Neural Network Based Route Weight Classification and Prediction for Traffic Management System

Submitted by

Md Ashfaqul Islam

Id. 2012-2-60-054

Supervised by

Dr. Shamim Akhter

Assistant Professor

Department of Computer Science and Engineering

East West University

A Project Submitted in Partial Fulfillment of the Requirements for the Degree of Bachelors of Science in Computer Science and Engineering

To the

Department of Computer Science and Engineering

East West University

Dhaka, Bangladesh

Abstract

Traffic jam is a major problem in Dhaka City, so a traffic management support system, with less cost, flexible, easily maintainable and secured is in demand. For monitoring road traffic condition, Internet based real time bi-directional communication provides a lot of benefits. For making traffic system more realistic and reliable, dynamic route computation is a vital requirement. Therefore, for predicting road weights, an integrated approach with multiple data feeds and back propagation neural network with Levenberg Marquardt optimization is applied. The traffic system where NN based dynamic weights computation is used and much more suitable to find the optimal routes. Inclusion of BPNN with LM achieved more than 90% accuracy. NARX time delay neural network is used to predict different feature's weights and those are applied in this neural network to determine the road weights of different roads. NARX neural network performs better than weighted mean moving average to predict different feature's weights.

Declaration

We hereby declare that, this project was done under CSE497 and has not been submitted elsewhere for requirement of any degree or diploma for any purpose except for publication.

Md Ashfaqul Islan

ID: 2012-2-60-054

Department of Computer Science and Engineering

East West University

Letter of Acceptance

I hereby declare that this thesis is from the student's own work and best effort of mine, and all other source of information used have been acknowledged. This thesis has been submitted with my approval.

Dr. Md. Shamim Akhter
Assistant Professor
Department of Computer Science and Engineering
East West University

Supervisor

Dr. Mozammel Huq Azad Khan
Chairperson & Professor
Department of Computer Science and Engineering
East West University

Chairperson

Acknowledgement

First of all, I would like to thank almighty Allah for giving me the strength and knowledge to complete this thesis work.

I would like to express my humble gratitude and thanks to my honorable supervisor Dr. Md. Shamim Akhter, Assistant Professor, Department of Computer Science and Engineering, East West University for his kind guidance throughout this thesis work.

I am grateful to the faculty of Computer Science and Engineering for their support.

I am thankful to my parents for their endless support.

TABLE OF CONTENTS

	Page
Abstract	**ii**
Declaration	**iii**
Letter of Acceptance	**iv**
Acknowledgement	**v**
List of Tables	**viii**
List of Figures	**ix**
Chapter 1:	
Introduction	**1**
1.1 Objective of the Research	1
1.2 Methodology	1
1.3 Outcomes of the Research	1
Chapter 2	
Background Study	**2-8**
2.1 BI-Directional Traffic management system	2-4
2.1.1 Test Zone/Study Area	2-3
2.1.2 Static Road Weight Matrix	3
2.1.3 Static Road Weight Matrix	3-5
2.2Road weight Forecast	5-8
Chapter 3	
Related Works	**9**
Chapter 4	
Materials and Method	**10-26**
4.1 Back Propagation methods	10-18
4.1.1 Gradient Decent Back Propagation method	10-17
4.1.2 Levenberg-Marquardt Back Propagation method	17-19
4.2 Closed loop NARX Time Delay Neural Network	19-26
Chapter 5	
Experimental Results and their Analysis	**27-31**
5.1 Results of GD and LM based Neural Network	27-29
5.2 Results of NARX Implementation	29-31

Chapter 6
Conclusion and Future Work **32**
6.1 Conclusion 32
6.2 Future Work 32
References **33-35**

LIST OF TABLES

	Page
Table 2.1: Part of the weight Matrix	4
Table 2.2: Part of the weight Matrix	5
Table 2.3: Temperature of First Fifteen days December 2006	6
Table 2.4: Temperature of Last Fifteen days December 2006	7
Table 4.1: Input in the network	10
Table 4.2: Output of the network	11
Table 4.3: Simple mean average for Temperature	20-21
Table 4.4: Oscillator for Temperature	21
Table 4.5: Input and output of first NARX Network	22
Table 4.6: Inputs of present and last five day's input in input buffer for getting output of sixth day	23
Table 4.7: Outputs of previous five days in output buffer for getting output of sixth day	23
Table 4.8: Inputs of present and last five day's input in input buffer for getting output of seventh day	24
Table 4.9: Outputs of previous five days in output buffer for getting output of seventh day	24
Table 4.10: Input and output of NARX Network for road status	25
Table 5.1: Comparison of regression analysis of NN training techniques (matlab)	28
Table 5.2: True error by K-Fold cross validation	29
Table 5.3: Accuracy of predicted temperature and rainfall for different time delay and different no of neurons in hidden layer	29-30
Table 5.4: Accuracy of predicted road status for different time delay and different no of neurons in hidden layer	30-31

LIST OF FIGURES

Page

Figure 2.1: Architecture of the proposed system. 2
Figure 2.2: Route Suggestion before and after weight increase 3
Figure 2.3: Route Suggestion before and after weight increase 3
Figure 2.4: Visualization of the generated Decision Tree 4
Figure 4.1: Neural Network with GD method 10
Figure 4.2: Neural Network with LM method 17
Figure 4.3: Closed loop NARX neural network. 20
Figure 4.4: Open loop NARX neural network. 23
Figure 4.5: K subsets for K-Fold cross validation. 26
Figure 5.1: Selection # of Neuron in Hidden Layer 27
Figure 5.2: Regression analysis of Gradient Decent (GD) NN 28
Figure 5.3: Regression analysis of Levenberg-Marquardt (LM) NN 28
Figure 5.4: Actual and Predicted weights of temperature and rainfall 30
Figure 5.5: Actual and Predicted weights of road status. 31

Chapter 1

Introduction:

1.1 Objective of the Research

Traffic jam is a very important problem in our country. Traffic problem is getting severe day by day. For solving traffic congestion through radio, television and several websites provide current traffic information. But prediction is not used to provide traffic information. For solving this problem different solutions have been proposed [4][5][6][7][8][24]. For these different solutions, surveillance system like infrared sensors, CCTV cameras, GSM, sound sensors, aerial surveillance etc can be used. For measuring travel times, Bluetooth and Wi-Fi signals can be used [29]. Each solution is very costly for the high cost of surveillance system. Heavy installations and regular maintenances are also responsible for this high cost. Again in extreme weather condition, some of these tools are not effective. These tools are effective in normal weather condition. The objective of proposed neural network based road weight prediction is to predict road weight accurately for different time condition and reduce the surveillance costs. Later, Dijkstra's Algorithm [15] is used to get the optimal route from a source to destination.

1.2 Methodology

In the proposed neural network [14][23], at first the weights of features are predicted by using closed loop NARX time delay neural network. Road maintenance's weights are not predicted by time delay neural network. Then a simple neural network is used to predict the route weights by using these predicted feature's weights. It has been ensured that each neural network is trained by considerable amount of data, so accuracy rate of predicted road weight is high. For training network, back propagation method is used.

1.3 Outcomes of the Research

For road weight prediction, two back propagation methods have been used. Those are gradient decent back propagation (BP) and Levenberg-Marquardt back propagation (LM_BP). In both cases, above 90% percent accuracy is achieved. It shows that, LM_BP perform better than BP. Closed loop NARX neural network perform better than weighted moving mean average.

Chapter 2

Background Study

2.1 BI-Directional Traffic management system

In this section, for the proposed real time system a generic architecture has been discussed. For different purposes like named traffic communication, data networking [24], video conferencing [26], chatting and telecommunication, bi-directional communication is used. We can see the elements of proposed architecture in figure 2.1. Their involved signaling can also be seen in figure 2.1. In this figure, we can see some vehicles, which are indicated as client. Clients have access to GPS. GPS allows clients to gather their current location (latitude and longitude). The clients are equipped with devices which are supported by WebSocket protocol RFC6455 [9][25][27]. Clients are also able to handle the web request/response over the HTTP [16]. Necessary scripts are requested by client from server at beginning of application. By utilizing proxy connection, scripts initiate a WebSocket connection. Server has been informed about the method available on client side. Server accepts WebSocket connection. Until any endpoint requests to disconnect, server remains open. The logical parts of the web services are hosted by WebSocket server. The server application consists the real time location plotting system on region based available data.

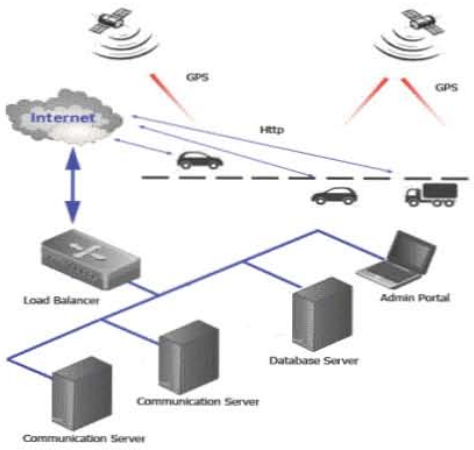

Figure 2.1 Architecture of the proposed system.

2.1.1 Test Zone/Study Area

Traffic jam is a serious problem in our city, which shows no signs of improvement. This problem causes air pollution, sound pollution, accident and late arrival at destination etc [1][2][3][28]. For city like Dhaka traffic jam is a severe problem. It causes lost of valuable time (approx 200%) being stuck in traffic jams and CO_2 emissions (approx 300%) has also increased due to the

current situation. Banani area is the heart of Dhaka road network. This area has been selected for testing vehicle routings. Like other intersections at Dhaka city it follows the all way stop control intersection resulting in huge traffic jam at peek hours which make this difficult for traffic police to control the vehicle flow in a nimble way. Inside Banani area road network shape as quadrangle which is suitable for our test simulation. In fig 2.2, each road is divided into segments depending on the junctions to test effectiveness of vehicle routing. Each segment is marked as a rectangle. Each segment has five points – four corners and midpoint. The proper road segment is indentified by the required four corners. The midpoint of a road defines the road marker. By using difference between client's location and segment's road marker, client can decide nearest segment.

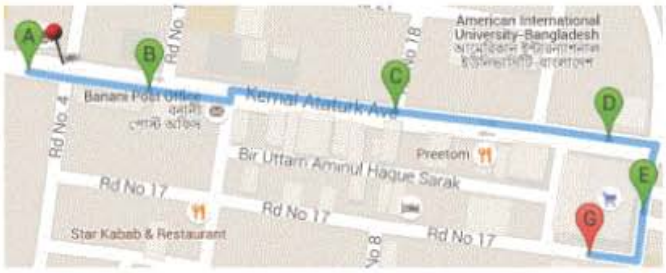

Figure 2.2: Route Suggestion before weight increase

Segment location is stored in data base and placed in Google Map [12][13].

