

**Software Implementation For Fruits And Vegetables Quality Determination**

by

**SUHAILI SHAZREENA BINTI SUDIN**

Dissertation submitted in partial fulfillment of  
requirements for the  
Bachelor of Engineering (Hons)  
(Electrical and Electronics Engineering)

**JUNE 2009**

**Universiti Teknologi PETRONAS  
Bandar Seri Iskandar  
31750 Tronoh  
Perak Darul Ridzuan**

**CERTIFICATION OF APPROVAL**

**Software Implementation For Fruits And Vegetables Quality Determination**

by

**Suhaili Shazreena Bt. Sudin**

A project dissertation submitted to the  
Electrical & Electronics Engineering Programme  
Universiti Teknologi PETRONAS  
in partial fulfillment of the requirement for the  
BACHELOR OF ENGINEERING (Hons)  
(ELECTRICAL & ELECTRONICS ENGINEERING)

Approved by,



---

(Dr John Ojur Dennis)

UNIVERSITI TEKNOLOGI PETRONAS  
TRONOH, PERAK

June 2009

## CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.



---

SUHAILI SHAZREENA BT SUDIN

## ABSTRACTS

Human always use their senses to detect the quality of the vegetables and fruits so then they will know how fresh the vegetables or fruits they want. However, human senses can only detect the freshness at a certain level making it hard for us to know how fresh the vegetables and fruits are. Even though the physical appearances of the vegetables and the fruits can easily indicate how fresh they are but it can be very deceiving sometimes as the fruits can be rotten on the inside. The objectives of this project are to analyze the waveforms obtained from gas emitted by fruits and vegetables in certain condition and use neural network to classify the readings from the sensors into three different conditions which are fresh, slightly fresh and rotten. C program is also constructed as another way to classify the fruits and vegetables into three different conditions as well. The experiment is conducted by obtaining the readings from the sensors and analyzed the readings using neural network. Based on the simulation from the neural network, the network will identify the conditions of the fruits and vegetables based on the readings from the sensors. C program also works similar ways like the neural network. By comparing both neural network and C program, neural network is able to determine the freshness of the fruits and vegetables with the accuracy of 99%. Unlike neural network, C program can only determine the condition of the fruits and vegetables based on the sensor range which it is not sensitive to the changes in the readings. Therefore, the best method for determining the freshness of the fruits and vegetables is by using neural network.

## **ACKNOWLEDGEMENT**

This project would not be possible without the assistance and the guidance from my supervisor, Dr. John Ojur Dennis. Therefore, I would like express my utmost gratitude to him for his valuable input and guidance throughout this final year project.

I would also like to express my sincere gratitude to my partner for this project, Zizie Zulaiqa Ahmad for helping me in understanding this project better so then I will be able to fully grasp the main idea for this project. Not just that, I am fortunate to have her as my partner for this project since we are able to communicate very well with each other, and also help each other a lot. Therefore, this project will not be successful without her contribution.

I would also like to thank my friend, Tan Mao Cheng for teaching me neural network since he is one of the persons I know who understands how to construct the neural network and also how to use MATLAB Neural Network Toolbox. Without his help, I may get the whole idea about neural network completely wrong, making the whole results absolutely invalid.

Another person I would like to express my gratitude is my Computer System Architecture lecturer, Mr. Patrick Sebastian for teaching me on how to access the file from Microsoft Excel with using MATLAB as it will simplify the data input process as neural network requires a lot of data to get an accurate result.

Last but not least, I would like to apologize if any party was inadvertently excluded from being mentioned above and would like to thank all parties who were involved making this final year project a success.

# TABLE OF CONTENT

CERTIFICATE OF APPROVAL .....	ii
CERTIFICATE OF ORIGINALITY .....	iii
ABSTRACTS.....	iv
ACKNOWLEDGEMENTS .....	v
LIST OF FIGURE.....	viii
LIST OF TABLES .....	ix
CHAPTER 1: INTRODUCTION .....	1
1.1 Background Of Study.....	1
1.2 Problem Statements.....	1
1.3 Objectives.....	2
1.4 Scope Of Study.....	2
1.5 Feasibility .....	2
1.6 Relevancy .....	3
CHAPTER 2: LITERATURE REVIEW .....	4
2.1 Artificial Neural Network (ANN).....	4
2.2 Character Recognition Using Back Propagation Network.....	6
2.3 Introduction To C.....	12
CHAPTER 3: METHODOLOGY .....	15
3.1 Project Workflow .....	15
3.2 Project Activities.....	20
3.3 Tools And Software .....	23
CHAPTER 4: RESULTS AND DISCUSSIONS.....	24
4.1 Designing Back Propagation Network.....	24
4.2 Simulation Of The Back Propagation Network .....	27
4.3 Alternative Way To Determine The Condition.....	29
4.4 Comparison Between Neural Network And C Program .....	31
CHAPTER 5: CONCLUSIONS AND RECOMMENDATION .....	33
5.1 Conclusions .....	33
5.2 Recommendations .....	34

References .....	35
Appendices .....	36

## LIST OF FIGURES

Figure 1: Overview of Artificial Neural Network (ANN) .....	5
Figure 2: Letter 'A' in Boolean values.....	6
Figure 3: Letter 'A' in Boolean values with the presence of noise.....	8
Figure 4: Noisy letter.....	9
Figure 5: The letter that the network picked correctly .....	9
Figure 6: Architecture of back propagation network .....	10
Figure 7: Project flowchart.....	19
Figure 8: Training and performance plot for the condition.....	26
Figure 9: Network architecture for condition determination .....	27
Figure 10: Simulation error plot.....	28
Figure 11: Flowchart for condition determination C program.....	30
Figure 12: Condition determination in C program.....	31



## **LIST OF TABLES**

<b>Table 1: List of the acronyms for the training algorithm .....</b>	<b>10</b>
<b>Table 2: Result of transient stability analysis in generator .....</b>	<b>16</b>
<b>Table 3: Olfactory evaluation for the fruits and vegetables .....</b>	<b>20</b>
<b>Table 4: Sensor range for fresh, slightly fresh and rotten condition .....</b>	<b>21</b>

# **CHAPTER 1**

## **Introduction**

### **1.1 Background of Study**

Nose is one of the human organs that can detect the smell and is also plays important role in tasting things. The nose is the main gate of the respiratory system which is our body system's breathing.

The nose detects the smell by relying on the hidden parts inside the human nasal cavity and head. The olfactory epithelium which is located on the roof of nasal cavity is the nose part which is responsible for smelling. The olfactory epithelium contains special receptors which are very sensitive to the odor molecules that travel via air. These receptors are very small and have at least 10 million of receptors in the nose. There are hundreds of different odor receptors that have the ability to sense specific odor molecules. The brain recognizes 10,000 different smells by interpreting the receptors. Once the receptors are stimulated, the signals will travel to the olfactory bulb through olfactory nerve. The signals will be sent to the other parts of the brain to interpret the smells which the humans are familiar with [1].

### **1.2 Problem Statement**

Certain smells cannot be detected by the human nose even though the nose can detect more than 10,000 different smells. This is due to insufficient amount of olfactory epithelium in the nose. Sometimes, the smells are so close to each other, making it hard for the brain to determine whether the fruits or vegetables are fresh or rotten. Therefore, humans need a device that can detect the freshness of fruits and vegetables which is beyond the capability of the human nose.

Utilizing various types of metal oxide gas sensors will enable the acquisitions of the right input signals of the odor. However, the presences of noise in the output signals which are from the sensors require the use of neural networks to obtain information about fruits and vegetables freshness.

### **1.3 Objectives**

- To find the similarities between three waveforms obtained from the three sensors which are operated at different temperature
- To analyze the results obtained from the inputs using neural network
- To classify the fruits and vegetables in terms of degree of freshness deduced from the neural network analysis
- To use C program as another way to determine the freshness of the fruits and vegetables besides neural network

### **1.4 Scope of Study**

- To study the characteristic of the waveforms obtained from the three sensors
- To study back propagation method used in pattern recognition

### **1.5 Feasibility**

Based on the results obtained from the neural network, it will be easily to tell whether the fruits or vegetables are still fresh or has become rotten. Therefore, this project is feasible to study to detect the quality of the fruits and vegetables in terms of its freshness.

## **1.6 Relevancy**

This project is relevant to the food industry such as canned food manufacturing where the quality of the fruits and vegetables needed to be clarified before the canning process begins. Besides that, this project is also useful for wine brewing industry where the fruits must be fresh before they start brewing the wine. This project can also be used by the consumers to detect the freshness of the fruits and vegetables before they purchased it.

## **CHAPTER 2**

### **Literature Review**

#### **2.1 Artificial Neural Network (ANN)**

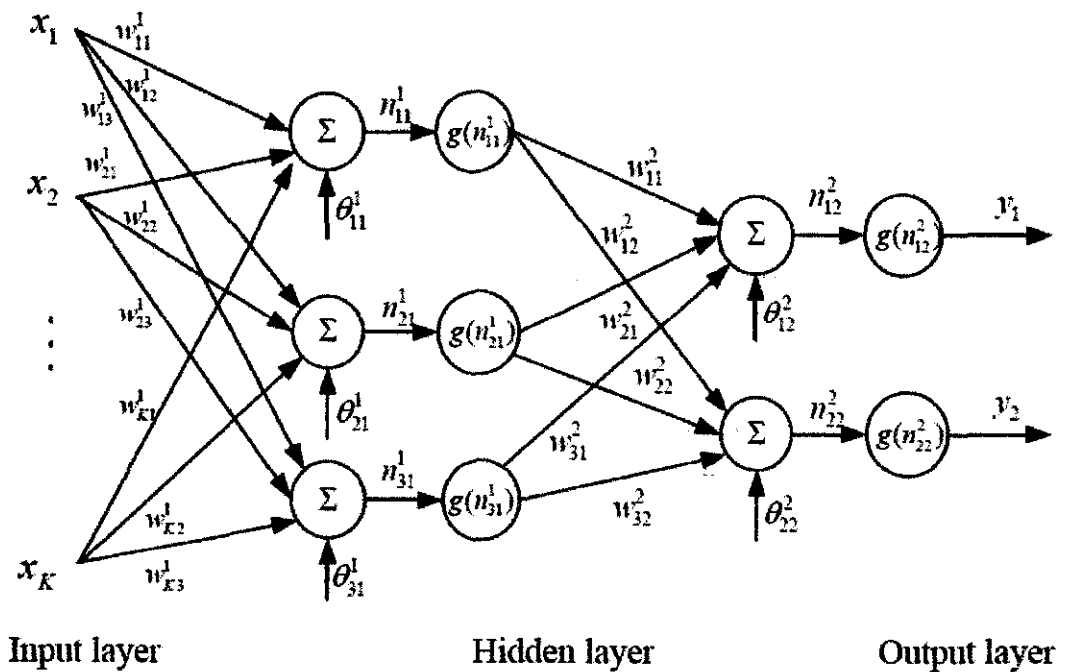
Artificial Neural Network (ANN) is a mathematical or computational model which imitates biological neurons. ANN can adapt the changes of the structure when external and internal information flows into the network during the learning phase. ANN is used to model complex relationships between the inputs and the outputs to find the similarities of the data provided [2].

