Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 1039613, 11 pages
https://doi.org/10.1155/2021/1039613

Research Article

Design and Implementation of a Robust Convolutional Neural Network-Based Traffic Matrix Estimator for Cloud Networks

Rashida Ali Memon[1], **Sameer Qazi**[2] and **Bilal Muhammad Khan**[1]

[1]*Department of Electronics & Power Engineering, PN Engineering College, National University of Sciences & Technology, Habib Ibrahim Road, Karachi 75350, Pakistan*
[2]*Department of Electrical Engineering, College of Engineering, PAF Karachi Institute of Economics & Technology (PAF-KIET), Korangi Creek, Karachi 75190, Pakistan*

Correspondence should be addressed to Rashida Ali Memon; rashida@pnec.nust.edu.pk

Received 16 April 2021; Accepted 10 May 2021; Published 3 June 2021

Academic Editor: Nawab Muhammad Faseeh Qureshi

Recent research literature shows promising results by convolutional neural network- (CNN-) based approaches for estimation of traffic matrix of cloud networks using different architectures. Although conventionally, convolutional neural network-based approaches yield superior estimation; however, these rely on assumptions of availability of a large training dataset which is completely accurate and nonsparse. In real world, both these assumptions are problematic as training data size may be limited, and it is also prone to missing (or incomplete) measurements as well as may have measurement errors. Similarly, the 2-D training datasets derived from network topology based may be sparse. We investigate these challenges and develop a novel architecture which can cater for these challenges and deliver superior performance. Our approach shows promising results for traffic matrix estimation using convolutional neural network-based techniques in the presence of limited training data and outlier measurements.

1. Introduction

Internet traffic load is increasing multifold every few years. In US alone, only the business internet traffic volume has seen a rapidly increasing trend growing from 45.1 billion GB in 2016 to 112.7 billion GB in 2020 and projected to increase up to 224.8 billion GB in 2023 [1]. Two technologies are primary contributors for this increased traffic load: (1) cloud computing services which conveniently offer the Internet as a one-stop solution to all users in the form of infrastructure, platform, and software as service (AaaS, PaaS, and SaaS). (2) software-defined networks (SDNs) devise intelligent technologies to cater for ensuring Quality of Service (QoS) for Internet users even in presence of an ever-increasing Internet cross traffic. This places significant burden on the network as traditional traffic engineering (TE) techniques are effectively bypassed in software-defined networks. Akyildis et al. provide a comprehensive review of a roadmap [2] and challenges [3] to implement TE in SDNs in their survey papers. While the traditional flow level-based routing strategies to ensure QoS like MPLS, ATM, IntServ, and DiffServ failed in achieving scalability and QoS guarantees, SDNs employ programmable OpenFlow (OF) switches that communicate with SDN controllers via OF protocol to dynamically modify the traditional forwarding tables of routers based on a flow-level control in scalable manner such as using hash-based approaches and employing more powerful hardware like multithreading controllers. The absence of an efficient network monitoring system (NMS) also makes the network vulnerable to security threats like DoS/DDoS [4] and Sybil attacks [5]. An NMS can timely detect and thwart such threats by timely detection of network traffic anomalies [6].

With conventional traffic engineering (TE) solutions bypassed in SDNs, it is only a matter of time when SDNs would have squeezed maximum QoS out of the network necessitating further capacity planning by network administrators. Thus, it becomes increasingly important to estimate the network parameters for an improved and reliable

network capacity planning and management. Estimation of dynamic live network parameters has been investigated by a lot of researchers [7]. Special emphasis on parameters affecting the QoS of network includes network delays [8], network traffic volumes [9], packet losses, and network capacity planning. Network delays are important as some network applications like voice over IP (VoIP) require end-to-end delays to be within a specified threshold usually, and the limit for one-way delay of data packets carrying voice should be less than 150 ms. Similarly, VoIP requires packet loss rates of less than 1% and average one-way packet jitter to be less than 30 msec. Similarly, capacity planning in the form of upgrading network bandwidth and/or employment of effective traffic shaping/policing techniques is equally important as in the absence of that rate adaptive flows may be hurt by nonrate adaptive flows [10].

Recent research literature shows promising results by using neural network-based approaches for estimation of traffic matrix using different architectures, such as use of Artificial Neural Networks (ANNs) [11] and recurrent neural networks (RNNs) [12]. Another new approach used recently is convolutional neural network- (CNN-) based which shows promising results for traffic matrix estimation. Although the results of traffic matrix prediction using a CNN-based architecture are impressive under ideal assumed conditions, the results will deteriorate if real-world problems are considered. We list ideal conditions here: (1) an assumption that a huge volume of training dataset is available, and this is often not the case in real situations; (2) an assumption that training dataset is complete, i.e., it is not having any missing measurements, measurement noise, or outliers.

A consequence of the violation of any of the above assumptions is that if we employ CNN using standard architectures, they may yield suboptimal performance, e.g., if a small volume of dataset is available for training or if we use an adaptive learning-rate optimizer such as Adagrad as used in [13], and it will yield inferior performance because of the incorrect assumption of the availability of large clean dataset; it will adaptively lower the learning rate as the training would progress over time.

In this paper, we develop a robust CNN-based traffic matrix estimation framework which can deliver superior performance in the presence of limiting training data or training data with outlier measurements.

The rest of this paper is organized as follows. Section II discusses the related work. We present the problem of traffic matrix estimation in Section III. Section IV outlines the methodology of our techniques, followed by Section V in which we present the simulation results, and finally, Section VI draws the conclusions.

2. Related Work

Network parameter measurement techniques have been covered exhaustively in research literature. Few popular technique-specific terms coined by researchers include Kriging [14], cartography [15], tomography [16], and compressed sensing [17, 18]. Technically, all of these network parameter estimation techniques can be separated as those which are space-based, time-based, and use spatiotemporal techniques [19]. The temporal aspect of the measured network parameters is obtained through the availability of network training data while the spatial aspect is generated from the knowledge of topology of the network and how it can potentially impact the measured parameters. The network parameter estimation or prediction algorithm in these spatiotemporal techniques belongs to approaches such as Bayesian estimation, Linear Optimization, and Maximum Likelihood (or Expectation Maximization) [9].

