Multivariate Time Series as Images: Imputation Using Convolutional Denoising Autoencoder

Abdullah Al Safi, Christian Beyer(✉), Vishnu Unnikrishnan, and Myra Spiliopoulou

Fakultät für Informatik, Otto-von-Guericke-Universität, Postfach 4120, 39106 Magdeburg, Germany
`abdullah.safi@st.ovgu.de`,
{`christian.beyer,vishnu.unnikrishnan,myra`}`@ovgu.de`

Abstract. Missing data is a common occurrence in the time series domain, for instance due to faulty sensors, server downtime or patients not attending their scheduled appointments. One of the best methods to impute these missing values is *Multiple Imputations by Chained Equations (MICE)* which has the drawback that it can only model linear relationships among the variables in a multivariate time series. The advancement of deep learning and its ability to model non-linear relationships among variables make it a promising candidate for time series imputation. This work proposes a modified Convolutional Denoising Autoencoder (CDA) based approach to impute multivariate time series data in combination with a preprocessing step that encodes time series data into 2D images using Gramian Angular Summation Field (GASF). We compare our approach against a standard feed-forward Multi Layer Perceptron (MLP) and MICE. All our experiments were performed on 5 UEA MTSC multivariate time series datasets, where 20 to 50% of the data was simulated to be missing completely at random. The CDA model outperforms all the other models in 4 out of 5 datasets and is tied for the best algorithm in the remaining case.

Keywords: Convolutional Denoising Autoencoder · Gramian Angular Summation Field · MICE · MLP. · Imputation · Time series

1 Introduction

Time series data resides in various domains of industries and research fields and is often corrupted with missing data. For further use or analysis, the data often needs to be complete, which gives the rise to the need for imputation techniques with enhanced capabilities of introducing least possible error into the data. One of the most prominent imputation methods is MICE which uses iterative regression and value replacement to achieve state-of-the-art imputation quality but has the drawback that it can only model linear relationships among variables (dimensions).

In past few years, different deep learning architectures were able to break into different problem domains, often exceeding previously achieved performances by other algorithms [7]. Areas like speech recognition, natural language processing, computer vision, etc. were greatly impacted and improved by deep learning architectures. Deep learning models have a robust capability of modelling latent representation of the data and non-linear patterns, given enough training data. Hence, this work presents a deep learning based imputation model called Convolutional Denoising Autoencoder (CDA) with altered convolution and pooling operations in Encoder and Decoder segments. Instead of using the traditional steps of convolution and pooling, we use deconvolution and upsampling which was inspired by [5]. The time series to image transformation mechanisms proposed in [12] and [13] were inherited as a preprocessing step as CDA models are typically designed for images. As rival imputation models, Multiple Imputation by Chained Equations (MICE) and a Multi Layer Perceptron (MLP) based imputation were incorporated.

2 Related Work

Three distinct types of missingness in data were identified in [8]. The first one is *Missing Completely At Random (MCAR)*, where the missingness of the data does not depend on itself or any other variables. In *Missing At Random (MAR)* the missing value depends on other variables but not on the variable where the data is actually missing and in *Missing Not At Random (MNAR)* the missingness of an observation depends on the concerned variable itself. All the experiments in this study were carried out on MCAR missingness as reproducing MAR and MNAR missingness can be challenging and hard to distinguish [5].

Multiple Imputation by Chained Equations (MICE) has secured its place as a principal method for imputing missing data [1]. Costa et al. in [3] experimented and showed that MICE offered the better imputation quality than a Denoising Autoencoder based model for several missing percentages and missing types.

