This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3195706

Spatiotemporal Urban Inference and Prediction in Sparse Mobile CrowdSensing: a Graph Neural Network Approach

En Wang, Weiting Liu, Wenbin Liu*, Yongjian Yang, Bo Yang, and Jie Wu, *Fellow, IEEE*

Abstract—Mobile CrowdSensing (MCS) has recently become a promising data acquisition paradigm, which recruits a large number of users to collect data from the target sensing areas. Obviously, with the increase of sensing scale and the decrease of sensing granularity, traditional MCS cannot fully cover the required sensing areas especially the inaccessible areas. As a variant, Sparse MCS can utilize the spatiotemporal correlations in sensing data to infer the whole sensing map only by sensing a few subareas. However, in many real-world scenarios, such as traffic congestion prediction or parking occupancy detection, inferring the current unsensed data may not be the final goal. By comparison, it is more important to get the future information through the sparse sensed data. In this paper, we turn attention from inferring the current unsensed data to predicting the future unknown data and propose an urban inference and prediction framework in Sparse MCS. To deal with the sparse sensed data, we first present a bipartite-graph-based matrix completion algorithm with spatiotemporal constraints to accurately recover the current full map. Then, by exploiting spatiotemporal correlations based on the inferred full map, we present a Graph Convolutional Networks (GCN) with spatiotemporal attention to predict the future maps. Furthermore, we design a spatiotemporal iterative method to repeatedly update the spatiotemporal attentions and constraints, in order to connect the urban inference and prediction to improve the accuracy of the whole framework. Extensive experiments have been conducted on two types of typical urban sensing tasks with four real-world data sets, which verify the effectiveness of our proposed algorithms in improving the inference and prediction accuracy with the sparse sensed data.

Index Terms—Mobile crowdsensing, matrix completion, graph convolution networks, spatiotemporal correlations

1 Introduction

With the rapid development of information society and wireless devices portability, Mobile CrowdSensing (MCS) [2] has become a new data collection mode combined with crowdsensing and mobile devices [3], [4]. It recruits users carrying mobile devices to collect data from target sensing areas in order to perform various sensing tasks, such as environment monitoring [5], traffic controlling [6] and urban sensing [7], etc. To obtain high-quality sensing results, the traditional MCS system usually has to recruit lots of users to cover the entire sensing map. However, considering the costs limitation, it cannot afford too many users. Even if it has recruited enough users with acceptable cost, some sensing subareas may still have no available users due to the users' uncertain mobility or subareas' inaccessibility [8]. Therefore, in most cases, the traditional MCS can only collect incomplete or even sparse data, especially facing large-scale and fine-grained urban sensing tasks.

As to this problem, researchers have proposed a more practical data acquisition paradigm, called Sparse MCS [9], [10], [11], [12], [13]. Sparse MCS aims to infer the data of unsensed subareas through the already sensed data to complete the entire sensing map. As shown in Fig. 1, we want to get complete fine-grained data from the sensing map at the current time in this Sparse MCS scenario, while only 3 out of 5×4 subareas have been sensed, and we need to infer the data in unsensed subareas from the sparse data. To deal with this problem, most of the existing work focuses on the data correlations among the sensed data to design various data inference algorithms. Bose *et al.* [14] use interpolation methods to infer the entire sensed data matrix by the sparse data. Liu *et al.* [15], Wang *et al.* [9], [10], and He *et al.* [16] use the Compressive Sensing and its variants (*e.g.*, Spatiotemporal-CS and Bayesian-CS) to recover the entire sensed matrix under the spatiotemporal limit. Besides, Liu *et al.* [11] and Xie *et al.* [12] find that subareas selection is also important and use different selection methods based on the Reinforcement Learning technology

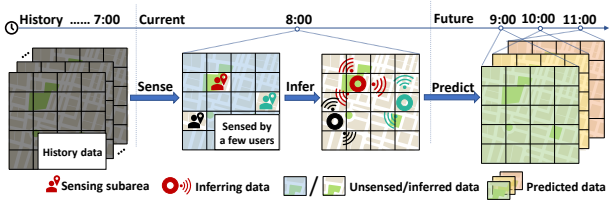


Fig. 1: Users sense data from a few subareas, which could be used to infer the data of unsensed subareas at the current and predict the data of all subareas in the future.

- *En Wang, Weiting Liu, Wenbin Liu, Yongjian Yang and Bo Yang are with the Department of Computer Science and Technology and Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, Changchun 130012, China. (e-mail: wangen@jlu.edu.cn; liuwt20@mails.jlu.edu.cn; liuwenbin@jlu.edu.cn; yyj@jlu.edu.cn; ybo@jlu.edu.cn)*
- *Jie Wu is with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122, USA. (e-mail: jiewu@temple.edu)*

Corresponding author: Wenbin Liu.

A preliminary version of this work appeared in the IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS 2020) [1].

and the sensing scheduling scheme to select more important sensing subareas in order to improve inference accuracy. In addition, some work is committed to the tradeoff between inference quality and inference cost [17], [18].

However, these existing works mainly focus on utilizing the partial sensing data to infer the current full sensing map, they ignore that not only the current data but also the future ones are important in many real-world scenarios. Taking traffic sensing task as an example, compared with sensing the destination's current traffic congestion or parking occupancy, it may be more useful to predict the future situation, because it will take a while for us to reach the destination. Therefore, in Sparse MCS, *inferring the current sensing data may not be the final goal, while it's highly significant to get the future information through the sparse sensed data*.

To achieve this goal, intuitively, we can ignore the data sparsity and predict the future data based on the already sensed ones or regard the future data as the missing ones and infer them directly. However, most of the traditional prediction methods require large amounts of complete sensing data, which are utilized to excavate the spatiotemporal correlations among them for predicting the future full map. While in sparse MCS, we are faced with *the sparse sensed data*, such methods can not effectively exploit the correlations and thus achieve bad performances in most cases. In addition, many existing data inference algorithms are not *designed for prediction*, they simply use data imputation as data preprocessing for prediction, or independently imputation and prediction with no connections. Generally, they fill in the missing values by satisfying some constraints or properties, e.g., the low-rank structure of sensing data matrix, which can hardly deal with the future inferring without any sensed data. Furthermore, considering the non-linear temporal relationships between the pair-wise spatial correlations across different subareas, *the spatiotemporal correlations* among sensing data are actually very complicated. We should not only extract them from the sparse sensed data but also utilize them for prediction. Therefore, how to effectively utilize the sparse sensed data to extract the spatiotemporal correlations for inference and prediction is the main challenge in our urban prediction problem via Sparse MCS.

In this paper, we turn attention from inferring the current unsensed data to predicting the future full map from the sparse sensed data. We propose an urban inference and prediction framework via Sparse MCS consisting of three parts: data inference, data prediction, and iterative update. Firstly, we propose a bipartite-graph-based matrix completion algorithm with spatiotemporal constraints to recover the current full sensing map from the sparse sensed data. With the help of the low-rank attributes and general spatiotemporal, we can solve the problem which is difficult to extract enough correlations from sparse data, so as to improve the inference accuracy. Note that the added spatiotemporal constraints not only guide the inference directions but also preserve the latent spatiotemporal correlations for prediction. We then present a Graph Convolutional Network (GCN) model with spatiotemporal attentions to predict the future. Once having the complete sensing information, we can utilize this GCN model to exploit more spatiotemporal correlations among sensing data and use attention to assist prediction. Finally, we use spatiotemporal attention matrix in data prediction to update spatiotemporal constraint matrix in data inference iteratively to enhance the correlations and improve the performances of data inference and prediction.

Our work has the following contributions:

- We formulate the fine-grained urban prediction problem via Sparse MCS, which turns attention from inferring the current unsensed data to predicting the future full map from the sparse sensed data.
- We propose a bipartite-graph-based matrix completion algorithm with spatiotemporal constraints to recover the current full sensing map from the sparse sensed data.
- We present a GCN model with spatiotemporal attentions to predict the full map based on the recovered sensing.
- We fuse the spatiotemporal constraints and spatiotemporal attentions iteratively to enhance the spatiotemporal correlations for inference and prediction.
- We evaluate the proposed algorithms on two types of typical urban sensing tasks (environmental sensing and traffic monitoring) with four real-world data sets (Humidity, PM2.5, Traffic Speed, and Traffic flow), which show that our methods can effectively improve the accuracy of inference and prediction. We also evaluate the influences and impacts of the spatiotemporal correlations.

The remainder of this paper is organized as follows. After reviewing the related works in Section 2, we introduce the system model and formulate the problem in Section 3. Then, the inference and prediction methods are proposed in Section 4 and 5, followed by the spatiotemporal matrix iteration in Section 6. Finally, we evaluate the performance in Section 7 and conclude this paper in Section 8.

2 Related Work

Sparse Mobile CrowdSensing [9], [10], [11], [12], [13] has played an essential role in the Mobile CrowdSensing because of its property of using little information to infer the full sensing map. Due to cost or other constraints [8], [19], it is more suitable for some real-world scenarios than the traditional MCS [2], [5].

Recently, many fine-grained urban sensing systems have been developed via Sparse MCS. Rana $et\ al.$ [20] conducted an urban noise monitoring system that randomly senses data from some target subareas and uses compressive sensing to infer a fine-grained urban noise map. Zhu $et\ al.$ [21] also used a modified compressive sensing approach to estimate the urban traffic speeds, based on the data periodically collected by probe vehicles. The above works are all solving the Sparse MCS problem from a specific direction, Wang $et\ al.$ [9], [10] further provided a general framework for Sparse MCS that includes three stages: cell selection, data inference, and quality assessment. Briefly, Sparse MCS will first select some subareas to sense and then uses the sensed data to infer the full map. If the inferred results are of poor quality, Sparse MCS continues to sense

data and infer the full map. The authors also conducted various experiments over four urban sensing tasks, including temperature, humidity, air quality and traffic monitoring, in order to verify the effectiveness of Sparse MCS. Similarly, Lana *et al.* [22] and Liu *et al.* [23] proposed a sparse MCS-based urban traffic monitoring system, which uses an imputation method for missing value filling. He *et al.* [16] and Liu *et al.* [15] also proposed the urban air pollution and signal mapping systems based on Sparse MCS, while they further added the incentive design to steer users to sense data from some subareas. Furthermore, to effectively sense the useful subareas which hold sufficient information for data inference, Liu *et al.* [11] introduced the Reinforcement Learning and Xie *et al.* [12] studied the sensing scheduling scheme to actively determine the next sensing subareas.

