Proceedings of Machine Learning Research 95:248-263, 2018 ACML 2018

End-to-End Time Series Imputation via Residual Short Paths

Lifeng Shen SCUTERLIFENG@FOXMAIL.COM
School of Computer Science and Engineering, South China University of Technology, Guangzhou

Qianli Ma* QIANLIMA@SCUT.EDU.CN
School of Computer Science and Engineering, South China University of Technology, Guangzhou
Guangdong Key Laboratory of Big Data Analysis and Processing, Guangzhou

Sen Li AWSLEE@FOXMAIL.COM
School of Computer Science and Engineering, South China University of Technology, Guangzhou

Editors: Jun Zhu and Ichiro Takeuchi

Abstract

Time series imputation (replacing missing data) plays an important role in time series analysis due to missing values in real world data. How to recover missing values and model the underlying dynamic dependencies from incomplete time series remains a challenge. A recent work has found that residual networks help build very deep networks by leveraging short paths due to skip connections (Veit et al., 2016). Inspired by this, we observe that these short paths can model underlying correlations between missing items and their previous non-missing observations in a graph-like way. Hence, we propose an end-to-end imputation network with residual short paths, called Residual IMPutation LSTM (RIMP-LSTM), a flexible combination of residual short paths with graph-based temporal dependencies. We construct a residual sum unit (RSU), which enables RIMP-LSTM to make full use of previous revealed information to model incomplete time series and reduce the negative impact of missing values. Moreover, a switch unit is designed to detect the missing values and a new loss function is then developed to train our model with time series in the presence of missing values in an end-to-end way, which also allows simultaneous imputation and prediction. Extensive empirical comparisons with other competitive imputation approaches over several synthetic and real world time series with various rates of missing data verify the superiority of our model.

Keywords: Time series imputation, LSTMs, end to end learning

1. Introduction

Time series is an important form of data in practical applications, including geo-sensory (Yi et al., 2016), financial markets (Qin et al., 2017) and action recognition (Ma et al., 2017). However, these practical time series data inevitably contain missing values due to noise or malfunctioning sensors. A survey of some incomplete time series in the UCI ML Repository is shown in Table 1. Traditional methods require complete data and have to impute missing values before prediction. This can be cumbersome and is not an end-to-end solution. Furthermore, missing values make any kind of inference more difficult (Rubin, 1976), such

* Corresponding Author

as prediction (Yu et al., 2016) and classification (Keogh and Pazzani, 1998). Time series imputation is a very challenging task, since it needs to model temporal dependencies from incomplete data. Especially in the case of missing a continuous chunk of data, a long-term and robust memory with history information will be required in the model. How to model these dependencies from incomplete time series is an important issue.

Table 1: Statistical results of incomplete time series in UCI.

Time Series	#Total	#Incomplete	Rate(%)
Dodgers Loop Sensor	50400	2903	6
Heterogeneity Activity Recognition	33741500	4643613	14
Ozone Level Detection	2536	688	27
SML2010	4137	2152	52
PM2.5 Data of Five Chinese Cities	262920	162236	62
OPP-Activity Recognition	869387	639853	74

Using a graph to describe the temporal dependencies between the missing item and its previously revealed points is a very explicit and natural strategy for time series imputation. A recent representative work is Temporal Regularized Matrix Factorization (TRMF) (Yu et al., 2016), in which a graph-based temporal regularization was introduced to model temporal dependencies. These dependencies are simplified to an autoregressive structure illustrated in Fig. 1. For example, assuming the missing variable at time step t is x_t, the autoregressive dependencies can be formulated as $x_t = \sum_{l \in \mathcal{L}} w^{(l)} x_{t-l}$, where $\mathcal{L}$ denotes a lag set. Although TRMF demonstrated the effectiveness of graph-based modeling in time series imputation, the graph-based dependency structure (such as $\mathcal{L}$) still depends on manual design and cannot automatically capture complex dynamic correlations in an end-to-end way.

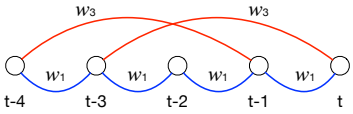
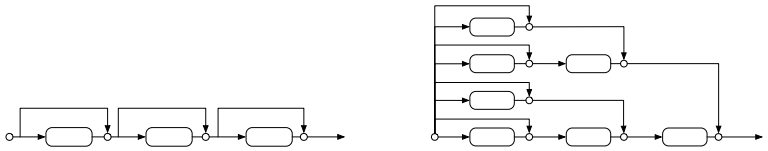


Figure 1: Graph-based regularization for temporal dependencies shown in Yu et al. (2016).

In the past decades, the deep learning community has developed powerful methods, such as LSTM units (Hochreiter et al., 1997), for learning temporal dependencies within data. However, the standard LSTM is not designed to fill in missing data; rather, it learns to remember data it needs for prediction. Missing items will have negative impact on the memory states in an LSTM. If we arbitrarily replace a missing item with the mean value or the previously revealed item, there is no mechanism for the LSTM to recognize this as not being real data, which could mislead the LSTM in a prediction task.

Recently, Veit et al. (2016) performed an enlightening analysis of residual networks (He et al., 2016). They argued that a residual network can be regarded as an ensemble of relatively shallow networks. As seen in Fig. 2, adapted from their paper, a 3-block residual network is a collection of $2^3 = 8$ short paths with different lengths. In their view, with the structure of short paths, the flow of the gradient information can be efficiently propagated in this corresponding shallow network. This is the main reason why He et al.'s residual



(a) 3-block residual network (b) unraveled view of (a)

Figure 2: As analyzed in Veit et al. (2016), residual networks (a) enable very deep networks by leveraging the short paths shown in (b). There are $2^3 = 8$ short paths in a 3-block residual network.

networks work so well. This work inspired us to use this structure of short paths to model time series with missing values. It has a close connection to the graph-based models in the aforementioned TRMF. With this structure, we can model longer-term dependencies on previous data using shorter paths. To the best of our knowledge, this residual structure has not been previously considered as an approach to the imputation problem in time series prediction.

