IGANI: Iterative Generative Adversarial Networks for Imputation With Application to Traffic Data

AMIR KAZEMI AND HADI MEIDANI[iD]

Department of Civil and Environmental Engineering, University of Illinois at Urbana–Champaign, Champaign, IL 61801, USA

Corresponding author: Hadi Meidani (meidani@illinois.edu)

ABSTRACT Increasing use of sensor data in intelligent transportation systems calls for accurate imputation algorithms that can enable reliable traffic management in the occasional absence of data. As one of the effective imputation approaches, generative adversarial networks (GANs) are implicit generative models that can be used for data imputation, which is formulated as an unsupervised learning problem. This work introduces a novel iterative GAN architecture, called Iterative Generative Adversarial Networks for Imputation (IGANI), for data imputation. IGANI imputes data in two steps and maintains the invertibility of the generative imputer, which will be shown to be a sufficient condition for the convergence of the proposed GAN-based imputation. The performance of our proposed method is evaluated on (1) the imputation of traffic speed data collected in the city of Guangzhou in China, and the training of short-term traffic prediction models using imputed data, and (2) the imputation of multi-variable traffic data of highways in Portland-Vancouver metropolitan region which includes volume, occupancy, and speed with different missing rates for each of them. It is shown that our proposed algorithm mostly produces more accurate results compared to those of previous GAN-based imputation architectures.

INDEX TERMS Generative adversarial networks, GAN, missing data, imputation, invertible neural networks (INN).

I. INTRODUCTION

Traffic data is generated faster than before as intelligent transportation systems (ITS) develop. Various and numerous operating sensors from road-side cameras to vehicles equipped with GPS tracking devices underscore the big-data features of traffic records. Traffic management and control highly depend on the collected data which is usually incomplete because of sensor or transmission failures [1], [2]. Imputation of traffic data is therefore crucial for an informative presentation of them as well as for training prediction models based on them. The choice of method for incomplete data analysis is of critical importance as it directly affects the conclusion validity. The methods for handling missing data ranges from naive deletion of incomplete samples to modern machine learning techniques for imputation. The propriety of a method for missing data analysis depends on the missing rate, missingness mechanism, data type and size, and the acceptable

The associate editor coordinating the review of this manuscript and approving it for publication was Razi Iqbal[iD].

accuracy of imputation or of downstream classification and regression tasks based on the imputed data. [3].

Myriads of techniques have been proposed for the imputation of missing data which are categorized into statistical and machine-learning based techniques as well as those which are tailored for traffic data to address their spatial-temporal correlations. In the past decade, statistical techniques including mean/mode, linear regression, least square, and expectation maximization have been outperformed by machine-learning based techniques with respect to accuracy and applicability to different missing mechanisms. These statistical methods, such as the mean/mode method, are now being implemented to rather serve as baseline for comparative studies [4]–[6]. Major machine learning techniques for data imputation include, but are not limited to, clustering-based methods (e.g. KNN and K-means) [7], [8], Support Vector Machine (SVM), Denoising Autoencoders (DAEs) [9], and Generative Adversarial Networks (GANs) [10]–[15]. Specialized imputation methods for traffic data are majorly matrix/tensor repair methods which use statistical techniques

such as matrix/tensor decomposition [16]–[19]. Tensor-based methods have been proposed to overcome the shortcoming of matrix based ones in the face of high missing rates as tensors are capable of preserving multi-way features of traffic data [2]. Despite the low-rank property of traffic data sets, which is the foundation for tensor decomposition methods, the choice of rank and decomposition method is subjective and challenging especially for incomplete data [19].

Further development of imputation techniques using GAN is highly motivated by ITS. First, the accuracy of GAN-based imputation architectures have been significantly competitive compared with both statistical and other machine-learning techniques [10]–[15], so they are appealing to ITS where poor traffic management based on erroneous data may cost millions of dollars [20]. Second, traffic data are usually high dimensional which can be represented by implicit models like GAN avoiding the bias of paramateric distributions [21]. Third, a generative imputer once trained can be applied to a single data, unlike clustering-based methods which must search a sample of data for finding neighbors of the incomplete one. Finally, GAN models like every DNN can be fine-tuned efficiently as ITS data are generated in large volumes and velocity. Despite all the advantages of the GAN-based imputation methods, they suffer from inaccuracy for high missing rates, and/or complicated architecture. Therefore, more exploration is needed for traffic data imputation using GAN in ITS.

The contribution of this work is as follows: A novel GAN-based architecture for data imputation is proposed which outperforms previous GAN-based architectures with respect to:

- accuracy,
- simplicity of architecture, and
- robustness.

The architecture, called Iterative Generative Adversarial Networks for Imputation (IGANI), is simple as it enjoys one generator and discriminator. The generator is iterated over imputed data and the intuition behind such iteration is to train a robust discriminator which can identify the first-hand imputed data as real and the second-hand one as fake. We will demonstrate how our proposed method compare with competing imputation methods in accurately imputing missing traffic (speed) data. As a comparison basis, we also demonstrate that the imputed traffic data from our method can better train a short-term traffic prediction model, compared with other GAN-based imputation methods, and that this superiority holds for different missing rates. Also, the imputation is performed and evaluated for multi-variable traffic data consisting of volume, occupancy, and speed with different missing rates for each of them. It is demonstrated that IGANI's accuracy is substantially more robust compared with other GAN-based architectures when the dimension of input layer is increased.

The paper is organized as follows. First, a brief background for generative models, GANs, and GAN-based imputation architectures is provided in Section II. Then in Section III we propose the architecture of IGANI. Section IV includes the application of the proposed imputation method on traffic data and the discussion of the results.
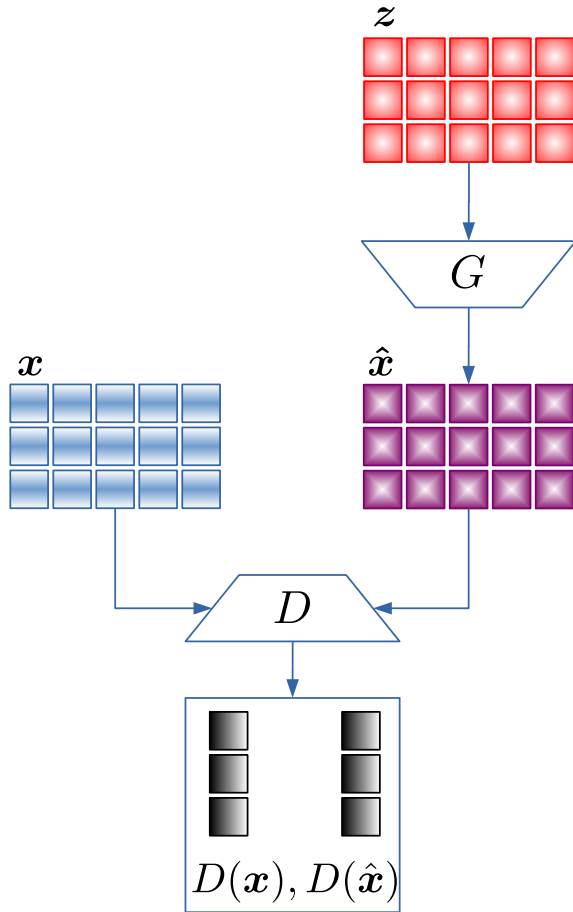
II. BACKGROUND
A. GENERATIVE MODELS

Generative models are models that generate data, either explicitly or implicitly. Explicit generative models considers the data to follow a density function $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ whose parameters $\boldsymbol{\theta}$ are estimated through maximum likelihood method using the log-likelihood function $\log p_{\boldsymbol{\theta}}(\boldsymbol{x})$. Though a computationally intractable log-likelihood may be substituted by other objectives like Jensen-Shannon divergence (JSD) [22]. Explicit generative models are also known as prescribed probabilistic models, because the parametric distribution is dictated in contrast to implicit models which merely generate data [23].

Selecting computationally tractable densities is an important step in explicit generative models, while the choice of a density function that is capable of capturing data complexity is not straightforward. Tractable densities are modeled either by fully visible belief networks (FVBNs) [24] or non-linear independent component estimation (NICE) [25]. FVBNs are based on the chain rule of probability as $p(\boldsymbol{x}) = p(x_1)p(x_2|x_1)\ldots p(x_n|x_1, x_2, \ldots, x_{n-1})$ which gives samples entry-by-entry at a cost of $\mathcal{O}(n)$ for each sample. NICE considers $p(\boldsymbol{x})$ to be a continuous, invertible, nonlinear transformation which maps a latent variable $\boldsymbol{z}$ to $\boldsymbol{x}$, i.e. $\boldsymbol{x} = g(\boldsymbol{z})$. Then $p_x(\boldsymbol{x}) = p_z(g^{-1}(\boldsymbol{x}))|\det(\partial g^{-1}(\boldsymbol{x})/\partial \boldsymbol{x})|$ becomes a tractable density function, if p_z and the determinant is tractable. Data generation by FVBNs is time consuming and cannot be parallelized and NICE requires g to be invertible with $\boldsymbol{x}$ and $\boldsymbol{z}$ having the same dimension [21]. If the explicit density is not tractable, variational methods can be employed. A typical example is variational autoencoders (VAE) which uses a tractable lower bound for an intractable log-likelihood [26]. Another option is Boltzman machines which is based on Markov Chain Monte Carlo [27]. Boltzman machines simulate a sequence of samples $\boldsymbol{x}' \sim q(\boldsymbol{x}'|\boldsymbol{x})$ where q is a transition probability density designed in such a way that the distribution of the samples will converge to $p(\boldsymbol{x})$. While variational methods like VAEs are affected by the accuracy of the posterior or prior distributions, MCMC methods, such as Boltzman machines, suffer from slow convergence [21].

