Int. J. Electron. Commun. (AEÜ) 65 (2011) 75–86

Contents lists available at ScienceDirect

Int. J. Electron. Commun. (AEÜ)

journal homepage: www.elsevier.de/aeue

Accurate estimation of large-scale IP traffic matrix

Dingde Jiang [a,b,c], Xingwei Wang [a,*], Lei Guo [a], Haizhuan Ni [a], Zhenhua Chen [a]

[a] College of Information Science and Engineering, Northeastern University, Shenyang 110004, China
[b] State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China
[c] Key Lab of Broadband Optical Fiber Transmission and Communication Networks, University of Electronic Science and Technology of China, Chengdu 610054, China

ARTICLE INFO

Article history:
Received 22 April 2008
Accepted 24 December 2009

Keywords:
Traffic matrix estimation
Backpropagation neural network
Inverse problem
Large-scale network

ABSTRACT

Traffic matrix (TM) estimation, which is an interesting and important research topic at present, is used to conduct network management, traffic detecting, provisioning and so on. However, because of inherent characteristics in the IP network, especially in the large-scale IP network, TM estimation itself is highly under-constrained, and so it is an very ill-posed problem. how fast and accurately to attain large-scale IP TM estimation is a challenge. Based on back-propagation neural network (BPNN), this paper proposes a novel method for large-scale IP TM estimation, called BPNN TM estimation (BPTME). In contrast to previous methods, BPTME can easily avoid the complex mathematical computation so that we can quickly estimate the TM. The model of large-scale IP TM estimation built on top of BPNN, whose outputs can sufficiently represent TM's spatial-temporal correlations, ensures that we can attain an accurate estimation result. Finally, we use the real data from the Abilene Network to validate and evaluate BPTME. Simulation results show that BPTME not only improves remarkably and holds better robustness, but it can also make more accurate estimation of large-scale IP TM and track quickly its dynamics.

1. Introduction

Traffic matrix (TM) estimation is an interesting and important research topic now. Network operators usually use TM estimation to conduct network management, load balancing, traffic detecting, provisioning and so on. TM consists of all origin-destination (OD) flows in the network. It reflects, from a global aspect, how the traffic flows in the network. Moreover, it is also a key input of traffic engineering. It is very important for network operators to accurately attain TM in the network, especially in the large-scale IP network. However, as commented in [1], it is very difficult and even impossible to directly attain TM in the large-scale IP network.

How fast and accurately to attain large-scale IP TM estimation is a challenge. The traffic in a network complies with certain routing strategy (denoted by routing matrix) when it flows through the network, and link loads are the aggregated traffic that consists of all OD flows traversing this link. Assume that there are n nodes and L links in the large-scale IP network. Afterwards, there exist $N=n^2$ OD flows. Thus large-scale IP TM estimation problem can be denoted as follows:

$$Y = AX, \tag{1}$$

where L-vector Y represents link loads, N-vector X TM, and A $L \times N$ routing matrix whose element is equal 1 if OD flow j traverses link i or zero otherwise [2–10]. This is an inverse problem. However, because the inequation $L \ll N$ always exists in a large-scale IP network, this problem is highly under-constrained and so it is also ill-posed. Hence, this is a highly ill-posed inverse problem which is very difficult to solve.

To solve the problem, based on back-propagation neural network (BPNN), we propose a novel method, called BPNN TM estimation (BPTME), from a new perspective. BPTME uses BPNN to model large-scale IP TM estimation problem, and then exploits iterative proportional fitting procedure (IPFP) to satisfy TM estimation with the constraints of (1). BPNN is studied extensively and often applied to signal processing, biomedicine, modeling and so on [11–13]. It is a powerful modeling tool. Hence, BPNN is usually suited for modeling the unknown and complex systems, linear and nonlinear. A useful advantage of using BPNN to model the systems is that one can avoid the complex mathematical computation, and especially when one cannot express the modeled system by the analytical expressions, it is more effective to use BPNN to model the systems. The parallel architectures and universally approximate properties [14] of BPNN make it suited for handling the problem of the large-scale IP TM estimation because BPNN can fast make the accurate prediction of the large-scale problems once one correctly constructs BPNN and successfully trains it. As mentioned in [15], for large-scale IP TM, Eq. (1) represents a highly

* Corresponding author.
E-mail addresses: jiangdingde@ise.neu.edu.cn (D. Jiang), wangxingwei@ise.neu.edu.cn (X. Wang).

1434-8411/$ - see front matter © 2010 Elsevier GmbH. All rights reserved.
doi:10.1016/j.aeue.2010.02.008

under-constrained system because the number of OD flows in a network is much larger than the number of links. Thus there are infinite solutions to (1). How one finds a meaning solution is a difficult problem. As noted above, BPNN is a good choose to the large-scale IP TM estimation. We use the real data [16] from the Abilene network to validate BPTME. Simulation results show that BPTME improves remarkably and holds better robustness, and it can also make more accurate estimation of large-scale IP TM and track quickly its dynamics.

1.1. Related work

Some papers have investigated this problem and proposed some solutions. Vardi [5], Cao et al. [6], and Tebaldi et al. [17] used the statistical inference method to estimate TM only over the local area network. As mentioned in [3], these methods are sensitive to the prior, and estimation errors are larger. Medina et al. [4] showed that the basic assumptions based on the statistical models are not justified, and they also showed that, when their underlying assumptions are violated, the estimated results are bad. Furthermore, because these methods need to perform the complex mathematical computation, it takes some time to estimate TM. Hence, it is difficult to scale these methods to large-scale networks. Zhang et al. [2,7] discussed the problem of large-scale IP TM estimation by introducing the gravity model. Though, as mentioned in [3], their method partially reduced the sensitivity to the prior, it also has the larger errors, because it only considered the spatial correlations among the OD flows. Nucci et al. [18] proposed the method that changed the under-constrained problem into the full rank one by changing the routing and then taking new SNMP measurements under this new routing map. Similarly, Soul et al. [19] presented a heuristic algorithm to compute the routing needed in order to obtain a full rank problem. Papagiannaki et al. [1] proposed a data-driven method that depends on measurements alone to obtain TM, without using the routing matrix and performing the inference, but based on measuring TM directly. Lakhina et al. [20] used the principal component analysis to solve the TM estimation problem. Soule et al. [21] introduced the Kalman filtering into TM estimation. However, all the methods need to establish mathematical model about OD flows and perform the statistical inference, or combine the direct measurement of partial OD flows to infer TM. Thus, they need the complex mathematical computations. Different from the above methods, BTTME uses BPNN to model large-scale IP TM estimation problem. Because BPNN is a powerful modeling tool, we avoid complex mathematical computation and can attain the accurate estimation results.

Liang et al. [10], based on game theory, proposed a fast lightweight approach to OD flow estimation. Bermolen et al. [22] derived analytically the Fisher information matrix under the second moment statistics with the functional mean–variance relationship and then obtained the Cramer–Rao lower bound for the variance of TM estimator. By this bound, they could attain TM estimation. Juva [23] studied the sensitivity of the estimation accuracy to those underlying assumptions in the case of the gravity model based and second moment methods. They showed that if the assumptions hold, the gravity model based methods are more accurate, or their accuracy declines faster than that of the second moment methods. However, BPTME does not make any assumption about OD flows. It only models TM estimation problem with BPNN. Due to the capacity of learning and generalizing of BPNN, BPTME is not sensitive to the assumption about OD flows and it is also robust to noise. Furthermore, because of the parallel structure of BPNN, BPTME is a lightweight method and can fast predict large-scale IP TM.

1.2. Our contributions

In this paper, we propose a novel method for large-scale IP TM estimation, called BPTME. Previous methods all make the assumptions about OD flows. This leads to the results that if these assumptions hold, the estimation is accurate, or it is highly bad otherwise. BPTME avoids these problems. The key idea of BPTME is first to introduce the temporal correlations of link loads into input data, then to use BPNN to model large-scale IP traffic matrix estimation problem, instead of modeling OD flows or making the assumptions about OD flows, and finally to combine IPFP to fast and accurately predict large-scale IP traffic matrix. Hence, BPTME successfully avoids the sensitivity to the assumptions about OD flows. Secondly, complex mathematical computations always exist in previous methods. BPTME do avoid subtly these problems by training BPNN. It only needs the simple input–output data pairs, instead of complex mathematical computations, to establish the estimation model and then infer TM. Because of parallel structure, training and predicting is very fast. This makes BPTME being a lightweight method of TM estimation. Finally, to fast and accurately predict TM, we propose a multi-input and multi-output estimation model. Our model can be extended to the larger network. Moreover, because of the inhere properties of BPNN, the outputs of our model can well capture the spatio-temporal properties of OD flows so that BPTME can more accurately predict large-scale IP TM. To the best of our knowledge, this is the first time to apply BPNN to handle the large-scale IP TM estimation problem.

