

**Weather Data Transmission Driven
By Artificial Neural Network based Prediction**

By

Muhammad Zulhilmi bin Md Salahan
16127

Dissertation submitted in partial fulfilment of
the requirements for the
Bachelor of Technology (Hons)
(Information & Communication Technology)

MAY 2015

Universiti Teknologi PETRONAS
32610, Bandar Seri Iskandar
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

**Weather Data Transmission Driven
By Artificial Neural Network based Prediction**

By

Muhammad Zulhilmi bin Md Salahan
16127

A project dissertation submitted to the
Information & Communication Technology Program
Universiti Teknologi PETRONAS
In partial fulfillment of the requirements for the
BACHELOR OF TECHNOLOGY (Hons)
(INFORMATION & COMMUNICATION TECHNOLOGY)

Approved by,

(DR NORDIN ZAKARIA)

Project Supervisor

UNIVERSITI TEKNOLOGI PETRONAS
TRONOH, PERAK

May 2015

CERTIFICATION OF ORIGINALITY

This was to certify that I am responsible for the work submitted in this project, that the original work was 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.

(MUHAMMAD ZULHILMI BIN MD SALAHAN)

Abstract

Nowadays, the trend of big data that can be describe as a massive volume and unstructured data have become more complicated because of its difficulty to be process using traditional database and software techniques. Due to increase in size of data, there also demand for big bandwidth for the transmission of big data. Data transmission is important in communication in providing information at different location. In this project we focus on big data transmission in the context of weather data. Weather data is important for meteorologist as it helps them to make weather prediction. Real time weather prediction is really important as it would help in making quick decision to react with the environment and planning for our daily activities. The purpose of this project is to develop a real time and low bandwidth usage for weather data transmission driven by an artificial neural network perform weather forecast using Adaptive Forecasting Model. This project seeks an application context offshore because the data transmission from offshore to onshore is very costly and requires high usage of network bandwidth. Other than that, offshore weather can change rapidly and cause offshore activity to be delayed.

This research is important as it would help in making faster decision for oil and gas daily activity and avoid loss of human life in natural disaster in the fastest way. In this research we propose a neural network based prediction to control weather data transmission. This system mainly has several parts which consist of data gathering module, neural network prediction module and transmission module. The data gathering module is developed using ODroid weatherboard sensor while the prediction module is developed using an artificial neural network technique and algorithm to make prediction. The transmission module is based on TCP/IP protocol. The prediction module works by using the data captured by sensor to be processed using neural network prediction. If the neural network predict major changes in weather condition or detect bad weather, it will send the weather data and result to the server so that the data can be published and alarm meteorologist and people on the oil platform to avoid loss of human life or another contingency plan on the current oil and gas activity. But if there are no major changes in weather prediction result, the module will only update the server on a daily basis. In this

project, we consider to capture atmospheric pressure, temperature, humidity, visibility, Infrared index, altitude and UV index. Real time processing and transmission weather data will show improvement in predicting weather and saving the network bandwidth transmission.

ACKNOWLEDGEMENT

I would like to express my deepest appreciation and gratitude to the following people in helping me completing the whole final year project.

First of all, I would like to offer my sincere gratitude to my supervisor, Dr Nordin Zakaria who has guided and assisted me throughout my final year project with patience and knowledge whilst allowing me to work in my own way. Although with a tight schedule as a Head of High Performance Computing Center, however, he still offers his valuable time to assisting me in completing the whole project even after the office hour. It has been such a great opportunity to be given the trust to proceed with my project proposed that have been presented in the competition.

Besides that, I would like to thank all the parties and friends who willing to assist me in completing the research survey and system evaluation. These responses helped me to acquire as much data and information as much as possible. Additionally, I would like to thank the organization that willing to spend their valuable time in answering my question.

Other than that, I would like to thank my parents for giving me the moral support along the completion of my final year project. Without their support, I would not make it this far in completing the whole final year project. Last but not least, thank God for giving His strength in solving all the challenge in this final year project.

TABLE OF CONTENTS

CERTIFICATION	ii
ABSTRACT	iv
ACKNOWLEDGEMENT	vi
CHAPTER 1 : INTRODUCTION	
1.1 Background of Study	1
1.2 Problem Statement	2
1.3 Objectives	3
1.4 Scope of Study	4
1.5 Relevancy of Project	5
1.6 Feasibility of the Project within the Scope and Time Frame	6
1.7 Comparative between Controlled and Uncontrolled Data Transmission	7
CHAPTER 2 : LITERATURE REVIEW	
2.1 Data Transmission	9
2.2 Artificial Neural Network	13
CHAPTER 3 : METHODOLOGY	
3.1 Data Gathering	25
3.2 Development Methodology	26
3.3 Requirement Definition	28
3.4 System Architecture	29
3.5 Tsunami UDP Data transmission	30
3.6 ANN Modeling approach	34
3.7 Hardware and Software Required	37
3.8 Planned Methodology	43
3.9 Gantt Chart	44
3.10 Key Milestones	45

CHAPTER 4 : RESULTS AND DISCUSSION	
4.1 User needs Assessment & Analysis	46
4.2 Activity Diagram of the System	47
4.3 Hardware Setup	48
4.4 Software setup on Neural Networks	51
4.5 System Interface Screenshot	52
4.6 System Testing Results	55
4.7 How system will help operation	56
CHAPTER 5 : CONCLUSION & RECOMMENDATION	
5.1 Conclusion	57
5.2 Recommendation	59
REFERENCE	60
APPENDICES	63

CHAPTER 1

INTRODUCTION

1.1 Background of Study

Real time weather data transmission is important nowadays to make quick decision and keep out of danger. Weather information is important for planning and making decision for our day-to-day activities. Weather information might help farmers plan for the planting and harvesting of their crops. Weather information also might help in predict natural disaster such as flood and El Niño so that we can make preparation before natural disasters strikes.

In offshore communication, they used satellite communication to communicate with onshore. They used satellite communication to transfer important data in real time and for normal communication. With this long distance communication, the offshore will easily be connected as offshore can be remotely controlled from onshore. With this communication also, no need for many people to be offshore as all decision can be decide from onshore remotely.

The applications of Artificial Neural Network (ANNs) are becoming famous in oil and gas industry. Its usage had increased widely in areas of engineering, communication and economic. ANNs had been proved its success in application of prediction market trends. Therefore, ANN also can be used for weather data transfer prediction as it is expected this model can save network bandwidth.

1.2 Problem Statement

How to improve network bandwidth usage for big weather data transfer.

There are major problems in weather data transmission from offshore platform to onshore that have cause by latency and speed of data transmission. The connection between offshore and onshore usually have limited bandwidth which make big data such as weather data slow and require high bandwidth to transfer. This problem has negatively impact the oil and gas activity on the oil platform as most of the activities require real time and accurate information to determine the production and activity on oil platform.

1.3 Objectives

The aim of this project is to develop a real time and effective use of network bandwidth for big weather data transmission driven by artificial neural network based prediction. In order to fulfill the aim the following objectives will need to meet:

- To develop a weather data gathering module based on using Odroid-C1 weatherboard and sensor.
- To develop an Artificial Neural Network modelling approach for weather prediction.
- To develop a real time weather data transmission triggered by neural network weather prediction.

1.4 Scope of Study / Limitation

This project scope of study is about providing a real time weather condition and data transmission gathered from sensor. The reason of using weather data is because of the complexity and extensiveness of weather data that can affect data transmission over the communication network. The continuous of weather data collected making the size of data big to be transferred and process onshore. Thus, by implement neural network to predict weather changes can reduce the usage of network bandwidth for data transfer. The experiment of the project need to be implement in real weather situation gathering weather data from environment so that major changes in weather condition can trigger data transmission and determine the network usage.

This project is using neural network because its ability to solve highly complex pattern recognition using a network of neurons. Neural network also have ability to train the network to identify the trend of weather condition. The accuracy of weather prediction is hard to measure and depend on the sample data set provided for the neural network training. Limited timeframe in doing this project lead to limited time to train the neural network weightage for every link node.

This project will require researcher to learn on the sensor to capture weather data and neural network programming to make prediction. The experiment is using Ethernet cable as a medium for data transmission while the target application is assumed to rely on satellite because the cost of testing data transmission using satellite communication that used on offshore platform is very high and complex to be implemented.

1.5 Relevancy of Project

The significance study of this project is this project can help in optimize the network bandwidth usage by using neural network to predict weather condition that can change at any time without notice by human observation. Based on the prediction of neural network, the data transfer to onshore can be reducing as only important data being transfer to onshore. This weather prediction module also can help in making faster decision in oil and gas daily activities. This project also might help save human life from natural disaster such as flood and thunderstorm.

1.6 Feasibility of the Project within the Scope and Time Frame

Network bandwidth is one of the important parts in data transmission especially for big data transfer. High demand in big data also increase demand for high bandwidth usage. In terms of time frame, this project requires high time commitment in the development process. Since the research period is very short, the process of handling the research outcome and transfer it into working system is quite challenging.

The time frame for this project to be developed is two semester of study. For the first semester the project will be focused on the planning, analysis, research and design phase. Meanwhile, in the second semester will be developing the prototype and usability testing.

Moreover, in term of technical, this project focused on three different features which are gathering weather data using ODroid Weatherboard, making weather forecast by using neural Network and data transmission module to transfer important weather data. Besides that, the function and process of this project is feasible within the time frame as the trend of transmission of big data have become popular nowadays.

1.7 Comparative between Controlled and Uncontrolled Data transmission

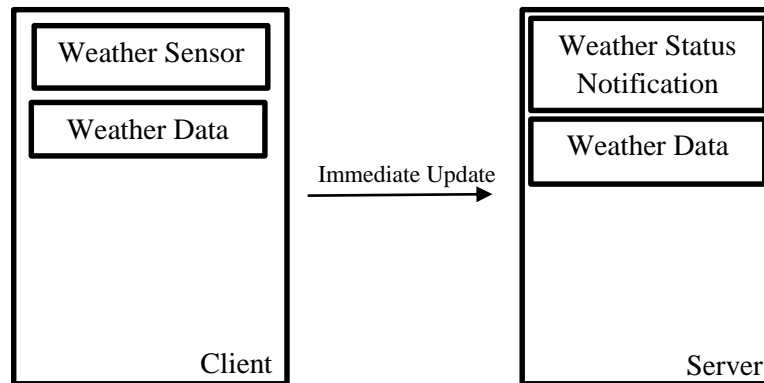


Figure 1 : Uncontrolled Data Transmission

Figure 1 above show example of uncontrolled data transmission between two destinations. In this type of communication, the data collected is directly transferred to another destination. For big data transmission and busy traffic, this could be a problem for the network and can cause the network to jam with data transfer. With this type of transmission the bandwidth demand for network usage is high and can be costly for long distance communication and satellite communication.

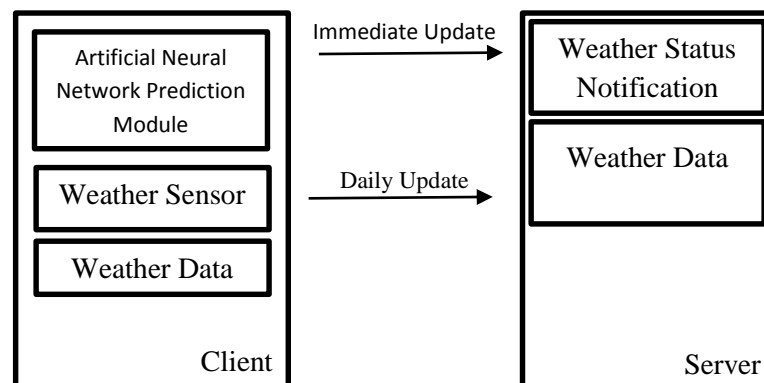


Figure 2 : Controlled Data transmission

Figure 2 above show example of controlled data transmission between two destinations. In this type of communication, the data collected is process and filtered first before transferred to another destination. For big data such as weather data, this can be done using neural network to process the weather data first before being transferred

to another location and filtered the unimportant data to be transfer later to allow important data being transferred first. With this type of transmission, network bandwidth can be saved and less usage thus reduces the cost of data transfer.

CHAPTER 2

LITERATURE REVIEW

2.1 Data Transmission

2.1.1 Basic of Data Transmission

Data transmission is a physical transfer of data from one place to another place (A.P. Clark, 1983). The existing data transmission can be classified into two different categories. Firstly the method in which entire data is transferred from a web server before being process and, secondly the method of streaming which only a part of the entire file is downloaded before processing begins and processing continues while the rest of the file is being downloaded.

There are 2 types of data transmission which is parallel and serial. Parallel transmission usually use for short distance communication and data is transfer simultaneously. This type of data transmission is fast as multiple data is transferred simultaneously at one time but very costly as require multiple numbers of lines to transmit. Serial Transmission used if the transmission long distance communication for a reason of cost and economical to use single line. There are 2 types of serial transmission which is asynchronous and synchronous transmission. Asynchronous transmission send data have start and end bit at interval time. This method of data transmission is cheaper than synchronous if the line is short but it is less efficient and slower due to overhead of extra bits and gaps between bit streams. Synchronous transmission combined bit into longer frames contain multiple bytes and have no gap between transmission. This method much more faster than asynchronous as there are no extra bit and gap between data bytes but it require accurate synchronized clock and long buffers that lead to increase in the cost.