As opposed to explicit methods, implicit generative models do not parameterize the density of data, but are still appealing for high-dimensional data representation, model-based reinforcement learning, data imputation, models with multi-modal outputs, and realistic data generation [21]. Implicit methods train functions which map from a latent variable to the data space; though the mapping is deterministic, the latent variable is the external stochastic source [23]. Implicit models ranges from basic non-uniform random variate generator [28] to Generative Stochastic Networks (GSNs) [29] and Generative Adversarial Networks (GANs) [30]. GSNs using MCMC

FIGURE 1. The GAN architecture, where the generator *G* generates data $\hat{x} = G(z)$ from the latent variable z and the discriminator *D* determines whether $\hat{x}$ or x is fake or not. *G* and *D* are adversarial networks, i.e. *G* is trained toward deceiving *D* and as a result the distribution of the generated data $\hat{x}$ tends to that of x. The number of columns in x, z, and $\hat{x}$ equals the dimension of data and their number of rows is the number of samples (or mini-batches).

are time-consuming especially for high-dimensional data as they use Markov chains, while GANs do not have most of the aforementioned shortcomings. First, data generation in GANs is performed in parallel, independently of the dimension of x. Second, Markov chains, tractable densities, invertible mapping from latent variable z, and variational bounds are not required by GANs [21]. In the next section we present a technical background on GANs.

B. GENERATIVE ADVERSARIAL NETWORK (GAN)

Generative Adversarial Networks (GANs) provide a game-theoretic framework for obtaining the implicit distribution of data. This is performed by adversarial networks where a discriminator network estimates the probability of a data instance being real or fake. The data instances are mapped from a latent variable z by a generative model called generator which is trained to deceive the discriminator. The discriminator is therefore trained in a supervised way, where real and fake data are labeled as one and zero, respectively [30].

Let G and D be the generator and discriminator, each a neural network that is differentiable with respect to its respective input, z and x, and also with respect to their respective neural parameters, $\boldsymbol{\theta}^{(G)}$ and $\boldsymbol{\theta}^{(D)}$. The generator G for each d-dimensional (noise) input z returns a d-dimensional output $\hat{x}$, and the discriminator D for each input data (either $\boldsymbol{x}$ or $\hat{\boldsymbol{x}}$) returns a scalar indicating (deterministically or probabilistically) whether the data is real or fake. The training involves minimizing two cost functions associated with the generator and discriminator, namely $J^{(G)}(\boldsymbol{\theta}^{(G)}, \boldsymbol{\theta}^{(D)})$, and $J^{(D)}(\boldsymbol{\theta}^{(G)}, \boldsymbol{\theta}^{(D)})$. Note that both functions depends on both $\boldsymbol{\theta}^{(G)}$ and $\boldsymbol{\theta}^{(D)}$. In each iteration of the training, a minibatch of inputs, i.e. z and x are selected for D and G and gradient-based optimization minimizes the cost functions of the generator and discriminator subsequently. The architecture of GAN in a training iteration is illustrated in Fig. (1) for the mini-batch size of 3 and a 5-dimensional data.

The discriminator parameters $\boldsymbol{\theta}^{(D)}$ are estimated by minimizing $J^{(D)}(\boldsymbol{\theta}^{(G)}, \boldsymbol{\theta}^{(D)})$, while $\boldsymbol{\theta}^{(G)}$ are estimated by minimizing $J^{(G)}(\boldsymbol{\theta}^{(G)}, \boldsymbol{\theta}^{(D)})$. The whole minimization is considered as a game, because each network can control only its own parameters [21]. The solution of such a game is that of Nash equilibrium [31], i.e. the pair $(\boldsymbol{\theta}^{(G)}, \boldsymbol{\theta}^{(D)})$ which locally minimizes $J^{(D)}(\boldsymbol{\theta}^{(G)}, \boldsymbol{\theta}^{(D)})$ with respect to $\boldsymbol{\theta}^{(D)}$, and minimizes $J^{(G)}(\boldsymbol{\theta}^{(G)}, \boldsymbol{\theta}^{(D)})$ with respect to $\boldsymbol{\theta}^{(G)}$.

Different schemes for cost functions can be used in GANs. These schemes usually differ in $J^{(G)}$, rather than in $J^{(D)}$ [21]. One simple choice for the discriminator's cost function is given by

$$J^{(D)}(\boldsymbol{\theta}^{(G)}, \boldsymbol{\theta}^{(D)}) = -\frac{1}{2}\mathbb{E}_x \log(D(\boldsymbol{x}))$$
$$-\frac{1}{2}\mathbb{E}_z(1 - \log(D(G(z)))). \quad (1)$$

where $\mathbb{E}_x$ denote the expected (or mean) value calculated over the distribution of x. In the zero-sum game, the cost of generator is selected such that it neutralizes the cost of discriminator, i.e.,
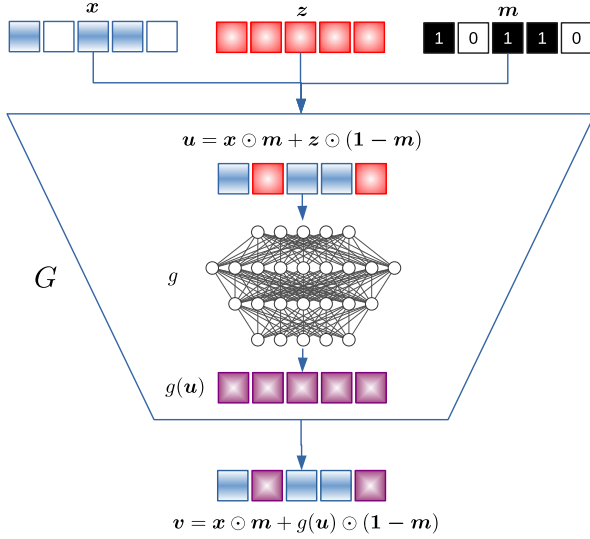
$$J^{(G)} = -J^{(D)}. \quad (2)$$

This is equivalent to a minimax optimization as follows

$$\boldsymbol{\theta}^{(G)*} = \arg\min_{\boldsymbol{\theta}^{(G)}} \max_{\boldsymbol{\theta}^{(D)}} -J^{(D)}(\boldsymbol{\theta}^{(G)}, \boldsymbol{\theta}^{(D)}), \quad (3)$$

which does not perform well in practice because both D and G are minimizing and maximizing the same cross-entropy, respectively. Instead of minimizing the probability of a correct discrimination, to avoid vanishing gradient issue in the optimization, G can be trained by maximizing the probability of a wrong discrimination, or effectively by minimizing the following cost function

$$J^{(G)}(\boldsymbol{\theta}^{(G)}, \boldsymbol{\theta}^{(D)}) = -\frac{1}{2}\mathbb{E}_z \log(D(z)). \quad (4)$$

The role of discriminator in predicting the probability of being real or fake data can be relaxed using Wasserstein distance in the cost functions. Wasserstein or Earth-Mover

FIGURE 2. Generative imputer: blue and white squares denote observed and missing values of incomplete data x. Noise is represented by red squares as shown for the noise vector z. The generated data $g(u)$ is shown in purple. Note that the imputed vector v shares the observed values with x.

