Joint Modeling of Local and Global Temporal Dynamics for Multivariate Time Series Forecasting with Missing Values

Xianfeng Tang,[†] Huaxiu Yao,[†] Yiwei Sun,[†] Charu Aggarwal,[‡] Prasenjit Mitra,[†] Suhang Wang[†]*
[†]College of Information Sciences and Technology, Pennsylvania State University, PA, USA
[‡]IBM T.J. Watson, NY, USA
[†]{xut10, huy144, yus162, pum10, szw494}@psu.edu; [‡]charu@us.ibm.com

Abstract

Multivariate time series (MTS) forecasting is widely used in various domains, such as meteorology and traffic. Due to limitations on data collection, transmission, and storage, real-world MTS data usually contains missing values, making it infeasible to apply existing MTS forecasting models such as linear regression and recurrent neural networks. Though many efforts have been devoted to this problem, most of them solely rely on local dependencies for imputing missing values, which ignores global temporal dynamics. Local dependencies/patterns would become less useful when the missing ratio is high, or the data have consecutive missing values; while exploring global patterns can alleviate such problem. Thus, jointly modeling *local* and *global* temporal dynamics is very promising for MTS forecasting with missing values. However, work in this direction is rather limited. Therefore, we study a novel problem of MTS forecasting with missing values by jointly exploring local and global temporal dynamics. We propose a new framework LGnet, which leverages memory network to explore global patterns given estimations from local perspectives. We further introduce adversarial training to enhance the modeling of global temporal distribution. Experimental results on real-world datasets show the effectiveness of LGnet for MTS forecasting with missing values and its robustness under various missing ratios.

Introduction

Multivariate time series (MTS) forecasting is widely used in many applications such as weather forecasting (Xingjian et al. 2015), clinical diagnosis (Che et al. 2018), sales forecasting (Wu et al. 2018; 2019) and traffic analysis (Yao et al. 2019b; 2018; 2019a; Tang et al. 2019). The popularity of MTS forecasting has attracted increasing attention, and many efforts have been taken to address the problem in the past few years (Box et al. 2015; Qin et al. 2017; Chang et al. 2018). Recurrent neural networks (RNNs), a class of deep learning frameworks designed for modeling sequential data, have been successfully applied to this problem. For example, LSTNet (Lai et al. 2018) adopts LSTM to capture long-term dependencies for time series forecasting.

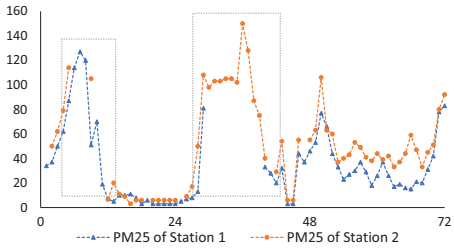

Figure 1: Two time series from Beijing air pollution dataset.

Qin et al. (Qin et al. 2017) designed two attention mechanisms in RNN to improve forecasting accuracy. Despite their success, *the majority of the aforementioned models assume MTS data is complete.*

In the real-world, MTS data are usually incomplete due to various reasons, such as broken sensors, failed data transmissions, or damaged storages. For example, Figure 1 gives two multivariate time series snippets from Beijing air pollution data, both of which contain apparent missing values marked by gray boxes (i.e., 20 of the 144 data points are unobserved). Missing values damage temporal dependencies in MTS sequences (Luo et al. 2018; Cao et al. 2018), make it hard to apply existing RNN-based models on incomplete sequences and increase the difficulty of MTS forecasting tasks. As shown in Figure 1, because of missing values, the second peak of the blue signal is not observed, and cannot be inferred by simply relying on existing RNNs. Therefore, it is vital to design models that handle missing values in MTS data to perform accurate forecasting. Many prior efforts have been dedicated to this direction. For example, two-step approaches that first omit or impute missing values then process time series forecasting based on the pre-processed data are explored in (Yi et al. 2016; García-Laencina, Sancho-Gómez, and Figueiras-Vidal 2010). End-to-end solutions, where the missing patterns are modeled jointly with forecasting tasks, are investigated in (Che et al. 2018; Cao et al. 2018; Luo et al. 2018). However, those methods only explore local statistical features, while the global temporal patterns in exogenous sequences are neglected.

Jointly modeling *local* and *global* temporal dynamics is

*Corresponding Author

very promising for MTS forecasting with missing values. Though constructing local statistics (e.g., empirical mean and last observations) to estimate missing variables have certain potential (Che et al. 2018), these local statistics are unreliable when the missing ratio raises up or consecutive missing values occur as illustrated in Figure 1. This problem can be alleviated by adopting global temporal dynamics. From a global perspective, there exist many MTS snippets with close temporal patterns. For example, in Figure 1, two MTS sequences from different air quality stations share similar temporal patterns. Although it is hard to recover consecutive missing values (i.e., circled by the grey dash boxes) purely from local statistics of one MTS, aggregating temporal patterns in both sequences is rather promising. The temporal patterns of one time series can be utilized for the other when dealing with missing values. However, how to take advantage of global temporal dynamics is a very challenging problem, which is under-explored in existing work.

To address the aforementioned challenges, we propose a novel framework LGnet to model **L**ocal and **G**lobal temporal dynamics jointly for MTS forecasting. LGnet absorbs model designs from previous work (Che et al. 2018), where LSTM is leveraged for MTS forecasting. Since the original LSTM is unable to handle incomplete input, we first construct estimations for missing values. Specifically, representative local statistic features are constructed for each variable in an MTS. Besides, a memory module is designed for LGnet to explicitly leverage knowledge from exogenous sequences to generate global estimations for missing values. This is achieved by using local statistics as keys to query a global optimized memory component. We further introduce adversarial training to enhance the modeling of global temporal distribution. A discriminator is built to identify the generated MTS from real samples. Meanwhile, LGnet aims at producing forecasting sequences that are hard to be identified, which are also closer to the actual global distribution of real MTS data. The main contributions of the paper are:

- We study a new problem of MTS forecasting with missing values by exploring local and global temporal dynamics.
- We propose a novel framework LGnet, with a memory module to capture global temporal dynamics for missing values and adversarial training to enhances the modeling of global temporal distribution.
- We conduct extensive experiments on four large-scale real-world datasets to validate the proposed approach.