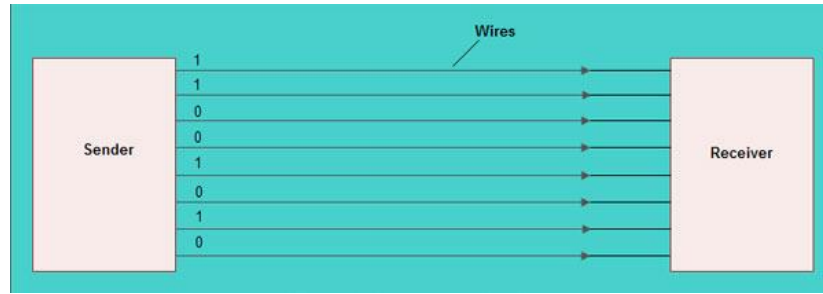


Figure 3: Parallel Transmission

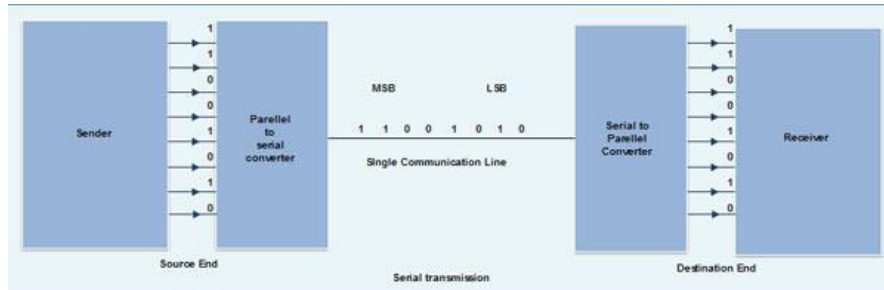


Figure 4 : Serial Transmission for long distance communication

2.1.2 Limited Communication Bandwidth

According to Control and Estimation Methods over Communication Networks (2014), one of the limitations in communication network is can only carry finite amount of information per unit of time. This limitation has become constraint for some application to operate. Examples of Network Control system that are afflicted by severe communication limitation included unmanned air vehicle (UAVs) due to stealth requirements, power starved vehicles such as planetary rovers, long endurance energy limited systems such as sensor networks, underwater vehicle and large arrays of micro actuators and sensor.

2.1.3 Offshore Data Transmission

According to Circadence (www.circadence.com/markets/oil_and_gas) website about oils and gas market, data transmission between offshore and onshore have become much more expensive as the increase in size of data collected. This big data is important for the expert to be analyses and make prediction and proper decision for oil and gas

industry. There are several problems in offshore data transmission such as low bandwidth, high latency and limited connectivity.

According to reference book (Communication Links for Offshore Platform, 2012), there is an increase demand for bandwidth for offshore, oil and gas production platform especially when one platform is a hub and provide radio links to remote platform in the vicinity of the hub platform. The increase demand for capacity is not just because of the number of platforms but due to the need for more services for voice, data, SCADA equipment, video surveillance, Internet and corporate LAN access, and channel to collocated radio networks such as ground to air and marine. Typically, most oil and gas operators are looking for in excess of 16Mb/s for offshore links, to replace low rate (1Mb/s), legacy VSAT links.

For years, satellite technology have served offshore drilling operation by incorporating rigs into corporate communications infrastructures, even where wires and towers can't reach and bringing them digitally closer to headquarters. However, as drilling technology continues to advance, the demand for bandwidth-intensive data transmission from operation to land has skyrocketed. To remain competitive, drillers now require real time data regarding operations and often data needs to be collected from remote, deep areas of the ocean.

For Certain operation, subsea fiber network offer most effective solution and the necessary bandwidth to provide real time transmission of drilling and platform data eliminating potential lag time for communication with headquarters (Rick Simonian, 2012).

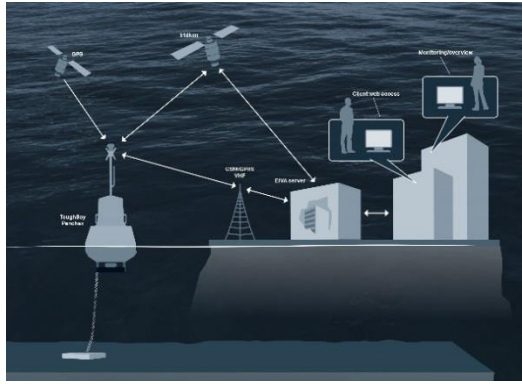


Figure 5 : Sample communication between offshore and onshore

2.2 Artificial Neural Network

2.2.1 Basic Theory

ANN generally mimic in structure and behavior human brain (S. Mohaghegh, 1995). An ANN is designed for specific application for example pattern recognition or data classification. With the capability of ANN, it cans effectively analyzing with simple mathematical method and dealing with non-linear and complex relationship [12], therefore it is the best model to predict the weather data transmission. An ANN is made up of one input layer, one or more hidden layers and one output layer. These layers are connected with the artificial neuron that has ability to stimulate the process and transmit the result to the output. The neurons are organized in different forms subject on the desired type of network. The ANN configuration (figure 4) can be as shown as follows.

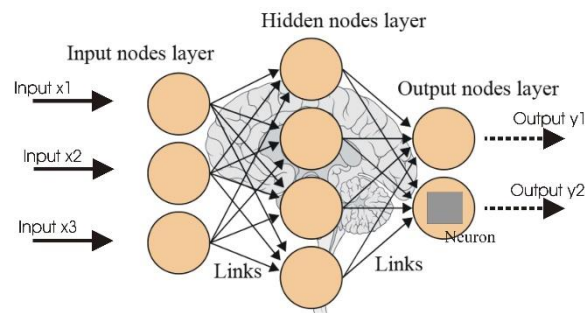


Figure 6: Basic structure of an Artificial Neural Network

2.2.2 History of Artificial Neural Network

The basic disciplines of neural networks originate from an understanding of the human brain. Human brain have an average of 3×10^{10} neurons of various types with each connected by up to 10^4 synapse (Muller B, Reinhardt J & Strickland M ,1995). Neural network is attractive since they consist of many neurons that each process information separately and simultaneously. All the neurons are connected by synapse with variable weights. Therefore, neural network are actually a parallel distributed processing system. The structure of ANN consists of input nodes layer, hidden layer and output nodes layer. Each layer has several numbers of nodes that responsible to send

signal to the next interconnected nodes. Each link interconnected between nodes has their own weighting system and bias that use to evaluate the output decision.

2.2.3 The Neuron

Neuron is a basic building block and processing unit in a neural network. The basic neuron is consists of a black box with weighted inputs and output. It is a node that process all fan in from other nodes and generates an output according to a transfer function called activation function. The activation function represents a nonlinear or linear mapping from the input to the output. A neuron is linked to other neurons by variable weights. Figure 4 shows the simple neuron model proposes by McCulloch and Pitts (McCulloch WS & Pitts W, 1943).

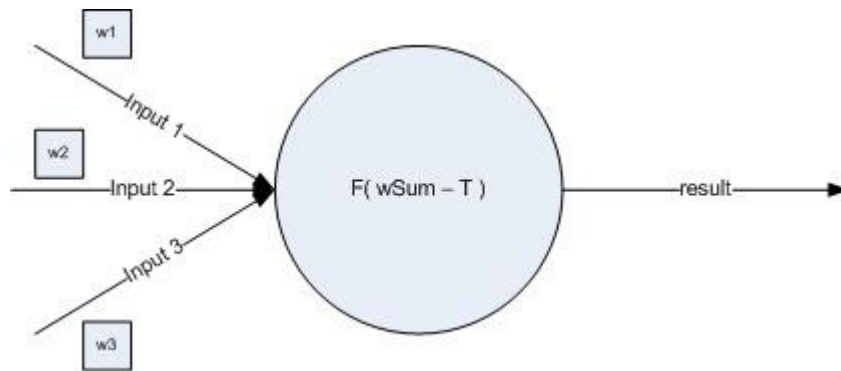


Figure 7: The McCulloch-Pitts Mathematical Neuron Model

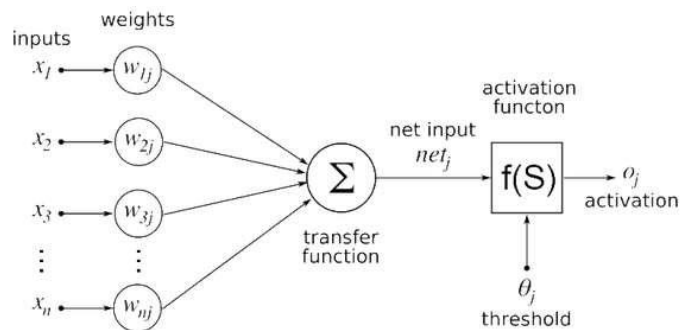
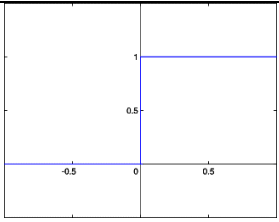
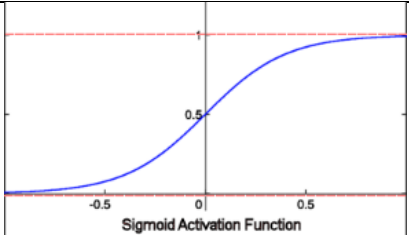


Figure 8: Network of ANN model:

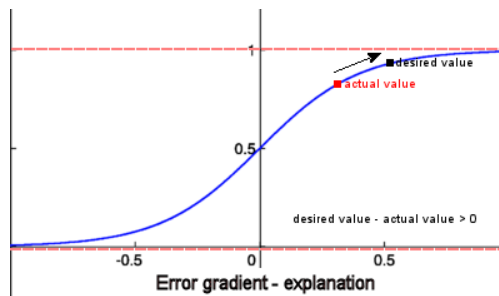
According to Wikibooks (en.wikibooks.org), the activation function is usually some continuous or discontinuous function mapping the real number into the interval (-1, 1) or (0, 1). There are several functions that can be used as activation function such as step function, linear combination, sigmoid function and softmax function. The weights are initializing to some small random values and during training the value get updated. The weighted sum is given below.

$$wSum = \sum_{i=1}^n weight_i \times input_i$$

Function Name	Function Formula	Function Graph
Step Function	$A1 = 1$ $A0 = 0$	
Linear Combination	$y = \zeta + b$	
Sigmoid Function	Continuous Log $\sigma(x) = \frac{1}{1 + e^{-x}}$ $\frac{d\sigma(x)}{dx} = \sigma(x) \times (1 - \sigma(x))$ Continuous tan $\sigma(t) = \tanh(t) = \frac{e^t - e^{-t}}{e^t + e^{-t}}$	
Softmax Function	$y_i = \frac{e^{\zeta_i}}{\sum_{j \in L} e^{\zeta_j}}$	

2.2.4 Neuron Error Gradient

In order to give correct output at the output layer, the weights need to be update. This is the basic of neural network training. Some of method that can be used to update the weight is backpropagation. Its means input is fed in, the error calculated and filtered back though the network making changes to the weight to try reduce the error. The weight changes are calculated by using the gradient descent method which means. The aim is to minimize the steepest path on the error function. Output neurons (desired value – actual value) are multiplied by gradient of the sigmoid function to get the error function. If the value is positive means need to move up the gradient of the activation function and if its value negative need to move down the gradient of the activation function.



Below shows the formula used to calculate the basic error gradient for each output neuron K

$$\delta_k = y_k(1 - y_k)(d_k - y_k)$$

where y_k is the value at output neuron k

and d_k is the desired value at output neuron k

There are difference between the error gradient at the output and hidden layer. The hidden layer error gradient is based on the output layer's error gradient (backpropagation). While the hidden layer the error gradient for each hidden neuron is

the gradient of the activation function multiplied by the weighted sum of the error at the output layer originating from neuron.

$$\delta_j = y_j(1 - y_j) \sum_{k=1}^n w_{jk} \delta_k$$

2.2.5 Weight Update

Weight update algorithm is used to update the weight of neural network to increase the accuracy. The alpha value shown in formula below is the learning rate which usually value between 0 and 1. This value affects how large the weight adjustments are and also affect the learning speed of the network. This value is carefully selected to provide the best result as if the value too low the learning taking long period but when too high the adjustment might be too large and the accuracy will suffer and the network will constantly jump over a better solution and generally get stuck at some sub optimal accuracy.

$$w_{ij} = w_{ij} + \Delta w_{ij} \text{ and } w_{jk} = w_{jk} + \Delta w_{jk}$$

$$\text{where } \Delta w_{ij}(t) = \alpha \cdot \text{inputNeuron}_i \cdot \delta_j$$

$$\text{and } \Delta w_{jk}(t) = \alpha \cdot \text{hiddenNeuron}_j \cdot \delta_k$$

α – learning rate

δ – error gradient

2.2.6 Learning Algorithm

ANN have the ability of learning from the input pattern and their associated output pattern, but before that, ANN should be trained to perform the specific task. The training process is essential is to select the weights and biases using a suitable learning algorithm. The validation division is used for ending the training process to prevent the network from over fitting the data. The test set is used to compare the result for different model.

