Traffic Flow Imputation Using Parallel Data and Generative Adversarial Networks

Yuanyuan Chen, Yisheng Lv, *Senior Member, IEEE*, and Fei-Yue Wang, *Fellow, IEEE*

***Abstract*— Traffic data imputation is critical for both research and applications of intelligent transportation systems. To develop traffic data imputation models with high accuracy, traffic data must be large and diverse, which is costly. An alternative is to use synthetic traffic data, which is cheap and easy-access. In this paper, we propose a novel approach using parallel data and generative adversarial networks (GANs) to enhance traffic data imputation. Parallel data is a recently proposed method of using synthetic and real data for data mining and data-driven process, in which we apply GANs to generate synthetic traffic data. As it is difficult for the standard GAN algorithm to generate time-dependent traffic flow data, we made twofold modifications: 1) using the real data or the corrupted ones instead of random vectors as latent codes to generator within GANs and 2) introducing a representation loss to measure discrepancy between the synthetic data and the real data. The experimental results on a real traffic dataset demonstrate that our method can significantly improve the performance of traffic data imputation.**

***Index Terms*— Parallel data, generative adversarial networks, traffic flow imputation, data augmentation, deep learning.**

I. Introduction

With the recent success in deep learning methods, large amounts of high-quality traffic data are becoming increasingly important to develop traffic models [1]–[5]. However, in practice, it is time-consuming and expensive to collect large-scale traffic data. And more often, traffic data collected from physical sensors in the real world are missing or corrupted due to detector failures. Thus, the task of imputing traffic data is required [6], [7].

Recently, parallel data, a new method, has been proposed to use synthetic and real data for data mining and data-driven processes [8]. The idea of using parallel data, i.e. synthetic data and real data, to develop a model has become appealing since the synthetic data can augment the real data automatically that complements the original data, which provide a way to obtain big data economically and help us to train robust and powerful models [9], [10]. For example, synthetic data have been applied in pose and gaze estimation tasks [11]. However, this problem is challenging as synthetic data may not capture characteristics of real data and there can be a huge gap between synthetic data and real data, which leads to models learned from synthetic data failing to work well. Therefore, we need to generate synthetic data as real as possible.

In this paper, we investigated the feasibility to impute traffic flow with synthetic data and real data simultaneously. Herein we call synthetic data and real data as parallel data, i.e. a mixed set of real data collected in the real world and artificial data generated based on real data. The starting point is to generate synthetic data with generative adversarial networks (GANs) taking real data as input, and then the parallel data are used to train the imputation model. We conducted a series of experiments to evaluate the imputation models on the parallel data and on the real data, respectively. Experiments showed that models trained on the parallel data have a higher accuracy.

Most successful examples of GANs focus on generating data especially in images and texts where the latent code fed into the generator does not need to embed time-serial dependence. However, those cannot be directly applied in generating traffic flow data as they have strong time-serial dependence. To address this issue, we propose a novel method to train GANs, which take original data or corrupted data as the input of the generator. In this paper, we used the synthetic data to augment the original data to help improve traffic flow imputation models, which needs to constrain the discrepancy between synthetic data and real data. Thus, we introduced a new loss function in the objective function of the generator, called representation loss, to decrease the reconstruction error between synthetic data and real data.

The main contributions of this paper are as follows:

(i) The parallel data paradigm, utilizing real data and synthetic data, was proposed to impute traffic flow. The algorithm to develop models with parallel data was given and we performed extensive experiments on real traffic dataset to investigate the effectiveness of the proposed method. Experimental results show that our method leads to significant improvements.

(ii) Different from existing variations of GANs, we proposed to take the original data or its corrupted data as the latent code and apply a representation loss to additionally optimize the generator.

The rest of this paper is organized as follows. Section II reviews related studies on traffic data imputation and GANs. Section III presents our approach to impute traffic data with parallel data paradigm. Section IV provides the experimental results to verify the effectiveness of our approach. Section V concludes this paper.

II. Related Works

In this section, we first review traffic data imputation algorithms. Then we present basics and recent applications of GANs.

A. Traffic Data Imputation

Traffic data imputation is targeted to estimate the missing or corrupted data points with known observed traffic data [7], [12]. Many methods have been proposed to solve traffic data imputation problem, which can be generally categorized to prediction/interpolation methods and statistical learning methods. The prediction/interpolation methods estimate the corrupted data with known historical data from the current and/or neighboring detector data. These methods include autoregressive integrated moving average [13], support vector regression [14], [15], k-Nearest neighbors [16], etc.

Manuscript received March 24, 2018; revised October 29, 2018; accepted March 31, 2019. Date of publication April 25, 2019; date of current version March 27, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant 61533019, Grant 61876011, and Grant U1811463. The Associate Editor for this paper was E. Kaisar. *(Corresponding author: Yisheng Lv.)*

Y. Chen is with the State Key Laboratory for Management and Control of Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, and also with the University of Chinese Academy of Sciences, Beijing 100190, China (e-mail: yychen5133@ia.ac.cn).

