Transformed Distribution Matching for Missing Value Imputation

He Zhao [1] **Ke Sun** [1] **Amir Dezfouli** [1] **Edwin V. Bonilla** [1]

Abstract

We study the problem of imputing missing values in a dataset, which has important applications in many domains. The key to missing value imputation is to capture the data distribution with incomplete samples and impute the missing values accordingly. In this paper, by leveraging the fact that any two batches of data with missing values come from the same data distribution, we propose to impute the missing values of two batches of samples by transforming them into a latent space through deep invertible functions and matching them distributionally. To learn the transformations and impute the missing values simultaneously, a simple and well-motivated algorithm is proposed. Our algorithm has fewer hyperparameters to fine-tune and generates high-quality imputations regardless of how missing values are generated. Extensive experiments over a large number of datasets and competing benchmark algorithms show that our method achieves state-of-the-art performance[1].

1. Introduction

In practice, real-world data are usually incomplete and consist of many missing values. For example, in the medical domain, the health record of a patient may have considerable missing items as not all of the characteristics are properly recorded or not all of the tests have been done for the patient (Barnard & Meng, 1999). In this paper, we are interested in imputing missing values in an unsupervised way (Van Buuren & Groothuis-Oudshoorn, 2011; Yoon et al., 2018; Mattei & Frellsen, 2019; Muzellec et al., 2020; Jarrett et al., 2022). Here the meaning of "unsupervised" is twofold: We do not know the ground truth of the missing values during training and we do not assume a specific downstream task.

[1]CSIRO's Data61, Australia. Correspondence to: He Zhao <he.zhao@ieee.org>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

[1]Code at https://github.com/hezgit/TDM

The key to missing value imputation is how to model the data distribution with a considerable amount of missing values, which is a notoriously challenging problem. To address this challenge, existing approaches either choose to model the conditional data distribution instead (i.e., the distribution of one feature conditioned on the other features), such as in Heckerman et al. (2000); Raghunathan et al. (2001); Gelman (2004); Van Buuren et al. (2006); Van Buuren & Groothuis-Oudshoorn (2011); Liu et al. (2014); Zhu & Raghunathan (2015) or use deep generative models to capture the data distribution, such as in Gondara & Wang (2017); Ivanov et al. (2018); Mattei & Frellsen (2019); Nazabal et al. (2020); Gong et al. (2021); Peis et al. (2022); Yoon et al. (2018); Li et al. (2018); Yoon & Sull (2020); Dai et al. (2021); Fang & Bao (2022); Richardson et al. (2020); Ma & Ghosh (2021); Wang et al. (2022). Alternatively, a recent interesting idea proposed by Muzellec et al. (2020) has found success, whose key insight is that any two batches of data (with missing values) come from the same data distribution. Thus, a good method should impute the missing values to make the empirical distributions of the two batches matched, i.e., distributionally close to each other. This is a more general and applicable assumption that can be used in various data under different missing value mechanisms. In this paper, we refer to this idea as *distribution matching* (DM), the appealing property of which is that it bypasses modelling the data distribution explicitly or implicitly, a difficult task even without missing values.

As the pioneering study, Muzellec et al. (2020) does DM by minimising the optimal transport (OT) distance whose cost function is the quadratic distance in the data space between data samples. However, real-world data usually exhibit complex geometry, which might not be captured well by the quadratic distance in the data space. This can lead to wrong imputations and poor performance. In this paper, we propose a new, straightforward, yet powerful DM method, which first transforms data samples into a latent space through a deep invertible function and then does distribution matching with OT in the latent space. In the latent space, the quadratic distance of two samples is expected to better reflect their (dis)similarity under the geometry of the data considered. In the missing-value setting, learning a good transformation is non-trivial. We propose a simple and elegant algorithm that learns the transformations and

imputes the missing values simultaneously, which is well-motivated by the theory of OT and representation learning.

The contributions of this paper include: **1)** As DM is a new and promising line of research in missing value imputation, we propose a well-motivated transformed distribution matching method, which significantly improves over previous methods. **2)** In practice, the ground truth of missing values is usually unknown, making it hard to fine-tune methods with complex algorithms and many hyperparameters. We develop a simple and theoretically sound learning algorithm with a single loss and very few hyperparameters, alleviating the need for extensive fine-tuning. **3)** We conduct extensive experiments over a large number of datasets and competing benchmark algorithms in multiple missing-value mechanisms and report comprehensive evaluation metrics. These experiments show that our method achieves state-of-the-art performance.

2. Background

2.1. Data with Missing Values

Here we consider N data samples with D-dimensional features stored in matrix $\boldsymbol{X} \in \mathbb{R}^{N\times D}$, where a row vector $\boldsymbol{X}[i,:] \in \mathbb{R}^D$ $(1 \le i \le N)$ represents the i^{th} sample. The missing values contained in $\boldsymbol{X}$ are indicated by a binary mask $\boldsymbol{M} \in \{0,1\}^{N\times D}$ such that $\boldsymbol{M}[i,d] = 1$ indicates that the d^{th} feature of sample i is missing and $\boldsymbol{M}[i,d] = 0$ otherwise. Moreover, we assign NaN (Not a Number) to the missing values and use the following Python/Numpy style matrix indexing $\boldsymbol{X}[\boldsymbol{M}] = \text{NaN}$ to denote the missing data. Similarly, the observed data can be denoted as $\boldsymbol{X}[\mathbf{1} - \boldsymbol{M}]$, where $\mathbf{1}_{N\times D}$ is the matrix with the same dimension as $\boldsymbol{M}$ filled with ones. The task of imputation is to fill $\boldsymbol{X}[\boldsymbol{M}]$ with the imputed values given the mask $\boldsymbol{M}$ and the observed values $\boldsymbol{X}[\mathbf{1} - \boldsymbol{M}]$.

In practice, three missing value patterns/mechanisms (i.e., ways of generating the mask) have been widely explored (Rubin, 1976; 2004; Van Buuren, 2018; Seaman et al., 2013): missing completely at random (MCAR) where the missingness is independent of the data; missing at random (MAR) where the probability of being missing depends only on observed values; missing not at random (MNAR) where the probability of missingness then depends on the unobserved values. In MCAR and MAR, the missing value patterns are "ignorable" as it is unnecessary to model the distribution of missing values explicitly while MNAR can be a harder case where missing values may lead to important biases in data (Muzellec et al., 2020; Jarrett et al., 2022).

2.2. Optimal Transport

Optimal transport (OT) provides sound and meaningful distances to compare distributions (Peyré et al., 2019), which has been used in many problems, such as computer vision (Ge et al., 2021; Zhang et al., 2022), text analysis (Huynh et al., 2020; Zhao et al., 2021; Guo et al., 2022c), adversarial robustness (Bui et al., 2022), probabilistic (generative) models (Vuong et al., 2023; Vo et al., 2023), and other machine learning applications (Nguyen et al., 2021; Guo et al., 2021; Nguyen et al., 2022; Guo et al., 2022a;b). Here we briefly introduce OT between two discrete distributions of dimensionality B and B', respectively. Let $\alpha(\boldsymbol{X}^1) := \sum_{i=1}^{B} a_i \delta_{\boldsymbol{X}^1[i,:]}$ and $\beta(\boldsymbol{X}^2) := \sum_{j=1}^{B'} b_j \delta_{\boldsymbol{X}^2[j,:]}$, where $\boldsymbol{X}^1 \in \mathbb{R}^{B\times D}$ and $\boldsymbol{X}^2 \in \mathbb{R}^{B'\times D}$ denote the supports of the two distributions, respectively; $\boldsymbol{a} \in \Delta^B$ and $\boldsymbol{b} \in \Delta^{B'}$ are two probability vectors; Δ^B is the $(B-1)$-dimensional probability simplex. The OT distance between $\alpha(\boldsymbol{X}^1)$ and $\beta(\boldsymbol{X}^2)$ can be defined as:

$$d_{\boldsymbol{G}}\left(\alpha(\boldsymbol{X}^1), \beta(\boldsymbol{X}^2)\right) := \inf_{\boldsymbol{P}\in U(\boldsymbol{a},\boldsymbol{b})} \langle \boldsymbol{P}, \boldsymbol{G}\rangle, \qquad (1)$$

where $\langle\cdot,\cdot\rangle$ denotes the Frobenius dot-product; $\boldsymbol{G} \in \mathbb{R}_{\ge 0}^{B\times B'}$ is the cost matrix/function of the transport; $\boldsymbol{P} \in \mathbb{R}_{>0}^{B\times B'}$ is the transport matrix/plan; $U(\boldsymbol{a},\boldsymbol{b})$ denotes the transport polytope of $\boldsymbol{a}$ and $\boldsymbol{b}$, which is the polyhedral set of $B \times B'$ matrices: $U(\boldsymbol{a},\boldsymbol{b}) := \left\{\boldsymbol{P} \in \mathbb{R}_{>0}^{B\times B'} | \boldsymbol{P}\mathbf{1}_{B'} = \boldsymbol{a}, \boldsymbol{P}^T\mathbf{1}_B = \boldsymbol{b}\right\}$; and $\mathbf{1}_B$ is the N-dimensional column vector of ones. The cost matrix $\boldsymbol{G}$ contains the pairwise distance/cost between B and B' supports, for which different distance metrics can be used. Specifically, if the cost is defined as the pairwise quadratic distance: $\boldsymbol{G}[i,j] = \|\boldsymbol{X}^1[i,:] - \boldsymbol{X}^2[j,:]\|^2$, where $\|\cdot\|$ is the Euclidean norm, the OT distance reduces to the p-Wasserstein distance ($p = 2$ here), i.e., $d_{\boldsymbol{G}}\left(\alpha(\boldsymbol{X}^1), \beta(\boldsymbol{X}^2)\right) = W_2^2\left(\alpha(\boldsymbol{X}^1), \beta(\boldsymbol{X}^2)\right)$.

3. Method

3.1. Previous Work: Optimal Transport for Missing Value Imputation

Our method is motivated by Muzellec et al. (2020), the pioneering method of distribution matching, recently proposed for missing value imputation. Consider two batches of data of $\boldsymbol{X}$: $\boldsymbol{X}^1$ and $\boldsymbol{X}^2$ with batch size of B, both of which can contain missing values. The key insight is that a good method should impute the missing values in $\boldsymbol{X}^1$ and $\boldsymbol{X}^2$ so that the empirical distributions of them are matched. To do so, Muzellec et al. (2020) propose to minimise the OT distance[2] between them in terms of the missing values.

$$\min_{\boldsymbol{X}^{1\cup 2}[\boldsymbol{M}^{1\cup 2}]} W_2^2\left(\mu(\boldsymbol{X}^1), \mu(\boldsymbol{X}^2)\right), \qquad (2)$$

where $\mu(\boldsymbol{X}^1) = \frac{1}{B}\sum_i \delta_{\boldsymbol{X}^1[i,:]}$ denotes the empirical measure associated to the B samples of $\boldsymbol{X}^1$ (simi-

[2]The paper actually uses the Sinkhorn divergence (Genevay et al., 2018; Feydy et al., 2019), a surrogate divergence to OT.

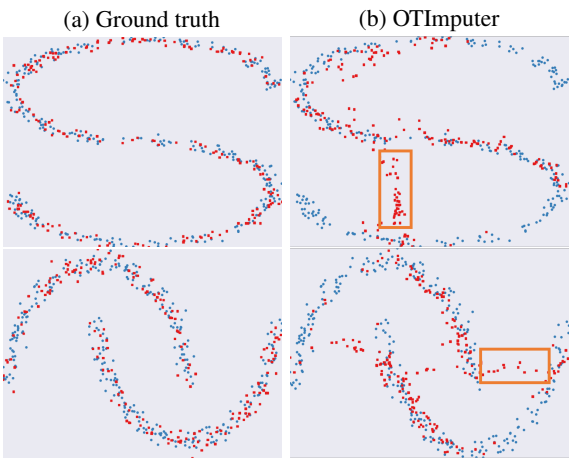

Figure 1. Two synthetic datasets (two rows) each of which is with 500 samples. (a) Ground truth: Blue points (60%) have no missing values and red points (40%) have one missing value on either coordinate (following MCAR). (b) The imputed values for the red points by OTImputer[4], where the orange rectangle highlights the region of interest.

larly for $\mu(\boldsymbol{X}^2)$); $\boldsymbol{X}^{1\cup 2}$ denotes the union of $\boldsymbol{X}^1$ and $\boldsymbol{X}^2$, i.e., the unique data samples of the two batches; $\boldsymbol{M}^{1\cup 2}$ denotes the union of $\boldsymbol{M}^1$ and $\boldsymbol{M}^2$, i.e., the mask of missing values in $\boldsymbol{X}^{1\cup 2}$. To impute the missing values, one can take an iterative update of $\boldsymbol{X}^{1\cup 2}[\boldsymbol{M}^{1\cup 2}]$ by gradient descent, e.g., RMSprop (Tieleman et al., 2012): $\boldsymbol{X}^{1\cup 2}[\boldsymbol{M}^{1\cup 2}] \leftarrow \boldsymbol{X}^{1\cup 2}[\boldsymbol{M}^{1\cup 2}] - \alpha\text{RMSprop}\left(\nabla_{\boldsymbol{X}^{1\cup 2}[\boldsymbol{M}^{1\cup 2}]} W_2^2\left(\mu(\boldsymbol{X}^1), \mu(\boldsymbol{X}^2)\right)\right)$. We refer to this method as "OTImputer".

3.2. Motivations

We now motivate the proposed method by giving a closer look at OTImputer.

Lemma 3.1. *For any given $\boldsymbol{X}^1$ and $\boldsymbol{X}^2$, we have:*

$$W_2^2\left(\mu(\boldsymbol{X}^1), \mu(\boldsymbol{X}^2)\right) = \min_{\pi} \frac{1}{B}\sum_{i=1}^{B} \|\boldsymbol{X}^1[i,:] - \boldsymbol{X}^2[\pi(i),:]\|^2,$$

where the minimum is taken over all possible permutations π of the sequence $(1,\ldots,B)$, and $\pi(i)$ is the permuted index of i with $1 \le \pi(i) \le B$.

