DEGAIN as tool for Missing Data Imputation

Reza Shahbazian[1,†], Irina Trubitsyna[1,*,†]

[1]*Department of Informatics, Modeling, Electronics and System Engineering, University of Calabria, Italy*

Abstract

This paper focuses on the use of machine learning methods to deal with missing data imputation. In particular, we describe a generative adversarial network (GAN) based model called DEGAIN to estimate the missing values in the dataset. We present the performance of the presented method comparing the results with some of the existing methods on the publicly available Letter Recognition and SPAM datasets. The Letter dataset consists of 20000 samples and 16 input features and the SPAM dataset consists of 4601 samples and 57 input features. The results show that the proposed DEGAIN outperforms the existing ones in terms of root mean square error and Frechet Inception distance metrics.

Keywords

Machine Learning, Missing Data, Data Imputation, Generative Networks

1. Introduction

Incomplete information arises naturally in many database applications, such as data integration, data exchange, inconsistency management, data cleaning, ontological reasoning, and many others [1]. In some applications it is natural to allow the presence of incomplete data in the database and to support this circumstance using the proper approaches. Some recent proposal in this direction can be found in [2, 3, 4]. In particular, in relational databases a single null value is used to replace both missing and non-existent values, and data selection queries must take these circumstances into account[5, 6]. The missing data can be used as a feature for other decision makings such as error estimation. For instance the authors in [7] estimate the physical-layer transmission errors in cable broadband networks by considering the missing values. In one of recent researches, the authors propose a new multiple imputation MB (MimMB) framework for causal feature selection with missing data [8]. In different applications missing values cannot be tolerated and must be replaced by the concrete values. The available strategies in this case can be divided into traditional methods and Machine Learning (ML) based algorithms. Traditional methods include the Case Deletion, Mean, Median, Mode, Principal Component Analysis (PCA) and Singular Value Decomposition (SVD) [9, 10]. Machine Learning algorithms are categorized into supervised, semi-supervised and un-supervised [11]. Some of the main ML based methods are clustering algorithms [12], k-nearest neighbors (KNN) [13], Gaussian process regression

SEBD 2023: 31st Symposium on Advanced Database System, July 02–05, 2023, Galzignano Terme, Padua, Italy

*Corresponding author.

†These authors contributed equally.

Reza.Shahbazian@unical.it (R. Shahbazian); i.trubitsyna@dimes.unical.it (I. Trubitsyna)

0000-0002-2313-6002 (R. Shahbazian); 0000-0002-9031-0672 (I. Trubitsyna)

(GPR) [14], support vector machine (SVM) [15], long short-term memory (LSTM) [16], decision trees (DT) [17], random forests (RF) [18], auto-encoder (AE) [19], Expectation Maximization (EM) [20] and Generative adversarial networks (GAN) [21]. One of the main differences of the traditional methods and the machine learning based methods to handle the missing data is the capability of the optimization in ML. The ML based methods follow an optimization process. ML based methods can also extract the relation between data points, and therefore more precise estimation on the missing values.

Among the ML based algorithms, Generative Adversarial Networks (GANs) has attracted many researchers in recent years. The GAN has many applications, mostly with focus on generating synthetic data. The missing value estimation could be considered as synthetic data generation. Therefore, it is possible to use such networks in handling the missing data problem. However, the performance of the algorithms are dependent to many variables such as data type, for instance, if the data belongs to the category of image, clinical dataset, energy dataset or etc. Accordingly, many variations of the GAN are introduced in the literature.

In this paper we present the fundamentals of GAIN [21], a GAN-based algorithm and describe an improved version of GAIN called DEGAIN recently presented in [22]. In our method we improve the GAIN by applying the idea of network deconvolution [23]. Convolutional kernels usually re-learn redundant data because of the strong correlations in many image based datasets. The deconvolution strategy is proven to be effective on images, however, it has not been applied to the GAIN algorithm. DEGAIN is capable of removing the data correlations. We evaluate the performance of the proposed DEGAIN with the publicly available Letter Recognition dataset (Letter dataset, for short) and SPAM dataset. In particular, we use Root Mean Square Error (RMSE) and Frechet Inception Distance (FID) metrics and compare the performance of the proposed DEGAIN with the GAIN, the Auto-Encoder [24] and the MICE [25] algorithms.