Y. Lv and F.-Y. Wang are with the State Key Laboratory for Management and Control of Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, and also with the Qingdao Academy of Intelligent Industries, Shandong 266109, China (e-mail: yisheng.lv@ia.ac.cn; feiyue@ieee.org).

Digital Object Identifier 10.1109/TITS.2019.2910295

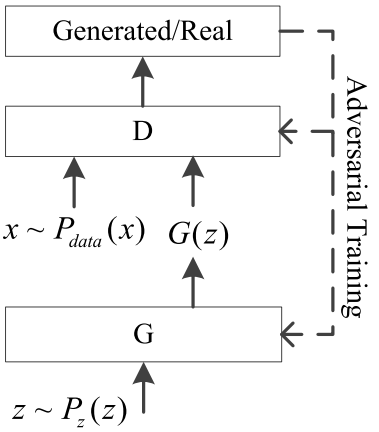

Fig. 1. Architecture of generative adversarial networks.

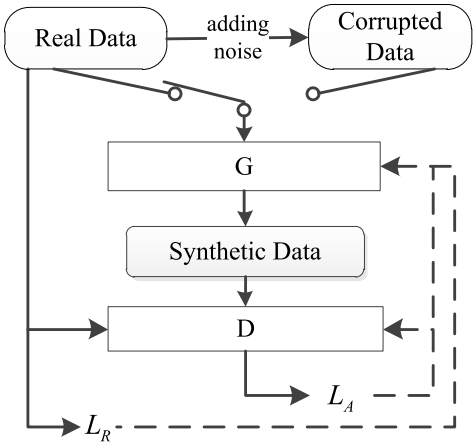

Fig. 2. Architecture of the proposed scheme to generate synthetic data.

The statistical learning methods estimate the corrupted data by learning the statistical characteristics of traffic data. Markov chain Monte Carlo and neural networks are two representative statistical methods [7], [17]–[20].

B. Generative Adversarial Networks

GANs are a class of generative models which can generate synthetic data that resemble or are to some extent related to the training data, which works by mimicking the distribution of real data in mathematics. GANs have been applied in generating images, videos and language [21], [22]. GANs provide a powerful framework to estimate generative models by adversarial training. Fig. 1 shows the basic structure of GANs, which have two competing components, named as the generator G and the discriminator D, respectively. Typical models for G and D are deep neural networks. The generator G learns to map given samples from a standard random distribution to samples that are drawn from the same distribution as the training data. The discriminator D takes in samples both from the true data and the artificial data generated by G. The generator G tries to fool the discriminator D to not classify generated samples as real, while the discriminator D tries to classify them as real or fake. Ideally, the discriminator D cannot distinguish the generated samples and real data when the generator G is well trained.

The generator G and the discriminator D are simultaneously trained as a minimax two-player game. In practice, G and D are trained in an alternating manner. Formally, let $P_z(z)$ be the input standard distribution and $P_{data}(x)$ be the training data distribution, then the minimax objective of GANs is defined as

$$\min_G \max_D E_{x\sim P_{data}(x)}[\log D(x)] + E_{z\sim P_z(z)}[\log(1 - D(G(z)))] \tag{1}$$

GANs implicitly model high-dimensional distributions of data and provide a methodology for probability density estimation, which make it flexible for both semi-supervised and unsupervised learning. GANs have become a popular topic since the year of 2014 and there are many variants of GANs in theory and application. In the field of theory and algorithm development, researchers focus on training methods and structures of GANs. Among these works, Wasserstein GAN and Loss-sensitive GAN are two significant improvements to train GANs by applying alternative cost functions [23], [24]. And conditional GAN, and InfoGAN are two significant advances to explore the structures of GANs [25], [26]. In the field of application, GANs have been applied to image dataset synthesis [11], image-to-image style translation [27], image super-resolution [28], dialogue generation [29], etc.

III. Methodology

To use the parallel data paradigm for traffic data imputation, we need firstly to generate synthetic data, and then develop imputation models with the real data and synthetic data. In this section, we first present generating synthetic traffic data with our improved GANs where we introduce the representation loss in the objective function. Then we present the data generating process and the method to train traffic data imputation models with parallel data.

A. Synthetic Data Generation

Intuitively, due to the time-serial dependence in traffic flow data, the latent code fed into the generator should also embed such dependence. Inspired by this idea, it is convenient to use the original data as the latent code. Besides, the corrupted data, created by adding small scale noises to the original data, is also suitable to be used as the latent code. These two kinds of latent codes can be fed into the generator G to generate synthetic data as shown in Fig. 2. The discriminator D takes real data and synthetic data and assigns a label to indicate its input coming from real data or synthetic data. The probability of assigning the correct label is defined as L_A, which is used to optimize the generator G and the discriminator D. As illustrated in the bottom of Fig. 2, to decrease the reconstruction error between real data samples and synthetic data samples, we propose to apply a representation loss L_R to optimize the generator G.