Proof. It is a special case of Proposition 2 of Nguyen (2011). □

The above lemma shows that 2-Wasserstein distance between two empirical distributions used in OTImputer is equivalent to the minimisation of a matching distance by finding the optimal permutation. In the missing value imputation context, OTImputer finds the closest pairs of data samples in the two batches in terms of the quadratic distance in the data space (by the computation of 2-Wasserstein) and then tries the hardest to minimise their quadratic distance to impute missing values (by the minimisation of 2-Wasserstein). Real-world data usually exhibit complex geometry, which can hardly be captured by the quadratic distance in the data space. Figure 1 shows the imputation results on two synthetic datasets. It can be seen that OTImputer incorrectly imputes a considerable amount of missing values within the orange rectangles. This is because these imputed samples have a small quadratic distance to others in the data space but they are not good imputations. Therefore, the quadratic distance in the data space is unable to reflect the data geometry.

3.3. Proposed Method

In this paper, we introduce **T**ransformed **D**istribution **M**atching (TDM), which carries out OT-based missing value imputation on a *transformed* space, where the distances between the transformed samples can reveal the similarity/dissimilarity between them better, respecting the underlying geometry of the data. Specifically, we aim to learn a deep transformation parameterised by θ, $f_\theta : \mathbb{R}^{D'} \to \mathbb{R}^D$ that projects a data sample $\boldsymbol{x} \in \mathbb{R}^D$ to a transformed one $\boldsymbol{z} \in \mathbb{R}^{D'}$: $\boldsymbol{z} := f_\theta(\boldsymbol{x})$. With a slight abuse of notation, we denote the batch-level transformation as $\boldsymbol{Z} = f_\theta(\boldsymbol{X})$ where f_θ is applied to each sample (row vector) in $\boldsymbol{X}$. Generalising Eq. (2), we learn f_θ and the imputations by:

$$\min_{\boldsymbol{X}^{1\cup 2}[\boldsymbol{M}^{1\cup 2}],\theta} \mathcal{L}^W(\boldsymbol{X}^1, \boldsymbol{X}^2), \tag{3}$$

$$\mathcal{L}^W(\boldsymbol{X}^1, \boldsymbol{X}^2) = W_2^2\left(f_{\#}\mu(\boldsymbol{X}^1), f_{\#}\mu(\boldsymbol{X}^2)\right), \tag{4}$$

where $f_{\#}\mu(\boldsymbol{X}) := \mu(f_\theta(\boldsymbol{X}))$. If f_θ is an isometry, then our TDM reduces to OTImputer (Muzellec et al., 2020). We provide more theoretical analysis of this loss in Section A of the appendix.

The above optimisation is straightforward, however, simply minimising the Wasserstein loss can lead to *model collapsing*, meaning that no matter what the input sample is, f_θ always transforms it into the same point in the latent space. It is easy to see that model collapsing is the trivial solution that minimises the Wasserstein distance between any two samples (or batches) (i.e., to be zero), regardless of how the missing values are imputed. To prevent model collapsing, we are inspired by the viewpoint of representation learning, where our method can be viewed to learn the latent representation $\boldsymbol{z}$ from the data sample $\boldsymbol{x}$ with missing values in an unsupervised way. Representation learning based on mutual information (MI) has been shown promising in several domains (Oord et al., 2018; Bachman et al., 2019; Hjelm et al.,

[4]Note that Figure 1 is not directly comparable with Figure 2 in Muzellec et al. (2020) because the synthetic dataset is generated differently, the missing value proportion is different, and the computation of OT is done differently.

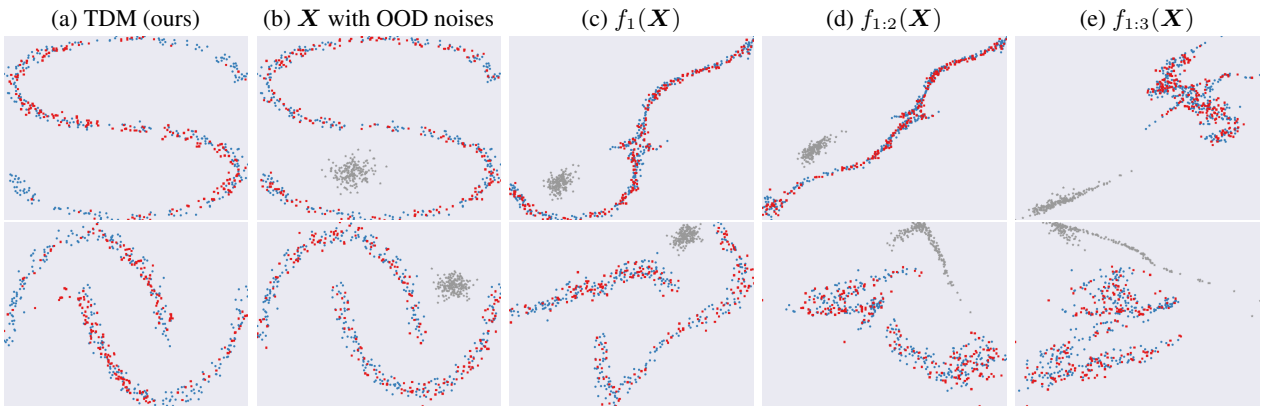

Figure 2. (a) The imputed values for the red points by TDM, corresponding to the ground truth of Figure 1(a). (b) $\boldsymbol{X}$ with OOD noises (grey points). (c-e) The transformed points by different blocks of f_θ.

2019; Tschannen et al., 2020), where it has been motivated by the InfoMax principle (Linsker, 1988). To avoid model collapsing, we propose to add a constraint to f_θ such that it also maximises the MI $I\left(\boldsymbol{X}, f_\theta(\boldsymbol{X})\right)$. Thus, we define

$$\mathcal{L}^{\text{MI}}(\boldsymbol{X}) = -I\left(\boldsymbol{X}, f_\theta(\boldsymbol{X})\right), \tag{5}$$

and aim to learn θ by minimising both $\mathcal{L}^W(\boldsymbol{X}^1, \boldsymbol{X}^2)$ and $\mathcal{L}^{\text{MI}}(\boldsymbol{X}^1) + \mathcal{L}^{\text{MI}}(\boldsymbol{X}^2)$.

Estimating the above MI in high-dimensional spaces has been known as a difficult task (Tschannen et al., 2020). Although several methods have been proposed to approximate the estimation, e.g., in Oord et al. (2018), they may inevitably add significant complexity and parameters to our method. Instead of maximising a tractable lower bound as in Oord et al. (2018); Poole et al. (2019), we propose a simpler approach that constrains f_θ to be a smooth invertible map.

Proposition 3.2. *If f_θ is a smooth invertible map, then $f_\theta \in \arg\max_{f'} I\left(\boldsymbol{X}, f'(\boldsymbol{X})\right)$.*

Proof. By definition, we have $I(\boldsymbol{X}, f'(\boldsymbol{X})) = H(\boldsymbol{X}) - H(\boldsymbol{X}|f'(\boldsymbol{X}))$ where $H(\boldsymbol{X})$ and $H(\boldsymbol{X}|f'(\boldsymbol{X}))$ are the entropy and conditional entropy, respectively. As we consider $\boldsymbol{X}$ and $f'(\boldsymbol{X})$ as empirical random variables with finite supports of their samples, $H(\boldsymbol{X}|f'(\boldsymbol{X})) \geq 0$. Therefore, $I(\boldsymbol{X}, f'(\boldsymbol{X})) \leq H(\boldsymbol{X}) = I(\boldsymbol{X}, \boldsymbol{X})$. If f_θ is a smooth invertible map, it is known that: $I(\boldsymbol{X}, f_\theta(\boldsymbol{X})) = I(\boldsymbol{X}, \boldsymbol{X})$ according to Eq. (45) of Kraskov et al. (2004). Therefore, $I(\boldsymbol{X}, f_\theta(\boldsymbol{X})) \geq I(\boldsymbol{X}, f'(\boldsymbol{X}))$. □

Proposition 3.2 shows that f_θ being invertible[5] (i.e., f_θ projects $\boldsymbol{x}$ to $\boldsymbol{z}$ and its inverse function f_θ^{-1} projects $\boldsymbol{z}$ back to $\boldsymbol{x}$) prevents model collapsing, without explicitly maximising the mutual information. Accordingly, we implement f_θ with invertible neural networks (INNs) (Dinh et al., 2014; 2017; Kingma & Dhariwal, 2018), which are approximators to invertible functions (Jacobsen et al., 2019; Gomez et al., 2017; Ardizzone et al., 2019; Kobyzev et al., 2020; Papamakarios et al., 2021). Specifically, the INNs for f_θ consists of a succession of T blocks, $f_\theta = f_1 \circ f_2 \circ \cdots f_T$, each of which is an invertible function[6]. Note that now we have $D' = D$, meaning that the output and input dimensions are the same for f_θ and every f_t $(1 \leq t \leq T)$.

For one block f_t, whose input and output vectors are denoted as $\boldsymbol{y}^{\text{in}} \in \mathbb{R}^D$ and $\boldsymbol{y}^{\text{out}} \in \mathbb{R}^D$ respectively, we implement it as an affine coupling block by following Ardizzone et al. (2019), which consists of two complementary affine coupling layers (Dinh et al., 2017):

$$\boldsymbol{y}^{\text{out}}_{1:d} = \boldsymbol{y}^{\text{in}}_{1:d} \odot \exp\left(g_1(\boldsymbol{y}^{\text{in}}_{d+1:D})\right) + h_1(\boldsymbol{y}^{\text{in}}_{d+1:D}), \tag{6}$$

$$\boldsymbol{y}^{\text{out}}_{d+1:D} = \boldsymbol{y}^{\text{in}}_{d+1:D} \odot \exp\left(g_2(\boldsymbol{y}^{\text{out}}_{1:d})\right) + h_2(\boldsymbol{y}^{\text{out}}_{1:d}), \tag{7}$$

where $\boldsymbol{y}^{\text{in}}$ and $\boldsymbol{y}^{\text{out}}$ are decomposed into two disjoint subsets, respectively: $\boldsymbol{y}^{\text{in}} = [\boldsymbol{y}^{\text{in}}_{1:d}, \boldsymbol{y}^{\text{in}}_{d+1:D}]$ and $\boldsymbol{y}^{\text{out}} = [\boldsymbol{y}^{\text{out}}_{1:d}, \boldsymbol{y}^{\text{out}}_{d+1:D}]$; d is set to $\lfloor D/2 \rfloor$; $\odot$ denotes the element-wise product. Moreover, g_1, g_2, h_1 and h_2 are neural networks, each of which is implemented by a succession of fully connected layers with the SELU activation (Klambauer et al., 2017). In addition, the output of g_1 and g_2 is clamped by the $\arctan$ function. The implementation ensures that f_t is invertible (Dinh et al., 2017) as proved in Section A.2 of the appendix. It is noticeable that our method is agnostic to the implementation of INNs and other coupling layers such as NICE (Dinh et al., 2014) and GLOW (Kingma & Dhariwal, 2018) can also be used as drop-in replacements.

3.4. When TDM Is Better?

We believe that TDM outperforms OTImputer when the data exhibit complex geometry. To demonstrate this, Figure 2(a) shows the imputation of TDM on the same data shown in

[5] One may also say that f_θ is bijective or f_θ is a diffeomorphism (Papamakarios et al., 2021).

[6] As $f_1, \ldots, f_T$ are invertible, so is f_θ (Kobyzev et al., 2020)

Figure 1, where it can be observed that the imputed values of TDM (with three blocks, i.e., $T = 3$) align with the data distribution significantly better than OTImputer. To demonstrate the learned transformed spaces, we feed the data with out-of-distribution (OOD) samples generated from a two-dimensional normal distribution shown in Figure 2(b) into the learned f_θ of TDM with a succession of three blocks. Note that TDM is trained without these OOD noises. Figures 2(c-e) show the latent space after the first ($f_1(\boldsymbol{X})$), second ($f_{1:2}(\boldsymbol{X})$), and third (final) ($f_{1:3}(\boldsymbol{X})$) block, respectively. In the latent spaces, the in-domain samples are close to each other, and are well separated from the OOD samples. That explains why TDM does not have the false imputations as OTImputer, because the OOD samples are far away from the in-domain ones in terms of the quadratic distance in the latent spaces. This also demonstrates that TDM does not simply push all the points in the data space close to each other and model collapsing is avoided. In Figure 7 of the appendix, we show two additional synthetic datasets, whose geometry is simpler than the previous ones. In these simpler cases, one can see that OTImputer is able to correctly impute the missing values as TDM. This is because the quadratic distance in the data space used in OTImputer can capture the data geometry well. In these cases, TDM does not have to learn a complex series of transformations. Therefore, the transformed spaces of TDM look similar to the data space, i.e., is close to an isometry, which leads TDM to reduce to OTImputer.

3.5. Implementation Details

As discussed before, we only need to minimise the Wasserstein distance without explicitly maximising the mutual information, which requires f_θ to be invertible, denoted as $f_\theta \in \mathcal{F}^{\text{INN}}$. Our final loss then becomes:

$$\min_{\boldsymbol{X}^{1\cup2}[\boldsymbol{M}^{1\cup2}],\theta} \mathcal{L}^W \text{ s.t. } f_\theta \in \mathcal{F}^{\text{INN}}, \tag{8}$$

where θ consists of parameters of the neural networks g_1, g_2, h_1, h_2 used in every block of f_θ.