Although Sparse MCS provides an effective way for fine-grained urban sensing systems, current research focuses primarily on inferring current unsensed data and is unable to anticipate the near future full map. To solve this problem, some prediction models based on spatiotemporal conditions have been developed. Williams *et al.* [24] proposed a univariate auto-regressive time series prediction model based on the Wold Decomposition Theorem to forecast vehicular traffic flow, which Zivot *et al.* [25] then expanded to multi variables. However, these models are unable to handle nonlinear data. To deal with nonlinear data prediction, Van *et al.* [26] proposed a hybrid short-term spatiotemporal data prediction approach based on the K-Nearest Neighbor (KNN) algorithm. Simultaneously, Jeong *et al.* [27] proposed a new online short-time data prediction model based on the temporal correlation, named the Online Weighted Support Vector Regression model (OLWSVR). Furthermore, relying on Deep Learning's more powerful ability to capture spatiotemporal features, Zhang *et al.* [28] designed an ST-RESNET model based on the residual convolutional unit to predict spatiotemporal data. Specifically, they first used a residual network to capture spatial features of data, then used multi networks to extract temporal features of data, and finally achieved a better prediction effect. Besides, Yao *et al.* [29] integrated a Convolutional Neural Network (CNN) and Long-Short Term Memory artificial neural network (LSTM) to jointly model the spatiotemporal correlation, and proposed the Deep Multi-View Spatial-Temporal Network (DMVST-NET) to deal with the prediction problem. Although the above works made great contributions to prediction, they do not take into account the fact that in a sparse environment the inference and prediction algorithms should be closely coupled to avoid the loss of data features. There are also some works that are specifically concerned with this issue. Chan *et al.* [30] and Gupta *et al.* [31] proposed a neural network for prediction with a missing data imputation system, tying inference and prediction together. Wang *et al.* [32] proposed an inference-prediction framework, which first infers the sparse data using Deep Matrix Factorization (DMF) and then performs short-time data prediction using Non-Linear Auto-Regressive (NAR) neural network and Stacked Denoising Auto-Encoder (SDAE). However, in the face of graph-structured data, it is challenging to capture, store and use spatiotemporal correlations, especially when these correlations are critical for inference and prediction.

3 System Model and Problem Formulation

3.1 System Model

We consider a general urban sensing scenario where the sensing system recruits some users to collect data from a large-scale target sensing area to provide the fine-grained urban sensing services. We divide the whole sensing campaign into many equal-length sensing cycles and the target area is split into m subareas, in order to provide the fine-grained results. Note that the lengths of cycles and the sizes of subareas are predetermined according to the tasks' requirements.

Under such a fine-grained urban sensing scenario, we recruit users to sense data from some subareas, and use the data inference algorithms to infer the unsensed data for each sensing cycle. Specifically, for each sensing cycle, we sense some data from a few subareas, which are recorded in a vector $\boldsymbol{y}' \in \mathbb{R}^{m\times 1}$, and the unsensed data are recorded as 0 (if 0 is not the reasonable sensing value, then we should use another value). Let the vector $\boldsymbol{y} \in \mathbb{R}^{m\times 1}$ denote the ground truth and the binary sensed vector $\boldsymbol{c} \in \mathbb{R}^{m\times 1}$ mark whether one subarea has been sensed: if subarea i has been sensed at the current cycle, $\boldsymbol{c}[i] = 1$; otherwise, $\boldsymbol{c}[i] = 0$, and thus

$$\boldsymbol{y}' = \boldsymbol{y} \bullet \boldsymbol{c}, \tag{1}$$

where $\bullet$ represents an element-wise product, i.e. , $\boldsymbol{y}'[i] = \boldsymbol{y}[i] \times \boldsymbol{c}[i]$. Then, we use a data inference algorithm I to infer the unsensed data from the sensed data $\boldsymbol{y}'$ with the spatiotemporal constraints, and the inference error is ε.

$$I(\boldsymbol{y}') = \hat{\boldsymbol{y}} \approx \boldsymbol{y}, \tag{2}$$

$$\varepsilon(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \sum\nolimits_{i=1}^{m} |\, \boldsymbol{y}[i] - \hat{\boldsymbol{y}}[i] \,|. \tag{3}$$

At the $n-$th sensing cycle, we have already sensed n vectors and obtained the actually sensed matrix $\boldsymbol{Y}'_n \triangleq \{\boldsymbol{y}'_1, \boldsymbol{y}'_2, ..., \boldsymbol{y}'_n\}$. Similarly, let $\boldsymbol{Y}_n \triangleq \{\boldsymbol{y}_1, \boldsymbol{y}_2, ...\boldsymbol{y}_n\}$ and $\boldsymbol{C}_n \triangleq \{\boldsymbol{c}_1, \boldsymbol{c}_2, ...\boldsymbol{c}_n\}$, we have $\boldsymbol{Y}'_n = \boldsymbol{Y}_n \bullet \boldsymbol{C}_n$. Then, with the goal to accurately predict the fine-grained near future full map from the sparse sensed data, we first use I to recover the historical sensed matrix $\hat{\boldsymbol{Y}}_n$ from $\boldsymbol{Y}'_n$ and then use a near-future prediction method P with the spatiotemporal correlations to predict the future vector:

$$P(I(\boldsymbol{Y}'_n)) = P(\hat{\boldsymbol{Y}}_n) = \hat{\boldsymbol{y}}_{n+1} \approx \boldsymbol{y}_{n+1}, \tag{4}$$

$$\varepsilon(\boldsymbol{y}_{n+1}, \hat{\boldsymbol{y}}_{n+1}) = \sum\nolimits_{i=1}^{m} |\, \boldsymbol{y}_{n+1}[i] - \hat{\boldsymbol{y}}_{n+1}[i] \,|. \tag{5}$$

We can add the predicted $\hat{\boldsymbol{y}}_{n+1}$ to $\hat{\boldsymbol{Y}}_n$ and obtained $\hat{\boldsymbol{Y}}_{n+1}$.

In order to more clearly show the spatiotemporal correlations in our framework, we use the $\hat{\boldsymbol{I}}$ to denote the spatiotemporal constraints in the inference and $\hat{\boldsymbol{P}}$ to denote the spatiotemporal attentions. Correspondingly, $I_{\hat{\boldsymbol{I}}}$ and $P_{\hat{\boldsymbol{P}}}$ represent the inference algorithm with spatiotemporal constraints and the prediction algorithm with spatiotemporal attention. Therefore, the Eq. 4 changes to:

$$P_{\hat{\boldsymbol{P}}}(I_{\hat{\boldsymbol{I}}}(\boldsymbol{Y}'_n)) = P_{\hat{\boldsymbol{P}}}(\hat{\boldsymbol{Y}}_n) = \hat{\boldsymbol{y}}_{n+1} \approx \boldsymbol{y}_{n+1}. \tag{6}$$

After a round of inference and prediction, we obtain the spatiotemporal constraints $\hat{\boldsymbol{I}}$ and the spatiotemporal correlations $\hat{\boldsymbol{P}}$. We can use the specific spatiotemporal correlations captured from the complete data to modify the general spatiotemporal correlations that constrain sparse data. Specifically, we use this correlations matrix $P_{\hat{\boldsymbol{P}}}$ to

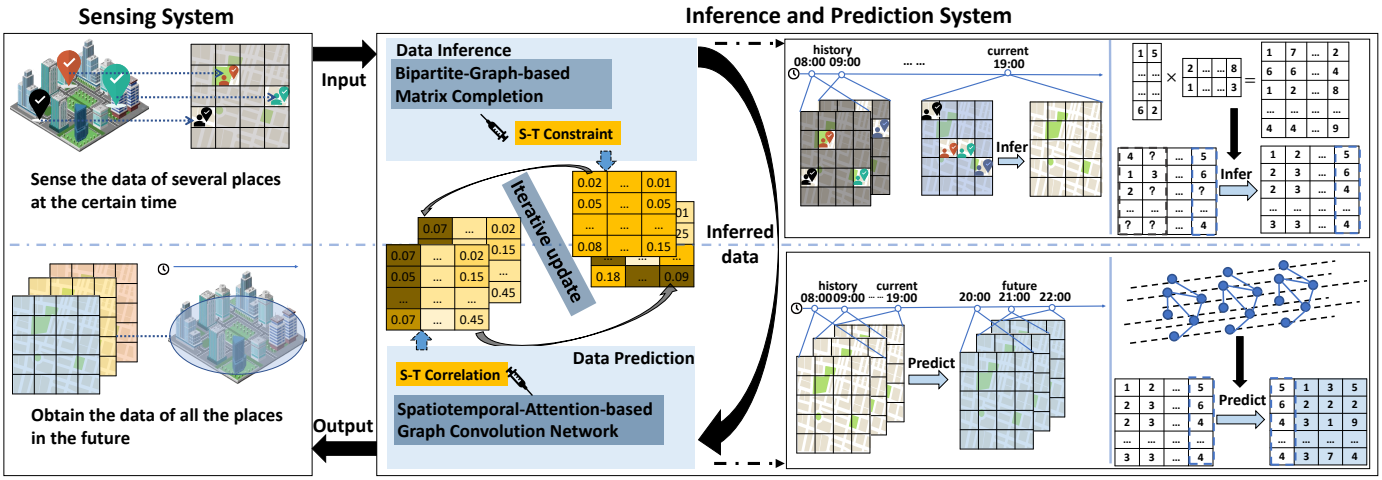


Fig. 2: The framework of urban inference and prediction: we collect sparse sensing data from the city and infer the full sensing data through the matrix completion. Then, we train a GCN model to predict the sensing data in the future of the city. In addition, we use an iterative update mechanism to strengthen the spatiotemporal correlations between inference and prediction system.

