Congestion avoidance by estimating proper round trip time (RTT) using neural networks

Mirza Waseem Hussain[1], Dr. Sanjay Jamwal[2], Dr. Majid Zaman[3]
[1] *Department of Computer Science,*
Baba Ghulam Shah Badshah University, Rajouri, J&K, India.
(E-mail: mirza.waseem.hussain@gmail.com)
[2] *Department of Computer Science,*
Baba Ghulam Shah Badshah University, Rajouri, J&K, India.
(E-mail: sanjayjamwal_2k6@rediffmail.com)
[3] *Directorate of Information Technology & Support System*
University of Kashmir, Hazratbal, Srinagar, J&K, India
(E-mail: zamanmajid@gmail.com)

***Abstract*—** This paper proposes a use of recurrent neural network model to solve the problem of congestion by estimating proper round trip time. Round trip time (RTT) is used as basic matric to foretell the value in a non-linear systems. The data is collected by an open source packer tracer software wire shark. Round trip data collected is then divided into training data using 70% of the data, test and evaluation which consist of 15% of data each using Matlab. The neural networks have been further optimized by changing the number of neurons and delay units in layers. Further this paper explores the algorithms to calculate RTT and its contrast to measured value of RTT

***Keywords*—** *RTT; RTO; MSS; TCP/IP; Congestion control; Recurrent neural networks;*

I. INTRODUCTION

Prediction of values is one of the common most practice in these days in engineering fields. Normally we use mean square error between the true and expected value that works fine for large data during long time scale. However in case of networking, prediction should be on diverse parameters based on most recent values. The older values should have least influence in predicting the values. The protocol TCP used in internet is based on retransmission time out. TCP is a peer to peer reliable, connection oriented protocol due to its constant exchange of control message. In addition to this data is transferred in enumerated manner with sequence number assigned to each byte to guarantee ordered transfer of data. The number of such segments generated depends upon Maximum Segment Size (MSS).The number of segments can be calculated using (1).

$$NOS = \lceil S/MSS \rceil \quad (1)$$

Where
NOS=Generated number of segments.
S=Size of file in bytes.
MSS=Maximum Segment Size.

The Retransmission of packet in TCP is directly related to round trip time. However packet traversal through its route may be exposed to noise, loss of packets, queuing delay thus alienating RTT. Now the question arises, what should be exact value of Retransmission time out? Should it be equal to round trip time, less than round trip time, more than round trip time or should it be fixed or dynamic. If the time will be too short there will be unnecessary retransmission, which will ultimately lead to congestion. If time will be too large response time will be too slow. The retransmission time should be little larger than round trip time. The fixed value of retransmission time out will not either help, due to dynamic nature of network parameters. Overestimating the retransmission time out will result in devastating of congestion Control mechanisms [1] [2].

A better approach is to estimate the round trip time of next packet by using the information of last recently sent packets

According to RFC 793 the round trip time can be estimated based on previous values as shown in (2).

$$RTTe\ (K+1) = \alpha * RTTm\ (k) + (1-\alpha) * RTTe\ (K+1) \quad (2)$$

Where
RTTe = Estimated Round Trip Time.
RTTm= Measured Round Trip Time.
α= constant whose value is calculated as 0.125.

Equation (2) can be rewritten as

$$RTTe\ (K+1)(1-\alpha) * RTTm\ (K+1) + \alpha(1-\alpha) * RTTm\ (K) + \cdots + \alpha^{(K+1)}\ (1-\alpha)\ \ RTTm\ (0) \quad (3)$$

Further solving the (3) to a form as

$$RTTe\ (K+1) = \sum_{K=0}^{K+1}\ {}_{\alpha}^{K}(1-\alpha) * RTTm\ (k) \quad (4)$$

Substituting the value of α in (4) we get.

$$RTTe\ (K+1) = \sum_{K=0}^{K+1}\ {}_{0.125}^{K}(0.875) * RTTm\ (k) \quad (5)$$

The above equation is a recursive one and is rewritten form of Jacobson's model for calculating Round trip time.