1) Latent Code Fed Into the Generator: The generator G is a deterministic function that maps the input space into the space of synthetic data. To generate synthetic data, previous GANs-based works utilized an n-dimensional vector, each element of which is a random sample from a particular distribution, as the input fed into G. While in the field of transportation, the data points are usually time dependent, i.e. they are time-series data, each element of the random input vector fed into the generator utilized in previous works, is independent from others. Thus, this is not a proper way to generate time-serial data. To generate diverse and realistic-like time serial data, obviously the input space should: (i) have large-scale and diverse input samples, and (ii) embed temporal dependence.

To fulfill the above two points, we propose to utilize the real data with or without adding noise as the input data as shown in Fig. 2. Here, we explain this intuition more specifically. The real data is

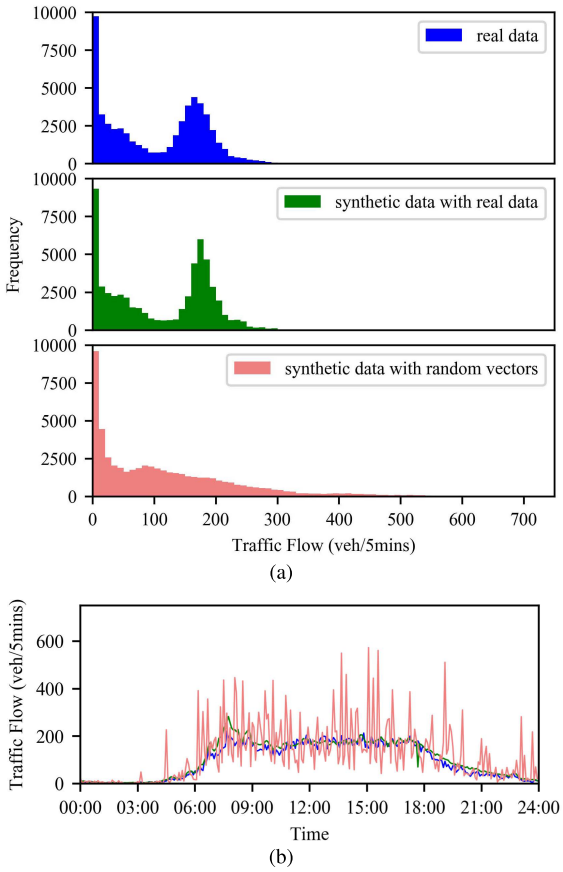

Fig. 3. Histogram and time series of synthetic data by GANs with real data and random vectors as latent codes. (a) Histogram of traffic flow. (b) Traffic flow series.

obviously time series but usually volume-limited that is a reason to apply parallel data. To obtain enough samples, motivated by the denoising autoencoder, we can add noises to the original data x to obtain its corrupted data $\tilde{x}$ by means of a stochastic mapping $\tilde{x} \sim q(\tilde{x}|s;\theta)$. Fig. 3 shows synthetic traffic flow data with random vectors and real data, respectively. Fig. 3(a) presents the histogram of real data, synthetic data with real data, and synthetic data with random vectors, and Fig. 3(b) shows their time series. Clearly, random vectors in previous GANs are not suitable to generate traffic data, and using real traffic flow data as the input can be used to generate data with similar patterns.

We have described the method of utilizing the real data with adding noises as the input of the generator. As for the method of utilizing the real data without adding noises, it can be done by firstly feeding the original data into the generator and thus obtaining the generated synthetic data, which then will be fed into the generator to get new synthetic data in a loop manner. The difference of these two methods is in the data generating stage, and more details are given in section Traffic data imputation with parallel data.

2) Representation Loss: In this study, our target is to generate artificial data to augment the real data, so the generated data samples are desired to resemble the corresponding input samples. Recall that in GANs, the generated data samples are desired to be realistic samples from the original dataset, where there is no constraint for a generated sample to resemble or be related to which input samples. To tackle this problem, we propose to apply a representation loss to constraint the generated sample to be similar to corresponding input samples. The representation loss is to measure the discrepancy

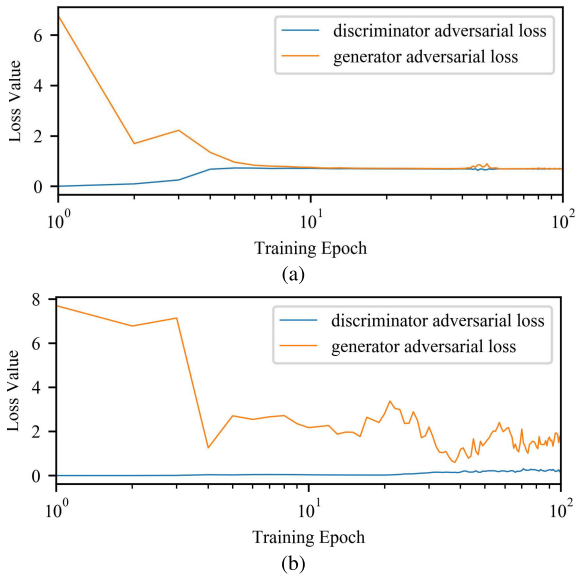

Fig. 4. Loss value evolution over training epoch. (a) With representation loss. (b) Without representation loss.