A novel approach was proposed in [14], incorporating General Adversarial Networks (GAN) to perform imputations, thus authors named it Generative Adversarial Imputation Nets (GAIN). The approach imputed significantly well against some state-of-the-art imputation methods including MICE. An Autoencoder based approach was proposed in [4], which was compared against an Artificial Neural Network (NN) model on MCAR missing type and several missing percentages. The proposed model performed well against NN. A novel Denoising Autoencoder based imputation using partial loss (DAPL) approach was presented in [9], where different missing data percentages and MCAR missing type were simulated in a breast cancer dataset. The comparisons incorporated statistical, machine learning based approaches and standard Denoising Autoencoder (DAE) model where DAPL outperformed DAE and all the other models. An MLP based imputation approach was presented for MCAR missingness in [10] and also outperformed other statistical models. A Convolutional Denoising Autoencoder model which did not impute missing data but denoised audio signals was presented in [15]. A Denoising Autoencoder with more units in the encoder layer than input layer was presented in [5] and achieved good imputation results against MICE. Our work was inspired from both of these works which is why we combined the two approaches into a Convolutional Denoising Autoencoder which maps input data into a higher subspace in the Encoder.

3 Methodology

In this section we first describe how we introduce missing data in our datasets, then we show the process used to turn multivariate time series into images which is required by one of our imputation methods and finally we introduce the imputation methods which were compared in this study.

3.1 Simulating Missing Data

Simulating missing data is a mechanism of artificially introducing unobserved data into a complete time series dataset. Our experiment incorporated 20%, 30%, 40% and 50% of missing data and the missing type was MCAR. Introducing MCAR missingness is quite a simple approach as it does not depend on observed or unobserved data. Many studies assume MCAR missing type quite often when there is no concrete evidence of missingness type [6]. In this experimental framework, values at randomly selected indices were erased from randomly selected variables which simulated MCAR missingness of different percentages.

3.2 Translating Time Series into Images

A novel approach of encoding time series data into various types of images using Gramian Angular Field (GAF) was presented in [12] to improve classification and imputation. One of the variants of GAF was Gramian Angular Summation Field (GASF), which comprised of multiple steps to perform the encoding. First, the time series is scaled within $[-1, 1]$ range.

$$x'_i = \frac{(x_i - Max(X)) + (x_i - Min(X))}{Max(X) - Min(X)} \tag{1}$$

Here, x_i is a specific value at timepoint i where x'_i is derived by scaling and X is the time series. The time series is scaled within $[-1, 1]$ range in order to be represented as polar coordinates achieved by applying angular cosine.

$$\theta_i = arccos(x'_i)\{-1 <= x'_i <= 1, x'_i \in X\} \tag{2}$$

The polar encoded time series vector is then transformed into a matrix. If the length of the time series vector is n, then the transformed matrix is of shape $(n \times n)$.

$$GASF_{i,j} = cos(\theta_i + \theta_j) \tag{3}$$

The GASF represents the temporal features in the form of an image where the timestamps move along top-left to bottom-right, thereby preserving the time factor in the data. Figure 1 shows the different steps of time series to image transformation.

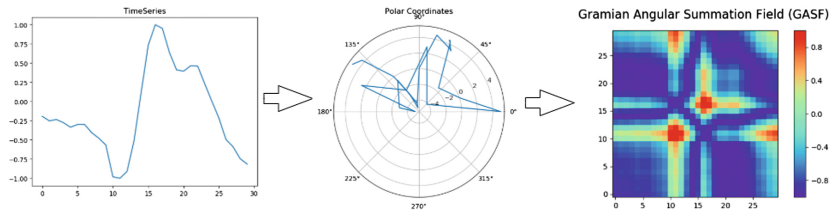

Fig. 1. Time series to image transformation

The methods of encoding time series into images described in [12] were only applicable for univariate time series. The GASF transformation generates one image for one time series dimension and thus it is possible to generate multiple images for multivariate time series. An approach which vertically stacked images transformed from different variables was presented in [13], see Fig. 2. The images were grayscaled and the different orders of vertical stacking (ascending, descending and random) were examined by performing a statistical test. The stacking order did not impact classification accuracy.