The objectives of ANN are to imitate human ability in adapting changing circumstances and the current environment. This will depend solely on the ability to learn from the events which have happened in the past and able to implement this in the future situations. In other words, ANN is using past events as a part of decision making process.

ANN consists of three layers where there are input layer, hidden layer and output layer. There are many nodes which work as processing units inside the hidden layer. These processing units are known as neurons and these neurons work similarly like the neurons in our brain. Each neuron has its own function which is used to determine the output of the neuron from the given input. Changing the local parameters will change the neuron function. The neurons which are in the information processing system process the information and transmitted by the connecting links. The output signal is obtained by applying activations to the net input [3]. Figure 1 [4] summarized the neural network architecture in mathematical model form.

There are a lot of hardware and software implementations based on Artificial Neural Network such as [3]:

- Hopfield net
- Multilayer Perceptron
- Self-Organizing Feature Map
- Learning Vector Quantization
- Radial Basis Function
- Cellular Neural
- Adaptive Resonance Theory (ART)
- Counter Propagation
- Back Propagation
- Neo-Cognitron



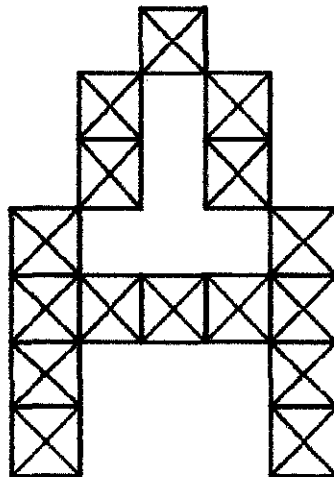
**FIGURE 1: Overview of Artificial Neural Network (ANN)**

## 2.2 Character Recognition Using Back Propagation Network

Back propagation network is widely used in pattern recognition. Therefore, this network will be used to design and train to recognize 26 letters of the alphabets.

This can be done by representing the letters in 5-by-7 grid of Boolean values. The image produced will not be perfect due to the presence of noise so the network must be able to handle noise. The network will recognize the pattern by receiving 35 Boolean values as 35-element input vector and will respond with 26-element output vector. The network will respond when 1 is in the position of the letter being represented in the network and other values in the output vector should be 0. Figure 2 shows the letter without the presence of noise and Figure 3 shows the letter with the presence of noise. The letters are shown in Boolean values. Firstly, alphabet definitions and their targets are loaded into the program.

```
[alphabet, targets] = prprob;
```



**FIGURE 2: Letter 'A' in Boolean values**

The network will be trained on both ideal and noisy vectors. The training starts with the ideal condition and it will stop until it has achieved low sum squared error. The same process is also repeated for condition with the presence of noise. 10 sets of vectors are used for the training and the vectors in ideal condition and vectors with the presence of noise are trained at the same time. This is to ensure that the network is able to

maintain its ability to classify the ideal input vectors. The network will be trained for a longer period and the number of neurons in the hidden layer needs to be increased if the network requires high accuracy.

#### **Creating two-layer network:**

```
Net = newff(alphabets,targets,10,{'logsig','logsig'});
```

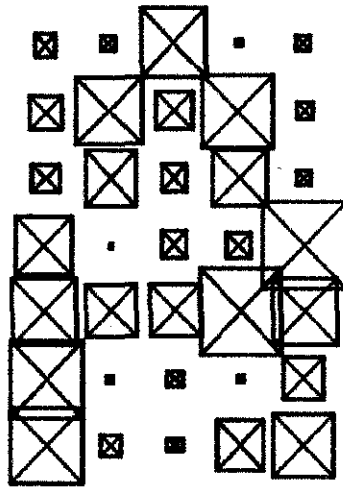
#### **Training without noise:**

```
P = alphabets;
T = targets;
net.performFCN = 'sse';
net.trainParam.goal = 0.1;
net.trainParam.show = 20;
net.trainParam.epochs = 5000;
net.trainParam.mc = 0.95;
[net,tr] = train (net,P,T);
```

#### **Training with noise:**

```
netn = net;
netn.trainParam.goal = 0.6;
netn.trainParam.epochs = 300;
T = [targets targets targets targets];
P1 = P;
[R,Q] = size (P);
for pass = 1:10
P = [P1,P1, (P1 + randn(R,Q)*0.1), (P1 + randn(R,Q)*0.2)];
[netn,tr] = train(netn,P,T);
end
```

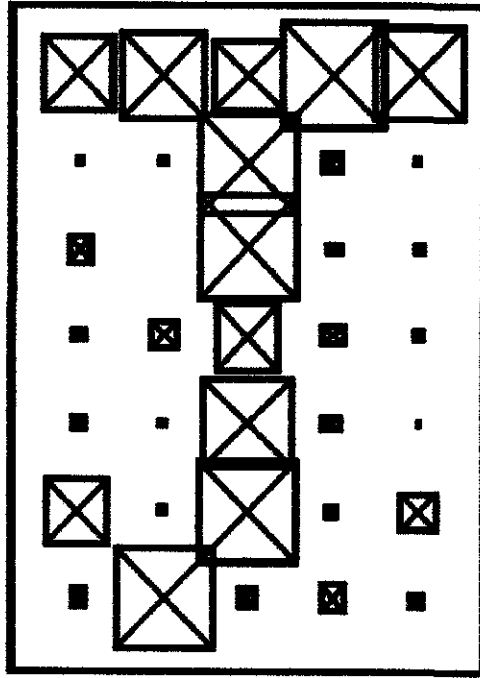




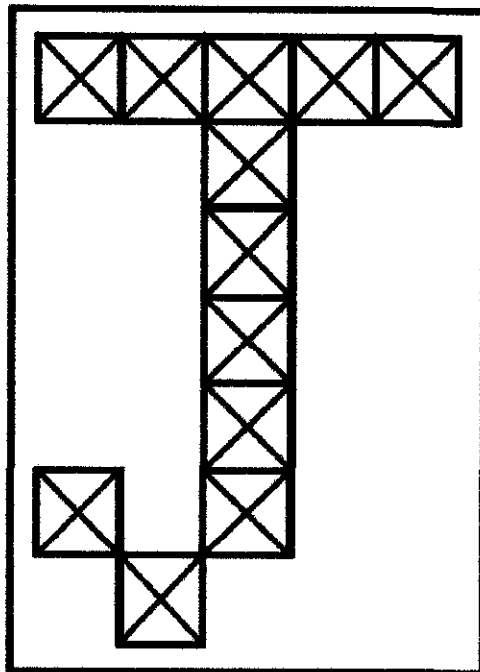
**FIGURE 3: Letter 'A' in Boolean values with the presence of noise**

The network can be tested by creating an input vector with the noise and present it to the network. The network can produce an accurate result if the network is able to recognize the letter even with the presence of noise. Figure 4 shows the noisy letter and Figure 5 shows the letter that network picked correctly.

```
noisyJ = alphabet(:,10)+randn(35,1)*0.2;  
plotchar(noisyJ);  
A2 = sim(net,noisyJ);  
A2 = compet(A2);  
answer = find(compet(A2) == 1);  
plotchar(alphabet(:,answer));
```

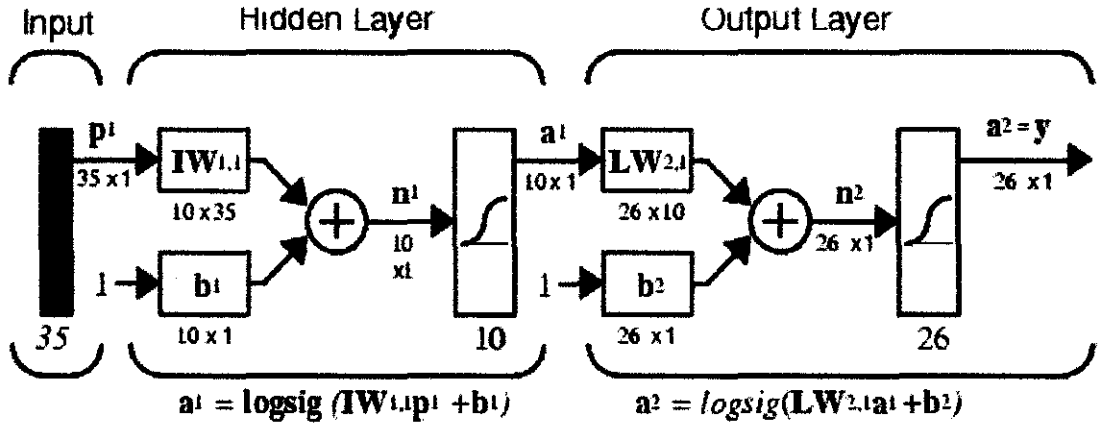


**FIGURE 4: Noisy letter**



**FIGURE 5: The letter that the network picked correctly**

Back propagation network also implies the same architecture just like the neural network in mathematical model form shown in Figure 1 where it consists of input layer, hidden layer and output layer. The input layer will provide the data to the hidden layer and the data analysis is done in the hidden layer. When the data analysis is done, the neurons in the hidden layer will send the analyzed data to the output layer. The process is shown in Figure 6.



**FIGURE 6: The architecture of back propagation network**

In back propagation network, the training relies on which training algorithm will give the fastest result for the given problem. The speed of the training depends on a few factors such as the complexity of the problem, the number of data in the training set, the number of weights and biases in the network, the error goal, and whether the network is used for pattern recognition or function approximation. Table 1 shows the list of the acronyms used in the training.

**TABLE 1: List of the acronyms for the training algorithm**

Acronym	Algorithm	Abbreviation
LM	trainlm	Levenberg-Marquardt
BFG	trainbfg	BFGS Quasi-Newton
RP	trainrp	Resilient Backpropagation
SCG	trainscg	Scaled Conjugate Gradient

CGB	<code>traincgb</code>	Conjugate Gradient with Powell/Beale Restarts
CGF	<code>traincgf</code>	Fletcher-Powell Conjugate Gradient
CGP	<code>traincgp</code>	Polak-Ribière Conjugate Gradient
OSS	<code>trainoss</code>	One Step Secant
GDX	<code>traingdx</code>	Variable Learning Rate Backpropagation

Generally, if the training algorithm is used in function approximation problem, Levenberg-Marquardt algorithm has the fastest convergence. This can be very noticeable if the network needs high accuracy. In most cases, `trainlm` is able to obtain lower mean square errors compared with any other algorithm. However, `trainlm` will decrease as the number of weights in the network increases and this algorithm is not suitable for pattern recognition as it has a poor performance in pattern recognition problems. Besides that, this training function requires larger storage compared to other algorithms. The storage can be reduced by adjusting the `mem_reduc` parameter, but it will increase the execution time.