between the generate sample and its input sample. Decreasing the representation loss would penalize the generated samples that does not resemble their corresponding input samples. And according to our experiments, decreasing representation loss also helps accelerate the convergence of generator. Formally, we define the representation loss as

$$L_R = V(G(x), x) \tag{2}$$

where V is the function to map the difference between the input sample x and the generated sample $G(x)$. A convenient choice for is the ℓ_1 norm, and the representation loss becomes

$$L_R = \|G(x) - x\|_1 \tag{3}$$

Our motivation of applying the representation loss is to constrain the synthetic data to resemble the real data, which is necessary for the tasks such as traffic flow imputation and prediction with augmented data. Fig. 4 shows the adversarial loss values evolution over training epochs. According to equations (4) and (5), when loss functions of the generator and the discriminator converge to a same value indicates the discriminator cannot identify whether traffic flow data is from a generated dataset or a real dataset. Thus, we can find that the advantages of using representative loss are to accelerate convergence. And the following section of experiments also illustrates the improvements by applying representation loss.

3) Algorithm and Optimization: We used the stochastic gradient descent (SGD) method to optimize the objective function. The parameter set of a generator network is updated by minimizing the following loss function:

$$L_G(\varphi) = \sum_j [\log(1 - D(G_\varphi(x_j))) + \lambda \left\| G_\varphi(x_j) - x_j \right\|_1] \tag{4}$$

where x_j belongs to a batch of training instances and λ is a parameter to balance between the adversarial loss and the representation loss. For a discriminator network, the parameter set ϕ is updated by minimizing the following loss function:

$$L_D(\phi) = -\sum_i \log(D_\phi(x_i)) - \sum_i \log(1 - D_\phi(G(x_j))) \tag{5}$$

Algorithm 1 GANs Model Training

Input: original dataset $X(m \times n)$, balance parameter λ
Output: the learned GANs model with parameters Θ
initial all the parameters
//pre-train D
generate data $\tilde{X}$ with X
train D by minimizing the loss function (5)
//adversarial training
repeat
  randomly select a bath of instances X_b from X
  generate data $\tilde{X}_b$ with X_b
  train D by minimizing the loss function (5)
  train G by minimizing the loss function (4) while D fixed
until stopping criteria is met

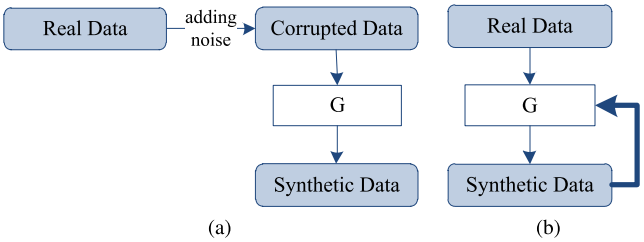

Fig. 5. Data generating schemes.

Algorithm 1 outlines the process of training GANs to generate traffic data for imputation. The discriminator is first pre-trained to learn the features of generated data and real data. Then the discriminator and the generator are adversarially trained.

B. Traffic Data Imputation With Parallel Data

One critical step in training traffic data imputation models with parallel data is to generate synthetic traffic data which augment the original dataset. Thus, we need to provide a large volume of latent code as the input of the generator of GANs. In previous section, we have proposed two schemes: one is to take corrupted data, built by adding noises to real data, as the input of the generator G; and the other is to use the real data as the input of the generator G to generate new synthetic data iteratively. For utilizing corrupted data as input, it is convenient to obtain a large volume of input data by adding noise vectors to the original data. And the synthetic data is generated by feeding different corrupted data to G as shown in Fig. 5(a). While for taking original data without adding noise as the input of the generator G, the generated data is volume-limited considering that one real data vector can only be used to generate one synthetic data vector. To solve this problem, we propose to use the synthetic data as the input in next generating process as shown in Fig. 5(b), which is implemented in a loop manner and make it possible to generate a large volume of data.

Algorithm 2 describes the training process of traffic data imputation models with parallel data. The first step is to build the training dataset, which consists of two major components: artificially masking the complete data to be corrupted or missing and normalizing the data by scaling between 0 and 1. The second step is to fit the imputation model with parallel data. There are three stages in this process, namely fitting the model with the corrupted data of original dataset, fitting the model with the corrupted data of synthetic dataset, and refine the model with the corrupted data of original dataset.

