:07 PM	--	Folder
▶ mipmap-anydpi-v26	1 Nov 2018 at 3:54 PM	--	Folder
▶ mipmap-hdpi	1 Nov 2018 at 3:54 PM	--	Folder
▶ mipmap-mdpi	1 Nov 2018 at 3:54 PM	--	Folder
▶ mipmap-xhdpi	1 Nov 2018 at 3:54 PM	--	Folder
▶ mipmap-xxhdpi	1 Nov 2018 at 3:54 PM	--	Folder
▶ mipmap-xxxhdpi	1 Nov 2018 at 3:54 PM	--	Folder
▼ values	Today at 7:57 PM	--	Folder
colors.xml	Yesterday at 3:34 PM	254 bytes	XML Document
dimens.xml	12 Nov 2018 at 4:20 PM	157 bytes	XML Document
strings.xml	Today at 7:57 PM	335 bytes	XML Document
styles.xml	19 Nov 2018 at 1:48 PM	1 KB	XML Document
▶ release	24 Nov 2018 at 4:19 PM	--	Folder
▶ test	1 Nov 2018 at 3:54 PM	--	Folder
build.gradle	19 Nov 2018 at 12:01 AM	616 bytes	Document
gradle	1 Nov 2018 at 3:54 PM	--	Folder
gradle.properties	1 Nov 2018 at 3:54 PM	726 bytes	Java Properties
gradlew	1 Nov 2018 at 3:54 PM	5 KB	Unix executable
gradlew.bat	1 Nov 2018 at 3:54 PM	2 KB	Batch File

Figure 57 : Well-organized file system in Android Studio project

Both of the points make the application developing in a sustainable way and the maintainability will also be enhanced due the organization of the source code files.

8.1.3. User Friendliness

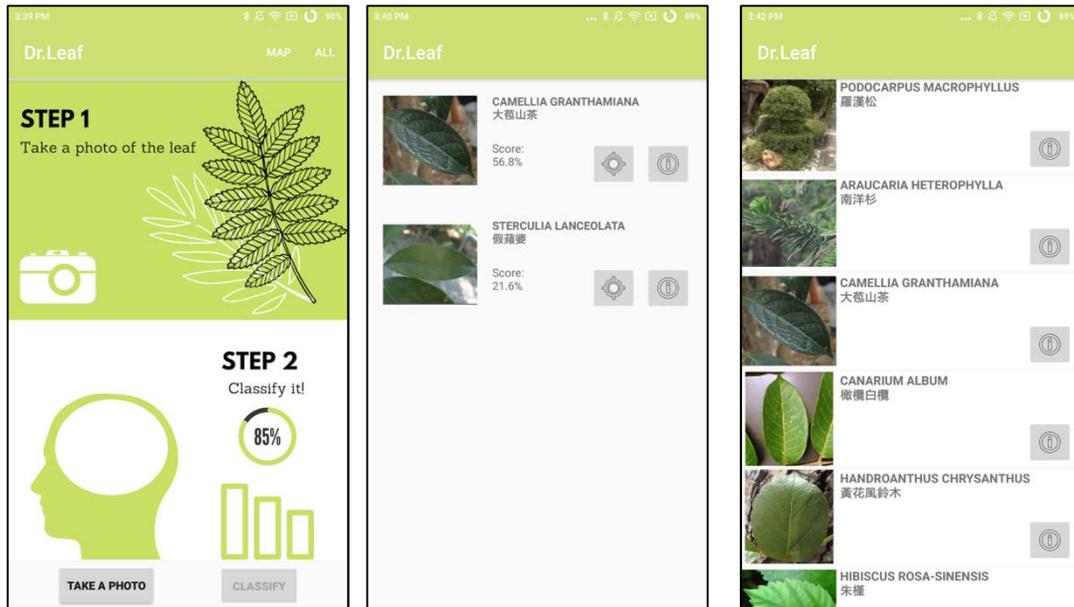


Figure 58 : Application captures

Although the application has developed with clear instruction for user to use and classify the plants, most the option and activity in the application rely on button click, which might not bring users some modern and intriguing experience that differ from other applications we found. Also, it is similar to the WIMP user interface style, which has been advocated some time ago. We might consider and redesign some user interface, for example animation, to deliver a more revolution experience to the user in the future.

8.2. AI Model

This application makes use of the state-of-art convolutional neural network to classifying species of trees. For concern on balance of size, processing speed and accuracy, we decided to use the MobileNet CNN architecture. To reduce our effort on collecting labelled data, we used the pretrained MobileNet and enjoy the benefits of transfer learning.