Computation of Wasserstein Distances The exact computation of Wasserstein distances can be done by network simplex methods that take $\mathcal{O}(D^3)$ (D is the feature dimension of $\boldsymbol{X}^1$ and $\boldsymbol{X}^2$) (Ahuja et al., 1995). Muzellec et al. (2020) uses Sinkhorn iterations (Cuturi, 2013) with the entropic regularisation to compute the 2-Wasserstein distances in $\mathcal{O}(D^2 \log D)$ (Altschuler et al., 2017; Dvurechensky et al., 2018): $\hat{d}_{\boldsymbol{G}}\left(\mu(\boldsymbol{X}^1), \mu(\boldsymbol{X}^2)\right) := d_{\boldsymbol{G}}\left(\mu(\boldsymbol{X}^1), \mu(\boldsymbol{X}^2)\right) + \epsilon r(\boldsymbol{P})$, where $r(\boldsymbol{P})$ is the negative entropy of the transport plan and ϵ is set in an ad-hoc manner: 5% of the median distance between initialised values in each dataset. We find that ϵ is critical to the imputation results and the ad-hoc setting may achieve sub-optimal performance. To avoid selecting ϵ, instead of Sinkhorn iterations, we use the network simplex method (Bonneel et al., 2011) efficiently implemented in the POT package (Flamary et al., 2021). Theoretically, the network simplex method has $\mathcal{O}(D^3)$ complexity, however, Bonneel et al. (2011) reports it behaves in $\mathcal{O}(D^2)$ in practice. Unlike W_2 used in TDM, neither $\hat{d}_{\boldsymbol{G}}$ nor the Sinkhorn divergence (Genevay et al., 2018) used in Muzellec et al. (2020) is guaranteed to be a metric distance.

Algorithm 1: TDM. Learnable parameters include missing values $\boldsymbol{X}[\boldsymbol{M}]$ and parameters θ of f.

input : Data $\boldsymbol{X}$ with missing values indicated by $\boldsymbol{M}$
output : $\boldsymbol{X}$ with $\boldsymbol{X}[\boldsymbol{M}]$ imputed, f_θ
Initialise θ of f;
 # Initialise missing values with noisy mean #
$\boldsymbol{X}[\boldsymbol{M}] \leftarrow \text{nanmean}(\boldsymbol{X}, \text{dim=0}) + \mathcal{N}(0, 0.1)$;
while *Not converged* **do**
  Sample two batches of B data samples $\boldsymbol{X}^1$ and $\boldsymbol{X}^2$;
  Feed $\boldsymbol{X}^1$ and $\boldsymbol{X}^2$ to f_θ;
  for $i = 1 \dots B$, $j = 1 \dots K$ **do**
    Compute $\boldsymbol{G}'[i, j]$; *# Quadratic cost function #*
  end
  Compute $\mathcal{L}^W$;
  Update the missing values $\boldsymbol{X}^{1\cup2}[\boldsymbol{M}^{1\cup2}]$ and θ with gradient update;
end

Computation of Gradients Recall that in Eq. (1) OT/Wasserstein distances are computed by finding the optimal transport plan $\boldsymbol{P} \in \mathbb{R}_{>0}^{B\times B}$ (B is the batch size in Eq. (8)). Given $\boldsymbol{P}$, if a quadratic cost function is used, the gradient of $\mathcal{L}^W$ in terms of $f_\theta(\boldsymbol{X}^1[i,:])$ ($\boldsymbol{X}^1[i,:]$ is the i^{th} sample of $\boldsymbol{X}^1$) is (Cuturi & Doucet, 2014; Muzellec et al., 2020) :

$$\frac{\partial \mathcal{L}^W}{\partial f_\theta(\boldsymbol{X}^1[i,:])} = \sum_{j=1}^{B} \boldsymbol{P}[i,j]\left(f_\theta(\boldsymbol{X}^1[i,:]) - f_\theta(\boldsymbol{X}^2[j,:])\right),$$

with which, one can use backpropagation to update θ and the missing values in $\boldsymbol{X}^1[i,:]$. The algorithm of our proposed method is shown in Algorithm 1.

4. Related Work

As missing values are ubiquitous in many domains, missing value imputation has been an active research area (Little & Rubin, 2019; Mayer et al., 2019). Existing methods can be categorised differently from different aspects (Muzellec et al., 2020) such as the type of the missing variables (e.g., real, categorical, or mixed) the mechanisms of missing data (e.g., MCAR, MAR, or MNAR discussed in Section 2.1). In this paper, we focus on imputing real-valued missing data. Besides simple baselines such as imputation with mean/median/most-frequent values, we introduce the related

Figure 3. MAE in the MCAR (top) and MAR (bottom) settings.

work from the perspective of whether a method treats data features (i.e., the columns in data $\boldsymbol{X} \in \mathbb{R}^{N \times D}$) separately or jointly, following Jarrett et al. (2022).

Methods in the former category estimate the distributions of one feature conditioned on the other features and perform iterative imputations for one feature at a time, such as in Heckerman et al. (2000); Raghunathan et al. (2001); Gelman (2004); Van Buuren et al. (2006); Van Buuren & Groothuis-Oudshoorn (2011); Liu et al. (2014); Zhu & Raghunathan (2015). As each feature's conditional distribution may be different, these methods need to specify different models for them, which can be cumbersome in practice, especially when the missing values are unknown.

In the latter category, methods learn a joint distribution of all the features explicitly or implicitly. To do so, various methods have been proposed, such as the ones based on matrix completion (Mazumder et al., 2010), (variational) autoencoders (Gondara & Wang, 2017; Ivanov et al., 2018; Mattei & Frellsen, 2019; Nazabal et al., 2020; Gong et al., 2021; Peis et al., 2022), generative adversarial nets (Yoon et al., 2018; Li et al., 2018; Yoon & Sull, 2020; Dai et al., 2021; Fang & Bao, 2022), graph neural networks (You et al., 2020; Vinas et al., 2021; Chen et al., 2022; Huang et al., 2022; Morales-Alvarez et al., 2022; Gao et al., 2023), normalising flows (Richardson et al., 2020; Ma & Ghosh, 2021; Wang et al., 2022), and Gaussian process (Dai et al., 2022).

In addition to these two categories, several recent work proposes general refinements to existing imputation methods. For example, Wang et al. (2021) introduces data augmentation methods to improve generative methods such as those in Yoon et al. (2018); Nazabal et al. (2020); Richardson et al. (2020). Kyono et al. (2021) proposes causally-aware (Mohan et al., 2013) refinements to existing methods. Recently, Jarrett et al. (2022) proposes to automatically select an imputation method among multiple ones for each feature, which shows improved results over individual methods. Our method is a stand-alone approach and many of the above refinements can be applied to ours as well such as the data augmentation and causally-aware methods.

Among the above works, the closest one to ours is OTImputer (Muzellec et al., 2020), whose differences from ours

have been comprehensively discussed. Muzellec et al. (2020) also introduces a parametric version of OTImputer trained in a round-robin fashion. The parametric algorithm does not work as well as the standard OTImputer and our method can be easily extended with the parametric algorithm if needed. Methods based on normalising flows, e.g., MCFlow (Richardson et al., 2020) and EMFlow (Ma & Ghosh, 2021) also use INNs for imputation. However, there are fundamental differences of TDM to them, the most significant one of which is that MCFlow and EMFlow can still be viewed as deep generative models using INNs as the encoder and decoder and their losses are still reconstruction losses or maximum likelihood in the data space, while ours uses a matching distance in the latent space as the loss. Going beyond missing value imputations, a recent work by Coeurdoux et al. (2022) proposes to learn sliced-Wasserstein distances with normalising flows, which we do not consider as a close related work to ours as the primary goal, motivation, and methodology are different.

5. Experiments

5.1. Experimental Settings

Datasets Similar to many recent works (Yoon et al., 2018; Mattei & Frellsen, 2019; Muzellec et al., 2020; Jarrett et al., 2022), UCI datasets[7] with different sizes are used in the experiments, the statistics of which are shown in Table 1 of the appendix. Each dataset is standardised by the scale function of sklearn[8]. Following Muzellec et al. (2020); Jarrett et al. (2022), we generate the missing value mask for each dataset with three mechanisms in four settings, which, to our knowledge, include all the cases used in the literature. Specifically, for MCAR, we generate the mask for each data sample by drawing from a Bernoulli random variable with a fixed parameter. For MAR, we first sample a subset of features (columns in $\boldsymbol{X}$) that will not contain missing values and then we use a logistic model with these non-missing columns as input to determine the missing values of the remaining columns and we employ line search of the bias term to get the desired proportion of missing values. Finally, we also generate the masks with MNAR in two ways: 1) MNARL: Using a logistic model with the input masked by MCAR; 2) MNARQ: Randomly sampling missing values from the range of the lower and upper p^{th} percentiles. For each of the four settings, we use 30% missing rate, sample 10 masks for one dataset with different random seeds (Jarrett et al., 2022), and report the mean and standard deviation (std) of the corresponding performance metric.

Evaluation Metrics We evaluate the performance of a method by examining how close its imputation is to the ground-truth values, which is measured by the mean absolute error (MAE) and the root-mean-square error (RMSE). Following Muzellec et al. (2020), we also use the 2-Wasserstein distance, W_2^2, between the imputed and the ground-truth distributions in the data space. Here W_2^2 is similar to the loss of OTImputer in Eq. (8) and the difference is W_2^2 as a metric is computed over all the imputed and ground-truth samples[9] while OTImputer minimises W_2^2 between two sampled batches. For all the three metrics, lower values indicate better performance. Although we do not assume a specific downstream task, we conduct the evaluations of classification on the imputed data of different methods, whose settings are as follows: **1)** We remove the datasets that do not have labels (e.g., parkinsons) or have binary labels (e.g., letter. In this case, the classification performance is almost the same regardless of how the missing values are imputed). **2)** After the missing values are imputed for a dataset, we train a support vector machine with the RBF kernel and auto kernel coefficient. We report the average accuracy of 5-fold cross-validations. **3)** We report the mean and std of average accuracies in 10 runs of an imputation method with different random seeds.

Settings of Our Method To minimise the loss in Eq. (8), we use RMSprop (Tieleman et al., 2012) as the optimiser with learning rate of 10^{-2} and batch size of 512[10]. Due to the simplicity of our method, there are only two main hyperparameters to set in terms of the architecture of the transformation. The first one is the number of INN blocks T. For the affine coupling layers (Dinh et al., 2017) in each transformation, we use three fully connected layers with SELU activation for each of h_1, h_2, g_1, and g_2. The size of each fully connected layer is set to $K \times D$ where D is the feature dimension of the dataset and K is the second hyperparameter. We empirically find that $T = 3$ and $K = 2$ work well in practice and show the hyperparameter sensitivity in Appendix B. We train our method for 10,000 iterations and report the performance based on the last iteration, which is the same for all the OT-based methods.

Baselines We compare our method against four lines of ten baselines. **1)** Iterative imputation methods: ICE (Van Buuren & Groothuis-Oudshoorn, 2011) with linear/logistic models used in Muzellec et al. (2020); Jarrett et al. (2022); MissForest with random forests (Stekhoven & Bühlmann, 2012). **2)** Deep generative models: GAIN (Yoon et al., 2018)[11] using generative adversarial networks (Goodfellow et al., 2020) where the generator outputs the imputations and the discriminator classifies the imputations in an element-

[7] https://archive-beta.ics.uci.edu

[8] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.scale.html

[9] Therefore, W_2^2 cannot be reported if the number of data points is too large.

[10] If the number of data samples N is less than 512, we use $2^{\lfloor N/2 \rfloor}$, following Muzellec et al. (2020).

[11] https://github.com/jsyoon0823/GAIN

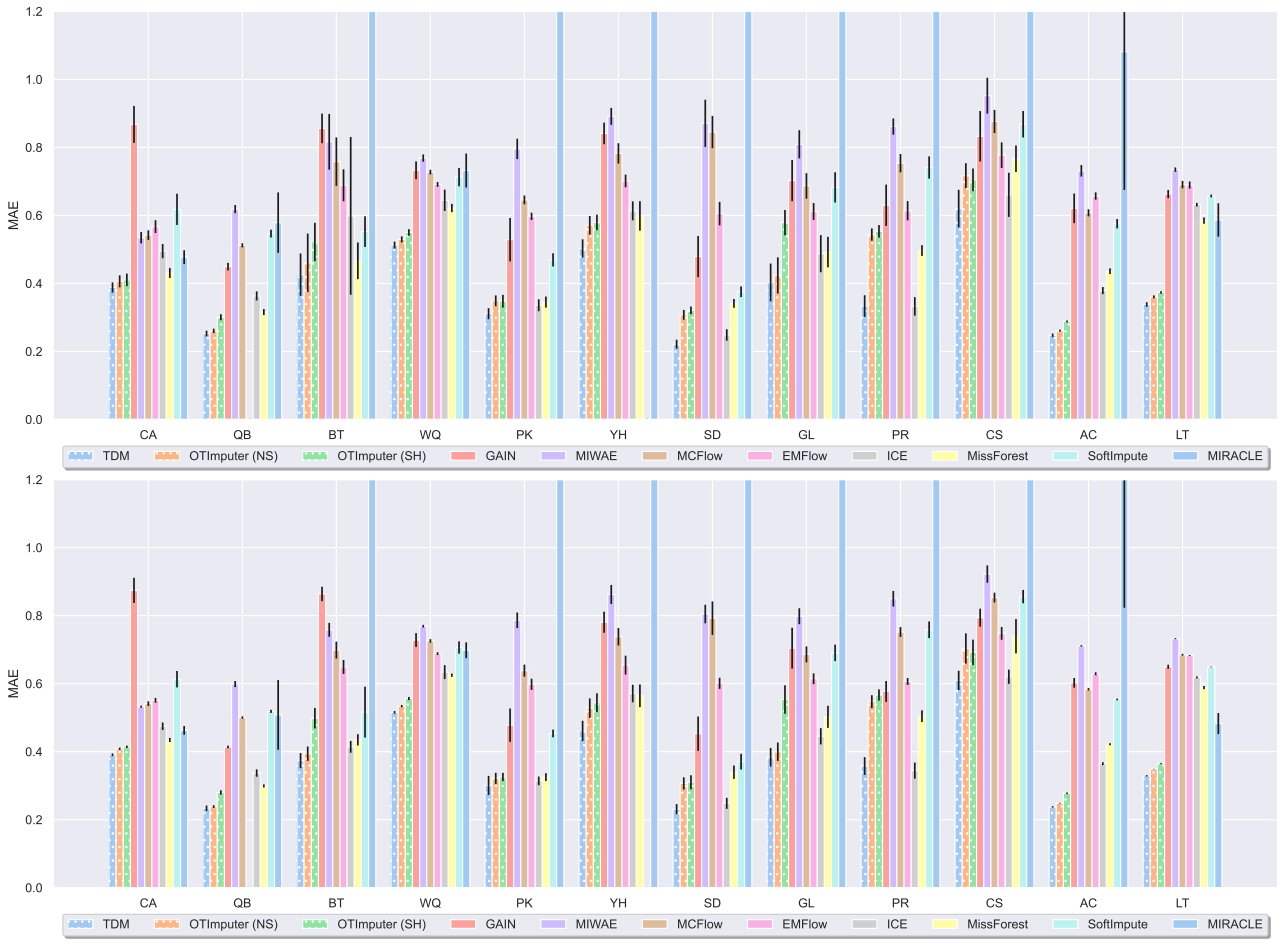

Figure 4. MAE in the MNARL (top) and MNARQ (bottom) settings.

