Matrix Factorization with Landmarks for Spatial Data

Chenguang Fang
BNRist, Tsinghua University
fcg19@mails.tsinghua.edu.cn

Yinan Mei
BNRist, Tsinghua University
myn18@mails.tsinghua.edu.cn

Shaoxu Song
BNRist, Tsinghua University
sxsong@tsinghua.edu.cn

***Abstract*—Matrix factorization (MF) is widely adopted to learn from data, e.g., for data representation and recommendation as well as many database applications such as data imputation and repairing. While it works for numerical values in general, for spatial data, without considering the locality w.r.t. the spatial information, the learned features could vary in spatial distribution. Even if smoothness in terms of close neighbors could be considered in the objective function to leverage the spatial information, the learned features are still uncontrolled in locations, and thus do not help much in learning from the data that are geographically distant. Therefore, in this study, we propose to introduce landmarks to control the locations of learned features and make them geographically close to the data observations. The proposed SMFL, Spatial Matrix Factorization with Landmarks, benefits from landmarks in more accurate learned features, along with better interpretability, and reduced computation cost. Our major contributions include (1) introducing landmarks to guide the locations of learned features and enhance the performance as well as the interpretability of the MF model, (2) proposing the SMFL method that cooperates landmarks with NMF and spatial regularization, for better utilizing the spatial information, and (3) devising updating rules with landmarks and proving the convergence for the proposed method. Experiments on real-world datasets highlight the advance of our proposal in various applications.**

I. Introduction

Matrix Factorization (MF) shows a good performance in learning from numerical data for various applications [48], [27]. It decomposes the original data matrix $\mathbf{X}$ into the product of two low-rank matrices $(\mathbf{V}, \mathbf{U})$ and then recovers the original matrix from the product. One of the decomposed matrices $\mathbf{V}$ records the latent features learned. The other $\mathbf{U}$ serves as a coefficient matrix for reconstructing the original matrix by linearly combining the features in $\mathbf{V}$. (See details in Section II.)

A. Motivation

For spatial data, typical numerical values, related to objects that occupy space [22], the MF-based method naturally provides good performance. In particular, we may use the first 2 columns of the original matrix $\mathbf{X}$ to express the spatial information (such as latitude and longitude illustrated in Figure 1). Interestingly, following the principle of matrix multiplication, the first 2 columns in the feature matrix $\mathbf{V}$ could also be interpreted as the latent spatial features, i.e., latitude and longitude of the learned latent features.

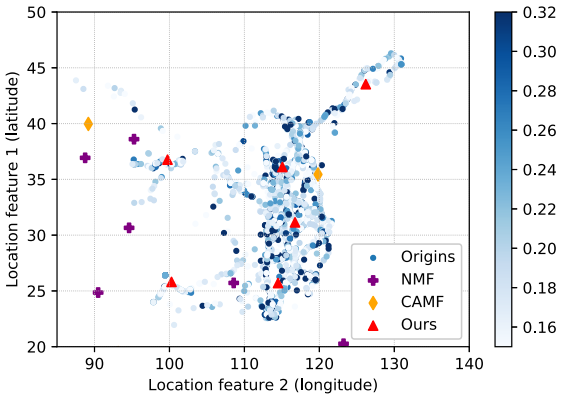

Fig. 1: Locations of observations and features learned over a real-world dataset. Points in blue colors denote the fuel consumption rates of observations. Purple and orange points are the spatial features learned by NMF [41] and CAMF [42], respectively, which are all distant from observations (some extremely distant points are not plotted for better visualization). We propose to employ landmarks in red points to control the locations of learned features and make them geographically close to data observations.

Example 1 (Motivation)**.** Figure 1 visualizes an example of the fuel consumption rate data collected by our partner company, a heavy machine maintenance service provider. One of its objectives is to reduce the fuel consumption rate of the maintained devices. The data are collected by vehicle sensors installed by the company. They could be incomplete due to broken sensors, hardware failures, unsynchronized collection frequencies, etc. The incompleteness may prevent downstream applications from performing, e.g., optimize the logistics route for more efficient fuel consumption rate. In general, the fuel consumption rates in the east region are higher (dark blue) than those in the west and northeast, revealing that the east region in lower altitudes with sufficient oxygen leads to a better fuel consumption rate. In this sense, as an important application of MF methods, the imputation of missing data could utilize such relationships on the spatial information.

The locations of learned features (i.e., the first two columns

TABLE I: An example of the input data

Latitude	Longitude	Speed	Torque	Fuel Consumption Rate
45.314585	130.939853	775.750	381.0	7.40
45.315147	130.939788	698.500	322.0	4.40
45.315058	130.939952	700.750	302.0	4.80
45.315058	130.939950	701.125	276.0	4.55
...	...	...	...	...

TABLE II: Notations

Symbol	Description
$\mathbf{X}, \mathbf{U}, \mathbf{V}$	observed matrix, decomposed matrices
$\mathbf{x}_i$, $\mathbf{u}_i$, $\mathbf{v}_i$	i-th row of $\mathbf{X}$, $\mathbf{U}$, $\mathbf{V}$
x_{ij}, u_{ij}, v_{ij}	j-th element of $\mathbf{x}_i$, $\mathbf{u}_i$, $\mathbf{v}_i$
Ω	the set of the observed entries in $\mathbf{X}$
Ψ	the set of the unobserved entries in $\mathbf{X}$
Φ	the set of the landmark entries in $\mathbf{V}$
$\mathcal{O}$	objective function
$\mathrm{Tr}(\cdot)$	the trace of a matrix
$NN_p(\mathbf{x}_i)$	p-nearest neighbors of $\mathbf{x}_i$

in $\mathbf{V}$ as aforesaid) by the existing methods NMF [41] in purple and CAMF [42] in orange are also illustrated in Figure 1. Unfortunately, without considering the locality in spatial data, the features learned by MF could vary in spatial distribution. As shown, they could be very distant from the (blue) data observations, which might not help much in imputing missing values. Even if smoothness in terms of close neighbors is considered in the objective function [40], [9], partially leveraging the spatial information, the learned features are still uncontrolled in locations and do not help much in imputing the missing data that are geographically distant. For instance, the purple point learned by NMF at the bottom (in the ocean) would have limited contribution in imputing the distant blue points at the top (on the continent).

Table I shows the example tabular data of Figure 1. The first two columns (latitude and Longitude) are the spatial information, i.e., $\mathbf{SI} = [\mathbf{SI}_1, \mathbf{SI}_2]$ and $L = 2$. The last three columns (Speed, Torque, and Fuel Consumption Rate) are the additional attributes in $\mathbf{X}$. In such a scenario, each tuple in $\mathbf{X}$ denotes the engine speed, torque and fuel consumption rate of the vehicle at the specific location. As shown in Figure 1, the fuel consumption rates are affected by the terrains in different locations, as well as engine speed and torque. ■

B. Intuition

In this study, we propose the **S**patial **M**atrix **F**actorization with **L**andmarks (SMFL) for spatial data. Intuitively, rather than arbitrarily distributed features, we introduce landmarks to control the locations of learned features. For instance, red points in Figure 1 are the landmarks in feature learning, which are forced to be close to data observations, thus contributing to more accurate learning.

The benefits of applying the landmarks are in three aspects.

(i) More accurate learned features and applications. Since the learned features on landmarks are much closer to data observations, analogous to the rationale of the nearest neighbor-based methods [18], it contributes to more accurate learned features, thus benefiting downstream applications, such as data imputation and data repair, as illustrated in Section IV-B.

(ii) Better interpretability. While the features distant from observations learned by the existing CAMF and NMF are difficult to explain, as presented in Figure 1, our learned features on landmarks are more interpretable. Indeed, as reported in Figure 5 in Section IV-C, it could also explain why some (carefully curated) landmarks show better imputation performance than others.

(iii) Reduced computation cost. It is worth noting that existing MF methods without fixing the locations of features will update all the columns of $\mathbf{V}$, including the columns of spatial information (e.g., latitude and longitude in Figure 1). In contrast, by fixing the landmarks, the columns of spatial information in $\mathbf{V}$ have already been given and have no need to be updated in each iteration, i.e., improved time cost as evaluated in Section IV-E.

There are many downstream applications of our proposal. (1) The application of optimizing the logistics route for more efficient fuel consumption rate is performed as follows. The data with spatial locations in the fuel consumption map can be utilized to simulate the accumulated fuel consumption of a given route, as shown Figure 1. Vehicles may select the logistics route with less fuel consumption, thus saving energy. Incomplete data obviously obstruct the fuel consumption simulation, while more accurate imputation helps in selecting the energy-efficient logistics route. (2) Another application is data clustering with missing values. The learned coefficient matrix $\mathbf{U}$ gives each tuple a weight of belonging to each cluster. Therefore, matrix factorization-based methods can first impute the missing values and then perform clustering [37]. In such an application, the spatial information helps not only in imputing missing values but also clustering.

C. Contribution

The major contributions of our work are summarized as follows.

(1) We introduce landmarks to guide the locations of learned features and enhance the performance as well as the interpretability of the MF model.

(2) We propose the SMFL method that cooperates landmarks with NMF and spatial regularization, for better utilizing the spatial information.

(3) We provide a updating strategy with landmarks for the proposed algorithm, and remarkably, prove the convergence of the strategy.

(4) Comprehensive experiments are conducted over different datasets and applications including data imputation and data repair. The results demonstrate that our proposed SMFL outperforms the state-of-the-art methods. Notably, we show that the proposed landmarks indeed contribute to higher accuracy, better interpretability and lower time cost.

The code and data of this paper are available at [1].

II. PRELIMINARIES

In this section, we first introduce some notations of our problem, the NMF algorithm and spatial regularization. We then define the **S**patial **M**atrix **F**actorization (SMF) problem (without landmarks) by combining the NMF algorithm and spatial regularization for spatial data. Table II lists some frequently used notations.

A. Notations

We will use bold upper-case letters to refer to matrices, bold lower-case letters to refer to vectors, and regular font lower-case letters to refer to entries in the matrix. For instance, $\mathbf{X}$ is a matrix, $\mathbf{x}_i$ is the i-th row of $\mathbf{X}$ and x_{ij} is the j-th element of $\mathbf{x}_i$.

