DESIGN OF AN AUDIO-BASED MACHINE LEARNING ALGORITHM TO DETECT OIL PALM FRESH FRUIT BUNCHES HARVESTING

By

TAN JYH CHYAN

A project report submitted in the partial fulfilment of the requirements for the degree of Master of Information Systems

Department of Internet Engineering and Computer Science
Lee Kong Chian Faculty of Engineering and Science
Universiti Tunku Abdul Rahman

April 2018
DECLARATION

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at UTAR or other institutions.

Signature : 

________________________________________

Name : Tan Jyh Chyan

________________________________________

ID No. : 13UEM07932

________________________________________

Date : 

________________________________________
I certify that this project report entitled “DESIGN OF AN AUDIO-BASED MACHINE LEARNING ALGORITHM TO DETECT OIL PALM FRESH FRUIT BUNCHES HARVESTING” was prepared by TAN JYH CHYAN has met the required standard for submission in partial fulfilment of the requirements for the award of MASTER OF INFORMATION SYSTEM at Universiti Tunku Abdul Rahman.

Approved by,

Signature : ________________________________
Supervisor : Dr. Tay Yong Haur
Date : ________________________________
The copyright of this report belongs to the author under the terms of the copyright Act 1987 as qualified by Intellectual Property Policy of Universiti Tunku Abdul Rahman. Due acknowledgement shall always be made of the use of any material contained in, or derived from, this report.

© 2018, Tan Jyh Chyan. All right reserved.
ABSTRACT

DESIGN OF AN AUDIO-BASED MACHINE LEARNING ALGORITHM TO DETECT OIL PALM FRESH FRUIT BUNCHES HARVESTING

Palm oil in Malaysia is a main economic income to the nation. The performance in Malaysia palm oil industry will directly affect the nation economy and government income.

The main unsolved problem in Malaysia’s palm oil industry is the shortage of labor, especially labor for harvesting the Fresh Fruit Bunch of palm oil. However, huge improvement had been achieved in the transportation of FFB from plantation to mill with the introduction of mechanization. But to date, it has no FFB harvesting machinery which can perform as efficient and reliable as human harvester. As the consequences, some percentages of the FFB are left rotten, without reaching to the mill.

To encounter the labor shortage of harvesters, some level of automation is necessary. An IoT device can be used to automate harvesting activities and to provide real time information about the number of harvested FFB and its GPS location. It will optimize the harvesting process. The real time information concluded of quantity of harvested FFB, where the FFB were harvested, and who was the harvester.

This project is to design an audio-based machine learning algorithm for the IoT, to record the quantity of fruit bunches harvested by harvesters, which are done manually by the plantation supervisors.
TABLE OF CONTENTS

ABSTRACT iv
TABLE OF CONTENTS v
LIST OF TABLES vii
LIST OF FIGURES viii
LIST OF SYMBOLS / ABBREVIATIONS ix
LIST OF APPENDICES x

CHAPTER 1: INTRODUCTION

1.1 Introduction 1
1.2 Important of the Study 1
1.3 Problem Statement 2
1.4 Aims and Objectives 2
1.5 Scope and Limitation of the Study 2
1.6 Contribution of the Report 3
1.7 Outline of the Report 3

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction 4
2.2 Malaysia Oil Palm Industry: Overview 4
2.3 The harvesting process of Fresh Fruit Bunch of Oil Palm 5
2.4 MPOB (Malaysia Palm Oil Board) 6
2.5 Mechanization Technology in Oil Palm Plantation 6
2.6 Labor Shortage Problem of FFB harvesters 7
2.7 Ripeness Measurement of FFB 7
2.8 Smart Agriculture and The Use of Internet of Thing (IoT) 8
2.9 Raspberry Pi in Agriculture 8
Summary 9
CHAPTER 3: METHODOLOGY

3.1 Introduction
3.2 Overview of the system
3.3 Sampling of audio samples
3.3.1 Audio samples collection
3.3.2 Text files
3.4 Features creation
3.5 Training the Multilayer Neural Network
3.6 Neural Network classifier
Summary

CHAPTER 4: RESULT AND DISCUSSION

4.1 Introduction
4.2 Experiment 1
4.3 Experiment 2
4.4 Experiment 3
4.5 Experiment 4
Summary

CHAPTER 5: CONCLUSIONS

5.1 Conclusion
5.2 Future works

REFERENCES

APPENDICES

APPENDIX A: Crude Palm Oil Milling Process
APPENDIX B: Oil Palm Harvesting
APPENDIX C: Implementation
LIST OF TABLES

Table 3.1: The sampling of audio files 13
Table 4.1: Testing the Experiment 1’s classifier with training samples 19
Table 4.2: Testing the Experiment 1’s classifier with audio files 20
Table 4.3: Testing the Experiment 3’s classifier with training samples 25
LIST OF FIGURES

Figure 1.1: Smart harvesting of oil palm FFB 1
Figure 3.1: The system overview of the machine learning algorithm 11
Figure 4.1: Power diagrams of Experiment 1 data samples 20
Figure 4.2: Spectrograms of Experiment 1 data samples 21
Figure 4.3: Power diagrams of Experiment 3 data samples 23
Figure 4.4: Spectrograms of Experiment 3 data samples 24
Figure A.1: Crude palm oil milling process 32
Figure B.1: Harvester harvests with mechanization cutter 33
Figure B.2: Harvesting machine harvests oil palm FFB 33
Figure B.3: The oil palm FFBs with different ripeness 34
Figure B.4: An oil palm FFBs harvesting machine 34
## LIST OF SYMBOLS / ABBREVIATIONS

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPO</td>
<td>Crude Palm Oil</td>
</tr>
<tr>
<td>FFB</td>
<td>Fresh Fruit Bunch</td>
</tr>
<tr>
<td>IoT</td>
<td>Internet of Thing</td>
</tr>
<tr>
<td>RGB</td>
<td>Red Green Blue</td>
</tr>
<tr>
<td>MACRES</td>
<td>Malaysia Centre For Remote Sensor</td>
</tr>
<tr>
<td>MPOB</td>
<td>Malaysia Palm Oil Board</td>
</tr>
<tr>
<td>NN</td>
<td>Neural Network</td>
</tr>
</tbody>
</table>
LIST OF APPENDICES

APPENDIX A: Crude palm oil milling process
APPENDIX B: Oil palm harvesting.
APPENDIX B: Implementation
CHAPTER 1: INTRODUCTION

1.1 Introduction
This project is to develop an audio-based machine learning algorithm to count the quantity of fruit bunches harvested by a harvester. The software can be installed in an IoT device to count the quantity of fruit bunches harvested by harvester in oil palm plantation.

The developed software will record down the number of the harvested fruit bunches by the harvester. This will provide the information for the management of plantation to manage the harvesting process efficiently.

As shown in the figure 1.1, the whole smart harvesting system consists of an IoT device attached to the harvester. The IoT device will record the audio data and GPS location of the harvester. This device will process the audio data and output the quantity of harvested FFB with the GPS location. Through wireless communication, the management staffs at the management office can view the number of the harvested FFB and the GPS location.

This project is to design an audio-based machine learning algorithm, to receive the environment sound and output the number of the harvested FFB by the harvester.