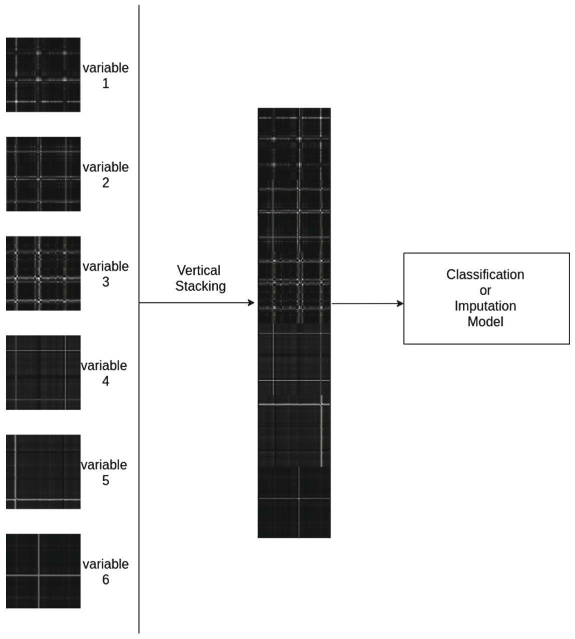

Fig. 2. Vertical stacking of images transformed from different variables

3.3 Convolutional Denoising Autoencoder

Autoencoder is a very popular unsupervised deep learning model frequently found in different application areas. Autoencoder is unsupervised in fashion and reconstructs the original input by discovering robust features in the hidden layer representation. The latent representation of high dimensional data in the hidden layer contributes in reconstructing the original data. The architecture of Autoencoder consists of two principal segments named Encoder and Decoder. The Encoder usually compresses the original representation of the data into lower dimension. The Decoder decodes the low dimensional representation of the input back into its original dimensional representation.

$$Encoder(x^n) = s(x^n W_E + b_E) = x^d \tag{4}$$

$$Decoder(x^d) = s(x^d W_D + b_D) = x^n \tag{5}$$

Here, x^n is the original input with n dimensions. s is any non-linear activation function, W is weight and b is bias.

Denoising Autoencoder model is an extension of Autoencoder where the input is reconstructed from a corrupted version of it. There are different ways of adding corruption, such as Gaussian noise, setting some values to zero etc. The noisy input is fed as input and the model minimizes the loss between the clean input and corrupted reconstructed input. The objective function looks as follows

$$RMSE(X, X')\frac{1}{n}\sqrt{|X_{clean} - X'_{reconstructed}|^2} \tag{6}$$

Convolutional Denoising Autoencoder (CDA) incorporates convolution operation which is ideally performed in Convolutional Neural Networks (CNN). CNN is a methodology, where the layers of perceptrons are replaced by convolution layers and convolution operation is performed on the data. Convolution is defined as multiplication of two function within a finite or infinite range, where two functions refer to input data (e.g. Image) and a fixed size kernel consecutively. The kernel traverses through the input space to generate feature maps. The feature maps consist of important features of the data. The multiple features are pooled, preserving important features.

The combination of convoluted feature maps generation and pooling is performed in the Encoder layer of CDA where the corrupted version of the input is fed into the input layer of the network. The Decoder layer performs Deconvoluiont and Upsampling which decompresses the output coming from Encoder layer back into the shape of input data. The loss between reconstructed data and clean data is minimized. In this work, the default architecture of CDA is tweaked in the favor of imputing multivariate time series data. Deconvolution and Upsampling were performed in the Encoder layer and Convolution and Maxpooling was performed in Decoder layer. The motivation behind this specific tweaking came from [5], where a Denoising Autoencoder was designed with more hidden units in the Encoder layer than input layer. The high dimensional representation in Encoder layer created additional feature which was the contributor of data recovery.

3.4 Competitor Models

Multiple Imputation by Chained Equations (MICE): MICE, which is sometimes addressed as fully conditional specification or sequential regression multiple imputation, has emerged in the statistical literature as the principal method of addressing missing data [1]. MICE creates multiple versions of the imputed datasets through multiple imputation technique.