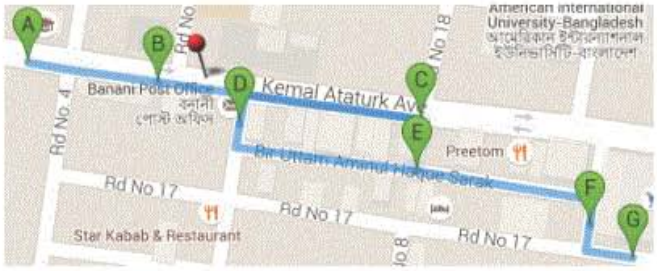

Figure 2.3: Route Suggestion after weight increase

2.1.2 Static Road Weight Matrix

The road weights are stored in database server. One way roads are not considered here. In real test weights must be calculated dynamically data which are coming from weather forecast, traffic congestion, road condition etc. Server application able to map any transmitted location to an existing location. Then it can plot these on an aerial map.

2.1.3 Static Road Weight Matrix

Data scraping technique is used to extract data from external data sources. Data is scraped for each of the road segments in a regular interval. Then data is stored in a historic table. Meanwhile, weights for the road segments are also updated in weight matrix (Table 2.1).

Table 2.1: Part of the weight Matrix

Serial	Weight ID	From ID	To ID	Road Weight
1	1	2	3	100
2	7	2	6	5
3	8	3	2	5
4	9	3	15	5
5	10	3	4	5
6	11	4	3	5
7	12	4	23	5
8	13	4	5	5

Decision tree ([10], [11], [19] and [40]) is used to update the weights of weight matrix. The top node of the tree is the influential piece of data that affects the response variable in model. For test purpose four weather attributes are considered- temperature, rainfall, humidity and wind. These attributes are used to determine the environmental status. We can change the number of attributes according to system requirements. Rainfall is an important attribute for determining road condition. For heavy rain road segments can be submerged. As a result, it causes slower traffic movements [28]. The attribute temperature is also important. The engines and air condition of the vehicle release heat which increases temperature of the road segment. Road safety is directly influenced by the gusts of wind. It causes slower traffic movement. Humidity, wind and temperature have direct influence to predict the future rainfall in a particular area. These four attributes have direct or indirect relationship with road congestion or traffic movement. Increased weights of a road segment by DT decrease the probability of choosing road segment as optimal path. By using decision tree [19][20] in figure 2. 4,

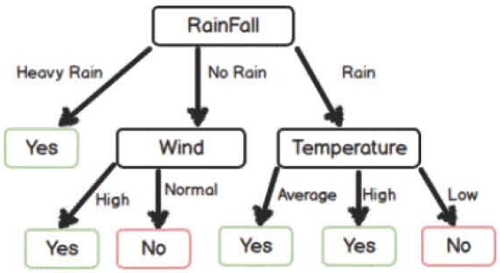

Figure 2.4: Visualization of the generated Decision Tree

Road weights are predicted in Table 2.2. Example: In figure 2.2, we can see the point 'D' is adjacent to point 'C' and it is on the right side of point 'C'. In figure 2.2 we can see the suggested route before weight of the point 'D' is increased. Now for using decision tree, the weight for point 'D' has increased. In figure 2.3, we can see the alternate route, where point 'D' is relocated.

Table 2.2: Part of the weight Matrix

Rain Fall	Temperature	Humidity	Wind	Increase Weight
No Rain	High	High	Normal	No
No Rain	High	High	High	Yes
Rain	High	High	Normal	Yes
Heavy Rain	High	High	Normal	Yes
Heavy Rain	Low	Normal	Normal	Yes
Heavy Rain	Low	Normal	High	Yes

2.2 Road weight Forecast

Here, future road weights are predicted by utilizing the structured data from the historic database. At first module predicts the weights of Rainfall, Temperature, road status etc. Weighted moving average algorithm [17] is used to predict the factor values.

$$S_{n+1} = \sum_{i=1}^{n} \frac{T_n}{n} = \frac{1}{n} T_n + \frac{1}{n} \sum_{i=1}^{n-1} T_{n-1} \quad (2.1)$$

Here, S_n represents predicted weights of each parameter (Rainfall, Temperature, road status). T_n represents actual weights of each parameter. n is the window size that means how many days are considered.

$$= \frac{1}{n} T_n + \frac{(n-1)}{n} \times \frac{1}{(n-1)} \sum_{i=1}^{n-1} T_{n-1} \quad (2.2)$$

Now,

$$S_n = \sum_{i=1}^{n-1} \frac{T_{n-1}}{(n-1)} \quad (2.3)$$

So,

$$S_{n+1} = \frac{1}{n} T_n + \frac{(n-1)}{n} S_n = \frac{1}{n} T_n + \left(1 - \frac{1}{n}\right) S_n \quad (2.4)$$

$$S_{n+1} = \alpha T_n + (1-\alpha) S_n \quad (2.5)$$

In this process most recent data have higher impact on prediction of future weights of each attributes. The coefficient α represents the degree of weighting decrease, a constant smoothing factor between zero and one. If value of α is close to one that means present data's priority is

higher than old data. So, value of α needs to be as large as possible. But it can not be one, because in that case all priority will be on latest data and there will be no priority on previous data. Now,

$$S_n = \alpha T_{n-1} + (1-\alpha) S_{n-1} \tag{2.6}$$

So,

$$\begin{aligned} S_{n+1} &= \alpha T_n + (1-\alpha)[\alpha T_{n-1} + (1-\alpha) S_{n-1}] = \alpha T_n + (1-\alpha)\alpha T_{n-1} + (1-\alpha)^2 S_{n-1} \\ &= \alpha T_n + (1-\alpha)\alpha T_{n-1} + (1-\alpha)^2 [\alpha T_{n-2} + (1-\alpha) S_{n-2}] \\ &= \alpha T_n + (1-\alpha)\alpha T_{n-1} + (1-\alpha)^2 \alpha T_{n-2} + (1-\alpha)^3 S_{n-2} \\ &= \alpha T_n + (1-\alpha)\alpha T_{n-1} + (1-\alpha)^2 \alpha T_{n-2} + (1-\alpha)^3 \alpha T_{n-3} + \cdots + (1-\alpha)^i \alpha T_{n-i} + \cdots \\ &\quad + (1-\alpha)^n S_{n-(n-1)} \end{aligned} \tag{2.7}$$

Among all parameters which are used in determining road weights, temperature is one of those.

Table 2.3: Temperature of First Fifteen days December 2006

Date	Day	Original Temperature(weighted form)	Predicted Temperature(weighted form)
12/1/2006	1	1	1
12/2/2006	2	1	0.8
12/3/2006	3	0	0
12/4/2006	4	1	1
12/5/2006	5	1	0.7
12/6/2006	6	1	1
12/7/2006	7	1	1
12/8/2006	8	1	1.1
12/9/2006	9	1	1
12/10/2006	10	1	0.9
12/11/2006	11	1	1
12/12/2006	12	1	1
12/13/2006	13	0	0
12/14/2006	14	0	0
12/15/2006	15	1	1

In table 2.3, we can see the actual and predicted temperatures of first fifteen days of December 2006. Now by applying algorithm of weighted moving average, temperature of coming days can be predicted. Here, $\alpha = 0.6$

$P.temp_{16} = \alpha \times A.temp_{15} + (1-\alpha) \times \alpha \times A.temp_{14} + (1-\alpha)^2 \times \alpha \times A.temp_{13} + (1-\alpha)^3 \times \alpha \times A.temp_{12} + (1-\alpha)^4 \times \alpha \times A.temp_{11} + (1-\alpha)^5 \times \alpha \times A.temp_{10} + (1-\alpha)^6 \times \alpha \times A.temp_{09} + (1-\alpha)^7 \times \alpha \times A.temp_{08} + (1-\alpha)^8 \times \alpha \times A.temp_{07} + (1-\alpha)^9 \times \alpha \times A.temp_{06} + (1-\alpha)^{10} \times \alpha \times A.temp_{05} + (1-\alpha)^{11} \times \alpha \times A.temp_{04} + (1-\alpha)^{12} \times \alpha \times A.temp_{03} + (1-\alpha)^{13} \times \alpha \times A.temp_{02} + (1-\alpha)^{14} \times \alpha \times A.temp_{01} + (1-\alpha)^{15} \times P.temp_{01} = 0.6 \times 1 + (1-0.6) \times 0.6 \times 0 + (1-0.6)^2 \times 0.6 \times 0 + (1-0.6)^3 \times 0.6 \times 1 + (1-0.6)^4 \times 0.6 \times 1 + (1-0.6)^5 \times 0.6 \times 1 + (1-0.6)^6 \times 0.6 \times 1 + (1-0.6)^7 \times 0.6 \times 1 + (1-0.6)^8 \times 0.6 \times 1 + (1-0.6)^9 \times 0.6 \times 1 + (1-0.6)^{10} \times 0.6 \times 1 + (1-0.6)^{11} \times 0.6 \times 1 + (1-0.6)^{12} \times 0.6 \times 0 + (1-0.6)^{13} \times 0.6 \times 1 + (1-0.6)^{14} \times 0.6 \times 1 + (1-0.6)^{15} \times 1 = 0.663989934$

Table 2.4: Temperature of Last Fifteen days December 2006

Date	Day	Original Temperature(weighted form)	Predicted Temperature(weighted form)
12/16/2006	16	1	0.663989934
12/17/2006	17	1	0.865595973
12/18/2006	18	1	0.946237316
12/19/2006	19	1	0.978496
12/20/2006	20	1	0.9913984
12/21/2006	21	1	0.99655936
12/22/2006	22	1	0.998623744
12/23/2006	23	1	0.999449498
12/24/2006	24	1	0.999779799
12/25/2006	25	1	0.99991192
12/26/2006	26	1	0.999964768
12/27/2006	27	0	0.999985907
12/28/2006	28	1	0.399993289
12/29/2006	29	0	0.759997316
12/30/2006	30	0	0.304
12/31/2006	31	0	0.121599639

In the same way, weights of rainfall and road status are determined by using weighted moving average algorithm.

There are a lot of factors which can influence the road weights in different ways. These predicted factor values are used to determine estimated road weights. These estimated road weights are stored in “predicted road weight” matrix.

Chapter 3

Related Works

In many researches neural network has been used to solve many problems [45][46]. Yujing Yang and Junhai Ma [33] use LM_BP based neural network in the deduction and application of Climate Index in Chinese real state market. HenryA.Rowley, Shumeet Baluja and Takeo Kanade [34] work on neural network based face detection system. N.Verma, M. S. Sobhan and T. Jalil [4] have proposed Novel Design Proposal For Real Time Traffic Monitoring & Management of Dhaka Metropolitan City with Rcap. M. R. Rahman and S. Akhter [10][11] work on Bi-directional traffic management support system with decision tree based dynamic routing and with GPS and websocket. V .Preetha Pallavi and V .Vaithiyanathan [35] work on Combined Artificial Neural Network and Genetic Algorithm for Cloud Classification. Meera Narvekar and Priyanca Fargose [36] work on Daily Weather Forecasting using Artificial Neural Network. Arti R. Naik and S.K.Pathan [37] work on Weather Classification and Forecasting using Back Propagation Feed-forward Neural Network. Pooja Malik, Prof Sranjeet Singh and Binni Arora [38] work on an effecting Weather Forecasting Using Neural Network. Shaminder Singh, Pankaj Bhambri and Jasmeen Gill [39] work on Time Series Based Temperature Prediction using Back Propagation with Genetic Algorithm Technique. Xiaoming Zheng and Sven Koenig [41] work on a project on Gesture Recognition with Neural Networks for “Introduction to Artificial Intelligence” Classes. Zhang Minli and Qiao Shanshan [42] work on Research on the Application of Artificial Neural Networks in Tender Offer for Construction Projects. Karan Kamdar and Amit Mathapati [43] work on Artificial Neural Networks for Cancer Research in Prediction & Survival. Vidushi Sharma, Sachin Rai and Anurag Dev [44] work on A Comprehensive Study of Artificial Neural Networks.

