

CHAPTER 1

INTRODUCTION

1.1 Background of Study

Electricity forecasting is necessary for the control and scheduling of power systems. But the system planners need to confront one problem which is the complexity in electricity demand values. The 1990's brought computer power into load forecasting and with it the opportunity to explore the complexity in the load data. Rule-based and fuzzy logic expert systems have been used to model the complexity in the data using domain knowledge. Although these methods are promising, they rely on rules that are extracted from experts and operators' experience, which are subject to inconsistencies and are thus unreliable.

Artificial neural networks have made it possible to experiment with the rich data that behave non-linear. ANN's models can identify the complex non-linear relationships in the data and infer future behaviour. The basic idea is that the networks learn through examples, which consist of the input signals and desired output. The result is that neural networks have the potential to model a far greater range of relationships than models that have a pre-specified form like ordinary least squares linear regressions [1].

Load forecasting has always been important for planning and operational decision conducted by utility companies. Besides, it is also useful because it can help to estimate load flows and to make decisions that can prevent overloading. Hence, this will lead to the improvement of network reliability as well as to reduce occurrences of equipment failures and blackouts [4].

Load forecasting can be categorized into three categories which are short term, medium term and long term. Short term forecasting means that the duration of load forecasting is from one hour to one week [4]. It represents a great saving

potential for economic and secure operation of power systems [5]. Medium term forecasting means that the load forecasting is ranged from a week to a year [4]. It deals with the scheduling of fuel supplies and maintenance operations [5]. As for long term load forecasting, it means that the load forecasting is for more than a year [4]. This type of load forecasting is useful for planning operations [5]. The main focus of this study is to forecast the electricity demand for short term load forecasting.

The purpose of this study is to forecast the electricity demand of the small scale power system. Hence, the study is conducted on GDC (UTP) as the example of a small scale power system. Co-generation/District Cooling Plant (GDC) for Universiti Teknologi Petronas (UTP) or GDC (UTP) is designed to produce electrical power and steam from Co-generation system (Cogeneration plant) and chilled water from chilled water system (District Cooling Plant). Electrical power produced from Cogeneration system is supplied to UTP and also consumed within the plant. 2 nos of Gas Turbine Generator (GTG) with Heat Recovery Steam Generator (HRSG) are installed in which are able to generate up to 8.4 MW of electrical power. Upon studying the electricity demand and pattern of GDC (UTP), four (4) forecasting models have been developed using Artificial Neural Network (ANN).

1.2 Problem Statement

The industry as for this study is GDC (UTP) has been faced the complexity electricity demand values. The complexity demand values are due to the many factors such as temperature and weather that effect the electricity demand values during generation. The effect of this complexity is that it made the electricity demand values to be complex and nonlinear. Thus, it is hard to analyze. In this project, the complexity electricity demand values can be seen in the data gathered as well as when data has been analyzed.

The complexity electricity demand values make the prediction of the electricity become not easy. Hence, the study is being made using ANN models to help the industry to forecast the electricity demand more accurately. This is based to the fact that Artificial Neural Networks have made it possible to experiment

with theoretically poor, but data rich, models that can identify the complex non-linear relationships in the data and infer future behaviour. The basic idea is that the networks learn through examples, which consist of the input signals and desired output. The result is that neural networks have the potential to model a far greater range of relationships than models that have a pre-specified form like ordinary least squares linear regressions.[1]

The study is also conducted since the models used to solve GDC (UTP) future load demand are not available. Hence, by developing the forecast models using ANN, the future load demand of GDC (UTP) can be predicted.

1.3 Significance of Study

The project has its significance to the GDC (UTP) specifically and the industry generally. The significances of this project are as follows:

1. Study the electricity behavior
2. Formulate the predictive model
3. Forecast electricity demand

One of the significances of this project is to study the electricity behavior of GDC (UTP) load demand. The importance of analyzing the electricity behaviors is that to understand and know the pattern of the load.

Once the pattern has been known, the predictive model can be formulated. The predictive model is being developed using Artificial Neural Network and it is being developed to forecast the electricity demand for several steps ahead.

Third, the project is used to forecast the electricity demand of GDC (UTP). The electricity forecasting is very essential in the power system industry. This is due to the fact that by forecasting the electricity demand for several steps ahead, therefore the exact amount of electricity can be generated at the exact time. Hence, the earlier prediction can lead to the optimization in the power generation as well as the reduction of power wastage during the generation.

1.4 Project's Objectives

The objectives of this study are as follows:

1. To understand the principle of ANN as the method to design the model of load forecasting.
2. To analyze the data gathered from GDC UTP.
3. To design and model the load forecasting model using ANN method
4. To forecast the GDC UTP load demand

1.5 Scope of Work

The scope of work of this study is to understand the principle of ANN. The understanding of ANN is being done by doing the literature review as well as the brief research about the topic. The ANN is used to develop models in order to forecast the electricity demand of UTP. Furthermore, the historical data is being gathered from UTP's power supplier, Gas District Cooling (GDC). The data gathered is for four years data which is from 1st January 2006 till 31st December 2009. The data gathered is then being used for the forecast model developments. Thus, the electricity demand of UTP is being predicted for the duration of seven days and thirty days ahead.

CHAPTER 2

LITERATURE REVIEW

2.1 Principal of ANN

There are many techniques that have been developed and used for short-term load forecasting. There are Similar-Day Approach, Regression Methods, Time Series, Expert Systems, Fuzzy Logic, Support vector machines and Neural Network [4]. The descriptions of each of the techniques are as follows:

Table 1: Description of Short Term Load Forecasting Techniques

Technique	Description	Implemented by
Similar-Day Approach	<ul style="list-style-type: none">• Search the historical data for days with similar characteristics.	Xunming Li, Changyin Sun, Dengcai Gong,” Application of Support Vector Machine and Similar Day Method for Load Forecasting”, Hohai University of China
Expert System (Ruled Based)	<ul style="list-style-type: none">• Make use of rules to do accurate forecasting.• Utilizes the historical relationship between the weather, load and the day to predict load [10].	K.L. Ho, Y.Y. Hsu, F.F. Chen, T.E. Lee, C.C. Liang, T.S. Lai, and K.K. Chen, “Short-Term Load Forecasting of Taiwan Power System using a Knowledge Based Expert System”, <i>IEEE Transactions on Power Systems</i> , 5:1214–1221, 1990.

Fuzzy Logic	<ul style="list-style-type: none"> • Generalization of Boolean Logic that takes on a truth value of ‘0’ and ‘1’ 	E. Srinivas and Amit Jain, “ A Methodology for Short Term Load Forecasting Using Fuzzy Logic and Similarity”, <i>IEEE</i>
Support Vector Machines	<ul style="list-style-type: none"> • Describe the nonlinear relationship between load and influencing factors. • Corrected the forecasted results by the curve of a similar day to avoid the appearance of large forecasting error [9]. 	Xunming Li, Changyin Sun, Dengcai Gong,” Application of Support Vector Machine and Similar Day Method for Load Forecasting”, Hohai University of China.
Time Series	<ul style="list-style-type: none"> • Assume the data have an internal structure and detect and explore the structure. • Forecasts the current value of a variable by means of a linear combination of previous values of the variable, previous values of noise and current value of noise [11]. 	H.T. Yang, C.M. Huang, and C.L. Huan, “Identification of ARMAX Model for Short-Term Load Forecasting: An Evolutionary Programming Approach”, <i>IEEE Transactions on Power Systems</i> , 11:403–408, 1996.
Regression Methods	<ul style="list-style-type: none"> • To model relationship of load consumption and other factors. • The models incorporate deterministic influences such as holidays, stochastic influences such as average loads, and exogenous influences such as weather [4]. 	W. Charytoniuk, M.S. Chen, and P. Van Olinda, “Nonparametric Regression Based Short-Term Load Forecasting” <i>IEEE Transactions on Power Systems</i> , 13:725–730, 1998.
Neural Network	<ul style="list-style-type: none"> • A class of models inspired by biological nervous systems • Trained to learn the relationship between various input variables and historical load 	Mohsen Hayati, Yazdan Shirvany, “Artificial Neural Network Approach for Short Term Load Forecasting for Illam Region”, World Academy of Science, Engineering

	<p>patterns</p> <ul style="list-style-type: none"> • Able to generalize among the training sets and produce a corresponding output [11]. 	and Technology 28, 2007
--	---	-------------------------

Table 1 shows the description of the techniques used for short term load forecasting. Based on the above descriptions, ANN has been chosen as the method for load forecasting. ANN literally means an interconnected assembly of simple processing elements, units or nodes, whose functionality is loosely based on the animal neuron. In this study, ANN is a class of models inspired by biological nervous systems. The models consist of many computing elements and working in parallel. The computing elements are usually being denoted as neurons. Neurons communicate via electrical signals that are short-lived impulses or ‘spikes’ in the voltage of the membrane. Below is the example of neural network:

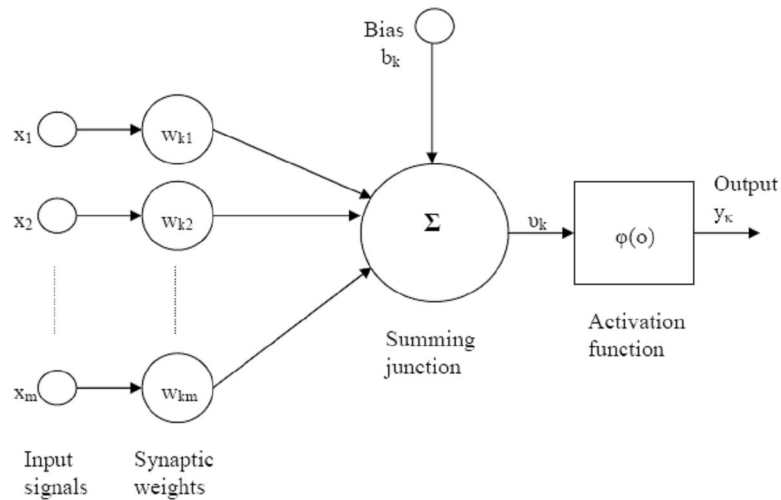


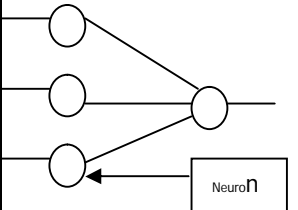
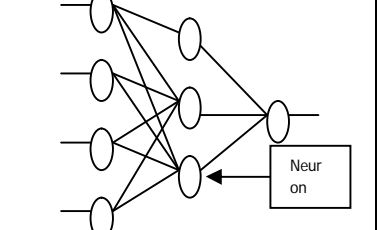
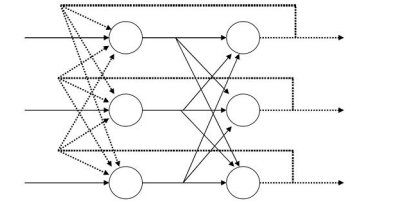
Figure illustrated from Konstantinos Adamopoulos, “Application of BackPropagation Learning Algorithms on Multilayer Perceptions”, University of Bradford Department of Computing, May 2000

Figure 1: The model of Neuron

Figure 1 shows the model of the neuron or also known as computing element. The model consists of three basic elements of the neuronal model: a set of synapses or synaptic (connecting) links, an adder (logical unit) and an activation function (threshold function). ANN has been widely used as the techniques for load forecasting. ANN has been used instead of other techniques most likely because ANN can demonstrate the capability to do non-linear curve fitting.

In applying the techniques, the architectures of the network should be selected from single layer feedforward networks, multilayer feedforward networks and recurrent networks. The comparison between single layer feedforward, multilayer feedforward and recurrent networks are as follows:

Table 2: Comparison between Network Architecture

Single Layer Feedforward	Multilayer Feedforward	Recurrent Networks
Has input layer and output layer.	<ul style="list-style-type: none"> • Has input layer, hidden layer and output layer. • Hidden layer essential in order to extract higher statistics and perform more complicated tasks [2]. 	<ul style="list-style-type: none"> • Have feedback networks that exhibit closed loops. • Feedback connections has a profound impact on the learning capability and the overall performance of the network • Increase the nonlinearity and the complexity of the derived dynamics [2].
<p>Input Layer Output Layer</p> 	<p>Input Layer Hidden Layer Output Layer</p> 	<p>Input layer Output Layer</p> 

In this study, multilayer feedforward networks have been chosen as the network architectures.

The process of ANN which is multilayer feedforward network is shown as follows:

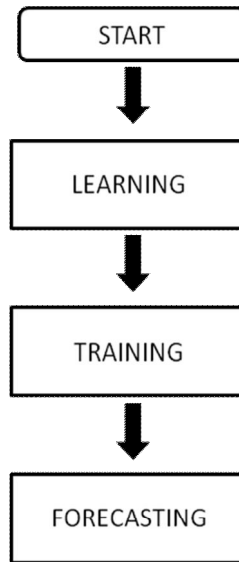


Figure 2: ANN Process

Generally, there are 2 types of ANN models that widely used in previous works which are supervised learning and unsupervised learning. The descriptions of the learning are as follows:

Table 3: Description of Training

Training Function Type	Descriptions
Supervised	<ul style="list-style-type: none">• Training and synaptic modification of a neural network that is provided with a number of training samples or task examples that construct the training set• To minimize the observed error between the desired output and the actual output• Learned to produce an output that closely matches to the desired.
Unsupervised	<ul style="list-style-type: none">• The network follows a self-supervised method and makes no use of external influences for synaptic weight modification.• An internal monitoring of the network's performance.• Looks for regularities and trends in the input signals and makes adaptations according to the function of the network [2].

The supervised learning network has been used instead of unsupervised learning network since under supervised learning, the actual numerical weights assigned to element inputs are determined by matching the historical data to desired outputs [3].

Furthermore, the learning algorithm of the network should be chosen. There are either Hebbian Learning, Widrow-Hoff Learning Rule, Gradient Descent Rule, Backpropagation Learning Algorithm or Kohonen's Learning Law. The descriptions of all the learning algorithms are as follows:

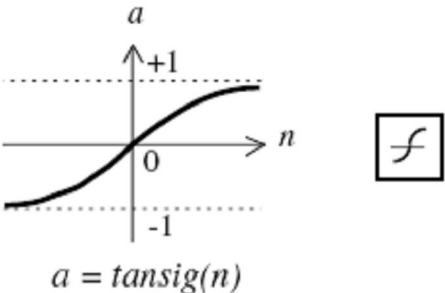
Table 4: Description of Networks' Learning Algorithm

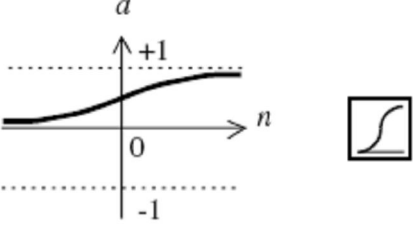
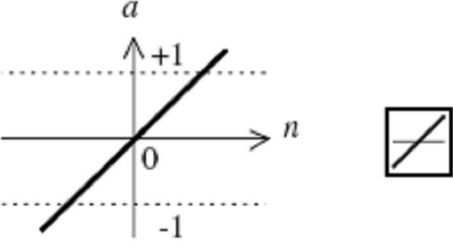
Learning Algorithms Techniques	Descriptions
Hebbian Learning	<ul style="list-style-type: none"> • Modify the synaptic efficiency according to the correlation of the presynaptic and postsynaptic activities as if a processing element receives an input from another processing element [2].
Widrow-Hoff Rule	<ul style="list-style-type: none"> • To reduce or minimize the difference between the desired output and the actual output, Least Mean Square [2].
Gradient Descent Rule	<ul style="list-style-type: none"> • To minimize the error between actual and desired output. • Adjusting the synaptic weights by an amount proportional to the first derivative of the mean squared error with respect to the synaptic weight [2].
Kohonen's Learning Law	<ul style="list-style-type: none"> • Applied only in unsupervised learning applications. • The processing elements compete for the opportunity of learning • The processing element with the largest output has the capability of inhibiting its competitors as well as exciting its neighbours [2].
Backpropagation Learning	<ul style="list-style-type: none"> • Error correction learning rule which is use the differences between the actual and the desired output [2].

Backpropagation Learning Algorithm has been chosen compared to others since it is based on error correction learning rule which is use the differences between the actual and the desired output. Backpropagation was created by generalizing the Widrow-Hoff learning rule to multiple-layer networks and nonlinear differentiable transfer functions. Input vectors and the corresponding target vectors are used to train a network until it can approximate a function, associate input vectors with specific output vectors, or classify input vectors in an appropriate way as defined to it. Properly trained backpropagation networks tend to give reasonable answers when presented with inputs that they have never seen. Typically, a new input leads to an output similar to the correct output for input vectors used in training that are similar to the new input being presented. This generalization property makes it possible to train a network on a representative set of input/target pairs and get good results without training the network on all possible input/output pairs [6].