9. Future Development

9.1. Mobile Application

9.1.1. Function

9.1.1.1. Recognition History

We would like to include the recognition history once the application has been installed in a device. Normally, application always have a user authentication system and integrated the history function with respect to each user. Some applications even support linking users' other social media account, for example Facebook or google account to reduce the step registering new account for the system.

However, we would like to deliver a service to user who take the minimized step of setting up and registering and enjoy the fast recognition and check the history back of what plants have already been recognized.

Moreover, we might try to add a bookmark system to the recognition history so that users are able the save the plants that is important to them or they are interested at among the history.

9.1.1.2. Advanced view all plants function

Currently, we have developed the first version of the application with complete function as a platform to recognize plants by leaf. In the view all page, we might try to improve the user experience by adding the searching function, sorting function or filtering function as the size of database will increase in the future.

9.1.1.3. Offline data for the plants

The mobile application provides fast recognition for the plant, but if the users want to acquire the details of the recognized plants, internet access is still required to log in the herbarium website. Offline data would be useful for users when they lack internet connection.

9.1.2. UI/UX Design

We have provided a simple user interface, in which user can easily follow the instruction to do the recognition on the leaf of the plants. However, the user interface may not be attractive or interactive enough for user as it is mainly depending on clicking button like operating the application in the computer, the WIMP³¹ style. In the future development, we might try to add more animation, follow the material design principle, for instance, expressive, diverse and reality-based³².

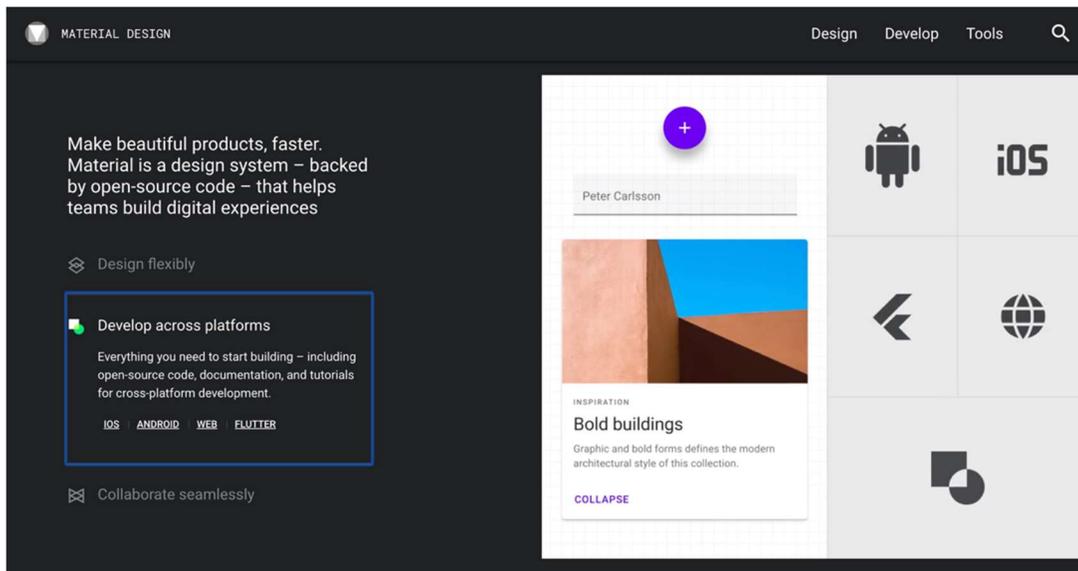


Figure 59: Webpage of Material Design

³¹ WIMP stands for Windows, Icons, Menus, Pointer, mentioned in the article Software Engineering Issues for Mobile Application Development in the section discussing user experience.

³² Design principles are being advocated by Google, <https://material.io/design/introduction/#principles>

9.1.3. Testing

As the scale of mobile application is increasing, testing play a vital role to ensure the normal operation of the application. We will try to emphasis testing in the further development, following the suggest ratio of 70% for small tests, 20% for medium tests and 10% for large tests³³. We should try to design and clearly determine the testing into those three categories in order to have a productive and effective testing.