![Image](image.png)

Figure 1.1 The smart harvesting of oil palm FFB.

1.2 Importance of the Study
This project is to develop a software to count the fruit harvested by the harvesters. The software can be installed to an IoT device to count the quantity of fruit bunches
harvested by harvester and the GPS location. With the ability of wireless network, the IoT device can provide real time information.

Currently, counting of quantity harvested fruit bunches are done by plantation supervisors. This fruit bunches counting software can automate the counting and decrease the workload of plantation supervisor, and the plantation may need less number of plantation supervisors. The management staff in the office also able to view the GPS location of the harvested area.

1.3 Problem Statement
The project problem statement is:

The labor shortage in Malaysia’s oil palm plantation can be eased by automating the counting of harvested FFB quantity with audio-based machine learning algorithm in an IoT.

1.4 Aims and Objectives
The aim of this project is to develop software to count the number of fruit bunches harvested by harvesters, to decrease the workload of manual counting.

The objectives of the project are listed as the following:
1. To do literature review on practices of the oil palm plantation.
2. To build software to process the sound data and to output the number of harvested fruits.
3. To test the software with the actual audio data collected from oil palm plantation.

1.5 Scope and Limitation of the Study
The scope of the project is listed as the following:
1. To collect the sound data of fruit bunches harvesting.
2. To collect basketball dropping sound data for testing purposes.
3. To build a program to process the sound data, including data storage, data processing, and output information storage.
4. To do a quantitative test of the software with the actual audio data collected from the oil palm plantation.
1.6 Contribution of the project
The contribution of the project is that with the developed software, it can be installed to an IoT device and attached to the harvester. The real-time information will facilitate the oil palm plantation management.

The automation of the counting process can decrease the workload of human worker, and less workers are required.

1.7 Outline of the report
This project proposal report will cover the introduction, literature review, methodology of development, preliminary finding, conclusion, and appendices.

The literature review will study the palm oil industry on the labor shortage problem, the usage of mechanization to ease the labor requirement, and the labor shortage problem.

In the methodology chapter, describe how the audio-based machine learning algorithm is built.

In the result and discussion chapter, the works and the result of the project was illustrated.

Follow by the conclusion and the appendices. The appendices consisted of the illustration of FFB oil extraction processes, FFB harvesting, and implementation, such as the installation of docker toolbox.
CHAPTER 2: LITERATURE REVIEW

2.1 Introduction
Many studies and researches had been conducted on different aspects in oil palm industry. Some of the projects were sponsored by government agency, such as MPOB, carried out by universities.

This literature review will cover on the overview of Malaysia's oil palm industry, the oil palm harvesting process, and the uses of technology to facilitate the oil palm harvesting.

2.2 Malaysia Oil Palm Industry: Overview
Referred to DOSM (2016), remarked that in 2015, “Malaysia's agriculture sector continued to expand with a contribution of 8.9 per cent to the Gross Domestic Product (GDP). Oil palm was a major contributor to the GDP of agriculture sector at 46.9 per cent”. Malaysia is doing well and successful in the palm oil industry.

Malaysia is the second largest of palm oil product, after overtaken by Indonesia. In year 2016, the export value of palm oil product is RM64.58bil. The palm oil industry currently on 2017 has about 431,357 workers in estates, of which 77% or 332,135 were foreigners (S. Puspadevi, 2017). “Malaysia is the world's largest producer and exporter with 50 percent share in palm oil production and 61 percent in export” (Ming, Chandramohan, 2002).

Comparing with seeds oil such as soya bean, rapeseed, sun flower, peanut and etc, palm oil is far more efficient in term of yield per ha (Murphy, 2014) Typical average global basic yield for palm oil is 4.0t per ha, whether other oil seed crops yield are 0.3 to 1.2 t per ha (Murphy, 2014).

The global demand for palm oil is increasing. The increase of global demand for palm oil is due to the continual population growth and economic development of the oil importer countries, such as India subcontinents, China, European Union, West Asia, USA, and etc (Murphy, 2014)

Palm oil industry is a labor intensive industrial, the shortage of labor in the palm oil industries has caused the decrease in production, due to wasted rotten FFB un-harvested. For example, in East Malaysia, the Sarawak Oil Palm Plantation Asso-
Association (SOPPA) had reported that 20%-30% labor shortage had resulted in 15% losses due to wasted FFBs un-harvested (Murphy, 2014).

As mentioned, the typical average of oil palm yield was 4.0 t per ha, the maximum oil yield potential at commercial-scale was estimated to be in the range of 10-11t per ha (Donough, et, al, 2010). In good management environment, the yield of the palm oil can be increased. This will avoid more land to be deforested to get the same yield.

2.3 The harvesting process of Fresh Fruit Bunch (FFB) of oil palm

The harvesting of FFB is an arduous task. The following are the typical harvesting process of FFB of oil palm (Hasry, 2012):

1. Tool preparation. Chisel is used for young plants and sickle is used for taller older trees.
2. Moving through the plantation, to identify the ripe brunched, by inspection of fruits color and quantities of loose fruits on the land.
3. After identified the ripe FFB, removing the subtending frond, then cut the FFB.
4. Picking FFBs and loading in the wheel-barrow.
5. Picking up all the loose fruits.
6. Load the FFB to the collection point, normally by road-side.

The harvesting works are usually carried out by team of two, one harvester and one helper, with a separate team of loading workers to load the fruits to the platform of truck and to transport to the mill.

The quality of the oil depends on the ripeness of the fruits, the unripe fruit may yield low quality of crude palm oil (CPO), due to the rapid rise in free fatty acids. It is recommended to process the cut fruits within 48 hours to ensure the quality of the oil.
2.4 MPOB (Malaysia Palm Oil Board)

Malaysia Palm Oil Board (MPOB) is a government agency responsible for the research and development (R&D), and the regulatory and licensing of Malaysia palm oil industry.

MPOB's R&D is directed by the following three prong strategies (Cheng, 2002):

1. High Income Strategy - increase productivity through technology and good management.
3. Value-Addition Strategy - to conduct R&D to increase the value of products of palm oil, for edible or non-edible products.

2.5 Mechanization Technology in Oil Palm Plantation

In other sectors of agriculture, the introduction of harvesting machines works effectively for some crops, such as rice plantation, potatoes plantation, radish plantation. Where the customized harvesting machine can effectively replace human for harvesting job and complete the job in a shorter time.

For the palm oil plantation, due to the terrain of the plantation and natural condition of palm oil tree and fruits, such as the undulated landscape, peat soil, fruits are scattering on tall trees, it is very challenging and difficult to create an efficient and effective harvesting machine.

However, MPOB has always tried to develop mechanized tool with the aim to increase productivity, reduce dependence on worker, and reduce operation cost (Ramdhan, Rahim, 2014). Referred to Ramdhani, Rahim (2014), some examples of the mechanization in the oil palm plantation are listed as the following:

1. Aluminium harvesting tool with the trade name Zirafah.
2. Hi-reaching pole.
4. Mechanical grabber on mini-tractor.
5. Track type transport for peat area (Beluga).

