Hindawi
Mobile Information Systems
Volume 2017, Article ID 8429878, 8 pages
https://doi.org/10.1155/2017/8429878

Research Article

A Traffic Prediction Model for Self-Adapting Routing Overlay Network in Publish/Subscribe System

Meng Chi,[1] Jianhua Yang,[1,2] Yabo Liu,[1] and Zhenhui Li[3]

[1]*College of Computer Science and Technology, Zhejiang University, 38 Zheda Road, Hangzhou, Zhejiang 310027, China*
[2]*The Sci-Tech Academy, Zhejiang University, 38 Zheda Road, Hangzhou, Zhejiang 310027, China*
[3]*College of Control Science and Engineering, Zhejiang University, 38 Zheda Road, Hangzhou, Zhejiang 310027, China*

Correspondence should be addressed to Jianhua Yang; jhyang@zju.edu.cn

Received 20 January 2017; Accepted 27 February 2017; Published 30 March 2017

Academic Editor: Jaegeol Yim

In large-scale location-based service, an ideal situation is that self-adapting routing strategies use future traffic data as input to generate a topology which could adapt to the changing traffic well. In the paper, we propose a traffic prediction model for the broker in publish/subscribe system, which can predict the traffic of the link in future by neural network. We first introduced our traffic prediction model and then described the model integration. Finally, the experimental results show that our traffic prediction model could predict the traffic of link well.

1. Introduction

Location-based services (LBS) have drawn more and more attention, which can provide us with location-aware experiences. Some LBS applications such as E-coupon and Mobile Buddy List were implemented based on publish/subscribe system.

Providing the fast LBS service to millions of users is a big challenge; self-adapting overlay network and reducing traffic based on traffic prediction in publish/subscribe system provide the solution to this problem, which could deliver a message to end user efficiently.

There are two improvements which will be achieved when the underlying infrastructure of the system is incorporated with publish/subscribe paradigm. One is that this paradigm will provide anonymous communication mechanism. The other is that this paradigm will decouple consumers and producers in terms of space, time, and synchronization [1]. As a result, the publish/subscribe paradigm has been researched both in academia and in industry recently.

In publish/subscribe system, producers send notifications, and consumers receive their interested notifications expressed by the subscriptions [2]. The overlay network forwards notifications to consumers according to the match and routing algorithm.

The content-based publish/subscribe system is commonly applied in many scenarios, and its cost of routing depends on the topology of the dispatching network which is usually defined at deployment time and never changes [3, 4], so it is important to reconfigure the dispatching network at runtime to reduce the overall routing cost. There is always an assumption in the current self-adapting routing strategies [3, 5, 6]; the assumption is that the traffic of the link l at time t is equal to traffic of the link l at time $t + 1$:

$$v_t^l = v_{t+1}^l. \tag{1}$$

Ideally, if a self-adapting routing strategy could reduce the traffic cost of the overlay network efficiently, v_{t+1}^l should be used as input data to figure out the appropriate dispatching network at time t, but actually it cannot get v_{t+1}^l at time t, so it could only assume that $v_t^l = v_{t+1}^l$. However, in fact the traffic of the link is constantly changing, so $v_t^l \neq v_{t+1}^l$, which means that the traffic of the link at time t which is used to figure out the dispatching network is not real traffic of the link at time

$t + 1$. Consequently, the reconfigured overlay network cannot adapt well to the traffic at time $t + 1$.

In the paper, we propose a traffic prediction model which could predict the traffic of the overlay network to overcome the limitation above. We predict v_{t+1}^{l} at time t and neural network is used in our traffic prediction model to predict the traffic of the link. In this way, the self-adapting routing strategy could use the predicted traffic to reconfigure the dispatching network well.

The paper is organized as follows. Section 2 discusses related work. Section 3 proves that the traffic is predictable. Section 4 presents the traffic prediction model. Section 5 presents the model integration. Section 6 shows and discusses the experimental results. Section 7 presents our conclusion.

2. Related Work

Several approaches of self-adapting routing strategy and prediction have been researched in the past. We present and discuss these approaches as follows.

In 2014, we proposed a self-adapting routing strategy for frequently changing application traffic in content-based publish/subscribe system and published in the literature [5]. The strategy firstly records the traffic information and then uses it to figure out a new topology. In the end, the strategy reconfigures the topology of the overlay network to reduce the overall traffic cost. Our research is based on assumption (1), so the reconfigured overlay network cannot adapt well to the traffic at time $t + 1$.

A self-organizing algorithm to solve the problem of publish/subscribe overlay decision problem (PSODP) is presented in the literature [6]. In another work [3], a distributed algorithm to solve the problem of optimal content-based routing (OCBR) was proposed. However, both of these two approaches also are based on assumption (1), so the reconfigured overlay network cannot adapt well to the traffic at time $t + 1$.

