Noname manuscript No.
(will be inserted by the editor)

Reducing Uncertainty of Dynamic Heterogeneous Information Networks: A Fusing Reconstructing Approach

Ning Yang · Lifang He · Zheng Li · Philip S. Yu

Received: date / Accepted: date

Abstract In real world, a heterogeneous information network (HIN) is often dynamic due to the time varying features of the nodes, and uncertain due to missing values and noise. In this paper, we investigate the problem of reducing the uncertainty of a dynamic HIN, which is an important task for HIN analysis. The challenges are three-fold, the heterogeneity of features, the heterogeneity of constraints, and the dynamic uncertainty. We propose a novel approach, called Fusing Reconstruction (FRec), which reconstructs the uncertain snapshots of a dynamic HIN in a homogeneous feature space combining two fusions, the fusion of heterogeneous features and the fusion of heterogeneous constraints. To address the challenge of the heterogeneity of features, we propose an Invertible Fusing Transformation (IFT) as the first part of FRec. IFT is a bidirectional transformation, which is able to learn unified latent homogeneous feature representations for heterogeneous nodes and transform them back to the raw heterogeneous feature space by its invertibility. To address the challenge of the heterogeneity of constraints and the challenge of dynamic uncertainty, we propose a Heterogeneous Constraints Fusion based Tensor Reconstruction Model (HCF-TRM) as the second part of FRec. HCF-TRM is able to denoise the uncertain snapshots of a dynamic HIN and recovers the missing values by fusing the spatial smoothness constraint and the temporal smoothness constraint into the tensor reconstruction. At last, the extensive exper-

Ning Yang
School of Computer Science, Sichuan University, China
E-mail: yangning@scu.edu.cn

Lifang He
Computer Vision Institute, Shenzhen University, China
E-mail: lifanghescut@gmail.com

Zheng Li
Corresponding author
School of Computer Science, Sichuan University, China
E-mail: lizheng@scu.edu.cn

Philip S. Yu
Department of Computer Science, University of Illinois at Chicago, USA
E-mail: psyu@uic.edu

iments conducted on real datasets and synthetic datasets verify the effectiveness and scalability of FRec.

Keywords Heterogeneous information network; Invertible fusing transformation; Graph embedding; Sparse tensor approximate

1 Introduction

Heterogeneous Information Networks (HINs) are networks consisting of interconnected objects of different types, and serve as a new network model for many offline and online applications including environment monitoring networks, e-commerce networks, social networks (e.g. LBSN), research publication networks (e.g. DBLP), etc. Generally, an HIN always evolves over time as nodes are associated with one or more features (measurements) of which the values vary over time. For example, in an environment monitoring network, the nodes representing observation stations in an environment monitoring network might record temperature, humidity, traffic, and some other measurements, and the values of these measurements always change over time. In DBLP, a node representing a conference may have a property of how active different topics are every year, and a node representing a topic term may have a property of how attractive that topic is to researchers every year, and both of the two properties are time-varying. The records of the feature values of all nodes at a specific time form a *snapshot* of a dynamic HIN, and all the snapshots arranged in chronological order form the *history* of the dynamic HIN. In a big data era, applications often need to leverage the rich knowledge hidden in histories of dynamic HINs to improve their service quality.

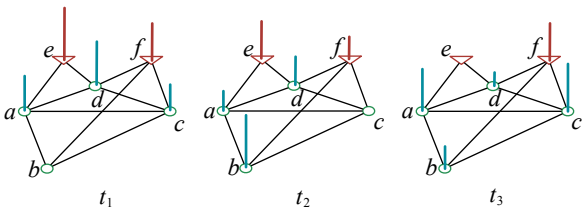


Fig. 1 Illustration of Dynamic Heterogeneous Information Networks

The historical snapshots of dynamic HINs, however, are always uncertain due to missing values and noise. Fig. 1 shows three snapshots of a environment monitoring network consisting of two types of monitoring stations over a town road network, where the green circle nodes represent the stations collecting meteorologic data (MD) such as humidity and pressure, the red triangle nodes represent the stations collecting air quality index (AQI), and the vertical line segments represent the values of MF and AQI, respectively. We can have two observations from Fig. 1: (1) The height of a vertical line segment changes over three snapshots, which means the measurements at MF nodes and AQI nodes are time-varying. (2) The nodes with missing value exist and change over time. For example, at time t_1, the node

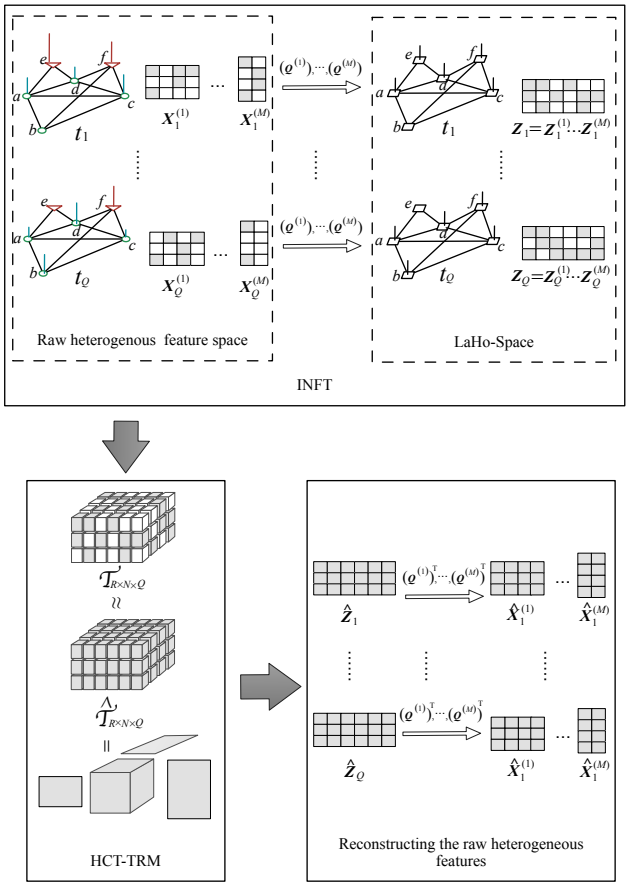


Fig. 2 Flow of FRec

with missing value is node b, while at time t_2, the node with missing value becomes node c, and at time t_3, node e. To get an insight of how our living environment changes over time, scientists need to recover the missing values of the historical snapshots.

In this paper, we investigate the problem of reducing the uncertainty of the history of a dynamic HIN, which is not easy due to the following challenges.

– **Heterogeneity of Features** In HINs, interconnected nodes of different types likely have heterogeneous features with different metric units and value ranges. For example, in Fig. 1, AQI is often measured by the density of aerosol particles smaller than 2.5 micrometers, which may take any real number greater than zero, while on an MF node, the humidity is often defined as the total mass of water vapor present in a given volume of air. It might lead to a large error to reconstruct missing values on a node simply by linearly combining the feature values on its heterogeneous neighbor nodes.

- **Heterogeneity of Constraints** A node usually generates its measurement under two different constraints. On one hand, since HINs always evolve over time, a node measurement depends on its own historical sequence, which we call *temporal constraint*. On the other hand, a node is in a network context, so a node measurement also depends on its connectivity to other nodes, which we call *spatial constraint*. It is a challenge to combine these two heterogeneous constraints in a unified framework.
- **Dynamic Uncertainty** As we can see from Fig. 1, in different snapshots, the nodes with missing value are different, which means the uncertainty of dynamic HINs is also dynamic. The dynamic uncertainty makes traditional methods such as matrix factorization and signal recovery inapplicable to dynamic HINs, since they are mostly motivated by static networks.

In this paper, we propose a novel approach to reducing the uncertainty of a dynamic HIN, called Fusion based Reconstruction (FRec). The main idea of FRec to address the aforementioned challenges is reconstructing the snapshots of a dynamic HIN in a homogeneous feature space by *two types of fusions, the fusion of heterogeneous features and the fusion of heterogeneous constraints*. The high level view of FRec is shown in Fig. 2.

To address the challenge of the heterogeneity of features, in this paper, we propose an Invertible Fusing Transformation (IFT) as the first part of FRec for the fusion of heterogeneous features, which is illustrated in the leftmost block of Fig. 2. IFT is able to transform the HIN nodes from their different raw heterogeneous feature spaces to a unified Latent Homogeneous Space (LaHo-Space), in which the nodes are represented by Latent Homogeneous Features (LaHo-Features) generated by IFT. Since the raw heterogeneous features are of different dimensions, we will learn an IFT matrix for each type of nodes. Note that the invertibility is a key property of IFT, by which IFT plays the role of a bridge between the raw heterogeneous feature space and the LaHo-Space. As one can see from the rightmost block of Fig. 2, once the LaHo-Features are reconstructed in the LaHo-Space, their corresponding raw heterogeneous features can be recovered by applying the inverse form of IFT to the reconstructed LaHo-Features.

To address the challenge of the heterogeneity of constraints and the challenge of dynamic uncertainty, we propose a Heterogeneous Constraints Fusion based Tensor Reconstruction Model (HCF-TRM) as the second part of FRec, to fulfil the reconstruction of the snapshots in the LaHo-Space. As shown in the middle block of Fig. 2, the core data structure of HCF-TRM is a homogeneous snapshot tensor, which is built by assembling the homogeneous snapshots in the LaHo-Space. To reconstruct the LaHo-Features, HCF-TRM approximates the homogeneous snapshot tensor by decomposing it into a tensor product of a low-rank core identity tensor and three factor matrices, with respect to a fusion of the two heterogeneous constraints. To meet the temporal constraint, HCF-TRM temporally smooths the reconstructed LaHo-Features, i.e., makes them similar between successive time points, while to meet the spatial constraint, HCF-TRM spatially smooths the reconstructed LaHo-Features, i.e., makes them similar with neighbors in the LaHo-Space. The major benefit of HCF-TRM is the reduction of the uncertainty, as it can infer the missing values by approximating the uncertain snapshot tensor, and denoise the features by smoothing them both temporally and spatially.

The major contributions of this paper can be summarized as follows:

(1) **FRec** We propose a novel approach, called Fusing Reconstruction (FRec), for reducing the uncertainty of dynamic HINs. FRec is able to reconstruct the historical snapshots of dynamic HINs by two fusions, heterogeneous features fusion and heterogeneous constraints fusion. To the best of our knowledge, FRec is the first work on uncertain dynamic HINs.
(2) **IFT** We propose an Invertible Fusing Transformation (IFT) as a bridge between the raw heterogeneous feature space and the LaHo-Space. IFT can transform the HIN nodes from their different raw heterogeneous feature spaces to a unified LaHo-Space, and its inverse form can recover the heterogeneous features from the reconstructed LaHo-Features in LaHo-Space. To our best knowledge, IFT is the first invertible transformation method for latent feature learning for HIN.
(3) **HCF-TRM** We propose a Heterogeneous Constraints Fusion based Tensor Reconstruction Model (HCF-TRM), which fulfills the LaHo-Feature reconstruction. HCF-TRM reduces the uncertainty by approximating and denoising the uncertain snapshot tensor.
(4) We conduct extensive experiments on real datasets and synthetic datasets to verify the performance of our approach.