As listed above, the mechanization are lighter pole, motorized pole, and vehicle to transport the FFB, there are some vehicles with crane to reach the tree to cut the FFB, but it is very time consuming and the vehicles are difficult to access soft and undulated land.
For oil palm harvesting, it is very challenging and difficult to build machinery that can harvest the FFB of oil palm effectively such as other crops' harvesting machinery, where the crops are planted in flat lands, a customized machine can harvest effectively and fast. For the undulated large scattered area of oil palm, the manual harvesting by harvesters is the most practical method to harvest the FFB of oil palm.

Barring the limitations of terrain, many mechanizations are successful applied to the operations such as herbicide sprayer, fertilizing, draining, FFB transportation of the FFB from the tree to the roadside and to the mills (Ming, Chandramohan, 2002).

The oil palm plantations planted on the 1970's and 1980's were built with poor infrastructure, causing accessing problems to the palms (Ming, Chandramohan, 2002).

2.6 Labor Shortage problem of FFB harvesters
Mechanization of operations in oil palm plantation can alleviate labor problem in some of the plantation operations. Automation in the palm oil mills can reduce 40 percent of labour (Ming, Chandramohan, 2002).

Unfortunately, there has been no machine that is effective, efficient, and reliable to replace human harvesters (Ming, Chandramohan, 2002). Referred to Ming, Chandramohan (2002), a human harvester can harvest up to 1.5 to 2 ha of oil palm trees per day. This can be improved to 10 ha per day as the target.

Referred to Ming, Chandramohan (2002), issues arise over wages of skill harvesters and loaders. Harvesters need skill to harvest FFBs, some individual with better skill can harvest more fruits. Some plantations have paid the harvesters extra incentive based on number of harvested FFB. However, labor shortage of harvesters is remained unsolved.

2.7 Ripeness Measurement of FFB
The ripeness of harvested FFB is crucial for the quality of the crude palm oil production.

The harvesters of the FFB need to make sure the ripeness of fruits before harvesting. The recent method is by viewing the color of the FFB or counting the loose fruits in the ground. This practice is not efficient, as viewing the color of FFB the person need to have experience and some of the palm trees are tall and difficult to view. There are some solutions of using camera to capture the image of FFB, and by machine vision and analysis to grade the ripeness of FFB. The FFB can be roughly
categorized as unripe, under-ripe, ripe, or overripe based on the color hue of the FFB image.

Referred to research by Alfatni, et al, (2008), the FFB image’s red green blue (RGB) color value, are processed by the computer to get the mean value, and this value is used to grade the ripeness of the FFB. With a grading tool attached to the harvesting pole, the ripeness of the FFB can be measure for more efficient oil palm harvesting. This study was conducted by the research team of University Putra Malaysia (UPM), in collaboration with Malaysia Palm Oil Board (MPOB) and Malaysia Centre For Remoter Sensor (MACRES), (Alfatni, et, al, 2008).

2.8 Smart Agriculture and The Use of Internet of Thing (IoT)

In today farm, the advance technologies with the IoT have pushed the future of farming and plantation to the next level. Smart agriculture with the use of sensor in the IoT allows the farmers or plantation operators to monitor their farm or plantation efficiently. They can monitor their farms or plantations through smart phone.

The collected data from the sensors is processed and sent to central servers may be cloud located, to supply precious information to the concerned stakeholders. As the assistant of the smart agricultural, the stakeholders can manage their farms or plantations efficiently.

The smart agricultural can be applied to the oil palm plantations as well. The application of satellite image and drone, have capture the images of the oil palm plantation. The image can be processed to create useful information, such as the health of the trees.

Referred to TongKe (2013), Smart Agricultural with the application cloud computing Internet of Thing (IoT) is the latest trend in agricultural, it provides information for the stakeholders to remote monitor the farms.

2.9 Raspberry Pi in agriculture

Raspberry Pi is a single board computer. Raspberry Pi Foundation in United Kingdom had developed it for the purposes of promoting computer science teaching promote in school or developing countries. Raspberry Pi has the size of a credit card, 3 inches by 2 inches.

In agriculture, Raspberry Pi IoT devices can be used to measure the soil humidity, temperature, leaf moisture, and to activate the actions such as irrigating,
fertilizing, or sending acknowledge messages to the concerned parties through wireless network.

Examples of Raspberry Pi application are listed as the following:

1. Usage of Raspberry Pi to capture images of growing of crop. Further analysis is conducted to relate the environment variables with the crop growth, in the objective to reduce farming costs (Sam, 2017).

2. Robots are built with Raspberry Pi. The parameters such as moisture, light, color, or temperature, are monitored by the Robot and send message by GSM technology to the farmer. The useful information will facilitate the farmers to take effective decisions (Balaj, Lakshmanan, 2017).

3. Design of Smart Irrigation System Using Raspberry Pi for Agriculture (Agrawal, Singhal, 2017). The above design consisted of automatic control of water motor, monitor the plant growth and live streaming of farm on PC by Wi-Fi.

Raspberry Pi is very useful in the design IoT devices. It is small and economic.

**Summary**

Malaysia is the main exporter of the crude oil palm. To remain competitive, the oil palm industries need to find solution to solve the problem of the labor shortage of oil palm plantation, especially labors for FFB harvesters.

The MPOB has put a lot of effort to increase the efficiency of palm oil industry in all areas, including planting, maintenance, harvesting, and mill processing. Many improvements had been achieved in these areas, but there is no invention of any mechanization to replace human harvester. The human harvester is still the most reliable and efficient.

The IoT devices have successfully applied to the agricultural, in the farming operations such as irrigating, fertilizing, and monitoring. The IoT devices can be applied as well to the oil palm plantations.
CHAPTER 3: METHODOLOGY

3.1 Introduction
This project is to develop a machine learning algorithm to count the number of fruit bunches harvested by harvester in oil palm plantation.

3.2 Overview of the system.
The audio-based machine learning algorithm was written using python Jupyter Notebook. The Docker Toolbox was installed in the Window 10 notebook. In the Docker toolbox, the image of ‘tensorflow/tensorflow’ from docker hub at hub.docker.com was pulled and a container was created.

Other required packages such as Librosa, scikits.talkbox, and etc were installed through ‘pip install xxxx’ inside the terminal of the tensorflow/tensorflow’s container. With Tensorflow, a multilayer neural network program is created to classify the audio data.

The audio-based machine learning algorithm was written with reference to Aeqib (2016), where a multilayer neural network classifier was used to classify the urban sound of ten classes: street music, siren, jackhammer, gun shot, children playing, drilling sound, car horn, engine-idling, air-conditioner and dog bark.

As shown in the following figure 3.1, is the system overview of the machine learning algorithm. The system consists of two parts, training of Neural Network (NN) and the classifier. The NN classifier is built based on the optimum weights from the training of Neural Network.

In part one, the training of Neural Network section, every audio wav file has two text files to keep, the times of the dropping sounds and the times of non-dropping sounds. Based on the times in the text files, 6 second audio samples consist of positive samples and negative samples are created.

From the audio samples, the MFCC features are extracted and saved in data files with their class label data file. This features data files and label files will be fed to train the Multilayer Neural Network. The optimum NN weight will be saved and used by NN classifier.

