Adversarial Recurrent Time Series Imputation

Shuo Yang, Minjing Dong, Yunhe Wang, and Chang Xu, *Member, IEEE*

***Abstract*— For the real-world time series analysis, data missing is a ubiquitously existing problem due to anomalies during data collecting and storage. If not treated properly, this problem will seriously hinder the classification, regression, or related tasks. Existing methods for time series imputation either impose too strong assumptions on the distribution of missing data or cannot fully exploit, even simply ignore, the informative temporal dependencies and feature correlations across different time steps. In this article, inspired by the idea of conditional generative adversarial networks, we propose a generative adversarial learning framework for time series imputation under the condition of observed data (as well as the labels, if possible). In our model, we employ a modified bidirectional RNN structure as the generator G, which is aimed at generating the missing values by taking advantage of the temporal and nontemporal information extracted from the observed time series. The discriminator D is designed to distinguish whether each value in a time series is generated or not so that it can help the generator to make an adjustment toward a more authentic imputation result. For an empirical verification of our model, we conduct imputation and classification experiments on several real-world time series data sets. The experimental results show an eminent improvement compared with state-of-the-art baseline models.**

***Index Terms*— Generative adversarial learning, missing data imputation, time series analysis.**

I. Introduction

TIME series data are one of the most common and important data forms in the physical world [1]. It has been studied in various application scenarios, such as medical diagnosis [2], meteorological prediction [3], and financial analysis [4]. However, the data missing problem is an almost inevitable and ubiquitously existing issue in time series analysis [5], [6]. There are various reasons leading to this problem, such as data collecting anomaly, device failure, or data transferring error. The missing measurements in time series will not only be harmful to critical data monitoring but also compromise the performance in real-world data analysis applications. Thus, there is an indispensable necessity to carefully deal with the missing gaps in the real-world time series.

As is well known, there exist some latent patterns and dependencies between collected data in each time step within a time series. Therefore, simply omitting the missing values [7], especially in a long consecutive term, may cause information loss so that it will be unstable for time series processing [8]. Another crude approach is case deletion that directly discards the samples with unobserved measurements. However, because of the ubiquity in the real-world time series data, this method will squander precious and important data resources (in general, the maximum acceptable deletion ratio is 5% [9]). Thus, it is of great significance to properly impute the incomplete time series before analysis.

However, time series imputation is certainly not an easy task, especially when the missing rate of the data set is relatively high. A number of researchers have paid unremitting efforts to address the missing value problem. Traditional statistical methods try to estimate the missing values based on some general statistical attributes of time series, globally or locally. A primitive idea is substituting the global mean value for the missing ones. Alternatively, one can also impute the missing values in a time series by the observed values from another similar time series, which is also known as hot-deck imputation [10]. However, these methods underestimate the temporal relationships between different time steps. There are also some traditional machine learning algorithms borrowing the statistical properties of time series for imputation in a sophisticated way, such as K-nearest neighbor (KNN) [11], expectation–maximization (EM)-based algorithms [9], kernel methods [12], and matrix factorization (MF) [13]. These methods also barely consider the temporal dependencies, which is vital for time series analysis [14].

In recent years, the development of deep learning algorithms has brought more inspirations for time series imputation. Recurrent neural network (RNN) [15] is a class of algorithms that is typically good at handling sequential data. The special recursive connection enables RNN to capture the temporal dynamic behavior of time series. With the help of the hidden states, RNN can memorize the temporal dependencies with the time moving on. Time series imputation based on RNN models has long been investigated since [16]. Latter works try to customize the common RNNs for incomplete time series by combining the missing pattern [2] and using a bidirectional structure [17]. Some studies find that for the multivariate time series, the correlations between different feature dimensions are informative for the reconstruction of time series [18]. These customizations make the RNN more suitable for the time series imputation task.

The generative adversarial networks (GANs) [19] propose an innovative framework for data generation. They create new data by learning a map from random noise to the real data distribution. GAN is such a flexible framework that can almost be adapted to train any differentiable deep model to generate data, including RNN. In contrast to generic generative training

Manuscript received 10 June 2019; revised 23 December 2019 and 8 April 2020; accepted 9 July 2020. Date of publication 4 August 2020; date of current version 5 April 2023. This work was supported in part by the Australian Research Council under Project DE180101438 and in part by USyd–SJTU Partnership Collaboration Award. *(Corresponding author: Chang Xu.)*

Shuo Yang, Minjing Dong, and Chang Xu are with the Faculty of Engineering, School of Computer Science, The University of Sydney, Sydney, NSW 2008, Australia (e-mail: syan9630@uni.sydney.edu.au; mdon0736@uni.sydney.edu.au; c.xu@sydney.edu.au).

Yunhe Wang is with the Noahs Ark Laboratory, Huawei Technologies Company Ltd., Beijing 100085, China (e-mail: yunhe.wang@huawei.com).

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2020.3010524

schemes of RNN (e.g., VAE implemented by an RNN), which minimizes the KL divergence between real and generated data distributions, GAN optimizes the JS divergence between them. Although the reason remains controversial, GAN is deemed to generate new data from fewer modes, suggesting that it can generally generate clearer and sharper samples [20]. Nonetheless, the direct implementation of GAN to impute time series is still not sufficient. For example, Luo *et al.* [14] simply view the time series imputation as a general generation task that generates an entire time series totally from random noise. However, compared with the general data generation, there exist strong dependencies and connections between the known and unknown parts in the imputation task. When imputing the unknown data, the generated data are supposed to be constrained by the observed part. Thus, it is more reasonable to employ the conditional GAN [21] to impute the incomplete time series conditioned on the observed values.

In this article, we propose a GAN-based end-to-end recurrent time series imputation framework. Inspired by the conditional GAN [21], we take the observed measurements as additional input. Compared with the former similar GAN-based works, our model learns a mapping to the completed time series directly from the underlying dependencies and correlations of the observed measurements rather than a random noise, which reasonably reduces the searching space of an appropriate estimation. This requires our model to thoroughly explore the underlying knowledge in time series. Thus, in contrast to the general recurrent networks, our model makes several customizations for dealing with the time series with missing values. In practice, the generator G in our model estimates the lost values from two different perspectives, which we call temporal decay estimation and feature correlation estimation, respectively. These two estimations are systematically combined by a tradeoff term to simultaneously capture the interdependencies across time steps and inner correlations between features within a single time step. Different from the common generation type of GAN that generates a whole sample each time our model completes the missing data step by step. Thus, in order to fit the way of generation, the discriminator D also changes from distinguishing the authenticity of an entire sample to each value in a whole time series. Inspired by [22], a hint mechanism is applied to our model in order to assist D to learn the real missing pattern. G and D are alternately trained, and the discriminative loss from D is used to adjust the training of G. As a result, the data generation procedure can be more explicit and accurate. Experimental results on several real-world time series data sets empirically demonstrate the advantage of our model. To sum up, our work makes the following contributions.

1) We propose a novel end-to-end adversarial recurrent time series imputation method that generates the missing values in the time series in a self-feeding manner and distinguishes the authenticity of each value to adversarially make amendments to the imputed values.
2) Our proposed model is based on the conditional GAN framework by taking the observed measurements as additional input. Thus, we can reconstruct the time series by using the temporal dependencies and feature correlations available in the observed part that makes the imputed time series more reasonable.
3) We achieved state-of-the-art results on both imputation and classification experiments, which demonstrates the superior imputation ability of our model.

For clear comprehension, it is worth introducing the organization of the remains of this article. Section II reviews previous related literature. Section III formulates the time series imputation problem. In Section IV, we demonstrate the detailed architecture of our model. Experiments are conducted in Section V. We draw a conclusion in Section VI.

II. Related Works

This section reviews existing works of time series imputation. We take apart the literature into two categories: one of which is statistical and RNN-based methods and another is GAN-based methods. Because of the myriad of related literature, we only review the ones that are most relevant to our work.