modify the constraints matrix $I_{\hat{I}}$ in order to improve the accuracy of the inference.

$$\hat{\boldsymbol{I}}_{i+1}, \hat{\boldsymbol{P}}_{i+1} = F(\hat{\boldsymbol{I}}_i, \hat{\boldsymbol{P}}_i), \tag{7}$$

where the F denotes the iterative algorithm. The new inference data constrained by modified matrix $\hat{\boldsymbol{I}}_{i+1}$ also has influence on the $\hat{\boldsymbol{P}}_i$ in prediction module and also improves the accuracy of the prediction, then the influenced $\hat{\boldsymbol{P}}_{i+1}$ modifies the inference module again. And we iterate this process until the satisfactory accuracy is achieved. After that, we continue to predict the next cycle $\hat{\boldsymbol{y}}_{n+2}$ with the $\boldsymbol{Y}'_{n+1}$. We denote the $P_{\hat{\boldsymbol{P}}}(I_{\hat{I}}(\boldsymbol{Y}'_n), k) = \hat{\boldsymbol{y}}_{n+k}$ as the prediction in the next k cycles.

3.1.1 Problem Formulation

Problem [Urban Prediction via Sparse MCS] : Given an MCS task with m subareas and n sensing cycles, for each cycle, we can sense data from quite a few subareas, and then use the sparse data to predict the full map of the k near future cycles, with the objective of minimizing the prediction errors:

$$\min \quad \sum_{j=1}^{n-k} \varepsilon(\boldsymbol{y}_{j+k}, \hat{\boldsymbol{y}}_{j+k}), \tag{8}$$

$$\text{s.t.} \quad P_{\hat{\boldsymbol{P}}}(I_{\hat{\boldsymbol{I}}}(\boldsymbol{Y}'_n), k) = \hat{\boldsymbol{y}}_{j+k}, j = 1, 2, ..., n-k. \tag{9}$$

We now provide a running example to describe the urban prediction problem in more detail, as shown in Fig. 2. Suppose that we have an urban sensing task that needs to sense data from a target area from 8:00 to 20:00. To provide the fine-grained results, we divide the target sensing area into 5×4 subareas and collect sensing data from these subareas every one hour (upper of the Sensing system shown in Fig. 2). To reduce the costs and deal with the unavailable subareas, for each sensing cycle, only a few subareas will be sensed, e.g., at 8:00, we only obtain the data sensed from 3 subareas. After 11 hours, the current time is 19:00, and we get the data sensed from 4 subareas in this sensing cycle. We can use a data inference algorithm to infer the unsensed data of the current sensing cycle (upper of the inference and prediction system shown in Fig. 2). Furthermore, we would like to predict the near future 20:00, 21:00 and 22:00 entire map based on the historical sparse sensed data (lower of the inference and prediction system shown in Fig. 2). In this paper, we first use a matrix completion algorithm with the spatiotemporal constraints to recover the historical complete matrix from the sparse sensed matrix, which provides accurate and sufficient data for prediction. Then, with the inferred matrix, we use a prediction method to learn the spatiotemporal correlations for predicting the near-future full map. Finally, we utilize the spatiotemporal correlations of both modules to iteratively modify each other in order to improve the accuracy.

4 Data Inference by Spatiotemporal Graph-Based Completion

4.1 Spatiotemporal Matrix Factorization

The urban sensing is actually to collect the various readings from the urban regions. Note that most of the sensing data are continuous in the physical world, which generally exhibits strong spatiotemporal correlations, and thus the complete sensing matrix $\boldsymbol{Y}$ usually has the low-rank property. Given an incompletely and randomly (or even sparse) sensed data matrix $\boldsymbol{Y}'$, we would like to recover the full sensing matrix $\hat{\boldsymbol{Y}}$ based on the low-rank property:

$$\min \ \ rank(\hat{\boldsymbol{Y}}), \tag{10}$$

$$\text{s.t.,} \ \ \hat{\boldsymbol{Y}} \bullet \boldsymbol{C} = \boldsymbol{Y}'. \tag{11}$$

Note that the above optimization is nonconvex, so we can hardly solve it directly. Given the complete sensing matrix $\boldsymbol{Y}$ with rank k [1], i.e., $rank(\boldsymbol{Y}) = rank(\hat{\boldsymbol{Y}}) = k$, we can factor our inferred matrix $\hat{Y}$ into the product of a spatial factor matrix $\boldsymbol{L}_{m \times k}$ and a temporal factor matrix $\boldsymbol{R}_{n \times k}$, as shown in Fig. 3 (left part), in order to capture the low-rank feature

1. k is a property of $\boldsymbol{Y}$, but in practice, we cannot obtain the complete matrix $\boldsymbol{Y}$ and have to collect it for some sensing cycles to estimate the initial rank k and readjust it according to the recovery errors during the sensing campaign.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3195706

and change the above rank optimization problem with constraints to the error minimization problem for missing data recovery:

$$\min \ \|(\boldsymbol{Y}-\hat{\boldsymbol{Y}})\bullet\boldsymbol{C}\|_{\mathrm{F}}^2=\|(\boldsymbol{Y}'-\boldsymbol{L}\boldsymbol{R}^{\intercal}\bullet\boldsymbol{C}\|_{\mathrm{F}}^2, \tag{12}$$

$$\text{s.t.}, \ rank(\boldsymbol{Y})=rank(\hat{\boldsymbol{Y}})=k, \hat{\boldsymbol{Y}}=\boldsymbol{L}\boldsymbol{R}^{\intercal}, \tag{13}$$

where $\|\cdot\|_{\mathrm{F}}$ is the Frobenius norm and used to denote the error between the inferred matrix and the actually sensed matrix.

To obtain the optimal $\hat{\boldsymbol{Y}}$ from $\boldsymbol{Y}'$, many existing methods, such as the Alternating Least Squares [21], [33] can be used to train the two factors to solve the problem, i.e., $\hat{\boldsymbol{Y}}=\boldsymbol{L}\boldsymbol{R}^{\intercal}$ according to the Eq. 10. However, the Eq. 10 only focuses on the learning from the sensed data but ignores the spatiotemporal correlations existing in the unsensed data. Therefore, we further consider the important and naturally occurring correlations as the supplement and constraint for Eq. 10, and thus obtain the error minimization problem with spatiotemporal constraints as follows:

$$\min \ \|\boldsymbol{Y}'-\hat{\boldsymbol{Y}}\bullet\boldsymbol{C}\|_{\mathrm{F}}^2+\lambda_t\|\hat{\boldsymbol{Y}}\mathbb{T}^{\intercal}\|_{\mathrm{F}}^2+\lambda_s\|\mathbb{S}\hat{\boldsymbol{Y}}\|_{\mathrm{F}}^2, \tag{14}$$

where $\mathbb{T}$ and $\mathbb{S}$ are the temporal and spatial constraint matrices:

- $\mathbb{T}$ presents the temporal constraint among the sensing data of the same subarea between different sensing cycles. Intuitively, two continuously sensed data from the same subarea are usually similar. Thus, we choose a temporal constraint matrix $\mathbb{T}_{n\times n}=Toeplitz(0,1,-1)$ [34], [35] for Eq. 14 to constrain that two continuous data from one subarea are the same. Moreover, the prior domain knowledge and the sufficient historical data may provide more temporal constraints, such as the periodicity and statistical characteristics, which can be used to conduct a more sophisticated $\mathbb{T}$.
- $\mathbb{S}$ presents the spatial constraint among the sensing data of the same sensing cycle between different subareas. Similar with the temporal constraint, the data sensed from the closer subareas usually have the similar values. Thus, we use the Euclidean distance to characterize the spatial correlation, denoted as $\mathbb{S}_{m\times m}[i,j]$=exp$(-\text{distance}(i,j)/\sigma_s^2)$. Then, for each row i in $\mathbb{S}$, we normalize them as $\sum_{j=1,j\neq i}^{m}\mathbb{S}[i,j]=1$ and set $\mathbb{S}[i,i]=-1,\forall i=\{1,...,m\}$.

Note that the temporal and spatial constraint matrices $\mathbb{T}$ and $\mathbb{S}$ are used as a supplement to further constrain the unsensed data in the sensing matrix, which also preserves the spatiotemporal correlations among all the subareas and sensing cycles for the near-future prediction. Meanwhile, as shown in Eq. 14, we can use the weighted parameters λ_t and λ_s to balance the weights of different elements. Furthermore, many other correlations could be easily modified into our error minimization problem.

4.2 Graph-based Matrix Completion

To conduct the matrix completion, we first consider the relationships between the spatiotemporal factors and the inferred results. As discussed above, we obtain that the inferred matrix $\hat{\boldsymbol{Y}}=\boldsymbol{L}\boldsymbol{R}^{\intercal}$. Specifically, as shown in

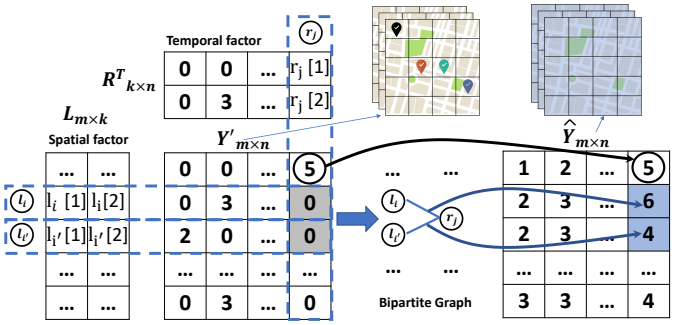


Fig. 3: Matrix factorization with bipartite graph.