In part two, the NN classifier section, each 6 second audio clip will be sampled from the beginning of the audio files with window step of one second and shifting forward. Each audio clip will be fed to the NN classifier for classification, every
positive classification will be recorded, and output the total number of dropping sound after classification of all the audio clips from the audio files.

Figure 3.1, The system overview of the machine learning algorithm.

3.3 Sampling of audio samples
The machine learning algorithm is designed and tested with the basketball dropping sound samples. Two text files are created to contain the dropping time and non-
dropping time of the basketball. The 6 second wav file samples which are labelled with 0 or 1 according to their class as non-dropping sound or dropping sound are created based on the times from the text files.

3.3.1 Audio raw data collection.
Twenty-two audio files with 10 minutes each were collected by dropping a basketball manually. Each file contained 60 units of basketball dropping sound. The total was 1320 units of basketball dropping sound.

The audio files were placed in a folder and ready to be loaded for sampling work. The audio files are listed in the following table 3.1, the non-dropping samples and dropping samples were placed in the folder named ‘Foldx’ as shown in the table. For example, the file 171121_222534.wav, the non-dropping samples were placed in Fold3 and the dropping samples were placed in Fold2.

The 6 seconds audio samples were labelled automatically with 0 or 1 according to their class as non-dropping sound or dropping sound.

<table>
<thead>
<tr>
<th>Nos:</th>
<th>File name</th>
<th>Non-dropping</th>
<th>dropping</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>171121_223939.wav</td>
<td>Fold1</td>
<td>Fold1</td>
</tr>
<tr>
<td>2</td>
<td>171123_222534.wav</td>
<td>Fold3</td>
<td>Fold2</td>
</tr>
<tr>
<td>3</td>
<td>171124_220025.wav</td>
<td>Fold3</td>
<td>Fold2</td>
</tr>
<tr>
<td>4</td>
<td>171124_221412.wav</td>
<td>Fold3</td>
<td>Fold2</td>
</tr>
<tr>
<td>5</td>
<td>171125_133507.wav</td>
<td>Fold2</td>
<td>Fold3</td>
</tr>
<tr>
<td>6</td>
<td>171125_140219.wav</td>
<td>Fold2</td>
<td>Fold3</td>
</tr>
<tr>
<td>7</td>
<td>171125_150208.wav</td>
<td>Fold2</td>
<td>Fold3</td>
</tr>
<tr>
<td>8</td>
<td>171125_153724.wav</td>
<td>Fold4</td>
<td>Fold5</td>
</tr>
<tr>
<td>9</td>
<td>171125_160257.wav</td>
<td>Fold4</td>
<td>Fold5</td>
</tr>
<tr>
<td>10</td>
<td>171125_174356.wav</td>
<td>Fold4</td>
<td>Fold5</td>
</tr>
<tr>
<td>11</td>
<td>171125_181712.wav</td>
<td>Fold5</td>
<td>Fold4</td>
</tr>
<tr>
<td>12</td>
<td>171125_183752.wav</td>
<td>Fold5</td>
<td>Fold4</td>
</tr>
<tr>
<td>13</td>
<td>171125_215320.wav</td>
<td>Fold5</td>
<td>Fold4</td>
</tr>
<tr>
<td>14</td>
<td>171125_221114.wav</td>
<td>Fold6</td>
<td>Fold7</td>
</tr>
<tr>
<td></td>
<td>Audio File</td>
<td>Folder 1</td>
<td>Folder 2</td>
</tr>
<tr>
<td>----</td>
<td>-------------------------</td>
<td>----------</td>
<td>----------</td>
</tr>
<tr>
<td>15</td>
<td>171126_110229.wav</td>
<td>Fold6</td>
<td>Fold7</td>
</tr>
<tr>
<td>16</td>
<td>171126_123253.wav</td>
<td>Fold6</td>
<td>Fold7</td>
</tr>
<tr>
<td>17</td>
<td>171126_142625.wav</td>
<td>Fold8</td>
<td>Fold6</td>
</tr>
<tr>
<td>18</td>
<td>171126_162454.wav</td>
<td>Fold8</td>
<td>Fold6</td>
</tr>
<tr>
<td>19</td>
<td>171127_211716.wav</td>
<td>Fold8</td>
<td>Fold6</td>
</tr>
<tr>
<td>20</td>
<td>171128_211311.wav</td>
<td>Fold7</td>
<td>Fold8</td>
</tr>
<tr>
<td>21</td>
<td>171128_221513.wav</td>
<td>Fold7</td>
<td>Fold8</td>
</tr>
<tr>
<td>22</td>
<td>171129_212319.wav</td>
<td>Fold7</td>
<td>Fold8</td>
</tr>
</tbody>
</table>

Table 3.1 The sampling of audio files.

An example of the creating of the samples are shown as the following.

```python
In [1]: from FfbSound import

In [12]: create_wav("Basketball_sound/171125_215320.wav","notDroptime.txt", "0", "5")
```

The raw audio file from directory of “Basketball_sound/171125_215320.wav” was read and by referring to the text file “notDroptime.txt”, the samples were labelled with “0” to the sample file name and saved to folder Fold “5” of FFB-sound-Data folder.

3.3.2 Text file

Text files were created in the docker’s container, which contain the times in second to indicate the time to sample the 6 second audio wav samples. The screen-shot of the dropping sound’s text file is shown as the following.

Text files were created to provide the time of dropping sound or non-dropping sound. Reading from the audio sound files, a python function created the six seconds wave
samples referring to the time from the text file, and the final samples were labelled with sequence number and class label accordingly.

3.4 Features Creation

The Mel Frequency Cepstral Coefficients (MFCC) of each 6 second sound samples were extracted, stored and been ready to feed into the multilayer neural network. The mfcc() function from Scikits.talkbox library was used to extract the MFCC features.

In extraction of MFCC features, the audio clip is Fourier transformed, mapped the spectrum with Mel scales, obtained the logarithm of spectrum, then Discrete Cosine transformed. The result spectrum amplitudes are the MFCC features.

Referred to Luis and Willi (2015), a classifier with MFCC features showed better performance compare with FFT (Fast Fourier Transform). The MFCC features classifier was applied to the classification of six distinct music genres: Classical, Jazz, Country, Pop, Rock, and Metal.

The following function ‘extract_mfcc’ was used to extract the MFCC features from the samples.

```python
def extract_mfcc(file_name):
    sample_rate, X = scipy.io.wavfile.read(file_name)
    print(X.shape)
    ceps, smceps, spec = mfcc(X)
    print(ceps.shape)
    X = np.mean(ceps, axis=0)
    print(X.shape)
    return X
```

The following function ‘parse_mfcc_features’ was called to extract the MFCC features from the samples and converted them into the program data structures, and ready to be fed into the machine learning program.

```python
def parse_mfcc_features(parent_dir, sub_dir, file_ext='*.wav'):
    features, labels = np.empty((0, 13)), np.empty(0)
    for label, sub_dir in os.walk(parent_dir):
        print('sub_dir: %s' % (sub_dir))
        for fn in glob.glob(os.path.join(parent_dir, sub_dir, file_ext)):
            try:
                # still print wav info (fn)
                print('extract file: %s' % (fn))
                except Exception as e:
                    print('Error unhandle wav file. %s' % (e))
                    continue
            try:
                ext_features = extract_mfcc(fn)
                except Exception as e:
                    print('Error extract feature error. %s' % (e))
                    continue
                features = np.vstack((features, ext_features))
            fnm, ext = os.path.splitext(fn)
            labels = np.append(labels, fnm.split('/')[-1][3])
            print(np.array(labels))
    return np.array(features), np.array(labels, dtype = np.int)
```
The MFCC features will be generated once from the samples and saved as `.npy` files, and read them instead of regenerating them each time we train our classifier.

