Learning Representations for Incomplete Time Series Clustering

Qianli Ma[1,3*], **Chuxin Chen** [1*], **Sen Li** [1*], **Garrison W. Cottrell** [2]

[1] South China University of Technology, Guangzhou, China
[2] University of California, San Diego, CA, USA
[3] Key Laboratory of Big Data and Intelligent RobotSouth China University of Technology, Ministry of Education,
qianlima@scut.edu.cn, sherlockem@foxmail.com, awslee@foxmail.com, gary@ucsd.edu

Abstract

Time-series clustering is an essential unsupervised technique for data analysis, applied to many real-world fields, such as medical analysis and DNA microarray. Existing clustering methods are usually based on the assumption that the data is complete. However, time series in real-world applications often contain missing values. Traditional strategy (imputing first and then clustering) does not optimize the imputation and clustering process as a whole, which not only makes performance dependent on the combination of imputation and clustering methods but also fails to achieve satisfactory results. How to best improve the clustering performance on incomplete time series remains a challenge. This paper proposes a novel unsupervised temporal representation learning model, named Clustering Representation Learning on Incomplete time-series data (CRLI). CRLI jointly optimizes the imputation and clustering process to impute more discriminative values for clustering and make the learned representations possessed good clustering property. Also, to reduce the error propagation from imputation to clustering, we introduce a discriminator to make the distribution of imputation values close to the true one and train CRLI in an alternating training manner. An experiment conducted on eight real-world incomplete time-series datasets shows that CRLI outperforms existing methods. We demonstrate the effectiveness of the learned representations and the convergence of the model through visualization analysis. Moreover, we reveal that the joint training strategy can impute values close to the true ones in those important sub-sequences, and impute more discriminative values in those less important sub-sequences at the same time, making the imputed sequence cluster-friendly.

Introduction

Time series data is ubiquitous in real life, and there are many application scenarios, such as medicine (de Jong et al. 2019), gene expression analysis (de Souto et al. 2008), and financial markets (Azoff 1994). When the data lacks category labels, data analysts typically will apply clustering methods (Paparrizos and Gravano 2015; Zhang et al. 2018; Madiraju et al. 2018; Ma et al. 2019) to extract interesting patterns and valuable information from complex and massive datasets (Aghabozorgi, Shirkhorshidi, and Wah 2015).

However, real-world time series usually contain missing values due to uncontrollable factors (e.g., device failure and communication errors), which violates the complete-data assumption of most state-of-the-art clustering methods. Incomplete data make any inference more difficult (Rubin 1976) and harms downstream applications' performance (Cheema and J. 2014; Cao et al. 2018). On the other hand, incomplete time series clustering is essential for analysis in some domains, where the data is often corrupted, and re-recording them is expensive and impractical. For example, in precision medicine, it is meaningful to stratify the patients and accordingly decide on the right type and time point of therapy for an individual (de Jong et al. 2019). In DNA microarray analysis, it is useful to cluster gene expression data to identify genes' function (de Souto et al. 2008). Under such circumstances, performing imputation first and then clustering is a natural solution. However, this two-stage strategy makes the clustering performance heavily depend on the best combination of the imputation method and the clustering method. In practical applications, there is usually no category information to help pick the best combination. Furthermore, two-stage approaches separate the imputation and its downstream analysis, which may lead to sub-optimal results (Che et al. 2018; Wells et al. 2013).

Deep learning is a data-driven method that can learn effective representations for various tasks and has achieved great success in recent years. Madiraju et al. (Madiraju et al. 2018) and Ma et al. (Ma et al. 2019) successfully applied deep learning to time series clustering and achieved state-of-the-art performance. Also, Generative Adversarial Networks (GANs) have shown potential for handling missing data (Yoon, Jordon, and Der Schaar 2018; Li, Jiang, and Marlin 2019; Ma, Li, and Cottrell 2020). Most recently, Jong et al. (de Jong et al. 2019) proposed a deep learning-based method, named VaDER, for clustering time series with missing values. However, these methods do not effectively prevent error propagation from imputation to clustering, leading to unsatisfactory performance. Moreover, VaDER uses a generative variational auto-encoder framework to perform the clustering, which requires a large amount of data to train and performs poorly when the data amount is small.

In this paper, we propose a novel unsupervised incomplete time series clustering model, Clustering Representation Learning on Incomplete time-series data (CRLI), which

*These authors contributed equally to this work can generate cluster-friendly temporal representations for incomplete data. Specifically, CRLI consists of two parallel branches (see Figure 1). The upper branch employs a bidirectional recurrent neural network with an adversarial strategy to reduce error propagation from imputation to clustering. Moreover, to enable the learned representations to have good clustering properties, the lower branch of CRLI integrates the soft K-means objective (Zha et al. 2002; Ma et al. 2019) into an encoder-decoder network to generate cluster-friendly representations. In this way, CRLI simultaneously optimizes the imputation and clustering process in a single deep learning framework. This joint training strategy not only alleviates imputed bias introduced by imputation to improve clustering performance, but also makes the imputed values more suitable for clustering. The main contributions of this work can be summarized as follows:

- We propose an incomplete temporal clustering representation learning method for directly clustering time series in the presence of missing values. Also, we introduce an adversarial strategy to reduce the error propagation from imputation to clustering, which boosts the clustering performance.
- We couple the imputation and clustering process together and optimize them simultaneously to generate cluster-friendly representations of the incomplete temporal data rather than in a two-stage manner. Additionally, we introduce a fully connected layer between the imputation and clustering process to alleviate the instability introduced by joint optimization.
- Experiments conducted on eight real-world incomplete time-series datasets show that CRLI achieves state-of-the-art clustering performance. We provide a visualization analyses to demonstrate the effectiveness of the learned representations and show the convergence of the proposed model. Quantitative and qualitative analysis are also provided to reveal that joint training strategy can impute more discriminative values to improve clustering performance.

Related Work

Incomplete time series clustering methods can be divided into two methods: two-stage methods (imputation and then clustering) and one-stage ones (joint optimization).

Two-stage methods

A common methodology to cluster the incomplete time series is to first impute the missing values and then apply existing clustering methods to the complete data. The simplest imputation is to replace the missing values with zero, mean, or median values. Recently, many more complex methods based on deep learning have been proposed. We only mention some closely related approaches. Cao et al. (Cao et al. 2018) proposed BRITS, a bidirectional recurrent dynamical system for imputing missing values without any specific assumption on the underlying data generating process, making their model quite general. Yoon et al. (Yoon, Jordon, and Der Schaar 2018) and Luo et al. (Luo et al. 2018) proposed a GAN-based generative framework for filling the missing values.

