Imputing Structured Missing Values in Spatial Data with Clustered Adversarial Matrix Factorization

Qi Wang[1], Pang-Ning Tan[1], Jiayu Zhou[1]
[1]Computer Science and Engineering, Michigan State University, East Lansing, MI 48824.

***Abstract*—Missing data problem often poses a significant challenge as it may introduce uncertainties into the data analysis. Recent advances in matrix completion have shown competitive imputation performance when applied to many real-world domains. However, there are two major limitations when applying matrix completion methods to spatial data. First, they make a strong assumption that the entries are missing-at-random, which may not hold for spatial data. Second, they may not effectively utilize the underlying spatial structure of the data. To address these limitations, this paper presents a novel *clustered adversarial matrix factorization* method to explore and exploit the underlying cluster structure of the spatial data in order to facilitate effective imputation. The proposed method utilizes an adversarial network to learn the joint probability distribution of the variables and improve the imputation performance for the missing entries that are not randomly sampled.**

***Index Terms*—Missing value imputation, deep adversarial network, spatial data**

I. Introduction

Many real-world applications are prone to the missing data problem. For spatial data, the missing values may arise due to various reasons. For example, missing values are common in forest inventory and monitoring databases due to the prohibitive cost needed to collect data for large land areas [3]. The past decade has witnessed extensive research on data imputation, from simple statistical approaches to complicated model-based ones. The model-based approaches such as low-rank matrix completion have brought huge success to many challenging applications such as recommender systems [9], image reconstruction [19], etc. These methods leverage the low-rank property of data matrices to bridge the missing values and observed ones in a matrix. Matrix factorization is one of the most commonly used low-rank matrix completion methods. It factorizes the input matrix into a product of two lower ranked matrices (latent factors) based on their observed entries. By minimizing the reconstruction error of the observed part, the two latent factors are learned, which are then combined to estimate the missing entries [18]. Other examples of low-rank matrix completion approaches include the singular value thresholding [1], which iteratively applies truncated SVD to fill the missing values.

These matrix completion approaches, though elegantly designed, have one key assumption that the entries are missing at random [2]. However, this assumption may not hold in spatial data, which often contain structured missing patterns. For example, a Canadian study of adolescents finds that those with missing household income information are less likely to reside in high-income neighborhoods [15]. When the missing values are not randomly sampled, minimizing the reconstruction error of the observed part no longer guarantees the reconstruction of the missing part.

Another limitation of classical matrix completion methods is that they do not incorporate prior knowledge of the structures of the datasets. In many spatial studies, such prior knowledge is especially critical to model missing values [12]. For example, freshwater lakes exhibit strong natural spatial clustering structures, as lakes in a similar neighborhood are likely to have similar nutrient regeneration cycles, and thus their feature values may be similar to each other [17]. When such existing neighborhood knowledge can be correctly identified by a matrix factorization approach, it is expected to significantly improve the quality of the imputed values because the clustering structure imposes a high-quality subspace on which information is transferred among the lakes.

To address the limitations of existing imputation methods, in this paper, we propose a clustered adversarial matrix factorization framework. The proposed framework identifies a low-dimensional subspace that is consistent with the clustering structure of the spatial data, and thus, facilitates knowledge transfer among data points within the same cluster. In addition, to alleviate the challenges from structured missing data, the framework encourages the imputed samples to have a similar probability distribution as that of the complete (non-missing) data. The benefit of this distribution alignment is that it relates the observed features to the missing features of each incomplete sample through the joint probability distribution of their combined features. If the imputed values deviate significantly from their true values, the joint probability distribution of such imputed sample is likely to be small. However, since the true distribution of the data is often unknown, the proposed framework adopts an adversarial learning strategy by introducing a distribution detector to discriminate between the complete samples from imputed samples. We conduct extensive experiments on both synthetic and real-world datasets to demonstrate the effectiveness of the proposed method.

II. Related work

Multiple low-rank matrix completion algorithms have been proposed in the past years and showed great success in various applications. Truncated SVD algorithm is one of the most widely used method [10]. It iteratively applies truncated SVD on the data matrix and reconstructs the whole matrix by keeping a small number of singular values. Matrix factorization [11], [20] is another popular method. This method

factorizes the matrix into two smaller matrices (latent factors) to guarantee the low-rank structure of the matrix and the reconstruction error of the observed part is minimized. The missing entries are filled based on the entries in the product of the two latent matrices. There are also simpler matrix completion methods that do not require a low-rank assumption. For example, mean imputation uses mean values to fill the missing entries. KNN imputation [13] imputes missing values by k nearest neighbors. While all the mentioned methods work well on some dataset, their underlying assumption is that the missing entries occur randomly in the data [2].