As shown in the above codes, the MFCC features data and the label data were stored as the ‘13ctr_features’ and ‘13ctr_labels’.npy files.

Referred to Aeqib(2016), 193 audio features can be extracted by methods of Melspectogram, Mfcc, Chorma-stft, Spectral-contrast and Tonnetz. The codes are showed as the following.

```python
def extract_feature(file_name):
    X, sample_rate = librosa.load(file_name)

    # stft
    stft = np.abs(librosa.stft(X))

    # mfcc
    mfcoc = np.mean(librosa.feature.mfcc(y=X, sr=sample_rate, n_mfcc=40), T, axis=0)

    # chroma
    chroma = np.mean(librosa.feature.chroma_stft(S=stft, sr=sample_rate), T, axis=0)

    # melspectrogram
    mel = np.mean(librosa.feature.melspectrogram(X, sr=sample_rate), T, axis=0)

    return mfcoc, chroma, mel, contrast, tonnetz

def parse_npy_features(parent_dir, sub_dirs, file_ext='*.wav'):
    features, labels = np.empty((0, 193)), np.empty(0)
    for label, sub_dir in onomatopoeia[sub_dirs]:
        for fn in glob.glob(os.path.join(parent_dir, sub_dir, file_ext)):
            try:
                X, sample_rate = librosa.load(fn)
                # stft, stft = np.abs(librosa.stft(X))
                mfcoc = np.mean(librosa.feature.mfcc(y=X, sr=sample_rate, n_mfcc=40), T, axis=0)
                chroma = np.mean(librosa.feature.chroma_stft(S=stft, sr=sample_rate), T, axis=0)
                mel = np.mean(librosa.feature.melspectrogram(X, sr=sample_rate), T, axis=0)
                features = np.vstack((features, mfcoc, chroma, mel, contrast, tonnetz))
                labels = np.append(labels, fn.split('_')[3])
                continue
            except Exception as e:
                print(f'[*] [Error] {e} in file {fn}
```

3.5 Training with Multilayer Neural Network

A multilayer neural network logistic regression classifier was built with Tensorflow. The multilayer neural network consisted input layer with 13 inputs, a hidden layer with 13 inputs and 26 outputs, and an output layer with 26 inputs and 2 outputs.

The following codes loaded the samples features and labels into the program data structures.

```plaintext
In [2]:
parent_dir = 'FB2-Sound-Data'
tr_sub_dirs = ['Fold1', 'Fold2', 'Fold3']
ta_sub_dirs = ['Fold5']

tr_features, tr_labels = parse_npy_features(parent_dir, tr_sub_dirs)
ta_features, ta_labels = parse_npy_features(parent_dir, ta_sub_dirs)

tr_labels = one_hot_encode(tr_labels)
ta_labels = one_hot_encode(ta_labels)

np.save("193str_features", tr_features)
np.save("193str_features", ta_features)
np.save("193str_labels", tr_labels)
np.save("193str_labels", ta_labels)
```

The input layer applied a rectified linear unit (ReLU) as the activation function and the output layer applied Softmax Regression as the activation function.

The optimum NN weights will be saved for later retraining, or the usage of NN classifier to classify and to count the number of dropping sounds.

3.6 Neural Network Classifier

The optimum NN weights are used to classify the audio sound, to count and output the number of the dropping sound in the audio sound.

A pointer will be pointed to the beginning of the audio wav file, and 6 seconds audio data is extracted, and feed in the Neural Network classifier to be classified. After the Neural Network classifier finished the task, the pointer will move forward with a predefined time and next sample is processed until the end of the audio signal. A counter will count the number of the dropping sound in the classification process and output the total number of dropping sound.
Summary

The system consisted of two sections, training section and classifier section.

In the training section, supervised learning of Multilayer Neural Network was carried out to obtain the optimum NN weights.

In the classifier section, the optimum NN weights were used to build the NN classifier. The NN classifier will scan through the audio file with a stipulated window size to find the dropping sound. Every positive count was added to a counter. The counter recorded the total number of dropping sound.
CHAPTER 4: RESULT AND DISCUSSION

4.1 Introduction
The Multilayer Neural Network was trained with the labelled audio samples to obtain the optimum NN weight, and this optimum NN weight will be use by the NN classifier to classify the audio wav file.

4.2 Experiment 1
Experiment 1:
- Basketball dropping sounds are used as audio data.
- Multilayer Neural Network was trained with all the audio samples from folder Fold1 to Fold8, was tested with the same audio samples.
- The input layer of NN was 13 nodes, hidden layer 26 nodes, and output layer 2 nodes.
- The Mel Frequency Cepstral Coefficients were extracted as the features data, which contained 13 coefficients for each 6 second audio sample, total of 1320 units of samples of each of the dropping class and non-dropping class.

Purposes:
The purposes of the Experiment 1 are to train the Multilayer Neural Network with the basketball dropping sound audio data and to test the NN classifier with the audio files

Result:
The achieved accuracy was 95% after training, the testing of the counting of the number of dropping sounds are as shown in the following table 4.1.

<table>
<thead>
<tr>
<th>Nos:</th>
<th>File name</th>
<th>Actual number of dropping sound:</th>
<th>NN classifier output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>171121_223939.wav</td>
<td>60</td>
<td>62</td>
</tr>
<tr>
<td>2</td>
<td>171123_222534.wav</td>
<td>60</td>
<td>59</td>
</tr>
<tr>
<td>3</td>
<td>171124_220025.wav</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>4</td>
<td>171124_221412.wav</td>
<td>60</td>
<td>59</td>
</tr>
</tbody>
</table>
Table 4.1 Testing the Experiment 1’s classifier with training audio files

<table>
<thead>
<tr>
<th>Nos</th>
<th>File name</th>
<th>Actual number of dropping sound</th>
<th>NN classifier output</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>171125_133507.wav</td>
<td>60</td>
<td>58</td>
</tr>
<tr>
<td>6</td>
<td>171125_140219.wav</td>
<td>60</td>
<td>64</td>
</tr>
<tr>
<td>7</td>
<td>171125_150208.wav</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>8</td>
<td>171125_153724.wav</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>9</td>
<td>171125_160257.wav</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>10</td>
<td>171125_174356.wav</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>11</td>
<td>171125_181712.wav</td>
<td>60</td>
<td>61</td>
</tr>
<tr>
<td>12</td>
<td>171125_183752.wav</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>13</td>
<td>171125_215320.wav</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>14</td>
<td>171125_221114.wav</td>
<td>60</td>
<td>61</td>
</tr>
<tr>
<td>15</td>
<td>171126_110229.wav</td>
<td>60</td>
<td>61</td>
</tr>
<tr>
<td>16</td>
<td>171126_123253.wav</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>17</td>
<td>171126_142625.wav</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>18</td>
<td>171126_162454.wav</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>19</td>
<td>171127_211716.wav</td>
<td>60</td>
<td>58</td>
</tr>
<tr>
<td>20</td>
<td>171128_211311.wav</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>21</td>
<td>171128_221513.wav</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>22</td>
<td>171129_212319.wav</td>
<td>60</td>
<td>56</td>
</tr>
</tbody>
</table>

With an overfitted NN classifier, it showed that the NN classifier can perform the classification tasks, 12 audio samples were fully tally, 5 audio samples were out of plus or minus 1, 3 audio samples were out of plus or minus 2, and 2 audio samples were out of plus or minus 4.