Let $\mathbf{X} \in \mathbb{R}^{N\times M}$ of N examples and M features be the target matrix, which is only partially observed. Due to the applications such as imputation and repair may have unobserved entries, we use Ω to denote the set of the observed entries in $\mathbf{X}$, and use Ψ to denote the set of the unobserved entries in $\mathbf{X}$. A mapping function $\mathcal{R}_\Omega : \mathbb{R}^{N\times M} \to \mathbb{R}^{N\times M}$ is thus defined, which serves as a mask function to filter the observed entries.

$$[\mathcal{R}_\Omega(\mathbf{X})]_{ij} = \begin{cases} x_{ij}, & \text{if } (i,j) \in \Omega \\ 0, & \text{otherwise} \end{cases}$$

In this study, we focus on the spatial data, i.e., the data related to objects that occupy space [22]. Without loss of generality, we use the first L columns of $\mathbf{X}$ to denote the spatial information of the samples, i.e., $\mathbf{SI} = [\mathbf{SI}_1, \mathbf{SI}_2, \ldots, \mathbf{SI}_L] \in \mathbb{R}^{N\times L}$, where $\mathbf{SI}_i$ is the i-th attribute (column) of the spatial information. For example, in Figure 1, the spatial information is provided with latitude and longitude, having $L = 2$, i.e., the first two columns of $\mathbf{X}$ are latitude and longitude, respectively.

An example diagram is also illustrated in Figure 2, where grey entries are unobserved corresponding to Ω and white entries are observed corresponding to Ψ. The first two columns of $\mathbf{X}$ in pink are spatial information of the data. In addition, we use $NN_p(\mathbf{x}_i)$ to denote the p-nearest neighbors of $\mathbf{x}_i$ on spatial information.

B. NMF Algorithm

Nonnegative Matrix Factorization (NMF) algorithm has been shown effective in learning localized features [25], [30], i.e., different parts of features related to different locations [28]. For instance, in face analysis tasks, different parts of faces, including eyes and noses, can be decomposed as localized features. Given a data matrix $\mathbf{X} \in \mathbb{R}^{N\times M}$, each row of $\mathbf{X}$ is an observed sample. NMF learns a decomposition of $\mathbf{X}$ to two nonnegative matrices $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_n] \in \mathbb{R}^{N\times K}$ and $\mathbf{V} \in \mathbb{R}^{K\times M}$ with $K < \min(N, M)$. The product of $\mathbf{U}$ and $\mathbf{V}$ should approximate the original matrix $\mathbf{X}$. A standard NMF objective function can be defined as,

$$\mathcal{O}_{NMF}(\mathbf{U}, \mathbf{V}) = ||\mathbf{X} - \mathbf{U}\mathbf{V}||_F^2, \tag{1}$$

$$s.t. \quad u_{ij}, v_{ij} \geq 0 \tag{2}$$

where $||\cdot||_F$ is the Frobenius norm. The nonnegative constraint on the decomposed matrices $\mathbf{U}$ and $\mathbf{V}$ allows only additive linear combinations of the features [29]. The learned features are thus localized [25]. $\mathbf{V}$ is called a feature matrix, where the rows record the localized features. $\mathbf{U}$ is called a coefficient matrix, which provides a linear combination of features to reconstruct the original matrix. Following a similar idea, since features may be related to spatial locations, spatial data can be potentially decomposed into localized features (e.g., features of different clusters) for imputation.

C. Spatial Regularization

In addition to NMF, which tries to capture localized features, we also introduce spatial regularization into the objective function. In spatial models, spatial regularization penalizes the distance between neighboring locations [23] and has been widely adopted in spatial and geometric models [40], [24]. In a word, the spatial regularization limits the distance of the neighbors and tries to maintain spatial smoothness in the objective function.

One of the commonly used spatial regularizers is the graph Laplacian regularizer [43], [9]. Let $\mathbf{D}$ be the similarity matrix for $\mathbf{X}$. Given p as the number of nearest neighbors, $\mathbf{D}$ is denoted as

$$d_{ij} = \begin{cases} 1, & \mathbf{x}_i \in NN_p(\mathbf{x}_j) \text{ or } \mathbf{x}_j \in NN_p(\mathbf{x}_i) \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

i.e., d_{ij} describes the spatial similarity of $\mathbf{x}_i$ and $\mathbf{x}_j$, where $NN_p(\mathbf{x}_j)$ denotes the p-nearest neighbors on spatial information $\mathbf{SI}$ of $\mathbf{x}_j$.

If some attribute of $\mathbf{SI}$ in a sample is unfortunately missing, the complete elements in the j-th column will be used for initializing x_{ij}. For example, if $x_{ij} \in \Omega$ ($j \in \{0, 1, \ldots, L\}$), we may initialize it with $x_{ij} = average(\{x_{tj} | (t, j) \notin \Omega\})$. This strategy is adopted only for computing the similarity matrix $\mathbf{D}$, while the precise imputation of x_{ij} will be computed later in our proposal (Section III-A).

The spatial regularization is thus defined as follows:

$$\begin{aligned} \mathcal{O}_{SR}(\mathbf{U}) &= \frac{1}{2}\sum_{i,j=1}^{N} d_{ij}||\mathbf{u}_i - \mathbf{u}_j||_2^2 \\ &= \sum_{i}^{N} w_{ii}\mathbf{u}_i^T\mathbf{u}_i - \sum_{i,j=1}^{N} d_{ij}\mathbf{u}_i^T\mathbf{u}_j \\ &= \mathrm{Tr}(\mathbf{U}^T\mathbf{L}\mathbf{U}) \end{aligned}$$

where $\mathrm{Tr}(\cdot)$ denotes the trace of a matrix, $\mathbf{L}$ is called a graph Laplacian matrix with $\mathbf{L} = \mathbf{W} - \mathbf{D}$, where $\mathbf{W}$ is a diagonal matrix, denoted as

$$w_{ij} = \begin{cases} \sum_t d_{it}, & i = j \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

which is called the symmetric similarity matrix.

Intuitively, this regularizer enables the close tuple pairs in $\mathbf{U}$ to have similar features, and thus the spatial smoothness is maintained. We will explain how to combine the NMF objective function and spatial regularization when the data are

incomplete, not considered in the existing works. Moreover, we will introduce landmarks into our proposed algorithm to further utilize the spatial information in Section III-A.

D. Problem Definition

In this paper, we employ matrix factorization for spatial data, to learn accurate features and better reconstruction of the matrix. As introduced in Section II, we will adopt NMF and spatial regularization to learn from the spatial data. To impute the entries $\mathcal{R}_\Psi(\mathbf{X})$, we deploy the mask function $\mathcal{R}$ to mask the unobserved entries by $\mathcal{R}_\Omega$ in the objective function.

$$\mathcal{O}'_{NMF}(\mathbf{U},\mathbf{V}) = ||\mathcal{R}_\Omega(\mathbf{X}-\mathbf{UV})||_F^2 \tag{5}$$

With the mask function, gradients for those unobserved entries are not computed. Combining the NMF with spatial regularization, we then propose the objective of our proposed method:

$$\begin{aligned}\mathcal{O}(\mathbf{U},\mathbf{V}) &= \mathcal{O}'_{NMF}(\mathbf{U},\mathbf{V}) + \lambda\mathcal{O}_{SR}(\mathbf{U}) \\ &= ||\mathcal{R}_\Omega(\mathbf{X}-\mathbf{UV})||_F^2 + \lambda\,\mathrm{Tr}(\mathbf{U}^T\mathbf{L}\mathbf{U})\end{aligned}$$

Hence, we formalize the optimization problem of our **S**patial **M**atrix **F**actorization (SMF) as below,

Problem 1. *Given an input matrix* $\mathbf{X}$*, the set of the observed entries* Ω*, graph Laplacian matrix* $\mathbf{L}$*, the* ***S****patial* ***M****atrix* ***F****actorization (SMF) tries to minimize the following objective function:*

$$\min_{\mathbf{U},\mathbf{V}} \mathcal{O}(\mathbf{U},\mathbf{V}) = \min_{\mathbf{U},\mathbf{V}} (||\mathcal{R}_\Omega(\mathbf{X}-\mathbf{UV})||_F^2 + \lambda\,\mathrm{Tr}(\mathbf{U}^T\mathbf{L}\mathbf{U})) \tag{6}$$

$$s.t. \qquad u_{ij}, v_{ij} \geq 0 \tag{7}$$

Let $\mathbf{U}^*$ and $\mathbf{V}^*$ be the solution to the optimization problem, $\mathbf{X}^* = \mathbf{U}^*\mathbf{V}^*$. We can thus recover unobserved entries in $\mathbf{X}$ with $\mathbf{X}^*$, i.e.,

$$\hat{\mathbf{X}} \leftarrow \mathcal{R}_\Omega(\mathbf{X}) + \mathcal{R}_\Psi(\mathbf{X}^*) \tag{8}$$

where Ψ denotes the set of the unobserved entries in $\mathbf{X}$. For repair task, following the same line, we can let Ψ denote the set of the dirty entries provided by error detection techniques (e.g., Raha [33]. The dirty entries are then replaced with learned values in $\mathbf{X}^*$ according to Formula 8.

III. Matrix Factorization with Landmarks

In this section, we will first introduce landmarks to enhance the matrix factorization methods for spatial data and propose the Spatial Matrix Factorization with Landmarks (SMFL) in Section III-A. Then, we devise updating rules for iteratively solving the problem in Section III-B. Finally, the convergence of the updating rules will be proved in Section III-C.