As to create the network, the transfer functions of the network must be chosen. There are three types of transfer function which are tansig, logsig and purelin. The description of the transfer functions are as follows:

Table 5: Description of Transfer Functions

Learning Algorithms Techniques	Descriptions
Tansig	<ul style="list-style-type: none"> • generates outputs between -1 and +1 as the neuron's net input goes from negative to positive infinity [6] • Graph representation <div style="text-align: center;">  <p>$a = \text{tansig}(n)$</p> </div>

<p style="text-align: center;">Logsig</p>	<ul style="list-style-type: none"> • generates outputs between 0 and +1 as the neuron's net input goes from negative to positive infinity [6] • Graph representation <div style="text-align: center;">  <p>$a = \text{logsig}(n)$</p> </div>
<p style="text-align: center;">Purelin</p>	<ul style="list-style-type: none"> • the linear transfer function [6] • Graph representation <div style="text-align: center;">  <p>$a = \text{purelin}(n)$</p> </div>

For the model development, two transfer functions have been chosen for input and output of the network. For input, tansig transfer function has been chosen instead of other function since it can generate wider range of output. This is to ensure that the MATLAB can works well without any delay and etc [8]. As for output, purelin transfer function has been chosen due to the fact that it will make the network outputs can take on any value.

There are also training function included in the model development. There training function for backpropagation can be categorized into two categories which are slow and faster training. The descriptions are as follows:

Table 6: Description of Backpropagation Training Function

Training Function Type	Descriptions	
Slow	<ul style="list-style-type: none"> • Too slow for practical problems. • Eg: gradient descent(<code>traingd</code>), and gradient descent with momentum (<code>traingdm</code>) 	
Faster	<ul style="list-style-type: none"> • High performance algorithms that can converge from ten to one hundred time faster than the slow type. • Two types: heuristic and standard numerical optimization techniques algorithm 	
	Heuristic	Standard numerical optimization techniques algorithm
	<ul style="list-style-type: none"> • developed from an analysis of the performance of the standard steepest descent algorithm • Eg: variable learning rate backpropagation, <code>traingda</code>; and resilient backpropagation <code>trainrp</code>[6] 	<ul style="list-style-type: none"> • Eg: conjugate gradient (<code>traincgf</code>, <code>traincgp</code>, <code>traincgb</code>, <code>traincsg</code>), quasi-Newton (<code>trainbfg</code>, <code>trainoss</code>), and Levenberg-Marquardt (<code>trainlm</code>)[6]

Table 6 shows the description of backpropagation training algorithm types and the example of training function that can be used for model development. For the purpose of the model development, Levenberg-Marquardt (`trainlm`) has been chosen as the training function.

The Levenberg-Marquardt (`trainlm`) has been chosen instead of other standard numerical optimization techniques algorithm of faster training algorithm based on the following reasons:

Table 7: Comparison between TRAINLM and other techniques

Training Algorithms Techniques	Descriptions
Conjugate gradient (traincgf, traincgp, traincgb, trainscg),	<ul style="list-style-type: none"> • faster convergence than steepest descent directions • The results will vary from one output to another.
Quasi-Newton (trainbfg, trainoss)	<ul style="list-style-type: none"> • Converges faster than conjugate gradient methods. • Complex and expensive to compute the Hessian matrix for feedforward neural networks
Levenberg-Marquardt (trainlm)	<ul style="list-style-type: none"> • The fastest method for training moderate-sized feedforward neural networks (up to several hundred weights). • Very efficient since the solution of the matrix equation is a built-in function, so its attributes become even more pronounced in a MATLAB setting [6].

Table 7 shows the description of Standard numerical optimization techniques algorithm and the reasons to choose Levenberg-Marquardt (trainlm) as the training algorithm.

2.2 Data Collected from GDC

Upon developing the model, the data had been gathered from GDC (UTP). The data of the electricity demand by UTP had been gathered starting from 1st January 2006 till 31st December 2009. The data gathered are based on daily interval data. The data have been categorized into two conditions based on UTP's Academic Calendar. The data have been categorized into two conditions based on UTP's Academic Calendar. The simplified Academic Calendar's of UTP ranging from 2006 until 2009 is attached in Appendix A. The two conditions are Semester ON and Semester OFF. The graph representations of daily data gathered from GDC are as follow:

2.3.1 Semester ON

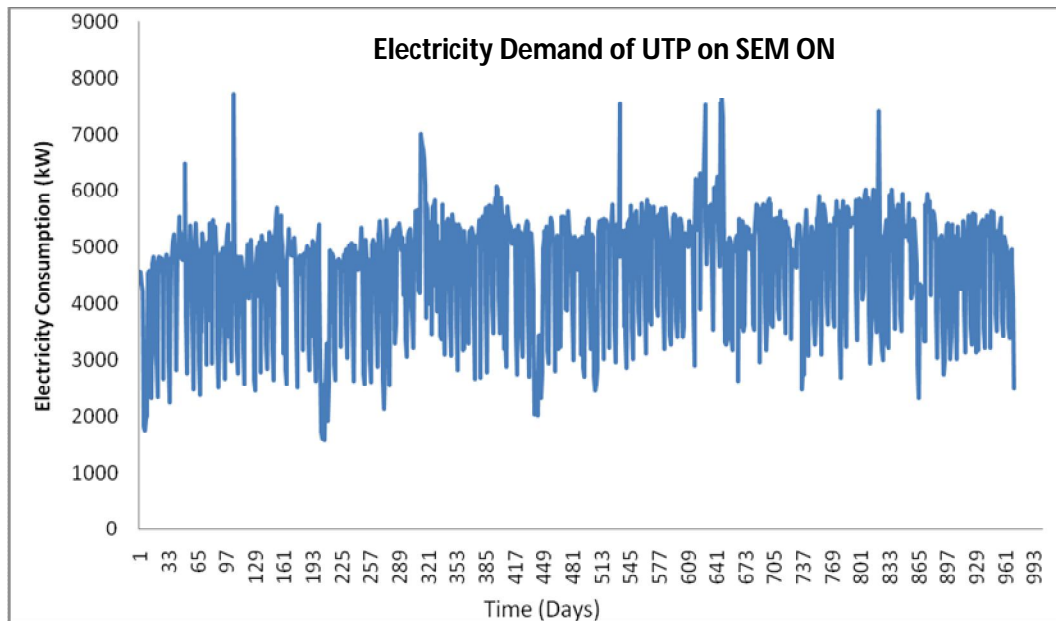


Figure 3: Graph of Electricity Demand of UTP on Semester ON

Figure 3 shows the electricity consumption of UTP during Semester ON. The average of electricity demand of UTP is 5 MW. There are also certain times that the electricity demand becomes higher. This is due to the special occasions that have been organized in UTP.

2.3.2 Semester OFF

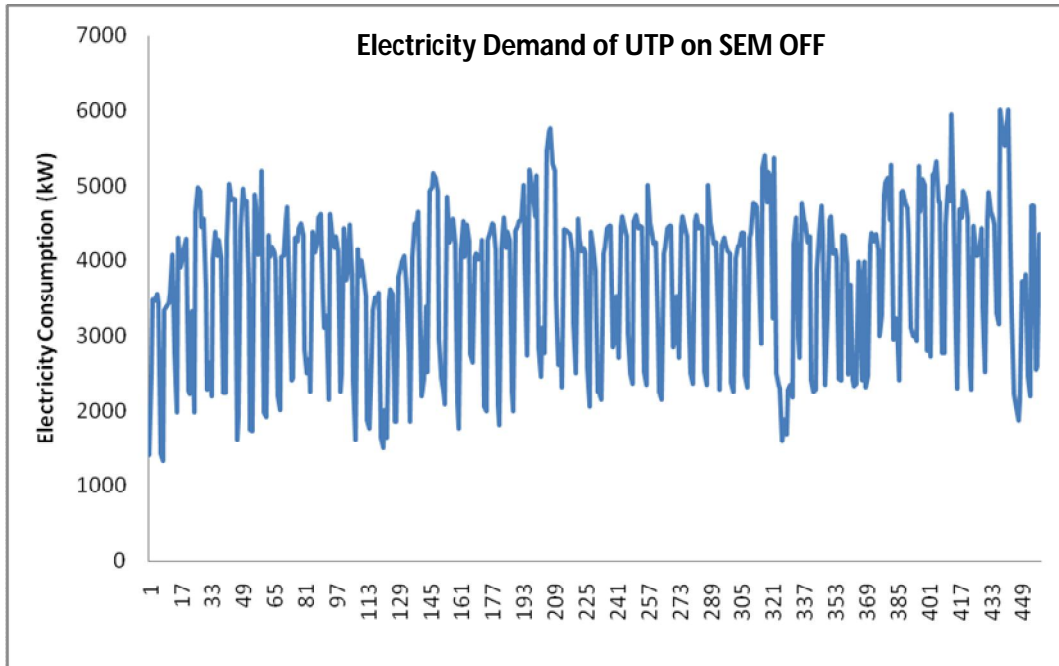


Figure 4: Graph of Electricity Demand of UTP on Semester OFF

Figure 4 shows the electricity demand of UTP during Semester OFF. The average of electricity consumption of UTP is 4 MW. It can be seen that at the beginning the data period, the electricity consumption of UTP is higher. This may be due to the fact that there are special event organized at that time that require more electricity demand.

2.3 Data Analysis

The daily data that has been gathered from GDC (UTP) are being analysed and the result are as follows:

2.4.1 Semester ON

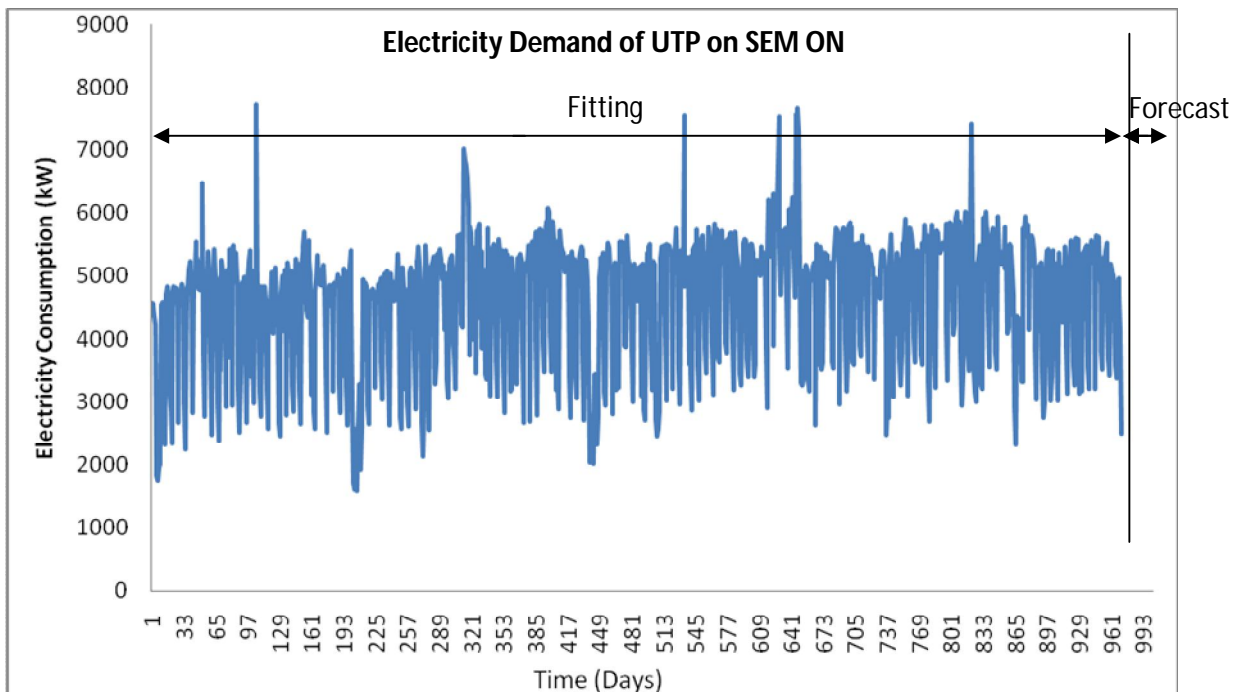


Figure 5: Graph Electricity Demand of UTP on Semester ON with Fitting Data and Forecast Data

Figure 5 shows the electricity demand of UTP during Semester ON together with fitting data and forecast data. The fitting data is the data that is being used to train, validate and test the models.

2.4.2 Semester OFF

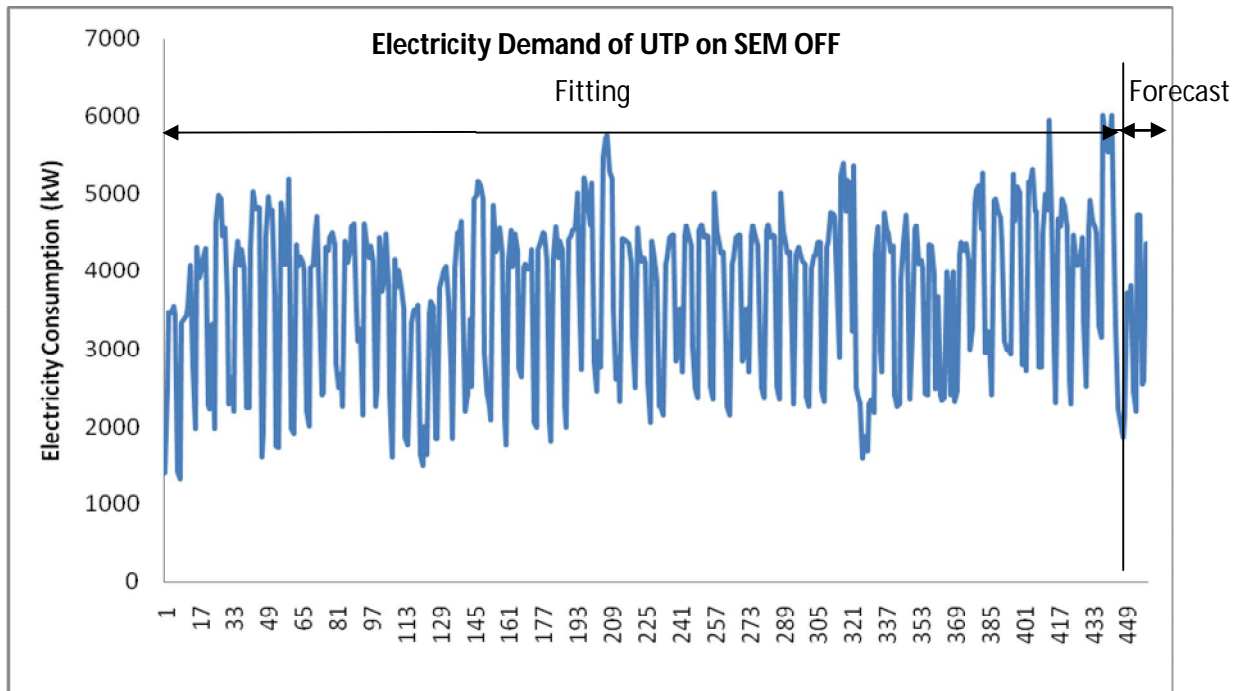


Figure 6: Graph Electricity Demand of UTP on Semester OFF with Fitting Data and Forecast Data

Figure 6 shows the electricity demand of UTP during Semester ON together with fitting data and forecast data. The fitting data is the data that is being used to train, validate and test the models.

CHAPTER 3

METHODOLOGY

3.1 Procedure Identification

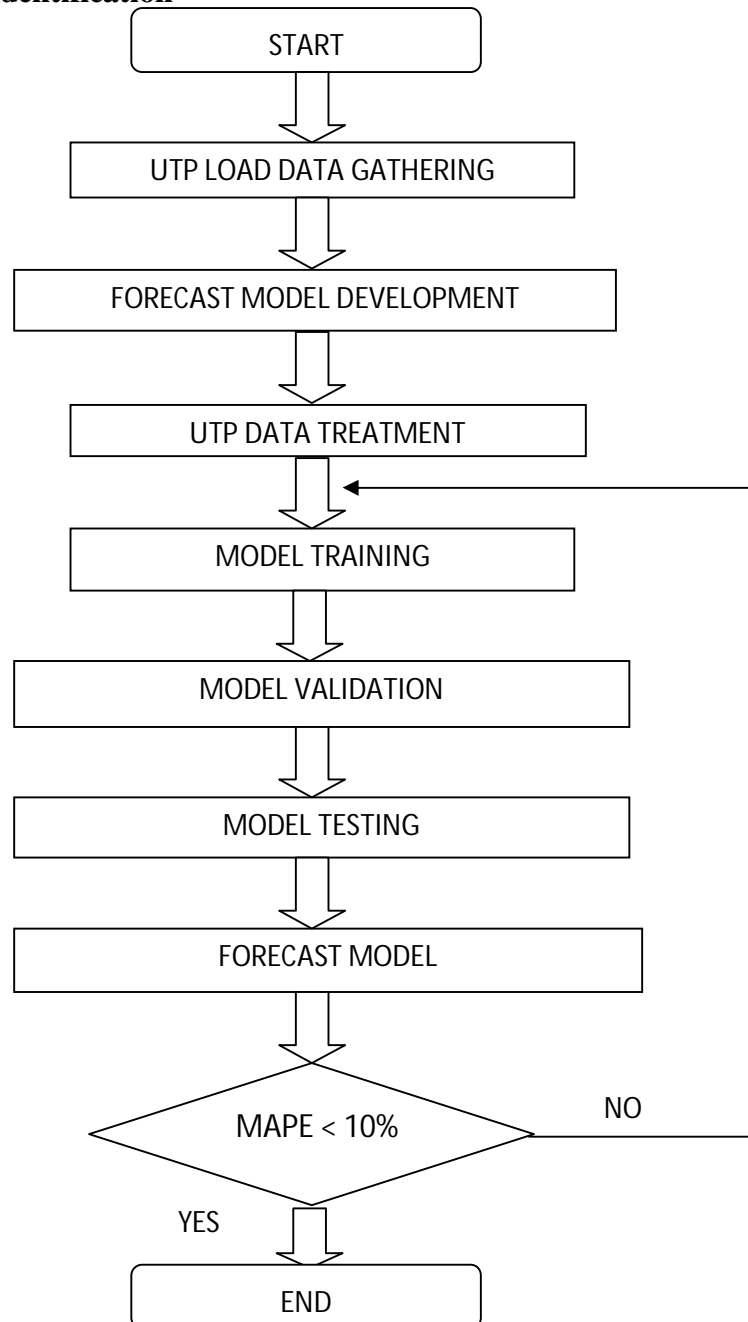


Figure 7: Project's Methodology

3.1.1 UTP load data gathering

The historical UTP daily load data have been gathered from GDC (UTP). The historical data is ranging from 1st January 2006 until 31st December 2009. The data have been categorized into Semester ON and Semester OFF based on UTP academic calendar.