We conduct extensive simulation experiments on the Abilene network topologies to demonstrate the performance improvement of BPTME, using the real data. Previous methods can often only predict accurately the larger OD flows. Simulation results show that BPTME cannot only predict accurately the larger OD flows, but also can infer accurately the smaller OD flows. Moreover, BPTME holds the lower spatial and temporal relative errors. It is not sensitive to noise and holds the strong robustness.

The rest of this paper is organized as follows. Section 2 describes TM, BPNN, and its training process. Section 3 derives our method, models large-scale IP TM estimation, and proposes two algorithms. Section 4 presents simulation results and performance analysis. We conclude our work in Section 5.

2. Problem statement

2.1. Traffic matrix

TM describes the traffic of flow between all source nodes and destination nodes, namely egress nodes and ingress nodes. As mentioned in [2,3], TM is time sequence. It evolves over time and holds the dynamics during the observed intervals. Previous studies show that, in the IP backbone network, OD flows, namely elements in TM, hold strongly daily pattern, and even weekly and monthly pattern. This suggests that OD flows in the IP backbone network hold the temporal correlations. Soule et al. [3] showed that TM also holds spatial correlations. They found that when the spatial and temporal correlations of TM are considered, estimation results are more accurate. However, since TM estimation is a highly ill-posed inverse problem, it is very difficult to solve it. Previous methods introduce additional information or assumptions about TM to solve this estimation problem. BPTME uses powerful modeling capacity of BPNN to build large-scale IP TM estimation model from link loads to TM.

To facilitate the following discussion, we bring forth the relevant notational convention used for this paper. As mentioned in introduction, assume that there are the n nodes and L links in a

large-scale IP network, with the result that there exist $N=n^2$ OD flows. At a particular time t, TM and link traffic are denoted as $X(t)$ and $Y(t)$, respectively. Hence, large-scale IP traffic matrix estimation problem can be denoted as follows:

$$Y(t)=AX(t), \tag{2}$$

where A denotes $L\times N$ routing matrix. In a large-scale IP network, TM and link loads evolve over time, and are satisfied with the above linear relationship.

2.2. BPNN for traffic matrix estimation

BPNN holds a highly parallel structure and is a multilayer feedforward network. It is a universally approximator, and can asymptotically approach any function [14]. For large-scale IP TM estimation, Fig. 1 plots the architecture representation of BPNN with three parts, namely input layer, hidden layer, and output layer, where $Y_U=(Y_1^U,Y_2^U,\ldots,Y_V^U)$ (V is the number of inputs of BPNN.) represents the V-vector inputs of BPNN, $Y_k^i=(y_1^i,y_2^i,\ldots,y_k^i)$ ($k=M_1,M_2,\ldots,M_U$, and M_i is the number of neural nodes in the i th hidden layer, and $i=1,2,\ldots,U$, and U is the number of hidden layers.) M_i-vector output of the i th hidden layer, "*tansig*" tansigmoid activation function, $X_N^p=(\hat{x}_1^p,\hat{x}_2^p,\ldots,\hat{x}_N^p)$ (N is the number of OD flows in the network, and p denotes the result before data posttreating.) N-vector output of output layer, and "*line*" linear activation function.

From Fig. 1, the output of hidden layer node is

$$\begin{cases} y_j^i=\varphi_{ji}(n_j^i) \\ n_j^i=\sum_{k=1}^{M_i} W_{jk}^i y_k^{i-1}+b_j^i, \end{cases} \tag{3}$$

where $i=1,2,\ldots,U$; $j=1,2,\ldots,M_i$; and y_j^i, W_{jk}^i, $\varphi_{ji}(.)$, and b_j^i denote the output of the j th neural node, the $j\times k$ th network forward connection weight value, the j th tansigmoid activation function, and the j th bias input in the i th hidden layer, respectively.

Likewise, the output of output layer node is

$$\begin{cases} \hat{x}_i^p=f_i(n_i) \\ n_i=\sum_{j=1}^{N} W_{ij}^{U+1} y_j^U+b_i, \end{cases} \tag{4}$$

where $i=1,2,\ldots,N$; y_i^U denotes the i th output of the final hidden layer; $f_i(.)$, W_{ij}^{U+1}, b_i, and $\hat{x}_i^p$ represent the i th linear activation function, the $i\times j$ th network forward connection weight value, the i th bias input, and the i th output in output layer, respectively.

At the time t, assume that the inputs and outputs of BPNN are $Y_U(t)=(y_1^U(t),y_2^U(t),\ldots,y_V^U(t))^T$ and $X^p(t)=(x_1^p(t),x_2^p(t),\ldots,x_N^p(t))^T$, respectively, and the outputs of the i th hidden layer are $Y_k^i(t)=(y_1^U(t),y_2^U(t),\ldots,y_k^U(t))^T$. Then Eqs. (3) and (4) can be denoted as follows, respectively,

$$\begin{cases} y_j^i(t)=\varphi_{ji}(n_j^i(t)) \\ n_j^i(t)=\sum_{k=1}^{M_i} W_{jk} y_k^{i-1}(t)+b_j^i, \end{cases} \tag{5}$$

where $i=1,2,\ldots,U$ and $j=1,2,\ldots,M_i$.

$$\begin{cases} \hat{x}_i^p(t)=f_i(n_i(t)) \\ n_i(t)=\sum_{j=1}^{N} W_{ij}^{U+1} y_j^U(t)+b_i, \end{cases} \tag{6}$$

where $i=1,2,\ldots,N$.

From (5) and (6), get the below mapping from input $Y^0(t)$ to output $X^p(t)$ of BPNN:

$$\begin{cases} \hat{x}_1^p(t)=F_1(Y^0(t)) \\ \hat{x}_2^p(t)=F_2(Y^0(t)) \\ \ldots, \\ \hat{x}_N^p(t)=F_1(Y^0(t)), \end{cases} \tag{7}$$

where $F_i(.)$ ($i=1,2,\ldots,N$) denotes the mapping from $Y^0(t)$ to $\hat{x}_i^p(t)$, namely from R^V to 1.

Thus, according to (7), get the following (8):

$$\hat{X}^p(t)=F(Y^0(t)), \tag{8}$$

where $F(.)=(F_1(.),F_2(.),\ldots,F_N(.))^T$. Eq. (8) clearly shows that BPNN in Fig. 1 establishes the mapping from input vector space R^V to output vector space R^N. Hence, by training appropriately BPNN, one can establish the accurate mapping from R^V to R^N and estimate accurately large-scale IP TM.

2.3. Training of BPNN

The training process of BPNN in Fig. 1 is simple and quick because BPNN is only a multilayer feed-forward network. Back-propagation algorithm is often used for updating and adjusting its weights. This algorithm can be performed quickly [24,25]. First, the input and output data pairs are pretreated and passed to the BPNN directly. Then back-propagation algorithm and other kinds of optimal algorithms are used to adjust and update the weights of BPNN. Hence, BPNN can be trained quickly and successfully with the input and output data pairs.

Because BPNN can establish the complex map from inputs to outputs and reflect the dynamic behavior of the handled systems, we use the BPNN to complete the map from $Y(t)$ to $X(t)$ of the time-varying system denoted by (2). Thus, by this way, we can avoid the complex computation that is used for obtaining the inversion of the matrix in (2). Once BPNN is trained successfully by the input–output data pairs, we can well build the model of large-scale IP TM estimation, which reflects the mapping from inputs to outputs. As discussed in the following, we use this method to be able to solve well the under-constrained problem denoted by (1) and (2).

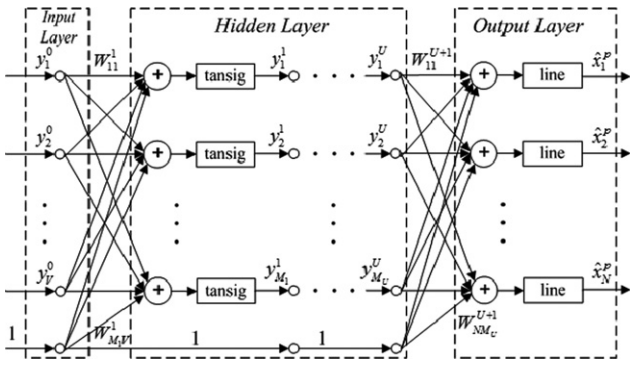


Fig. 1. Architecture representation of BPNN with input layer, hidden layer, and output layer.