A. Landmarks for Spatial Matrix Factorization

In the formalization of SMF in Problem 1, we propose to decompose the input matrix $\mathbf{X}$ into two matrices $\mathbf{U}$ and $\mathbf{V}$. According to the motivation mentioned above, the NMF objective function $\mathcal{O}'_{NMF}$ contributes to the localized features in $\mathbf{V}$ and spatial regularization $\mathcal{O}_{SR}$ restricts the spatial smoothness. Moreover, owing to the principle of matrix multiplication, the first L columns of $\mathbf{X}$ are only related to the first L columns of $\mathbf{V}$, according to the following formula:

$$x_{ij} = \sum_{k=1}^{K} u_{ik}v_{kj}, 1 \leq j \leq L$$

This is to say, the spatial information recorded in $\mathbf{X}$ is only reconstructed by the linear combinations of the first L columns of $\mathbf{V}$. In original matrix factorization problems, the features matrix $\mathbf{V}$ is first randomly initialized and completely learned by the updating rules. Considering the spatial information of the data, we propose to utilize the spatial information from the original matrix $\mathbf{X}$ to generate some landmarks and inject these landmarks into the first L columns of $\mathbf{V}$. During the updating of $\mathbf{U}$ and $\mathbf{V}$ for solving the objective function, we set the gradients of those landmarks in $\mathbf{V}$ to 0. Therefore, the landmarks in $\mathbf{V}$ do not update in iteration. This is why we call it "landmark".

Definition 1. *We denote* Φ *the set of the landmark entries in* $\mathbf{V}$*, i.e.,* $\Phi = \{(i,j)|1 \leq i \leq K, 1 \leq j \leq L\}$.

The next question is how to decide their values. We novelly propose to set landmarks as the centers of the clusters of spatial information $\mathbf{SI}$. We use the K-means algorithm to provide clustering. This is achieved by setting the number of cluster K' in K-means equal to K of the NMF problem. Thus the landmarks can serve as the bridge to connect the NMF algorithm and K-means, as illustrated in Figure 2. Moreover, the features learned in $\mathbf{V}$ can represent the features of each cluster, respectively, due to the characteristics of the NMF algorithm to learn the localized features.

Specifically, let $\mathbf{C} = [\mathbf{C}_1, \mathbf{C}_2, \ldots, \mathbf{C}_K] \in \mathbb{R}^{K\times L}$ denote the centers of the K clusters on $\mathbf{SI}$ computed by K-means. We use $\mathbf{C}$ as landmark values and inject them into the first L columns of $\mathbf{V}$, i.e.,

$$v_{ij} = c_{ij}, (i,j) \in \Phi \tag{9}$$

The gradients of these landmarks in $\mathbf{V}$ will be set to 0. Thus this part of $\mathbf{V}$ will not update during the iterations. As stated in Section I, the benefits of them for spatial matrix factorization are in three aspects (accuracy, interpretability and efficiency).

Finally, by introducing landmarks into our problem, we define the optimization problem of our **S**patial **M**atrix **F**actorization with **L**andmarks (SMFL) as below.

Problem 2. *Given an input matrix* $\mathbf{X}$*, the set of clean entries* Ω*, graph Laplacian matrix* $\mathbf{L}$*, the set of landmark entries* Φ*, and the landmark matrix* $\mathbf{C}$ *computed by K-means on* $\mathbf{SI}$*, the* ***S****patial* ***M****atrix* ***F****actorization with* ***L****andmarks (SMFL) minimizes the following objective function:*

$$\min_{\mathbf{U},\mathbf{V}} \mathcal{O}(\mathbf{U},\mathbf{V}) = \min_{\mathbf{U},\mathbf{V}} (||\mathcal{R}_\Omega(\mathbf{X}-\mathbf{UV})||_F^2 + \lambda\,\mathrm{Tr}(\mathbf{U}^T\mathbf{L}\mathbf{U})) \tag{10}$$

$$s.t. \qquad v_{ij} = c_{ij}, (i,j) \in \Phi \tag{11}$$

$$u_{ij}, v_{ij} \geq 0 \tag{12}$$

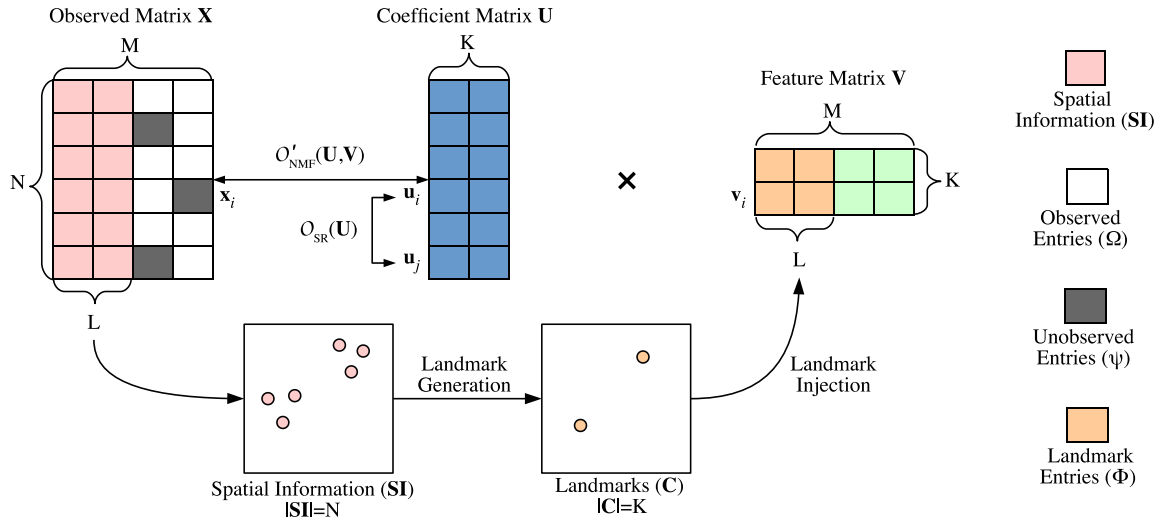

Fig. 2: Overall structure of SMFL. $\mathcal{O}'_{NMF}(\mathbf{U},\mathbf{V})$ is the reconstruction error when approximating $\mathbf{X}$ by the product of $\mathbf{U}$ and $\mathbf{V}$. $\mathcal{O}_{SR}(\mathbf{U})$ denotes the spatial smoothness highlighted in the rows learned in the Coefficient Matrix $\mathbf{U}$, corresponding to the rows in $\mathbf{X}$. The spatial information **SI** with N rows and L columns of the original $\mathbf{X}$ will be utilized to generate landmarks. The landmarks $\mathbf{C}$ will then be injected into the first L columns of the Feature Matrix $\mathbf{V}$, which will not change during the learning iterations and thus guide the training of the (green) features in $\mathbf{V}$.

B. Updating Rules

We now discuss the solution to the proposed SMFL problem. The NP-hardness of the matrix completion with missing data has been proved in the existing study [19]. Therefore, in this section, we focus on iteratively obtaining a local minimum for the SMFL problem. We introduce a multiplicative updating method with the consideration of landmarks.

1) Gradient Descent Method: To optimize, we exploit gradient descent which is a classical optimization technique.

$$\begin{aligned} u_{ik} &= u_{ik} - \theta_{ik}\frac{\partial \mathcal{O}(\mathbf{U},\mathbf{V})}{\partial U_{ik}} \\ &= u_{ik} + 2\theta_{ik}(\mathcal{R}_\Omega(\mathbf{X})\mathbf{V}^T - \mathcal{R}_\Omega(\mathbf{U}\mathbf{V})\mathbf{V}^T - \lambda\mathbf{L}\mathbf{U})_{ik} \end{aligned}$$

Similarly, we have

$$v_{kj} = \begin{cases} v_{kj} - \delta_{kj}(-2(\mathbf{U}^T\mathcal{R}_\Omega(\mathbf{X}))_{kj} \\ \qquad + 2(\mathbf{U}^T\mathcal{R}_\Omega(\mathbf{U}\mathbf{V}))_{kj}), & (k,j) \notin \Phi \\ c_{kj}, & (k,j) \in \Phi \end{cases}$$

where θ_{ik} and δ_{kj} are learning rates for updating u_{ik} and v_{kj} respectively. Gradient descent updates $\mathbf{U}$ and $\mathbf{V}$ iteratively until convergence to obtain a local minimum for the problem.

2) Multiplicative Updating Method: In the gradient descent method, there are about $(N+M)*K$ learning rates $(\theta_{ik}, \delta_{ki})$ to set manually. Too large parameters can increase the loss, while too small parameters may defer the convergence [7]. Even if learning rate decay is applied to adjust the learning rate dynamically, it still requires additional settings of hyper-parameters.

To deal with the challenge, we present our multiplicative updating method with landmarks, inspired by the updating rules in NMF [25], [26], [9]. It is exactly a self-adaptive gradient descent method, addressing the challenge of tuning parameters for gradient descent methods. We will also analyze the convergence of the proposed method in Section III-C.

According to the given objective function $\mathcal{O}$ subject to the equality constraints $u_{ij}, v_{ij} \geq 0, v_{kj} = c_{kj}, (k,j) \in \Phi$, by adopting the method of Lagrange multipliers, we can obtain the Lagrangian function from the original objective function and constraints, i.e.,

$$\begin{aligned} \mathcal{L}(\mathbf{U},\mathbf{V}) =& ||\mathcal{R}_\Omega(\mathbf{X} - \mathbf{U}\mathbf{V})||_F^2 + \lambda \operatorname{Tr}(\mathbf{U}^T\mathbf{L}\mathbf{U}) - \operatorname{Tr}(\mu\mathbf{U}^T) \\ & - \operatorname{Tr}(\rho\mathbf{V}^T) - \operatorname{Tr}(\eta(\mathbf{V}-\mathbf{C})^T) \\ =& \operatorname{Tr}(\mathbf{X}\mathbf{X}^T) - 2\operatorname{Tr}(\mathbf{X}\mathbf{V}^T\mathbf{U}^T) + \operatorname{Tr}(\mathbf{U}\mathbf{V}\mathbf{V}^T\mathbf{U}^T) \\ & + \lambda \operatorname{Tr}(\mathbf{U}^T\mathbf{L}\mathbf{U}) - \operatorname{Tr}(\mu\mathbf{U}^T) \\ & - \operatorname{Tr}(\rho\mathbf{V}^T) - \operatorname{Tr}(\eta(\mathbf{V}-\mathbf{C})^T) \end{aligned}$$

where $\mu_{ik}, \rho_{kj}, \eta_{kj}$ are lagrangian multipliers, and $\eta_{kj} = 0, (k,j) \in \Phi$.