It is pertinent to highlight that for the problem under consideration in this paper, namely, traffic matrix estimation, the accuracy of training data is of utmost importance. For example, the expectation maximum-based estimation approach uses statistical inference tools that are based on distribution of elements of the traffic matrix. These are then used to compute, based on information of link counts, an expectation of the elements of the traffic matrix. Such techniques, however, rely heavily on initial or given knowledge of the traffic flow mean and variance. Many researchers, for this purpose, have utilized various distributions such as Gaussian and Poisson. [20]. Towards that end, even in the case when initial or preliminary estimates of a recognized distribution of traffic flows are known, these techniques very much rely on knowledge of the initial prior [9], which acts as a starting point for optimization algorithms to yield a solution obeying all the spatial constraints of the problem. Hence, these techniques are highly dependent upon 'goodness' of a known or prior solution. Earlier research work done by Tebaldi et al. [21] and Vardi [22] for estimation of traffic elements in IP backbones introduced the idea of adapting a hybrid approach, in which elements of the traffic matrix are either estimated or algebraically computed based on link load measurements. This approach significantly reduces complexity of the problem; since in this case, estimation of a much lesser number of origin/destination (OD) pair traffic flows is required. Such schemes have shown to be beneficial in cases where the routing matrix exhibits a highly decaying Eigen spectrum [23]. A few noteworthy research contributions related to problem of traffic matrix estimation include Bayesian Learning techniques which have been carried out by Nie [6] and Fan [24].

Recent research literature shows promising results by neural network-based approaches for estimate of traffic matrix using different architectures. Jiang et al. [11] fuse the neural network approach with time frequency analysis for network traffic matrix estimation. Emami et al. [13] have devised a convolutional neural network- (CNN-) based traffic matrix estimation architecture in which one part of the training data (link loads) is transformed into one higher dimension by considering the network topology. This enables exploitation of convolutional neural network-based techniques which are optimized for image (2D) datasets and are considered more robust than other types of neural networks.

Qian et al. [12] present another technique for traffic matrix estimation without any training data using only current but partial or incomplete OD flow data. The technique exploits recurrent neural networks (RNNs) to estimate the unknown OD flows from known OD flow measurements.

Qazi et al. [18] have developed a compressed sensing model for network traffic matrix estimation based on a dynamic network measurement model. The model is based on traffic demands instead of a stationary routing matrix. This technique shows that link count measurements could be further reduced with traffic matrix estimation with tolerable errors.

3. The Traffic Matrix Estimation Problem

The traffic matrix estimation problem can be represented by the following relationship:

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

where $Y \in \Re^{mxt}$ is the observable matrix of measured link counts, and m is the number of network links, recorded at indexed time intervals. $A \in \Re^{m \times n^2}$ is called routing matrix that expresses routing configurations where m and n denote the number of wide area links and those of cloud nodes, respectively. These result in n^2 possible paths between cloud nodes. We assume that the routing matrix is stationary, i.e., remains stable (unlike typical wireless networks) over long periods of time. This is valid especially for the case of Internet; since in this case, majority of the wide area paths can remain stable for several hours or days. Finally, $X \in \Re^{n\wedge 2xt}$ is an unobservable traffic matrix which is required to be estimated over the specified time intervals as indexed by t.

Solution to the traffic matrix problem as given in Equation (1) is oftentimes quite a challenging one. This is because of a variety of reasons: (a) the routing matrix A is a "fat" matrix; and since $m << n$, the system of linear equations under consideration is ill-posed, underdetermined, and underconstrained. It is pertinent to highlight the fact that for an underdetermined system of equations, a unique solution may not exist. On the other hand, if we have a balanced, well-posed, or an overdetermined linear system of equations (e.g., if we have A as a 'tall' or a 'square' matrix, i.e., when $m \geq n$), then we may have a unique solution. (b) In this modern age of dynamic software defined, validity of assumption for A being stationary is no longer acceptable. Furthermore, in these modern times, routing is no longer dictated by any deterministic algorithms. Instead, more "opportunistic" software-defined strategies are being used to allow for user-centric routing decisions rather than being network-centric in nature. In case of software-defined cloud computing platforms, several decisions while being user-centric are still treated as network oriented in order to obtain good load balancing on the network servers or the network as a whole. 1st generation research works using the assumption of stationary routing matrix A have investigated range of spatiotemporal methods as explained in the previous section. More recent works, e.g., [13] have devised strategies in which the topology information is embedded into the model using flexibility, e.g., by using the link load measurements and embedding them into link adjacency matrix. This relaxes the spatial constraints in the original problem while still enabling accurate estimation through an efficient learning process of a neural network.

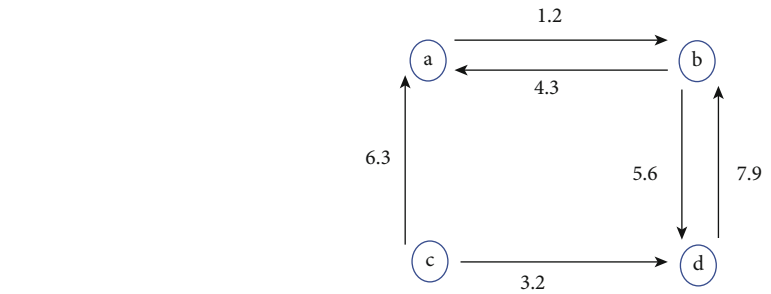
It is a well-known fact that in network traffic estimation problem, quick anomaly detection is one of the prime motivators of the development of these systems; hence, the estimator should have superior temporal prediction performance.

4. Robust CNN-Based Traffic Matrix Estimation

Recent work by Emami et al. [13] incorporating graph embedding and convolutional neural network proposed an idea to fuse the topology of network into the neural network-based estimation framework by using the graph embedding approach. This enables to add one dimension to the training data. The training data is traditionally composed of two, 1D vectors of link loads and OD flows. Using the above mentioned graph embedding technique, the observable data of link loads is transformed from 1D measurements into 2D measurements by implicit introduction of the topology into the measurement framework. This is through a 2D link adjacency matrix referred to as L2AM [13]. Figure 1 shows this graph embedding technique using a toy example as well as the CNN-based architecture to learn OD flow features using this L2AM matrix training dataset. In the original Emami et al. [13] architecture as shown in Figure 1, the output of the convolution part consists of N matrices which are vectorized to generate N different feature vectors of size $W \times W$ (corresponding to each OD flow) fed to N parallel fully connected networks (FCN) with one hidden layer and one output layer corresponding to each OD flow. The values of L and W are 13 and 9, respectively. The architecture that proposed (CNTME) gives superior performance over other contemporary approaches; however, we find that the performance becomes suboptimal when we consider that 2D training data supplied via the network topology information through link adjacency matrix is sparse (since number of OD pairs is much greater than the number of links in the network), and further, it may also have errors or noise in it. This can cause the CNN-based estimator to have suboptimal performance in terms of feature learning about the OD flow or generate outlier predictions especially when the problem at hand is a regression problem and not a classification problem. Motivated by the findings above, we see that the CNN-based traffic matrix estimator is not robust to outliers or errors in the training data.