Related Work

Various methods have been proposed for MTS forecasting, such as Autoregressive (AR), Vector Autoregression (VAR), Autoregressive moving average (ARMA), standard regression models (e.g., support vector regression (Smola and Schölkopf 2004), linear regression, and regression tree methods (Chen and Guestrin 2016)). Inspired by the recent success of deep neural networks, many RNN-based methods (Lai et al. 2018; Qin et al. 2017) are developed for MTS forecasting. Even some vanilla RNNs, such as GRU (Chung et al. 2014) and LSTM (Hochreiter and Schmidhuber 1997), can outperform the non deep learning models significantly (Chang et al. 2018). However, *none of those approaches can handle input with missing values*.

To handle missing values in MTS, the simplest solution would be removing all samples with missing values, such as pairwise deletion (Marsh 1998). Obviously, such methods *ignore many useful information*, especially with a high missing ratio (King et al. 1998). General data imputation methods such as statistical imputation (e.g., mean, median), EM-based imputation (Nelwamondo, Mohamed, and Marwala 2007), K-nearest neighborhood (Friedman, Hastie, and Tibshirani 2001), and matrix factorization (Friedman, Hastie, and Tibshirani 2001) can be applied for the unobserved variables. However, those general approaches *fail to model temporal dynamics of time series*. Even if MTS imputation methods, such as multivariate imputation by chained equations (Azur et al. 2011) and generative adversarial network Luo et al., can be applied to fill in missing values first, *training a forecasting model on pre-processed MTS data would lead to sub-optimal results, since the temporal patterns of missing values are totally isolated from forecasting models* (Wells et al. 2013). To tackle this issue, some researchers propose end-to-end frameworks that jointly estimate missing values and forecast future MTS. Che et al. introduce GRU-D that imputes missing values using the linear combination of statistical features. Yoon, Zame, and van der Schaar propose M-RNN that leverages bi-directional RNN for the imputation. Cao et al. model the relationships between missing variables to simultaneously perform imputation and classification/regression in one neural graph. However, *those solutions focus on localized temporal dependencies and fail to model global temporal dynamics*.

Problem Formulation

In practice, many multivariate time series signals are sampled evenly. Thus, we assume time span is divided into equal-length time intervals. Let $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ denote one MTS of length n, where $\mathbf{x}_i \in \mathbb{R}^d$ is the observation at the i-th time interval, x_i^j is the j-th variable of $\mathbf{x}_i$, and d is the number of variables. Let mask matrix $\mathbf{M} = \{\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_n\}, \mathbf{m}_i \in \{0, 1\}^d$ denote the missing status of each variable, where $m_i^j = 0$ if x_i^j is unobserved/missing, otherwise, $m_i^j = 1$. Note that we can use a symbol to denote missing values in $\mathbf{X}$ (e.g., $null$).

We are interested in a general MTS forecasting task. Given N incomplete MTS observation $\{\mathbf{X}^j\}_{j=1}^N$ and their masks $\{\mathbf{M}^j\}_{j=1}^N$, we aim at learning a function f that can forecast the values in future k time intervals ($\{\mathbf{x}_{n+1}, \ldots, \mathbf{x}_{n+k}\}$) of any new MTS, given its historical n observations $\mathbf{X}$ with the mask matrix $\mathbf{M}$.

The Proposed Framework

Figure 2 illustrates the proposed framework LGnet. LGnet is built on LSTM to forecast future MTS values. We design a memory module which contains temporal information from exogenous sequences to impute MTS during the running time of LSTM. Specifically, we first extract local

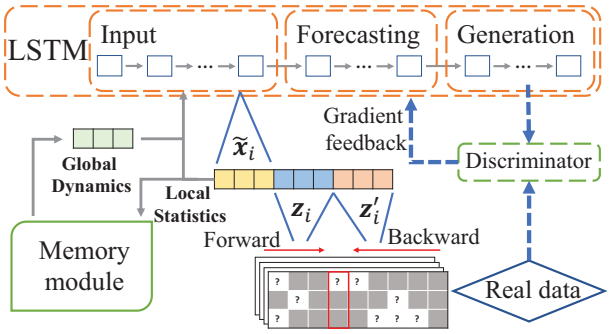

Figure 2: An illustration of LGnet.

statistic features for every time interval, then use them as keys to query a memory component, which is jointly optimized with LSTM on all MTS data. The query results, which preserve global temporal dynamics, are further combined with local statistic features to serve as the input of the LSTM. We also introduce adversarial training on forecasted sequences to make sure they follow the global distribution. The whole framework of LGnet is trained in an end-to-end manner. Next, we introduce each module of LGnet in detail.

MTS forecasting with LSTM

Recurrent neural networks (RNNs) have demonstrate remarkable success in various MTS forecasting tasks (Lai et al. 2018; Chang et al. 2018). To leverage the advances of RNN, we build LGnet on the top of Long short-term memory (LSTM) network, a variant of RNN which is able to capture long/short term dependency. Note that other RNN variants such as GRU (Cho et al. 2014) can serve as the replacement of the LSTM. Formally, LSTM takes one data point of the time series as input in each step, and iterates from $\mathbf{x}_1$ to $\mathbf{x}_n$. Suppose $\mathbf{x}_t$ is currently fed to the LSTM, $\mathbf{h}_{t-1}$ and $\mathbf{c}_{t-1}$ are the hidden state and memory cell of LSTM at previous step $t-1$, then the hidden state of memory cell at time t can be calculated as:

$$\mathbf{i}_t = \sigma\big(\mathbf{x}_t\mathbf{W}^i + \mathbf{h}_{t-1}\mathbf{U}^i + \mathbf{b}^i\big),\ \mathbf{f}_t = \sigma\big(\mathbf{x}_t\mathbf{W}^f + \mathbf{h}_{t-1}\mathbf{U}^f + \mathbf{b}^f\big)$$
$$\mathbf{o}_t = \sigma\big(\mathbf{x}_t\mathbf{W}^o + \mathbf{h}_{t-1}\mathbf{U}^o + \mathbf{b}^o\big),\ \tilde{\mathbf{c}}_t = \tanh\big(\mathbf{x}_t\mathbf{W}^c + \mathbf{h}_{t-1}\mathbf{U}^c\big)$$
$$\mathbf{c}_t = f_t \odot \mathbf{c}_{t-1} + i_t \odot \tilde{\mathbf{c}}_t, \qquad \mathbf{h}_t = \tanh(\mathbf{c}_t) \odot \mathbf{o}_t$$

where $\mathbf{i}_t$ is the input gate, $\mathbf{f}_t$ is the forget gate, $\mathbf{o}_t$ is the output gate, $\odot$ is the element-wise product, σ is the sigmoid function, and $\mathbf{W}^*$, $\mathbf{U}^*$, $\mathbf{b}^*$ are parameters. Given the current hidden state $\mathbf{h}_t$, the forecasting of next data point $\tilde{\mathbf{x}}_{t+1}$ can be generated recurrently as:

$$\tilde{\mathbf{x}}_{t+1} = \mathbf{h}_t\mathbf{U} + \mathbf{b}. \tag{1}$$

However, the original LSTM cannot handle missing values in its input. Obviously, if $\mathbf{x}_t$ contains unobserved variables, matrix productions such as $\mathbf{x}_t\mathbf{W}^i$ are invalid. One solution is using $\tilde{\mathbf{x}}_t$ as an alternative of $\mathbf{x}_t$. However, early errors in $\tilde{\mathbf{x}}_t$ can be quickly amplified in the following steps (Bengio et al. 2015), leading to inaccurate forecasting.

Therefore, appropriate estimations of missing values should be constructed for the LSTM. We tackle the problem by exploring temporal dynamics from both local and global perspectives with a memory module. In the next section, we discuss its technical details.

Memory Module

The basic idea of the memory module is to learn a parameterized memory which caches global temporal patterns and projects each variable to the same feature space with the memory. For each variable in a MTS, we first capture informative statistics from the local context of this time series, then leverage local statistics as keys to query the memory component, which returns representation vectors with global temporal dynamics. The memory module brings two advantages: (i) learn and store meaningful temporal patterns from a global perspective; and (ii) utilize the knowledge of temporal patterns to construct global representations. *Note that the memory module is not the memory cell of the LSTM as shown in Figure 2* and 3.

Capturing Local Statistics We extract useful local statistic features using the contextual information from observed parts of the time series for missing values. Following prior studies (Che et al. 2018), we first generate empirical mean and last observation of every time stamp as follows:

Empirical Mean: for variable x_i^j, we construct its empirical mean using all available observations of x_*^j before time i, i.e., $\bar{x}_i^j = \sum_{l=1}^{i-1} m_l^j x_l^j / \sum_{l=1}^{i-1} m_l^j$. The mean of previous observations reflects the time-aware data distribution of x_i^j and serves as the prior knowledge of the variable.

Last Observation: the last observation of x_i^j is the first available j-th variable before time interval i, which is the most temporally close neighbor. We use $\dot{x}_i^j$ to denote the last observation of x_i^j. Note that $\dot{x}_i^j$ isn't necessary equal to x_{i-1}^j because x_{i-1}^j could also be missing. We further introduce an indicator $\delta \in \mathbb{R}_+^{d\times n}$ to record the temporal distance between each x_i^j and its last observation, which reflects the confidence and trustworthy of previous values:

$$\delta_i^j = \begin{cases} 0, & \text{if } i = 1 \text{ or } m_i^j = 1 \\ \delta_{i-1}^j + 1, & \text{if } m_i^j = 0 \end{cases}. \tag{2}$$

Generally, when δ_i^j is small, we tend to trust $\dot{x}_i^j$ more; and when δ_i^j becomes larger, the averaged value $\bar{x}_i^j$ would be more representative. Based on the above assumption, we propose the following decaying mechanism to balance empirical mean and last observation:

$$\gamma(\delta_i^j) = \exp\left(-\max(0, w^j\delta_i^j + b^j)\right), \tag{3}$$

where w^j and b^j are parameters. The above decay mechanism leverages an exponentiated negative rectifier so that the decay value γ is monotonous decreasing in the range between 0 and 1 w.r.t δ_i^j (Che et al. 2018; Luo et al. 2018). The localized estimation for x_i^j is constructed as follows:

$$z_i^j \leftarrow m_i^j x_i^j + (1 - m_i^j)[\gamma(\delta_i^j)\dot{x}_i^j + (1 - \gamma(\delta_i^j))\bar{x}_i^j]. \tag{4}$$

We use $\mathbf{z}_i = [z_i^1, \ldots, z_i^d]$ to denote local statistic features for $\mathbf{x}_i$. For observed variables, their original values are directly

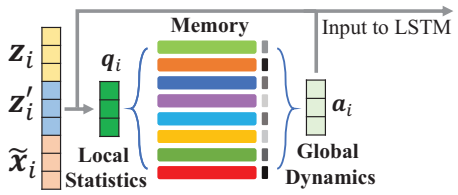

Figure 3: Architecture of the memory module.

used. For missing values, we combine empirical mean with last observation to construct $\mathbf{z}_i$.

However, $\mathbf{z}_i$ only takes data points observed before the i-th time interval into consideration. Similar local statistics can also be extracted from time interval $i+1$ to n. This is achieved by first reverse $\mathbf{X}$ and $\mathbf{M}$ on the temporal dimension, then extracting local statistics following the same definition of $\mathbf{z}_i$ on time interval 1 to $n-i+1$. We use $\mathbf{z}'_i$ to denote local statistics from time interval $i+1$ to n. As shown in Figure 2, $\mathbf{z}_i$ and $\mathbf{z}'_i$ are forward and backward local statistic features, respectively.