A review of neural networks for the prediction and forecasting of water resources variables was summarized in literature [7]. These include the choice of performance criteria, the division and preprocessing of the available data, the determination of appropriate model inputs and network architecture, optimization of the connection weights (training), and model validation. Also, the neural networks prediction theory is applied in other fields, such as traffic flow prediction and stock prediction [8, 9]. However, it is firstly introduced in publish/subscribe system.

3. Predictability Analysis

We shall first prove that the traffic of the publish/subscribe overlay network is predictable before we introduce our traffic prediction model.

In the overlay network, a broker node, connected by several links which contains the inputs and outputs, is our research object, and we want to predict the traffic of a certain link in the broker node. However, the traffic in the broker node always exhibits complicated, irregular behaviors which

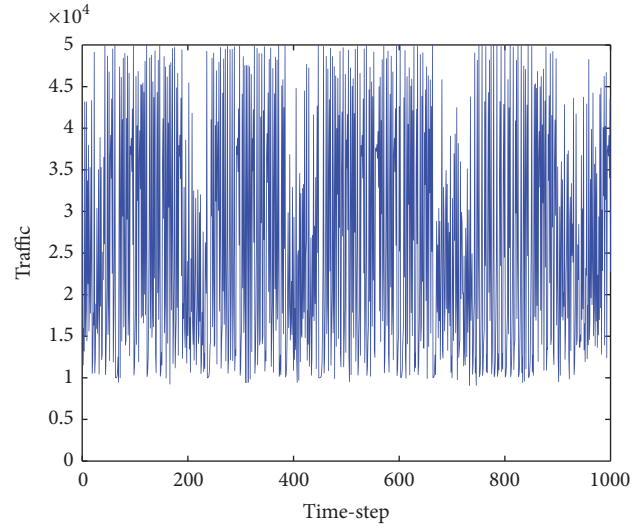

Figure 1: Traffic-time series.

are seemingly random, and they are largely determined by the business logic of the upper level, so we can come to the conclusion that it is impossible to make a prediction for the next hour or day traffic.

On the other hand, short-term (1–15 minutes) prediction has been analyzed as below. The data was recorded from an air traffic control system which is used to monitor the designated airplane by different terminals, and the communications layer of this system is supported by the publish/subscribe system. A node n was chosen randomly in inner brokers, and a link l was chosen randomly as well in node n. We set the node n to record the routing notifications in every 5 minutes, and we call this period as a time-step. We recorded notifications routed by link l 1000 time-steps in the node, which comprise a time series ts. The data are shown in Figure 2.

In Figure 1, x-coordinate is the number of the time-steps and the y-coordinate is the number of the notifications. From the figure, the curve does not show obvious periodicity or complete randomness, and it is still difficult to determine whether there is a certain rule in this time series ts. However, the subscriptions and the advertisement in the publish/subscribe system certainly have the prediction information, because they indicate what type of notification will be sent or received. To find this order and pattern, we introduced chaos theory into our proof procedure.

In chaos theory, chaos refers to an apparent lack of order in a system that nevertheless obeys particular laws or rules, and it is not disorder but a higher order of the universe [10, 11]. If the time series ts exhibits chaos, the traffic of link l could be predictable. The existence of a positive Lyapunov exponent is usually taken as an indication of the chaotic character [12]. In practice, we only need to calculate the largest Lyapunov exponent λ from this time series ts by small data sets method. If $\lambda > 0$, this time series ts is chaotic [13].

All calculation procedures will be done in MATLAB, and the input parameters of the largest Lyapunov exponent algorithm should be calculated firstly, which is shown as follows: reconstruction delay is 2 calculated by the fast

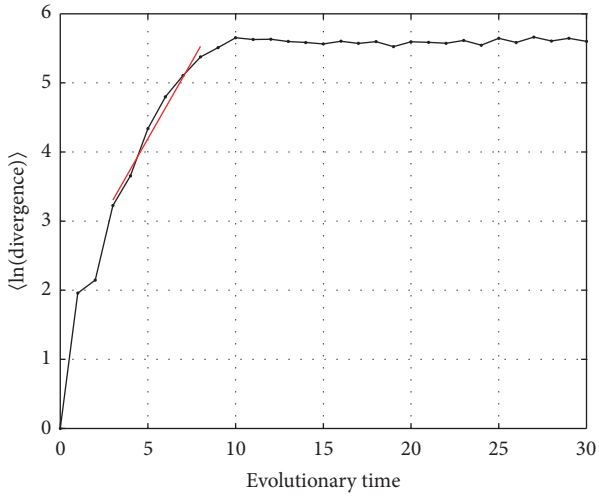

Figure 2: Largest Lyapunov exponent figure.

Fourier transform, mean period is 2 calculated by the fast Fourier transform, and embedding dimension is 3 calculated by C-C algorithm [14]. Now the largest Lyapunov exponent λ could be calculated, and the result is shown in Figure 2.

In Figure 2, x-coordinate is the evolutionary time and the y-coordinate is the $\langle \ln(\text{divergence}) \rangle$. The red line is calculated using least-squares fit method. The slope of this red line is the largest Lyapunov exponent λ. The slope is 0.4363, and the slope is larger than 0, so the time series ts exhibits chaos. We could arrive at the conclusion that the traffic of the publish/subscribe overlay network is predictable.