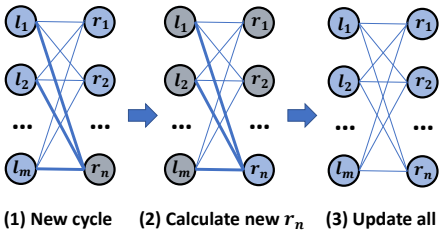


Fig. 4: Matrix completion based on bipartite graph.

Fig. 3 (right part), let $\boldsymbol{L}\triangleq\{\boldsymbol{l}_1^{\intercal},\boldsymbol{l}_2^{\intercal},...,\boldsymbol{l}_m^{\intercal}\}^{\intercal}$ and $\boldsymbol{R}\triangleq\{\boldsymbol{r}_1,\boldsymbol{r}_2,...,\boldsymbol{r}_m\}^{\intercal}$, we have $\hat{\boldsymbol{Y}}[i,j]=\boldsymbol{l}_i\boldsymbol{r}_j^{\intercal}=\sum_{c=1}^{k}\boldsymbol{l}_i[c]\boldsymbol{r}_j[c]$. Actually, the elements in the inferred matrix $\hat{\boldsymbol{Y}}$ can be seen as the linear combination of the elements in spatiotemporal factors $\boldsymbol{L}$ and $\boldsymbol{R}$. As an example, suppose that we obtain the rank $k=2$ and the well trained spatiotemporal factors $\boldsymbol{L}$ and $\boldsymbol{R}$, as shown in Fig. 3, the unsensed data $\boldsymbol{Y}'[i,j]=0$ can be inferred as follows:

$$\begin{aligned}\hat{\boldsymbol{Y}}[i,j]&=\boldsymbol{l}_i\boldsymbol{r}_j^{\intercal}=\boldsymbol{l}_i[1]\boldsymbol{r}_j[1]+\boldsymbol{l}_i[2]\boldsymbol{r}_j[2],\\ \hat{\boldsymbol{Y}}[i',j]&=\boldsymbol{l}_{i'}\boldsymbol{r}_j^{\intercal}=\boldsymbol{l}_{i'}[1]\boldsymbol{r}_j[1]+\boldsymbol{l}_{i'}[2]\boldsymbol{r}_j[2].\end{aligned} \tag{15}$$

Thus, we conduct the bipartite graph that consists of $\boldsymbol{l}_i$ and $\boldsymbol{r}_j$ as the vertexes and $\hat{\boldsymbol{Y}}[i,j]$ as the edges, where $i\in[1,m]$ and $j\in[1,n]$. Then, the matrix completion can be seen as the linear calculations of $\boldsymbol{l}_i$ and $\boldsymbol{r}_j$ to obtain $\hat{\boldsymbol{Y}}[i,j]$.

After understanding the relationships between the factors and the inferred matrix, we further consider how to train the spatiotemporal factors $\boldsymbol{L}$ and $\boldsymbol{R}$. Based on the error minimization problem in Eqs. 11 and 12, we have $\boldsymbol{Y}'=\boldsymbol{L}\boldsymbol{R}^{\intercal}\bullet\boldsymbol{C}$. Go back to the example in Fig. 3 (left part), we have

$$\boldsymbol{Y}'[1,n]=\boldsymbol{l}_1[1]\boldsymbol{r}_n[1]+\boldsymbol{l}_1[2]\boldsymbol{r}_n[2]=5. \tag{16}$$

Consider that in the first $(n-1)$ sensing cycles, we already calculate the suitable $\boldsymbol{L}_{m\times k}$ and $\boldsymbol{R}_{(n-1)\times k}$. For the current n-th sensing cycle, we know the sensed data $\boldsymbol{Y}'[1,n]$ and the spatial factor $\boldsymbol{l}_1$, and then we can use the linear Eq. 16 to calculate the unknown temporal factor $\boldsymbol{r}_n$. With the well-trained $\boldsymbol{R}=\{\boldsymbol{R}_{(n-1)\times k},\boldsymbol{r}_n\}$ and $\boldsymbol{L}$, we can recover the unsensed data, e.g., the $\hat{\boldsymbol{Y}}[i,j]$ and $\hat{\boldsymbol{Y}}[i',j]$ in our example, through the linear calculations.

The detailed graph-based matrix completion algorithm is summarized in Alg. 1 with an example shown in Fig. 4. For the current n-th, we have already held the well-trained temporal and spatial factors $\boldsymbol{L}_{m\times k}$ and $\boldsymbol{R}_{(n-1)\times k}$ for the historical sensing matrix. We first build the linear system as Eq. 16 (line 2) to calculate the newly added temporal factor

Algorithm 1 Graph-based Matrix Completion

Input: $\boldsymbol{Y}'_{m\times n} = \{\boldsymbol{Y}'_{m\times(n-1)}, \boldsymbol{y}'^{\intercal}_n\}$, $\boldsymbol{L}_{m\times k} = \{\boldsymbol{l}_1, \boldsymbol{l}_2, ..., \boldsymbol{l}_m\}$, $\boldsymbol{R}_{(n-1)\times k} = \{\boldsymbol{r}_1, \boldsymbol{r}_2, ..., \boldsymbol{r}_{n-1}\}$

Output: $\hat{\boldsymbol{Y}}_{m\times n}$

1: Init $\boldsymbol{r}_n$, $\boldsymbol{R}_{n\times k} = \{\boldsymbol{R}_{(n-1)\times k}, \boldsymbol{r}_n\}$, $count = 0$;
2: Build the linear system by using $\boldsymbol{y}'_n$, $\boldsymbol{L}_{m\times k}$, and $\boldsymbol{R}_{(n-1)\times k}$, and then calculate $\boldsymbol{r}_n$;
3: **while** not convergent **and** $count <$ MAX_ITER **do**
4: Fix $\boldsymbol{R}_{n\times k}$ and treat $\boldsymbol{L}_{m\times k}$ as unknown, build the linear system by using $\boldsymbol{Y}'_{m\times n}$, $\boldsymbol{L}_{m\times k}$ and $\boldsymbol{R}_{n\times k}$, and then calculate and update $\boldsymbol{L}_{m\times k}$;
5: Fix $\boldsymbol{L}_{m\times k}$ and treat $\boldsymbol{R}_{n\times k}$ as unknown, build the linear system by using $\boldsymbol{Y}'_{m\times n}$, $\boldsymbol{L}_{m\times k}$ and $\boldsymbol{R}_{n\times k}$, and then calculate and update $\boldsymbol{R}_{n\times k}$, $\hat{\boldsymbol{Y}} = \boldsymbol{L}\boldsymbol{R}^T$, and $count$++;
6: **return** $\hat{\boldsymbol{Y}}$.

$\boldsymbol{r}_n$. Note that the current factors $\boldsymbol{L}_{m\times k}$ and $\boldsymbol{R}_{(n-1)\times k}$ only hold the temporal and spatial information learned from the historical matrix $\boldsymbol{Y}'_{m\times(n-1)}$, we then iteratively train and update the factor $\boldsymbol{L}$ or $\boldsymbol{R}$ by using an Alternating Least Squares [10], [12], [21], [33] while keeping the others fixed (lines 3-5), until the inferred $\hat{\boldsymbol{Y}}$ is convergent or the maximum number of iterations is reached. Finally, the graph-based matrix completion algorithm outputs the complete inferred matrix $\hat{\boldsymbol{Y}}$, which further provides sufficient data with effective spatiotemporal correlations for the near-future prediction in the next section.

5 Data Prediction by Spatiotemporal Graph Convolutional Networks

In common with the inference part, we still need to use the strong spatiotemporal correlations among sensing data to predict the near future. For this problem, many methods have been proposed. With the rise of big data and deep learning in recent years, the CNN is considered to be an efficient prediction method for capturing the feature relationship between data [36], [37]. It should be noted deep learning can also be applied to the inference part. However, in this work, due to the sparse data(the sparse ratio is 0.1 or even 0.05) in the inference, many unsensed data will lead to the inability to extract effective spatiotemporal correlations in the process of deep learning training. While our inference method makes use of the structural correlation of the data itself, such as the low-rank attribute of the matrix, added with the general spatiotemporal constraints, to enrich the correlations among the data in the inference stage. Compared with using neural networks to learn the correlations among data from sparse data for prediction, our method performs better. And we also proved this conclusion in the following related experiments. While in the prediction part, we are faced with complete data after inference, neural networks can capture enough data correlations from the complete sensing data, not only that, we also extract the spatiotemporal correlations specifically to improve the accuracy. Therefore, we apply the algorithm based on matrix completion method in the inference part, while using the neural network model based on the deep learning in the prediction part.

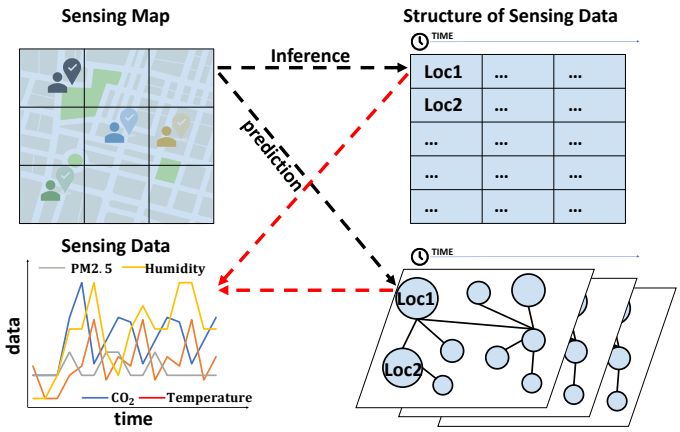


Fig. 5: Graph transformation: the data collected from the sensing map are represented by matrix in the inference and transformed into graph in the prediction.