In addition to the forward and backward local statistic features, the LSTM naturally provides estimations for missing variables. Specifically, we follow Equation 1 and use the output of LSTM at the previous time step $i-1$ as another local statistic from a model view:

$$\tilde{\mathbf{x}}_i = \mathbf{h}_{i-1}\mathbf{U} + \mathbf{b} \tag{5}$$

Modeling Global Dynamics The above imputations $\mathbf{z}_i$, $\mathbf{z}'_i$, and $\tilde{\mathbf{x}}_i$ can fed directly in LSTM to train a MTS forecasting model (Che et al. 2018). However, such an approach is sub-optimal. $\mathbf{z}_i$ and $\mathbf{z}'_i$ become less trustful as the missing ratio raises up. Besides, purely relying on local statistics ignores global temporal dynamics from exogenous sequences, which potentially benefit the estimation of missing values. It is likely to capture time series snippets/patterns from other sequences that are temporally similar (e.g., periodicity) to the contextual of a missing value. For example, to impute one missing data point in a trajectory, similar time series snippet may be found from other trajectories.

However, capturing such global temporal dynamics to find informative temporal patterns for missing values is very challenging. Simply comparing with all potential snippets to find similar ones is impractical due to high computational costs. Recently, memory network (Weston, Chopra, and Bordes 2014) has shown promising results in capturing patterns for sequential data (Sukhbaatar et al. 2015; Chang et al. 2018). Generally, a memory network initializes a memory component to store feature representations that optimized explicitly on the whole dataset. Those stored representations can be retrieved and utilized for specific tasks (Tang et al. 2017; Kumar et al. 2016). We design a memory module to capture global temporal dynamics explicitly, as shown in Figure 3. We assume there are L temporal patterns existing in the dataset (L is a hyperparameter), and initialize a parameterized memory $\mathcal{G} \in \mathbb{R}^{L \times d^{\mathcal{G}}}$, where $d^{\mathcal{G}}$ is the dimension of pattern representation. The memory $\mathcal{G}$ is updated jointly with the LSTM. We utilize local statistics as keys to query the memory module because they can represent the uniqueness of variables. Specifically, queries to the memory module are constructed as follows:

$$\mathbf{q}_i = \mathbf{W}_q[\mathbf{z}_i || \mathbf{z}'_i || \tilde{\mathbf{x}}_i] + \mathbf{b}_q, \tag{6}$$

where $||$ denotes concatenation on column. $\mathbf{W}_q$ and $\mathbf{B}_q$ are parameters. Then we calculate the similarity between $\mathbf{q}_i$ and the memory component $\mathcal{G}$:

$$\mathbf{s}_i = \text{Softmax}(\mathcal{G} \cdot \mathbf{q}_i). \tag{7}$$

The similarity scores measure the importance of each temporal patterns in the memory. Any pattern with a higher attention score is more similar to the context of targeting missing value. The representation vector of $\mathbf{x}_i$ that preserves global temporal dynamics is then constructed from the weighted sum of all temporal patterns in $\mathcal{G}$:

$$\mathbf{a}_i = \sum_{l=1}^{L} s_i^l \mathcal{G}(l), \tag{8}$$

where $\mathcal{G}(l)$ represents the l-th row of the memory component. Besides, since variables at the same time interval interact with each other in Equation 6, $\mathbf{a}_i$ also preserves inner correlations of variables at the same time interval. We combine local statistic features and global representations to construct the input of LSTM. Specifically, $\mathbf{z}_i$, $\mathbf{z}'_i$, $\tilde{\mathbf{x}}_i$, and $\mathbf{a}_i$ are averaged as the input at time interval i. Note that some of the local statistics can become unavailable in some cases. For example, we cannot construct $\mathbf{z}_i$ for the first missing value, and no $\mathbf{z}'_i$ is available for the forecasting stage. We set unavailable local statistics to $\mathbf{0}$. However, we can generate either the forward or the backward local statistic features unless the whole time series is empty. This ensures LGnet is more reliably than those purely using the forward local statistic (Che et al. 2018). The forecasting results $\mathbf{X}_p = [\tilde{\mathbf{x}}_{n+1}, \ldots, \tilde{\mathbf{x}}_{n+k}]$ are generated after n-th iteration of LSTM. The aligned ground truth data for $\mathbf{X}_p$ is denoted as $\hat{\mathbf{X}}_p$, which also contains missing values. Therefore, we incorporate the mask matrix into mean-square-error and propose the following objective function to train LGnet:

$$\min_{\theta} \mathcal{L}_p(\theta) = \frac{1}{N} \sum_{j=1}^{N} \|(\mathbf{X}_p^j - \hat{\mathbf{X}}_p^j) \odot \mathbf{M}_p^j\|_2, \tag{9}$$

where θ are parameters of LGnet, including parameters of the LSTM and the memory component, $\mathbf{M}_p^j$ is the mask matrix of the j-th MTS data sample $\mathbf{X}^j$ over the predicted variables, and $\odot$ is dot-production. Because of the mask matrix, LGnet is optimized over the observed part of $\hat{\mathbf{X}}_p^j$.

Adversarial Training

The objective function of LGnet in Equation 9 only considers available variables. When the missing ratio is relatively high, the proposed objective function becomes inefficient because most values of $\mathbf{M}_p$ is zero when sampling under the same data distribution. The predicted future sequences should also follow the same data distribution of the true MTS data. If we can encourage LGnet to generate more

realistic data distribution, the overall accuracy of MTS forecasting can be improved. To achieve this goal, we introduce adversarial training to control the distribution of generated MTS.