4. Publish/Subscribe Traffic Prediction Model

The publish/subscribe traffic prediction model (PSTPM) worked in content-based publish/subscribe system, and the advertisements mechanism is applied in the system. We assume that the subscription routing table and the advertisement routing table are updated by covering-based routing algorithm in the broker node.

In the publish/subscribe overlay network, a broker node is our modeling object, it is connected by several links which contains the inputs and outputs, and the traffic of a certain link in the node is our prediction object.

In Figure 3, if the traffic of the link L_k at time $t+1$ is our prediction object, we need to find out which relevant factors the traffic of the link L_k at time $t+1$ depends on. We consider these factors in two categories. One is the unchanged part which is relevant to the history traffic of link L_k, such as the traffic at time t, $t-1$ or $t-2$: here we use two history values $v_t^{L_k}$ and $v_{t-1}^{L_k}$ as the one part of the relevant factors. The other part is changed part which is the traffic that will increase or decrease in the link L_k at time $t+1$: if the subscription entry (S_i, D_i) was added to (deleted from) the subscription routing table at time t in node N_i and the destination D_i is node N_j, it denotes that the matching notifications have to (could not) be forwarded to the destination D_i through the link L_k at time $t+1$. If the advertisement entry (A_i, D_i) was added to (deleted from) the advertisement routing table at time t in node N_i and the destination D_i is node N_j, it denotes that the notifications might (could not) be received from the destination D_i through the link L_k at time $t+1$. Both cases will lead to traffic change in link L_k at time $t+1$, so the number of the change values $\text{sub}v_t^{L_k}$ and the number of the change values $\text{ad}v_t^{L_k}$ are the other two relevant factors. $\text{sub}v_t^{L_k}$ is the sum of forwarding notifications matched by S_i in node N_i and $\text{ad}v_t^{L_k}$ is the sum of forwarding notifications matched by A_i in node N_i.

$$\begin{aligned} \text{sub}v_t^{L_k} &= \sum_{j=1}^{n} \text{Num}\left(\text{Match}\left(\text{Notification}_j, S_i\right)\right), \\ \text{ad}v_t^{L_k} &= \sum_{j=1}^{n} \text{Num}\left(\text{Match}\left(\text{Notification}_j, A_i\right)\right). \end{aligned} \tag{2}$$

In formula (2), the function Num() returns the number of notifications; the function match() returns notification if the notification is matched by filter S_i or A_i; n is the length of the routing table.

In sum, we find out four factors: $v_t^{L_k}$, $v_{t-1}^{L_k}$, $\text{sub}v_t^{L_k}$, and $\text{ad}v_t^{L_k}$, and we use these factors to predict the traffic of link L_k $v_{t+1}^{L_k}$. We need to find some approaches to represent the relation between these two parts.

Neural networks have become extremely popular for prediction and forecasting in many areas [15, 16]. The advantage of neural networks lies in their ability to represent both linear and nonlinear relationships and in their ability to learn these relationships directly from the data being modeled. Now the neural network is used to model our problem and network parameters are discussed as below.

4.1. PSTPM Inputs and Outputs. As in any prediction model, the selection of appropriate model inputs and outputs is very important. In our prediction model, the inputs and outputs are determined by problem itself, so the inputs and outputs are discussed above.

Inputs: $v_t^{L_k}$, $v_{t-1}^{L_k}$, $\text{sub}v_t^{L_k}$, and $\text{ad}v_t^{L_k}$

Outputs: $v_{t+1}^{L_k}$

4.2. PSTPM Data Source. The prediction model is deployed in each broker node, and it is set to record the routing notifications in every t minutes and its default setting is 5 minutes. As a result, the recorded data comprise a time series, input data, and target data, which are used in training the network and could be figured out from the time series according to the selection of inputs and outputs.

4.3. PSTPM Data Division. Cross-validation technique [14] is used in our prediction model to divide the inputs data and targets data into three subsets: a training set which is used for training and accounted for 70%; a validation set which is used to validate that the network is generalizing and to stop training before overfitting and accounted for 15%; a testing