A. Statistical and RNN-Based Methods

Some simple examples of the statistical methods attempt to fill the gaps of missing values in time series by directly utilizing some naive statistical characteristics of the time series. For example, Press *et al.* [23] used the mean value of data to replace the unobserved elements in a sample. Thus, the variation of data is entirely neglected by the mean smoothing. By assuming that the data changes very little in a short period of time, Little and Rubin [24] introduced a method to fill the vacancies in data by the values last observed. KNNs imputation can be regarded as an updated hot deck method. This method first finds K nearest candidates according to a similarity metric (e.g., dynamic time warping (DTW) in [25]). Then, the lost part can be estimated by averaging the known values of the k neighbors in the corresponding position. Multiple imputations by chained equations (MICE) were investigated in [26]. It first fills the incomplete data with mean value and then employs regression models (e.g., linear, logistic, or Poisson regression depending on what assumption is made on data) to impute the missing values based on the chained equations. However, these methods hardly consider the temporal correlations, so they can hardly generalize well on time series imputation tasks.

Some advanced methods use a more sophisticated scheme. For example, the temporal regularized MF (TRMF) [13] method incorporates temporal dependencies in time series to regularize the MF model. Another example is spatiotemporal multiview-based learning (ST-MVL) [5], a multiview learning-based synthetic method. It consists of empirical statistic models (e.g., inverse distance weighting (IDW) [27]) and data-driven algorithms (e.g., collaborative filtering (CF) [28]), and with a special design for meteorological data, the ST-MVL method takes both temporal correlation and geographical information into consideration and achieves the state-of-the-art result of meteorological time series imputation. However, these methods usually impose assumptions on the time series and are only suitable for specific problems.

RNN and its variants are typically good at handling time series. Thus, the vast majority of neural-network-related methods for time series analysis are based on RNNs. Bengio and Gingras [16] first introduced RNN to the time series imputation problem. Their proposed method first initializes the empty positions with mean imputation and then feeds the completed data to an RNN. The imputation values will be updated by the feedback links. Furthermore, multidirectional RNN (M-RNN) [18] employed not only the bidirectional RNN structure to acquire the temporal dependencies "horizontally" but also a fully connected layer for exploiting correlations of different variables "vertically" within a time step. Some latter studies try to capture more information from incomplete time series in terms of its missing pattern. As suggested in [29], the missing pattern can provide valuable information for time series analysis. To this end, Che *et al.* [2] introduced a decay mechanism to the common gated recurrent unit (GRU) in order to explore the connection between the missing pattern and data labels. However, GRU with decay (GRU-D) mechanism pays more attention to the classification task, while the completion is on the basis of mean and last seen imputation. Thus, GRU-D is proven to perform worse in imputation but better in classification tasks [6]. Intuitively, in an incomplete time series, information from time steps with different time elapse will make a different contribution to the current missing value estimation, which means that the missing pattern can also be informative for imputation. The BRITS model proposed in [6] not only takes advantage of the idea of M-RNN structure but also systematically combines the decay mechanism to the recurrent dynamical system for missing data estimation. Experimental results show that there is a mutual promotion of both imputation and classification results.

B. GAN-Based Methods

In recent years, GAN has received constant focus. In a number of research areas, such as image generation [30], [31], text generation [32], domain adaptation [33], point process [34], and robust methods [35], impressive improvement has been made by taking advantage of GANs. GAN introduces a novel generative framework that tries to learn the real distribution of data via the contest between two neural networks, namely, generator (G) and discriminator (D). Although a number of works have been applied to image generation and other nontemporal tasks [22], [36], [37], very few studies employ GAN for time series imputation. An example is [14] in which a GRU combined with a decay vector (which is called GRUI) was employed to generate missing values from a noise vector. However, the imputation of [14] is a rather implicit process because the generation of a time series is totally from random noise and there is no sufficient control on the pattern of time series generation. Meanwhile, Luo *et al.* [14] viewed the imputation and classification as a two-phase task that tends to result in suboptimal results. Thus, even if [14] achieved the state-of-the-art classification result on the PhysioNet data set, the estimation accuracy of missing values may be still very poor. In order to address the abovementioned issues, we borrow the idea of conditional GAN [21] that takes additional information (e.g., labels) as input to control the mode of generated data. In our model, we generate the missing values under the condition of information from the observed data and labels so that the imputation of unknown values will be reasonable and clear.

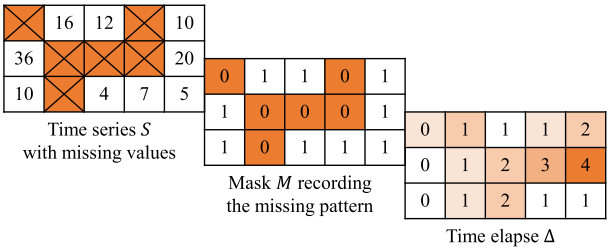

Fig. 1. Illustration of the mask matrix and time interval matrix.

III. Problem Formulation and Preliminary

Consider a time series $S = (s_1, s_2, \ldots, s_T) \in \mathbb{R}^{T\times D}$ consisting of T time steps. For each $t \in \{1, 2, \ldots, T\}$, $s_t \in \mathbb{R}^D$ with D dimensions of features observed sequentially. s_t^d represents the data of the dth feature in the tth time step, and any of s_t^d can be a missing value in the time series S. In the real-world background, the time intervals between collected measurements are not uniform, so we denote $\mathrm{T} = (\tau_1, \tau_2, \ldots, \tau_T) \in \mathbb{R}^T$ to be the time gap between two adjacent observations. We set τ_1 to be 0 because there is no predecessor for the first observation. Next, we denote $M = (m_1, m_2, \ldots, m_T) \in \mathbb{R}^{T\times D}$ as the mask matrix that records the missing pattern of S. Especially, if s_t^d is missing, $m_t^d = 0$; otherwise $m_t^d = 1$, correspondingly.

In order to impute the missing values in a time series, we have to explore the relationship and dependencies between the observed data and missing data. However, the contribution that the observed values make to the estimation of missing values may depend on the time interval between them. Thus, we introduce $\Delta = (\delta_1, \delta_2, \ldots, \delta_T) \in \mathbb{R}^{T\times D}$ to record the information of time interval. Literally, Δ represents the time gaps between the last observation and the present unobserved value. We can formulate each element of Δ by the following equation:

$$\delta_t^d = \begin{cases} \tau_t + \left(1 - m_{t-1}^d\right)\delta_{t-1}^d, & t > 1 \\ 0, & t = 1 \end{cases} \tag{1}$$

where δ_t^d denotes the time that has elapsed since the last value at the dth feature in the tth time step was observed. Fig. 1 demonstrates an example of the relations of notations in the imputation problem. Without loss of generality, in this example, we assume that the time series is sampled every other time unit.

In some of the time series data sets, there may be some labels of categories for instances. We denote the label as y. Based on y, we can also conduct a classification task for the purpose of assisting to impute the missing data and verify the result of imputation.

In this article, our main objective is to generate a completed time series $\hat{S}$ to make an accurate estimation of the missing data in the incomplete time series S. Because we have no access to the missing data, it is prone to generate the missing

values under an incorrect distribution. In order to avoid this situation, we try to learn the distribution of real data so that we can force the imputed data to be more authentic. Thus, we actually simulate the distribution of missing data under the conditional information of known data and labels, i.e., $P(\hat{S}|S, y)$, so that we can generate the missing data in a more explicit and accurate manner.

IV. Method

The proposed GAN-based method for time series imputation in [14] employs a modified RNN structure called GRUI as generator to create time series by stacking a fully connected layer on every hidden state. Especially, GRUI adopts the GRU as the basic RNN unit, and a decay vector α is introduced into the update of GRU as follows:

$$\alpha_t = 1/e^{\max(0, W_\alpha \delta_t + b_\alpha)}, \quad h_{t-1} = h_{t-1} \odot \alpha_t \tag{2}$$

$$\bar{x}_t = W_{\bar{x}} h_{t-1} + b_{\bar{x}}, \quad h_t = \text{GRU}(\bar{x}_t, h_{t-1}) \tag{3}$$

where δ_t is the time gaps at time step t, and $e^{(\cdot)}$, $\max(\cdot)$, and $\odot$ are elementwise exponential, maximizing, and multiplication operation, respectively. For simplicity, we state that, without special specification, all the variables, hereafter noted as W, are weights and b are bias regardless of subscripts.

