Spatiotemporal Fracture Data Inference in Sparse Urban CrowdSensing

En Wang[1,2], Mijia Zhang[1,2], Yuanbo Xu[1,2,*], Haoyi Xiong[3], Yongjian Yang[1,2]
[1]*College of Computer Science and Technology*, *Jilin University*, China
[2]*Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education*, *Jilin University*, China
[3]*Big Data Laboratory*, *Baidu Research*, CA, USA
Email: wangen@jlu.edu.cn; zhangmj19@mails.jlu.edu.cn; yuanbox@jlu.edu.cn; xionghaoyi@baidu.com; yyj@jlu.edu.cn

***Abstract*—While Mobile CrowdSensing (MCS) has become a popular paradigm that recruits mobile users to carry out various sensing tasks collaboratively, the performance of MCS is frequently degraded due to the limited spatiotemporal coverage in data collection. A possible way here is to incorporate sparse MCS with data inference, where unsensed data could be completed through prediction. However, the spatiotemporal data inference is usually "fractured" with poor performance, because of following challenges: 1) the sparsity of the sensed data, 2) the unpredictability of a spatiotemporal fracture and 3) the complex spatiotemporal relations. To resolve such fracture data issues, we elaborate a data generative model for achieving spatiotemporal fracture data inference in sparse MCS. Specifically, an algorithm named Generative High-Fidelity Matrix Completion (GHFMC) is proposed through combining traditional Deep Matrix Factorization (DMF) and Generative Adversarial Networks (GAN) for generating spatiotemporal fracture data. Along this line, GHFMC learns to extract the features of spatiotemporal data and further efficiently complete and predict the unsensed data by using Binary Cross Entropy (BCE) loss. Finally, we conduct experiments on three popular datasets. The experimental results show that our approach performs higher than the state-of-the-art (SOTA) baselines in both data inference accuracy and fidelity.**

***Index Terms*—Mobile CrowdSensing, data inference, spatiotemporal fracture data, Generative Adversarial Network**

I. Introduction

With the rapid developments and applications of 5th Generation Mobile Communication Technology (5G) [1], [2] and the increasing popularity of portable and smart Internet of Things (IoT) [1] devices. Mobile CrowdSensing (MCS) [3], [4], which recruits users with carrying out mobile smart devices to collect various types of data [5]–[7], has played an extremely powerful role in the smart city. MCS can help the manager to not only monitor the current and historical status, but also predict the future situation. Due to the limited cost of MCS, sparse MCS [8], which senses only limited subareas, came into being and played an extremely significant role in MCS. Sparse MCS relies on the data inference techniques for improving service quality [9] in the context of sparse data. It significantly lower the sensing cost, but also brings a series of problems such as the spatiotemporal fracture data inference problem.

This work was supported in part by the National Key R&D Program of China under Grant 2021ZD0112501 and Grant 2021ZD0112502, in part by the National Natural Science Foundations of China under Grant 61772230, Grant 61972450, Grant 62102161 and Grant 62072209, and in part by the CCF-Baidu Open Fund under Grant 2021PP15002000.

* Yuanbo Xu is the corresponding author.

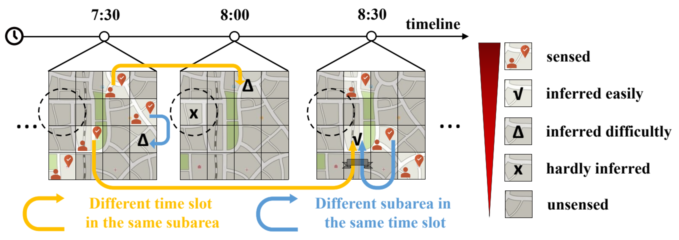


Fig. 1: Examples to describe the spatiotemporal fracture data inference problem of sparse urban crowdsensing.

The problem description for *spatiotemporal fracture data inference in sparse urban crowdsensing* is shown in Fig. 1. Given a whole city map, we artificially divide the map into 4×4 subareas and sense the data from these subareas. At each time slot, due to the uncertainty of mobile users' location and the sensing budget constraint, we can hardly get all m ($m = 16$) subareas' sensed data. Specially, it may not achieve any data of each subareas at some time slots (e.g. the time slot $8:00$ in Fig. 1). By collecting the sensed data after n time slots, we can obtain the spatiotemporal matrix with the size of $m \times n$. It means that the spatiotemporal matrix is sparse and even has fracture rows or columns. For the general sparse case, the previous research has been able to complete with high accuracy. But for the spatiotemporal fracture data, which usually happens in real situations, the previous methods will not be able to deal with. The fracture of the data may cause some vital information to be lost, which even results emergency. For example, there is an air pollution monitoring system. If the system fails in some time slots, data collection will not be carried out. So the other work that needs to use these air pollution data will be affected in these time slots. This phenomenon is called *temporal fracture*. Similarly, there is also another phenomenon named *spatio fracture*. For example, due to the limitation of data acquisition equipment, some blind areas will not be able to collect air pollution data. Therefore, both temporal and spatial fracture data are also needed to be inferred correctly. However, the inference of spatiotemporal fracture data under the background of sparse MCS faces the following three challenges: 1) lack of enough context information, 2) the irregular spatiotemporal fracture data and 3) the complex spatiotemporal relationships.

Actually, how to infer spatiotemporal fracture data in sparse urban crowdsensing has not been perfectly solved. Some existing methods (e.g., IGMC [10], DMF [11], etc.) are strongly dependent on the distribution of sensed data. They usually take the sensed data as the convergence target. However, for fracture data, we cannot provide any context element as the convergence target, which weakens the performance of existing methods by producing a series of meaningless chaos, meaningless data. Therefore, how to *implement data inference without enough context information* is the first challenge.

In a practical application scenario, the occurrence of spatiotemporal fracture is usually random and unpredictable. Moreover, the scale of spatiotemporal fracture is also uncertain. (We have no knowledge of when the temporal data fracture will appear and how long it will last.) Similarly, it is also difficult to predict where spatial fracture will appear and what scale the spatial fracture will be. Therefore, how to *deal with the irregular spatiotemporal fracture phenomenon* is the second challenge.

In previous studies [10], [12], [13], existing methods are more inclined to obtain less error inference results. But in fact, sometimes we are more interested in the spatiotemporal distribution of data. For example, in a city traffic flow monitoring system, we may pay more attention to the distribution of vehicle's number in each area of the city and the change tendency over time, rather than the number of vehicles. In sparse MCS, due to the sparsity of data, it is extremely difficult to extract the high fidelity spatiotemporal features of data. Therefore, how to *generate inferential data from sparse sensed data that are more consistent with the real data distribution* is the third challenge.

To deal with the challenges above, we propose a complete model named Generative High-Fidelity Matrix Completion (GHFMC) to address the spatiotemporal fracture data inference problem in sparse urban crowdsensing. To solve the problem of insufficient context information, we combine the Conditional Generative Adversarial Nets (CGAN) [14] with the traditional Deep Matrix Factorization (DMF) [11]. Using generative algorithm instead of inferential algorithm can avoid the problem caused by the lack of convergence object. At the same time, we use the idea of DMF to make use of the low rank property of spatiotemporal matrix. In consideration of the irregular of the spatiotemporal fracture, we use the conditional vector estimation method which only faces the adjacent time slot or the adjacent position. Finally, we have higher requirements for the effect of sparse matrix completion and hope to achieve high fidelity matrix completion through generative algorithm. Although the accuracy of matrix completion is not significantly improved, high fidelity of matrix completion is more conducive to the generation of spatiotemporal fracture data. Compared with the traditional sparse matrix completion algorithm, RMSE and R-squared of this method are better than other methods. It shows that this method not only solves the problem of spatiotemporal fracture data, but also can restore fracture data with high accuracy and high fidelity, which thanks to the generative algorithm we designed.

Our work has the following contributions:

- We formalize the sparse urban crowdsensing problem, with the goal of recovering the spatiotemporal fracture data from the sparse sensed data.
- We propose a multi-step urban crowdsensing method named GHFMC, which aims to solve the problem of inferring the spatiotemporal fracture data from sparse sensed data and adapt to the irregular fracture.
- Compared with the traditional data inference methods in MCS, our approach can effectively extract the complex spatiotemporal relationship and make the characteristics of the generated data more closer to the ground truth data.
- We evaluate our approach on three different types of typical urban sensing tasks. The experimental results verify that our approach can improve the inference accuracy and fidelity effectively when applying to spatiotemporal fracture data in sparse MCS.

