System Identification of Internet transmission rate control factors

Sung-Goo Yoo*, Young-Seok Kim**, and Kil-To Chong***

* Department of Control and Instrumentation Engineering, Chonbuk University, Jeonju, Korea
(Tel : +82-63-270-2478; E-mail: ding5@hanmail.net)
** Division of Mobile Communication, Samsung Electronics Co., LTD, Suwon, Korea
(Tel : +82-31-279-8502; E-mail: yongseok528.kim@samsung.com)
***Division of Electronics and Information, Chonbuk University, Jeonju, Korea
(Tel : +82-63-270-2478; E-mail: kitchong@chonbuk.ac.kr)

Abstract: As the real-time multimedia applications through Internet increase, the bandwidth available to TCP connections is oppressed by the UDP traffic, result in the performance of overall system is extremely deteriorated. Therefore, developing a new transmission protocol is necessary. The TCP-friendly algorithm is an example meeting this necessity. The TCP-friendly (TFRC) is an UDP-based protocol that controls the transmission rate based on the available round transmission time (RTT) and the packet loss rate (PLR). In the data transmission processing, transmission rate is determined based on the conditions of the previous transmission period. If the one-step ahead predicted values of the control factors are available, the performance will be improved significantly. This paper proposes a prediction model of transmission rate control factors that will be used for the transmission rate control, which improves the performance of the networks. The model developed through this research is predicting one-step ahead variables of RTT and PLR. A multiplayer perceptron neural network is used as the prediction model and Levenberg-Marquardt algorithm is used for the training. The values of RTT and PLR were collected using TFRC protocol in the real system. The obtained prediction model is validated using new data set and the results show that the obtained model predicts the factors accurately.

Keywords: multimedia transmission, round trip time, packet loss rate, one-step ahead prediction model, neural network

1. INTRODUCTION

Most of Internet traffic are protocol based on TCP such as HTTP(Hypertext Transfer Protocol), SMTP(Simple Mail Transfer Protocol), FTP(File Transfer Protocol). According as the real-time audio/video streaming application such as IP telephony, Internet audio player, VOD services increase, these became important cause of Internet traffic. Usually, the real-time application uses the UDP algorithm that does not consider congestion control instead of the TCP algorithm that uses complex retransmission mechanism. If congestion condition happens when TCP and UDP share equal link, the TCP reduces transmission rate to solve congestion problem but the UDP keeps transmission rate of determined. So the UDP has most of effective bandwidth and make congestion condition intricately. Furthermore, Unfairness happens in use of networks.

To solve these problems, need to apply transmission rate adjustment rule in non-TCP traffic for coexist with TCP's transmission rate adjustment mechanism. This transmission rate adjustment rule must make the non-TCP applications have TCP-friendly property and the system supports uniform distribution. Several TCP-friendly algorithms that solve unequal distribution problem proposed. Present, proposed main TCP-friendly algorithm examples are Rate Adaptation Protocol(RAP) and TCP Friendly Rate Control(TFRC) etc.

The TCP-friendly is a protocol that can control the transmission rate adaptively on the Internet congestion condition. But this emphasized fairness with TCP flow and did not consider QoS(Quality of Service) that influence in quality of transmitted picture. Also current TCP-friendly is an UDP-based protocol that controls transmission rate by using one-step prior round trip time(RTT) and packet loss rate(PLR). This research proposes a prediction model of transmission rate control factors. This prediction model is able to manage transmission rates actively in the congestion condition through the considering of Internet bandwidth depending on time. It is a model that predicts RTT and PLR by using neural networks.

There are several prediction-modeling methods such as Decision Tree, Rule, and Neural Networks. This research gives a prediction based on the Neural Network method, which has the ability of modeling and implementation of an uncertain nonlinear system. The Neural Network model is made of Multi-layer perceptron (MLP) structure, which improves performance of Neural Networks through the non-linearization system of input/output characteristics. Levenberg-Marquardt Back-propagation(LMBP) is used for training method, which can overcome the local minimum point problem and improve training convergence time. For the purpose of collecting training data of Neural Network Prediction Model (NNPM), each of two Linux OS computers is placed in Chonbuk National University and Seoul National University. TFRC algorithm is used for packet transmission experiment between two computers. The IPERF (traffic generator) is used to produce various network congestion conditions. This paper is organized as follow. In chapter 2, a TCP- friendly rate control method will be described. In chapter 3, neural network structure and LMBP algorithm is illustrated. In chapter 4, the prediction model set up and analysis of simulation results will be described. Finally, we conclude our paper and outline future work in chapter 5.