From (2) and (3), we can see that GRUI is a self-feed recurrent structure, which means that the current biased estimation $\bar{x}_t$ is directly fed to the next GRU unit as input. Note that the generator is initialized with a random noise z and generates an entire time series with GRUI, e.g., $G(z) = z \rightarrow \bar{X}$, so there is no interaction between the update of GRUI and the original time series during the generating process. As a result, the temporal dependencies and feature correlations of the original observed data could not be extracted and leveraged effectively, which makes the imputation result very uncertain and vague. Even though a reconstruction loss is introduced to control the similarity of the generated and observed data, it could still be difficult to yield time series, which obeys the real data distribution learned by discriminator because of the uncertainty of generation. After the generator is well-trained, GAN + GRUI comes to the next training stage in which it tries to find the best-matched noise vector for the generation.

In order to address the issues in [14], we modify both of the RNN structure and the GAN architecture in GAN + GRUI. For a clear comparison, the major differences between GAN + GRUI and our model are summarized as follows.

1) Instead of directly generating a whole sample from a random noise z, our model borrows the idea of conditional GAN [21], which constrains the generated data conditioned on the observed values. Thus, our model can make use of the temporal dependencies and feature correlation in the observed data by the well-designed components and will have a more reasonable imputation result.
2) The GRUI-GAN model generates a whole sample $\hat{X}$ at a time; thus, the target for the discriminator is to identify the whole generated sample $\hat{X}$ from the original X as the traditional GANs do, while, in our model, we gradually reconstruct the time series by imputing the missing values within one time step and then the next. Thus, there exist both observed values x_t and imputed values $\hat{x}_t$ within one time series. In order to be compatible with the generation pattern, we use a novel discriminator that distinguishes the authenticity of each value in the time series.
3) GRUI-GAN uses a two-stage strategy to train the model that may be more difficult to find a simultaneous optimal solution, while we train our model in an end-to-end manner that can lead to more stable performance.

In the remainder of this section, we introduce the details of our model.

A. Exploration of Temporal Dependencies and Feature Correlations With RNN

The previous analysis emphasizes the importance of the temporal dependencies and feature correlations for time series imputation. Accordingly, we modify the RNN structure for missing values estimation in time series from two perspectives. On the one hand, we make an estimation of missing values based on the temporal dependencies learned from previous observations decaying in terms of the missing pattern, which we call temporal decay estimation. On the other hand, we explore the correlations of different feature dimensions for imputation by another term called feature correlation estimation. Finally, these two terms will be systematically combined by a tradeoff term for the final imputation result.

1) Temporal Decay Estimation: Intuitively, as the time gap between previously observed values and missing values becomes increasingly larger, the influence of the observed values on the missing data reconstruction will gradually fade away. For example, a medical feature is significant to diagnosis only within a certain time span [38]. Thus, we also use a negative exponential term named decay term [see (2)] to capture this fading effect as in previous works [2], [6], [14]. Note that in order to exploit the temporal dependencies of the observed part in the time series, the estimated values in the positions where the values were originally observed are replaced by real values before fed into the next RNN unit as follows:

$$\bar{x}_t = m_t x_t + (1 - m_t)\bar{x}_t \tag{4}$$

where x_t is the decomposition of the original time series S by time, i.e., $x_t = s_t$, and $\bar{x}$ is the estimation of missing values with respect to the decaying temporal dependencies across previous time steps. Therefore, with the substitution of observed measurements in (4), the temporal dependencies of the original time series could be extracted and memorized in h_t to provide sufficient guidance for the recursive filling of the missing values in subsequent time steps.

2) Feature Correlation Estimation: Another deficiency of the generation process in GRUI is the omission of feature correlations. Many existing studies mentioned that there may be a certain correlation between different features in the same time step [6], [18], particularly in the medical and meteorological related data because, in these kinds of time series data, similar properties are generally concatenated as a feature vector in one time step. For example, in the clinical diagnosis records, the physiological measurements recorded

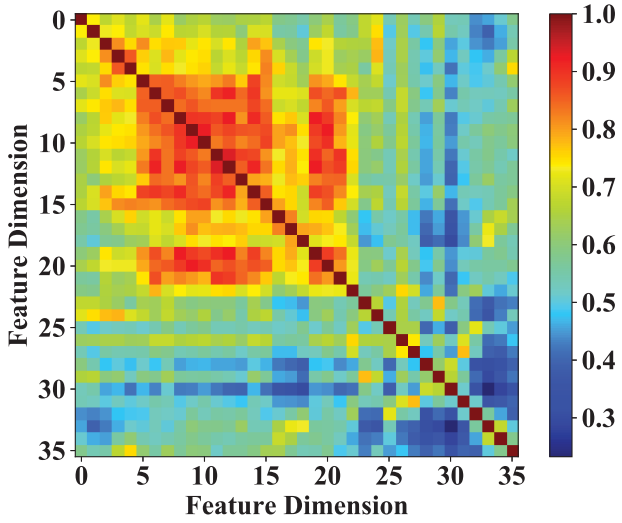

Fig. 2. Average Pearson correlation coefficients between different features. Each pixel in this figure corresponds to the Pearson correlation coefficient of two feature dimensions corresponding to the coordinates. The redder color means a stronger correlation. In order to derive this figure, we first eliminate feature vectors with missing values within a sampling time span (i.e., 36 h in this data set). Then, we compute the average of pairwise Pearson correlation coefficient across all the samples in the data set.

at the same time, such as heart rate, arterial blood pressure, and cholesterol, may have mutual influences and connections with each other because, as a systematic organism, human's vital signs will simultaneously change to make necessary physiological regulation. Another example is shown in Fig. 2. This figure depicts a visualization of average Pearson correlation coefficients between any two features vectors in the PM25 concentration data set. Different feature dimensions correspond to measurements from different meteorological stations. A clear positively correlated pattern is presented in this figure, which indicates the usefulness of the feature correlation for estimating the missing values. Thus, if the pattern of the interactions between different features can be learned, we may complete the missing measurements more precisely.

Inspired by this motivation, we design another estimation of the missing values based on the observed features within one time step as follows:

$$\tilde{x}_t = W_{\tilde{x}}\bar{x}_t + b_{\tilde{x}}, \quad \tilde{x}_t = m_t x_t + (1 - m_t)\tilde{x}_t \tag{5}$$

where $\tilde{x}_t$ is the imputed feature vector in one time step t. Note that the diagonal elements in weight matrix $W_{\tilde{x}}$ are all fixed to be zeros because, assuming that x_t^d is the missing feature, $\tilde{x}_t^d$ is estimated only according to the information from other feature dimensions other than the dth dimension. Similar to (4), we also substitute the observed values for the estimated ones via (5) to obtain the feature correlation estimation of completed data.

3) Combination of Two Forms of Estimation: We now have $\bar{x}_t$ and $\tilde{x}_t$ that are estimations from temporal dependencies and feature correlations, respectively. The upcoming problem is how to determine the proportions of these two parts to the final imputation. Similar to [6], we learn the tradeoff coefficient term β_t from the masking information m_t and temporal decay α_t as follows:

$$\beta_t = \sigma(W_\beta(m_t \circ \alpha_t) + b_\beta) \tag{6}$$

where $\sigma(\cdot)$ is the activation function with the purpose of scaling β within the range of $(0, 1)$, and $\circ$ denotes the concatenation operation. Then, the final imputation $\hat{x}_t$ is designed to be a combination of $\bar{x}_t$ and $\tilde{x}_t$

$$\hat{x}_t = \beta_t \odot \bar{x}_t + (1 - \beta_t) \odot \tilde{x}. \tag{7}$$

4) Update of RNN Unit: Recall (2), and the last hidden state h_{t-1} has been restricted by the decay term α_t. Taking the final imputation result in (7) as input, the update of the modified RNN can be formulated as follows:

$$h_t = \sigma(W_{hh}h_{t-1} + W_{hx}(\hat{x}_t \circ m_t) + b_h). \tag{8}$$

From the abovementioned description, we can find that the biggest difference between our modified RNN structure and GRUI is that the update of the RNN unit in our model is conditioned by the temporal dependencies and feature correlations extracted from the original time series. Consequently, the estimation of the missing values in the next time step would be more explicit and accurate. Furthermore, if the training data are accompanied by category labels, we can make a prediction of the data label and based on the last output of RNN

$$\hat{y} = \text{softmax}(W_y h_T + b_y). \tag{9}$$

Note that in order to demonstrate the power of our model, we simply choose a fully connected layer as the classifier for the classification task, and the empirical experiment shows a competitive result. The classification result is not just a verification of our model; it also acts as part of conditional information to assist the updating of our model toward a more accurate result.