The reminder of this paper is organized as follows. In Section 2, we introduce the system model with mathematical relations and describe the proposed DEGAIN algorithm. The performance evaluation is presented in Section 3. Finally, Section 4 concludes the paper.

2. From GAIN to DEGAIN

A GAN-based data imputing method, called Generative Adversarial Imputation Nets (GAIN), has been introduced in [21]. In GAIN the generator component G takes real data vectors, imputes the missing values conditioned on the really observed data, and gives a completed vector. Then, the discriminator component D gets a completed vector and tries to determine which element is really observed and which one is synthesized. To learn the desired distribution in the G component, some additional information is deployed for the discriminator D in the form of a hint vector. The hint vector shows the pieces of information about the missingness of the real data to the D component, and D concentrates its heed on the quality of imputation for particular missing values. In other words, the hint vector assures the G component to be learned for generating data based on the actual data distribution [21].

Convolution, which applies a kernel to overlapping sections shifted across the data, is a crucial operation in many Convolutional Neural Networks. Although utilizing CNN to synthesize images is not required in GANs, it is frequently done in order to learn the distribution of

images [26]. The generator architecture is typically composed of the following layers:

- **Linear layer**: The noise vector is fed into a fully connected layer, and its output is reshaped into a tensor.
- **Batch Normalization Layer**: Stabilizes learning by normalizing inputs to zero mean and unit variance, avoiding training issues such as vanishing or exploding gradients and allowing the gradient to flow through the network.
- **Up sample Layer**: Instead of using a convolutional transpose layer to up sample, it mentions using upsampling and then applying a simple convolutional layer on top of it. Convolutional transpose is sometimes used instead.
- **Convolutional Layer**: To learn from up-sampled data, the matrix is passed through a convolutional layer with a stride of 1 and the same padding as it is up sampled.
- **ReLU Layer**: for the generator because it allowed the model to quickly saturate and cover the training distribution space.
- **TanH Activation**: TanH enables the model to converge more quickly.

Convolutional kernels are in fact relearning duplicate data because of the high correlations in real-world data. The convolution makes the neural network training difficult. In the following, we review the mathematical presentation of GAIN.

Problem Formulation In a d-dimensional space $\mathcal{X} = \mathcal{X}_1 \times ... \times \mathcal{X}_d$ the $\mathbf{X} = (X_1, ..., X_d)$ is a random variable taking values in $\mathcal{X}$ with distribution $P(\mathbf{X})$. $\mathbf{M} = (M_1, ..., M_d)$ is a random variable in $\{0, 1\}^d$. The $\mathbf{X}$ is called data vector, and $\mathbf{M}$ is called the mask vector.

A new space $\tilde{\mathcal{X}}_i = \mathcal{X}_i \cup \{*\}$ is defined for $i \in \{1, ..., d\}$ where the start, $*$ does not belong to any $\mathcal{X}_i$, and represents an unobserved value. Defining $\tilde{\mathcal{X}} = \tilde{\mathcal{X}}_1 \times ... \times \tilde{\mathcal{X}}_d$ The variable $\tilde{\mathbf{X}} = (\tilde{X}_1, ..., \tilde{X}_d) \in \tilde{\mathcal{X}}$ is presented in Eq. (1):

$$\tilde{X}_i = \begin{cases} X_i, & \text{if } M_i = 1 \\ *, & \text{otherwise} \end{cases} \tag{1}$$