3.1.2 Forecast model development

The forecast models have been developed using the data gathered. There are four (4) models that have been developed. The models are based on Semester ON and Semester OFF of UTP and the duration of the load forecasting. The summary of the four (4) models are as follows:

Table 8: Forecast Models

Model Type	Semester Type	Forecasting Duration
Model 1	Sem OFF	7 days
Model 2	Sem ON	7 days
Model 3	Sem OFF	30 days
Model 4	Sem ON	30 days

The input and output of the model are depended on the forecasting type. The summary of input and output of each of the model are as follows:

Table 9: Models' input and output

Model Type	Input(Ain)	Output (Aout)
Model 1	14	7
Model 2	14	7
Model 3	60	30
Model 4	60	30

The transfer functions used for the model development can be categorized into three (3) which are transfer function for input, output and training. The transfer functions used for each of the models are the same and they are as follows:

Table 10: Transfer Functions of the models

Parameter	Transfer functions
Input	Tansig
Output	Purelin
Training	Trainlm

The models have been developed using MATLAB Version 7.1. The coding for the four models has been included in the appendix. See Appendix B, C, D and E for more details of the coding.

3.1.3 UTP data treatment

The data treatment has been done in order to create robust models. The data treatment consists of data normalization and data partitioning.

a) Data normalization

The data used for the model development should be in the range of -1 to +1 since the transfer function used for input of the model is *tansig*. This is to ensure that the MATLAB can works well without any delay and etc [8]. Hence the daily data has been normalized by dividing them with 10000. As the forecasted load has been obtained, the value then should be converted back by multiplying with 10000.

b) Data partitioning

The gathered data has been partitioned into three (3) partitions for the purpose of training, validation and testing. The partitioning has been done based on the total data as follows:

Table 11: Total Data

Semester Type	Total Data
Semester OFF	482
Semester ON	965

The partitions are as follows:

- i) Training data – 40%
- ii) Validation data – 30%
- iii) Testing data – 30%

The partitioning of the data is based on the non-randomization data that need to done to the historical data in order to obtain accurate result [8].

3.1.4 Model training

The training of the forecast model involves the 40% of the gathered data. The numbers of data for each of the model are as follows:

Table 12: Numbers of Training data

Model Type	Training data
Model 1	193
Model 2	386
Model 3	184
Model 4	377

The transfer function in the model development for the purpose of training is TRAINLM. The Levenberg-Marquardt (trainlm) was designed to approach second-order training speed without having to compute the Hessian matrix. This algorithm appears to be the fastest method for training moderate-sized feedforward neural networks (up to several hundred weights) [6]. The training occurs based on the training parameter. The descriptions of the training parameter are as follows:

Table 13: Training parameter

Training parameter	Values	Description
net.trainParam.epochs	100	Maximum number of epoch that can be used to train.
net.trainParam.goal	0.001	The performance goal that training and validation should meet to stop training
net.trainparam.show	1	Number of epoch between each displays

3.1.5 Model validation

The purpose of validation is to guide the training of the model. As the validation reaches the performance goal, the training should stop. The validation data consists of 30% of the entire data. The numbers of data for validation are as follows:

Table 14: Numbers of validation data

Model Type	Training data
Model 1	145
Model 2	290
Model 3	138
Model 4	283

3.1.6 Model testing

The purpose of testing is to observe the efficiency of the developed models. The lower MAPE indicates the higher efficiency. The numbers of data for testing are as follows:

Table 15: Numbers of testing data

Model Type	Training data
Model 1	144
Model 2	289
Model 3	137
Model 4	282

3.1.7 Forecast model

The forecast model is used to forecast for seven (7) days ahead as well as for thirty (30) days ahead. The output data or the actual data for the models are as follows:

Table 16: Actual Data Range

Model Type	Actual Data
Model 1	25 th Dec – 31 st Dec '09
Model 2	21 st Nov – 27 th Nov '09
Model 3	2 nd Dec -31 st Dec '09
Model 4	29 th Oct – 27 th Nov '09

Once the forecasted load has been obtained, the program then will compares the values with the actual load. Hence, the Mean Absolute Percentage Error (MAPE) can be calculated.

3.1.8 MAPE

MAPE or the error calculation between actual and forecast load has been included in the model development. The MAPE values will indicate which model would be the best model to be used for load forecasting. The error's calculations are based on the following formula:

$$\text{RelativeError} = \frac{\text{ForecastLoad} - \text{ActualLoad}}{\text{ActualLoad}} \times 100\%$$

$$\text{AbsoluteError} = \frac{|\text{ForecastLoad} - \text{ActualLoad}|}{\text{ActualLoad}} \times 100\%$$

3.2 Project Duration

As to ease the progress of this project, a planner had been conducted known as Project Duration to monitor the progress of this study. See Appendix F for more details of the project duration of this project. The planner is consisting of the planning for one year duration of this project.

CHAPTER 4

RESULT AND DISCUSSION

4.1 Result

The four (4) developed models have been simulated using five different numbers of hidden neurons. The numbers of hidden neurons used for each of the models are as follows:

Table 17: Number of Hidden Neurons Used

Number of Hidden Layer	Number of Hidden Neurons
1	3
1	5
1	7
1	9
1	11

Table 17 shows the number of hidden neurons used for the simulation. For the purpose of accurate result, the simulations have been carried out up to twenty (20) simulations for each of the hidden neurons. Then, the averages of the results have been obtained. The results are as follows:

4.1.1 Model 1

4.1.1.1 3 Neurons

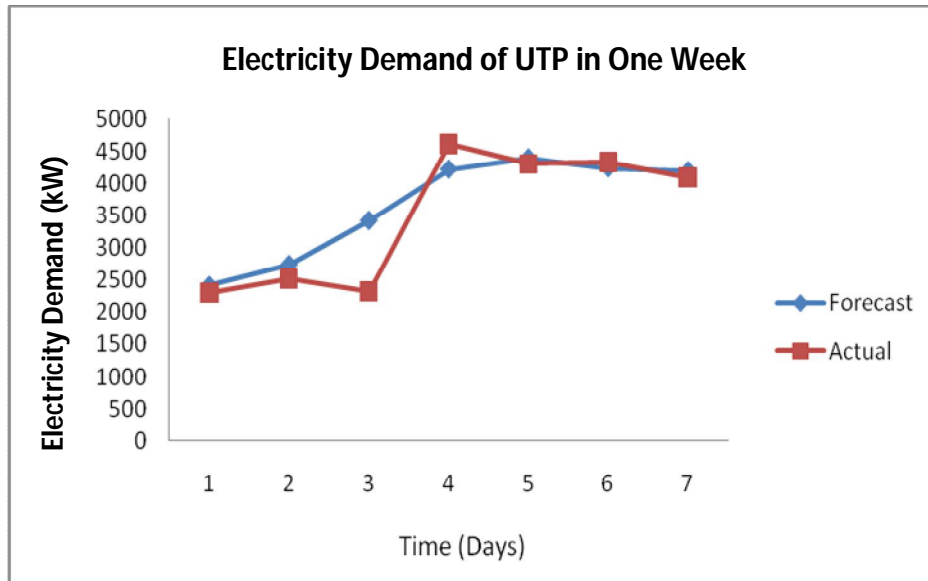


Figure 7: Comparison between Actual and Forecast Load for 3 Hidden Neurons

4.1.1.2 5 Neurons

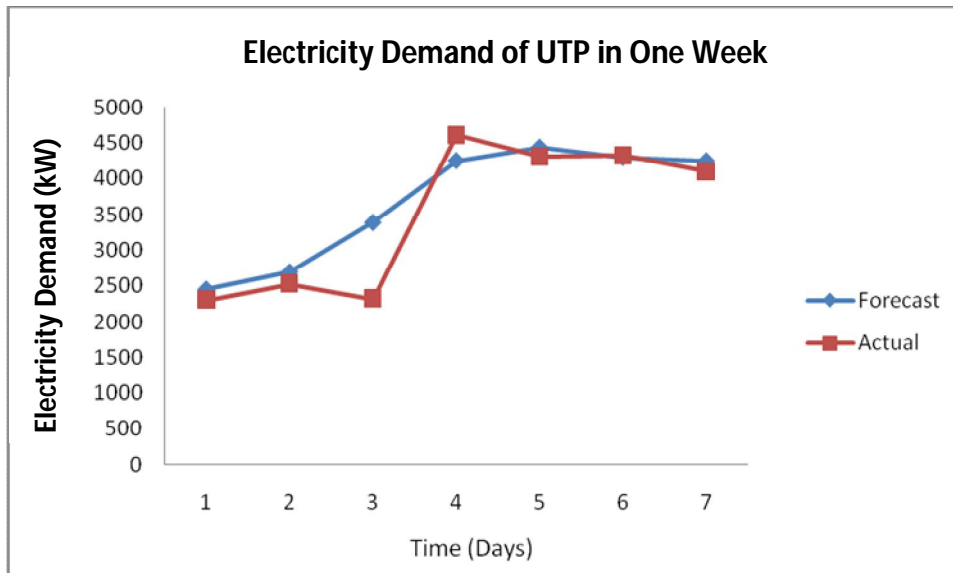


Figure 8: Comparison between Actual and Forecast Load for 5 Hidden Neurons

4.1.1.3 7 Neurons

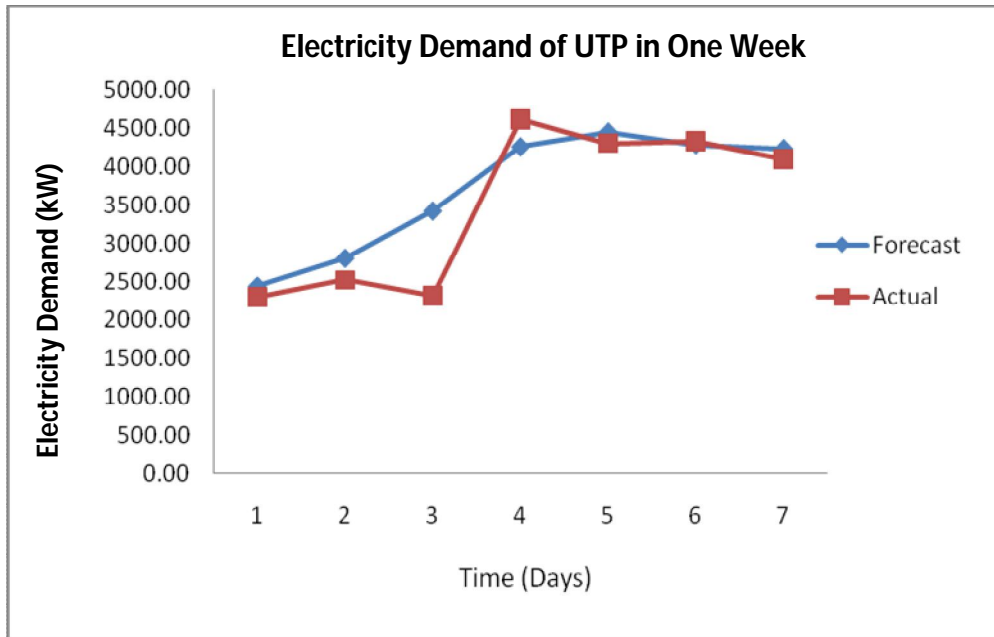


Figure 9: Comparison between Actual and Forecast Load for 7 Hidden Neurons

4.1.1.4 9 Neurons

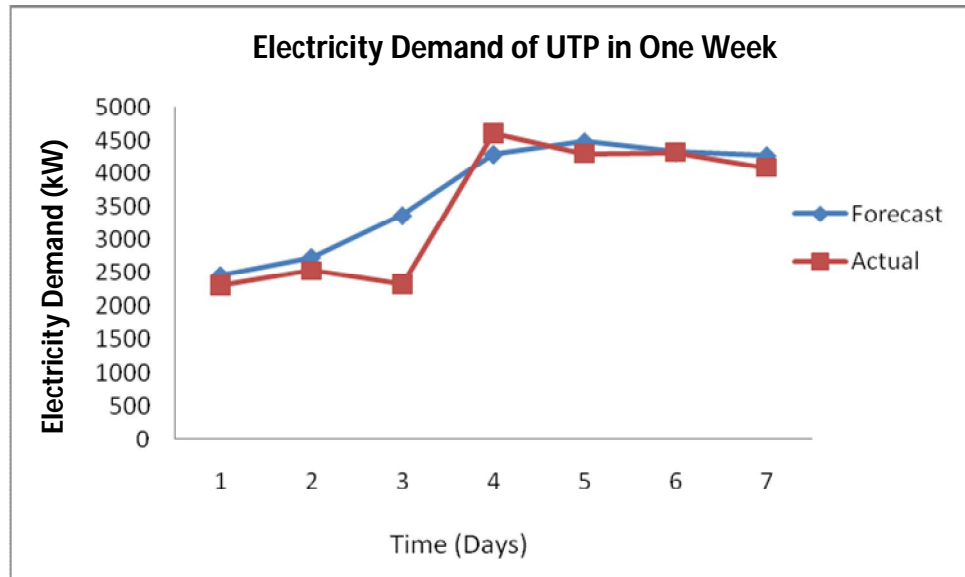


Figure 10: Comparison between Actual and Forecast Load for 9 Hidden Neurons

4.1.1.5 Model 5: 11 Neurons

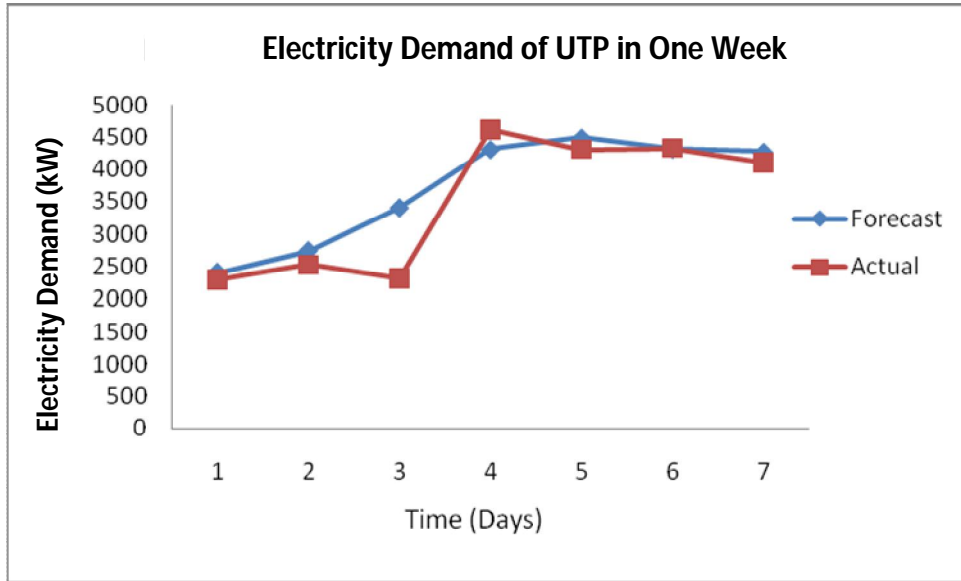


Figure 11: Comparison between Actual and Forecast Load for 11 Hidden Neurons

4.1.1.6 MAPE Values

Table 18: MAPE values for Model 1

(Sem OFF 7 Days)

No. Hidden Neurons	MAPE (%)
3	8.9348
5	9.4634
7	8.9196
9	9.5734
11	9.1301

Table 18 shows the values of MAPE obtained as Model 1 is simulated using five different numbers of hidden neurons for twenty simulation. Based on the result obtained, it is found that Model 1 with the number of hidden neurons seven (7) has the less value of MAPE which is 8.9196%. The training, validation, testing and comparison of Model 1 with 7 hidden neurons are as follows:

Training and Validation

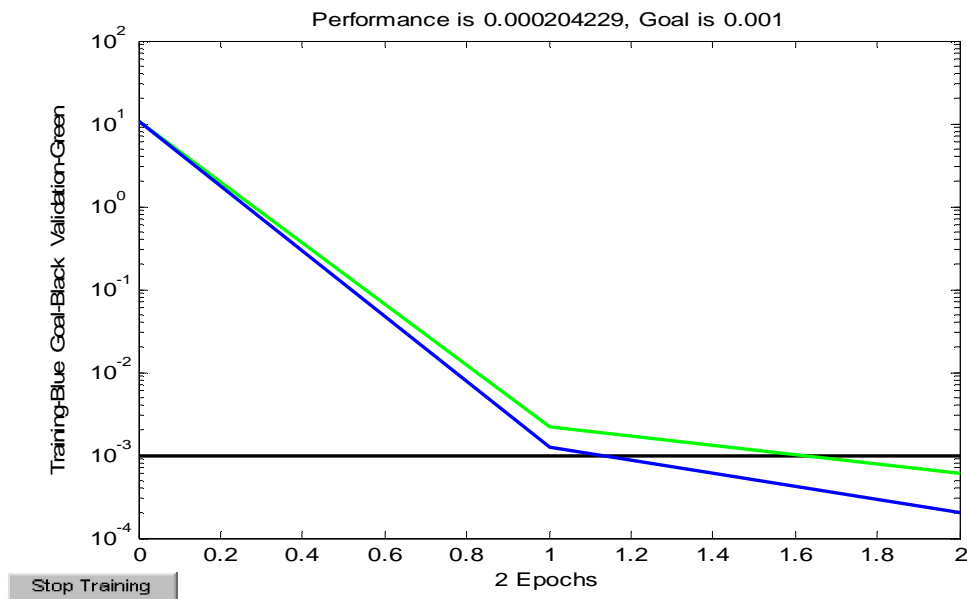


Figure 12: Training and Validation of Model 1 with 3 neurons

Test 1

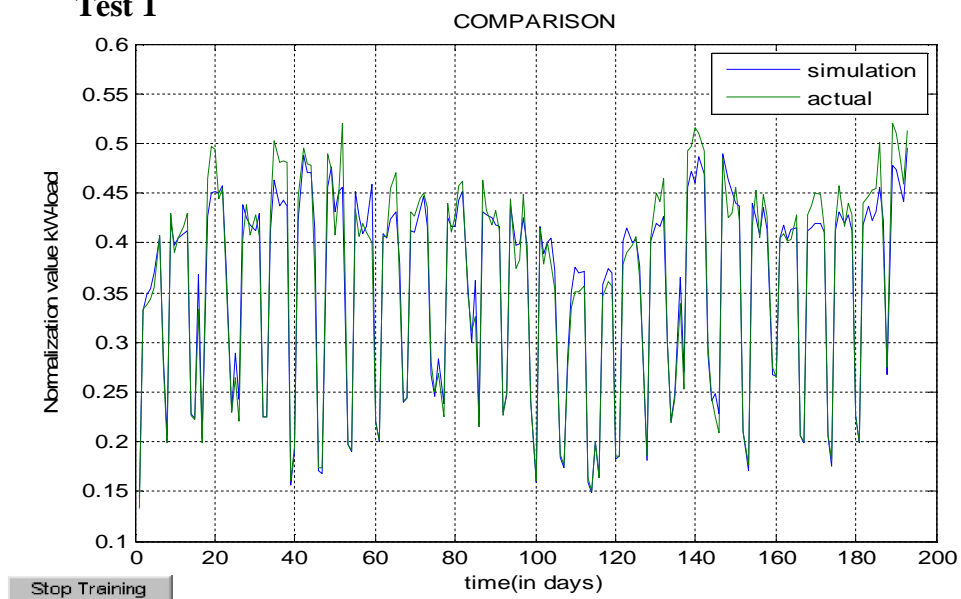


Figure 13: Test 1 Result using training data

Test 2

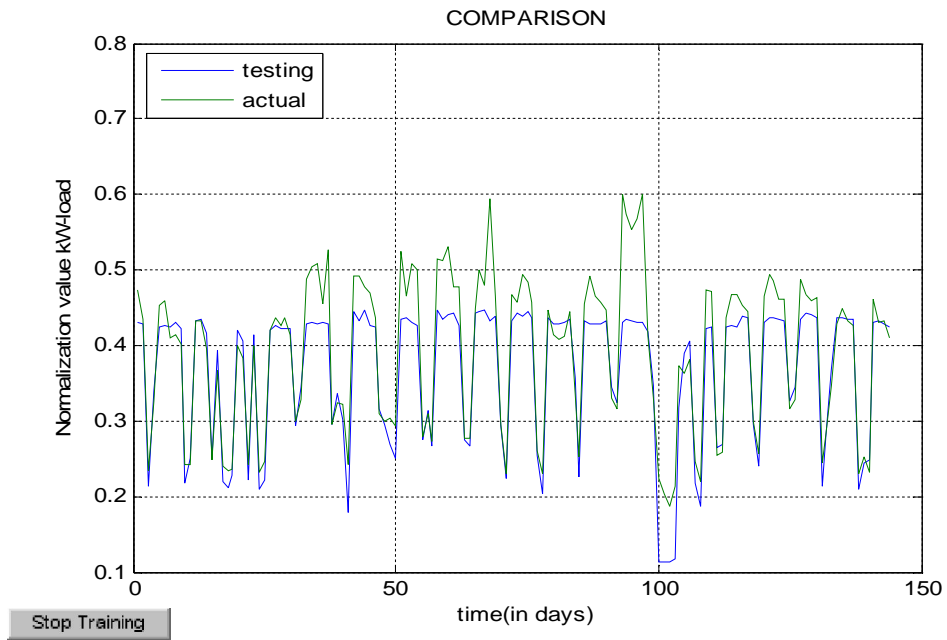


Figure 14: Test 2 result using testing data

Comparison

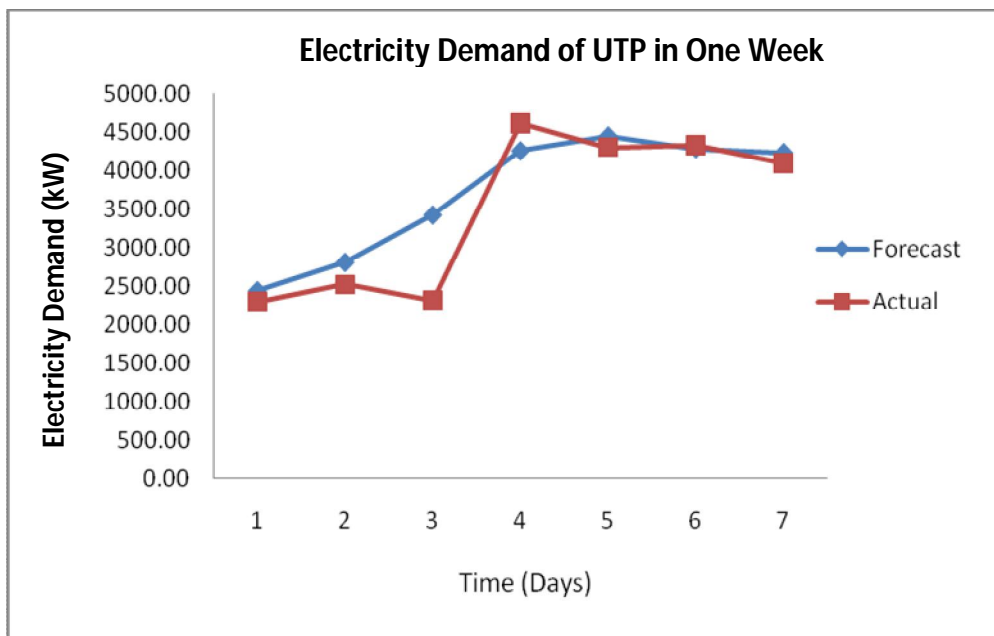


Figure 15: Comparison between Actual and Forecast Load for 7 Hidden Neurons

4.1.2 Model 2

4.1.2.1 3 Neurons

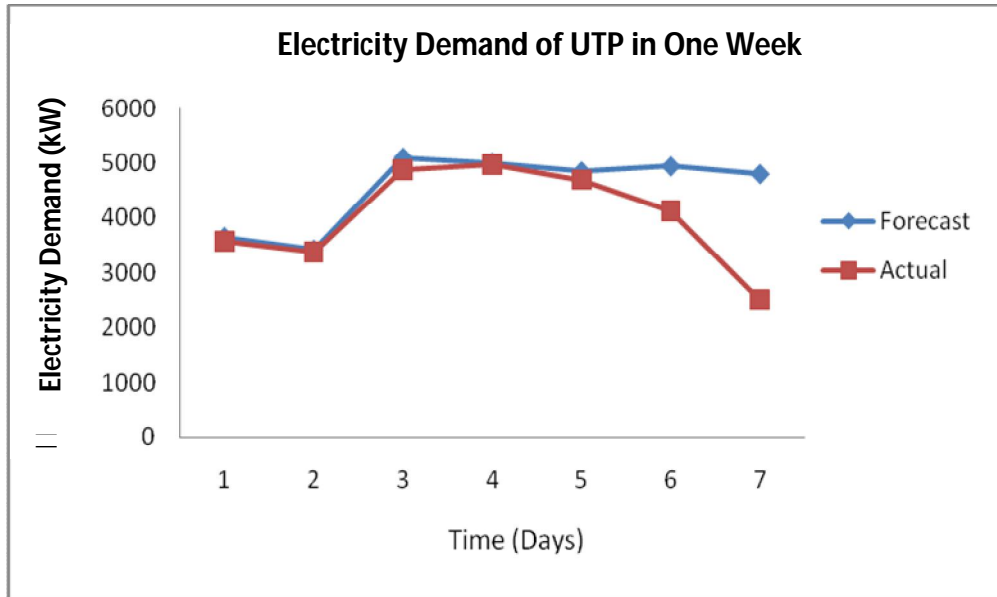


Figure 16: Comparison between Actual and Forecast Load for 3 Hidden Neurons

4.1.2.2 5 Neurons

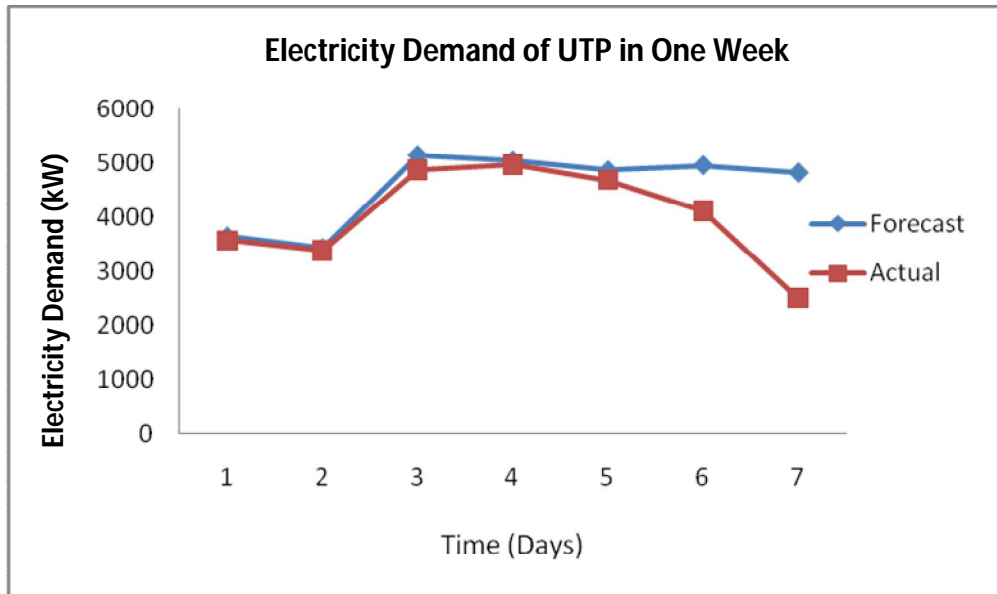


Figure 17: Comparison between Actual and Forecast Load for 5 Hidden Neurons

4.1.2.3 7 Neurons

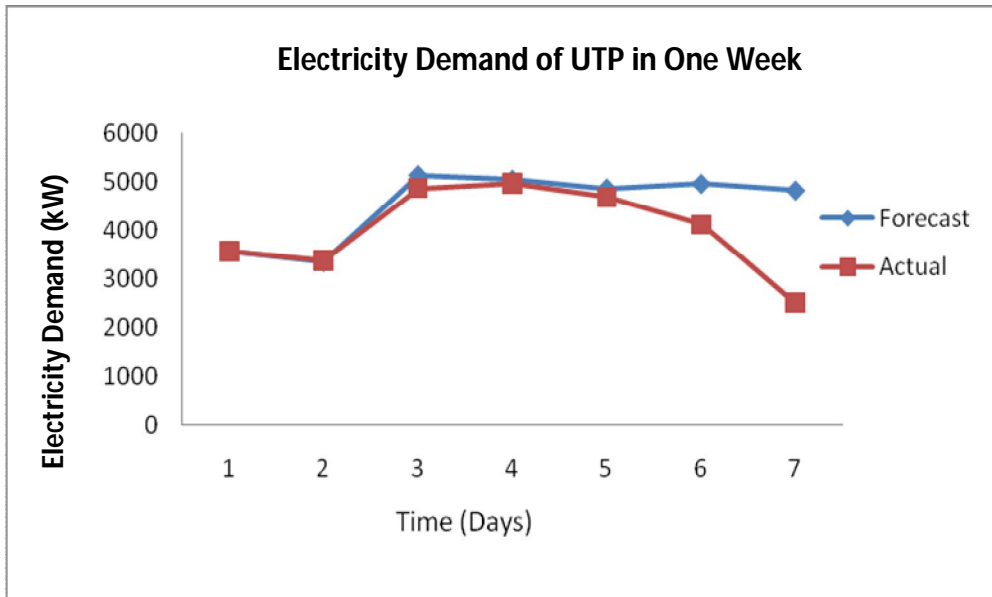


Figure 18: Comparison between Actual and Forecast Load for 7 Hidden Neurons

4.1.2.4 9 Neurons

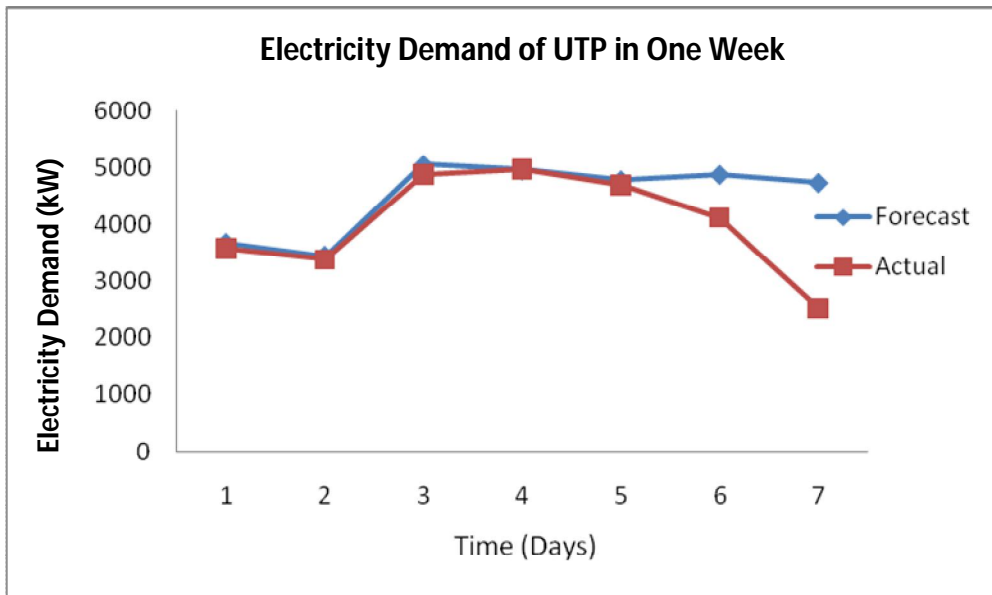


Figure 19: Comparison between Actual and Forecast Load for 9 Hidden Neurons

4.1.2.5 11 Neurons

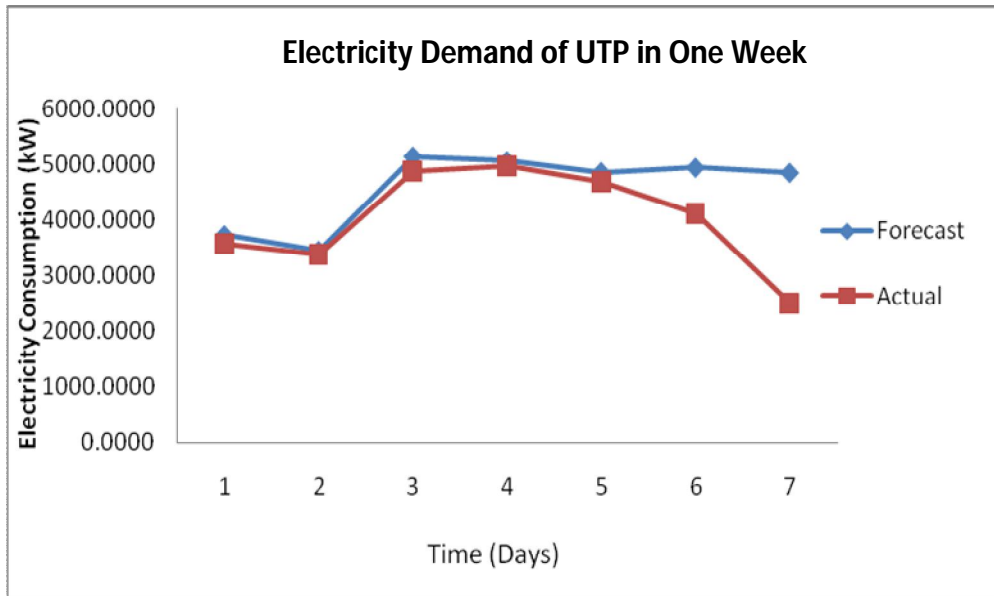


Figure 20: Comparison between Actual and Forecast Load for 11 Hidden Neurons

4.1.2.6 MAPE Values

Table 19: MAPE values for Model 2

(Sem ON 7 Days)