Chapter 4

Materials and Method

In this chapter we are going to describe two ways of calculating road weights using two back propagation methods- Gradient Decent back propagation (BP) and Levenberg-Marquardt back propagation. We are also going to describe the way of calculating feature's (temperature, rainfall and road status) weight using closed loop NARX Time Delay Neural Network.

4.1 Back Propagation methods

4.1.1 Gradient Decent Back Propagation method

This method can be better understood if preceded by an example. So this Section will aim at expanding the algorithm with an example.

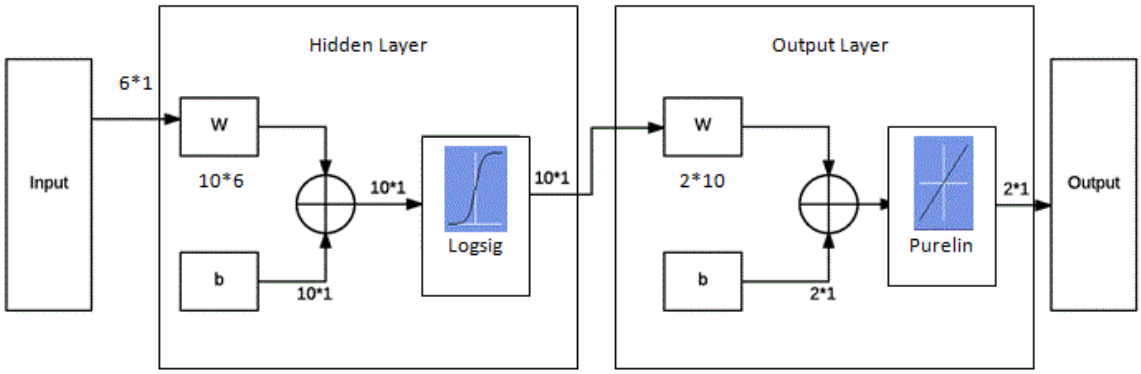

Figure 4.1: Neural Network with GD method

Table 4.1: Input in the network

Temperature	Rainfall	Wind	Humidity	Road status	Road maintenance
1	0	1	1	1	1
1	0	1	1	2	0
1	0	1	1	0	0
1	0	1	1	3	0
1	0	0	1	1	1
1	0	0	1	3	0
1	0	0	1	2	0
1	0	0	1	3	0
0	0	1	2	2	1

Table 4.2: Output of the network

Road weight	Binary Form of Road Weight	
	Rw1	Rw2
2	1	0
0	0	0
0	0	0
3	1	1
2	1	0
3	1	1
0	0	0
3	1	1
2	1	0

Here, as input weights of six features is got into the network. Table 4.1 showing a set of data set (input) which will be used for training of the network. Table 4.2 showing corresponding output of each data point of table 4.1. Now this network is providing output in binary form, so column two and three of table 4.2 is representing the binary result of column one. Output layer of network has two neurons for this binary form of output. Output matrix is 2*1. Input contains six features, so input matrix is 6*1. Now, hidden layer no of neurons are not fixed. Here, in this network hidden layer has ten neurons. As input has six features and hidden layer has 10 neurons, weight matrix of hidden layer is 10*6 and bias matrix is 10*1. Output matrix of hidden layer is 10*1. Output layer's weight and bias matrix are correspondingly 2*10 and 2*1. Data point's weights never change. By updating weight and bias matrixes in every epoch, outputs are getting close to original outputs. Gradient Decent Back propagation method is used for updating weights and biases and for measuring performance mean square method is used.

In equation 4.1 and 4.2, initial weights of weights and bias matrix of hidden layer can be seen respectively which is set randomly before training.

$$\begin{bmatrix} -1.8 & 1.3 & 1.5 & -2.1 & 0.4 & 0.6 \\ -1.8 & 2.1 & 0.3 & 0.3 & -2.5 & -2.7 \\ -1.8 & 1.7 & 1.1 & 0.0 & 2.8 & 1.3 \\ -2.5 & 1.5 & 1.5 & 0.7 & 0.0 & 2.0 \\ 0.7 & 1.3 & 2.6 & -2.5 & -0.5 & 0.7 \\ -0.7 & 1.4 & 2.0 & -2.4 & -2.0 & 0.3 \\ 0.7 & 2.0 & -1.2 & -2.0 & -0.9 & 2.3 \\ 1.4 & 1.2 & 0.6 & -2.4 & -0.7 & 2.5 \\ 1.8 & -1.1 & -0.3 & -1.6 & 2.8 & -1.2 \\ 2.7 & 0.0 & -2.0 & -1.4 & 0.0 & 1.6 \end{bmatrix} \quad (4.1)$$

$$\begin{bmatrix} -4.4 \\ -3.2 \\ 1.9 \\ -1.6 \\ 0.4 \\ -0.4 \\ -1.2 \\ 1.5 \\ 3.2 \\ 4.0 \end{bmatrix} \tag{4.2}$$

For showing, how gradient decent back propagation method works during training the first data point of table 4.1 has been selected. Below input matrix of that data point can be seen,

$$\begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

For generating output of hidden layer, input matrix is multiplied with weight matrix of hidden layer and then it is added bias matrix of hidden layer. Now transfer function of hidden layer is logsig. So logistic function is used on this intermediate result to get the output of hidden layer.

$$a_{hidden} = logsig(W_{hidden} \times P + b_{hidden}) \tag{4.3}$$

$$a_{hidden} = logsig\left(\begin{bmatrix} -1.8 & 1.3 & 1.5 & -2.1 & 0.4 & 0.6 \\ -1.8 & 2.1 & 0.3 & 0.3 & -2.5 & -2.7 \\ -1.8 & 1.7 & 1.1 & 0.0 & 2.8 & 1.3 \\ -2.5 & 1.5 & 1.5 & 0.7 & 0.0 & 2.0 \\ 0.7 & 1.3 & 2.6 & -2.5 & -0.5 & 0.7 \\ -0.7 & 1.4 & 2.0 & -2.4 & -2.0 & 0.3 \\ 0.7 & 2.0 & -1.2 & -2.0 & -0.9 & 2.3 \\ 1.4 & 1.2 & 0.6 & -2.4 & -0.7 & 2.5 \\ 1.8 & -1.1 & -0.3 & -1.6 & 2.8 & -1.2 \\ 2.7 & 0.0 & -2.0 & -1.4 & 0.0 & 1.6 \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} + \begin{bmatrix} -4.4 \\ -3.2 \\ 1.9 \\ -1.6 \\ 0.4 \\ -0.4 \\ -1.2 \\ 1.5 \\ 3.2 \\ 4.0 \end{bmatrix}\right) = logsig\left(\begin{bmatrix} -5.8 \\ -9.6 \\ 5.3 \\ 0.1 \\ 1.4 \\ -3.2 \\ -2.3 \\ 2.9 \\ 4.7 \\ 4.9 \end{bmatrix}\right)$$

$$= \begin{bmatrix} \frac{1}{1+e^{5.8}} \\ \frac{1}{1+e^{9.6}} \\ \frac{1}{1+e^{-5.3}} \\ \frac{1}{1+e^{-0.1}} \\ \frac{1}{1+e^{-1.4}} \\ \frac{1}{1+e^{3.2}} \\ \frac{1}{1+e^{2.3}} \\ \frac{1}{1+e^{-2.9}} \\ \frac{1}{1+e^{-4.7}} \\ \frac{1}{1+e^{-4.9}} \end{bmatrix} = \begin{bmatrix} 0.00301 \\ 0.00006 \\ 0.99503 \\ 0.52497 \\ 0.80218 \\ 0.03916 \\ 0.09112 \\ 0.94784 \\ 0.99098 \\ 0.99260 \end{bmatrix}$$

In output layer, output matrix of hidden layer is the input of output layer. In equation 4.4 and 4.5, initial weights of weights and bias matrix of output layer can be seen which is set randomly before training.

$$\begin{bmatrix} 0.36 & 0.11 & 0.29 & 0.40 & -2.35 & 0.21 & 0.54 & 1.01 & 0.13 & 0.18 \\ 0.77 & -0.30 & 0.70 & 0.02 & -0.69 & -0.30 & -1.60 & 0.16 & 0.32 & 0.01 \end{bmatrix} \quad (4.4)$$

$$\begin{bmatrix} -0.17 \\ 0.09 \end{bmatrix} \quad (4.5)$$

Output matrix of hidden layer is multiplied with weight matrix of output layer, then added with bias matrix. Output layer neuron's transfer function is purelin. Purelin function just forwards the result.

$$a_{output} = f(W_{output} \times a_{hidden} + b_{output}) \quad (4.6)$$

$$a_{output} = purelin\left(\begin{bmatrix} 0.36 & 0.11 & 0.29 & 0.40 & -2.35 & 0.21 & 0.54 & 1.01 & 0.13 & 0.18 \\ 0.77 & -0.30 & 0.70 & 0.02 & -0.69 & -0.30 & -1.60 & 0.16 & 0.32 & 0.01 \end{bmatrix} \times \begin{bmatrix} 0.00301 \\ 0.00006 \\ 0.99503 \\ 0.52497 \\ 0.80218 \\ 0.03916 \\ 0.09112 \\ 0.94784 \\ 0.99098 \\ 0.99260 \end{bmatrix} + \begin{bmatrix} -0.17 \\ 0.09 \end{bmatrix}\right) = purelin\left(\begin{bmatrix} -0.23 \\ 0.48 \end{bmatrix}\right) = \begin{bmatrix} -0.23 \\ 0.48 \end{bmatrix}$$

In this way, rest of the data points corresponding output is calculated. Then MSE method is applied to calculate total mean square error of the dataset. Let there are Q no of data points in dataset.

$$e = \frac{1}{Q}\sum_{q=1}^{Q}(t_q - a_q)^T(t_q - a_q) \tag{4.7}$$

Assume, mse error is 0.67 (probability). Total gradient of the mean square error is the mean of the gradients of the individual data point's squared errors. Individual data point's square error is,

$$\hat{F} = (t-a)^T(t-a) \tag{4.8}$$

Assume, first data point's square error is 0.6 (probability).