where $\mathbf{M}$ indicates which components of $\mathbf{X}$ are observed. The $\mathbf{M}$ could be recovered from $\tilde{\mathbf{X}}$. In missing data re-construction, n i.i.d. copies of $\tilde{\mathbf{X}}$ are realized, denoted by $\tilde{\mathbf{x}}^1, ..., \tilde{\mathbf{x}}^n$ and defined in the dataset $\mathcal{D} = \{(\tilde{\mathbf{x}}^i, \mathbf{m}^i)\}_{i=1}^n$, where $\mathbf{m}^i$ is simply the recovered realization of $\mathbf{M}$ corresponding to $\tilde{\mathbf{x}}^i$. The goal is to estimate the unobserved values in each $\tilde{\mathbf{x}}_i$. The samples are generated according to $P(\mathbf{X}|\tilde{\mathbf{X}} = \tilde{\mathbf{x}}^i)$, that is the conditional distribution of $\mathbf{X}$ given $\tilde{\mathbf{X}} = \tilde{\mathbf{x}}^i$, for each i, to fill in the missing data points in $\mathcal{D}$.

The generator, G, takes as input $\tilde{\mathbf{X}}$, $\mathbf{M}$ and a noise variable $\mathbf{Z}$, and outputs $\bar{\mathbf{X}}$. Where $\bar{\mathbf{X}}$ is a vector of synthetic data. Let $G : \tilde{\mathcal{X}} \times \{0, 1\}^d \times [0, 1]^d \to \mathcal{X}$ be a function, and $\mathbf{Z} = (Z_1, ..., Z_d)$ be d-dimensional noise (independent of all other variables) [21]. The random variables $\bar{\mathbf{X}}, \hat{\mathbf{X}} \in \mathcal{X}$ is defined by Eq. (2) and Eq. (3).

$$\bar{\mathbf{X}} = G(\tilde{\mathbf{X}}, \mathbf{M}, (\mathbf{1} - \mathbf{M}) \odot \mathbf{Z}) \tag{2}$$

$$\hat{\mathbf{X}} = \mathbf{M} \odot \tilde{\mathbf{X}} + (\mathbf{1} - \mathbf{M}) \odot \bar{\mathbf{X}} \tag{3}$$

where $\odot$ denotes element-wise multiplication. $\bar{\mathbf{X}}$ corresponds to the vector of estimated values and $\hat{\mathbf{X}}$ corresponds to the completed data vector.

The discriminator, D will be used to train G. However, unlike the standard GAN where the output of the generator is either real or synthetic, the output of GAIN is comprised of some components that are real and some that are synthetic. Rather than identifying that an entire vector is real or synthetic, the discriminator attempts to distinguish which are real (observed) or synthetic. The mask vector $\mathbf{M}$ is pre-determined by the dataset. Formally, the discriminator is a function $D : \mathcal{X} \rightarrow [0,1]^d$ with the i-th component of $D(\hat{\mathbf{x}})$ corresponding to the probability that the i-th component of $\hat{\mathbf{x}}$ was observed. The D is trained to maximize the probability of correctly predicting $\mathbf{M}$ and G is trained to minimize the probability of D predicting $\mathbf{M}$. The quantity $V(D,G)$ is defined as presented in Eq. (4).

$$V(D,G) = \mathbb{E}_{\hat{\mathbf{X}},\mathbf{M},\mathbf{H}}\Big[\mathbf{M}^T \log D(\hat{\mathbf{X}},\mathbf{H}) + (\mathbf{1}-\mathbf{M})^T \log\big(\mathbf{1} - D(\hat{\mathbf{X}},\mathbf{H})\big)\Big], \tag{4}$$

where $\log$ is element-wise logarithm and dependence on G is through $\hat{\mathbf{X}}$. The goal of GAIN is presented in Eq. (5).

$$\min_G \max_D V(D,G). \tag{5}$$

where the loss function $\mathcal{L} : \{0,1\}^d \times [0,1]^d \rightarrow \mathbb{R}$ is defined as presented in Eq. (6).