By adopting Karush-Kuhn-Tucker (KKT) conditions and rearranging the formulas, we finally obtain the following updating rules for $\mathbf{U}$ and $\mathbf{V}$.

$$u_{ik} \leftarrow u_{ik}\frac{(\mathcal{R}_\Omega(\mathbf{X})\mathbf{V}^T)_{ik} + \lambda(\mathbf{D}\mathbf{U})_{ik}}{(\mathcal{R}_\Omega(\mathbf{U}\mathbf{V})\mathbf{V}^T)_{ik} + \lambda(\mathbf{W}\mathbf{U})_{ik}} \tag{13}$$

$$v_{kj} \leftarrow \begin{cases} v_{kj}\dfrac{(\mathbf{U}^T\mathcal{R}_\Omega(\mathbf{X}))_{kj}}{(\mathbf{U}^T\mathcal{R}_\Omega(\mathbf{U}\mathbf{V}))_{kj}}, & (k,j) \notin \Phi \\ c_{kj}, & (k,j) \in \Phi \end{cases} \tag{14}$$

An example of applying the multiplicative updating rules is illustrated in Figure 3.

The whole pipeline of SMFL is outlined in Algorithm 1.

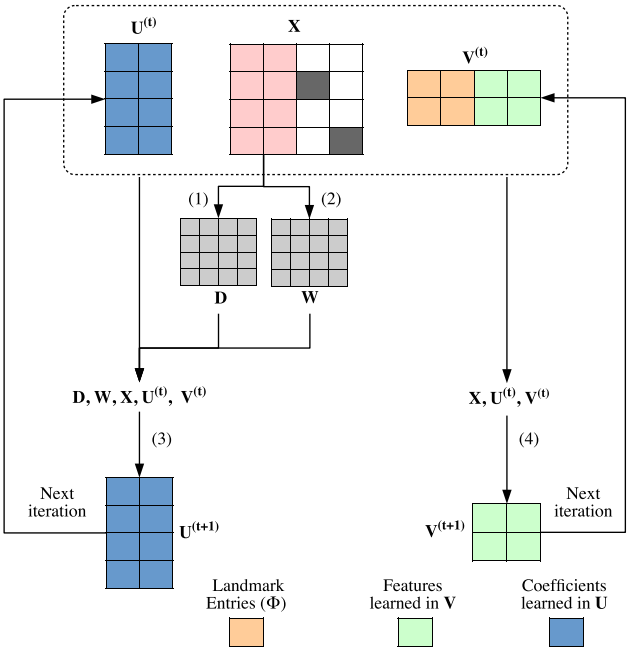

Fig. 3: Procedure of multiplicative updating in an iteration. In preprocessing, (1) compute $\mathbf{D}$ by Formula 3 and (2) compute $\mathbf{W}$ by Formula 4. In each iteration, (3) update $\mathbf{U}$ by Formula 13 and (4) update $\mathbf{V}$ by Formula 14.

Proposition 1. *The SMFL algorithm runs in $O(t_1NMK + N^2L + t_2KNL)$ time, where t_1 is the iteration number of updating rules, and t_2 is the iteration number of K-means.*

Proof sketch. We divide the time cost into three parts:

(1) The iteration of the updating rules is deduced in Section III-B (Formulas 13 and 14). The most time-cost operation is the matrix multiplication of $\mathbf{UV}$ and $\mathbf{U}^T\mathbf{X}$, both of which are in $O(NMK)$ time. That is, the updating rules run in $O(t_1NMK)$ time, where t_1 is the iteration number.

(2) Following Section II-C, SMFL also takes $O(N^2L)$ time to construct the similarity matrix $\mathbf{D}$ for $\mathbf{X}$.

(3) The K-means algorithm takes $O(t_2KNL)$ time to perform, where t_2 is the iteration number.

To conclude, the time complexity of the SMFL algorithm is $O(t_1NMK + N^2L + t_2KNL)$. ■

Proposition 1 shows that SMFL runs in $O(t_1NMK + N^2L + t_2KNL)$ time. The matrix updating runs t_1 iterations. By default, t_1 is set to 500. The updating algorithm will stop early if it already converges. Likewise, the K-means algorithm runs t_2 iterations. By default, t_2 is set to 300. Again, the clustering algorithm will stop early if it already converges. We also observe that L is generally small (i.e., with 2 spatial information columns), while the matrix updating rules indeed include multiple times of matrix multiplication in $O(NMK)$ time. That is, in practice, the matrix updating part dominates the time cost, and the K-means algorithm is not the bottleneck of the whole algorithm. It also leads to the result that SMFL runs faster than SMF. Section IV-E reports the experimental results in detail.

Algorithm 1: SMFL $(\mathbf{X}, \Omega, \Psi, L)$

Input: $N \times M$ matrix $\mathbf{X}$, observed entries Ω, unobserved entries Ψ, number of spatial information attributes (columns) L

Output: imputed matrix $\hat{\mathbf{X}}$

initialize $\mathbf{U} \in \mathbb{R}^{N\times K}$, $\mathbf{V} \in \mathbb{R}^{K\times M}$;

$\mathbf{D} \leftarrow$ computed by $\mathbf{X}$ according to Formula 3;

$\mathbf{W} \leftarrow$ computed by $\mathbf{D}$ according to Formula 4;

$\mathbf{SI} \leftarrow$ the first L columns of $\mathbf{X}$;

$\mathbf{C} \leftarrow$ centers of clusters computed by K-means($\mathbf{SI}$) according to Section III-A;

Inject $\mathbf{C}$ into $\mathbf{V}$ according to Formula 9;

while *not converged* **do**

  $u_{ik} \leftarrow u_{ik} \frac{(\mathcal{R}_\Omega(\mathbf{X})\mathbf{V}^T)_{ik}+\lambda(\mathbf{DU})_{ik}}{(\mathcal{R}_\Omega(\mathbf{UV})\mathbf{V}^T)_{ik}+\lambda(\mathbf{WU})_{ik}}$ (Formula 13) ;

  $v_{kj} \leftarrow v_{kj} \frac{(\mathbf{U}^T\mathcal{R}_\Omega(\mathbf{X}))_{kj}}{(\mathbf{U}^T\mathcal{R}_\Omega(\mathbf{UV}))_{kj}}, (k,j) \notin \Phi$ (Formula 14) ;

$\mathbf{U}^*, \mathbf{V}^* \leftarrow \mathbf{U}, \mathbf{V}$;

$\mathbf{X}^* \leftarrow \mathbf{U}^*\mathbf{V}^*$;

$\hat{\mathbf{X}} \leftarrow \mathcal{R}_\Omega(\mathbf{X}) + \mathcal{R}_\Psi(\mathbf{X}^*)$;

return $\hat{\mathbf{X}}$;

C. Convergence Analysis

In this section, we will show the convergence of the proposed multiplicative updating method. Following the similar pipeline proposed in [26], we will first design auxiliary functions for the proposed method. Next, by utilizing the auxiliary functions, we can prove that the objective function is non-increasing under the given updating rules of $\mathbf{U}$ and $\mathbf{V}$, in Propositions 5 and 7, respectively.

Definition 2. *Let A, G be functions and h, h' be variables, $A(h, h')$ is an auxiliary function for $G(h)$ if the conditions*

$$A(h, h') \geq G(h), \quad A(h, h) = G(h)$$

are satisfied.

This auxiliary function would then lead to the below lemma.

Lemma 2. *If A is an auxiliary function for G, then G is non-increasing under the updating rule*

$$h^{(t+1)} = \arg\min_h A(h, h^{(t)})$$

Proof sketch. The updating rule indicates that $A(h^{(t+1)}, h^{(t)}) \leq A(h^{(t)}, h^{(t)})$. Combining this inference with Lemma 2, we will have $G(h^{(t+1)}) \leq A(h^{(t+1)}, h^{(t)}) \leq A(h^{(t)}, h^{(t)}) = G(h^{(t)})$ ■

We can also conclude from the proof that, when $h^{(t+1)} = h^{(t)}$, $G(h^{(t+1)}) = G(h^{(t)})$. It indicates the convergence of the updating rule and the function reaches the local minimum.

Next, by utilizing Lemma 2, we will design auxiliary functions for the updating rule of $\mathbf{U}$ and $\mathbf{V}$ respectively and

then show the satisfactory of Lemma 2 with the help of the auxiliary functions.

1) Updating Rule of $\boldsymbol{U}$*:* We first show the convergence of the updating rule of $\mathbf{U}$. According to Formula 6, let

$$G(\mathbf{U}) = ||\mathcal{R}_\Omega(\mathbf{X} - \mathbf{UV})||_F^2 + \lambda \operatorname{Tr}(\mathbf{U}^T\mathbf{LU}) \tag{15}$$

and by a little abuse of the notation, let $G(u_{ik})$ denote the part in $G(\mathbf{U})$ that is just related to u_{ik}. Then according to Formula 15, we define the derivative and the second derivative of $G(u_{ik})$ as below,

$$G'(u_{ik}) = -2(\mathcal{R}_\Omega(\mathbf{X})\mathbf{V}^T)_{ik} + 2(\mathcal{R}_\Omega(\mathbf{UV})\mathbf{V}^T)_{ik} + 2\lambda(\mathbf{LU})_{ik}$$
$$G''(u_{ik}) = 2\sum_{(i,j)\in\Omega} v_{kj}v_{jk} + 2\lambda l_{ii} \tag{16}$$

Lemma 3. *Function*

$$A(u_{ik}, u_{ik}^{(t)}) = G(u_{ik}^{(t)}) + G'(u_{ik}^{(t)})(u_{ik} - u_{ik}^{(t)}) + \frac{(\mathcal{R}_\Omega(\mathbf{UV})\mathbf{V}^T)_{ik} + \lambda(\mathbf{WU})_{ik}}{u_{ik}^{(t)}}(u_{ik} - u_{ik}^{(t)})^2$$

is an auxiliary function for $G(u_{ik})$*.*

Proof sketch. (1) By setting $u_{ik}^{(t)} = u_{ik}$, $A(u_{ik}, u_{ik}) = G(u_{ik})$ is naturally satisfied; (2) To prove $A(u, u') \geq G(u)$, we give the Taylor series expansion of $G(u_{ik})$. Combining it with Formula 16, we find $A(u, u') \geq G(u)$ is equivalent to

$$\frac{(\mathcal{R}_\Omega(\mathbf{UV})\mathbf{V}^T)_{ik} + \lambda(\mathbf{WU})_{ik}}{u_{ik}^{(t)}} \geq \sum_{(i,j)\in\Omega} v_{kj}v_{jk} + \lambda l_{ii} \tag{17}$$

By rearranging the formulas, we can prove that Formula 17 holds, thus proving Lemma 3. (See full proof in [1].) ■

Now we have proved that $A(u_{ik}, u_{ik}^{(t)})$ is an auxiliary function for $G(u_{ik})$. The last step is to show that the updating rule follows the form of Lemma 2.