FIGURE 3. GAIN architecture: the generative imputer G is trained along with the discriminator D which predicts the mask (m) using the imputed data (v) and the hint (h).

distance facilitate the convergence of a probability distribution sequence for which popular measures for distributional distance like Jensen-Shannon divergence, Kullback-Leibler divergence do not converge. Therefore, training GANs using Wasserstein distance (WGANs) is more stable and the results are less sensitive to hyper-parameters and architectures [32]. WGANs uses a linear activation output layer instead of sigmoid function for the discriminator (critic), employing the Wasserstein distance in the cost functions, and constraining the parameters (namely weights) of discriminator to fall into a compact support by clipping. Clipping underuses the capacity of discriminator and, if not tuned well, may lead to exploding or vanishing gradient. This can be avoided by incorporating a gradient penalty (GP) in WGAN Lipschitz constraint on the discriminator loss [33] as follows

$$J_W^{(D)}(\boldsymbol{\theta}^{(G)}, \boldsymbol{\theta}^{(D)}) = \mathbb{E}_x D(\boldsymbol{x}) - \mathbb{E}_z D(G(\boldsymbol{z}))$$
$$+ \lambda \mathbb{E}_y(\|\nabla_{\boldsymbol{y}} D(\boldsymbol{y})\| - 1)^2 \quad (5)$$
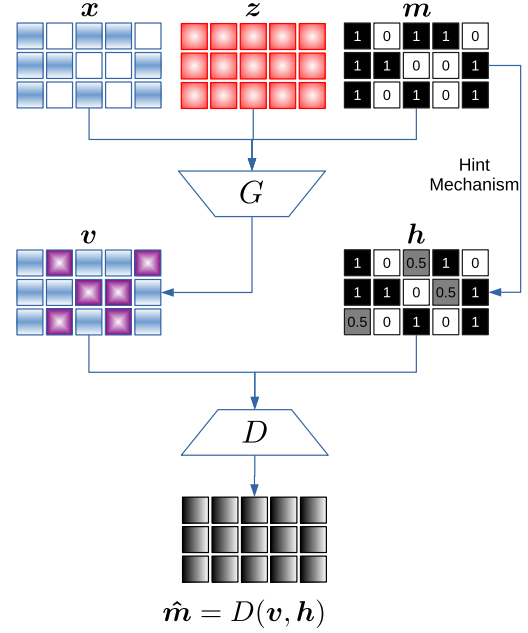
where $\boldsymbol{y} = t G(\boldsymbol{z}) + (1 - t)\boldsymbol{x}$ with $0 \leq t \leq 1$ and λ is a coefficient, usually set as $\lambda = 10$. The cost function of generator like Eq. (4) can be written as

$$J_W^{(G)}(\boldsymbol{\theta}^{(G)}, \boldsymbol{\theta}^{(D)}) = -\mathbb{E}_z D(\boldsymbol{z}). \quad (6)$$

The proposed algorithm in this work uses WGAN-GP with the cost functions defined in Eqs. (5-6).

C. GAN-BASED IMPUTATION METHODS

As mentioned above, GAN is an implicit generative model which can be applied to data imputation. A pioneering method is Generative Adversarial Imputation Network (GAIN) [10] which has a sweet epoch in the training phase urging early-stoping. GAIN has not been successful on block
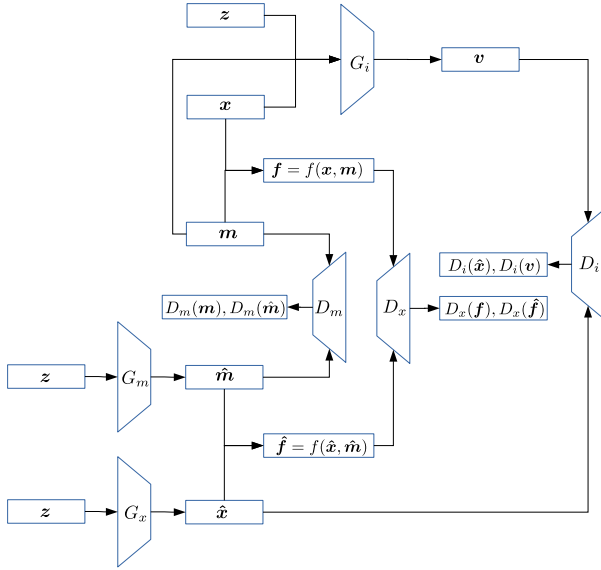
missing patterns. For these patterns, another architecture, called MisGAN, has been shown to be more effective [11]. In MisGAN, three generators for mask, complete data, and imputed data are simultaneously trained together with their corresponding discriminators. In [12], the performance of MisGAN was questioned in uniform missing pattern and a Generative Imputation (GI) was introduced as an accurate method for both block and uniform missing patterns. The problem of imputing multi-view and multi-modal data which are observed from heterogeneous sources is addressed in [13], [14]. GAN is also leveraged by Recurrent Neural Network (RNN) to impute incomplete multivariate time series in [15], and demonstrated accuracy improvement over GAIN and MisGAN. In this work, we used GAIN and MisGAN as the prominent competing architectures against which our proposed IGANI method is compared. Before that, we define what a generative imputer is:

Definition 1: Let $\boldsymbol{x} \in \mathbb{R}^d$ denote a random vector and $\boldsymbol{m} \in \{0, 1\}^d$ be its random mask where $m_j = 0$ or $m_j = 1$ means that x_j is observed or missing, respectively. A generative imputer is defined as $(\boldsymbol{u}, \boldsymbol{v}) = G(\boldsymbol{x}, \boldsymbol{m}, \boldsymbol{z})$ where

$$\boldsymbol{u} = \boldsymbol{x} \odot \boldsymbol{m} + \boldsymbol{z} \odot (1 - \boldsymbol{m})$$
$$\boldsymbol{v} = \boldsymbol{x} \odot \boldsymbol{m} + g(\boldsymbol{u}) \odot (1 - \boldsymbol{m}) \quad (7)$$

where $\boldsymbol{z} \in \mathbb{R}^d$ is noise and $g(\cdot)$ is a function to be learned, and $\boldsymbol{v}$ is the imputed vector where $v_j = x_j$ if x_j is observed. Alternatively, one may write $(\boldsymbol{u}, \boldsymbol{v}) = G(\boldsymbol{x}^{(m)}, \boldsymbol{m}, \boldsymbol{z}^{(1-m)})$ where $\boldsymbol{x}^{(m)} = \boldsymbol{x} \odot \boldsymbol{m}$ and $\boldsymbol{z}^{(1-m)} = \boldsymbol{z} \odot (1 - \boldsymbol{m})$.

In GAIN, shown in Fig. 3, a discriminator D maximizes the probability of predicting mask $\boldsymbol{m}$ based on the imputed data $\boldsymbol{v}$. Prediction of the mask can also be interpreted as

FIGURE 4. MisGAN architecture: three pairs of generators and discriminators are trained simultaneously. G_m, G_x, and G_i generate mask, data, and imputed data, respectively. A mask function $f(x, m) = x \odot m + \tau(1 - m)$ is used to train the data generator G_x and its corresponding discriminator (D_x). Note that G_i denotes the generative imputer and v is the imputed data.
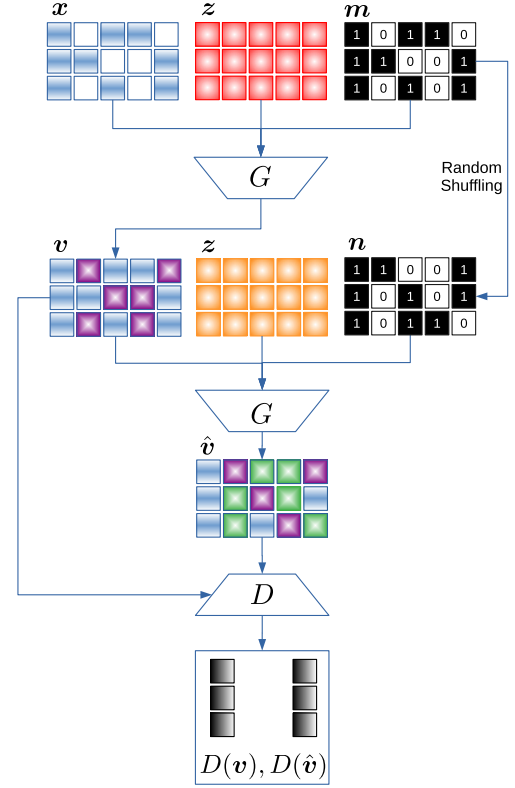

FIGURE 5. IGANI architecture: iteration of the generative imputer G gives a second-hand imputed data $\hat{v}$ which has less observed values compared with the first-hand imputed data v; compare the blue squares between v and $\hat{v}$. So G and D are trained to compensate the decline of observed values after the secondary imputation, by means of imputing more accurately.

how real/fake every element of the imputed vector is. This generalizes the role of conventional discriminators in GANs which give a scalar score for the whole generated vector. Mask prediction is realized by using a partial information about the original mask matrix which is referred to as the hint h. Specifically in GAIN, the hint mechanism chooses one element of each row of the mask matrix randomly and set it to be 0.5. Using the imputed data v and the hint h, the discriminator D predicts the mask as $\hat{m} = D(h, \hat{x})$ whose distance from m is minimized. Using the Sigmoid function as the last layer, we force the entries of $\hat{m}$ to be in [0, 1]. The cost functions for the discriminator and generator in Eqs. (1, 4) are therefore modified as:

$$J^{(D)}(\boldsymbol{\theta}^{(G)}, \boldsymbol{\theta}^{(D)}) = -\mathbb{E}_{\boldsymbol{m}, \hat{\boldsymbol{m}}}\Big[\sum_{i=1}^{d}[\boldsymbol{m}_i \log(\hat{\boldsymbol{m}}_i) + (1 - \boldsymbol{m}_i)\log(1 - \hat{\boldsymbol{m}}_i)]\Big] \quad (8)$$

and

$$J^{(G)}(\boldsymbol{\theta}^{(G)}, \boldsymbol{\theta}^{(D)}) = -\mathbb{E}_{\boldsymbol{m}, \hat{\boldsymbol{m}}}\left[\sum_{i=1}^{d}(1 - \boldsymbol{m}_i)\log(\hat{\boldsymbol{m}}_i)\right] \quad (9)$$

MisGAN, compared to GAIN, has a more complicated architecture and includes three pairs of generators and discriminators trained simultaneously by the WGAN loss functions in Eqs (5) and (6). Generators are G_m, G_x, and G_i which generate mask, data, and imputed data, respectively. Mask generation is the first step, because we access to the real mask matrix whose distribution can be implicitly recovered by training G_m within a GAN scheme. The same procedure can be applied to train G_x, i.e. the generator of observed

data represented by the mask function $f(x, m) = x \odot m + \tau(1 - m)$. Note that the arbitrary constant τ is a proxy for the unobserved data and the choice of its value does not affect the convergence. Given the distribution of mask and data as described, one may train the imputer generator (G_i) whose output (v) is discriminated from the data generated by G_x, i.e. ($\hat{x}$).