The rest of this paper is organized as follows. The notations, preliminary concepts and the target problem are defined in Section 2. The details of IFT are described in Section 3. The details of HCF-TRM are described in Section 4. The overall algorithm of FRec is presented in Section 5. We analyze the experimental results in Section 6. At last, we briefly review the related work in Section 7 and conclude in Section 8.

2 Preliminaries

In this section, we first introduce the mathematical symbols, then define the basic concepts, and finally give the formal statement of the target problem.

2.1 Notations

In this paper, we use bold lower case letters such as $\boldsymbol{x}, \boldsymbol{y}$ to represent column vectors, and the ith component of $\boldsymbol{x}$ is represented by x_i. We use bold upper case letters such as $\boldsymbol{A}, \boldsymbol{B}$ to represent matrices, and the ith row of $\boldsymbol{A}$ is represented by $\boldsymbol{A}_{i:}$ and the jth column $\boldsymbol{A}_{:j}$. Sets are represented by regular upper case letters such as V, E, particularly, empty set is represented by Φ. The cardinality of V is denoted by $|V|$. Tensors are represented by calligraphic letters such as $\mathcal{T}, \mathcal{Q}$.

2.2 Basic Definitions

A heterogeneous information network [7,25] is defined as follow:

Definition 1 Heterogeneous Information Network (HIN) An HIN is an undirected graph $G = (V, \boldsymbol{W})$, where $V = \{v_1, \cdots, v_N\}$ is a set of N nodes, and $\boldsymbol{W}$ is a weighted adjacent matrix of $N \times N$, where element $\boldsymbol{W}_{ij} > 0$ if and only if there exists a link between nodes v_i and v_j, otherwise $\boldsymbol{W}_{ij} = 0$. There exists a

type mapping function $f : V \mapsto O$, where $O = \{o_1, \cdots, o_M\}$ is a set of M node types.

Note that in this paper we assume that the weighted adjacent matrix $\boldsymbol{W}$ is known as input. Essentially, an entry $\boldsymbol{W}_{ij}$ measures the similarity between nodes v_i and v_j, so $\boldsymbol{W}$ can be generated based on the priori knowledge about the similarity between nodes, which depends on the concrete application scenarios one is handling. As examples, in Section 6, we will describe how to decide the $\boldsymbol{W}$ for the datasets used for the experiments.

Obviously, the whole node set V can be divided into M disjoint subsets $V^{(m)}, m = \{1, \cdots, M\}$, respectively corresponding to M different types, i.e., $V^{(m)} \bigcap V^{(l)} = \Phi$ for $\forall m \neq l$ and $V = \bigcup_{m=1}^{M} V^{(m)}$. Any node can uniquely fall into only one subset, and a node $v_i, i = \{1, \cdots, N\}$ belongs to a subset $V^{(m)}$, if and only if $f(v_i) = o_m$. Any node $v_i \in V^{(m)}$ is associated with a raw feature vector $\boldsymbol{x}_i^{(m)}$ of d_m dimensions. In a heterogeneous context, for any two nodes $v_i \in V^{(m)}, u_j \in V^{(l)}, m \neq l$, and $i, j = \{1, \cdots, N\}$, their raw feature vectors $\boldsymbol{x}_i^{(m)}$ and $\boldsymbol{x}_j^{(l)}$ often have different dimensionalities, i.e., $d_m \neq d_l$. For each node subset $V^{(m)}, m = \{1, \cdots, M\}$, we can build a raw feature matrix $\boldsymbol{X}^{(m)} \in \mathbb{R}^{d_m \times |V^{(m)}|}$ where the column vectors of $\boldsymbol{X}^{(m)}$ consist of the raw feature vectors $\boldsymbol{x}_i^{(m)}$ of the nodes $v_i \in V^{(m)}$ as shown in the leftmost block of Fig. 2. Now we can define the snapshot of an HIN as follow:

Definition 2 HIN Snapshot Given an HIN $G = (V, E)$, its snapshot at time point q is defined as the set $S_q = \{\boldsymbol{X}_q^{(1)}, \cdots, \boldsymbol{X}_q^{(M)}\}$, where $\boldsymbol{X}_q^{(m)}, m = \{1, \cdots, M\}$, is the state of $\boldsymbol{X}^{(m)}$ at time point q.

2.3 Problem Statement

As we have mentioned before, the snapshots of an HIN is often uncertain due to the missing values, and noise. The target problem of this paper can be formally stated as follow:

Given an HIN $G = (V, \boldsymbol{W})$, and its uncertain historical snapshot sequence $\langle S_1, \cdots, S_Q \rangle$, we want to reconstruct these snapshots in which the true values of the raw features are approximated in an acceptable accuracy.

3 Invertible Fusing Transformation (IFT)

In this section, we describe the details of the Invertible Fusing Transformation (IFT), which is an invertible mapping function able to transform HIN nodes from their corresponding raw heterogeneous feature spaces to a unified LaHo-Space.

3.1 Definition of IFT

As we have mentioned, an IFT should be orthogonal for its invertibility. Based on this ideas, we formally define IFT as follow:

Definition 3 IFT The IFT for node subset $V^{(m)}, m = \{1, \cdots, M\}$, is a transformation

$$\boldsymbol{Z}^{(m)} = \boldsymbol{Q}^{(m)} \boldsymbol{X}^{(m)}, \tag{1}$$

where $\boldsymbol{X}^{(m)} \in \mathbb{R}^{d_m \times |V^{(m)}|}$ and $\boldsymbol{Z}^{(m)} \in \mathbb{R}^{R \times |V^{(m)}|}$ are the raw feature matrix and the LaHo-Feature matrix of $V^{(m)}$, respectively, and $\boldsymbol{Q}^{(m)} \in \mathbb{R}^{R \times d_m}$ is the transformation matrix for $V^{(m)}$. $\boldsymbol{Q}^{(m)}$ satisfies the orthogonal constraint that the column vectors of $\boldsymbol{Q}^{(m)}$ are orthogonal with each other, i.e., ${\boldsymbol{Q}^{(m)}}^T \boldsymbol{Q}^{(m)} = \boldsymbol{I}$, where $\boldsymbol{I}$ is a $d_m \times d_m$ identity matrix.

It is easy to show that the orthogonality of the row vectors of $\boldsymbol{Q}^{(m)}$ leads to the invertibility of IFT, which indicates that the raw feature matrix can be recovered by the equation

$$\boldsymbol{X}^{(m)} = {\boldsymbol{Q}^{(m)}}^T \boldsymbol{Z}^{(m)}, \tag{2}$$

once the LaHo-Feature matrix $\boldsymbol{Z}^{(m)}$ is constructed.

3.2 Optimizing Objective

We learn an IFT $\boldsymbol{Q}^{(m)}$ for each node subset $V^{(m)}, m = \{1, \cdots, M\}$, and the M IFTs $\boldsymbol{Q}^{(1)}, \cdots, \boldsymbol{Q}^{(M)}$ can transform the nodes from their respective raw feature spaces to a unified LaHo-Space of R dimensions.

Objective Function We formulate IFT learning as the following minimization problem:

$$\begin{aligned} \underset{\boldsymbol{Q}^{(1)}, \cdots, \boldsymbol{Q}^{(M)}}{\operatorname{argmin}} \; & \mathcal{F}(\boldsymbol{Q}^{(1)}, \cdots, \boldsymbol{Q}^{(M)}) + \beta \mathcal{R}(\boldsymbol{Q}^{(1)}, \cdots, \boldsymbol{Q}^{(M)}), \\ & \text{s.t. } {\boldsymbol{Q}^{(m)}}^T \boldsymbol{Q}^{(m)} = \boldsymbol{I}, \boldsymbol{Q}^{(m)} \geq 0, m = \{1, \cdots, M\}, \end{aligned} \tag{3}$$

where $\mathcal{F}(\boldsymbol{Q}^{(1)}, \cdots, \boldsymbol{Q}^{(M)})$ is the loss function defined in LaHo-Space, $\mathcal{R}(\boldsymbol{Q}^{(1)}, \cdots, \boldsymbol{Q}^{(M)})$ is the regularization function for avoiding overfitting, and $\beta > 0$ is the balancing parameters. Now we describe the loss function and the regularization function respectively.

Loss Function $\mathcal{F}$ Our idea to design the loss function is first based on the fact that in a LaHo-Space, any two objects, even of different types, are homogeneous and consequently we can measure the similarity between them. For any two nodes $v_i \in V^{(m)}, u_j \in V^{(l)}, i, j = \{1, \cdots, n\}, m, l = \{1, \cdots, M\}$, the similarity between them in the LaHo-Space can be defined as the inner product of their corresponding LaHo-Features, i.e,

$$s(\boldsymbol{z}_i^{(m)}, \boldsymbol{z}_j^{(l)}) = \frac{{\boldsymbol{z}_i^{(m)}}^T}{\sqrt{w_i}} \frac{\boldsymbol{z}_j^{(l)}}{\sqrt{w_j}} = \frac{(\boldsymbol{Q}^{(m)} \boldsymbol{x}_i^{(m)})^T \boldsymbol{Q}^{(l)} \boldsymbol{x}_j^{(l)}}{\sqrt{w_i} \sqrt{w_j}} \tag{4}$$

where $w_i = \sum_{k=1}^{N} \boldsymbol{W}_{ik}$, $w_j = \sum_{k=1}^{N} \boldsymbol{W}_{jk}$. Note that the second equality in Equation (4) can be derived according to Equation (1), and when $m = l$, Equation (4) returns the similarity between two nodes of the same type.

The loss function is supposed to reflect the linkage information of an HIN. It is reasonable to assume that the nodes that are neighbors with each other

should incur less penalty than the nodes that are not neighbors [7]. Based on this assumption, we design the following indicator function:

$$c_{ij} = \begin{cases} 1 & \text{if } \boldsymbol{W}_{ij} > 0, \\ -1 & \text{if } \boldsymbol{W}_{ij} = 0. \end{cases}$$

Based on this indicator function and Equation (4), for any $m, l \in 1, \cdots, M, m \neq l$, the loss incurred by IFTs $\boldsymbol{Q}^{(m)}$ and $\boldsymbol{Q}^{(l)}$ can be defined as:

$$g(\boldsymbol{Q}^{(m)}, \boldsymbol{Q}^{(l)}) = \sum_{i=1}^{|V^{(m)}|} \sum_{j=1}^{|V^{(l)}|} -c_{ij} s(\boldsymbol{z}_i^{(m)}, \boldsymbol{z}_j^{(l)}). \tag{5}$$

Based on Equation (5), we finally define the overall loss function as follow:

$$\begin{aligned} \mathcal{F}(\boldsymbol{Q}^{(1)}, \cdots, \boldsymbol{Q}^{(M)}) = & \sum_{m=1}^{M-1} \sum_{l=m+1}^{M} g(\boldsymbol{Q}^{(m)}, \boldsymbol{Q}^{(l)}) \\ & + \sum_{m=1}^{M} g(\boldsymbol{Q}^{(m)}, \boldsymbol{Q}^{(m)}). \end{aligned} \tag{6}$$

Note that in Equation (6), the first term represents the loss incurred by IFTs of different types ($m \neq l$), while the second term represents the loss incurred by IFT of the same type.