The steps for performing MICE are the following:

– A simple imputation method is performed across the time series (mean, mode or median). The missing time points are referred as "placeholders".
– If there are total m variables having missing points, then one of the variables are set back to missing state. The variable with "missing state" label is considered as dependent variable and other variables are considered as predictors.
– A regression is performed over these settings and "missing state" variable is imputed. Different regressions are supported in this architecture but since the dataset only contains continuous values, linear, ridge or lasso regression are chosen.
– The remaining $m - 1$ "missing state" are regressed and imputed by the same way. Once all the m variables are imputed, one iteration is completed. More iterations are performed and the imputations are placed in the time series in each iteration.
– The number of iterations can be determined by observing whether coefficients of the regression model are converged or not.

According to the experimental setup of our work, MICE had three different regression supports, namely Linear, Ridge and Lasso regression.

Multi Layer Perceptron (MLP) Based Imputation: The imputation mechanism of MLP is inspired by the MICE algorithm. Nevertheless, MLP based imputation models do not perform the chained or multiple imputations like MICE but improve the quality of imputation over several epochs as stochastic gradient descent optimizes the weights and biases per epoch. A concrete MLP architecture was described in literature [10] which was a three layered MLP with the hyperbolic tangent activation function in the hidden layer and the identity function (linear) as the activation function for the output layer. The train and test split were slightly different, where training set and test set consisted of both observed and unobserved data.

The imputation process of MLP model in our work is similar to MICE but the non-linear activation function of MLP facilitates finding complex non-linear patterns. However, the imputation of a variable is performed only once, in contrast to the multiple iterations in MICE.

4 Experiments

In this section we present the used datasets, the preprocessing steps that were conducted before training, the chosen hyperparameters and our evaluation method. Our complete imputation process for the CDA model is depicted in Fig. 3. The process for the competitors is the same except that corrupting the training data and turning the time series into images is not being done.

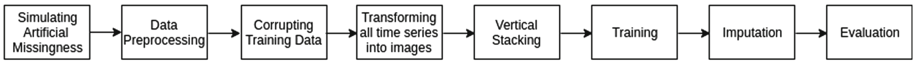

Fig. 3. Experiment steps for the CDA model

4.1 Datasets and Data Preprocessing

Our experiments were conducted on 5 time series datasets from the UEA MTSC repository [2]. Each dataset in UEA time series archive has training and test splits and specific number of dimensions. Each training or test split represents a time series. The table below presents all the relevant structural details (Table 1).

Table 1. A structural summary of the 5 UEA MTSC dataset

Dataset name	Number of series	Dimensions	Length	Classes
ArticularyWordRecognition	275	9	144	25
Cricket	108	6	1197	12
Handwriting	150	3	152	26
StandWalkJump	12	4	2500	3
UWaveGestureLibrary	120	3	315	8

The Length column of the table denotes the length of each time series. In our framework, each time series was transformed into images. The number of time series for any of the datasets was not very high in number. As we had selected a deep learning model for imputation, such low number of samples could cause overfitting. Experiments showed us that the default number of time series could not perform well. Therefore, the main idea was to increase the number of time series by splitting them into multiple parts and reducing their corresponding lengths. This modification facilitated us by introducing more patterns for learning which aided in imputation. The final lengths chosen were those that yielded the best results. The table below presents the modified number of time series and lengths for each dataset (Table 2).