For pattern recognition, `trainrp` function is the most suitable for this kind of situation due to having the fastest algorithm in pattern recognition problems. However, it does not perform well in function approximation problems and the performance of this algorithm is decreased when the error goal is reduced. The memory requirement for this function is relatively small compared with other training algorithm.

The `trainscg` function seems to perform very well in a wide variety of problems especially with the networks that have large number of weights. The SCG algorithm is also as fast as LM algorithm in function approximation problems which it is a lot faster in the network that has large network and it is also as fast as `trainrp` function in pattern recognition problems. The best part about this function is it does not degrade as fast as `trainrp` function in terms of performance when the error is reduced. Putting the memory requirement as a part of the consideration, this algorithm has a modest memory requirement.

The `trainbfg` is similar to `trainlm` function in terms of performance. It does not need as much storage as `trainlm` but the computation will increase with the size of the network as the equivalent of a matrix inverse needs to be computed for each iteration. The size of storage is the same as the storage for `trainrp`.

`traingdx` function is slower compared with other algorithms and has the same storage like `trainrp` function. Therefore, this function is suitable for problems that require slow convergence. Using this function will prevent the training from converging too quickly and overshooting when minimizing the error on the validation set [5].

### 2.3 Basic Structure of C Language

C is a general-purpose and structured programming language where the instruction set is compatible with the human language and human thought process. The instructions in C resemble algebraic expressions, implemented with the English words such as *if*, *else*, *for*, *do* and *while*. C has an additional feature which allows it to be used in low-level programming to bridge the gap between the machine language and conventional high-level languages. C is very flexible, allowing it to be used in system programming such as writing operating systems and application programming such as writing a program to solve complicated mathematical equations or writing a program to bill the customers.

C has a small instruction set and it can enhance the basic instructions by implementing the extensive *library functions*. C also has the flexibility for the user to write their own additional library functions, thus extending the capabilities and the features of the language.

C compilers are widely available for computers and very compact since these compilers are able to generate object programs which are small and highly efficient compared to other compilers for other high-level languages. It is easy to create a new program but the interpreters are not so efficient.

C program is highly portable because it relegates most computer-dependant features to its own library functions. Therefore, every version of C will come together with its own set of library functions, written with the particular characteristics of the host computer. The library functions are standardized and the individual library functions can be accessed in the same manner from one version of C to another version. Therefore, C program is convenient because it can be executed in different computers with minor or no alteration.

Generally, C program has one or more modules which are called functions. One of these functions must be declared as `main`. Other functions need to be defined separately whether it is ahead or after `main`. Each function must have a function heading, which has the function name and followed by the list of arguments which are enclosed in parentheses. The arguments must be listed if the arguments are included in the heading and it must also have compound statement, which is the remainder of the function.

The arguments are symbols which represents the information that passed between the function and other parts of the program. As for compound statement, it will be enclosed within a pair of curly braces (`{}`). Inside the braces, it can have expression statements and other compound statements. Therefore, the compound statements can be nested and must be end with the semicolon (`;`).

Comment of the program can be included in the program as long as it is replaced within the delimiters `/*and*/`. These comments are useful to identify the feature or explained certain functions. Example below summarized C program structure [7].

```
/* program to calculate the area of a circle*/      /*Title (comment)*/
#include<stdio.h>                                  /*Library file access*/

main()                                             /*Function heading*/
{
```

```
float radius, area;                /*Variable declarations*/

printf("Radius = ?");              /*Output statement (prompt)*/
scanf("%f", &radius);             /*Input statement*/
area = 3.1414159*radius*radius;    /*Assignment statement*/
printf ("Area = %f", area);       /*Output assignment*/
}
```

## **CHAPTER 3**

### **Methodology**

#### **3.1 Project Work Flow**

##### *Research on related work*

Research is done by finding technical papers and journal related to the scope of study. This is essential to get a basic understanding on how similar project is conducted in terms of how they set up the experiment, the conditions of the experiment that are needed to be considered, how they do the measurement, how they implement neural network in their analysis, how they present their analysis and how they conclude the experiment.

##### *Understand how neural network works and how to create it*

Understanding how neural network works is important to make the analysis of the data becomes easier. Since there are so many methods to implement neural network, it is crucial to find the best method for the analysis as not all method will produce the expected result and some method must fulfill certain conditions. Based on research done previously, the possible methods that can be used for the experiment will be Self-Organizing and Learning Vector Quantization Network and Back Propagation Network.

Self-Organizing and Learning Vector Quantization works by identifying sets of data and will group the data based on the specific condition. This network is suitable for this project but it is only meant for the data that remain constant in any condition, making this network unable to adapt the changes in data. Therefore, back propagation network is used instead because back propagation network is able to classify the data even if there are changes in data.



Once the method has been chosen, the network needs to be created to get a better picture on how to use neural network in this project. At this stage, the actual readings from the sensors have not been obtained yet so raw data is created to get a full understanding about neural network. The network is created based on the raw data shown in Table 1.

**TABLE 2: Result of transient stability analysis in generator**

<b>Fault clearing time (s)</b>	<b>Machine 1</b>	<b>Machine 2</b>
0.5	0	1
1.0	0	1
1.5	0	1
2.0	0	1
2.5	0	1
3.0	0	1
3.5	0	1
4.0	0	1
4.5	0	1
5.0	0	1
5.5	0	1
6.0	0	1
6.5	0	1
7.0	0	1
7.5	0	1
8.0	0	1
8.5	0	1
9.0	0	1
9.5	0	1
10.0	0	1

The network for this raw data is created by indicating whether the machine 1 and machine 2 is stable or not based on the specific fault creating time. From this data, 10 readings are used for the training and 10 readings are used for the simulation. This is the initial step to understand on how to create training and performance plot and simulate the network. By creating a network with this raw data, it will be easy to understand how to manipulate the network and training parameters so then these parameters can increase the accuracy of the network.

#### *Constructing and simulating the circuit*

Before constructing the circuit, which type of sensors will be used for this project is decided. Once the type of sensors has been decided, circuit modeling is done to get a clear idea how the sensor operates and to identify what is the suitable resistor values for the sensors. After the simulation is successful, the circuit for connecting the sensor is constructed.

#### *Obtaining the waveforms using interface*

The waveform is captured from the sensors using an interface which is connected to the computer. By obtaining the waveforms, the readings of the waveforms can be recorded so then the readings can be used for analysis using neural network.

#### *Analyzing the reading using neural network*

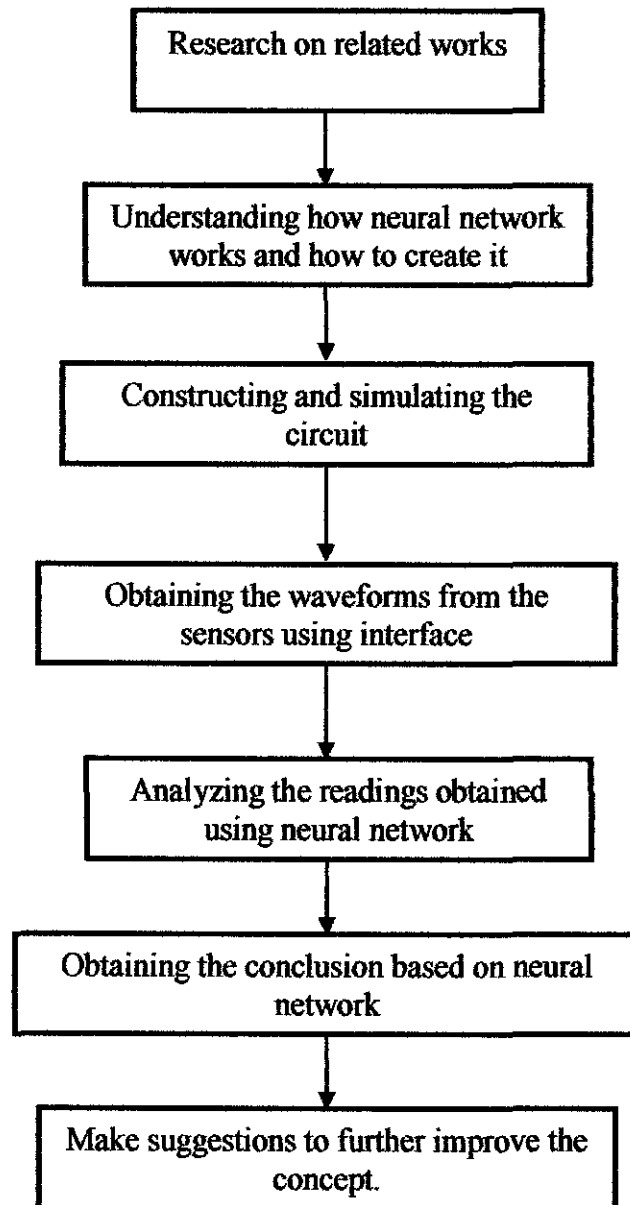
The data is analyzed by classifying the data into fresh and rotten condition data. Once the data has been classified, the neural network is created based on these data obtained from the experiment.

### *Obtaining the conclusion based on neural network*

The conclusion can be obtained based on the neural network analysis. This can be done by identifying the accuracy of the network and the simulation error produced during the simulation.

### *Make suggestions to improve the concept*

If the network does not give an accurate result, an improvement is needed to be done for the network. The improvement can be done by identifying the factors that affect the accuracy of the network. Once the cause has been identified, an improvement can be made to produce a better result. Figure 7 summarized the overall process flow of the project in a form of flowchart.



**FIGURE 7: Project Flow Chart**

### 3.2 Project Activities

#### *Stage 1: Collecting the data*

Samples of the waveforms for each condition (fresh and rotten) are taken for analysis.

#### *Stage 2: Decide what method will be used for building the network*

In this project, back propagation network will be used because back propagation network is widely used for pattern recognition.

#### *Stage 3: Decide how the network determines the result*

The values of 1 and 0 are used to determine the freshness of the fruits and vegetables. Therefore, the result will be classified as below:

1 - fresh

0 - rotten

The conditions of the fruits and vegetables can be categorized into different condition. Table 3 [6] summarized the conditions and the results.

**TABLE 3: Olfactory evaluation for the fruits and vegetables**

<b>Olfactory evaluation</b>	<b>Results</b>
Fresh	Accepted
Rotten	Rejected

#### *Stage 4: Determine the range for all sensors*

The range for the sensor is important for creating C program. The range of the sensor is based on the readings obtained from the sensors. The range for fresh condition is based on the reading taken on the first day and the range for rotten condition is based on the

reading taken on Day 11. Table 4 shows the sensor range for both fresh and rotten condition. The range for each sensor is based on the minimum and the maximum values of the sensors.