II. Related Work

A. Sparse Mobile CrowdSensing

Mobile CrowdSensing (MCS) technology utilized smart devices carried by mobile users to perform different series of urban crowdsensing tasks [4], [15]. An example of traffic speed measurement system [16] would reveal how powerful the MCS was. By collecting a large number of spatiotemporal data from numerous mobile users, the system mapped a large-scale urban environment. However, in most cases, the sensing cost is limited, and we can't recruit users to collect data without restraint. In the face of this situation, we will achieve similar sensing result by sensing only a small amount of specific data if we make full use of spatiotemporal relationships among different subareas. By utilizing the spatiotemporal relationships mined from sensed data, the value of the other unsensed areas can be infered approximately [17]. The past several years has witnessed a growing number of researchers have developed a series of sparse MCS-based urban sensing systems. Sparse MCS, an advanced data sensing paradigm, senses only limited subareas and infers all the other unsensed subareas. The example of urban noise monitoring system by Rana *et al.* [18] achieved a fine-grained urban noise map. Rana *et al.* applied the compressive sensing theory to the sparse MCS in order to infer all unsensed data. There are also many examples about sparse MCS in recent years. Liu *et al.* [19] and He *et al.* [20] proposed a sparse MCS-based urban air pollution and signal mapping systems. We can get some inspiration from the existing applications of sparse MCS.

B. Matrix Completion for Spatiotemporal Data

From the perspective of the mathematical model, sparse MCS can be equivalent to the completion problem of a sparse spatiotemporal matrix. In the real application scenarios, because the values of adjacent time slots or adjacent subareas are close, the rank of the spatiotemporal matrix should be small (low-rank). Based on this assumption, Matrix Factorization (MF) for data inference came into being, which was effective in processing spatiotemporal data with strong linear

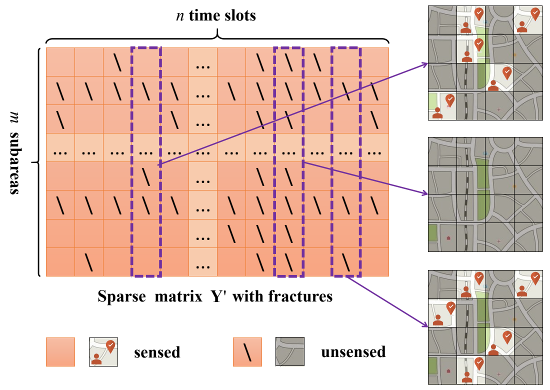


Fig. 2: The relationship between the physical model and the mathematical model.

TABLE I: Main notations

Symbol	Meaning
m, m_0	number of all subareas and fracture subareas
n, n_0	number of all time slots and fracture time slots
i, j	index of a subarea and a time slot
k, l	index of a column and a row of a sliced matrix
$\mathbf{Y}$, $\mathbf{Y}'$, $\hat{\mathbf{Y}}$	ground truth of the complete data, sparse data with fractures, estimated data
$\mathbf{C}$, $\mathbf{C}_\mathrm{O}$	labels of which subareas are sensed
$\mathbf{A}'$	non-sparse matrix with spatiotemporal fractures
$\hat{\mathbf{A}}$, $\hat{\mathbf{B}}$	temporal and spatial non-fracture matrix
$\hat{\mathbf{A}}_\mathrm{s}$, $\hat{\mathbf{B}}_\mathrm{s}$	sliced matrix of matrix $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$
$\mathbf{z}_\mathrm{O}$, $\widetilde{\mathbf{z}}_\mathrm{T}$, $\widetilde{\mathbf{z}}_\mathrm{S}$	input vector of the generator for original step, temporal generative step and spatial generative step
$\mathbf{z}_\mathrm{T}$, $\mathbf{z}_\mathrm{S}$	common vector of the generator for temporal and spatial generative step
$\mathbf{z}_\mathrm{cT}$, $\mathbf{z}_\mathrm{cS}$	conditional vector of the generator for temporal and spatial generative step

characteristics but takes large error in processing non-linear data. Fan and Cheng [11] took advantage of the non-linear neural network and proposed the Deep Matrix Factorization (DMF) which improved the limitations of MF. On the basis of DMF, Wang *et al.* [12] proposed an end-to-end model for sparse industrial sensing and prediction. It means that sparse MCS can not only realize the sensing of the current time slot, but also achieve high-precision future prediction. With the popularity of graph neural network (GNN), Zhang *et al.* [10] showed a new idea named Inductive Graph-based Matrix Completion (IGMC) about matrix completion by using GNN. The performance of IGMC is better than DMF for some datasets. What's more, Wang *et al.* [13] considered the influence of outlier data and proposed a new algorithm to address the problem of outlier data inference. Considering the spatiotemporal fracture, Xie *et al.* [21] propose a two-phase MC-based data recovery scheme, which exploits the inherent features of environmental data to recover the fracture data. In the field of sparse matrix completion, existing work usually ignored the fidelity of a matrix completion method. Generative algorithms, such as Generative Adversarial Network (GAN), are popular in recent years, and seem to achieve this effect.

III. System Model and Problem Formulation

A. System Model

As shown in Fig. 1, our research is based on a classical urban crowdsensing task that recruits mobile device users to collect data from limited areas for recovering spatiotemporal fracture data. In this subsection, we are going to build the mathematical model of the system we propose. Fig. 2 shows the relationship between the physical model and the mathematical model and Table I includes the main notations that are used in this paper.

Given a whole urban sensing map which is divided into m subareas artificially, we aim at obtaining all values from the m subareas with the condition of only $\tilde{m}$ sensed subareas ($\tilde{m} << m$) at each time slot. In order to represent each subarea easily, we code these subareas by one-hot encoding. We use an unit vector $\mathbf{e}^{(i)}_{m\times 1}$ to denote the i-th subarea and only the i-th element of the vector $\mathbf{e}^{(i)}$ equals 1. So as to mark the sensed subareas at the j-th time slot, we introduce the logical value c_{ij} to denote whether the i-th subarea is sensed or not. $c_{ij} = 1$ means the i-th subarea is sensed at the j-th time slot and the sensed value is named y'_{ij}. Conversely, if there is no sensed data from the i-th subarea at the j-th time slot, we will set $c_{ij} = 0$ and the value of y'_{ij} will be evaluated as a meaningless value (e.g., $y'_{ij} = \infty$). Therefore, we can get a logical vector $\mathbf{c}^{(j)}_{m\times 1} = \sum_{i=1}^{m} c_{ij}\mathbf{e}^{(i)}$, which marks the sensed subareas, and a sparse vector $\mathbf{y}'^{(j)}_{m\times 1} = \sum_{i=1}^{m} y'_{ij}\mathbf{e}^{(i)}$, which denotes the sensed values of the j-th time slot. Similarly, we use $\mathbf{y}^{(j)}_{m\times 1} = [y_{1j}, y_{2j}, \ldots, y_{mj}]^\intercal$ to denote the ground truth vector of the j-th time slot.

The vectors that we present above can be combined to matrices after n time slots. $\mathbf{C}_{m\times n} = [\mathbf{c}^{(1)}, \mathbf{c}^{(2)}, \ldots, \mathbf{c}^{(n)}]$ denotes the situation of sensed subareas at each time slot; $\mathbf{Y}'_{m\times n} = [\mathbf{y}'^{(1)}, \mathbf{y}'^{(2)}, \ldots, \mathbf{y}'^{(n)}]$ denotes the sparse sensed data of n time slots; $\mathbf{Y}_{m\times n} = [\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \ldots, \mathbf{y}^{(n)}]$ denotes the complete ground truth. Then the mathematical relation of these three matrices is

$$\mathbf{Y}' = \mathbf{Y} \circ \mathbf{C}, \tag{1}$$

where $\circ$ denotes the Hadamard product (element product) of two matrices. Then, the main task is building a data inference function $f(\cdot)$ so as to infer all the unsensed data from the sparse matrix $\mathbf{Y}'$. The estimated matrix $\hat{\mathbf{Y}}$ can be calculated by the following equation:

$$\hat{\mathbf{Y}} = f(\mathbf{Y}'). \tag{2}$$

B. Problem Formulation

Problem [Spatiotemporal Fracture Data Inference via Sparse MCS]: Given a sparse matrix $\mathbf{Y}'_{m\times n}$ with spatiotemporal fractures, we aim to achieve the following two targets:

- Find a function $f(\cdot)$ to infer all unsensed value of the sparse matrix $\mathbf{Y}'_{m\times n}$ and make the complete matirx $\hat{\mathbf{Y}} = f(\mathbf{Y}')$ be established.
- Ensure that the spatiotemporal fracture data can be recovered correctly with a high accuracy and a high fidelity.

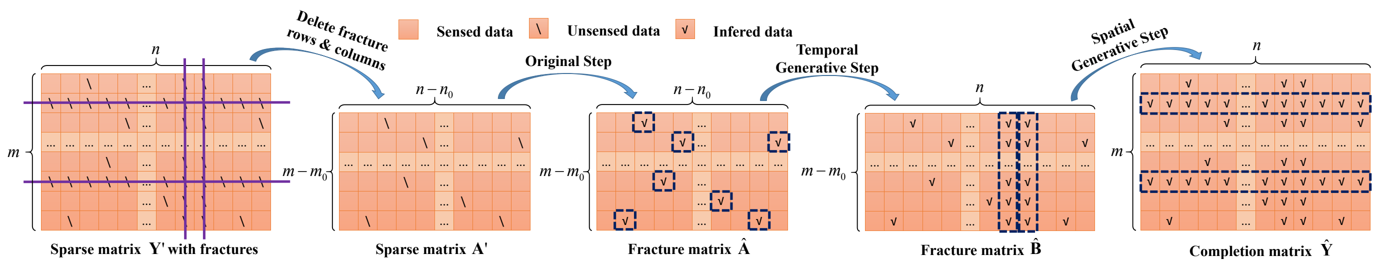


Fig. 3: Framework overview.