2. TCP FRIENDLY RATE CONTROL

When the multimedia data is provided from the web server, there are two ways to regenerate the multimedia at the client side. One of them is a down loading method, which plays the multimedia after getting all of the data. The other one is a streaming method, which plays the multimedia while the data is being downloaded. This streaming method is better for the real time audio or video broadcasting when the data is transmitted through the Internet.

Packet losses in the Internet arise when there exist transmission error or the network is under congestion condition. When there are packet losses, TCP protocol method resulted in transmission rate reduction due to the congestion control, if each TCP connection shares the channel when the connections have similar RTT, every TCP connections share bandwidth with same priority. So far, most of the data transmission has used TCP base protocol, thus the bandwidth sharing was not a problem. However, non-TCP traffic from real-time services such as IP-telephony, image conference and the streaming services for voice/video are increasing. Thus, the bandwidth sharing becomes a very important problem. Since the non-TCP protocol does not have congestion control mechanism compatible to TCP protocol, non-TCP traffic is transmitted at the consistent rate while the TCP reduces the transmission rate when there is congestion in the networks. Therefore, the congestion in the networks is getting worse. To solve this problem, the non-TCP protocol must have certain mechanism controlling the transmission rate, which can be compatible with TCP protocol. TCP-friendly rate control is the method, which can solve the problem mentioned in the above. This method makes the TCP and non-TCP share the bandwidth fairly. The TCP-friendly rate control method regulates the transmission rate with the rule of slow start and with the congestion avoidance algorithm as shown in Fig. 1. The transmission rate can be expressed as in equation(1)

$$R = f(P, RTT) \tag{1}$$

where, R is transmission rate, P is packet loss rate, and RTT indicates the round trip time. Generally, TCP protocol adjusts the congestion window size on the Fig 1 if it is assumed to be in steady state condition. This is corresponding to the operational principle of TCP Reno model.

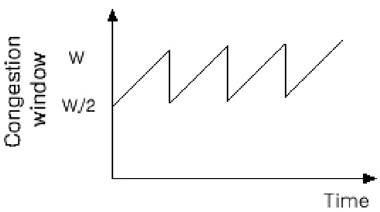


Fig 1. TCP Reno's operation mechanism in steady state

The average transmission rate is $R = \frac{W \times s}{RTT}$, and the window size is reduced to $\frac{w}{2}$ if there is packet loss. And then, the average transmission rate becomes $R = 0.5 \times \frac{W \times s}{RTT}$. The average transmission rate in total range turn to be $R = 0.75 \times \frac{W \times s}{RTT}$, where W is window size and s is packet size. The packet loss rate p of single saw tooth cycle in Fig 1 is $\frac{1}{p} = \left(\frac{W}{2}\right)^2 + \frac{1}{2}\left(\frac{W}{2}\right)^2$. The window size can be approximated to $W \approx \sqrt{\frac{8}{3p}}$. Finally, the effective transmission rate $R(t)$ can be obtained as in equation(2) in terms of time t.

$$R(t) = \frac{1.22 \times s}{RTT(t) \times \sqrt{p(t)}} \tag{2}$$

In this paper, RTT and PLR is collected using equation (2).

3. NEURAL NETWORKS

3.1 Structure of Neural Networks

This research used multi-layer perceptron neural network(MLP) by prediction model structure. The structure is consisted of 3 layers as shown Fig 2. Input layer, Hidden layer and Output layer are it.

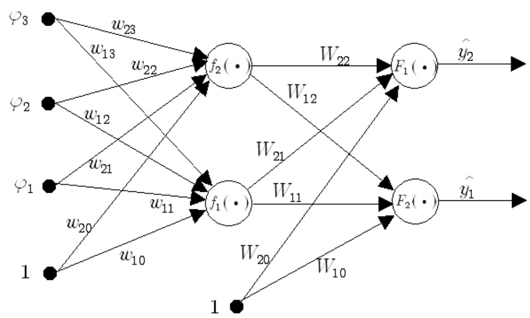


Fig 2. Multi-layer perceptron neural network structure

The MLP's output $\hat{y}$ is

$$\begin{aligned} \hat{y}_i(t) &= g_i[\varphi, \theta] \\ &= F_i\left[\sum_{j=1}^{n_h} W_{i,j} f_j\left(\sum_{l=1}^{n_\varphi} w_{j,l}\varphi_l + w_{j,0}\right) + W_{i,0}\right] \end{aligned} \tag{3}$$