**Table 4: Sensor range for fresh, slightly fresh and rotten condition**

<b>Fresh (V)</b>	<b>Slightly fresh (V)</b>	<b>Rotten (V)</b>
TGS = 2.19 – 2.32	TGS = 2.28 – 2.43	TGS = 2.05 – 2.25
Alcohol = 2.37 – 3.18	Alcohol = 3.09 – 3.66	Alcohol = 3.17 – 3.37
MQ6 = 0.73 – 0.96	MQ6 = 0.79 – 0.97	MQ6 = 1.02 – 1.77

*Stage 5: Building the model*

Before the model is constructed, the number of samples for training and testing the network will be decided. For the time being, 50 samples for fresh condition and 50 samples for rotten conditions will be used for training the model and 50 samples for each condition will be used for testing the network.

After that, the number of neurons will be determined for the hidden layer of the model. This can be done by using trial and error method. The number of neurons in the hidden layer will determine the success classification rate of the freshness recognition.

Another thing that needs to be considered when building the network is what is the suitable the number of epochs. Number of epochs is a number of cycles needed to execute the calculation.

Besides that, which training algorithm used needs to be determined as well because different training algorithm will determine whether the training will or will not exceed the error goal. The error goal can also be determined with trial and error method.

### *Stage 6: Simulation of the network*

Simulation of the network is done to identify whether the network is able to give an accurate result or not.

### *Stage 7: Make improvement to the network*

If the network does not give an accurate result, improvement will be done until the network is able to produce an accurate result.

### *Stage 8: Building flowchart for C program*

Flowchart is created to get a brief idea on how the program is supposed to execute. Besides that, it will make things easier for creating a C program because the flowchart is served as a guideline.

### *Stage 9: Creating C program*

Once the analysis of the flowchart is done, the C program can be created using a C compiler.

### *Stage 10: Troubleshooting the program*

If the program created produced a lot of syntax errors, the troubleshooting will be done until the program is able to execute smoothly without any errors.

### **3.3 Tools and Software**

The software that will be used for this project is MATLAB where it will be used to write the coding for neural network and to plot simulation error. This software is feasible due to the availability of this software and it is easy to use. MATLAB Neural Network toolbox is also used to find the simulation error. As for C Program, the program is created by using Borland C++. This software is a C compiler and it is feasible to use this software because it is easy to understand how this software works and this software is widely available. Besides, most people are more familiar with Borland C++ compared with other C compilers.



## **CHAPTER 4**

### **Result and Discussions**

#### **4.1 Designing Back Propagation Network**

Before the construction of the network, the data is needed so then the network will be able to determine the result. In this experiment, three types of sensors are used to determine the condition of the fruits or vegetables which are TGS sensor, alcohol sensor and MQ6 sensor. The initial values for the sensors are taken as the baseline for the voltage. This is needed because the baseline will be used as a reference. The sensors are left in the room temperature for 30 minutes. The data below is the minimum value for the baseline.

TGS sensor = 2.09 V

Alcohol sensor = 2.02 V

MQ6 sensor = 0.71 V

After the sensors are stabilized, the fruit will be placed in the container where the sensors are connected. From the readings obtained from the sensors, the condition of the fruits or vegetables will be determined which are fresh, slightly fresh and rotten.

By using sensors alone, it is hard to determine the condition of the fruits or vegetables as the voltage reading from the sensors did not yield much information about the condition other than the voltage keeps on increasing day by day. Therefore, back propagation network is used to determine the condition of the fruits as the network is able to recognize the condition even though there is a slight difference in voltage reading.

In order to construct the back propagation network, 100 sets of data from each sensor readings will be used where 50 sets of data will be used for training input and 50 sets are for simulation input. Only selected days will be included in the neural network since the readings for Day 1 and Day 2 for three sensors do not differ much. Therefore, only Day 1 and Day 11 will be included in the neural network where Day 1 is the data for fresh condition and Day 11 is the data for rotten condition. By using these parameters, the training and performance plot is obtained as shown in Figure 8.

### Network architecture

Network type = feed-forward backprop

Number of hidden layers = 2

Number of neurons in the first hidden layer = 15

Transfer function for first hidden layer = logsig

Number of neurons in the second hidden layer = 1

Transfer function for second hidden layer = purelin

Training function = traingd

### Training parameters

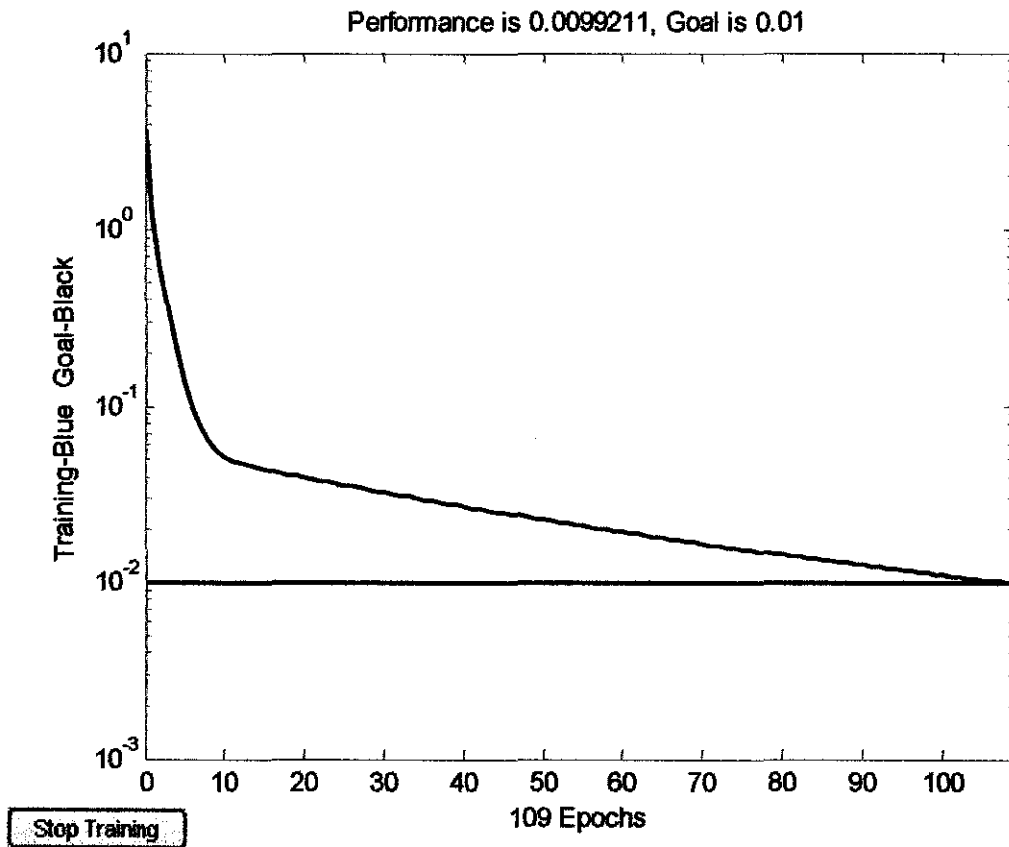
Show = 1

Number of epochs = 1000

Error goal = 0.01

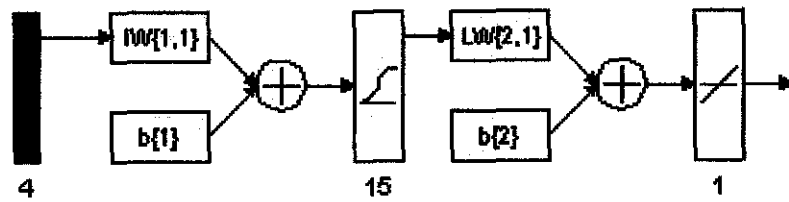
Learning rate = 0.05

Maximum fail = 50



**FIGURE 8: Training and performance plot**

From the training and performance plot, a decaying plot is obtained because the network is trying to minimize the error. The training stopped at 109 epochs even though the number of epochs in the setting is 1000 epochs because the network has reached the error goal which is 0.01. Usually, the network will stop the training when the network has reached its error goal or the number of epochs has reached 1000. Figure 9 shows the network architecture for back propagation network.



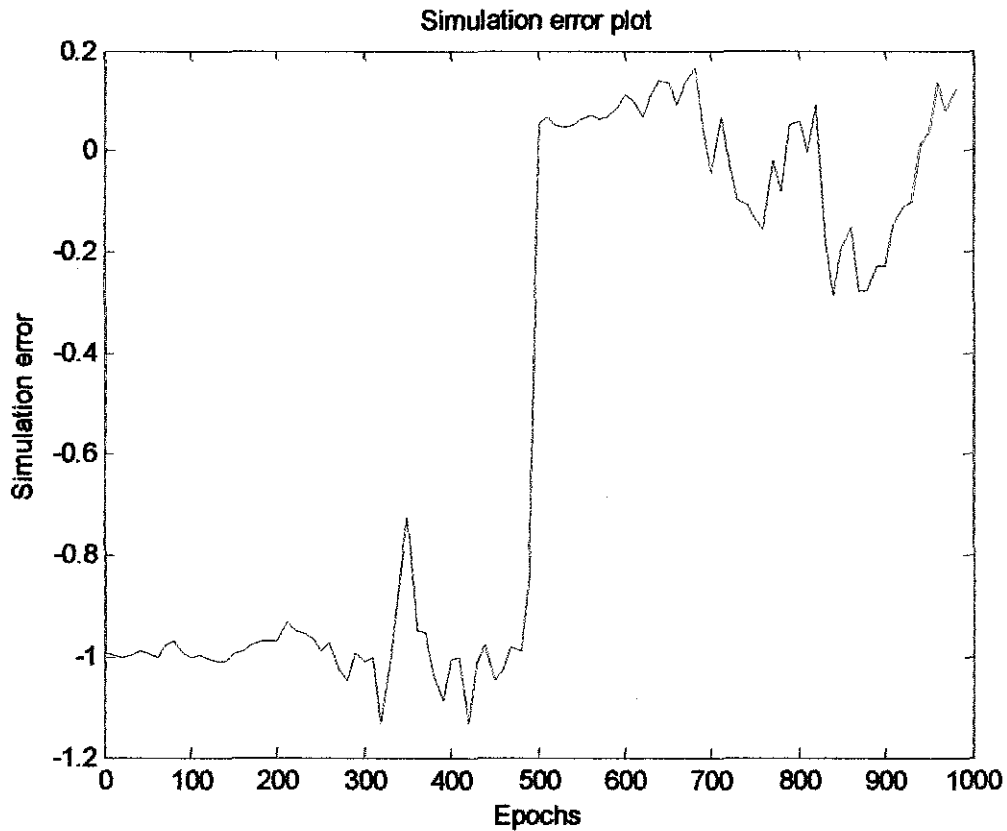
**FIGURE 9: Network architecture for back propagation network**

#### 4.2 Simulation of The Back Propagation Network

Network simulation is done in order to identify the reliability of the neural network where the neural network is able to determine whether the fruits or vegetables are fresh or not.