wise fashion; MIWAE (Mattei & Frellsen, 2019)[12] extending the importance weighted autoencoders (Burda et al., 2016) for missing value imputation; MCFlow (Richardson et al., 2020)[13] and EMFlow (Ma & Ghosh, 2021)[14] extending normalising flows for imputation. **3)** Methods based on OT: OTImputer (SH), the original implementation of OTImputer (Muzellec et al., 2020)[15] where OT distances are computed by Sinkhorn iterations; OTImputer (NS), the same to OTImputer (SH) except that the OT distances are computed by the network simplex methods with POT (Flamary et al., 2021). **4)** Other methods: SoftImpute (Hastie et al., 2015) using matrix completion and low-rank SVD for imputation; MIRACLE (Kyono et al., 2021)[16] introducing causal learning as a regulariser to refine imputations. For ICE and MissForest, we use the implementations in sklearn (Pedregosa et al., 2011)[17] For SoftImpute, we use the implementation of Jarrett et al. (2022), which follows the original implementation. For the other methods, we use their original implementations (links of code listed above) with the best reported settings.

5.2. Results

Now we show the MAE results[18] in the four missing value settings in Figures 3 and 4. The results of RMSE and W_2^2 are shown in Figures 8 and 9 of the appendix. From these results, it can be observed that our proposed method, TDM, consistently achieves the best results in comparison with others in almost all the settings, metrics, and datasets. Specifically, for OT-based methods, we can see that one may gain

[12] https://github.com/pamattei/miwae

[13] https://github.com/trevor-richardson/MCFlow

[14] https://openreview.net/attachment?id=bmGLlsX_iJl&name=supplementary_material

[15] https://github.com/BorisMuzellec/MissingDataOT

[16] https://github.com/vanderschaarlab/MIRACLE

[17] https://scikit-learn.org/stable/modules/impute.html

[18] The empty results are due to the failure of running the code.

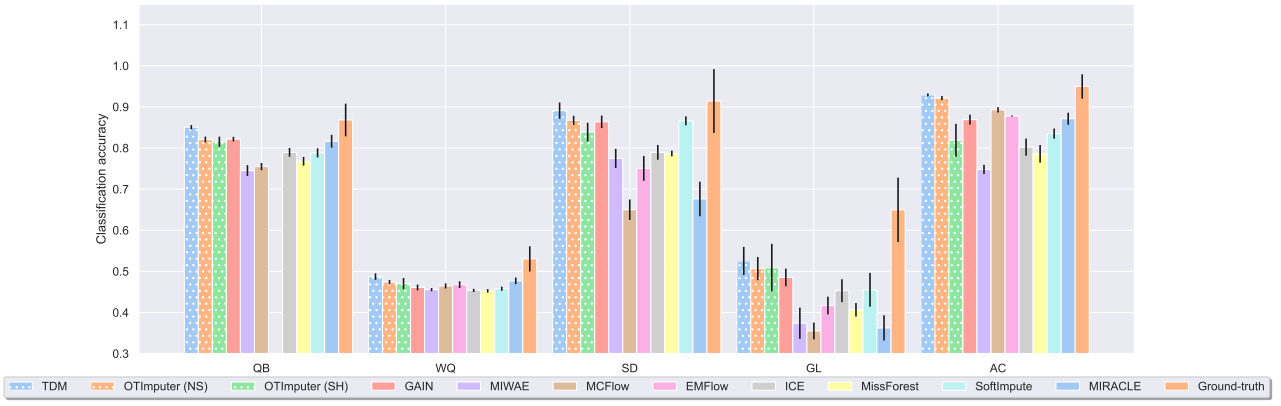

Figure 5. Classification accuracy in the MCAR setting.

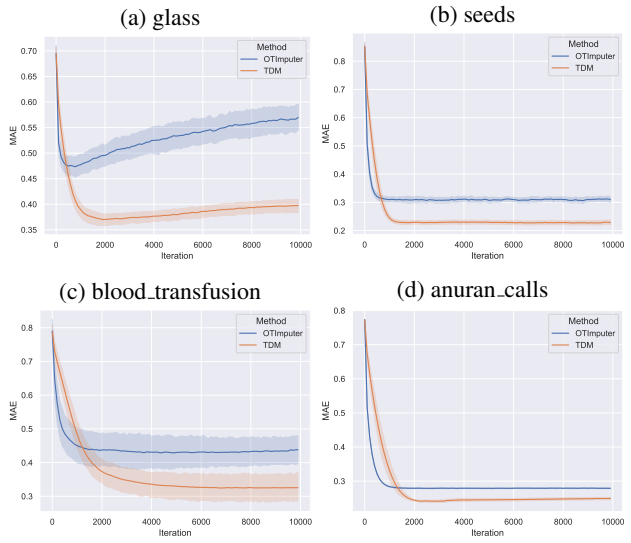

Figure 6. MAE over training iterations of TDM and OTImputer on four datasets in MCAR. The results are averaged over 10 runs.

marginal yet consistent improvement in most cases by using the network simplex methods to compute the OT distance in the comparison between OTImputer (NS) and OTImputer (SH). With the help of the learned transformations, TDM significantly outperforms both OT methods. Note that shown in Eq. (2), OTImputer directly minimises the metric of W_2^2 between two batches in the data space. Alternatively, TDM minimises the Wasserstein distance in the transformed space. Interestingly, although TDM does not directly minimises W_2^2 in the data space, it outperforms OTImputer on W_2^2 (shown in the appendix). In the comparison with MCFlow and EMFlow that also use INNs as ours, their performance is not as good as TDM's.

Figure 5 shows the classification accuracy with the imputed data by different approaches in the MCAR setting (the other settings are shown in Figure 10 of the appendix). We also report the accuracy on the ground-truth data as a reference. From the results, it can be seen that TDM in general performs the best in the classification task, showing that good imputations do help with downstream tasks.

Figure 6 shows the MAE over the training iterations of TDM and OTImputer on four datasets in the MCAR settings (the other metrics and settings are shown in Figures 17, 18, 19, and 20 of the appendix). It can be observed that TDM converges slightly slower than OTImputer as a function of the number of iterations, as TDM additionally learns the deep transformations. The average running time (seconds) per iteration for the two methods in the same computing environment is as follows: glass: OTImputer (1.14), TDM (3.20); seeds: OTImputer (1.03), TDM (3.19); blood_transfusion: OTImputer (2.39), TDM (3.35); anuran_calls: OTImputer (2.33), TDM (4.06). The running time per iteration of TDM is about 2 to 3 times that of OTImputer. For larger datasets, the running time gap between the two appears to be smaller. In several datasets, e.g., glass, OTImputer exhibits overfitting, while TDM is more stable during training.

6. Conclusion

We propose transformed distribution matching (TDM) for missing value imputation. TDM matches data samples in a transformed space, where the distance of two samples is expected to better reflect their (dis)similarity under the geometry of the data considered. The transformations are implemented with invertible neural networks to avoid model collapsing. By minimising the Wasserstein distance between the transformed samples, TDM learns the transformations and imputes the missing values simultaneously. Extensive experiments show that our method significantly improves over previous approaches, achieving state-of-the-art performance. The limitations of TDM include: **1)** Due to the learning of neural networks, TDM is relatively slower than OTImputer. **2)** TDM (OTImputer as well) does not work with categorical data. We leave the development of more efficient and applicable algorithms of TDM to future work.

References

Ahuja, R. K., Magnanti, T. L., Orlin, J. B., and Reddy, M. Applications of network optimization. *Handbooks in Operations Research and Management Science*, 7:1–83, 1995.

Altschuler, J., Niles-Weed, J., and Rigollet, P. Near-linear time approximation algorithms for optimal transport via sinkhorn iteration. In *NeurIPS*, 2017.

Ardizzone, L., Kruse, J., Rother, C., and Köthe, U. Analyzing inverse problems with invertible neural networks. In *ICLR*, 2019.

Bachman, P., Hjelm, R. D., and Buchwalter, W. Learning representations by maximizing mutual information across views. In *NeurIPS*, 2019.

Barnard, J. and Meng, X.-L. Applications of multiple imputation in medical studies: from AIDS to NHANES. *Statistical methods in medical research*, 8(1):17–36, 1999.

Barthe, F. and Bordenave, C. *Combinatorial Optimization Over Two Random Point Sets*, pp. 483–535. Springer International Publishing, Heidelberg, 2013.

Bonneel, N., Van De Panne, M., Paris, S., and Heidrich, W. Displacement interpolation using Lagrangian mass transport. In *SIGGRAPH Asia*, pp. 1–12, 2011.

Bui, A. T., Le, T., Tran, Q. H., Zhao, H., and Phung, D. A unified Wasserstein distributional robustness framework for adversarial training. In *ICLR*, 2022.

Burda, Y., Grosse, R. B., and Salakhutdinov, R. Importance weighted autoencoders. In *ICLR*, 2016.

Chen, K., Liang, X., Zhang, Z., and Ma, Z. GEDI: A graph-based end-to-end data imputation framework. *arXiv preprint arXiv:2208.06573*, 2022.

Coeurdoux, F., Dobigeon, N., and Chainais, P. Learning optimal transport between two empirical distributions with normalizing flows. In *ECML PKDD*, 2022.

Cuturi, M. Sinkhorn distances: Lightspeed computation of optimal transport. In *NeurIPS*, 2013.

Cuturi, M. and Doucet, A. Fast computation of Wasserstein barycenters. In *ICML*, pp. 685–693, 2014.

Dai, Z., Bu, Z., and Long, Q. Multiple imputation via generative adversarial network for high-dimensional blockwise missing value problems. In *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 791–798, 2021.

Dai, Z., Bu, Z., and Long, Q. Multiple imputation with neural network Gaussian process for high-dimensional incomplete data. In *ACML*, 2022.

Dinh, L., Krueger, D., and Bengio, Y. NICE: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using Real NVP. In *ICLR*, 2017.

Dvurechensky, P., Gasnikov, A., and Kroshnin, A. Computational optimal transport: Complexity by accelerated gradient descent is better than by Sinkhorn's algorithm. In *ICML*, pp. 1367–1376, 2018.

Fang, F. and Bao, S. FragmGAN: Generative adversarial nets for fragmentary data imputation and prediction. *arXiv preprint arXiv:2203.04692*, 2022.

Feydy, J., Séjourné, T., Vialard, F.-X., Amari, S.-i., Trouvé, A., and Peyré, G. Interpolating between optimal transport and MMD using Sinkhorn divergences. In *AISTATS*, pp. 2681–2690, 2019.

Flamary, R., Courty, N., Gramfort, A., Alaya, M. Z., Boisbunon, A., Chambon, S., Chapel, L., Corenflos, A., Fatras, K., Fournier, N., Gautheron, L., Gayraud, N. T., Janati, H., Rakotomamonjy, A., Redko, I., Rolet, A., Schutz, A., Seguy, V., Sutherland, D. J., Tavenard, R., Tong, A., and Vayer, T. POT: Python optimal transport. *JMLR*, 22(78): 1–8, 2021.

Gao, Z., Niu, Y., Cheng, J., Tang, J., Xu, T., Zhao, P., Li, L., Tsung, F., and Li, J. Handling missing data via max-entropy regularized graph autoencoder. In *AAAI*, 2023.

Ge, Z., Liu, S., Li, Z., Yoshie, O., and Sun, J. OTA: Optimal transport assignment for object detection. In *CVPR*, pp. 303–312, 2021.

Gelman, A. Parameterization and Bayesian modeling. *Journal of the American Statistical Association*, 99(466):537–545, 2004.

Genevay, A., Peyré, G., and Cuturi, M. Learning generative models with Sinkhorn divergences. In *AISTATS*, pp. 1608–1617, 2018.

Gomez, A. N., Ren, M., Urtasun, R., and Grosse, R. B. The reversible residual network: Backpropagation without storing activations. In *NeurIPS*, 2017.

Gondara, L. and Wang, K. Multiple imputation using deep denoising autoencoders. *arXiv preprint arXiv:1705.02737*, 280, 2017.

Gong, Y., Hajimirsadeghi, H., He, J., Durand, T., and Mori, G. Variational selective autoencoder: Learning from partially-observed heterogeneous data. In *AISTATS*, pp. 2377–2385, 2021.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

Guo, D., Tian, L., Zhang, M., Zhou, M., and Zha, H. Learning prototype-oriented set representations for meta-learning. In *ICLR*, 2021.