Algorithm 2 Traffic Data Imputation Model Training With Parallel Data

Input: original complete traffic dataset $X(m \times n)$, generator G, missing rate α
Output: the learned imputation model with parameters Ψ
Step 1: construct training instances by artificially corrupting the original data
normalize the values in dataset into the range (0, 1]
generate $m \times n$ uniformly distributed random numbers in the interval (0, 1)
mark the element in dataset with zero if the corresponding random number at the same position is less than α
obtain training instances $\tilde{X}$
Step 2: train the imputation model
initialize all parameters Ψ
fit the model with X and $\tilde{X}$
repeat
  generate data X_{syn} with G
  corrupt X_{syn} as in Step 1 and obtain $\tilde{X}_{syn}$
  fit the model with synthetic data X_{syn} and $\tilde{X}_{syn}$
until augmenting count is met
refine the imputation model with X and $\tilde{X}$

IV. Experiments

A. Dataset and Experiments Settings

1) Dataset Description: In this paper, we evaluated the proposed method on traffic flow datasets obtained from Caltrans Performance Measurements Systems (PeMS) [30]. Caltrans PeMS has placed over 39,000 individual detectors spanning the freeway system across all major metropolitan areas of the State of California and its dataset is widely used for researcher to develop and evaluate traffic models. The proposed method is applied to the 5-min traffic flow data of District 5 in the whole year of 2013 except two days due to missing points, which is the same as our previous study [7]. There are 153 vehicle detector stations in this district, and we used the data of 147 vehicle detector stations among them as there are null numbers in the data of the other 6 stations.

2) Preprocessing: In this study, we assumed that the data directly obtained from Caltrans PeMS are accurate and there are no corrupted or missing points in them. Then the missing points are randomly chosen over the whole dataset by generating a same-sized matrix of random numbers ranging from 0 to 1. And if one data point's random number is less than the missing rate, it is set to be corrupted and marked with zero. Traffic patterns on weekdays differ from those on non-weekdays in the dataset, which is shown in Fig. 6. We trained traffic imputation models for these two kinds of traffic patterns separately. As non-weekdays may consist of weekends and holidays or other days, e.g. the day of 2013-01-21 is Monday and the federal holiday, it is more convenient to determine which days have weekday patterns by applying a k-means clustering algorithm other than the statement of holidays and weekends [7]. Finally, the data of 244 days show weekday patterns, and the data of the other 119 days show non-weekday patterns. For weekday patterns, the data of former 195 days are split into the training set and the data of the remaining 49 days are split into the test set. For non-weekday patterns, the data of former 95 days are split into the training dataset and the data of the remaining 24 days are split into the test dataset.

3) Model Configurations: We used Tensorflow, an open source software library for numerical computation, to build our models and evaluate their performances. As shown in Fig. 7(a), the generator network is built by stacking 3 fully connected layers. We used the

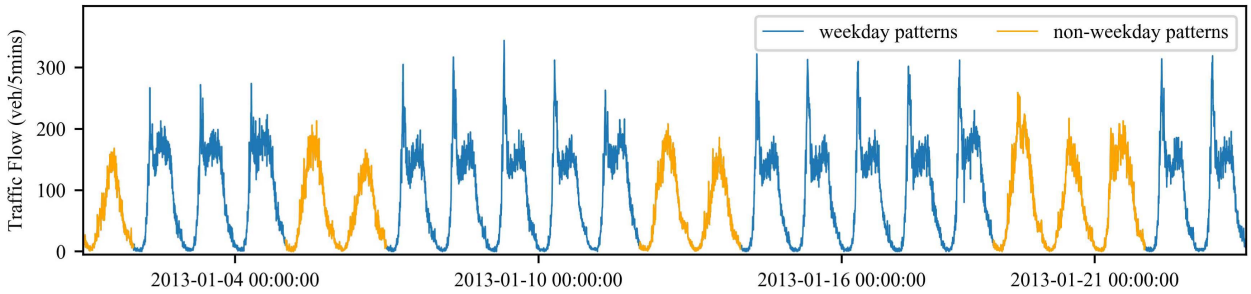

Fig. 6. Traffic flow series of VDS 500010102 between 2013-01-01 00:00:00 and 2013-01-23 23:55:00.

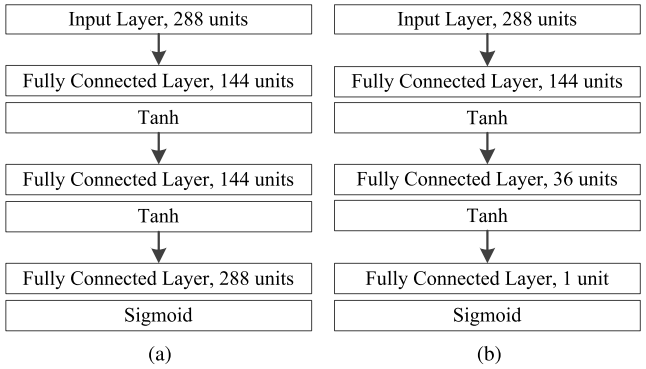

Fig. 7. An Implementation of GANs for generating data. (a) Generator architecture. (b) Discriminator architecture.