Recently, generative adversarial networks (GANs) (Goodfellow et al. 2014) have been widely applied to various domains (Yu et al. 2017; Sun et al. 2019; Shu et al. 2018). Typical GAN consists of a generator and a discriminator. The discriminator tries to distinguish samples from the generator and those from read data. The generator tries to generate samples that can "fool" the discriminator by modeling data distribution With such a min-max game, the generator can create more realistic samples. This motivates us to adopt adversarial learning to enhance the forecasting. We design a discriminator D, as illustrated in Figure 2. LSTM generates future sequences as the forecasting to "fool" the discriminator D; while D is trained to identify whether the input sequence is fake. Through iterative training, the LSTM is more capable of generating time series that fit the underlying distribution (Goodfellow et al. 2014), which makes the forecasting result more accurate.

Specifically, we adopt W-GAN (Arjovsky, Chintala, and Bottou 2017) and construct a two-layer convolution net as D. Given a MTS s as input, D outputs a real value $D(s)$, which is higher if s is real, and lower if s is "fake". The "fake" multivariate time series of length k' are generated after the forecasting part. Let $\mathcal{X} = \{[\tilde{\mathbf{x}}_{n+k+1}, \ldots, \tilde{\mathbf{x}}_{n+k+k'}]\}$ denote a generated (fake) time series. To compile a "true" dataset that preserves latent data distribution, we sample a subset of complete time series snippets with same length k' from the raw dataset. Let $\mathcal{S}$ denote the sampled subset of time series snippets. Empirically, it is not a difficult task when k' is small (i.e., $k' \leq 5$). The training objective of the discriminator is:

$$\min_{\theta_D} \mathcal{L}_D = \min_{\theta_D} - \mathop{\mathbb{E}}_{s\sim\mathcal{S}} D(s) + \mathop{\mathbb{E}}_{s\sim\mathcal{X}} D(s), \tag{10}$$

where $\sim$ denotes "sampling from", and θ_D is parameters of the discriminator. Generally, time series with a high probability of being a "true" sample will receive a higher score. To generate more realistic sequences, the objective function for the LSTM is defined as:

$$\min_{\theta} \mathcal{L}_a = \min_{\theta} - \mathop{\mathbb{E}}_{s\sim\mathcal{X}} D(s), \tag{11}$$

which aims at faking the discriminator. Note that there is no overlap between the forecasting and the generated part, as we imperially find that adding adversarial loss on the forecasting part may hurt the performance. A potential reason is that the best time series to "fool" the discriminator might not be the most accurate forecasting result. Therefore, we put $\mathcal{L}_a$ on extra generated sequences to achieve the best performance. Luo et al. state a similar conclusion.

Objective Function and Training

We define the overall objective function to learn model parameters θ for an accurate MTS forecasting with adversarial training as follows:

$$\theta, \theta_D = \min_{\theta} \mathcal{L}_p + \lambda \big[\max_{\theta_D} \mathop{\mathbb{E}}_{s\sim\mathcal{S}} D(s) - \mathop{\mathbb{E}}_{s\sim\mathcal{X}} D(s) \big],$$

where λ balances the MTS forecasting part and the adversarial training part.

We use stochastic gradient descent to update model parameters. The discriminator and the LSTM are trained alternatively until converged. We first update θ_D with real MTS snippets and generated ones, then optimize θ for the LSTM and the memory module while fixing θ_D.

Experiment

In this section, we present experiments to evaluate the proposed framework LGnet. Specifically, we aim at answering the following research questions: (i) **RQ1**: Can LGnet improve the accuracy of MTS forecasting with missing values? (ii) **RQ2** How robust is LGnet w.r.t different missing ratios? (iii) **RQ3** How the memory module benefits LGnet? (iv) **RQ4** How adversarial training contributes to LGnet? Next, we start by introducing various experiments on MTS forecasting to answer the above questions.

Datasets

Four large-scale real-world MTS datasets from different domains are selected to validate LGnet: **Beijing Air**[1]: This dataset is introduced by KDD Cup 2018. We extract PM2.5 values from 35 monitoring stations in Beijing, and formulate multivariate time series. The values are reported every hour between 05/01/2014 and 04/30/2015. It has a missing rate of 13% over the temporal dimension. We use past 9-hour observations to train each model, and forecast PM2.5 values for the following 3 hours. **PhysioNet**: PhysioNet (Silva et al. 2012) provides 4000 multivariate clinical time series from intensive care unit (ICU). Each time series records 35 measurements such as glucose and heart-rate from the first 48 hours since the patient entered the hospital. We compile time series from 12 important measurements such as heart-rate and temperature. The missing ratio of PhysioNet is about 78% over the temporal dimension. We use the past 6 observations to forecast values in the coming 3 hours. **Porto Taxi**[2]: This dataset includes approximately one million trajectories for 442 taxis running in the city of Porto during a complete year (from 01/07/2013 to 30/06/2014). Each trajectory contains many GPS coordinates (i.e., longitude and latitude) recorded chronologically. The sampling speed is 15 second per coordinates. We use the past 7 GPS coordinates to forecast the location of future points. **London Weather**[1]: The dataset includes temperature, pressure, humidity, wind direction, and wind speed from 861 regions in London from 01/01/2017 to 03/27/2018. All features are collected hourly. We use the past 5 observations to forecast the coming values.

The first and second datasets naturally contain missing values, which are used as real-world settings to answer the first question. For the rest two datasets, we randomly remove $p\%$ of observed values ($p \in \{10, 30, 50, 70, 90\}$) to study the robustness of LGnet and compared methods.