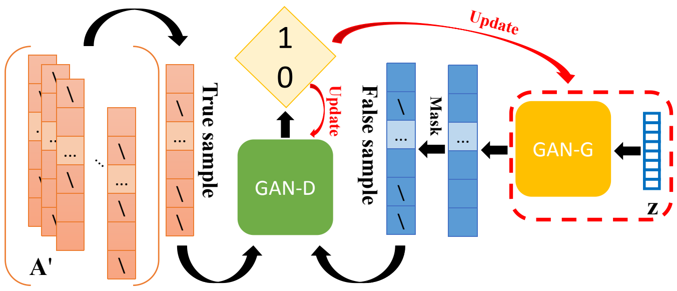


Fig. 4: Structure of GHFMC (Original Step).

In this process, the following objective value obj should be kept as small as possible:

$$obj = \lambda_1 \varepsilon(f(\mathbf{Y}'), \mathbf{Y}) - \lambda_2 R^2(f(\mathbf{Y}'), \mathbf{Y}), \tag{3}$$

where $\varepsilon(\cdot,\cdot)$ and $R^2(\cdot,\cdot)$ are used to measure the accuracy and fidelity of data inference, respectively. In order to minimize the objective value obj, we should first minimize the error of the inference of non-fracture data and then focus on the subsequent operations for fracture data. It will be a multi-step process in the following approaches.

IV. Sparse Data Inference of Generative High-Fidelity Matrix Completion

Aiming at the problem of spatiotemporal fracture data inference in sparse urban crowdsensing, we design a framework like Fig. 3 and our idea is shown as follows: Firstly, the rows and columns corresponding to the fracture data are deleted and we can get a sparse matrix with a smaller shape. Then, the sparse matrix completion algorithm is used to infer the unsensed data. Finally, the complete matrix we get is used to infer the fracture data. There are essential differences between sparse data and fracture data, and temporal fractures and spatial fractures also need to be handled separately, so different methods should be used for processing. In order to make the final generated fracture data inference results have higher fidelity, it is necessary to ensure that the inference results of sparse data have high similarity with the ground truth. Previous researches on sparse matrix completion mainly focus on high accuracy matrix completion rather than high fidelity. Therefore, we propose a high accuracy and fidelity data inference algorithm for sparse data in this paper.

Assuming that there are m_0 temporal fracture rows and n_0 spatial fracture columns. We first delete these rows or columns and then try to complete a pure sparse matrix $\mathbf{A}'$ with the shape of $(m - m_0) \times (n - n_0)$. Matrix $\mathbf{A}'$ can be divided into columns: $\mathbf{A}' = [\mathbf{a}'^{(1)}, \mathbf{a}'^{(2)}, \cdots, \mathbf{a}'^{(n-n_0)}]$. The corresponding logical matrix becomes to $\mathbf{C}_\text{O} = [\mathbf{c}_\text{O}^{(1)}, \mathbf{c}_\text{O}^{(2)}, \cdots, \mathbf{c}_\text{O}^{(n-n_0)}]$.

Unless the traditional matrix completion, we hope to make the data inference result more similar to the ground truth on the premise of ensuring high-precision matrix completion. As shown in Fig. 4, inspired by GAN [22] and Deep Matrix Factorization (DMF) [11], we design a novel method of sparse matrix completion. The detailed algorithm of the original step is introduced in the following several paragraphs. In order to make it more easy for readers to understand, we give a pseudo code flow table of original step in Alg. 1.

The traditional GAN includes a generator and a discriminator. We tried to generate a group of fake data, which can confuse the most advanced true and false discriminator. Therefore, we need to constantly input labeled true and false data to the discriminator in order to get the most advanced discriminator. With the continuous improvement of the discriminator, the data generated by the generator is forced to be more realistic. The generator and the discriminator improve each other in repeated competition, so we can get the most advanced discriminator and generator by this type of training mode.

It seems that the traditional GAN can not directly solve the problem of sparse matrix completion that we want to generate the unsensed data by imitating sensed data. When DMF is used for matrix completion, we use the low rank property of spatiotemporal matrix. Supposing that the rank of a complete spatiotemporal matrix is r, the input of the neural network is set to an r-dimensional vector $\mathbf{z}_{r\times 1}$. In the process of training DMF, $\mathbf{z}$ is regarded as a network parameter to be updated. It can be proved that $\mathbf{z}$ is the low dimensional embedding representation of the complete vector and contains all the information of the complete vector. The Embedding vectors are more suitable for complex mathematical operations than complete vectors. Similarly, based on the same assumption, we also set the input of the generator to an r-dimensional vector. In this step, we name the generator $G_\text{O}(\cdot)$ and the discriminator $D_\text{O}(\cdot)$. The j-th input vector of the generator $G_\text{O}(\cdot)$ is named $\mathbf{z}_\text{O}^{(j)}$ and $\mathbf{Z}_\text{O} = [\mathbf{z}_\text{O}^{(1)}, \mathbf{z}_\text{O}^{(2)}, \cdots, \mathbf{z}_\text{O}^{(n-n_0)}]$.

In addition, the sensed data is sparse, but the output of the generator $G_\text{O}(\cdot)$ is complete. Therefore, we cannot input the complete data into the discriminator without masking. When training the discriminator, we want the discriminator to have the ability to judge the true and false data, so the j-th output of the discriminator corresponding to the positive label sample is $D_{\mathrm{O}}(\mathbf{a}'^{(j)})$ and the j-th output of the discriminator corresponding to the negative label sample is $D_{\mathrm{O}}(G_{\mathrm{O}}(\mathbf{z}_{\mathrm{O}}^{(j)}) \circ \mathbf{c}_{\mathrm{O}}^{(j)})$. In the case of training the generator, the treatment is different. We hope that the data generated by the generator can confuse the discriminator, so we make the output of the generator correspond to the positive label sample.

Finally, it is necessary to introduce the loss function of our neural network. Our direct observation results only come from the discriminator. Therefore, we choose the Binary Cross Entropy (BCE) loss function. The loss functions of the discriminator and the generator are

$$L_{\mathrm{OD}}^{(j)} = -\log(D_{\mathrm{O}}(\mathbf{a}'^{(j)})) - \log(1 - D_{\mathrm{O}}(G_{\mathrm{O}}(\mathbf{z}_{\mathrm{O}}^{(j)}) \circ \mathbf{c}_{\mathrm{O}}^{(j)})), \quad (4)$$

$$L_{\mathrm{OG}}^{(j)} = -\log(D_{\mathrm{O}}(G_{\mathrm{O}}(\mathbf{z}_{\mathrm{O}}^{(j)}) \circ \mathbf{c}_{\mathrm{O}}^{(j)})). \quad (5)$$

By training the generator and discriminator repeatedly, the output of the generator will converge and the discriminator will not be able to distinguish between true and false data. At this point, the output vector of the generator is $G_{\mathrm{O}}(\mathbf{z}_{\mathrm{O}}^{(j)})$. Considering that some sensed data are known, we use these sensed data instead of inferential data to represent the estimate:

$$\hat{\mathbf{a}}^{(j)} = G_{\mathrm{O}}(\mathbf{z}_{\mathrm{O}}^{(j)}) \circ (\mathbf{1} - \mathbf{c}_{\mathrm{O}}^{(j)}) + \mathbf{a}'^{(j)} \circ \mathbf{c}_{\mathrm{O}}^{(j)}. \quad (6)$$

Finally, we can easily combine all the $(n - n_0)$ vectors to get a complete matrix which as follows:

$$\hat{\mathbf{A}} = [\hat{\mathbf{a}}^{(1)}, \hat{\mathbf{a}}^{(2)}, \cdots, \hat{\mathbf{a}}^{(n-n_0)}]. \quad (7)$$

V. Fracture Data Inference of Generative High-Fidelity Matrix Completion

Through the method of section IV, we get a complete matrix $\hat{\mathbf{A}}$ with high precision and high fidelity. In this section, we will deal with the matrix $\hat{\mathbf{A}}$ further and propose a spatiotemporal fracture data generation method based on conditional GAN (CGAN). According to the idea in Fig. 3, we deal with the temporal fracture and spatial fracture step by step.