5.1 Overall Structure

5.1.1 Graph transformation

The traditional CNN usually faces structured grid data based on its working principle. However, in real life, many data are not stored structurally, but unstructured like a graph, such as chemical molecular formula, road network flow and so on. The graph structure can not only preserve the data well, but also preserve the correlations among the data. For example, the road network can show the traffic flow and connectivity of each section. Therefore, in the prediction, in order to obtain more exact spatiotemporal correlations and improve the accuracy of prediction, we use graph structure to represent sensing map. As shown in Fig. 5, let the undirected graph $G = (V, E)$ denote the sensing map, where V is a set of $|V| = m$ nodes denotes the subareas in the sensing map and E is a set of edges indicating the connectivity [2] between the nodes. Each node on the graph holds the data collected by the user at that point or the data inferred subsequently (shown in Fig. 5). That means we can use the $\hat{\boldsymbol{Y}}_n = \{\hat{\boldsymbol{y}}^1_n, ..., \hat{\boldsymbol{y}}^m_n\}$ of the inference part to denote the data of all nodes at n-th sensing cycle and the $\hat{\boldsymbol{y}}^i_n$ represents the data of node i at n-th sensing cycle. Correspondingly, we denote the data of all nodes over n sensing cycles as $\mathcal{Y} = \{\hat{\boldsymbol{Y}}_1, ..., \hat{\boldsymbol{Y}}_n\}$.

5.1.2 Graph Convolutional Network

Although graph structure can provide more correlations among data, its topology is also more complex, which is difficult to deal with by traditional neural network. Therefore, we need GCN to convolute the irregular graph data and extract feature correlation among data to predict. To extract spatiotemporal correlations, we propose a GCN model based on spatiotemporal attention as shown in Fig. 6. It consists of three identical and independent substructures that represent different time spans, and each substructure contains several convolution layers. It's reasonable to design different time spans because the data of current time can not only be predicted from the previous hours, periodic

2. Connectivity depends on the type of sensing task. It is set for traffic data according to the road network and for environment data according to their own settings flexibly.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3195706

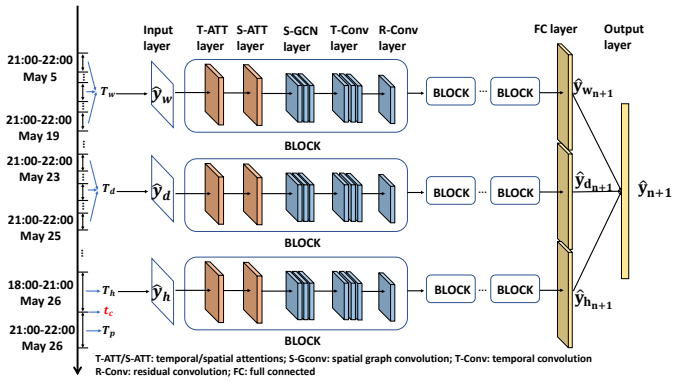


Fig. 6: Graph Convolutional Network with Spatiotemporal Attention.

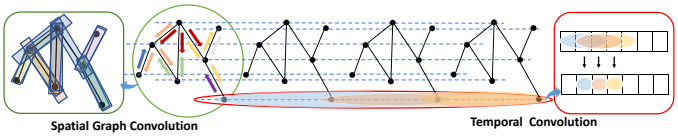


Fig. 7: Spatiotemporal convolution.

changes in a few days or weeks can also help prediction, like the number of vehicles in the morning peak or the number of people in the parks on weekends. In this model, we designed hour, day, and week time spans to explore time correlation. For input data, let $\hat{\mathcal{Y}}_h$, $\hat{\mathcal{Y}}_d$, $\hat{\mathcal{Y}}_w$ denote the hour, day, and week time span. We assume that the current time is t_c and the prediction time span is T_p. To predict the data in T_p, we get the corresponding time period on the whole historical time axis T_h, T_d, T_w. For example, suppose it's 21:00 and we want to predict the data in $21{:}00-22{:}00$ on May 26 with the sampling frequency of one hour. We intercept the data in $18{:}00-21{:}00$ on May 26 as $\hat{\mathcal{Y}}_h = \{\hat{\boldsymbol{Y}}_{18:00-19:00}, \hat{\boldsymbol{Y}}_{19:00-20:00}, \hat{\boldsymbol{Y}}_{20:00-21:00}\}$, similarly, the data in $21{:}00-22{:}00$ on May $23, 24, 25$ as $\hat{\mathcal{Y}}_d$ and the data in $21{:}00-22{:}00$ on May $5, 12, 19$ as $\hat{\mathcal{Y}}_w$. We show this interception process on the left of the network in the Fig. 6 and we use the following formula to explain in more detail how to use the data of the intercepted time period as the input $\hat{\mathcal{Y}}_h = \{\hat{\boldsymbol{Y}}_{T_{h_1}}, \hat{\boldsymbol{Y}}_{T_{h_2}}, ..., \hat{\boldsymbol{Y}}_{T_{h_n}}\}$, $\hat{\mathcal{Y}}_d = \{\hat{\boldsymbol{Y}}_{T_{d_1}}, \hat{\boldsymbol{Y}}_{T_{d_2}}, ..., \hat{\boldsymbol{Y}}_{T_{d_n}}\}$, $\hat{\mathcal{Y}}_w = \{\hat{\boldsymbol{Y}}_{T_{w_1}}, \hat{\boldsymbol{Y}}_{T_{w_2}}, ..., \hat{\boldsymbol{Y}}_{T_{w_n}}\}$. After we obtain the input data for each substructure, the data enters into several consecutive blocks, which contains spatiotemporal attention layer and convolution layer, and then the corresponding output is obtained through a full connection layer, finally we aggregate the three outputs $\hat{\boldsymbol{Y}}_{h_{n+1}}$, $\hat{\boldsymbol{Y}}_{d_{n+1}}$, $\hat{\boldsymbol{Y}}_{w_{n+1}}$ and get the final prediction data $\hat{\boldsymbol{Y}}_{n+1}$.

5.2 Spatiotemporal Convolution

When data enter into the convolution layer, we convolute the data from two dimensions of time and space. As shown in Fig. 7, in the spatial dimension, we use graph convolution to extract spatial features of nodes based on graph structure, in the temporal dimension, we use traditional convolution to extract temporal features of nodes based on time slots.

5.2.1 Spatial Convolution

Borrowed from Spectral Graph Theory [38], [39], we can extend convolution operation from matrix data to graph structure data. First, we regard the data in nodes as a signal. For signal processing, it's difficult to analyze in the time domain so we need to analyze the signal in the frequency domain by Fourier transform. [40]. We need to use the eigenvectors of Laplacian matrix as the basis of Fourier Transform. For signal in the graph, the Laplacian operator is equal to Laplacian matrix, so we utilize Laplacian matrix $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$ to represent the graph structure. $\boldsymbol{L}$ is Laplacian matrix of the graph structure, $\boldsymbol{D}$ is the Degree matrix and $\boldsymbol{A}$ is the Adjacency matrix. After $\boldsymbol{L}$ is obtained, we can use its eigenvector as the basis of the graph Fourier Transform to carry out the Fourier Transform of the graph signal. Since it's an undirected graph, then $\boldsymbol{L}$ is a symmetric matrix and certainly able to eigenvalue decompose as $\boldsymbol{L} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\intercal$, where $\boldsymbol{U}$ is a matrix in which each column is the eigenvector of $\boldsymbol{L}$ and $\boldsymbol{\Lambda}$ is a Diagonal matrix with the eigenvalue of $\boldsymbol{L}$. Then, we obtain the graph Fourier Transform $\hat{f} = \boldsymbol{U}^\intercal f$ and deduce the graph convolution formula.

$$(f * g)_G = \boldsymbol{U}(\boldsymbol{U}^\intercal f \odot \boldsymbol{U}^\intercal g). \tag{17}$$

If $\boldsymbol{U}^\intercal g$ is regarded as convolution kernel g_θ, the above formula can be expressed as

$$(f * g)_G = \boldsymbol{U} g_\theta \boldsymbol{U}^\intercal f. \tag{18}$$

Taking the signal $\hat{\boldsymbol{y}}_t$ of all nodes at t-th sensing cycle as an example, $\hat{\boldsymbol{y}}_t$ is filtered by a kernel g_θ:

$$(\hat{\boldsymbol{y}}_t * g_\theta)_G = \boldsymbol{U} g_\theta \boldsymbol{U}^\intercal \hat{\boldsymbol{y}}_t. \tag{19}$$

Due to the high cost of calculating all the eigenvectors and the eigenvalues, we use Chebyshev polynomials [41] to accelerate the solution of the eigenmatrix.

$$g_\theta = \sum_{k=0}^{k-1} \theta_k T_k(\hat{\boldsymbol{\Lambda}}), \tag{20}$$

$$\hat{\boldsymbol{\Lambda}} = \frac{2\Lambda}{\lambda_{max}} - \boldsymbol{I}_N, \tag{21}$$

where θ is polynomial coefficients, $T_(k)$ is the recursive function and its definition is $T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$, the function of $\hat{\boldsymbol{\Lambda}}$ is normalize eigenvector matrix to $[-1, 1]$, λ_{max} is the maximum eigenvalue of the Laplacian matrix and $\boldsymbol{I}_N$ is a unit matrix. Using the Chebyshev polynomials, the convolution changes to

$$(\hat{\boldsymbol{y}}_t * g_\theta)_G = \sum_{k=0}^{k-1} \theta_k T_k(\hat{\boldsymbol{\Lambda}}) \hat{y}_t. \tag{22}$$

After the node collects the neighbor information based on the convolution kernel, the Rectified Linear Unit(ReLU) activation function $ReLU((\hat{\boldsymbol{y}}_t * g_\theta)_G)$ is used to complete the design of the whole module.