$$\mathcal{L}(\mathbf{a},\mathbf{b}) = \sum_{i=1}^{d} \Big[a_i \log(b_i) + (1-a_i)\log(1-b_i)\Big]. \tag{6}$$

DEGAIN DEGAIN is originated from GAIN. The main idea behind the DEGAIN is to use the deconvolution in the generator and discriminator. Convolution applies a kernel to overlapping regions shifted across the data. However, because of the strong correlations in real-world data, convolutional kernels are in effect re-learning redundant data. This redundancy makes the neural network training challenging. The deconvolution can remove the correlations before the data is fed into each layer. It has been shown in [23] that the deconvolution can be efficiently calculated at a fraction of the computational cost of a convolution layer. The deconvolution strategy is proven to be effective on images, however, it has not been applied to GANs including the GAIN.

Given a data matrix $X_{N\times F}$ where N is the number of samples, and F is the number of features, the covariance matrix is calculated as $Cov = \frac{1}{N}(X-\mu)^T(X-\mu)$.

An approximated inverse square root of the covariance matrix could be calculated as $D = Cov^{-\frac{1}{2}}$ multiplied with the centered vectors $(X-\mu)\cdot D$. Accordingly, the correlation effects could be removed. If computed perfectly, the transformed data has the identity matrix as covariance: $D^T(X-\mu)^T(X-\mu)D = Cov^{-0.5}\cdot Cov \cdot Cov^{-0.5} = I$.

The process to construct X and $D \approx (Cov + \epsilon\cdot I)^{-\frac{1}{2}}$ is presented in Algorithm 1, where $\epsilon\cdot I$ improves the stability. The deconvolution operation is further applied via matrix multiplication to remove the correlation between neighboring pixels. The deconvolved data is then multiplied with w. The architecture of proposed method is depicted in Figure 1. When the training phase is completed, the G component is able to impute the dataset. There are two main loops for

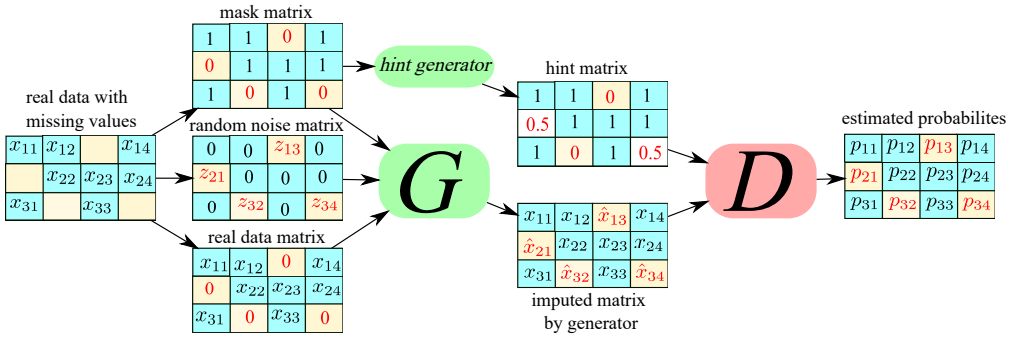

Figure 1: The gerenal structure of DEGAIN

Algorithm 1 (The DEGAIN Algorithm: Deconvolution and then Training)

Input: N channels of input features $[x_1, x_2, ..., x_N]$, Number of epochs e; Number of Iteration of inner loop n; Updating rates (α_g and α_d);
for $i \in \{1, ..., N\}$ **do**
    $X_i = im2col(x_i)$
$X = [X_1, ..., X_N]$
$X = Reshape(X)$
$Cov = \frac{1}{M} X^t X \ \%[x_i]$ has M rows
$D \approx (Cov + \epsilon \cdot I)^{-\frac{1}{2}}$
Training of G **and** D
for $(i = 0; i < e; i{+}{+})$ **do**
    for $j = 0; j < n; j{+}{+}$ **do**
        batch samples from noise $Z \in R^{B \times L} \sim p_z(z)$;
        batch samples from noise $X \in R^{B \times M}$;
        $L(\theta_d) = \frac{1}{B} \Sigma_{b=1}^{B} logD(X_b, \theta_d) + log(1 - D(G(2Z_b), \theta_g))$;
        $\xi_d = \frac{\delta}{\delta \theta_d} L(\theta_d)$;
        $\theta_d^{j+1} = \theta_d^t + \alpha_d \xi_d$
    batch samples from noise $Z \in R^{B \times L} \sim p_z(z)$;
    $L(\theta_d) = \frac{1}{B} \Sigma_{b=1}^{B} log(1 - D(G(Z_b, \theta_g)))$
    $\xi_d = \frac{\delta}{\delta \theta_d} L(\theta_d)$;
    $\theta_g^{i+1} = \theta_g^t - \alpha_g \xi_g$;