For back propagation we need to calculate sensitivities. For this chain rule needs to be applied. Sensitivities of layer m is calculated from sensitivities of layer m+1. The sensitivities are propagated backward through the network from the last layer to the first layer.

$$S^M \rightarrow S^{M-1} \rightarrow \cdots \rightarrow S^2 \rightarrow S^1$$

Sensitivities for all layers for all data point must be calculated, when outputs for all data points are calculated .There are two layers for this network. So, Maximum value of M is two. Now, for layer one, if there are ten points in dataset, there will be ten S^1 for each individual data point. At the same time there will be ten $S^2$for layer two. Now, here derivation and calculation of

sensitivities for all layers are shown for first data point. For deriving relationship of sensitivities for a particular data point, we will use following Jacobian matrix which is actually the output matrix of a layer.

$$\frac{\partial \mathbf{n}^{m+1}}{\partial \mathbf{n}^{m}} \equiv \begin{bmatrix} \frac{\partial n_1^{m+1}}{\partial n_1^m} & \frac{\partial n_1^{m+1}}{\partial n_2^m} & \cdots & \frac{\partial n_1^{m+1}}{\partial n_{S^m}^m} \\ \frac{\partial n_2^{m+1}}{\partial n_1^m} & \frac{\partial n_2^{m+1}}{\partial n_2^m} & \cdots & \frac{\partial n_2^{m+1}}{\partial n_{S^m}^m} \\ \vdots & \vdots & & \vdots \\ \frac{\partial n_{S^{m+1}}^{m+1}}{\partial n_1^m} & \frac{\partial n_{S^{m+1}}^{m+1}}{\partial n_2^m} & \cdots & \frac{\partial n_{S^{m+1}}^{m+1}}{\partial n_{S^m}^m} \end{bmatrix}. \tag{4.9}$$

Here, m representing a layer's particular no.

An expression for this matrix needs to be found, i and j are the elements of the matrix,

$$\frac{\partial n_i^{m+1}}{\partial n_j^m} = \frac{\partial\left(\sum_{l=1}^{S^m} w_{i,l}^{m+1} a_l^m + b_i^{m+1}\right)}{\partial n_j^m} = w_{i,j}^{m+1} \frac{\partial a_j^m}{\partial n_j^m}$$

$$= w_{i,j}^{m+1} \frac{\partial f^m(n_j^m)}{\partial n_j^m} = w_{i,j}^{m+1} \dot{f}^m(n_j^m), \tag{4.10}$$

Where,

$$\dot{f}^m(n_j^m) = \frac{\partial f^m(n_j^m)}{\partial n_j^m} \tag{4.11}$$

Jacobian matrix can be written as,

$$\frac{\partial n^{m+1}}{\partial n^m} = W^{m+1} \begin{bmatrix} \dot{f}^m(n_1^m) & 0 & \cdots & 0 \\ 0 & \dot{f}^m(n_2^m) & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & \dot{f}^m(n_{S^m}^m) \end{bmatrix} \tag{4.12}$$

Now, sensitivity of final layer S^2 is,

$$S_i^2 = \frac{\partial \hat{F}}{\partial n_i^2} = \frac{\partial}{\partial n_i^2}\left((t-a)^T(t-a)\right) = -2(t_i - a_i)\frac{\partial a_i}{\partial n_i^2} = -2(t_i - a_i)\dot{f}^2(n_i^2) \tag{4.13}$$

This can be written in matrix form as,

$$S^2 = -2(t-a)\dot{F}^2(n^2) \tag{4.14}$$

Now, sensitivity of hidden layer S^1 is,

$$S^1 = \frac{\partial \hat{F}}{\partial n^1} = \left(\frac{\partial n^2}{\partial n^1}\right)^T \frac{\partial \hat{F}}{\partial n^2} = \begin{bmatrix} \dot{f}^1(n_1^1) & 0 & \cdots & 0 \\ 0 & \dot{f}^1(n_2^1) & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & \dot{f}^1(n_{S^1}^1) \end{bmatrix} (W^2)^T S^2 \tag{4.15}$$

Outer layer's transfer function is purelin. So,

$$\dot{f}^2(n) = \frac{d}{dn}(n) = 1 \tag{4.16}$$

$$\dot{F}^2(n^2) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$S^2 = -2\left(t - a_{output}\right)\dot{F}^2(n^2) = -2 * 0.6 * \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} -1.2 \\ 0 \end{bmatrix}$$

Hidden layer's transfer function is logsig. So,

$$\dot{f}^1(n) = \frac{d}{dn}\left(\frac{1}{1+e^{-n}}\right) = \frac{e^{-n}}{(1+e^{-n})^2} = \left(1 - \frac{1}{1+e^{-n}}\right)\left(\frac{1}{1+e^{-n}}\right) = (1-a^1)a^1 \tag{4.17}$$

$$S^1 = \begin{bmatrix} \dot{f}^1(n_1^1) & 0 & \cdots & 0 \\ 0 & \dot{f}^1(n_2^1) & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & \dot{f}^1(n_{S^1}^1) \end{bmatrix} (W^2)^T S^2 = \begin{bmatrix} (1-a_1^1)a_1^1 & 0 & \cdots & 0 \\ 0 & (1-a_2^1)a_2^1 & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & (1-a_{10}^1)a_{10}^1 \end{bmatrix} (W^2)^T S^2$$

$$= \begin{bmatrix} 0.003001 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.000059 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.99503 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.00495 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.15869 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.037626 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.08282 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.04944 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.00894 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.00894 \end{bmatrix}$$

$$\begin{bmatrix} 0.36 & 0.77 \\ 0.11 & -0.30 \\ 0.29 & 0.70 \\ 0.40 & 0.02 \\ -2.35 & -0.69 \\ 0.21 & -0.30 \\ 0.54 & -1.60 \\ 1.01 & 0.16 \\ 0.13 & 0.32 \\ 0.18 & 0.01 \end{bmatrix} \begin{bmatrix} -1.2 \\ 0 \end{bmatrix} = \begin{bmatrix} 0.00108036 & 0.00231077 \\ 0.00000649 & -0.0000177 \\ 0.2885586 & 0.696521 \\ 0.0019800 & 0.000099 \\ -0.3729215 & 0.1094960 \\ 0.0079014 & -0.01128779 \\ 0.04472280 & -0.1325120 \\ 0.0499344 & 0.0079104 \\ 0.0011622 & 0.0028608 \\ 0.00160919 & 0.0000894 \end{bmatrix} \begin{bmatrix} -1.2 \\ 0 \end{bmatrix} = \begin{pmatrix} -0.001296 \\ -0.000007 \\ -0.346270 \\ -0.002376 \\ 0.447505 \\ -0.009481 \\ -0.053667 \\ -0.059921 \\ -0.001394 \\ -0.001931 \end{pmatrix}$$

In this way, sensitivity for two layers for all data points will be calculated. Then weight and bias matrix of every layer is updated from output layer to layer upper.

$$\mathbf{W}^m(k+1) = \mathbf{W}^m(k) - \frac{\alpha}{Q}\sum_{q=1}^{Q} \mathbf{s}_q^m(\mathbf{a}_q^{m-1})^T, \tag{4.18}$$

$$\mathbf{b}^m(k+1) = \mathbf{b}^m(k) - \frac{\alpha}{Q}\sum_{q=1}^{Q} \mathbf{s}_q^m. \tag{4.19}$$

This way one epoch is completed. Updating weights and biases are continued, until mse is less than 0.001 or 8000 epochs are completed. If validation checks satisfy the goal, then the procedure is stopped.

For making the weight more accurate and for removing fractional part, we use following condition,

$$new(a_{output} > 0.5) = 1;$$
$$new(a_{output} <= 0.5) = 0;$$

Later, this binary value is converted to decimal value.

4.1.2 Levenberg-Marquardt Back Propagation method

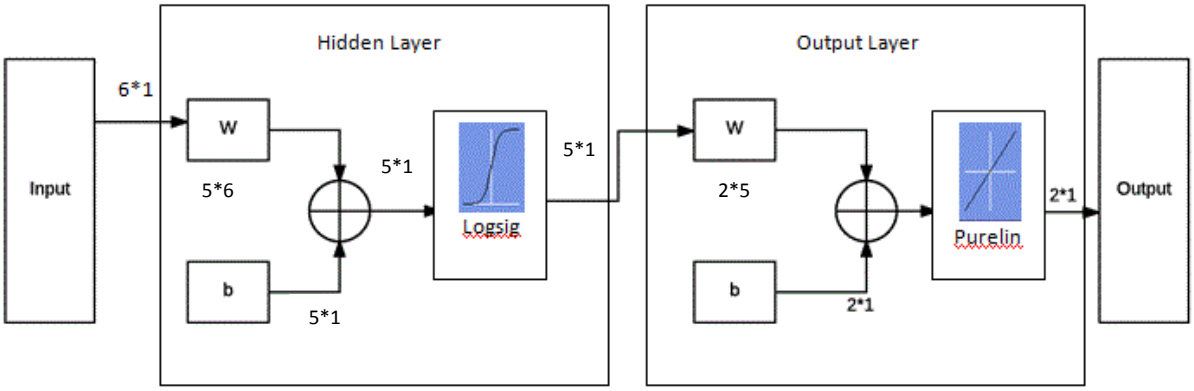

Figure 4.2: Neural Network with LM method

The Levenberg-Marquardt backpropagation algorithm can be described as follows,

In equation 4.20, 4.21, 4.22 and 4.23 initial weights of weights and bias matrix of hidden layer and output layer can be seen respectively which are set randomly before training.

$$\begin{bmatrix} 0.5947 & -0.0208 & -0.0458 & 0.1334 & -12.2995 & 8.9070 \\ -7.1442 & -0.2859 & 0.0297 & 0.1391 & 4.3932 & 3.0632 \\ -1.4037 & -0.4677 & -0.9799 & 7.4351 & 4.3747 & -0.1389 \\ -0.6449 & 0.0579 & 0.0623 & -0.1469 & 13.2747 & 1.7449 \\ -5.2820 & 0.2455 & -0.0228 & -0.1112 & -26.3978 & 2.6347 \end{bmatrix} \tag{4.20}$$

$$\begin{bmatrix} -5.4887 \\ 10.4484 \\ -2.6328 \\ -5.2890 \\ 15.0875 \end{bmatrix} \tag{4.21}$$

$$\begin{bmatrix} 2.1163 & 0.2434 & -0.0004 & 2.0138 & -0.1070 \\ 0.0046 & -2.0173 & -0.0042 & -0.0040 & -2.0109 \end{bmatrix} \tag{4.22}$$

$$\begin{bmatrix} -1.2552 \\ 3.0213 \end{bmatrix} \tag{4.23}$$

For, every data points in network calculate corresponding output.

$$a^q_{output} = purelin\left(w_{output} logsig(w_{hidden} * p^q + b_{hidden}) + b_{output}\right) \tag{4.24}$$

Then, the mean square error for all inputs is calculated.

$$e = \frac{1}{Q}\sum_{q=1}^{Q}(t_q - a_q)^T(t_q - a_q) \tag{4.25}$$

Marquardt Sensitivities for each layer are calculated for each data points. There are two layers in this network. At first for outer layer sensitivity is computed for each data point,

$$\tilde{S}^2_1 = -\dot{F}^2(n^2_1) \tag{4.26}$$

$$\tilde{S}^2 = \left[\tilde{S}^2_1 \middle| \tilde{S}^2_2 \middle| \tilde{S}^2_3 \middle| \cdots \middle| \tilde{S}^2_Q\right] \tag{4.27}$$

Then, for hidden layer sensitivity is computed,

$$\tilde{S}^1_1 = \dot{F}^1(n^1_1)(W^2)^T\tilde{S}^2_1 \tag{4.28}$$

$$\tilde{S}^1 = \left[\tilde{S}^1_1 \middle| \tilde{S}^1_2 \middle| \tilde{S}^1_3 \middle| \cdots \middle| \tilde{S}^1_Q\right] \tag{4.29}$$