where θ is the parameter vector, which contains all the adjustable parameters of the network, the weights and biases $\{w_{j,l}, W_{i,j}\}$. Usually biases are 1. To determine the weight values one must have a set of examples of how the outputs, $\hat{y}_i$, should relate to the inputs, φ_l. The task of determining the weights from these examples is called training or learning. Error function E, which indicates difference of training output and MLP's output is

$$E = \frac{1}{2}\sum_{n=1}^{k}(y_n - o_n)^2 \tag{4}$$

where y_n is training output that desired value, o_n is output of MLP.

3.2 LMBP Training Method

Several training algorithms such as steepest descent, Newton, Gauss-Newton are used neural networks. Among these methods, the steepest descent has a convergence problem. The Newton method has a good performance of convergent but this method has a complex process because using a second order derivative. So generally the Gauss-Newton method is used many neural network applications.

The LM method to be one of a Gauss-Newton method can solve the steepest descent and the Newton's problems dynamically. The LM method gives high worth to steepest descent and learns at early training process. When the convergence rate decreases, on the contrary, the LM gives high worth to Newton method. So this is converged by local minimum. And then, the LM method does to converge rapid by optimal solution using again steepest descent method.

That is, weights are attained as that training using following equation.

$$w_{i+1} = w_i - (H + \lambda I)^{-1} \nabla F(w_i) \tag{7}$$

where

$$\nabla F(w_i) = \frac{\partial F}{\partial w_i} : gradient\ ,\ \ i \text{ is } \ i \text{ th weights} \tag{8}$$

$$F = \sum_{k=0}^{N} e_k^2 \tag{9}$$

is SSE(square-sum error), k is k th sample

$$H = \nabla^2 F(w) \text{ is hessian matrix} \tag{10}$$

λ is controlled dynamically.

However, Gauss-Newton method is used BP neural network that using LM algorithm(LMBP) because Hessian matrix of Newton method is used second order derivative. The Gauss-Newton method applies changing second order derivative with first order derivative. The Hessian matrix of Newton method is attained as following,

$$\begin{aligned} H &= \left[\nabla^2 F(w)\right]_{ij} = \frac{\partial^2 F(x)}{\partial w_i \partial w_j} \\ &= 2\sum_{k=0}^{N} \left[\frac{\partial e_k(w)}{\partial w_i} \frac{\partial e_k(w)}{\partial w_j} + e_k(w) \frac{\partial^2 e_k(w)}{\partial w_i \partial w_j} \right] \end{aligned} \tag{11}$$

And the second term of equation (11) is able to ignore enough. Therefore we can write as following.

$$\begin{aligned} \left[\nabla^2 F(w)\right]_{ij} &\cong 2\sum_{k=0}^{N} \frac{\partial e_k(w)}{\partial w_i} \frac{\partial e_k(w)}{\partial w_j} \\ &= 2J^T(w)J(w) \end{aligned} \tag{12}$$

where $J_{ki} = \dfrac{\partial e_k}{\partial w_i}$ is Jacobian matrix.

Applying this approximation, we can remove necessity of second order derivative. And $\nabla F(w_i)$ of equation (12) is redefined as following.

$$\nabla F(w_i) = J^T(w_i) e(w_i) \tag{13}$$

Finally, we are able to attained modified LM algorithm.

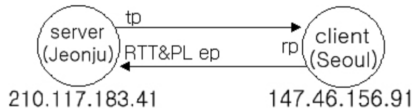
$$w_{m+1} = w_m - \left[J^T(w_m)J(w_m) + \lambda_m I\right]^{-1} J^T(w_m) e(w_m) \tag{14}$$

The equation (14) is weights parameter that is controlled at m th repeat process.

4. NEURAL NETWORK PREDICTION MODEL

4.1 Collect training and validation data

For the purpose of collecting training and validation data, constructed experiment system as Fig 3. Each of two Linux OS computers is placed in Chonbuk National University and Seoul National University. And we programmed transmission processor(TP), receive-retransmission processor(RP) and RTT-PL estimation processor(RTT-PL EP), respectively by using ANSI c language socket program. The TP and RTT-PL EP is installed server in Chonbuk National University and the RP is installed client in Seoul National University.