3. Proposed method: BPTME

In this section, we will discuss the details of the proposed BPTME method. We first briefly present the basic idea of BPTME. Then, we describe the details of TM estimation modeling, which is followed by proposing how fast and accurately to predict large-scale IP TM by this model. Finally, the two algorithms are given, which is followed by a complete BPTME method.

BPTME includes the training and predicting phases. First, we introduce the temporal correlations of link loads into input data.

Then we train the BPNN to complete the modeling of large-scale IP TM estimation. Finally, by combining IPFP with the established model, we can fast and accurately attain large-scale IP TM estimation. BPTME avoids complex mathematical computations and the sensitivity to the assumptions about OD flows.

3.1. Modeling of traffic matrix estimation

Since traffic matrix estimation is a highly ill-posed inverse problem, it is very difficult to solve it. To estimate accurately TM, various kinds of models about TM are built to add the side information. However, previous studies show that it is very difficult to accurately build the model of TM because TM has many characteristics, e.g., daily pattern, weekly pattern, spatial correlations, temporal correlations and so on. To use a certain model to capture these characteristics is difficult. Moreover, even though the model is successfully established, one will meet the complex mathematical computation and the identifiable problem of the established model.

To overcome these problems, we use BPNN to model large-scale IP TM estimation problem because BPNN is a powerful modeling tool. To ensure the estimation accuracy and capture accurately the properties of TM estimation problem, the spatio-temporal correlations of link loads are introduced into the input data of the model. Furthermore, to estimate fast and accurate TM, all OD flows are simultaneously estimated. Hence, we propose a multi-input and multi-output large-scale IP TM estimation model.

Architecture of multi-input and multi-output estimation model is plotted in Fig. 2, where $Y(t)$ (its dimensions are L) and $Y(t-i)$ ($i=1,2,\ldots,k$, k is the number of the time points (before the current time point t) to add) is the input vector of our model, and $\hat{X}(t)=(\hat{x}_1(t),\hat{x}_2(t),\ldots,\hat{x}_N(t))^T$ (N is the number of OD flows in a network) is the output vector. The data pretreating unit in Fig. 2 transforms the input data into those suited for BPNN handling. By this way, it is able to speed up the training process and improve the performance.

Multi-input and multi-output model denoted by Fig. 2, includes two parts, namely training and predicting. Training part is used for modeling large-scale IP TM estimation problem. Predicting part is used for estimating large-scale TM by the established model. According to Fig. 2, attain the following equation

$$U(t)=(Y(t),Y(t-1),\ldots,Y(t-k))^T, \tag{9}$$

where $Y(t)=(y_1(t),y_2(t),\ldots,y_L(t))^T$ denotes link loads at time point t, and $U(t)$ denotes the inputs of our model. The inputs $U(t)$ consist of link loads of the current time point t and the last k time points.

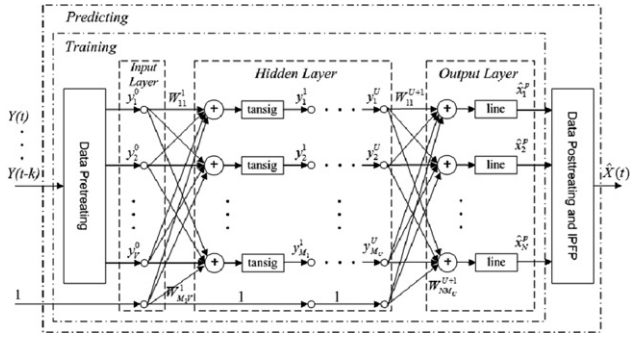


Fig. 2. Architecture representation of multi-input and multi-output model used for training and predicting large-scale IP TM, based on BPNN and combined with IPFP, including the training and predicting phases, with last several time slot link loads introduced into input data.

This shows that we introduce the spatio-temporal correlations of link loads into the inputs of our model.

Thus, according to the training frame in Fig. 2, and Eqs. (8) and (9), attain the below equation:

$$\begin{cases} \hat{X}^p(t)=F(Y^0(t)), \\ Y^0(t)=G(U(t)), \\ U(t)=(Y(t), \quad Y(t-1),\ldots, \quad Y(t-k))^T, \end{cases} \tag{10}$$

where G denotes the data pretreating process, $Y^0(t)=(y_1^0(t),y_2^0(t),\ldots,y_V^0(t))^T$ the inputs of BPNN. Eq. (10) represents the established model by training BPNN. From (10), get the below mapping:

$$\begin{cases} \hat{X}^p(t)=H(U(t)), \\ U(t)=(Y(t), \quad Y(t-1),\ldots, \quad Y(t-k))^T, \end{cases} \tag{11}$$

where $Xp(t)=(\hat{x}_1^p(t),\hat{x}_2^p(t),\ldots,\hat{x}_N^p(t))T$; $\hat{x}_i^p(t)=h_i(U(t))$ ($i=1,2,\ldots,N$); $H=(h_1,h_2,\ldots,h_N)^T$ denotes the mapping from R^V to R^N. Hence, the training frame in Fig. 2 denotes the mapping represented by (11).

In the training frame in Fig. 2, we exploit one step secant (OSS) algorithm, which is an improved back-propagation algorithm, to train BPNN. In contrast to other training algorithm, OSS algorithm requires less storage and computation per epoch, and can converge faster. Hence, for very large BPNN, e.g., corresponding to large-scale IP TM estimation problem, to use OSS algorithm to train BPNN is better [24]. Hence, when training BPNN in terms of the training frame in Fig. 2, one can quickly establish large-scale IP TM estimation model. Once this estimation model is established, it is fast and accurate to predict large-scale IP TM. Up to now, the model of large-scale IP TM estimation is successfully established.

3.2. Predicting of traffic matrix

Generally, once successfully trained by the input–output data pairs, BPNN can describe the characteristics of TM estimation problem, and build appropriately large-scale IP TM estimation model. Thus, we can quickly complete the modeling about large-scale IP TM estimation problem. According to the established model, we can predict TM at other time point. Because BPNN holds the powerful capability of learning and generalizing, it can be used to accurately predict TM. Though large-scale IP TM holds the complex characteristics and the estimation problem about it is significantly difficult, BPNN can well capture the properties of TM estimation problem by learning, and simultaneously can make the generalization and conclusion, as long as the data set used for training is sufficient enough. As a result, BPNN can also describe the characteristics that the data set used for training do not reflect, and appropriately predict the corresponding OD flows.

As noted above, TM must be satisfied with the constraints denoted by (2). Moreover, each element of TM, namely OD flow, must be nonnegative. However, the estimations of TM are not often satisfied with the constraints. To solve this problem, we propose the data posttreating process and then use IPFP to make the estimated results satisfied with the constraints. Hence, the predicting frame in Fig. 3 denotes the predicting process of TM. The transformation from the estimation $\hat{x}^p(t)$ to $\hat{x}(t)$ is

$$\begin{cases} \hat{X}(t)=Q(\hat{X}p(t)) \\ Y(t)=A\hat{X}(t) \\ \hat{x}_i(t)\geq 0, \quad i=1,2,\ldots, \quad N, \end{cases} \tag{12}$$

where Q denotes the data posttreating and IPFP process. According to (11) and (12), the predicting frame can be denoted

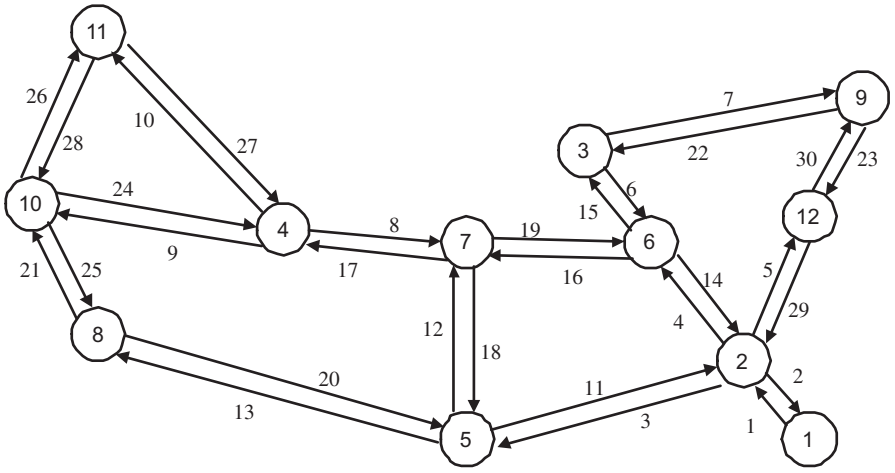


Fig. 3. Abilene network topologies for simulation.

as follows:

$$\begin{cases} \hat{X}(t) = P(U(t)), \\ U(t) = (Y(t), Y(t-1), \ldots, Y(t-k))^T, \\ Y(t) = A\hat{X}(t), \\ Y(t-1) = A\hat{X}(t-1), \\ \ldots, \\ y(t-k) = A\hat{X}(t-k), \\ \hat{x}_i(t-j) \geq 0, \quad i = 1, \ldots, N; \ j = 0, 1, \ldots, k \end{cases} \tag{13}$$

where P denotes the mapping from $U(t)$ to $\hat{X}(t)$. Eq. (14) denotes the resulting estimation results.