The neural network learns during a training iteration, that several iteration need to go through to make the neural network have sufficiently learn to handle all the data provided and produce a satisfactory result. Several steps is taken during the learning which first for each input entry in the training dataset the feed input data in (feed forward) then check the output against desired value and feedback error(back propagate). The backpropagation consist of calculate the error gradient and update the weight.

According, to Simon Haykin (2009), *Neural Network and Learning Machine*. Pearson Prentice Hall Company, a neural network has a natural ability for pattern classification. The network is trained by presenting it with it number of different examples of the same object. In this case, weather data collected by Odroid Weatherboard will be trained by using function approximation to determine the hidden layer within the function. The training of the network is repeated for many examples in the set until the network reaches a steady state where there are no further significant changes in the synaptic weights. Thus the network will learn from the data during training session.

2.2.7 Stopping Condition

There are several measure used to decide stopping the learning process of neural network.

- **Maximum Epochs Reached** – This is the easiest stopping condition, all it means is that the NN will stop once a set number of epochs have elapsed. Epoch is actually an iteration of sample training data.
- **Training Set Accuracy** – This is the number of correctly classified patterns over the total number of patterns in the training set for an epoch. Obviously the higher the accuracy the better but this value is the accuracy on previously seen patterns.
- **Generalization Set Accuracy** – This is the number of correctly classified patterns over the total number of patterns in the generalization set for an epoch. This gets calculated at the end of an epoch once all the weight changes have been completed. This represents the accuracy of the network in dealing with unseen patterns. This value alone cannot be used to measure stopping condition since this could have a much higher accuracy that the training set error.
- **Training Set Mean Squared Error (MSE)** – this is the average of the sum of the squared errors (desired – actual) for each pattern in the training set. This gives a more detailed measure of the current networks accuracy, the smaller the MSE the better the network performs.
- **Generalization Set Mean Squared Error (MSE)** – this is the average of the sum of the squared errors (desired – actual) for each pattern in the generalization set.

$$MSE = \frac{\sum_{i=0}^n (\text{desired value } i - \text{actual value } i)^2}{n}$$

where n = number of patterns in set

2.2.8 Time Series Neural Network

Time series is a sequence of vector which depends on time. The model of time series generated by a collection of source can be used not only in classification problem, but also prediction and classification.

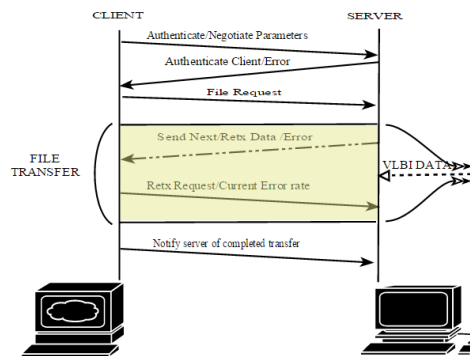
2.2.9 Artificial Neural Network in forecasting

Artificial Neural Network (ANN) has become popular and more frequently used because it has the ability to give solution without the use of conventional mathematical model (C. Siruvuri, S. Nagarakanti and R. Samuel, 2006) (R. A. Neto, N. Ebecken, L. P. Caloba and R. P. Bedregal) and it is not new. ANN is a powerful data modelling tool use complex computational to capture and represent complex input and output relationship that inspired from biological neural network. ANN was introducing together with Knowledge-Based Expert system (KBE) and Artificial Intelligence (AI) (T. Sadiq and I. Nashawi, 2000). ANN has proved that they were outstanding in predictive tools. Prediction of pattern and trend are example of success of ANN (T. Sadiq and I. Nashawi, 2000). ANN is a method which has ability of receiving signals and making proper response for the desired output through pattern recognition or data classification. ANN have very special characteristic such as it has large degree of freedom, makes them to solve the non-linear inside the system. Besides it also can adapt and ability to learn to the new environment. Therefore, with the introduction of new data, ANN can evaluate the data. In addition, ANN has the input and output mapping ability and has fault tolerance. With these qualities, ANNs are superior to conventional regression techniques.

The reason of this neural network is to develop an artificial system that could perform “intelligent” tasks similar to those performed by the human brain. Artificial Neural Network have been used in many fields like pattern recognition, signal processing, function approximation, and process simulation (Guo et al 2001). The neural

network is used to study the weather data transmission based on prediction and the establish model of neural network to provide efficient data transfer and saving bandwidth. The study on prediction of the weather by established a Backpropagation Neural Network (BPNN). The application of ANN is used in prediction of the weather condition and weather data transfer in saving the network bandwidth. This study was revealed another successful of ANN in application of data transmission which achievement in efficient big data transfer to save network bandwidth by using data priority.

2.2.10 Tsunami UDP (TCP & UDP)



A fast user-space file transfer protocol that uses TCP control and UDP data for transfer over very high speed long distance networks designed to provide more throughput than possible with TCP over the same networks. The original early Tsunami UDP transfer protocol developed and released to the public in 2002 by Mark Meiss et al. at the Pervasive Technology Labs at Indiana University (IU). After release, several people have improved the code and have added more functionality. Currently two branched versions of the Tsunami UDP protocol, generic and real-time, are maintained by Metsähovi Radio Observatory / Jan Wagner.

Tsunami is one of several currently available UDP-based transfer protocols that were developed for high speed transfer over network paths that have a high bandwidth-delay product. Such paths can for example be found in the European GEANT research

network. It can for example be a route from a local server PC to a local GEANT2 access node such as FUNET or SURFNET, then via GEANT's internal 10G to another country, and finally a local link via another node such as NORDUNET to some client PC. Currently these network links are capable of 1G..10G and can have some hundred milliseconds of roundtrip delay between the client and server PCs.

Custom UDP protocols are needed because average TCP/IP is not very well suited for paths with a large bandwidth-delay product. To take full advantage of the available bandwidth, the standard TCP slow-start congestion control algorithm needs to be replaced with e.g. HighSpeed TCP, Compound TCP, or one of several others. Some TCP parameters need to be tweaked, such as SACK, dynamic TCP window size. Most of the extended TCP features are already part of Windows Vista, with some partly implemented but not enabled in Windows XP and 2000. In Linux, far more extensive support and a much broader choice of options and congestion control algorithms is available.

Currently Tsunami runs in Linux and Unix. The source code should be largely POSIX compliant. It might be easily ported to Windows XP and Vista as they have a POSIX layer, but, porting has not been attempted yet.

In Very Long Baseline Interferometry, an interferometry based observation method in radio astronomy, the digitized noise recorded from stellar radio sources is often streamed over the network, and there's no requirement for reliable data transport as is guaranteed with TCP. A fraction of data can be lost without degrading the sensitivity of the method too much. In some cases, for example to maintain a high data stream throughput, it may be preferred to just stream the data and not care about transmission errors i.e. not request retransmission of old missing data.

The Tsunami UDP protocol has several advantages over TCP and most other UDP-based similar protocols: it is high-speed (a maintained 900Mbps through 1Gbit NICs and switches isn't unusual), it offers data transmission with default priority for data

integrity, but may also be switched to rate priority by disabling retransmissions. It is the most stable of the other available UDP-based similar protocols. Tsunami is completely implemented in user land and thus doesn't depend on Linux kernel extensions. It can be compiled and run from a normal user account, even a guest account, and does not need root privileges. The global settings for a file transfer are specified by the client - this is useful since the Tsunami user often has a priori knowledge about the quality of their network connection and the speed of their harddisks, and can pass the suitable settings through the client application to the server. Another benefit of Tsunami is that the client already contains a command line interface, no external software is necessary. The commands available in the Tsunami client are similar to FTP commands.

CHAPTER 3

METHODOLOGY

This chapter covers all the methodology that will be used throughout the project. In general, simulation software will be used in the development of model by using ANN approach. After defining the problems and objectives that need to be achieved in Chapter 1, the next step is conducting the literature review on publications, previous works and books relevant to the project. The time allocated for each section is sufficient in order to meet with the project dateline for each chapter. For the studies done by previous authors, most of the information is obtained from books and journals which can be found in the Information Resource Centre as well as websites.

Once a better understanding of the method to determine the data transmission is achieved, the next step to determine the important parameters from the existing methods which will be used as a screening process in our methodology section. After writing the methodology, the next step is data collection and the simulation process.

3.1 Data Gathering

The research methodology requires gathering relevant data from the specified papers for literature review and gathering information from web search. These are the method and platform used to collect data and information about the project.

1. Online Resources

ODroid dokuwiki (<http://odroid.com/dokuwiki/doku.php?id=en:odroid-c1>) is the most useful resources to learn the ODroid hardware. This website provides information about the hardware architecture and experts to help new developers in open source hardware. IEEE website also other online resources that provide information about pass study about data communication. All this online resource help researcher to gather information and gain knowledge on current trend in data communication technology and identify the problem to be solve using knowledge that researchers have.

2. Academic Research

Through Google Scholar (<http://scholar.google.com>), researcher able to grab few researches and literatures on similar topic of what will assess. UTP Library resources also being used to gather more data related to the topic. Through reading and analyzing the academic research, this give more insight in current trend and problem in data communication and help researcher recognize issues and gap to be solved.

3. Interview from experts

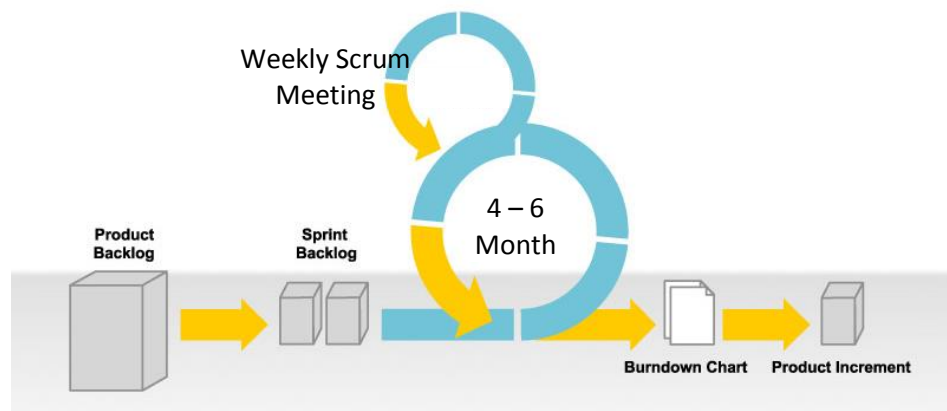
Researcher also has sought experts to gather data and advice from data communication industry and academic. In industry, researcher receives some knowledge and information from expert during industrial training about the data

communication and method what should be used for the execution. In academic, researcher seek theoretical and conceptual advice from supervisor, lecturer and postgraduate who working on the same niche; data communication.

4. Empirical studies, personal experience and keen experience in data communication

University only provides the basic theory of this area (artificial neural network and data communication), so this require researcher to learn the advance topic and the way to implement this area in real life. By learning from online tutorial and reading few titles on the particular topic help researcher to have deep understanding on the research area.

3.2 Development Methodology



In this project, the “Scrum methodology” was implemented. Scrum is an incremental agile software development methodology for managing product development. In this model, the whole requirement is divided into various builds. Multiple development cycle take place making each module smaller and more easily manage modules. During the first module software lifecycle, a working version of software is produced and each subsequent release of the module adds the function to the previous release. This process continues until the complete system is achieved. Scrum is

a framework which people can address complex adaptive problem while productively and creatively delivering product of the highest possible value.

3.2.1 Scrum Roles

1. Product Owner: Muhammad Zulhilmi

Responsibility: Responsible for determine what to be built in the next 7 days or less.