B. Adversarial Imputation Architecture

Our model is trained under a generative adversarial learning framework. The traditional GAN training framework is generally considered as a minimax game between the contest of two networks, namely, generator G and discriminator D. The objective of the traditional GAN can be formalized as

$$\min_G \max_D V(D, G) = \mathbb{E}_{X\sim p_{\text{data}}(X)}[\log D(X)] + \mathbb{E}_{Z\sim p_Z(Z)}[\log(1 - D(G(Z)))] \tag{10}$$

where X is the real data sampled from $p_{\text{data}}(X)$, and Z is a random noise sampled from $p_Z(Z)$. Especially, G learns a mapping from random noise distribution to generated data distribution. D is an estimation of the JS divergence between the real data distribution and the generated data distribution (see [19, Th. 1]), which means that D can guide the G to generate more realistic samples. When D achieves an equilibrium state, the distribution of generated data can match that of real data. Different from the typical GAN-based time series imputation strategy, such as the model in [14], our model first reconstructs the time series based on the prior knowledge of observed data rather than a mapping from random noise; then, the assessment of the authenticity of imputed data will be fed

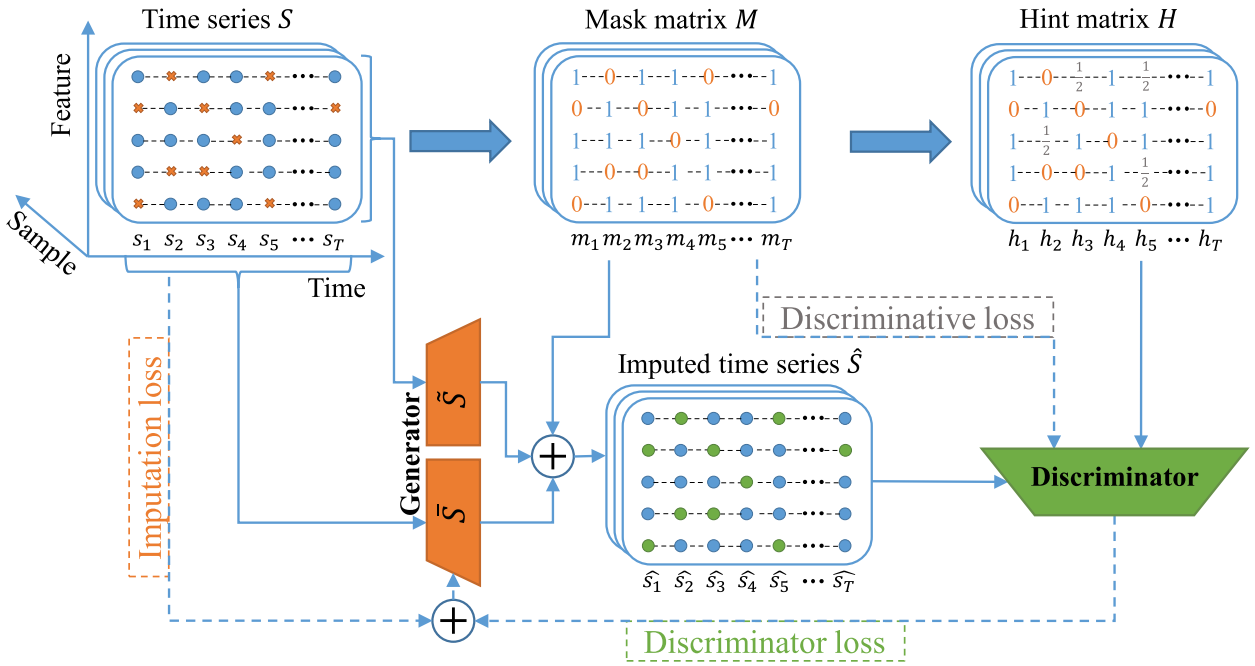

Fig. 3. Adversarial imputation training architecture.

back to adjust the estimation of missing values. Fig. 3 shows a visualized overview of our model. Especially, the discriminator D tries to identify whether each value in the input time series is original or imputed, while the job of generator G is to generate the missing value based on the observed data and to confuse D so that D cannot pick out the imputed value from a completed time series. In the remainder of this section, we show the details of G and D and the adversarial training procedure.

1) Design of Generator: The generator G employs the modified RNN structure detailed in Section IV-A for generating missing values. Especially, in order to exploit the dependencies between both the past and future measurements with the current missing value, a bidirectional RNN structure is adopted in the G block, and for stable training, we choose the long short-term memory (LSTM) [15] as the basic RNN unit. Our model can certainly adopt the GRU as the basic unit; however, one of the comparative model named BRITS [6] that utilizes a similar structure as our generator uses LSTM in their model in practice. Thus, for the sake of fair comparison, we also employ LSTM as a basic recurrent unit. G takes the original missing time series S and its corresponding mask matrix M as input. With the updating of the LSTM units via (3)–(8), the unobserved values will be filled step by step.

2) Design of Discriminator: By contrast, the discriminator D adopts a more generic bidirectional LSTM structure compared with G. Given the time series $\hat{X}$ completed by G, the job of D is to distinguish which elements in $\hat{X}$ are imputed by G. As shown in the following equation, we simply put the previous hidden state h_{t-1} in D into a fully connected layer whose output $\hat{m}_t$ is the estimation of current mask vector m_t

$$\hat{m}_t = W_{\hat{m}} h_{t-1} + b_{\hat{m}}. \tag{11}$$

Because D does not involve in dealing with incomplete data, we do not have to introduce a decay term to D. Alternatively, we implant the hint mechanism which is proposed by [22] to the input of D. The intuition of hint mechanism is to define a hint matrix N, which introduces partial information of M to D so that D can help G to learn the distribution of the real data [22]. The hint matrix N is designed to be the following:

$$N = B \odot M + 0.5(1 - B) \tag{12}$$

where $B \in \{0, 1\}^{T \times D}$ is a random variable that obeys the following Bernoulli distribution to select elements in mask M to pass to N

$$\Pr\left(b_t^d = b\right) = \begin{cases} p, & b = 0 \\ 1 - p, & b = 1 \end{cases} \tag{13}$$

where b_t^d is an element of B located at position (t, b). It is obvious that if n_t^d equals to 0 or 1, we can infer that m_t^d also equals to 0 or 1, correspondingly. It is noteworthy that if the p is very high, we will discard too much hint information, and it will be very difficult for D to work normally because the discriminations of both "real" values and "fake" values are all based on the data reconstructed by G. In contrast, if we provide too much knowledge about M, there will be a huge risk of overfitting.

Combining with $\hat{X}$, N is utilized as part of the input for the updating of D as follows (with a little bit abuse of notation):

$$h_t = \sigma\left(W_{hh} h_{t-1} + W_{hx}(\hat{x}_t \circ n_t) + b_h\right). \tag{14}$$

3) Objective: The generative adversarial training framework is generally understood to mean a two-player minimax game. In our model, inspired by [21], we extend the ordinary GAN to a conditional model. The generation of missing values is based on the prior information of observed measurements (and labels, if available). Therefore, our minimax game with

value function can be described as

$$\min_G \max_D V(D,G) = \mathbb{E}_{\hat{X}\sim p_{\text{data}}(\hat{X}|X,y)}[M^T \log D(\hat{X},N) + (1-M)^T \log(1-D(\hat{X},N))] \quad (15)$$

where $\hat{X}$ is the estimation of the completed data, which is generated by G, i.e., $\hat{X} = G(X)$.