The NN classifier were tested with other sound samples, the results are shown as the following table 4.2. The result showed that the NN classifier failed to classify correctly the sound samples which are new or never learned.

<table>
<thead>
<tr>
<th>Nos</th>
<th>File name</th>
<th>Actual number of dropping sound</th>
<th>NN classifier output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>180405_232653.wav</td>
<td>12</td>
<td>11</td>
</tr>
</tbody>
</table>
Basketball dropping sound in the same environment as the samples

<table>
<thead>
<tr>
<th></th>
<th>Audio File</th>
<th>Class 1</th>
<th>Class 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>180405_233015.wav</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>180405_233321.wav</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 4.2. Testing the Experiment 1’s classifier with audio files

The classification of basketball dropping sound is simple task and as the purpose to design the pilot machine learning algorithm. The different between the dropping and non-dropping sample is obvious through human hearing, wave diagram, and spectrogram shown as the following Figure 4.1 and Figure 4.2.

Figure 4.1 Power diagrams of the Experiment 1 data samples
Figure 4.2 Spectrograms of the Experiment 1 data samples

4.3 Experiment 2

Experiment 2:
- Basketball dropping sounds were used as audio data.
- Multilayer Neural Network was trained with all the audio samples from folder Fold1 to Fold8, was tested with the same audio samples.
- The input layer of NN was 13 nodes, hidden layer 26 nodes, and output layer 2 nodes.
- 193 audio features were extracted by methods of Melspectogram, Mfcc, Chorma-stft, Spectral-contrast and Tonnetz.
Purposes:

The purposes of the Experiment 2 are to use the 193 audio features of ‘Melspectogram, Mfcc, Chorma-stft, Spectral-contrast and Tonnetz’ to train the Multilayer Neural Network and to compare with the 13 audio features of MFCC.

Result:

The achieved accuracy was 0.49981 or 50% after 10 thousand epochs of training. This showed that the NN failed to learn, as for two class classification, the accuracy of 50% was the worst and could be achieved by random guessing.

This may be caused by the higher complexity of the 193 audio features of ‘Melspectogram, Mfcc, Chorma-stft, Spectral-contrast and Tonnetz’, with 193 dimensions of features. The designed Multilayer Neural Network was too simple to carry out the training task. To further tested the 193 audio features, more complicated and better model need to be applied. However, if the 13 features MFCC is sufficient, we may use the 13 features MFCC in the reason of simplicity.

4.4 Experiment 3

Experiment 3:

- FFB sounds from plantation were used as audio data.
- Multilayer Neural Network was trained with all the audio samples from folder Fold1 to Fold8, was tested with the same audio samples.
- The input layer of NN was 13 nodes, hidden layer 26 nodes, and output layer 2 nodes.
- The Mel Frequency Cepstral Coefficients were extracted as the features data, which contained 13 coefficients for each 6 second audio sample. Total of 481 audio samples are used for each class, positive class or negative class.

Purposes:

The purposes of Experiment 3 are to train the Multilayer Neural Network with a small quantities of actual audio samples from the oil palm plantation.
Result:

The achieved accuracy was 0.501037 or 50% after 10 thousand epochs of training. This showed that the NN failed to learn, as for two class classification, the accuracy of 50% was the worst and could be achieved by random guessing.

The poor result was due to poor quality of the sounds samples. The sound samples of the FFB dropping sound was recorded by a recorder standing near the harvester. The actual time of FFB dropping was manually recorded. Total of 100 minute of several audio samples are collected, with total of 93 FFB dropping sounds.

The classification of FFB dropping sound in Experiment 4 was fail due to the poor quality of the sound samples. The different between the dropping and non-dropping sample are not obvious through human hearing, wave diagram, and spectrogram shown as the following Figure 4.3 and Figure 4.4.

Figure 4.3 Power diagram of the Experiment 3 data samples
4.5. Experiment 4.

Experiment 4:

- Basketball dropping sounds were used as audio data.
- Multilayer Neural Network was trained with all the audio samples from folder Fold1 to Fold8, was tested with the same audio samples.
- The input layer of NN was 13 nodes, hidden layer 26 nodes, and output layer 2 nodes.
- The Mel Frequency Cepstral Coefficients were extracted as the features data, which contained 13 coefficients for each 3 second audio sample.
Purposes:

The purposes of the Experiment 4 are to train the Multilayer Neural Network with the 3 second audio samples instead of 6 second audio samples, and to compare the performance with the 6 second audio samples.

Result:

The achieved accuracy was 98% after training, the testing of the counting of the number of dropping sounds are as shown in the following table 4.1. The result of the training showed better accuracy of 98% compared with the accuracy of 95% of the ‘6 second audio samples’. But when tested with audio files, it did not show better output, only six testings were correct comparing to 12 were correct of the ‘6 second audio samples’.

<table>
<thead>
<tr>
<th>Nos:</th>
<th>File name</th>
<th>Actual number of dropping sound:</th>
<th>NN classifier output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>171121_223939.wav</td>
<td>60</td>
<td>64</td>
</tr>
<tr>
<td>2</td>
<td>171123_222534.wav</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>3</td>
<td>171124_220025.wav</td>
<td>60</td>
<td>59</td>
</tr>
<tr>
<td>4</td>
<td>171124_221412.wav</td>
<td>60</td>
<td>61</td>
</tr>
<tr>
<td>5</td>
<td>171125_133507.wav</td>
<td>60</td>
<td>61</td>
</tr>
<tr>
<td>6</td>
<td>171125_140219.wav</td>
<td>60</td>
<td>66</td>
</tr>
<tr>
<td>7</td>
<td>171125_150208.wav</td>
<td>60</td>
<td>61</td>
</tr>
<tr>
<td>8</td>
<td>171125_153724.wav</td>
<td>60</td>
<td>61</td>
</tr>
<tr>
<td>9</td>
<td>171125_160257.wav</td>
<td>60</td>
<td>64</td>
</tr>
<tr>
<td>10</td>
<td>171125_174356.wav</td>
<td>60</td>
<td>59</td>
</tr>
<tr>
<td>11</td>
<td>171125_181712.wav</td>
<td>60</td>
<td>61</td>
</tr>
<tr>
<td>12</td>
<td>171125_183752.wav</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>13</td>
<td>171125_215320.wav</td>
<td>60</td>
<td>59</td>
</tr>
<tr>
<td>14</td>
<td>171125_221114.wav</td>
<td>60</td>
<td>61</td>
</tr>
<tr>
<td>15</td>
<td>171126_110229.wav</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>16</td>
<td>171126_123253.wav</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>17</td>
<td>171126_142625.wav</td>
<td>60</td>
<td>60</td>
</tr>
</tbody>
</table>
Table 4.3 Testing the Experiment 4’s classifier with training audio files.