updating the parameters of the G and D components in Algorithm 1. First, batch samples from the noise and samples of real data are presented to the inner loop for updating the parameters of the D component. The cost function of the D component is then calculated by the given samples. Then, the D component's parameters are updated based on the initiated rate.

Table 1
Evaluation metrics used to assess the proposed algorithm's performance

Metric	Formula
RMSE	$RMSE = \sqrt{\frac{1}{\hat{N}} \sum_{i=1}^{\hat{N}} (y_i - \hat{y}_i)^2}$
FID	$d^2((m_1, C_1)(m_2, C_2)) = \|m_1 - m_2\|_2^2 + \mathrm{Tr}(C_1 + C_2 - 2(C_1 C_2)^{\frac{1}{2}})$

3. Performance Evaluation

We evaluate the performance of the DEGAIN and compare the results with MICE [25], GAIN [21] and AE [24]. Multivariate imputation by chained equations (MICE) has emerged in addressing missing data. The chained equations approach can handle variables of varying types, for instance the continuous or binary as well as complexities such as bounds. MICE is also a software package presented in R [25]. In multiple imputation algorithms such as AE, multiple copies of the dataset replaced by slightly different imputed values in each copy. In this method the variability is modeled into the imputed values [24].

We perform each experiment 10 times and within each experiment we use 5-cross validations in terms of RMSE, which directly measures the error distance, and FID, which takes the distribution of imputed values into account. We use the Letter and SPAM datasets for comparing algorithms, and samples are missed with the rate of 20%.

Evaluation Metrics In our experiments we consider the Root-Mean-Square Error (RMSE) and Frechet Inception Distance (FID) metrics to evaluate the performance of the proposed DEGAIN on handling the missing data. It should be noted that besides RMSE and FID, several other metrics are introduced in the literature including but not limited to Mean Absolute Error (MAE), Area under ROC curve or AUC score and F1-score. These metrics perform in different dataset or operations. For example, the F1-score is mostly used for binary classification. MAE is simialr to RMSE, however, RMSE is more common in the literature. Comparing our method with GAIN, AE and MICE, we have chosen the related RMSE and FID metrics. Remember that RMSE is used for continuous values and measures the error between real values and imputed values for an incomplete dataset. FID converts a group of imputed samples to a feature space using a particular inception net layer. Assuming the converted layer as continuous multivariate Gaussian distribution, the mean and covariance are predicted for both the imputed and real samples.

The mathematical presentation is shown in Table 1, where $\hat{N}$ is the number of missing values, y_i is the real missing value, and $\hat{y}_i$ is the imputed value. Also the m_1 and m_2 denote the mean of the real and imputed data, respectively. C_1 and C_2 indicate the covariance of real and imputed data, respectively.