In this paper we address the problem of incomplete time series imputation from the viewpoint of modeling graph dependencies and propose a novel end-to-end imputation network called the Residual IMPutation LSTM (RIMP-LSTM). We introduce the residual-short-path structure into LSTM and construct a Residual Sum Unit (RSU) to fuse the residual information flows. In particular, the role of RSU at each time step is to fuse the LSTM's hidden states and the RSUs from previous time steps. From the point of view of graph dependencies, the current value of the RSU directly integrates the information from its historical states via residual paths, much as a weighted graph does. In this way, it can take full advantage of the previous observed information and reduce the negative impact of missing values. Moreover, we propose a switch unit to detect the missing values. If the next input value is known, the switch will be off, and the output of RSU at current time will be trained to approximate the known next input value; while if the next input is missing, the switch will be on and the missing value is imputed by the output of RSU. A new loss function is then developed to achieve switching the states adaptively. Finally, all the parameters are learned via the standard BPTT algorithm. In this way, the RIMP-LSTM can be trained with incomplete time series, simultaneously imputing the missing values and conducting time series prediction. Since RIMP-LSTM is a flexible framework based on RNNs, the LSTM can be replaced with any other type of RNN.

Our contributions can be summarized as follows:

- We propose an end-to-end Residual IMPutation LSTM (RIMP-LSTM) to address the problem of time series imputation.
- This model unifies the idea of residual short paths with the method of graph-based modeling of temporal dependencies.
- RIMP-LSTM can be trained with incomplete time series in an end-to-end way, simultaneously achieving imputation and prediction.
- RIMP-LSTM is evaluated by experiments on several synthetic and real-world time series with different levels of missing values. Results show that our model obtains state of the art imputation and prediction performance.

The remainder of this paper is organized as follows. Section 2 discusses related work on time series imputation. Section 3 presents our method formally. Section 4 reports the detailed experimental settings and results. We draw our conclusion in Section 5.

2. Related Work

The demand for imputing missing data arises in many areas, giving rise to a lot of relevant studies. Traditional time series imputation methods, such as interpolation, splines and moving averages (MA), are commonly used to impute missing values in time series. All of them estimate the missing value from immediately preceding or succeeding values. Hence, they will achieve poor performance when encountering a large number of missing values. The Expectation Maximization (EM) algorithm (Dempster et al., 1977) is also widely applied in dealing with missing values in time series. Sinopoli et al. (2004) combine it with a Kalman filter. Oba et al. (2003) combine it with PCA and variational Bayes methods. Both of them reconstruct the missing values by iterative EM steps over the available values.

Similar to the Kalman filter, Li et al. (2009) propose Dynammo, using a sequence of latent variables to model the underlying linear dynamical system and hidden patterns of the observation sequences for multivariate time series imputation. White et al. (2011) propose MICE, a sequential linear regression multivariate imputation method, in which the variable with missing value is regressed on other available variables and draws from the corresponding posterior predictive distribution to replace the missing value. Anava et al. (2015) use an autoregressive (AR) model to address online time series prediction with missing values. In particular, they assume that the missing item can be represented as a recursive autoregressive form of its previous non-missing points and missing ones. However, all of them assume the time series has underlying linear dynamics, while non-linear dynamics is more common in time series.

Recently, modeling temporal dependencies with graph-based regularization provides a new insight into time series imputation. The aforementioned TRMF (Yu et al., 2016) employed low-rank matrix factorization to deal with the correlation among multiple variables and further generalized the AR model as a weighted dependency graph-based regularizer to learn the temporal dependencies between non-missing observations and missing values at different time steps, which allows it simultaneous imputation and prediction. However, TRMF is limited to linear dependency with manually-designed structures.

Recurrent neural networks (RNNs) are suitable for modeling non-linear temporal dependencies for both univariate and multivariate time series. However, conventional RNNs are based on sequential memory and can not be trained in the presence of missing values. Although Brakel et al. (2013) presented a training strategy for time series imputation, their method still required the guidance of ground truth in the training stage. Recently, Lipton et al. (2016) used an RNN with an added binary variable to indicate whether the value is missing or not, and set the missing value to zero when it is missing. This allowed them to train a recurrent network with missing data, which was especially important in their medical data domain. However, their use case was not filling in the missing variables. In medical data, lack of data is actually useful information (e.g., that a test was not run). More recently, Che et al. (2018) further combined the indicator based approach with a decay mechanism for clinical data classification with missing values. However, as they claimed, their mod-

el is not explicitly designed for filling in the missing values in the data. It requires the missing patterns are informative (not missing-at-random), otherwise, it may gain limited improvements or even fail.

Our method is related to TRMF with a temporal dependency graph and LSTM units. We address time series imputation from the viewpoint of modeling graph dependencies with weighted residual short paths. Moreover, our model is an end-to-end imputation network adopting a novel learning mechanism, which takes full advantage of the previous observed information of incomplete time series and reduces the negative impact of missing values to the memory in LSTM.