Although these methods have achieved promising imputation performance, for incomplete clustering, imputing missing values first means imputation and clustering are independent, degrading the performance. Moreover, the two-stage method makes the overall clustering performance rely on the best combination of imputation and clustering methods. However, it is difficult to choose the best combination when category information is lacking.

One-stage methods

A series of variants based on the classical Fuzzy C-Means (FCM) algorithm has appeared to cluster incomplete data without pre-imputation. Hathaway et al. (Hathaway and Bezdek 2001) proposed modified versions of FCM with four strategies to deal with missing values. Zhang et al. (Zhang and Chen 2003) further introduced the kernel method to FCM. Li et al. (Li, Gu, and Zhang 2010) transform the incomplete dataset into an interval-valued one and then applied the interval FCM algorithm. Li et al. (Li et al. 2017) further introduce the kernel method to interval FCM. However, these raw-data based methods are usually sensitive to outliers and noise (Ferreira and Zhao 2016). In addition, these methods are not designed for time series, so they cannot effectively capture the non-linear temporal dynamic characteristics. Recently, based on the variational auto-encoder clustering framework (VaDE) proposed by Jiang et al. (Jiang et al. 2017), Jong et al. (de Jong et al. 2019) instantiated the auto-encoder as an RNN and proposed VaDER for clustering incomplete time series, integrating the imputation into the model to reduce error propagation to the clustering. However, since VaDER has no constraints on missing values on the encoder side, it is likely to introduce errors into the clustering process when encoding incomplete sequences into hidden representations. Therefore, VaDER inevitably suffers from the negative impact of missing values on the clustering process.

Proposed Method

Here we introduce our model named Clustering Representation Learning on Incomplete time-series data (CRLI) to improve the incomplete time series clustering performance. The general structure of CRLI is illustrated in Figure 1. On the upper branch, the generator imputes the missing values and obtains the imputed data. The discriminator then takes the imputed data as input and outputs the probability that the variable at each time step is an observed value. On the lower branch, the last hidden states of the generator are concatenated and fed into a fully-connected layer to get latent representations. After that, to obtain cluster-friendly representations for incomplete temporal data, we integrate a soft K-means objective into an encoder-decoder network to reconstruct the original input data. The whole training process of CRLI is trained end-to-end, optimizing the imputation and clustering simultaneously.

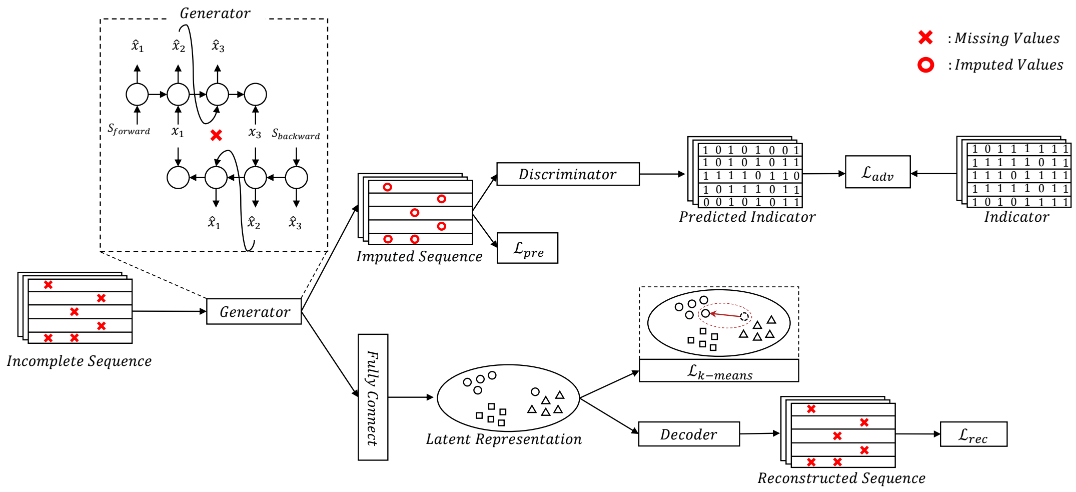


Figure 1: The general architecture of the Clustering Representation Learning on Incomplete time-series data (CRLI).

Learning Clustering Representations on Incomplete Temporal Data

Given a multivariate time series dataset $D_{ata} = \{X_1, X_2, \dots, X_N\}$, each time series $X_i \in R^{T\times d}$ is denoted as $X_i = (x_1, x_2, \dots, x_T)$, where $x_t \in R^d$ and d is the input dimension of each time step. In our missing case, there is also a corresponding indicator set $M = \{M_1, M_2, \dots, M_N\}$, and each indicator M_i is denoted as $M_i = (m_1, m_2, \dots, m_T)$, where $m_t \in \{0, 1\}^d$. When the component of x_t is missing, we set the corresponding component of m_t to 0 and vice versa.

To capture the dynamics of the time series, we set the generator architecture of CRLI to a recurrent neural network (RNN). A standard recurrent network (Rumelhart, Hinton, and Williams 1986) can be formulated as follows:

$$h_t = tanh(W_h h_{t-1} + W_x x_t + b) \tag{1}$$

where W_h,W_x and b are parameters, and h_{t-1} is the hidden state of the previous time step. However, the value of x_t cannot directly be fed into the RNN since it could contain missing values. Hence, we use an imputed value u_t. Formally, the revised RNN can be updated as follows:

$$\hat{x}_t = W_{imp} h_{t-1} + b_{imp} \tag{2}$$

$$u_t = m_t \odot x_t + (1 - m_t) \odot \hat{x}_t \tag{3}$$

$$h_t = tanh(W_h h_{t-1} + W_x u_t + b) \tag{4}$$

where W_{imp} and b_{imp} are learnable parameters, and $\odot$ denotes the element-wise product. Eq. (2) denotes the model is trained to predict the value x_t with previous hidden state h_{t-1}. Eq. (3) replacing the missing components in x_t with the corresponding elements of predicted values $\hat{x}_t$. Eq. (4) denotes the RNN can deal with the missing values with the imputed value u_t. Once the model reads the entire input sequence, we can get the corresponding predicted sequence $\hat{X}$ by Eq. (2), which can be denoted as $\hat{X} = \{\hat{x}_1, \hat{x}_2, \dots, \hat{x}_T\}$. To deal with the situation where the first value of the sequence may be missing, we introduced an identifier as the starting signal of the RNN. Therefore, the identifier and the predicted loss of all time-series samples are defined as follows:

$$S = W_s s \tag{5}$$

$$\mathcal{L}_{pre} = \frac{1}{N} \sum_{i=1}^{N} ||(X_i - \hat{X}_i) \odot M_i)||_2^2 \tag{6}$$

where $s \in R^1$ is a constant value and $W_s \in R^{d\times 1}$ is a adaptable weight matrix used to transfer s to the same dimension as x_t. Here, we only give the updating process when the generator is unidirectional RNN for simplicity. It should note that, to reduce the error of the imputed value, following (Cao et al. 2018), the generator is instantiated by a bidirectional RNN in our experiments. The backward RNN only needs to reverse the input sequence order and then use it as input. We used two different identifiers ($S_{forward}$ and $S_{backward}$) for the forward and backward RNN so that CRLI knows the different directions.

As shown in Figure 1, the generator reads the entire incomplete input sequence and then encode it to its representation, so that we can also regard it as an encoder. Therefore, the generator (denoted as $f_{gen/enc}$) can be rewritten as follows:

$$H_i = f_{gen/enc}(X_i, M_i) \tag{7}$$

Eq. (7) denotes that the generator takes an incomplete time series sample and its missing indicator as input, and directly encodes it into a latent temporal representation without any pre-imputation. Specifically, H is the concatenation of the last hidden state of the forward and backward RNN. To enable H capture the original time series's informative features, we fed it to a decoder network (denotes as f_{dec}) to reconstruct the incomplete time series X. We use masked Mean Square Error (MSE) as the reconstruction loss, which is defined as follows:

$$\mathcal{L}_{rec} = \frac{1}{N} \sum_{i=1}^{N} ||(X_i - f_{dec}(H_i)) \odot M_i||_2^2 \tag{8}$$

Eq. (8) denotes that we only reconstruct the observed components of X, regardless of missing ones. However, the learned representations H may not suitable for the clustering task. Hence, following (Zha et al. 2002; Ma et al. 2019), we encourage H to form cluster structures by a soft K-means objective, defined as follows:

$$\mathcal{L}_{k-means} = Tr(H^T H) - Tr(F^T H^T H F) \; s.t. \; F^T F = I \tag{9}$$

where $H \in R^{h \times N}$, h and N are the dimensions of the representation and training batch size, respectively. $F \in R^{N \times k}$ denotes the cluster indicator matrix, k is the number of clusters. Since the learning of H is dynamic instead of static, the training process of Eq. (9) consists of updating F and H iteratively. When fixing F, updating H can follow the standard stochastic gradient descent (SGD), encouraging the representations to form cluster structures. While fixing H, according to the Ky Fan theorem, F can be obtained by computing the k-truncated singular value decomposition (SVD) of H.

However, we found empirically that directly applying Eq. (9) to optimize H reduces the stability of CRLI, harming the clustering performance. Since cluster-friendly representation H is not necessarily beneficial to the imputation process, jointly optimizing the same H in the imputation and clustering process may lead to more difficult optimization. Hence, as shown in Figure 1, before we apply Eq. (8) and Eq. (9) to H, H is fed to a fully connected layer to alleviate model instability. We will show its effectiveness in the experimental part.

Adversarial Strategy

The clustering process relies on the representations generated by the generator. However, the generator inevitably introduces imputation bias when the input series has missing values, such that it will propagate errors from imputation to clustering, hindering clustering performance. To this end, we introduce an adversarial strategy to reduce this error.

Given a time series $X_i \in R^{T \times d}$, according to Eq. (3), there is an imputed sequence $U_i = (u_1, u_2, \dots, u_T)$ and its corresponding missing indicator sequence $M_i = (m_1, m_2, \dots, m_T)$. For a bidirectional RNN, U_i is the average of the imputed sequence of the forward and backward RNN. We can know which values in U_i are predicted and which values are real with M_i. We use this to train a discriminator D and then use it to teach the generator in turn, so that the distribution of the predicted values is closer to the distribution of the true one. Note that D is reliable since it reads the real values and the predicted values at the same time. Formally, D is trained by minimizing the following loss function:

$$\mathcal{L}_{dis} = -\,[E \underbrace{\log(D(X_{real}))}_{X_{real} \sim U} + E \underbrace{\log(1 - D(\hat{X}_{pre}))}_{\hat{X}_{pre} \sim U}] \tag{10}$$

$$= -\frac{1}{N}\sum\nolimits_{i=1}^{N}[M_i \odot \log(D(U_i)) + (1 - M_i) \odot \log(1 - D(U_i))] \tag{11}$$

where $D(\cdot)$ denotes the predicted indicator probability $\hat{M}_i \in R^{T \times d}$ of the discriminator, indicating the discriminator detects the real or predicted of each value in X_i. Eq. (10) denotes the discriminator maximizes its output for real values while minimizes its output for predicted values. In this way, discriminator learns to know what's actually-observed value and what's imputed value in an imputed sequence.

By fooling the discriminator D, CRLI is trained to generate imputed values closer to the distribution of real values. The adversarial loss of CRLI can be defined as follows:

$$\mathcal{L}_{adv} = \frac{1}{N}\sum\nolimits_{i=1}^{N}(1 - M_i) \odot \log(1 - D(U_i)) \tag{12}$$

Hence CRLI tries to maximize the discriminator output for imputed values. We alternately update the parameters of the discriminator and the rest of the CRLI.

Overall Loss Function

Finally, the overall training loss $\mathcal{L}_{CRLI}$ of CRLI is defined by:

$$\mathcal{L}_{CRLI} = \mathcal{L}_{pre} + \mathcal{L}_{rec} + \mathcal{L}_{adv} + \lambda * \mathcal{L}_{k-means} \tag{13}$$

where λ is the coefficient. Eq. (13) enables CRLI directly to learn the cluster-friendly temporal representations on incomplete time series. $\mathcal{L}_{pre}$ captures the temporal dynamics of the incomplete time series. $\mathcal{L}_{rec}$ aims to capture informative features that can reconstruct original samples. $\mathcal{L}_{k-means}$ encourages the learned representations to form cluster structures. $\mathcal{L}_{adv}$ reduces the error propagation from imputation to clustering. The detailed training method of CRLI is presented in Algorithm 1.