3.3. Algorithm

In this subsection, we proposed two algorithms, namely Algorithms 3.3.1 and 3.3.2, and a complete BPTME method. Algorithm 3.3.1 is used for the training process, and Algorithm 3.3.2 is used for the predicting process.

Algorithm 3.3.1. *Step* 1. Initialize the network model denoted by Fig. 2. Set the error bound δ, total iterative steps T, iterative variable $m=0$, the number k(of the last time points to add).

Step 2. Construct the input of the model at the current time t, namely $U(t) = (Y(t), Y(t-1), \ldots, Y(t-k))^T$.

Step 3. Pass the input-output data pairs to the model, and make the data pretreatment. According to (11), get the output $\hat{X}^p(t)$ of the training frame in Fig. 2.

Step 4. Compute the gradient of BPNN by using the OSS algorithm. And then update the weights of BPNN.

Step 5. Calculate the total estimation error $\varepsilon = \|Y(t) - A\hat{X}^p(t)\|$ ($\|.\|_2$ denotes the L_2 norm). If $\varepsilon < \delta$ or $k > T$, then save the network weights to the file and exit the training, or set $m = m+1$ and go back to Step 2.

Algorithm 3.3.2. *Step* 1. By Algorithm 3.3.1, attain the weights of BPNN and initialize the model denoted by Fig. 2.

Step 2. Construct the input of the model at the current time t, namely $U(t) = (Y(t), Y(t-1), \ldots, Y(t-k))^T$.

Step 3. Present the input data to the model in Fig. 2, and make the data pretreatment.

Step 4. According to the predicting frame in Fig. 2, get the output $\hat{X}^p(t)$ of BPNN. Make the data posttreatment. And perform the IPFP process. By (13), get the resulting estimation $\hat{X}(x)$ of TM.

Step 5. If the predicting process is over, then save the estimations to the file and exit, or go back to Step 2.

As discussed above, BPTME includes two phases, namely training phase and predicting phase. Hence, we here summarized the complete BPTME method as follows.

Step 1. Initialize the network model denoted by Fig. 2.

Step 2. According to Algorithm 3.3.1, train quickly BPNN and update its weights with the OSS algorithm, and then establish large-scale IP TM estimation model.

Step 3. According to Algorithm 3.3.2, make the fast prediction of TM, and combine the IPFP process to improve the accuracy of the resulting estimation results.

4. Simulation results

In this section, we conduct a series of simulations to study the performance of BPTME. We compare BPTME with TomoGravity. TomoGravity has so far reported as the accurate method of large-scale IP TM estimation, and is validated by the real data [2]. Moreover, it is now used for the practical network engineering.

We use the real data [16] from the Abilene network to simulate the performance of two methods, namely analyzing TM tracking, estimation errors, and their robustness. Fig. 3 shows the Abilene network topology for simulation, where there are 12 nodes, 30 inner links and 24 outside links. In Fig. 3, the number in the circle denotes the serial number of routers, the number beside the directed line denotes the serial number of the inner links, and there are two outside links in each router, but the outside links are not plotted. According to the practical simulation, we select the eleven related time points as the inputs of the multi-input and multi-output estimation model presented in Section 3, and in BPNN select one input layer with the 11×54 inputs, two hidden layers with each hidden layer holding one neural node, one linear output layer with the 144 outputs. By training the multi-input and multi-output estimation model, we can fast and accurately estimate TM in large-scale IP network showed in Fig. 3.

The following discusses the performance of two methods. The data of the 24 files given by [16] are all simulated with two methods. The simulation results show that, in contrast to TomoGravity, BPTME can more accurately track TM, and hold the smaller estimation errors and the stronger robustness. Thus we, from a new perspective, use a novel method to solve the problem that large-scale IP TM estimation is very difficult. Here we use the data in [16], from 29th in May to 9th in July, 2004

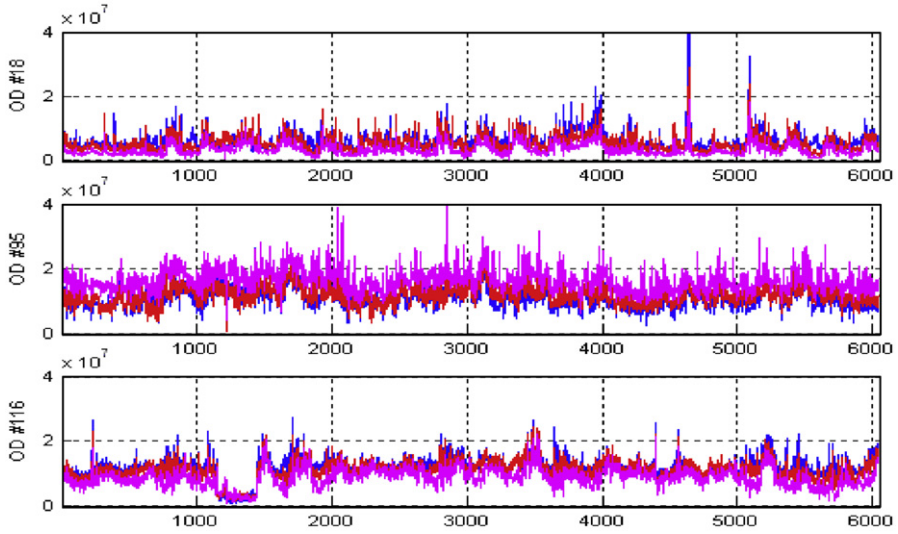


Fig. 4. Estimation results about OD flows 18th, 95th, and 116th in Abilene network, for three weeks. True in blue, estimation with TomoGravity in magenta, estimation with BPTME in red. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

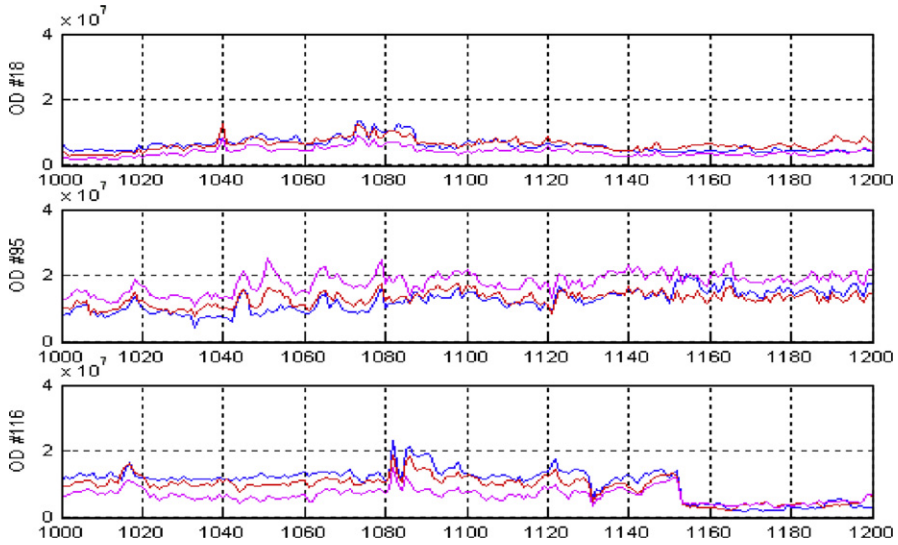


Fig. 5. Corresponding to Fig. 4, from time slot 1000 to 1200. True in blue, estimation with TomoGravity in magenta, estimation with BPTME in red. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

(the 6-week data in all), to simulate two methods. The data of the first three weeks, namely 6048 time points, are used for training the multi-input and multi-output estimation model. The rest are used for the consecutive estimation of TM. Figs. 4, 5, 6, 7, 8, and 9 show TM tracking, and Figs. 10 and 11 denote the estimation errors.