2. Scrum Master : Dr Nordin Zakaria

Responsibility: Ensure the scrum process happens smoothly and help improve the process. During this project, scrum master provide the task to be complete in every sprint cycle that take 1 week. Scrum master

3. Developer: Muhammad Zulhilmi

Responsibility: Develop the system based on item determined by product owner

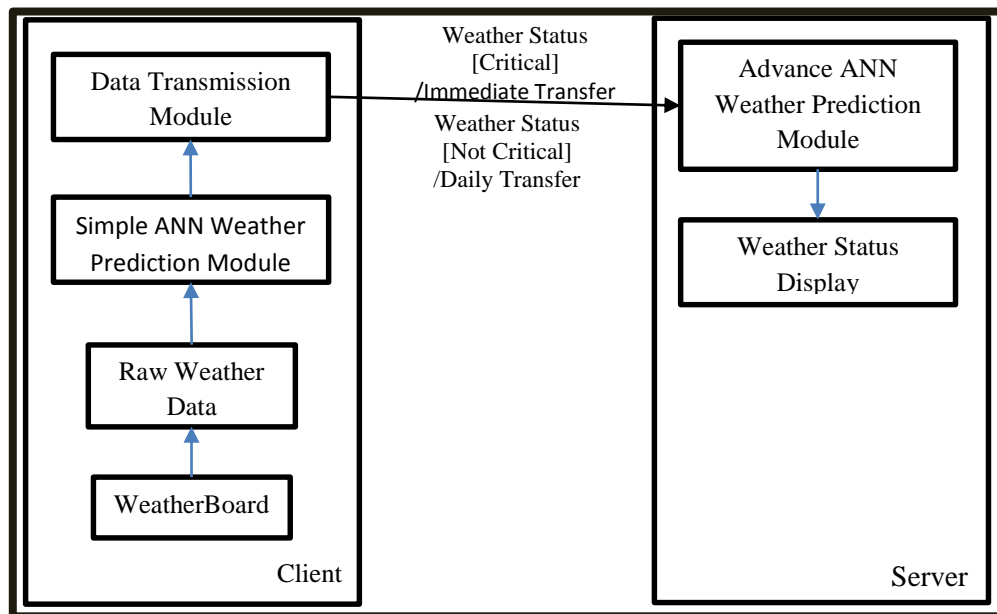
3.3 Requirement Definition

The existence of this project is clearly defined by the problem statement which mainly involve in bandwidth network usage. Hence the requirement of this project is analyze via validation of background of study as well as definition of study scope and objective. In addition, literature review is carried out to further understand the data mining algorithm that have been develop to serve the purpose of saving network bandwidth transmission for weather data and prediction on offshore oil platform using artificial neural network.

The goal of this project is to develop an ANN that is able to predict weather and trigger data transmission for major weather changes. In order to meet the objective, resource and data set of raw weather is determined because this information is crucial in providing input data. Training and testing of the algorithm required as many dataset as possible to increase the accuracy as well as validity of the output patterns. Last but not least, the software used to fulfill all this requirement is Arduino IDE and C++ application with neural network function is prepared in a platform without sophisticated programming skills are needed. Defining of such necessary requirement could ease the task in later stage.

3.4 System Architecture

During the design phase, system architecture is established to identify and describe the fundamental structure abstraction as well as process flow of the whole system. The system architecture consisted of 3 core focuses of the whole system which are weather data gathering, weather forecasting and weather data transmission. Collected weather data is process to make it readable by neural network. Supervised learning is adopted in ANN learning process as training dataset and target dataset are provided to the ANN in order to recognize the pattern of training dataset and categorized it based on the target dataset. Result produced is in binary form and it is converted into understandable phrase by the system to enhance system's user friendliness.



As shown above, the neural network prediction is used to make weather forecast and then if there are major changes in weather condition, the system will transfer the data to the main server for further process. But if there are no major changes in weather condition, the system will only update the server on daily basis. This concept believes can reduce the network bandwidth usage by allowed only important data being transfer.

3.5 Tsunami UDP Data transmission

Tsunami performs a file transfer by sectioning the file into numbered blocks of usually 32kB size. Communication between the client and server applications flows over a low bandwidth TCP connection. The bulk data is transferred over UDP. Most of the protocol intelligence is worked into the client code - the server simply sends out all blocks, and resends blocks that the client requests. The client specifies nearly all parameters of the transfer, such as the requested file name, target data rate, blocksize, target port, congestion behavior, etc, and controls which blocks are requested from the server and when these requests are sent.

Clients connect to the server and authenticate with a MD5 hash in challenge-response using a shared secret. The default is "kitten". In the current client, per default the user is not asked for a password. The client starts file transfers with a get-file request. At the first stage of a transfer the client passes all its transfer parameters to the server inside the request. The server reports back the length of the requested file in bytes, so that the client can calculate how many blocks it needs to receive. Immediately after a get-file request the server begins to send out file blocks on its own, starting from the first block. It flags these blocks as "original blocks". The client can request blocks to be sent again. These blocks are flagged as "retransmitted blocks" by the server.

When sending out blocks, to throttle the transmission rate to the rate specified by the client, the server pauses for the correct amount of time after each block before sending the next. The client regularly sends error rate information to the server. The server uses this information to adjust the transmission rate; the server can gradually slow down the transmission when the client reports it is too slow in receiving and processing the UDP packets. This, too, is controlled by the client. In the settings passed from client to server at the start of a transfer, the client configures the server's speed of slowdown and recovery/"speed-up", and specifies an acceptable packet loss percentage (for example 7%).

The client keeps track of which of the numbered blocks it has already received and which blocks are still pending. This is done by noting down the received blocks into a simple bitfield. When a block has been received, in the bitfield the bit corresponding to the received block is set to '1'. If the block number of a block that the client receives is larger than what would be the correct and expected consecutive block, the missing intervening blocks are queued up for a pending retransmission. The retransmission "queue" is a simple sorted list of the missing block numbers. The list size is allowed to grow dynamically, to a limit. At regular intervals, the retransmission list is processed - blocks that have been received in the meantime are removed from the list, after which the list of really missing blocks is sent as a normal block transmission request to the server. When adding a new pending retransmission to the client's list makes the list exceed a hard-coded limit, the entire transfer is reinitiated to start at the first block in the list i.e. the earliest block in the entire file that has not been successfully transferred yet. This is done by sending a special restart-transmission request to the server.

When all blocks of the file have been successfully received, the client sends a terminate-transmission request to the server. During a file transfer, both server and client applications regularly output a summary of transfer statistics to the console window, reporting the target and actual rate, transmission error percentage, etc. These statistics may be used in e.g. Matlab to graph the characteristics of the transfer and network path. All client file I/O is performed in a separate disk thread, with a memory ring buffer used to communicate new data from the main process to the I/O thread for writing to disk.

In pseudo-code, the server and client operate approximately like this:

Server	Client
<pre> start while(running) { wait(new incoming client TCP connection) fork server process: [</pre>	<pre> start, show command line while(running) { read user command; switch(command) { case command_exit: { clean up; exit; } </pre>

<pre> check_authenticate(MD5, "kitten"); exchange settings and values with client; while(live) { wait(request, nonblocking) switch(request) { case no request received yet: { send next block in sequence; } case request_stop: { close file, clean up; exit; } case request_retransmit: { send requested blocks; } } sleep(throttling) } } } </pre>	<pre> case command_set: { edit the specified parameter; } case command_connect: { TCP connect to server; auth; protocol version compare; send some parameters; } case command_get && connected: { send get-file request containing all transfer parameters; read server response - filesize, block count; initialize bit array of received blocks, allocate retransmit list; start separate disk I/O thread; while (not received all blocks yet) { receive_UDP(); if timeout { send retransmit request(); } if block not marked as received yet in the bit array { pass block to I/O thread for later writing to disk; if block nr > expected block { add intermediate blocks to retransmit list; } } </pre>
---	---

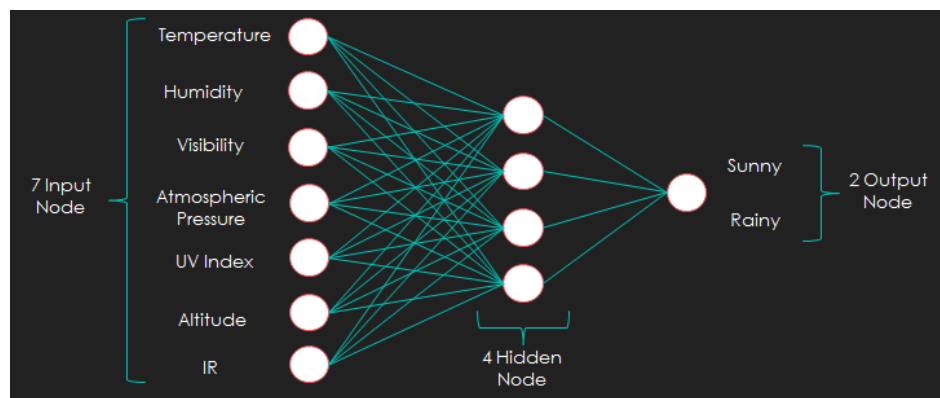
	<pre> if it is time { process retransmit list, send assembled request_retransmit to server; send updated statistics to server, print to screen; } } send request_stop; sync with disk I/O, finalize, clean up; } case command_help: { display available commands etc; } }</pre>
--	--

3.6 ANN Modelling Approach

In this project, the author was selected Backpropagation Neural Network model. The BPNN is the most common supervised learning technique used in training neural network. The term “backpropagation” is the process to minimize the network error by adjusting the weights to resolve closer to desired outputs. As stated before, there are steps to construct the model, BPNN have four major steps (G. Sun, S. J. Hoff, B. C. Zelle and M. A. Smith, 2008): (1) collecting the data, (2) create a network (3) train the network (4) stimulate the network.

The first step was done by determine the parameter that need to predict pore pressure. This process was done by evaluate the parameter that have been used in conventional method. For the network construction, the data set was divided into three subset, training, validation and test set. The ratio of this subset is 6:2:2. The training set was used for calculating the gradient and updating the network weight and biases while validation set was used to improve generalization. The test set was used to validate the network performance that was created. The next step is training the network and then it wills competence to stimulate the network to the test data. The output of this network will be recorded and analyses.

3.6.1 Development of Artificial Neural Network Model



1. **Determine the input parameters** - The input parameter of neural network consist of value that have been collected using weatherboard sensor.

Temperature, UV Index, Altitude, Visibility, Humidity, IR and Pressure is the input parameter for this neural network model.

2. **Determine the number of network layer** - The network layer that researcher use is 3 which consist of input layer, hidden layer and output layer. The input layer will use to input the parameter while the output to show the result of learning process of neural network.
3. **Determine the link condition of the network layers** - The link condition is determined using a weightage calculation set on every link. This weightage value is set at low value before training process of neural network. This value will be update to increase the effectiveness of every link.
4. **The number of neuron on each layers** - For the input layer it consist of 7 input neuron same as number of input parameters used in neural network. While the output layer consist of 1 output neuron to give result on critical weather (raining weather) or not critical weather (sunny weather). The hidden layer consists of 4 as the number of hidden layer usually is between input and output layer.
5. **Activation function** - For the activation function, researcher use sigmoid as this is the most suitable to measure the value that always changing like weather data. The value collected is change every minute making this activation function fit with this project.
6. **Output** - The output of this neural network consist of only 2 output which is critical (raining weather) and not critical (sunny weather). This output will determine whether to send data or not over the network.

3.6.2 Integration and System Testing

This project require integration module between hardware, software and data communication. This project is created separately and the output is used in next module. The module is separated because this module is creating separately in every scrum cycle. This project is incrementally integrated to make it a complete project to control the network data transmission.

First this system needs to be integrating between hardware and hardware which consist of ODroid-SHOW and Weatherboard. This integration is using Arduino IDE programming because the hardware is compatible with Arduino board making the process of integration between this 2 hardware easy.

Second, integration between ODroid-SHOW with Weather Display application in computer. The integration require the application to read value displayed on the ODroid-SHOW and display in the computer screen and log the data in file that later will be used by neural network application. This integration using QT app that use to make User interface of the software. And at the same time write log file of weather data.

Third, integration between Weather Display applications with Neural Network application. This only requires the neural network application to read log data created by Weather Display application and process the output. The integration only about providing readable and structure data for neural network.

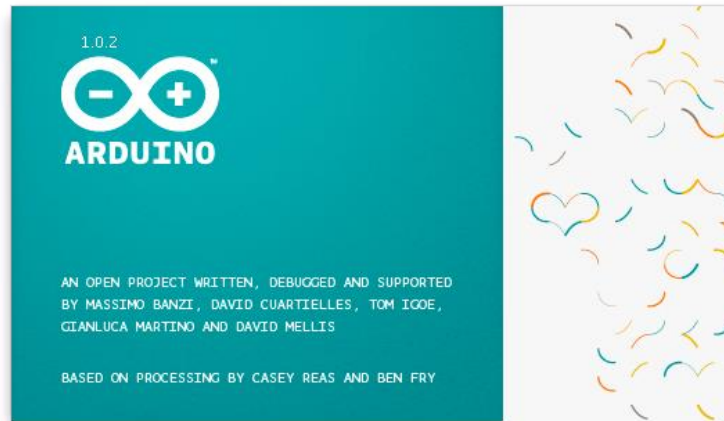
Finally, the integration between neural network applications to data communication module. This integration uses the output of neural network application to trigger the data transmission. This integration of module can control the data transfer thus control the usage of network bandwidth.

3.7 Hardware and Software Required

i. Required Software

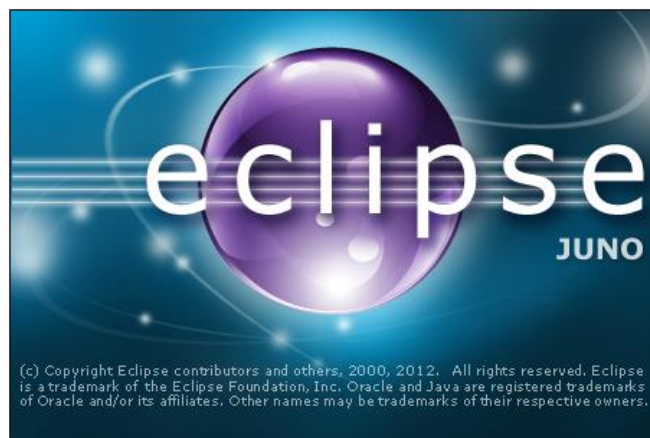
➤ Arduino IDE

- Arduino IDE is an Arduino open source software used to program the ODroid_SHOW. This software using C programming language to control the ODroid display and get the sensor value from weatherboard.