Regularization Function $\mathcal{R}$ The regularisation function $\mathcal{R}(\boldsymbol{Q}^{(1)}, \cdots, \boldsymbol{Q}^{(M)})$ is defined as:

$$\mathcal{R}(\boldsymbol{Q}^{(1)}, \cdots, \boldsymbol{Q}^{(M)}) = \sum_{m=1}^{M} \| \boldsymbol{Q}^{(m)} \|_F^2. \tag{7}$$

where $\| \cdot \|_F$ represents the Frobenius norm of a matrix.

3.3 IFT Learning

There are no closed-form solution to the optimization problem of Equation (3) due to the orthogonality constraints [30]. In this subsection, we will develop an iteratively learning algorithm for IFT with the preservation of the orthogonal constraint. Our basic idea is to alternately learn $\boldsymbol{Q}^{(1)}, \cdots, \boldsymbol{Q}^{(M)}$ by employing a variant of the classical gradient descent method [30]. We first fix $\boldsymbol{Q}^{(2)}, \cdots, \boldsymbol{Q}^{(M)}$, and learn $\boldsymbol{Q}^{(1)}$, and then fix $\boldsymbol{Q}^{(1)}, \boldsymbol{Q}^{(3)}, \cdots, \boldsymbol{Q}^{(M)}$, and learn $\boldsymbol{Q}^{(2)}$, and so on. Without loss of generality, we first describe the learning algorithm for a particular IFT $\boldsymbol{Q}^{(m)}, k = \{1, \cdots, m\}$, then give the learning algorithm for all IFTs.

3.3.1 Learning A Particular IFT $\boldsymbol{Q}^{(m)}$

Now we describe how to learn a single IFT $\boldsymbol{Q}^{(m)}$ when $\boldsymbol{Q}^{(1)}, \cdots, \boldsymbol{Q}^{(m-1)}, \cdots, \boldsymbol{Q}^{(m+1)}, \cdots, \boldsymbol{Q}^{(M)}$ are fixed. For simplicity, we omit the superscript of $\boldsymbol{Q}^{(m)}$ and

just denote it by $\boldsymbol{Q}$, then the minimization problem represented by Equation (3) becomes the following optimization problem:

$$\begin{aligned} &\underset{\boldsymbol{Q}}{\operatorname{argmin}} \ \mathcal{F}(\boldsymbol{Q}) + \beta\mathcal{R}(\boldsymbol{Q}), \\ &\text{s.t. } \boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{I}, \end{aligned} \tag{8}$$

where β is a weight parameter used to control the contribution of $\mathcal{R}(\boldsymbol{Q})$.

The Lagrangian function of Equation (8) is

$$\mathcal{L}(\boldsymbol{Q}, \boldsymbol{\Lambda}) = \mathcal{H}(\boldsymbol{Q}) - \frac{1}{2}\text{tr}(\boldsymbol{\Lambda}(\boldsymbol{Q}^T\boldsymbol{Q} - \boldsymbol{I})),$$

where $\mathcal{H}(\boldsymbol{Q}) = \mathcal{F}(\boldsymbol{Q}) + \beta\mathcal{R}(\boldsymbol{Q})$, and $\boldsymbol{\Lambda}$ is the Lagrangian multiplier matrix. Note that since the matrix $\boldsymbol{Q}^T\boldsymbol{Q}$ is symmetrical, $\boldsymbol{\Lambda}$ is also a symmetrical matrix. Now, the solution of Equation (8) is equivalent to the solution minimizing $\mathcal{L}(\boldsymbol{Q}, \boldsymbol{\Lambda})$. Note that when $\boldsymbol{Q}$ is a local optimizer of Equation (8), $\frac{\partial\mathcal{L}}{\partial\boldsymbol{Q}} = \boldsymbol{B} - \boldsymbol{Q}\boldsymbol{\Lambda} = 0$, where

$$\begin{aligned} \boldsymbol{B} = \frac{\partial\mathcal{H}}{\partial\boldsymbol{Q}} &= \nabla_{\boldsymbol{Q}}\mathcal{F}(\boldsymbol{Q}) + \beta\nabla_{\boldsymbol{Q}}\mathcal{R}(\boldsymbol{Q}) \\ &= (\sum_{l\neq m} \nabla_{\boldsymbol{Q}} g(\boldsymbol{Q}, \boldsymbol{Q}^{(l)}) + \nabla_{\boldsymbol{Q}} g(\boldsymbol{Q}, \boldsymbol{Q})) + 2\beta\boldsymbol{Q}, \end{aligned}$$

where

$$\nabla_{\boldsymbol{Q}} g(\boldsymbol{Q}, \boldsymbol{Q}^{(l)}) = \sum_{i=1}^{|V^{(m)}|}\sum_{j=1}^{|V^{(l)}|} c_{ij}\frac{\boldsymbol{Q}^{(l)}\boldsymbol{x}_j^{(l)}\boldsymbol{x}_i^T}{\sqrt{w_i}\sqrt{w_j}},$$

$$\nabla_{\boldsymbol{Q}} g(\boldsymbol{Q}, \boldsymbol{Q}) = \sum_{i=1}^{|V^{(m)}|}\sum_{j=1}^{|V^{(m)}|} c_{ij}\frac{\boldsymbol{Q}(\boldsymbol{x}_i\boldsymbol{x}_j^T + \boldsymbol{x}_j\boldsymbol{x}_i^T)}{\sqrt{w_i}\sqrt{w_j}}.$$

Considering $\boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{I}$, we have $\boldsymbol{\Lambda} = \boldsymbol{B}^T\boldsymbol{Q}$, then the gradient of $\mathcal{L}$ with respect to $\boldsymbol{Q}$ is

$$\nabla_{\boldsymbol{Q}}\mathcal{L} = \boldsymbol{B} - \boldsymbol{Q}\boldsymbol{B}^T\boldsymbol{Q} = \boldsymbol{A}\boldsymbol{Q}, \tag{9}$$

where

$$\boldsymbol{A} = \boldsymbol{B}\boldsymbol{Q}^T - \boldsymbol{Q}\boldsymbol{B}^T. \tag{10}$$

Naturally, according to Equation (9) the naive iterative update schema for $\boldsymbol{Q}$ is

$$\boldsymbol{Q}_i = \boldsymbol{Q}_{i-1} - \tau\nabla_{\boldsymbol{Q}_{i-1}}\mathcal{L} = \boldsymbol{Q}_{i-1} - \tau\boldsymbol{A}\boldsymbol{Q}_{i-1}, \tag{11}$$

where $\boldsymbol{Q}_i$ is the updated $\boldsymbol{Q}$ at i-th iteration, and τ is a step size. However, this update schema can not serve our goal because it can not guarantee that the orthogonality constraint ($\boldsymbol{Q}_i^T\boldsymbol{Q}_i = \boldsymbol{I}$) is always satisfied at every iteration [30].

In order to preserve the orthogonality constraint at each iteration, inspired by the idea presented in [30], we use a novel iterative update equation by modifying the gradient term of Equation (11) as follow:

$$\boldsymbol{Q}_i = \boldsymbol{Q}_{i-1} - \tau\boldsymbol{A}(\boldsymbol{Q}_{i-1} + \boldsymbol{Q}_i). \tag{12}$$

ALGORITHM 1: *LearningQ*$(\boldsymbol{X}^{(m)}, \{\boldsymbol{Q}^{(1)}, \cdots, \boldsymbol{Q}^{(m-1)}, \boldsymbol{Q}^{(m+1)}, \cdots, \boldsymbol{Q}^{(M)}\}, \tau, \epsilon)$

Input:
$\boldsymbol{X}^{(m)}$: the raw feature matrix of node subset of type m;
$\{\boldsymbol{Q}^{(1)}, \cdots, \boldsymbol{Q}^{(m-1)}, \boldsymbol{Q}^{(m+1)}, \cdots, \boldsymbol{Q}^{(M)}\}$: the fixed transformation matrices as constants;
τ: the step size;
η, ϵ: the thresholds for convergence;
Output:
$\boldsymbol{Q}^{(m)}$: the IFT matrix for $\boldsymbol{X}^{(m)}$;

1: Initialize $\boldsymbol{Q}_0^{(m)}$ s.t. ${\boldsymbol{Q}_0^{(m)}}^T \boldsymbol{Q}_0^{(m)} = \boldsymbol{I}$;
2: $i \leftarrow 0$;
3: **repeat**
4: $\quad i \leftarrow i + 1$
5: $\quad$ Build $\boldsymbol{A}$ according to Equation (10);
6: $\quad$ Build $\boldsymbol{P}$ according to Equation (14);
7: $\quad \boldsymbol{Q}_i^{(m)} \leftarrow \boldsymbol{P}\boldsymbol{Q}_{i-1}^{(m)}$, according to Equation (13);
8: **until** $\|\nabla_{\boldsymbol{Q}_i} \mathcal{H}(\boldsymbol{Q}_i)\| < \epsilon$
9: $\boldsymbol{Q}^{(m)} \leftarrow \boldsymbol{Q}_i^{(m)}$

After a simple linear algebra derivation, we can get the solution to Equation (12):

$$\boldsymbol{Q}_i = \boldsymbol{P}\boldsymbol{Q}_{i-1}, \tag{13}$$

where

$$\boldsymbol{P} = (\boldsymbol{I} + \tau\boldsymbol{A})^{-1}(\boldsymbol{I} - \tau\boldsymbol{A}). \tag{14}$$

The following theorem ensures that $\boldsymbol{Q}_i$ is orthogonal if $\boldsymbol{Q}_{i-1}$ is orthogonal.

Theorem 1 *If* $\boldsymbol{Q}_{i-1}^T \boldsymbol{Q}_{i-1} = \boldsymbol{I}$, *then* $\boldsymbol{Q}_i^T \boldsymbol{Q}_i = \boldsymbol{I}$.

Proof As $\boldsymbol{A}$ is an anti-symmetric matrix, for any nonzero vector $\boldsymbol{a}$, $\boldsymbol{a}^T(\boldsymbol{I}+\tau\boldsymbol{A})\boldsymbol{a} = \|\boldsymbol{a}\|_2^2$ holds true [10], i.e., $(\boldsymbol{I} + \tau\boldsymbol{A})$ is positive definite. Therefore $(\boldsymbol{I} + \tau\boldsymbol{A})$ is invertible, and $\boldsymbol{P}^T\boldsymbol{P} = (\boldsymbol{I} + \tau\boldsymbol{A})(\boldsymbol{I} + \tau\boldsymbol{A})^{-1}(\boldsymbol{I} - \tau\boldsymbol{A})^{-1}(\boldsymbol{I} - \tau\boldsymbol{A}) = \boldsymbol{I}$. Then $\boldsymbol{Q}_i^T\boldsymbol{Q}_i = \boldsymbol{Q}_{i-1}^T\boldsymbol{P}^T\boldsymbol{P}\boldsymbol{Q}_{i-1} = \boldsymbol{Q}_{i-1}^T\boldsymbol{Q}_{i-1} = \boldsymbol{I}$.

According to Theorem 1, if we initialize $\boldsymbol{Q}_0$ to be orthogonal, the orthogonality constraint will be preserved at each iteration. Algorithm 1 outlines the procedure of learning a particular IFT. In the ith outer iteration (Lines 3 to 8), Algorithm 1 updates $\boldsymbol{Q}_i^{(m)}$ with respect to Equation (13) until the gradient $\nabla_{\boldsymbol{Q}_i}\mathcal{H}(\boldsymbol{Q}_i)$ converges to a value less than the given threshold ϵ.