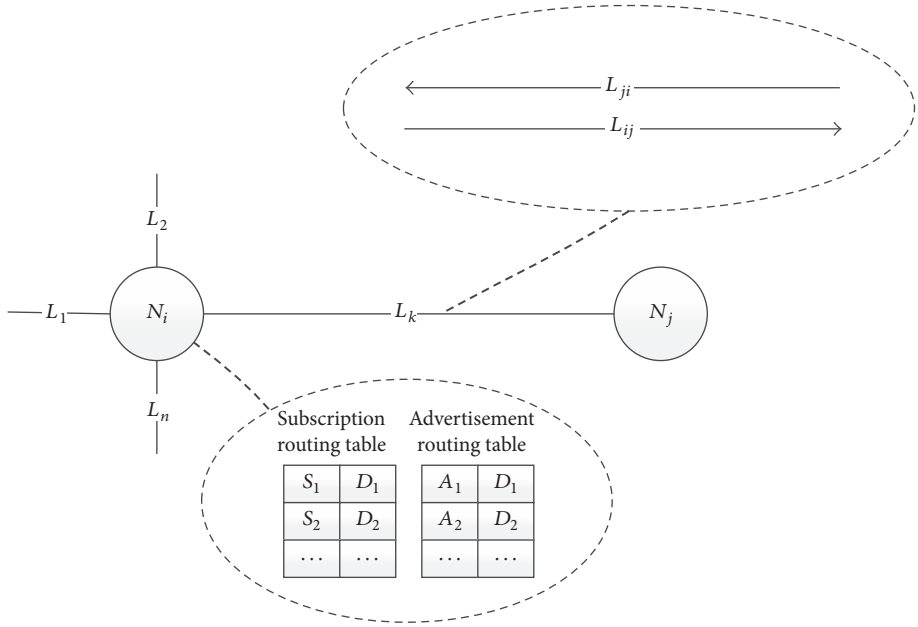

Figure 3: Broker node.

set which is used as a completely independent test of network generalization and accounted for 15%.

4.4. PSTPM Data Preprocessing. Generally, inputs data and targets data span different ranges. In order to ensure that all the data will be commensurate with the limits of the activation functions, in our prediction model, inputs data and targets data should be normalized to a value between 0 and 1 using formula (3).

$$V_t = \frac{v_t - v_{\min}}{v_{\max} - v_{\min}}. \quad (3)$$

In the formula, v_t is the sample data in the time series at time t. $v_{\max}$ and $v_{\min}$ are the maximum and minimum values in the time series. Obviously, the output results of the network which are between 0 and 1 should be carried out with reductions using formula (3).

4.5. PSTPM Network Structure. Feedforward network is used in our prediction model and it is arranged by three layers: an input layer, a hidden layer, and an output layer. The number of neurons in the input layer is fixed by the number of model inputs: 4; the number of neurons in the output layer equals the number of model outputs: 1; the number of neurons in hidden layer nodes was obtained using trial and error and the initial number of neurons is calculated by formula (4):

$$l = \sqrt{n + m} + a. \quad (4)$$

In formula (4), n is the number of the neurons in the input layer; m is the number of the neurons in the output layer; a is one number between 1 and 10.

In addition, we also consider the experience that the number of the hidden nodes in each layer should be between the size of input and output layer.

At last, the number of neurons in the hidden layer is 3.

4.6. PSTPM Network Optimization. In our prediction model, the back-propagation algorithm is used to train network, and the Levenberg-Marquardt (LM) optimization method is used to update the weight and bias. The initial weights are initialized to zero-mean random values in (−1,1). The Tan-Sigmoid transfer function is used in hidden layer and the Log-Sigmoid transfer function is used in output layer. The learning rate is set to 0.1. The error function is used as the mean squared error function, and it is set to 0.001. The maximum epoch size is set to 1000.

5. Model Integration

In this section, we show how the traffic prediction model integrates into the self-adapting routing strategy. Many approaches have been presented to solve the publish/subscribe overlay optimization problem [3, 5, 6]. Normally, the main idea of the self-adapting overlay routing strategy is to reduce the distance between brokers that consume a lot of identical notifications. In this process, the brokers need to know which links will forward a lot of the identical messages in the future, and they will reconfigure the topology of the overlay network to reduce the traffic cost according to these pieces of information. In other words, the more precise the prediction for the traffic that could be given by broker, the more the traffic cost reduction that will be achieved. Consequently, the prediction model presented in this paper could be applicable in all the strategies.

By the above analysis, we could know that the traffic prediction models are running in each broker node, and the model instances are created for each link. The working procedure figure is shown in Figure 4.

In Figure 4, the working procedure contains three procedures: training procedure, prediction procedure, and adapting procedure. In the training procedure, the brokers

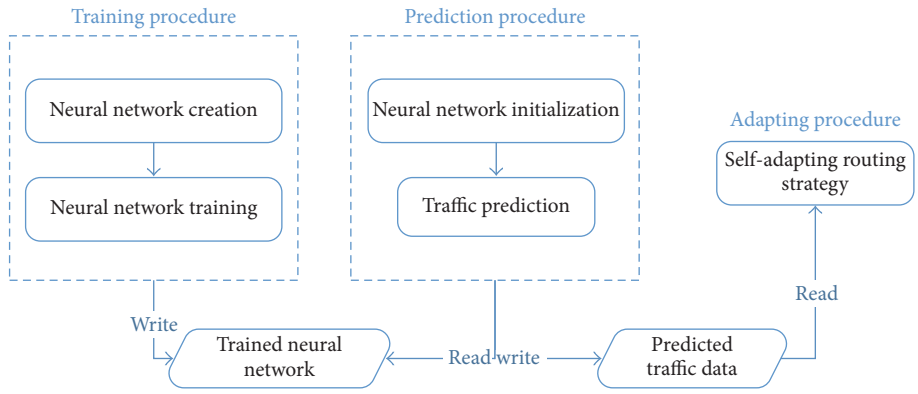

Figure 4: Working procedure.

create the neural network, train the network, and store the trained network for each link. The training procedure runs once a day. In the prediction procedure, the brokers predict the traffic for each link and save the predicted data. In the adapting procedure, the brokers reconfigure the topology of the overlay network using the predicted data. The brokers enter adapting procedure after finishing the prediction procedure; the prediction procedure is triggered after a certain time interval set by system administrator.