Lemma 4. *The updating rule of* u_{ik} *and the auxiliary function* $A(u_{ik}, u_{ik}^{(t)})$ *follows the form of* $u_{ik}^{(t+1)} = \arg\min_{u_{ik}} A(u_{ik}, u_{ik}^{(t)})$ *in Lemma 2.*

Proof sketch. Let

$$\frac{\partial A(u_{ik}^*, u_{ik}^{(t)})}{\partial u_{ik}^*} = 0$$

We then obtain

$$u_{ik}^* = u_{ik}^{(t)} \frac{(\mathcal{R}_\Omega(\mathbf{X})\mathbf{V}^T)_{ik} + \lambda(\mathbf{DU})_{ik}}{(\mathcal{R}_\Omega(\mathbf{UV})\mathbf{V}^T)_{ik} + \lambda(\mathbf{WU})_{ik}}$$

Since A is convex, u_{ik}^* is the global minimum. It follows

$$u_{ik}^{(t+1)} = u_{ik}^* = u_{ik}^{(t)} \frac{(\mathcal{R}_\Omega(\mathbf{X})\mathbf{V}^T)_{ik} + \lambda(\mathbf{DU})_{ik}}{(\mathcal{R}_\Omega(\mathbf{UV})\mathbf{V}^T)_{ik} + \lambda(\mathbf{WU})_{ik}}$$

follows the form $u_{ik}^{(t+1)} = \arg\min_{u_{ik}} A(u_{ik}, u_{ik}^{(t)})$ in Lemma 2. ■

Finally, we conclude the convergence of the updating rule of $\mathbf{U}$.

Proposition 5. *The objective function* $\mathcal{O}$ *of SMFL in Formula 10 is non-increasing under the updating rule of* $\boldsymbol{U}$ *in Formula 13.*

Proof sketch. (1) The updating rule is element-wise for each u_{ik}; (2) By finding an auxiliary function for $G(u_{ik})$, we prove that $G(u_{ik})$, the part of objective function $\mathcal{O}$ that is related to u_{ik}, is non-increasing under the updating rule of $\mathbf{U}$, with the combination of Lemma 2 and Lemma 3.

Therefore, we can conclude that the objective function $\mathcal{O}$ of SMFL in Formula 10 is non-increasing under the updating rule of $\mathbf{U}$ in Formula 13. ■

2) Updating rule of $\boldsymbol{V}$*:* For $\mathbf{V}$, we need to treat two parts of v_{kj} respectively: $(k, j) \in \Phi$ and $(k, j) \notin \Phi$.

When $(k, j) \in \Phi$ (the landmark entry), the updating rule is

$$v_{kj} \leftarrow c_{kj}, (k, j) \in \Phi$$

Since c_{kj} is fixed, we do not update v_{kj} when $(k, j) \in \Phi$, i.e., the objective function $\mathcal{O}$ is non-increasing under the updating rule of $(k, j) \in \Phi$.

When $(k, j) \notin \Phi$, the updating rule is

$$v_{kj} \leftarrow v_{kj} \frac{(\mathbf{U}^T\mathcal{R}_\Omega(\mathbf{X}))_{kj}}{(\mathbf{U}^T\mathcal{R}_\Omega(\mathbf{UV}))_{kj}}, (k, j) \notin \Phi$$

The convergence of the updating rule can be proved similarly by designing an auxiliary function like Section III-C1. For simplicity, we will reuse G and A to denote the original and auxiliary functions corresponding to the proof of $\mathbf{V}$. Let

$$G(\mathbf{V}) = ||\mathcal{R}_\Omega(\mathbf{X} - \mathbf{UV})||_F^2 + \lambda \operatorname{Tr}(\mathbf{U}^T\mathbf{LU})$$

and let $G(v_{kj})$ denote the part in $G(\mathbf{V})$ that is just related to v_{kj}. Then another auxiliary function for v_{kj} is designed.

Lemma 6. *Function*

$$A(v_{kj}, v_{kj}^{(t)}) = G(v_{kj}^{(t)}) + G'(v_{kj}^{(t)})(v_{kj} - v_{kj}^{(t)}) + \frac{(\mathbf{U}^T\mathcal{R}_\Omega(\mathbf{UV}))_{kj}}{v_{kj}^{(t)}}(v_{kj} - v_{kj}^{(t)})^2$$

is an auxiliary function for $G(v_{kj})$*.*

Proof sketch. The lemma can be proved similarly by replacing the roles of u_{ik} in Lemma 3. ■

With similar steps in Lemma 4, we can finally conclude the convergence of the updating rule of $\mathbf{V}$.

Proposition 7. *The objective function* $\mathcal{O}$ *of SMFL in Formula 10 is non-increasing under the updating rule of* $\boldsymbol{V}$ *in Formula 14.*

Hence, we analyze the convergence of the updating rules of $\mathbf{U}$ and $\mathbf{V}$, in the presence of landmarks.

IV. Experiments

In this section, we validate the performance of our proposed SMFL over several read-world spatial datasets and applications, including data imputation and data repair. The settings of the experiments are described in Section IV-A. In the first

set of experiments, we will qualitatively evaluate the accuracy of SMFL over several datasets, comparing with state-of-the-art imputation and repair techniques. Next, we conduct ablation studies to test the effectiveness of our proposed landmarks. We also analyze the parameter sensitivity of our proposal. Finally, the efficiency of the proposal is evaluated. In each experiment, we conduct it five times and take the average of the results.

TABLE III: Dataset summary

Dataset	Tuples	Columns	Examples of additional Columns
Economic [2]	27k	13	Precipitation, Temperature, ...
Farm [3]	0.4k	13	Nitrogen Fertilizer Application, ...
Lake [39]	8k	7	Lake Area, Elevation, ...
Vehicle	100k	7	Speed, Fuel Consumption Rate, ...

A. Experimental Settings

The experiments run on a machine with 2.1GHz CPU and Nvidia 1080ti GPU. The code and data are available at [1].

1) Datasets and Pre-Processing: Table III summarizes the six real datasets used in the experiments.

Economic dataset [2] includes economic data such as climates and population to analyze economic activities. Close locations may have similar Precipitation, Temperature, and so on, resulting in similar economic activities. Farm dataset [3] dataset records data such as Nitrogen Fertilizer Application to study location-specific nitrogen management for corn production. Fertilize application is various in different locations, due to climates and other location features. Lake dataset [39] includes lake information (e.g., Lake Area and Elevation) to study lake ecology in different places. Vehicle dataset is collected by our industrial partner with Speed, Torque, and Fuel Consumption Rate data to analyze vehicle fuel consumption. The features such as Fuel Consumption Rate are affected by the terrains in different locations, as the example illustrated in Figure 1.

There are some existing data quality issues in the original datasets. For example, Economic [2] is naturally incomplete without ground truth. We thereby select complete rows of the dataset. Farm [3] contains a large number of zero values without any description. Therefore, for each dataset, we first remove the incomplete tuples, and then choose the complete tuples without unclear zero values to serve as the ground truth for conducting experiments.

We devise different error injections for imputation and repair tasks respectively. For the imputation task, following [38], errors are injected artificially by randomly removing values from several columns, controlled by missing rate. For the repair task, we inject errors into all columns by randomly replacing the original values with other values in the same domain, controlled by the error rate. Unless otherwise stated, both the missing rate and error rate are set to 10%.

Note that, when the missing rate increases, some datasets with a large number of columns (such as Economic and Farm) may fail to preserve any complete rows. Due to the reasons that some baselines require complete rows to perform, we first randomly extract 100 complete tuples from the dataset for a fair comparison. Error injection is performed on the remaining data. Finally, we will conduct min-max normalization on all datasets and transform them into the range [0, 1] to balance the influences of the different scales of different columns.

2) Criteria: To evaluate the application accuracy, we compare the imputed or repaired matrix $\mathcal{R}_\Psi(\mathbf{X}^*)$ to the corresponding ground truth $\mathcal{R}_\Psi(\mathbf{X}^\#)$ with root-mean-square error:

$$\mathbf{RMS} = \sqrt{\frac{||\mathcal{R}_\Psi(\mathbf{X}^* - \mathbf{X}^\#)||_F^2}{|\Psi|}}$$

3) Implementation Details: The major competitors of our proposal are implemented as below.

(1) Neighbor-based kNN Ensemble (kNNE) [16] builds a NN classifier on each subset of complete columns for the incomplete tuples. Then, the NN classifiers will be collected to obtain the final results.

(2) Value regression model-based LOESS [13] learns regression models among nearest neighbors for imputation. Incomplete attributes are taken as the dependent attributes, while complete attributes are the determinants of the models.

(3) Individual regression model-based IIM [47] learns an individual regression model for each tuple together with its neighbors.

(4) Matrix-completion-based method MC [10] aims to complete a data matrix from a sampling of its entries. The imputation problem is formed as a convex optimization program, which can be solved by finding the matrix with the minimum nuclear norm.

(5) Distance-based DLM [38] learns a statistical model over the distances of a tuple to its neighbors. It determines the most likely filling by maximizing the distance likelihood.

(6) GAN-based GAIN [46] proposes to impute missing values by adapting Generative Adversarial Nets (GAN) framework. The generator (G) completes the data matrix according to the complete entries. The discriminator (D) learns to identify which entries were imputed or observed.

(7) Matrix Factorization-based SoftImpute [5], [35] iteratively replaces the missing elements by applying soft-thresholded SVD.

(8) Clustered Adversarial Matrix Factorization CAMF [42] combines matrix factorization and the structure of GAN to enhance the spatial data imputation.