3.3.2 Learning All IFTs

The learning procedure for all the IFTs of all node types is shown as Algorithm 2. As we have mentioned, Algorithm 2 learns IFT matrices of all node subsets one by one (Lines 2 to 4) by repeatedly invoking Algorithm 1 (Line 3), and repeats this procedure until every learned IFT matrix converges (Lines 2 to 6). Note that the learning order of $\boldsymbol{Q}^{(m)}$ does not affect the convergency of Algorithm 2, because once a $\boldsymbol{Q}^{(m)}$ meets the stop learning condition $\|\nabla_{\boldsymbol{Q}^{(m)}}\mathcal{H}(\boldsymbol{Q}^{(m)})\| < \epsilon$ (Line 5), it will be fixed and will not be updated anymore. We will verify the convergency of Algorithm 2 in Section 6.

ALGORITHM 2: *LearningIFT*$(\boldsymbol{X}^{(1)}, \cdots, \boldsymbol{X}^{(M)}, \tau, \epsilon)$

Input:
$\boldsymbol{X}^{(1)}, \cdots, \boldsymbol{X}^{(M)}$: the raw feature matrices;
τ: the step size;
η, ϵ: the thresholds for convergence;
Output:
$\boldsymbol{Q}^{(1)}, \cdots, \boldsymbol{Q}^{(M)}$: the IFT matrices;

repeat
  for all $m \in \{1, \cdots, M\}$ **do**
    $\boldsymbol{Q}^{(m)} \leftarrow LearningQ(\boldsymbol{X}^{(m)}, \{\boldsymbol{Q}^{(1)}, \cdots, \boldsymbol{Q}^{(m-1)}, \boldsymbol{Q}^{(m+1)}, \cdots, \boldsymbol{Q}^{(M)}\}, \tau, \epsilon)$;
  end for
until $\|\nabla_{\boldsymbol{Q}^{(m)}} \mathcal{H}(\boldsymbol{Q}^{(m)})\| < \epsilon$ for all $m \in \{1, \cdots, M\}$

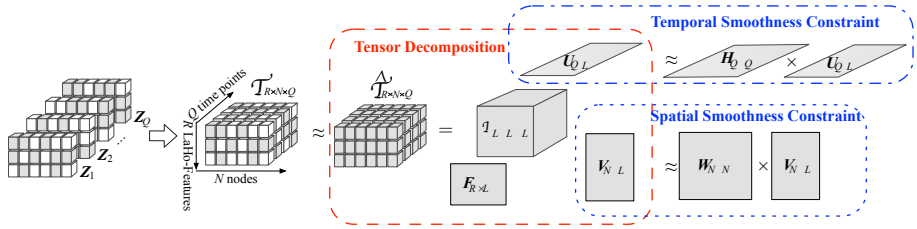


Fig. 3 HCF-TRM

4 Heterogeneous Constraints Fusion based Tensor Reconstruction Model (HCF-TRM)

In this section, we describe the details of our Heterogeneous Constraints Fusion based Tensor Reconstruction Model (HCF-TRM).

4.1 Homogeneous Snapshot Tensor

By IFTs, the raw feature matrices of each type of heterogeneous nodes are transformed to LaHo-Feature matrices $\boldsymbol{Z}^{(m)}$, $m \in \{1, \cdots, M\}$, and each HIN snapshot S_q, $q \in \{1. \cdots, Q\}$, has a corresponding homogenous snapshot $\widetilde{S}_q = \{\boldsymbol{Z}_q^{(1)}, \cdots, \boldsymbol{Z}_q^{(M)}\}$ in the LaHo-Space. Since the column vectors of each $\boldsymbol{Z}_q^{(m)}$, $m \in \{1, \cdots, M\}$, are of R-dimensional, we can build a bigger matrix $\boldsymbol{Z}_q$ of $R \times N$ by horizontally assembling $\boldsymbol{Z}_q^{(m)}$, i.e., $\boldsymbol{Z}_q = \boldsymbol{Z}_q^{(1)} \cdots \boldsymbol{Z}_q^{(M)}$.

Along the time dimension, we further assemble $\boldsymbol{Z}_1, \cdots, \boldsymbol{Z}_Q$ to a homogenous snapshot tensor. As shown in Fig. 3, the homogeneous snapshot tensor $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{R \times N \times Q}$ consists of three modes which respectively represent R LaHo-Features, N nodes, and Q time points. We call $\boldsymbol{Z}_q$, $q \in \{1, \cdots, Q\}$, a **temporal slice** of $\boldsymbol{\mathcal{T}}$ at time point q.

4.2 Tensor Reconstruction Model

In LaHo-Space, the homogeneous snapshot tensor $\boldsymbol{\mathcal{T}}$ still stay uncertain since IFT does not eliminate the uncertainty. As illustrated in Fig. 3, we construct $\widehat{\boldsymbol{\mathcal{T}}}$ as an estimate of $\boldsymbol{\mathcal{T}}$ by a CATD decomposition [14],

$$\boldsymbol{\mathcal{T}} \approx \widehat{\boldsymbol{\mathcal{T}}} = \boldsymbol{\mathcal{I}} \times_1 \boldsymbol{F} \times_2 \boldsymbol{V} \times_3 \boldsymbol{U}, \tag{15}$$

i.e., a core identity tensor $\boldsymbol{\mathcal{I}} \in \mathbb{R}^{L \times L \times L}$ multiplied by three latent factor matrices, $\boldsymbol{F} \in \mathbb{R}^{R \times L}$, $\boldsymbol{V} \in \mathbb{R}^{N \times L}$, $\boldsymbol{U} \in \mathbb{R}^{Q \times L}$, along its three modes respectively, where L is the target rank, and the symbol $\times_i$ $(1 \leq i \leq 3)$ stands for the tensor multiplication along the ith mode. One can note that the decomposition expressed by Equation (15) is meaningful. Actually, $\boldsymbol{F}$, $\boldsymbol{V}$, and $\boldsymbol{U}$ can be regarded as the latent feature factors about LaHo-Space, nodes, and time, respectively.

To construct $\widehat{\boldsymbol{\mathcal{T}}}$ as an estimate of uncertain $\boldsymbol{\mathcal{T}}$, we need additional knowledge about the tensor. In this paper, we fuse the following three additional constraints into the reconstruction:

(1) **Spatial Smoothness**. Spatial smoothness requires the LaHo-Feature variation with respect to the network structure of nodes be as small as possible, or equivalently the LaHo-Features of one node is supposed to be as close as possible to a linear combination of the LaHo-Feature vectors of its neighbors. As we can see from Fig. 3, we impose this constraint on HCF-TRM through the node latent factor matrix $\boldsymbol{V}$. Based on this idea, we define the spatial variation function as follow:

$$\phi(\boldsymbol{V}) = \|\boldsymbol{V}^T - \boldsymbol{V}^T\boldsymbol{W}\|_F^2, \tag{16}$$

where $\boldsymbol{W}$ is the weighted adjacent matrix of the HIN, and $\|\cdot\|_F$ is the Frobenius norm. The jth column vector of $\boldsymbol{V}^T$, $\boldsymbol{V}^T_{:j}$, can be regarded as the latent feature factor of the jth node, and the jth column vector of $\boldsymbol{V}^T\boldsymbol{W}$, $(\boldsymbol{V}^T\boldsymbol{W})_{:j}$, is a combination of the latent features factor of the neighbors of the jth node. In fact, $(\boldsymbol{V}^T\boldsymbol{W})_{:j} = \sum_{i=1}^{N} \boldsymbol{W}_{ij}(\boldsymbol{V}^T)_{:i}$, and note that $\boldsymbol{W}_{ij} = 0$ if $i = j$.

(2) **Temporal Smoothness**. Temporal smoothness requires the LaHo-Feature variation with respect to time be as small as possible. Similar as what we do for spatial smoothness, we impose the temporal smoothness constraint through the temporal latent factor matrix $\boldsymbol{U}$. We define the temporal variation function as follow:

$$\psi(\boldsymbol{U}) = \|\boldsymbol{U}^T - \boldsymbol{U}^T\boldsymbol{H}\|_F^2, \tag{17}$$

where $\boldsymbol{H}$ of $Q \times Q$ is a weighted temporal link matrix. An element $\boldsymbol{H}_{ij}$ represents the link weight between ith time point and jth time point, which is defined as

$$\boldsymbol{H}_{ij} = e^{-\alpha|i-j|}, \tag{18}$$

where $\alpha \geq 1$ is a decay factor. The intuition of the definition of $\boldsymbol{H}_{ij}$ is that closer two time points are, more dependence between them, and greater weight between them.

(3) **Small Noise**. We assume the noise is small, which means that the reconstructed LaHo-Features of the observable nodes (i.e. the nodes whose value are

ALGORITHM 3: *IDA*$(\boldsymbol{\mathcal{T}}, \boldsymbol{W}, \boldsymbol{H}, \epsilon)$

Input:
$\boldsymbol{\mathcal{T}}$: the homogeneous snapshot tensor;
$\boldsymbol{W}$: the weighted adjacent matrix of HIN;
$\boldsymbol{H}$: the weighted temporal link matrix;
ϵ: the error threshold;
Output:
$\boldsymbol{F}, \boldsymbol{V}, \boldsymbol{U}$: the factor matrices;
Initialize $\boldsymbol{F}, \boldsymbol{V}, \boldsymbol{U}$ with small random values;
$t \leftarrow 0$; Initialize Γ_0;
Set η as step size;
repeat
  $t \leftarrow t + 1$;
  for all $\boldsymbol{\mathcal{T}}_{rnq} \neq 0$ **do**
    $\boldsymbol{F}_{r:} = \boldsymbol{F}_{r:} - \eta \partial_{\boldsymbol{F}_{r:}} \Gamma$;
    $\boldsymbol{V}_{n:} = \boldsymbol{V}_{n:} - \eta \partial_{\boldsymbol{V}_{n:}} \Gamma$;
    $\boldsymbol{U}_{q:} = \boldsymbol{U}_{q:} - \eta \partial_{\boldsymbol{U}_{q:}} \Gamma$;
  end for
  Compute Γ_t;
until $|\Gamma_t - \Gamma_{t-1}| < \epsilon$

not missing) are supposed to be as similar as possible to their LaHo-Features before reconstruction. We express the noise as

$$\delta(\widehat{\boldsymbol{\mathcal{T}}}) = \|\widehat{\boldsymbol{\mathcal{T}}}_{\Omega} - \boldsymbol{\mathcal{T}}_{\Omega}\|_2^2, \tag{19}$$

where Ω represents the set of the indices of the observable tensor cells.

Based on these assumptions, we can define the optimization objective of HCF-TRM as to minimize the following loss function:

$$\begin{aligned} \Gamma(\boldsymbol{F}, \boldsymbol{V}, \boldsymbol{U}) = & \frac{\beta_1}{2}\delta(\widehat{\boldsymbol{\mathcal{T}}}) + \frac{\beta_2}{2}\phi(\boldsymbol{V}) + \frac{\beta_3}{2}\psi(\boldsymbol{U}) \\ & + \frac{\beta_4}{2}(\|\boldsymbol{F}\|_2^2 + \|\boldsymbol{V}\|_2^2 + \|\boldsymbol{U}\|_2^2) \end{aligned} \tag{20}$$

where $\|\boldsymbol{F}\|_2^2$, $\|\boldsymbol{V}\|_2^2$, and $\|\boldsymbol{U}\|_2^2$ are the regularization terms, and β_1, β_2, β_3, β_4 are the nonnegative parameters used to control the respective contributions of the terms, and $\sum_{i=1}^{4} \beta_i = 2$. Then HCF-TRM is reduced to the following optimization problem:

$$\underset{\boldsymbol{F}, \boldsymbol{V}, \boldsymbol{P}}{\operatorname{argmin}} \Gamma(\boldsymbol{F}, \boldsymbol{V}, \boldsymbol{U}). \tag{21}$$

4.3 Iterative Decomposition Algorithm

As it is hard to derive the closed-form solution to Equation (21), we present an Iterative Decomposition Algorithm (IDA) shown in Algorithm 3, which searches a suboptimal solution with the strategy of gradient descent.