<table>
<thead>
<tr>
<th></th>
<th>171126_162454.wav</th>
<th>171127_211716.wav</th>
<th>171128_211311.wav</th>
<th>171128_221513.wav</th>
<th>171129_212319.wav</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>60</td>
<td>55</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>60</td>
<td>61</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>60</td>
<td>61</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>60</td>
<td>58</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Summary**

The machine learning algorithm can easily achieve the accuracy of 0.9 with the training of the basketball sound. It shows that the designed Multilayer is possible to be applied to the actual audio signal from the plantation. Due to unavailable of the good quality actual FFB dropping sound samples, the algorithm is pending for more promising testing.

The sound recording should be done with a recorder properly attached to the harvester and a follower to record the actual time of FFB dropping time tally with the recorder. The time is very important for the data samples preparation.
CHAPTER 5: CONCLUSIONS

5.1 Conclusion.
Palm oil in Malaysia is a main economic income to the nation. The performance in Malaysia palm oil industry will directly affect the nation economy and government income.

The main unsolved problem in Malaysia’s palm oil industry is the shortage of labor, especially labor for harvesting the Fresh Fruit Bunch of palm oil. To encounter the labor shortage of harvesters, some level of automation is necessary. An IoT device can be used to automate harvesting activities and to provide real time information about the quantity of harvested FFB and its GPS location. It will optimize the harvesting process. The real time information concluded of number of harvested FFB, where the FFB were harvested, and who was the harvester.

This project is to design an audio-based machine learning algorithm for the IoT to record the number of fruit bunches be harvested by harvesters, which are done manually by the plantation supervisors.

The built audio-based machine learning algorithm was tested with the basketball dropping sound, the result showed that it is possible to build the final audio-based machine learning algorithm for the actual audio signal from the plantation. The training of the audio-based learning algorithm with actual audio samples was not work out, due to the unavailability of actual FFB dropping audio samples.

The collection of actual FFB dropping audio samples is not an easy task. The collection of the audio samples must be carried out to further testing the audio-based machine learning.
5.2 Future work.

For future work, the main work should be focused on the collection of sound samples at the oil palm plantation. More studies need to be made on how to record the sound samples at the plantation. With the availability of the audio samples, the samples can be fed to the machine learning algorithm to create and to tweak the classifier model.

Some of the future tasks are suggested as the following:

- Collection of FFB audio samples: The better way of collection of audio samples is to attach the recorder to the harvester, and a follower to note down the dropping time accurately.
- Processing of the audio samples: It is better to have a huge quantity of audio samples, may be in the quantities of a few thousand. Every sample should be checked by hearing and wave diagram inspection.
REFERENCES


Department Of Statistical Malaysian (DOSM)a, Official Portal, 2016. Selected Agriculture Indicator, Malaysia, 2016. [online] Available at :


Sam Francis, 20 June 2017. Raspberry Pi could help farmers manage crops better and more cheaply that current system, say scientists. [online] Available at: https://roboticsandautomationnews.com/2016/06/20/raspberry-pi-helps-farmers-manage-crops-better-and-more-cheaply-than-current-systems-say-scientists/5823/ [Accessed on 21 July 2017]


TongKe, F., 2013. Smart agriculture based on cloud computing and IOT. Journal of Convergence Information Technology, 8(2). Available at: http pdfs.semanticscholar.org/62ee/b701c40626811a1111ca5d1db37650f1ea0b.pdf [Accessed on 28 June 2017]
APPENDICES

APPENDIX A: Crude palm oil milling process

The following figure shows the processes of oil palm FFB, from the plantation to the mill and the processes to extract the crude palm oil.

Figure A.1 Crude palm oil milling process. (Plantation.com, 2015)
APPENDIX B: Oil palm harvesting.

The following are the image of the mechanization cutter, oil palm FFB harvesting machine and FFB on the tree. The mechanization cutter is not suitable for tree taller than 5 meters, and the harvesting machines are not efficient as human harvesters.

Figure B.1 A harvester harvests oil palm FFB with mechanization cutter. (Pemandu.gov.my, 2017)

Figure B.2 A harvesting machine harvests oil palm FFB. (Researchgate.net, 2017)
Figure B.3 The oil palm FFBs with different ripeness. (Agrifarming.in, 2017)

Figure B.4 An oil palm FFB harvesting machine. (Crendon, 2017)
APPENDIX C: Implementation

C.1 The Docker container.

The Docker container was built with the docker ‘run’ command such as the following.

```bash
$ docker run -it -p 8888:8888 --MyProject -v ~/documents/notebook:notebooks
```

With the above run command, a container with name ‘MyProject’ was created. The ‘-it’ option created an iterative terminal in the container. The ‘-p’ option matched the port 8888 of docker localhost to the pc port 8888, to enable the view in the web browser. The ‘-v’ option created a volume, which was named notebook as the storage external to the container.

After you have started the container with the docker toolbox’s Kitematic, it will ask you to copy and paste the link, as show in the following.

You can copy and paste the link to the browser and replace the localhost with the Ip ‘192.168.99.100’, the Jupyter notebook is ready in the browser as the following.
When you click the ‘new’ button, it will drop down to show the list you can choose as the following.

You can open the Jupyter notebook by clicking the ‘python 2’ button. Other programs can be chosen are ‘Text File’, ‘Folder’ and ‘Terminal.

You can install new python library such as the Librosa in the Terminal as shown in the following.
C.2 Implementation.