6. Experiments

In this section, our experiments were divided into two parts: one part is to validate the traffic prediction model and the other is to validate self-adapting strategy integrated into the prediction model.

For part one, we designed the experiments:

(i) To validate whether the prediction model proposed in this paper has the ability to predict the traffic

For part two, we designed the experiments:

(i) To validate whether the strategy integrated into the prediction model has the ability to reduce the traffic cost of the overlay network and to compare with other strategies

The part one experiments were simulated on MATLAB and the part two experiments were simulated on ProtoPeer [17] which is a distributed systems prototyping toolkit.

6.1. Part One Experiments. The part one experiments are to validate whether the prediction model proposed in this paper has the ability to predict the traffic. The data were recorded from the ProtoPeer environment and analyzed in MATLAB. We designed two experiments in this part: one is inner broker case where the traffic is relatively large and the other one is the border case where the traffic is relatively small. The experiment process is as follows.

A node n_{inn} was chosen randomly in inner brokers and a link l_{inn} was chosen randomly in node n_{inn}. We recorded notifications routed by link l_{inn} 1000 time-steps. The first few

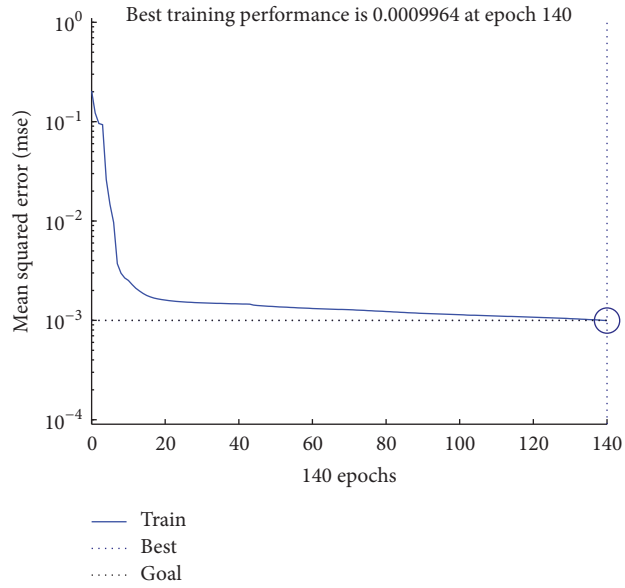

Figure 5: l_{inn} performance.

950 time-steps were used for training the network and the rest were used to test the network. The records were loaded and trained in MATLAB. The training performance figure is shown in Figure 5.

In Figure 5, x-coordinate is the number of the epochs and the y-coordinate is the mse. The figure shows that the mse reached 0.0009964 at iteration of 140 epochs, so the convergence speed of the error is very fast with LM algorithm. The regression figure is shown in Figure 6.

In Figure 6, x-coordinate is the targets and the y-coordinate is the network outputs. The best linear fit is indicated by a dashed line. The perfect fit is indicated by the solid line. In this figure, we cannot find out the dashed line and the solid line because they covered by the data points which composed a line. This means that the fit is very good. The results of the prediction are shown in Figure 7.

In Figure 7, x-coordinate is the number of the time-steps and the y-coordinate is the number of the notifications. The symbol "∘" denotes the actual value and the symbol "+"

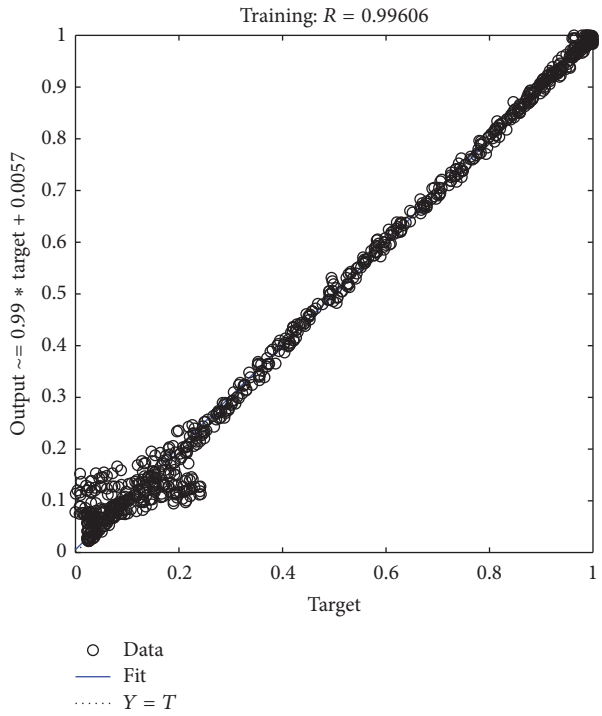

Figure 6: l_{inn} regression.