Algorithm 1 CRLI Training Method

```
Input: Incomplete time series dataset: $D$; Number of clusters: $K$; Maximum iteration: $Maxiter$
Output: Cluster result: $S$
 1: Orthogonal initialize cluster indicator matrix $F$.
 2: for $iter = 1$ to $MaxIter$ do
 3:     (1) Discriminator optimization
 4:     Update discriminator $D$ using SGD based on Eq. (11).
 5:     (2) CRLI optimization (fixed $D$)
 6:     Update representations $H$ using SGD based on Eq. (13).
 7:     Update $F$ by computing the $k$-truncated SVD of $H$.
 8: end for
 9: Apply K-means to the learned representations $H$.
10: return Clustering result $S$.
```

Experiments

We collected eight real-world incomplete time-series datasets from various existing works in several domains (Alizadeh et al. 2000; Bianchi, Mikalsen, and Jenssen 2017; Chen et al. 2002; Dua and Graff 2017; Liang et al. 2005; Silva et al. 2012) and conducted experiments on these datasets to evaluate performance. The statistics of these 8 datasets are shown in Section A of the Supplementary Material. Following (Xie, Girshick, and Farhadi 2016; Guo et al. 2017; Madiraju et al. 2018; Ma et al. 2019), we train the model on the training set and evaluate it on the test set.

Imputation	ZERO					GAIN					BRITS					VaDER	CRLI
Dataset \ Cluster	KS	DEC	IDEC	DTC	DTCR	KS	DEC	IDEC	DTC	DTCR	KS	DEC	IDEC	DTC	DTCR		
Ali-v1	0.50(0.06)	**0.85(0.00)**	0.72(0.00)	0.47(0.01)	**0.85(0.00)**	0.63(0.24)	0.46(0.00)	0.62(0.00)	0.49(0.04)	0.56(0.16)	0.50(0.06)	**0.85(0.00)**	0.82(0.06)	0.47(0.01)	0.82(0.06)	0.47(0.04)	**0.85(0.00)**
Ali-v2	0.69(0.09)	0.77(0.00)	0.79(0.06)	0.74(0.05)	0.47(0.03)	0.70(0.16)	0.51(0.00)	0.60(0.00)	0.54(0.10)	0.78(0.07)	0.60(0.02)	0.77(0.00)	0.83(0.08)	0.70(0.02)	0.48(0.01)	0.49(0.08)	**0.93(0.04)**
Ali-v3	0.75(0.03)	0.77(0.00)	0.77(0.02)	0.73(0.03)	0.63(0.06)	0.75(0.03)	0.51(0.00)	0.64(0.00)	0.48(0.18)	0.72(0.01)	0.72(0.03)	0.81(0.00)	0.74(0.03)	0.70(0.01)	0.69(0.01)	0.29(0.06)	**0.92(0.05)**
BloodSample	0.59(0.07)	0.67(0.00)	0.67(0.00)	0.72(0.06)	0.50(0.00)	0.62(0.14)	0.67(0.00)	0.65(0.00)	0.55(0.04)	0.51(0.00)	0.69(0.00)	0.62(0.00)	0.64(0.00)	0.75(0.04)	0.59(0.05)	0.67(0.06)	**0.85(0.02)**
Chen	0.59(0.09)	0.51(0.00)	0.50(0.00)	0.56(0.06)	0.51(0.01)	0.58(0.12)	**0.77(0.00)**	0.49(0.00)	0.50(0.01)	0.51(0.02)	0.54(0.06)	0.65(0.00)	0.61(0.00)	0.54(0.01)	0.50(0.00)	0.50(0.01)	0.63(0.05)
Vote	0.57(0.02)	0.83(0.00)	0.84(0.00)	0.57(0.09)	0.50(0.00)	0.69(0.05)	0.81(0.00)	0.76(0.00)	0.63(0.15)	0.50(0.01)	0.62(0.07)	0.81(0.00)	0.87(0.00)	0.69(0.11)	0.76(0.01)	0.58(0.03)	**0.91(0.05)**
Liang	0.55(0.15)	0.64(0.00)	0.70(0.00)	0.64(0.00)	**0.82(0.17)**	0.64(0.10)	0.42(0.00)	0.42(0.00)	0.56(0.00)	0.75(0.14)	0.58(0.12)	0.64(0.00)	0.70(0.00)	0.64(0.00)	0.68(0.13)	0.70(0.13)	0.67(0.03)
Physionet	0.50(0.00)	0.52(0.00)	0.52(0.00)	0.50(0.00)	0.64(0.00)	0.50(0.00)	0.51(0.00)	0.51(0.00)	0.66(0.04)	0.73(0.00)	0.50(0.00)	0.51(0.00)	0.50(0.00)	0.50(0.00)	0.52(0.01)	0.70(0.10)	**0.76(0.00)**
Best	0	1	0	0	2	0	1	0	0	0	0	1	0	0	0	0	**6**
AVG RANK	11.25	5.875	5.625	9.375	10.25	9.625	10.625	11.625	12.625	8.75	10.5	5.5	5.875	9.75	10.625	10.75	**2**
AVG RI	0.5936	0.6938	0.6890	0.6180	0.6134	0.6387	0.5835	0.5866	0.5520	0.6332	0.5948	0.7068	0.7134	0.6227	0.6299	0.5503	**0.8140**
***p*-value**	5.41E-04	3.48E-03	2.86E-03	2.65E-03	3.25E-02	6.75E-04	1.02E-02	2.30E-05	8.59E-04	1.54E-02	4.82E-04	1.84E-02	2.77E-02	1.82E-03	8.79E-03	1.18E-02	-

Table 1: Rand Index (RI) comparisons on 8 real incomplete time series datasets (the values in parentheses present standard deviations)

As mentioned above, we employ the adapted bidirectional multi-layer RNN as the generator (namely, encoder). The decoder is a single-layer RNN. The discriminator is a 5-layer RNN, and the number of units of each layer is fixed to $\{32, 16, 8, 16, 32\}$, respectively. We use Gated Recurrent Units (GRU) in the RNN (Cho et al. 2014). The number of layers of the encoder is $l \in \{1, 2\}$. The number of units of each layer in the encoder is $h \in \{50, 100\}$. The last hidden states of encoder are concatenated and then sent to a fully-connected layer to obtain the representations. The decoder takes the learned representation as its initial state and the forward identifier $S_{forward}$ as the starting drive signal to perform iterative prediction, i.e., the output at time $t-1$ is fed as the input at time t. The λ in Eq.(13) $\in \{1e\text{-}3, 1e\text{-}6, 1e\text{-}9\}$. The batch size is 32. The experiments are run on the TensorFlow (Abadi et al. 2016) platform using an Intel Core i7-6850K, 3.60-GHz CPU, 64-GB RAM and a GeForce GTX 1080-Ti 11G GPU. The Adam (Kingma and Ba 2015) optimizer is employed with an initial learning rate of $5e$-3.