	OS	CPU	RAM
SERVER	Rethat Linux 8.0 (Kernal 2.4.20)	Pentium IV 1.7GHz	512Mbytes
CLIENT	Redhat Linux 8.0 (Kernal 2.4.20)	AMD Athlon 1200	512Mbytes

Fig 3. Experiment system

The TP transmits packet by TFRC method to refer in Chapter 2. A packet transmission experiment progressed using equation (2). The initial sending rate is 100Kb/s, the packet size is 625byte and 64byte is size of probe header. The probe header attached header to measure of RTT and PLR. This probe header consists of sequence number that displays the sequence of transmitted packet and time stamp that saves the transmission time. Fig 4 shows composition of the probe header. After the TP sends packet to RP, the RP that receive packet separates the probe header and attaches packet sequence number and present time to the probe header. Then, retransmit packet to RTT-PL EP. The RTT-PL EP that receive the probe header measures a RTT and PLR using equation (15) and (16).

$$RTT(ms) = time(ep) - time(tp) \tag{15}$$

$$\begin{aligned} PL(\%) &= 1 - \frac{\text{All sum of packet that receive in R } i}{\text{All sum of packet that transmit in R } i} \times 100 \\ &= 1 - \frac{R_i - R_{i-1}}{S_i - S_{i-1}} \times 100 \end{aligned} \tag{16}$$

R_i : Last sequence number of data that receive in Round i

S_i : Last sequence number of data that transmit in Round i

R_{i-1} : Last sequence number of data that receive in Round $i-1$

S_{i-1} : Last sequence number of data that transmit in Round $i-1$

where Round(R) means interval of two seconds.

Sequence Number (tp)
Sequence Number (rp)
Sequence Number (ep)
Time Stamp (tp)
Time Stamp (rp)
Time Stamp (ep)
User Data

Fig 4. Probe Header

When occur congestion in the Internet, RTT and PLR are changed rapidly. If control transmission rate applying the RTT and PLR that change rapidly on a TFRC mechanism, congestion condition problem solve can quickly. But in this case, a service of real-time applications grows worse suddenly. To prevent this, the TFRC algorithm’s calculation rule of RTT and PLR follows the TCP algorithm’s rule. The TCP changes RTT and PLR estimation value smoothly using low-pass filter. The estimation values of RTT and PLR use a moving average as follows.

$$RTT^* = \alpha RTT^* + (1-\alpha) new\_RTT$$
$$PLR^* = \alpha PLR^* + (1-\alpha) new\_PLR \tag{17}$$

where α is a coefficient that have recommendation price 0.9, new_RTT and new_PLR are RTT and PLR that measured value newly. The moving average is used transmission rate control making variety of the RTT and PLR softly at congestion condition. The RTT and PLR measured each Round(two seconds) at transmission experiment. In this research, we train the RTT, PLR and the moving average RTT, moving average PLR, respectively and validate the prediction results.

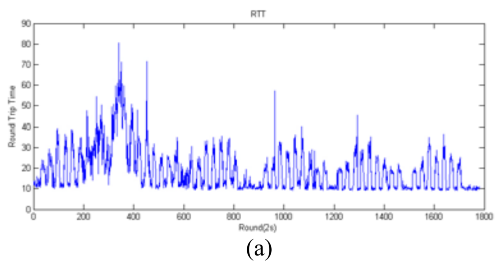
The IPERF(a traffic generator) is used to produce various network congestion conditions and regulated a traffic of networks at experiment process.

Table 1 The RTT and PLR according to traffic size

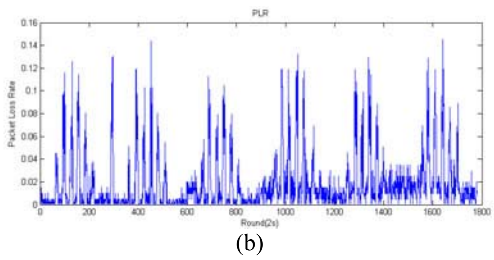
	0 MB	1 MB	2 MB	5 MB	7 MB
Max RTT	10.8ms	17.3ms	23.9ms	45.2ms	79.3ms
Min RTT	10.1ms	11.5ms	19.4ms	29.7ms	34.9ms
Max PLR	1.2%	2.4%	8.1%	12.7%	14.4%
Min PLR	0%	0.5%	3.8%	11.3%	12.5%

Table 1 shows traffic size and RTT and PLR’s results of measurements that corresponded to traffic size. When there is no congestion, the average of RTT is 10.3ms and average of PLR is 0.5%. A load of traffic increases, the RTT and PLR increase sharply.