[1] https://www.kdd.org/kdd2018/kdd-cup

[2] https://www.kaggle.com/c/pkdd-15-predict-taxi-service-trajectory-i

Compared Baselines

We compare LGnet with classical and state-of-the-art baselines, including two non-RNN methods, two time series imputation methods, and two RNN methods:

- **Linear Regression** (LR): Because conventional linear regression model cannot directly handle missing values, we concatenate each MTS with its mask matrix as the input features to train LR for the forecasting task.
- **XGBoost** (Chen and Guestrin 2016): XGBoost is widely used in time series analysis and machine learning fields. We use the same setting of LR to train XGBoost.
- **MICE** (Azur et al. 2011): MICE fills the missing values using multiple imputations with chained equations. We first apply MICE to impute miss values. We then train LSTM for the forecasting task.
- **GRUI** (Luo et al. 2018): GRUI leverages GAN and RNN for time series imputation. Similar to MICE, we first train GRUI for time series imputation. Then we train LSTM on imputed data as the forecasting model.
- **GRU-D** (Che et al. 2018): GRU-D combines statistical features and linear decay in RNN to tackle missing variables in time series. It is proposed for multivariate time series forecasting task.
- **BRITS** (Cao et al. 2018): BRITS designs bi-direction recurrent neural architecture for time series imputation and forecasting. It models missing patterns explicitly and improves forecasting accuracy.

Experimental Settings

We normalize each dataset and ensure all time series variables have the same scale (i.e., mean and variance) on each dataset so that their averaged results are comparable. For each dataset, we randomly select 70% of MTS data for training, 10% for validation to tune hyperparameters, and the remaining 20% for testing. We set the dimension of the hidden unit to 32 for the LSTM. We select L from $\{8, 16, 32, 64\}$ to create the memory component according to the performance on validation sets. The dimension of each memory slot is 128. We tune λ that balance MTS forecasting and adversarial training on validation sets for the best performance. The discriminator contains two convolutional layers following by two fully-connected layers. 3×3 kernels are used for both convolutional layers. The channel sizes are 64 and 128 for the first and second convolutional layer, respectively. The dimensions of fully connected layers are 1024 and 1.

Two widely used evaluation metrics, i.e., *root mean square error* (RMSE) and *mean absolute error* (MAE), are adopted. Since different variables have different scales, we report the RMSE and MAE on their normalized values. *The smaller RMSE and MAE are, the better the performance is.*

Performance Comparisons

To answer **RQ1**, we compare LGnet with baselines on Beijing Air and PhysioNet, where missing values naturally exist. We report the performance on the two datasets for $k = 1, 2, 3$ (forecasting horizon) in Table 1, and make the following observations: ***(i)*** LGnet outperforms all the baseline methods for the majority of the cases, which shows the effectiveness of the memory module and adversarial learning for multivariate time series forecasting with missing values. The memory module explores global temporal dynamics and generates appropriate estimations for missing values; ***(ii)*** when k increases, i.e., when forecasting far future values, the performance of all the methods decreases, which is reasonable because it's more difficult to forecast far future values than near ones. However, LGnet still significantly outperform the compared methods, which is because we adopt adversarial training on the predicted sequences to make the forecasting more realistic; ***(iii)*** In addition, the performance improvement of LGnet is much more significant on PhysioNet than Beijing Air. Compared with Beijing Air, PhysioNet has a higher missing ratio, which challenges the baseline methods; while LGnet can still handle such high missing ratio, which further implies the effectiveness of LGnet by designing memory network and adopting adversarial training.

Robustness of LGnet

Real-world applications could encounter various data missing conditions. It is interesting to understand the robustness of LGnet under different missing ratios. To this end, we design experiments on two complete MTS datasets, including Porto taxi and London weather. In particular, for each dataset, we randomly drop $p\%$ of observed values to generate synthetic missing condition and we alter p from $\{10, 30, 50, 70, 90\}$. We train LGnet and compared methods to forecast the next observation of all variables (i.e., $k = 1$). The performance of LGnet and all compared methods in terms of RMSE and MAE varying p is reported in Figure 4. Clearly, the forecasting error of non-RNN methods raises dramatically as p increasing, because they fail to model missing temporal patterns for the forecasting. GRU-D and BRITS explicitly handle missing values and achieve lower errors compared with LR and XGBoost. However, they fail to maintain accurate forecasting when the missing ratio is high. LGnet achieves the highest accuracy even if the data is extremely sparse (e.g., $p = 90$), which illustrate the effectiveness of the memory module and the adversarial schema. The global temporal patterns in memory module help LGnet perform well as the missing ratio increasing. Extra guidance from the discriminator improves the capability of LSTM in modeling the global distribution of MTS.

Ablation Study

Memory Module Analysis We analyze the contribution of the memory module. We create an ablation named LGnet$_{adv}$ by removing the memory module from LGnet, and use $\mathbf{q}_i$ as the input for LSTM. The performance of LGnet$_{adv}$ is reported in Table 2. Obviously, LGnet significantly out-performs LGnet$_{adv}$, indicating that modeling global temporal dynamics with the memory module benefits the forecasting. Moreover, the performance improvement of LGnet over LGnet$_{adv}$ is relatively bigger as the missing ratio raises up. This is because local statistic features are less reliable with a high missing ratio. Under such circumstances,

Table 1: MTS forecasting performances on Beijing Air and PhysioNet.