To calculate retransmission time out, the value of round trip time is multiplied by a constant factor ρ having value of 2.

$$RTO\ (K+1) = \rho * RTTe\ (K+1) \quad (6)$$

Another method of calculating round trip time is by using weighted median average.

$$MDEV = \delta(\ulcorner E\ \ N \urcorner, W) \quad (7)$$

E= Set of last previously calculated estimator of round trip time.
N= Set of last previously calculated measurement of round trip time.
W= Set consisting of corresponding weights of data.

Equation (7) can be rewritten as

$$MDEV(x) = E\ulcorner |x - E(x)| \urcorner \quad (\ulcorner\urcorner \text{ represents Ceiling function}) \quad (8)$$

II. PROPOSED TECHNIQUE

Artificial neural networks are widely used for estimating the future values for systems that are non-linear in nature due to their dynamic adaptive nature and ability to learn new sequences[3] [4]. So Artificial Neural networks are by and large used for Routing protocols, Flow Control Algorithms, Monitoring Algorithms and predicting end to end delays[5] [6] [7] [8] [9].

Packet analyzing software Wireshark is used for analyzing the packets in the Research lab of Baba Ghulam Shah Badshah University (BGSBU) as shown in Fig. 1 Round trip time is calculated in Matlab using Jacobson's Method as in (2).

Taking about 10 lakh samples and frame 1 as base frame whose RTTm is measured by wire shark as 0.1499800 seconds.

RTTe0 = 0.00187475.

RTTe1= (0.875 * 0.00187475) + (0.125* 0.000143000)

= 0.00164040625 + 0.000017875

=0.00165828125

RTTe2= (0.875*0.00165828125) + (0.125 *0.000102000)

= 0.00145099609375 + 0.00001275

=0.00146374609375

RTTe3= (0.875 * 0.00146374609375) + (0.125 *0.093561000)

= 0.0012807778320313 + 0.011695125

= 0.0129759028320313

RTTe4= (0.875 * 0.0129759028320313) + (0.125 * 0.000206000)

= 0.0113539149780274 + 0.00002575

= 0.0113796649780274

RTTe5= (0.875 * 0.0113796649780274) + (0.125 * 0.093032000)

=0.009957206855773975 + 0.011629

= 0.021586206855773975

RTTe6= (0.875 * 0.021586206855773975) + (0.125 * 0.000155000)

= 0.018887930998802228125 + 0.000019375

=0.018907305998802228125

RTTe7= (0.875 * 0.018907305998802228125) + (0.125 * 0.073195000)

= 0.016543892748951949609375 + 0.009149375

= 0.025693267748951949609375

RTTe8= (0.875 * 0.025693267748951949609375) + (0.125 * 0.000224000)

= 0.022481609280332955908203125 + 0.000028
=0.022509609280332955908203125

RTTe9= (0.875 *0.022509609280332955908203125) + (0.125 * 0.073202000)

= 0.019695908120291336419677734375 + 0.00915025

= 0.028846158120291336419677734375

The graph of calculated and estimated round trip time is shown in Fig. 3 and Fig. 4 respectively.

From Fig. 5 it is evident that estimated round trip value should be greater than calculated value in order to avoid unnecessary retransmission of data that may leading to congestion. Thus effective estimation of round trip time is very important as it helps in avoiding congestion in a computer network.

A. Methodology of predicting values and learning mechanism

Our justification for using neural networks is due to its ability to learn and to predict time series of values. Therefore in this method historical data i.e. History of inputs is used to predict the output by utilizing feedback information. This recurrent neural network is shown in Fig. 6.

The model is created by analysing Round trip time for a specific time period. The supervised learning is used by recurrent neural networks to predict the future values.

Let

$$RTT(Z) = \{RTT(Z-1), RTT(Z-2), \ldots RTT(1)\} \quad (9)$$

The method for predicting value is defines as in (10).

$$RTT¥(z) = fn(RTT(Z-4), RTT(Z-3), RTT(Z-2), RTT(Z-1)) \quad (10)$$

Where fn denotes the non-linear function used in recurrent neural network. And the error is defines as in (11).

$E = [|RTT - RTT¥|]^n$ Where n is minimum for appropriate value of n (11)

By giving the previous historical measurements $RTT(Z-4), RTT(Z-3), RTT(Z-2), RTT(Z-1)$ we like to predict the value of $RTT¥(z)$ that is as close as possible to target value.