Generative adversarial network (GAN) has been widely used for image generation [5], [16]. In [8], [14], the authors proposed to use the idea from GAN to infer arbitrary missing regions of an image based on the image semantics. While this method is quite effective to address the semantic image inpainting problems. it considers each sample independently when inpainting them. In contrast, spatial data exhibits strong spatial dependencies, which must be taken into account to improve imputation performance.

III. Methodology

A. Low-rank matrix completion

Given a matrix with missing values, matrix completion aims to estimate the missing entries of the matrix by exploiting the latent structures in its observed entries. One commonly used latent structure is the low-rank structure of the matrix, as it suggests a low-dimensional subspace to account for the redundancy in the matrix. Given a matrix with missing values, the low-rank matrix completion approaches learn a decomposition of the matrix to constrain a desired upper bound of its rank. There are convex and non-convex approaches to formulating the low-rank matrix completion problem. Convex approaches based on trace norm can guarantee a global optimal with nice theoretical properties, whereas non-convex approaches such as matrix factorization conduct a local search procedure and provide much more flexibility and efficiency. Given a matrix $X \in \mathbb{R}^{d\times n}$ with n represents sample size and d represents feature dimension, matrix factorization approximates X by UV with $U \in \mathbb{R}^{d\times r}$ and $V \in \mathbb{R}^{r\times n}$, where $r < \min(n, d)$. U and V can be solved by minimizing the reconstruction error of the observed entries as follows:

$$\min_{U,V} \frac{1}{2}\|M \odot (X - UV)\|_F^2 \quad \text{s.t. } U \in S_1, V \in S_2, \qquad (1)$$

where $\|\cdot\|_F$ denotes the Frobenius norm. S_1 and S_2 are constraints on U and V to specify the feasible regions of the two factors. $\odot$ denotes Hadamard product. M is an indicator matrix that has the same size as X. The i-th row and j-th column of M is defined as: $M_{ij} = 1$ if X_{ij} is observed, and 0 otherwise. A locally optimal solution of Eq. (1) can be obtained by a block coordinate descent procedure. Denote the local solution as U^* and V^*. Then, U^* and V^* can be used to reconstruct X and estimate the missing values as $X_r = X \odot M + (U^*V^*) \odot (1 - M)$, where X_r denotes the reconstructed matrix (imputed samples). Matrix factorization is widely used in recommender systems to estimate the ratings of users on new items based on their ratings on other items [11].

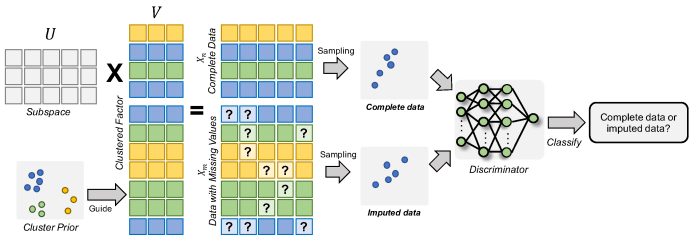


Fig. 1. Overview of clustered adversarial matrix factorization. Our proposed method utilizes the spatial clustering information and the probability distribution of complete data to improve the imputation performance. In X_m, X_n and V, each color represents one spatial cluster.

When applying this method to spatial datasets, matrix factorization does not incorporate prior knowledge on the spatial clustering structure in the datasets. However, such prior knowledge usually helps in shaping the solution space and thus leads to improved convergence to high-quality solutions. Also, for the structured missing value problem, classical matrix completion may deliver poor imputation performance, since the missing values are not random..

B. Clustered adversarial matrix factorization

To address the two limitations of matrix factorization mentioned in the last subsection, we propose a novel *clustered adversarial matrix factorization* framework. In our framework, we jointly model the clustering pattern of the entire dataset and align the probability distribution of the imputed samples to be close to the distribution of the complete samples, which is an approximation of the true data distribution.. Let X be our data matrix where each column is a data point. Some of the data points have complete feature values, while others have structured missing values. We denote the complete part as X_n and the submatrix of data points with missing values as X_m, respectively. We assume that all the samples of X are i.i.d. and each data point is sampled from a probability distribution $p_{data}(x)$. The proposed formulation contains two components: *reconstruction* and *distribution alignment*.