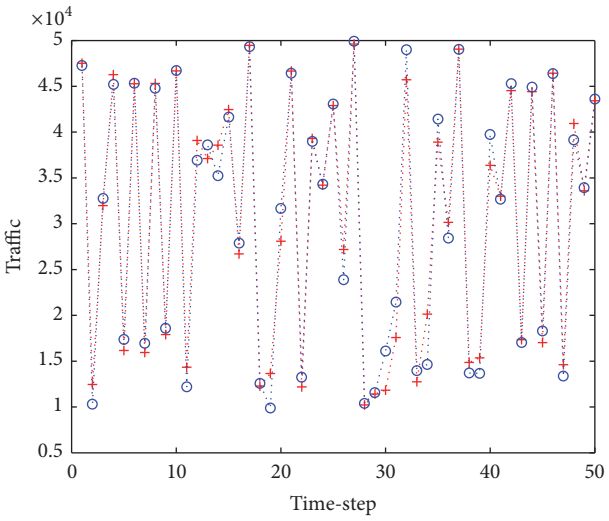

Figure 7: l_{inn} prediction.

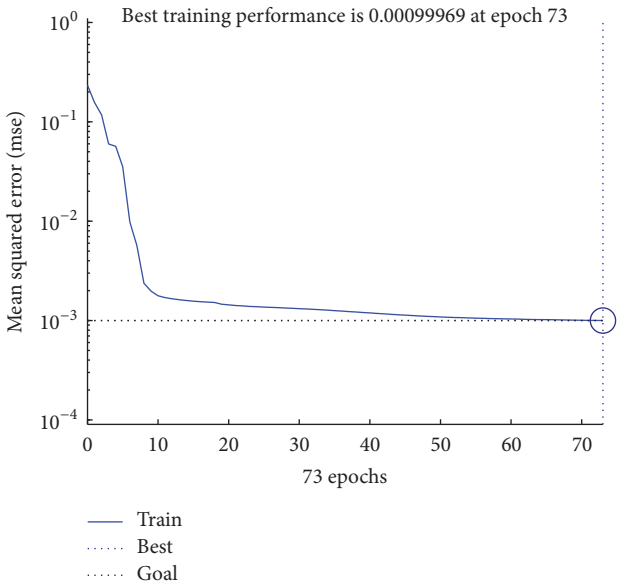

Figure 8: l_{brd} performance.

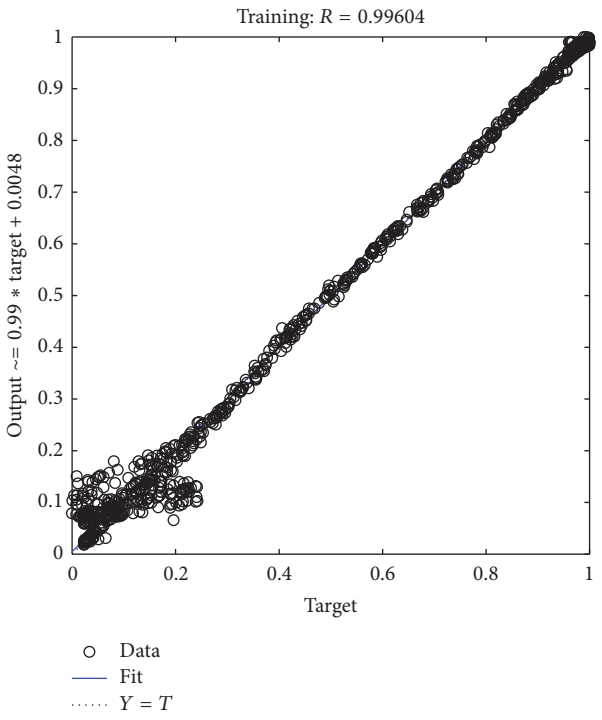

Figure 9: l_{brd} regression.

denotes the predicted value. From the figure, we can see that the predicted values series could well fit the actual ones. Next, border broker case was also given.

A node n_{brd} was chosen randomly in border brokers [18] and a link l_{brd} was chosen randomly in node n_{brd}. The rest of the environment is identical with the previous. The training performance figure is shown in Figure 8.

In Figure 8, the figure shows that the mse reached 0.00099969 at iteration of 73 epochs, so the convergence speed of the error is also very fast in border broker case. The regression figure is shown in Figure 9.

In Figure 9, the fit is also very good in border broker case. The results of the prediction are shown in Figure 10.

In Figure 10, we can see that the predicted values series could well fit the actual ones in border broker case. As a result, we can come to the conclusion that the prediction model proposed in this paper has the ability to predict the traffic.