At first, note that a cell of the homogeneous snapshot tensor, $\boldsymbol{\mathcal{T}}_{rnq}$, can be computed by

$$\boldsymbol{\mathcal{T}}_{rnq} = \boldsymbol{\mathcal{I}} \times_1 \boldsymbol{F}_{r:} \times_2 \boldsymbol{V}_{n:} \times_3 \boldsymbol{U}_{q:},$$

where $r \in \{1, \cdots, R\}$, $n \in \{1, \cdots, N\}$, and $q \in \{1, \cdots, Q\}$.

Next, note that at each iteration step, the update of the tensor is actually reduced to the update of each cell. For a cell $\boldsymbol{\mathcal{T}}_{rnq} \neq 0$, the gradient of the objective function $\Gamma(\boldsymbol{F}, \boldsymbol{V}, \boldsymbol{U})$ can be computed by the following equations:

$$\begin{aligned}
\partial_{\boldsymbol{F}_{r:}}\Gamma &= \beta_1(\widehat{\boldsymbol{\mathcal{T}}}_{rnq} - \boldsymbol{\mathcal{T}}_{rnq}) \times \boldsymbol{\mathcal{I}} \times_2 \boldsymbol{V}_{n:} \times_3 \boldsymbol{U}_{q:} + \beta_4 \boldsymbol{F}^T_{r:}, \\
\partial_{\boldsymbol{V}_{n:}}\Gamma &= \beta_1(\widehat{\boldsymbol{\mathcal{T}}}_{rnq} - \boldsymbol{\mathcal{T}}_{rnq}) \times \boldsymbol{\mathcal{I}} \times_1 \boldsymbol{F}_{r:} \times_3 \boldsymbol{U}_{q:} \\
&\quad + \beta_2(\boldsymbol{V}^T(\boldsymbol{I} - \boldsymbol{W}) \times (\boldsymbol{I} - \boldsymbol{W}))_{:n} + \beta_4 \boldsymbol{V}^T_{n:}, \\
\partial_{\boldsymbol{U}_{q:}}\Gamma &= \beta_1(\widehat{\boldsymbol{\mathcal{T}}}_{rnq} - \boldsymbol{\mathcal{T}}_{rnq}) \times \boldsymbol{\mathcal{I}} \times_1 \boldsymbol{F}_{r:} \times_2 \boldsymbol{V}_{n:} \\
&\quad + \beta_3(\boldsymbol{U}^T(\boldsymbol{I} - \boldsymbol{H}) \times (\boldsymbol{I} - \boldsymbol{H}))_{:q} + \beta_4 \boldsymbol{U}^T_{q:}.
\end{aligned} \tag{22}$$

4.4 Parallel Decomposition Algorithm

ALGORITHM 4: *PDA*$(\boldsymbol{\mathcal{T}}, \boldsymbol{W}, \boldsymbol{H}, \epsilon)$

Input:
$\boldsymbol{\mathcal{T}}$: the homogeneous snapshot tensor;
$\boldsymbol{W}$: the weighted adjacent matrix of HIN;
$\boldsymbol{H}$: the weighted temporal link matrix;
ϵ: the error threshold;
Output:
$\boldsymbol{F}, \boldsymbol{V}, \boldsymbol{U}$: the factor matrices;
1: Partition $\boldsymbol{\mathcal{T}}$ into a grid of sub-tensors, $\{\boldsymbol{\mathcal{T}}^{(\mathbf{k})}\}$;
2: Concurrently call $IDA(\boldsymbol{\mathcal{T}}^{(\mathbf{k})}, \boldsymbol{W}^{(\mathbf{k})}, \boldsymbol{H}^{(\mathbf{k})}, \epsilon)$ for each sub-tensor $\boldsymbol{\mathcal{T}}^{(\mathbf{k})}$, where $\boldsymbol{W}^{(\mathbf{k})}, \boldsymbol{H}^{(\mathbf{k})}$ are respectively the parts of $\boldsymbol{W}, \boldsymbol{H}$ corresponding to $\boldsymbol{\mathcal{T}}^{(\mathbf{k})}$;
3: Concurrently build $\boldsymbol{V}, \boldsymbol{U}, \boldsymbol{B}$ by iteratively concatenating the factors of the sub-tensors, $\boldsymbol{V}^{(\mathbf{k})}, \boldsymbol{U}^{(\mathbf{k})}, \boldsymbol{B}^{(\mathbf{k})}$, in terms of Lemma 1;

A huge and sparse tensor $\boldsymbol{\mathcal{T}}$ will incur a very high computational cost when we update the loss function (Equation (20)). To overcome this issue, we further design a Parallel Decomposition Algorithm (PDA) shown in Algorithm 4, which takes a divide-and-conquer strategy to fulfill the tensor decomposition. PDA first partitions the tensor $\boldsymbol{\mathcal{T}}$ into a grid of sub-tensors, $\mathfrak{T} = \{\boldsymbol{\mathcal{T}}^{(\mathbf{k})} | \mathbf{k} \in \mathcal{K}\}$, where $\mathcal{K}$ is a collection of sub-tensor indexes, $\mathcal{K} = \{[k_1, k_2, k_3] | 1 \leq k_1 \leq K_1, 1 \leq k_2 \leq K_2, 1 \leq k_3 \leq K_3\}$, and $K_i (1 \leq i \leq 3)$ is the number of sub-tensors along i-th mode. Then PDA concurrently factorizes the sub-tensors by invoking Algorithm 3, and at last, PDA integrates the partial results to produce the final factor matrices for the whole tensor. The following lemma ensures that a large-scale tensor decomposition can be obtained by integrating the factors of its sub-tensors [20].

Lemma 1 *If a tensor $\boldsymbol{\mathcal{T}}$ can be partitioned into two sub-tensors $\boldsymbol{\mathcal{T}}^{(1)}$ and $\boldsymbol{\mathcal{T}}^{(2)}$, and they are factorized as $\boldsymbol{\mathcal{T}}^{(1)} = \boldsymbol{\mathcal{I}}^{(1)} \times_1 \boldsymbol{F}^{(1)} \times_2 \boldsymbol{V}^{(1)} \times_3 \boldsymbol{U}^{(1)}$ and $\boldsymbol{\mathcal{T}}^{(2)} = \boldsymbol{\mathcal{I}}^{(2)} \times_1 \boldsymbol{F}^{(2)} \times_2 \boldsymbol{V}^{(2)} \times_3 \boldsymbol{U}^{(2)}$ respectively, then the tensor $\boldsymbol{\mathcal{T}}$ can be factorized as $\boldsymbol{\mathcal{T}} = \boldsymbol{\mathcal{I}} \times_1 \boldsymbol{F} \times_2 \boldsymbol{V} \times_3 \boldsymbol{U}$, where $\boldsymbol{\mathcal{I}} = \begin{bmatrix} \boldsymbol{\mathcal{I}}^{(1)} & \\ & \boldsymbol{\mathcal{I}}^{(2)} \end{bmatrix}$, $\boldsymbol{F} = \begin{bmatrix} \boldsymbol{F}^{(1)} & \boldsymbol{F}^{(2)} \end{bmatrix}$, $\boldsymbol{V} = \begin{bmatrix} \boldsymbol{V}^{(1)} & \boldsymbol{V}^{(2)} \end{bmatrix}$, and $\boldsymbol{U} = \begin{bmatrix} \boldsymbol{U}^{(1)} & \\ & \boldsymbol{U}^{(2)} \end{bmatrix}$.*

ALGORITHM 5: $FRec(V, \boldsymbol{W}, \langle S_1, \cdots, S_Q\rangle, \{\boldsymbol{Q}^{(1)}, \cdots, \boldsymbol{Q}^{(M)}\}, \epsilon)$

Input:
V: the HIN node set;
$\boldsymbol{W}$: the weighted adjacent matrix of HIN;
$\langle S_1, \cdots, S_Q\rangle$: the sequence of uncertain snapshots of HIN;
$\{\boldsymbol{Q}^{(1)}, \cdots, \boldsymbol{Q}^{(m)}\}$: the IFT matrices;
ϵ: the error threshold;
Output:
$\langle \widehat{S}_1, \cdots, \widehat{S}_Q\rangle$: the reconstructed snapshots;
1: **for all** $S_q, q \in \{1, \cdots, Q\}$ **do**
2: **for all** $\boldsymbol{X}_q^{(m)}, m \in \{1, \cdots, M\}$ **do**
3: $\boldsymbol{Z}_q^{(m)} = \boldsymbol{Q}^{(m)}\boldsymbol{X}_q^{(m)}$, according to Equation (1);
4: **end for**
5: Build the homogeneous time slice $\boldsymbol{Z}_q = \boldsymbol{Z}_q^{(1)} \cdots \boldsymbol{Z}_q^{(M)}$;
6: **end for**
7: Build the homogeneous snapshot tensor $\boldsymbol{\mathcal{T}}$ by assembling $\boldsymbol{Z}_q, q \in \{1, \cdots, Q\}$;
8: Build the weighted temporal link matrix $\boldsymbol{H}$ according to Equation (18);
9: Set the error threshold ϵ;
10: Generate the factor matrices $\boldsymbol{F}$, $\boldsymbol{V}$, and $\boldsymbol{U}$ by calling $PDA(\boldsymbol{\mathcal{T}}, \boldsymbol{W}, \boldsymbol{H}, \epsilon)$ (Algorithm 4);
11: Build the estimate tensor $\widehat{\boldsymbol{\mathcal{T}}} = \boldsymbol{\mathcal{I}} \times_1 \boldsymbol{F} \times_2 \boldsymbol{V} \times_3 \boldsymbol{U}$;
12: **for all** $q \in \{1, \cdots, Q\}$ **do**
13: **for all** $m \in \{1, \cdots, M\}$ **do**
14: Reconstruct the raw feature matrix for nodes of type m: $\widehat{\boldsymbol{X}}_q^{(m)} = {\boldsymbol{Q}^{(m)}}^T \widehat{\boldsymbol{Z}}_q^{(m)}$, according to Equation (23);
15: **end for**
16: Reconstruct the snapshot at time point q: $\widehat{S}_q = \{\widehat{\boldsymbol{X}}_q^{(1)}, \cdots, \widehat{\boldsymbol{X}}_q^{(M)}\}$;
17: **end for**

5 Fusing Reconstruction (FRec)

As illustrated in Fig. 2, once we generate the estimate tensor $\widehat{\boldsymbol{\mathcal{T}}}$, we can apply the inverse form of IFTs to reconstruct the raw heterogeneous feature matrices, $\widehat{\boldsymbol{X}}_1, \cdots, \widehat{\boldsymbol{X}}_Q$, from the temporal slices, $\widehat{\boldsymbol{Z}}_1 = \widehat{\boldsymbol{\mathcal{T}}}_{::1}, \cdots, \widehat{\boldsymbol{Z}}_Q = \widehat{\boldsymbol{\mathcal{T}}}_{::Q}$ (see the rightmost part of Fig. 2). Actually, according to Equation (2) introduced in Section 3.1, for $\forall q \in \{1, \cdots, Q\}$, and $\forall m \in \{1, \cdots, M\}$,

$$\widehat{\boldsymbol{X}}_q^{(m)} = {\boldsymbol{Q}^{(m)}}^T \widehat{\boldsymbol{Z}}_q^{(m)}, \tag{23}$$

where $\widehat{\boldsymbol{X}}_q^{(m)}$ is the raw feature matrix of the nodes of type m at time point q, and $\widehat{\boldsymbol{Z}}_q^{(m)}$ is its LaHo-Feature sub-matrix extracted from the temporal slice $\widehat{\boldsymbol{Z}}_q$.