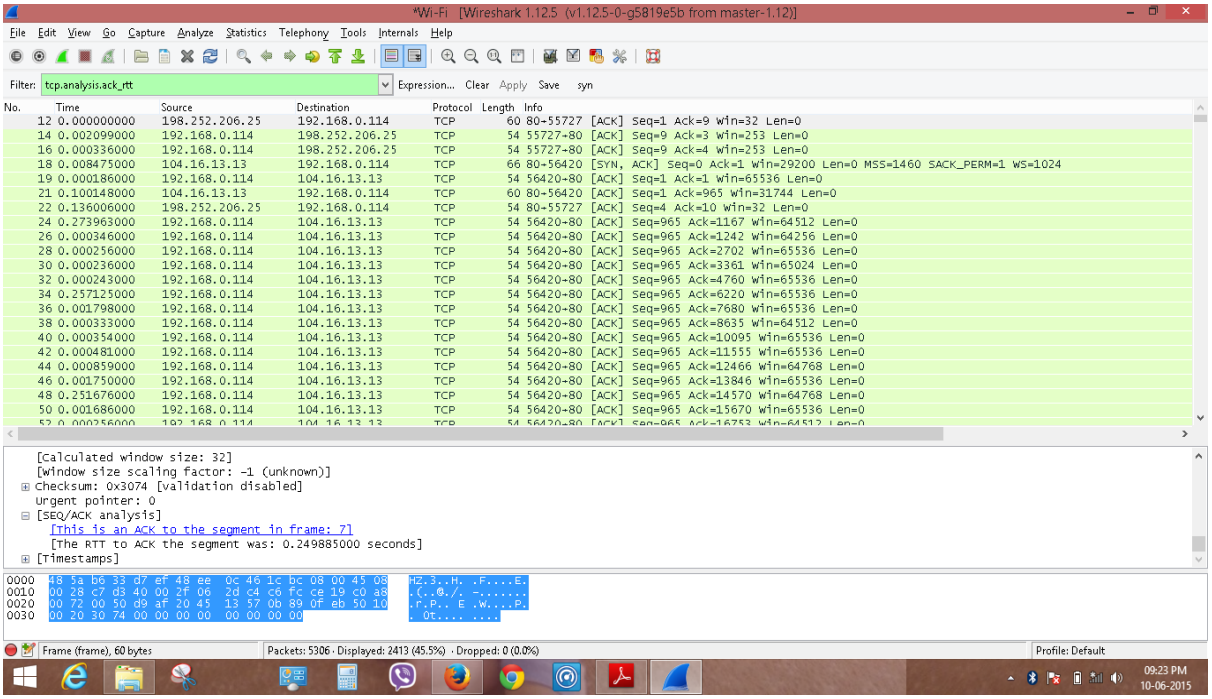

Fig. 1: Representing Wireshark Packet Capturing

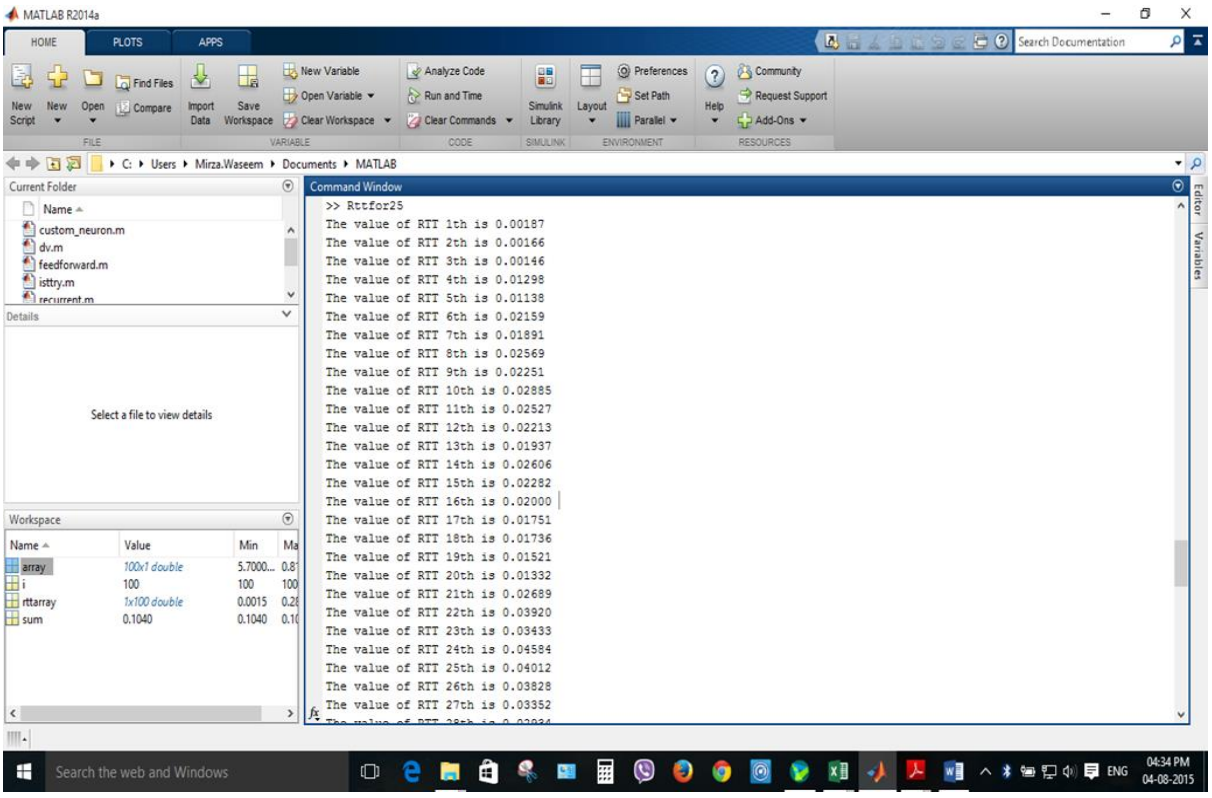

Fig. 2: Representing Calculated Round Trip time

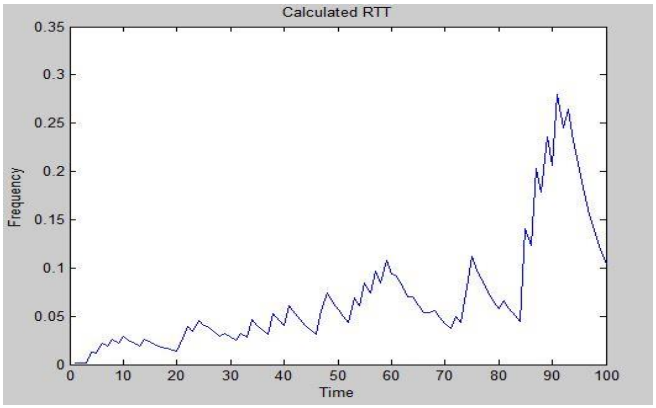

Fig. 3: Represents The Calculated Value of Round trip time

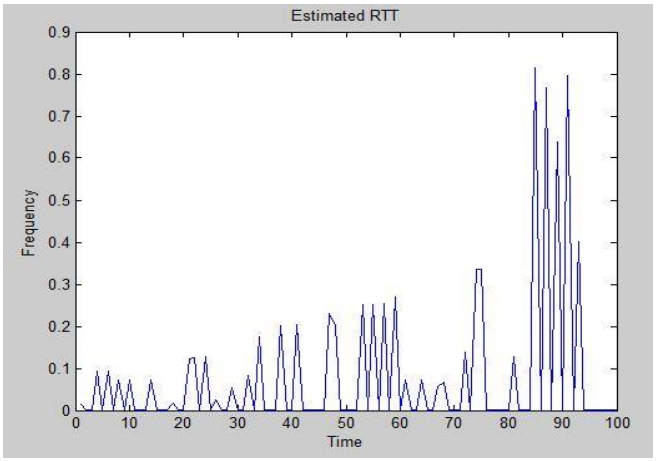

Fig. 4: Represents Estimated Round Trip time

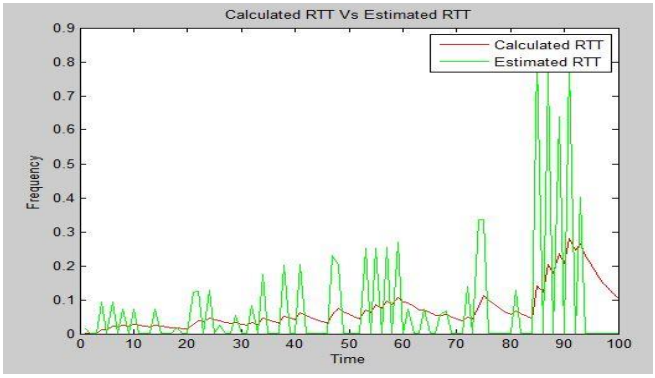

Fig. 5: Represents calculated vs Estimated round trip time

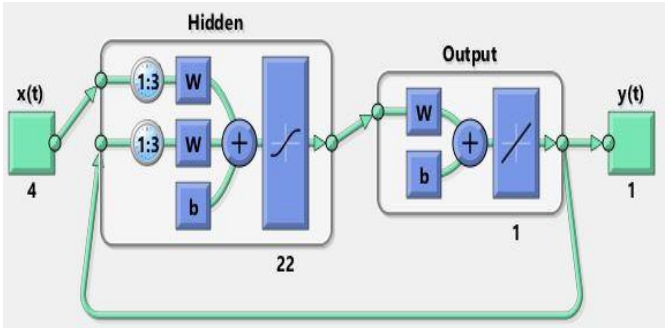