Basically, the network will determine the freshness of the fruits or vegetables by indicating 1 as fresh and 0 as rotten. Based on the simulation result in Appendix VI, the values to indicate the freshness are either almost close to 1 or 0 based on the result obtained from the simulation. Even though the values are almost close to 1 or 0, the network is still able to classify the freshness of fruits or vegetables accurately. Therefore, the network has the accuracy of 99% for determining the conditions of the fruits or vegetables.

Error can also be obtained from the simulation. Based on the simulation error, the error is very small but there is a dramatic increase at some point. This is due to the transition between fresh and rotten condition where the values difference for both conditions are very big. Figure 10 shows the simulation error plot.



**FIGURE 10: Simulation error plot**

The accuracy of the network depends on the number of neurons in the hidden layers where the result will be more accurate if there are a lot of neurons in the hidden layer. But somehow, too many neurons in the hidden layer do not guarantee it will produce accurate result as the network will not learn anything. Besides that, the training will take much longer to complete due to too many neurons. In this case, 15 neurons in the first hidden layer are sufficient to train the network and 1 neuron in the second hidden layer. There is only 1 neuron in the second hidden layer because only 1 answer is required for identifying the result based on the simulation data.

Another way to increase the accuracy of the network is by increasing the number or epochs as the training will take longer to reach the maximum epoch. However, the training will stop if the network has reached the error goal. Sometimes, the training will still continue even though the network has exceeded the error goal. This is because the network is still trying to minimize the error until it has reached its

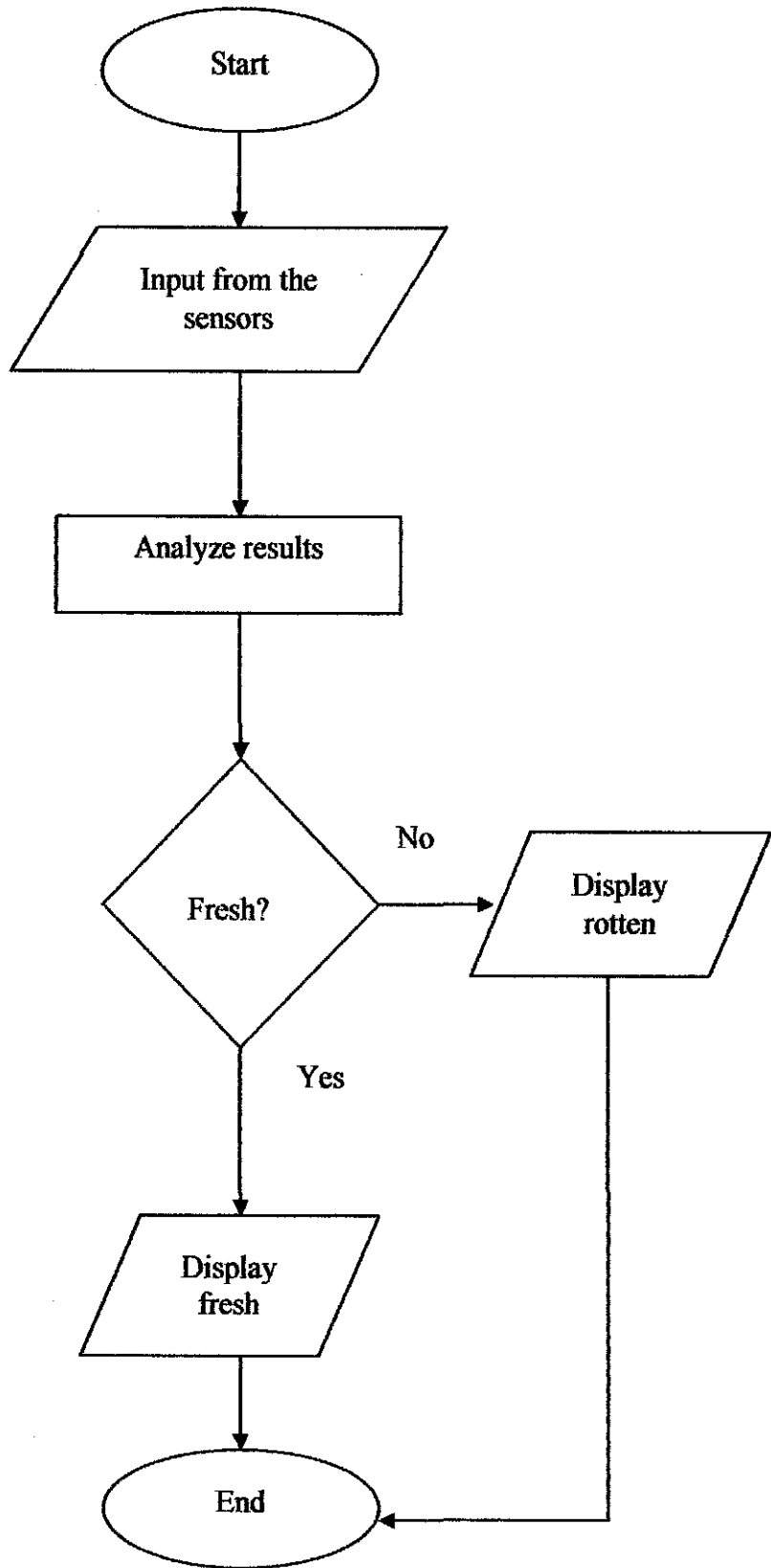
maximum epochs. It can also be indicated that the error goal is not sufficient for the network.

Which types of training algorithm used also affect the accuracy of the network as some of the training algorithm can only be used for specific problems. In this project, `traingd` function is used because this training algorithm is a general training algorithm and can be used for any problem. Other training algorithm such as `trainrp` is also another training algorithm that can be used for this project other than `traingd` which can also produced an accurate result.

### **4.3 Alternative Way to Determine The Condition Of The Fruits Or Vegetables**

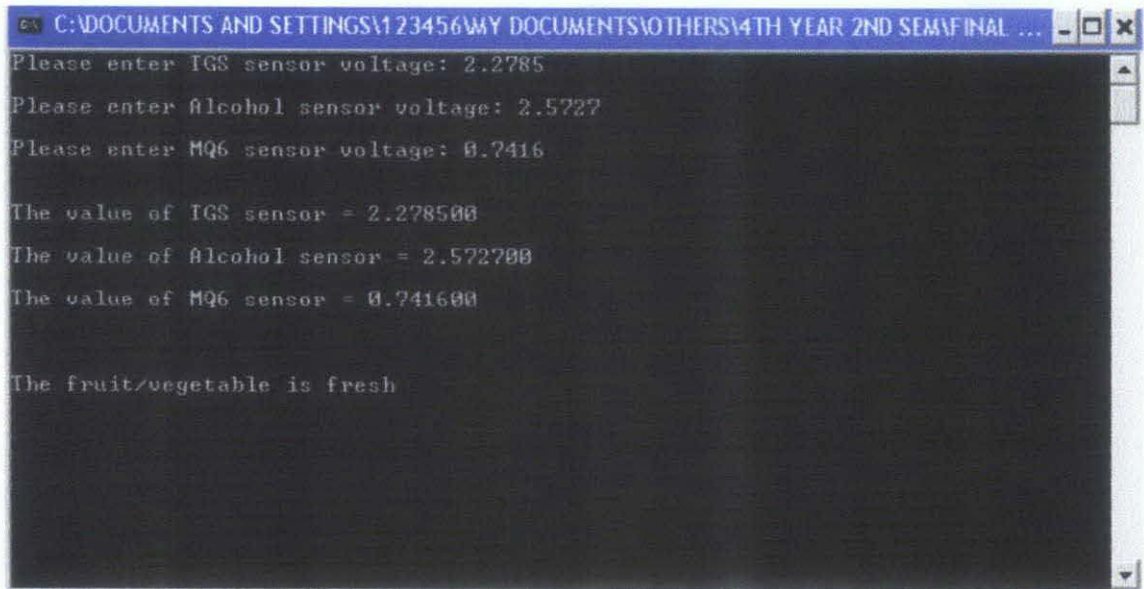
Another way to determine whether the fruit or vegetable is fresh or rotten is by using C program. C program works by inputting the three sensors' voltages and the program will determine whether the fruit or vegetable is fresh or not based on the values provided to the program. Unlike neural network, C program will give a direct result instead of 1 or 0. Therefore, the result from C program is a lot easier to understand because it gives results in words.

C program can also be the desktop version of the portable device since it will produce the result straightaway just like a portable device would. Not just that, C program does not require an additional software in order to execute the program. Therefore, the program does not need to be installed and can be used straightaway. Figure 11 shows the flowchart on how the C program will be executed.



**FIGURE 11: Flowchart for condition determination program**

Once the program is executed, the values for each sensor will be entered and the condition of the fruit or vegetable will be known immediately. This program is created based on the sensor range in Table 4. Figure 12 shows command window of the program.



```
C:\DOCUMENTS AND SETTINGS\123456\MY DOCUMENTS\OTHERS\4TH YEAR 2ND SEM\FINAL ...
Please enter IGS sensor voltage: 2.2785
Please enter Alcohol sensor voltage: 2.5727
Please enter MQ6 sensor voltage: 0.7416

The value of IGS sensor = 2.278500
The value of Alcohol sensor = 2.572700
The value of MQ6 sensor = 0.741600

The fruit/vegetable is fresh
```

**FIGURE 12: Condition determination in C program**

#### **4.4 Comparison between Neural Network And C Program**

The result produced from C program is easier to understand compared to the output simulation obtained from the neural network because C program will give direct result instead of integers in neural network. Therefore, C program is more convenient to use compared to the neural network because C program is compatible with human language.

However, C program is not capable to identify the sensor reading correctly because it relies solely on the sensor range. From Table 2, the sensor range for fresh, slightly fresh and rotten are overlapping, making it hard for C program to identify the condition because the value can be any of those conditions. Another thing is needed to be considered is the C program analyzed the given inputs in real time unlike neural network where it recalls the data from the memory storage it has stored previously.



Therefore, neural network plays an important role in identifying the condition because neural network is able to classify the reading according to the right condition. Since neural network itself is based on the past events, the results obtained are more accurate than the results in C program.

Similarly, both neural network and C program are high-level language where in the case of C program, it is a general-purpose language. Therefore, it can run on any type of computers. It also works with neural network where C++ program can be integrated into neural network.

## **CHAPTER 5**

### **Conclusions and Recommendations**

#### **5.1 Conclusions**

Three waveforms for each condition which are obtained from the sensors are similar in a way that the voltage for each sensor increased day by day. As for rotten condition, the voltage is decreasing for TGS sensor and MQ6 sensor but increasing for Alcohol sensor. The neural network is able to classify the readings from these three sensors because there are similarities in the fresh condition readings but there is a slight difference in readings for rotten condition. Despite of the change in the readings for rotten condition, the neural network is still able to determine the condition based on the readings from the sensors. With the right amount of neurons, epochs and the correct training algorithm, neural network can be used to analyze the data accurately because it is recalled from the data stored in the memory during training. The amount of data can be stored is based on which training algorithm used in the training. Therefore, the network is able to classify the fruits and vegetables correctly with the accuracy of 99% which means that the network will give correct result of 99% most of the time.