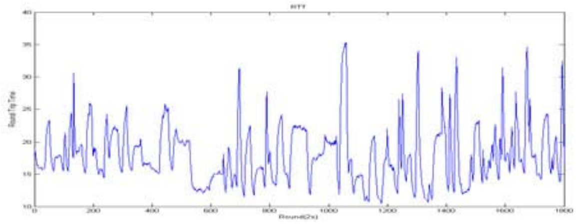
The packet transmission experiments were performed for a week in every time and each of them took thirty minutes. A measurement of RTT and PLR used Round unit that it means two seconds. Fig 5 shows the collected RTT and PLR through a transmission experiments.



(a)



(b)


(c)

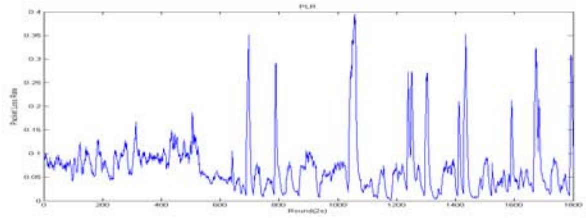

(d)

Fig 5. Value of RTT and PLR that is measured according to transmission experiment result

(a) and (b) in Fig 5 are the result graphs of RTT and PLR that corresponded in case of imposed traffic every sixty seconds. And (c), (d) are the result graphs that imposed random traffic and time.

4.2 Modeling

In this research, we collected about 150,00 RTT and PLR data respectively through a packet transmission experiments. Seventy percent among these was used for training to measure parameters and remainder thirty percent was used in validation to measure performance of neural network prediction model.

The prediction model used MLP structure that is shown in Fig 2. And the structure of neural networks is by input layer 20, hidden layer 8, and output layer 1. A training method is LMBP algorithm that explained in chapter 3.

The activation function that use to neuron is hyperbolic tangent. As follows,

$$f(x) = \tanh(x) = 1 - \frac{2}{\exp(2x)+1} \tag{18}$$

We obtained 168 weights between input layer and hidden layer and 9 weights between hidden layer and output layer through the training.

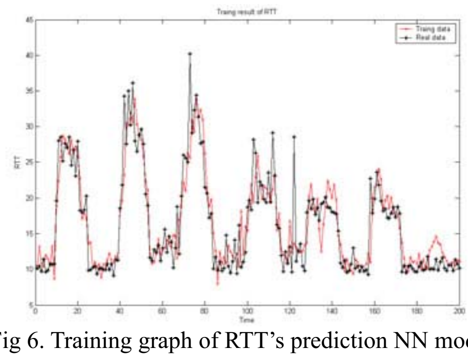


Fig 6. Training graph of RTT’s prediction NN model

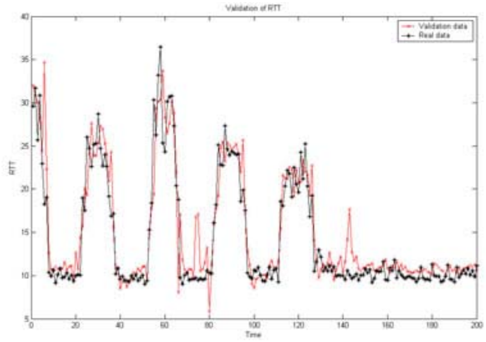


Fig 7. Validation graph of round trip time's prediction NN model

Fig 6 is graph that shows some part of training result that round trip time data and Fig 7 shows validation result that applies data, which unused to training to neural networks model. Point that the RTT size increases are part, which regulate the traffic size of networks using IPERF traffic generator that referred to section 4.1. Through the training result, we can confirm that NN prediction model's prediction error is small to when the RTT by congestion condition of networks increases suddenly. There is some error in result graph that validate using unused data to training. But we can confirm that prediction error is less in the congestion condition. The root-mean-square(RMS) error of training result is 0.34ms and validation result is 1.004ms. These results are small error within extent of the RTT.