A. Temporal Generative Step

In this step, we use a non-sparse spatiotemporal matrix $\hat{\mathbf{A}} = [\hat{\mathbf{a}}^{(1)}, \hat{\mathbf{a}}^{(2)}, \cdots, \hat{\mathbf{a}}^{(n-n_0)}]$ to calculate the n_0 deleted column vectors $\hat{\mathbf{a}}_{\mathrm{TG}}^{(1)}, \cdots, \hat{\mathbf{a}}_{\mathrm{TG}}^{(n_0)}$. Assuming that the fracture column vectors appear between $\hat{\mathbf{a}}^{(k_0)}$ and $\hat{\mathbf{a}}^{(k_0+1)}$, then the temporal non-fracture matrix $\hat{\mathbf{B}}$ can be expressed as:

$$\hat{\mathbf{B}} = [\hat{\mathbf{a}}^{(1)}, \cdots, \hat{\mathbf{a}}^{(k_0)}, \hat{\mathbf{a}}_{\mathrm{TG}}^{(1)}, \cdots, \hat{\mathbf{a}}_{\mathrm{TG}}^{(n_0)}, \hat{\mathbf{a}}^{(k_0+1)}, \cdots, \hat{\mathbf{a}}^{(n-n_0)}]. \quad (8)$$

We must assume that there are similar spatiotemporal distribution characteristics in a period of time slots before and after the temporal fracture data. In fact, we also verify that the real data set does have such characteristics. However, the time slot far away from the temporal fracture data is no longer similar to the fracture. Therefore, we can only select a limited number of columns as the input data of the training model.

Algorithm 1 The Original Step of Generative High-Fidelity Matrix Completion

```
Input:   the sparse matrix $\mathbf{A}' = [\mathbf{a}'^{(1)}, \mathbf{a}'^{(2)}, \cdots, \mathbf{a}'^{(n-n_0)}]$
         and the logical matrix $\mathbf{C}_{\mathrm{O}} = [\mathbf{c}_{\mathrm{O}}^{(1)}, \mathbf{c}_{\mathrm{O}}^{(2)}, \cdots, \mathbf{c}_{\mathrm{O}}^{(n-n_0)}]$
Output:  the complete matrix $\hat{\mathbf{A}}$
 1: Build the Generator and the Discriminator by using $G_{\mathrm{O}}(\cdot)$
    and $D_{\mathrm{O}}(\cdot)$, respectively;
 2: Random Init $\mathbf{Z}_{\mathrm{O}} = [\mathbf{z}_{\mathrm{O}}^{(1)}, \mathbf{z}_{\mathrm{O}}^{(2)}, \cdots, \mathbf{z}_{\mathrm{O}}^{(n-n_0)}]$ and the NN
    structure parameters of $G_{\mathrm{O}}(\cdot)$ and $D_{\mathrm{O}}(\cdot)$;
 3: $count := 0$;
 4: while not convergent and $count <$ MAX_ITER do
 5:     for $j$ is from 1 to $n - n_0$ do
 6:         Calculate $D_{\mathrm{O}}(G_{\mathrm{O}}(\mathbf{z}_{\mathrm{O}}^{(j)}) \circ \mathbf{c}_{\mathrm{O}}^{(j)})$ and $D_{\mathrm{O}}(\mathbf{a}'^{(j)})$;
 7:         Fix the NN structure parameters of $D_{\mathrm{O}}(\cdot)$ to make
            $L_{\mathrm{OD}}^{(j)}$ reduce;
 8:         Calculate $D_{\mathrm{O}}(G_{\mathrm{O}}(\mathbf{z}_{\mathrm{O}}^{(j)}) \circ \mathbf{c}_{\mathrm{O}}^{(j)})$ and $D_{\mathrm{O}}(\mathbf{a}'^{(j)})$ again;
 9:         Fix $\mathbf{z}_{\mathrm{O}}^{(j)}$ and the NN structure parameters of $G_{\mathrm{O}}(\cdot)$
            at the same time to make $L_{\mathrm{OG}}^{(j)}$ reduce;
10:     end for
11:     $count := count + 1$;
12: end while
13: return $\hat{\mathbf{A}} = G_{\mathrm{O}}(\mathbf{Z}_{\mathrm{O}}) \circ (\mathbf{1} - \mathbf{C}_{\mathrm{O}}) + \mathbf{A}' \circ \mathbf{C}_{\mathrm{O}}$.
```

Suppose that the column vectors we choose are $\hat{\mathbf{a}}^{(k_1)}, \cdots, \hat{\mathbf{a}}^{(k_0)}, \hat{\mathbf{a}}^{(k_0+1)}, \cdots, \hat{\mathbf{a}}^{(k_2)}$, then these column vectors will form a matrix $\hat{\mathbf{A}}_{\mathrm{s}} = [\hat{\mathbf{a}}^{(k_1)}, \cdots, \hat{\mathbf{a}}^{(k_0)}, \hat{\mathbf{a}}^{(k_0+1)}, \cdots, \hat{\mathbf{a}}^{(k_2)}]$.

We hope that matrix $\hat{\mathbf{A}}_{\mathrm{s}}$ contains as more vectors as possible for training. It means we need a smaller k_1 and a larger k_2. As the value of k_1 decreases, the similarity between $\hat{\mathbf{a}}^{(k_1)}$ and $\hat{\mathbf{a}}^{(k_0+1)}$ will decrease together. Similarly, As the value of k_2 increases, the similarity between $\hat{\mathbf{a}}^{(k_2)}$ and $\hat{\mathbf{a}}^{(k_0)}$ will decrease. We must ensure that the similarity is within a certain range, otherwise the assumption that our method is applicable is no longer valid. So, we set a threshold value of R-squared which named th_{T}. Then, the values of k_1 and k_2 shall satisfy the following two constraint relationships:

$$\max\{R^2(\hat{\mathbf{a}}^{(k_1)}, \hat{\mathbf{a}}^{(k_0+1)}), \cdots, R^2(\hat{\mathbf{a}}^{(k_0-1)}, \hat{\mathbf{a}}^{(k_0+1)})\} < th_{\mathrm{T}}, \quad (9)$$

$$\max\{R^2(\hat{\mathbf{a}}^{(k_0+2)}, \hat{\mathbf{a}}^{(k_0)}), \cdots, R^2(\hat{\mathbf{a}}^{(k_2)}, \hat{\mathbf{a}}^{(k_0)})\} < th_{\mathrm{T}}. \quad (10)$$

Next, our task is to calculate vectors $\hat{\mathbf{a}}_{\mathrm{TG}}^{(1)}, \cdots, \hat{\mathbf{a}}_{\mathrm{TG}}^{(n_0)}$ through matrix $\hat{\mathbf{A}}_{\mathrm{s}}$. Different from the original step, the data obtained in the temporal general step is not sparse, and the generated data is completely unknown. The original GAN is no longer applicable for inferring such data. Compared with GAN, the input vector of CGAN adds r_{c} dimensions as the condition vector. As shown in Fig. 5, inspired by CGAN [14], we design a novel method of fracture vector inference. The detailed algorithm of the temporal generative step is introduced in the following several paragraphs. In order to make it more easy for readers to understand, we give a pseudo code flow table of temporal generative step in Alg. 2.

In this step, the k-th input vector of the generator is $\tilde{\mathbf{z}}_{\mathrm{T}}^{(k)} = [\mathbf{z}_{\mathrm{T}}^{\intercal}, \mathbf{z}_{\mathrm{cT}}^{(k)\intercal}]^{\intercal}$, which includes a common vector $\mathbf{z}_{\mathrm{T}}$