C program can also be used to determine the freshness of the fruits and vegetables but it is not able to adapt the change in readings because the C program is created based on the sensor range, making it not as sensitive to changes like neural network. In conclusion, neural network is the right software to measure the freshness of fruits and vegetables for this project.

## **5.2 Recommendations**

This project can be improved further by creating a program embedded in microcontroller as a part of the portable device. The program can be created by identifying the voltage readings produced by the sensors and converted the readings into digital form via analog to digital conversion. Based on these conversion, the result can be displayed by using LCD as the LCD itself is connected to the microcontroller altogether.

## REFERENCES

- [1] Gavin M, 8 August 2008 <<http://kidshealth.org/kid/htbw/nose.html>>
- [2] Wikipedia, 8 August 2008  
<[http://en.wikipedia.org/wiki/Artificial\\_neural\\_network](http://en.wikipedia.org/wiki/Artificial_neural_network)>
- [3] Sivanandam S.N, Sumathi S, Deepa S.N. 2006. *Introduction to Neural Network Using Matlab 6.0*. McGraw Hill. p. 3 – 4
- [4] Koivo H.N, 8 August 2008  
<[http://www.control.hut.fi/Kurrsit./AS-74.3115/2008/Material/NN\\_Basics\\_2006.pdf](http://www.control.hut.fi/Kurrsit./AS-74.3115/2008/Material/NN_Basics_2006.pdf)>
- [5] Demuth H, Beale M, Hagan M. 2008. *Neural Network Toolbox™ 6 User's Guide*. The MathWorks Inc. p384-389, p189-206
- [6] Suman M, Rianin G, Dalcanale E. "MOS-based artificial olfactory system for the assessment of egg products freshness," *Sensors and Actuators B* 125(2007) 40 – 47
- [7] Gottfried B. 1990. *Schaum's Outlines Programming With C Second Edition*. McGraw Hill. p.7-9

## Appendix 1: Neural Network Source Code

```

clear all

Input = [2.2858 2.2901 2.298 2.306 2.3096 2.309 2.3048 2.3127
2.3145 2.3084 2.3041 2.3048 2.2987 2.2956 2.2913 2.2767 2.2724 2.2663
2.262 2.259 2.2486 2.2272 2.2169 2.2193 2.2065 2.1931 2.1973 2.1876
2.1924 2.1998 2.2022 2.2071 2.2108 2.2279 2.2333 2.2266 2.2065 2.2065
2.2065 2.2156 2.2327 2.2419 2.2523 2.2803 2.2864 2.2852 2.2987 2.3109
2.309 2.3176 2.0307 2.0356 2.0283 2.0386 2.0374 2.0338 2.0301 2.024
2.0307 2.0338 2.0222 2.0093 2.0167 2.0466 2.0008 1.9922 1.9861 2.0142
2.0069 1.9514 1.9379 1.9562 1.9391 1.9288 1.9288 1.9251 1.9227 1.9416
1.933 1.9532 1.955 1.9434 1.9623 1.9196 1.9092 1.9184 1.9245 1.9117
1.9105 1.916 1.9166 1.9251 1.9288 1.93 1.9459 1.952 1.9831 1.9623
2.0038;

2.9914 3.0189 3.0653 3.1214 3.1489 3.1483 3.122 3.1745
3.1849 3.1483 3.1172 3.1098 3.0793 3.0677 3.0396 2.9505 2.9298 2.8925
2.8645 2.851 2.7961 2.6405 2.5837 2.6014 2.5202 2.436 2.4519 2.4207
2.4671 2.486 2.5141 2.5422 2.6527 2.6832 2.635 2.5141 2.5056 2.5074
2.5581 2.6551 2.71 2.7668 2.9536 2.9872 2.9774 3.0549 3.1367 3.1159
3.1806 3.3479 3.3583 3.3387 3.3638 3.3656 3.3583 3.3503 3.3412 3.3509
3.3479 3.3253 3.2942 3.3143 3.3509 3.2624 3.2472 3.2386 3.285 3.271
3.1544 3.1208 3.1611 3.122 3.0976 3.0921 3.0805 3.0757 3.1092 3.0988
3.1452 3.1465 3.1495 3.1245 3.1404 3.0671 3.0335 3.0543 3.0616 3.0226
3.0287 3.0427 3.0409 3.0592 3.0793 3.0873 3.1343 3.1501 3.2112 3.1849
3.2868;

0.8942 0.9052 0.9149 0.9302 0.9394 0.9387 0.9308 0.9461
0.9516 0.9406 0.932 0.9302 0.9217 0.918 0.9095 0.885 0.8789 0.8698
0.8637 0.86 0.8429 0.8032 0.7874 0.791 0.7697 0.7453 0.7483 0.7331
0.7404 0.7526 0.7593 0.766 0.7752 0.8039 0.8136 0.8008 0.7666 0.766
0.766 0.7813 0.8051 0.8216 0.8362 0.8869 0.8954 0.8911 0.9125 0.9333
0.929 0.9467 1.7621 1.7646 1.7573 1.7682 1.767 1.7627 1.7585 1.7536
1.7585 1.7566 1.7469 1.7322 1.7414 1.7609 1.7188 1.7084 1.7048 1.7286
1.7206 1.6614 1.6474 1.6651 1.648 1.6352 1.6315 1.6266 1.6248 1.6425
1.6364 1.659 1.6602 1.648 1.6614 1.6224 1.6102 1.6187 1.623 1.6071
1.6108 1.6138 1.6114 1.6224 1.6291 1.6297 1.6486 1.6541 1.6834 1.6669
1.7121;

1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
11 11 11 11 11 11 11 11 11 11
11 11 11 11 11 11 11 11 11 11
11 11 11 11 11 11 11 11 11 11
11 11 11 11 11 11 11 11 11 11
11 11 11 11 11 11 11 11 11];

Output =[1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1

```

```
1    1    1    1    1    1    1    1    1    1
0    0    0    0    0    0    0    0    0    0
0    0    0    0    0    0    0    0    0    0
0    0    0    0    0    0    0    0    0    0
0    0    0    0    0    0    0    0    0    0
0    0    0    0    0    0    0    0    0    0
```

```
Range = [2.05 2.32; 2.37 3.66; 0.73 1.77; 1 11];
```

```
net = newff(Range, [15,1], {'logsig', 'purelin'}, 'traingd');
net.trainParam.show=1;
net.trainParam.epochs=1000;
net.trainParam.goal=0.01;
net.trainParam.lr=0.05;
[net,tr]=train(net,Input,Output);
```

```
return
```

## Appendix II: C Program Source Code

```
#include <stdio.h>

int main ()
{
    float tgs, alcohol, mq6;

    printf("Please enter TGS sensor voltage: ");
    scanf ("%f", &tgs);
    printf("\nPlease enter Alcohol sensor voltage: ");
    scanf ("%f", &alcohol);
    printf("\nPlease enter MQ6 sensor voltage: ");
    scanf ("%f", &mq6);

    printf ("\n\nThe value of TGS sensor = %f\n", tgs);
    printf ("\n\nThe value of Alcohol sensor = %f\n", alcohol);
    printf ("\n\nThe value of MQ6 sensor = %f\n", mq6);

    if ((2.37 <= alcohol <= 3.18 ))
    {
        printf ("\n\n\nThe fruit/vegetable is fresh", alcohol);
    }
    else if (0.73 <= tgs <= 0.96)
    {
        printf ("\n\n\nThe fruit/vegetable is fresh", tgs);
    }
    else if (2.262 <= mq6 <= 3.636)
    {
        printf ("\n\n\nThe fruit/vegetable is fresh", mq6);
    }
    else if ((alcohol >= 3.17))
    {
        printf ("\n\n\nThe fruit/vegetable is rotten", alcohol);
    }
    else if ((tgs >= 2.05))
    {
        printf ("\n\n\nThe fruit/vegetable is rotten", tgs);
    }
    else if ((mq6 >= 1.02))
    {
        printf ("\n\n\nThe fruit/vegetable is rotten", mq6);
    }

    else

        printf ("\n\n\nERROR - serves you right, lol");

    return 0;
}
```

**Appendix III: TGS Sensor Reading For Fresh and Rotten Condition**

Day 1	Day 3	Day 8	Day 9	Day 11
2.2858	2.3872	2.4097	2.3676	2.0252
2.2858	2.3737	2.3963	2.364	2.0307
2.2901	2.3682	2.4049	2.3609	2.0356
2.298	2.3823	2.4226	2.3572	2.0283
2.306	2.3878	2.4262	2.353	2.0386
2.3096	2.3872	2.4287	2.3347	2.0374
2.309	2.4006	2.4323	2.3084	2.0338
2.3048	2.3963	2.4293	2.317	2.0301
2.3127	2.392	2.4311	2.3304	2.024
2.3145	2.3988	2.4268	2.3267	2.0307
2.3084	2.3981	2.4238	2.317	2.0338
2.3041	2.3969	2.4207	2.3157	2.0222
2.3048	2.3951	2.4244	2.317	2.0093
2.2987	2.3945	2.4256	2.3225	2.0167
2.2956	2.3841	2.4311	2.328	2.0466
2.2913	2.3908	2.4305	2.3212	2.0008
2.2767	2.3939	2.4256	2.3139	1.9922
2.2724	2.3927	2.4342	2.3011	1.9861
2.2663	2.3933	2.4342	2.3078	2.0142
2.262	2.3927	2.4305	2.2895	2.0069
2.259	2.392	2.4299	2.2846	1.9514
2.2486	2.3762	2.425	2.2938	1.9379
2.2272	2.3835	2.411	2.284	1.9562
2.2169	2.3688	2.4134	2.2871	1.9391
2.2193	2.3634	2.3975	2.2889	1.9288
2.2065	2.3597	2.3927	2.2901	1.9288
2.1931	2.3591	2.3872	2.3035	1.9251
2.1973	2.3481	2.3811	2.2993	1.9227
2.1876	2.3145	2.3414	2.3182	1.9416
2.1924	2.3255	2.3542	2.3212	1.933
2.1998	2.3347	2.3463	2.3249	1.9532
2.2022	2.3188	2.3286	2.3298	1.9532
2.2071	2.3103	2.3243	2.3603	1.955
2.2108	2.3139	2.3188	2.3676	1.9434
2.2279	2.3023	2.3176	2.3499	1.9623
2.2333	2.2987	2.3225	2.3481	1.9196
2.2266	2.3084	2.3243	2.3524	1.9092
2.2065	2.3005	2.3218	2.3591	1.9184