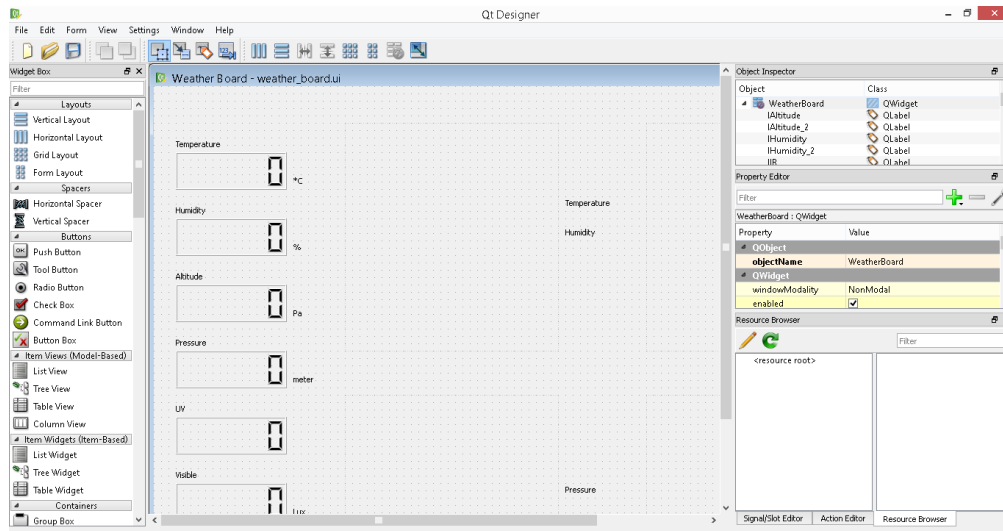
➤ Eclipse

- Eclipse IDE is software use to develop an application. This software can use many languages programming to code. In this project, researcher use to code and design the neural network using C++ using this software.



➤ QT Application

- QT app is a software use to design and simulate the design of software. Researcher use this software to get the value of sensor to be display on screen and generate raw weather data in more structured and organize to be use in Neural Network processing.



ii. Required Hardware

➤ Odroid-C1

Processor	Amlogic S805 : Quad Core Cortex™-A5 processor with Dual Core Mali™-450 GPU
RAM	Samsung K4B4G1646D : 1GByte DDR3 32bit RAM (512MByte x 2pcs)
eMMC module	8GB/64GB : Toshiba eMMC 16GB/32GB : Sandisk iNAND Extreme

socket	<p>The eMMC storage access time is 2-3 times faster than the SD card. You can purchase 4 size options: 8GB, 16GB, 32GB and 64GB. Using an eMMC module will increase speed and responsiveness; similar to the way in which upgrading to a Solid State Drive (SSD) in a typical PC also improves performance over a mechanical hard drive (HDD).</p>	
Micro Secure Digital (MicroSD) Card slot	<p>There are two different methods of storage for the operating system. One is by using a MicroSD Card and another is using an eMMC module, which is normally used for external storage for smartphones and digital cameras. The ODROID-C1 can utilize the newer UHS-1 SD model, which is about 2 times faster than a normal class 10 card.</p> <p>Note that there are some cards which needs additional booting delay time around 30 seconds.</p> <p>According to our test, most Sandisk Micro-SD cards don't cause the booting delay. We will make a compatibility list soon.</p>	
5V2A DC input	<p>This is for 5V power input, with an inner diameter of 0.8mm, and an outer diameter of 2.5mm. The ODROID-C1 consumes less than 0.5A in most cases, but it can climb to 2A if many passive USB peripherals are attached directly to the main board.</p>	
USB host ports	<p>There are four USB 2.0 host ports. You can plug a keyboard, mouse, WiFi adapter, storage or many other devices into these ports. You can also charge your smartphone with it! If you need more than 4 ports, you can use a powered external USB hub to reduce the power load on the main device.</p>	
Micro HDMI port	<p>To minimize the size of the board, we used the Type-D micro-HDMI connector.</p>	
Ethernet RJ-45 jack	<p>The standard RJ45 Ethernet port for LAN connection supports 10/100/1000Mbps speed.</p>	
	Green	Flashes when there is 100Mbps connectivity
	Yellow	Flashes when there is 1000Mbps connectivity
Status /	<p>The ODROID-C1 has four indicator LEDs that provide visual feedback.</p>	

Power LEDs	Red	Power	Hooked up to 5V power
	Blue	Alive	Solid light : u-boot is running Flashing : Kernel is running (hear beat)
Infrared (IR) receiver	This is a remote control receiver module that can accept standard 37.9Khz carrier frequency based wireless data in NEC format.		
Micro USB OTG port	You can use the standard micro-USB connector with Linux Gadget drivers on your host PC, which means that the resources in the ODROID-C1 can be shared with typical PCs. You can also add a micro-USB to HOST connector if you need an additional USB host port. Note that this port cannot be used for power input.		
General Purpose Input and Output (GPIO) ports	<p>These 40pin GPIO port can be used as GPIO/I2C/SPI/UART/ADC for electronics and robotics.</p> <p>The 40 GPIO pins on an ODROID-C1 are a great way to interface with physical devices like buttons and LEDs using a lightweight Linux controller. If you're a C/C++ or Python developer, there's a useful library called WiringPi that handles interfacing with the pins. We've already ported the WiringPi v2 library to ODROID-C1.</p> <p>Please note that pins #37, #38 and #40 are not compatible with Raspberry Pi B+ 40pin header. Those pins are dedicated for Analog input function. Note that all the GPIO ports are 3.3Volt. But the ADC inputs are limited to 1.8Volt.</p>		
Serial console port	Connecting to a PC gives access to the Linux console. You can see the log of the boot, or to log in to the C1 to change the video or network settings. Note that this serial UART uses a 3.3 volt interface. We recommend the USB-UART module kit from Hardkernel.		
RTC (Real Time Clock) backup	<p>If you want to add a RTC functions for logging or keeping time when offline, just connect a Lithium coin backup battery (CR2032 or equivalent).</p> <p>All of the RTC circuits are included on the ODROID-C1 by default.</p>		

battery connector	Molex 53398-0271 1.25mm pitch Header, Surface Mount, Vertical type (Mate with Molex 51021-0200)
Gigabit Ethernet PHY	Realtek RTL8211F is a highly integrated Ethernet transceiver that complies with 10Base-T, 100Base-TX, and 1000Base-T IEEE 802.3 standards.
USB MTT hub controller	GENESYS LOGIC GL852G is used to implement the 4-port Hub function which fully complies with Universal Serial Bus Specification Revision 2.0.
USB VBUS controller	NCP380 Protection IC for USB power supply from OnSemi.
Boot media selector	If this port is opened, the first boot media is always eMMC. If this port is closed, the first boot media is always SD-card.
Power switch port	You can add a slide switch or rocker switch on this port if you want to implement a hardware on/off switch. If this port is closed, the power is off. If this port is opened, the power is on.
Power supply circuit	Discrete DC-DC converters and LDOs are used for CPU/DRAM/IO power supply.
Power protector IC	NCP372 Over-voltage, Over-current, Reverse-voltage protection IC from OnSemi.

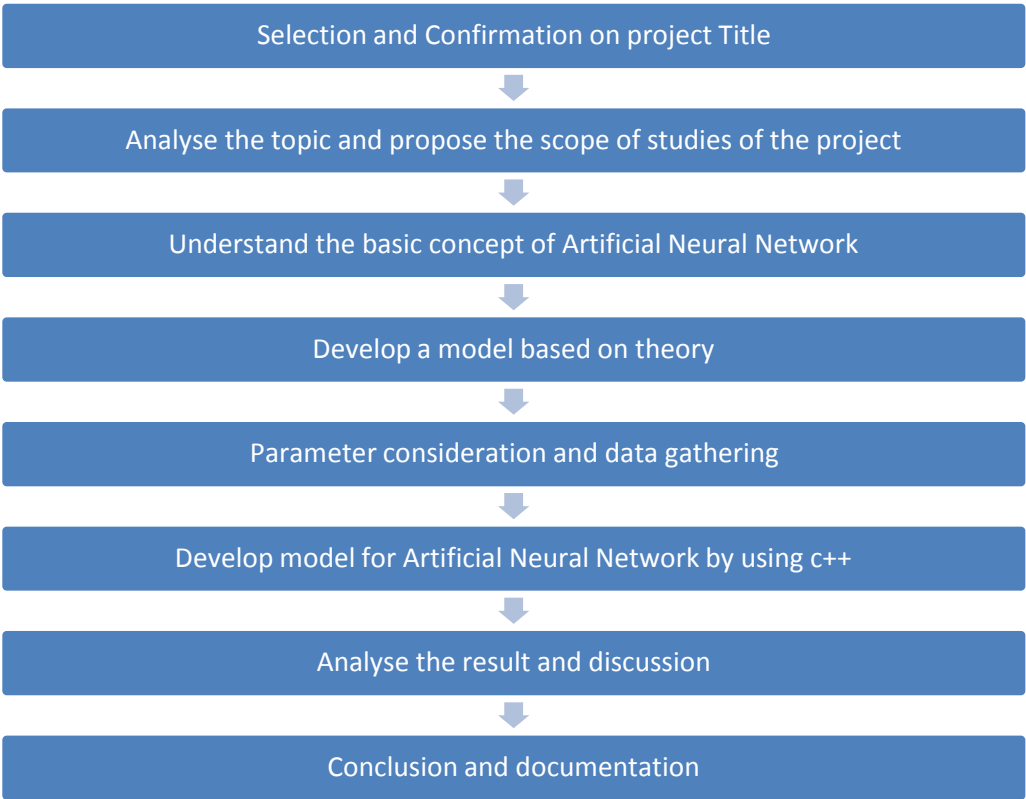
➤ ODroid SHOW

MCU	ATmega328P at 16Mhz
LCD	2.2 inch 240 x 320 TFT LCD (SPI 8Mhz interface)
Host Interface	USB to UART via on-board CP2104
Input Voltage	3.7 ~ 5.5 volt
Power Consumption	60mA @ 5Volt
Serial Port Settings	Baud rate : 500,000 bps (0.5 Mbps), Stop bits : 8-N-1, No H/W, S/W Flow Control
MCU/LCD Voltage	3.45 V from CP2104 on chip regulator

➤ Weatherboard

Interface	I2C
Input Voltage	3.45 V
Current Consumption with Odroid Show	60 mA

3.8 Planned Methodology

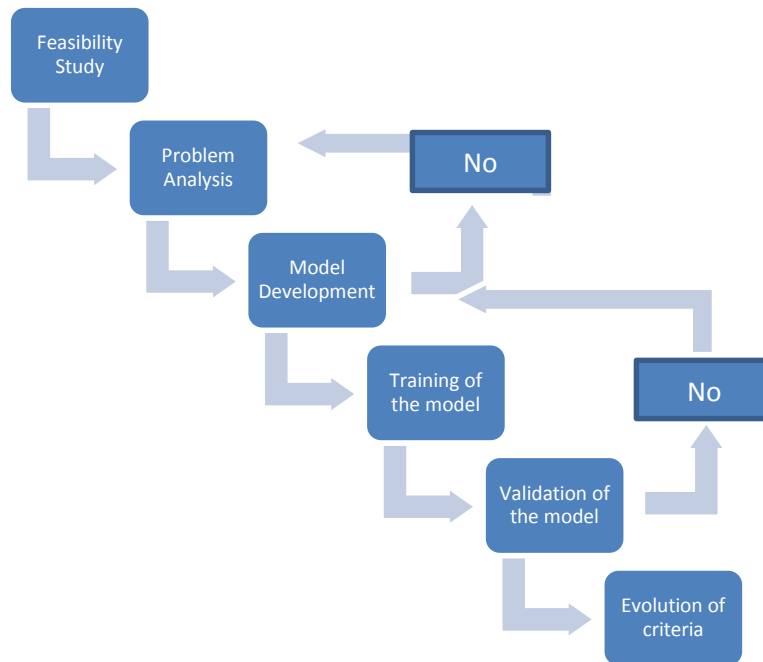


3.9 Gantt Chart

Activity/Period	Week																												
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	
Final Year Project I																													
Planning																													
Discussion with SV	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█	█
Selection of project topic	█	█	█	█																									
Identify the problem				█	█																								
Define objectives of project					█																								
Study on project background						█																							
Preliminary research work						█	█																						
Writing complete proposal							█	█	█	█	█	█																	
Learn Basic Neural Network						█	█	█	█	█	█																		
Learn Hardware architecture					█	█	█			█																			
Data Gathering											█	█																	
Setup Hardware											█	█																	
Implements Neural Network											█	█																	
Final Year Project II																													
Design and Development																													
Design system component												█																	
Prototype development													█																
Design interface													█																
Coding ANN application														█	█	█													
Training																													
Trained the neural network															█	█													
Improvement of prototype																█	█												
Implementation																													
System Implementation																					█	█							
Maintenance																						█	█						
Important Dates																													
Submission of interim report												█																	
Proposal defense													█																
Pre-SEDEX																									█				
Viva																												█	
Project submission																												█	

Table 1: Final Year Project Gantt Chart Planning

3.10 Key Milestones



The key milestones of this project focus on the development of artificial neural network. The feasibility study is the root understanding for this project. The second key milestone is the problem analysis, to analyses the problems in this project that need to solve. The model development is the next stage where the problem had been identified. The training and validation will be the next stage and the result will be determined whether the model is good or bad.