(9) Iterative is IterativeImputer from the machine learning toolkit scikit-learn devised for multiple imputation [4].

(10) HoloClean [36] is a data repairing system driven by probabilistic inference. Due to the lack of integrity rules, we ran HoloClean with statistical signals to fix errors holistically.

(11) Baran [32] is an error correction system that fixes data errors with respect to their value, vicinity, and domain contexts. Following the default setting of Baran, we set the number of labels as 20.

(12) Nonnegative Matrix Factorization NMF is introduced in Section II-B, trying to impute missing values in data using NMF with the form of Formula 1. It is also studied in [41] to impute missing values with NMF.

TABLE IV: Imputation RMS error of our proposals compared to the existing methods on various datasets (missing rate 10%). OOT: Out of time error (take more than 24h). OOM: Out of memory error (take more than 128GB).

Dataset	kNNE	LOESS	IIM	MC	DLM	GAIN	SoftImpute	Iterative	CAMF	NMF	SMF	SMFL
Economic	0.101	0.097	0.069	0.135	0.032	0.076	0.055	0.043	0.137	0.135	0.042	**0.028**
Farm	0.243	0.487	0.137	0.215	0.082	0.259	0.067	0.058	0.253	0.221	0.066	**0.051**
Lake	0.141	0.280	0.511	0.083	0.060	0.085	0.064	0.053	0.122	0.086	0.057	**0.048**
Vehicle	0.221	0.256	OOT	0.301	0.163	0.204	0.161	0.150	OOM	0.224	0.067	**0.055**

TABLE V: Imputation RMS error of our proposals compared to the existing methods when spatial information is also missing. OOT: Out of time error (take more than 24h). OOM: Out of memory error (take more than 128GB).

Dataset	kNNE	LOESS	IIM	MC	DLM	GAIN	SoftImpute	Iterative	CAMF	NMF	SMF	SMFL
Economic	0.232	0.231	0.314	0.311	0.135	0.198	0.161	0.153	0.231	0.243	0.133	**0.126**
Farm	0.311	0.791	0.158	0.286	0.136	0.255	0.163	0.071	0.322	0.227	0.068	**0.058**
Lake	0.181	0.602	0.168	0.193	0.116	0.174	0.190	0.110	0.179	0.125	0.117	**0.096**
Vehicle	0.208	0.213	OOT	0.366	0.175	0.214	0.172	0.132	OOM	0.363	0.096	**0.090**

(13) Spatial Matrix Factorization (SMF) and Spatial Matrix Factorization with Landmark (SMFL) are introduced in Section II and Section III, respectively.

B. Comparison with Existing Approaches

In this section, we compare our proposal with the state-of-the-art methods both in imputation and repair, to verify the effectiveness of the proposed SMFL.

1) Imputation on Various Datasets: Table IV reports the imputation comparison results under the 10% missing rate. First, we observe that SMFL outperforms all the baselines over all datasets. This is not surprising due to the effectiveness of the MF methods and the guidance of the landmarks in SMFL.

Among all the baselines, distance-based DLM and statistic-based Iterative Imputer show better results. DLM tries to impute incomplete values via distance likelihood maximization. The procedure of maximizing the distance likelihood indeed leverages the spatial smoothness potentially. IterativeImputer predicts a column value by other columns, thus leveraging the spatial information during prediction. However, it (1) does not fully utilize the spatial information as SMFL and (2) cannot perform if many other columns are also missing. Nevertheless, they both account for the effectiveness of our SMFL by using smoothness and spatial information.

GAN-based methods GAIN and CAMF do not perform on spatial data. It is not surprising due to the difficulty of training the generative adversarial networks. In the meantime, the structure of the Generator and Discriminator does not include the usage of the spatial information.

Regression model-based IIM and LOESS do not show stable performance over the datasets. Both rely on the regression relationships among the tuples and columns, while the spatial information is not considered. The regression model relies on complete tuples and does not consider the spatial distribution.

Compared to NMF and SMF, it is apparent that SMFL outperforms both methods with lower RMS error. This demonstrates the advance of the landmarks again.

For time-consuming IIM and memory-consuming CAMF, the data size of Vehicle is too large to perform, with either out of time error (take more than 24h) or out of memory error (take more than 128GB). Our proposed SMFL still performs and shows the best performance among all the competitors. This again verifies the efficacy of the proposed SMFL.

TABLE VI: Repair RMS error of our proposals compared to the existing methods (error rate 10%)

Dataset	Baran	HoloClean	NMF	SMF	SMFL
Economic	0.163	0.065	0.133	0.040	**0.037**
Farm	0.227	0.194	0.257	0.083	**0.057**
Lake	0.135	0.078	0.110	0.067	**0.064**
Vehicle	0.266	0.188	0.261	0.151	**0.137**

Table V provides the results when the spatial information is also incomplete. Due to the importance of the spatial information, in most cases, the performances of these methods are worse than those with complete spatial information (Table IV). Nevertheless, SMFL still outperforms all the competitors.

2) Repair on Various Datasets: Table VI reports the repair RMS error of SMFL against repair methods Baran and HoloClean. Analogously, SMFL outperforms all baselines over all datasets, and SMF also shows satisfying repair results compared to baselines. Existing data repair techniques such as Baran and HoloClean are difficult to learn from the spatial information, thus do not perform in these datasets.

3) Application in Vehicle Route Planning: To evaluate the imputation results in the vehicle route planning application, we use different imputation methods to impute vehicle routes with missing fuel consumption rate, and then compute the accumulated fuel consumption of the route based on the imputed fuel consumption rate. The computed fuel consumption is then compared to the truth fuel consumption collected by our partner company. Figure 4(a) reports the absolute accumulated fuel consumption error of different methods. SMFL shows the lowest error of 0.0136L. It indicates that SMFL is more precise and effective when used for vehicle route planning.

4) Application in Clustering: We also analyze the matrix factorization-based methods in clustering applications over Lake dataset, since these methods all provide unsupervised clustering results. Following [8], the results are evaluated by accuracy. Given n tuples, we have

$$Accuracy = \max_{\sigma}\left(\frac{\sum_{i=1}^{n}\delta(truth[i], \sigma(pred[i]))}{n}\right),$$

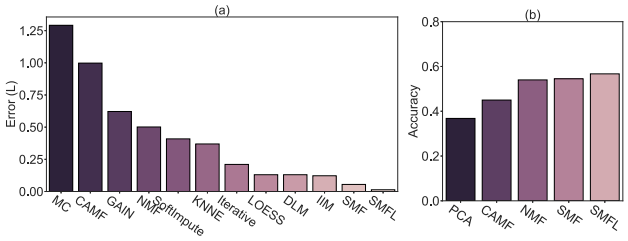

Fig. 4: Results of downstream applications. (a) Accumulated fuel consumption error of different approaches in the vehicle route planning application. (b) Clustering accuracy of different approaches in the clustering application.

TABLE VII: Imputation RMS error of NMF, SMF and SMFL over six datasets with different missing rates

Dataset	Algorithm	Missing Rate				
		10%	20%	30%	40%	50%
Economic	NMF	0.131	0.133	0.133	0.135	0.136
	SMF	0.037	0.037	0.042	0.041	0.044
	SMFL	**0.028**	**0.028**	**0.031**	**0.032**	**0.033**
Farm	NMF	0.221	0.219	0.219	0.221	0.222
	SMF	0.066	0.079	0.093	0.106	0.120
	SMFL	**0.051**	**0.062**	**0.078**	**0.091**	**0.101**
Lake	NMF	0.086	0.088	0.088	0.091	0.090
	SMF	0.057	0.062	0.063	0.066	0.069
	SMFL	**0.048**	**0.052**	**0.055**	**0.062**	**0.064**

where $\delta(x, y) = 1$ if $x = y$ and otherwise 0, $truth[i]$ and $pred[i]$ denote the i-th clustering label of ground truth and predicted data, respectively, and σ is a permutation mapping function from the predicted labels to the truth labels. The σ function aims to find a mapping to maximize the accuracy, which is determined by Kuhn-Munkres algorithm [31]. The results are provided in Figure 4(b). PCA [44] is a popular MF-based clustering method based on SVD. It is not surprising that SMFL still outperforms other baselines, since it fully utilizes the spatial data with landmarks.

C. Ablation Studies

We conduct ablation studies on SMFL to test the effectiveness of the proposed landmarks. In Section III-A, we propose landmarks to guide the generation of the latent features. The generation and injection of the landmarks are introduced, and SMFL is proposed in Problem 2. This experiment mainly evaluates two aspects: (1) do the landmarks truly improve the imputation performance; (2) do the generation and injection of the landmarks indeed improve the rationality and interpretability of the learned spatial locations?

Table VII presents the imputation results over several datasets with different missing rates. Compared to SMF, SMFL provides more accurate and robust results over different datasets.

According to the results in Table VII, spatial regularization (i.e., SMF) is truly effective, which motivates us to improve SMF by further investigating the spatial information such as landmarks. Although the further improvement by landmarks may not be as significant as spatial regularization, such an improvement is clearly observed and essential to beat some competitors. (1) Landmarks indeed improve the results of SMF in all the cases, as illustrated in Table VII. (2) SMFL outperforms all the baselines, while SMF fails in some datasets, according to Table IV. For instance, SMFL is the best in all the datasets of Table IV, while SMF shows worse results than DLM and Iterative in the Economic and Lake datasets, respectively.

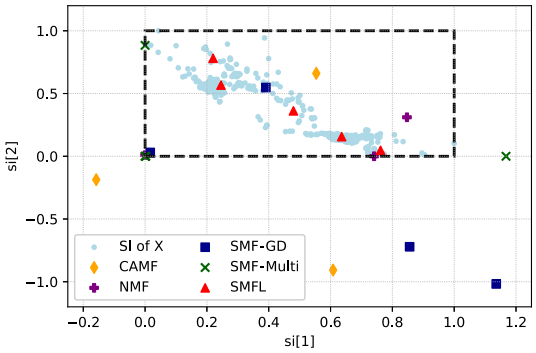

Fig. 5: Visualization of feature locations over a dataset, with $L = 2$ and $K = 5$. The dashed box is the region of all data observations. Some extremely distant features are not plotted in the figure for better visualization.