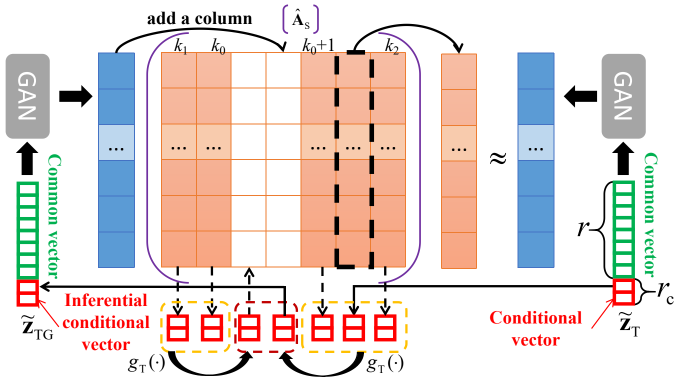


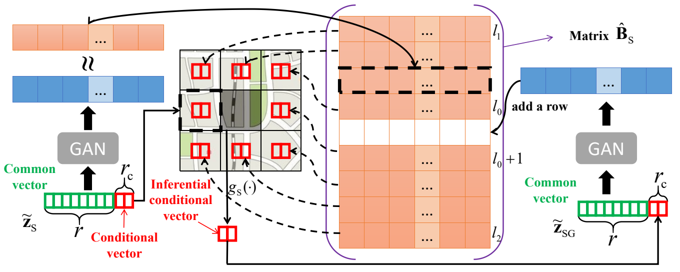
Fig. 5: Structure of GHFMC (Temporal Generative Step).

```
Algorithm 2 The Temporal Generative Step of Generative High-Fidelity Matrix Completion
Input: the sliced temporal fracture matrix $\hat{\mathbf{A}}_s = [\hat{\mathbf{a}}^{(k_1)}, \cdots, \hat{\mathbf{a}}^{(k_0)}, \hat{\mathbf{a}}^{(k_0+1)}, \cdots, \hat{\mathbf{a}}^{(k_2)}]$
Output: the temporal generative vectors $\hat{\mathbf{a}}^{(1)}_{\mathrm{TG}}, \cdots, \hat{\mathbf{a}}^{(n_0)}_{\mathrm{TG}}$
 1: Build the Generator and the Discriminator by using $G_{\mathrm{T}}(\cdot)$ and $D_{\mathrm{T}}(\cdot)$, respectively;
 2: Random Init $\mathbf{z}_{\mathrm{T}}, \mathbf{z}^{(k_1)}_{\mathrm{cT}}, \cdots, \mathbf{z}^{(k_2)}_{\mathrm{cT}}$ and the NN structure parameters of $G_{\mathrm{T}}(\cdot)$ and $D_{\mathrm{T}}(\cdot)$;
 3: $count := 0$;
 4: while not convergent and $count <$ MAX_ITER do
 5:     for $k$ is from $k_1$ to $k_2$ do
 6:         Calculate $D_{\mathrm{T}}(G_{\mathrm{T}}(\mathbf{z}_{\mathrm{T}}, \mathbf{z}^{(k)}_{\mathrm{cT}}))$ and $D_{\mathrm{T}}(\hat{\mathbf{a}}^{(k)})$;
 7:         Fix the NN structure parameters of $D_{\mathrm{T}}(\cdot)$ to make $L^{(k)}_{\mathrm{TD}}$ reduce;
 8:         Calculate $D_{\mathrm{T}}(G_{\mathrm{T}}(\mathbf{z}_{\mathrm{T}}, \mathbf{z}^{(k)}_{\mathrm{cT}}))$ and $D_{\mathrm{T}}(\hat{\mathbf{a}}^{(k)})$ again;
 9:         Fix $\mathbf{z}_{\mathrm{T}}$, $\mathbf{z}^{(k)}_{\mathrm{cT}}$ and the NN structure parameters of $G_{\mathrm{T}}(\cdot)$ at the same time to make $L^{(k)}_{\mathrm{TG}}$ reduce;
10:     end for
11:     $count := count + 1$;
12: end while
13: Calculate $\mathbf{z}^{(1)}_{\mathrm{cTG}}, \mathbf{z}^{(2)}_{\mathrm{cTG}}, \cdots, \mathbf{z}^{(n_0)}_{\mathrm{cTG}}$ by (13);
14: return $\hat{\mathbf{a}}^{(1)}_{\mathrm{TG}} = G_{\mathrm{T}}(\mathbf{z}_{\mathrm{T}}, \mathbf{z}^{(1)}_{\mathrm{cTG}})$, $\hat{\mathbf{a}}^{(2)}_{\mathrm{TG}} = G_{\mathrm{T}}(\mathbf{z}_{\mathrm{T}}, \mathbf{z}^{(2)}_{\mathrm{cTG}})$, $\cdots$, $\hat{\mathbf{a}}^{(n_0)}_{\mathrm{TG}} = G_{\mathrm{T}}(\mathbf{z}_{\mathrm{T}}, \mathbf{z}^{(n_0)}_{\mathrm{cTG}})$.
```

and a conditional vector $\mathbf{z}^{(k)}_{\mathrm{cT}}$. In particular, in the process of training GAN, the parameter $\mathbf{z}_{\mathrm{T}}$ will be shared with all columns because we slice the data with similar features. Different columns only correspond to different conditional vector $\mathbf{z}^{(k)}_{\mathrm{cT}}$. The training processes of the generator $G_{\mathrm{T}}(\cdot)$ and the discriminator $D_{\mathrm{T}}(\cdot)$ are similar to original step. Each column of the matrix $\hat{\mathbf{A}}_s$ is as the training target of the generator $G_{\mathrm{T}}(\cdot)$ and also as the positive sample of the discriminator $D_{\mathrm{T}}(\cdot)$. Similarly, the BCE loss functions of the discriminator and the generator are as follows:

$$L^{(k)}_{\mathrm{TD}} = -\log(D_{\mathrm{T}}(\hat{\mathbf{a}}^{(k)})) - \log(1 - D_{\mathrm{T}}(G_{\mathrm{T}}(\mathbf{z}_{\mathrm{T}}, \mathbf{z}^{(k)}_{\mathrm{cT}})), \quad (11)$$

$$L^{(k)}_{\mathrm{TG}} = -\log(D_{\mathrm{T}}(G_{\mathrm{T}}(\mathbf{z}_{\mathrm{T}}, \mathbf{z}^{(k)}_{\mathrm{cT}}))). \quad (12)$$



Matrix $\hat{\mathbf{B}}_s$
$\approx$
GAN
GAN
Common vector
Common vector
r_c
r_c
r
r
$\widetilde{\mathbf{z}}_s$
$\widetilde{\mathbf{z}}_{\mathrm{SG}}$
Inferential conditional vector
Conditional vector
$g_s(\cdot)$
l_1
l_0
l_0+1
l_2
add a row

Fig. 6: Structure of GHFMC (Spatial Generative Step).

When the training of GAN is finished, we can get a common vector $\mathbf{z}^{(k_1)}_{\mathrm{cT}}$ and a series of condition vectors $\mathbf{z}^{(k_1)}_{\mathrm{cT}}, \cdots, \mathbf{z}^{(k_2)}_{\mathrm{cT}}$ corresponding to the columns of matrix $\hat{\mathbf{A}}_s$. In order to generate temporal fracture data, it is necessary to generate conditional vectors of similar forms for the fracture columns. These condition vectors can be easily inferred by time series inference algorithms (e.g. Bi-LSTM [23]). Assuming that the condition vectors are $\mathbf{z}^{(1)}_{\mathrm{cTG}}, \cdots, \mathbf{z}^{(n_0)}_{\mathrm{cTG}}$ and the time series inference algorithm is $g_{\mathrm{T}}(\cdot)$, it is obvious that

$$(\mathbf{z}^{(1)}_{\mathrm{cTG}}, \cdots, \mathbf{z}^{(n_0)}_{\mathrm{cTG}}) = g_{\mathrm{T}}(\mathbf{z}^{(k_1)}_{\mathrm{cT}}, \cdots, \mathbf{z}^{(k_2)}_{\mathrm{cT}}). \quad (13)$$

So, we can easily generate fracture data with high fidelity through $\hat{\mathbf{a}}^{(j)}_{\mathrm{TG}} = G_{\mathrm{T}}(\mathbf{z}_{\mathrm{T}}, \mathbf{z}^{(j)}_{\mathrm{cTG}})$, where j is from 1 to n_0. Finally, the temporal non-fracture matrix $\hat{\mathbf{B}}$ can be calculated by (8) and subsequent processing can be carried out.

B. Spatial Generative Step

In this step, we use a non-sparse spatiotemporal matrix $\hat{\mathbf{B}} = [\hat{\mathbf{b}}^{(1)\intercal}, \hat{\mathbf{b}}^{(2)\intercal}, \cdots, \hat{\mathbf{b}}^{(m-m_0)\intercal}]^\intercal$ to calculate the m_0 deleted row vectors $\hat{\mathbf{b}}^{(1)}_{\mathrm{SG}}, \cdots, \hat{\mathbf{b}}^{(m_0)}_{\mathrm{SG}}$. Assuming that the fracture row vectors appear between $\hat{\mathbf{b}}^{(l_0)}$ and $\hat{\mathbf{b}}^{(l_0+1)}$, then the spatial non-fracture matrix $\hat{\mathbf{Y}}$ can be expressed as:

$$\begin{aligned}\hat{\mathbf{Y}} = [&\hat{\mathbf{b}}^{(1)\intercal}, \cdots, \hat{\mathbf{b}}^{(l_0)\intercal}, \hat{\mathbf{b}}^{(1)\intercal}_{\mathrm{SG}}, \cdots, \\ &\cdots, \hat{\mathbf{b}}^{(m_0)\intercal}_{\mathrm{SG}}, \hat{\mathbf{b}}^{(l_0+1)\intercal}, \cdots, \hat{\mathbf{b}}^{(m-m_0)\intercal}]^\intercal.\end{aligned} \quad (14)$$