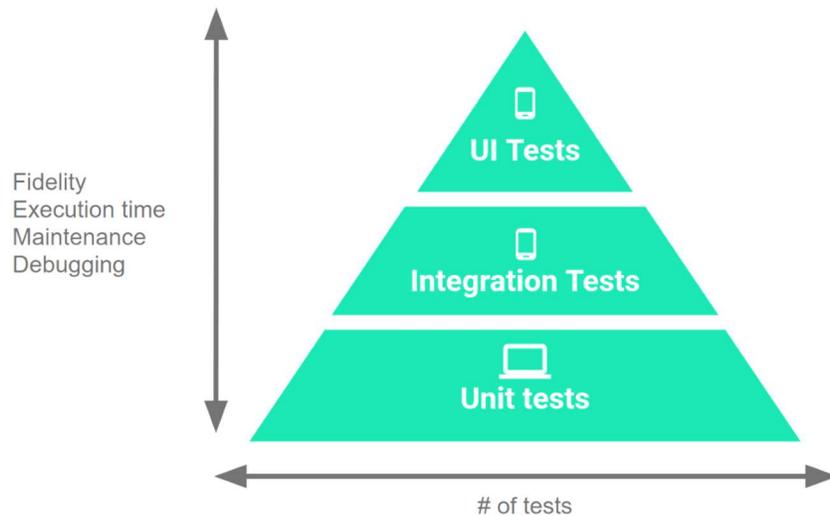


Figure 60 : Picture the relationship of different testing from the document

³³ Promoted principle of testing by the Android development document on <https://developer.android.com/training/testing/fundamentals>

9.2. AI Model

9.2.1. More Species

As a starting point, we only support the classification of 10 different species of trees in the semester. In the next phase, we expect we could support 10 more different species of trees.

Besides, as mentioned above, classifying trees with only leaves feature is not sufficient. We hope to extend the application by supporting classification using more features like tree bark, in order to increase the accuracy of the application on classifying trees.



Figure 61: Other than leaves, tree bark can be a feature for classifying trees.³⁴

³⁴ The image is obtained from <https://www.britannica.com/science/bark-plant-tissue>

9.2.2. Visualization

The neural network is still like a black box to us. Although the model achieves satisfied results, we do not know what exactly the model has done. By using some visualization approaches, we can inspect what features each layer extracted. In the next stage. We hope to visualize the neural network and improve our algorithm.