Fig. 6: Representing recurrent neural networks

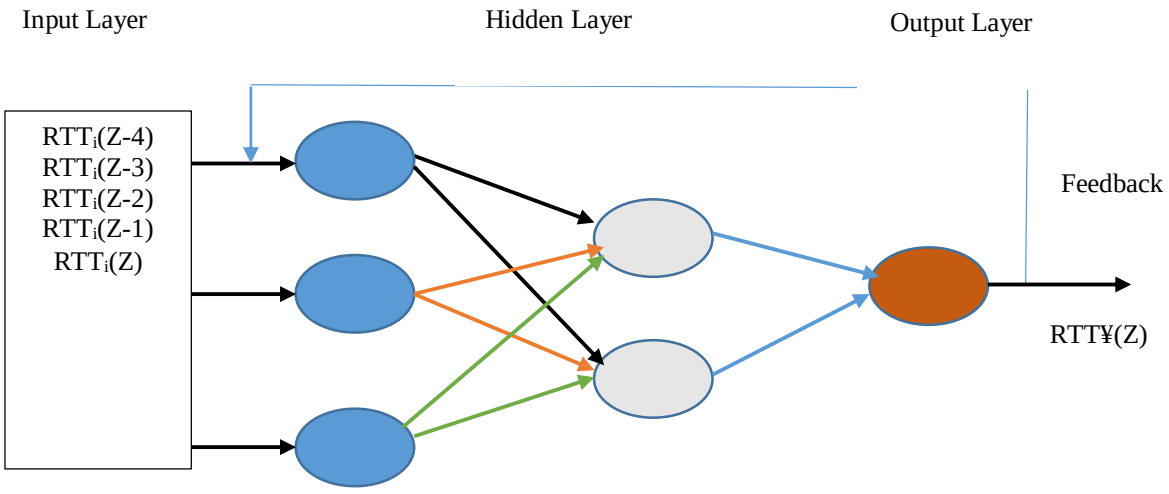

Fig. 7: Representing Proposed Recurrent Neural network model

III. SIMULATION

This approach has been verified by using Matlab Neural network toolbox. The Recurrent neural networks has 5 inputs Nodes, varying hidden nodes and one output node .The number of hidden nodes are varied deliberately to check the performance of changing the hidden nodes. The output is

provided as feedback .Therefore data is organized as 5 value set, with first four as input data and fifth as target to calculate the error. The data is organized in two forms; in first form the data is organized in such a way such that when new value is entered the oldest value disappears with step size 1 as shown in table I.

TABLE I. REPRESENTING INPUT DATA WITH STEP SIZE 1

Input values				Target
1	2	3	4	5
2	3	4	5	6
3	4	5	6	7

In the second form the data is organised in such way that each time 4 new data are used to replace 4 oldest data to predict the target with step size of 4 as shown in table II.