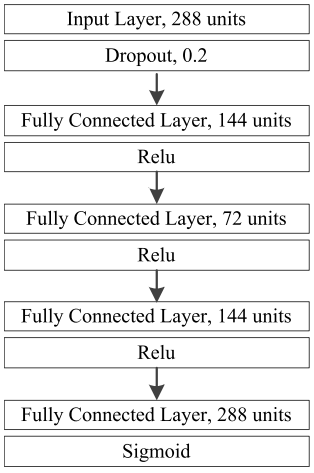

Fig. 8. An implementation of neural network for traffic flow imputation.

Adam algorithm to optimize gradient descent of the generator and set the learning rate to 0.0001. The discriminator network is illustrated in Fig. 7(b). It is also built by stacking 3 fully connected layers and the Adam optimizer is used, and the learning rate is set to 0.0002. The imputation model is built as an autoencoder network as in [7], which show advantages on traffic flow data imputation, but we used ReLU as the activation function in the intermediate layers, as shown in Fig. 8. And we used Adadelta [31] optimizer and the learning rate is set to 1.0.

4) Evaluation Metrics: To evaluate the effectiveness of the proposed method, we adopted three performance metrics, namely, the mean absolute error (MAE), the root mean square error (RMSE)

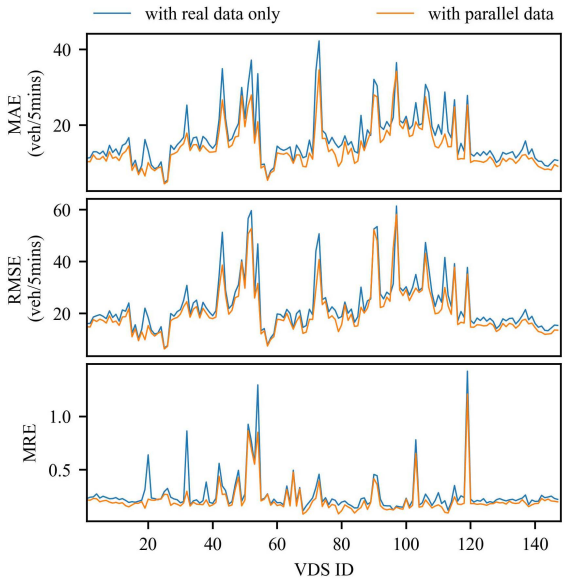

Fig. 9. Performance of traffic flow imputation with real data only and parallel data for different VDS at weekdays.

and the mean relative error (MRE), which are defines as:

$$MAE = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N} I_{ij}\left|x_{ij} - y_{ij}\right|}{\sum_{i=1}^{M}\sum_{j=1}^{N} I_{ij}} \tag{6}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{M}\sum_{j=1}^{N} I_{ij}(x_{ij} - y_{ij})^2}{\sum_{i=1}^{M}\sum_{j=1}^{N} I_{ij}}} \tag{7}$$

$$MRE = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N} I_{ij}\frac{\left|x_{ij}-y_{ij}\right|}{x_{ij}}}{\sum_{i=1}^{M}\sum_{j=1}^{N} I_{ij}} \tag{8}$$

where x_{ij} is the observed value before corrupted, and y_{ij} is the predicted value. M is the number of days in the test set and N is the number of slots in a single day. I is an indicator function define as:

$$I_{ij} = \begin{cases} 1 & \text{if data point } x_{ij} \text{ is corrupted} \\ 0 & \text{else} \end{cases} \tag{9}$$

B. Performance of Traffic Flow Imputation

To evaluate the improvement brought by parallel data paradigm, we compared traffic flow imputation performances with parallel data and with real data only. We designed experiments with missing rate 0.3, 0.4, 0.5, 0.6, 0.7 and 0.8 for each VDS, respectively. The performance for each VDS is different, and we firstly give the results with missing rate 0.3 at weekdays in Fig. 9. It can be observed that MAE, RMSE and MRE measures are all improved for every VDS

TABLE I
PERFORMANCE OF TRAFFIC FLOW IMPUTATION WITH ORIGINAL DATA, AND PARALLEL DATA

Missing Rate	Original Data			Parallel Data without Representation Loss			Parallel Data with Representation Loss		
	MAE	$RMSE$	MRE	MAE	$RMSE$	MRE	MAE	$RMSE$	MRE
0.3	18.1865	27.9297	0.3097	**14.4347**	**22.9350**	**0.2322**	14.6927	23.3517	0.2396
0.4	18.2405	27.9496	0.3110	15.8069	24.6357	0.2604	**15.0609**	**23.7471**	**0.2451**
0.5	18.2813	27.9496	0.3113	16.2155	25.2418	0.2684	**15.6254**	**24.6242**	**0.2563**
0.6	18.1981	27.8290	0.3074	16.6102	25.7980	0.2742	**15.8005**	**24.8771**	**0.2566**
0.7	18.3140	27.9890	0.3082	17.0500	26.4220	0.2824	**16.2863**	**25.5696**	**0.2647**
0.8	18.4796	28.3563	0.3102	17.6099	27.2840	0.2921	**17.0209**	**26.6346**	**0.2792**