3. Proposed Methods

3.1. Brief Review of LSTM Networks

Given $\mathbf{x} = \{\mathbf{x}_t\}$, a T-length time series without any missing values, an LSTM network can encode this time series as a hidden sequence $\mathbf{h} = \{\mathbf{h}_t\}$, where input at each time is a n-dimensional vector $\mathbf{x}_t \in \mathbb{R}^n$, and the corresponding hidden output is a m-dimensional vector $\mathbf{h}_t \in \mathbb{R}^m$, $t = 1, 2, \ldots, T$. As a variant of an RNN, the LSTM neurons consist four special types: a memory cell $\mathbf{c}_t \in \mathbb{R}^m$, an input gate $\mathbf{i}_t \in \mathbb{R}^m$, a forget gate $\mathbf{f}_t \in \mathbb{R}^m$, and an output gate $\mathbf{o}_t \in \mathbb{R}^m$. With these four units, we can formulate the basic LSTM as follows:

$$
\begin{aligned}
\widetilde{\mathbf{c}}_t &= \tanh(\mathbf{W}_c[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_c) \\
\mathbf{i}_t &= \sigma(\mathbf{W}_i[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_i) \\
\mathbf{f}_t &= \sigma(\mathbf{W}_f[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_f) \\
\mathbf{o}_t &= \sigma(\mathbf{W}_o[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_o) \\
\mathbf{c}_t &= \mathbf{f}_t \otimes \mathbf{c}_{t-1} + \mathbf{i}_t \otimes \widetilde{\mathbf{c}}_t \\
\mathbf{h}_t &= \mathbf{o}_t \otimes \tanh(\mathbf{c}_t)
\end{aligned} \tag{1}
$$

where $\sigma(\cdot)$ denotes the sigmoid function $\sigma(x) = 1/(1 + exp(-x))$, the weight matrices $\mathbf{W}_c$, $\mathbf{W}_i$, $\mathbf{W}_f$, $\mathbf{W}_o$ are learned parameters that control the memory cells and gates. The operator $\otimes$ denotes the element-wise product. We can simplify the notation of an LSTM as a function $\mathcal{F}_{LSTM}$:

$$
\mathbf{h}_t = \mathcal{F}_{LSTM}(\mathbf{h}_{t-1}, \mathbf{x}_t, \mathbf{c}_{t-1}) \tag{2}
$$

Note that we ignore the other output $\mathbf{c}_t$ in (2). This form is convenient for explaining our model later.

3.2. Proposed RIMP-LSTM

Denote a T-length time series with missing values as $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$, where $\mathbf{x}_t \in \mathbb{R}^n$ $(t = 1, 2, \ldots, T)$ is a n-dimensional vector at time t. The time series is allowed to be missing some x_t's or some components of the x_t's.

Our RIMP-LSTM architecture is illustrated in Fig. 3.

We introduce a unit called a Residual Sum Unit (RSU) into the LSTM network (colored green in Fig. 3). The value of the RSU at time step t, $\mathbf{r}_t \in \mathbb{R}^m$, is called the residual sum,

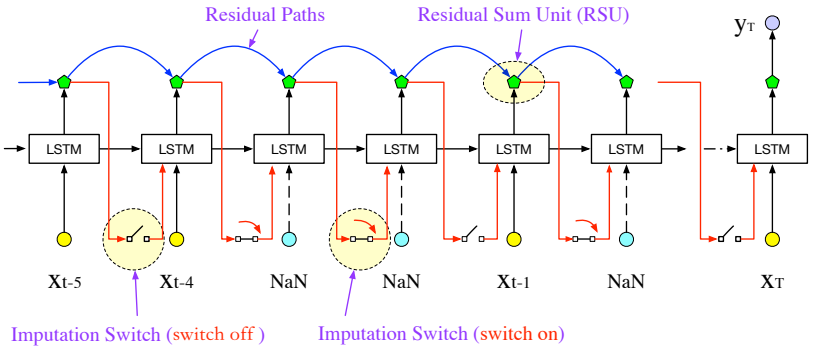


Figure 3: The RIMP-LSTM architecture. We use green to denote Residual Sum Units (RSUs), yellow for observed inputs, blue for missing values and violet for the task-related output.

and is given by:

$$\mathbf{r}_t = \mathcal{F}_{RSU}(\mathbf{h}_t, \mathbf{r}_{t-1}) = \begin{cases} f(\mathbf{h}_t) & t = 1 \\ f(\mathbf{h}_t + g(\mathbf{W}_r \mathbf{r}_{t-1})) & t \geq 1 \end{cases} \tag{3}$$

where g and f are identity functions (used for generality), $\mathbf{h}_t$ denotes the hidden state of LSTM at the time step t, and $\mathbf{W}_r \in \mathbb{R}^{m \times m}$ is a learned weight matrix, which allows the RSU to have the same dimensionality as the hidden states.

RIMP-LSTM approximates the next input value with a linear transformation of the residual sum:

$$\mathbf{z}_t = \mathbf{W}_{ro} \mathbf{r}_t \tag{4}$$

where $\mathbf{W}_{ro} \in \mathbb{R}^{n \times m}$ is a learned transformation matrix.

The imputation switches in Fig. 3 are used for detecting and filling in the missing items. Unifying the next revealed input and the imputed one as $\mathbf{u}_t$, then we have

$$\mathbf{u}_t = (\mathbf{x}_t \otimes \mathbb{I}\{\mathbf{x}_t \ is \ revealed\}) \oplus (\mathbf{z}_{t-1} \otimes \mathbb{I}\{\mathbf{x}_t \ is \ missing\}) \tag{5}$$

where $\otimes$ and $\oplus$ denote element-wise product and addition, respectively, and $\mathbb{I}\{\mathbf{x}_t\}$ denotes the element-wise indicator function, which returns an n-dimensional binary vector to indicate the missing attributes. Hence, we will input $\mathbf{x}_t$ when $\mathbf{x}_t$ is revealed, and input $\mathbf{z}_{t-1}$ when $\mathbf{x}_t$ is missing. In this way, we can formulate the updating process of LSTM hidden states by $\mathbf{h}_t = \mathcal{F}_{LSTM}(\mathbf{h}_{t-1}, \mathbf{u}_t, \mathbf{c}_{t-1})$, where $\mathbf{c}_{t-1} \in \mathbb{R}^m$ denotes the LSTM's memory cell.