III. METHOD

In this section, we propose the architecture of IGANI as a GAN-based data imputation method. As can be seen in Fig. 5, this approach first applies the generative imputer G to produce the 'imputed' data v. Then in an iteration, it again applies G, this time on v and a reshuffled mask matrix, to produce a 're-imputed' data $\hat{v}$. Note that v has more observed values compared to $\hat{v}$. Then, the discriminator D seeks to distinguish between 'imputed' and 're-imputed' data, by giving scalar scores to v and $\hat{v}$ vectors separately, as done in conventional GANs. We train IGANI, or the two neural networks G and D specifically, by minimizing the following WGAN loss functions

$$J_W^{(D)} = \mathbb{E}_v D(v) - \mathbb{E}_z D(\hat{v}) + \lambda \mathbb{E}_y(\|\nabla_{\boldsymbol{y}} D(\boldsymbol{y})\| - 1)^2,$$
$$J_W^{(G)} = -\mathbb{E}_z D(\hat{v}), \quad (10)$$

TABLE 1. Dimension of layers for different architectures.

Architecture	DNN	Layer dimensions
IGANI	G	Dense (214×512), ReLU, Dropout (p = 0.05), Dense (512×512), ReLU, Dropout (p = 0.05), and Dense (512×214)
	D	Dense (214×256), ReLU, Dense (256×256), ReLU, and Dense (256×214)
GAIN	G	Dense (214×512), ReLU, Dropout (p = 0.05), Dense (512×512), ReLU, Dropout (p = 0.05), and Dense (512×214)
	D	Dense ((2×214)×256), ReLU, Dense (256×256), ReLU, and Dense (256×214)
MisGAN	G_m	Dense (214×256), ReLU, Dense (256×256), ReLU, and Dense (256×214)
	D_m	Dense (214×256), ReLU, Dense (256×256), ReLU, and Dense (256×214)
	G_x	Dense (214×256), ReLU, Dense (256×256), ReLU, and Dense (256×214)
	D_x	Dense (214×256), ReLU, Dense (256×256), ReLU, and Dense (256×214)
	G_i	Dense (214×512), ReLU, Dropout (p = 0.05), Dense (512×512), ReLU, Dropout (p = 0.05), and Dense (512×214)
	D_i	Dense (214×256), ReLU, Dense (256×256), ReLU, and Dense (256×214)
Short-term prediction	Net	Dense (214×424), ReLU, Dropout (p = 0.05), Dense (424×424), ReLU, Dropout (p = 0.05), and Dense (424×214)

Note: The dimension of inputs for Guangzhou data is always 214 for all the networks. The dimension of Portland-Vancouver data (discussed in Section IV-B) is 480, which replaces 214 in this table.

FIGURE 6. MAE of normalized imputed values by GAIN, MisGAN, and IGANI where shaded area represents ±3σ for 5 models. The baseline, mean imputation, shown by black star marker, does not outperform IGANI even for higher missing rates.

where $y = tG(z) + (1 - t)v$ with $0 \le t \le 1$ and $\lambda = 10$.

According to the architecture in Fig. 5, the distribution of $\hat{v}$ tends to that of v, if the GAN converges. However, the ultimate goal is to show that the distribution of the imputed data with the new mask ($v^{(n)} = v \odot n$) tends to that of the observed data ($x^{(m)} = x \odot m$). Briefly speaking, we want to show that $p_{\hat{v}} \to p_v$ gives $p_{v^{(n)}} \to p_{x^{(m)}}$. This is guaranteed by the invertiblity of the generative imputer G in Def. (1) which is preceded by showing that m is recovered almost everywhere from (u, v). This is rather intuitive; as Fig. 2 shows, v differs from u only for missing values:

Lemma 1: For a nonlinear function g in Eq. (7), $m = \mathbb{1}_{(u=v)}$ holds almost everywhere.

Proof: If g is a nonlinear function, $g(u)$ cannot be a linear combination of z. Therefore, the measure of $\{z \in \mathbb{R}^d | \exists i \le d : z_i = g(u)_i\}$ is zero, which means that u and v are not equal for unobserved indices almost everywhere and $m = \mathbb{1}_{(u=v)}$. Note that the function sequence $\{g_i\}_{i=1}^{\infty}$ estimating g is trained by deep neural networks which are nonlinear because of their nonlinear activation functions. □
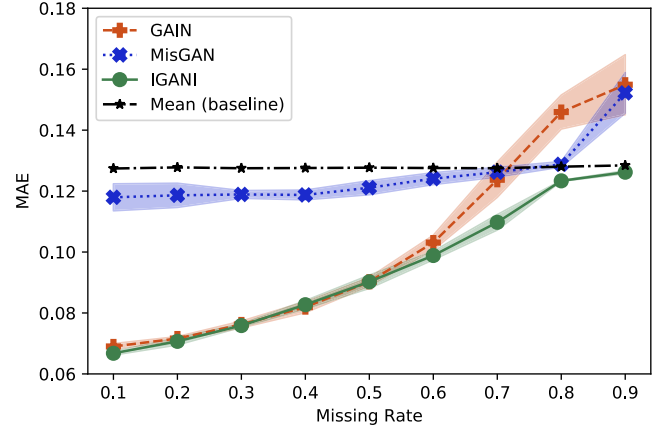
Lemma 2: The generative imputer $(u, v) = G(x^{(m)}, m, z^{(1-m)})$ in Def. 1 is invertible for $m = \mathbb{1}_{(u=v)}$.

Proof: Eq. (7) is written as

$$u = x^{(m)} + z^{(1-m)}$$
$$v = x^{(m)} + g(u) \odot (1 - m) \quad (11)$$

Then $(x^{(m)}, m, z^{(1-m)})$ can be written explicitly in terms of (u, v) as follows

$$x^{(m)} = v - g(u) \odot \mathbb{1}_{(u \neq v)}$$
$$z^{(1-m)} = u - v + g(u) \odot \mathbb{1}_{(u \neq v)} \quad (12)$$

which proves the invertibility of $(u, v) = G(x^{(m)}, m, z^{(1-m)})$ for $m = \mathbb{1}_{(u=v)}$. □

Now consider the architecture in Fig. (5) which gives $p_{\hat{v}} \to p_v$ provided that GAN converges. The aim is to show that $p_{v^{(n)}} \to p_{x^{(m)}}$. Let $(u, v) = G(x^{(m)}, m, z^{(1-m)})$ and $(\hat{u}, \hat{v}) = G(v^{(n)}, n, z^{(1-n)})$ where G is the generative imputer in Def. 1 and $n \sim p_m$:

Theorem 1: $p_{\hat{u}, \hat{v}} \to p_{u,v}$ gives $p_{v^{(n)}} \to p_{x^{(m)}}$.

Proof: If $p_{\hat{u}, \hat{v}} \to p_{u,v}$, i.e.

$$p_{G(v^{(n)}, n, z^{(1-n)})} \to p_{G(x^{(m)}, m, z^{(1-m)})} \quad (13)$$

invertibility of the generative imputer G as shown in Lemma 2 gives

$$p_{v^{(n)}, n, z^{(1-n)}} \to p_{x^{(m)}, m, z^{(1-m)}}. \quad (14)$$

As the limit holds for joint distributions, it must hold for marginals and $p_{v^{(n)}} \to p_{x^{(m)}}$. □

Lemma 3: $p_{\hat{v}} \to p_v$ gives $p_{\hat{u}, \hat{v}} \to p_{u,v}$.