Now, Jacobian matrix can be computed.

$$J(x) = \begin{bmatrix} \frac{\partial e_{1,1}}{\partial w^1_{1,1}} & \frac{\partial e_{1,1}}{\partial b^1_1} & \frac{\partial e_{1,1}}{\partial w^2_{1,1}} & \frac{\partial e_{1,1}}{\partial b^2_1} \\ \frac{\partial e_{1,2}}{\partial w^1_{1,1}} & \frac{\partial e_{1,2}}{\partial b^1_1} & \frac{\partial e_{1,2}}{\partial w^2_{1,1}} & \frac{\partial e_{1,2}}{\partial b^2_1} \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} \tag{4.30}$$

$$[J]_{1,1} = \frac{\partial e_{1,1}}{\partial w^1_{1,1}} = \frac{\partial e_{1,1}}{\partial n^1_{1,1}} \times \frac{\partial n^1_{1,1}}{\partial w^1_{1,1}} = \tilde{S}^1_{1,1} \times \frac{\partial n^1_{1,1}}{\partial w^1_{1,1}} = \tilde{S}^1_{1,1} \times P^1 \tag{4.31}$$

$$[J]_{1,2} = \frac{\partial e_{1,1}}{\partial b_1^1} = \frac{\partial e_{1,1}}{\partial n_{1,1}^1} \times \frac{\partial n_{1,1}^1}{\partial b_1^1} = \tilde{S}_{1,1}^1 \times \frac{\partial n_{1,1}^1}{\partial b_1^1} = \tilde{S}_{1,1}^1 \tag{4.32}$$

$$[J]_{1,3} = \frac{\partial e_{1,1}}{\partial w_{1,1}^2} = \frac{\partial e_{1,1}}{\partial n_{1,1}^2} \times \frac{\partial n_{1,1}^2}{\partial w_{1,1}^2} = \tilde{S}_{1,1}^2 \times \frac{\partial n_{1,1}^2}{\partial w_{1,1}^2} = \tilde{S}_{1,1}^2 \times a_{1,1}^1 \tag{4.33}$$

$$[J]_{1,4} = \frac{\partial e_{1,1}}{\partial b_1^2} = \frac{\partial e_{1,1}}{\partial n_{1,1}^2} \times \frac{\partial n_{1,1}^2}{\partial b_1^2} = \tilde{S}_{1,1}^2 \times \frac{\partial n_{1,1}^2}{\partial b_1^2} = \tilde{S}_{1,1}^2 \tag{4.34}$$

$$[J]_{2,1} = \frac{\partial e_{1,2}}{\partial w_{1,1}^1} = \frac{\partial e_{1,2}}{\partial n_{1,2}^1} \times \frac{\partial n_{1,2}^1}{\partial w_{1,1}^1} = \tilde{S}_{1,2}^1 \times \frac{\partial n_{1,2}^1}{\partial w_{1,1}^1} = \tilde{S}_{1,2}^1 \times P^2 \tag{4.35}$$

$$[J]_{2,2} = \frac{\partial e_{1,2}}{\partial b_1^1} = \frac{\partial e_{1,2}}{\partial n_{1,2}^1} \times \frac{\partial n_{1,2}^1}{\partial b_1^1} = \tilde{S}_{1,2}^1 \times \frac{\partial n_{1,2}^1}{\partial b_1^1} = \tilde{S}_{1,2}^1 \tag{4.36}$$

$$[J]_{2,3} = \frac{\partial e_{1,2}}{\partial w_{1,1}^2} = \frac{\partial e_{1,2}}{\partial n_{1,2}^2} \times \frac{\partial n_{1,2}^2}{\partial w_{1,1}^2} = \tilde{S}_{1,2}^2 \times \frac{\partial n_{1,2}^2}{\partial w_{1,1}^2} = \tilde{S}_{1,2}^2 \times a_{1,2}^1 \tag{4.37}$$

$$[J]_{2,4} = \frac{\partial e_{1,2}}{\partial b_1^2} = \frac{\partial e_{1,2}}{\partial n_{1,2}^2} \times \frac{\partial n_{1,2}^2}{\partial b_1^2} = \tilde{S}_{1,2}^2 \times \frac{\partial n_{1,2}^2}{\partial b_1^2} = \tilde{S}_{1,2}^2 \tag{4.38}$$

$$\vdots$$

$$\vdots$$

Then, Δx_k is calculated, which is used to update weights of weight and bias matrix of each layer.

$$\Delta \mathbf{x}_k = -[\mathbf{J}^T(\mathbf{x}_k)\mathbf{J}(\mathbf{x}_k) + \mu_k \mathbf{I}]^{-1} \mathbf{J}^T(\mathbf{x}_k)\mathbf{v}(\mathbf{x}_k) \tag{4.39}$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k - [\mathbf{J}^T(\mathbf{x}_k)\mathbf{J}(\mathbf{x}_k) + \mu_k \mathbf{I}]^{-1} \mathbf{J}^T(\mathbf{x}_k)\mathbf{v}(\mathbf{x}_k) \tag{4.40}$$

μ sets to small value at the beginning of the algorithm. The mean square errors using $x_k + \Delta x_k$ are recalculated. If new mean square error is smaller than old one, then divide μ by some factor $\vartheta > 1$, and let $x_{k+1} = x_k + \Delta x_k$ and then do the same procedure again. If new mean square error is larger than old one, then μ is multiplied by ϑ and Δx_k is recalculated. Then mean square errors are recalculated for comparison with old one as a result which will occur multiplication or division of μ by ϑ can be decided.

4.2 Closed loop NARX Time Delay Neural Network

Before predicting road weights of different road, we need to know weights of all features (temperature, rainfall, road status). Now the future weights of these features need to be predicted. For this closed loop time delay neural network is used to predict future weights of these features. For three individual features. Now for predicting weights of temperature and rainfall, a NARX time delay network is used. This network has 20 neurons in hidden layer. A separate NARX time delay network is used to predict weights for road status. Now for predicting weights for and rainfall, seven features are used. Those are day, simple moving mean

average for temperature, oscillator [31] for temperature, rate of change for temperature, simple moving mean average for rainfall, oscillator for rainfall and rate of change for rainfall.

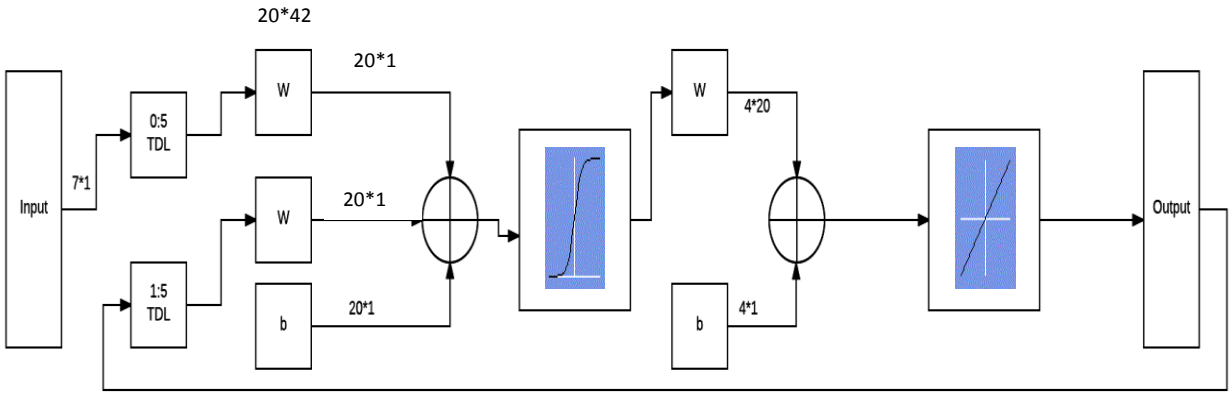

Figure 4.3: Closed loop NARX neural network.

Moving Average: It is calculated progressively as an average of N number data values over the certain period.

$$S_{n+1} = \sum_{i=1}^{n} \frac{T_i}{n} \tag{4.41}$$

Now, in table 4.3, we can see the simple moving mean average for temperature and real temperatures [32] of a few days. Here, period size is 15.

Table 4.3: Simple mean average for Temperature

Date	Temperature(℃)	Ma15
12/3/2016	18	20
12/4/2016	20	19.33333
12/5/2016	20	19.5
12/6/2016	21	19.6
12/7/2016	22	19.83333
12/8/2016	21	20.14286
12/9/2016	22	20.25
12/10/2016	22	20.44444
12/11/2016	22	20.6
12/12/2016	23	20.72727
12/13/2016	19	20.91667
12/14/2016	18	20.76923
12/15/2016	20	20.57143
12/16/2016	20	20.53333
12/17/2016	22	20.53333

12/18/2016	20	20.66667
12/19/2016	22	20.8
12/20/2016	22	20.93333
12/21/2016	22	21.06667
12/22/2016	22	21.13333
12/23/2016	22	21.13333
12/24/2016	22	21.2
12/25/2016	20	21.2

Here, mean average [17] for temperature of a day is average temperature of last fifteen days.

$$S_{16}^{Temperature} = \sum_{i=1}^{15} \frac{T_i^{Temperature}}{15} \tag{4.42}$$

In the same way mean average for rainfall is calculated

Oscillator: It is used to define the rising and falling trend available in the time series. It is defined as difference of moving averages of different time periods.

$$OSC = MA_{N1} - MA_{N2} \tag{4.43}$$

Where, N1 and N2 are different periods and N1>N2.

Table 4.4 Oscillator for Temperature

DATE	Ma15	Ma30	OSC=Ma30-Ma15
1/1/2007	20.4	20.46667	0.066667
1/2/2007	20.13333	20.4	0.266667
1/3/2007	19.93333	20.36667	0.433333
1/4/2007	19.6	20.26667	0.666667
1/5/2007	19.06667	20.06667	1
1/6/2007	18.4	19.76667	1.366667
1/7/2007	18	19.56667	1.566667
1/8/2007	17.4	19.3	1.9
1/9/2007	17.06667	19.13333	2.066667
1/10/2007	16.8	18.93333	2.133333
1/11/2007	16.53333	18.73333	2.2
1/12/2007	16.53333	18.56667	2.033333
1/13/2007	16.06667	18.36667	2.3
1/14/2007	15.86667	18.26667	2.4
1/15/2007	15.86667	18.16667	2.3

In the same way, oscillator for rainfall is calculated.

Rate of change: ROC is the rate of change of data.

$$ROC = \frac{D_N - D_{t-N}}{N} \tag{4.44}$$

Now, weights of simple mean average for rainfall, oscillator for rainfall and rate of change for rainfall are calculated in same way.