Reconstruction Component. In order to utilize the low-rank property and the spatial clustering structure of the data, we propose to use a ℓ_2 clustering term in matrix factorization:

$$\begin{aligned} l_r =& \frac{1}{2}\|M \odot (X - UV)\|_F^2 + \gamma_1\|U\|_F^2 \\ &+ \gamma_2\|V\|_F^2 + \gamma_3 \sum\nolimits_{i<j} d_{ij}\|v_i - v_j\|_2^2, \end{aligned} \qquad (2)$$

where v_i denote the i-th column of V, and $\gamma_1, \gamma_2, \gamma_3$ are regularization parameters. In this formula, the second and third terms are used to control model complexity and make the model robust against overfitting. The last term $\gamma_3 \sum_{i<j} d_{ij}\|v_i - v_j\|_2^2$ is used to transfer knowledge between spatial clusters. d_{ij} is a customizable similarity value between the i-th and the j-th sample, which can be used to inject prior knowledge. When d_{ij} is large, there is a high chance that v_i is close to v_j and are in the same cluster. For example, if we know two samples are likely to be in the same cluster, we can

set the corresponding d_{ij} to be large, otherwise it can be set to a smaller value. γ_3 is used to control the cluster strength on V. When γ_3 is large, more samples become similar as the columns in V become closer to each other. Imputation of one sample will borrow information from more related samples compared with that with small γ_3. When γ_3 is 0, $(v_i - v_j)$ will not be constrained for all i's and j's. This reduces the formulation to standard matrix factorization. From a projection perspective, in (2), U serves as a mapping factor to bridge X and V. V is the sample latent factor to capture the sample difference. We project X to V and add cluster constraint on V since this factor is the latent sample factor that is not affected by the feature factor.

We would like to point out the difference between the cluster constraint used in this paper and the constraint in convex clustering [7] and network lasso [6]. For convex clustering and network lasso, the cluster constraint is sum of l_p norms. If used in our case, the cluster constraint is given by $\sum_{i<j} d_{ij}\|v_i - v_j\|_2$. This constraint leads to the sample latent factor for points in the same cluster to be identical. Since the imputed values are given by $X = UV$, this means the missing features for all points in the same cluster are the same. However, in spatial data, the points in the same cluster are similar but not identical. Therefore, we use $\sum_{i<j} d_{ij}\|v_i - v_j\|_2^2$ instead to encourage the points in same cluster to be similar but not necessarily identical.

Distribution Alignment Component: We propose to use an adversarial strategy to encourage the imputed samples to have a similar probability distribution as that of the complete data. To achieve this goal, we use a discriminator to distinguish the distribution differences between the imputed and complete samples.:

$$l_d = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{x_r\sim p_r(x_r)}[\log(1 - D(x_r))] \tag{3}$$

where $p_r(x_r)$ represents the probability distribution of the imputed samples, which will be estimated from the reconstructed matrix X_r. x_r represents a data point sampled from $p_r(x_r)$. D denotes a discriminator, which in this paper, is implemented using a fully connected deep neural network with a SOFTMAX output layer. The discriminator will output a probability whether the input sample comes from the complete data or the imputed data distribution. Eq. (3) is a negative cross-entropy loss function. By maximizing l_d with respect to D, the discriminator will be trained to distinguish the complete samples from the imputed ones.

Proposed Formulation. By combining the two aforementioned components, we arrive at the following min-max problem:

$$\begin{aligned}\min_{U,V}\max_{D}\ &\frac{1}{2}\|M\odot(X-UV)\|_F^2 \\ &+\gamma_1\|U\|_F^2+\gamma_2\|V\|_F^2+\gamma_3\sum\nolimits_{i<j} d_{ij}\|v_i-v_j\|_2^2 \\ &+\lambda(\mathbb{E}_{x\sim p_{data}(x)}[\log D(x)]+\mathbb{E}_{x_r\sim p_r(x_r)}[\log(1-D(x_r))]),\end{aligned} \tag{4}$$

where λ is a parameter to balance the tradeoff between the reconstruction and distribution alignment components of the framework. When minimizing Eq. (4) with respect to U and V, the proposed formulation not only seeks a small reconstruction error on the observed portion of X, but also encourages the alignment between the probability distribution of imputed samples and that of complete samples, through the discriminator. This min-max process is similar to playing an adversarial game. On one hand, the discriminator tries to distinguish the differences in the probability distributions, whereas imputation process tries to mimic the distribution of complete samples to trick the discriminator. When the algorithm converges, the distribution of X_r, i.e., $p_r(x_r)$, will be close to the distribution of complete data, i.e., $p_{data}(x)$, given that the imputed samples are able to fool a very strong discriminator. Note that the maximization part and minimization part are connected by the imputed samples. In the minimization part, we minimize the reconstruction error of completed samples by solving U and V, which are then used to impute the missing values. Meanwhile, the minimization part encourages the imputed samples to trick the discriminator. In the maximization part, the discriminator updates itself by distinguishing the complete samples and the imputed samples obtained from the minimization part. We show the overview of our proposed clustered adversarial matrix factorization in Fig. 1. In this overview, we have 3 clusters marked by different colors.