Guo, D., Li, Z., Zhao, H., Zhou, M., and Zha, H. Learning to re-weight examples with optimal transport for imbalanced classification. In *NeurIPS*, 2022a.

Guo, D., Tian, L., Zhao, H., Zhou, M., and Zha, H. Adaptive distribution calibration for few-shot learning with hierarchical optimal transport. In *NeurIPS*, 2022b.

Guo, D., Zhao, H., Zheng, H., Tanwisuth, K., Chen, B., Zhou, M., et al. Representing mixtures of word embeddings with mixtures of topic embeddings. In *ICLR*, 2022c.

Hastie, T., Mazumder, R., Lee, J. D., and Zadeh, R. Matrix completion and low-rank SVD via fast alternating least squares. *JMLR*, 16(1):3367–3402, 2015.

Heckerman, D., Chickering, D. M., Meek, C., Rounthwaite, R., and Kadie, C. Dependency networks for inference, collaborative filtering, and data visualization. *JMLR*, 1 (Oct):49–75, 2000.

Hjelm, R. D., Fedorov, A., Lavoie-Marchildon, S., Grewal, K., Bachman, P., Trischler, A., and Bengio, Y. Learning deep representations by mutual information estimation and maximization. In *ICLR*, 2019.

Huang, B., Zhu, Y., Usman, M., Zhou, X., and Chen, H. Graph neural networks for missing value classification in a task-driven metric space. *TKDE*, 2022.

Huynh, V., Zhao, H., and Phung, D. OTLDA: A geometry-aware optimal transport approach for topic modeling. In *NeurIPS*, volume 33, pp. 18573–18582, 2020.

Ivanov, O., Figurnov, M., and Vetrov, D. Variational autoencoder with arbitrary conditioning. In *ICLR*, 2018.

Jacobsen, J.-H., Behrmann, J., Zemel, R., and Bethge, M. Excessive invariance causes adversarial vulnerability. In *ICLR*, 2019.

Jarrett, D., Cebere, B. C., Liu, T., Curth, A., and van der Schaar, M. HyperImpute: Generalized iterative imputation with automatic model selection. In *ICML*, pp. 9916–9937, 2022.

Kingma, D. P. and Dhariwal, P. Glow: Generative flow with invertible 1x1 convolutions. In *NeurIPS*, 2018.

Klambauer, G., Unterthiner, T., Mayr, A., and Hochreiter, S. Self-normalizing neural networks. *NeurIPS*, 30, 2017.

Kobyzev, I., Prince, S. J., and Brubaker, M. A. Normalizing flows: An introduction and review of current methods. *IEEE TPAMI*, 43(11):3964–3979, 2020.

Kraskov, A., Stögbauer, H., and Grassberger, P. Estimating mutual information. *Physical review E*, 69(6):066138, 2004.

Kyono, T., Zhang, Y., Bellot, A., and van der Schaar, M. MIRACLE: Causally-aware imputation via learning missing data mechanisms. In *NeurIPS*, volume 34, pp. 23806–23817, 2021.

Li, S. C.-X., Jiang, B., and Marlin, B. MisGAN: Learning from incomplete data with generative adversarial networks. In *ICLR*, 2018.

Linsker, R. Self-organization in a perceptual network. *Computer*, 21(3):105–117, 1988.

Little, R. J. and Rubin, D. B. *Statistical analysis with missing data*, volume 793. John Wiley & Sons, 2019.

Liu, J., Gelman, A., Hill, J., Su, Y.-S., and Kropko, J. On the stationary distribution of iterative imputations. *Biometrika*, 101(1):155–173, 2014.

Ma, Q. and Ghosh, S. K. EMFlow: Data imputation in latent space via em and deep flow models. *arXiv preprint arXiv:2106.04804*, 2021.

Mattei, P.-A. and Frellsen, J. MIWAE: Deep generative modelling and imputation of incomplete data sets. In *ICML*, pp. 4413–4423, 2019.

Mayer, I., Sportisse, A., Josse, J., Tierney, N., and Vialaneix, N. R-miss-tastic: A unified platform for missing values methods and workflows. *arXiv preprint arXiv:1908.04822*, 2019.

Mazumder, R., Hastie, T., and Tibshirani, R. Spectral regularization algorithms for learning large incomplete matrices. *JMLR*, 11:2287–2322, 2010.

Mohan, K., Pearl, J., and Tian, J. Graphical models for inference with missing data. In *NeurIPS*, volume 26, 2013.

Morales-Alvarez, P., Gong, W., Lamb, A., Woodhead, S., Jones, S. P., Pawlowski, N., Allamanis, M., and Zhang, C. Simultaneous missing value imputation and structure learning with groups. In *NeurIPS*, 2022.

Muzellec, B., Josse, J., Boyer, C., and Cuturi, M. Missing data imputation using optimal transport. In *ICML*, pp. 7130–7140, 2020.

Nazabal, A., Olmos, P. M., Ghahramani, Z., and Valera, I. Handling incomplete heterogeneous data using VAEs. *Pattern Recognition*, 107:107501, 2020.

Nguyen, T., Le, T., Zhao, H., Tran, Q. H., Nguyen, T., and Phung, D. Most: Multi-source domain adaptation via optimal transport for student-teacher learning. In *UAI*, pp. 225–235, 2021.

Nguyen, T., Nguyen, V., Le, T., Zhao, H., Tran, Q. H., and Phung, D. Cycle class consistency with distributional optimal transport and knowledge distillation for unsupervised domain adaptation. In *UAI*, pp. 1519–1529, 2022.

Nguyen, X. Wasserstein distances for discrete measures and convergence in nonparametric mixture models. *arXiv preprint arXiv:1109.3250v1*, 2011.

Oord, A. v. d., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Papamakarios, G., Nalisnick, E. T., Rezende, D. J., Mohamed, S., and Lakshminarayanan, B. Normalizing flows for probabilistic modeling and inference. *JMLR*, 22(57): 1–64, 2021.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. Scikit-learn: Machine learning in python. *JMLR*, 12:2825–2830, 2011.

Peis, I., Ma, C., and Hernández-Lobato, J. M. Missing data imputation and acquisition with deep hierarchical models and Hamiltonian Monte Carlo. In *NeurIPS*, 2022.

Peyré, G., Cuturi, M., et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

Poole, B., Ozair, S., Van Den Oord, A., Alemi, A., and Tucker, G. On variational bounds of mutual information. In *ICML*, pp. 5171–5180, 2019.

Raghunathan, T. E., Lepkowski, J. M., Van Hoewyk, J., Solenberger, P., et al. A multivariate technique for multiply imputing missing values using a sequence of regression models. *Survey methodology*, 27(1):85–96, 2001.

Richardson, T. W., Wu, W., Lin, L., Xu, B., and Bernal, E. A. MCFLOW: Monte Carlo flow models for data imputation. In *CVPR*, pp. 14205–14214, 2020.

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

Rubin, D. B. *Multiple imputation for nonresponse in surveys*, volume 81. John Wiley & Sons, 2004.

Seaman, S., Galati, J., Jackson, D., and Carlin, J. What is meant by "missing at random"? *Statistical Science*, 28 (2):257–268, 2013.

Stekhoven, D. J. and Bühlmann, P. MissForest—nonparametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1):112–118, 2012.

Tieleman, T., Hinton, G., et al. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4 (2):26–31, 2012.

Tschannen, M., Djolonga, J., Rubenstein, P. K., Gelly, S., and Lucic, M. On mutual information maximization for representation learning. In *ICLR*, 2020.

Van Buuren, S. *Flexible imputation of missing data*. CRC press, 2018.

Van Buuren, S. and Groothuis-Oudshoorn, K. MICE: Multivariate imputation by chained equations in r. *Journal of statistical software*, 45:1–67, 2011.

Van Buuren, S., Brand, J. P., Groothuis-Oudshoorn, C. G., and Rubin, D. B. Fully conditional specification in multivariate imputation. *Journal of statistical computation and simulation*, 76(12):1049–1064, 2006.

Vinas, R., Zheng, X., and Hayes, J. A graph-based imputation method for sparse medical records. *arXiv preprint arXiv:2111.09084*, 2021.

Vo, V., Le, T., Vuong, L.-T., Zhao, H., Bonilla, E., and Phung, D. Learning directed graphical models with optimal transport. *arXiv preprint arXiv:2305.15927*, 2023.

Vuong, T.-L., Le, T., Zhao, H., Zheng, C., Harandi, M., Cai, J., and Phung, D. Vector quantized Wasserstein auto-encoder. *arXiv preprint arXiv:2302.05917*, 2023.

Wang, S., Li, J., Miao, H., Zhang, J., Zhu, J., and Wang, J. Generative-free urban flow imputation. In *CIKM*, pp. 2028–2037, 2022.

Wang, Y., Li, D., Xu, C., and Yang, M. Missingness augmentation: A general approach for improving generative imputation models. *arXiv preprint arXiv:2108.02566*, 2021.

Yoon, J., Jordon, J., and Schaar, M. GAIN: Missing data imputation using generative adversarial nets. In *ICML*, pp. 5689–5698, 2018.

Yoon, S. and Sull, S. GAMIN: Generative adversarial multiple imputation network for highly missing data. In *CVPR*, pp. 8456–8464, 2020.

You, J., Ma, X., Ding, Y., Kochenderfer, M. J., and Leskovec, J. Handling missing data with graph representation learning. In *NeurIPS*, volume 33, pp. 19075–19087, 2020.

Zhang, C., Cai, Y., Lin, G., and Shen, C. Deepemd: Differentiable earth mover's distance for few-shot learning. *TPAMI*, 2022.

Zhao, H., Phung, D., Huynh, V., Le, T., and Buntine, W. Neural topic model via optimal transport. In *ICLR*, 2021.

Zhu, J. and Raghunathan, T. E. Convergence properties of a sequential regression multiple imputation algorithm. *Journal of the American Statistical Association*, 110(511): 1112–1124, 2015.

A. Theoretical Analysis

A.1. Properties

In this section, we discuss the basic properties of the proposed imputation method. Essentially, the loss of TDM is an empirical estimation of

$$\mathbb{E}W_2^2(f_{\#}\mu(\boldsymbol{X}^1), f_{\#}\mu(\boldsymbol{X}^2)), \tag{9}$$

where $\mathbb{E}$ denotes the expectation wrt the random mini-batches $\boldsymbol{X}^1$ and $\boldsymbol{X}^2$ that are sampled independently. We assume each sample in each random mini-batch is sampled independently and identically based on the uniform distribution on $\{1, 2, \cdots, N\}$.

Proposition A.1.

$$\mathbb{E}W_2^2(f_{\#}\mu(\boldsymbol{X}^1), f_{\#}\mu(\boldsymbol{X}^2)) \geq \mathbb{E}W_2^2(f_{\#}\mu(\boldsymbol{X}^1), f_{\#}\mu(\boldsymbol{X})),$$

where $f_{\#}\mu(\boldsymbol{X}) := \frac{1}{N}\sum_{i=1}^{N} \delta_{f_\theta(\boldsymbol{X}[i,:])}$ *is the pushforward empirical measure with respect to all observed samples in* $\boldsymbol{X}$.

Proof. We first show that $\mu \to W_2^2(\mu, \nu)$ is convex. Consider $\mu = \lambda\mu_1 + (1-\lambda)\mu_2$, where $\lambda \in (0, 1)$. The optimal transport plan of μ_1 (resp. μ_2) is $\boldsymbol{P}_1$ (resp. $\boldsymbol{P}_2$). Then, $\lambda\boldsymbol{P}_1 + (1-\lambda)\boldsymbol{P}_2$ is a valid transport plan from μ to ν. We have

$$\begin{aligned} W_2^2(\mu, \nu) &:= \inf_{\boldsymbol{P} \in \boldsymbol{U}(\mu,\nu)} \langle \boldsymbol{P}, \boldsymbol{G} \rangle \\ &\leq \langle \lambda\boldsymbol{P}_1 + (1-\lambda)\boldsymbol{P}_2, \boldsymbol{G} \rangle \\ &= \lambda\langle \boldsymbol{P}_1, \boldsymbol{G} \rangle + (1-\lambda)\langle \boldsymbol{P}_2, \boldsymbol{G} \rangle \\ &= \lambda W_2^2(\mu_1, \nu) + (1-\lambda) W_2^2(\mu_2, \nu). \end{aligned}$$

By Jensen's inequality,

$$\mathbb{E}W_2^2(f_{\#}\mu(\boldsymbol{X}^1), f_{\#}\mu(\boldsymbol{X}^2)) \geq W_2^2(f_{\#}\mu(\boldsymbol{X}^1), \mathbb{E}f_{\#}\mu(\boldsymbol{X}^2)).$$

Based on our assumption, all samples in $\boldsymbol{X}^2$ are sampled uniformly, and therefore

$$\mathbb{E}f_{\#}\mu(\boldsymbol{X}^2) = \frac{1}{N}\sum_{i=1}^{N} \delta_{f_\theta(\boldsymbol{X}[i,:])}$$

regardless of the minibatch size B. In summary,

$$\mathbb{E}W_2^2(f_{\#}\mu(\boldsymbol{X}^1), f_{\#}\mu(\boldsymbol{X}^2)) \geq \mathbb{E}W_2^2(f_{\#}\mu(\boldsymbol{X}^1), f_{\#}\mu(\boldsymbol{X})).$$

On the LHS the operator $\mathbb{E}(\cdot)$ is taken with respect to the two random batches $\boldsymbol{X}^1$ and $\boldsymbol{X}^2$; on the RHS $\mathbb{E}(\cdot)$ is with respect to $\boldsymbol{X}^1$. □

The inequality in Proposition A.1 holds for some given $\boldsymbol{X}$ (with the missing entries imputed) and f_θ. As learning goes on, both sides of the inequality change with the missing entries as well as θ, while the inequality is always valid regardless of how the missing values are set.