Algorithm 5 outlines the procedures of FRec, which consists of three successive parts. The first part (Lines 1 to 6) applies IFT of each node type to transform the raw heterogeneous features to LaHo-Features, which corresponds to the leftmost block of Fig. 2. The second part (Lines 7 to 11) reconstructs the homogeneous snapshot tensor by applying HCF-TRM, which corresponds to the middle block of Fig. 2. At last, the third part (Lines 12 to 17) reconstructs the snapshots by applying the inverse form of IFT to transform the LaHo-Features back to the raw heterogeneous features, which corresponds to the rightmost block of Fig. 2.

6 Experiments

In this section, we first investigate the sensitivity of LaHo-Space dimensionality R, and the target rank L of HCF-TRM, and then verify the effectiveness and efficiency of FRec (IFT + HCF-TRM). The experiments are conducted on a hadoop cluster consisting of two PCs, each of which is equipped with a 2.7 GHz INTEL CPU and 32GB RAM. All the programs are written with MATLAB.

6.1 Experiment Setting

6.1.1 Datasets

We use three real datasets of different types and five synthetic datasets for our experiments. On each dataset, we use 80% of the data as the training set, and the remaining part as the test set.

Environment Monitoring Network (EMN) EMN is a dataset contains one-year (from Feb. 8, 2013 to Feb. 8, 2014) AQI data consisting of the measurements of $PM_{2.5}$, PM_{10}, SO_2, and NO_2, and meteorological data (MD) consisting of the measurements of temperature, humidity, and barometer pressure, collected by 22 environment monitoring stations in Beijing and 11 ones in Shanghai [36]. To build an EMN for each city, we randomly choose some stations as the AQI nodes, and some other stations as the MD nodes, and weight a link between any two nodes (monitoring stations) of different types or the same type according to the distance between monitoring stations. Since the data were recorded every one hour, we build a snapshot of EMN for each hour.

DBLP Base on the DBLP data from 2000 to 2012, we build a dynamic HIN consisting of two types of nodes, the 40 conference nodes and the 25 term nodes. The feature of a conference node is a term distribution vector where each component represents the number of accepted papers that relate to a specific term. The feature of a term node is a conference distribution of papers relating to that term. The weight schema of links is defined as follows: (1) A link between a conference node and a term node is weighted in terms of the proportion of the papers related to that term in the total papers published in that conference; (2) A link between two conference nodes is weighted in terms of the number of the papers that are accepted by the conferences and share at least one term; (3) A link between two term nodes is weighted in terms of the number of papers that share those two terms. We build 13 snapshots for the DBLP network, one for each year.

Foursquare Zheng *et al.* [4] release a location based social network (LBSN) dataset collected from Foursquare, which contains a collection of the check-in records that were generated at four USA cities in the years from 2008 to 2012. Based on this dataset, we build a dynamic heterogeneous network consisting of 5,918 venues nodes and 1,116 user nodes. We define the feature of a venue node as a 8-dimensional vector as representing the distribution of the check-ins at that venue over 8 time intervals of a day. The feature of a user node is defined as a 10-dimensional vector representing a frequency distribution of the check-ins of that user over 10 venue categories. The weight of a link between two venue nodes is defined in terms of the distance between them, the weight of a link between two user nodes is defined in terms of the number of their cooccurrences, and the

Table 1 Synthetic Datasets

Name	N	D	Q
SYN1	2K	10	10
SYN2	4K	10	10
SYN3	8K	10	10
SYN4	16K	10	10
SYN5	32K	10	10

weight of a link between a venue node and a user node is defined in terms of the times that user checked in at that venue. Similarly, we build 5 snapshots for the Foursquare network, one for each year.

Synthetic Datasets We use synthetic dataset to verify the efficiency of FRec. A synthetic dynamic network with a given number of nodes N is generated through 4 steps. First, we choose the raw feature space dimensionality D, and randomly generating a D-dimensional nonnegative feature vector for each node. Second, we build the weighted adjacent matrix in terms the cosine similarity between two nodes. Third, we choose the number Q of time points, and building Q snapshots, where each node feature linearly decays along the Q time points. At each snapshot, we also add a white noise of mean 0 and variance 1 to each node feature. We generate 5 synthetic datasets in total, as described in Table 1.

6.1.2 Metrics

To evaluate the effectiveness of FRec, we use the following two metrics: root mean square error (RMSE) and mean absolute error (MAE), which are defined as

$$RMSE = \sqrt{\frac{1}{QND}\sum_{m=1}^{M}\sum_{q=1}^{Q}\sum_{n=1}^{N}\sum_{r=1}^{d_m}((\boldsymbol{X}_q^{(m)})_{rn} - (\widehat{\boldsymbol{X}}_q^{(m)})_{rn})^2},$$

$$MAE = \frac{1}{QND}\sum_{m=1}^{M}\sum_{q=1}^{Q}\sum_{n=1}^{N}\sum_{r=1}^{d_m}|(\boldsymbol{X}_q^{(m)})_{rn} - (\widehat{\boldsymbol{X}}_q^{(m)})_{rn}|,$$

where $D = \sum_{m=1}^{M} d_m$.

6.1.3 Baseline Methods

We use the l_1-norm Low-Rank Matrix Approximation (LRMA) [1] and the CATD tensor decomposition model [29,37] as the alternatives of our HCF-TRM, which are both classical methods for missing value recovery. We will compare our FRec (IFT + HCF-TRM) with the different combinations of the alternatives, i.e., IFT+LRMA, IFT+CATD, LRMA, and CATD.

6.2 Sensitivity of R and L

In our experiments, we set the LaHo-Space dimensionality $R = 11$ and the HCF-TRM target rank $L = 10$ for the dataset EMN, $R = 8$ and $L = 11$ for the dataset

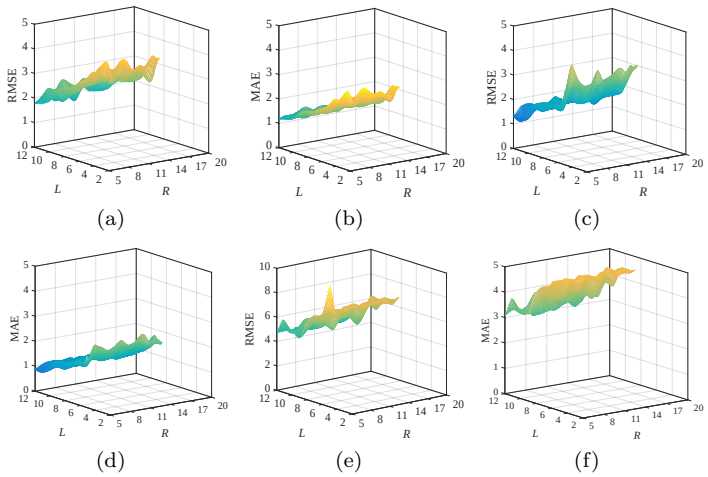


Fig. 4 Sensitivity of R and L. (a) RMSE on EMN. (b) MAE on EMN. (c) RMSE on DBLP. (d) MAE on DBLP. (e) RMSE on Foursquare. (d) MAE on Foursquare.

DBLP, and $R = 15$ and $L = 10$ for the dataset Foursquare. To see why they are good choices of R and L, we investigate their sensitivity to the performance of FRec by running FRec with different combinations of the values of R and L on the three real datasets. Fig. 4 (a), (b) show the results on EMN, and Fig. 4 (c), (d) the results on DBLP.

Theoretically, given a specific value of R, FRec will perform best when L equals the rank of the tensor. However, there is no straightforward algorithm to determine the rank of a given tensor [14]. We search a suboptimal value of L in a space defined by an upper bound of the rank of a tensor, which is stated in the following Lemma 2 [16].

Lemma 2 *For a general third-dimensional tensor* $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{I \times J \times K}$*, its rank* $L \leq min\{IJ, IK, JK\}$.

From Fig. 4 we see that fixing R at a value, all the metrics roughly decrease as L increases. This is consistent with the nature of tensor decomposition which is analogous to a Taylor expansion where a function can be approximated better with more derivative terms of higher order, and a low-rank approximation of a matrix where the matrix can be approximated more precisely by more leading factors of its SVD. In our HCF-TRM the homogeneous snapshot tensor will be reconstructed more precisely with a decomposition of higher target rank. However, a decomposition with too high a target rank will incur an overfitting, and that is why we see the RMSE and MAE become larger when $L > 10$ on EMN, $L > 11$ on DBLP, and $L > 10$ on Foursquare.

From Fig. 4 we also see that fixing L at a value, each metric roughly exhibits a curve that first falls and then rises after $R > 11$ on EMN, $R > 8$ on DBLP, and $R > 15$ on Foursquare. Intuitively, more latent features will make IFT more informative and raise the precision of the reconstruction from the LaHo-Space to the original raw feature space. However, again, IFT with too many latent features will incur an overfitting.

At last, from Fig. 4 we find that the points (11, 10), (8, 11) and (15, 10) on the $R - L$ plane are the local optimal points for EMN, DBLP, and Foursquare, respectively.

6.3 Effectiveness of FRec

Now we compare FRec with the baseline methods IFT+LRMA, IFT+CATD, LRMA, and CATD, to verify the effectiveness of FRec. On each dataset, we use 80% of the data as the training set and the remaining part as the test set. On the test set, we randomly remove 30% of cells and use their original values as the ground truth. Note that to verify IFT, we also compare FRec with applying LRMA and CATD alone to reconstruct the missing values in the raw feature space without feature transformation. To directly apply LRMA and CATD to the raw feature space where heterogeneous nodes have different dimensionality of features, we extend the raw feature vectors of nodes of each type to the same dimensionality which is sum of the dimensionalities of heterogeneous node features.

The results on EMN and DBLP are shown in Table 2 and Table 3, respectively, which show that FRec outperforms all the baseline methods on all datasets. We can make some reasonable analyses as follows.