6.2. Part Two Experiments. The part two experiments are to validate whether the strategy integrated into the prediction

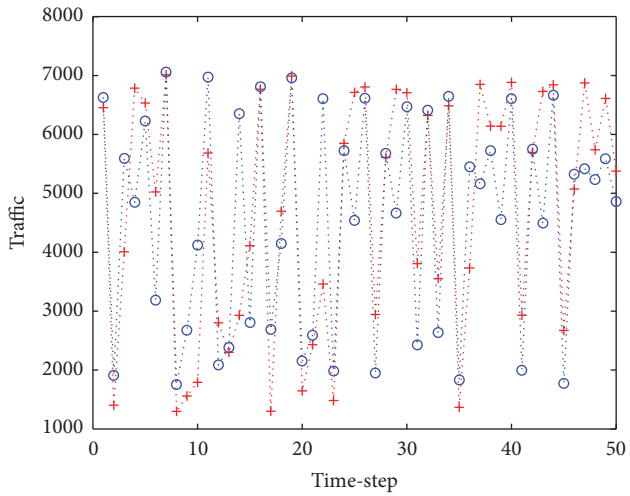

Figure 10: l_{brd} prediction.

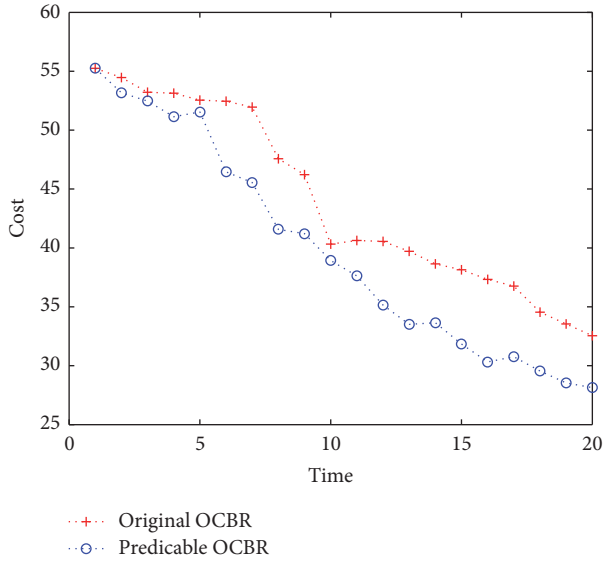

Figure 11: OCBR case.

model has the ability to reduce the traffic cost of the overlay network and to compare with other strategies. The traffic prediction model was implemented in ProtoPeer, and then the experiment analysis procedure was run in the ProtoPeer and the analysis results were recorded. Finally, the results were plotted in MATLAB. We designed two experiments in this part: one is PSOO-FCAT strategy and the other one is OCBR strategy. The strategies integrated into our traffic prediction model will be compared with the original ones. The experiment process and parameters are as follows.

An overlay network topology with 5000 nodes has been generated randomly, and 1000 nodes have been deployed on the brokers randomly. We randomly generated 2000 different types of subscription, 10000 different types of events based on the 1500 subscriptions, 8000 advertisements according to 8000 different types of events, 200 producers, 200 consumers, and 50 pairs of the ⟨producer, consumer⟩ that connected to brokers randomly, each producer published 40 types of events, each consumer subscribed to 8 types of events, and the event is sent every 2 simulation ticks by a producer. The simulation experiment was performed in 100 minutes. The results were recorded by us at every 500 ticks. The average value was calculated as shown in Figure 11.

In Figure 11, the original OCBR and the predicable OCBR cost are both decreased, but predicable OCBR cost decreases sharper than original OCBR, which indicates that both the original OCBR and the predicable OCBR strategy have the ability to reduce the traffic cost of the overlay network and the efficiency of the predicable OCBR strategy is more obvious.

In Figure 12, the original PSOO-FCAT and the predicable PSOO-FCAT cost are both decreased, but predicable PSOO-FCAT cost decreases sharper than original PSOO-FCAT, which indicates that both the original PSOO-FCAT and the predicable OCBR strategy have the ability to reduce the traffic cost of the overlay network and the efficiency of the predicable PSOO-FCAT strategy is more obvious.

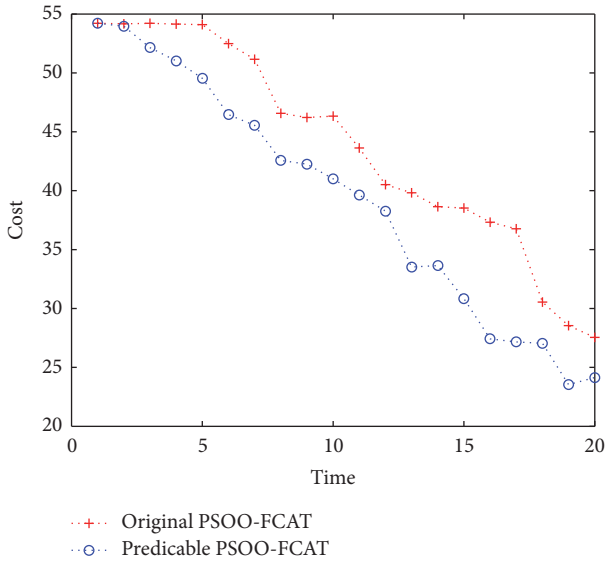

Figure 12: PSOO-FCAT case.