We propose to rewire the original architecture so that the final feature set learned is that of the entire system rather than of individual OD flows. This has the effect of diluting any learning errors causing incorrect features to be learnt about individual OD flows. We call this new proposed architecture as R-CNTME. We present more details about this in the following paragraph.

If the 2D training data has limited measurements or outliers, CNTME architecture has the effect of magnifying the errors in the estimation of each individual OD flows via N parallel FCN. Our architecture differs that the feature matrices generated after the downsampled output of the convolution layers are flattened before the FCN layer (instead of afterwards) to generate one feature vector of the entire system (N OD flows) instead of individual OD flows. This has



Src\Dst	a	b	c	d
a	0	1.2	0	0
b	4.3	0	0	5.6
c	6.3	0	0	3.2
d	0	7.9	0	0

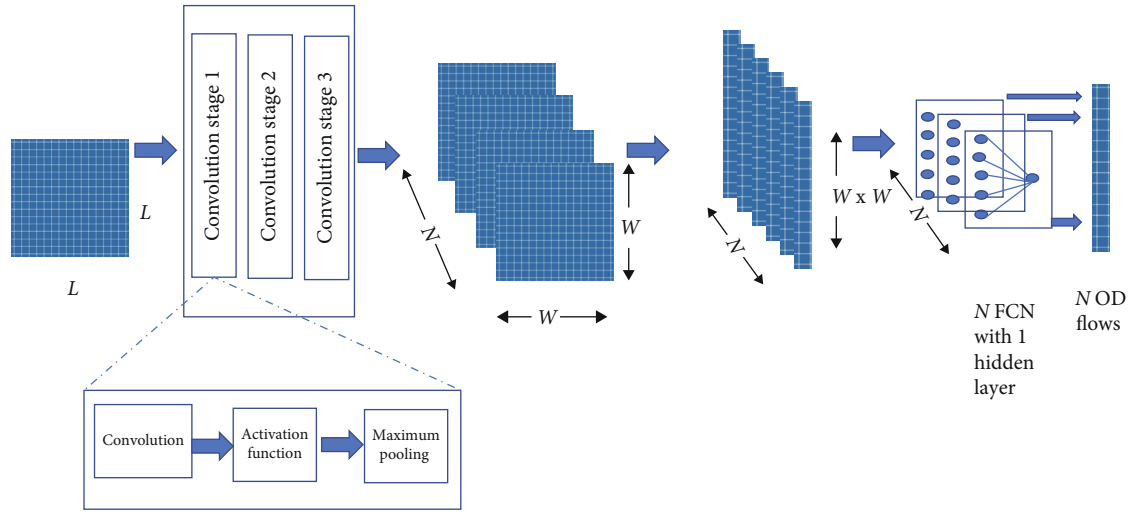


Figure 1: (top) Graph embedding technique using link adjacency matrix and numbers over arrows shows traffic volume over directed links; (bottom) convolutional neural network technique (CNTME) proposed by Emami et al. [13].

the effect of diluting the learning errors in the CNN stage caused due to sparsity of 2D training data, limited training data, and potential outlier measurements in it. Detailed schematic of the proposed architecture is given in Figure 2 below.

5. Description of ABILENE Dataset

We use the real dataset of the Abilene network (a backbone network in the United States) to evaluate the performance of our proposed ROBUST CNN-based technique. This network consists of a total of 12 nodes. Accordingly, there are a total of 144 OD flows (since $n = 12 \Rightarrow N = 12^2 = 144$). There are a total of 54 links (i.e., $m = 54$), of which 30 links provide for the connectivity between the near-neighbor nodes. The rest connects all other nodes that are available over the Internet. The Abilene network dataset consists of end-to-end traffic measurement for 24 weeks. Since the measurement time slots are considered as 5-minute intervals, therefore, there are a total of 2016 measurement points (i.e., 7 times 24 times 12 which equals to 2016) per week.

In this work, we consider the external network as an independent node. We add it with the 12 nodes of the Abilene network to cater for the load between the Abilene network and the external network. Accordingly, the L2AMs become matrices of 13×13 size.

5.1. Performance Evaluation. To test the performance of the traffic matrix estimation architecture proposed in this paper and to do a fair comparison with Emami et al. [13], we use Adam optimizer rather than Adagrad because we also simulate noise in the training data, and Adagrad lowers its learning rates with training epochs. We use mean squared error (MSE) as the loss function.

Like Emami et al. [13], we use the Softplus activation functions for the benefit that it does not give the problem of negative estimated values; so, no additional steps are needed for negative values treatment. Softplus generates output values as per function $f(x)$ for input values x.

$$f(x) = \log\left(1 + e^{x}\right) \tag{2}$$

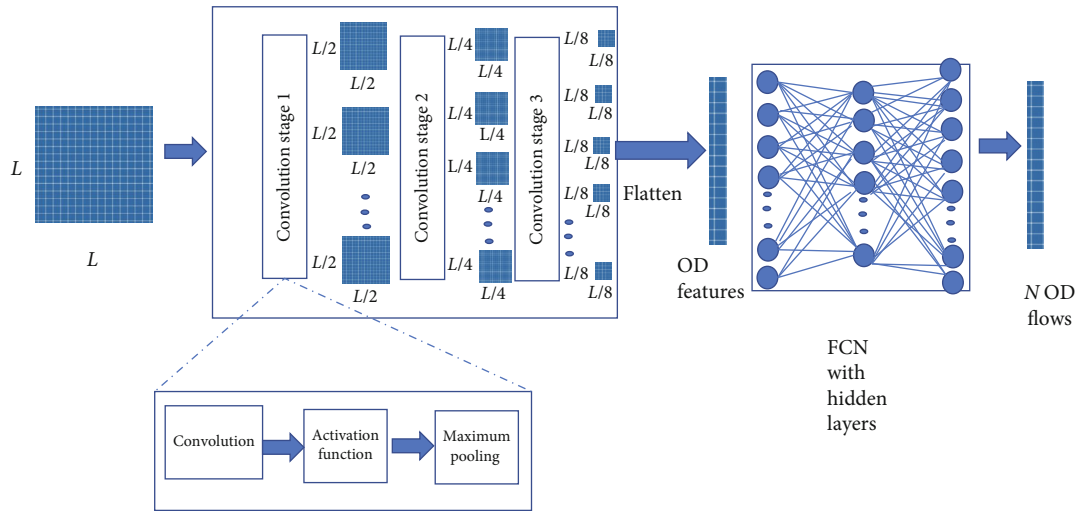


Figure 2: Schematic of the proposed architecture for robust convolutional neural network-based traffic matrix estimator (R-CNTME).