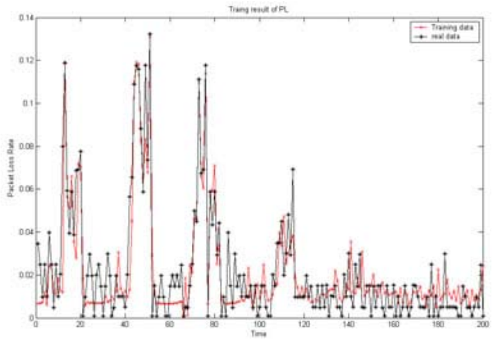


Fig 8. Training graph of PLR's prediction NN mode
l

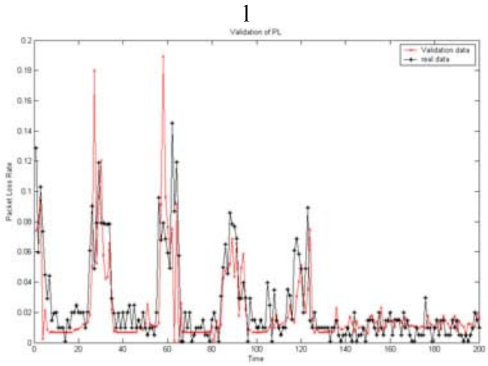


Fig 9. Validation graph of packet loss rate's prediction NN model

Fig 8 and 9 show NN prediction model's training and validation about the packet loss rate. The measured PLR is very small numerical value. So the graphs oscillate extremely in smallish oscillation of PLR. However, if see the result of graphs we can confirm that prediction error is small in the congestion condition. The RMS error of prediction result is as following. The training result is 1.52(%) and validation result is 3.82(%).

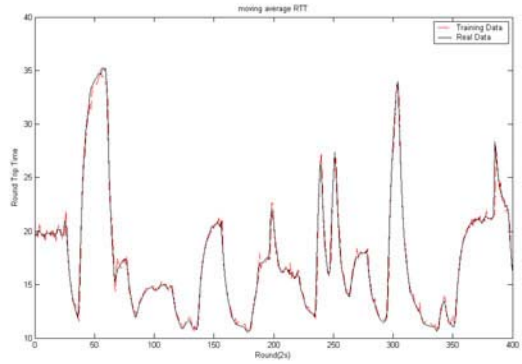


Fig 10. Training graph of moving average RTT's prediction NN model

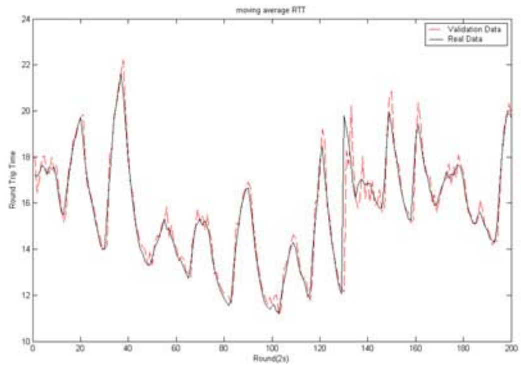


Fig 11. Validation graph of moving average RTT's prediction NN model

Fig 10 and 11 show proposed prediction model's training and validation results about the moving average RTT that measure using equation (17). The moving average RTT has a slackness oscillation range than RTT that oscillate extremely. Therefore, we can confirm that validation results improve that display performance of prediction than Fig 9 and 10. The RMS error of prediction result is as following. The training result is 0.680ms and validation result is 0.7502ms. As this result, we can show that improved performance of prediction system.

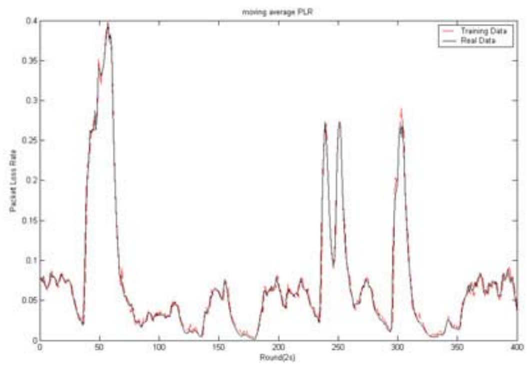


Fig 12. Training graph of moving average PLR's prediction NN model

Fig 12 and 13 show training and validation results about the moving average PLR. Similarly, we can confirm that training and performance of prediction improve than PLR that show sudden oscillation. The RMS error of prediction result is as following. The training result is 0.87(%) and validation result is 0.9(%).