CHAPTER 4

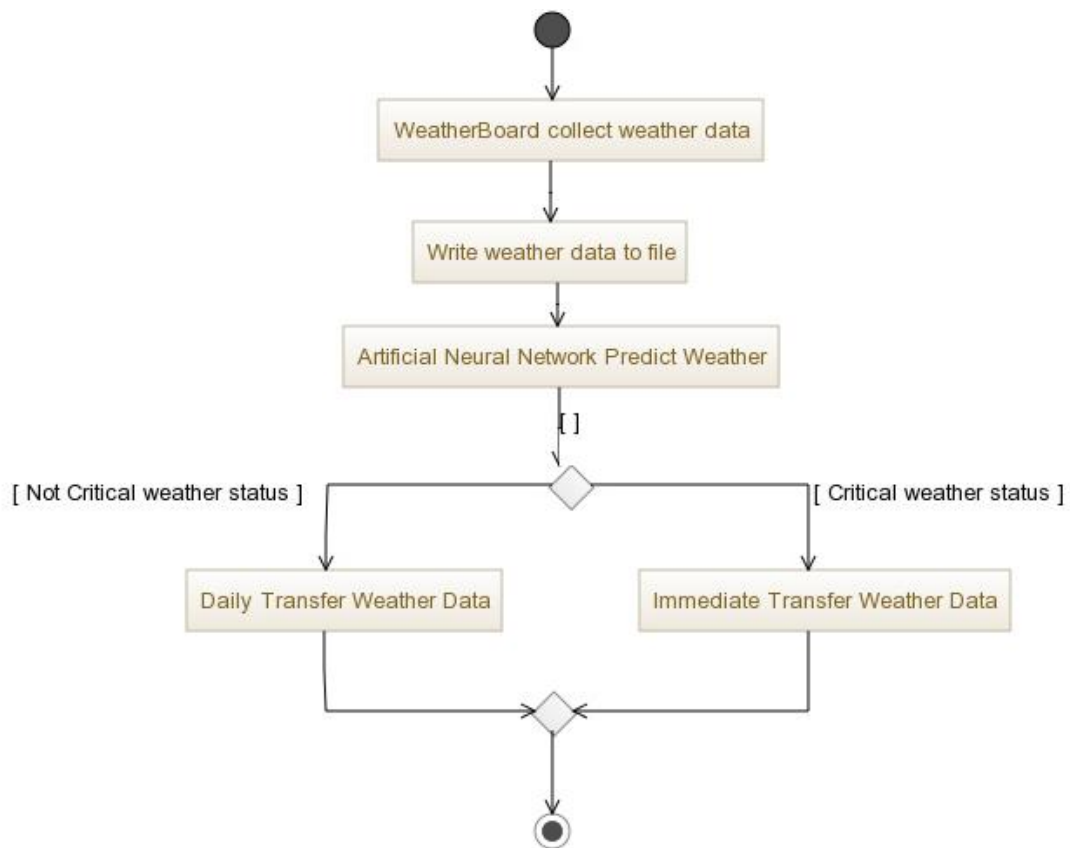
RESULTS AND DISCUSSIONS

4.1 User Needs Assessment and Analysis

According to Asiacube (www.asiacube.com), the data transmission for long distance communication using satellite internet to transfer data. This technology is highly cost as in Malaysia, the cost of the satellite internet service between RM 2500 to RM 4500. This not includes the equipment which cost around RM 7000 to RM 9000. This is for satellite internet for rural area which has speed of 512kbps because of latency.

This showed that for big scale communication used on the offshore surely high cost and problematic with limited bandwidth and slow speed because of network latency. Based on interview with network engineer from Schlumberger also state that this is one of the problem in long distance data communication as Oil and Gas industry need real-time data to process and make decision on operating.

4.2 Activity Diagram of the System



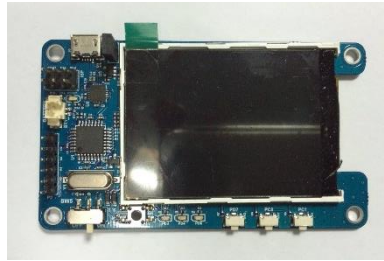
Activity Diagram shows the flow of this system from collection of weather data to transmission of weather data. The system starts with collecting weather data using ODroid weatherboard. Then the weather data is saving in a text file separated by every hour to be read by neural network. After that, the neural network predicts the weather using weather data stored. This neural network application is installing in the ODroid-C1 board. If the weather is not critical then it will only transfer at the end of day when the network not busy. But if the weather is critical which means have big changes in weather or bad condition, the data will be immediately transfer to be further process and notify the people on offshore and onshore. This type of data transmission surely will help reduce the usage of network bandwidth especially during critical or busy network.

4.3 Hardware Setup

Hardware List



ODroid-C1



ODroid-SHOW2



Weatherboard



Micro HDMI Cable





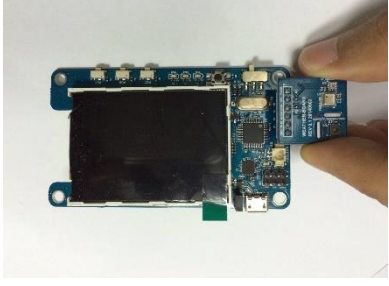




USB-DC Plug Cable 2.5m


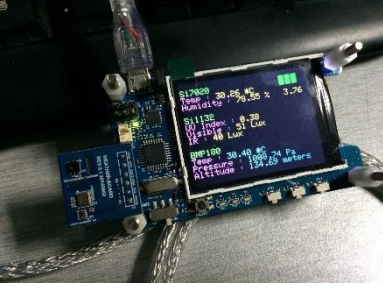


8GB MicroSD UHS-1 Linux

Step by Step Setup Hardware

No	Picture	Instruction
Step 1		<p>Connect the USB-DC Plug Cable to the ODroid-C1. This cable is used to provide 5V power to power on the ODroid-C1 single board computer</p>
Step 2		<p>Connect the Micro HDMI Cable Type D to the ODroid C-1. This cable used to connect the ODroid-C1 to the monitor for screen display.</p>

<p>Step 3</p>		<p>Connect the Weatherboard to ODroid-SHOW2. Weatherboard is add on hardware for ODroid-SHOW2 that use sensor to collect weather data. Plug in also the Micro size Type B cable to the ODroid SHOW2 used to connect between ODroid-SHOW2 and ODrid-C1.</p>
<p>Step 4</p>		<p>Connect the USB 2.0 cable that used to connect between ODroid-C1 and ODroid-SHOW2 to the USB port.</p>
<p>Step 5</p>		<p>Check and make sure all the connection has been made correctly to avoid short circuit on board.</p>
<p>Step 6</p>		<p>Connect the HDMI Cable to the monitor for the display while the ODroid-C1 boot the system.</p>
<p>Step 7</p>		<p>Power on the ODroid-C1 and wait until the booting of Operating System done. The ODroid-C1 sill display the booting process on the monitor.</p>

<p style="text-align: center;">Step 8</p>		<p>After booting complete, open Arduino IDE and open the source code for display sensor value on ODroid-SHOW. Compile the code and burn the code into ODroid-SHOW2. The IDE will show success after burning process complete.</p>
<p style="text-align: center;">Step 8</p>		<p>After done burning the code into ODroid-SHOW, the ODroid-SHOW will start using the code and display all the sensor value on the 2.2inch TFT LCD Module.</p>

4.4 Software Setup on Neural Network

Training Input

Pressure	Altitude	Temperature	Humidity	UV	Visible	IR	Output
1007.11	148.29	39.79	60.03	349.75	45695.00	27543.00	1
1007.14	148.04	39.81	60.01	349.75	45695.00	27527.00	0
1007.14	148.04	39.79	60.04	349.75	45695.00	27510.00	0
1007.12	148.21	39.78	60.02	349.75	45695.00	27493.00	1
1007.13	148.13	39.79	60.04	349.75	45695.00	27480.00	1
1007.12	148.21	39.80	60.04	349.75	45695.00	27476.00	1
1007.14	148.04	39.79	60.06	349.75	45695.00	27469.00	0
1007.10	148.38	39.79	60.06	349.75	45695.00	27464.00	0
1007.16	147.87	39.79	60.04	349.75	45695.00	27469.00	1
1007.11	148.04	39.78	60.04	349.75	45695.00	27466.00	0

Randomly captured Training Input Dataset

Sample input for neural network dataset used to train the network. This dataset consist of 2 part which is the patterns and target. This sample dataset is used to train neural network for letter recognition. The patterns consist of criteria value from table column 1 to column 7. The last column is the target output of the letter recognition. This dataset will be implementing for weather neural network as the structure of dataset almost the same which consist of 7 value of sensor and 1 output as target.

Output

Table above show the neural network weightage for every linkages between nodes after training with sample dataset. The neural network will determine the best weightage in order to make weather prediction accurate based on previous data that used to train the network. Based on the optimal weightage calculated, neural network will predict the weather. Output 1 will evaluate as raining while output 0 as sunny.

4.5 System Interface Screenshot

The system interface of this project consists of 2 which is interface for hardware and software. This 2 interface using different programming language to be develop and provide different output for different user. The hardware interface will displayed using ODroid-SHOW2 to display the data being collected from the weatherboard. The second interface is display on a computer screen that display the data being process by neural network.

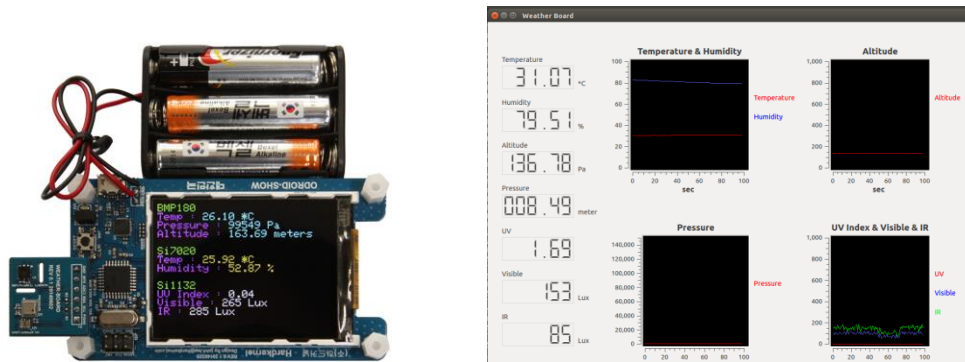


Figure 1 : Data Gathering Module

Expected display on hardware. This interface is displayed on ODroid-SHOW2. This output will display the value of sensor such as temperature, pressure, altitude, humidity, UV index and visibility. All this data is then send to ODroid-C1 to be displayed on screen monitor and save in file for further used by neural network to make weather prediction.

```

C:\Users\wan\Desktop\Project\Final Neural Network Project\Neural Network Analysis.exe
#####
# Weather data prediction using neural network at 1 hour #
# time interval and transfer if probability of prediction #
# above 50% #
#####
Enter server ip address
localhost
Enter request filename
WeatherBoard.csv
Set interval time in day > hour > minute > second
0 0 15
Set max interval time in day > hour > minute > second
0 0 1 0
Timeout : 15
Start Neural Network
Initialising training data
Done execute training data
Read real data
Done read real data
Total Possibility 64 127 : 50.3937
Start data transfer
Complete data transfer

Timeout : 30
Start Neural Network
Initialising training data
Done execute training data
Read real data
Done read real data
Total Possibility 64 127 : 50.3937
Start data transfer
Complete data transfer

```

Figure 10 : Neural network learning process and prediction using console

Expected display for neural network application on console. This neural network process will use terminal to display all the process and calculation. The application will read weather data from file and process the output for future weather condition. Before the neural network can make prediction, the neural need to be train using sample data to determine the best weightage for links connected between nodes.