2.2065	2.2962	2.3157	2.3676	1.9245
2.2065	2.2987	2.3157	2.3695	1.9117
2.2156	2.2993	2.32	2.3621	1.9105
2.2327	2.2974	2.3182	2.3627	1.916
2.2419	2.3029	2.3145	2.3621	1.9166
2.2523	2.3139	2.3145	2.3676	1.9251
2.2803	2.3084	2.3157	2.3658	1.9288
2.2864	2.3237	2.3127	2.367	1.93
2.2852	2.328	2.3115	2.3591	1.9459
2.2987	2.3316	2.3109	2.3615	1.952
2.3109	2.3316	2.3164	2.3798	1.9831
2.309	2.3347	2.3139	2.378	1.9623
2.3176	2.3585	2.3267	2.3707	2.0038
2.3066	2.3719	2.3261	2.3798	1.9831
2.3041	2.356	2.3408	2.378	1.9947
2.298	2.3634	2.3475	2.3731	2.0118
2.2822	2.3756	2.3511	2.375	2.0167
2.2736	2.3774	2.3536	2.364	2.024
2.2669	2.3786	2.3597	2.3646	2.0276
2.2645	2.3896	2.3933	2.3615	2.0203
2.2394	2.389	2.392	2.3585	2.0197
2.2217	2.3817	2.3798	2.3591	2.0301
2.223	2.3914	2.3908	2.3572	2.0228
2.2236	2.3884	2.4036	2.3426	2.0252
2.2181	2.3872	2.4097	2.3383	2.0173
2.2077	2.3872	2.4134	2.3328	2.0032
2.2016	2.3835	2.4073	2.331	2.0063
2.1973	2.3676	2.4097	2.3231	1.9916
2.1955	2.3743	2.4165	2.3066	1.9861
2.1992	2.3743	2.4128	2.281	1.9868
2.1949	2.3713	2.4067	2.2987	1.9807
2.1894	2.3664	2.4104	2.281	1.9782
2.1918	2.3676	2.4024	2.2755	1.9715
2.2004	2.3701	2.3933	2.2633	1.9678
2.1992	2.3719	2.3914	2.2535	1.9636
2.2077	2.3579	2.3786	2.2645	1.9538
2.2047	2.3499	2.3762	2.2523	1.9288
2.2211	2.3493	2.375	2.2559	1.9349
2.2272	2.3457	2.3737	2.2584	1.952
2.2315	2.3426	2.375	2.2571	1.944
2.2358	2.3121	2.3798	2.273	1.9337

2.2394	2.3017	2.3853	2.2639	1.9446
2.2633	2.3182	2.3884	2.2718	1.9385
2.2785	2.3109	2.3878	2.2846	1.9288
2.2828	2.2968	2.3847	2.2883	1.9404
2.2785	2.298	2.3823	2.2901	1.9288
2.2803	2.2919	2.3817	2.2919	1.9459
2.2883	2.2803	2.3743	2.2962	1.9825
2.2987	2.2877	2.3725	2.2962	1.9343
2.3035	2.2895	2.3365	2.2926	1.9544
2.3084	2.2773	2.3261	2.2864	1.9239
2.3103	2.2816	2.3353	2.2858	1.9147
2.3054	2.2864	2.3218	2.2871	1.908
2.2919	2.2852	2.3084	2.2895	1.916
2.2846	2.2999	2.3005	2.2944	1.9147
2.2858	2.2944	2.2962	2.3292	1.9086
2.295	2.3115	2.2944	2.328	1.9098
2.306	2.3176	2.295	2.3133	1.9123
2.3096	2.3176	2.2956	2.3231	1.9111
2.306	2.3231	2.2901	2.3377	1.9257
2.3139	2.3267	2.2852	2.3286	1.9227
2.3048	2.3615	2.2864	2.3469	1.941
2.3005	2.3518	2.2901	2.3511	1.9459

**Appendix IV: Alcohol Sensor Reading For Fresh and Rotten Condition**

Day 1	Day 3	Day 8	Day 9	Day 11
2.9945	3.3363	3.5945	3.3967	3.3314
2.9914	3.3076	3.553	3.404	3.3479
3.0189	3.2942	3.5798	3.4205	3.3583
3.0653	3.332	3.6256	3.5188	3.3387
3.1214	3.3473	3.6384	3.5176	3.3638
3.1489	3.3479	3.6464	3.4754	3.3656
3.1483	3.3766	3.6555	3.5011	3.3583
3.122	3.3705	3.6482	3.5456	3.3503
3.1745	3.3558	3.6518	3.52	3.3412
3.1849	3.3766	3.6433	3.5774	3.3509
3.1483	3.3747	3.6348	3.5932	3.3479
3.1172	3.3705	3.6225	3.5652	3.3253
3.1098	3.368	3.6299	3.5823	3.2942
3.0793	3.3705	3.6348	3.5896	3.3143
3.0677	3.3479	3.6506	3.589	3.3509
3.0396	3.3662	3.6409	3.5908	3.2624
2.9505	3.3729	3.625	3.5878	3.2472
2.9298	3.3686	3.6525	3.5829	3.2386
2.8925	3.3692	3.6567	3.5731	3.285
2.8645	3.3705	3.6427	3.5707	3.271
2.851	3.3644	3.6427	3.5719	3.1544
2.7961	3.3192	3.628	3.5395	3.1208
2.6405	3.3357	3.5896	3.5322	3.1611
2.5837	3.2978	3.5987	3.5261	3.122
2.6014	3.2826	3.5517	3.4748	3.0976
2.5202	3.2746	3.5353	3.459	3.0921
2.436	3.2649	3.5206	3.451	3.0805
2.4519	3.2429	3.5011	3.4315	3.0757
2.3927	3.166	3.3717	3.3009	3.1092
2.4207	3.1758	3.4138	3.354	3.0988
2.4671	3.2063	3.3912	3.3308	3.1452
2.486	3.1672	3.346	3.2875	3.1465
2.5141	3.1471	3.3326	3.3052	3.1495
2.5422	3.152	3.3161	3.2466	3.1245
2.6527	3.1257	3.3168	3.2398	3.1404
2.6832	3.1153	3.3229	3.2545	3.0671
2.635	3.1318	3.3351	3.26	3.0335
2.5141	3.1196	3.3247	3.2313	3.0543

2.5056	3.108	3.3033	3.2331	3.0616
2.5074	3.1153	3.3088	3.2417	3.0226
2.5581	3.119	3.3241	3.2392	3.0287
2.6551	3.1166	3.321	3.2344	3.0427
2.71	3.1294	3.3094	3.2301	3.0409
2.7668	3.1562	3.3058	3.2282	3.0592
2.9536	3.1458	3.3161	3.2282	3.0793
2.9872	3.1812	3.3058	3.2307	3.0873
2.9774	3.1941	3.3033	3.238	3.1343
3.0549	3.2014	3.3076	3.2411	3.1501
3.1367	3.2124	3.3222	3.2386	3.2112
3.1159	3.2228	3.3131	3.2807	3.1849
3.1806	3.274	3.3528	3.2661	3.2868
3.1074	3.3119	3.3546	3.2838	3.2478
3.0909	3.2728	3.3961	3.3296	3.2679
3.0647	3.2875	3.415	3.3375	3.3119
2.9554	3.3265	3.4266	3.3479	3.3247
2.9084	3.3247	3.4431	3.354	3.3406
2.8578	3.332	3.4559	3.3735	3.3552
2.8358	3.3631	3.5591	3.4565	3.3393
2.6887	3.357	3.5591	3.4931	3.3412
2.5745	3.3436	3.5176	3.4443	3.3601
2.5727	3.3656	3.553	3.4584	3.3485
2.5788	3.3619	3.5939	3.498	3.3497
2.5483	3.3583	3.614	3.5145	3.3363
2.4866	3.3564	3.6299	3.5041	3.3021
2.4409	3.3491	3.6073	3.5463	3.3143
2.4189	3.3119	3.6195	3.5548	3.2771
2.4049	3.3296	3.6378	3.5408	3.2667
2.4311	3.329	3.6274	3.5292	3.263
2.4104	3.3235	3.6116	3.5151	3.2508
2.3719	3.3088	3.6201	3.5426	3.2472
2.3988	3.3222	3.5957	3.5694	3.2325
2.4433	3.318	3.5652	3.5542	3.227
2.4519	3.3271	3.5536	3.5292	3.2179
2.4964	3.2887	3.5096	3.564	3.1898
2.4946	3.2746	3.4968	3.5475	3.1343
2.5776	3.2704	3.4809	3.5401	3.1416
2.6283	3.2569	3.4632	3.5371	3.177
2.6582	3.2435	3.4626	3.4791	3.166
2.6978	3.1758	3.4669	3.5066	3.1404

2.721	3.141	3.4761	3.4462	3.1709
2.851	3.1849	3.4828	3.4358	3.1581
2.9621	3.1684	3.4754	3.4193	3.1367
2.989	3.1294	3.4645	3.4114	3.1629
2.9713	3.1288	3.4553	3.3906	3.1349
2.9859	3.122	3.4413	3.2624	3.1727
3.0226	3.0879	3.4205	3.2948	3.2374
3.0811	3.1013	3.4059	3.3204	3.1538
3.1196	3.1092	3.2875	3.3186	3.1819
3.1355	3.0854	3.2643	3.3168	3.1288
3.1434	3.0927	3.3204	3.3033	3.1135
3.1135	3.0952	3.2881	3.3143	3.0909
3.025	3.0958	3.2502	3.3125	3.1092
2.981	3.1336	3.2417	3.2649	3.105
2.989	3.1202	3.2295	3.2282	3.0927
3.0476	3.1599	3.2313	3.2545	3.1037
3.1251	3.1727	3.2466	3.1843	3.1062
3.1434	3.1727	3.2557	3.1758	3.108
3.1214	3.1861	3.2331	3.2026	3.1513
3.1806	3.2002	3.2289	3.1819	3.1416
3.1269	3.2923	3.235	3.1721	3.1831
3.0988	3.2722	3.2514	3.1837	3.1935