5.2.2 Temporal Convolution

After spatial convolution, each node extract surrounding neighbor information in the spatial dimension. Then, we add a standard convolution layer to extract the time information between neighbor time slices. In order to obtain a wide range of temporal correlation, we design multiple spatiotemporal convolution layers. Assuming $\hat{\mathcal{Y}}_{h,l}$ represents the signal of all the nodes with the time span of hour and convolution operation to layer l, then the convolution of time dimension is

$$\hat{\mathcal{Y}}_{h,l+1} = ReLU(\mu \cdot ReLU(\hat{\mathcal{Y}}_{h,l} * g_\theta)_G), \tag{23}$$

where $\cdot$ denotes the traditional convolution operation and μ is the parameters of the convolution kernel. Like the spatial convolution, we choose ReLU as the activation function.

In brief, through multiple spatiotemporal convolution layers, we collect enough spatiotemporal dimension information from graph signals.

5.3 Submodule Aggregation

It's worth noting that our whole neural network consists of three submodules, each module outputs different dimension prediction results. We need to aggregate these three results [42], as follows

$$\hat{\boldsymbol{y}}_{n+1} = \boldsymbol{\phi}_h \odot \hat{\boldsymbol{y}}_{h_{n+1}} + \boldsymbol{\phi}_d \odot \hat{\boldsymbol{y}}_{d_{n+1}} + \boldsymbol{\phi}_w \odot \hat{\boldsymbol{y}}_{w_{n+1}}, \quad (24)$$

where $\boldsymbol{\phi}_h$, $bm\phi_d$, $\boldsymbol{\phi}_w$ are the learning parameters reflecting the influence degrees of the three submodules on the prediction target and $\hat{\boldsymbol{y}}_{h_{n+1}}$, $\hat{\boldsymbol{y}}_{d_{n+1}}$, $\hat{\boldsymbol{y}}_{w_{n+1}}$ are the prediction results.

6 Iterative Update based on Spatiotemporal Correlations

In the data inference part, we use the spatiotemporal correlation extracted from the spatiotemporal constrain matrix to improve the accuracy of inference. Similarly, we also use the spatiotemporal correlation in the prediction part.

6.1 Spatiotemporal Attention

By adding spatiotemporal attention mechanism [43] to graph neural network, we can further extract the spatiotemporal correlation between data. Attention mechanism can be explained directly by human visual mechanism. For example, our visual system tends to pay attention to some information in the image and ignore the irrelevant information. In neural networks, such as image-caption [44], only some regions in the input image may be more relevant to the word of caption. The attention model allows the neural network model to dynamically focus on certain parts of the input that contribute to the execution of the current task. In our model, we focus on the spatial attention and temporal attention. More specifically, we focus on the places that need attention among all places, and the same is true for time.

6.1.1 Spatial Attention

For urban data, each location is affected by the surrounding location in different degrees in the spatial dimension. It means that we need to use the spatial attention to capture the dynamic spatial correlations between nodes in the graph. We take the spatial attention module in the day span as an example:

$$\boldsymbol{S} = \boldsymbol{V}_s \cdot \sigma((\hat{\mathcal{Y}}_{d,l-1}\boldsymbol{\Psi}_1)\boldsymbol{\Psi}_2(\boldsymbol{\Psi}_3\hat{\mathcal{Y}}_{d,l-1})^\intercal + \boldsymbol{b}_s), \quad (25)$$

$$Softmax(\boldsymbol{S}_{i,j}) = \frac{exp(\boldsymbol{S}_{i,j})}{\sum_{j=1}^{N} exp(\boldsymbol{S}_{i,j})}, \quad (26)$$

where $\mathcal{X}$ is the input data of the l^{th} spatiotemporal layer. $\boldsymbol{V}_s$, $\boldsymbol{b}_s$, $\boldsymbol{\Psi}_1$, $\boldsymbol{\Psi}_2$, $\boldsymbol{\Psi}_3$ are learnable parameters and sigmoid σ is the activation function. The value of an element $\boldsymbol{S}_{i,j}$ in $\boldsymbol{S}$ represents the correlation strength between node i and node j. Then we use a softmax function to ensure the weights of a node sum to one. With the connectivity of nodes provided by adjacency matrix $\boldsymbol{A}$, we can use spatial attention matrix to capture the dynamic influence between nodes. When we put the attention module into the GCN, the convolution changes to:

$$(\hat{\boldsymbol{y}}_t * g_\theta)_G = \sum_{k=0}^{k-1} \theta_k T_k(\hat{\boldsymbol{\Lambda}} \odot \boldsymbol{S})\hat{\boldsymbol{y}}_t. \quad (27)$$

6.1.2 Temporal Attention

The urban data in each location is also affected by the surrounding time slices in different degrees in the temporal dimension. We use the temporal attention to capture the temporal correlations between time slices. As same as the spatial attention, the temporal is:

$$\boldsymbol{E} = \boldsymbol{V}_e \cdot \sigma(((\hat{\mathcal{Y}}_{d,l-1})^\intercal\boldsymbol{\Omega}_1)\boldsymbol{\Omega}_2(\boldsymbol{\Omega}_3\hat{\mathcal{Y}}_{d,l-1}) + \boldsymbol{b}_e), \quad (28)$$

$$Softmax(\boldsymbol{E}_{i,j}) = \frac{exp(\boldsymbol{E}_{i,j})}{\sum_{j=1}^{T} exp(\boldsymbol{E}_{i,j})}, \quad (29)$$

where $\boldsymbol{V}_e$, $\boldsymbol{b}_e$, $\boldsymbol{\Omega}_1$, $\boldsymbol{\Omega}_2$, $\boldsymbol{\Omega}_3$ are learnable parameters and sigmoid σ is the activation function. The value of an element $\boldsymbol{E}_{i,j}$ in $\boldsymbol{E}$ represents the correlation strength between time slice i and time slice j. $\boldsymbol{E}$ is normalized by the softmax function like the spatial attention. When we put the attention module into the convolution neural network, the input of convolution layer changes to: $\hat{\mathcal{Y}}'_{d,l-1} = \hat{\mathcal{Y}}_{d,l-1}\boldsymbol{E}$.

6.2 Spatiotemporal Iteration

Now that we've used spatiotemporal correlations in both two parts, we need a way to connect them to provide more accurate spatiotemporal correlations. In fact, the spatiotemporal correlation in data inference is more inclined to the realistic level, like the distance of the place and the size of the time interval, while the spatiotemporal correlation in data prediction is more inclined to the data level. More specifically, assuming that there are two locations A and B and we want to infer the temperature of location B from A. If A and B are far away and due to the climate, the temperature of A and B are similar. In this case, we can infer that the temperature of B is similar to A by using the spatial correlations extracted at the data level, while the spatial correlations extracted from the real locations are likely to have a large error due to the long distance. It means that we can make use of flexible correlations at the data level to make up for the tough constraints at the realistic level to make the inferred data more accurate. Therefore, we propose an iterative approach, using the spatiotemporal correlation in data prediction to modify the correlation in data inference iteratively as shown in Fig. 8.

In this way, the tough constraints of the inference part will become more flexible and accurate after several rounds of iteration, and can also react to the flexible correlations of the prediction part to make it more accurate. We use the following formulas to express this iterative relationship:

$$\hat{\boldsymbol{I}}_{i+1} = \lambda_{inf}\hat{\boldsymbol{I}}_i + \lambda_{pre}\hat{\boldsymbol{P}}_i, \quad (30)$$

$$\hat{\boldsymbol{P}}_{i+1} = I(\hat{\boldsymbol{I}}_{i+1}), \quad (31)$$

TABLE 1: Statistics of four evaluation data sets

Data Type	*Urban environment*		*Urban traffic*	
Datasets	***Sensor-Scope***	***U-air***	***TaxiSpeed***	***Traffic Volume Viewer***
City	Lausanne(Switzerland)	Beijing(China)	Beijing(China)	NSW(Australia)
Data	Humidity	PM2.5	Traffic Speed	Traffic flow
Subareas	57 subareas ($50 * 30m^2$)	36 subareas ($1000 * 1000m^2$)	100 roads as subareas	30 checkpoints as subareas
Cycle & Duration	$0.5h$ & $7d$	$1h$ & $11d$	$1h$ & $4d$	$1d$ & $1y$
Mean ± Std.	$84.52 \pm 6.32(\%)$	$79.11 \pm 81.21(\mu g/m^3)$	$13.01 \pm 6.97(m/s)$	$19095.73 \pm 26750.79(n)$

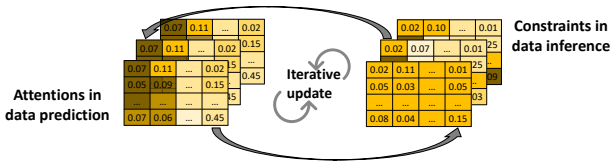


Fig. 8: Spatiotemporal matrix iteration: the spatiotemporal correlations matrix is updated with the iteration cycles and different colors represent different intensities of spatiotemporal constraints and spatiotemporal attentions.

where $\hat{\boldsymbol{I}}_{\boldsymbol{i}}$ is the constraint matrix in the inference part, $\hat{\boldsymbol{P}}_{\boldsymbol{i}}$ is the attention matrix in the prediction part, λ_{inf}, λ_{pre} are the weighted parameters and the I represents the inference algorithm. It means that new inferred data will be generated after the inference algorithm with updated spatiotemporal constraints matrix and the spatiotemporal attention matrix captured from the new data in the prediction part will also be update. The iterative process will continue until the the satisfactory prediction result $\hat{\boldsymbol{y}}_{n+1}$ is obtained. Then we predict the next sensing cycle $\hat{\boldsymbol{y}}_{n+2}$ with the new sensing data $\boldsymbol{Y}'_{n+1}$. Finally, we will get the prediction of multiple sensing cycles in the future under the sparse urban data.