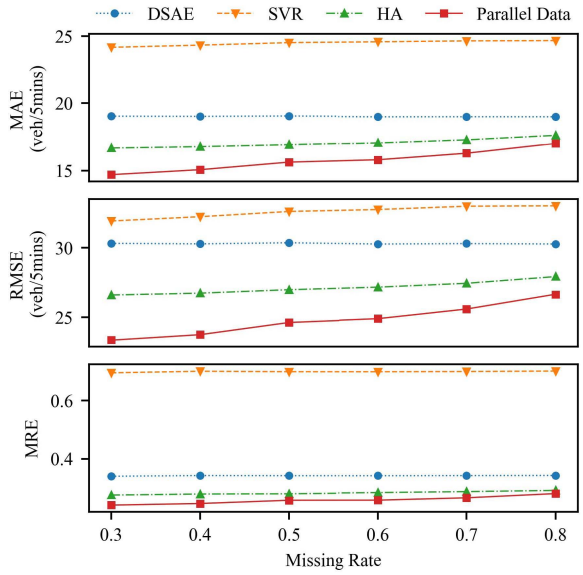

Fig. 10. Comparisons on performance of DSAE, SVR, HA and parallel data approach.

by parallel data paradigm against utilizing real data only. Then we evaluated the overall imputation performance. There are 147 VDSs used in this paper, and we conducted experiments for weekdays and weekend separately, so that 294 GANs models and traffic flow imputation models are developed. The overall imputation performance is given in Table I. It is clear that parallel data paradigm work well to improve the performance of traffic flow imputation for all the tasks conducted in this paper. Taking the task with missing rate 0.3 as example, the parallel data paradigm decreases the MAE from 18.1865 to 14.6927, which improves 19.21%; For RMSE and MRE, the percentage improvement is 16.39% and 46.39%, respectively; And the approach of parallel data without applying representation loss achieves slightly better performance than the approach of parallel data with applying representation loss at missing rate 0.3. While in other five tasks, the parallel data approach with applying representation loss achieve the best performance.

For comparison, we have conducted a series experiments to impute traffic flow by applying different competitive approaches, including support vector regression (SVR) model, History Average (HA) model, Denoising Stacked Auto-encoder (DSAE) model. To make a fair comparison, we used the same train dataset and test dataset in all experiments, and the predictions of corrupted data points were treated as the observations if required to impute following data points in test dataset. In this paper, we tuned the parameters of SVR and HA by ten-fold cross validation with grid search, and the parameters of DSAE were kept the same with [7]. Fig. 10 shows the comparing results, from which we can find out that parallel data approach achieves the lowest MAE, RMSE and MRE for all missing rate. With the increasing of missing ratio, the performances of SVR, HA and Parallel Data approaches become slight worse while DSAE approach remains nearly the same performance. Given a lower missing rate, Parallel Data approach improves imputation performance more significantly. Taking RMSE as an example, the percent decrease by proposed approach is 12.2% compared to HA approach at missing rate of 0.3, and this number is 4.6% at missing rate of 0.8.

V. CONCLUSION

In this paper, we proposed applying parallel data (synthetic data + real data) paradigm for traffic data imputation. To generate synthetic time-serial traffic data, we proposed to use the real data as the latent code to generator, and introduced a representation loss within GANs framework. We evaluated the performances of the proposed approach on traffic flow data from Caltrans PeMS, and experimental results showed that the proposed method leads to a great improvement in traffic flow data imputation. The advantages of our contributions are to generate time dependent series in an easy and quick way, and to develop imputation models using limited volume of real traffic data and large volume of synthetic traffic data in an economic way. The method developed in this paper can be applied in serial data generation tasks and other traffic data mining applications, especially for a system with limited volume of data, such as a newly deployed system, to tackle the so-called cold start issue. In the future, we plan to explore more methods to generate synthetic traffic data such as variational autoencoders (VAE) approach, and use the proposed method to tackle other tasks in the field of intelligent transportation systems, such as traffic prediction and traffic control.

REFERENCES

[1] F.-Y. Wang, "Parallel control and management for intelligent transportation systems: Concepts, architectures, and applications," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 3, pp. 630–638, Sep. 2010.
[2] J. Zhang, F.-Y. Wang, K. Wang, W.-H. Lin, X. Xu, and C. Chen, "Data-driven intelligent transportation systems: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 4, pp. 1624–1639, Dec. 2011.
[3] Y.-Y. Chen, Y. Lv, Z. Li, and F.-Y. Wang, "Long short-term memory model for traffic congestion prediction with online open data," in *Proc. IEEE 19th Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2016, pp. 132–137.
[4] X. Ma, Z. Tao, Y. Wang, H. Yu, and Y. Wang, "Long short-term memory neural network for traffic speed prediction using remote microwave sensor data," *Transp. Res. C, Emerg. Technol.*, vol. 54, pp. 187–197, May 2015.