Datasets	k	1		2		3	
	Metric	RMSE	MAE	RMSE	MAE	RMSE	MAE
Beijing Air	LR	0.0398±0.0018	0.0261±0.0003	0.0550±0.0014	0.0371±0.0001	0.0661±0.0015	0.0454±0.0007
	XGB	**0.0389±0.0004**	**0.0229±0.0017**	0.0542±0.0010	0.0376±0.0015	0.0663±0.0009	0.0406±0.0013
	MICE	0.0646±0.0001	0.0417±0.0004	0.0703±0.0012	0.0452±0.0017	0.0766±0.0018	0.0493±0.0008
	GRUI	0.0601±0.0012	0.0387±0.0009	0.0667±0.0001	0.0433±0.0001	0.0754±0.0007	0.0502±0.0012
	GRU-D	0.0459±0.0014	0.0305±0.0001	0.0554±0.0020	0.0366±0.0011	0.0649±0.0003	0.0432±0.0001
	BRITS	0.0501±0.0001	0.0335±0.0012	0.0570±0.0003	0.0384±0.0008	0.0707±0.0017	0.0493±0.0002
	LGnet	0.0451±0.0010	0.0300±0.0009	**0.0519±0.0008**	**0.0332±0.0006**	**0.0597±0.0002**	**0.0373±0.0006**
PhysioNet	LR	0.2401±0.0003	0.1839±0.0006	0.2520±0.0009	0.1948±0.0007	0.2622±0.0001	0.1997±0.0004
	XGB	0.2308±0.0004	0.1753±0.0006	0.2481±0.0005	0.1913±0.0019	0.2598±0.0017	0.1972±0.0001
	MICE	0.1113±0.0000	0.0783±0.0011	0.1148±0.0008	0.0793±0.0006	0.1116±0.0018	0.0789±0.0019
	GRUI	0.1142±0.0017	0.0776±0.0012	0.1176±0.0001	0.0812±0.0008	0.1270±0.0005	0.0813±0.0016
	GRU-D	0.1125±0.0013	0.0998±0.0003	0.1202±0.0011	0.0796±0.0014	0.1348±0.0002	0.0971±0.0011
	BRITS	0.1082±0.0002	0.0720±0.0010	0.1158±0.0012	0.0734±0.0018	0.1158±0.0001	0.0785±0.0003
	LGnet	**0.1021±0.0004**	**0.0706±0.0002**	**0.0998±0.0002**	**0.0713±0.0002**	**0.1080±0.0005**	**0.0762±0.0007**

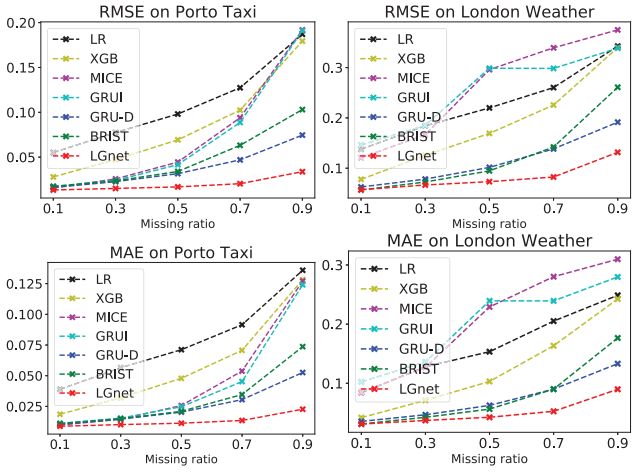

Figure 4: MTS forecasting performance on Porto taxi and London Weather with varying missing ratios.

Table 2: MTS forecasting performance of variants.

p	Porto Taxi			London Weather		
	LGnet$_{adv}$	LGnet$_{mem}$	LGnet	LGnet$_{adv}$	LGnet$_{mem}$	LGnet
0.1	0.0157	**0.0132**	0.0137	0.0627	0.0595	**0.0573**
0.3	0.0241	0.0153	**0.0151**	0.0773	0.0745	**0.0666**
0.5	0.0303	0.0170	**0.0168**	0.1020	0.0816	**0.0732**
0.7	0.0451	0.0208	**0.0204**	0.1287	0.0852	**0.0825**
0.9	0.0710	0.0348	**0.0337**	0.2015	0.1327	**0.1316**

it is vital to leverage global patterns stored in the memory component as support to estimate missing values.

Adversarial Schema Analysis We further study the effectiveness of adversarial training. The parameter λ balances the weight between the forecasting loss and the adversarial part. A variant of LGnet without the adversarial training (i.e., $\lambda = 0$) is denoted as LGnet$_{mem}$, and its performance is reported in Table 2. Clearly, the adversarial training contributes a lot to the forecasting, reducing RMSE by 2% – 10% under different circumstances. More concretely, more significant error reductions occur on London weather dataset compared with Proto taxi dataset. One possible reason is that MTS from London weather dataset contain more variables and have a better description of the realistic data distribution. Besides, improvements from incorporating the discriminator are relatively greater when the missing ratio increases. This is because the original MTS forecasting objective is less efficient with a high missing ratio, as it only relies on observed parts of the time series. In conclusion, it is beneficial to introduce adversarial training for LGnet.

Table 3: Analysis of hyper-parameter λ.

λ	0	0.1	1	10	100
RMSE	0.0483	0.0451	0.0471	0.0522	0.0518
MAPE	0.0322	0.0300	0.0315	0.0360	0.0354

Hyper-parameter Analysis

We investigate the sensitivity of λ, which balances the forecasting loss and the adversarial training part. Generally, More emphasis is put to the forecasting part when λ is closer to 0. We alter the value of λ among $\{0, 0.1, 1, 10, 100\}$ and report the performance of LGnet on Beijing Air dataset with $k = 1$. As shown in Table 3, the forecasting accuracy of LGnet first increases as λ becomes larger. However, extremely large values of λ result in low performances.

Conclusion

In this paper, we investigate a novel problem of exploring local and global temporal dynamics for MTS forecasting with missing values. We propose a new framework LGnet, which adopts memory network to capture global temporal patterns using local statistics as keys. To make the generated MTS more realistic, we further adopt adversarial training to enhance the modeling of global temporal data distribution. Experimental results on four large-scale real-world datasets show the efficacy of LGnet.

Acknowledgments

This material is based upon work supported by, or in part by, the National Science Foundation (NSF) under grant #1909702.

References

Arjovsky, M.; Chintala, S.; and Bottou, L. 2017. Wasserstein gan. *arXiv:1701.07875*.

Azur, M. J.; Stuart, E. A.; Frangakis, C.; and Leaf, P. J. 2011. Multiple imputation by chained equations: what is it and how does it work? *International journal of methods in psychiatric research* 20(1):40–49.

Bengio, S.; Vinyals, O.; Jaitly, N.; and Shazeer, N. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *NeurIPS*, 1171–1179.

Box, G. E.; Jenkins, G. M.; Reinsel, G. C.; and Ljung, G. M. 2015. *Time series analysis: forecasting and control*. John Wiley & Sons.