Performance of the proposed techniques has been evaluated using Abilene network datasets based on real captured network traffic as described in the previous section. For this purpose, we have used two different platforms, namely, jupyter notebook (online) and spyder python 3.7 at AMD Radeon R7 240 with a PC. The specifications of the PC are as follows. CPU: Intel Core i7 7770 @3.6 GHz with 32 GB RAM (DDR4). We have extensively evaluated all our simulations on Colab, which uses GPU Tesla K80 for TensorFlow and Keras. In all the considered cases, we use the first 1500 samples of the first week data as training dataset and the last 500 samples for testing unless specified explicitly. To compare the performance of our scheme, we use the established metrics that are widely used in the relevant recent research literature [25]. These metrics are defined as follows:

Spatial relative error (SRE):

$$\mathrm{SRE}(i) = \frac{\left\|\widehat{X}_i - X_i\right\|_2}{\|X_i\|_2}; i = 1, 2, \cdots, n^2 \tag{3}$$

where X_i represents the actual OD flows, and $\widehat{X}_i$ represents estimated OD flows for $i \in 1, 2, \cdots, n^2$ at time t.

Temporal relative error (TRE):

$$\mathrm{TRE}(t) = \frac{\left\|\widehat{X}_t - X_t\right\|_2}{\|X_t\|_2}; t = 1, 2, \cdots, T \tag{4}$$

in which X_t represents the actual OD flow,and $\widehat{X}_t$ represents estimated OD flow for time t while T denotes the length of the period of testing.

Bias:

$$\mathrm{Bias}(i) = \frac{1}{T}\sum_{t=1}^{T}\left(\widehat{X}_{i,t} - X_{i,t}\right) \tag{5}$$

Table 1: Simulated errors in the L2AM training data.

Error scenario	% of Noisy L2AM entries	Noise locations in all time indices	Noise distribution
E1	30%	Random	Gaussian N(0, 25e12)

Table 2: Number of CNN and FCN layers.

CNN layers (filter size 2×2)	FCN hidden layers
3	81 : 144
2	81 : 100 : 144
2	81 : 100 : 121 : 144
2	81 : 90 : 100:121 : 144
1	81 : 144

Standard deviation:

$$\text{Standard Deviation}\,(i) = \sqrt{\frac{1}{T-1}\sum_{t=1}^{T}\left(\left(\widehat{X}_{i,t} - X_{i,t}\right) - \mathrm{bias}(i)\right)^2} \tag{6}$$

where $X_{i,t}$ represents the actual OD flow, and $\widehat{X}_{i,t}$ represents the estimated OD flow for $i \in 1, 2, \cdots, n^2$at time t, while T denotes the length of the period of testing.

We find that this framework works well when there are no errors in the training data. We generate random noise for the L2AM matrix training data to 30% of the nonzero entries, respectively. We call it error scenario (E1) in which we generate noise in random 30% locations of the nonzero entries of the training data using the Gaussian distribution with zero mean and standard deviation equal to 5×10^6.

5.2. Performance of Proposed Architecture with Different Numbers of CNN and FCN Layers with Simulated Errors in Abilene Dataset. To validate our architecture, we test the

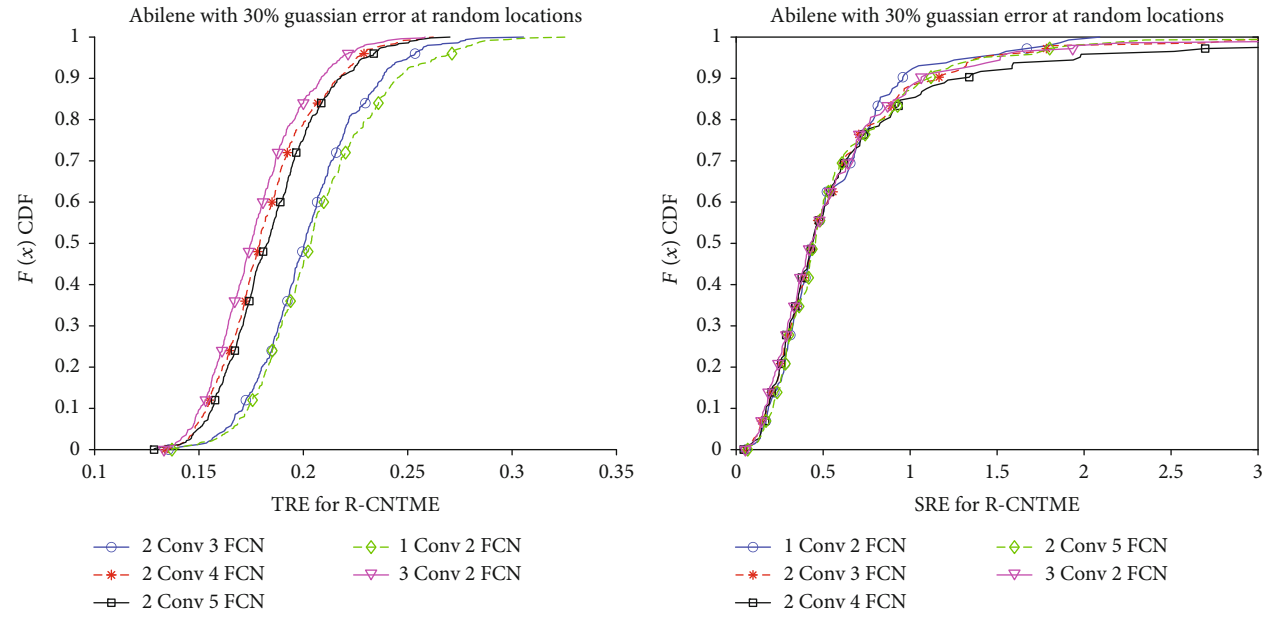


Figure 3: SRE and TRE performance of proposed R-CNTME architecture when errors are introduced in the training data and number of CNN layers and FCN layers are varied.