Optimization. For the last two terms in Eq. (4), we do not know the exact forms since both the probability distributions of complete data and that of the imputed data are unknown in practice. In this case, we use the sample expectations to replace the exact expectation. At each optimization turn, we randomly sample k samples from X_n and X_r and calculate $\mathbb{E}_{x\sim p_{data}(x)}[\log D(x)]$ and $\mathbb{E}_{x_r\sim p_r(x_r)}[\log(1 - D(x_r))]$ approximately as:

$$\mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] \approx \frac{1}{k}\sum\nolimits_{i=r_1}^{r_k}\log D(X_n^i), \tag{5}$$

$$\mathbb{E}_{x_r\sim p_r(x_r)}[\log(1 - D(x_r)] \approx \frac{1}{k}\sum\nolimits_{i=q_1}^{q_k}(1-\log D(X_r^i)), \tag{6}$$

where r_1 and r_k index the first sample and last sample of the k samples sampled from X_n. q_1 and q_k index the first sample and last sample of the k samples sampled from X_r. X_n^i, X_r^i denote the i-th sample of X_n and X_r, respectively. The min-max problem in Eq. (4) can thus be solved by iteratively optimizing a minimization problem and a maximization problem as follows:

$$\max_{D}\frac{1}{k}\sum\nolimits_{i=r_1}^{r_k}\log D(X_n^i)+\frac{1}{k}\sum\nolimits_{i=q_1}^{q_k}(1-\log D(X_r^i)) \tag{P:MAX}$$

$$\begin{aligned}\min_{U,V}\ &\|M\odot(X-UV)\|_F^2+\gamma_1\|U\|_F^2+\gamma_2\|V\|_F^2 \\ &+\gamma_3\sum\nolimits_{i<j} d_{ij}\|u_i-u_j\|_2^2+\frac{\lambda}{k}\sum\nolimits_{i=q_1}^{q_k}(1-\log D(X_r^i)).\end{aligned} \tag{P:MIN}$$

The complete procedure is summarized in Algorithm 1.

How to train the network. Similar to existing adversarial frameworks, the training may face convergence challenges. To make it converge, we adopted some training strategies. We first pre-train the reconstruction part by solving Eq. (2). Then, we

use the pre-trained reconstruction model to initialize U and V, and optimize the whole network. The reconstruction error is quite low after the pre-training procedure. The procedure in Algorithm 1 mainly focuses on adjusting the probability distribution of X_r which makes it easier to train the entire network. Also, to make the discriminator strong enough, we pre-train the discriminator as well. After pre-train the discriminator, the network starts with a relative good discriminator compared with a randomly initialized discriminator. When facing such a strong discriminator, the reconstruction component is forced to learn from it and trick it. During the training, we also found that the setting $t_2 > t_1$ benefits the convergence of the adversarial training. The balance between discriminator and the reconstruction part can be observed from the score calculated as the average probability of samples being classified as completed samples in discriminator. If the two components are balanced, the scores for completed samples and the imputed samples should converge to 0.5. After the training step has been completed, if the score of completed sample is larger than 0.5, that means the reconstruction part is too weak. We can increase λ or t_2 until the two components are balanced.

Algorithm 1: The algorithm for solving the proposed clustered adversarial matrix factorization

```
for number of training iterations do
  for t_1 steps do
    Sample k samples {X_n^{r_1}, X_n^{r_2}, ...X_n^{r_k}} from X_n
    Sample k samples {X_r^{q_1}, X_r^{q_2}, ...X_r^{q_k}} from X_r
    Update discriminator by Eq. (P:MAX)
  end for
  for t_2 steps do
    Sample k samples {X_r^{q_1}, X_r^{q_2}, ...X_r^{q_k}} from X_r
    Update reconstruction component by Eq. (P:MIN)
  end for
end for
```

IV. Experiment

In this section, we evaluate the proposed method on several synthetic datasets, LAGOS dataset [17], and other benchmark spatial datasets. The methods we compared in the experiments are mean imputation (Mean), KNN [4], truncated SVD (SVD) [1], Low-rank matrix factorization (MF) [18], adversarial matrix factorization (AMF), which uses matrix factorization with the distribution alignment component, clustered matrix factorization (CMF), which uses matrix factorization with the cluster constraint, and clustered adversarial matrix factorization (CAMF), which uses matrix factorization with cluster constraint and distribution alignment component.