TABLE II. REPRESENTING INPUT DATA WITH STEP SIZE 4

Input values				Target
1	2	3	4	5
5	6	7	8	9
9	10	11	12	13

Furthermore the data is using divided randomly using 70% of data for training, 15% for validation, remaining 15% for testing purpose and network is trained using Leverberg-Marquardt algorithm. Fig. 8, Fig. 9 shows the best value of validation performance with different step size 4 with different size of hidden layer and delay units.

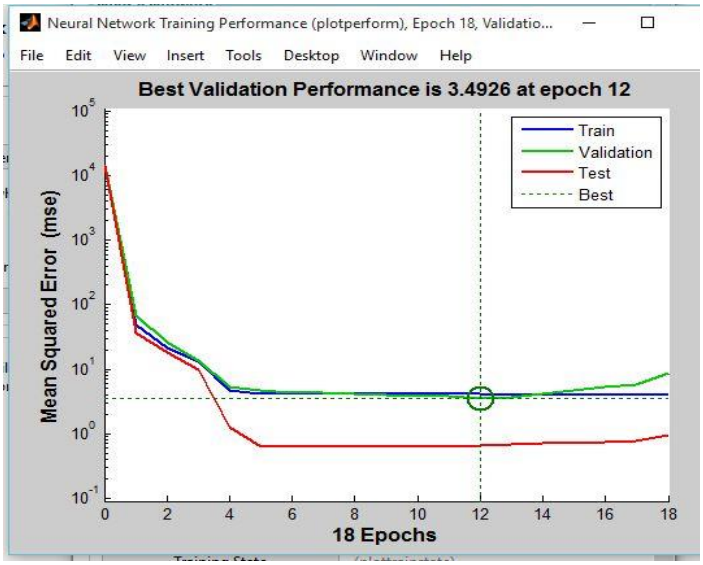

Fig. 8: Representing performance of RNN with Step size 4, 22 hidden layers and 1 delay unit

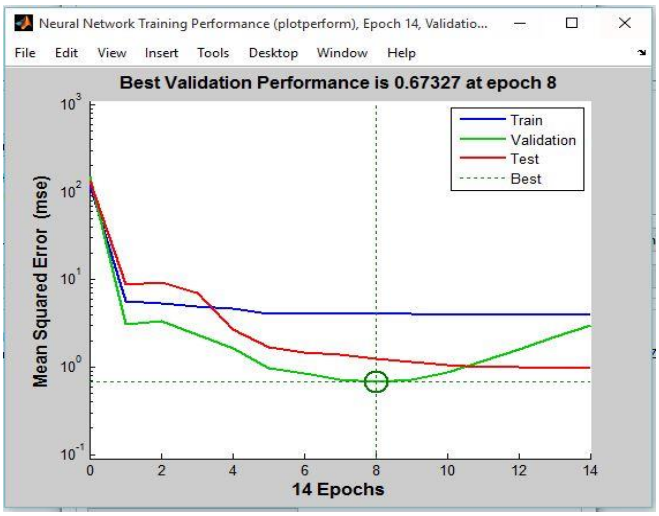

Fig. 9: Representing performance of RNN with Step size 4, 26 hidden layers and 2 delay unit

From the Fig. 8 and Fig. 9 it is clear that the performance of predicting the best value of round trip time increases as the number neurons in hidden layer and delay units are increased .This is evident because weights are adjusted to their best values as the number of neuron and delay units increases. The Fig. 11 shows the changing value of throughput with increase in time. Initially the throughput start to increase once it reached to highest load it started to decreases due to increase in the load of a network during the peak hours. Fig. 12 shows the change in window size .Initially the size of window is small then the size of windows is increased until the size of window becomes constant