Proof: As shown in Fig. (2), the generative imputer preserves the observed values of data, i.e. $x^{(m)} = v^{(m)}$ and $v^{(n)} = \hat{v}^{(n)}$. So Eq. (11) implies
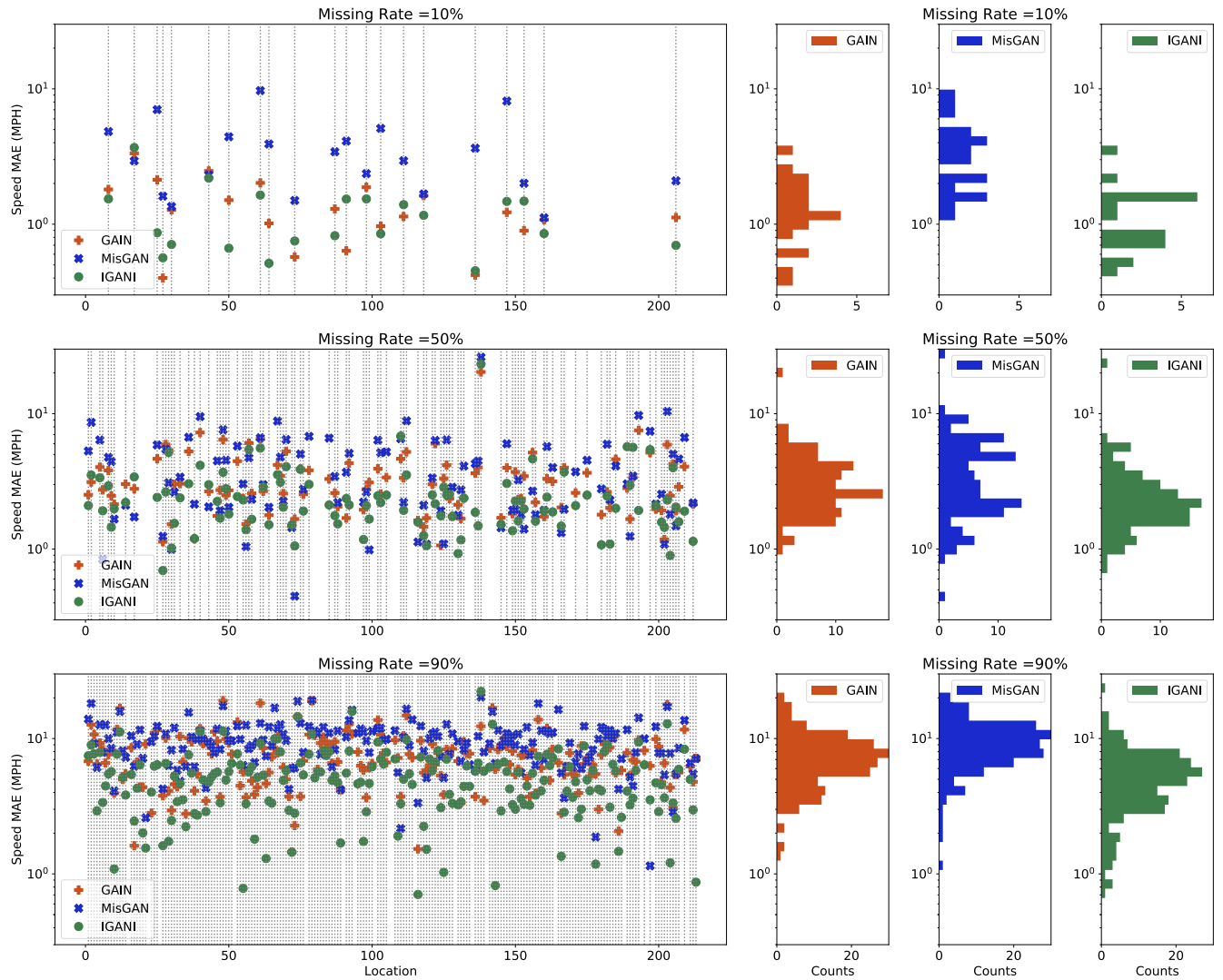
$$p_{u,v} = p_{v^{(m)} + z^{(1-m)}, v}$$
$$p_{\hat{u}, \hat{v}} = p_{\hat{v}^{(n)} + z^{(1-n)}, \hat{v}} \quad (15)$$

Also, let $\gamma(v, m, z) = v - v^{(m)} - g(v^{(m)} + z^{(1-m)}) = 0$ which gives $\gamma(\hat{v}, n, z) = 0$ because the same generative model G (and g) is used in the iterative imputation. Therefore, for the same relation γ, assuming $n \sim p_m$ and $p_{\hat{v}} \to p_v$, it is concluded that $p_{\hat{u}, \hat{v}} \to p_{u,v}$. □

Corollary 1: $p_{\hat{v}} \to p_v$ gives $p_{v^{(n)}} \to p_{x^{(m)}}$.

Proof: Lemma 3 states $p_{\hat{u}, \hat{v}} \to p_{u,v}$ which according to Theorem 1 is enough to have $p_{v^{(n)}} \to p_{x^{(m)}}$. □

The GAN architecture described in Fig. 5 and implemented in Algorithm (1) imputes the incomplete data x, provided that GAN gives $p_{\hat{v}} \to p_v$ which according to Corollary 1 means $p_{v^{(n)}} \to p_{x^{(m)}}$.

FIGURE 7. Logarithmic MAE of imputations by GAIN, MisGAN, and IGANI for an individual sample. IGANI outperforms GAIN and MisGAN especially for higher missing rates. Vertical lines on left subplots represent missing value indices. The histograms of MAE for imputed values are also plotted.

IV. EXPERIMENTS

A. GUANGZHOU (CHINA) SPEED DATA

1) DATA DESCRIPTION

The dataset represents the speed of 214 road segments in Guangzhou (China) collected over two months (from August 1, 2016 to September 30, 2016) at 10-minute time steps. [34]. The data is recommended by the providers for data imputation, traffic prediction and pattern discovery. The spatial and temporal window of the data, as mentioned above, implies a structure of three-dimensional tensor of size $61 \times 144 \times 214$, whose dimensions are day, time, and road segment, respectively. For learning, the dataset is reshaped as a 8784×214 tensor, meaning that the sample size is 8784. The original missing rate of data is 1.29%, which is almost negligible for imputation purposes, and the missing data is calculated using an iterative imputer (that estimates each feature as a function of all the others) before creating a ''complete'' reference data.
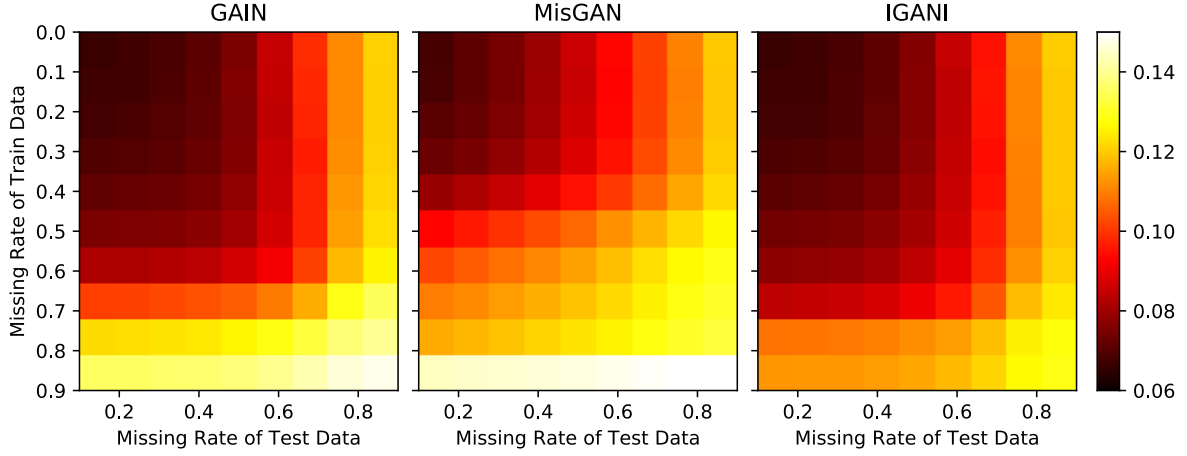
The described dataset is divided into three portions: (i) 10% for training the imputers, (ii) 80% for training the short-term traffic prediction models, and (iii) 10% for testing the prediction models. The imputation accuracy is tested on portions (ii) and (iii).

2) ARCHITECTURE

Network dimensions of the architectures used in the experiments is shown in table 1. To have a fair comparison of imputer performance, the same capacity of it is used for them; i.e. G for IGANI and GAIN and G_i for MisGAN have the same layer and dimensions.

3) IMPUTATION OF MISSING TRAFFIC DATA

The proposed method in this work, IGANI, is applied to the mentioned data with different missing rates. The missing mechanism is MCAR which is the common case

FIGURE 8. MAE of normalized short-term traffic predictions using missing data imputed by GAIN, MisGAN, and IGANI, averaged over 5 trained imputers for each architecture.
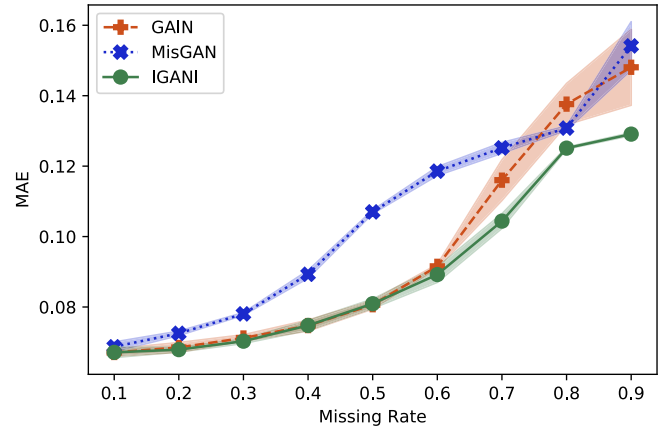
Algorithm 1: Training of IGANI