Our loss is a surrogate of $\mathbb{E}W_2^2(f_{\#}\mu(\boldsymbol{X}^1), f_{\#}\mu(\boldsymbol{X}))$. During learning, our imputation method tries to make the local distribution $f_{\#}\mu(\boldsymbol{X}^1)$ to be close to the global distribution $f_{\#}\mu(\boldsymbol{X})$. Similar lower- and upper-bounds for W_1 (the 1-Wasserstein distance) between empirical measures appeared in Barthe & Bordenave (2013).

We also have an upper bound of $\mathbb{E}W_2^2(f_{\#}\mu(\boldsymbol{X}^1), f_{\#}\mu(\boldsymbol{X}^2))$ through the triangle inequality.

Proposition A.2. $\forall\mu$,

$$\mathbb{E}W_2^2(f_{\#}\mu(\boldsymbol{X}^1), f_{\#}\mu(\boldsymbol{X}^2)) \leq 4\mathbb{E}W_2^2(f_{\#}\mu(\boldsymbol{X}^1), \mu). \tag{10}$$

Proof. The 2-Wasserstein distance is a metric distance and therefore satisfies the triangle inequality. We have

$$\forall\mu, \boldsymbol{X}^1, \boldsymbol{X}^2, \quad W_2^2(f_{\#}\mu(\boldsymbol{X}^1), f_{\#}\mu(\boldsymbol{X}^2)) \leq \left(W_2(f_{\#}\mu(\boldsymbol{X}^1), \mu) + W_2(f_{\#}\mu(\boldsymbol{X}^2), \mu)\right)^2.$$

Take the expectation wrt our random sampling protocol on both sides, and by noting that $\boldsymbol{X}^1$ and $\boldsymbol{X}^2$ are sampled independently, we get

$$\begin{aligned}
&\mathbb{E}W_2^2\left(f_{\#}\mu(\boldsymbol{X}^1), f_{\#}\mu(\boldsymbol{X}^2)\right)\\
&\leq \mathbb{E}\left(W_2(f_{\#}\mu(\boldsymbol{X}^1),\mu) + W_2(f_{\#}\mu(\boldsymbol{X}^2),\mu)\right)^2.\\
&= \mathbb{E}W_2^2(f_{\#}\mu(\boldsymbol{X}^1),\mu) + \mathbb{E}W_2^2(f_{\#}\mu(\boldsymbol{X}^2),\mu) + 2\mathbb{E}W_2(f_{\#}\mu(\boldsymbol{X}^1),\mu)\cdot\mathbb{E}W_2(f_{\#}\mu(\boldsymbol{X}^2),\mu)\\
&= 2\mathbb{E}W_2^2(f_{\#}\mu(\boldsymbol{X}^1),\mu) + 2\left(\mathbb{E}W_2(f_{\#}\mu(\boldsymbol{X}^1),\mu)\right)^2.
\end{aligned}$$

As $\left(\mathbb{E}W_2(f_{\#}\mu(\boldsymbol{X}^1),\mu)\right)^2 \leq \mathbb{E}W_2^2(f_{\#}\mu(\boldsymbol{X}^1),\mu)$, we have

$$\mathbb{E}W_2^2\left(f_{\#}\mu(\boldsymbol{X}^1), f_{\#}\mu(\boldsymbol{X}^2)\right) \leq 4\mathbb{E}W_2^2(f_{\#}\mu(\boldsymbol{X}^1),\mu).$$

□

Proposition A.2 is valid for an arbitrary measure μ. Let

$$\mu = \bar{\mu} := \arg\min_{\mu} W_2^2\left(f_{\#}\mu(\boldsymbol{X}^1),\mu\right) \tag{11}$$

be the Wasserstein barycentre of the random measure $f_{\#}\mu(\boldsymbol{X}^1)$. Then $\mathbb{E}W_2^2(f_{\#}\mu(\boldsymbol{X}^1),\bar{\mu})$ is the Wasserstein variance which bounds the loss from above.

Given $\boldsymbol{X}$, the random distance $W_2\left(f_{\#}\mu(\boldsymbol{X}^1),\bar{\mu}\right)$ is bounded. We have

$$0 \leq W_2\left(f_{\#}\mu(\boldsymbol{X}^1),\bar{\mu}\right) \leq R,$$

where $R := \max W_2(f_{\#}\mu(\boldsymbol{X}^1),\bar{\mu})$ is a constant depending on f_θ and the dataset $\boldsymbol{X}$. Therefore

$$\mathrm{var}\left(W_2\left(f_{\#}\mu(\boldsymbol{X}^1),\bar{\mu}\right)\right) \leq \frac{1}{4}R^2,$$

where $\mathrm{var}(\cdot)$ denotes the variance. By Proposition A.2 and the Popoviciu's inequality, we have

$$\begin{aligned}
\mathbb{E}W_2^2\left(f_{\#}\mu(\boldsymbol{X}^1), f_{\#}\mu(\boldsymbol{X}^2)\right) &\leq 4\left(\mathbb{E}W_2(f_{\#}\mu(\boldsymbol{X}^1),\bar{\mu})\right)^2 + 2\mathrm{var}\left(W_2\left(f_{\#}\mu(\boldsymbol{X}^1),\bar{\mu}\right)\right)\\
&\leq 4\left(\mathbb{E}W_2(f_{\#}\mu(\boldsymbol{X}^1),\bar{\mu})\right)^2 + \frac{1}{2}R^2.
\end{aligned}$$

Hence, the (expected) loss is bounded above by the 2-Wasserstein distance between the random measure $f_{\#}\mu(\boldsymbol{X}^1)$ and its center. It is therefore *a variance-like measure*.

Proposition A.3. *Let $\boldsymbol{X}^1$, $\boldsymbol{X}^2$ be independent random batches of size B, $\boldsymbol{X}^3$, $\boldsymbol{X}^4$ be independent random batches of size $2B$, then*

$$\mathbb{E}W_2^2(f_{\#}\mu(\boldsymbol{X}^3), f_{\#}\mu(\boldsymbol{X}^4)) \leq \mathbb{E}W_2^2(f_{\#}\mu(\boldsymbol{X}^1), f_{\#}\mu(\boldsymbol{X}^2)). \tag{12}$$

If $B = 1$, then

$$\mathbb{E}W_2^2(f_{\#}\mu(\boldsymbol{X}^1), f_{\#}\mu(\boldsymbol{X}^2)) = \mathbb{E}\|f_\theta(\boldsymbol{X}[i,:]) - f_\theta(\boldsymbol{X}[j,:])\|^2. \tag{13}$$

Therefore, as the batch size B increases, the loss will decrease. Eventually, when B is sufficiently large, the distance between the two measures $f_{\#}\mu(\boldsymbol{X}^1)$ and $f_{\#}\mu(\boldsymbol{X}^2)$ will be close to zero as they both become close to $f_{\#}\mu(\boldsymbol{X})$.

To prove the above proposition, we introduce the lemma below first.

Lemma A.4. *Let $\boldsymbol{X}^1$ and $\boldsymbol{X}^2$ (resp. $\boldsymbol{X}^3$ and $\boldsymbol{X}^4$) be multisets of the same size B (resp. B').*

$$(B+B')W_2^2(\mu(\boldsymbol{X}^1\cup\boldsymbol{X}^3),\mu(\boldsymbol{X}^2\cup\boldsymbol{X}^4)) \leq BW_2^2(\mu(\boldsymbol{X}^1),\mu(\boldsymbol{X}^2)) + B'W_2^2(\mu(\boldsymbol{X}^3),\mu(\boldsymbol{X}^4)). \tag{14}$$

Proof. By Lemma 3.1,

$$BW_2^2\left(\mu(\boldsymbol{X}^1),\mu(\boldsymbol{X}^2)\right)=\min_{\pi}\sum_{i=1}^{B}\|\boldsymbol{X}^1[i,:]-\boldsymbol{X}^2[\pi(i),:]\|^2,$$
$$B'W_2^2\left(\mu(\boldsymbol{X}^3),\mu(\boldsymbol{X}^4)\right)=\min_{\pi'}\sum_{j=1}^{B'}\|\boldsymbol{X}^3[j,:]-\boldsymbol{X}^4[\pi'(j),:]\|^2,$$

where the minimum is taken over all possible permutations of $(1,\cdots,B)$ (resp. $(1,\cdots,B')$). Denote the optimal permutation as $\pi^\star$ (resp. $\pi'^\star$). Then we can construct a permutation $\sigma^\star$ of the index set $(1,\cdots,B,B+1,\cdots,B+B')$ by permuting $(1,\cdots,B)$ wrt $\pi^\star$ and permuting $(B+1,\cdots,B+B')$ wrt $\pi'^\star$. Thus we have

$$\begin{aligned}(B+B')W_2^2\left(\mu(\boldsymbol{X}^1\cup\boldsymbol{X}^3),\mu(\boldsymbol{X}^2\cup\boldsymbol{X}^4)\right)&=\min_{\sigma}\sum_{i=1}^{B+B'}\|(\boldsymbol{X}^1\cup\boldsymbol{X}^3)[i,:]-(\boldsymbol{X}^2\cup\boldsymbol{X}^4)[\sigma(i),:]\|^2\\&\le\sum_{i=1}^{B+B'}\|(\boldsymbol{X}^1\cup\boldsymbol{X}^3)[i,:]-(\boldsymbol{X}^2\cup\boldsymbol{X}^4)[\sigma^\star(i),:]\|^2\\&=\sum_{i=1}^{B}\|\boldsymbol{X}^1[i,:]-\boldsymbol{X}^2[\pi^\star(i),:]\|^2+\sum_{i=1}^{B'}\|\boldsymbol{X}^3[i,:]-\boldsymbol{X}^4[\pi'^\star(i),:]\|^2\\&=BW_2^2\left(\mu(\boldsymbol{X}^1),\mu(\boldsymbol{X}^2)\right)+B'W_2^2\left(\mu(\boldsymbol{X}^3),\mu(\boldsymbol{X}^4)\right).\end{aligned}$$

□

The proof of Proposition A.3 is as follows.

Proof. In Lemma A.4, let $B'=B$, and take expectation on both sides of the inequality, then Proposition A.3 is immediate. If $B=1$, the 2-Wasserstein distance between empirical measures becomes the Euclidean distance. □

A.2. Invertibility Analysis

We rewrite Eq. 6 and Eq. 7 as

$$\begin{aligned}\boldsymbol{z}_{1:d}&=\boldsymbol{y}^{\text{in}}_{1:d}\odot\exp\left(g_1(\boldsymbol{y}^{\text{in}}_{d+1:D})\right)+h_1(\boldsymbol{y}^{\text{in}}_{d+1:D}),\\\boldsymbol{z}_{d+1:D}&=\boldsymbol{y}^{\text{in}}_{d+1:D},\\\boldsymbol{y}^{\text{out}}_{1:d}&=\boldsymbol{z}_{1:d},\\\boldsymbol{y}^{\text{out}}_{d+1:D}&=\boldsymbol{z}_{d+1:D}\odot\exp\left(g_2(\boldsymbol{z}_{1:d})\right)+h_2(\boldsymbol{z}_{1:d}),\end{aligned}$$

where $\boldsymbol{z}$ is a D-dimensional intermediate vector. Both the mappings $\boldsymbol{y}^{\text{in}}\to\boldsymbol{z}$ and $\boldsymbol{z}\to\boldsymbol{y}^{\text{out}}$ are invertible (the inverse has a simple closed form (Dinh et al., 2017) and is omitted here), and therefore $\boldsymbol{y}^{\text{in}}\to\boldsymbol{y}^{\text{out}}$ is invertible.

Note that $\boldsymbol{y}^{\text{out}}_i$ (the i'th dimension of $\boldsymbol{y}^{\text{out}}$) only depends on $\boldsymbol{y}^{\text{in}}_{1:i}$. The Jacobian of the mapping $\boldsymbol{y}^{\text{in}}\to\boldsymbol{y}^{\text{out}}$ has a *lower triangular* structure.

A.3. Proof of Proposition 3.2

Proof. By definition, we have $I(\boldsymbol{X},f'(\boldsymbol{X}))=H(\boldsymbol{X})-H(\boldsymbol{X}|f'(\boldsymbol{X}))$ where $H(\boldsymbol{X})$ and $H(\boldsymbol{X}|f'(\boldsymbol{X}))$ are the entropy and conditional entropy, respectively. As we consider $\boldsymbol{X}$ and $f'(\boldsymbol{X})$ as empirical random variables with finite supports of their samples, $H(\boldsymbol{X}|f'(\boldsymbol{X}))\ge 0$. Therefore, $I(\boldsymbol{X},f'(\boldsymbol{X}))\le H(\boldsymbol{X})=I(\boldsymbol{X},\boldsymbol{X})$. If f_θ is a smooth invertible map, it is known that: $I(\boldsymbol{X},f_\theta(\boldsymbol{X}))=I(\boldsymbol{X},\boldsymbol{X})$ (proof shown in Eq. (45) of Kraskov et al. (2004)). Therefore, $I(\boldsymbol{X},f_\theta(\boldsymbol{X}))\ge I(\boldsymbol{X},f'(\boldsymbol{X}))$. □

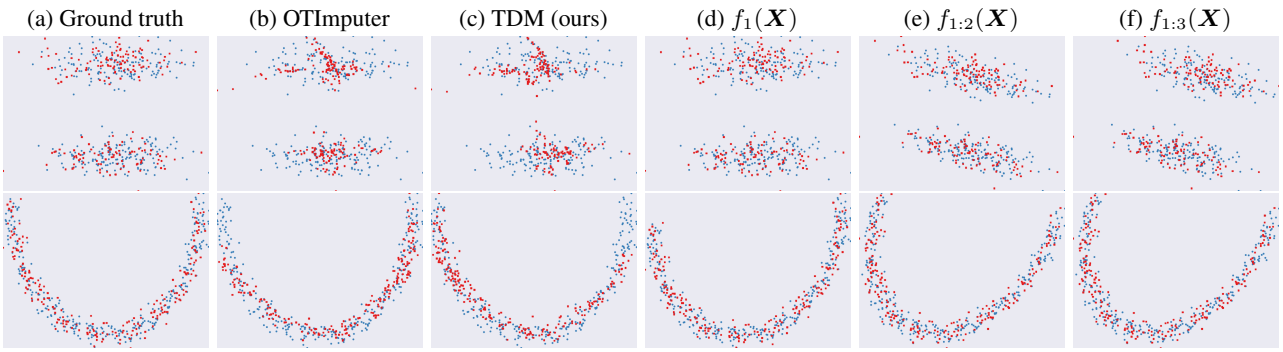

Figure 7. Two synthetic datasets (two rows) each of which is with 500 samples. (a) Ground truth: Blue points (60%) have no missing values and red points (40%) have one missing value on either coordinate (following MCAR). (b) The imputed values for the red points by OTImputer. (c) The imputed values for the red points by TDM. (d-f) The transformed points by different blocks of f_θ.