Our training process runs under two cases: approximation and imputation. As shown in Fig. 3, it uses the approximation process when the switch is turned off, and conducts the imputation when the switch is turned on. If the next input $\mathbf{x}_t$ is revealed, we train the output $\mathbf{z}_{t-1}$ of the RSU to approximate $\mathbf{x}_t$, aiming to model temporal dependencies between $\mathbf{x}_t$ (including the case of missing terms) and its previous information. When $\mathbf{x}_t$ is missing, we directly copy $\mathbf{z}_{t-1}$ to $\mathbf{x}_t$. Unifying these two cases, we can formulate a Residual Imputation Loss (RIMP Loss) $\mathcal{L}_t(\mathbf{z}_{t-1}, \mathbf{x}_t)$ at time t as

$$\mathcal{L}_t = \|(\mathbf{z}_{t-1} - \mathbf{x}_t) \otimes \mathbb{I}\{\mathbf{x}_t \ is \ revealed\}\|_2^2 \tag{6}$$

where $\mathbb{I}\{\mathbf{x}_t\}$ denotes the element-wise indicator function, $t = 2, 3, \ldots, T$. If we let the superscript k denote the k-th sample of time series collections ($k = 1, 2, \ldots, N$), then we have the overall training loss $\mathcal{L}_{total}$ as

$$\mathcal{L}_{total} = \sum_{k=1}^{N} \{ \underbrace{\sum_{t=2}^{T} \| (\mathbf{z}_{t-1}^{(k)} - \mathbf{x}_t^{(k)}) \otimes \mathbb{I}\{\mathbf{x}_t^{(k)} \ is \ revealed\} \|_2^2}_{\text{Residual Imputation Loss (RIMP Loss)}} + \lambda_{target} \underbrace{\mathcal{L}_{target}(\mathbf{d}^{(k)}, \mathbf{y}_T^{(k)})}_{\text{Task-related Loss}} \} \quad (7)$$

where $\mathbf{d}^{(k)}$ and $\mathbf{y}_T^{(k)}$ denote task-related target and output of k-th sample. The second term of $\mathcal{L}_{total}$ is task-related. For example, in a prediction task, $\mathcal{L}_{target}$ is the square loss. Finally, the training of the RIMP-LSTM is the same as other RNNs/LSTMs and uses the BPTT algorithm to learn their parameters.

During the testing stage, the transferred output $\mathbf{z}_{t-1}$ of the RSU fills in the missing value $\mathbf{x}_t$ in an online manner.

3.3. Discussion

RIMP-LSTM combines the merits of graph-based models with explicitly modeled temporal dependencies via weighted residual connection between nodes, with the ones of LSTM that can accumulate historical residual information and learn the underlying patterns of incomplete time series automatically. Compared to other general graph methods (such as Yu et al. (2016), shown in Fig. 1), our RIMP-LSTM has two advantages:

- The temporal dependency graph in RIMP-LSTM considers all direct connections among variables (e.g., given K previous points, the number of residual short paths is 2^K), which avoids handcrafted design of dependency structure.
- These residual short paths in RIMP-LSTM can be automatically learned in an end-to-end way using BPTT, which does not limit the system to some set of user-intuited assumptions, like the dependency length (delay) in autoregression (AR).

RIMP-LSTM thus models temporal dependencies with weighted residual short paths, takes advantage of RSU to accumulate historical residual information, and learns the underlying patterns of time series with missing data in an end-to-end manner.

4. Experimental Evaluations

In this section, we demonstrate the effectiveness of our RIMP-LSTM for time series imputation of different-level missing values. We consider two kinds of time series: univariate time series and multivariate ones. Univariate time series imputation depends on the temporal dependencies between the missing item and its previous history (or neighbors) only, due to the data are one dimensional points at each time step. On the other hand, multivariate time series contain multiple variables and each variable has a corresponding univariate time series. In this case, imputation can use the correlations among variables, since some variables at specific time steps may be missing, while other non-missing variables are still helpful for modeling temporal dependencies. Based on these observations, we conduct two types of independent imputation experiments on six synthetic and real world datasets.

Table 2: Experimental Datasets Summarization

Data type	data set	dim	length	source	missing rate
univariate	Sanity check	1	496	synthetic	10% - 50%
	Daily births	1	5113	real world	
	Electricity_MT124	1	17536		
multivariate	DSIM	16	1440	synthetic	5% - 50%
	SCITOS G5	24	5456	real world	
	Traffic volume	10	4272		10%

4.1. Datasets

Three univariate time series (Sanity check, Daily births and Electricity_MT124 datasets) and three multivariate time series (DSIM, SCITOS G5 and Traffic volume datasets) are listed in Table 2. More details are introduced as follows:

1. **Sanity check** (Anava et al., 2015): This is a synthetic time series generated from a fifth-order autoregression (AR) equation:

$$\mathbf{x}_t = \phi_0 + \sum_{i=1}^{5} \phi_i \mathbf{x}_{t-i} + \epsilon_t \tag{8}$$

   where ϕ_0 and $\{\phi_i\}(i \in 1, \ldots, 5)$ are set to 0, 0.6, -0.5, 0.4, -0.4, 0.3, respectively. The noise terms $\{\epsilon_t\}$ are sampled from a distribution of $\mathcal{N}(0, 0.3^2)$. The first five points $\{\mathbf{x}_i\}(i \in 1, 2 \ldots, 5)$ are initialized by $1, 2, 3, 4, 5$, respectively. The length of this time series is 496.

2. **Daily births** (Hipel and McLeod, 1994): This is a time series of the number of daily births in Quebec from Jan, 1977 to Dec, 1990. There are 5113 records.

3. **Electricity-MT124** (Dheeru and Karra Taniskidou, 2017) records clients' electricity consumption every 15 minutes with 140,256 data points. We select one of the client's data as a univariate time series and downsample to 17536 points.

4. **DSIM dataset** (Rahman et al., 2014): DSIM is a simulated diabetes multivariate dataset. 16-dimensional data with additive Gaussian noise is generated for each simulated minute, yielding 1440 data points.