Set N_E (number of epochs);
Set N_{DU} (number of discriminator updates);
while *epoch* $\leq N_E$ **do**
    for $(x, m) \in (X, M)$ **do**
        Sample noise $z \sim p_z$;
        for *iter* $\leq N_{DU}$ **do**
            $(u, v) \leftarrow G(x, m, z)$;
            $n \leftarrow$ a random shuffling of m;
            $(\hat{u}, \hat{v}) \leftarrow G(v, n, z)$;
            Update discriminator D by minimizing the loss $J_W^{(D)}$ in Eqs. 10;
        Update Generator G by minimizing the loss $J_W^{(G)}$ in Eqs. 10.;

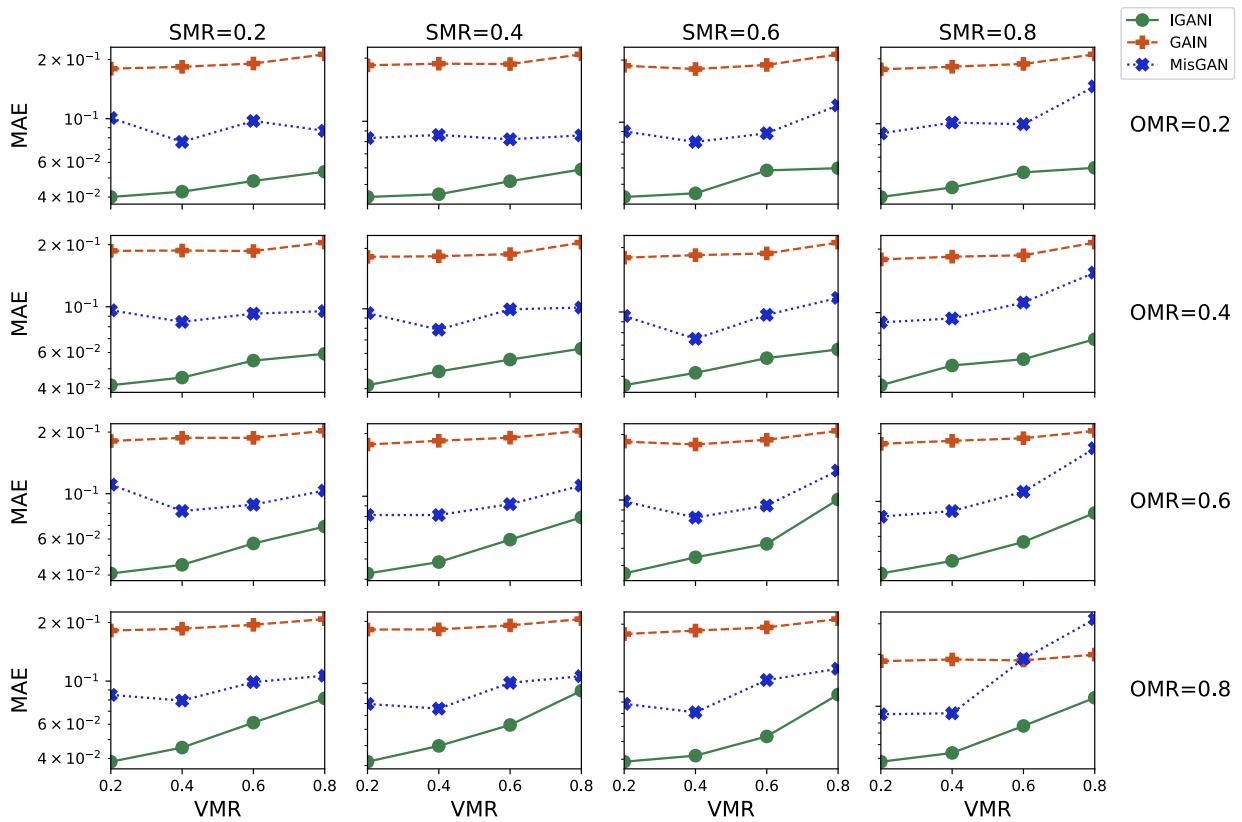Training parameters for this paper: learning rate = 1e-4, $N_E = 200$, $N_{DU} = 30 + [epoch/10]$

FIGURE 9. MAE of normalized short-term speed predictions based on imputed data by GAIN, MisGAN, and IGANI. Shaded area represents $\pm 3\sigma$ for the 5 trained imputers. Training and testing data have the same missing rate.

for incomplete traffic data. The performance of IGANI with respect to test accuracy is compared with two recent GAN-based imputation methods, i.e. GAIN and MisGAN, and the results are shown in Fig. 6. For each of the three approaches, we ran 5 separate trainings, and the mean and standard deviation results in Fig. 6 are calculated using 5 separate tests on the 5 trained models.

As can be seen, the mean absolute error (MAE) of imputation by IGANI, compared with other methods, is lower for all the missing rates. Also, GAIN and MisGAN are less accurate than the baseline (mean imputation) for missing rates higher than 70%, while IGANI always outperforms the baseline. Superiority of IGANI is also visible for an individual sample of data; see Fig. 7. The outperformance of IGANI may be justified as follows. MisGAN architecture, as can be seen from Fig. 4, is complicated with three pairs of generator-discriminator trained together at every epoch. The discriminator of imputer (D_i) treats the output of data generator (G_x) as the real data ($\hat{x}$), which is far from being

real in the early stages of training and mislead the imputer. Such an adverse early-stage effect of NNs on each-other, does not occur in GAIN and IGANI where both have only one pair of generator and discriminator. However, the mask hint as an input to the discriminator of GAIN (see Fig. 3) is spoiler in the sense that the discriminator has an easier job to distinguish the observed data from the missing one. The effect of spoiling the discriminator becomes bold for higher missing rate. The discriminator of IGANI does not benefit from such a hint while training, instead it is trained to recognize which data is imputed twice without any hint of the masks. Therefore, the superiority of IGANI compared with MisGAN and GAIN is attributed to its simple architecture and mature discriminator.

4) SHORT-TERM TRAFFIC PREDICTION USING IMPUTED DATA

In addition to assessing the accuracy of data imputation, it is also important to evaluate how imputation methods, which are unsupervised learning, perform when used in subsequent analyses or predictions. In this study, we choose

FIGURE 10. MAE of normalized imputed *volume* data by IGANI, GAIN, and MisGAN. VMR, OMR, and SMR denote volume, occupancy, and speed missing rates, respectively.

short-term traffic prediction using supervised learning-based neural network using imputed data. In particular, the time interval between two subsequent data samples is considered to be 10 minutes and the aim is to predict an instance (the speed) from its preceding one using a fully connected neural network model. The authors admit that more advanced methods are available for predicting temporal data (like recurrent neural networks (RNN) and its variants, e.g. long short-term memory (LSTM) and gated recurrent unit (GRU)). But, the choice of prediction model is beyond the aim of this work, and our aim is to assess how the errors cause by different imputation methods transcend into *a* subsequent prediction task.

In this work, short-term prediction models are trained and tested for different missing rates. These rates range from 0 to 0.9 for training and from 0.1 to 0.9 for testing models. The imputation methods are GAIN, MisGAN, and IGANI which implies a total of $10 \times 9 \times 3 = 270$ models. The MAE of testing models for predicting normalized data is observed for all cases in Figs. 8 and 9. As can be seen from these results, the superiority of IGANI is more significant for higher missing rates where both GAIN and MisGAN lose robustness in supervised learning-based tasks that use imputed data. Generally, IGANI is more accurate and stable compared with previous GAN-based imputation methods and is strictly recommended for missing rates higher than 50%.
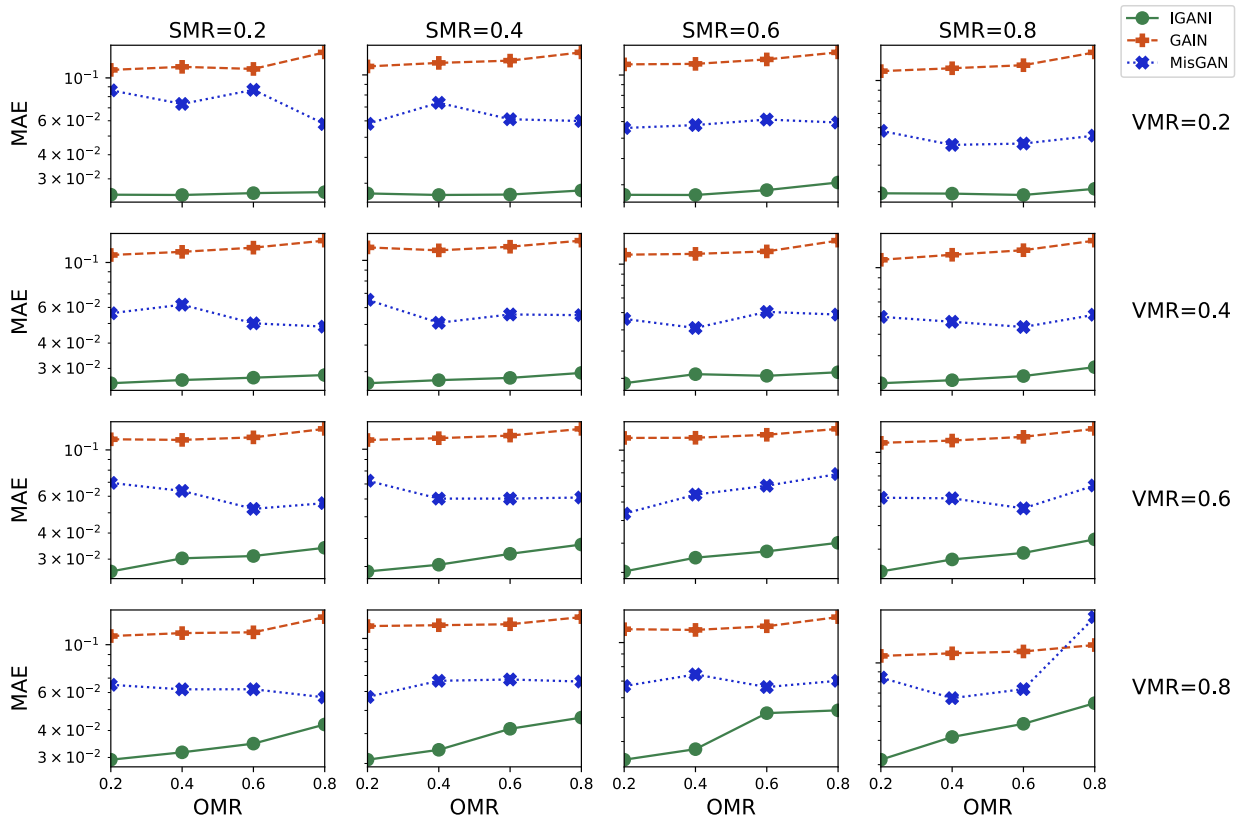
B. PORTLAND-VANCOUVER METROPOLITAN REGION: VOLUME, OCCUPANCY, AND SPEED