A. Synthetic data experiments

1) Setting 1: In the first experiment, we compare the performance of multiple methods under different missing rates. **Data synthesis and missing value generation.** We create 3 clusters by sampling V from 3 Gaussian distributions. Each cluster has 500 samples. All entries of U are randomly sampled from $0.1 \times \mathcal{N}(0, 1)$. The rank for X is 25. The feature dimension is 50. To create structured missing values, we first partition the data into two equal parts. Then, we pick one part and let the entries whose values are within certain range to be missing (this range is determined by the missing rate). Detail data synthesis process can be found in the Supplementary Materials [1].

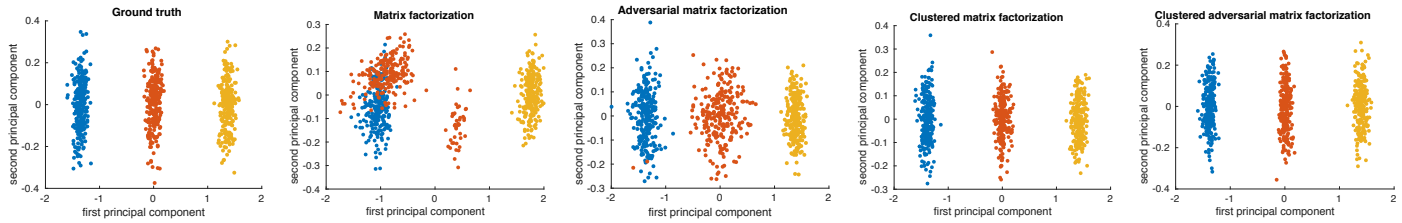
Parameters setting. γ_1, γ_2 and γ_3 are tuned over $\{1e-4, 1e-3, 1e-2, 1e-1, 1\}$. λ is tuned over {1e-2, 1e-1, 1, 5, 10, 15, 20}. For the discriminator, the nonlinear layers number are tuned over {2, 3, 4, 5}. The neuron number for all the nonlinear layers are set to be the same and tuned over {128, 256, 512}. The activation function we use is ReLU. Detail parameter settings including the similarity matrix setting can be found in the Supplementary Materials [1].

Imputation Performance. We repeat the whole process including the data synthesis part and missing value creation part for five times and perform experiments on those datasets to obtain the imputation RMSE. The results are shown in TABLE I. From the table, we see MF performs the best among all the classical matrix completion methods. For all the different missing rates, CMF outperforms MF, especially when missing rates are high. When adding the adversarial process to whether MF or CMF, the performance is better than those without the adversarial process, which shows that aligning the probability distribution could help the estimation of structured missing values. Also, we see CAMF works better than AMF. Therefore, when the data have cluster information, adding cluster information helps imputation.

Distribution Study. To show how well each method can learn the distribution of the data, we visualize the imputed samples obtained by each method, and compare them with the ground truth. We set the missing rate to be 0.7. We apply PCA on the imputed samples and visualize them on 2-d figures using the first two principal components. The results are shown in Fig. 2. In those figures, different colors represent different spatial clusters. From these figures, we see for MF, two clusters are mixed and can not be separated. After adding distribution alignment, it is much better. Three clusters can be well separated. Compared (d) with (b), we see that the cluster information helps a lot if the data has strong clustering structure. For CMF, we see the yellow cluster has smaller variance compared with the other two. But the ground truth is that three clusters should have the same variance. The probability distribution for the results of CAMF is almost identical to the ground truth.

2) Setting 2: For the second experiment, we compare the performance of different methods under different sample size. **Data synthesis and missing value generation.** In this setting, we synthesize the data with more than one latent factors and add some nonlinearity into the data to test if we can estimate the missing values only by linear clustered adversarial matrix factorization. The data are synthesized by three factors $U_1 \in \mathbb{R}^{d \times r_1}, U_2 \in \mathbb{R}^{r_1 \times r_2}, V \in \mathbb{R}^{r_2 \times n}$ as $X = f(U_1 f(U_2 f(V)))$,

[1] https://github.com/illidanlab/CAMF-MissingValueImputation



(a) Ground truth (b)MF AMF (d) CMF (e) CAMF

Fig. 2. Distribution of imputed samples. The proposed clustered adversarial matrix factorization outperforms baselines. Clustered matrix factorization is also decent, but the variance of the yellow cluster is smaller than expected.