Table 2. Modified number of time series and lengths

Dataset name	Number of series	Dimension	Length
ArticularyWordRecognition	6600	9	6
Cricket	6804	6	19
Handwriting	1200	3	19
StandWalkJump	3000	4	10
UWaveGestureLibrary	1800	3	21

The evaluation of the imputation models require a complete dataset and the corresponding incomplete dataset. Therefore, artificial missingness was introduced at different percentages (20%, 30%, 40% and 50%) into all the datasets. After simulating artificial missingness, each dataset has an observed part, which contains all the time series segments where no variables are missing and an unobserved part, where at least one variable is missing. After simulating artificial missingness, each dataset had an observed and unobserved split and the observed data was further processed for training. As CDA models learn denoising from a corrupted version of the input, we introduced noise by discarding a certain amount of values for each observed case from specific variables and replacing them by the mean of the corresponding variables. A higher amount of noise has seen to be contributing more in learning dependencies of different variables, which leads to denoising of good quality [11]. The variables selected for adding noise were the same variables having missing data in unobserved data. Different amount of noise was examined but 90% noise lead to good results. Unobserved data was also mean imputed as the CDA model would apply the denoising technique on the "mean-noise" for imputation. So the CDA learns to deal with "mean-noise" on the observed part and is then applied on mean imputed unobserved part to create the final imputation.

The next step was to perform time series to image transformation where, all the observed and unobserved chunks were rescaled between -1 to 1 using min-max scaling. Rescaled data was further transformed into polar coordinates and then GASF encoded image was achieved for each dimension. Multiple images referring to multiple variables were vertically aggregated. Finally, both observed and unobserved splits consisted their own set of images.

Note that, the following data preprocessing was performed only for CDA based imputation models. The competitor models imputed using the raw format of the data.

4.2 Model Architecture and Hyperparameters

Our Model architecture was different from a general CDA, where the Encoder layer incorporates Deconvolution and Upsampling operations and the Decoder layer incorporates Convolution and Maxpooling operations. The Encoder and Decoder both have 3 layers. The table below demonstrates the structure of the imputation model (Table 3).

Table 3. The architecture of CDA based imputation model

	Operation	Layer name	Kernel size	Number of feature maps
Encoder	Upsampling	up_0	(2, 2)	–
	Deconvolution	deconv_0	(5, 5)	64
	Upsampling	up_1	(2, 2)	–
	Deconvolution	deconv_1	(7, 7)	64
	Upsampling	up_2	(2, 2)	–
	Deconvolution	deconv_2	(5, 6)	128
Decoder	Convolution	conv_0	(5, 6)	128
	Maxpool	pool_0	(2, 2)	–
	Convolution	conv_1	(7, 7)	64
	Maxpool	pool_1	(2, 2)	–
	Convolution	conv_2	(5, 5)	64
	Maxpool	pool_2	(2, 2)	–

Hyperparameter specification was achieved by performing random search on different random combinations of hyperparameter values and the root mean square error (RMSE) was used to decide on the best combination. The random search allowed us to avoid the exhaustive searching unlike grid search. Applying random search, we selected stochastic gradient descent (SGD) as optimizer, which backpropagates the error to optimize the weights and biases. The number of epochs was 100 and the batch size was 16.

4.3 Competitor Model's Architecture and Hyperparameters

As competitor models, MICE and MLP based imputation models were selected. MLP based model had 3 hidden layers and number of hidden units were 2/3 of the number of input units in each layer. The hyperparameters for both of the models were tuned by using random search.

Hyperbolic Tangent Function was selected as activation function with a dropout of 0.3. Stochastic Gradient Descent operated as optimizer for 150 epochs and with a batch size of 20.

MICE based imputation was demonstrated using Linear, Ridge and Lasso regression and 10 iterations were performed for each of them.

4.4 Training

Based on the preprocessed data and model architecture described above, the training is started. L2 regularization was used with weight of 0.01 and stochastic gradient descent was used as the optimizer which outperformed Adam and Adagrad optimizers. The whole training process was about learning to minimize loss between the clean and corrupted data so that it can be applied on the unobserved data (noisy data after mean imputation) to perform imputation. The training and validation split was 70% and 30%. Experiments show that, the training and validation loss was saturated approximately after 10–15 epochs, which was observed for most of the cases.

The training was conducted on a machine with Nvidia RTX 2060 with RAM memory of 16 GB. The programming language for the training and all the steps above was Python 3.7 and the operating system was Ubuntu 16.04 LTS.