No. Hidden Neurons	MAPE (%)
3	13.6948
5	14.3462
7	14.3444
9	13.6007
11	14.4097

Table 19 shows the values of MAPE obtained as Model 2 is simulated using five different numbers of hidden neurons for twenty simulation. Based on the result obtained, it is found that Model 2 with the number of hidden neurons nine (9) has the less value of MAPE which is 13.6007%. The training, validation and testing of Model 2 with 9 hidden neurons are as follows:

Training and Validation

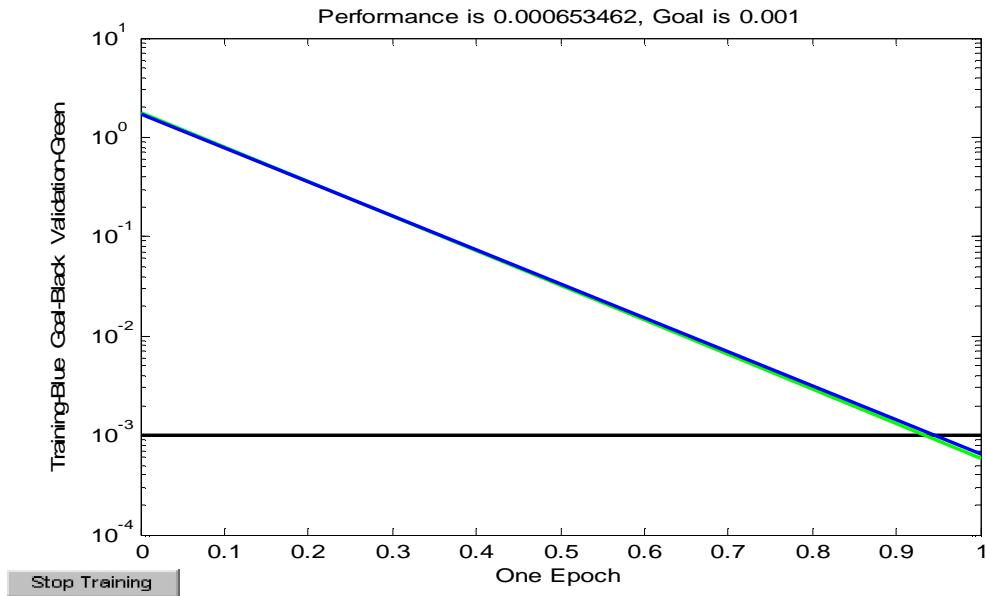


Figure 21: Training and Validation of Model 2 with 9 Neurons

Test 1

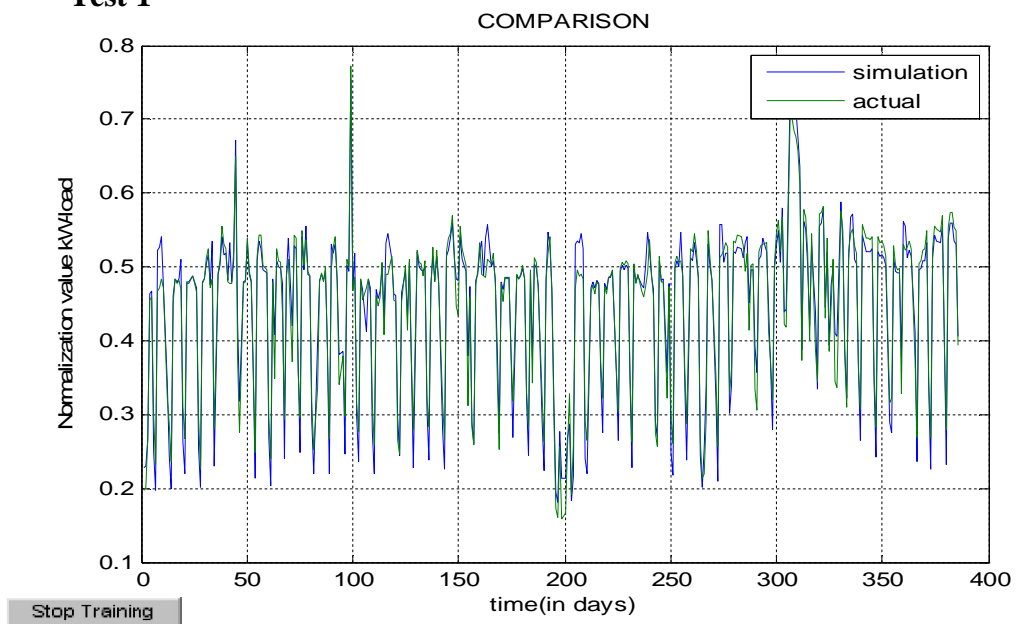


Figure 22: Test 1 result using training data

Test 2

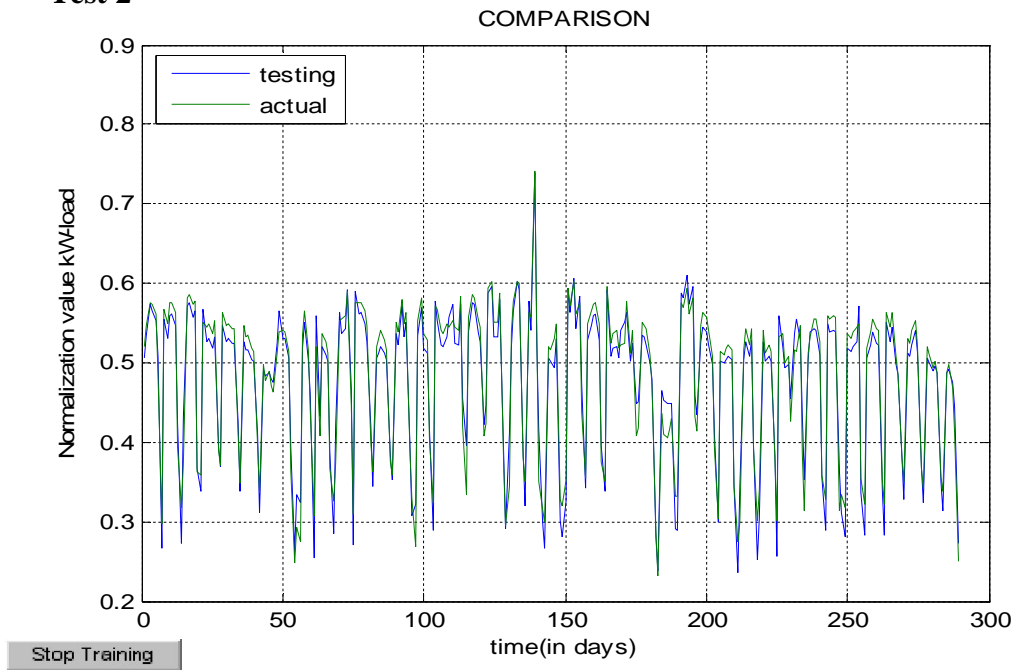


Figure 23: Test 2 result using testing data

Comparison

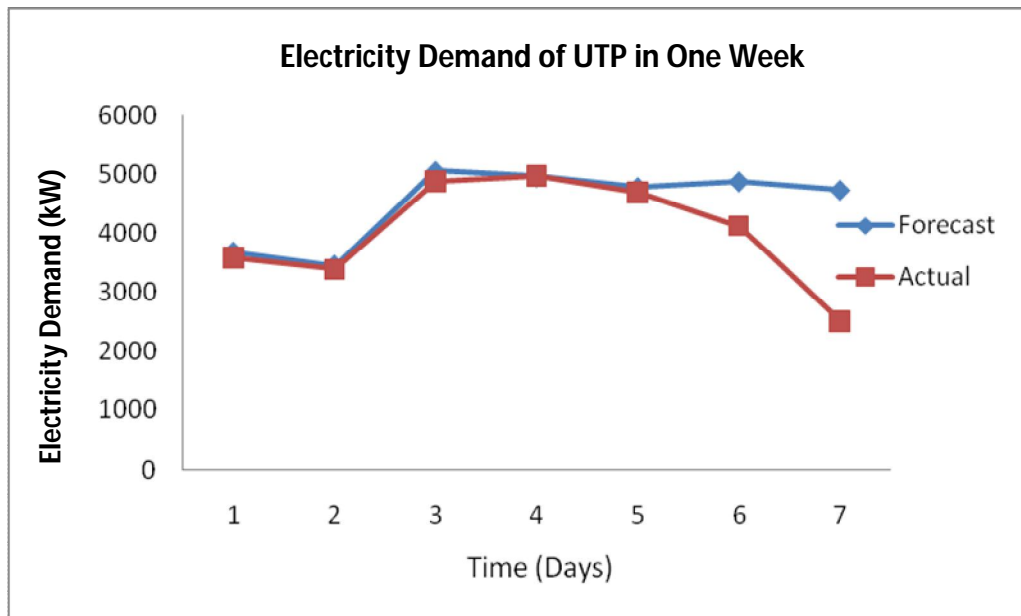


Figure 24: Comparison between Actual and Forecast Load for 9 Hidden Neurons

4.1.3 Model 3

4.1.3.1 3 Neurons

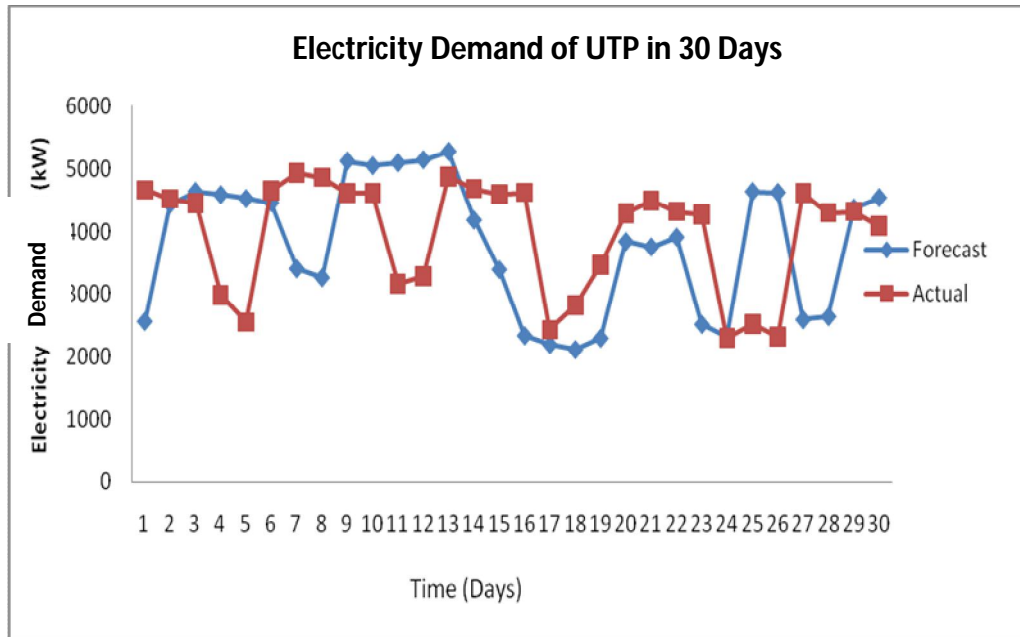


Figure 25: Comparison between Actual and Forecast Load for 3 Hidden Neurons

4.1.3.2 5 Neurons

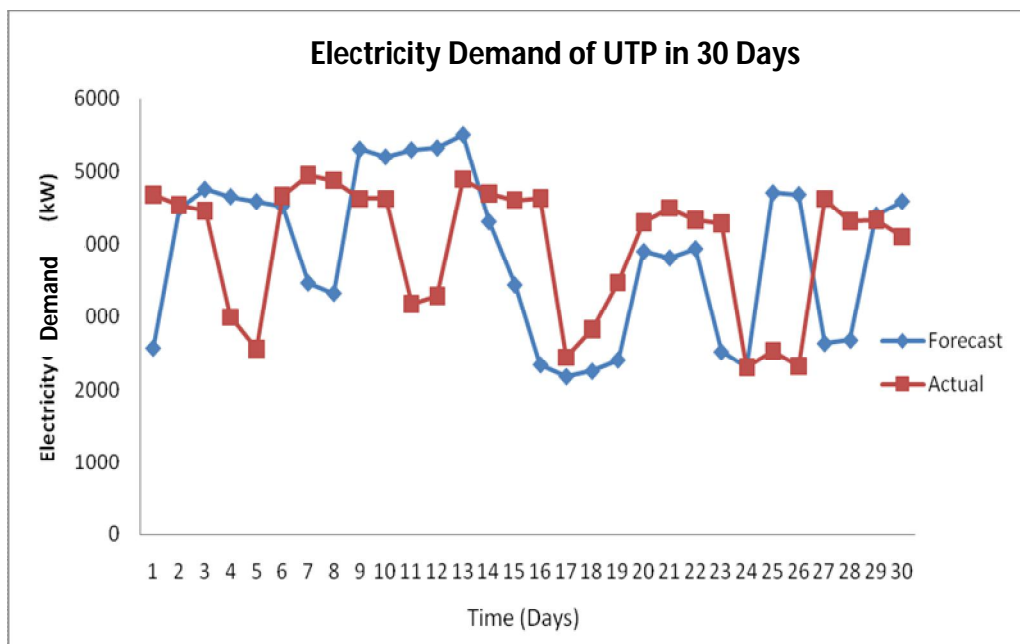


Figure 26: Comparison between Actual and Forecast Load for 5 Hidden Neurons

4.1.3.3 7 Neurons

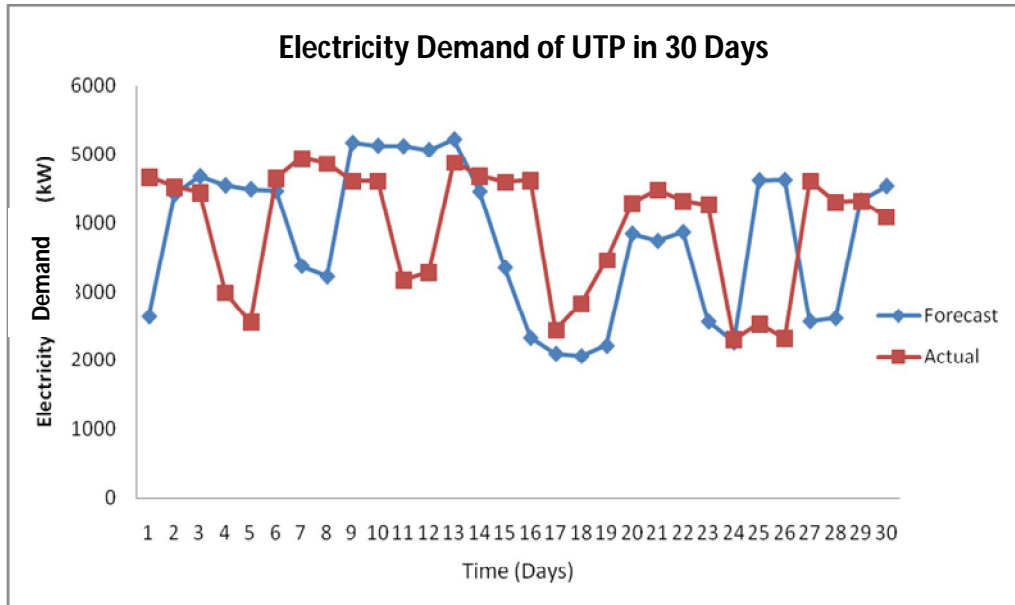


Figure 27: Comparison between Actual and Forecast Load for 7 Hidden Neurons

4.1.3.4 9 Neurons

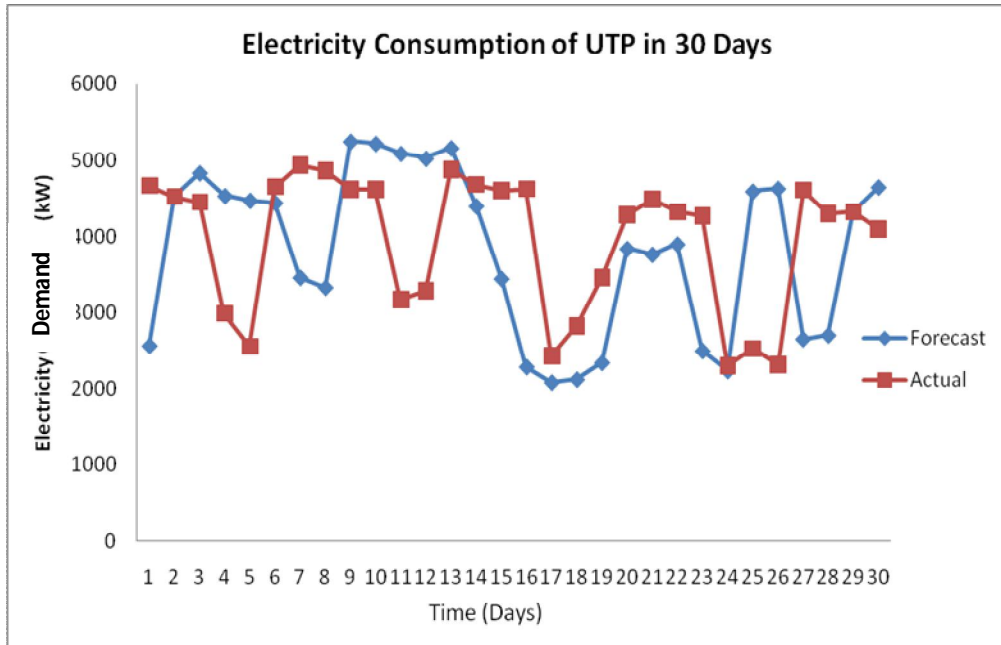


Figure 28: Comparison between Actual and Forecast Load for 9 Hidden Neurons

4.1.3.5 11 Neurons

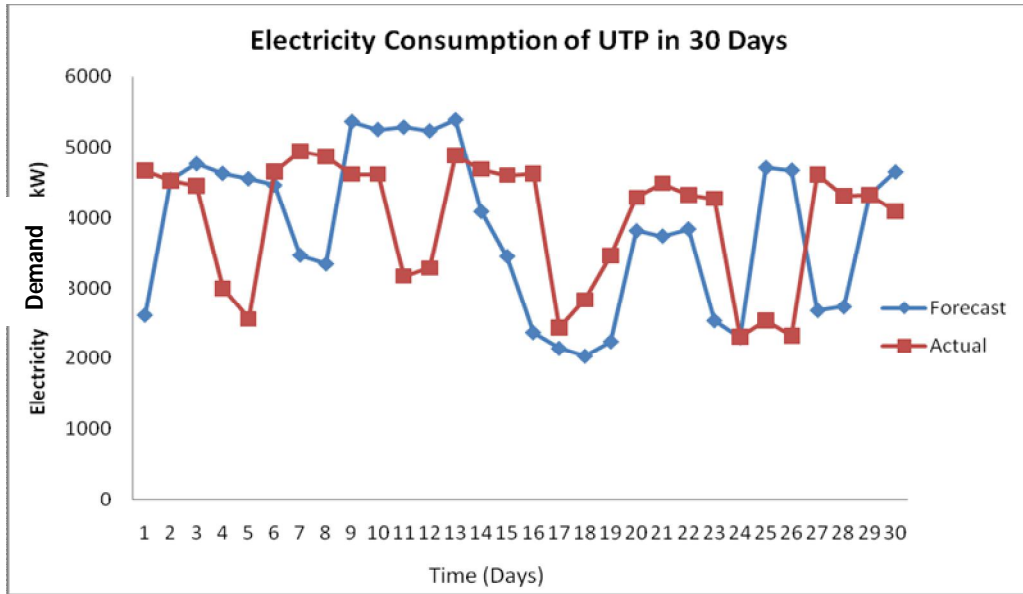


Figure 29: Comparison between Actual and Forecast Load for 11 Hidden Neurons

4.1.3.6 MAPE Values

Table 20: MAPE values for Model 3

(Sem OFF 30 Days)

No. Hidden Neurons	MAPE (%)
3	27.7642
5	28.1650
7	28.4150
9	28.3395
11	28.8203

Table 20 shows the values of MAPE obtained as Model 3 is simulated using five different numbers of hidden neurons for twenty simulation. Based on the result obtained, it is found that Model 2 with the number of hidden neurons three (3) has the less value of MAPE which is 27.7642%. The training, validation and testing of Model 3 with 3 hidden neurons are as follows:

Training and Validation

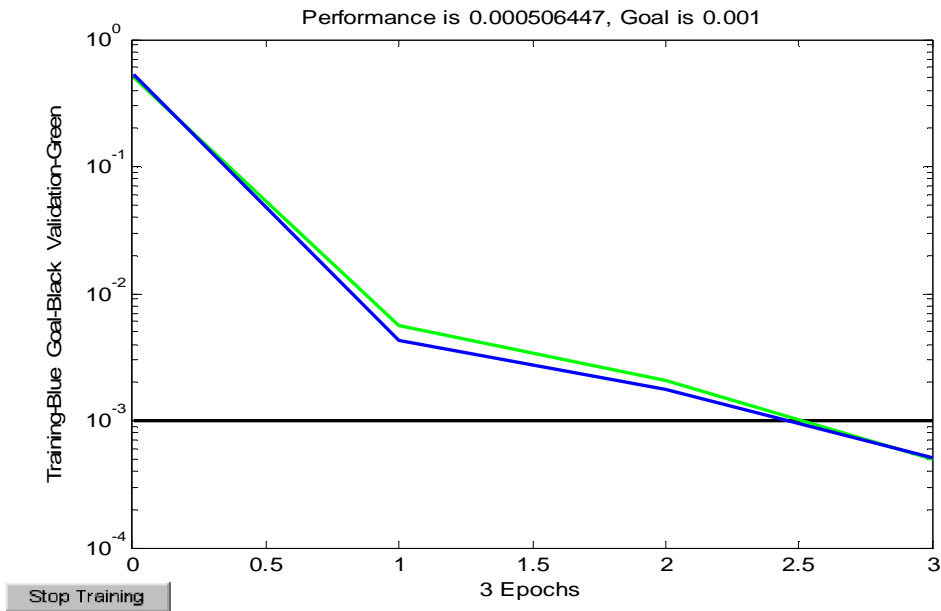


Figure 30: Training and Validation of Model 3 with 3 neurons

Test 1

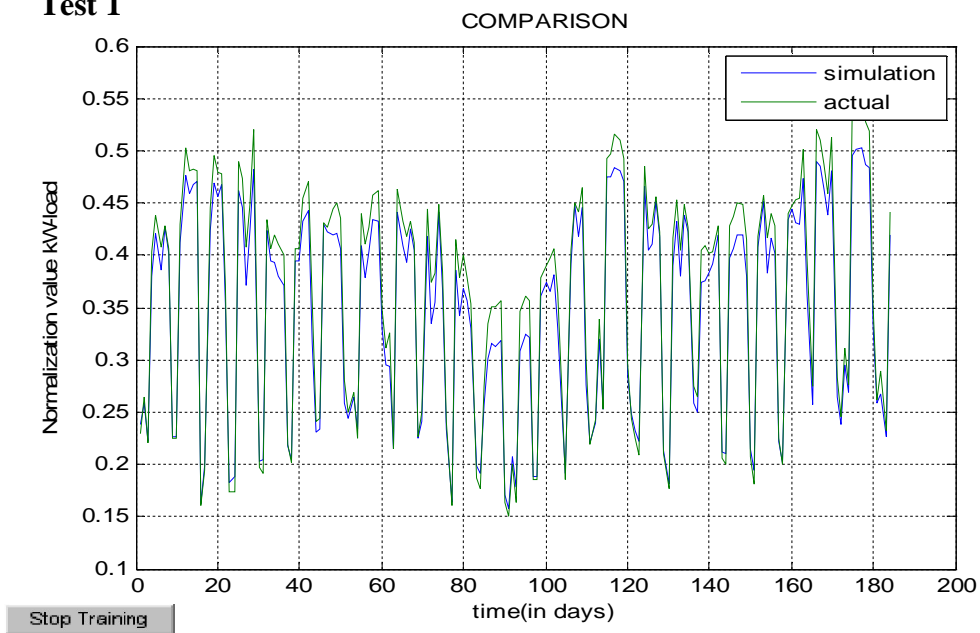


Figure 31: Test 1 result using training data

Test 2

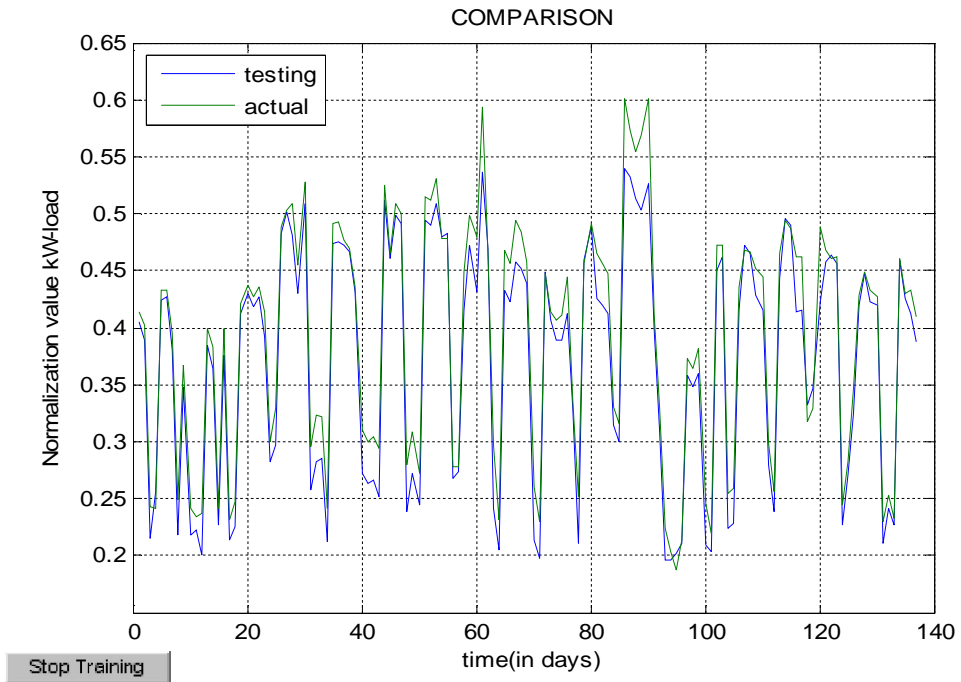


Figure 32: Test 2 result using testing data

Comparison

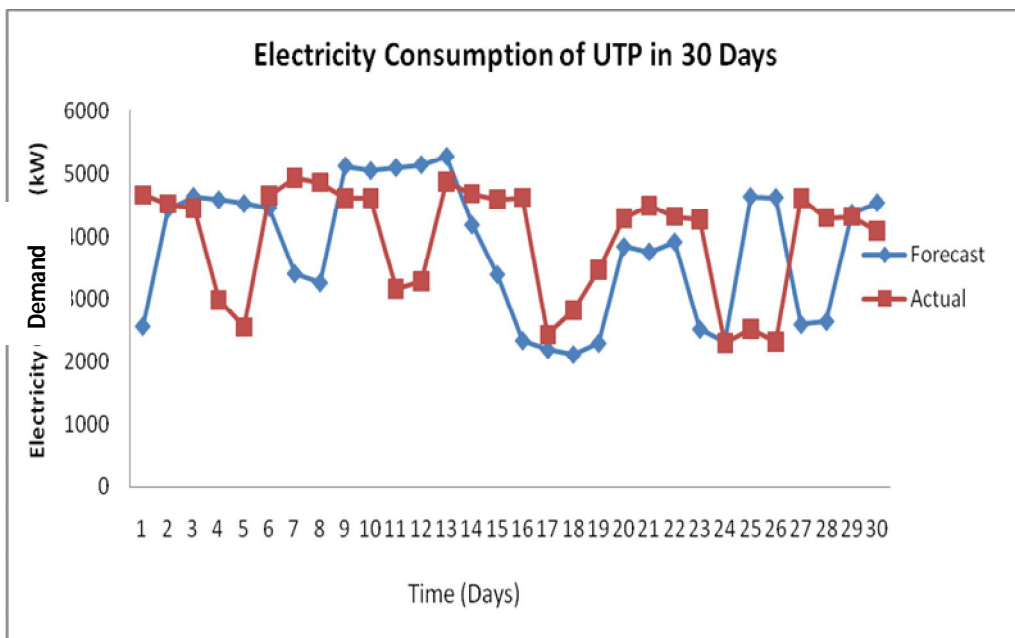


Figure 33: Comparison between Actual and Forecast Load for 3 Hidden Neurons

4.1.4 Model 4

4.1.4.1 3 Neurons

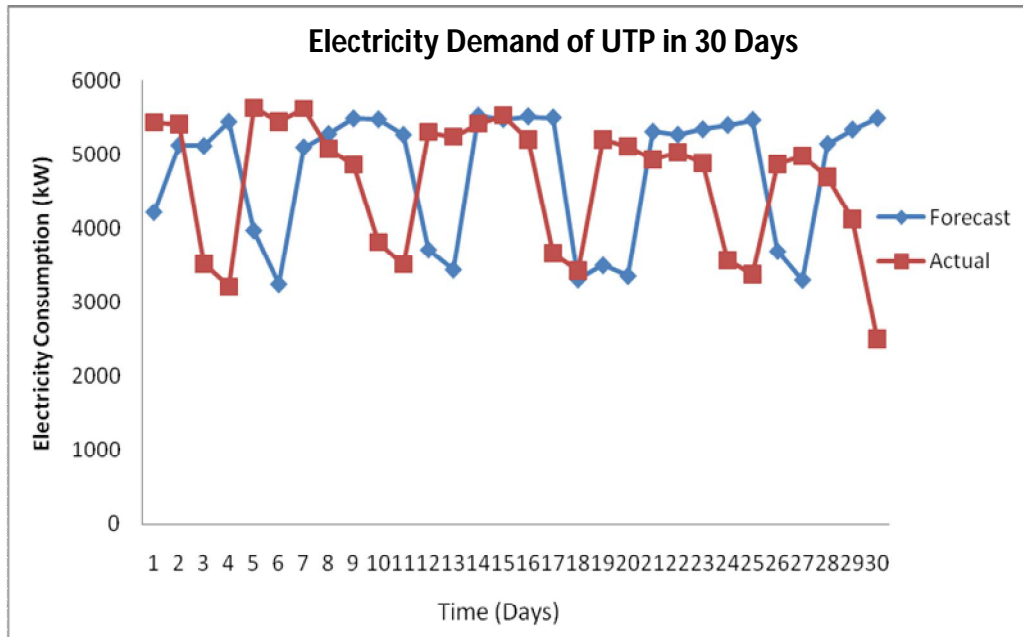


Figure 34: Comparison between Actual and Forecast Load for 3 Hidden Neurons

4.1.4.2 5 Neurons

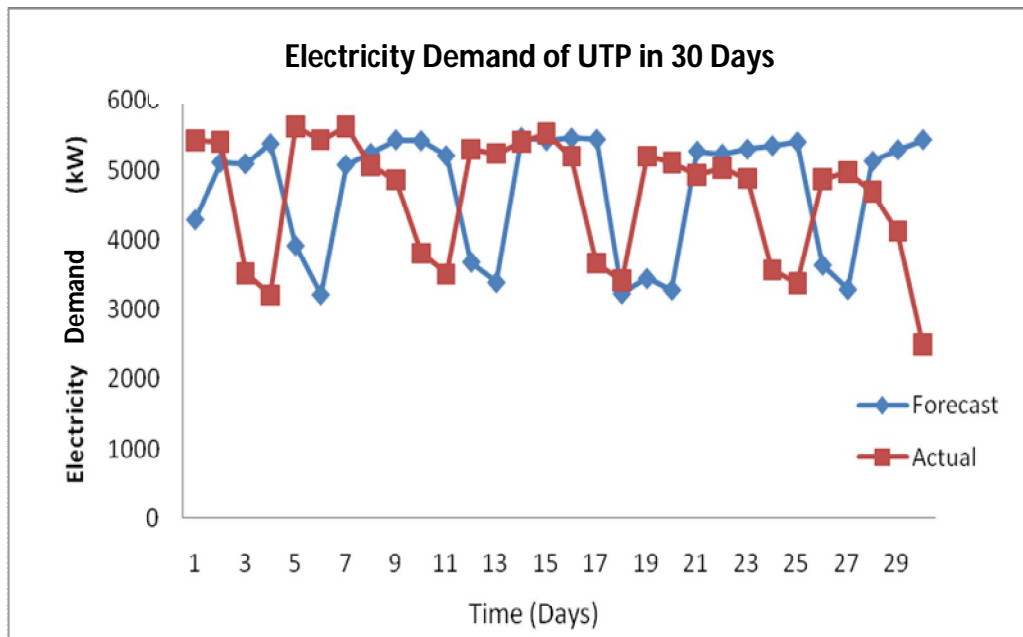


Figure 35: Comparison between Actual and Forecast Load for 5 Hidden Neurons

4.1.4.3 7 Neurons

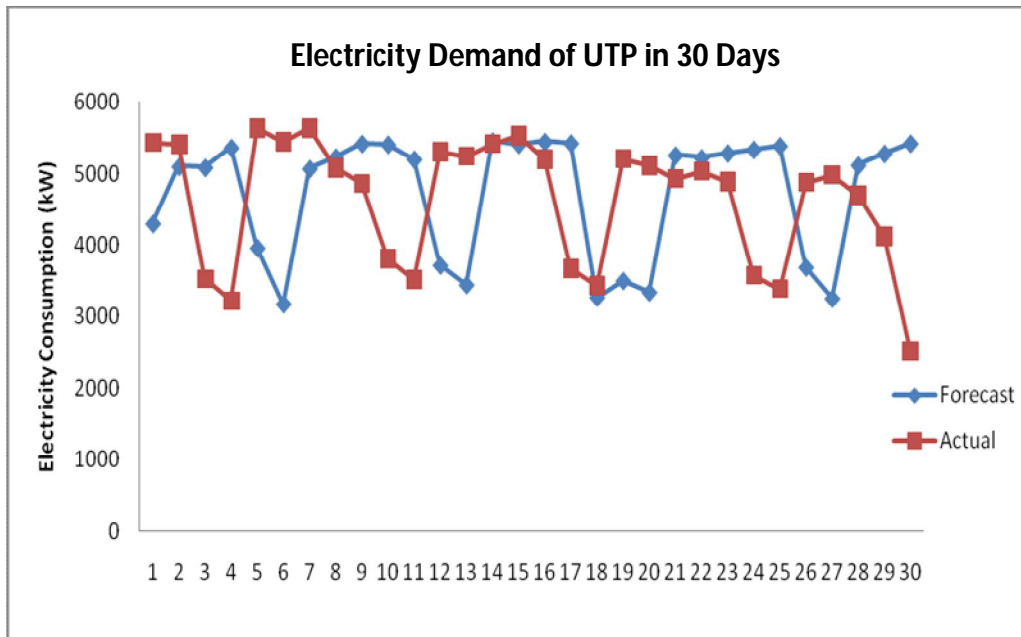


Figure 36: Comparison between Actual and Forecast Load for 7 Hidden Neurons

4.1.4.4 9 Neurons

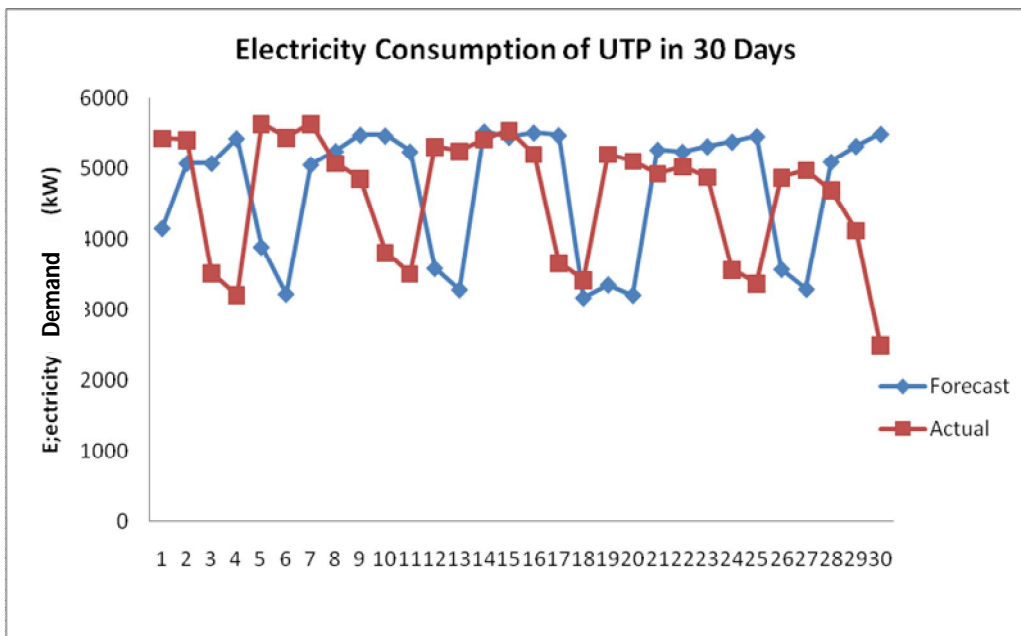


Figure 37: Comparison between Actual and Forecast Load for 9 Hidden Neurons

4.1.4.5 11 Neurons

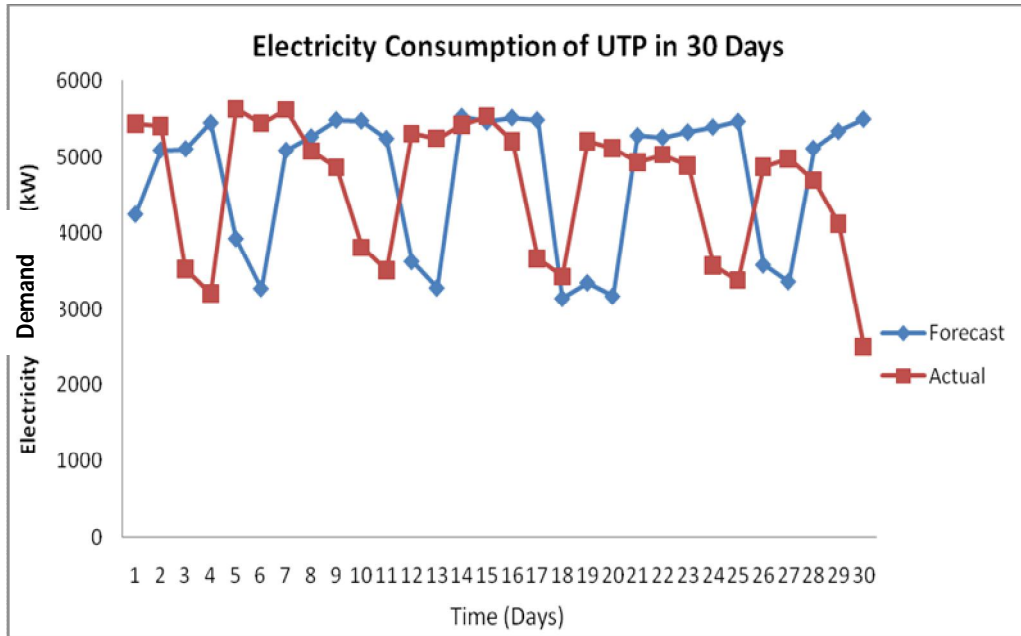


Figure 38: Comparison between Actual and Forecast Load for 11 Hidden Neurons

4.1.4.6 MAPE Values

Table 21: MAPE values for Model 4

(Sem ON 30 Days)

No. Hidden Neurons	MAPE (%)
3	21.9488
5	25.6626
7	25.3413
9	26.4561
11	26.4704

Table 21 shows the values of MAPE obtained as Model 4 is simulated using five different numbers of hidden neurons for twenty simulation. Based on the result obtained, it is found that Model 4 with the number of hidden neurons three (3) has the less value of MAPE which is 21.9488%. The training, validation and testing of Model 4 with 3 hidden neurons are as follows:

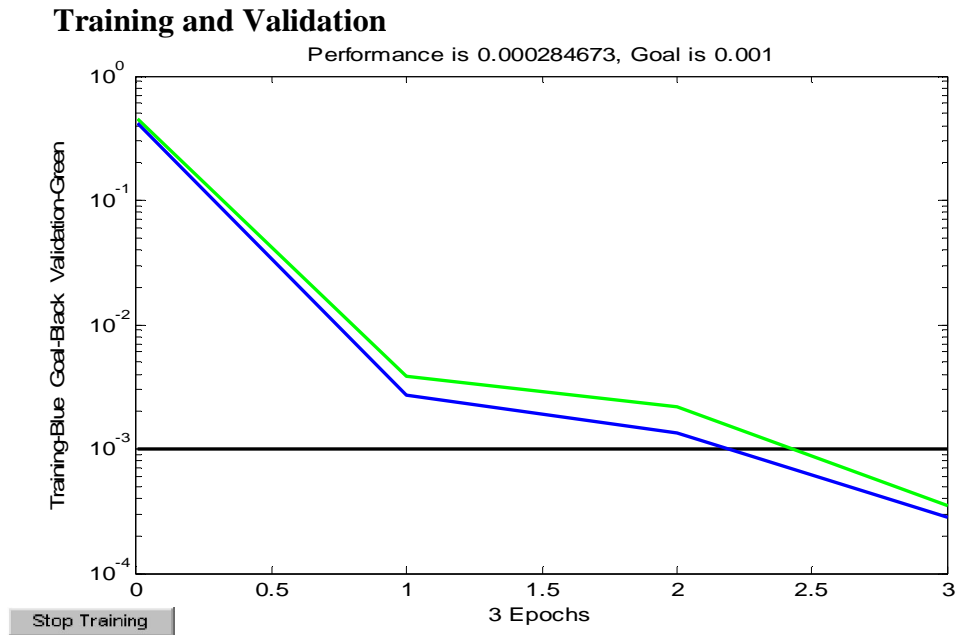


Figure 39: Training and Validation of Model 4 with 3 Neurons

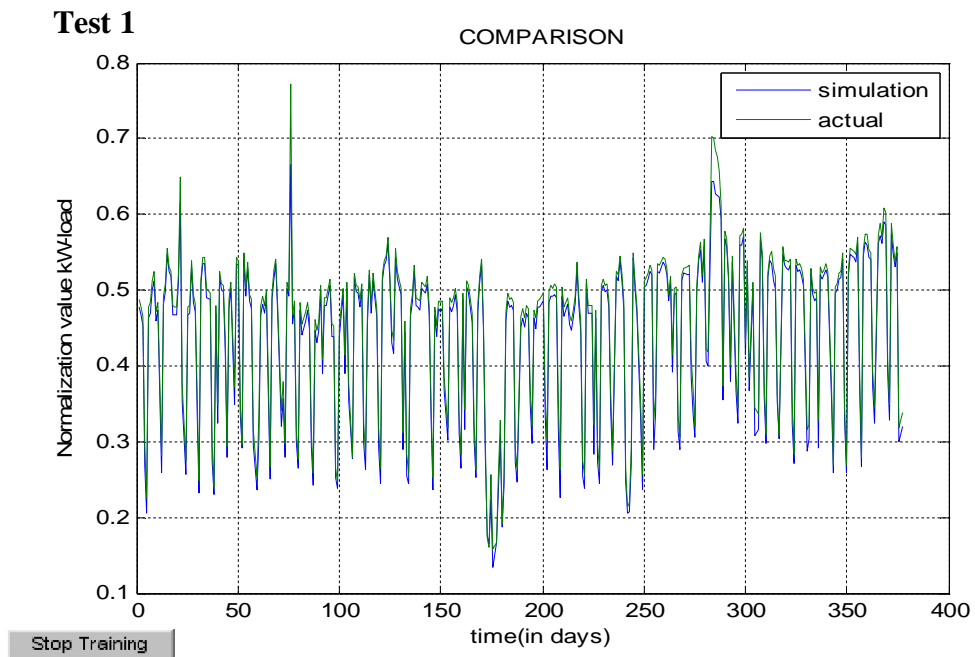