In sum, we can come to the conclusion that the strategy integrated into the prediction model has the ability to reduce the traffic cost of the overlay network.

7. Conclusion

In the paper, we propose a traffic prediction model for the broker in publish/subscribe system, and it uses neural network to predict the traffic of the link. We first prove that the traffic of the link is predictable by chaos theory and introduce our traffic prediction model and the model integration. Finally, the experimental results show that our traffic prediction model could predict the traffic of link in the broker well and the strategy integrated into our traffic

prediction model could reduce the traffic cost of the overlay network efficiently.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was sponsored by the National Science Foundation of China, NSFC no. 61173177.

References

[1] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec, "The many faces of publish/subscribe," *ACM Computing Surveys*, vol. 35, no. 2, pp. 114–131, 2003.

[2] J. Yin, W. Lo, S. Deng, Y. Li, Z. Wu, and N. Xiong, "Colbar: a collaborative location-based regularization framework for QoS prediction," *Information Sciences*, vol. 265, pp. 68–84, 2014.

[3] M. Migliavacca and G. Cugola, "Adapting publish-subscribe routing to traffic demands," in *Proceedings of the Inaugural International Conference on Distributed Event-Based Systems (DEBS '07)*, pp. 91–96, ACM, Ontario, Canada, June 2007.

[4] Y. Yin, S. Aihua, G. Min, X. Yueshen, and W. Shuoping, "QoS prediction for web service recommendation with network location-aware neighbor selection," *International Journal of Software Engineering and Knowledge Engineering*, vol. 26, no. 4, pp. 611–632, 2016.

[5] M. Chi, S. Liu, and C. Hu, "Self-adapting routing overlay network for frequently changing application traffic in content-based publish/subscribe system," *Mathematical Problems in Engineering*, vol. 2014, Article ID 362076, 2014.

[6] M. A. Jaeger, H. Parzyjegla, G. Mühl, and K. Herrmann, "Self-organizing broker topologies for publish/subscribe systems," in *Proceedings of the ACM Symposium on Applied Computing (SAC '07)*, pp. 543–550, ACM, March 2007.

[7] H. R. Maier and G. C. Dandy, "Neural networks for the prediction and forecasting of water resources variables: a review of modelling issues and applications," *Environmental Modelling and Software*, vol. 15, no. 1, pp. 101–124, 2000.

[8] T. Kimoto, K. Asakawa, M. Yoda, and M. Takeoka, "Stock market prediction system with modular neural networks," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN '90)*, pp. 1–6, June 1990.

[9] B. L. Smith and M. J. Demetsky, "Short-term traffic flow prediction: neural network approach," Transportation Research Record 1453, 1994.

[10] K. T. Alligood, T. D. Sauer, and J. A. Yorke, *Chaos*, Springer, New York, NY, USA, 1996.

[11] J. Gleick, *Chaos: Making a New Science*, Random House, 1997.

[12] F. Fernández-Rodríguez, S. Sosvilla-Rivero, and J. Andrada-Félix, "A new test for chaotic dynamics using Lyapunov exponents," Documento de Trabajo 9, 2003.

[13] M. T. Rosenstein, J. J. Collins, and C. J. De Luca, "A practical method for calculating largest Lyapunov exponents from small data sets," *Physica D: Nonlinear Phenomen*, vol. 65, no. 1-2, pp. 117–134, 1993.

[14] H. S. Kim, R. Eykholt, and J. D. Salas, "Nonlinear dynamics, delay times, and embedding windows," *Physica D: Nonlinear Phenomena*, vol. 127, no. 1-2, pp. 48–60, 1999.

[15] H. S. Hippert, C. E. Pedreira, and R. C. Souza, "Neural networks for short-term load forecasting: a review and evaluation," *IEEE Transactions on Power Systems*, vol. 16, no. 1, pp. 44–55, 2001.

[16] C. N. Lu, H.-T. Wu, and S. Vemuri, "Neural network based short term load forecasting," *IEEE Transactions on Power Systems*, vol. 8, no. 1, pp. 336–342, 1993.

[17] W. Galuba, K. Aberer, Z. Despotovic, and W. Kellerer, "ProtoPeer: a P2P toolkit bridging the gap between simulation and live deployement," in *Proceedings of the 2nd International Conference on Simulation Tools and Techniques*, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), Rome, Italy, March 2009.

[18] G. Mühl, L. Fiege, and P. Pietzuch, *Distributed Event-Based Systems*, vol. 1, Springer, Heidelberg, Germany, 2006.