C.2.1 FfbSound.py

```python
# -*- coding: utf-8 -*-

from __future__ import print_function
import scipy.io
import scipy.io.wavfile
import os
import numpy as np
from scikit_talkbox_features import mfcc
from matplotlib.pyplot import specgram
from pylab import*
import librosa
import glob
import os
import tensorflow as tf

def readTxt(txt):
    f = open(txt, "r")
    myList =[]
    for line in f:
        myList.append(line)
    f.close()
    return myList

def create_fft(fn, txt, Label):
    myList = readTxt(txt)
    sample_rate, X = scipy.io.wavfile.read(fn)
    print(X.shape)
    X=X[:,0]
    base_fn, ext = os.path.splitext(fn)
    for count, line in enumerate(myList):
        print(line)
        start=int(line)*sample_rate
        stop=int(line)*sample_rate+sample_rate
        fft_features = abs(scipy.fft(X)[start:stop])
        print(fft_features.shape)
        data_fn = base_fn + "_" + str(count) +"_" + Label + ".fft"
        np.save(data_fn, fft_features)

def create_wav(fn, txt, Label, folder):
    myList = readTxt(txt)
    sample_rate, X = scipy.io.wavfile.read(fn)
    print(X.shape)
    X=X[:,0] # only consider one channel only
    fn=fn.split('/')[1]
    base_fn, ext = os.path.splitext(fn)
    for count, line in enumerate(myList):
        print(line)
        start=int(line)*sample_rate
        stop=int(line)*sample_rate+sample_rate
        X_sample=X[start:stop]
        print(X_sample.shape)
        data_fn = "Ffb-Sound-Data/Fold="+folders+"/"+base_fn + "_" + str(count) +"_" + Label + ".wav"
        scipy.io.wavfile.write(data_fn, 44100, X_sample)
```

```
def extract_feature(file_name):
    X, sample_rate = librosa.load(file_name)
    
    # stft
    stft = np.abs(librosa.stft(X))
    
    # mfcc
    mfccs = np.mean(librosa.feature.mfcc(y=X, sr=sample_rate, n_mfc=40).T, axis=0)
    
    # chroma
    chroma = np.mean(librosa.feature.chroma_stft(S=stft, sr=sample_rate).T, axis=0)
    
    # spectral_contrast
    contrast = np.mean(librosa.feature.spectral_contrast(S=stft, sr=sample_rate).T, axis=0)
    
    tonnets = np.mean(librosa.feature.tonnetz(y=librosa.effects.harmonic(X), sr=sample_rate).T, axis=0)

    return mfccs, chroma, contrast, tonnets

def parse_npy_features(parent_dir, sub_dirs, file_ext='*.wav'):
    features, labels = np.empty((0,193)), np.empty(0)
    for label, sub_dir in enumerate(sub_dirs):
        print(f'label: {label} % (label))
        print(f'sub dir: {sub_dir}')
        for fn in glob.glob(os.path.join(parent_dir, sub_dir, file_ext)):
            print(f'extract file: %s % (fn))
            try:
                # util.print_wave_info(fn)
                print(f'extract file: %s % (fn))
                except Exception as e:
                    print(f'[Error] unhandle npy file. %s % (e))
                continue
                try:
                    mfccs, chroma, mel, contrast, tonnets = extract_feature(fn)
                    except Exception as e:
                        print(f'[Error] extract feature error. %s % (e)
                    continue
                ext_features = np.hstack([mfccs, chroma, mel, contrast, tonnets])
                features = np.vstack([features, ext_features])
            fn, ext = os.path.splitext(fn)
            labels = np.append(labels, fn.split('_')[3])
            print(np.array(labels))

    return np.array(features), np.array(labels, dtype = np.int)

def one_hot_encode(labels):
    n_labels = len(labels)
    print(np.array(labels))
    n_unique_labels = len(np.unique(labels))
    print(n_unique_labels)
    one_hot_encode = np.zeros((n_labels, n_unique_labels))
    one_hot_encode[n_labels, labels] = 1
    print (one_hot_encode)
    return one_hot_encode

def extract_mfcc(file_name):
    sample_rate, X = scipy.io.wavfile.read(file_name)
    print ([X.shape)
    ceps, mspec, spec = mfcc(X)
    print (ceps.shape)
    X=np.mean( cepe, axis=0 )
    print ([X.shape)
    return X
C.2.2 Train Neural Network

```python
39
In [1]:
from FBSound import *
import tensorflow as tf

In [2]:
# To load the audio features from features.npy file
tr_features= np.load("/13dr_features.npy")
tst_features= np.load("/13dr_features.npy")
ttr_features= np.load("/13e_features.npy")
tst_features= np.load("/13e_features.npy")

In [3]:
x= tf.placeholder(tf.float32, [None, 22])  # 28x28
ys= tf.placeholder(tf.float32, [None, 2])

# input layer, In-Add layer (xs, 13, 26, activation_function=tf.nn.relu)
W1= tf.Variable(tf.random_normal([13, 26]), name= 'W1')
bias1= tf.Variable(tf.zeros([1, 26]), name= 'b1')
Wx_plus_b1= tf.nn.relu(Wx_plus_b1, 1)

# output layer, prediction = add layer(12, 26, activation_function=tf.nn.softmax)
W2= tf.Variable(tf.random_normal([26, 2]), name= 'W2')
bias2= tf.Variable(tf.zeros([1, 26]), name= 'b2')
Wx_plus_b2= tf.nn.relu(Wx_plus_b2, 1)

cross_entropy= tf.reduce_mean(tf.log(prediction))
loss= tf.train.GradientDescentOptimizer(0.1).minimize(cross_entropy)
```
C.2.3 Neural Network Classifier

```python
if int(tf.__version__).split('.')[0] < 12 and int(tf.__version__).split('.')[1] < 0:
    init = tf.initialize_all_variables()
else:
    init = tf.global_variables_initializer()

with tf.Session() as sess:
    for i in range(5000):
        # batch size = nn.train.next_batch(n)
        sess.run(train_step, feed_dict={xs: tr_features, ys: tr_labels})
        if i % 50 == 0:
            print("compute_accuracy(\n        xs=features, ys=labels, sess, prediction, xs, ys )")

        save_path = saver.save(sess, "trialid")
    print("Save to path: ", save_path)
```

```python
In [1]: from fb掌声 import *

In [2]: sample_rate, X = scipy.io.wavfile.read('Basketball_sound/171124_220023.wav')
    # sample_rate, X = scipy.io.wavfile.read('I.wav')

    print(X.shape)
    print(len(X))
    print(len(X)/sample_rate)
    soundlen=len(X)/sample_rate

count = 0
previous = 0

# util.print_wave_info(fn)
xs = tf.placeholder(tf.float32, [None, 13]) # 28x28
ys = tf.placeholder(tf.float32, [None, 2])

# input layer, 11+add_layer(xs, 13, 26, activation_function=tf.nn.relu)
Weights_1 = tf.Variable(tf.random_normal([13, 26]), name='w1')
biases_1 = tf.Variable(tf.zeros([1, 26]) + 0.1, name='b1')
Wx_plus_b_1 = tf.matmul(xs, Weights_1) + biases_1
output_1 = tf.nn.relu(Wx_plus_b_1,)

# output layer, prediction = add_layer(11, 26, 2, activation_function=tf.nn.softmax)
Weights_2 = tf.Variable(tf.random_normal([26, 2]), name='w2')
biases_2 = tf.Variable(tf.zeros([1, 2]) + 0.1, name='b2')
Wx_plus_b_2 = tf.matmul(output_1, Weights_2) + biases_2
prediction = tf.nn.softmax(Wx_plus_b_2,)

with tf.Session() as sess:
    saver = tf.train.Saver() # Renaming in jupyter will cause "key w-1 not found in checkpoint" error.
    saver.restore(sess, "trialid")

    #x1=extract_mfcc(fn)
    #k=wav.reshape(1,13)
    #print (sess.run(prediction, feed_dict={k:x1}))

for i in range(sample_rate*6, len(X)/sample_rate*6):
    signal = X[i:i+sample_rate*6])
    print (signal.shape)
    signal = np.mean( signal, axis=0 )
    signal = signal.reshape(1,13)
    out = sess.run(prediction, feed_dict={xs:signal})

    if out[0,1] < 0.7:
        if previous == 0:
            print("one")
        previous = 1
        count = count +1
    else:
        previous = 0

print (count)
```