Comparison with State-of-the-art Methods

We compare CRLI with a one-stage method (state-of-the-art incomplete time-series deep clustering, VaDER (de Jong et al. 2019)) and some two-stage methods. For two-stage methods, we first impute missing values with two SOTA imputation methods (i.e., BRITS (Cao et al. 2018), GAIN (Yoon, Jordon, and Der Schaar 2018)) and the commonly used zero imputation method (ZERO). Then we apply existing SOTA clustering methods (i.e., KS (Paparrizos and Gravano 2015), DEC (Xie, Girshick, and Farhadi 2016), IDEC (Guo et al. 2017), DTC (Madiraju et al. 2018), DTCR (Ma et al. 2019)). All the following results are obtained by running their published code (If a method provides a combination of parameters, we go through all the parameters; if not, we use its default parameter values). The details of these methods are described in section B of the Supplementary Material. Following recent works (Ma et al. 2019; Zhang et al. 2018), we use Rand Index (RI) (Rand 1971) to evaluate clustering performance. The RI is defined as:

$$RI = \frac{TP + TN}{n(n-1)/2} \tag{14}$$

where TP (True Positive) is the number of pairs of time series that are correctly assigned in the same cluster, TN (True Negative) is the number of pairs that are correctly assigned in different clusters and n is the size of the dataset.

Due to space limitations, we here only provide RI comparisons of the various methods. More metrics (including Normalized Mutual Information (NMI), Cluster Purity (PUR), and Cluster Accuracy (ACC)) are provided in section C of the Supplementary Material. Note that CRLI also achieves the lowest average rank of 2.5, 1.6 and 1.9 on those metrics, respectively.

As shown in Table 1, CRLI achieves the best performance in terms of the highest number of best results 6, the lowest average RANK of 2, and the highest average RI of 0.8140. In two-stage methods, involving a large number of combinations, the best results can be achieved on some datasets, but it is challenging to pick the best combination when there is a lack of labels in real scenarios. Also, we observed the same phenomenon as in (De Souto, Jaskowiak, and Costa 2015) that sometimes a simple imputation strategy can achieve comparable results as complex ones.

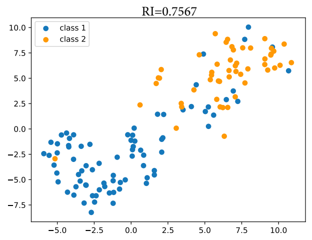
Compared with the two-stage method, CRLI achieved better overall clustering performance, which shows that jointly optimize the imputation and clustering process indeed improve the performance of clustering on incomplete time series. As for the existing one-stage method (VaDER), CRLI is superior to it on 7 out of 8 datasets, indicating that our approach can effectively reduce the error from imputation propagation to clustering, boosting the performance. As mentioned before, VaDER is a generative clustering framework. It performs well on relatively large datasets (the training sizes of BloodSample and Physionet are all greater than 400), while performing poorly on small ones. To statistically analyze the performance, we perform a pairwise comparison for each method against CRLI. Specifically, we conduct the Wilcoxon signed-rank test (Demsar 2006) to measure the significance of the difference. As shown in the last row of Table 1, CRLI is significantly better than all of the other methods at $p < 0.05$ level.

Ablation Study

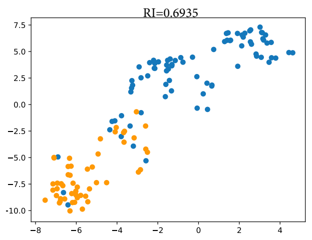
To verify the effectiveness of each component of CRLI, we show a comparison between the full CRLI model and its four ablation models: 1) CRLI without feature learning (namely, directly applying K-means to the imputed data, denoted as w/o featured); and 2) CRLI without the fully-connected layer (w/o fc); 3) CRLI without the adversarial strategy (w/o adv); 4) CRLI without the joint optimizing strategy (namely, it first performs imputation and then does the clustering, denoted as w/o jointly). Table 2 shows that the full CRLI is significantly superior to all of its ablations at $p < 0.05$ level, demonstrating the effectiveness of all its components. The detailed result is reported in Section D of the Supplementary Material.

Dataset	w/o featured	w/o fc	w/o adv	w/o jointly	CRLI
Best	0	0	0	0	**8**
AVG RANK	2.75	4.25	2.625	3.625	**1**
AVG RI	0.6413	0.5139	0.6535	0.5750	**0.8140**
***p*-value**	1.32E-02	6.60E-05	3.77E-03	3.04E-03	-

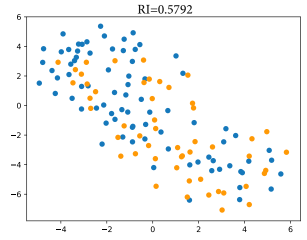
Table 2: Rand Index(RI) ablation study of CRLI



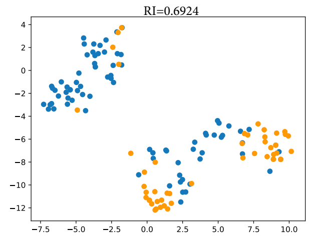
(a) BRITS-DTCR



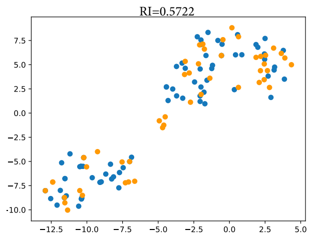
(b) BRITS-DTC



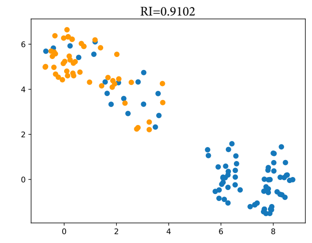
(c) VaDER



(d) CRLI w/o adv



(e) CRLI w/o joint optimizing



(f) CRLI

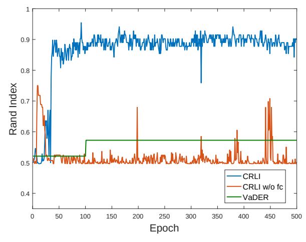
Figure 2: The learned representations of SOTA deep time series clustering methods on dataset *Vote*.