5. **SCITOS G5** (Freire et al., 2009) is a real world dataset, which consists of the measurements of the 24 ultrasound sensors of a SCITOS G5 robot navigating a room. The 5456 sensor readings were sampled at a rate of 9Hz and the robot was following the wall of the room in a clockwise direction, making four trips around the room.

6. **Traffic volume**: This is a real traffic volume dataset collected from 10 stations in the freeway network in a province of China. Each station records flow every 5 minutes from Feb to Apr, resulting in 25632 records. We downsample the recording interval every 30 minutes, and obtain 4272 10-dim records.

In our experiments, we randomly remove some values according to a certain missing rate. For univariate imputation, the missing rates are from 10% to 50% with increments of

10%. For the multivariate imputation on DSIM and SCITOS G5, the missing rates are from 5% to 50% with increments of 5%. Moreover, we imitate a consecutive missing case (we set the total missing rate as 10%) on realistic Traffic volume dataset to show the imputation results of different methods.

4.2. Compared Methods

In the univariate imputation, we compare RIMP-LSTM with 6 representative time series imputation methods both on imputation and prediction tasks. Notice that since these baselines cannot conduct prediction by themselves, they need to be combined with predictors such as ARMA (Kashyap, 1982) or LSTM after imputation. In particular, for Sanity check, we use ARMA as predictors since this data is derived from the autoregression equation. For other univariate data sets, we use LSTM as predictors due to their non-linearity. The univariate imputation methods are as follows:

1. **Forward imputation**: Filling in a missing item with its last observed value.
2. **Indicator approach** (Lipton et al., 2016): Adding an indicator value at each time step. We then obtain a two variable time series from the original univariate time series. If the current input is missing, the added variable is set to 1, otherwise 0.
3. **Spline imputation** (Schoenberg, 1973): Filling in a missing item with spline interpolation.
4. **Moving average (MA) imputation** (Moritz and Bartz-Beielstein, 2015): Filling in missing item with the mean of a tuned window near the missing value.
5. **Regularized EM imputation** (Schneider, 2001): This is a regularized variant of expectation maximization (EM) algorithm for imputation.
6. **Kalman imputation** (Grewal, 2001): One of the most used univariate imputation methods.

For the multivariate dataset (DSIM and SCITOS G5), we compare advanced multivariate imputation methods with RIMP-LSTM. These techniques include:

1. **BPCA** (Oba et al., 2003): An estimation method based on Bayesian principal component analysis (BPCA).
2. **k-NN imputation**: Using the nearest neighbors to fill in the missing values.
3. **MICE** (White et al., 2011): Using a chained equation to fill the missing values.
4. **FLk-NN** (Rahman et al., 2014): Incorporating time lagged correlations both within and across variables by combining k-NN and Fourier transform.
5. **Dynammo** (Li et al., 2009): Learns a linear dynamical system in presence of missing values and fills them.
6. **TRMF** (Yu et al., 2016): This is a most recently proposed framework for time series imputation, based on matrix factorization and graph regularization.

We also design a baseline model called **IMP-LSTM** to verify the effectiveness of residual-short-path modeling. IMP-LSTM does not have residual structures, but uses the same loss function proposed in (7). In this way, IMP-LSTM can fill the missing values and conduct the prediction simultaneously as RIMP-LSTM does.

4.3. Experimental Settings

We evaluate results on two criteria: mean absolute error (MAE) for DSIM task, in order to directly compare with results reported in Rahman et al. (2014), and root mean square error (RMSE) for other datasets. These two criteria can be formulated by

$$\text{RMSE} = (\frac{1}{n}\sum_{i=1}^{n}(x_i^{real} - x_i^{imp})^2)^{\frac{1}{2}} \tag{9}$$

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|(x_i^{real} - x_i^{imp})| \tag{10}$$

where n is the number of missing data points. x_i^{real} and x_i^{imp} denote the truth and the imputed value of i-th missing item respectively.

For the configuration, a single-layer LSTM is used in the RIMP-LSTM architecture, with 128 hidden units. Empirically, we found that more hidden units will slightly improve the imputation performance. For the univariate time series, we randomly remove data with a given missing rate (e.g., 15%), and divide the whole time series into two parts: the first 70% for training and remaining 30% for testing. In this task, we compute the imputation error in the training set (we do not give any ground truth as targets) and report the prediction performance in the test set. For the multivariate time series, we follow the methods in previous work (Li et al., 2009; Rahman et al., 2014; Yu et al., 2016), and compute the imputation performance on the whole dataset.

We use the ADAM (Kingma and Ba, 2014) optimizer and early stopping in our training stage. In the following experiments, λ_{target} is set to 1. All the following experiments are repeated 10 times and their average results reported.

The experiments are run on the tensorflow platform using an Intel Core i5-6500, 3.20-GHz CPU 32-GB RAM and a GeForce GTX 980-Ti 6G. For the Sanity check dataset, we use mean normalization. For other datasets, we adopted min-max normalization. A 5-step sliding window smoothing is adopted for the TraVol dataset due to its strong non-linearity.

4.4. Experimental Results

We report our results of univariate imputation and prediction on three datasets with different missing rates in Table 3 and Table 4, respectively.

For the imputation results (see Table 3), RIMP-LSTM is superior to others in almost all cases. IMP-LSTM is a strong baseline, outperforming the other methods while is inferior to RIMP-LSTM, which verifies the effectiveness of modeling temporal dependency with residual short paths. The one-step-ahead prediction results are shown in Table 4. RIMP-LSTM achieves the best prediction performance on three datasets with various missing rates. Here, IMP-LSTM displays unstable prediction performance on Sanity Check (e.g. the case of 40-50%), on which IMP-LSTM performs worse than other methods. Without residual short paths, IMP-LSTM imputes the missing value only depending on the last state preceding the missing item, and the long short-term memory will be impacted when it is corrupted by inaccurate imputation. Hence, inaccurate imputation will disrupt later prediction.