[5] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: A deep learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 865–873, Apr. 2015.
[6] L. Qu, L. Li, Y. Zhang, and J. Hu, "PPCA-based missing data imputation for traffic flow volume: A systematical approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 3, pp. 512–522, Sep. 2009.
[7] Y. Duan, Y. Lv, Y.-L. Liu, and F. Wang, "An efficient realization of deep learning for traffic data imputation," *Transp. Res. C, Emerg. Technol.*, vol. 72, pp. 168–181, Nov. 2016.
[8] X. Liu, X. Wang, W. Zhang, J. Wang, and F. Y. Wang, "Parallel data: From big data to data intelligence," *Pattern Recognit. Artif. Intell.*, vol. 30, no. 8, pp. 673–681, 2017.
[9] K. Wang, C. Gou, Y. Duan, Y. Lin, X. Zheng, and F.-Y. Wang, "Generative adversarial networks: Introduction and outlook," *IEEE/CAA J. Autom. Sinica*, vol. 4, no. 4, pp. 588–598, Sep. 2017.
[10] F.-Y. Wang, "Artificial intelligence and intelligent transportation: Driving into the 3rd axial age with ITS," *IEEE Intell. Transp. Syst. Mag.*, vol. 9, no. 4, pp. 6–9, Oct. 2017.
[11] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from simulated and unsupervised images through adversarial training," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 2242–2251.
[12] Y. Li, Z. Li, and L. Li, "Missing traffic data: Comparison of imputation methods," *IET Intell. Transp. Syst.*, vol. 8, no. 1, pp. 51–57, Feb. 2014.
[13] A. Guin, "Travel time prediction using a seasonal autoregressive integrated moving average time series model," in *Proc. IEEE Intell. Transp. Syst. Conf.*, Sep. 2006, pp. 493–498.
[14] M. Castro-Neto, Y.-S. Jeong, M.-K. Jeong, and L. D. Han, "Online-SVR for short-term traffic flow prediction under typical and atypical traffic conditions," *Expert Syst. Appl.*, vol. 36, no. 3, pp. 6164–6173, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957417408004740
[15] X. Jin, Y. Zhang, and D. Yao, "Simultaneously prediction of network traffic flow based on PCA-SVR," in *Advances in Neural Networks*, D. Liu, S. Fei, Z. Hou, H. Zhang, and C. Sun, Eds. Berlin, Germany: Springer, 2007, pp. 1022–1031.
[16] O. Troyanskaya *et al.*, "Missing value estimation methods for DNA microarrays," *Bioinformatics*, vol. 17, no. 6, pp. 520–525, 2001.
[17] J. Farhan and T. F. Fwa, "Airport pavement missing data management and imputation with stochastic multiple imputation model," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2336, no. 1, pp. 43–54, 2013. doi: 10.3141/2336-06.
[18] M. Zhong, S. Sharma, and P. Lingras, "Genetically designed models for accurate imputation of missing traffic counts," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1879, pp. 71–79, Jan. 2004.
[19] J. M. de M. Goulart, A. Y. Kibangou, and G. Favier, "Traffic data imputation via tensor completion based on soft thresholding of Tucker core," *Transp. Res. C, Emerg. Technol.*, vol. 85, pp. 348–362, Dec. 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0968090X17302516
[20] B. Bae, H. Kim, H. Lim, Y. Liu, L. D. Han, and P. B. Freeze, "Missing data imputation for traffic flow speed using spatio-temporal cokriging," *Transp. Res. C, Emerg. Technol.*, vol. 88, pp. 124–139, Mar. 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0968090X18300469
[21] M. Arjovsky and L. Bottou. (Jan. 2017). "Towards principled methods for training generative adversarial networks." [Online]. Available: https://arxiv.org/abs/1701.04862
[22] I. J. Goodfellow *et al.* "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2014, pp. 2672–2680.
[23] M. Arjovsky, S. Chintala, and L. Bottou. (Jan. 2017). "Wasserstein GAN." [Online]. Available: https://arxiv.org/abs/1701.07875
[24] G.-J. Qi. (2017). "Loss-sensitive generative adversarial networks on Lipschitz densities." [Online]. Available: https://arxiv.org/abs/1701.06264
[25] M. Mirza and S. Osindero. (2014). "Conditional generative adversarial nets." [Online]. Available: https://arxiv.org/abs/1411.1784
[26] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2172–2180.
[27] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1125–1134.
[28] C. Ledig *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 105–114.
[29] J. Li, W. Monroe, T. Shi, S. Jean, A. Ritter, and D. Jurafsky, "Adversarial learning for neural dialogue generation," in *Proc. Conf. Empirical Methods Natural Lang. Process.* Copenhagen, Denmark: Association for Computational Linguistics, Sep. 2017, pp. 2157–2169. [Online]. Available: https://www.aclweb.org/anthology/D17-1230
[30] PeMS. *Caltrans Performance Measurement System (PeMS)*. Accessed: Apr. 1, 2019. [Online]. Available: http://pems.dot.ca.gov/
[31] M. D. Zeiler. (2012). "ADADELTA: An adaptive learning rate method." [Online]. Available: https://arxiv.org/abs/1212.5701