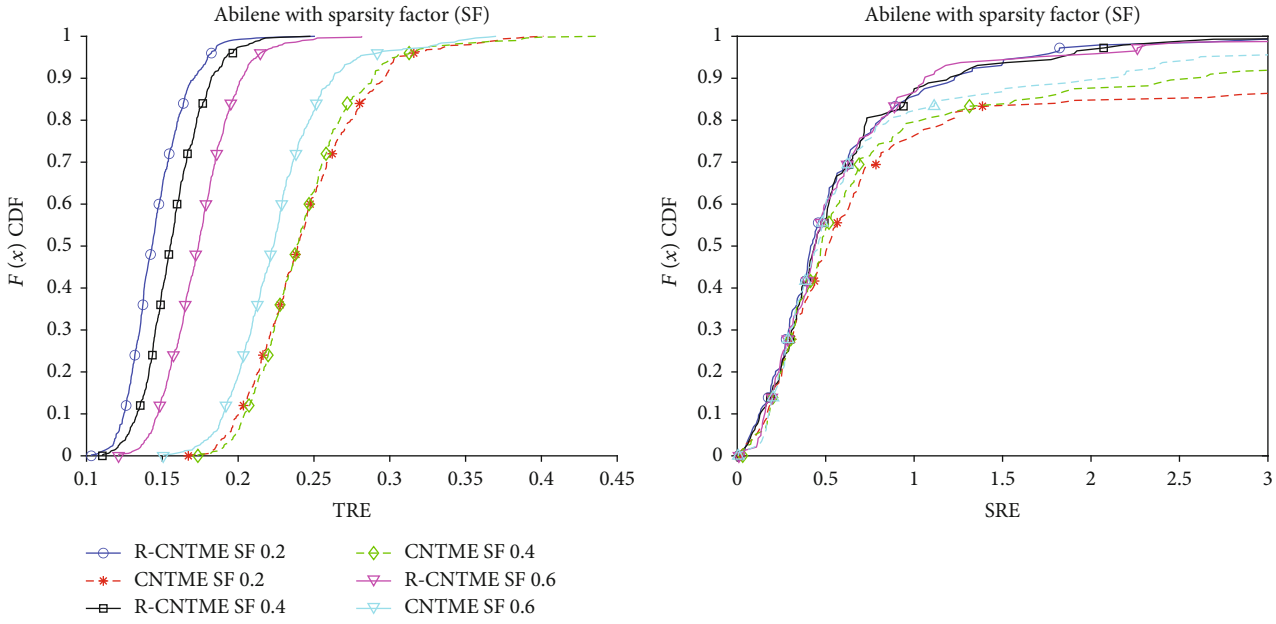


Figure 4: Comparison of performance of R-CNTME AND CNTME for Abilene dataset for different sparsity.

performance of our proposed scheme by first introducing errors in the 2D training L2AM data using simulated error scenario E1 as shown in Table 1. This is so that we can simulate both the effects of sparsity and training data errors while validating the model. We vary the number of CNN and FCN layers from 3 to 5 using the specified hidden layers as shown in Table 2.

We observe from the result in Figure 3 that as number of CNN layers and FCN layers are varied, we see marginal differences in the SRE performance. However, we notice substantial improvement in TRE performance (as CNN and FCN layers are increased); optimum trade-off for TRE metrics is best when the number of CNN layers is 3 and number of FCN layers are equal 2. This is because increasing the number of neurons in the hidden layers of the FCN is well known to cause the problem of overfitting, and too few neurons in hidden layers are well known to cause the problem of underfitting. Hence, CNN and FCN layers cannot be increased indefinitely as network anomaly prediction requires the estimator to be good in both the spatial and temporal aspects.

Addition of more hidden layers in the FCN part may add more nonlinearities and may improve the SRE performance

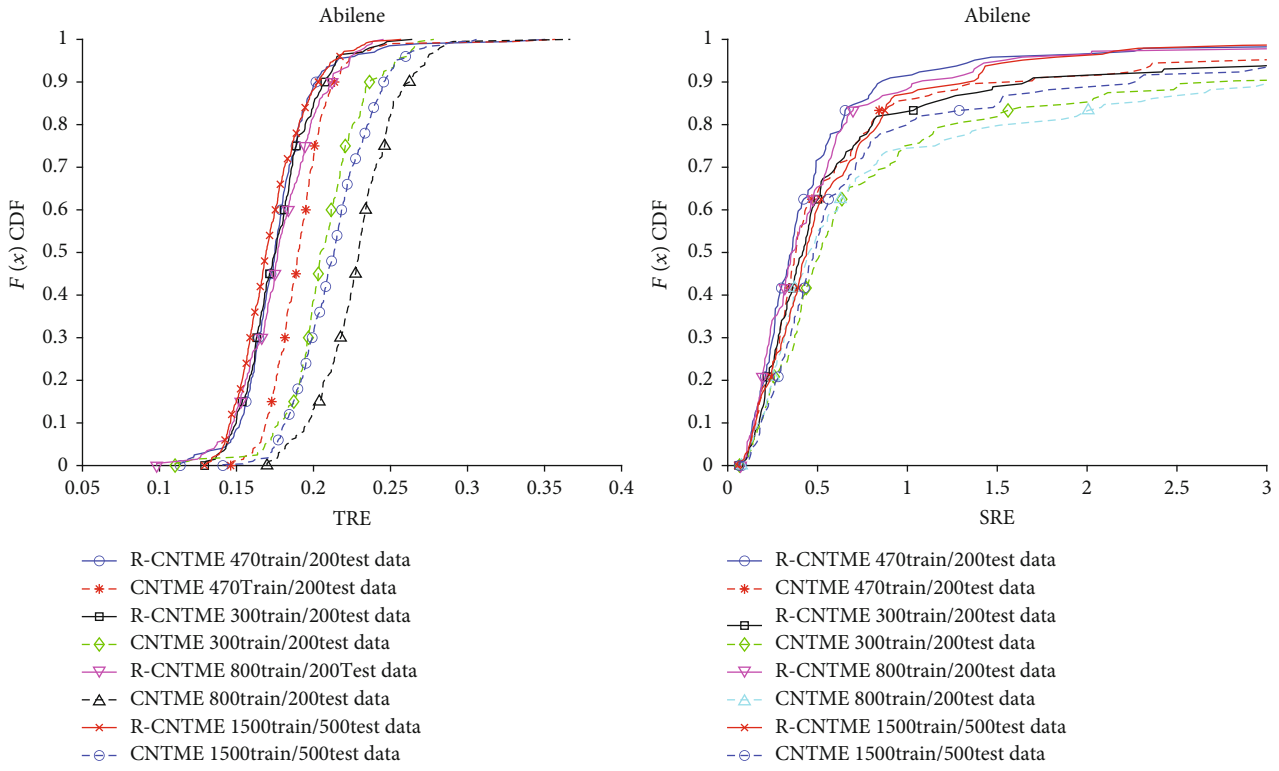


Figure 5: Comparison of performance of R-CNTME AND CNTME for the Abilene dataset for different train-test splits.