Table 3: Imputation results (RMSE) for univariate time series.

Dataset	missing (%)	Forward	Indicator	Spline	MA	EM	Kalman	IMP -LSTM	RIMP -LSTM
Sanity check	10	0.296	0.280	0.282	0.247	0.280	0.258	0.261	**0.246**
	20	0.482	0.332	0.317	0.317	0.347	0.326	0.310	**0.295**
	30	0.454	0.378	0.394	**0.363**	0.372	0.400	0.371	0.365
	40	0.464	0.342	0.490	0.389	0.438	0.355	0.366	**0.346**
	50	0.460	0.354	0.508	0.389	0.474	0.352	0.342	**0.335**
Daily births	10	0.371	0.342	0.284	0.327	0.285	0.327	0.256	**0.208**
	20	0.415	0.361	0.345	0.334	0.299	0.318	0.258	**0.215**
	30	0.417	0.376	0.360	0.348	0.323	0.323	0.263	**0.225**
	40	0.438	0.357	0.398	0.353	0.338	0.322	0.266	**0.229**
	50	0.450	0.369	0.531	0.372	0.361	0.328	0.307	**0.237**
Electricity-MT124	10	0.352	0.272	0.334	0.284	0.292	0.265	0.201	**0.189**
	20	0.369	0.277	0.362	0.293	0.302	0.273	0.227	**0.219**
	30	0.385	0.302	0.405	0.304	0.317	0.277	0.239	**0.226**
	40	0.389	0.323	0.411	0.315	0.326	0.281	0.245	**0.229**
	50	0.396	0.332	0.467	0.319	0.330	0.279	0.257	**0.240**

Table 4: One-step-ahead prediction results (RMSE) for univariate time series.

Dataset	missing (%)	Forward	Indicator	Spline	MA	EM	Kalman	IMP -LSTM	RIMP -LSTM
Sanity check	10	0.357	0.315	0.335	0.335	0.288	0.335	0.289	**0.283**
	20	0.359	0.333	0.337	0.336	0.335	0.336	0.293	**0.291**
	30	0.353	0.344	0.339	0.337	0.324	0.338	0.329	**0.312**
	40	0.391	0.341	0.343	0.341	0.343	0.340	0.346	**0.312**
	50	0.372	0.351	0.334	0.335	0.334	0.334	0.348	**0.327**
Daily Births	10	0.274	0.377	0.253	0.271	0.262	0.269	0.275	**0.244**
	20	0.308	0.380	0.307	0.282	0.279	0.275	0.268	**0.225**
	30	0.342	0.383	0.299	0.321	0.280	0.299	0.285	**0.246**
	40	0.359	0.383	0.325	0.328	0.299	0.313	0.275	**0.254**
	50	0.413	0.389	0.386	0.359	0.348	0.330	0.327	**0.269**
Electricity-MT124	10	0.259	0.294	0.266	0.232	0.247	0.243	0.220	**0.210**
	20	0.268	0.298	0.267	0.252	0.251	0.255	0.225	**0.213**
	30	0.284	0.310	0.329	0.271	0.261	0.263	0.233	**0.227**
	40	0.296	0.323	0.350	0.288	0.276	0.270	0.238	**0.238**
	50	0.312	0.332	0.365	0.298	0.279	0.274	0.244	**0.243**

In Fig. 4, the nonparametric tests (Nemenyi test) (Demšar, 2006) at the significance level of 0.05 are conducted for statistical comparisons (overall results of 8 methods on imputation. RIMP-LSTM is slightly better than IMP-LSTM and Kalman models in the case of univariate time seires, and significantly outperforms other methods.

For DSIM and SCITOS G5 datasets, we summarize our average imputation results in Fig. 5(a) and Fig. 5(b) respectively, where the results of BPCA, EM, k-NN, MICE and FLk-NN are published in Rahman et al. (2014) and we reported the results of Dynammo and TRMF by running their source codes.

As seen in Fig. 5, we find that our RIMP-LSTM both has the best imputation performance than other methods both on DSIM and SCITOS G5 datasets. IMP-LSTM and FLk-NN also perform well, and on SCITOS G5, our RIMP-LSTM is equivalent or slightly better than IMP-LSTM. k-NN has inferior performance when the missing ratio increases,

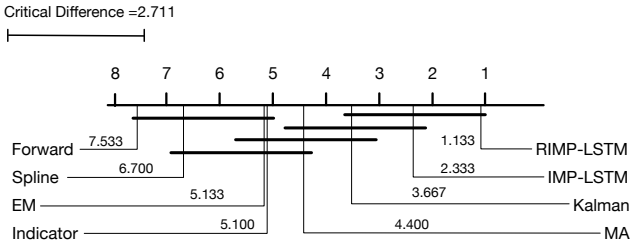
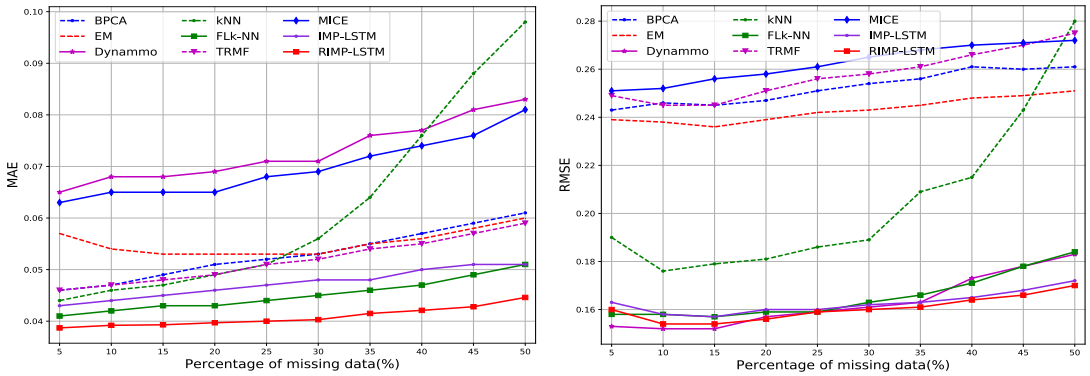


Figure 4: Comparison of RIMP-LSTM on univariate time series imputation with seven baselines on the Nemenyi test. The methods connected in one group are not significantly different at 0.05 significance level.