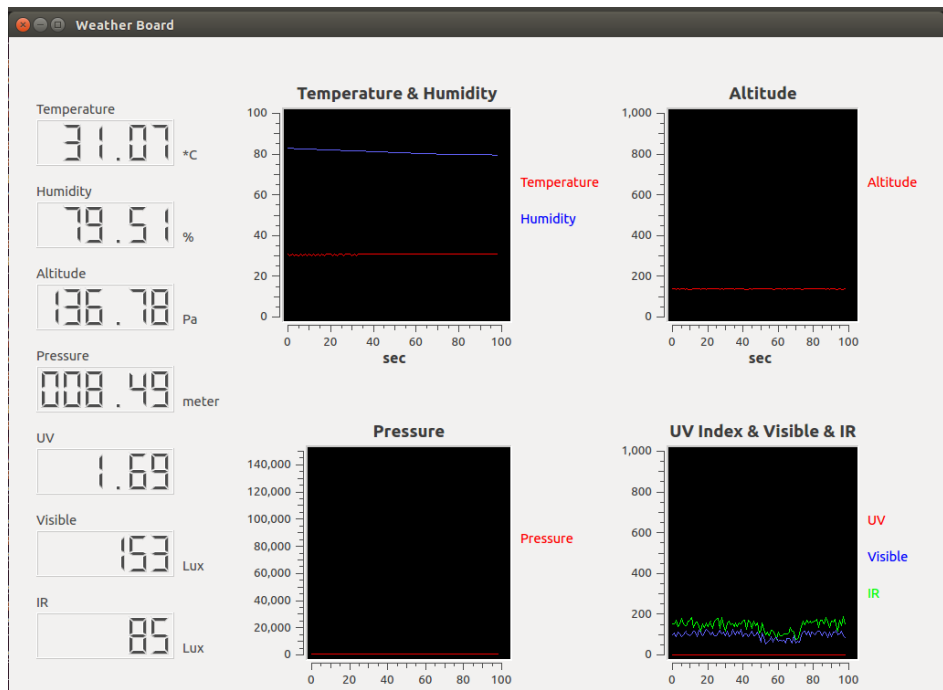


Figure 11 : Expected interface for weather data display on computer

Figure above show the expected interface of weather data collection. This interface is used to display the data collected using weatherboard by reading serial port. This application will also record the weather data and save in file that to be read in neural network application for weather prediction. This interface is develop using QT apps that use GUI to display all the information needed.

```

Block size: 1024
Buffer size: 20000000
Port: 46224
Tsunami Server for protocol rev 20061025
Revision: v1.1 devel cvsbuild 43
Compiled: Jun  8 2015 20:22:53
Waiting for clients to connect.
New client connecting from 127.0.0.1...
Client authenticated. Negotiated parameters are:
Block size: 1024
Buffer size: 20000000
Port: 46224
Request for file: 'WeatherBoard.csv'
Sending to client port 46224
erate   ipd target  block  %done  srvNr
   0 30.00us  12us  4686  58.97   1
Transmsstion of WeatherBoard.csv complete.
Server 1 transferred 8136251 bytes in 0.71 seconds (87.9 Mbps)

```

```

wakjong@wakjong-Aspire-4540:~$ cd Desk
bash: cd: Desk: No such file or directory
wakjong@wakjong-Aspire-4540:~$ cd Desktop/
wakjong@wakjong-Aspire-4540:~/Desktop$ tsunami connect localhost
Tsunami Client for protocol rev 20061025
Revision: v1.1 devel cvsbuild 43
Compiled: Jun  8 2015 20:22:51
Connected.

tsunami> get WeatherBoard.csv
Receiving data on UDP port: 46224

```

time	blk	data	rate	rexmit	blk	data	rate	rexmit	queue	ring	transfer_remaning	OS	UDP
00:00:00.351	4650	8.80K	103.4Mbps	0.0%	4650	8.80K	103.4Mbps	0.0%	0	5	3296	0	--

```

Transfer complete. Flushing to disk and signaling server to stop...
!!!!
PC performance figure : 0 packets dropped (if high this indicates receiving PC overload)
Transfer duration      : 0.71 seconds
Total packet data     : 62.88 Mbit
Goodput data          : 62.88 Mbit
File data              : 62.87 Mbit
Throughput            : 87.91 Mbps
Goodput w/ restarts   : 87.90 Mbps
Final file rate        : 87.90 Mbps
Transfer mode          : lossless

```

Figure 12 : Interface for data transmission module

Figure above show the interface of data transmission module using tsunami UDP transmission protocol. This interface is used to display the file transfer and the rate and performance of data transfer over the network. The interface will show and able to configure the setting of file transfer using tsunami client.

4.6 System Testing Results

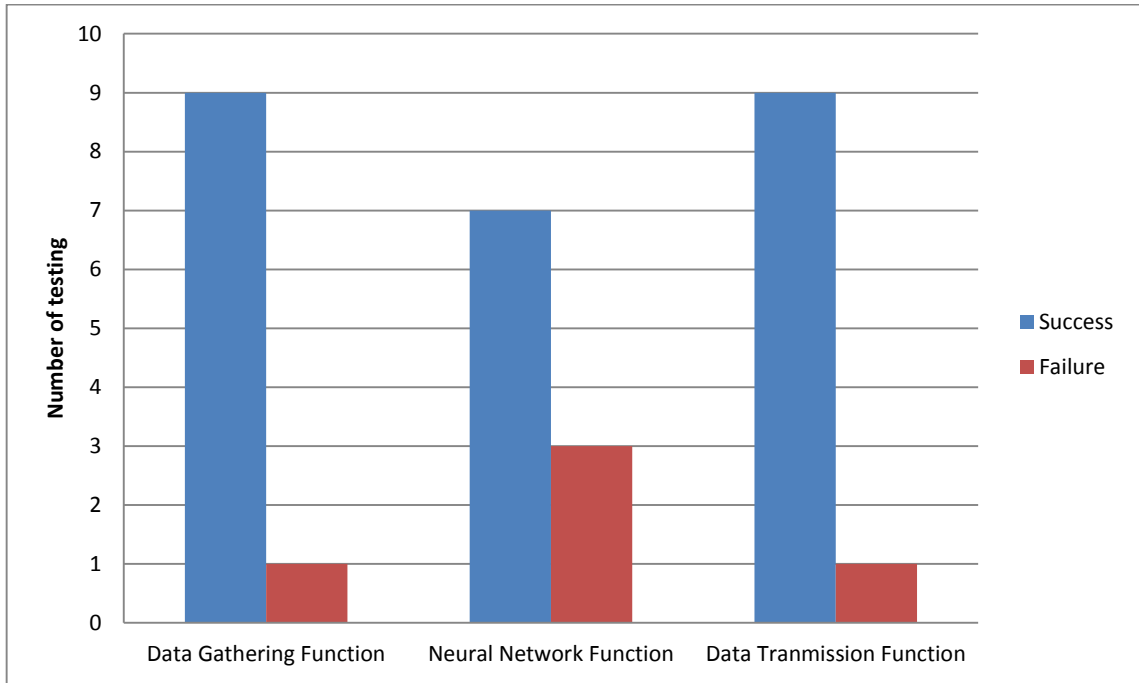


Figure above shows the testing result of module for weather data transmission driven by neural network. It shows that most of the modules worked as expected reduce the network bandwidth usage by neural network controlling the data transmission.

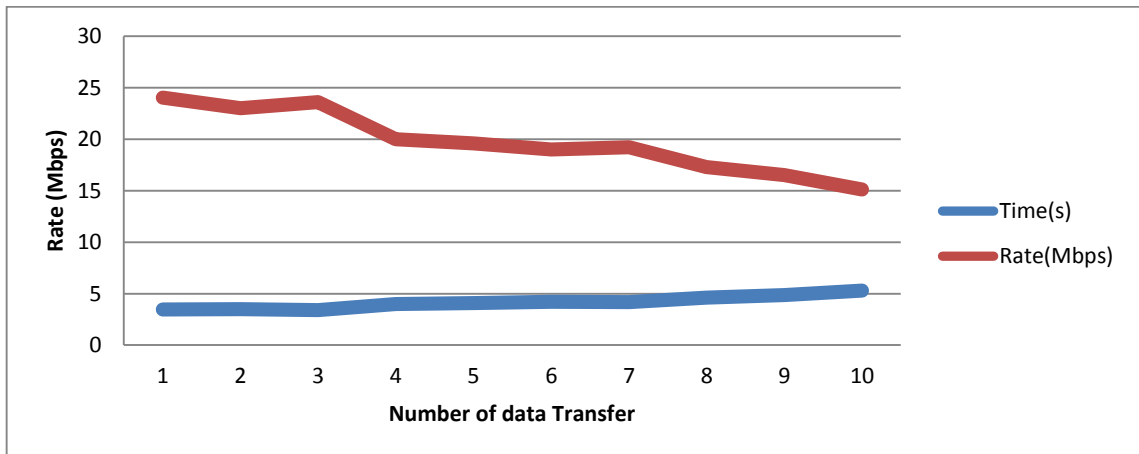


Figure above show the rate of data transfer using the application. The result show the rate is decrease as the time increase. This is because the size of data increases as time increase making the file transfer time longer.

4.7 How System will Help the Operations

This system will help in reducing the network bandwidth usage as the data transfer over the network is controlled by neural network. In offshore data communication, the usage of network was extensively high and use to transfer big data. All this data was not being structured and organize and contained a lot of unwanted data that increase the size of data. This increase in size of data surely will increase the usage of network bandwidth for data transfer. By processing the data first before transmit over network is one of the ways to reduce the size of data transfer. The network control also another way help in optimize the network bandwidth usage as only allow a high priority data being transfer over the network.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

This research consists of 3 parts. The first was learning the hardware architecture and data collection. This part focused on weather data gathering using ODroid Weatherboard. The second part was learning the basic implementation of artificial neural network to predict weather condition based on data collected during part 1. This part researcher learns to develop a C++ application that implement neural network and train the neural network. The final part is learning data transmission to transfer weather data after being process by part 2. This part require researcher to use networking knowledge to transfer the weather data based on priority to control the data transfer thus save the network bandwidth usage.

To prove this assessment, development will be taking place to answer the concept. The idea of saving network bandwidth usage in data communication will give a lot of benefit to control the big data transfer and saving cost. The transmission of weather data is divided into 2 which are immediate transfer and daily transfer. The immediate transfer works when a major change in weather condition is detected while daily transfer works when there are no major changes in weather condition. This weather condition is determine using neural network that use weather data to make weather prediction by provide training to neural network to make the weather prediction accurate.

During the collection of data, the resource about big data transfer from offshore is very limited and learning a new hardware is very challenging. The implementation of neural network also hard to learn as this technique is complicated to develop and not much resource to referred. The communication technology used for data transfer in

offshore also has high cost require researcher to change the data transmission method form satellite data transfer to network cable data transfer.

Based on the result, the network bandwidth usage was reduced as compared with normal data transfer at the same time reduce the cost of data transfer from one location to another location. The results shows that the total data being transferred over the network was reduce and the rate of data transfer also was improve with help of neural network based prediction to trigger data transmission.

5.2 Recommendation

For further recommendation, this application require to do further testing to improve on the performance of data transfer over a long distance transmission. The neural network also could be improve by make the weightage calculation and analysis type wider rather than limited to number of iteration. The traning data set and real data also could be increase by using dynamic array instead of static array as dynamic array make the neural network able to read big data and not cause any problem to the system.

REFERENCE

1. A. P. Clark, "Principles of Digital Data Transmission", Published by Wiley, 1983
2. "Offshore communication", 3 28 2015, [Online] Available: http://www.circadence.com/markets/oil_and_gas
3. Communication Links for Offshore Platform, 2012
4. Rick Simonian (2012, June). Subsea Fiber and Digital Oilfield – Diving into Offshore Communications, *Scandinavian Oil-Gas Magazine* (5), 34-36
5. S. Mohaghegh, "Neural Network: What It Can Do For Petroleum Engineers," SPE Paper, 1995.
6. Muller B, Reinhardt J, Strickland M (1995) Neural Networks : An introduction. 2nd edition, Springer-Verlag, Berlin.
7. McCulloch WS, Pitts W (1943) A logical calculus of the ideas immanent in nervous activity. *Bull of Math Biophysics* 5:115-133
8. "Neural Network Activation Function", 3 28 2015, [Online] Available : http://en.wikibooks.org/wiki/Artificial_Neural_Networks/Activation_Functions
9. C. Siruvuri, S. Nagarakanti and R. Samuel, "Stuck Pipe Prediction and Avoidance: A Convolutional Neural Network Approach," SPE Paper, 2006.
10. R. A. Neto, N. Ebecken, L. P. Caloba and R. P. Bedregal, "Artificial Neural Network use on Simulation of geological process," SPE Paper.
11. T. Sadiq and I. Nashawi, "Using Neural Networks for Prediction of Formation Fracture Gradient," SPE Paper, 2000.
12. "A Basic Introduction to Neural Network," 8 2 2014. [Online]. Available: <http://pages.cs.wisc.edu/~bolo/shipyard/neural/local.html> . 31
13. G. Sun, S. J. Hoff, B. C. Zelle and M. A. Smith, "Development and Comparison of Backpropagation and Generalized Regression Neural Network Models to Predict Diurnal and Seasonal Gas and PM10 Concentrations and Emissions from Swine Buildings," American Society of Agricultural and Biological Engineers, 2008.