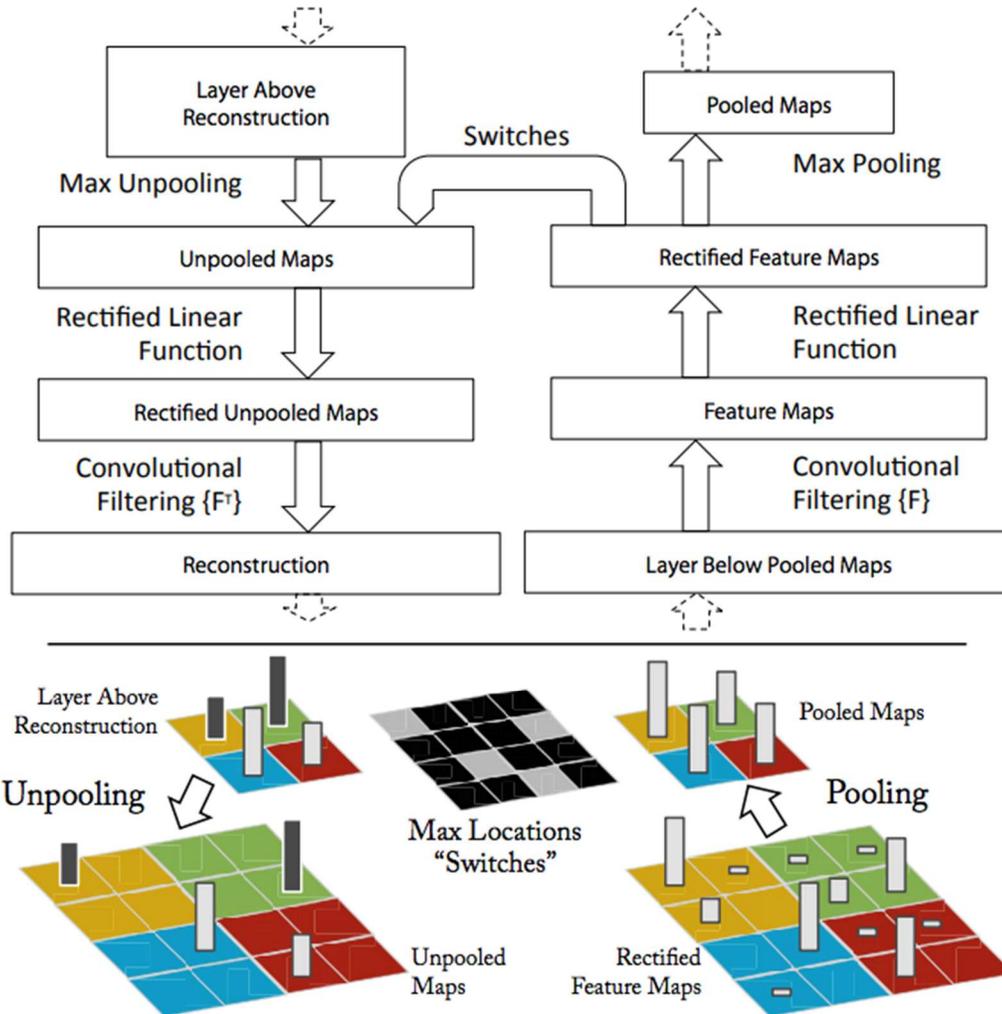


Figure 62: Use deconvnet(similar to inverse function of convolution) to perform visualization³⁵

³⁵ The detail of deconvnet is discussed in the article "Visualizing and understanding convolutional networks." The image is obtained from the article.

10. Acknowledgement

We would like to express our sincere and deep gratitude to our final year project supervisor Prof. Lyu Rung Tsong Michael and our tutor Xu hui for providing us the assistance regarding ideation as well as technical knowledge to overcome the difficulties we encounter on the road of the project, by directing us to the right path accomplishing the milestones and missions. Moreover, we would like to give credit to Dr. Tai Wai David Lau, the Curator of Shiu-Ying Hu Herbarium CUHK. Without the guidance from him on the knowledge of plants and the suggestion of make our own dataset, we might not able to locate correctly the plants in CUHK and struggle to get the raw data by ourselves.

11. References

- [1] Wasserman, A. (2010). Software engineering issues for mobile application development. *Proceedings of the FSE/SDP Workshop on Future of Software Engineering Research*, 397-400.
- [2] IDC - Smartphone Market Share - OS. (n.d.). Retrieved November 26, 2018, from <https://www.idc.com/promo/smartphone-market-share/os>
- [3] Activities | Android Developers. (n.d.). Retrieved from <https://developer.android.com/guide/components/activities/>
- [4] CardView | Android Developers. (n.d.). Retrieved from <https://developer.android.com/reference/android/support/v7/widget/CardView>
- [5] Save data in a local database using Room | Android Developers. (n.d.). Retrieved from <https://developer.android.com/training/data-storage/room/>
- [6] Smit, R. (n.d.). Blog | The Intent extras size limit. Retrieved from <https://www.neotechsoftware.com/blog/android-intent-size-limit>
- [7] Google. (n.d.). Material Design. Retrieved from <https://material.io>
- [8] Fundamentals of Testing | Android Developers. (n.d.). Retrieved from <https://developer.android.com/training/testing/fundamentals>
- [9] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016, 770-778.
- [10] Ghezzi, C., Jazayeri, M., & Mandrioli, D. (2003). *Fundamentals of software engineering* (2nd ed.). Upper Saddle River, N.J.: Prentice Hall.
- [11] CUHK campus 100 plants | Shiu Ying Hu Herbarium Collections. (n.d.). Retrieved from <http://syhuherbarium.sls.cuhk.edu.hk/collections/courseware/campus-100-plants/>

- [12] Image - Keras Documentation. (n.d.). Retrieved from <https://keras.io/preprocessing/image/>
- [13] Stack Overflow - Where Developers Learn, Share, & Build Careers. (n.d.). Retrieved from <https://stackoverflow.com/>
- [14] K. Simonyan and A. Zisserman. (2015). Very deep convolutional networks for large-scale image recognition. In ICLR, 2015
- [15] Szegedy, Wei Liu, Yangqing Jia, Sermanet, Reed, Anguelov, . . . Rabinovich. (2015). Going deeper with convolutions. In CVPR, 2015.
- [16] Hornik, Stinchcombe, & White. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5), 359-366.
- [17] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... & Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.
- [18] Zeiler, M. D., & Fergus, R. (2014, September). Visualizing and understanding convolutional networks. In *European conference on computer vision* (pp. 818-833). Springer, Cham.