Figure 40: Test 1 result using training data

Test 2

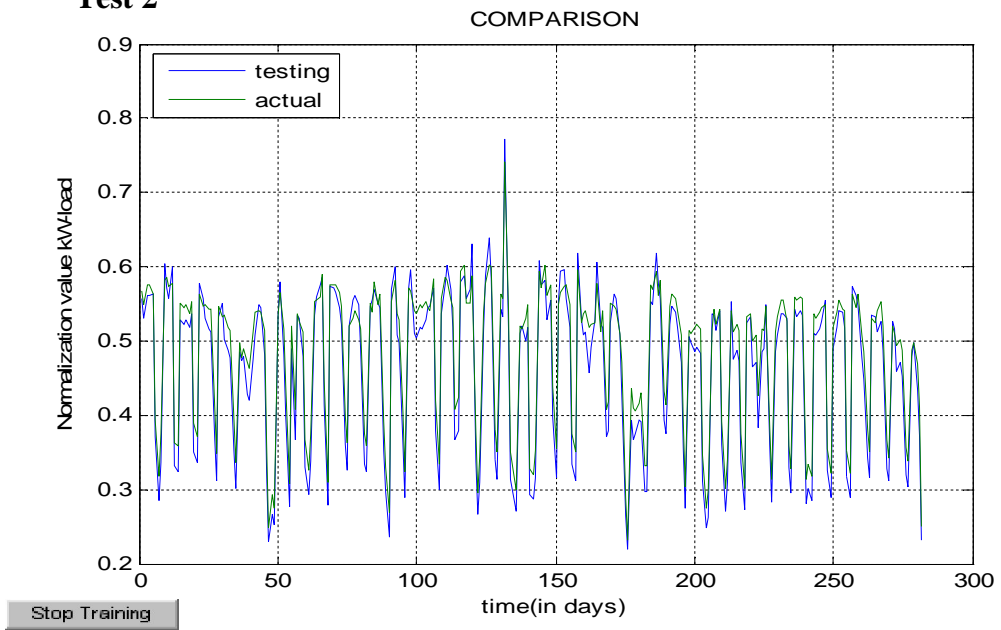


Figure 41: Test 2 result using testing data

Comparison

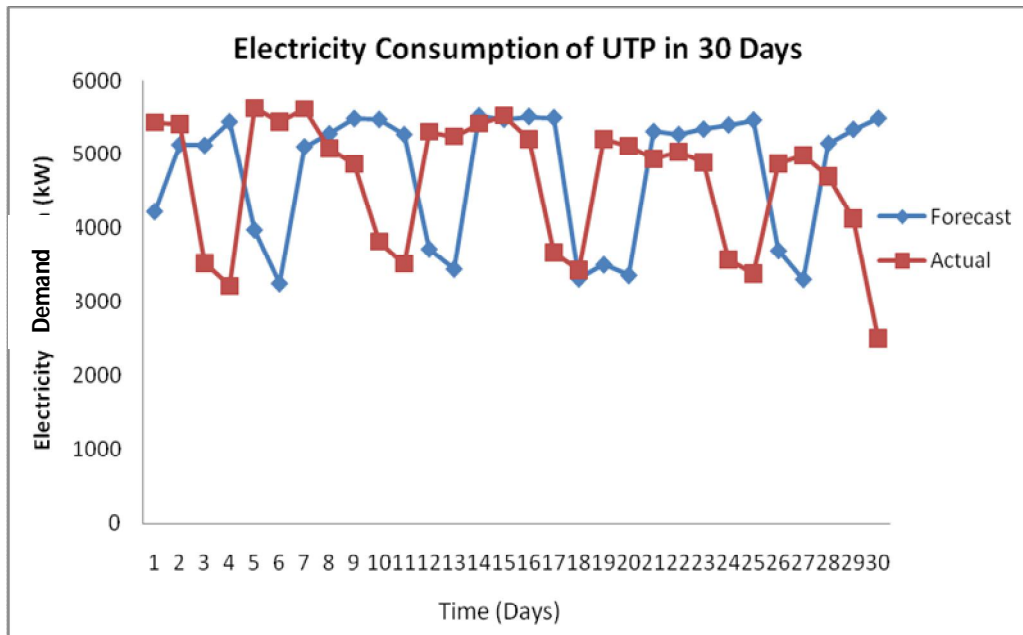


Figure 42: Comparison between Actual and Forecast Load for 3 Hidden Neurons

4.2 Result Discussion

Based on the result, it can be seen that the values of MAPE of Model 1 and Model 2 is less compared to the values of MAPE of Model 3 and 4. As mentioned in Chapter 3, Model 1 and 2 is developed to forecast the electricity demand for one week ahead. As for Model 3 and 4 is designed to forecast the electricity demand for 30 days ahead. Hence, the four developed models are only applicable for short term load forecasting. This is based on the fact that short term forecasting means the load forecasting usually from one hour to one week [4] while medium term forecasting means the load forecasting usually from a week to a year [4]. Since 30 days lies in medium term load forecasting, so it is the reason for getting greater values of MAPE for Model 3 and 4.

Model 1 with 7 hidden neurons is the best forecast model for Semester OFF and to be used as the forecast model since it got less value of MAPE. As for Semester ON, Model 2 with 9 hidden neurons is found to be the best forecast model. In comparison with the actual electricity demand, the pattern of the electricity demand is seems to be match each other. Due to that, the model can be used to forecast another sets of data as well as to use for larger power system. The values of MAPE obtained are slightly greater since the prediction error should less than 5 % [8]. Hence, new models can be developed to improve the existed model as well as to obtain more efficient models. Thus, more accurate result electricity demand can be generated.