**Appendix V: MQ6 Sensor Reading For Fresh and Rotten Condition**

Day 1	Day 3	Day 8	Day 9	Day 11
0.8936	0.9949	0.9143	1.3306	1.756
0.8942	0.9729	0.8856	1.2952	1.7621
0.9052	0.9626	0.8936	1.3044	1.7646
0.9149	0.99	0.8667	1.3245	1.7573
0.9302	1.0022	0.8515	1.3147	1.7682
0.9394	1.0034	0.8429	1.3135	1.767
0.9387	1.0242	0.8368	1.3141	1.7627
0.9308	1.0193	0.8313	1.3086	1.7585
0.9461	1.0096	0.7874	1.305	1.7536
0.9516	1.0254	0.7471	1.3196	1.7585
0.9406	1.0242	0.7697	1.333	1.7566
0.932	1.0199	0.7727	1.3422	1.7469
0.9302	1.015	0.7392	1.3251	1.7322
0.9217	1.0157	0.727	1.3208	1.7414
0.918	0.9992	0.7239	1.3337	1.7609
0.9095	1.0108	0.7263	1.3233	1.7188
0.885	1.0157	0.7251	1.3178	1.7084
0.8789	1.0144	0.7276	1.316	1.7048
0.8698	1.015	0.7624	1.283	1.7286
0.8637	1.015	0.7807	1.3001	1.7206
0.86	1.0108	0.7801	1.2812	1.6614
0.8429	0.9809	0.7813	1.2616	1.6474
0.8032	0.9943	0.7605	1.2555	1.6651
0.7874	0.9644	0.7337	1.25	1.648
0.791	0.9552	0.7257	1.2421	1.6352
0.7697	0.9473	0.73	1.2348	1.6315
0.7453	0.9412	0.705	1.2049	1.6266
0.7483	0.9241	0.7184	1.1597	1.6248
0.7331	0.8661	0.7074	1.1719	1.6425
0.7404	0.8741	0.7141	1.1933	1.6364
0.7526	0.893	0.7202	1.1841	1.659
0.7593	0.8637	0.744	1.1701	1.6602
0.766	0.8478	0.7611	1.1658	1.6602
0.7752	0.8515	0.7794	1.1689	1.648
0.8039	0.8313	0.788	1.1713	1.6614
0.8136	0.8203	0.7978	1.1829	1.6224
0.8008	0.8344	0.8081	1.1713	1.6102
0.7666	0.8258	0.871	1.1585	1.6187

0.766	0.8173	0.8844	1.1383	1.623
0.766	0.8234	0.8771	1.1487	1.6071
0.7813	0.8252	0.8863	1.117	1.6108
0.8051	0.8246	0.9162	1.1078	1.6138
0.8216	0.8301	0.9387	1.1225	1.6114
0.8362	0.8521	0.9412	1.1035	1.6224
0.8869	0.8405	0.9638	1.1103	1.6291
0.8954	0.8679	0.9583	1.1127	1.6297
0.8911	0.8777	0.9632	1.1115	1.6486
0.9125	0.882	0.9662	1.1328	1.6541
0.9333	0.8869	0.9607	1.128	1.6834
0.929	0.8942	0.9601	1.1518	1.6669
0.9467	0.932	0.9632	1.1591	1.7121
0.9265	0.9632	0.968	1.164	1.6926
0.921	0.9308	0.9577	1.1719	1.7017
0.9137	0.9442	0.9546	1.2226	1.7206
0.8869	0.9693	0.9626	1.2305	1.7255
0.8734	0.9687	0.9674	1.2049	1.7335
0.8606	0.9754	0.9601	1.2006	1.7377
0.8521	0.9955	0.9583	1.2061	1.7286
0.8124	0.9925	0.9461	1.2195	1.7298
0.7849	0.9815	0.9235	1.2311	1.7377
0.7843	0.9973	0.9156	1.2299	1.7316
0.7868	0.9961	0.8875	1.2201	1.7304
0.777	0.9949	0.8795	1.2177	1.7249
0.7624	0.9912	0.8698	1.2189	1.7084
0.7471	0.9851	0.8551	1.225	1.7115
0.7422	0.9571	0.7801	1.2232	1.6944
0.7379	0.9699	0.7868	1.2232	1.6883
0.7459	0.9687	0.8057	1.211	1.6852
0.7416	0.9632	0.7892	1.2128	1.6791
0.7294	0.9522	0.7575	1.2403	1.6767
0.7367	0.9619	0.7489	1.2378	1.6675
0.7501	0.9607	0.7459	1.2268	1.6651
0.7514	0.9644	0.7288	1.2384	1.659
0.7654	0.9375	0.741	1.2372	1.6456
0.7617	0.929	0.7245	1.2293	1.6181
0.7855	0.921	0.7288	1.2293	1.6224
0.7984	0.9125	0.7312	1.2116	1.6376
0.8075	0.9033	0.7404	1.211	1.6309
0.8203	0.857	0.766	1.2037	1.6169

0.8246	0.8301	0.7556	1.1975	1.6315
0.8594	0.8606	0.7758	1.1988	1.6272
0.8893	0.846	0.7904	1.1957	1.6144
0.8954	0.8234	0.7959	1.1725	1.626
0.8899	0.8228	0.8002	1.1652	1.612
0.896	0.8161	0.8063	1.1542	1.6291
0.9046	0.7898	0.8112	1.1512	1.6626
0.9198	0.7996	0.8197	1.1408	1.6181
0.9296	0.8063	0.8881	1.1121	1.6309
0.9363	0.7868	0.8893	1.0694	1.6041
0.9369	0.7935	0.8637	1.0968	1.5949
0.9302	0.7996	0.8808	1.07	1.5821
0.9058	0.7953	0.9137	1.0602	1.5918
0.893	0.824	0.918	1.037	1.5882
0.8972	0.8167	0.9125	1.023	1.5815
0.9125	0.8405	0.9143	1.0376	1.5857
0.9308	0.8539	0.9131	1.0193	1.5857
0.9375	0.8576	0.8924	1.0248	1.587
0.932	0.8649	0.8625	1.0285	1.6065
0.9479	0.8728	0.8625	1.0224	1.601
0.9326	0.9381	0.8728	1.0547	1.6211
0.9247	0.9247	0.8985	1.0358	1.6272



## Appendix VI: Simulation Input and Simulation Result From The Neural Network

```
>> Test_1 = [2.3066 2.3041 2.298 2.2822 2.2736 2.2669 2.2645 2.2394
2.2217 2.223 2.2236 2.2181 2.2077 2.2016 2.1973 2.1955 2.1992 2.1949
2.1894 2.1918 2.2004 2.1992 2.2077 2.2047 2.2211 2.2272 2.2315 2.2358
2.2394 2.2633 2.2785 2.2828 2.2785 2.2803 2.2883 2.2987 2.3035 2.3084
2.3103 2.3054 2.2919 2.2846 2.2858 2.295 2.3096 2.306 2.3139 2.3048
2.3005 1.9831 1.9947 2.0118 2.0167 2.024 2.0276 2.0203 2.0197 2.0301
2.0228 2.0252 2.0173 2.0032 2.0063 1.9916 1.9861 1.9868 1.9807 1.9782
1.9715 1.9678 1.9636 1.9538 1.9288 1.9349 1.952 1.944 1.9337 1.9446
1.9385 1.9288 1.9404 1.9288 1.9459 1.9825 1.9343 1.9544 1.9239 1.9147
1.908 1.916 1.9147 1.9086 1.9098 1.9123 1.9111 1.9257 1.9227 1.941
1.9459 2.0038;
3.1074 3.0909 3.0647 2.9554 2.9084 2.8578 2.8358 2.6887
2.5745 2.5727 2.5788 2.5483 2.4866 2.4409 2.4189 2.4049 2.4311 2.4104
2.3719 2.3988 2.4433 2.4519 2.4964 2.4946 2.5776 2.6283 2.6582 2.6978
2.721 2.851 2.9621 2.989 2.9713 2.9859 3.0226 3.0811 3.1196 3.1355
3.1434 3.1135 3.025 2.981 2.989 3.0476 3.1251 3.1434 3.1214 3.1806
3.1269 3.0988 3.2478 3.2679 3.3119 3.3247 3.3406 3.3552 3.3393 3.3412
3.3601 3.3485 3.3497 3.3363 3.3021 3.3143 3.2771 3.2667 3.263 3.2508
3.2472 3.2325 3.227 3.2179 3.1898 3.1343 3.1416 3.177 3.166 3.1404
3.1709 3.1581 3.1367 3.1629 3.1349 3.1727 3.2374 3.1538 3.1819 3.1288
3.1135 3.0909 3.1092 3.105 3.0927 3.1037 3.1062 3.108 3.1513 3.1416
3.1831 3.1935;
0.9265 0.921 0.9137 0.8869 0.8734 0.8606 0.8521 0.8124
0.7849 0.7843 0.7868 0.777 0.7624 0.7471 0.7422 0.7379 0.7459 0.7416
0.7294 0.7367 0.7501 0.7514 0.7654 0.7617 0.7855 0.7984 0.8075 0.8203
0.8246 0.8594 0.8893 0.8954 0.8899 0.896 0.9046 0.9198 0.9296 0.9363
0.9369 0.9302 0.9058 0.893 0.8972 0.9125 0.9308 0.9375 0.932 0.9479
0.9326 0.9247 1.6926 1.7017 1.7206 1.7255 1.7335 1.7377 1.7286 1.7298
1.7377 1.7316 1.7304 1.7249 1.7084 1.7115 1.6944 1.6883 1.6852 1.6791
1.6767 1.6675 1.6651 1.659 1.6456 1.6181 1.6224 1.6376 1.6309 1.6169
1.6315 1.6272 1.6144 1.626 1.612 1.6291 1.6626 1.6181 1.6309 1.6041
1.5949 1.5821 1.5918 1.5882 1.5815 1.5857 1.5857 1.587 1.6065 1.601
1.6211 1.6272;
1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1
```

```

1      1      1      1      1      1      1      1      1      1
1      1      1      1      1      1      1      1      1      1
1      11     11     11     11     11     11     11     11     11
11     11     11     11     11     11     11     11     11     11
11     11     11     11     11     11     11     11     11     11
11     11     11     11     11     11     11     11     11     11
11     11     11     11     11     11     11     11     11     11

```

```
11});
```

```
>> Answer = sim(net,Test_1)
```

```
Answer =
```

```
Columns 1 through 11
```

```

0.8784    0.8898    0.9046    0.9640    0.9859    1.0088    1.0195
1.0801    1.1384    1.1412    1.1375

```

```
Columns 12 through 22
```

```

1.1539    1.1904    1.2189    1.2310    1.2396    1.2239    1.2349
1.2569    1.2392    1.2158    1.2066

```

```
Columns 23 through 33
```

```

1.1824    1.1788    1.1354    1.1061    1.0911    1.0707    1.0604
1.0097    0.9567    0.9434    0.9515

```

```
Columns 34 through 44
```

```

0.9424    0.9251    0.8917    0.8659    0.8559    0.8524    0.8711
0.9261    0.9503    0.9450    0.9123

```

```
Columns 45 through 55
```

```

0.8675    0.8471    0.8711    0.8089    0.8550    -0.4730    0.0169
0.0560    0.0348    0.0500    0.0485

```

```
Columns 56 through 66
```

0.0128 0.0248 0.0597 0.0184 0.0366 0.0104 -0.0218  
0.0139 -0.0351 -0.0226 -0.0120

Columns 67 through 77

-0.0240 -0.0209 -0.0338 -0.0298 -0.0353 -0.0476 -0.0714  
-0.0209 0.0081 -0.0355 -0.0445

Columns 78 through 88

-0.0023 -0.0381 -0.0469 -0.0072 -0.0470 0.0075 0.0712  
-0.0882 0.0135 -0.0634 -0.0374

Columns 89 through 99

-0.0331 -0.0017 -0.0194 -0.0231 -0.0109 -0.0148 -0.0171  
0.0041 -0.0338 0.0070 -0.0176

Column 100

0.1358