Recall that the discriminator D is trained to recognize which values in a completed time series are imputed by G. To put it in another way, given a completed time series $\hat{X}$, D tries to estimate its original mask matrix M. Consequently, the loss of D is set to be the binary cross entropy (BCE) to measure the similarity of $\hat{M}$ and M as

$$\mathcal{L}_D(D,G) = \text{BCE}(\hat{M},M) = \sum_{t=1}^{T}\sum_{d=1}^{D}\left[m_t^d \log\left(\hat{m}_t^d\right) + \left(1-m_t^d\right)\log\left(1-\hat{m}_t^d\right)\right]. \quad (16)$$

Then, for a fixed generator G, we update the discriminator D via the following minimization criterion:

$$\arg\min_{\theta_D} \mathcal{L}_D(D,G) \quad (17)$$

where θ_D is the set of parameters in D. Apparently, (17) forces D to recognize whether data are observed or unobserved in $\hat{X}$.

The training process of G is followed by the update of D. The loss function of G consists of two major parts: one of which is the discriminator loss $\mathcal{L}_D$, and another part is the difference between the imputed time series and the original observed values. Note that when the label is available in training data, we will also care about the performance of the classifier in (9) by taking the classification loss into consideration. We formulate the entire loss function of G in the following equation:

$$\mathcal{L}_G(D,G) = -\mathcal{L}_D(D,G) + \gamma\,\mathcal{L}_I(\hat{X},X) + \lambda\mathcal{L}_P(\hat{y},y) \quad (18)$$

where γ and λ are hyperparameters to control the proportion of three parts of loss, $\mathcal{L}_I(\hat{X},X)$ denotes the imputation loss to measure the disparity between real values and corresponding imputed values, and $\mathcal{L}_P$ is the prediction loss. $\mathcal{L}_I(G(X),X,M)$ is designed to be

$$\mathcal{L}_I(\hat{X},X) = \sum_{t=1}^{T}\sum_{d=1}^{D}\left[\left|\left(x_t^d - \bar{x}_t^d\right)m_t^d\right| + \left|\left(x_t^d - \tilde{x}_t^d\right)m_t^d\right| + \left|\left(x_t^d - \hat{x}_t^d\right)m_t^d\right|\right]. \quad (19)$$

We have flexible options for the prediction loss $\mathcal{L}_P$, i.e., cross entropy loss or mean square loss, and so on. In this article, we use the cross entropy to evaluate the prediction error.

Then, G is updated by the following minimization criterion:

$$\arg\min_{\theta_G} \mathcal{L}_G(D,G) \quad (20)$$

where θ_G represents the parameters of G. If we train G by minimizing $\mathcal{L}_G$ as in (20), the loss of D is implicitly maximized based on the current output of G; thus, G will be updated in the direction of confusing D so that D cannot recognize the imputed values. In another word, the imputations generated by G will have a close distribution compared with that of the original data. It is worth to note that the entire discriminator loss rather than the mere discriminative loss of generated values is included in $\mathcal{L}_G$ because the discrimination of original observed values is also based on the imputed data. Besides, inspired by [39], we use the second term in the loss of G, i.e., $\mathcal{L}_G$, to make the difference between the evaluations and observed values as small as possible, which guarantees that we can generate more reasonable data. Algorithm 1 demonstrates a more detailed training procedure.

Algorithm 1 Minibatch Stochastic Gradient Descent (SGD) Training Procedure of Our Model

for number of training epochs **do**
  for number of iterations in one epoch **do**
    Sample K minibatch samples $\{X^{(k)}\}_{k=1}^K$ as input from original incomplete time series.
    Sample K minibatch samples $\{N^{(k)}\}_{k=1}^K$ based on the Bernoulli process in Equation (13).
    Training D:
    Forward propagate G to derive $\{\hat{X}^{(k)}\}_{k=1}^K$.
    Update D by descending the following gradient while fixing G:

$$\nabla_{\theta_D}\frac{1}{K}\Sigma_{k=1}^K\mathcal{L}_D\left[D(\hat{X}^{(k)},N^{(k)}),G(X^{(k)})\right]$$

    Training G:
    Forward propagate D for $\{\hat{M}^{(k)}\}_{k=1}^K$.
    Update G by descending the gradient while fixing D:

$$\nabla_{\theta_G}\frac{1}{K}\Sigma_{k=1}^K\mathcal{L}_G\left[D(\hat{X}^{(k)},N^{(k)}),G(X^{(k)})\right]$$

  end for
end for

TABLE I
ORIGINAL MISSING RATES OF DATA SETS

Dateset	PhysioNet	PM25 Concentration	NO2 Concentration	Wind Speed
Missing rate	78.0 %	13.3 %	16.0 %	30.3 %

V. EXPERIMENTS

In this section, we evaluate our model on four real-world time series data sets: one of which is the PhysioNet Challenge 2012 data set [40], and the other three are air quality and meteorological data sets introduced by [41]. There corresponding missing rates are shown in Table I. In order to demonstrate the performance of our model, we make a comparison of the imputation results with a range of strong time series imputation methods including both statistical and neural network-based ones. We further manually mask different ratios of observed values to fully prove the advantage of our model.

A. Data Sets' Description

1) PhysioNet Challenge 2012 Data Set: This data set collects 4000 intensive care unit (ICU) records of patients. Every record is a multivariate time series lasting approximately 48 h, and the time series records 41 features of patients, including five general descriptors (e.g., age, gender, and weight) and

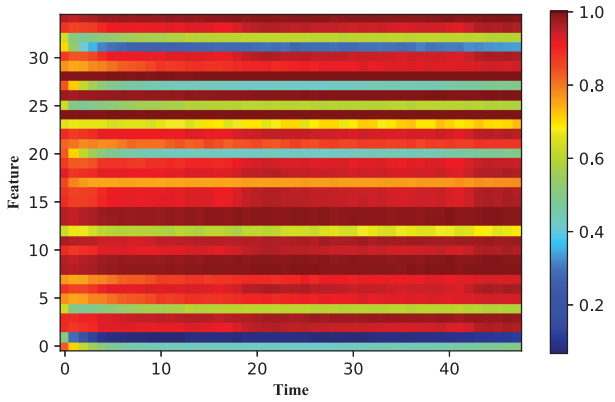

Fig. 4. Average missing rate of the PhysioNet data set.

36 measurements of vital signs (e.g., albumin, bilirubin, and cholesterol). When conducting experiments on this data set, we discarded the general descriptors because these five features are regarded as constants during the observation period. We further eliminated one of the 36 measurements as in [6], so the input size of each time series is 48×35. We note that there are numerous short- and long-term consecutive missing gaps in the original PhysioNet data set. For a clear demonstration of the missing pattern of the data set that we used, we plot the average missing rate of the PhysioNet data set in Fig. 4.

We conducted both imputation and classification experiments on this data set. For the imputation experiment, we manually masked ten percent observed values as ground truth for testing. For the classification experiment, we utilized the completed data to predict whether the patient died in ICU.

2) Air Quality and Meteorological Data Sets: The air quality and meteorological data sets are collected in the Urban Air[1] project. This project records air quality data, meteorological data, and weather forecast data from four major cities (i.e., Beijing, Tianjin, Guangzhou, and Shenzhen) in China over one year (from May 1, 2014, to April 30, 2015). In our experiments, we selected three categories, including PM25_Concentration, NO2_Concentration, and Wind_Speed, from all the meteorological factors specifically in Beijing. Note that for the PM25_Concentration and NO2_Concentration categories, there are totally 36 weather stations measuring the values, while the Wind_Speed data are collected from 16 weather stations. We arranged each measurement from each station as one dimension of the feature vector in a time step. Therefore, the feature dimensions of these three categories are 36 and 16, respectively. Besides, the data are collected by an hour. Thus, there are 8759 time steps in each category. We cut the original data into a time series every 36 h, which indicates that the input size of the air quality and meteorological data sets as 36×36 and 36×16, respectively. Particularly, we used the time series from March, June, September, and December as the test set. However, the real-world missing pattern of the meteorological data may not random [5]. Therefore, for the test set of these data sets, we did not casually mask the observed measurements but manually masked the nonmissing values in the position of these four months where the values in the corresponding positions of the next month are missing.