Table 4.5: Input and output of first NARX Network

Input							Output	
Day	Ma15 (Temperature)	OSC (Temperature)	ROC (Temperature)	Ma15 (Rainfall)	OSC (Rainfall)	ROC (Rainfall)	Temperature (Weighted)	Rainfall (Weighted)
1	20.4	0.066667	-0.03333	0	0	0	0	0
2	20.13333	0.266667	-0.06667	0	0	0	0	0
3	19.93333	0.433333	-0.03333	0	0	0	0	0
4	19.6	0.666667	-0.1	0	0	0	0	0
5	19.06667	1	-0.2	0	0	0	0	0
6	18.4	1.366667	-0.3	0	0	0	0	0
7	18	1.566667	-0.2	0	0	0	0	0
8	17.4	1.9	-0.26667	0	0	0	0	0
9	17.06667	2.066667	-0.16667	0	0	0	0	0
10	16.8	2.133333	-0.2	0	0	0	0	0
11	16.53333	2.2	-0.2	0	0	0	0	0
12	16.53333	2.033333	-0.16667	0	0	0	0	0
13	16.06667	2.3	-0.2	0	0	0	0	0
14	15.86667	2.4	-0.1	0	0	0	0	0
15	15.86667	2.3	-0.1	0	0	0	0	0

Outputs (temperature and rainfall) of the network will be in binary form. Later, which is converted to decimal form.

Here, as input weights of seven features is got into the network as 7*1 matrix. For training the network at first open loop is used, that means past predicted output does not need to circle back as input in the network. Original past output is used as input. This is done in training because at the beginning the result produced by network is not accurate, so if we use this faulted result as input, it may take a lot of time to train the network. Now, there are two buffers called time delay line (one for input and one for output).

Table 4.6: Inputs of present and last five day's input in input buffer for getting output of sixth day

Day	Ma15 (Temperature)	OSC (Temperature)	ROC (Temperature)	Ma15 (Rainfall)	OSC (Rainfall)	ROC (Rainfall)
1	20.4	0.066667	-0.03333	0	0	0
2	20.13333	0.266667	-0.06667	0	0	0
3	19.93333	0.433333	-0.03333	0	0	0
4	19.6	0.666667	-0.1	0	0	0
5	19.06667	1	-0.2	0	0	0
6	18.4	1.366667	-0.3	0	0	0

The input buffer contains present input data as well as last five day's input data. The output buffer contains the last five day's output data. In the input buffer all data form a matrix which size is 42*1.

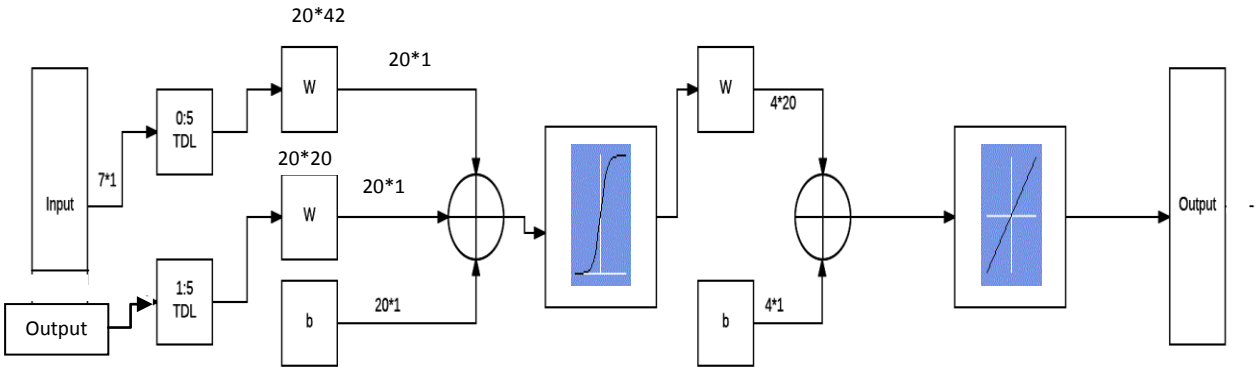

Figure 4.4: Open loop NARX neural network.

Table 4.7: Outputs of previous five days in output buffer for getting output of sixth day

	Output (In Output Buffer)	
Day	Temperature (Weighted)	Rainfall (Weighted)
1	0	0
2	0	0
3	0	0
4	0	0
5	0	0

In the output buffer, all present outputs form a matrix which size is 20*1. Now, in the hidden layer there are two weight matrices- W_1 (20*42) and W_2 (20*20) and a bias matrix b_1 (20*1). W_1 is multiplied with input buffer matrix and W_2 is multiplied with output buffer matrix, than

the result is added with b_1. Hidden layer transfer function is logsig (which is differentiable). So logistic function is used on this intermediate result to get the output of hidden layer,

$$a_{hidden} = logsig(W_1 \times P_{input\ buffer} + W_2 \times P_{output\ buffer} + b_1) \quad (4.45)$$

Output of hidden layer is 20*1 matrix. Output layer's weight matrix is W_{output} (4*20) and bias matrix is b_{output}. Output of hidden layer is multiplied with $W_{output}$and the result is added with b_{output}. Output layer transfer function is purelin. Purelin function is used on this intermediate result to get the output of output layer.

$$a_{Output} = Purelin(W_{output} \times a_{hidden} + b_{output}) \quad (4.46)$$

Now, when an output is calculated for a data point, the most past input is omitted from input buffer and next input gets into the input buffer. In the same way, in output buffer, the most past output is omitted from output buffer and next output gets into the output buffer.

Table 4.8: Inputs of present and last five day's input in input buffer for getting output of seventh day

Day	Ma15 (Temperature)	OSC (Temperature)	ROC (Temperature)	Ma15 (Rainfall)	OSC (Rainfall)	ROC (Rainfall)
2	20.13333	0.266667	-0.06667	0	0	0
3	19.93333	0.433333	-0.03333	0	0	0
4	19.6	0.666667	-0.1	0	0	0
5	19.06667	1	-0.2	0	0	0
6	18.4	1.366667	-0.3	0	0	0
7	18	1.566667	-0.2	0	0	0

Table 4.9: Outputs of previous five days in output buffer for getting output of seventh day

	Output (In Output Buffer)	
Day	Temperature (Weighted)	Rainfall (Weighted)
2	0	0
3	0	0
4	0	0
5	0	0
6	0	0

Then, in the same way, that data point's output is calculated. Now, in training the network, Levenberg-Marquardt back propagation algorithm is used to update weights and biases of hidden and output layers.

When training is over, network is converted into closed loop network. As a result, predicted output can now multiplied with weight matrix W_2, instead of original output, during using the network.

The NARX network for road status is trained and later used in same way.

Table 4.10: Input and output of NARX Network for road status

		Input				Output
Date	Weekday	Time slot	Ma24	OSC	ROC	Road Status
1/1/2007	Monday	8	10.54166667	-1.27083	1.708333	2
1/1/2007	Monday	5	10.04166667	-0.4375	-2	3
1/1/2007	Monday	4	10	-0.08333	1.625	2
1/1/2007	Monday	7	10	0.083333	-1.875	3
1/2/2007	Tuesday	8	9.666666667	0.4375	1.875	2
1/2/2007	Tuesday	5	10.125	-0.04167	-1.5	3
1/2/2007	Tuesday	4	10.16666667	-0.08333	1.666667	2
1/2/2007	Tuesday	7	10.29166667	-0.14583	-1.41667	3
1/3/2007	Wednesday	8	10.29166667	-0.3125	1.416667	2
1/3/2007	Wednesday	5	10.41666667	-0.14583	-1.41667	3
1/3/2007	Wednesday	4	10.625	-0.22917	1.75	2
1/3/2007	Wednesday	7	10.83333333	-0.27083	-1.20833	3
1/4/2007	Thursday	8	10.79166667	-0.25	1.25	2
1/4/2007	Thursday	5	10.29166667	0.0625	-2.125	3
1/4/2007	Thursday	4	10.375	0.125	1.625	2
1/4/2007	Thursday	7	10.375	0.229167	-1.16667	3
1/5/2007	Friday	8	10.54166667	0.125	1.916667	0
1/5/2007	Friday	5	9.583333333	0.354167	-3.16667	2
1/5/2007	Friday	4	7.083333333	1.645833	-0.875	0
1/5/2007	Friday	7	5.375	2.5	-3.04167	3
1/6/2007	Saturday	8	4.5	3.020833	2	1
1/6/2007	Saturday	5	4.666666667	2.458333	-0.5	3
1/6/2007	Saturday	4	6.166666667	0.458333	2.333333	2
1/6/2007	Saturday	7	7.166666667	-0.89583	-1.16667	3
1/7/2007	Sunday	8	7.833333333	-1.66667	2.5	2
1/7/2007	Sunday	5	9	-2.16667	-0.83333	3
1/7/2007	Sunday	4	9.666666667	-1.75	2	2
1/7/2007	Sunday	7	10	-1.41667	-1.5	3

In table 4:10, we can see the sample of input and output of closed loop narx network for road status. In all network, we use 70% data for training, 15% data for validation and for 15% data for testing.

For dividing data randomly, we use dividerand function. For cross Validation [22], K-Fold cross validation is used. In K-Fold cross validation [40], total data set is divided in K subsets. Than K-1 subsets are used for training, remaining validation sets are used for testing. This process runs k times for different validation sets. Than, the true error is calculated through the average error on the validation sets.

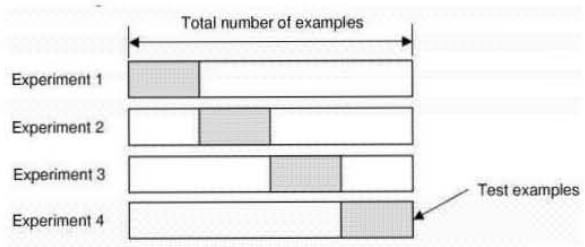

Figure 4.5: K subsets for K-Fold cross validation.

Chapter 5

Experimental Results and their Analysis

5.1 Results of GD and LM based Neural Network

For training and testing network, six attributes are considered. Those are rainfall, temperature, humidity, wind, road maintenance and road status. We can change the number of attributes according to system requirements. DT implementation system has two decision classes (Yes/ NO). There are four classes (three to zero) in neural network based system. Decision tree is not good for online learning because data is coming continuously and model needs to be updated. Data may include some exceptional situation which will force tree to be fall apart and need to be constructed again. By just changing the weight values, neural network is capable of reflecting the information of new instance on a model very efficiently. In all neural network, determining optimal neurons in hidden layer is a tricky part. We start with one neuron in the hidden layer and test the performance of the neural network on the basis of a fixed test set, and then we increase the number of neurons until we achieve maximum accuracy. When the maximum accuracy is found, we select that neuron number as optimal neuron in hidden layer. In GD method no of optimal neurons are ten and in LM method no of optimal neurons are five.