4.5 Evaluation Criteria

As all the time series dataset contain continuous numeric values, Root Mean Square Error (RMSE) was selected for evaluation. In out experimental setup, RMSE is not calculated on overall time series but only missing data points are taken into account to be compared with ground truth while calculating RMSE $RMSE = \sqrt{\frac{1}{m}\Sigma_{i=1}^{m}(x_i - x_i')^2}$. Where m is the total number of missing time points and I represents all the indices of missing values across the time series.

5 Results

Our proposed CDA based imputation model was compared with MLP and three different versions of MICE, each using a different type of regression. Figure 4 presents the RMSE values for 20%, 30% 40% and 50% missingness.

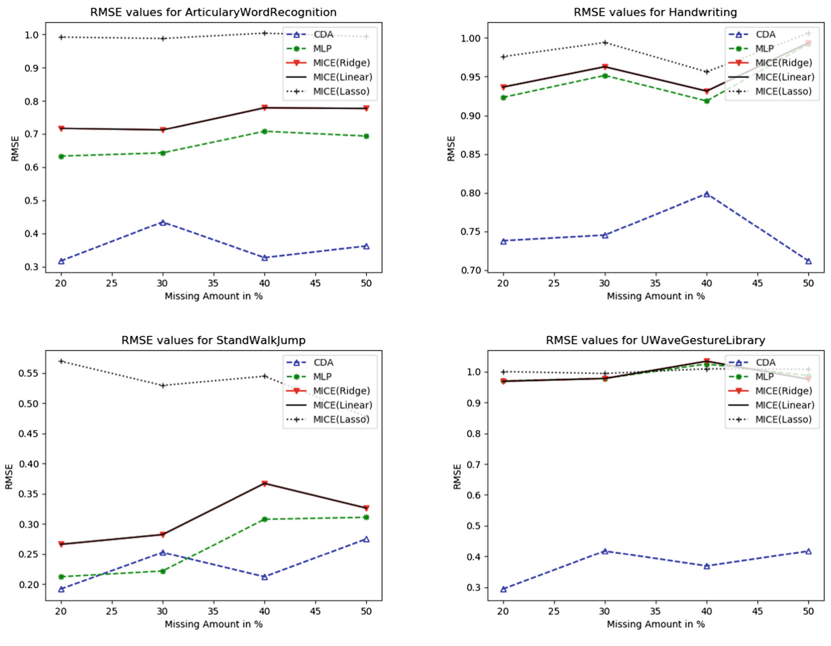

Fig. 4. RMSE plots for different missing proportions

The RMSE values for the CDA based model are the lowest at every percentage of missingness on the *Handwriting, ArticularyWordRecognition, UWaveGestureLibrary and Cricket* dataset. The depiction of the results on the *Cricket* dataset is omitted due to space limitations. Unexpectedly, in *StandWalkJump* dataset the performance of MLP and CDA model are very similar, and MLP is even better at 30% missingness. MICE (Linear) and MICE (Ridge) are identical in imputation for all the datasets. MICE (Lasso) performed worst of all the models, which implies that changing the regression type could potentially cause an impact on the imputation quality. The MLP model beat all the MICE models but was outperformed by the CDA model in at least for 80% of the cases.

6 Conclusion

In this work, we introduce an architecture of a Convolutional Denoising Autoencoder (CDA) adapted for multivariate time series imputation which inflates the size of the hidden layers in the Encoder instead of reducing them. We also employ a preprocessing step that turns the time series into 2D images based on Gramian Angular Summation Fields in order to make the data more suitable for our CDA. We compare our method against a standard Multi Layer Perceptron (MLP) and the state-of-the-art imputation method Multiple Imputations by Chained Equations (MICE) with three different types of regression (Linear, Ridge and Lasso). Our experiments were conducted on five different multivariate time series datasets, for which we simulated 20%, 30%, 40% and 50% missingness with data missing completely at random. Our results show that the CDA based imputation outperforms MICE on all five datasets and also beats the MLP on four datasets. On the fifth dataset CDA and MLP perform very similarly, but CDA is still better on four out of the five degrees of missingness. Additionally we present a preprocessing step on the datasets which manipulates the time series lengths to generate more training samples for our model which led to a better performance. The results show that the CDA model performs strongly against both linear and non-linear regression based imputation models. Deep Learning Networks are usually computationally more intensive than MICE but the imputation quality of CDA was convincing enough to be chosen over MICE or MLP based imputation.