Missing rate	0.9	0.8	0.7	0.6
Mean	0.415 ± 0.012	0.432 ± 0.022	0.444 ± 0.025	0.452 ± 0.028
SVD	0.457 ± 0.156	0.338 ± 0.159	0.215 ± 0.092	0.150 ± 0.039
KNN	0.716 ± 0.039	0.607 ± 0.065	0.540 ± 0.049	0.469 ± 0.044
MF	0.443 ± 0.162	0.325 ± 0.155	0.184 ± 0.108	0.099 ± 0.051
AMF	0.152 ± 0.023	0.117 ± 0.016	0.090 ± 0.013	0.075 ± 0.017
CMF	0.082 ± 0.007	0.072 ± 0.005	0.063 ± 0.004	0.050 ± 0.005
CAMF	$\mathbf{0.071 \pm 0.006}$	$\mathbf{0.065 \pm 0.004}$	$\mathbf{0.059 \pm 0.005}$	$\mathbf{0.046 \pm 0.005}$

TABLE I
IMPUTATION RMSE UNDER DIFFERENT MISSING RATE.

Sample # per cluster	500	1000	1500	2000
Mean	0.291 ± 0.025	0.287 ± 0.028	0.318 ± 0.015	0.313 ± 0.038
SVD	0.117 ± 0.013	0.117 ± 0.018	0.133 ± 0.012	0.132 ± 0.023
KNN	0.044 ± 0.010	0.044 ± 0.011	0.059 ± 0.003	0.052 ± 0.018
MF	0.045 ± 0.010	0.048 ± 0.012	0.061 ± 0.006	0.055 ± 0.016
AMF	0.044 ± 0.009	0.044 ± 0.009	0.054 ± 0.004	0.050 ± 0.011
CMF	0.035 ± 0.007	0.034 ± 0.006	0.040 ± 0.0020	0.036 ± 0.007
CAMF	0.034 ± 0.007	0.032 ± 0.005	0.035 ± 0.002	$\mathbf{0.031 \pm 0.006}$
DMF	0.060 ± 0.003	0.047 ± 0.009	0.055 ± 0.003	0.057 ± 0.016
DCMF	$\mathbf{0.030 \pm 0.006}$	0.032 ± 0.006	0.037 ± 0.002	0.033 ± 0.008
DCAMF	$\mathbf{0.030 \pm 0.006}$	$\mathbf{0.029 \pm 0.005}$	$\mathbf{0.034 \pm 0.002}$	$\mathbf{0.031 \pm 0.006}$

TABLE II
IMPUTATION RMSE UNDER DIFFERENT SAMPLE SIZES.

where we choose $f(x) = tanh(x)$ as an example and $r_1 = 20, r_2 = 40, d = 80$. The column of V are sampled as the same way as the Setting 1. All entries of U_1 and U_2 are randomly sampled from $0.1 \times \mathcal{N}(0, 1)$. The missing values are created the same way with the Setting 1.

Imputation Performance. Except for the methods we compared in the Setting 1, we add another three baselines: deep matrix factorization (DMF), deep clustered matrix factorization (DCMF) and deep clustered adversarial matrix factorization (DCAMF). The first one is to estimate the missing values by minimizing $\frac{1}{2}\|M \odot (X - f(U_1 f(U_2 f(V))))\|_2^2$ with the same constraints as matrix factorization. DCMF is the method to use deep matrix factorization with cluster constraint on V. DCAMF is to add distribution alignment to deep clustered matrix factorization to align the probability distribution of the data. The results are shown in TABLE II. From the table, we see a similar pattern as the results of the first experiment, i.e., cluster helps imputation and adding the distribution alignment improves the performance. We also see that when the sample size is 500 per cluster, adding adversarial network does not help a lot. That is because when the data's probability distribution is not simple, we need enough data samples to learn it. We also see when the sample size is 1000, CAMF's performance is much lower than DCAMF, which means with this sample size, provide exact latent factors information of X helps. However, when we increase the sample size to 1500 and 2000, we see CAMF works almost as well as DCMAF. From this, we conclude, if the sample size is large enough, it is sufficient to estimate the missing values by linear CAMF even if the data have complicated structure.

Method	KNN	SVD	MF
RMSE	0.2296 ± 0.02000	0.1042 ± 0.0129	0.0623 ± 0.0085
Method	ADF	CMF	CAMF
RMSE	0.0631 ± 0.0093	0.0378 ± 0.0047	0.0375 ± 0.0045

TABLE III
IMPUTATION RMSE OF DIFFERENT METHODS WHEN THE MISSING ENTRIES ARE RANDOMLY SAMPLED.