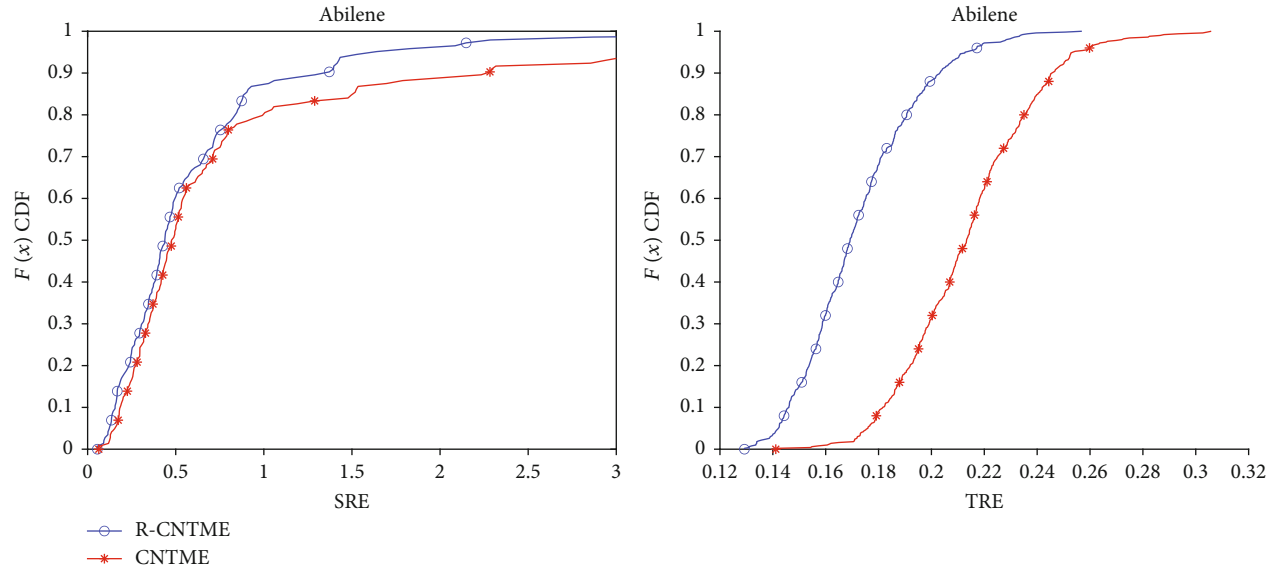


Figure 6: CDF of SRE and TRE for R-CNTME and CNTME for the Abilene dataset.

of the system at the cost of TRE performance since estimator performance becomes poorer on dataset outside of training data due to overfitting. It is pertinent to highlight here the fact that in the network traffic estimation problem, quick anomaly detection is one of the prime motivators of the development of these systems; hence, the estimator should have superior temporal prediction performance.

5.3. Comparison of Performance of R-CNTME and CNTME

5.3.1. Impact of Sparsity of Training Data. For this experiment, we generate different sparsity factors on the Abilene dataset. We synthetically modify the topology; so, it has 12 nodes, but the number of links is determined as per the sparsity factor. Sparsity factor can be defined as the number of

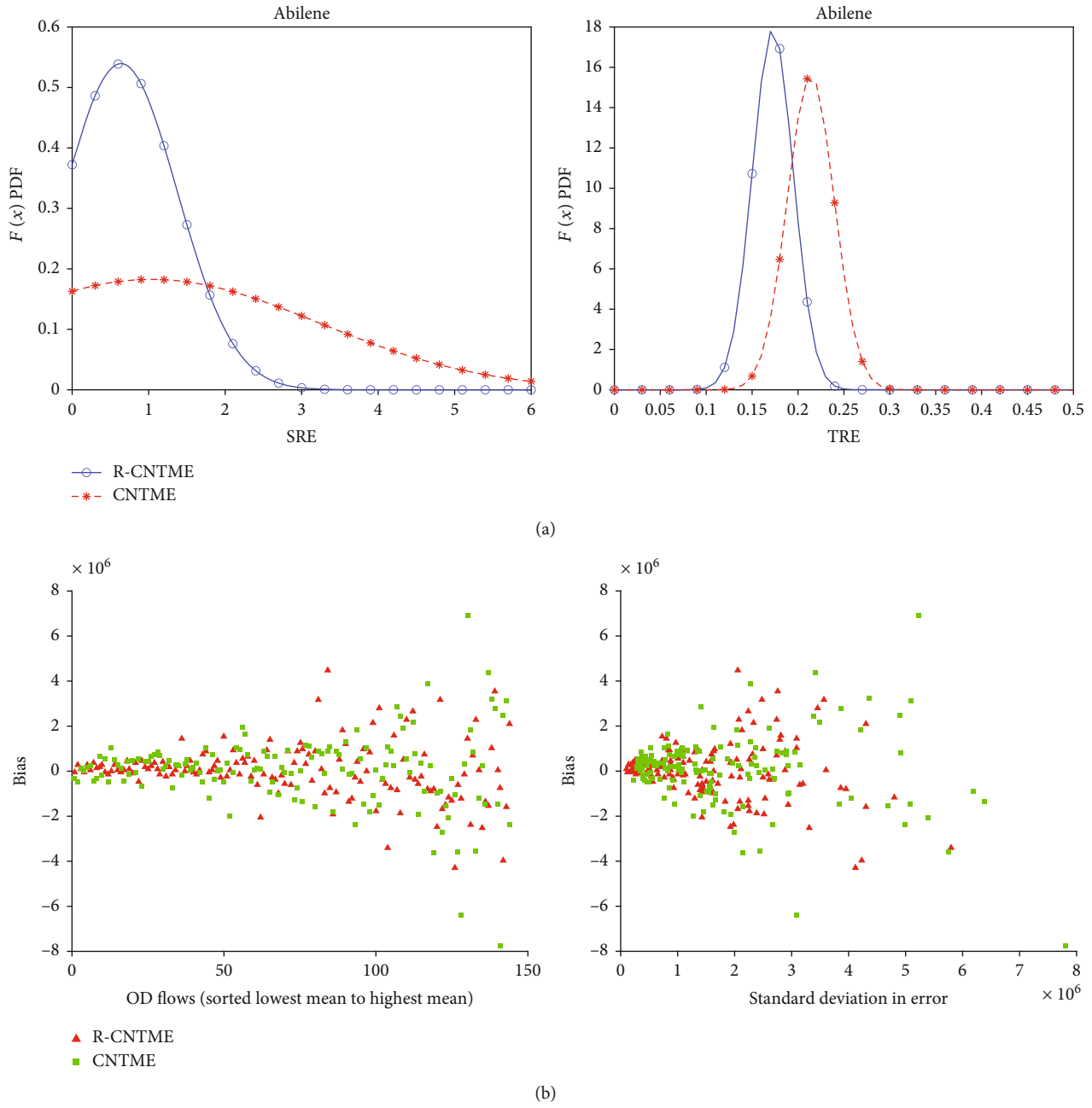


Figure 7: (a) PDF of SRE and TRE for R-CNTME and CNTME for the Abilene dataset. (b) Bias and standard deviation performance of R-CNTME and CNTME for the Abilene dataset.

zero entries to total number of entries in the L2AM training data. We ensure that the topology is sufficiently connected so that there is a path between each OD pair. Note that the OD traffic volumes between the 144 OD pairs are same as that of the original Abilene network; however, link counts are varied as per the simulated topology with variable sparsity factors. Figure 4 shows the results of this experiment. As the sparsity factor is reduced to 0.2, R-CNTME has superior TRE of 0.187 or lower for 90% of cases; for CNTME, the corresponding value is 0.27. On increasing sparsity factor to 0.6, the previous values increase to 0.2 and 0.29, respectively. SRE graphs show negligible degradation for R-CNTME and CNTME as sparsity is increased.