Visualization Analysis

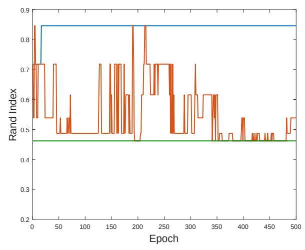
Through visualization, we analyze the benefits of the cluster-friendly representations and illustrate the convergence of our model. We use t-SNE (Der Maaten and Hinton 2008) to map the learned representations into $2D$ and plot it.

Comparison of the Learned Representations We visualize the learned representations of several SOTA deep time series clustering methods (including DTC (Madiraju et al. 2018), DTCR (Ma et al. 2019), VaDER (de Jong et al. 2019), CRLI, and its two ablations). For the representations of DTC and DTCR, we firstly apply the BRITS (Cao et al. 2018) to complete the data, and then use these two methods to obtain the representations. As shown in Figure 2, although all methods learned the clustering prototype except VaDER, CRLI performs better in terms of smaller intra-class and larger inter-class distances. Comparing CRLI with its two ablations, it can be seen that the joint training strategy that simultaneously optimizes the imputation and clustering, and the adversarial strategy that alleviates errors propagation from imputation to clustering, boost the performance of CRLI.

Convergence Analysis We investigate the convergence analysis of VaDER, CRLI, and its ablation (CRLI w/o fc) by reporting the test RI score with increasing training epochs. As shown in Figure 3, on the dataset *Vote*, VaDER's RI score no longer improves after the 100th epoch, and there is no change at all on the *Ali-v1*, which implies that it may suffer overfitting. The sizes of the training sets of these datasets are less than 400. In particular, *Ali-v1* has only 29 training samples. Therefore, there is not enough data to train VaDER since it's a generative clustering framework. On the other hand, due to adversarial training in CRLI, the training process is unstable at first, but eventually, it is relatively stable after the 300th epoch. As mentioned in **Proposed Method**, to alleviate the instability introduced by jointly optimizing the imputation and clustering process, we do not directly apply the clustering loss term to the representations of the generator, but apply the it after feeding the representations into a fully connected layer. With the fully connected layer, CRLI is more stable and has better performance.



(a) *Vote*



(b) *Ali-v1*

Figure 3: Convergence comparison of two datasets. The x-axis denotes the number of training epochs, and the y-axis denotes the corresponding test RI score.

Analysis of the imputed data

We experimentally investigate the characteristics of the imputed data to see why our joint training method can achieve better clustering performance. We firstly randomly drop the values of the first 20 data sets in the UCR time series data set archive (Chen et al. 2015) with a missing rate of 20%. Then, we adapt CRLI and BRITS methods to complete the data set, and use Root Mean Square Error (RMSE) to measure the imputation error. Finally, to compare the impact of the imputed data on the clustering accuracy, we apply Dynamic Time Warping(DTW) (Berndt and Clifford 1994) + K-means (Hartigan and Wong 1979) to the imputed complete data and the raw data respectively to record the RI. The experiment was repeated 5 times, and we report the average performance and standard deviation.

As shown in Table 3, although CRLI is not as good as BRITS in imputation, its clustering performance is significantly better ($p < 0.05$ level) than BRITS, and even better than DTW+K-means on the raw data. It indicates that the gradient backpropagated from clustering process enable the imputation process to impute more discriminative values for clustering.

Metric	RMSE		RI		
Method	BRITS	CRLI	Raw_data	BRITS	CRLI
50words	**0.67(0.19)**	1.44(0.29)	0.941(0.01)	**0.952(0.00)**	**0.952(0.00)**
Adiac	**0.93(0.47)**	6.78(2.14)	0.921(0.02)	**0.937(0.00)**	0.934(0.00)
ArrowHead	**1.36(0.50)**	2.01(0.43)	0.556(0.00)	0.503(0.01)	**0.575(0.01)**
Beef	**2.12(1.23)**	2.90(0.99)	**0.651(0.01)**	0.648(0.01)	**0.651(0.00)**
BeetleFly	**1.33(0.39)**	3.03(0.16)	0.479(0.02)	0.516(0.01)	**0.528(0.02)**
BirdChicken	**1.94(1.15)**	3.88(0.85)	**0.474(0.00)**	**0.474(0.00)**	**0.474(0.00)**
Car	**1.24(0.73)**	3.04(1.12)	**0.702(0.05)**	0.678(0.02)	0.698(0.01)
CBF	3.37(0.20)	**3.21(0.08)**	0.704(0.01)	0.699(0.01)	**0.710(0.01)**
Chlorine	**1.15(0.08)**	3.10(0.17)	**0.528(0.00)**	**0.528(0.00)**	**0.528(0.00)**
CinC	**2.19(0.80)**	3.96(0.24)	0.693(0.01)	0.685(0.01)	**0.695(0.00)**
Coffee	**1.11(0.36)**	4.43(1.21)	0.484(0.00)	0.484(0.00)	**0.506(0.01)**
Computers	**5.80(0.37)**	6.40(1.03)	0.502(0.01)	**0.510(0.00)**	0.509(0.00)
Cricket_X	**3.52(0.25)**	4.07(0.60)	0.855(0.01)	0.855(0.00)	**0.859(0.00)**
Cricket_Y	**2.89(0.34)**	3.05(0.22)	0.858(0.01)	0.856(0.00)	**0.861(0.00)**
Cricket_Z	**3.53(0.32)**	4.15(0.89)	0.858(0.01)	0.855(0.00)	**0.863(0.00)**
Diatom	**1.64(0.68)**	2.00(0.38)	0.926(0.00)	0.925(0.00)	**0.927(0.00)**
DPO.AgeGroup	**0.59(0.25)**	1.04(0.04)	**0.764(0.00)**	0.763(0.00)	0.763(0.00)
DPO.Correct	**0.58(0.28)**	0.92(0.05)	**0.505(0.00)**	0.504(0.00)	0.504(0.00)
DP.TW	**0.85(0.50)**	1.07(0.09)	0.774(0.02)	**0.777(0.00)**	0.775(0.00)
Earthquakes	13.02(0.13)	**10.49(0.37)**	0.499(0.03)	0.504(0.01)	**0.511(0.01)**
Average	2.492	3.548	0.684	0.683	0.691
AVG RANK	1.1	1.9	2.2	2.3	1.4
***p*-value**	-	-	1.05E-02	3.19E-02	-