Table 2 Effectiveness on EMN

Method	RMSE	MAE
FRec (IFT+HCF-TRM)	**1.2449**	**0.5191**
IFT+CATD	1.5370	1.0110
IFT+LRMA	1.9921	1.0816
CATD	3.1708	1.1220
LRMA	3.8253	1.3394

Table 3 Effectiveness on DBLP

Method	RMSE	MAE
FRec (IFT+HCF-TRM)	**3.3519**	**1.1749**
IFT+CATD	3.8100	1.2201
IFT+LRMA	3.8612	1.2307
CATD	4.3519	1.3050
LRMA	4.9073	1.4001

Table 4 Effectiveness on Foursquare

Method	RMSE	MAE
FRec (IFT+HCF-TRM)	**2.2960**	**1.6226**
IFT+CATD	4.0098	2.5982
IFT+LRMA	4.5331	3.0373
CATD	7.7675	5.0356
LRMA	8.0618	5.5237

At first, one can see that on all datasets, the RMSEs and MAEs of IFT+CATD and IFT+LRMA are much lower than those of applying CATD and LRMA alone to reconstruct the dynamic HINs directly in the raw feature space. This result verifies the proposed idea of reconstructing an uncertain dynamic HIN from a transformed homogeneous feature space (i.e., LaHo-Space) generated by IFT. In contrast with reconstruction without feature transformation, the advantage of IFT lies on its ability to overcome the challenge of feature heterogeneity by fusing the heterogeneous features into a homogeneous space in which missing features of a node can be recovered from the heterogeneous neighbors of that node.

Next, one can note that on all datasets, the the RMSE and MAE of FRec (IFT+HCF-TRM) are lower than those of IFT+CATD and IFT+LRMA, which verifies the effectiveness of our HCF-TRM. Unlike CATD and LRMA, HCF-TRM is able to overcome the challenge of constraint heterogeneity by fusing the temporal constraint and spatial constraint into the tensor decomposition model. LRMA treats each snapshot separately and neglects both the temporal smoothness and spatial smoothness of the features of HIN nodes. Although CATD takes the time dimension into account and assembles the snapshots to a unified tensor, it neither smoothes the factor matrix $\boldsymbol{U}$ along temporal dimension, nor smoothes the factor matrix $\boldsymbol{V}$ along spatial dimension, while HCF-TRM does both (see Fig. 3).

6.4 Efficiency of FRec

We investigate the efficiency of FRec by comparing it with its native version called Native-FRec which fulfils the HCF-TRM by directly invoking IDA (Algorithm 3) instead of PDA (Algorithm 4). The efficiency of FRec mainly depends on the tensor size $R \times N \times Q$ (where R is the dimensionality of LaHo-Feature space, N is the number of nodes, and Q is the number of snapshots), the number M of different node types, and the number of parallel threads. We thus first compare the running time of FRec with that of Native-FRec over the 6 synthetic datasets which have different tensor sizes, and then over different M. At last, we check the speedup ratio of FRec and Native-FRec versus the number of parallel threads.

6.4.1 Running time vs Tensor Size

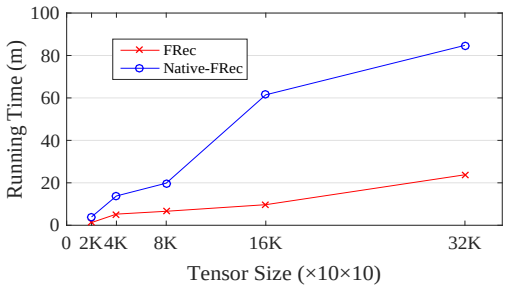


Fig. 5 Running Time over Different Tensor Sizes

Fig. 5 shows the result of the running time comparison between FRec and Native-FRec over the 6 synthetic datasets where R, L and M are set to 10, 10 and 2, respectively. We can see that the running time of FRec almost linearly grows from 1.27 minutes to 23.78 minutes as the tensor size increases from $2\text{K} \times 10 \times 10$ to $32\text{K} \times 10 \times 10$, while the running time of Native-FRec sharply grows from 3.92 minutes to 84.76 minutes. A larger tensor size will result in a bigger time cost of the tensor decomposition in HCF-TRM. At the same time, we also note that the running time of Native-FRec is significantly longer than that of FRec at scales equal to or larger than $2\text{K} \times 10 \times 10$. This efficiency gain of FRec is mainly due to the PDA algorithm which takes a divide-and-conquer strategy to fulfill the HCF-TRM.

6.4.2 Running Time vs M

We investigate how the running time changes with different M on the synthetic dataset SYN1, where R and L are both set to 10. To simulate M types of nodes, we randomly divide the nodes of SYN1 into M groups with an equal size. Fig. 6

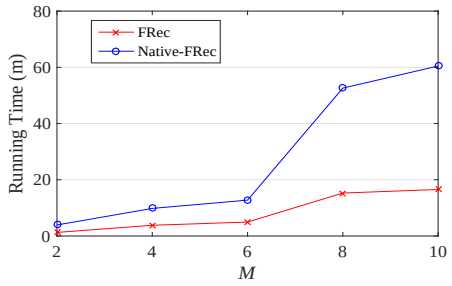


Fig. 6 Running Time over Different M

shows the running time of FRec and Native-FRec over a range of M from 2 to 10. We can see that as M increases, the running time of FRec and Native-FRec both grows in a similar trend. A larger M just will make IFT (the first part of FRec) more time consuming while with little impact on HCF-TRM (the second part of FRec).

6.4.3 Speedup Ratio vs Number of Parallel Threads

We check the speedup ratio of FRec and Native-FRec on the largest synthetic dataset SYN5, where R, L and M are set to 10, 10 and 2.

Fig. 7 shows the speedup ratio of FRec and Native-FRec with respect to the number of threads. We can see that the speeds of FRec and Native-FRec both accelerate as more threads are generated for optimization. However, the speedup ratio of FRec is a lot larger than that of Native-FRec at each number of threads. As shown in Fig. 7, the speedup ratio of FRec grows almost linearly from 5% to 26% as the number of threads increases from 2 to 20, while the speedup ratio of Native-FRec is at most about 5%.

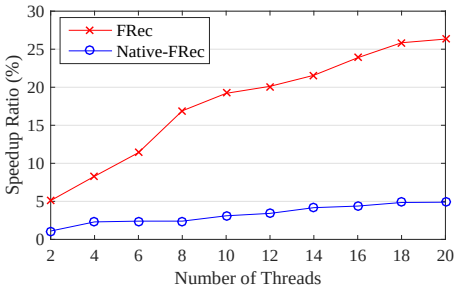
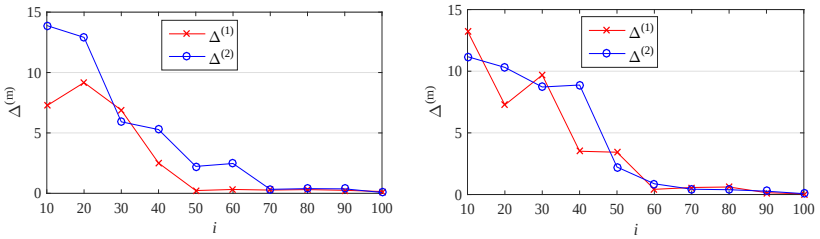


Fig. 7 Speedup Ratio Comparison

6.5 Convergency of IFT Learning

At last, we investigate the convergency of Algorithm 2 *LeraningIFT* using the largest synthetic dataset SYN5 which has two types of nodes (i.e., $M = 2$). Fig. 8(a) and Fig. 8(b) show that how $\Delta^{(m)} = \|\boldsymbol{Q}_i^{(m)} - \boldsymbol{Q}_{i-1}^{(m)}\|$ $(m = 1, 2)$ changes with respect to the iteration number i under the learning orders of $(\boldsymbol{Q}^{(1)}, \boldsymbol{Q}^{(2)})$ and the learning order of $(\boldsymbol{Q}^{(2)}, \boldsymbol{Q}^{(1)})$, respectively. As we can see from Fig. 8, $\Delta^{(1)}$ and $\Delta^{(2)}$ gradually decline to about zero after 30 and 50 iterations, respectively, which indicates that both $\boldsymbol{Q}^{(1)}$ and $\boldsymbol{Q}^{(2)}$ eventually converge to a fixed point regardless of the learning order of them.



(a) (b)

Fig. 8 Convergency of *LearningIFT*. (a) Learning Order of $(\boldsymbol{Q}^{(1)}, \boldsymbol{Q}^{(2)})$. (b) Learning Order of $(\boldsymbol{Q}^{(2)}, \boldsymbol{Q}^{(1)})$.

7 Related Work

The work of this paper is related to three domains, graph embedding, low-rank approximation of tensor, and heterogeneous information network analysis.

7.1 Graph Embedding

The purpose of graph embedding is to learn a low dimensional matrix as a latent feature representation with an optimization objective of best preserving the simi-

larity between original data points [32, 5]. He *et al.* [11] propose a subspace learning algorithm called Neighbourhood Preserving Embedding, which aims at preserving the local neighborhood structure on the data manifold. In contrast, Shaw *et al.* [21] propose a structure preserving embedding algorithm which can embed graphs to a low-dimensional Euclidean space and preserve the global topological properties of an input graph. Ahmed *et al.* [2] propose a graph factorization algorithm based on stochastic gradient descent for the low-dimensional. Perozzi *et al.* propose a latent representation learning algorithm called DeepWalk which realizes a social network embedding via a truncated random walk. Cao *et al.* [6] propose a model called GraRep which can learn low dimensional representations of nodes appearing in a graph with global structural information of the graph preserved. Wang *et al.* [28] propose a Structural Deep Network Embedding (SDNE) method for capturing non-linear network structure with preserving global and local structure. Ou *et al.* [19] develop a High-Order Proximity preserved Embedding (HOPE) algorithm, which is scalable to preserve high-order proximities of graphs and capable of capturing asymmetric transitivity. The aforementioned methods, however, are motivated by homogeneous settings and unable to serve dynamic heterogeneous information networks.

Recently, several latent representation learning methods for heterogeneous situations have been proposed. Yuan *et al.* [35] present a deep learning based framework for latent feature learning in social media networks. Tang *et al.* [27] propose an heterogeneous information network embedding algorithm, called LINE, which can preserve both 1-step and 2-step relationships between objects. Chang *et al.* [7] propose an embedding algorithm, called HNE, based on deep neural networks, which is able to map different heterogeneous objects into a unified latent space so that objects from different spaces can be directly compared. In contrast to the IFT proposed in this paper, however, these methods focus only on static heterogeneous networks, and are not invertible, which makes them unable to serve our goal.

7.2 Low-rank Approximation of Tensor

A branch of methods for the low-rank approximation of tensor have been proposed, which roughly fall into two classes. The methods of the first class are mainly based on the high order singular value decomposition (HOSVD). For example, Chen *et al.* [8] propose a low-rank orthogonal approximations of tensors with a theoretical guarantee of the existence of an optimal approximation. Koch *et al.* [13] propose an algorithm for low Tucker rank approximation which works in an incremental fashion. Lubich *et al.* [18] propose a dynamic low-rank approximation algorithm for time-dependent tensors. Anandkumar *et al.* [3] propose a robust tensor power method based on the perturbation theorem for the singular vectors.The second class methods are often based on vector outer production combined with norm constraints for optimization. For example, Liu *et al.* [17] propose an algorithm to reconstruct a tensor with its trace norm minimized. Jia *et al.* [12] propose a low-rank tensor completion method for action classification, as well as image recovery. Yu *et al.* [34] propose a low-rank tensor learning algorithm which can approximate a sparse tensor with its spectral norm minimized. Goldfarb *et al.* [9] propose a series of algorithms for robust low-rank tensor recovery in a convex optimization framework. Sun *et al.* [23] propose a general framework, incremental tensor analysis

(ITA), which efficiently computes a compact summary for high-order and high-dimensional data, and also reveals the hidden correlations. However, the above methods mainly do not adapt to the data from multiple heterogeneous sources and do not optimize the tensor decomposition by fusing heterogenous constraints, which makes them unable serve our heterogeneous setting in this paper.