The process of the spatial generative step is basically the same as the temporal generative step. Therefore, in this subsection, we only briefly introduce the meaning of each symbol and give the algorithm flow in Alg. 3 and the sketch map in Fig. 6. $\hat{\mathbf{B}}_s = [\hat{\mathbf{b}}^{(l_1)\intercal}, \cdots, \hat{\mathbf{b}}^{(l_0)\intercal}, \hat{\mathbf{b}}^{(l_0+1)\intercal}, \cdots, \hat{\mathbf{b}}^{(l_2)\intercal}]^\intercal$ denotes the sliced spatial fracture matrix and th_{S} denotes the threshold value of R-squared we set. The values of l_1 and l_2 shall satisfy the following relationships:

$$\max\{R^2(\hat{\mathbf{b}}^{(l_1)}, \hat{\mathbf{b}}^{(l_0+1)}), \cdots, R^2(\hat{\mathbf{b}}^{(l_0-1)}, \hat{\mathbf{b}}^{(l_0+1)})\} < th_{\mathrm{S}}, \quad (15)$$

$$\max\{R^2(\hat{\mathbf{b}}^{(l_0+2)}, \hat{\mathbf{b}}^{(l_0)}), \cdots, R^2(\hat{\mathbf{b}}^{(l_2)}, \hat{\mathbf{b}}^{(l_0)})\} < th_{\mathrm{S}}. \quad (16)$$

The l-th input vector of the generator is $\widetilde{\mathbf{z}}^{(l)}_{\mathrm{S}} = [\mathbf{z}_{\mathrm{S}}, \mathbf{z}^{(l)}_{\mathrm{cS}}]$, which includes a common vector $\mathbf{z}_{\mathrm{S}}$ and a conditional vector $\mathbf{z}^{(l)}_{\mathrm{cS}}$. Similarly, the BCE loss functions of the discriminator and the generator are shown as follows:

$$L^{(l)}_{\mathrm{SD}} = -\log(D_{\mathrm{S}}(\hat{\mathbf{b}}^{(l)})) - \log(1 - D_{\mathrm{S}}(G_{\mathrm{S}}(\mathbf{z}_{\mathrm{S}}, \mathbf{z}^{(l)}_{\mathrm{cS}})), \quad (17)$$

$$L^{(l)}_{\mathrm{SG}} = -\log(D_{\mathrm{S}}(G_{\mathrm{S}}(\mathbf{z}_{\mathrm{S}}, \mathbf{z}^{(l)}_{\mathrm{cS}}))). \quad (18)$$

Algorithm 3 The Spatial Generative Step of Generative High-Fidelity Matrix Completion

```
Input: the sliced spatial fracture matrix $\hat{\mathbf{B}}_s = [\hat{\mathbf{b}}^{(l_1)\intercal}, \cdots, \hat{\mathbf{b}}^{(l_0)\intercal}, \hat{\mathbf{b}}^{(l_0+1)\intercal}, \cdots, \hat{\mathbf{b}}^{(l_2)\intercal}]^\intercal$
Output: the spatial generative vectors $\hat{\mathbf{b}}_{\mathrm{SG}}^{(1)}, \cdots, \hat{\mathbf{b}}_{\mathrm{SG}}^{(m_0)}$
 1: Build the Generator and the Discriminator by using $G_{\mathrm{S}}(\cdot)$ and $D_{\mathrm{S}}(\cdot)$, respectively;
 2: Random Init $\mathbf{z}_{\mathrm{S}}$, $\mathbf{z}_{\mathrm{cS}}^{(l_1)}$, $\cdots$, $\mathbf{z}_{\mathrm{cS}}^{(l_2)}$ and the NN structure parameters of $G_{\mathrm{S}}(\cdot)$ and $D_{\mathrm{S}}(\cdot)$;
 3: $count := 0$;
 4: while not convergent and $count <$ MAX_ITER do
 5:     for $l$ is from $l_1$ to $l_2$ do
 6:         Calculate $D_{\mathrm{S}}(G_{\mathrm{S}}(\mathbf{z}_{\mathrm{S}}, \mathbf{z}_{\mathrm{cS}}^{(l)}))$ and $D_{\mathrm{S}}(\hat{\mathbf{b}}^{(l)})$;
 7:         Fix the NN structure parameters of $D_{\mathrm{S}}(\cdot)$ to make $L_{\mathrm{SD}}^{(l)}$ reduce;
 8:         Calculate $D_{\mathrm{S}}(G_{\mathrm{S}}(\mathbf{z}_{\mathrm{S}}, \mathbf{z}_{\mathrm{cS}}^{(l)}))$ and $D_{\mathrm{S}}(\hat{\mathbf{b}}^{(l)})$ again;
 9:         Fix $\mathbf{z}_{\mathrm{S}}$, $\mathbf{z}_{\mathrm{cS}}^{(l)}$ and the NN structure parameters of $G_{\mathrm{S}}(\cdot)$ at the same time to make $L_{\mathrm{SG}}^{(l)}$ reduce;
10:     end for
11:     $count := count + 1$;
12: end while
13: Calculate $\mathbf{z}_{\mathrm{cSG}}^{(1)}, \mathbf{z}_{\mathrm{cSG}}^{(2)}, \cdots, \mathbf{z}_{\mathrm{cSG}}^{(n_0)}$ by (19);
14: return $\hat{\mathbf{b}}_{\mathrm{SG}}^{(1)} = G_{\mathrm{S}}(\mathbf{z}_{\mathrm{S}}, \mathbf{z}_{\mathrm{cSG}}^{(1)})$, $\hat{\mathbf{b}}_{\mathrm{SG}}^{(2)} = G_{\mathrm{S}}(\mathbf{z}_{\mathrm{S}}, \mathbf{z}_{\mathrm{cSG}}^{(2)})$, $\cdots$, $\hat{\mathbf{b}}_{\mathrm{SG}}^{(m_0)} = G_{\mathrm{S}}(\mathbf{z}_{\mathrm{S}}, \mathbf{z}_{\mathrm{cSG}}^{(m_0)})$.
```

It should be noted that the conditional vector inference method in the spatial generative step is different from that in the temporal generative step. The premise of spatial inference is that there is continuity and data correlation between each subareas. For example, the air quality of a subarea is close to its neighbor area. However, if the subareas are different parking lots, the relationship between the parking numbers of these parking lots may not be strong. In this case, we are trying to estimate the number of cars in a parking lot without any prior information. This is simply impossible. Therefore, we only infer the spatial fracture data in the case of continuous subareas. These condition vectors can be easily inferred by data inference algorithms (e.g. DMF [11]). Assuming that the condition vectors are $\mathbf{z}_{\mathrm{cSG}}^{(1)}, \cdots, \mathbf{z}_{\mathrm{cSG}}^{(m_0)}$ and the data inference algorithm is $g_{\mathrm{S}}(\cdot)$, it is obvious that

$$(\mathbf{z}_{\mathrm{cSG}}^{(1)}, \cdots, \mathbf{z}_{\mathrm{cSG}}^{(m_0)}) = g_{\mathrm{S}}(\mathbf{z}_{\mathrm{cS}}^{(l_1)}, \cdots, \mathbf{z}_{\mathrm{cS}}^{(l_2)}). \qquad (19)$$

So, we can easily generate fracture data with high fidelity through $\hat{\mathbf{b}}_{\mathrm{SG}}^{(i)} = G_{\mathrm{S}}(\mathbf{z}_{\mathrm{S}}, \mathbf{z}_{\mathrm{cSG}}^{(i)})$, where i is from 1 to m_0. Finally, the spatial non-fracture matrix $\hat{\mathbf{Y}}$ can be calculated by (14) and we achieve our targets by our approach GHFMC.

VI. Performance Evaluation

In this section, we first show the details of all the three typical datasets and the latest or popular baseline methods in the field of sparse data inference. Then we present the performance evaluation results about each datasets for our approach. In particular, in order to explain the purpose of the experiments we test, the main research questions are given as the following points:

- **RQ1**: Does our approach improve the accuracy and fidelity of sparse matrix completion?
- **RQ2**: Can our approach be more suitable for inferring fracture data than other methods?
- **RQ3**: Is high fidelity data inference more conducive to fracture data inference?
- **RQ4**: Can our approach cope with a larger data fracture scale than other existing methods?

A. Datasets

Aiming at evaluating the spatiotemporal fracture data inference problem, we test on three popular urban crowdsensing datasets, including *U-Air* [24], *Sensor-Scope* [25], and *Parking in Birmingham* [26]. We provide a main content description of these three datasets in Table II.

B. Baselines & Measures

1) Baselines: In order to effectively utilize the sparse sensed data to infer spatiotemporal fracture data, we first present the data inference algorithm with GHFMC. We mainly compare our approach with the following multiple types of data inference algorithms:

- KNN [8]: K-Nearest Neighbor, which calculates the average value of the top-K nearest sensed time slots.
- GPR [27], [28]: Gaussian Process Regression, which uses Gaussian distribution to fit the sensed data values and generate unsensed data.
- GRU [29]: Gated Recurrent Unit, which is a popular time series prediction method and used to infer the value of temporal fracture data.
- DMF [11]: Deep Matrix Factorization, which is a matrix factorization-based neural network and usually used to complete a sparse matrix.
- IGMC [10]: Inductive Graph-based Matrix Completion, which is a Graph Neural Networks-based data inference method and usually used to predict the unknown value of recommender systems.
- AR-CF [30]: Augmented Reality Collaborative Filtering, which is a new method to deal with the cold start problem of recommender system.