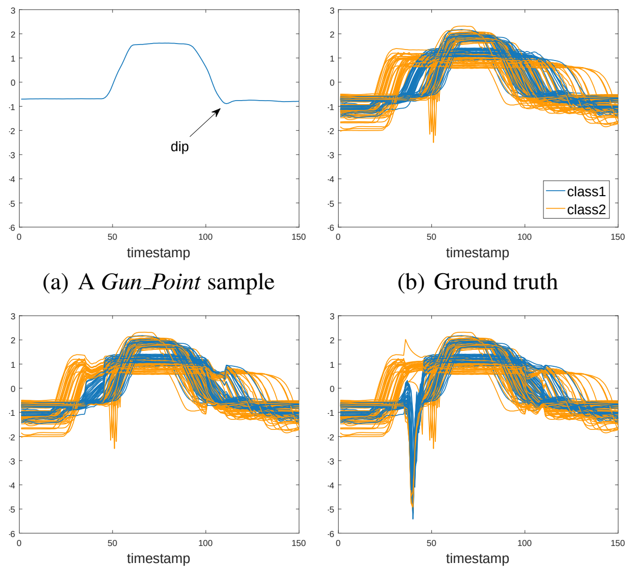
Table 3: RMSE & clustering RI comparison on the first 20 synthetic incomplete data sets of UCR archive. (the values in parentheses present standard deviations)

To explore the effect of the clustering process on imputation, we use the famous *Gun_Point* (Lines et al. 2012) dataset to visualize the imputation curve. A *Gun_Point* example denotes the movement of centroid of the actor's right hand when the actor lifts his hand, points to a target for around a second and then lowers his hand. The dataset consists of two classes, denoting whether the actor uses a replicate gun or his index finger to point the target. The most discriminative sub-sequence of *Gun_Point* is considered at the period when actors lower their hands, since usually there will be a dip if actors hold no gun (Lines et al. 2012) (Figure 4 (a)). Hence, we drop the values of the $[100, 110]$ period (an important time period), and the $[35, 45]$ period (a less important period). We then apply CRLI and BRITS to impute such a missing dataset and record clustering performance on imputed dataset and RMSE in both missing period respectively. We train both methods on the dropped train set and evaluate on the dropped test set. We apply DTW + K-means on the imputed test set to record the RI.

As shown in Table 4 and Figure 4, CRLI achieves comparable RMSE at the important sub-sequence, while RMSE in the less important part is much higher than BRITS, resulting similar curves in $[100, 110]$ that close to the ground truth and discriminative curves in $[35, 45]$ that separate sequences of different classes. The results reveal that CRLI not only can identify the important sub-sequence and imputes realistic values in such period to maintain the discrimination of original data, but also learns to impute discriminative values in the less important part to make it more separable. RI results on imputed dataset and complete dataset indicates that the CRLI-imputed sequences are more cluster-friendly.

Dataset	Total_RMSE	RMSE in $[35, 45]$	RMSE in $[100, 110]$	RI
Ground_truth	0	0	0	0.50
BRITS-imputed	**0.85**	**0.57**	0.63	0.50
CRLI-imputed	5.22	5.18	**0.62**	**0.62**

Table 4: RMSE & clustering RI on imputed *Gun_Point* dataset



(a) A *Gun_Point* sample (b) Ground truth

(c) BRITS-Imputed (d) CRLI-Imputed

Figure 4: Comparison of imputed *Gun_Point* dataset

Conclusions

We proposed the CRLI method to address challenging issues in real-world time-series applications, namely, clustering incomplete time series. CRLI jointly optimizes the imputation and clustering process, enabling it to learn cluster-friendly representations for time-series with missing values without pre-imputation. Moreover, we employ an adversarial strategy to prevent error propagation from the imputation to clustering. The effectiveness of CRLI is verified on eight real incomplete time-series datasets. We provide a visualization analysis to demonstrate the effectiveness of the learned representations and show the convergence of the proposed model. We also provide quantitative and qualitative analysis to reveal that joint training can impute more discriminative values to improve clustering performance.

Acknowledgments

We thank the anonymous reviewers for their helpful feedbacks and the authors of related work for providing their valuable datasets. The work described in this paper was partially funded by the National Natural Science Foundation of China (Grant Nos. 61502174, 61872148), the Natural Science Foundation of Guangdong Province (Grant Nos. 2017A030313355, 2019A1515010768), the Guangzhou Science and Technology Planning Project (Grant Nos. 201704030051, 201902010020), the Key RD Program of Guangdong Province (No. 2018B010107002), and the Fundamental Research Funds for the Central Universities.

References

Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; et al. 2016. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, 265–283.

Aghabozorgi, S.; Shirkhorshidi, A. S.; and Wah, T. Y. 2015. Time-series clustering–A decade review. *Information Systems* 53: 16–38.

Alizadeh, A. A.; Eisen, M. B.; Davis, R. E.; Ma, C.; Lossos, I. S.; Rosenwald, A.; Boldrick, J. C.; Sabet, H.; Tran, T.; Yu, X.; et al. 2000. Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature* 403(6769): 503–511.

Azoff, E. M. 1994. *Neural network time series forecasting of financial markets*. John Wiley & Sons, Inc.

Berndt, D. J.; and Clifford, J. 1994. Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, 359–370. Seattle, WA, USA:.

Bianchi, F. M.; Mikalsen, K. Ø.; and Jenssen, R. 2017. Learning compressed representations of blood samples time series with missing data. In *European Symposium on Artificial Neural Networks*.

Cao, W.; Wang, D.; Li, J.; Zhou, H.; Li, L.; and Li, Y. 2018. Brits: Bidirectional recurrent imputation for time series. In *Advances in Neural Information Processing Systems*, 6775–6785.

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): 1–12.