Zheng et al. [29,37] recently propose a Context-Aware Tensor Decomposition (CATD) model, which is most related to our HCF-TRM. To achieve a higher accuracy of filling in the missing entries, CATD utilizes additional heterogeneous data as context constraints to optimize the latent feature matrices of the tensor. To adapt to uncertain dynamic heterogeneous information networks, our model extends CATD from two aspects. First, CATD focuses on a homogeneous tensor in essence, since the tensor cells store values of the measurements of the same type. On the contrary, the tensor that HCF-TRM handles is heterogeneous, since the tensor cells store values of the measurements of various types. Second, CATD can not work without additional data available, while HCF-TRM can still impose the heterogeneous constraints on the latent factor matrices without the requirement for additional data.

7.3 Heterogeneous Information Network Analysis

Heterogeneous information network analysis has become a popular topic which is attracting increasing interest from researchers. The existing researches mainly concern link prediction [33,24], clustering [26,38], classification [15], relevance search [22], and similarity search [31], etc. Yang *et al.* [33] propose a probabilistic method, called Multi-Relational Influence Propagation (MRIP), for predicting links in a sparse heterogeneous information network. Sun *et al.* [24] extend the link prediction problem to the relationship prediction problem, and propose a meta-path based approach to time prediction for a certain relationship in a heterogeneous information network. Sun *et al.* [26] also propose an algorithm called NetClus, which utilizes links across heterogeneous objects to generate high-quality net-clusters. Zhou *et al.* [38] present a social influence based clustering algorithm called SI-CLUSTER for heterogeneous information networks. Kong *et al.* [15] proposed a meta path based collective classification algorithm called HCC to effectively assign labels to a group of instances that are interconnected through different meta-paths. She *et al.* [22] presented a model called HeteSim for relevance search in HINs. Xiong *et al.* [31] propose a path-based similarity join (PS-join) method to return the top k similar pairs of objects based on any user specified join path in a heterogeneous information network.

The existing researches, however, often focus on static heterogeneous information networks without uncertainty considered. To our best knowledge, the problem of reducing uncertainty of dynamic heterogeneous information networks has not been studied yet.

8 Conclusion

In this paper, we propose a novel approach called FRec for the problem of reducing uncertainty of dynamic HINs. To address the challenge of the heterogeneity of

features, we propose an Invertible Fusing Transformation (IFT) as the first part of FRec. IFT realizes a bidirectional transformation between the unified latent feature representations and the raw features of the nodes in a heterogeneous information network. To address the challenge of the heterogeneity of constraints and the challenge of dynamic uncertainty, we propose a Heterogeneous Constraints Fusion based Tensor Reconstruction Model (HCF-TRM) as the second part of FRec. HCF-TRM denoises the snapshots of a dynamic HIN and recovers the missing values by fusing the spatial smoothness constraint and the temporal smoothness constraint into the tensor decomposition. The extensive experiments conducted on real datasets and synthetic datasets verify the effectiveness and scalability of FRec.

Acknowledgment

This work is supported by National Science Foundation of China through Grant 61173099, and in part by NSF through grants CNS-1115234 and OISE-1129076.

References

1. Achlioptas, D., Mcsherry, F.: Fast computation of low-rank matrix approximations. Journal of the ACM (JACM) **54**(2), 9 (2007)
2. Ahmed, A., Shervashidze, N., Narayanamurthy, S., Josifovski, V., Smo: Distributed large-scale natural graph factorization. In: Proceedings of the 22nd International Conference on World Wide Web, WWW '13, pp. 37–48. International World Wide Web Conferences Steering Committee (2013)
3. Anandkumar, A., Ge, R., Hsu, D., Kakade, S.M., Telgarsky, M.: Tensor decompositions for learning latent variable models. The Journal of Machine Learning Research **15**(1), 2773–283 (2014)
4. Bao, J., Zheng, Y., Mokbel, M.F.: Location-based and preference-aware recommendation using sparse geo-social networking data. In: Proceedings of the 20th International Conference on Advances in Geographic Information Systems, SIGSPATIAL '12, pp. 199–208. ACM, New York, NY, USA (2012). DOI 10.1145/2424321.2424348. URL http://doi.acm.org/10.1145/2424321.2424348
5. Brand, M.: Continuous nonlinear dimensionality reduction by kernel eigenmaps. In: IJCAI, pp. 547–554 (2003)
6. Cao, S., Lu, W., Xu, Q.: Grarep: Learning graph representations with global structural information. In: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, CIKM '15, pp. 891–900. ACM (2015)
7. Chang, S., Han, W., Tang, J., Qi, G.J., Aggarwal, C.C., Huang, T.S.: Heterogeneous network embedding via deep architectures. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'15, pp. 119–128. ACM (2015)
8. Chen, J., Saad, Y.: On the tensor svd and the optimal low rank orthogonal approximation of tensors. SIAM Journal on Matrix Analysis and Applications **30**(4), 1709–1734 (2009)
9. Goldfarb, D., Qin, Z.: Robust low-rank tensor recovery: Models and algorithms. SIAM Journal on Matrix Analysis and Applications **35**(1), 225–253 (2014)
10. Golub, G.H., Van Loan, C.F.: Matrix computations, vol. 3. JHU Press (2013)
11. He, X., Cai, D., Yan, S., Zhang, H.J.: Neighborhood preserving embedding. In: Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on, vol. 2, pp. 1208–1213. IEEE (2005)
12. Jia, C., Zhong, G., Fu, Y.: Low-rank tensor learning with discriminant analysis for action classification and image recovery. In: Twenty-Eighth AAAI Conference on Artificial Intelligence (2014)

13. Koch, O., Lubich, C.: Dynamical tensor approximation. SIAM Journal on Matrix Analysis and Applications **31**(5), 2360–2375 (2010)
14. Kolda, T.G., Bader, B.W.: Tensor decompositions and applications. SIAM review **51**(3), 455–500 (2009)
15. Kong, X., Yu, P.S., Ding, Y., Wild, D.J.: Meta path-based collective classification in heterogeneous information networks. In: Proceedings of the 21st ACM International Conference on Information and Knowledge Management, CIKM '12, pp. 1567–1571. ACM (2012)
16. Kruskal, J.B.: Rank, decomposition, and uniqueness for 3-way and n-way arrays. Multiway data analysis **33**, 7–18 (1989)
17. Liu, J., Musialski, P., Wonka, P., Y, J.: Tensor completion for estimating missing values in visual data. Pattern Analysis and Machine Intelligence, IEEE Transactions on **35**(1), 208–220 (2013)
18. Lubich, C., Rohwedder, T., Schneider, R., Vandereycken, B.: Dynamical approximation by hierarchical tucker and tensor-train tensors. SIAM Journal on Matrix Analysis and Applications **34**(2), 470–494 (2013)
19. Ou, M., Cui, P., Pei, J., Zhang, Z., Zhu, W.: Asymmetric transitivity preserving graph embedding. In: Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, pp. 1105–1114. ACM, New York, NY, USA (2016). DOI 10.1145/2939672.2939751. URL http://doi.acm.org/10.1145/2939672.2939751
20. Phan, A.H., Cichocki, A.: Parafac algorithms for large-scale problems. Neurocomputing **74**(11), 1970–1984 (2011)
21. Shaw, B., Jebara, T.: Structure preserving embedding. In: Proceedings of the 26th Annual International Conference on Machine Learning, pp. 937–944. ACM (2009)
22. Shi, C., Kong, X., Yu, P.S., Xie, S., Wu, B.: Relevance search in heterogeneous networks. In: Proceedings of the 15th International Conference on Extending Database Technology, EDBT '12, pp. 180–191. ACM (2012)
23. Sun, J., Tao, D., Papadimitriou, S., Yu, P.S., Faloutsos, C.: Incremental tensor analysis: Theory and applications. ACM Trans. Knowl. Discov. Data **2**(3), 11:1–11:37 (2008)
24. Sun, Y., Han, J., Aggarwal, C.C., Chawla, N.V.: When will it happen?: Relationship prediction in heterogeneous information networks. In: Proceedings of the Fifth ACM International Conference on Web Search and Data Mining, WSDM '12, pp. 663–672. ACM (2012)
25. Sun, Y., Norick, B., Han, J., Yan, X., Yu, P.S., Yu, X.: Integrating meta-path selection with user-guided object clustering in heterogeneous information networks. In: Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD'12, pp. 1348–1356. ACM (2012)
26. Sun, Y., Yu, Y., Han, J.: Ranking-based clustering of heterogeneous information networks with star network schema. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'09, pp. 797–806. ACM (2009)
27. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line: Large-scale information network embedding. In: Proceedings of the 24th International Conference on World Wide Web, WWW'15, pp. 1067–1077. International World Wide Web Conferences Steering Committee (2015)
28. Wang, D., Cui, P., Zhu, W.: Structural deep network embedding. In: Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, pp. 1225–1234. ACM, New York, NY, USA (2016). DOI 10.1145/2939672.2939753. URL http://doi.acm.org/10.1145/2939672.2939753
29. Wang, Y., Zheng, Y., Xue, Y.: Travel time estimation of a path using sparse trajectories. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD'14, pp. 25–34. ACM (2014)
30. Wen, Z., Yin, W.: A feasible method for optimization with orthogonality constraints. Mathematical Programming **142**(1-2), 397–434 (2013)
31. Xiong, Y., Zhu, Y., Yu, P.: Top-k similarity join in heterogeneous information networks. Knowledge and Data Engineering, IEEE Transactions on **27**(6), 1710–1723 (2015)
32. Yan, S., Xu, D., Zhang, B., Zhang, H.J., Yang, Q., Lin, S.: Graph embedding and extensions: a general framework for dimensionality reduction. Pattern Analysis and Machine Intelligence, IEEE Transactions on **29**(1), 40–51 (2007)
33. Yang, Y., Chawla, N., Sun, Y., Hani, J.: Predicting links in multi-relational and heterogeneous networks. In: Data Mining (ICDM), 2012 IEEE 12th International Conference on, pp. 755–764. IEEE (2012)

34. Yu, Y., Cheng, H., Zhang, X.: Approximate low-rank tensor learning. In: 7th NIPS Workshop on Optimization for Machine Learning (2014)
35. Yuan, Z., Sang, J., Liu, Y., Xu, C.: Latent feature learning in social media network. In: Proceedings of the 21st ACM International Conference on Multimedia, MM'13, pp. 253–262. ACM (2013)
36. Zheng, Y., Liu, F., Hsieh, H.P.: U-air: When urban air quality inference meets big data. In: Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD'13, pp. 1436–1444. ACM (2013)
37. Zheng, Y., Liu, T., Wang, Y., Zhu, Y., Liu, Y., Chang, E.: Diagnosing new york city's noises with ubiquitous data. In: Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing, pp. 715–725. ACM (2014)
38. Zhou, Y., Liu, L.: Social influence based clustering of heterogeneous information networks. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'13, pp. 338–346. ACM (2013)