Cao, W.; Wang, D.; Li, J.; Zhou, H.; Li, L.; and Li, Y. 2018. Brits: Bidirectional recurrent imputation for time series. *arXiv:1805.10572*.

Chang, Y.-Y.; Sun, F.-Y.; Wu, Y.-H.; and Lin, S.-D. 2018. A memory-network based solution for multivariate time-series forecasting. *arXiv:1809.02105*.

Che, Z.; Purushotham, S.; Cho, K.; Sontag, D.; and Liu, Y. 2018. Recurrent neural networks for multivariate time series with missing values. *Scientific reports* 8(1):6085.

Chen, T., and Guestrin, C. 2016. Xgboost: A scalable tree boosting system. In *KDD*, 785–794. ACM.

Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv:1406.1078*.

Chung, J.; Gulcehre, C.; Cho, K.; and Bengio, Y. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv:1412.3555*.

Friedman, J.; Hastie, T.; and Tibshirani, R. 2001. *The elements of statistical learning*, volume 1. Springer series in statistics New York, NY, USA:.

García-Laencina, P. J.; Sancho-Gómez, J.-L.; and Figueiras-Vidal, A. R. 2010. Pattern classification with missing data: a review. *Neural Computing and Applications* 19(2):263–282.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. In *NeurIPS*, 2672–2680.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

King, G.; Honaker, J.; Joseph, A.; and Scheve, K. 1998. List-wise deletion is evil: what to do about missing data in political science. In *APSA*.

Kumar, A.; Irsoy, O.; Ondruska, P.; Iyyer, M.; Bradbury, J.; Gulrajani, I.; Zhong, V.; Paulus, R.; and Socher, R. 2016. Ask me anything: Dynamic memory networks for natural language processing. In *ICML*, 1378–1387.

Lai, G.; Chang, W.-C.; Yang, Y.; and Liu, H. 2018. Modeling long-and short-term temporal patterns with deep neural networks. In *SIGIR*. ACM.

Luo, Y.; Cai, X.; Zhang, Y.; Xu, J.; et al. 2018. Multivariate time series imputation with generative adversarial networks. In *NeurIPS*, 1603–1614.

Marsh, H. W. 1998. Pairwise deletion for missing data in structural equation models: Nonpositive definite matrices, parameter estimates, goodness of fit, and adjusted sample sizes. *Structural Equation Modeling: A Multidisciplinary Journal* 5(1).

Nelwamondo, F. V.; Mohamed, S.; and Marwala, T. 2007. Missing data: A comparison of neural network and expectation maximization techniques. *Current Science*.

Qin, Y.; Song, D.; Chen, H.; Cheng, W.; Jiang, G.; and Cottrell, G. 2017. A dual-stage attention-based recurrent neural network for time series prediction. *arXiv:1704.02971*.

Shu, K.; Wang, S.; Le, T.; Lee, D.; and Liu, H. 2018. Deep headline generation for clickbait detection. In *ICDM*. IEEE.

Silva, I.; Moody, G.; Scott, D. J.; Celi, L. A.; and Mark, R. G. 2012. Predicting in-hospital mortality of icu patients: The physionet/computing in cardiology challenge 2012. *Computing in cardiology* 39:245.

Smola, A. J., and Schölkopf, B. 2004. A tutorial on support vector regression. *Statistics and computing* 14(3):199–222.

Sukhbaatar, S.; Weston, J.; Fergus, R.; et al. 2015. End-to-end memory networks. In *NeurIPS*, 2440–2448.

Sun, Y.; Wang, S.; Hsieh, T.-Y.; Tang, X.; and Honavar, V. 2019. Megan: a generative adversarial network for multi-view network embedding. *arXiv preprint arXiv:1909.01084*.

Tang, J.; Wang, Y.; Zheng, K.; and Mei, Q. 2017. End-to-end learning for short text expansion. In *KDD*, 1105–1113. ACM.

Tang, X.; Gong, B.; Yu, Y.; Yao, H.; Li, Y.; Xie, H.; and Wang, X. 2019. Joint modeling of dense and incomplete trajectories for citywide traffic volume inference. In *WWW*. ACM.

Wells, B. J.; Chagin, K. M.; Nowacki, A. S.; and Kattan, M. W. 2013. Strategies for handling missing data in electronic health record derived data. *Egems* 1(3).

Weston, J.; Chopra, S.; and Bordes, A. 2014. Memory networks. *arXiv:1410.3916*.

Wu, X.; Shi, B.; Dong, Y.; Huang, C.; Faust, L.; and Chawla, N. V. 2018. Restful: Resolution-aware forecasting of behavioral time series data. In *CIKM*, 1073–1082. ACM.

Wu, X.; Shi, B.; Dong, Y.; Huang, C.; and Chawla, N. V. 2019. Neural tensor factorization for temporal interaction learning. In *WSDM*, 537–545. ACM.

Xingjian, S.; Chen, Z.; Wang, H.; Yeung, D.-Y.; Wong, W.-K.; and Woo, W.-c. 2015. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *NeurIPS*, 802–810.

Yao, H.; Wu, F.; Ke, J.; Tang, X.; Jia, Y.; Lu, S.; Gong, P.; Ye, J.; and Li, Z. 2018. Deep multi-view spatial-temporal network for taxi demand prediction. In *AAAI*.

Yao, H.; Liu, Y.; Wei, Y.; Tang, X.; and Li, Z. 2019a. Learning from multiple cities: A meta-learning approach for spatial-temporal prediction. *WWW*.

Yao, H.; Tang, X.; Wei, H.; Zheng, G.; and Li, Z. 2019b. Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction. In *AAAI*.

Yi, X.; Zheng, Y.; Zhang, J.; and Li, T. 2016. St-mvl: filling missing values in geo-sensory time series data.

Yoon, J.; Zame, W. R.; and van der Schaar, M. 2017. Multi-directional recurrent neural networks : A novel method for estimating missing data.

Yu, L.; Zhang, W.; Wang, J.; and Yu, Y. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *AAAI*.