2) Measures: In the following experiments, we will evaluate our approach from the error and fidelity of data inference.

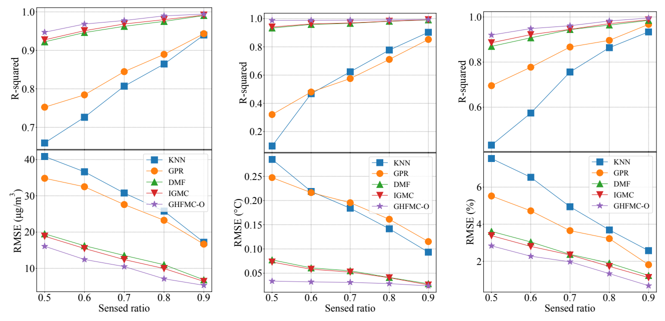
- RMSE: Root Mean Square Error, which denotes the data inference error. A lower value of RMSE means the high data inference accuracy.
- R-squared: Goodness of Fit, which denotes the data inference fidelity. A higher value of R-squared means the high data inference fidelity.

*C. Sparse Spatiotemporal Data Inference **(RQ1)***

We start to test the accuracy and fidelity of sparse spatiotemporal matrix completion for the GHFMC (Origianl Step) algorithm. We artificially randomly extract a certain ratio of sparse data, although the original datasets provide complete

TABLE II: Statistics of three evaluation datasets

	Datasets		
	U-Air	***Sensor-Scope***	***Parking in Birmingham***
Country - City	China - Beijing	Switzerland - Lausanne	UK - Birmingham
Data (Unit)	PM2.5 ($\mu g/m^3$)	Temperature ($°C$)	Parking occupancy rate ($\%$)
Subarea	36 subareas each with $1km \times 1km$	57 subareas each with $50m \times 30m$	30 parking lots
Period & Duration	1h & 11d	0.5h & 7d	0.5h & 77d
Mean $\pm$ Std.	79.11 ± 81.21	6.04 ± 1.87	53.6 ± 26.3



(a) PM2.5 (b) Temperature (c) Occupancy rate

Fig. 7: Accuracy and fidelity of sparse data inference over *U-Air*, *Sensor-Scope* and *Parking in Birmingham*.

spatiotemporal matrices. We set the sensed ratio from 50% to 90% and complete the sparse matrix by different data inference methods, which include KNN, GPR, DMF, IGMC and GHFMC-O. In this experiment, we calculate both RMSE and R-squared value to evaluate the data inference accuracy and fidelity of each method for sparse spatiotemporal matrix completion. The results of this experiment are shown in Fig. 7, which includes all three typical urban sensing tasks. In order to compare different experimental results of the same task easily, we use a special coordinate system, which omits one common abscissa axis and two independent ordinate axes.

The experimental results in Fig. 7 show that with the increase of sensed ratio, the data inference error decreases and the fidelity of data inference increases. It is easy to understand, because the sensing task with larger sensed ratio is more easier to achieve. KNN has better inference effect for data with strong linear characteristics, and GPR assumes that the distribution of data approximately obeys Gaussian distribution. However, the real data are more nonlinear and the distribution characteristics are more complex, so the performance of KNN and GPR is poor. DMF, IGMC and GHFMC-O can extract complex features from sensed data, so the accuracy and fidelity of data inference are better than KNN and GPR.

Generative algorithm is a new attempt in the field of sparse data inference. From the experimental results in Fig. 7, we find that generative algorithm (GHFMC-O) seems to be more suitable for sparse data completion than inferential algorithm (DMF and IGMC) because of the smallest data inference error and the largest data inference fidelity. In other words, the generative algorithm can generate batch data with real world distribution characteristics.

TABLE III: R-squared of fracture data inference under different methods over *U-Air*, *Sensor-Scope* and *Parking in Birmingham*

	DMF	**GRU**	**IGMC**	**AF-CF**	**GHFMC-T**
PM2.5	0.2648	0.9172	0.9194	0.9237	<u>0.9443</u>
Tem.	0.3081	0.7757	0.7867	0.8198	<u>0.8282</u>
Par.	0.6028	0.8731	0.8848	0.9006	<u>0.9171</u>

D. Spatiotemporal Fracture Data Inference (***RQ2***)

Then, we test the accuracy and fidelity of spatiotemporal fracture data inference for the GHFMC (Temporal Generative Step) algorithm. In this experiment, we randomly extract a column from the complete matrix of the original dataset as temporal fracture data. We try to recover the unsensed column by different methods, which include GRU, AR-CF, DMF, IGMC and GHFMC-T. We calculate the value of R-squared in order to test the fidelity of fracture data inference, which is the technical index we are most concerned about.

The experimental results are shown in TABLE III. By comparing the R-squared of different methods, we find that the R-squared value of DMF is the smallest, or even worse than that using GRU directly for time series predition. This is because DMF algorithm needs a convergence target to run effectively. When a whole column of data is missing, there is no context infomation for DMF. Then the output of the DMF will be completely random. IGMC algorithm is a GNN-based method which takes advantage of complex relationships, but it also performs generally for temporal fracture data. It shows that the increase of calculation cannot make up for the influence of data fracture. AR-CF algorithm is the latest method to deal with the cold start problem of the recommended system. The cold start problem of the recommended system is similar to the spatiotemporal fracture problem, so we also compare the AR-CF. Indeed, AR-CF still performs well in the urban crowdsensing dataset. However, due to the difference between spatiotemporal data and user-item data, GHFMC-T performs better than AR-CF. In the recommendation system, users and projects exist in isolation. So, we may draw the conclusion that our approach is more suitable for continuous data than discrete data.

E. Influence of Sparse Data Inference Accuracy on Fracture Data Inference (***RQ3***)

From the first experiment, we found that GHFMC-O performs best in sparse data inference accuracy and fidelity. In this experiment, we will verify whether the high fidelity we pursue

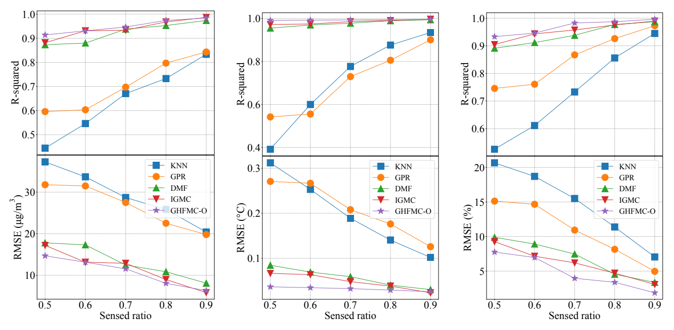

(a) PM2.5 (b) Temperature (c) Occupancy rate

Fig. 8: Accuracy and fidelity of fracture data inference over *U-Air*, *Sensor-Scope* and *Parking in Birmingham*.

in the sparse data completion step is meaningful. We directly use the data inference results of different sparse data inference methods under different sensed ratios. Then, we used GHFMC, which has been verified to be most suitable for spatiotemporal fracture data inference, to perform operations similar to the last experiment. we calculate both RMSE and R-squared value to evaluate the data inference accuracy and fidelity of each method for spatiotemporal fracture data inference. The results of this experiment are shown in Fig. 8.

The experimental results show that our approach GHFMC-O not only performs best in sparse data inference, but also is conducive to further processing fracture data of sparse data background. It is easy to understand that we need data as real as possible to train our GHFMC-T. Under the sparse background, the fidelity of complete data obtained by KNN, GPR, DMF and IGMC is all worse than GHFMC-O. Considering that the sparse data inference error of KNN and GPR is much higher than that of DMF, IGMC and GHFMC-O, we only compare DMF, IGMC and GHFMC-O in the next discussion. These three methods can effectively extract the non-linear features of data and show similar sparse data inference error. Therefore, on this premise, we have reason to admit that GHFMC-O can indeed generate inference results that are more consistent with the distribution law of real data.

F. Impact of Fracture Data Scale ***(RQ4)***

Finally, we will test how large scale fracture data our approach can adapt. It is obvious that with the increase of continuous fracture data scale, the inference of fracture data is becoming more and more difficult. Reflected in the technical indicators, the error is getting larger and larger. In the actual urban crowdsensing scene, we only tolerate errors within a certain range. Therefore, we test the number of effective fracture data scale that can be recovered by different methods under different error thresholds. We first test the RMSE with the fracture data scale of 1-10 time slots by different methods. Then, by setting different RMSE thresholds, we count how many time slots of the test results are below the threshold. The experimental results are shown in the Fig. 9.