4.1. Tracking dynamic changes

Figs. 4, 5, 6, 7, 8, and 9 show TM tracking, with true value in blue, estimation with TomoGravity in magenta, and estimation with BPTME in red. Fig. 4 shows TM tracking for three weeks. From Fig. 4, we can see that TomoGravity and BPTME can well track the dynamics of large-scale IP TM. However, BPTME is more accurate than TomoGravity. Fig. 4(a) shows that though TM holds the burst changes, TomoGravity and BPTME can accurately predict this type of changes. But the estimation results with TomoGravity divert from the average values of TM, and hold the larger estimation errors. BPTME can accurately predict the burst changes and its estimation errors are lower. Fig. 4(b) shows that when TM holds the period properties, two methods can well track the tendency of changes. In contrast to the overestimations yielded by TomoGravity, the estimation results of BPTME are reasonably consistent to the real TM, and estimation errors are much lower. Fig. 4(c) indicates that when dramatic and burst changes exist in TM, two methods can fairly accurately capture these changes. Burst changes happen at time points 232, 1711, 4391 and so on. Dramatic changes take place from time points 1152 to 1153, and from time points 1440 to 1441. Though two methods can immediately follow these changes, TomoGravity yields the underestimations, and BPTME can make the more accurate estimations. Thus, once our multi-input and multi-output model is successfully trained, BPTME can more accurately estimate large-scale IP TM than TomoGravity.

More importantly, we only use the data for three weeks to train our multi-input and multi-output model, but we can fairly precisely predict TM for three weeks or even longer time. We ever use the data for three weeks to train our model, and fast and accurately predict TM for six weeks. Hence, BPTME holds the fairly strong robustness. Once our model is built, we can

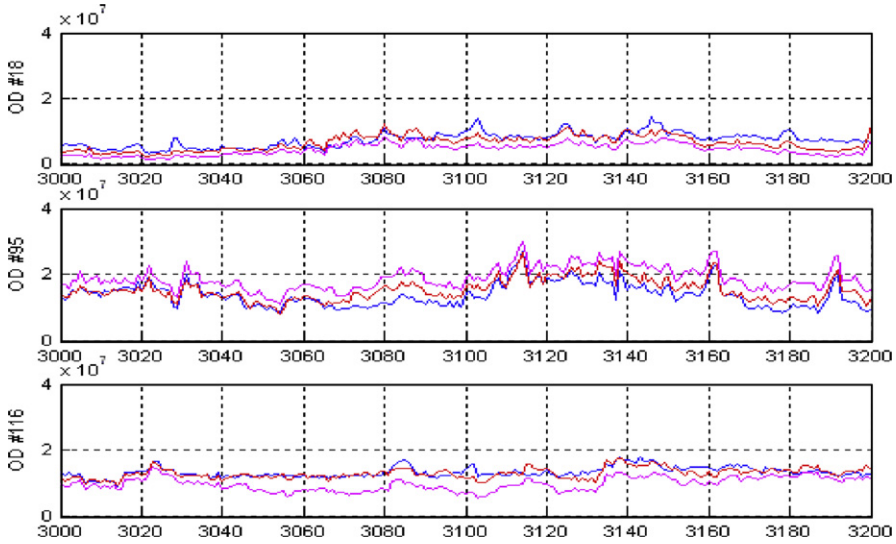


Fig. 6. Corresponding to Fig. 4, from time slot 3000 to 3200. True in blue, estimation with TomoGravity in magenta, estimation with BPTME in red. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

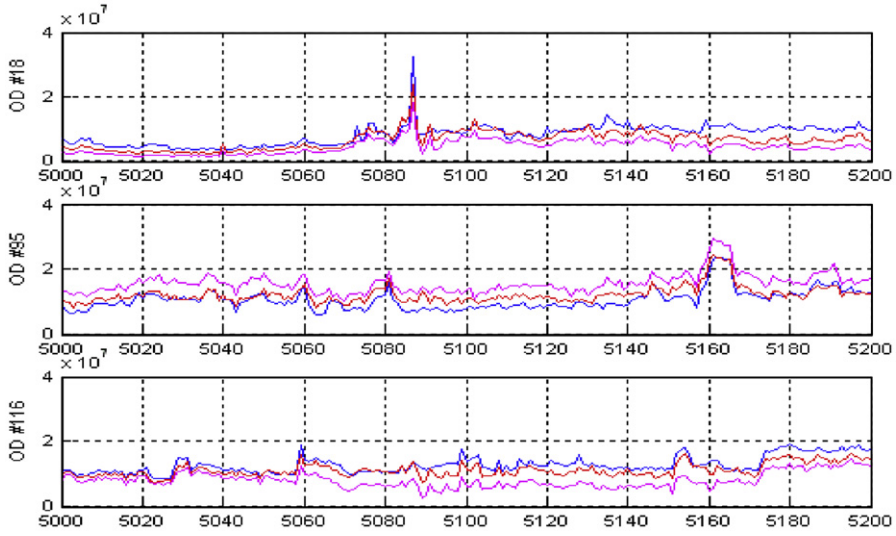


Fig. 7. Corresponding to Fig. 4, from time slot 5000 to 5200. True in blue, estimation with TomoGravity in magenta, estimation with BPTME in red. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

accurately make the long prediction of TM. Fig. 5 denotes the estimations from time points 1000 to 1200 in more details. Fig. 5(a), (b), and (c) show that beginning with the 1000 time points after training is over, BPTME is still more accurate than TomoGravity. Fig. 6 enlarges the estimations from time points 3000 to 3200. Fig. 6(a)–(c) indicate that, though the time point to estimate is 3000 time points far from the time point which training is over at, BPTME holds the considerable accuracy to TM estimations, and the estimation results of BPTME is still closer to the real value than those of TomoGravity. Fig. 7 plots the estimation results from time points 5000 to 5200. As noted in Figs. 5 and 6, Fig. 7(a), (b), and (c) state that, 5000 time points far from the termination for training, BPTME can more accurately predict large-scale IP TM than TomoGravity. Fig. 8 elaborates the estimations of two methods. Form a long scope, Fig. 8 shows that, though burst changes, e.g., Fig. 8(a), or period changes, e.g., Fig. 8(b), or slow changes, e.g., Fig. 8(c) exist in TM to estimate, two methods can capture these changes, whereas the estimations of BPTME is much better than ones of TomoGravity. Fig. 9 plots the larger burst traffic, e.g., Fig. 9(a), and the traffic of dramatic changes, e.g., Fig. 9(b), in more details. Fig. 9 shows that, in the two cases, BPTME can also make the more accurate prediction of TM than TomoGravity.

Hence, BPTME can attain the more accurate estimation results than TomoGravity, and it can also predict TM for long. As discussed in [3], because the gravity model captures the spatial correlations of TM, TomoGravity can attain the accurate estimations and track the dynamics of OD flows, but this method still makes the estimations holding the larger errors. However, because BPNN is a powerful modeling tool, the outputs of our multi-input and multi-output model based on BPNN can well describe the spatio-temporal correlations of TM. Thus BPTME can more accurately predict TM and even make the longer prediction.

4.2. Estimation errors

In this subsection, we discuss estimation errors of two methods, including spatial relative error (SRE) and temporal relative error (TRE). SRE denotes the estimation errors of each OD

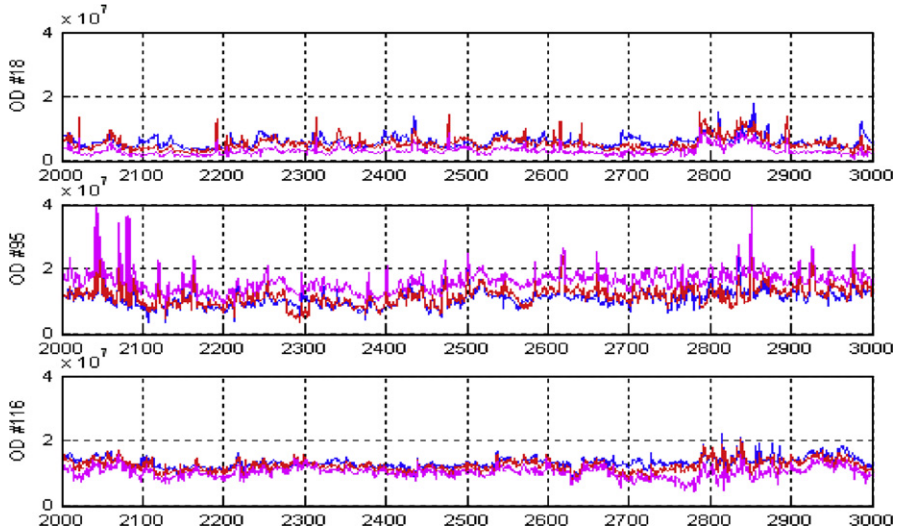


Fig. 8. Corresponding to Fig. 4, from time slot 2000 to 3000. True in blue, estimation with TomoGravity in magenta, estimation with BPTME in red. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