12. Appendix

12.1. List of Figures

Figure 1: Data Augmentation of shearing using OpenCV and Keras	7
Figure 2: Statistic on Smartphone OS Market share from IDC	8
Figure 3: 583,521 questions have been asked about ios in stackoverflow	9
Figure 4: 1,155,094 questions have been asked about android in stackoverflow	9
Figure 5: Excel Table show the plants we collected	10
Figure 6 : Araucaria heterophylla Figure 7 : podocarpus macrophyllus.....	11
Figure 8 : Photo of "machilus chekiangensis" Figure 9: Photo of " ficus altissima" ...	11
Figure 10 : Photo of "cinnamomum burmannii"	11
Figure 11: Example photo of single leaf Figure 12: Example photo of Multiple Leaves	12
Figure 13 : Network Error in PictureThis and Pl@ntNet.....	18
Figure 14 : Method using in pl@ntNet and PictureThis as assistance of recognition....	20
Figure 15 : Community for users in PictureThis and Pl@ntNet.....	22
Figure 16 : Failure of capturing picture in PlantSnap and capture in Google Play of the UI.....	24
Figure 17 : Architecture Design Diagram.....	26
Figure 18 : Data Flow Diagram of mobile application	28
Figure 19 : Table Plant	30
Figure 20 : Use Case Diagram of application	31
Figure 21 : Activity Diagram of Basic Recognition	32
Figure 22 : Activity Diagram of Navigation	34
Figure 23 : Main page of Dr. Leaf Figure 24: Main page after taking photo	37
Figure 25: Top 3 Recognition Result Figure 26: Browser Page of plant details.....	38
Figure 27: Location mark of the plant in Google Map.....	38
Figure 28 : View all plants in CardView Figure 29: View all plants in Map	39
Figure 30 : Navigation in Google Map	39
Figure 31 : Main Page of Dr.Leaf	40
Figure 32 : Taken photo will be displayed	41
Figure 33 : Recognition Result	42
Figure 34 : Google Map with dropped pin.....	43
Figure 35 : Herbarium Website showing the Plant with tag "ca"	44
Figure 36 : Main Page of Dr. Leaf.....	45
Figure 37 : Show all plants in list	46
Figure 38 : View all in Google Map	47
Figure 39: Classification between cats and dogs using 2 features.....	51
Figure 40: A neuron in neural network	52

Figure 41: A feed forward neural network	53
Figure 42: Output image(right) was obtained by performing convolution operation using the convolution kernel(middle) on the input image(left).....	54
Figure 43: Visualization using deconvolutional approach. The figure shows features extracted in different layers.	55
Figure 44: MobileNet architecture	58
Figure 45: Normal convolutional layer(The symbol '*' is the convolution operation) ..	59
Figure 46: Depth-wise Separable Convolution(The symbol '*' is the convolution operation)	60
Figure 47: Trade-off between accuracy and speed in terms of number of mult-add operation	61
Figure 48: Trade-off between accuracy and size in terms of number of parameters ...	61
Figure 49: Ratio of calculation amount between depth-wise separable convolution and standard convolution. Numerator: Amount of calculation of depth-wise separable convolution. Denominator: Amount of calculation of standard convolution. D_k : Length of convolution kernel. M : Number of channels of input image. N : Number of channels of output image. D_F : Length of input image.	62
Figure 50: Comparison on performance between standard MobileNet and some popular CNN architectures. First column shows the accuracy of different architecture. Second column compares the speed in terms of million mult-adds. Third column compares the size of differ.	63
Figure 52: Search results of ficus altissima on Google Image	66
Figure 53: Flowers of Bauhinia variegata(left) and Hong Kong Orchid Tree(right).....	68
Figure 54: Architecture Design Diagram.....	69
Figure 55: Json file of schemas of database in current version	71
Figure 56 : Example of showing file color control xml in the application	73
Figure 57 : Example of showing String control in the application.....	73
Figure 58 : Well-organized file system in Android Studio project.....	74
Figure 59 : Application captures	75
Figure 60: Webpage of Material Design	79
Figure 61 : Picture the relationship of different testing from the document	80
Figure 62: Other than leaves, tree bark can be a feature for classifying trees.	81
Figure 63: Use deconvnet(similar to inverse function of convolution) to perform visualization	82

12.2. List of Tables

Table 1: Single leaf data set	13
Table 2: Multiple leaves data set.....	13
Table 3: Augmented Single leaf data set	14
Table 4: Augmented multiple leaves data set	15
Table 5: All data set	15
Table 6: Summary of mobile applications	17
Table 7: Accuracy of pretrained MobileNet re-trained by different datasets	65