(a) DSIM (b) SCITOS G5

Figure 5: Imputation results for multivariate time series.

since they ignore the temporal dependencies. TRMF achieves temporal dependency modeling with its graph-based temporal regularization while its performance limits on its lag settings. FLk-NN incorporates time-lagged correlations both within and across variables. And our RIMP-LSTM model temporal dependencies via the graph-based residual-short-path structures and LSTM.

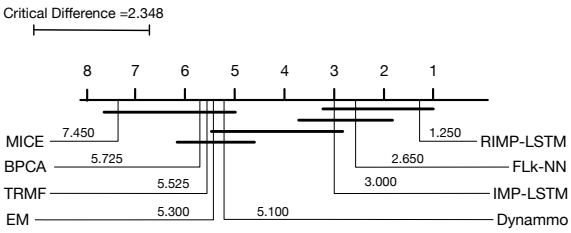


Figure 6: Comparison of RIMP-LSTM on multivariate time series imputation with seven baselines on the Nemenyi test. The methods connected in one group are not significantly different at 0.05 significance level.

In Fig. 6, Nemenyi test (Demšar, 2006) at the significance level of 0.05 are also conducted for statistical comparisons on multivariate imputation. RIMP-LSTM is slightly better than FLk-NN and IMP-LSTM models, and significantly outperforms other methods.

To further analyze the computational efficiency of our RIMP-LSTM, we compare it with vanilla LSTM (with forward imputation first) and IMP-LSTM with the same configurations and report the runtime (per 100 epochs of training) on DSIM in Fig. 7. The results show that our RIMP-LSTM only slightly increases the cost of runtime due to the added structure of residual short paths and switch units.

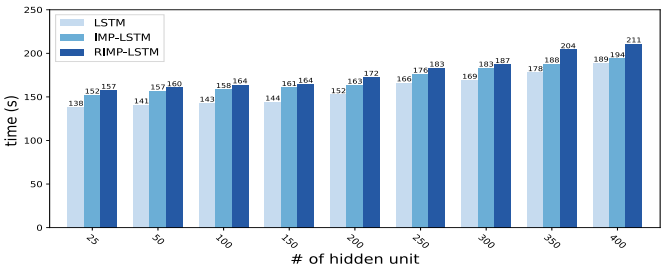


Figure 7: The runtime comparisons among vanilla LSTM, IMP-LSTM and RIMP-LSTM on the DSIM dataset.

4.5. Visualization of Imputation

We visualize the imputation results on multivariate traffic volume dataset which contains 10% consecutive missing block.

In Fig. 8, we found that our end-to-end methods (RIMP-LSTM and IMP-LSTM) achieved better imputation performance than FLk-NN, Dynammo and TRMF, as the imputed values (red points) are located in or closed to the original curves. Dynammo and TRMF can hardly work in this case. TRMF models the missing values in two ways: by correlation with other variables at the same time step, and by the dependency graph regularization. Since all the values of variables at the same time interval are deleted, there is nothing to correlate with (it also causes the failure of Dynammo). Moreover, since the missing time interval is relatively large, the dependency graph regularization is not effective.

On the other hand, compared with IMP-LSTM, RIMP-LSTM has better performance as it is good at modeling temporal dependencies with weighted residual short paths, which demonstrates that the reasonability of using these weighted residual paths to model graph-like temporal dependencies for imputation.

5. Conclusions

In this work, we consider recovering missing values and modeling the dynamic dependencies from incomplete time series in an end-to-end way. From the point of view of a temporal dependency graph, we regard residual-short paths as a specific graph topology and integrate these structures into an LSTM network, called RIMP-LSTM. This novel model can be trained end-to-end with missing data, and at run-time, simultaneously imputes missing values and predicts the next value. To the best of our knowledge, this is the first work that

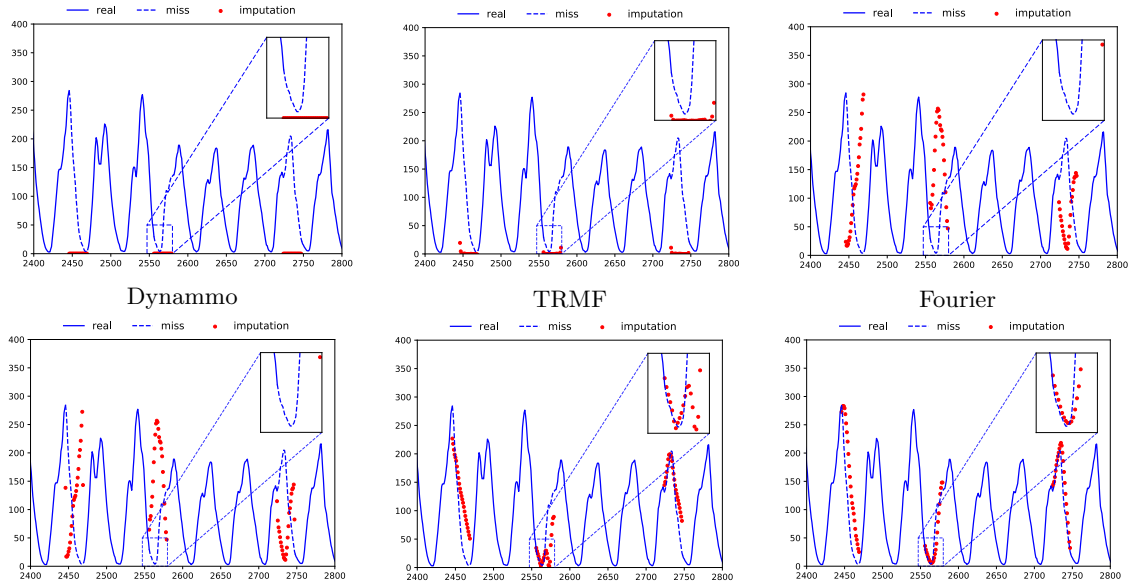


Figure 8: The visualization result of imputation on the traffic volume dataset. The dashed lines denote cases of consecutive missing values (three parts). We zoom in the imputed values in the missing region.