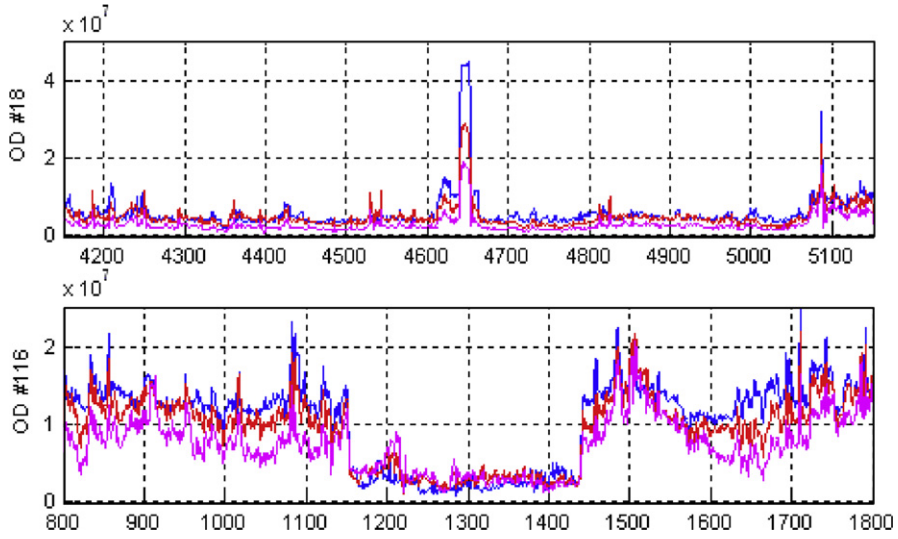


Fig. 9. Burst traffic in OD flow 18th, and dramatic change in OD flow 116th. True in blue, estimation with TomoGravity in magenta, estimation with BPTME in red. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

flow, which reflect the relationships between spatial estimation errors and OD flows, whereas TRE shows the estimation errors at each time point, which represent how estimation errors evolve over time. We also use the cumulative distribution function (CDF) of spatial or temporal relative errors, for short CSRE and CTRE, respectively, to evaluate the estimation errors. CSRE and CTRE indicate the cumulative distributions of SRE and TRE over space and time, respectively.

SRE [3] is denoted as follows:

$$err_{sp}(n) = \frac{\|\hat{x}_T(n) - x_T(n)\|_2}{\|x_T(n)\|_2}, \tag{14}$$

where $n = 1,2,\ldots,N$ and N denotes the number of OD flows in a network, T the number of the estimated time points, and $\|.\|_2$ the L_2 norm. Fig. 11(a) plots the SRE of two methods and Fig. 11(b) represents the CSRE, where TomoG denotes TomoGravity method, with BPTME in red and TomoGravity in blue. In Fig. 11(a), x-axis denotes OD flow ID which is attained by ordering all OD flows from the smallest OD flow to the largest OD flow in mean, and y-axis indicates SRE of TM estimations, which is obtained in terms of (14). In Fig. 11(b), x-axis represents the SRE, whereas y-axis indicates the CSRE.

From Fig. 10(a), we can see that, when OD flow is larger, TomoGravity holds the fairly low SRE, but when OD flow is smaller, it holds the fairly large SRE. This shows that TomoGravity can accurately predict the larger OD flows, whereas it will yield the larger estimation errors when it is used to predict the smaller OD flows. This conclusion is consistent to that in [3]. However, BPTME does not only hold the lower SRE when OD flow is larger, but its SRE also is lower when OD flow is smaller. This shows that BPTME can be used to estimate the larger OD flows, but it can also be used to estimate the smaller OD flows. More importantly, from Fig. 10(a), we can see that in the case that OD flow is larger or smaller, SRE of BPTME is lower than those of TomoGravity. Furthermore, when OD flow is smaller, SRE of BPTME is much lower than that of TomoGravity, even low 20 times or so. Thus, Fig. 10(a) shows that BPTME can more accurately estimate all the OD flows than TomoGravity. From Fig. 10(b), we can see that SRE of 82% above OD flows with BPTME is lower than 0.8, whereas only 75% OD flows with TomoGravity. Likewise, we can also see

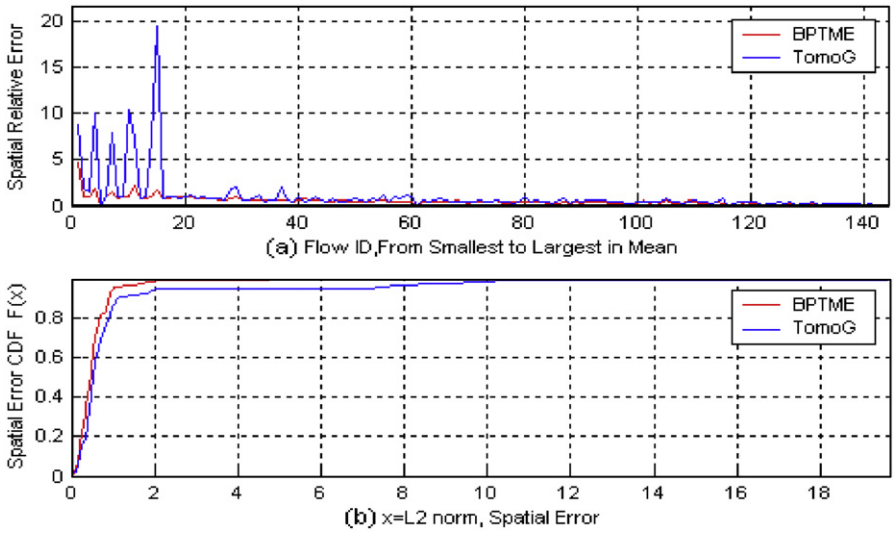


Fig. 10. (a) Spatial relative error (x-axis is flow ID; flows ordered from smallest to largest in mean); (b) CDF of spatial errors. TomoGravity in blue; BPTME in red. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

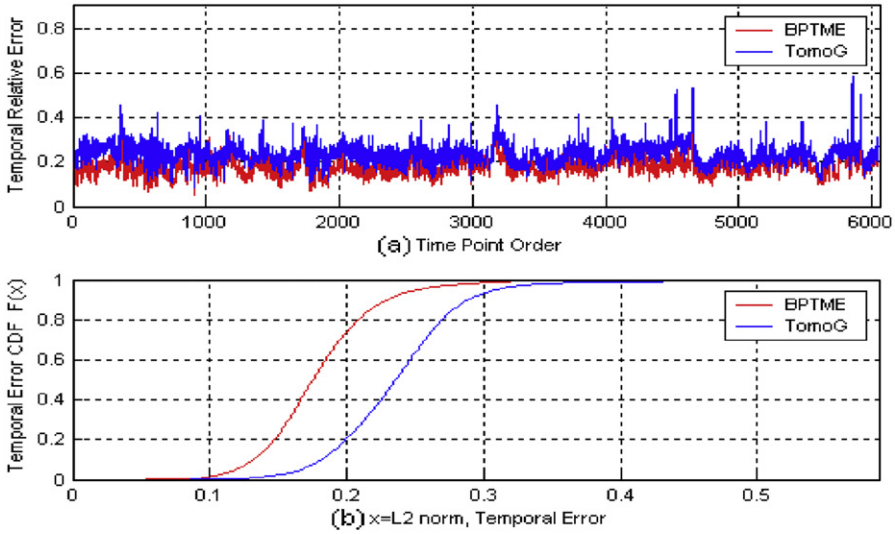


Fig. 11. (a) Temporal relative error (x-axis in time units of 5 min); (b) CDF of temporal errors. TomoGravity in blue; BPTME in red. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

that about 99% OD flows with BPTME hold SRE lower than 2, whereas about 95% with TomoGravity. Hence, Fig. 10(b) shows that SRE of all the OD flows with BPTME is nearly lower than those with TomoGravity.