7 Performance Evaluation

7.1 Data Sets

To truly evaluate our proposed urban prediction scheme, we introduce four data sets, two of which are Sensor-Scope and U-Air under the urban environment sensing, and the other two are TaxiSpeed and Traffic Volume Viewer under the urban traffic sensing. We show the main information in the table 1 with more details as follows:

- The **Sensor-Scope** is a dataset [3] of environment information, which includes temperature, humidity, and other variables. We choose the humidity dataset for performance evaluation. This dataset contains 57×336 data collected from static sensors deployed on the EPFL campus. It monitors the humidity variation in 57 areas for 7 days, recording the average value every half hour as the real value.
- The **U-Air** is an air quality dataset [4], which includes $PM2.5$, SO_2, and other variables. We choose the $PM2.5$ dataset for performance evaluation. This dataset contains 36×264 data collected from monitor stations deployed in Beijing, China. It monitors the PM2.5 variation in 36 areas for 7 days, recording the average value every half hour as the real value.
- The **TaxiSpeed** is a dataset [5] of vehicle mobility information. This dataset contains 100×96 data collected from GPS devices deployed on taxis. It monitors the speed variation in 100 road segments for 7 days, recording the average value every half hour as the real value.
- The **Traffic Volume Viewer** is a dataset [6] of traffic flow information. Data in this dataset is collected through sensors deployed at traffic collection stations in NSW, Australia. The Traffic Volume Viewer monitors the traffic flow at over 60 stations since 2006. We select traffic flow data every day at 30 locations in 2018 for performance evaluation.

In the above datasets, The U-air and The Traffic Volume Viewer, as the representatives of the urban environment and traffic, have the characteristics of large data fluctuation, so they can well measure the performance of inference and prediction in our model. And the data in The Sensor-Scope, The TaxiSpeed are closely related in time and space so that they can well measure the influence of the spatiotemporal constraints in our model. For all datasets, we set certain data as unsensed in inference experiment based on sense ratio. The training set accounts for 60% of prediction experiment, whereas the validation set and test set each account for 20%.

Besides, it's worth noting that for all the data by the static sensors, we can use mobile devices or employee works to collect the same data from the subareas. Moreover, the data in these datasets are needed in typical data acquisition tasks, which is of great significance for data inference and prediction. So we choose these datasets to verify the effectiveness of our model.

7.2 Comparison Algorithms

7.2.1 Data Inference

In order to effectively utilize the sparse sensed data to conduct the inference and prediction, we present the Bipartite-Graph based Matrix Completion algorithm with spatiotemporal constraints (BGMC-st). Since the subarea selection is not the main point of this work, we randomly sense some subareas in each sensing cycle, and send the sensed data to data inference algorithms to recover the unsensed data. We mainly compare it with the following algorithms:

- **BGMC** is an algorithm with the same main part as BGMC-st, but there are no spatiotemporal constraints.

3. http://sensorscope.epfl.ch/network_code
4. https://www.microsoft.com/en-us/research/project/urban-computing/
5. https://www.microsoft.com/en-us/research/publication/inferring-gas-consumption-and-pollution-emission-of-vehicles-throughout-a-city
6. https://www.rms.nsw.gov.au/about/corporate-publications/statistics/traffic-volumes/aadt-map/

- **KNN-S** and **KNN-T** are modified from the famous KNN algorithm, using the weighted average of k data sensed from the nearest subareas (spatial/temporal correlations) as the inferred value of the current subarea in each sensing cycle.
- **GP** is the Gaussian process, it assumes that the spatial distribution of data in the same cycle obeys the Gaussian distribution. The unknown data is inferred by calculating the expectation and variance of known ones.
- **DMF** is the traditional Matrix factorization with deep learning multi-layer network, which can decompose a matrix $\boldsymbol{X}$ with missing values into two (or more) matrices. And the result of multiplying these decomposed matrices $\boldsymbol{X}'$ is the approximation of the original matrix [32].

7.2.2 Data Prediction

Based on the inference algorithm, we obtain the recovered sensing matrix, then we conduct the Spatiotemporal-Attention based Graph Convolution Network (AGCN-st) to do the near-future prediction. Note that in this paper, we mainly focus on predicting the near future sensing map from the sparse sensed data, which is such a difficult scenario that most of the existing works on urban prediction cannot work well, and all prediction comparison algorithms use the same sensing matrix recovered by the BGMC-st as the input to ensure fairness. We thus mainly compare it with the following algorithms:

- **GCN-st** is an algorithm with the same main part as AGCN-st, but there are no spatiotemporal attentions.
- **LSTM** is the classic Long Short-Term Memory method, which has the temporal recurrent structure to capture the temporal relationships among data.
- **NAR** is the Non-linear Autoregressive Neural Network, which is composed of autoregressive model and neural network to effectively extract nonlinear features for prediction [32].
- **NAR-SDAE** uses the NAR to exploit the temporal correlation between the sensed data and a Stacked Denoised AutoEncoder to utilize the spatial correlation.
- **LINEAR** uses the Linear Regression model to predict the near-future full map. It assumes that the future results are linearly related to the historical sensed data.
- **WNN** combines Wavelet transform and Neural Network. WNN is similar to STFNets, which is also good at extracting periodic features of time series.

7.3 Evaluation Results: Data Inference

7.3.1 Sense ratio

In the aspect of data inference, we first test the performance of the algorithm under the sensed ratio indicator. We set the index from 0.1 to 0.5, which means that only 10% to 50% of subareas can be sensed at each sensing cycle. As shown in Fig. 9, we can clearly see that the inference error decreases with the increase of sensing ratio, because more sensing subareas are collected and the spatiotemporal correlations contained in the data also increases. With the help of spatiotemporal constraint matrix, our method BGMC-st achieves the best performance under almost every indicator compared with other methods. It is because the matrix completion method makes up for the lack of the correlations among sparse data by using the low-rank attribute of the data itself and the added spatiotemporal constraint matrix. Meanwhile, based on the same inference algorithm, the BGMC has slightly worse performance than the BGMC-st without spatiotemporal constraints. For KNN, some data are more affected by spatial factors while some are more affected by temporal factors, so the performance of KNN-T and KNN-S is not consistent. For example, the Humidity is greatly affected by time, so the KNN-T achieves better results than KNN-s, while the KNN-S has better performance for Traffic Flow. In addition, as a matrix completion algorithm based on neural networks, the performance of DMF is not as good as our method. It also proves our view that our method is better than neural networks when it is difficult to extract data correlations from sparse data.

7.3.2 Sensed subareas

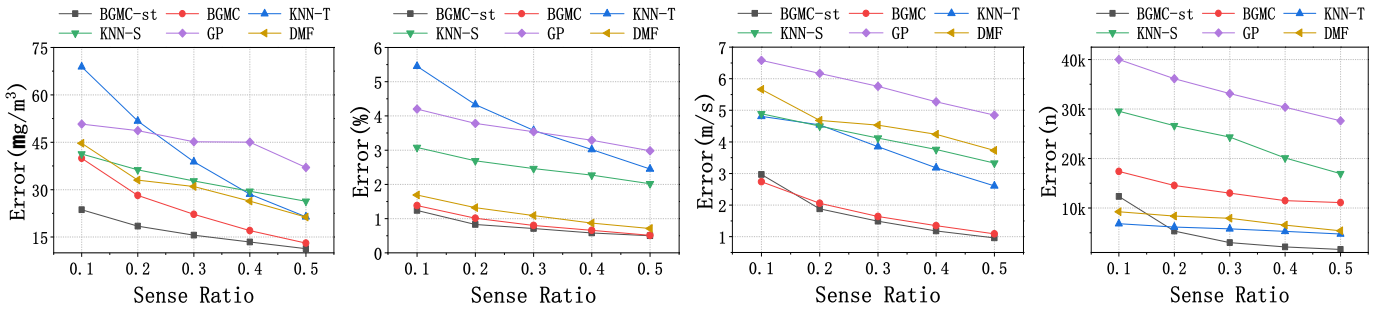
Furthermore, we measure the performance of the inference algorithm by the number of sensing subareas. We test how many sensing subareas data need to be collected to achieve a given certain error by all algorithms. As shown in Fig. 10, the result is the same as the sensed ratio indicator. The BGMC-st needs the least subareas and the BGMC sense more subareas than the BGMC-st. Conversely, other methods need more subareas. Both the sensed ratio and sensed subareas verify the effectiveness of BGMC-st.

It is worth noting that for the Error indicator, our method can still use fewer subareas to achieve the goal when the error is low. However, due to the reason that other methods need too many subareas under low error, the experiment is of little significance. Therefore, we only select two suitable error indicators to measure the experimental results.