14. "Malaysia broadband satellite internet", 3 28 2015. [Online]. Available : <http://blog.asiacube.com/malaysia-broadband-satellite-internet-provider-review.html>
15. Magdi S. Mahmoud, Control and Estimation Methods over Communication Networks, 2014
16. "Introduction", Neural Networks in a Softcomputing Framework, 2006
17. Yong Seok Kim. "A Real-Time Multimedia Data Transmission Rate Control Using a Neural Network Prediction Algorithm", Lecture Notes in Computer Science, 2005
18. Nor Azam Ramli, Norrimi Rosaida, and Hazrul Abdul Hamid. "Future daily PM10 concentrations prediction by combining regression models and feedforward backpropagation models with principle component analysis (PCA)", Atmospheric Environment, 2013.
19. Mohammed Ayoub. "Evaluation of Below Bubble Point Viscosity Correlations and Construction of a New Neural Network Model", Proceedings of Asia Pacific Oil and Gas Conference and Exhibition APOGCE, 10/2007
20. Pazand, Kamran, and Hawshin Feizi. "Using Artificial Neural Networks in Investigations on the Effects of Ultrasonic Vibrations on the Extrusion Process", Applied Mechanics and Materials, 2011.
21. P. Melin. "Neural Network optimization with a hybrid evolutionary method that combines Particle Swarm and Genetic Algorithms with fuzzy rules", NAFIPS 2008 - 2008 Annual Meeting of the North American Fuzzy Information Processing Society, 05/2008
22. Danisman, K.. "Modelling of the hysteresis effect of target voltage in reactive magnetron sputtering process by using neural networks", Surface & Coatings Technology, 20091215
23. Naraghi, Morteza Elahi. "Adaptive Neuro Fuzzy Inference System and Artificial Neural Networks: reliable approaches for pipe stuck prediction", Australian Journal of Basic & Applied Sciences, 2013.
24. Mahmood, M.A.. "Evaluation of empirically derived PVT properties for Pakistani crude oils", Journal of Petroleum Science and Engineering, 199612

25. Tang, Lujia, Lina Wang, Shuming Pan, Yi Su, and Ying Chen. "A neural network to pulmonary embolism aided diagnosis with a feature selection approach", 2010 3rd International Conference on Biomedical Engineering and Informatics, 2010.
26. Nishiyama, K.. "Identification of typical synoptic patterns causing heavy rainfall in the rainy season in Japan by a Self-Organizing Map", Atmospheric Research, 200702
27. El-Sayed Osman. "ARTIFICIAL NEURAL NETWORK MODEL FOR ACCURATE PREDICTION OF PRESSURE DROP IN HORIZONTAL AND NEAR-HORIZONTAL MULTIPHASE FLOW", Petroleum Science and Technology, 1/1/2002

APPENDIX

Weather Sensor Source Code fragments

```
#include <SPI.h>
#include <Wire.h>
#include "TimerOne.h"
#include <Adafruit_GFX.h>
#include <ODROID_Si1132.h>
#include <ODROID_Si70xx.h>
#include <Adafruit_ILI9340.h>
#include <Adafruit_BMP085_U.h>
#include <Adafruit_Sensor.h>

// These are the pins used for the UNO
// for Due/Mega/Leonardo use the hardware
// SPI pins (which are different)
#define _sclk 13
#define _miso 12
#define _mosi 11
#define _cs 10
#define _dc 9
#define _rst 8

const char version[] = "v1.3";

uint8_t ledPin = 5;
uint8_t pwm = 255;
uint8_t textSize = 2;
uint8_t rotation = 1;

float battery = 0;
uint8_t batState = 0;
```

```
uint16_t foregroundColor,
backgroundcolor;

Adafruit_ILI9340 tft =
Adafruit_ILI9340(_cs, _dc, _rst);
Adafruit_BMP085_Unified bmp =
Adafruit_BMP085_Unified(10085);
ODROID_Si70xx si7020;
ODROID_Si1132 si1132;

float BMP180Temperature = 0;
float BMP180Pressure = 0;
float BMP180Altitude = 0;

float Si7020Temperature = 0;
float Si7020Humidity = 0;

float Si1132UVIndex = 0;
uint32_t Si1132Visible = 0;
uint32_t Si1132IR = 0;

void setup()
{
    Serial.begin(500000);
    Serial.println("Welcome to the
WEATHER-BOARD");

    // initialize the sensors
    si1132.begin();
    bmp.begin();
```

```

tft.begin();

sensor_t sensor;
bmp.getSensor(&sensor);

// initialize the digital pins
  initPins();

  analogReference(INTERNAL);

//Timer one setting
Timer1.initialize(200000);
Timer1.attachInterrupt(timerCallback);

tft.setRotation(rotation);
tft.setTextSize(textSize);
tft.setCursor(50, 50);
tft.print("Hello          WEATHER-
BOARD!");
tft.setCursor(250, 200);
tft.print(version);
delay(1000);
tft.fillScreen(backgroundColor);
  displayLabel();
}

void displayLabel()
{
  tft.setCursor(0, 1);
  tft.setTextColor(ILI9340_GREEN,
backgroundColor);

tft.println("Si7020");
tft.setTextColor(ILI9340_MAGENTA,
backgroundColor);
tft.println("Temp : ");
tft.println("Humidity :");
tft.setCursor(0, 5);
tft.setTextColor(ILI9340_GREEN,
backgroundColor);
tft.println("Si1132");
tft.setTextColor(ILI9340_MAGENTA,
backgroundColor);
tft.println("UV Index :");
tft.println("Visible :");
tft.println("IR :");
tft.setCursor(0, 10);
tft.setTextColor(ILI9340_GREEN,
backgroundColor);
tft.println("BMP180");
tft.setTextColor(ILI9340_MAGENTA,
backgroundColor);
tft.println("Temp : ");
tft.println("Pressure :");
tft.println("Altitude :");
tft.drawRect(240, 10, 70, 30, 870);
  tft.fillRect(244, 13, 14, 24, 10000);
  tft.fillRect(260, 13, 14, 24, 10000);
  tft.fillRect(276, 13, 14, 24, 10000);
  tft.fillRect(292, 13, 14, 24, 10000);
}

void initPins()

```



```

{
    pinMode(ledPin, OUTPUT);
    pinMode(3, OUTPUT);
    pinMode(4, OUTPUT);
    pinMode(6, OUTPUT);
    pinMode(7, INPUT);
    pinMode(A0, INPUT);
    pinMode(A1, INPUT);
    analogWrite(ledPin, pwm);
}

void timerCallback()
{
    readBtn();
}

unsigned char btn0Presses = 0;
unsigned char btn0Releases = 0;
unsigned char btn1Presses = 0;
unsigned char btn1Releases = 0;
unsigned char btn2Presses = 0;
unsigned char btn2Releases = 0;
unsigned char btn0Pushed = 0;
unsigned char btn1Pushed = 0;
unsigned char btn2Pushed = 0;

void readBtn()
{
    if (!digitalRead(A1) && (btn2Presses
== 0)) {
        btn2Releases = 0;
        btn2Pushed = 1;
        digitalWrite(6, LOW);
    }
    if (digitalRead(A1) && (btn2Releases
== 0)) {
        btn2Releases = 1;
        btn2Presses = 0;
        btn2Pushed = 0;
        digitalWrite(6, HIGH);
    }
    if (!digitalRead(7) && (btn0Presses ==
0)) {
        btn0Presses = 1;
        btn0Releases = 0;
        btn0Pushed = 1;
        if (pwm > 225)
            pwm = 255;
        else
            pwm += 30;
        analogWrite(ledPin, pwm);
        digitalWrite(3, LOW);
    }
    if (digitalRead(7) && (btn0Releases ==
0)) {
        btn0Releases = 1;
        btn0Presses = 0;
        btn0Pushed = 0;
        digitalWrite(3, HIGH);
    }
}

```

```

    if (!digitalRead(A0) && (btn1Presses
== 0)) {
        btn1Presses = 1;
        btn1Releases = 0;
        btn1Pushed = 1;
        if (pwm < 30)
            pwm = 0;
        else
            pwm -= 30;
        analogWrite(ledPin, pwm);
        digitalWrite(4, LOW);
    }

    if (digitalRead(A0) && (btn1Releases
== 0)) {
        btn1Releases = 1;
        btn1Presses = 0;
        btn1Pushed = 0;
        digitalWrite(4, HIGH);
    }
}

void sendToHost()
{
    Serial.print("w0");
    Serial.print(BMP180Temperature);
    Serial.print("\ew1");
    Serial.print(BMP180Pressure);
    Serial.print("\ew2");
    Serial.print(BMP180Altitude);
    Serial.print("\ew3");

    Serial.print(Si7020Temperature);
    Serial.print("\ew4");
    Serial.print(Si7020Humidity);
    Serial.print("\ew5");
    Serial.print(Si1132UVIndex);
    Serial.print("\ew6");
    Serial.print(Si1132Visible);
    Serial.print("\ew7");
    Serial.print(Si1132IR);
    Serial.print("\ew8");
    Serial.print(battery);
    Serial.print("\e");
}

void getBMP180()
{
    sensors_event_t event;
    bmp.getEvent(&event);
    if (event.pressure) {
        bmp.getTemperature(&BMP180Temperature);
        BMP180Pressure = event.pressure;
        BMP180Altitude = bmp.pressureToAltitude(1025, event.pressure);
    }
}

void getSi1132()

```

```

{
    Si1132UVIndex          = void displaySi7020()
si1132.readUV()/100.0;    {
    Si1132Visible = si1132.readVisible();        tft.setTextColor(ILI9340_YELLOW,
    Si1132IR = si1132.readIR();                backgroundColor);
}
                                tft.setCursor(7, 2);
                                tft.print(Si7020Temperature);
                                tft.println(" *C ");
                                delay(20);
                                tft.setCursor(11, 3);
                                tft.print(Si7020Humidity);
                                tft.println(" % \n");
                                delay(20);
}

void getSi7020()
{
    Si7020Temperature      = tft.setCursor(11, 3);
si7020.readTemperature();    tft.print(Si7020Humidity);
    Si7020Humidity         = tft.println(" % \n");
si7020.readHumidity();      delay(20);
}
}

void displayBMP180()
{
    tft.setTextColor(ILI9340_CYAN,
backgroundColor);
    tft.setCursor(7, 11);
    tft.print(BMP180Temperature);
    tft.println(" *C ");
    delay(20);
    tft.setCursor(11, 12);
    tft.print(BMP180Pressure);
    tft.println(" Pa ");
    delay(20);
    tft.setCursor(11, 13);
    tft.print(BMP180Altitude);
    tft.println(" meters ");
}

void displaySi1132()
{
    tft.setCursor(11, 6);
    tft.print(Si1132UVIndex);
    tft.println(" ");
    delay(20);
    tft.setCursor(10, 7);
    tft.print(Si1132Visible);
    tft.println(" Lux ");
    delay(20);
    tft.setCursor(5, 8);
    tft.print(Si1132IR);
    tft.println(" Lux \n");
    delay(20);
}
}

```

```

void batteryCheck()
{
    battery = analogRead(A2)*1.094/1024/3.9*15.9;
    tft.setCursor(21, 3);
    tft.print(battery);
    if (battery >= 3.95) {
        if (batState != 4) {
            batState = 4;
            tft.fillRect(244, 13, 24, 24, 10000);
            tft.fillRect(260, 13, 24, 24, 10000);
            tft.fillRect(276, 13, 24, 24, 10000);
            tft.fillRect(292, 13, 24, 24, 10000);
        }
        } else if (battery > 3.75 && battery <= 3.95) {
            if (batState != 3) {
                batState = 3;
                tft.fillRect(244, 13, 24, 24, 10000);
                tft.fillRect(260, 13, 24, 24, 10000);
                tft.fillRect(276, 13, 24, 24, 10000);
                tft.fillRect(292, 13, 24, 24, 0);
            }
        } else if (battery > 3.65 && battery <= 3.75) {
            if (batState != 2) {
                batState = 2;
                tft.fillRect(244, 13, 24, 24, 10000);
                tft.fillRect(260, 13, 24, 24, 10000);
                tft.fillRect(276, 13, 24, 24, 0);
                tft.fillRect(292, 13, 24, 24, 0);
            }
        } else if (battery > 3.5 && battery <= 3.65) {
            if (batState != 1) {
                batState = 1;
                tft.fillRect(244, 13, 24, 24, 10000);
                tft.fillRect(260, 13, 24, 24, 0);
                tft.fillRect(276, 13, 24, 24, 0);
                tft.fillRect(292, 13, 24, 24, 0);
            }
        } else if (battery <= 3.5) {
            if (batState != 0) {
                batState = 0;
                tft.fillRect(244, 13, 24, 24, 0);
            }
        }
    }
}

```

```
        tft.fillRect(260, 13, 14, 24,
0);
        tft.fillRect(276, 13, 14, 24,
0);
        tft.fillRect(292, 13, 14, 24,
0);
    }
}
```

```
void loop(void)
{
    getBMP180();
    getSi7020();
    getSi1132();
    displayBMP180();
    displaySi7020();
    displaySi1132();
    sendToHost();
    batteryCheck();
}
```