Dataset Here we use the **Letter** and **SPAM** datasets. Letter is publicly available in the UC Irvine Machine Learning Repository and can be accessed through https://archive.ics.uci.edu/. In

Table 2
Performance evaluations of proposed DEGAIN, GAIN [21], AE [24], MICE [25] in terms of RMSE and FID metrics

	Proposed DEGAIN	GAIN [21]	AE [24]	MICE [25]
RMSE (Letter dataset)	0.096	0.101	0.142	0.166
Normalized FID (Letter dataset)	0.492	0.513	0.826	1
RMSE (SPAM dataset)	0.047	0.050	0.064	0.068
Normalized FID (SPAM dataset)	0.898	0.946	0.973	1

this dataset, the objective is to identify each of a large number of black-and-white rectangular pixel displays as one of the 26 capital letters in the English alphabet. The character images are based on 20 different fonts and each letter within these 20 fonts are randomly distorted to produce a file of 20,000 unique stimuli. Typically, the first 16000 items are used for training and then the resulting model is capable to predict the letter category for the remaining 4000. The SPAM dataset consists of 4601 samples and 57 input features. In this dataset the goal is to predict the spam emails based on input features. SPAM is also publicly available at http://archive.ics.uci.edu/ml/datasets/Spambase/. These datasets have no missing values and therefore, we use a 20% rate of missed samples.

Results The evaluation are performed in google colab, on python 3 with 12 GB of RAM. We used the base codes of GAIN [21] publicly accessible in https://github.com/jsyoon0823/GAIN. We have modified the code and added the deconvolution to the Generator and Discriminator. The results are presented in Table 2. Both GAIN and DEGAIN are performing much better than the auto encoder (AE) [24] and MICE [25]. The DEGAIN is slightly better compared with the GAIN. The main advantage of the DEGAIN could be explored running on correlated large datasets of images. Therefore, as expected, the GAIN and proposed DEGAIN can be most profitable in image datasets.

4. Conclusions

Dealing with incomplete information is an important problem that includes different aspects. In this paper we described an algorithm called DEGAIN to estimate the missing values in the dataset. The DEGAIN is based on the known GAIN algorithm that is already used in missing data imputation. We added deconvolution to remove the correlation between data. We evaluated the performance of the presented method on two publicly available datasets, called Letter and SPAM. The RMSE and FID metrics on the results confirmed that the GANs are effective on re-constructing the missing values compared with the earlier Auto-Encoder or MICE algorithms. Also the proposed DEGAIN performs well and improves the performance of GAIN. We believe that the main advantages of DEGAIN could be explored running on large image datasets, although it shows improvement even on our chosen datasets. In this paper we presented DEGAIN as tool for Missing Data Imputation. Inconsistent information, which could be handled using methods such as arbitration [27], is still an open issue that GAN-based methods and the proposed DEGAIN need to address in future works.

Acknowledgments

This research was supported by MISE Project *True Detective 4.0.*

References

[1] S. Greco, C. Molinaro, I. Trubitsyna, Approximation algorithms for querying incomplete databases, Information Systems 86 (2019) 28–45.

[2] G. Alfano, S. Greco, C. Molinaro, I. Trubitsyna, An approximation algorithm for querying inconsistent knowledge bases, Intelligent Systems with Applications 17 (2023) 200146.

[3] M. Calautti, S. Greco, C. Molinaro, I. Trubitsyna, Query answering over inconsistent knowledge bases: A probabilistic approach, Theor. Comput. Sci. 935 (2022) 144–173.

[4] M. Calautti, S. Greco, C. Molinaro, I. Trubitsyna, Preference-based inconsistency-tolerant query answering under existential rules, Artificial Intelligence 312 (2022).

[5] P. Guagliardo, L. Libkin, Making sql queries correct on incomplete databases: A feasibility study, in: 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, 2016, pp. 211–223.

[6] E. Toussaint, P. Guagliardo, L. Libkin, J. Sequeda, Troubles with nulls, views from the users, Proceedings of the VLDB Endowment 15 (2022) 2613–2625.

[7] J. Hu, Z. Zhou, X. Yang, Characterizing Physical-Layer transmission errors in cable broadband networks, in: 19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22), USENIX Association, Renton, WA, 2022, pp. 845–859. URL: https://www.usenix.org/conference/nsdi22/presentation/hu.