[1]https://www.microsoft.com/en-us/research/project/urban-air/

B. Implementation Details

Based on the structure of D and G described in Section IV, we implemented our model under the PyTorch framework. The dimensions of the hidden states in both D and G were controlled to be 64 in all the experiments, and the p was set to be 0.85 for the hint mechanism. We used Adam [42] optimizer to update our networks, and we did a grid search for the learning rates of D and G in the range of {0.01, 0.001, 0.0001, 0.00001}. The best results were achieved at 0.001 for G and 0.0001 for D, respectively. Meanwhile, when training all the abovementioned four data sets, the size of the minibatch was set to be 64. Besides, we explored the hyperparameter γ and λ in {1, 2, 4, 8}. The best results were obtained at $\gamma = 8$ and $\lambda = 1$ for the PhysioNet data set and $\gamma = 4$ for the air quality and meteorological data sets. We also conducted the classification task on the PhysioNet data set as an auxiliary verification of the imputation result, whose objective is to predict whether the patient died in ICU or not. Note that the labels of the PhysioNet data set are very imbalanced, the mortality patients of which is only 13.85%. According to previous work [6], [14], the area under the ROC curve (AUC) score was employed as the evaluation metric for comparison. For the imputation tasks, we used the mean absolute error (MAE) and mean relative error (MRE) to compare the performances of different methods.

C. Baseline Methods for Comparison

In order to demonstrate the outstanding performance of our model, we make comparisons with a number of commonly used methods for time series imputation including both traditional statistical and RNN-based methods.

1) Traditional Statistical Methods: One of the most fundamental methods is mean value imputation. As the name implies, it uses the global mean value of the time series to fill the empty positions. Another method is called MF that is usually applied in recommendation systems. Especially, the MF method reconstructs the original sparse matrix by the composition of two low-rank matrices. KNN uses the neighbor samples with high similarity to impute the missing values in the time series. MICE generates missing values by chained equations, and STMVL proposes a multiview learning framework for meteorological data imputation by using geographical information.

2) RNN-Based Methods: M-RNN employs a bidirectional RNN structure to exploit information both within and across time series for imputation. BRIST applies the decay term from [2] to an RNN structure similar to M-RNN. Luo *et al.* [14] also borrowed the decay term to the GRUI model but trains the model under a GAN architecture. However, Luo *et al.* [14] impute the missing values in an implicit way and do not exploit information within the time series, such as M-RNN and BRITS do.

D. Imputation Performance

In this section, we evaluate the imputation performance of our model based on the data sets introduced earlier.

TABLE II
IMPUTATION RESULTS [IN MAE (MRE%)] ON DIFFERENT DATA SETS

Method		PhysioNet	PM25	NO2	Wind Speed
Statistical Model	Mean imputation	0.4602(65.41%)	34.55(49.63%)	21.56(44.02%)	3.44(50.38%)
	KNN imputation	0.3674(52.22%)	32.28(46.37%)	27.07(55.27%)	6.30(92.05%)
	MICE imputation	0.5159(73.32%)	29.45(42.31%)	18.19(37.14%)	3.21(47.01%)
	ST-MVL	/	12.12(17.40%)	**8.45(17.10%)**	2.74(41.10%)
RNN-Based Model	M-RNN	0.4450(61.87%)	14.05(20.16%)	10.23(20.89%)	2.47(36.12%)
	BRITS	0.2780(38.72%)	11.56(16.65%)	8.71(17.78%)	2.44(35.55%)
Our Model	Vanilla LSTM	0.3811(54.16%)	12.65(18.18%)	8.86(18.15%)	2.45(35.84%)
	Only decay mechanism	0.3489(50.31%)	11.41(16.39%)	8.76(17.90%)	2.44(35.71%)
	Only feature correlations	0.2786(38.79%)	12.13(17.42%)	8.68(17.72%)	2.42(35.41%)
	Complete model	**0.2612(37.09%)**	**11.21(16.11%)**	8.52(17.39%)	**2.39(34.99%)**

Besides, we also conduct some auxiliary experiments to analyze the effectiveness of our model with respect to convergence, feature correlation term, and noise resistance. For the purpose of stably training the models, we conduct all the experiments based on the data normalized to be 0 mean and 1 standard deviation. Because the value scales of different types of physiological measurements in the PhysioNet data set vary vastly, we demonstrate the results in a normalized scale. For the other data sets, the values of data are in the same range, so the results are shown in the original scale.

1) Imputation Results With Different Data Sets: Table II demonstrates the imputation results of different data sets. From Table II, we can see that statistical methods except ST-MVL perform consistently worse in all the four data sets. Among these three basic methods, MICE imputation shows relatively better performance on the meteorological data sets. One possible reason is that the different feature dimensions in these data sets are meteorological measurements from different weather stations collected at the same time. The geographical proximity of the stations makes the correlation between different features slightly stronger, which is easier for the regression algorithms in MICE to capture. For the PhysioNet data set, the KNN imputation can achieve a lower estimation error. We ascribe this to the similarities of physiological indices between the patients who have a similar illness. The ST-MVL method is implemented under a multiview learning framework. It shows very competitive results in the meteorological data sets, especially in the data set of NO2 Concentration. The success of ST-MVL can be partially attributed to the extra information from the geographical view. However, in other application scenarios where the multiview information is not available (e.g., the PhysioNet data set), it can hardly perform well or even be implemented. Thus, there are always certain limitations on the applications of traditional statistical methods. It is of high difficulty to generalize these methods to more general cases. In our model, there are two specially designed components for dealing with incomplete time series, i.e., temporal decay term and feature correlation term. In order to investigate the effectiveness of them, we further conducted an ablation study by manually eliminating one or both of these two components. From the results in Table II, we can see that two components can both contribute to a more accurate reconstruction of the time series. However, the decay mechanism endows our model with marginal performance improvement in the PhysioNet data set compared with other data sets. It may be caused by the especially high missing rate which partly interferes with the reconstruction via temporal dependencies.

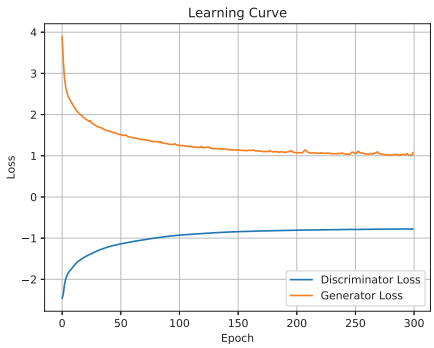

Fig. 5. Learning curve of our model.

In general, it is much easier for RNN-based models to capture the potential dynamics in time series. For example, M-RNN demonstrates an impressive improvement compared with the three basic statistical methods because of the bidirectional recurrent connections and well-designed structures for the exploitation of the multidirection correlations in time series. BRITS and our model all yield remarkable imputation results that exceed M-RNN to a large extent. Although there are some similarities between the behaviors of BRITS and our model, the experimental results reveal that our model uniformly outperforms the BRITS on the four data sets, which empirically demonstrates the benefit of the adversarial training strategy.

2) Convergence: Fig. 5 demonstrates the training losses of the discriminator and the generator. The BCE loss of the discriminator approximately converges to $-0.7(\approx \log(0.5))$ after training for 300 epochs, which means that D has been fooled by the generated values. Gs loss also converges to a small value, which indicates a good reconstruction performance of G.

3) Feature Correlation Visualization: In Section IV, we illustrate the motivation of the feature correlation mechanism by plotting the average Pearson correlation coefficients

TABLE III
IMPUTATION RESULTS [IN MAE (MRE%)] ON WIND SPEED DATA SET WITH STANDARD NORMAL NOISE UNDER DIFFERENT NOISED DATA RATIOS

Noised Data Ratio	Mean imputation	KNN imputation	M-RNN	BRITS	Our model
0%	3.44(50.38%)	6.39(92.05%)	2.47(41.10%)	2.44(35.55%)	2.39(34.99%)
1%	3.44(50.37%)	6.38(91.51%)	2.49(41.17%)	2.44(35.77%)	2.40(35.03%)
5%	3.45(50.38%)	6.39(92.17%)	2.52(41.25%)	2.49(36.37%)	2.42(35.40%)
10%	3.48(50.40%)	6.39(92.45%)	2.59(41.42%)	2.52(36.88%)	2.46(36.10%)
100%	3.50(51.42%)	6.41(93.75%)	2.76(41.90%)	2.64(38.93%)	2.55(37.29%)

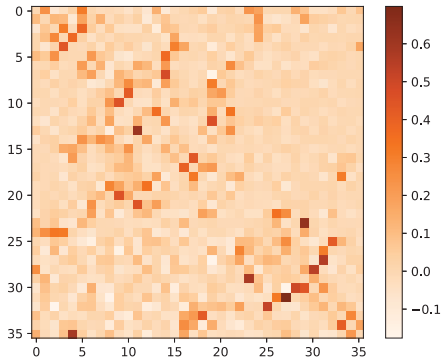

Fig. 6. Visualization of the learned feature correlation weights' matrix.

between any two features vectors in the PM25 concentration data set, which shows a clear positively correlated pattern. It is reasonable to postulate that one can estimate the missing PM25 concentration value by referring to the data from nearby observatories. In Fig. 6, we visualize the learned feature correlation weights matrix $W_{\tilde{x}}$ in (5) of the PM25 concentration data set in order to explicitly demonstrate the function of the feature correlation term. From Fig. 6, we can find that the elements in the matrix with significantly large absolute values are all positive, which indicates that the input features are positively correlated with some other features within the same time step. This phenomenon is consistent with the Pearson correlation coefficients of the original data. Moreover, Fig. 6 demonstrates an analogous pattern similar to than in Fig. 2, which means that the feature correlation term can indeed capture the correlation between different features.