5.3.2. Impact of Size of Training Data. Figure 5 shows the SRE and TRE performance as the size of the training and test data is varied. It is observed from TRE results that as size of training data is increased, the TRE performance of R-CNTME is better. When the training data size is 1500, 90% of TRE values for R-CNTME are 0.19 or lower; when training data is reduced to 300, this increases to just 0.2. For CNTME, the TRE performance is lower at all ranges of training data

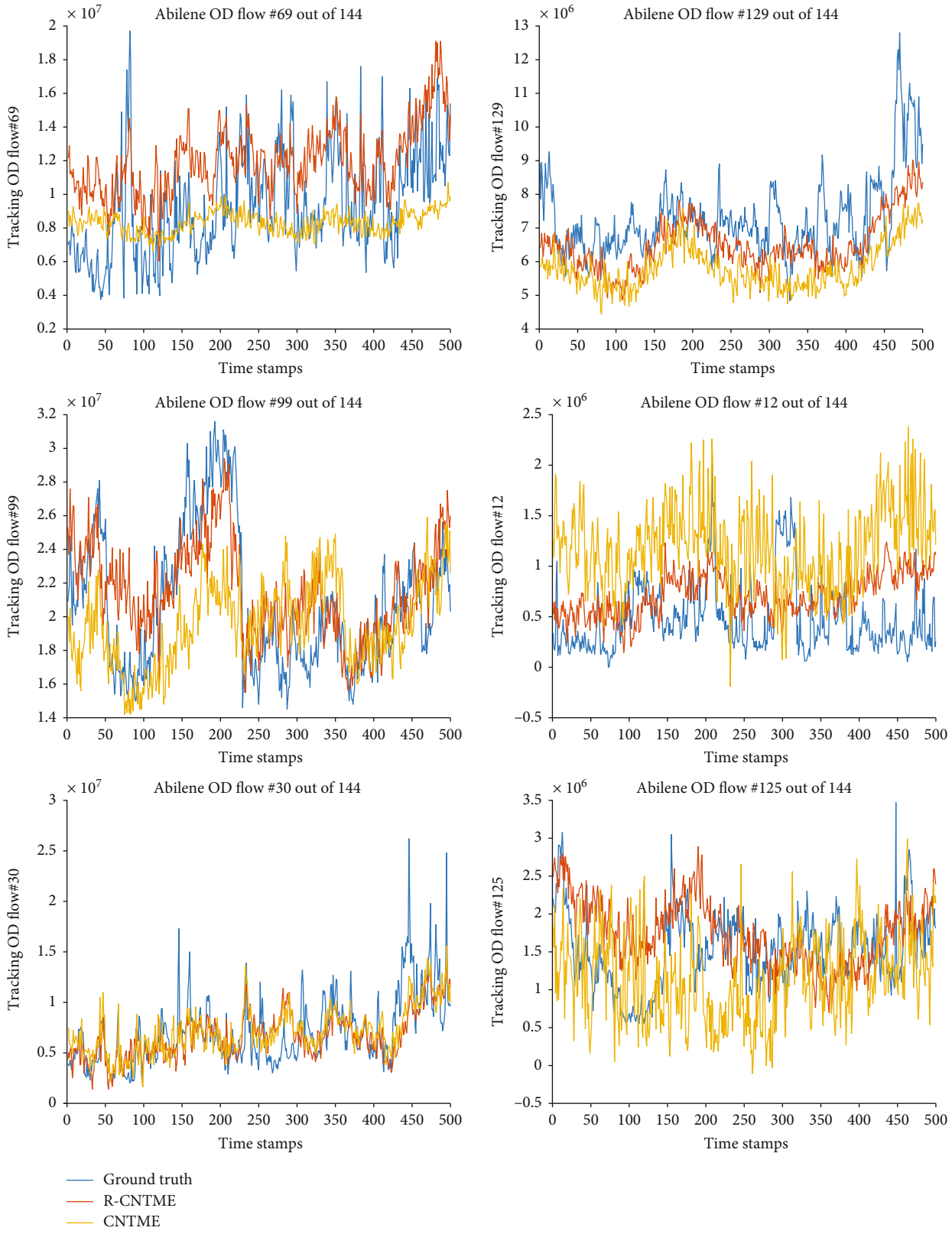


Figure 8: OD tracking performance.

sizes, and it also follows an erratic behavior due to the model being affected by underfitting and overfitting problem explained earlier. This shows the robustness of R-CNTME for limited training data availability too. The SRE performance of R-CNTME is also better than CNTME for all ranges of the training data size.

5.4. Performance Comparison of R-CNTME and CNTME on Clean Abilene Dataset. Figure 6 shows the CDF of the SRE and TRE for the clean Abilene dataset without any artificial effects of simulated errors or sparsity in the L2AM training data.

We see that the performance of R-CNTME is best for both TRE and SRE metrics. R-CNTME has a SRE value 1.5 or below for 95% of cases compared to 82% for CNTME. Similarly, the TRE performance of R-CNTME shows TRE values of less than 0.2 for 90% of the cases. The corresponding values for CNTME are 0.24. The SRE and TRE behavior is also shown as fitted to a normal pdf (Figure 7(a)). These graphs confirm the superior behavior of R-CNTME over CNTME. In addition, RCNTME is also displaying better bias and standard deviation performance than CNTME (Figure 7(b)).

5.5. OD Tracking Performance of R-CNTME and CNTME. Figure 8 shows the OD prediction performance when compared with the ground truth for six random OD flows. It is clear that R-CNTME displays superior behavior in OD flow prediction from link flow data compared to CNTME. It not only predicts the OD flows closely but also accurately tracks the anomalies in correspondence with the actual ground truth.