As the RMSE threshold decreases, more data points become ineffective. It means that our requirements for data inference

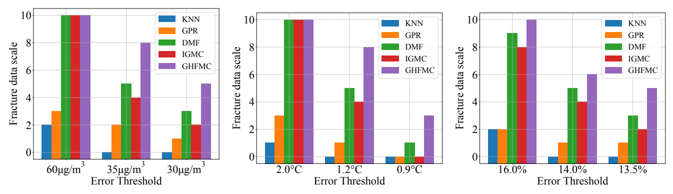

(a) PM2.5 (b) Temperature (c) Occupancy Rate

Fig. 9: Inference accuracy under different fracture data scales over *U-Air*, *Sensor-Scope*, and *Parking in Birmingham*.

results are becoming more and more strict. We find that even under such a strict threshold (There are a little number of effective fracture data scale by DMF, IGMC, GPR and KNN), GHFMC can also perform best. This is because the data inference error of our approach is less than that of other baseline methods for any scale of fracture. This experiment indirectly verifies that our approach can generate high fidelity fracture data. Although other methods perform well in the field of sparse data inference of non-fracture data, when encountering spatiotemporal fracture data, I suggest to apply GHFMC to deal with spatiotemporal fracture data especially when the fracture scale is large.

VII. Conclusion

In this paper, we propose the approach to address the problem of spatiotemporal fracture data inference in sparse urban crowdsensing. In detail, we propose a general urban sensing method named GHFMC, which aims to generate all unsensed value with a high accuracy and fidelity. With the benefit from the idea of DMF and GAN, GHFMC is proposed to extract the spatiotemporal features of the sensed data more effectively, and generates inference data more close to the real data distribution more easily. By the test about three different types of classical urban sensing tasks from three popular datasets, we verify that the performance of our approach outperforms the state-of-the-art (SOTA) baselines, which include the methods not only in the field of MCS but also other similar fields. In addition, it's necessary to explain that every sub-module (Original step, temporal generative step and spatial generative step) could be used in an ad-hoc manner with functionalities independently. For example, for the case of non-fracture data, GHFMC-O can also be used for sparse data inference and achieve better data inference accuracy and fidelity. For the case of non-sparse fracture data, we can skip matrix completion step and use GHFMC-T or GHFMC-S to infer the fracture data directly. It seems that our work can be improved. For example, 1) the algorithm based on GaN has high time complexity, and 2) the temporal and spatial fracture data inference are not carried out at the same time. Moreover, our approach can be applied in a wider range of crowdsensing tasks (e.g. congestion detection, traffic speed monitor, etc.), although we test only three tasks in the evaluation experiments.

References

[1] A. Nieto, A. Acien, and G. Fernandez, "Crowdsourcing analysis in 5g iot: Cybersecurity threats and mitigation," *Mob. Networks Appl.*, vol. 24, no. 3, pp. 881–889, 2019.

[2] L. Tan, H. Xiao, K. Yu, M. Aloqaily, and Y. Jararweh, "A blockchain-empowered crowdsourcing system for 5g-enabled smart cities," *Comput. Stand. Interfaces*, vol. 76, p. 103517, 2021.

[3] J. Bian, H. Xiong, Y. Fu, and S. K. Das, "CSWA: aggregation-free spatial-temporal community sensing," in *AAAI 2018*. AAAI Press, 2018, pp. 2087–2094.

[4] R. K. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: current state and future challenges," *IEEE Commun. Mag.*, vol. 49, no. 11, pp. 32–39, 2011.

[5] C. Xiang, P. Yang, C. Tian, L. Zhang, H. Lin, F. Xiao, M. Zhang, and Y. Liu, "CARM: crowd-sensing accurate outdoor RSS maps with error-prone smartphone measurements," *IEEE Trans. Mob. Comput.*, vol. 15, no. 11, pp. 2669–2681, 2016.

[6] C. Xiang, P. Yang, C. Tian, H. Cai, and Y. Liu, "Calibrate without calibrating: An iterative approach in participatory sensing network," *IEEE Trans. Parallel Distributed Syst.*, vol. 26, no. 2, pp. 351–361, 2015.

[7] X. Fan, C. Xiang, C. Chen, P. Yang, L. Gong, X. Song, P. Nanda, and X. He, "Buildsensys: Reusing building sensing data for traffic prediction with cross-domain learning," *IEEE Trans. Mob. Comput.*, vol. 20, no. 6, pp. 2154–2171, 2021.

[8] L. Wang, D. Zhang, A. Pathak, C. Chen, H. Xiong, D. Yang, and Y. Wang, "CCS-TA: quality-guaranteed online task allocation in compressive crowdsensing," in *UbiComp 2015*. ACM, 2015, pp. 683–694.

[9] L. Wang, D. Zhang, D. Yang, A. Pathak, C. Chen, X. Han, H. Xiong, and Y. Wang, "SPACE-TA: cost-effective task allocation exploiting intradata and interdata correlations in sparse crowdsensing," *ACM Trans. Intell. Syst. Technol.*, vol. 9, no. 2, pp. 20:1–20:28, 2018.

[10] M. Zhang and Y. Chen, "Inductive matrix completion based on graph neural networks," in *ICLR 2020*. OpenReview.net, 2020.

[11] J. Fan and J. Cheng, "Matrix completion by deep matrix factorization," *Neural Networks*, vol. 98, pp. 34–41, 2018.

[12] E. Wang, M. Zhang, X. Cheng, Y. Yang, W. Liu, H. Yu, L. Wang, and J. Zhang, "Deep learning-enabled sparse industrial crowdsensing and prediction," *IEEE Trans. Ind. Informatics*, vol. 17, no. 9, pp. 6170–6181, 2021.

[13] E. Wang, M. Zhang, Y. Yang, Y. Xu, and J. Wu, "Exploiting outlier value effects in sparse urban crowdsensing," in *IWQoS 2021*. IEEE, 2021, pp. 1–10.

[14] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *CoRR*, vol. abs/1411.1784, 2014.

[15] D. Zhang, L. Wang, H. Xiong, and B. Guo, "4w1h in mobile crowd sensing," *IEEE Commun. Mag.*, vol. 52, no. 8, pp. 42–48, 2014.

[16] Y. Zhu, Z. Li, H. Zhu, M. Li, and Q. Zhang, "A compressive sensing approach to urban traffic estimation with probe vehicles," *IEEE Trans. Mob. Comput.*, vol. 12, no. 11, pp. 2289–2302, 2013.

[17] K. Xie, X. Li, X. Wang, G. Xie, J. Wen, and D. Zhang, "Active sparse mobile crowd sensing based on matrix completion," in *SIGMOD 2019*. ACM, 2019, pp. 195–210.

[18] R. K. Rana, C. T. Chou, S. S. Kanhere, N. Bulusu, and W. Hu, "Ear-phone: an end-to-end participatory urban noise mapping system," in *IPSN 2010*. ACM, 2010, pp. 105–116.

[19] T. Liu, Y. Zhu, Y. Yang, and F. Ye, "Alc^2: When active learning meets compressive crowdsensing for urban air pollution monitoring," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 9427–9438, 2019.

[20] S. He and K. G. Shin, "Steering crowdsourced signal map construction via bayesian compressive sensing," in *INFOCOM 2018*. IEEE, 2018, pp. 1016–1024.

[21] K. Xie, X. Ning, X. Wang, D. Xie, J. Cao, G. Xie, and J. Wen, "Recover corrupted data in sensor networks: A matrix completion solution," *IEEE Trans. Mob. Comput.*, vol. 16, no. 5, pp. 1434–1448, 2017.

[22] 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 2014*, 2014, pp. 2672–2680.

[23] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, 1997.

[24] Y. Zheng, F. Liu, and H. Hsieh, "U-air: when urban air quality inference meets big data," in *KDD 2013*. ACM, 2013, pp. 1436–1444.

[25] F. Ingelrest, G. Barrenetxea, G. Schaefer, M. Vetterli, O. Couach, and M. Parlange, "Sensorscope: Application-specific sensor network for environmental monitoring," *ACM Trans. Sens. Networks*, vol. 6, no. 2, pp. 17:1–17:32, 2010.

[26] D. H. Stolfi, E. Alba, and X. Yao, "Predicting car park occupancy rates in smart cities," in *Smart-CT 2017*, ser. Lecture Notes in Computer Science, vol. 10268. Springer, 2017, pp. 107–117.

[27] F. Yin and F. Gunnarsson, "Distributed recursive gaussian processes for RSS map applied to target tracking," *IEEE J. Sel. Top. Signal Process.*, vol. 11, no. 3, pp. 492–503, 2017.

[28] H. Yen and C. Wang, "Cross-device wi-fi map fusion with gaussian processes," *IEEE Trans. Mob. Comput.*, vol. 16, no. 1, pp. 44–57, 2017.

[29] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *EMNLP 2014*. ACL, 2014, pp. 1724–1734.

[30] D. Chae, J. Kim, D. H. Chau, and S. Kim, "AR-CF: augmenting virtual users and items in collaborative filtering for addressing cold-start problems," in *SIGIR 2020*. ACM, 2020, pp. 1251–1260.