In the previous experiment, a single variable of traffic condition, i.e. speed, was imputed. Nowadays, traffic condition is measured by detectors which are capable of capturing more than one variable. Dual loop detectors are a common type of traffic detector which measure volume, occupancy and speed data. While dysfunctions such as communication errors or hardware issues may lead to missing data from dual loop detectors, faults such as chattering, pulse breakup, and hanging on/off may produce low fidelity data [35]. As all variable are not available and valid at the same time and location, imputation of multi-variable traffic data is of critical importance.

In this experiment, a dateset composed of volume, occupancy and speed is used to evaluate our proposed method and other GAN-based imputers. We considered different missing rates for volume, occupancy and speed in order to study the mutual effect of measured variables on the imputation accuracy of the missing one.

1) DATA DESCRIPTION

The data used in this section is collected from the dual loop detectors of highways in Portland-Vancouver Metropolitan region [36]. The time span includes the first half of January 2021 with a time resolution of 15 minutes for detector

FIGURE 11. MAE of normalized imputed *occupancy* data by IGANI, GAIN, and MisGAN. VMR, OMR, and SMR denote volume, occupancy, and speed missing rates, respectively.

IDs between 5000 and 6000. Detectors and time spots which are entirely missing are removed from data, after which the original missing rates for volume, occupancy, and speed were found to be 0, 0, and 0.5%, respectively. Similarly to the previous section, the small number of missing speed data are calculated using iterative imputer to form the "complete" data. This data consists of three matrices of the size 1500×160 for volume, occupancy, and speed measured at 160 locations (detectors) and 1500 time points. Out of the total data length (1500 time points), 85% of the data is used to train the imputer and 15% is used to test it.
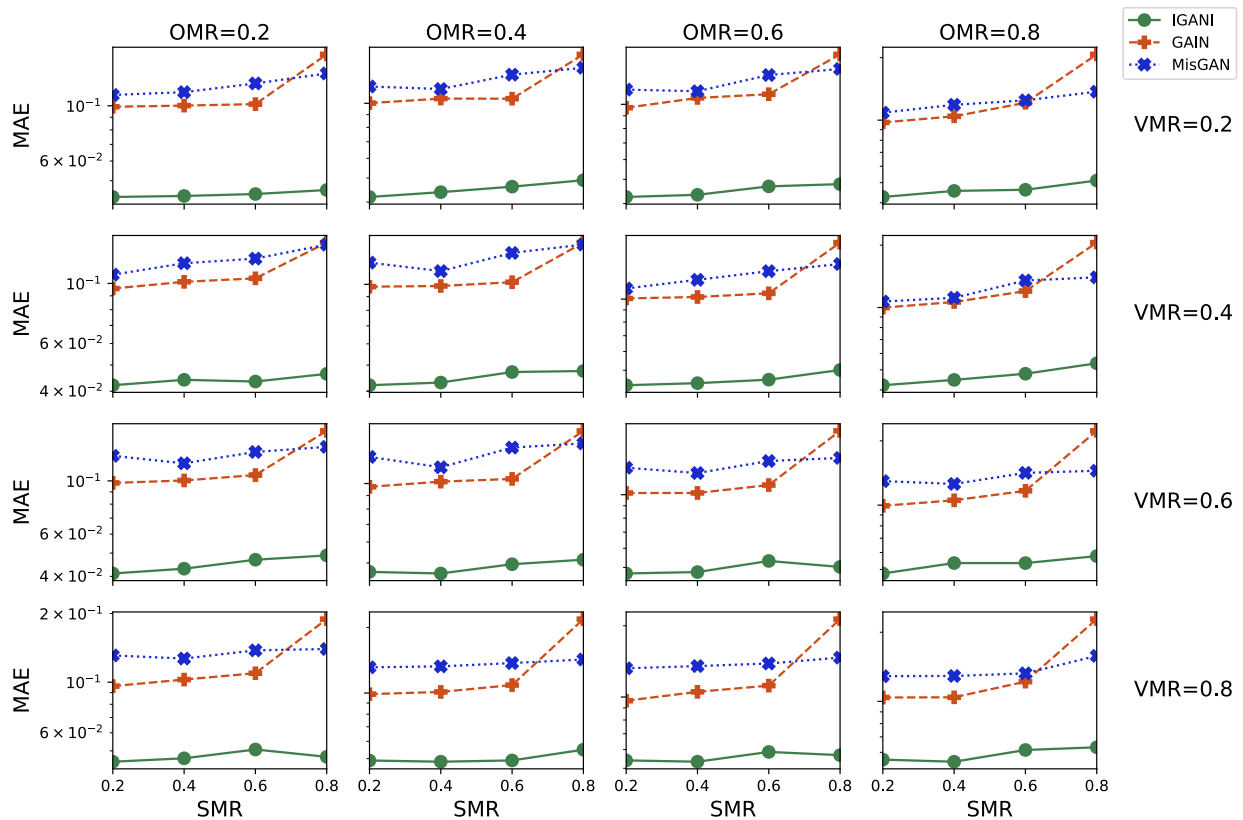
2) ARCHITECTURES

The imputation is performed with the same DNN layers, as listed in Table (1), to evaluate the performance of architectures under fixed model capacity when only the input dimension increases. The only difference is that in this case study, the dimension of the input is 480, instead of 214.

3) IMPUTATION OF MISSING TRAFFIC DATA

In this case study, we consider four missing rates of 0.2, 0.4, 0.6 and 0.8 for each of the three variable (i.e. volume, occupancy, and speed). The imputation results for the 4^3 combinations of these rates are shown in Figs. 10 - 12 for different GAN-based imputation methods. As can be seen, IGANI significantly outperforms GAIN and Mis-

GAN whose performance are adversely affected in this case where the dimension of input has increased from 214 to 480. Also, the expected error increase in higher missing rates can hardly be seen in MisGAN results. The observed inaccuracy for GAIN and MisGAN models, as well as the erratic error behavior for MisGAN results can be due to the fact that the hint mechanism of GAIN and complexity of MisGAN architecture have become inefficient now that we are applying them, with unchanged architecture and model capacity, to a new dataset with a different dimension. This shows that IGANI architecture is more robust and can show similar performance levels for a larger data set, while the dimension of layers in GAIN and MisGAN may need to be re-designed.

The results obtained from the IGANI imputation show that the change in the missing rates of the three variables affects the imputation accuracy of other variables. This is more obvious in Figs. 10 and 11, which show that the imputation accuracy of volume and occupancy depends on the missing rates of one another. This effect is less apparent in 12, however, where the speed imputation accuracy is not significantly influenced by the availability of volume and occupancy data. Such a behavior is probably because speed variation is not significant for this set of highway data, which makes the speed imputation an easier task, regardless of how incomplete the volume and occupancy data are.

FIGURE 12. MAE of normalized imputed *speed* data by IGANI, GAIN, and MisGAN. VMR, OMR, and SMR denote volume, occupancy, and speed missing rates, respectively.