3) Setting 3: In the third setting, we show the results when the data is random missing. The data are synthesized the same way with that of the Setting 1 except that the missing entries are randomly sampled. The missing rate we use is 0.6. The results are shown in TABLE III. From the table, we see that when the missing values are randomly sampled, cluster constraint still helps because the data have clustering structure. However, distribution alignment does not improve the performance, since when the missing values are randomly sampled, minimizing the reconstruction error of the observed part is equivalent to minimizing the reconstruction error of the missing part. Therefore, we do not need to add the distribution alignment to learn the probability distribution.

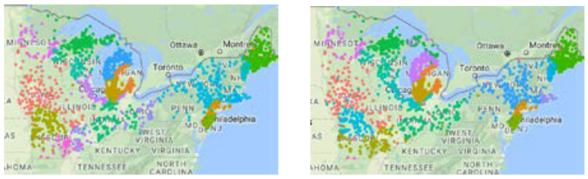
B. Case study: LAGOS dataset

In this sub-section, we present the results on the LAGOS dataset [17]. This dataset describes the features of lakes in north-east of United States. It has in total 2419 samples and 53 features including lakes local features like chemical measurements, longitude, latitude, and lakes regional features, i.e., climate, land use land cover. Since this dataset contains longitude and latitude of each lake, we use them to calculate the d_{ij} in the same way with the synthetic data experiments. The missing rate is set to be 0.7, and the missing values are created the same way as the first synthetic data experiment.

The results are shown in TABLE IV. We see that CAMF has the best performance. Cluster information and distribution information help a lot on the imputation performance. To compare CMF and CAMF, we also provide the cluster structure each method captures. To illustrate the difference, we applied hierarchical clustering with complete linkage on the V learned. We shows the results of 20 clusters Fig. 3. In the figures, each color represents a cluster. We see in the figures, most clusters are the same. However, for CMF, we see that the clusters in Ohio and Indiana are mixed. For CAMF,

Method	KNN	SVD	MF
RMSE	0.2296 ± 0.02000	0.1042 ± 0.0129	0.0623 ± 0.0085
Method	ADF	CMF	CAMF
RMSE	0.6169 ± 0.0114	0.5799 ± 0.0145	0.5552 ± 0.0147

TABLE IV
IMPUTATION RMSE OF DIFFERENT METHODS ON THE LAGOS DATASET.



(a) CMF (b) CAMF

Fig. 3. The cluster structures of V learned by CMF and CAMF. The number of clusters is set to be 20. Different colors represent different clusters.

Dataset	Income	School	G-Econ	YDNFT
Mean	1.083 ± 0.010	1.104 ± 0.004	1.121 ± 0.020	1.003 ± 0.002
SVD	0.988 ± 0.036	0.934 ± 0.008	0.812 ± 0.034	0.617 ± 0.0265
KNN	1.656 ± 0.108	1.356 ± 0.027	1.839 ± 0.140	1.098 ± 0.062
MF	0.762 ± 0.031	0.758 ± 0.007	0.689 ± 0.029	0.556 ± 0.016
AMF	0.697 ± 0.039	0.686 ± 0.009	0.619 ± 0.028	0.474 ± 0.028
CMF	0.738 ± 0.030	0.736 ± 0.008	0.553 ± 0.030	0.411 ± 0.011
CAMF	$\mathbf{0.649 \pm 0.029}$	$\mathbf{0.662 \pm 0.012}$	$\mathbf{0.509 \pm 0.027}$	$\mathbf{0.376 \pm 0.018}$

TABLE V
PERFORMANCE ON BENCHMARK DATASETS. CLUSTERED ADVERSARIAL MATRIX FACTORIZATION OUTPERFORMS ALL THE OTHER METHODS.

we see it groups Indian and Ohio into one cluster. Hence, regarding spatial continuity, CAMF is better than CMF.

C. Experiments on other real datasets

In this subsection, we compare the performance of different methods on 4 benchmark datasets: Income dataset[2], School dataset[3], G-Econ dataset[4] and YDNFT[5] dataset.

These spatial datasets all contain the location of each sample which are used to calculate d_{ij} in the same way as the synthetic data experiments. The missing values are created in the same way with synthetic data experiments, and the missing rates are set to be 0.7 for all the four datasets. The results are shown in TABLE V. Compared MF with CMF, and AMF with CAMF, we see clustering information can improve the performance on all the datasets. By adding the distribution alignment to MF or CMF, the imputation RMSE is also much lower than those without distribution alignment. The best performance is achieved by CAMF for all methods, which shows the effectiveness of the proposed method.