Table 1. Dataset statistics

Dataset	N	D	Abbreviation
california	20,640	8	CA
qsar_biodegradation	1,055	41	QB
blood_transfusion	748	4	BT
wine_quality	4,898	11	WQ
parkinsons	195	23	PK
yacht_hydrodynamics	308	6	YH
seeds	210	7	SD
glass	214	9	GL
planning_relax	182	12	PR
concrete_slump	103	7	CS
anuran_calls	7,195	22	AC
letter	20,000	16	LT

B. Hyperparameter Sensitivity

To test the sensitivity of TDM to the two main hyperparameters T and K, we vary them from 1 to 4 and show the imputation metrics in Figure 11, 12, 13. It can be observed that increasing T improves the performance in general but comparing $T = 4$ with $T = 3$, the improvement becomes marginal and overfitting can also be observed in a few datasets, e.g., QB and AC. In addition, one can see that varying K does not have significant impact on the performance.

We also report the sensitivity of our method to different batch sizes (128, 256, 512) in the comparison with OTImputer, shown in Figure 14, 15, 16. It can be seen that larger batch sizes in general give better performance of both TDM and OTImputer.

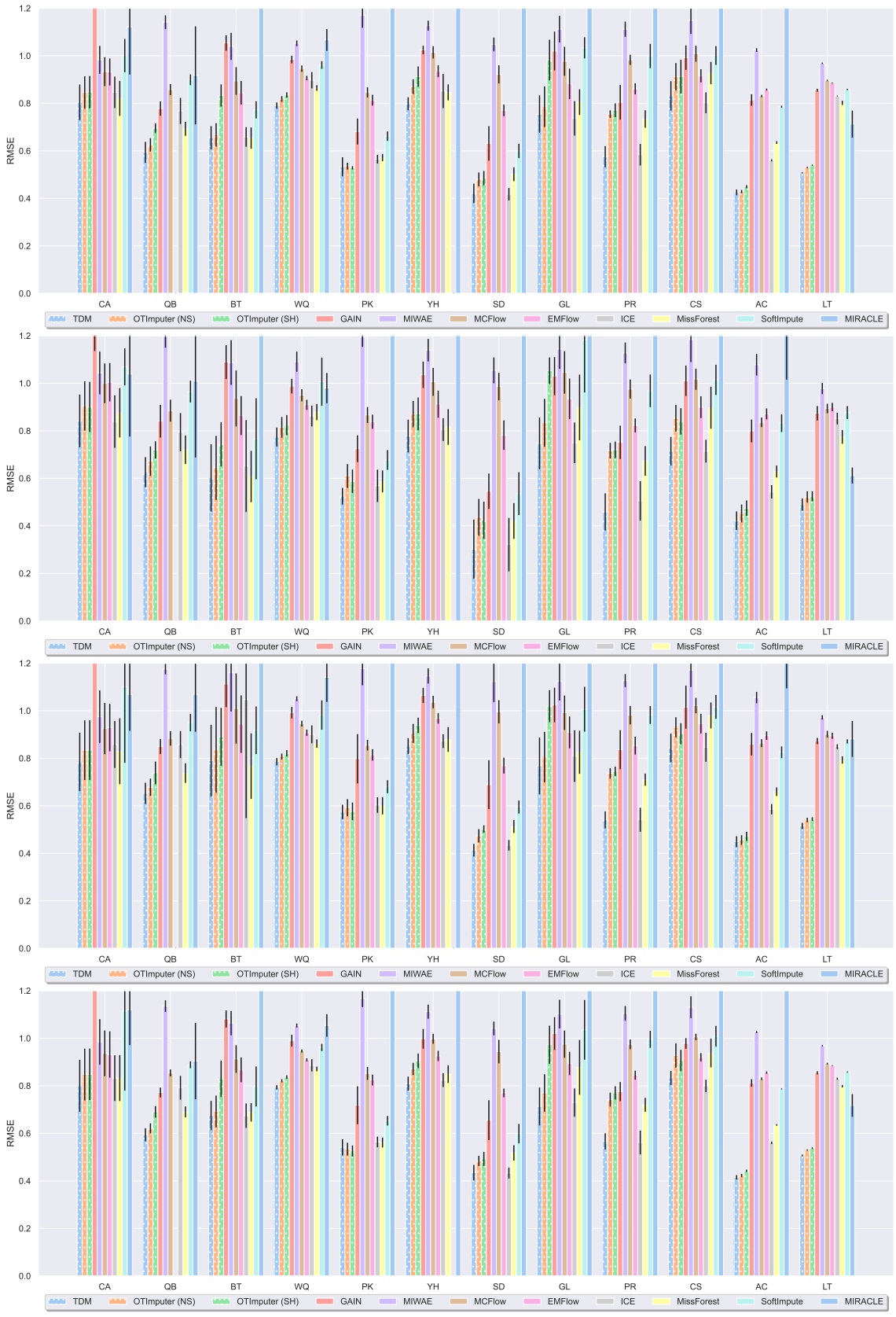

Figure 8. From top to bottom: RMSE in the MCAR, MAR, MNARL, and MNARQ settings.

Figure 9. From top to bottom: W_2^2 in the MCAR, MAR, MNARL, and MNARQ settings.

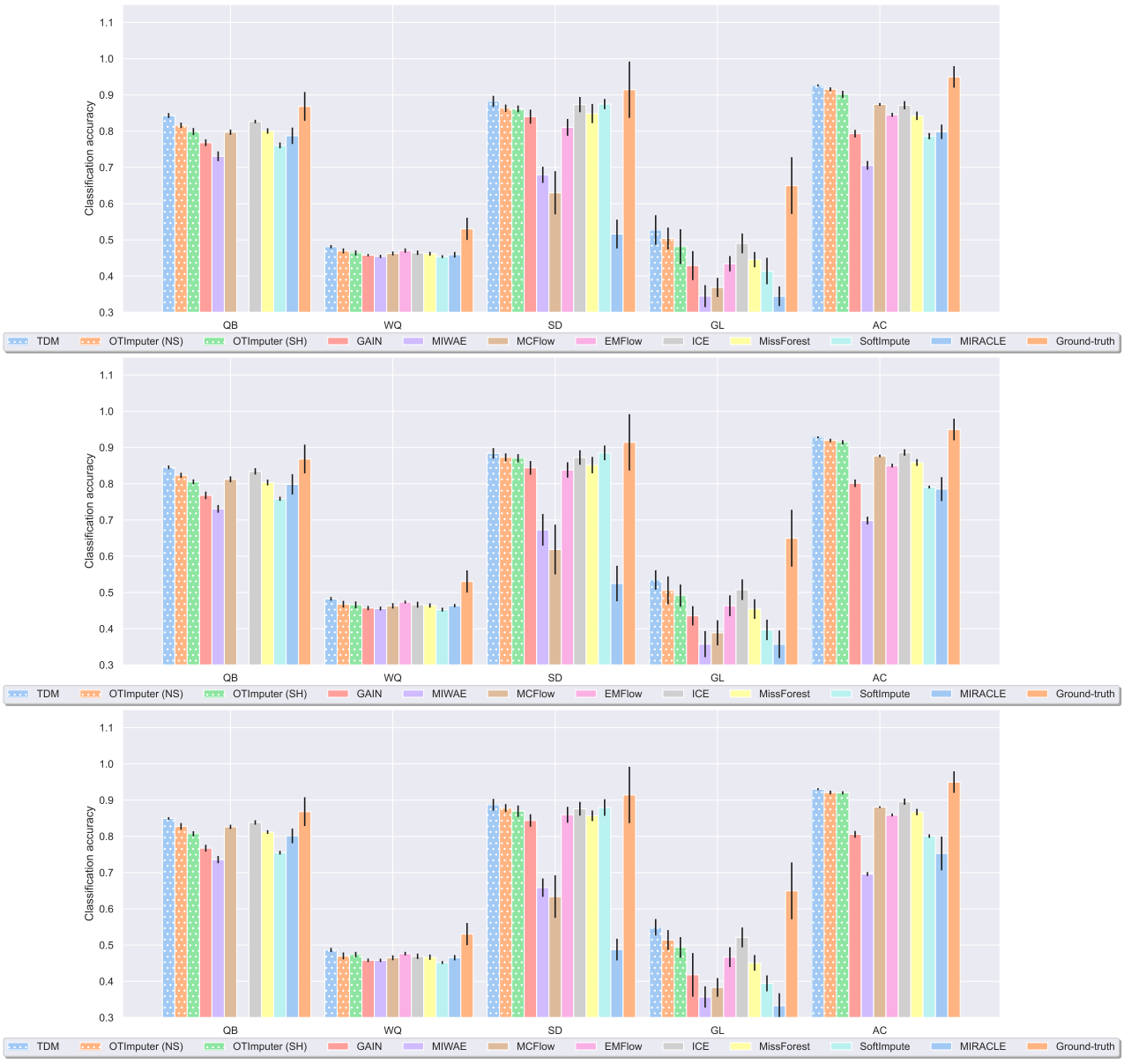

Figure 10. From top to bottom: Classification accuracy in the MAR, MNARL, and MNARQ settings.

Figure 11. From top to bottom: MAE in the MCAR, MAR, MNARL, and MNARQ settings for TDM with different T and K.

Figure 12. From top to bottom: RMSE in the MCAR, MAR, MNARL, and MNARQ settings for TDM with different T and K.

Figure 13. From top to bottom: W_2^2 in the MCAR, MAR, MNARL, and MNARQ settings for TDM with different T and K.

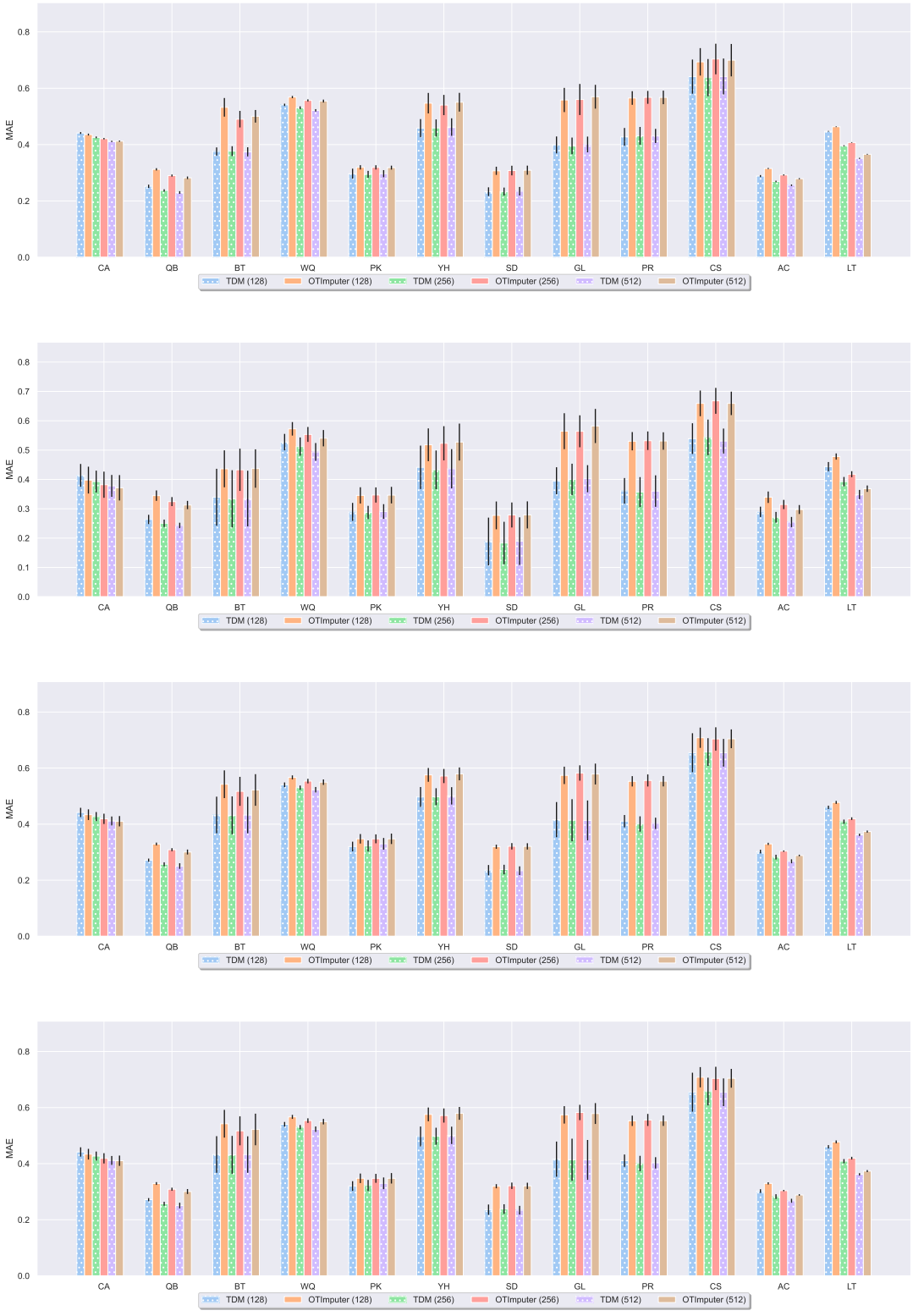

Figure 14. From top to bottom: MAE of TDM and OTImputer with batch size varied in MCAR, MAR, MNARL, and MNARQ.