In the future we plan to investigate also other types of missing data apart from *Missing Completely At Random (MCAR)* and want to incorporate more datasets as well as other deep learning based approaches for imputation.

Acknowledgments. This work is partially funded by the German Research Foundation, project OSCAR "Opinion Stream Classification with Ensembles and Active Learners". The principal investigators of OSCAR are Myra Spiliopoulou and Eirini Ntoutsi. Additionally, Christian Beyer is also partially funded by a PhD grant from the federal state of Saxony-Anhalt.

References

1. Azur, M.J., Stuart, E.A., Frangakis, C., Leaf, P.J.: Multiple imputation by chained equations: what is it and how does it work? Int. J. Methods Psychiatr. Res. **20**(1), 40–49 (2011)
2. Bagnall, A., et al.: The UEA multivariate time series classification archive, 2018. arXiv preprint arXiv:1811.00075 (2018)
3. Costa, A.F., Santos, M.S., Soares, J.P., Abreu, P.H.: Missing data imputation via denoising autoencoders: the untold story. In: Duivesteijn, W., Siebes, A., Ukkonen, A. (eds.) IDA 2018. LNCS, vol. 11191, pp. 87–98. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01768-2_8
4. Duan, Y., Lv, Y., Kang, W., Zhao, Y.: A deep learning based approach for traffic data imputation. In: 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), pp. 912–917. IEEE (2014)
5. Gondara, L., Wang, K.: MIDA: multiple imputation using denoising autoencoders. In: Phung, D., Tseng, V.S., Webb, G.I., Ho, B., Ganji, M., Rashidi, L. (eds.) PAKDD 2018, Part III. LNCS (LNAI), vol. 10939, pp. 260–272. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93040-4_21
6. Kang, H.: The prevention and handling of the missing data. Korean J. Anesthesiol. **64**(5), 402–406 (2013)
7. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436–444 (2015)
8. Little, R.J., Rubin, D.B.: Statistical Analysis with Missing Data, vol. 793. Wiley, Hoboken (2019)
9. Qiu, Y.L., Zheng, H., Gevaert, O.: A deep learning framework for imputing missing values in genomic data. bioRxiv, p. 406066 (2018)
10. Silva-Ramírez, E.L., Pino-Mejías, R., López-Coello, M., Cubiles-de-la Vega, M.D.: Missing value imputation on missing completely at random data using multilayer perceptrons. Neural Netw. **24**(1), 121–129 (2011)
11. Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P.A.: Extracting and composing robust features with denoising autoencoders. In: Proceedings of the 25th International Conference on Machine Learning, pp. 1096–1103 (2008)
12. Wang, Z., Oates, T.: Imaging time-series to improve classification and imputation. In: Twenty-Fourth International Joint Conference on Artificial Intelligence (2015)
13. Yang, C.L., Yang, C.Y., Chen, Z.X., Lo, N.W.: Multivariate time series data transformation for convolutional neural network. In: 2019 IEEE/SICE International Symposium on System Integration (SII), pp. 188–192. IEEE (2019)
14. Yoon, J., Jordon, J., Van Der Schaar, M.: Gain: missing data imputation using generative adversarial nets. arXiv preprint arXiv:1806.02920 (2018)
15. Zhao, M., Wang, D., Zhang, Z., Zhang, X.: Music removal by convolutional denoising autoencoder in speech recognition. In: 2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA), pp. 338–341. IEEE (2015)