TRE [3] is represented as follows:

$$err_{tm}(t) = \frac{\|\hat{x}_N(t) - x_N(t)\|_2}{\|x_N(t)\|_2}, \tag{15}$$

where $t = 1,2,\ldots,T$. Fig. 11(a) plots the TRE of two methods and Fig. 11(b) represents the CTRE, where TomoG denotes TomoGravity method, with BPTME in red and TomoGravity in blue. In Fig. 11(a), x-axis denotes the time point order which is in time units of 5 min, and y-axis indicates TRE of TM estimations, which is obtained in terms of (15). In Fig. 11(b), x-axis represents the TRE, whereas y-axis indicates the CTRE.

From Fig. 11(a), we can see that TomoGravity holds about 0.22 TRE on average, whereas BPTME holds about 0.15 TRE on average. Over the whole time points, TRE of TomoGravity makes up and down fluctuation near 0.22, but it yields the burst changes, e.g., at time points 362, 4522, 5856 and so on. However, TRE of BPTME makes up and down fluctuation near 0.15, and it does not yield the burst changes. Moreover, TRE of BPTME is lower than those of TomoGravity in all. This shows that BPTME holds the higher estimation accuracy over the time than TomoGravity. More importantly, from Fig. 11(a), we also see that TRE of BPTME is more stationary than those of TomoGravity. Over the whole time points, BPTME does not yield the burst or dramatic TRE, but TomoGravity do yield these errors. From Fig. 11(b), we can see that TRE of about 74% time points with BPTME is lower than 0.2, whereas only about 20% time points with TomoGravity. Similarly, we can also see that about 94.6% time points with BPTME hold TRE lower than 0.25, whereas about 63.5% with TomoGravity. Hence, Fig. 11(b) shows that TRE of all the time points with BPTME is nearly lower than those with TomoGravity method.

Generally speaking, BPTME holds lower SRE and TRE than TomoGravity. This shows that the outputs of our model can more accurately capture the characteristics of large-scale IP TM than gravity model. Thus BPTME can attain the more accurate TM estimations than TomoGravity. BPTME cannot be used to predict the larger OD flows, but it also can be used to predict the smaller

OD flows. This makes BPTME significantly suited for estimating large-scale IP TM.

4.3. Robustness analysis

This subsection discusses the robustness of two methods. By introducing the noise into link loads which can be directly measured by SNMP at each link, we simulate the robustness of two methods. As mentioned in [2,7], we introduce an error term $\varepsilon(t)$ in the time t to (2), and attain the following equation:

$$Y_N(t) = AX(t) + \varepsilon(t), \tag{16}$$

where $\varepsilon(t) = Y_N(t) * N(0,\delta)$, and $N(0,\delta)$ denotes a normal distribution with mean 0 and standard deviation δ, $Y_N(t)$ link loads including the noise, $X(t)$ TM, and A still routing matrix. We discuss the robustness of two methods in three cases of $\delta = 0.01$, $\delta = 0.03$ and $\delta = 0.05$.

According to (14), (15), and (16), we attain Table 4.3.1, where noise level includes noise free, $\delta = 0.01$, $\delta = 0.03$ and $\delta = 0.05$, and CSRE and CTRE of two methods are attained with 0.5 SRE and 0.2 TRE, respectively. Table 1 shows that for TomoGravity, when TRE is 0.2, CTRE is reduced by 0.032 with noise from free to $\delta = 0.01$, by 0.033 with noise from $\delta = 0.01$ to $\delta = 0.03$, and by 0.039 from $\delta = 0.03$ to $\delta = 0.05$. Similarly, when SRE is 0.5, CSRE is reduced by 0.031, 0.031, and 0.039. This shows that when noise changes from free to $\delta = 0.05$, about 3% of time points is to be added to yield TRE larger than 0.2 every time, and about 3% of OD flows is to be added to yield SRE larger than 0.5 every time. However, for BPTME, the changes of CTRE and CSRE are very small, and even can be omitted. Hence, from Table 1, we can find that, as noise is added from noise free to $\delta = 0.05$, for TomoGravity, its CTRE and CSRE are always increasingly small, whereas for BPTME, its CTRE and CSRE nearly keep constant. This shows that the impact with noise on BPTME is much lower than on TomoGravity, i.e., BPTME is nearly not sensitive to noise in the process of predicting large-scale IP TM.

Alternatively, to avoid the relative single metric above, we here use other several metrics, namely spatial root mean squared error (SRMSE), spatial root mean squared relative error (SRMSRE), temporal root mean squared error (TSRMSE), and temporal root mean squared relative error (TSRMSRE), to evaluate the robustness of two methods. They are defined as follows:

$$\begin{cases} SRMSE = \dfrac{1}{N}\displaystyle\sum_{n=1}^{N} \|\hat{X}_T(n) - X_T(n)\|_2, \\ SRMSRE = \dfrac{1}{N}\displaystyle\sum_{n=1}^{N} \dfrac{\|\hat{X}_T(n) - X_T(n)\|_2}{\|X_T(n)\|_2}, \end{cases} \tag{17}$$

where N is the number of OD flows in a network.

$$\begin{cases} TRMSE = \dfrac{1}{T}\displaystyle\sum_{n=1}^{T} \|\hat{X}_N(t) - X_N(t)\|_2, \\ TRMSRE = \dfrac{1}{T}\displaystyle\sum_{n=1}^{T} \dfrac{\|\hat{X}_N(t) - X_N(t)\|_2}{\|X_N(t)\|_2}, \end{cases} \tag{18}$$

Table 1
CDF of the spatial and temporal relative errors.

Noise level	TomoGravity		BPTME	
	CTRE(0.2)	CSRE(0.5)	CTRE(0.2)	CSRE(0.5)
Free	0.204	0.473	0.739	0.591
$\delta = 0.01$	0.172	0.442	0.743	0.589
$\delta = 0.03$	0.139	0.411	0.744	0.599
$\delta = 0.05$	0.100	0.372	0.749	0.590

Table 2
Impact with noise on two methods.

Noise level	$\delta = 0.01$	$\delta = 0.03$	$\delta = 0.05$
Link loads			
SRMSE (Mbps)	72.572	217.740	365.400
SRMSRE	1.00%	3.01%	5.06%
TRMSE (Mbps)	8.018	24.090	40.330
TRMSRE	0.90%	2.95%	4.94%
Estimation error with BPTME			
SRMSE (Mbps)	174.370	174.371	174.373
SRMSRE	53.05%	53.05%	53.06%
TRMSE (Mbps)	33.916	33.916	33.917
TRMSRE	17.94%	17.94%	17.95%
Estimation error with TomoGravity			
SRMSE (Mbps)	226.560	238.600	253.490
SRMSRE	87.24%	89.87%	94.14%
TRMSE (Mbps)	43.208	44.899	47.569
TRMSRE	23.82%	24.59%	25.95%

where T is the number of he estimated time points. SRMSE and TRMSE give an overall metric for the spatial and temporal errors in the estimates, respectively. SRMSRE and TRMSRE present a relative measure for the spatial and temporal errors in the estimates, respectively.

According to (17) and (18), we compute the several metrics and attain Table 2. According to the several metrics, Table 2 summarizes the impact with noise on two methods. Table 2 shows that, in the case that noise level is changed from $\delta = 0.01$ to $\delta = 0.05$, SRMSE of TomoGravity increases correspondingly by 1.691 and 2.670 Mbps when SRMSE of link loads increases by 16.072 and 16.240 Mbps, respectively. TRMSE of TomoGravity is added correspondingly by 12.04 and 14.89 Mbps when TRMSE of link loads is added by 145.168 and 147.66 Mbps, respectively. Moreover, SRMSRE of TomoGravity is added correspondingly by 2.63% and 4.27% every time when SRMSRE of link loads is added by 2.01% and 2.05%, respectively. TRMSRE of TomoGravity increases correspondingly by 0.775% and 1.355% when TRMSRE of link loads increases by 2.05% and 1.99%, respectively. This shows that for TomoGravity, the overall performance degradation is smaller than the introduced errors on link loads. But its SRMSRE degradation is larger than that of link loads, whereas its TRMSRE degradation is smaller. For BPTME, when link errors increase, its SRMSE, TRMSE, SRMSRE, and TRMSRE are 174.371 Mbps, 33.916 Mbps, 53.05%, and 17.94% or so, respectively. They nearly keep constant. This shows further that BPTEM is nearly not sensitive to noise. Moreover, these metrics of BPTME is much smaller than those of TomoGravity. Hence, Table 2 shows that BPTME is more robust to noise than TomoGravity.