Figure 15. From top to bottom: RMSE of TDM and OTImputer with batch size varied in MCAR , MAR, MNARL, and MNARQ.

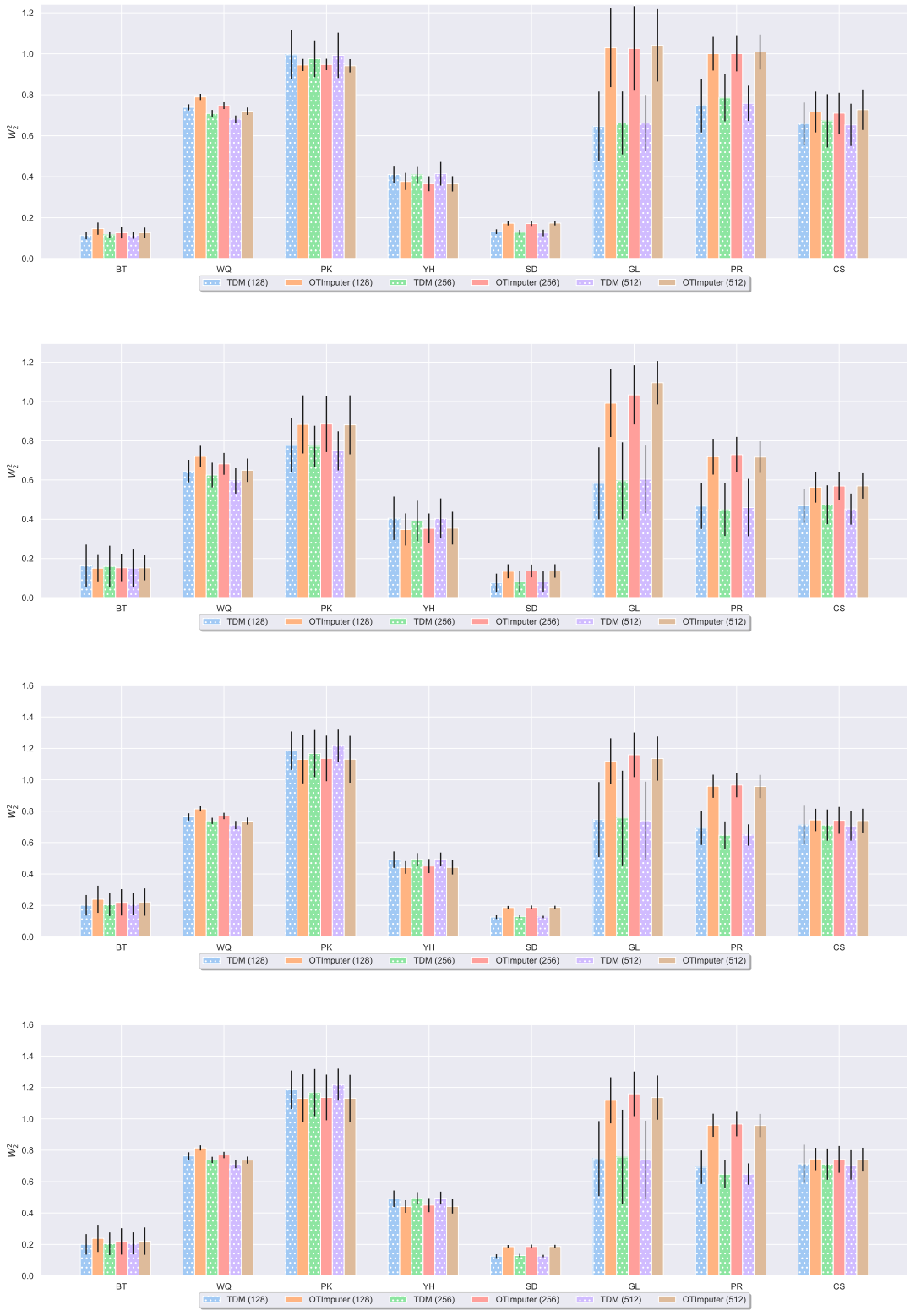

Figure 16. From top to bottom: W_2^2 of TDM and OTImputer with batch size varied in MCAR , MAR, MNARL, and MNARQ.

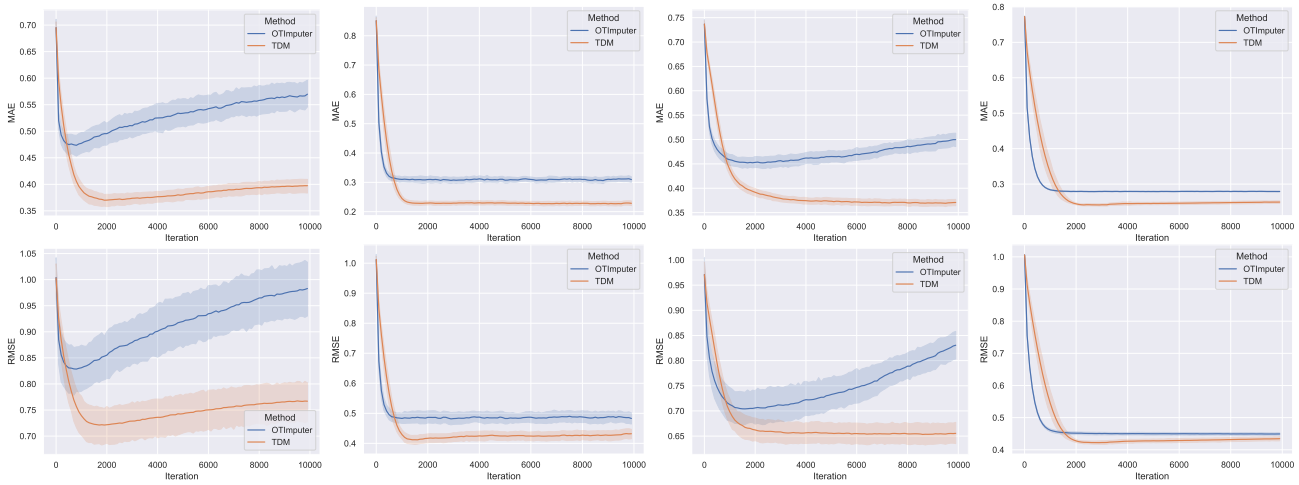

Figure 17. MAE and RMSE over training iterations of TDM and OTImputer on four datasets (from left to right: glass, seeds, blood_transfusion, anuran_calls) in MCAR.

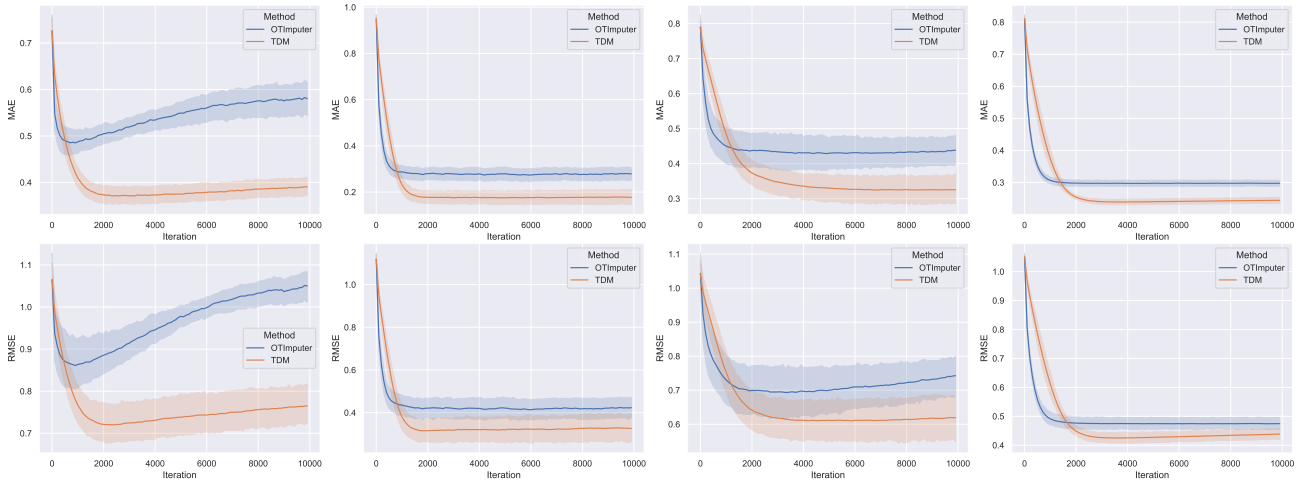

Figure 18. MAE and RMSE over training iterations of TDM and OTImputer on four datasets (from left to right: glass, seeds, blood_transfusion, anuran_calls) in MAR.

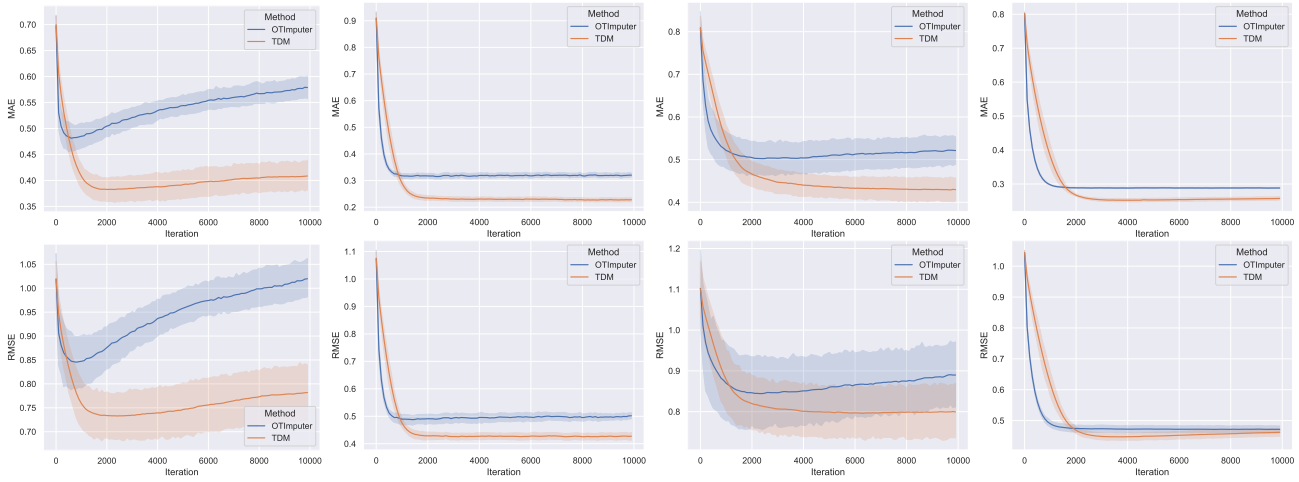

Figure 19. MAE and RMSE over training iterations of TDM and OTImputer on four datasets (from left to right: glass, seeds, blood_transfusion, anuran_calls) in MNARL.

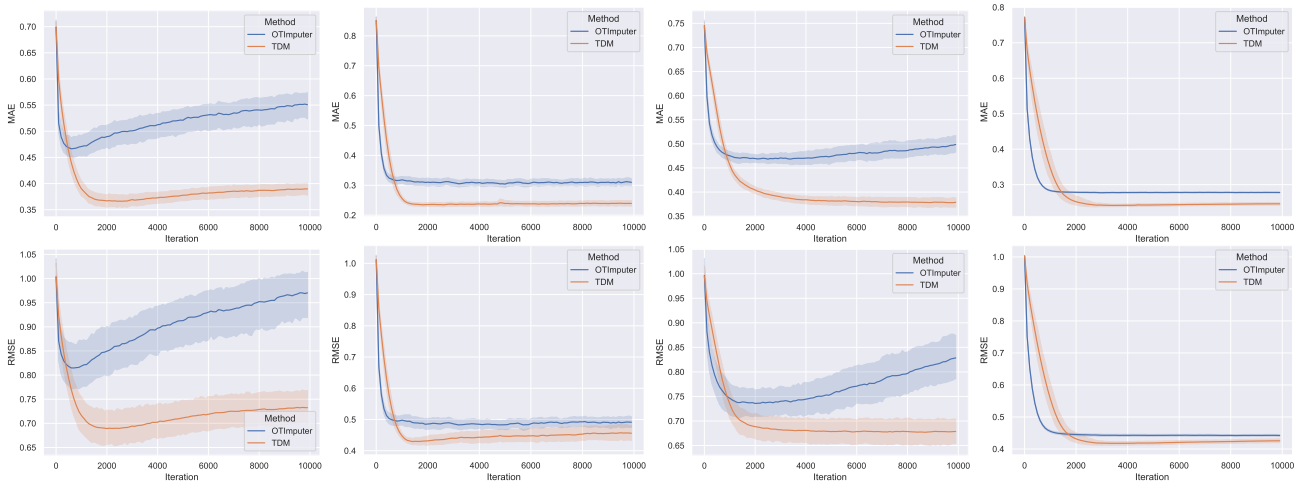

Figure 20. MAE and RMSE over training iterations of TDM and OTImputer on four datasets (from left to right: glass, seeds, blood_transfusion, anuran_calls) in MNARQ.