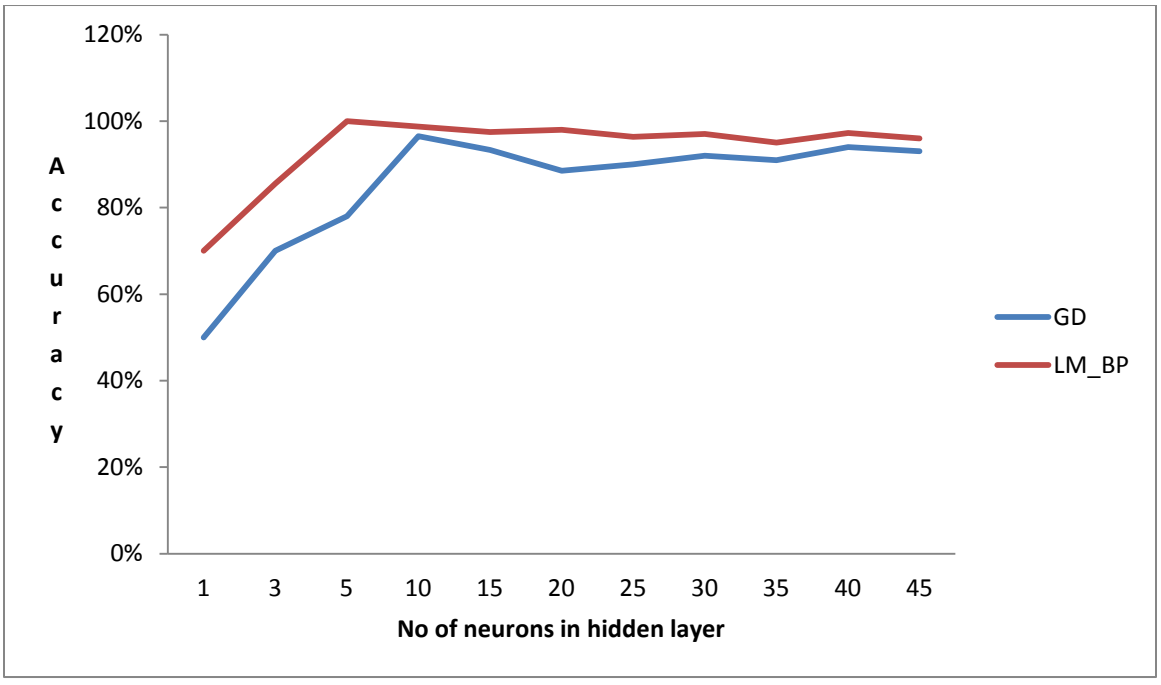

Figure 5.1: Selection # of Neuron in Hidden Layer

In GD Based neural network, we have achieved 96.8% accuracy and in LM based neural network, we have achieved 100% accuracy. We can see LM perform better than GD. We have

used total 1456 data points for training and testing in both methods. In fig 5.2 and fig 5.3, we can see the regression analysis of GD and LM methods respectively.

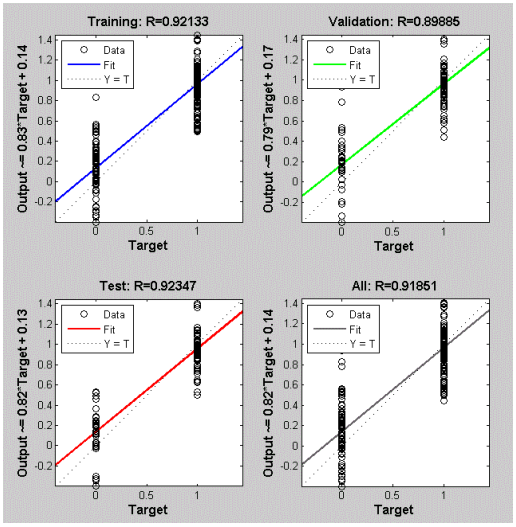

Figure 5.2: Regression analysis of Gradient Decent (GD) NN

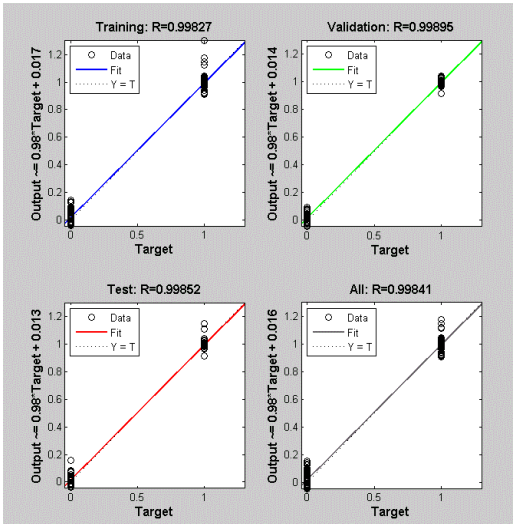

Figure 5.3: Regression analysis of Levenberg-Marquardt (LM) NN

For measuring the overall performance, the training process performs several times for analyzing the confidence interval of training, testing and validation. In table 5.1, we can see the training, testing, validation and overall performance of training algorithms.

Table 5.1: Comparison of regression analysis of NN training techniques (matlab)

Training Technique	Training	Testing	Validation	All
GD_BP_NN	0.92133	0.92347	0.89885	0.91851
LM_BP_NN	0.99827	0.99852	0.99895	0.99841

For measuring the true accuracy, k-fold cross validation [22] is used to estimate the true accuracy of the model. Thus LM based BP model has achieved 98%-100% accuracy.

Table 5.2: True error by K-Fold cross validation

Training Data set	Testing Data set	Average Error (Five repeatations)
1-972	973-1456	0%
485-1456	1-484	1.08%
1-487 and 972-1456	488-971	0%
Total average error		0.36%
Accuracy		99.64%

5.2 Results of NARX Implementation:

We have used closed loop NARX time delay networks to predict the weights of temperature, rainfall and road status. We have used 250 data points for training and 50 data points for testing for NARX network which is for temperature and rainfall. According to table 5.3, NARX network for temperature and rainfall achieves maximum accuracy, when time delay in input and output buffer are zero to five and one to five respectively and no of neurons are 15 in hidden layer.

Table 5.3: Accuracy of predicted temperature and rainfall for different time delay and different no of neurons in hidden layer.

Tapped Delay Input	Tapped Delay output	Number of Neurons in the hidden layer	Mse(btrain) (probability)	Mse(btest) (probability)	Mse(temp) (probability)	Mse(rain) (probability)	Mse(test) (probability)
0-5	1-5	3	0.2755	0.2400	-0.2200	0.7000	0.8200
		5	0.6041	0.5100	0.3200	0.7000	0.8750
		10	0.3163	0.3000	-0.0800	0.6800	0.7750
		15	0.6633	0.8100	0.9200	0.7000	0.9300
		20	0.7306	0.4500	0.9200	-0.0200	0.8300
		25	0.5837	0.6000	0.7200	0.4800	0.8550
0-7	1-7	3	0.7388	0.7200	0.7400	0.7000	0.9100
		5	0.6898	0.6800	0.6600	0.7000	0.8750
		10	0.5939	0.4800	0.3600	0.6000	0.8700
		15	0.6980	0.7700	0.8400	0.7000	0.9250
		20	0.3980	0.4300	0.2600	0.6000	0.8150
		25	0.2245	0.5600	0.7600	0.3600	0.8400
0-10	1-10	3	0.7184	0.7500	0.9200	0.5800	0.8800
		5	0.4510	-0.3800	-1.4600	0.7000	0.6850
		10	0.6041	0.5700	0.4400	0.7000	0.9000
		15	0.5469	0.5800	0.5800	0.5800	0.8500
		20	0.4367	0.3800	0.1200	0.6400	0.8550
		25	0.4041	-0.3500	-0.9200	0.2200	0.4700
0-15	1-15	3	0.4265	0.8100	0.9200	0.7000	0.9300
		5	0.3673	0.8100	0.9200	0.7000	0.9300
		10	0.7102	0.7800	0.9200	0.6400	0.8950

		15	0.5776	0.3300	0.1600	0.5000	0.7550
		20	0.5265	0.2600	0.7600	-0.2400	0.7650
		25	0.6837	0.4600	0.8400	0.0800	0.7950

The accuracy on testing data that we achieve is 81%. In fig 5.4 we can see predicted weights almost overlap on actual weights. There are some extreme condition, on those days predicted values fail to match with actual values.

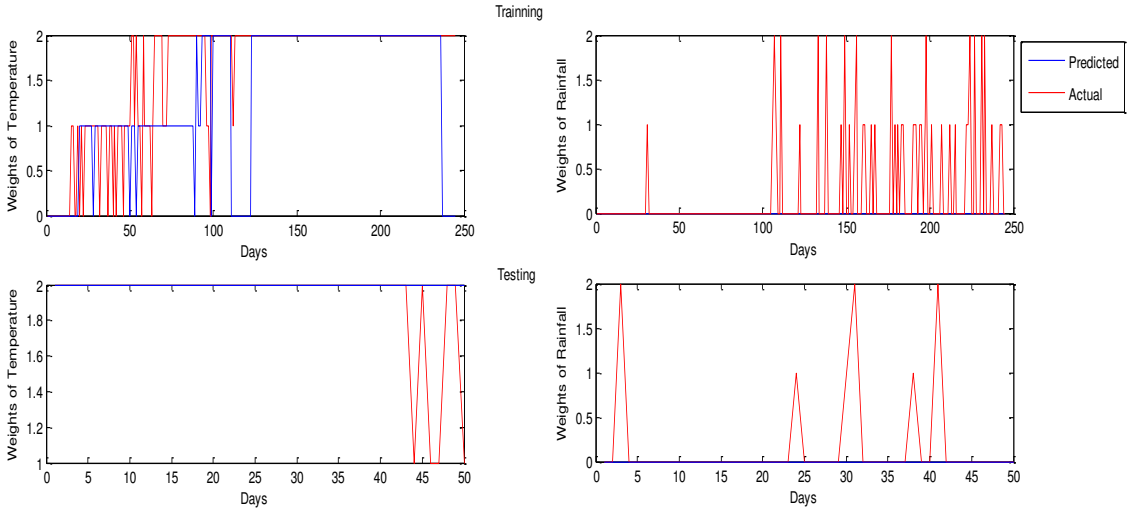

Figure 5.4: Actual and Predicted weights of temperature and rainfall

If we predict weights of temperature and rainfall by using weighted moving mean average algorithm, we will achieve 79.20% accuracy. NARX neural network perform better than weighted moving mean average algorithm.

Table 5.4: Accuracy of predicted road status for different time delay and different no of neurons in hidden layer

Tapped Delay Input	Tapped Delay Output	Number of Neurons in the hidden layer	Mse(btrain)	Mse(btest)	Mse(Rs)
0-5	1-5	3	0.9924	0.6200	0.9150
		5	0.5316	0.2300	0.7400
		10	-0.7646	-0.9400	0.5800
		15	-0.5468	-0.3600	0.6200
		20	-1.0430	-0.6300	0.6050
		25	0.1443	-0.1700	0.7600
0-7	1-7	3	0.1959	0.1500	0.7550
		5	0.2519	-0.2600	0.6950

		10	0.4529	0.4400	0.8450
		15	-1.5394	-2.1100	0.4550
		20	-0.8499	-0.7500	0.6150
		25	-1.0204	-1.3700	0.5700
0-10	1-10	3	0.3410	-0.1300	0.6600
		5	0.0333	-0.3300	0.6350
		10	-0.0769	-0.3800	0.5950
		15	-0.2282	0.0500	0.7400
		20	-0.4282	-0.4400	0.7300
		25	-0.1154	-0.3300	0.7350

According to table 5.4, NARX network for road statuse achieves maximum accuracy, when time delay in input and output buffer are zero to five and one to five respectively and no of neurons are three in hidden layer.