6. Conclusion

Accurate traffic estimation is very necessary in back-haul cloud networks for quick detection and prevention of an anomaly. In this paper, motivated by recent work for convolutional neural network-based network traffic matrix estimation, an architecture for a robust convolutional neural network-based traffic matrix estimator was proposed for Cloud Network Traffic Estimation which displays better performance in presence of sparse 2D training datasets and 2D datasets having errors and limited training dataset availability. The comprehensive simulations with real-world datasets reveal that the proposed architecture has stable performance with the training data artifacts and also exhibits superior error performance and anomaly detection performance.

Data Availability

We use the real dataset of the Abilene network (a backbone network in the United States) to evaluate the performance of our proposed technique.

Conflicts of Interest

The author(s) declare(s) that they have no conflicts of interest.

References

[1] S. O'Dea, "The economic value of Wi-Fi: A global view (2018 and 2023)," *Telecom Advisory Services; Cisco Systems*, p. 39, 2018, https://www.statista.com/statistics/995060/business-internet-traffic-in-the-us/.

[2] I. F. Akyildiz, A. Lee, P. Wang, M. Luo, and W. Chou, "A roadmap for traffic engineering in SDN-OpenFlow networks," *Computer Networks*, vol. 71, pp. 1–30, 2014.

[3] I. F. Akyildiz, A. Lee, P. Wang, M. Luo, and W. Chou, "Research challenges for traffic engineering in software defined networks," *IEEE Network*, vol. 30, no. 3, pp. 52–58, 2016.

[4] M. A. Saleh and A. Abdul Manaf, "Optimal specifications for a protective framework against HTTP-based DoS and DDoS attacks," in *2014 International Symposium on Biometrics and Security Technologies (ISBAST)*, pp. 263–267, Kuala Lumpur, Malaysia, Aug. 2014.

[5] J. Mao, X. Li, Q. Lin, and Z. Guan, "Deeply understanding graph-based Sybil detection techniques via empirical analysis on graph processing," *China Communications*, vol. 17, no. 10, pp. 82–96, 2020.

[6] L. Nie, D. Jiang, and Z. Lv, "Modeling network traffic for traffic matrix estimation and anomaly detection based on Bayesian network in cloud computing networks," *Annales des Telecommunications*, vol. 72, no. 5–6, pp. 297–305, 2017.

[7] M. Mardani and G. B. Giannakis, "Estimating traffic and anomaly maps via network tomography," *IEEE/ACM Transactions on Networking*, vol. 24, no. 3, pp. 1533–1547, 2016.

[8] N. Etemadi Rad, Y. Ephraim, and B. L. Mark, "Delay network tomography using a partially observable bivariate Markov chain," *IEEE/ACM Transactions on Networking*, vol. 25, no. 1, pp. 126–138, 2017.

[9] A. Medina, N. Taft, K. Salamatian, S. Bhattacharyya, and C. Diot, "Traffic matrix estimation: existing techniques and new directions," in *Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications - SIGCOMM '02*, p. 161, Pittsburgh, Pennsylvania, USA, 2002.

[10] D. Daniel-Simion and G. Dan-Horia, "Traffic shaping and traffic policing impacts on aggregate traffic behaviour in high speed networks," in *2011 6th IEEE International Symposium on Applied Computational Intelligence and Informatics (SACI)*, pp. 465–467, Timisoara, May 2011.

[11] D. Jiang, Z. Zhao, Z. Xu, C. Yao, and H. Xu, "How to reconstruct end-to-end traffic based on time-frequency analysis and artificial neural network," *AEU - International Journal of Electronics and Communications*, vol. 68, no. 10, pp. 915–925, 2014.

[12] F. Qian, G. Hu, and J. Xie, "A recurrent neural network approach to traffic matrix tracking using partial measurements," in *2008 3rd IEEE Conference on Industrial Electronics and Applications*, pp. 1640–1643, Singapore, June 2008.

[13] M. Emami, R. Akbari, R. Javidan, and A. Zamani, "A new approach for traffic matrix estimation in high load computer networks based on graph embedding and convolutional neural network," *Transactions on Emerging Telecommunications Technologies*, vol. 30, no. 6, 2019.

[14] K. Rajawat, E. Dall'Anese, and G. B. Giannakis, "Dynamic network kriging," in *2012 IEEE Statistical Signal Processing Workshop (SSP)*, pp. 121–124, Ann Arbor, MI, USA, Aug. 2012.

[15] K. Rajawat, E. Dall'Anese, and G. B. Giannakis, "Dynamic network delay cartography," *IEEE Transactions on Information Theory*, vol. 60, no. 5, pp. 2910–2920, 2014.

[16] L. Nie, "A novel network tomography approach for traffic matrix estimation problem in large-scale IP backbone networks," in *2015 International Conference on Computer Science and Mechanical Automation (CSMA)*, pp. 97–101, Hangzhou, China, Oct. 2015.

[17] M. Coates, Y. Pointurier, and M. Rabbat, "Compressed network monitoring for ip and all-optical networks," in *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement - IMC '07*, p. 241, San Diego, California, USA, 2007.

[18] S. Qazi, S. M. Atif, and M. B. Kadri, "A novel compressed sensing technique for traffic matrix estimation of software defined cloud networks," *KSII Transactions on Internet and Information Systems*, vol. 12, no. 10, 2018.

[19] P. Tune and M. Roughan, "Spatiotemporal traffic matrix synthesis," in *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, pp. 579–592, London United Kingdom, Aug. 2015.

[20] A. Soule, A. Nucci, R. Cruz, E. Leonardi, and N. Taft, "How to identify and estimate the largest traffic matrix elements in a dynamic environment," *ACM SIGMETRICS Performance Evaluation Review*, vol. 32, no. 1, pp. 73–84, 2004.

[21] C. Tebaldi and M. West, "Bayesian inference on network traffic using link count data: rejoinder," *Journal of the American Statistical Association*, vol. 93, no. 442, p. 576, 1998.

[22] Y. Vardi, "Network tomography: estimating source-destination traffic intensities from link data," *Journal of the American Statistical Association*, vol. 91, no. 433, pp. 365–377, 1996.

[23] D. B. Chua, E. D. Kolaczyk, and M. Crovella, "Network Kriging," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 12, pp. 2263–2272, 2006.

[24] X. Fan and X. Li, "Network tomography via sparse Bayesian learning," *IEEE Communications Letters*, vol. 21, no. 4, pp. 781–784, 2017.

[25] L. Nie and D. Jiang, "A compressive sensing-based network tomography approach to estimating origin-destination flow traffic in large-scale backbone networks," *International Journal of Communication Systems*, vol. 28, no. 5, pp. 889–900, 2015.