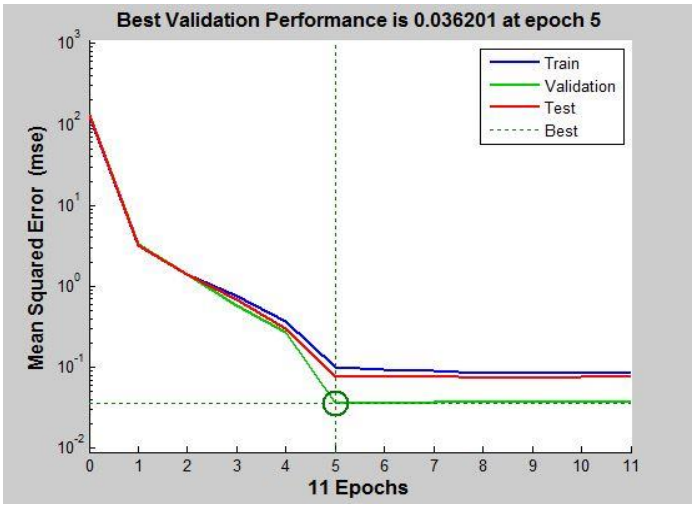

Fig. 10: Representing performance of RNN with Step size 1, 20 hidden layers and 3 delay unit