Following the motivation that landmarks will enhance the rationality and interpretability of the matrix factorization, we attempt to visualize the learned spatial features of SMFL and SMF in the first L dimensions of feature matrix $\mathbf{V}$. As shown in Figure 5, we visualize the learned spatial locations of matrix $\mathbf{V}$ learned by SMF and SMFL over one dataset ($L = 2$). Owing to the two updating methods (introduced in Section III-B) for SMF will lead to different matrix $\mathbf{V}$, we illustrate the matrix $\mathbf{V}$ learned by gradient descent (SMF-GD) and multiplicative updating rules (SMF-Multi), respectively. The x-axis and y-axis represent the locations of the first dimension and the second dimension of the spatial information $\mathbf{SI}$. Locations that represent spatial information in original matrix $\mathbf{X}$ are also shown in light blue.

According to Figure 5, features learned by SMF-GD (in dark blue) and SMF-Multi (in green) might be very distant from the observations. However, landmarks (in red) generated by SMFL control the locations of learned features, to make them geographically closer to the data observations. To conclude, compared to SMF, SMFL controls the locations of learned features by landmarks, thus providing more accurate and interpretable results. Similar to Figure 1, it also explains why the existing MF-based methods such as CAMF and NMF do not perform.

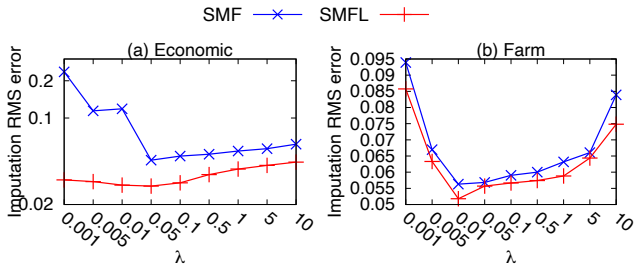

Fig. 6: Varying the regularization parameter λ

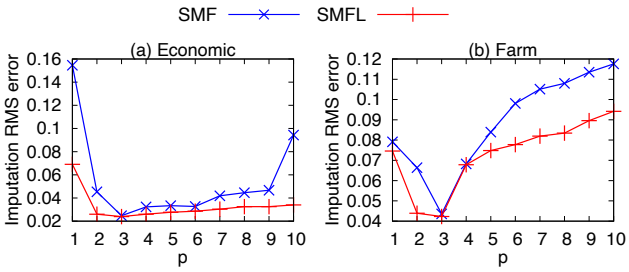

Fig. 7: Varying the number of spatial nearest neighbors p

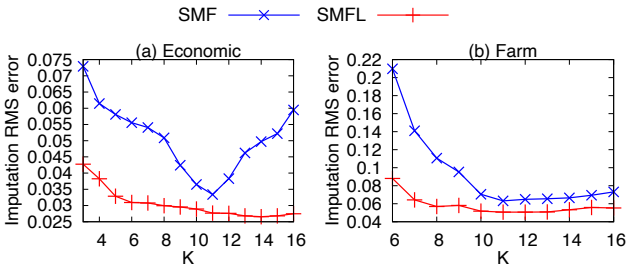

Fig. 8: Varying the number of landmarks K

D. Parameter Sensitivity

We conduct additional experiments to evaluate the parameter sensitivity of our proposal. Although the multiplicative updating rules are proposed in Section III-B to eliminate difficulties of tuning learning rates, parameters of the objective function still affect the performance of the proposed SMFL and SMF, including λ, p, and K.

1) Varying the Regularization Parameter λ: The regularization parameter λ is used to control the weight of the spatial regularization in the objective function of SMF and SMFL. We vary λ from 0.001 to 10 to test the effect of λ on performance. The results are reported in Figure 6. In general, a moderately small λ (0.05-0.1) is preferred since too large λ will overemphasize the spatial smoothness while ignoring the reconstruction of the original matrix. However, a too small λ may also neglect the spatial smoothness, thus restricting the performance of SMF and SMFL, also illustrated in Figure 6. Nevertheless, SMFL outperforms SMF almost over all datasets with various λ, demonstrating the robustness and effectiveness of the landmarks.

2) Varying the Number of Spatial Nearest Neighbors p: The parameter p controls the number of the spatial nearest neighbors to be considered in matrix $\mathbf{D}$ and graph Laplacian matrix $\mathbf{L}$. We vary p from 1 to 10 to test the impact of p on accuracy. The experimental results are illustrated in Figure 7. It is observed that a moderately small p is better for the algorithm. When p increases, the performance of both SMFL and SMF decreases. This is not surprising since a too large p will introduce tuples with low relevance into the graph Laplacian matrix and try to maintain the spatial smoothness over a large distance, which is unreasonable. In addition, a too small p could make the performance worse. Intuitively, a small p might prevent the model from learning from neighbors, thus lowering the performance. In our experiments, we find $p = 3$, i.e., the 3-nearest neighbor shows the best performance.

3) Varying the Number of Landmarks K: The parameter K is not only the number of the latent features in $\mathbf{V}$, but also the number of clusters for generating landmarks. The results are presented in Figure 8. According to the results, a moderately large K may potentially contribute to better performance. This is not surprising that too small K limits the ability of the matrix factorization model to learn the features, thus resulting in low accuracy results. Therefore, it is recommended to set a moderately large K.

E. Efficiency Evaluation

As explained in Section III-A, by fixing the landmarks during iterations, the time cost of updating landmark locations in matrix $\mathbf{V}$ can be saved. Compared to SMF, in implementation, the landmarks in $\mathbf{V}$ $(v_{kj}, (k, j) \in \Phi)$ are computed once and reserved for each iteration. Therefore, $\mathbf{V}$ is divided into two parts and only one of them is updated during iterations. According to Formula 14, the saving computation is related to data size, i.e., the efficiency of SMFL is affected by the data size. We thereby conduct efficiency evaluation by measuring the running time while varying the size of the data.

Figure 9 illustrates the time performance results of different methods over two datasets. Overall, SMFL is more efficient than neighbor-based methods (KNNE), GAN-based methods (GAIN and CAMF) and statistics-based methods (DLM). This is because neighbor-based and statistics-based methods may consider complex relationships among multiple columns, thus requiring higher time. GAN-based methods are also time-consuming in model training.

SMFL shows comparable efficiency among MF-based methods (MC, SoftImpute and SMF) and Regression-based methods (IterativeImputer). IterativeImputer and SoftImpute are faster in Lake dataset (with 7 columns), while slower in Economic dataset (with 11 columns). In other words, SMFL is more scalable in higher dimension of data.

It is notable that SMFL also shows slightly better efficiency than SMF, due to the following reasons. (1) While K-means clustering is not a low-overhead approach, it does not dominate in terms of time complexity in SMFL, as analyzed in Propositions 1. (2) Landmarks could reduce computation cost

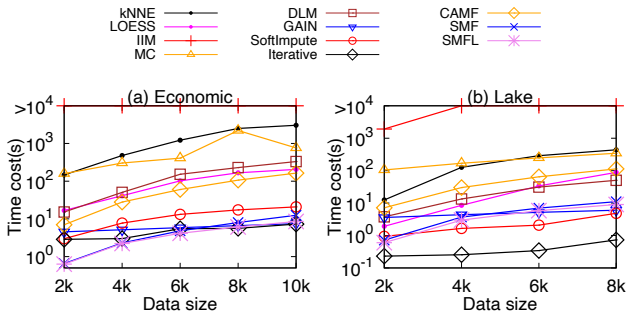

Fig. 9: Time cost of the proposed methods over two datasets while varying the number of tuples

in the matrix updating process, according to Section III-B, where the reduced part is related to the data size.

V. Related Work

A. Matrix Factorization

Matrix Factorization (MF) aims to decompose an original matrix into the product of two or three matrices and then tries to utilize the decomposed matrices for downstream applications such as graph mining [21], recommender systems [49], and data imputation [35].

MC [10] and SoftImpute [35] are both based on Singular Value Decomposition (SVD), which decomposes the original matrix into three matrices, including two unitary matrices and one singular value matrix. The benefit is that they provide exact factorization of the original matrix and learn the eigenvalues. However, for spatial data imputation, SVD fails to consider the spatial information.

Another widely used matrix factorization algorithm is Nonnegative Matrix Factorization (NMF) [25]. NMF attempts to factorize the original matrix into the product of two nonnegative matrices. [9] utilizes NMF and graph Laplacian regularizer for data representation and benefits downstream clustering tasks. However, it does not take missing values into consideration. [41] is a matrix completion method that adopts the original NMF algorithm.

B. Missing Data Imputation

As discussed in Section IV-A3, existing missing data imputation methods that support numerical data can be categorized as follows.

1) Neighbor-based Methods: kNN [6] imputes an incomplete tuple according to its top-K nearest neighbors. kNNE [16] builds a set of kNN models over various subsets of attributes, and combines their results. Owing to the dependency of complete neighbors, neighbor-based methods are limited by data redundancy, especially when the missing rate is high.

2) Regression-based Methods: LOESS [13] learns a regression model over the complete neighbors for imputation rather than simply aggregating candidates returned by neighbors. IIM [47] proposes to learn a regression model individually for each tuple together with its neighbors. Analogously, IterativeImputer [4] also employs regression models to learn from other attributes. These methods rely on the regression relationships among the tuples and attributes, thus not stable and robust when applied to various spatial data, especially with high missing rates.

3) Statistics-based Methods: ERACER [34] models the probabilistic correlations between attributes. DLM [38] learns a statistical model over the distances of a tuple to its neighbors, which indeed leverages the spatial smoothness potentially. Both methods determine the imputation values which maximize the likelihood referring to the statistical models.

4) GAN-based Methods: GAIN [46] leverages the structure of Generative Adversarial Nets (GAN) [20] for imputation. CAMF [42] combines matrix factorization and GAN to impute missing data. Moreover, in CAMF, spatial information is regarded as the prior knowledge to evaluate the distance and is used in the reconstruction component. However, since spatial smoothness and hidden guidance of the spatial information are not considered, CAMF does not perform.