Cheema; and J., R. 2014. A Review of Missing Data Handling Methods in Education Research. *Review of Educational Research* 84(4): 487–508.

Chen, X.; Cheung, S. T.; So, S.; Fan, S. T.; Barry, C.; Higgins, J.; Lai, K.-M.; Ji, J.; Dudoit, S.; Ng, I. O.; et al. 2002. Gene expression patterns in human liver cancers. *Molecular Biology of the Cell* 13(6): 1929–1939.

Chen, Y.; Keogh, E.; Hu, B.; Begum, N.; Bagnall, A.; Mueen, A.; and Batista, G. 2015. The UCR Time Series Classification Archive. www.cs.ucr.edu/~eamonn/time_series_data/.

Cho, K.; Van Merrienboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Conference on Empirical Methods in Natural Language Processing*, 1724–1734.

de Jong, J.; Emon, M. A.; Wu, P.; Karki, R.; Sood, M.; Godard, P.; Ahmad, A.; Vrooman, H.; Hofmann-Apitius, M.; and Fröhlich, H. 2019. Deep learning for clustering of multivariate clinical patient trajectories with missing values. *GigaScience* 8(11): giz134.

de Souto, M. C.; Costa, I. G.; de Araujo, D. S.; Ludermir, T. B.; and Schliep, A. 2008. Clustering cancer gene expression data: a comparative study. *BMC Bioinformatics* 9(1): 497.

De Souto, M. C.; Jaskowiak, P. A.; and Costa, I. G. 2015. Impact of missing data imputation methods on gene expression clustering and classification. *BMC Bioinformatics* 16(1): 64.

Demsar, J. 2006. Statistical Comparisons of Classifiers over Multiple Data Sets. *Journal of Machine Learning Research* 7: 1–30.

Der Maaten, L. V.; and Hinton, G. E. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9: 2579–2605.

Dua, D.; and Graff, C. 2017. UCI Machine Learning Repository. URL http://archive.ics.uci.edu/ml.

Ferreira, L. N.; and Zhao, L. 2016. Time series clustering via community detection in networks. *Information Sciences* 326: 227–242.

Guo, X.; Gao, L.; Liu, X.; and Yin, J. 2017. Improved deep embedded clustering with local structure preservation. In *International Joint Conference on Artificial Intelligence*, 1753–1759.

Hartigan, J. A.; and Wong, M. A. 1979. Algorithm AS 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 28(1): 100–108.

Hathaway, R. J.; and Bezdek, J. C. 2001. Fuzzy c-means clustering of incomplete data. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 31(5): 735–744.

Jiang, Z.; Zheng, Y.; Tan, H.; Tang, B.; and Zhou, H. 2017. Variational deep embedding: An unsupervised and generative approach to clustering. In *International Joint Conference on Artificial Intelligence*, 965–1972.

Kingma, D. P.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations*.

Li, D.; Gu, H.; and Zhang, L. 2010. A fuzzy c-means clustering algorithm based on nearest-neighbor intervals for incomplete data. *Expert Systems with Applications* 37(10): 6942–6947.

Li, S. C.; Jiang, B.; and Marlin, B. M. 2019. MisGAN: Learning from Incomplete Data with Generative Adversarial Networks. In *International Conference on Learning Representations*.

Li, T.; Zhang, L.; Lu, W.; Hou, H.; Liu, X.; Pedrycz, W.; and Zhong, C. 2017. Interval kernel fuzzy c-means clustering of incomplete data. *Neurocomputing* 237: 316–331.

Liang, Y.; Diehn, M.; Watson, N.; Bollen, A. W.; Aldape, K. D.; Nicholas, M. K.; Lamborn, K. R.; Berger, M. S.; Botstein, D.; Brown, P. O.; et al. 2005. Gene expression profiling reveals molecularly and clinically distinct subtypes of glioblastoma multiforme. *Proceedings of the National Academy of Sciences* 102(16): 5814–5819.

Lines, J.; Davis, L. M.; Hills, J.; and Bagnall, A. 2012. A shapelet transform for time series classification. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 289–297.

Luo, Y.; Cai, X.; Zhang, Y.; Xu, J.; et al. 2018. Multivariate time series imputation with generative adversarial networks. In *Advances in Neural Information Processing Systems*, 1596–1607.

Ma, Q.; Li, S.; and Cottrell, G. 2020. Adversarial Joint-Learning Recurrent Neural Network for Incomplete Time Series Classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 1–1. doi:10.1109/TPAMI.2020.3027975.

Ma, Q.; Zheng, J.; Li, S.; and Cottrell, G. W. 2019. Learning Representations for Time Series Clustering. In *Advances in Neural Information Processing Systems*, 3781–3791.

Madiraju, N. S.; Sadat, S. M.; Fisher, D.; and Karimabadi, H. 2018. Deep Temporal Clustering: Fully unsupervised learning of time-domain features. *ArXiv Preprint ArXiv:1802.01059* .

Paparrizos, J.; and Gravano, L. 2015. K-shape: Efficient and accurate clustering of time series. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, 1855–1870.

Rand, W. M. 1971. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association* 66(336): 846–850.

Rubin, D. B. 1976. Inference and missing data. *Biometrika* 63(3): 581–592.

Rumelhart, D. E.; Hinton, G. E.; and Williams, R. J. 1986. Learning Representations by Back Propagating Errors. *Nature* 323(6088): 533–536.

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. In *2012 Computing in Cardiology*, 245–248. IEEE.

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).

Xie, J.; Girshick, R.; and Farhadi, A. 2016. Unsupervised deep embedding for clustering analysis. In *International Conference on Machine Learning*, 478–487.

Yoon, J.; Jordon, J.; and Der Schaar, M. V. 2018. GAIN: Missing Data Imputation using Generative Adversarial Nets. In *International Conference on Machine Learning*, 5675–5684.

Zha, H.; He, X.; Ding, C.; Gu, M.; and Simon, H. D. 2002. Spectral relaxation for k-means clustering. In *Advances in Neural Information Processing Systems*, 1057–1064.

Zhang, D.-Q.; and Chen, S.-C. 2003. Clustering incomplete data using kernel-based fuzzy c-means algorithm. *Neural Processing Letters* 18(3): 155–162.

Zhang, Q.; Wu, J.; Zhang, P.; Long, G.; and Zhang, C. 2018. Salient subsequence learning for time series clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41(9): 2193–2207.