combines the idea of a graph-based model with residual short paths and learns temporal dependencies from incomplete time series in an end-to-end way. We evaluated RIMP-LSTM on several data with different levels of random missing data and consecutive missing ones. The results show that our model achieves state of the art performance of imputation and prediction.

Acknowledgements

The work described in this paper was partially funded by the National Natural Science Foundation of China (Grant No. 61502174, 61872148), the Natural Science Foundation of Guangdong Province (Grant No. 2017A030313355), the Guangzhou Science and Technology Planning Project (Grant No. 201704030051), the Opening Project of Guangdong Province Key Laboratory of Big Data Analysis and Processing (Grant No. 2017014) and the Guangdong University of Finance & Economics Big Data and Educational Statistics Application Laboratory (Grant No. 2017WSYS001).

References

Oren Anava, Elad Hazan, and Assaf Zeevi. Online time series prediction with missing data. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2191–2199. PMLR, 2015.

Philémon Brakel, Dirk Stroobandt, and Benjamin Schrauwen. Training energy-based models for time-series imputation. *Journal of Machine Learning Research*, 14:2771–2797, 2013.

Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. Recurrent neural networks for multivariate time series with missing values. *Scientific reports*, 8(1):6085, 2018.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30, December 2006. ISSN 1532-4435.

Dua Dheeru and Efi Karra Taniskidou. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

A. L Freire, G. A Barreto, M Veloso, and A. T Varela. Short-term memory mechanisms in neural network learning of robot navigation tasks: A case study. In *Robotics Symposium*, pages 1–6, 2009.

Mohinder S. Grewal. *Kalman Filtering*. J. Wiley, 2001.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

Hipel and McLeod. Number of daily births in quebec, jan. 01, 1977 to dec. 31, 1990. Time Series Data Library, 1994. https://datamarket.com/data/set/235j.

Hochreiter, Sepp, Schmidhuber, and Jürgen. Long short-term memory. *Neural Computation*, 9(8):1735, 1997.

Rangasami L Kashyap. Optimal choice of ar and ma parts in autoregressive moving average models. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 4(2):99–104, 1982.

Eamonn J Keogh and Michael J Pazzani. An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback. In *International Conference on Knowledge Discovery and Data Mining*, pages 239–243, 1998.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *Computer Science*, 2014.

Lei Li, James Mccann, Nancy S. Pollard, and Christos Faloutsos. Dynammo: mining and summarization of coevolving sequences with missing values. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 507–516, 2009.

Zachary C. Lipton, David C. Kale, and Randall Wetzel. Directly modeling missing data in sequences with rnns: Improved classification of clinical time series. 2016.

Qianli Ma, Lifeng Shen, Enhuan Chen, Shuai Tian, Jiabing Wang, and Garrison W. Cottrell. Walking walking walking: Action recognition from action echoes. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 2457–2463, 2017. doi: 10.24963/ijcai.2017/342.

Steffen Moritz and Thomas Bartz-Beielstein. imputets: Time series missing value imputation. *R package version 0.4*, 2015.

S Oba, M. A. Sato, I Takemasa, M Monden, K Matsubara, and S Ishii. A bayesian missing value estimation method for gene expression profile data. *Bioinformatics*, 19(16):2088–96, 2003.

Yao Qin, Dongjin Song, Haifeng Chen, Wei Cheng, Guofei Jiang, and Garrison W. Cottrell. A dual-stage attention-based recurrent neural network for time series prediction. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 2627–2633, 2017. doi: 10.24963/ijcai.2017/366.

S. A Rahman, Yuxiao Huang, J Claassen, and S Kleinberg. Imputation of missing values in time series with lagged correlations. In *IEEE International Conference on Data Mining Workshop*, pages 753–762, 2014.

Donald B Rubin. Inference and missing data. *Biometrika*, pages 581–592, 1976.

Tapio Schneider. Analysis of incomplete climate data: Estimation of mean values and covariance matrices and imputation of missing values. *Journal of Climate*, 14(5):853–871, 2001.

I. J Schoenberg. Cardinal spline interpolation. 7(1):1–42, 1973.

B Sinopoli, L Schenato, M Franceschetti, K Poolla, M. I Jordan, and S. S Sastry. Kalman filtering with intermittent observations. In *Decision and Control, 2003. Proceedings. IEEE Conference on*, pages 701–708 Vol.1, 2004.

Andreas Veit, Michael J Wilber, and Serge Belongie. Residual networks behave like ensembles of relatively shallow networks. In *Advances in Neural Information Processing Systems 29*, pages 550–558. 2016.

I. R. White, P Royston, and A. M. Wood. Multiple imputation using chained equations: Issues and guidance for practice. *Statistics in Medicine*, 30(4):377–99, 2011.

Xiuwen Yi, Yu Zheng, Junbo Zhang, and Tianrui Li. St-mvl: filling missing values in geo-sensory time series data. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-16*, pages 2704–2710, 2016.

Hsiang-Fu Yu, Nikhil Rao, and Inderjit S Dhillon. Temporal regularized matrix factorization for high-dimensional time series prediction. In *Advances in Neural Information Processing Systems 29*, pages 847–855. 2016.