4) Noise Resistance: Noise inevitably exists in real-world data set. In order to better analyze the noise resistance of our model, we add noises that obey the standard normal distribution to the wind speed data set under different ratios between added noises and original data and then conduct experiments to compare the performance of different models. The results are shown in Table III. From Table III, we can see that the imputation results of the statistic models only receive marginally decrease, while the performance decline of the RNN-based models is more significant. This is most probably because the statistic models barely consider the temporal dependencies that may be interfered with by the noise. Among the RNN-based models, our model can still achieve the best imputation performance with a relatively small drop of MAE(MRE), which means that our model enjoys better noise resistance.

E. Classification Results

In this section, we deploy the classification experiments on the PhysioNet data set to predict whether a patient died in ICU.

TABLE IV
CLASSIFICATION RESULTS (IN AUC) ON PHYSIONET DATA SET

Method	AUC Score
RNN-Based Model Named GRU-D [2]	0.8424 ± 0.012
RNN-Based Model Named BRITS [6]	0.8500 ± 0.002
GAN-Based Model Named GAN+GRUI [14]	0.8603 ± 0.008
Our Model	0.8631 ± 0.006

The classification result can be viewed as an indirect proof of the imputation accuracy. For our model, we stack a fully connected layer on the last hidden state of the modified RNN with a 50% dropout rate [43]. For comparison, we choose GRU-D, BRITS, and GAN + GRUI as baseline models. Although GRU-D and GAN + GRUI do not focus on an accurate imputation result, their performances on the classification task are impressive [6]. As indicated in Section V-B, we evaluate the classification task according to the AUC score as comparative methods do. We run our model under different random seeds for three times. The mean and variance of the AUC scores are demonstrated in Table IV.

F. Imputation Results With Different Masking Rate

In some real-world examples, the missing rate of time series data can be extremely high. In this section, we simulate this situation by manually masking some of the measurements in the time series as missing values, and these values are viewed as ground-truth for validation. To make the experiment more convincing, we specifically choose the PhysioNet data set for analysis because the original missing rate is fairly high. After randomly removing 10%–90% of the original observed data, we construct the whole data set with less and less available data remains. We conduct imputation experiments on these subsampled data sets with both statistical and RNN-based models, including the GAN + GRUI model. The results are shown in Table V. As the results imply, our model substantially outperforms other baseline methods except for BRITS regardless of the masking rate. Although our performance improvement compared with the BRITS model is not as significant as with other methods, we still consistently perform better. It is worth noting that in the results of GAN + GRUI, there is hardly a discernible trend with the increasing missing rate. Besides, the MAE and MRE values are exaggeratedly high, which indicates that the imputation of GAN + GRUI is fairly vague. Fig. 7 shows the variation trends of the imputation results (in MAE) of different methods with increasing masking rate. From the figure, we can see that for all the imputation methods, the estimation errors of missing values become larger when there is fewer data to be used. Another interesting fact is that the improvements in the estimation errors of

TABLE V
IMPUTATION RESULTS [IN MAE (MRE%)] ON PHYSIONET DATA SET WITH DIFFERENT MASKING RATES

Masking rate	Mean imputation	KNN imputation	MICE imputation	MF imputation	M-RNN	BRITS	GAN+GRUI	Our model
10%	0.4602(65.41%)	0.3674(52.22%)	0.5159(73.32%)	0.4405(62.60%)	0.3597(51.12%)	0.2780(38.72%)	0.7092(91.20%)	**0.2612(37.09%)**
20%	0.4663(66.08%)	0.4008(56.80%)	0.5326(75.48%)	0.4727(66.99%)	0.3696(52.38%)	0.2875(40.75%)	0.6678(86.01%)	**0.2773(39.29%)**
30%	0.4678(66.26%)	0.4274(60.53%)	0.5481(77.64%)	0.5043(71.42%)	0.3789(53.67%)	0.3027(42.87%)	0.8133(104.94%)	**0.2968(42.04%)**
40%	0.4702(66.71%)	0.4488(63.68%)	0.5627(79.84%)	0.5303(75.24%)	0.3849(54.62%)	0.3228(45.81%)	0.7220(92.94%)	**0.3121(44.29%)**
50%	0.4724(66.98%)	0.4644(65.85%)	0.5792(82.12%)	0.5575(79.05%)	0.3927(55.68%)	0.3360(47.64%)	0.9080(116.41%)	**0.3311(46.94%)**
60%	0.4753(67.43%)	0.4776(67.76%)	0.5929(84.13%)	0.5809(82.43%)	0.4005(56.82%)	0.3527(50.05%)	0.9065(117.15%)	**0.3457(49.06%)**
70%	0.4778(67.87%)	0.4876(69.27%)	0.6072(86.26%)	0.6021(85.53%)	0.4084(58.02%)	0.3615(51.35%)	0.7118(91.88%)	**0.3578(50.83%)**
80%	0.4816(68.25%)	0.5005(70.92%)	0.6196(87.80%)	0.6245(88.50%)	0.4167(59.05%)	0.3727(52.82%)	0.8717(112.71%)	**0.3689(52.27%)**
90%	0.4853(68.83%)	0.5142(72.94%)	0.6284(89.13%)	0.6405(90.84%)	0.4227(59.95%)	0.3842(54.49%)	0.9219(118.67%)	**0.3803(53.94%)**

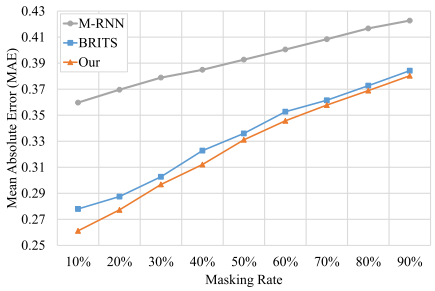

Fig. 7. Imputation results with increasing masking rate.

our model tend to be decreasing as the masking rate grows. A possible reason is that when the masking rate getting higher, the adversarial training architecture can capture less and less effective information to assist in the generation of more real data, which may cause a degradation of our model.

VI. CONCLUSION

In this article, in order to address the missing values problem of time series, we propose a novel learning strategy based on a generative adversarial learning framework. In our model, the generator is a modified RNN that is customized for dealing with incomplete time series. In order to fully exploit the interdependencies across time steps and inner correlations between variables within a single time step, we borrow the decay mechanism and the feature correlation term to the RNN structure. Besides, the missing pattern of time series also provides valuable information for the update of RNN. The discriminator takes the reconstructed time series and the hint information to predict whether a value in the time series is imputed or not. In return, the discriminative loss is utilized to assist the generator to generate missing values toward a more authentic distribution. Experimental results of several real-world data sets show that our model can achieve more accurate imputation results compared with baseline methods, even if under the extremely high missing rate.

REFERENCES

[1] S. Xiao, J. Yan, M. Farajtabar, L. Song, X. Yang, and H. Zha, "Learning time series associated event sequences with recurrent point process networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 10, pp. 3124–3136, Oct. 2019.

[2] Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu, "Recurrent neural networks for multivariate time series with missing values," *Sci. Rep.*, vol. 8, no. 1, p. 6085, Dec. 2018.

[3] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-C. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 802–810.