C. Data Repairing

There exist a variety of data repairing methods. SCARE [45] repairs errors with clean values based on their statistical likelihood. Holistic [11], NADEEF [15] and IncRep [14] leverage integrity rules to clean data. KATARA [12] and eRs [17] fix the data errors by considering external data sources. Baran [32] trains multiple corrector models and then combines them into a final correction for each data error. HoloClean [36] leverages integrity constraints, matching dependencies and statistical properties for data repairing.

VI. Conclusion

In this paper, to fully utilize the spatial information, we propose a novel approach of **S**patial **M**atrix **F**actorization with **L**andmarks (SMFL). Intuitively, we interpret the learned features in terms of locations. Rather than arbitrarily distributed features in the general MF learning, we use landmarks to ensure the learned features are geographically close to the data observations and thus better guide the training of the features. The landmarks not only enhance the imputation and repair accuracy but also improve the interpretability of the features and reduce the computation cost. Towards the iterative optimization problem, updating rules are devised together with convergence analysis. Finally, we conduct comprehensive experiments to demonstrate the superiority of our proposal SMFL over real-world datasets.

Acknowledgement: This work is supported in part by the National Natural Science Foundation of China (62021002, 62072265, 62232005), the National Key Research and Development Plan (2021YFB3300500), Beijing National Research Center for Information Science and Technology (BNR2022RC01011), and Alibaba Group through Alibaba Innovative Research (AIR) Program. Shaoxu Song (https://sxsong.github.io/) is the corresponding author.

REFERENCES

[1] *https://github.com/imputation-landmark/imputation-landmark*.

[2] *https://gecon.yale.edu/data-and-documentation-g-econ-project*.

[3] *https://geodacenter.github.io/data-and-lab/lasrosas/*.

[4] Iterativeimputer. *https://scikit-learn.org/stable/index.html*.

[5] Softimpute. *https://github.com/iskandr/fancyimpute*.

[6] N. S. Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992.

[7] Y. Bengio. Practical recommendations for gradient-based training of deep architectures. In *Neural networks: Tricks of the trade*, pages 437–478. Springer, 2012.

[8] D. Cai, X. He, and J. Han. Document clustering using locality preserving indexing. *IEEE Trans. Knowl. Data Eng.*, 17(12):1624–1637, 2005.

[9] D. Cai, X. He, J. Han, and T. S. Huang. Graph regularized nonnegative matrix factorization for data representation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(8):1548–1560, 2011.

[10] E. J. Candès and B. Recht. Exact matrix completion via convex optimization. *Found. Comput. Math.*, 9(6):717–772, 2009.

[11] X. Chu, I. F. Ilyas, and P. Papotti. Holistic data cleaning: Putting violations into context. In *ICDE*, pages 458–469. IEEE Computer Society, 2013.

[12] X. Chu, M. Ouzzani, J. Morcos, I. F. Ilyas, P. Papotti, N. Tang, and Y. Ye. KATARA: reliable data cleaning with knowledge bases and crowdsourcing. *Proc. VLDB Endow.*, 8(12):1952–1955, 2015.

[13] W. S. Cleveland and C. Loader. Smoothing by local regression: Principles and methods. In *Statistical theory and computational aspects of smoothing*, pages 10–49. Springer, 1996.

[14] G. Cong, W. Fan, F. Geerts, X. Jia, and S. Ma. Improving data quality: Consistency and accuracy. In *VLDB*, pages 315–326. ACM, 2007.

[15] M. Dallachiesa, A. Ebaid, A. Eldawy, A. K. Elmagarmid, I. F. Ilyas, M. Ouzzani, and N. Tang. NADEEF: a commodity data cleaning system. In *SIGMOD Conference*, pages 541–552. ACM, 2013.

[16] C. Domeniconi and B. Yan. Nearest neighbor ensemble. In *17th International Conference on Pattern Recognition, ICPR 2004, Cambridge, UK, August 23-26, 2004*, pages 228–231. IEEE Computer Society, 2004.

[17] W. Fan, J. Li, S. Ma, N. Tang, and W. Yu. Towards certain fixes with editing rules and master data. *VLDB J.*, 21(2):213–238, 2012.

[18] P. J. García-Laencina, J.-L. Sancho-Gómez, A. R. Figueiras-Vidal, and M. Verleysen. K nearest neighbours with mutual information for simultaneous classification and missing data imputation. *Neurocomputing*, 72(7-9):1483–1493, 2009.

[19] N. Gillis and F. Glineur. Low-rank matrix approximation with weights or missing data is np-hard. *SIAM J. Matrix Anal. Appl.*, 32(4):1149–1165, 2011.

[20] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS*, pages 2672–2680, 2014.

[21] P. Han, P. Yang, P. Zhao, S. Shang, Y. Liu, J. Zhou, X. Gao, and P. Kalnis. GCN-MF: disease-gene association identification by graph convolutional networks and matrix factorization. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pages 705–713. ACM, 2019.

[22] D. Harel and Y. Koren. Clustering spatial data using random walks. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, San Francisco, CA, USA, August 26-29, 2001*, pages 281–286. ACM, 2001.

[23] Z. Jiang. A survey on spatial prediction methods. *IEEE Trans. Knowl. Data Eng.*, 31(9):1645–1664, 2019.

[24] A. Karpatne, A. Khandelwal, S. Boriah, and V. Kumar. Predictive learning in the presence of heterogeneity and limited training data. In *Proceedings of the 2014 SIAM International Conference on Data Mining, Philadelphia, Pennsylvania, USA, April 24-26, 2014*, pages 253–261. SIAM, 2014.

[25] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[26] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *NIPS*, pages 556–562. MIT Press, 2000.

[27] M. Li, F. D. Mauro, K. S. Candan, and M. L. Sapino. Matrix factorization with interval-valued data. In *36th IEEE International Conference on Data Engineering, ICDE 2020, Dallas, TX, USA, April 20-24, 2020*, pages 2042–2043. IEEE, 2020.

[28] S. Z. Li, X. Hou, H. Zhang, and Q. Cheng. Learning spatially localized, parts-based representation. In *2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2001), with CD-ROM, 8-14 December 2001, Kauai, HI, USA*, pages 207–212. IEEE Computer Society, 2001.

[29] X. Li, W. K. Cheung, J. Liu, and Z. Wu. A novel orthogonal nmf-based belief compression for pomdps. In *ICML*, volume 227, pages 537–544. ACM, 2007.

[30] N. K. Logothetis and D. L. Sheinberg. Visual object recognition. *Annual review of neuroscience*, 19(1):577–621, 1996.

[31] L. Lovász and M. D. Plummer. *Matching theory*, volume 367. American Mathematical Soc., 2009.

[32] M. Mahdavi and Z. Abedjan. Baran: Effective error correction via a unified context representation and transfer learning. *Proc. VLDB Endow.*, 13(11):1948–1961, 2020.

[33] M. Mahdavi, Z. Abedjan, R. C. Fernandez, S. Madden, M. Ouzzani, M. Stonebraker, and N. Tang. Raha: A configuration-free error detection system. In *SIGMOD Conference*, pages 865–882. ACM, 2019.

[34] C. Mayfield, J. Neville, and S. Prabhakar. ERACER: a database approach for statistical inference and data cleaning. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2010*, pages 75–86. ACM, 2010.

[35] R. Mazumder, T. Hastie, and R. Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. *J. Mach. Learn. Res.*, 11:2287–2322, 2010.

[36] T. Rekatsinas, X. Chu, I. F. Ilyas, and C. Ré. Holoclean: Holistic data repairs with probabilistic inference. *Proc. VLDB Endow.*, 10(11):1190–1201, 2017.

[37] M. Scholz, F. Kaplan, C. L. Guy, J. Kopka, and J. Selbig. Non-linear pca: a missing data approach. *Bioinformatics*, 21(20):3887–3895, 2005.

[38] S. Song and Y. Sun. Imputing various incomplete attributes via distance likelihood maximization. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 535–545. ACM, 2020.

[39] P. A. Soranno, L. C. Bacon, M. Beauchene, K. E. Bednar, E. G. Bissell, C. K. Boudreau, M. G. Boyer, M. T. Bremigan, S. R. Carpenter, J. 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):gix101, 2017.

[40] K. Subbian and A. Banerjee. Climate multi-model regression using spatial smoothing. In *Proceedings of the 13th SIAM International Conference on Data Mining, May 2-4, 2013. Austin, Texas, USA*, pages 324–332. SIAM, 2013.

[41] T. Suyama, Y. Kishino, Y. Shirai, S. Mizutani, and H. Sawada. Interpolation of missing data in sensor networks using nonnegative matrix factorization. In *UbiComp/ISWC*, pages 263–266. ACM, 2018.

[42] Q. Wang, P. Tan, and J. Zhou. Imputing structured missing values in spatial data with clustered adversarial matrix factorization. In *IEEE International Conference on Data Mining, ICDM 2018, Singapore, November 17-20, 2018*, pages 1284–1289. IEEE Computer Society, 2018.

[43] K. Q. Weinberger, F. Sha, Q. Zhu, and L. K. Saul. Graph laplacian regularization for large-scale semidefinite programming. In *Advances in Neural Information Processing Systems 19*, pages 1489–1496. MIT Press, 2006.

[44] S. Wold, K. Esbensen, and P. Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

[45] M. Yakout, L. Berti-Équille, and A. K. Elmagarmid. Don't be scared: use scalable automatic repairing with maximal likelihood and bounded changes. In *SIGMOD Conference*, pages 553–564. ACM, 2013.

[46] J. Yoon, J. Jordon, and M. van der Schaar. GAIN: missing data imputation using generative adversarial nets. In *ICML*, volume 80, pages 5675–5684. PMLR, 2018.

[47] A. Zhang, S. Song, Y. Sun, and J. Wang. Learning individual models for imputation. In *35th IEEE International Conference on Data Engineering, ICDE 2019, Macao, China, April 8-11, 2019*, pages 160–171. IEEE, 2019.

[48] X. Zhang, K. Xie, S. Wang, and Z. Huang. Learning based proximity matrix factorization for node embedding. In *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*, pages 2243–2253. ACM, 2021.

[49] X. Zheng, H. Ding, H. Mamitsuka, and S. Zhu. Collaborative matrix factorization with multiple similarities for predicting drug-target interactions. In *KDD*, pages 1025–1033. ACM, 2013.