5. Conclusions

Based on BPNN, this paper has proposed a novel method for large-scale IP TM estimation, namely BPTME. Large-scale IP TM estimation is significantly difficult since it is a highly ill-posed inverse problem. How fast and accurately to estimation large-scale IP TM is a challenge that network operators so far face with. BPTME is simple and fast (taking less than 10 min to estimate TM in the Abilene network for three weeks). Based on the powerful modeling capacity of BPNN, we use it to model large-scale IP TM estimation problem, and then we can easily avoid the complex mathematical computations so that we can quickly make TM estimation. Because the multi-input and multi-output model

proposed by us is a parallel structure, this ensures that BPTME can fast estimate TM and be extended to larger network.

We have validated BPTME by using the real data from the Abilene network. Simulation results show that BPTME holds the lower estimation errors and is more robust to noise than TomoGravity. Moreover, BPTEM is not nearly sensitive to noise. It cannot only predict the larger OD flows, but also predict the smaller OD flows. These mainly benefit in the capability of learning and generalizing that BPNN holds. By training BPNN, the outputs of our multi-input and multi-output model can fairly accurately capture the properties of large-scale IP TM. As discussed in session 4, BPTME can make the fairly accurate estimation of large-scale IP TM and track its dynamics.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (Nos. 60673159, 70671020, 70931001, 60802023), the National High-Tech Research and Development Plan of China (No. 2007AA041201), the National Science & Technology Pillar Program (Nos. 2008BAH37B03, 2008BAH37B07), the Specialized Research Fund for the Doctoral Program of Higher Education (No. 20070145017), the State key Laboratory of Networking and Switching Technology (Beijing University of Posts and Telecommunications) (No. SKLNST-2009-1-04), and the Chinese Universities Scientific Fund (Nos. N090404014, N090504003, N090504006). The authors wish to thank the reviewers for their helpful comments.

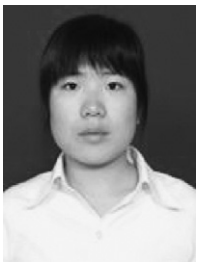
References

[1] Papagiannaki K, Taft N, Lakhina A. A distributed approach to measure traffic matrices. In: Proceedings of ACM internet measurement conference, Taormina, Italy, October 2004.
[2] Zhang Y, Roughan M, Duffield N, Greenberg A. Fast accurate computation of large-scale ip traffic matrices from link loads. In: Proceedings of ACM sigmetrics'03, San Diego, CA, 2003.
[3] Soule A, Lakhina A, Taft N, et al. Traffic matrices: balancing measurements, inference and modeling. In: Proceedings of ACM sigmetrics'05, Banff, June 2005.
[4] Medina A, Taft N, Salamatian K, et al. Traffic matrix estimation: existing techniques and new directions. In: Proceedings of ACM SIGCOMM'02, Pittsburgh, USA, August 2002.
[5] Vardi Y. Network tomography: estimating source-destination traffic intensities from link data. Journal of the American Statistical Association 1996;91(433):365–77.
[6] Cao J, Davis D, Vander W-S, et al. Time-varying network tomography. Journal of the American Statistical Association 2000;95:1063–75.
[7] Zhang Y, Roughan M, Lund C, et al. An information theoretic approach to traffic matrix estimation. In: Proceedings of ACM SIGCOMM'03, Karlsruhe, Germany, August 2003.
[8] Jiang D, Chen J, He L. An accurate approach of large-scale ip traffic matrix estimation. IEICE Transactions on Communications 2007;E90-B(12): 3673–6.
[9] Jiang D, He G. Garch model-based large-scale ip traffic matrix estimation. IEEE Communications Letters 2009;13(1):52–4.
[10] Liang G, Taft N, Yu B. A fast lightweight approach to origin-destination ip traffic estimation using partial measurements. IEEE Transactions on Information Theory 2006;52(6):2634–48.
[11] Parekh R, Balakrishnan K, Honavor V. An empirical comparison of flat-spot elimination techniques in back-propagation networks. In: Proceedings of the third workshop on neural networks – WNN'92; 1992. p. 55–60.
[12] Pethick M, Liddle M, Werstein P, et al. Parallelization of a backpropagation neural network on a cluster computer. In: Proceedings of the 15th IASTED international conference on parallel and distributed computing and systems (PDCS 2003), Marina Del Rey, California; 2003. p. 574–82.
[13] Hinton G-E, Osindero S, Welling M, et al. Unsupervised discovery of non-linear structure using contrastive backpropagation. Cognitive Science 2006;30(4):725–31.
[14] Hornik K, Stinchcombe M, White H. Multilayer feedforward networks are universal approximators. Neural Networks 1989;2:359–66.
[15] Gunnar A, Johansson M, Telkamp T. Traffic matrix estimation on a large ip backbone. In: Proceedings of ACM SIGCOMM IMC 2004, Taormina, Italy, 2004.
[16] Datasets: 〈http://www.cs.utexas.edu/∼yzhang/research/abilene-tm/〉.
[17] Tebaldi C, West M. Bayesian inference on network traffic using link count data. Journal of the American Statistical Association 1998;93(442):557–76.
[18] Nucci A, Cruz R, Taft N, et al. Design of igp link weight changes for estimation of traffic matrices. In: Proceedings of IEEE infocom, Hong Kong, March 2004.
[19] Soule A, Nucci A, Leonardi E, et al. How to identify and estimate the largest traffic matrix elements in a dynamic environment. In: Proceedings of ACM sigmetrics, New York, June 2004.
[20] Lakhina A, Papagiannaki K, Crovella M, et al. Structural analysis of network traffic flows. In: Proceedings of ACM sigmetrics, New York, June 2004.
[21] Soule A, Salamatian K, Nucci A, et al. Traffic matrix tracking using kalman filtering. LIP6 Research report RP-LIP6-2004-07-10, LIP6, 2004.
[22] Bermolen P, Vaton S, Juva I. Search for optimality in traffic matrix estimation: a rational approach by Cramr–Rao lower bounds. In: Proceedings of the second EuroNGI NGI conference on next generation internet design and engineering; 2006. p. 224–31.
[23] Juva I. Sensitivity of traffic matrix estimation techniques to their underlying assumptions. In: Proceedings of ICC'07; 2007. p. 562–8.
[24] Battiti R. First and second order methods for learning: between steepest descent and Newton's method. Neural Computation 1992;4(2): 141–66.
[25] Rumelhart D-E, Hinton G-E, Williams R-J. Learning representations by back-propagating errors. Nature 1986;323:533–6.

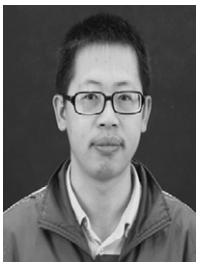
Dingde Jiang received the Ph.D. degree in Communication and Information Systems from School of Communication and Information Engineering, University of Electronic Science and Technology of China, Chengdu, China, in 2009. He is currently an Associate Professor in College of Information Science and Engineering, Northeastern University, Shenyang, China. His research interests include network measurement, network security, Internet traffic engineering, communication networks, and cognitive networks. Dr. Jiang is a member of IEEE and IEICE.

Xingwei Wang received the B.Sc., M.Sc., and Ph.D degrees in Computer Science from Northeastern University, China, in 1989, 1992, and 1998, respectively. He is currently a Professor in the College of Information Science and Engineering, Northeastern University, China. His research interests are mainly on routing algorithms and protocols, QoS control schemes, fault-tolerance and survivability models, mobility management mechanisms and resource assignment methods in NGI, IP over DWDM optical Internet and mobile Internet. He has published over 100 technical papers in the above areas. As one of the co-investigators, he has won the first-level science and technology advancement award of China ministry of education twice.

Lei Guo received the Ph.D. degree in communication and information systems from School of Communication and Information Engineering, University of Electronic Science and Technology of China, Chengdu, China, in 2006. He is currently a Professor in College of Information Science and Engineering, Northeastern University, Shenyang, China. His research interests include network survivability, optical networks, and wireless mesh networks. He has published over 100 technical papers in the above areas. Dr. Guo is a member of IEEE and OSA. He was the recipient of the Best Paper Award from the International Conference on Communications, Circuits and Systems (ICCCAS'04). He is currently servicing as the Editorial Board Member of The Open Optics Journal.



Haizhuan Ni received the B.Sc. degree in Electronic Information Science and Technology from Shandong University of Science and Technology, Shandong, China, in 2009. She is currently a Master in Communication and Information System, Northeastern University, China. Her research interests include network measurement and network security.



Zhenhua Chen will receive the B.Sc. degree in College of Information Science and Engineering, Northeastern University, Shenyang, China, in 2010. His research interests include network measurement and cognitive networks.