V. CONCLUSION

In this work, a new GAN architecture, named IGANI, is introduced for data imputation and its performance is evaluated on imputation of missing traffic data and also short-term traffic prediction. It is shown that IGANI significantly outperforms the previous GAN-based imputation architectures (like GAIN and MisGAN) accuracy. It is also shown that when IGANI-imputed data is used in a supervised learning framework to train short-term traffic predictions, the prediction accuracy is higher compared to the cases where GAIN- or MisGAN-imputed data is used. The proposed architecture is especially instrumental for the imputation of big data, such as traffic data generated in transportation systems. This is because IGANI imputes traffic data with a higher accuracy compared with other GAN-based methods at various missing rates, and as opposed to clustering-based imputation methods (like KNN) does not require searching within a large pool of data to find closest neighbors for imputation. In case of multi-variable data, like those obtained from traffic loop detectors, IGANI outperforms GAIN and MisGAN, with respect to both accuracy and behavior. This outperformance is demonstrated through a dataset of volume, occupancy, and speed which studies the effect of missing rate of one variable on the imputation accuracy of the other one. Although the architecture is shown to be accurate, robust, and simple, future works may

address how to use IGANI for multiple, noisy data, and time series imputation.

REFERENCES

[1] 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.

[2] P. Wu, L. Xu, and Z. Huang, "Imputation methods used in missing traffic data: A literature review," in *Proc. Int. Symp. Intell. Comput. Appl.* Singapore: Springer, 2019, pp. 662–677.

[3] R. J. Little and D. B. Rubin, *Statistical Analysis With Missing Data*, vol. 793. Hoboken, NJ, USA: Wiley, 2019.

[4] W.-C. Lin and C.-F. Tsai, "Missing value imputation: A review and analysis of the literature (2006–2017)," *Artif. Intell. Rev.*, vol. 53, no. 2, pp. 1487–1509, 2020.

[5] A. N. Baraldi and C. K. Enders, "An introduction to modern missing data analyses," *J. School Psychol.*, vol. 48, no. 1, pp. 5–37, 2010.

[6] J. S. Murray, "Multiple imputation: A review of practical and theoretical findings," *Stat. Sci.*, vol. 33, no. 2, pp. 142–159, May 2018.

[7] S. Zhang, "Nearest neighbor selection for iteratively kNN imputation," *J. Syst. Softw.*, vol. 85, no. 11, pp. 2541–2552, Nov. 2012.

[8] L. Beretta and A. Santaniello, "Nearest neighbor imputation algorithms: A critical evaluation," *BMC Med. Inform. Decis. Making*, vol. 16, no. 3, p. 74, Jul. 2016.

[9] A. F. Costa, M. S. Santos, J. P. Soares, and P. H. Abreu, "Missing data imputation via denoising autoencoders: The untold story," in *Proc. Int. Symp. Intell. Data Anal.* Cham, Switzerland: Springer, 2018, pp. 87–98.

[10] J. Yoon, J. Jordon, and M. van der Schaar, "Gain: Missing data imputation using generative adversarial nets," in *Proc. 35th Int. Conf. Mach. Learn.* (Proceedings of Machine Learning Research), vol. 80, J. Dy and A. Krause, Eds. Stockholm, Sweden: PMLR, Jul. 2018, pp. 5689–5698.

[11] S. C.-X. Li, B. Jiang, and B. Marlin, "MisGAN: Learning from incomplete data with generative adversarial networks," in *Proc. Int. Conf. Learn. Represent.*, 2019.

[12] M. Kachuee, K. Karkkainen, O. Goldstein, S. Darabi, and M. Sarrafzadeh, "Generative imputation and stochastic prediction," 2019, *arXiv:1905.09340*. [Online]. Available: http://arxiv.org/abs/1905.09340

[13] C. Shang, A. Palmer, J. Sun, K.-S. Chen, J. Lu, and J. Bi, "VIGAN: Missing view imputation with generative adversarial networks," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2017, pp. 766–775.

[14] L. Cai, Z. Wang, H. Gao, D. Shen, and S. Ji, "Deep adversarial learning for multi-modality missing data completion," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 1158–1166.

[15] Y. Luo, X. Cai, Y. Zhang, and J. Xu, "Multivariate time series imputation with generative adversarial networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 1596–1607.

[16] H. Tan, G. Feng, J. Feng, W. Wang, Y.-J. Zhang, and F. Li, "A tensor-based method for missing traffic data completion," *Transp. Res. C, Emerg. Technol.*, vol. 28, pp. 15–27, Mar. 2013.

[17] M. T. Asif, N. Mitrovic, J. Dauwels, and P. Jaillet, "Matrix and tensor based methods for missing data estimation in large traffic networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 7, pp. 1816–1825, Jul. 2016.

[18] X. Chen, Z. He, and J. Wang, "Spatial-temporal traffic speed patterns discovery and incomplete data recovery via SVD-combined tensor decomposition," *Transp. Res. C, Emerg. Technol.*, vol. 86, pp. 59–77, Jan. 2018.

[19] X. Chen, Z. He, and L. Sun, "A Bayesian tensor decomposition approach for spatiotemporal traffic data imputation," *Transp. Res. C, Emerg. Technol.*, vol. 98, pp. 73–84, Jan. 2019.

[20] A. Zambrano, M. Zambrano, E. Ortiz, X. Calderón, and M. Botto-Tobar, "An intelligent transportation system: The quito city case study," *Int. J. Adv. Sci., Eng. Inf. Technol.*, vol. 10, no. 2, pp. 507–519, 2020.

[21] I. Goodfellow, "NIPS 2016 tutorial: Generative adversarial networks," 2017, *arXiv:1701.00160*. [Online]. Available: http://arxiv.org/abs/1701.00160

[22] L. Theis, A. van den Oord, and M. Bethge, "A note on the evaluation of generative models," 2015, *arXiv:1511.01844*. [Online]. Available: http://arxiv.org/abs/1511.01844

[23] S. Mohamed and B. Lakshminarayanan, "Learning in implicit generative models," 2016, *arXiv:1610.03483*. [Online]. Available: http://arxiv.org/abs/1610.03483

[24] B. J. Frey, J. F. Brendan, and B. J. Frey, *Graphical Models for Machine Learning and Digital Communication*. Cambridge, MA, USA: MIT Press, 1998.

[25] L. Dinh, D. Krueger, and Y. Bengio, "NICE: Non-linear independent components estimation," 2014, *arXiv:1410.8516*. [Online]. Available: http://arxiv.org/abs/1410.8516

[26] D. Jimenez Rezende and S. Mohamed, "Variational inference with normalizing flows," 2015, *arXiv:1505.05770*. [Online]. Available: http://arxiv.org/abs/1505.05770

[27] G. E. Hinton, T. J. Sejnowski, and D. H. Ackley, "Boltzmann machines: Constraint satisfaction networks that learn," Dept. Comput. Sci. Pittsburgh, Carnegie Mellon Univ., Pittsburgh, PA, USA, 1984.

[28] L. Devroye, "Random variate generation in one line of code," in *Proc. 28th Conf. Winter Simulation (WSC)*, Dec. 1996, pp. 265–272.

[29] Y. Bengio, E. Laufer, G. Alain, and J. Yosinski, "Deep generative stochastic networks trainable by backprop," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 226–234.

[30] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[31] L. J. Ratliff, S. A. Burden, and S. S. Sastry, "Characterization and computation of local Nash equilibria in continuous games," in *Proc. 51st Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Oct. 2013, pp. 917–924.

[32] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," 2017, *arXiv:1701.07875*. [Online]. Available: http://arxiv.org/abs/1701.07875

[33] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of Wasserstein GANs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5767–5777.

[34] X. Chen, Y. Chen, and Z. He, "Urban traffic speed dataset of Guangzhou, China," Guangdong Provincial Key Lab. Intell. Transp. Syst., Res. Center Intell. Transp. Syst., Sun Yat-Sen Univ., Guangzhou, China, Tech. Rep., Mar. 2018. [Online]. Available: https://doi.org/10.5281/zenodo.1205229

[35] A. Ariannezhad and Y.-J. Wu, "Large-scale loop detector troubleshooting using clustering and association rule mining," *J. Transp. Eng., A, Syst.*, vol. 146, no. 7, Jul. 2020, Art. no. 04020064.

[36] *PORTAL, the Official Transportation Data Archive for the Portland-Vancouver Metropolitan Region*. Accessed: Aug. 10, 2021. [Online]. Available: https://portal.its.pdx.edu/home

AMIR KAZEMI received the bachelor's degree in civil engineering from Bu-Ali Sina University and the master's from Iran University of Science and Technology. He is currently pursuing the Ph.D. degree in sustainable and resilient infrastructure systems program with the Department of Civil and Environmental Engineering, University of Illinois at Urbana–Champaign. He served as a Budget Planner of civil projects at Plan and Budget Organization for about five years. His research interests include uncertainty quantification, data-driven and stochastic modeling, and machine learning techniques for infrastructure systems.

HADI MEIDANI received the M.S. degree in electrical engineering and the Ph.D. degree in civil engineering from the University of Southern California (USC), in 2012. He was a Postdoctoral Scholar with the Department of Aerospace and Mechanical Engineering, USC, from 2012 to 2013, and a Postdoctoral Research Associate with the Scientific Computing and Imaging Institute, University of Utah, from 2013 to 2014. Since 2014, he has been an Assistant Professor with the Department of Civil and Environmental Engineering, University of Illinois at Urbana–Champaign. In 2018, he received an NSF CAREER Award to create advanced computational methods for the analysis and design of interdependent infrastructure systems.

● ● ●