[4] S. Bauer, B. Schölkopf, and J. Peters, "The arrow of time in multivariate time series," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 2043–2051.

[5] X. Yi, Y. Zheng, J. Zhang, and T. Li, "ST-MVL: Filling missing values in geo-sensory time series data," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, 2016, pp. 1–7.

[6] W. Cao, D. Wang, J. Li, H. Zhou, L. Li, and Y. Li, "BRITS: Bidirectional recurrent imputation for time series," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 6776–6786.

[7] P. E. McKnight, K. M. McKnight, S. Sidani, and A. J. Figueredo, *Missing Data: A Gentle Introduction*. New York, NY, USA: Guilford Press, 2007.

[8] J. W. Graham, "Missing data analysis: Making it work in the real world," *Annu. Rev. Psychol.*, vol. 60, no. 1, pp. 549–576, Jan. 2009.

[9] P. J. García-Laencina, J.-L. Sancho-Gómez, and A. R. Figueiras-Vidal, "Pattern classification with missing data: A review," *Neural Comput. Appl.*, vol. 19, no. 2, pp. 263–282, Mar. 2010.

[10] J. L. Schafer, *Analysis of Incomplete Multivariate Data*. London, U.K.: Chapman & Hall, 1997.

[11] G. Batista and M. C. Monard, "A study of K-nearest neighbour as an imputation method," *Hybrid Intell. Syst., Frontiers Artif. Intell. Appl.*, vol. 87, no. 46, pp. 251–260, 2002.

[12] K. Rehfeld, N. Marwan, J. Heitzig, and J. Kurths, "Comparison of correlation analysis techniques for irregularly sampled time series," *Nonlinear Processes Geophysics*, vol. 18, no. 3, pp. 389–404, Jun. 2011.

[13] H.-F. Yu, N. Rao, and I. S. Dhillon, "Temporal regularized matrix factorization for high-dimensional time series prediction," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 847–855.

[14] Y. Luo *et al.*, "Multivariate time series imputation with generative adversarial networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 1603–1614.

[15] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[16] Y. Bengio and F. Gingras, "Recurrent neural networks for missing or asynchronous data," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 1996, pp. 395–401.

[17] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, 1997.

[18] J. Yoon, W. R. Zame, and M. van der Schaar, "Estimating missing data in temporal data streams using multi-directional recurrent neural networks," *IEEE Trans. Biomed. Eng.*, vol. 66, no. 5, pp. 1477–1490, May 2019.

[19] I. J. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

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

[21] M. Mirza and S. Osindero, "Conditional generative adversarial nets," 2014, *arXiv:1411.1784*. [Online]. Available: http://arxiv.org/abs/1411.1784

[22] J. Yoon, J. Jordon, and M. van der Schaar, "GAIN: Missing data imputation using generative adversarial nets," 2018, *arXiv:1806.02920*. [Online]. Available: http://arxiv.org/abs/1806.02920

[23] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes: The Art of Scientific Computing*, 3rd ed. New York, NY, USA: Cambridge Univ. Press, 2007.
[24] R. J. Little and D. B. Rubin, *Statistical Analysis with Missing Data*, vol. 333. Hoboken, NJ, USA: Wiley, 2014.
[25] H.-H. Hsu, A. C. Yang, and M.-D. Lu, "KNN-DTW based missing value imputation for microarray time series data," *J. Comput.*, vol. 6, no. 3, pp. 418–425, Mar. 2011.
[26] M. J. Azur, E. A. Stuart, C. Frangakis, and P. J. Leaf, "Multiple imputation by chained equations: What is it and how does it work?" *Int. J. Methods Psychiatric Res.*, vol. 20, no. 1, pp. 40–49, Mar. 2011.
[27] G. Y. Lu and D. W. Wong, "An adaptive inverse-distance weighting spatial interpolation technique," *Comput. Geosci.*, vol. 34, no. 9, pp. 1044–1055, Sep. 2008.
[28] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Adv. Artif. Intell.*, vol. 2009, pp. 421–425, Oct. 2009.
[29] D. B. Rubin, "Inference and missing data," *Biometrika*, vol. 63, no. 3, pp. 581–592, 1976.
[30] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015, *arXiv:1511.06434*. [Online]. Available: http://arxiv.org/abs/1511.06434
[31] X. Chen, C. Xu, X. Yang, L. Song, and D. Tao, "Gated-GAN: Adversarial gated networks for multi-collection style transfer," *IEEE Trans. Image Process.*, vol. 28, no. 2, pp. 546–560, Feb. 2019.
[32] L. Yu, W. Zhang, J. Wang, and Y. Yu, "Seqgan: Sequence generative adversarial nets with policy gradient," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 2852–2858.
[33] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 7167–7176.
[34] S. Xiao, M. Farajtabar, X. Ye, J. Yan, L. Song, and H. Zha, "Wasserstein learning of deep generative point process models," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 3247–3257.
[35] T. Guo, C. Xu, B. Shi, C. Xu, and D. Tao, "Learning from bad data via generation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 6042–6053.
[36] Y. Li, S. Liu, J. Yang, and M.-H. Yang, "Generative face completion," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3911–3919.
[37] P. Liu, X. Qi, P. He, Y. Li, M. R. Lyu, and I. King, "Semantically consistent image completion with fine-grained details," 2017, *arXiv:1711.09345*. [Online]. Available: http://arxiv.org/abs/1711.09345
[38] L. Zhou and G. Hripcsak, "Temporal reasoning with medical data—A review with emphasis on medical natural language processing," *J. Biomed. Inform.*, vol. 40, no. 2, pp. 183–202, 2007.
[39] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery," in *Proc. 25th Int. Conf. Inf. Process. Med. Imag.* Cham, Switzerland: Springer, 2017, pp. 146–157.
[40] I. Silva, G. Moody, J. S. Daniel, L. A. Celi, and R. G. Mark, "Predicting in-hospital mortality of ICU patients: The PhysioNet/Computing in cardiology challenge 2012," *Comput. Cardiol.*, vol. 39, pp. 245–248, Sep. 2012.
[41] Y. Zheng *et al.*, "Forecasting fine-grained air quality based on big data," in *Proc. 21st ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*. New York, NY, USA: ACM, 2015, pp. 2267–2276.
[42] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980
[43] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

Shuo Yang received the B.E. degree from Sun Yat-sen University, Guangzhou, China, in 2018. He is currently pursuing the Ph.D. degree with the School of Computer Science, The University of Sydney, Sydney, NSW, Australia.

His research interests include primarily in time series analysis and machine learning theory.

Minjing Dong received the bachelor's degree from The University of Sydney, Sydney, NSW, Australia, in 2018, where he is currently pursuing the M.Phil. degree with the School of Computer Science.

His research interests lie in human behavior analysis and network compression.

Yunhe Wang received the B.E. degree from Xidian University, Xi'an, China, in 2013, and the Ph.D. degree from Peking University, Beijing, China, in 2018.

He is currently a Senior Researcher with the Noah's Ark Laboratory, Huawei Technologies Company Ltd., Beijing. His research interests lie primarily in deep learning, computer vision, and machine learning.

Chang Xu (Member, IEEE) received the Ph.D. degree from Peking University, Beijing, China, in 2016.

He is currently a Lecturer and an ARC DECRA Fellow with the School of Computer Science, The University of Sydney, Sydney, NSW, Australia. He has published over 80 articles in prestigious journals and top tier conferences, including the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE (T-PAMI), IEEE T-NNLS, the IEEE TRANSACTIONS ON IMAGE PROCESSING (T-IP), the International Conference on Machine Learning (ICML), the Advances in Neural Information Processing Systems (NeurIPS), the International Joint Conference on Artificial Intelligence (IJCAI), and the AAAI Conference on Artificial Intelligence (AAAI). His research interests lie in machine learning algorithms and related applications in computer vision.

Dr. Xu has regularly severed as a PC Member or a Senior PC Member for many conferences, e.g., NeurIPS, ICML, the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), the IEEE International Conference on Computer Vision (ICCV), IJCAI, and AAAI. He has been recognized as a Top Ten Distinguished Senior PC Member in IJCAI 2017. He received several paper awards, including the Distinguished Paper Award in IJCAI 2018.