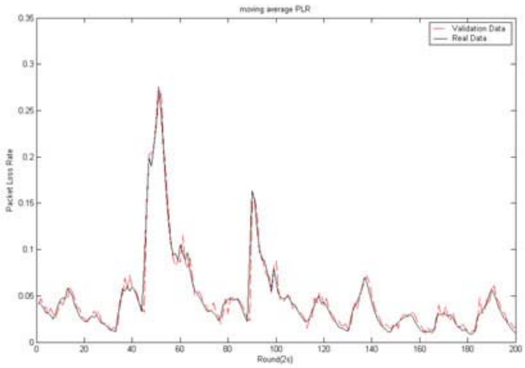


Fig 13. Validation graph of moving average PLR's prediction NN model

5. CONCLUSIONS

In this paper, we proposed a prediction model of transmission rate control factors. This prediction model is able to manage transmission rate actively in the congestion condition by considering time-dependent Internet bandwidth. The neural network prediction modeling method is used, which can model a nonlinear system. Levenberg-Marquardt Back-propagation training method is adopted, which has a high convergence characteristic. RTT and PLR data were collected by using TFRC protocol that is a UDP-based adaptive transmission control method. Those data are used for training of NN prediction model. Through neural network training and validation, we have shown that the prediction model can predict the one-step ahead RTT and PLR, with that the proposed model confirmed that it operates as a predictor with small error. The proposed approach allows predicting Internet congestion conditions. In order to cope with congestion conditions actively, a transmission mechanism will also be implanted by the prediction model. Through this research, we will implement system that control transmission rate of real multimedia data using proposed neural network prediction model. And then we are going to confirm performance as that compare with existent algorithms. These are future research.

REFERENCES

[1] A.S Tanenbaum, *Computer Networks(third edition)*, Prentice Hall International, Inc, 1996.

[2] V. Jacobson, "Congestion Avoidanced and Control," *SIGCOMM Symposium on Communications Architectures and Protocols*, pages 214-329, 1988.

[3] V.Paxson, "Automated packet trace analysis of TCP implementations," *IN Proceedings of SIGCOMM 97*, 1997.

[4] Joerg Widmer, Robert Denda, and Martin Mauve, Parkitsche Informatic IV, "A Survey on TCP-Friendly Congestion Control," *IEEE Network*, vol. 3, pp.28-37, May/June, 2001.

[5] IKJUN YEOM, "ENDE : An End-To-End Network Delay Emulator," Texas A&M University, 1998.

[6] Jitendra Padhye, Victor Firoiu, Don Towsley, Jim Kurose. "Modeling TCP Throughput : A simple Model and its Emirical Validation," *ACM SIGCOMM*, 1998.

[7] Michael J. Donahoo, Kenneth L. Calvert, *The Pocket Guide to TCP/IP Sockets : C Version*, Morgan Kaufmann Publishers, Inc. 2001.

[8] M. Norgaard, O.Ravn, N.K. Poulsen and L.K. Hansen, *Neural Networks for Modelling and Control of Dynamic Systems*, A practitioner's Handbook, Springer.

[9] Jacek M. Zurada, *Introduction to Artificial Neural Systems*, West Publishing Company, 1992.

[10] The IPERF, "http://dast.nlanr.net/Projects/Iperf/".

[11] M. Mathis, J. Semke, J. Mahdavi, and T. Ott, "The macroscopic behavior of TCP Congestion Avoidance Algorithm," *ACM Computer Communication Review*, 27(3):67-82, July 1997.

[12] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural network," *IEEE Trans. Neural networks*, vol. 1, no. 1, pp. 4-27, March, 1990.

[13] S. Haykin, "Neural Networks," MacMillan, 1994.

[14] J. Mahadavi and S. Floyd, "TCP-Friendly unicast rate-based flow control," Tech. Rep., *Technical note sent to the end2end interest mailing list*, January 1997,

[15] Finschi, "An implementation of the Levenberg-Marquardt algorithm," *clausiusstrasses 45*, CH-8092, Zuerich, 1996.

[16] Jae-Gi Lee, Jin Young Choi, "Modeling of Nuclear Power Plant Steam Generator using Neural Networks," *Jonal of Control, Automation and system Engineering,* Vol. 4, No. 4, August, 1998.