CHAPTER 5

CONCLUSIONS AND RECOMMENDATIONS

5.1 Conclusions

Short term load forecasting can help GDC (UTP) in generator scheduling in which can minimize operating cost. Once the electricity demand by UTP is known, the exact amount of electricity can be generated at the exact time. Hence, when the demand is less than 4.2 MW, one generator can be shut down. This will minimize the operating cost since just one generator is being operated.

Short term load forecasting is essential since it assists system planner for scheduling the maintenance works. This can be done because once the electricity demand by UTP is known; the exact amount of electricity can be generated at the exact time. Hence, when the demand is less than 4.2 MW, one generator can be shut down. Upon that, maintenance work can be done to the generator that has been shut down.

Electricity forecasting activities stabilize the generation system. The stabilize generation is very essential since the non-stabilize generation system will create fault or malfunction to the system. The non-stabilize generation system existed when there is unbalanced amount of electricity demand with the electricity generation. In simple word, the electricity demand is more compared to the electricity generation by the system. Therefore, by knowing the electricity for the several steps ahead, the power generation site may notify earlier the demand to supply in order to avoid such an unbalanced system.

This project also has eased the forecast electricity demand values which are due to complexity electricity demand values. This is due the fact that Artificial Neural Networks have made it possible to experiment with theoretically poor, but data rich, models that can identify the complex non-linear relationships in the data and infer future behaviour [1]. Hence, by using ANN as the forecast model, the prediction of the nonlinear can be obtained successfully.

Upon completing the project, the ANN had being studied, analyzed and used to develop the forecast model. Using the developed forecast model, the UTP's electricity demand had been forecasted.

5.2 Recommendations

Based on the result obtained, there are some improvements need to be done to this study in order to improve the efficiency of the model. First, in this study, the effects of weather are not included in the model development. Basically, weather is also contributed main effects to the electricity consumption of GDC (UTP). Hence, the new model should be developed to include the effects of weather.

Furthermore, the hourly data can be used for the model development instead of daily data. The electricity consumption is different for every hour for each day. This difference should be considered because it will give more impact to the result obtained. Hence, by using hourly data, the result will be more accurate.

REFERENCES

- [1] James W. Taylor, Lilian M. de Menezes, Patrick E. McSharry, “A Comparison of Univariate Methods for Forecasting Electricity Demand Up to a Day Ahead” , University of Oxford, May 2005
- [2] Konstantinos Adamopoulos, “Application of Back Propagation Learning Algorithms on Multilayer Perceptions”, University of Bradford Department of Computing, May 2000
- [3] Mohsen Hayati and Yazdan Shirvany, “Artificial Neural Network Approach for Short Term Load Forecasting for Illam Region”, World Academy of Science, Engineering and Technology 28, 2007
- [4] Eugene A. Feinberg, Dora Genethliou, “Load forecasting”, State University of New York, Stony Brook
- [5] T. Yalcinoz *, U. Eminoglu, “Short term and medium term power distribution Load forecasting by neural networks”, Department of Electrical and Electronic Engineering, Nigde University, September 2004
- [6] Howard Demuth, Mark Beale, “Neural Network Toolbox”, The Math Work, 2004
- [7] Kevin Gurney, “An Introduction to Neural Networks”, 1997
- [8] Mohd Izlan Bin Mohd Ilias, “Load Profile Study By Using Artificial Neural Network”, December 2005
- [9] Xunming Li, Changyin Sun, Dengcai Gong,” Application of Support Vector Machine and Similar Day Method for Load Forecasting”, Hohai University of China.

- [10] K.L. Ho, Y.Y. Hsu, F.F. Chen, T.E. Lee, C.C. Liang, T.S. Lai, and K.K. Chen,” Short-Term Load Forecasting of Taiwan Power System using a Knowledge Based Expert System”, *IEEE Transactions on Power Systems*, 5:1214–1221, 1990.
- [11] Nima Amjady, “Short-Term Hourly Load Forecasting Using Time-Series Modelling With Peak Load Estimation Capability”, *IEEE TRANSACTIONS ON POWER SYSTEMS*, VOL. 16, NO. 4, NOVEMBER 2001.
- [12] A.J. Al-Shareef, E.A. Mohamed and E.Al-Judaibi, “ Next 24-Hours Load Forecasting Using Artificial Neural Network (ANN) for the Western Area of Saudi Arabia”, *JKAU: Eng. Sci.*, Vol.19 No.2, pp: 25-40 (2008A.D. / 1429 A.H.)
- [13] K.Y. Lee and Y. T. Cha, “Short-Term Load Forecasting Using An Artificial Neural Network”, *Transaction on Power System*, Vol. 1, No. 1, February 1992
- [14] Mohammed El-Telbany, Fawwaz El-Karmi, “Short-Term Forecasting of Jordanian Electricity Demand Using Particle Swarm Optimization”, Computer Engineering Department, Al-Ahlyyia Amman University, Jordan, Electronics and Communication Department, Al-Ahlyyia Amman University, Jordan, April 2005
- [15] Danilo Bassi, Oscar Olivares, “Medium Term Electric Load Forecasting Using TLFN Neural Networks”, *International Journal of Computers, Communications & Control*, 2006
- [16] Mohammed K. Abd, “Electricity Load Forecasting based on Framelet Neural Network Technique”, *American Journal of Applied Sciences* 6 (5): 970-973, 2009
- [17] G.A. Adepoju, S.O.A. Ogunjuyigbe, K.O. Alawode, “Application of Neural Network to Load Forecasting in Nigerian Electrical Power System”, *The Pacific Journal of Science and Technology*, May 2007

APPENDICES

APPENDIX A

UTP's Academic Calendar 2006-2009

JANUARY 2006 SEMESTER (UNDERGRADUATE)

PARTICULAR	NO. OF WEEKS	DATE	
		START	ENDS
Registration of New Students	1	14 Jan 2006	22 Jan 2006
Registration OF Existing Students	1 day	22 Jan 2006	
Lecture	7	23 Jan 2006	10 March 2006
Mid-Semester Break	1	11 March 2006	19 March 2006
Lecture	7	20 March 2006	5 May 2006
Study Week	1	6 May 2006	14 May 2006
Examination Week	3	15 May 2006	2 June 2006
End of Semester Break	7	3 June 2006	23 July 2006

JULY 2006 SEMESTER (UNDERGRADUATE)

PARTICULAR	NO. OF WEEKS	DATE	
		START	ENDS
Registration of New Students	1	15 July 2006	23 July 2006
Registration OF Existing Students	1 day	23 July 2006	
Lecture	7	24 July 2006	8 Sept 2006
Mid-Semester Break	1	9 Sept 2006	17 Sept 2006
Lecture	7	18 Sept 2006	3 Nov 2006
Study Week	1	4 Nov 2006	12 Nov 2006
Examination Week	3	13 Nov 2006	1 Dec 2006
End of Semester Break	7	2 Dec 2006	21 Jan 2007

JANUARY 2007 SEMESTER (UNDERGRADUATE)

PARTICULARS	NO. OF WEEKS	DATE	
		START	ENDS
Registration of New Students	1	13 Jan 2007	21 Jan 2007
Registration of Existing Students	1 day	21 Jan 2007	
Lecture	7	22 Jan 2007	9 March 2007
Mid-Semester Break	1	10 March 2007	18 March 2007
Lecture	7	19 March 2007	4 May 2007
Study Week	1	5 May 2007	13 May 2007
Examination Week	3	14 May 2007	1 June 2007
End of Semester Break	7	2 June 2007	22 July 2007

JULY 2007 SEMESTER (UNDERGRADUATE)

PARTICULARS	NO. OF WEEKS	DATE	
		START	ENDS
Registration of New Students	1	14 July 2007	22 July 2007
Registration of Existing Students	1 day	22 July 2007	
Lecture	7	23 July 2007	7 Sept 2007
Mid-Semester Break	1	8 Sept 2007	16 Sept 2007
Lecture	7	17 Sept 2007	2 Nov 2007
Study Week	1	3 Nov 2007	11 Nov 2007
Examination Week	3	12 Nov 2007	30 Nov 2007
End of Semester Break	7	1 Dec 2007	20 Jan 2008

JANUARY 2008 SEMESTER (UNDERGRADUATE)

PARTICULARS	NO. OF WEEKS	DATE	
		START	ENDS
Registration and Orientation of New Students	1	12 Jan 2008	20 Jan 2008
Registration of Existing Students	1 day	20 Jan 2008	
Lecture	7	21 Jan 2008	7 March 2008
Mid-Semester Break	1	8 March 2008	16 March 2008
Lecture	7	17 March 2008	2 May 2008
Study Week	1	3 May 2008	11 May 2008
Examination Week	3	12 May 2008	30 May 2008
End of Semester Break	7	31 May 2008	20 July 2008

JULY 2008 SEMESTER (UNDERGRADUATE)

PARTICULARS	NO. OF WEEKS	DATE	
		START	ENDS
Registration and Orientation of New Students	1	12 July 2008	20 July 2008
Registration of Existing Students	1 day	20 July 2008	
Lecture	10	21 July 2008	26 Sept 2008
Mid-Semester Break	1	27 Sept 2008	7 Oct 2008
Lecture	4	8 Oct 2008	31 Oct 2008
Study Week	1	1 Nov 2008	9 Nov 2008
Examination Week	3	10 Nov 2008	28 Nov 2008
End of Semester Break	7	29 Nov 2008	18 Jan 2009

JANUARY 2009 SEMESTER (UNDERGRADUATE)

PARTICULARS	NO. OF WEEKS	DATE	
		START	ENDS
Registration and Orientation of New Students	1	10 Jan 2009	18 Jan 2009
Registration of Existing Students	1 day	18 Jan 2009	
Lecture	9	19 Jan 2009	20 March 2009
Mid-Semester Break	1	21 March 2009	29 March 2009
Lecture	5	30 March 2009	1 May 2009
Study Week	1	2 May 2009	10 May 2009
Examination Week	3	11 May 2009	29 May 2009
End of Semester Break	7	30 May 2009	19 July 2009

JULY 2009 SEMESTER (UNDERGRADUATE)

PARTICULARS	NO. OF WEEKS	DATE	
		START	ENDS
Registration and Orientation of New Students	1	11 July 2009	19 July 2009
Registration of Existing Students	1 day	19 July 2009	
Lecture	9	20 July 2009	18 Sept 2009
Mid-Semester Break	1	19 Sept 2009	29 Sept 2009
Lecture	5	30 Sept 2009	30 Oct 2009
Study Week	1	31 Oct 2009	8 Nov 2009
Examination Week	3	9 Nov 2009	27 Nov 2009
End of Semester Break	7	28 Nov 2009	17 Jan 2010

APPENDIX B

MATLAB Coding of Model 1

```
clear;
clc;
echo on;
pause
load utploaddatasemoff;
p=trdat';
pt=trtgdat';
VV.P=val';
VV.T=valtg';
ts=tsdat';
tst=tstgdat';
pause
net=newff((minmax(p)),[7 1],{'tansig' 'purelin'},'trainlm');
pause
net.trainParam.epochs = 100;
net.trainParam.goal = 0.001;
net.trainparam.show=1;
%Start training the model. Please wait.
pause
net=train(net,p,pt,[],[],VV);
pause
test1=sim(net,p);
pause
day=[1:1:193];
plot(day,test1,day,pt)
xlabel('time(in days)')
ylabel('Normalization value kW-load')
title('COMPARISON')
legend('simulation','actual',1)
grid on
error_test1=(sum(abs(test1-pt))/size(pt,2))*100
pause
test2=sim(net,ts);
pause
error=(sum(abs(test2-tst))/size(tst,2))*100
day=[1:1:144];
plot(day,test2,day,tst)
xlabel('time(in days)')
ylabel('Normalization value kW-load')
title('COMPARISON')
legend('testing','actual',2)
grid on
pause
ain=ain';
aout=aout';
```

```
pload=sim(net,ain);
pload=(10000*(pload));
pload
aload=(10000*(aout));
aload
pause
MAPE=(sum(abs(pload-aload))/sum(abs(aload)))*100
day=[1:1:7];
plot(day,pload,day,aload)
xlabel('time(in days)')
ylabel('kW load')
title('COMPARISON')
legend('predicted','actual',2)
grid on
pause
```

APPENDIX C

MATLAB Coding of Model 2

```
clear;
clc;
echo on;
pause
load utploaddatasemon;
p=trdat';
pt=trtgdat';
VV.P=val';
VV.T=valtg';
ts=tsdat';
tst=tstgdat';
pause
net=newff((minmax(p)),[9 1],{'tansig' 'purelin' },'trainlm');
pause
net.trainParam.epochs = 100;
net.trainParam.goal = 0.001;
net.trainparam.show=1;
%Start training the model. Please wait.
pause
net=train(net,p,pt,[],[],VV);
pause
test1=sim(net,p);
pause
day=[1:1:386];
plot(day,test1,day,pt)
xlabel('time(in days)')
ylabel('Normalization value kW-load')
title('COMPARISON')
legend('simulation','actual',1)
grid on
error_test1=(sum(abs(test1-pt))/size(pt,2))*100
pause
test2=sim(net,ts);
pause
error=(sum(abs(test2-tst))/size(tst,2))*100
day=[1:1:289];
plot(day,test2,day,tst)
xlabel('time(in days)')
ylabel('Normalization value kW-load')
title('COMPARISON')
legend('testing','actual',2)
grid on
pause
ain=ain';
aout=aout';
```

```
pload=sim(net,ain);
pload=(10000*(pload));
pload
aload=(10000*(aout));
aload
pause
MAPE=(sum(abs(pload-aload))/sum(abs(aload)))*100
day=[1:1:7];
plot(day,pload,day,aload)
xlabel('time(in days)')
ylabel('kW load')
title('COMPARISON')
legend('predicted','actual',2)
grid on
pause
```

APPENDIX D

MATLAB Coding of Model 3

```
clear;
clc;
echo on;
pause
load utploaddatasemoff;
p=trdat';
pt=trtgdat';
VV.P=val';
VV.T=valtg';
ts=tsdat';
tst=tstgdat';
pause
net=newff((minmax(p)),[3 1],{'tansig' 'purelin' },'trainlm');
pause
net.trainParam.epochs = 100;
net.trainParam.goal = 0.001;
net.trainparam.show=1;
%Start training the model. Please wait
pause
net=train(net,p,pt,[],[],VV);
pause
test1=sim(net,p);
pause
day=[1:1:184];
plot(day,test1,day,pt)
xlabel('time(in days)')
ylabel('Normalization value kW-load')
title('COMPARISON')
legend('simulation','actual',1)
grid on
error_test1=(sum(abs(test1-pt))/size(pt,2))*100
pause
test2=sim(net,ts);
pause
error=(sum(abs(test2-tst))/size(tst,2))*100
day=[1:1:137];
plot(day,test2,day,tst)
xlabel('time(in days)')
ylabel('Normalization value kW-load')
title('COMPARISON')
legend('testing','actual',2)
grid on
pause
ain=ain';
aout=aout';
```

```
pload=sim(net,ain);
pload=(10000*(pload));
pload
aload=(10000*(aout));
aload
pause
MAPE=(sum(abs(pload-aload))/sum(abs(aload)))*100
day=[1:1:30];
plot(day,pload,day,aload)
xlabel('time(in days)')
ylabel('kW load')
title('COMPARISON')
legend('predicted','actual',2)
grid on
pause
```


APPENDIX E

MATLAB Coding of Model 4

```
clear;
clc;
echo on;
pause
load utploaddatasemon;
p=trdat';
pt=trtgdat';
VV.P=val';
VV.T=valtg';
ts=tsdat';
tst=tstgdat';
pause
net=newff((minmax(p)),[3 1],{'tansig' 'purelin' },'trainlm');
pause
net.trainParam.epochs = 100;
net.trainParam.goal = 0.001;
net.trainparam.show=1;
%Start training the model. Please wait.
pause
net=train(net,p,pt,[],[],VV);
pause
test1=sim(net,p);
pause
day=[1:1:377];
plot(day,test1,day,pt)
xlabel('time(in days)')
ylabel('Normalization value kW-load')
title('COMPARISON')
legend('simulation','actual',1)
grid on
error_test1=(sum(abs(test1-pt))/size(pt,2))*100
pause
test2=sim(net,ts);
pause
error=(sum(abs(test2-tst))/size(tst,2))*100
day=[1:1:282];
plot(day,test2,day,tst)
xlabel('time(in days)')
ylabel('Normalization value kW-load')
title('COMPARISON')
legend('testing','actual',2)
grid on
pause
ain=ain';
aout=aout';
```

```
pload=sim(net,ain);
pload=(10000*(pload));
pload
aload=(10000*(aout));
aload
pause
MAPE=(sum(abs(pload-aload))/sum(abs(aload)))*100
day=[1:1:30];
plot(day,pload,day,aload)
xlabel('time(in days)')
ylabel('kW load')
title('COMPARISON')
legend('predicted','actual',2)
grid on
pause
```

APPENDIX F

PROJECT'S GANTT CHART