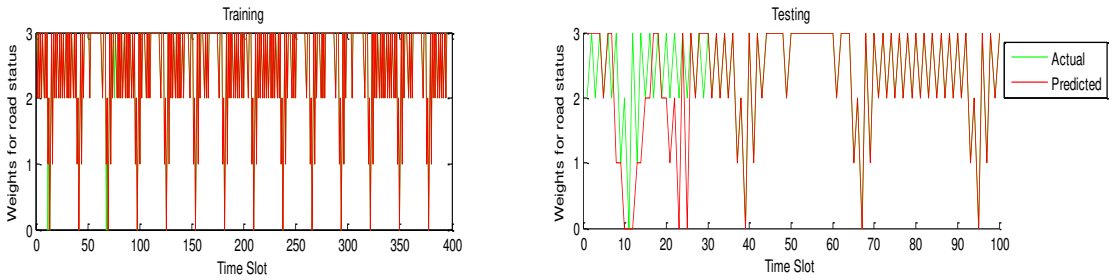

Figure 5.5: Actual and Predicted weights of road status.

In fig 5.4 we can see for training of the network, 400 data points have been used and for testing 100 data points have been used. We can also see predicted output almost cover the actual output. There are some extreme situation for that predicted output fail to match with actual output some time. For testing data we have achieved 62% accuracy. If we calculate weights by using weighted moving mean average algorithm, then we only achieve 16.45% accuracy. So, in all cases, NARX neural network is better than weighted moving mean average algorithm.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

We use neural network to predict the road weight. Two types of back propagation techniques (Gradient decent based back propagation and Levenberg-Marquardt based back propagation) are used to predict the road weight. LM back propagation is performing better than GD back propagation. NARX time delay neural network is used to predict feature's weights for road segment. We have succeeded to achieve better accuracy in predicting feature's weights.

6.2 Future Work

Genetic algorithm [21] can be integrated to the neural network to improve the performance. Dynamic web based system can be used as an altered static web based system.

REFERENCES

[1] "Traffic Pollution Kills Thousands," BBC News, 2000. Available at: http://news.bbc.co.uk/2/hi/health/905016.stm

[2] "Cutting Greenhouse Gas Emissions has Major Direct Health Benefits," The Lancet Medical Journal, December 2009. Available at: http://climateprogress.org/2009/12/01/the-lancetmedical-journal-cuttinggreenhouse-gas-emissions-hasmajor-direct-health-benefits.

[3] "Road Pollution Can Damage Kid's Lungs, Hearts," MSNBC Health, 2007. Available at: http://www.msnbc.msn.com/id/16831975/

[4] N.Verma, M. S. Sobhan and T. Jalil, "*Novel Design Proposal For Real Time Traffic Monitoring & Management of Dhaka Metropolitan City with (Rcap)*", Global Engineering, Science and Technology Conference, BIAM Foundation, Dhaka, 28-29 December 2012

[5] F. Aloul, A. Sagahyroon, A. Nahle, M. Abou Dehn and R. Al Anani , "*GuideME: An Effective RFID-Based Traffic Monitoring System*", Proc. of IASTED-International Conference on Advances in Computer Science and Engineering (ACSE), Phuket, Thailand, April 2-4, 2012.

[6] A. Dhar, "*Traffc and Road Condition Monitoring System*", M.Tech report, Indian Institute of Technology, Bombay, 2008.Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.518.2687&rep=rep1&type=pdf (Access Date: 15th May, 2015)

[7] T.Romão, L.Rato, P.Fernandes, N.Alexandre, A.Almada, and N. Capeta, "*M-Traffic - A Traffic Information and Monitoring System for Mobile Devices*", Proc. of International Workshop on Ubiquitous Computing (IWUC 2006), Pages 87-92, ICEIS Publisher, 2006.

[8] A. Angel, M. Hickman, P. Mirchandani, and D. Chandnani, "*Methods of Analyzing Traffic Imagery Collected From Aerial Platforms*" IEEE Transactions on Intelligent Transportation Systems, vol. 4, no. 2, Page 99-107 June 2003.

[9] HTML5 Web Sockets: *A Quantum Leap in Scalability for the Web*. Available at: http://www.websocket.org/quantum.html (Access Date: 10th May, 2015)

[10] M. R. Rahman and S.Akhter, "*Real Time Bi-directional Traffic Management Support System with GPS and WebSocket*" Proc. of the 15th IEEE International Conference on Computer and Information Technology (CIT-2015), Liverpool, UK, 26-28 Oct, 2015.

[11] M. R. Rahman and S. Akhter, "*Bi-directional traffic management support system with decision tree based dynamic routing*", Proc. of 10th International Conference for Internet Technology and Secured Transactions, ICITST 2015, London, United Kingdom, December 14-16, 2015.

[12] Global Positioning System. Available at: http://en.wikipedia.org/wiki/Global_Positioning_System (Access Date: 10th May, 2015)

[13] Google Maps. Available at: http://en.wikipedia.org/wiki/Google_Maps (Access Date: 10th May, 2015)

[14] Hagan, M.T., H.B. Demuth, and M.H. Beale, Neural Network Design, Boston, MA: PWS Publishing, 1996.

[15] Dijkstra's Algorithm. Available at: https://en.wikipedia.org/wiki/Dijkstra's_algorithm

(Access Date: 25th May, 2015)
[16] XML Path Language (XPath) at: https://www.w3.org/TR/xpath/ (Access Date: 6th March, 2016)
[17] Moving average at: https://en.wikipedia.org/wiki/Moving_average (Access Date: 6th March, 2016)
[18] AccuWeather at: http://www.accuweather.com/en/bd/dhaka/28143/january-weather/28143?monyr=1%2F1%2F2016&view=table (Access Date: 7th March, 2016)
[19] ID3 Decision Tree Algorithm - Part 1 at: http://www.codeproject.com/Articles/259241/ID-Decision-TreeAlgorithm-Part
[20] https://datajobs.com/data-science-repo/Decision-Trees-[Rokach-andMaimon].pdf
[21] D. E. Goldberg, *"Genetic Algorithms in Search, Optimization, and Machine Learning"*, 1989, 1st Edition, Addison-Wesley Professional, ISBN-13: 978-0201157673.
[22] https://en.wikipedia.org/wiki/Cross-validation_(statistics)
[23] M.T.Hagan, H. B. Demuth, M.H.Beale and O.D Jesús , "Book Neural Network Design", 2nd Edition,ISBN 987-0-9717321-1-7.
[24] M. Almishari, P. Gasti, N. Nathan, G. Tsudik," Optimizing BiDirectional Low-Latency Communication in Named Data Networking", ACM SIGCOMM Computer Communication Review, Volume 44, Number 1, January 2014.
[25] S.M. Rakhunde, "*Real Time Data Communication over Full Duplex Network Using Websocket*", IOSR Journal of Computer Science (IOSRJCE) e-ISSN: 2278-0661, p-ISSN: 2278-8727 PP 15-19
[26] S. Panagiotakis, K. Kapetanakis, A. G. Malamos, *Architecture for Real Time Communications over the Web*, International Journal of Web Engineering 2013, 2(1): 1-8, DOI: 10.5923/j.web.20130201.01
[27] J.-P. Erkkilä*, WebSocket Security Analysis*, Aalto University T-110.5291 Seminar on Network Security, Autumn 2012.
[28] http://www.newsbangladesh.com/english/details/6448
[29] http://www.itsinternational.com/categories/detection-monitoringmachine-vision/features/bluetooth-and-wi-fi-offer-new-options-fortravel-time-measurements/
[30] http://www.gmp.police.uk/content/section.html?readform&s=353EDD067D2055288025796100399283
[31] Shaminder Singh, Jasmeen Gill, "Temporal Weather Prediction using BackPropagation based Genetic Algorithm Technique", Published Online November 2014 in MECS, PP.55-61, DOI: 10.5815/ijisa.2014.12.08. (http://www.mecs-press.org/)
[32] https://www.wunderground.com/history/wmo/
[33] Yujing Yang, Junhai Ma, "The Deduction and Application of Climate Index Based on the L-M BP Neural Network Algorithm in Chinese Real Estate Market", E-ISSN: 2224-2856. 539.
[34] HenryA.Rowley, Shumeet Baluja, Takeo Kanade, "Neural Network-Based Face Detection", available at https://courses.cs.washington.edu/courses/.../rowley.**pdf**
[35] V .Preetha Pallavi, V .Vaithiyanathan, "Combined Artificial Neural Network and Genetic Algorithm for Cloud Classification", International Journal of Engineering and Technology (IJET) ISSN: 0975-4024
[36] Meera Narvekar, Priyanca Fargose, "Daily Weather Forecasting using Artificial Neural

Network", International Journal of Computer Applications (0975-8887)

[37] Arti R. Naik, Prof S.K.Pathan, "Weather Classification and Forecasting using Back Propagation Feed-forward Neural Network", International Journal of Scientific and Research Publications ISSN: 2250-3153.

[38] Pooja Malik, Prof Sranjeet Singh, Binni Arora, "An Effecting Weather Forecasting Using Neural Network", International Journal of Emerging Engineering Research and Technology, Volume 2, Issue 2, May 2014, PP 209-212

[39] Shaminder Singh, Pankaj Bhambri, Jasmeen Gill, "Time Series Based Temperature Prediction using Back Propagation with Genetic Algorithm Technique", IJCSI International Journal of Computer Science Issues, Vol.8, Issue 5, No 3, September 2011, ISSN (Online): 1694-0814 (www.IJCSI.org)

[40] Akhter, S, Rahman, M. R. & Islam, M. A. (2016). " Neural Network (NN) based Route Computation for Bi-directional Traffic Management System", Special Issue on Emerging Research Trend in Computing and Communication Technologies, International Journal of Applied Evolutionary Computation (IJAEC), Vol 7, Issue 4, IGI Global.

[41] Xiaoming Zheng, Sven Koenig ,"A Project on Gesture Recognition with Neural Networks for "Introduction to Artificial Intelligence" Classes", available at {xiaominz, skoenig}@usc.edu

[42] Zhang Minli , Qiao Shanshan, "Research on the Application of Artificial Neural Networks in Tender Offer for Construction Projects", Available online at www.sciencedirect.com

[43] Karan Kamdar, Amit Mathapati, "Artificial Neural Networks for Cancer Research in Prediction & Survival", Available at https://webfiles.uci.edu/.../project**work**/.../anncrips.pdf

[44] Vidushi Sharma, Sachin Rai, Anurag Dev, "A Comprehensive Study of Artificial Neural Networks", Available online at: www.ijarcsse.com, ISSN: 2277 128X

[45] Atiqul Islam, Shamim Akhter, Tumnun E. Mursalin, "Automated Textile Defect Recognition System Using Computer Vision and Artificial Neural Networks", ISSN 1307-6884

[46] Md. Atiqul Islam, Shamim Akhter, Tamnun E. Mursalin and M. Ashraful Amin, "A Suitable Neural Network to Detect Textile Defects" , Conference: Neural Information Processing, 13th International Conference, ICONIP 2006, Hong Kong, China, October 3-6, 2006, Proceedings, Part II, DOI: 10.1007/11893257_48 · Source: DBLP