[8] K. Yu, Y. Yang, W. Ding, Causal feature selection with missing data 16 (2022). URL: https://doi.org/10.1145/3488055. doi:10.1145/3488055.

[9] L. Peng, L. Lei, A review of missing data treatment methods, Intell. Inf. Manag. Syst. Technol 1 (2005) 412–419.

[10] A. Folch-Fortuny, F. Arteaga, A. Ferrer, Pca model building with missing data: New proposals and a comparative study, Chemometrics and Intelligent Laboratory Systems 146 (2015) 77–88.

[11] S. L. Mirtaheri, R. Shahbazian, Machine Learning: Theory to Applications, CRC Press, 2022.

[12] G. Nagarajan, L. D. Babu, Missing data imputation on biomedical data using deeply learned clustering and l2 regularized regression based on symmetric uncertainty, Artificial Intelligence in Medicine 123 (2022) 102214.

[13] T. Emmanuel, T. Maupong, D. Mpoeleng, T. Semong, B. Mphago, O. Tabona, A survey on missing data in machine learning, Journal of Big Data 8 (2021) 1–37.

[14] Y. Ma, Y. He, L. Wang, J. Zhang, Probabilistic reconstruction for spatiotemporal sensor data integrated with gaussian process regression, Probabilistic Engineering Mechanics 69 (2022) 103264.

[15] F. Camastra, V. Capone, A. Ciaramella, A. Riccio, A. Staiano, Prediction of environmental missing data time series by support vector machine regression and correlation dimension estimation, Environmental Modelling & Software 150 (2022) 105343.

[16] A. J. Saroj, A. Guin, M. Hunter, Deep lstm recurrent neural networks for arterial traffic volume data imputation, Journal of Big Data Analytics in Transportation 3 (2021) 95–108.
[17] D. Cenitta, R. V. Arjunan, K. Prema, Missing data imputation using machine learning algorithm for supervised learning, in: 2021 International Conference on Computer Communication and Informatics (ICCCI), IEEE, 2021, pp. 1–5.
[18] F. Tang, H. Ishwaran, Random forest missing data algorithms, Statistical Analysis and Data Mining: The ASA Data Science Journal 10 (2017) 363–377.
[19] S. Ryu, M. Kim, H. Kim, Denoising autoencoder-based missing value imputation for smart meters, IEEE Access 8 (2020) 40656–40666.
[20] F. V. Nelwamondo, S. Mohamed, T. Marwala, Missing data: A comparison of neural network and expectation maximization techniques, Current Science (2007) 1514–1521.
[21] J. Yoon, J. Jordon, M. Schaar, Gain: Missing data imputation using generative adversarial nets, in: International Conference on Machine Learning, PMLR, 2018, pp. 5689–5698.
[22] R. Shahbazian, I. Trubitsyna, DEGAIN: generative-adversarial-network-based missing data imputation, Inf. 13 (2022) 575.
[23] C. Ye, M. Evanusa, H. He, A. Mitrokhin, T. Goldstein, J. A. Yorke, C. Fermüller, Y. Aloimonos, Network deconvolution, arXiv preprint arXiv:1905.11926 (2019).
[24] L. Gondara, K. Wang, Multiple imputation using deep denoising autoencoders, arXiv preprint arXiv:1705.02737 (2017).
[25] S. Van Buuren, K. Groothuis-Oudshoorn, mice: Multivariate imputation by chained equations in r, Journal of statistical software 45 (2011) 1–67.
[26] B. Benjdira, A. Ammar, A. Koubaa, K. Ouni, Data-efficient domain adaptation for semantic segmentation of aerial imagery using generative adversarial networks, Applied Sciences 10 (2020) 1092.
[27] P. Z. Revesz, On the semantics of arbitration, International journal of Algebra and Computation 7 (1997) 133–160.

5. Online Resources

The datasets used in this paper are publicly available at
https://archive.ics.uci.edu/ml/datasets/letter+recognition and
http://archive.ics.uci.edu/ml/datasets/Spambase/