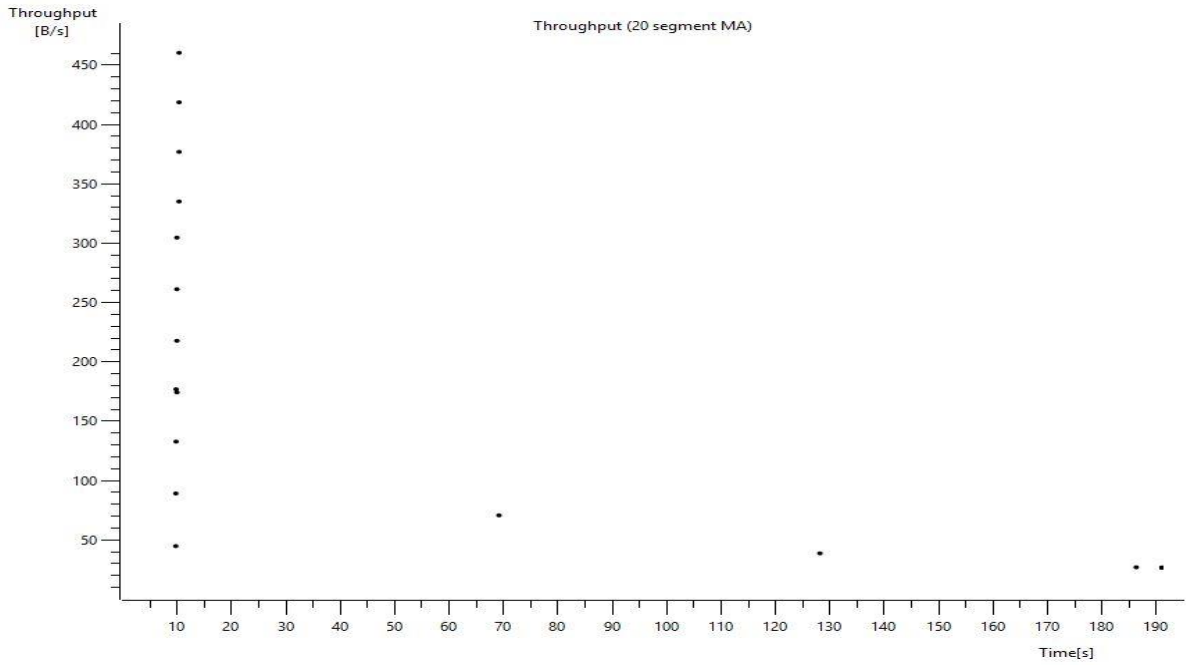

Fig. 11: Representing throughput of network vs time

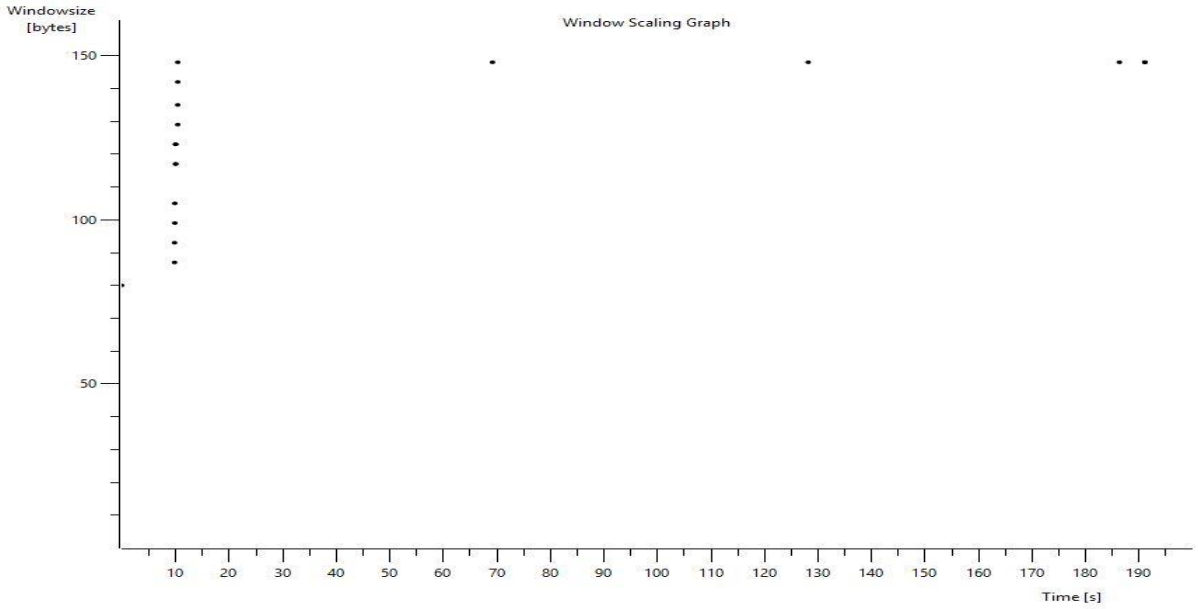

Fig. 12: Represents window size vs time

IV. CONCLUSION

To control the congestion in TCP/IP network it is very important to estimate the accurate value of RTT. Using neural networks approach, the obtained results confirms that neural networks are the best alternate solution for calculating the round trip time to alleviate the state of congestion in a network. Further it was found that by increasing the number of layers and delay units, the performance of a network increased .This prediction based neural network model suggests the prediction of future RTT single and multistep ahead. This research is still in early stage of study. In future, work will be done to further improve the performance and to optimize the error in predicting the values.

V. REFERENCES

[1] L. Ma, K. Barner, and G. Arce, "Statistical analysis of TCP's retransmission Timeout algorithm," *IEEE/ACM Trans. Networking*, vol. 14, no.2, Apr. 2006.

[2] L. Ma, G. R. Arce, and K. Barner, "TCP retransmission timeout algorithm using weighted medians," *IEEE Signal Process. Lett.*, vol.11, no. 6, pp. 569–572, June 2004.

[3] Yang, M., Li., X.R., and Chen, H. "Predicting Internet end to-end delay: An overview," In Proc. of the 36th South.Sym. on Sys. Theo., pp. 210-214, March 2004.

[4] Scarelli, F. and Chung Tsoi, A. "Universal approximation using feedforward neural networks: A survey of some existing methods, and some new results". In Neural Networks, Vol. 11, pp. 15-37, January 1998.

[5] Zhihao, Guo and Malakooti, B., "Delay prediction for intelligent routing in wireless networks using neural networks," In Proc. of the IEEE Int. Conf. on Net., Sens. and Cont. (ICNSC'06), pp. 625– 630, April 2006.

[6] Jacobson, V., "Congestion avoidance and control," In Proceedings of ACM/SIGCOMM'88, August 1988. Available:ftp://ftp.ee.lbl.gov/papers/congavoid.ps.Z

[7] Lawrence, S. B. and Larry L. P., "TCP Vegas: end-to-end congestion avoidance on a global Internet," IEEE Jour. On Sel. Ar. in Com., vol. 13, no. 8, pp. 1465-1480, October 1995, Available:http://cs.princeton.edu/nsg/papers/jsacvegas.ps

[8] Jin, C., Wei, D. X. and Low, S. H., "FAST TCP:Motivation, Architecture, Algorithms, Performance," In Proceedings of the IEEE INFOCOM'04, pp. 2490-2501, March 2004, Available:http://netlab.caltech.edu

[9] Rao, N.S.V., "Overlay networks of in-situ instruments for probabilistic guarantees on message delays in wide area networks," IEEE Jour. in on Sel. Ar. in Com., vol. 22, no. 1, pp. 79-90, January 2004.