V. CONCLUSION

In this paper, we proposed clustered adversarial matrix factorization to deal with the structure missing problems in spatial datasets. We utilized the cluster structure of the spatial data and the probability distribution of the data to improve the imputation performance on the missing data. In our model, we only considered a very simple way of adding prior cluster knowledge. In some cases, it is possible that this prior knowledge is inaccurate and may bring negative affect on the imputation. More research is needed in the future regarding to this point.

[2]https://geodacenter.github.io/data-and-lab//co_income_diversity_variables/
[3]https://www.kaggle.com/lazyjustin/ncschools
[4]https://gecon.yale.edu/data-and-documentation-g-econ-project
[5]https://geodacenter.github.io/data-and-lab/lasrosas/

ACKNOWLEDGMENTS

This material is based in part upon work supported by the National Science Foundation under Grant EF-1638679 (JZ, PT), IIS-1714741 (JZ), IIS-1749940 (JZ) and Office of Naval Research N00014- 17-1-2265. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

REFERENCE

[1] Jian-Feng Cai, Emmanuel J Candès, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. *SIOPT*, 20(4):1956–1982, 2010.
[2] Emmanuel J Candès and Benjamin Recht. Exact matrix completion via convex optimization. *FoCM*, 9(6):717, 2009.
[3] Bianca NI Eskelson, Hailemariam Temesgen, Valerie Lemay, Tara M Barrett, Nicholas L Crookston, and Andrew T Hudak. The roles of nearest neighbor methods in imputing missing data in forest inventory and monitoring databases. *Scand J For Res*, 24(3):235–246, 2009.
[4] Pedro J García-Laencina, José-Luis Sancho-Gómez, Aníbal R Figueiras-Vidal, and Michel Verleysen. K nearest neighbours with mutual information for simultaneous classification and missing data imputation. *Neurocomputing*, 72(7):1483–1493, 2009.
[5] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*. Curran Associates, Inc., 2014.
[6] David Hallac, Jure Leskovec, and Stephen Boyd. Network lasso: Clustering and optimization in large graphs. In *SIGKDD*. ACM, 2015.
[7] Toby Dylan Hocking, Armand Joulin, Francis Bach, and Jean-Philippe Vert. Clusterpath: An algorithm for clustering using convex fusion penalties. In *ICML*. Omnipress, 2011.
[8] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and locally consistent image completion. *ACM TOG*, 36(4):107, 2017.
[9] Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi. Low-rank matrix completion using alternating minimization. In *STOC*. ACM, 2013.
[10] Rebecka Jörnsten, Hui-Yu Wang, William J Welsh, and Ming Ouyang. Dna microarray data imputation and significance analysis of differential expression. *Bioinformatics*, 21(22):4155–4161, 2005.
[11] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.
[12] John Molitor, Paul Marjoram, and Duncan Thomas. Fine-scale mapping of disease genes with multiple mutations via spatial clustering techniques. *Am. J. Hum. Genet*, 73(6):1368–1384, 2003.
[13] Danh V Nguyen, Naisyin Wang, and Raymond J Carroll. Evaluation of missing value estimation for microarray data. *JDS*, 2(4):347–370, 2004.
[14] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *CVPR*. IEEE, 2016.
[15] Beth K Potter, Kathy N Speechley, Iris A Gutmanis, M Karen Campbell, John J Koval, and Douglas Manuel. A comparison of measures of socioeconomics status for adolescents in a canadian national health survey. *ADRD*, 26(2-3):80, 2005.
[16] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR*, abs/1511.06434, 2015.
[17] Patricia A Soranno, Linda C Bacon, Michael Beauchene, Karen E Bednar, Edward G Bissell, Claire K Boudreau, Marvin G Boyer, Mary T Bremigan, Stephen R Carpenter, Jamie W Carr, et al. Lagos-ne: a multi-scaled geospatial and temporal database of lake ecological context and water quality for thousands of us lakes. *GigaScience*, 6(12):1–22, 2017.
[18] Gábor Takács, István Pilászy, Bottyán Németh, and Domonkos Tikk. Matrix factorization and neighbor based algorithms for the netflix prize problem. In *RecSys*. ACM, 2008.
[19] Wadood Javed Yahya. *Image reconstruction from a limited number of samples: a matrix-completion-based approach*. PhD thesis, McGill University Libraries, 2011.
[20] Hsiang-Fu Yu, Cho-Jui Hsieh, Si Si, and Inderjit Dhillon. Scalable coordinate descent approaches to parallel matrix factorization for recommender systems. In *ICDM*. IEEE, 2012.