7.3.3 Spatiotemporal ratios

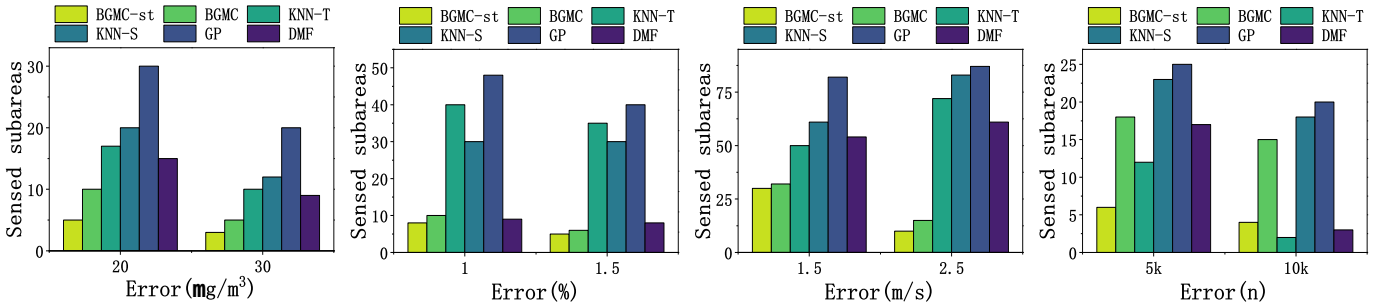
Through the above experiments, we find that the spatiotemporal influence is different on different data sets, so we should also control the spatiotemporal ratio when imposing spatiotemporal constraints. We test all data sets to find the best spatiotemporal constraint ratio for each data set. We denote the spatial weight and temporal weight as λ_s and λ_t, and $\lambda_t = 1 - \lambda_s$. As shown in Fig. 11, it is obvious that the spatial weight plays a more important role in the urban environmental data and the temporal weight is more important in the urban traffic data. Because the traffic data itself has stronger spatial attributes and relatively weak temporal attributes, the data inference accuracy can be improved by strengthening the temporal constraints. Similarly, the spatial of the environmental data is weaker than temporal so that the inference algorithm has better performance in the high spatial weight. All other experiments on inference in this paper are carried out with the optimal spatiotemporal ratio. From the above experiments, we can see that for different sensing tasks, using different spatiotemporal ratios can achieve better results.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3195706



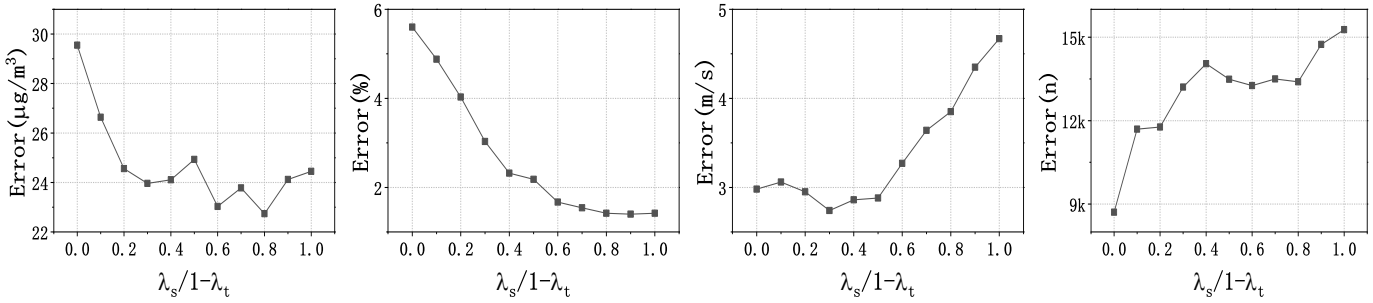
(a) PM2.5 (b) Humidity (c) Traffic Speed (d) Traffic Flow

Fig. 9: Inference accuracy under different sense ratios.



(a) PM2.5 (b) Humidity (c) Traffic Speed (d) Traffic Flow

Fig. 10: Inference accuracy under different sense subareas.



(a) PM2.5 (b) Humidity (c) Traffic Speed (d) Traffic Flow

Fig. 11: Inference accuracy under different spatiotemporal ratios.

7.4 Evaluation Results: Data Prediction

7.4.1 *Sense ratios*

In the aspect of data prediction, because our prediction method needs the entire data from inference, we firstly test the performance of the prediction algorithm under the sense ratio indicator. Like inference experiments, we set the sense ratio range from 0.1 to 0.5. As shown in Fig. 12, Fig. 13, with enough spatiotemporal information, our method obtained the lowest error in every sense ratio in all data sets. On the contrary, due to the lack of spatial correlation capture, LSTM is not good enough on each data set, especially for those data sets with important spatial features like Sensor-Scope and Traffic Volume Viewer. Interestingly, NAR has already got good performance on each data set, NAR-SDAE should have better results with the SDAE component modifying the spatial correlation, but in some data sets, excessive correction leads to bad results.

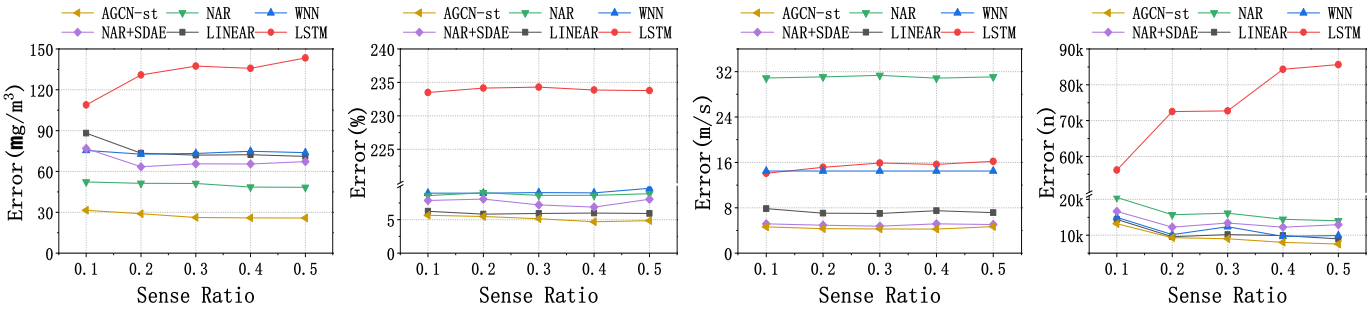
One more thing to note, due to the randomness of subareas selection, under low sense ratio, some subareas with strong spatiotemporal correlation may be selected at 0.1 sense ratio and not selected at 0.2 sense ratio, resulting in an error of 0.2 slightly higher than 0.1.

7.4.2 *Prediction cycles*

After testing the effect of the sense ratio in the inference algorithm on the prediction, we turn attention to the prediction algorithm itself. The above experiments only test the results of one prediction cycle, then we experiment the algorithm with the multi prediction cycles. We set the prediction cycle from 1 to 5 because we want the prediction results of the near future and we set sense ratio to 0.1. Our method still gets better performance than the others in every data set as shown in Figs. 14 and 15.

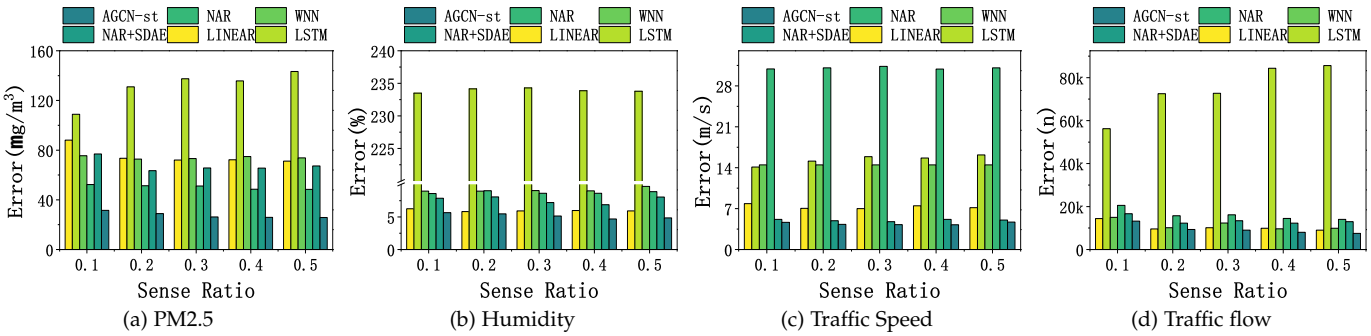
Moreover, with the sufficient spatiotemporal correlations captured by graph neural network and attention mechanism, the error of our method is relatively more stable than others in multiple prediction cycles. LINEAR got a

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3195706



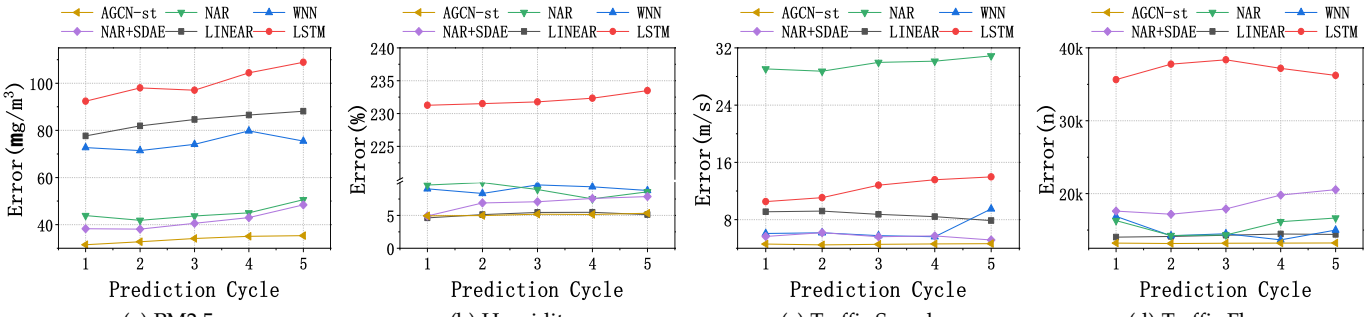
(a) PM2.5 (b) Humidity (c) Traffic Speed (d) Traffic Flow

Fig. 12: Prediction accuracy under different sensed ratios.



(a) PM2.5 (b) Humidity (c) Traffic Speed (d) Traffic flow

Fig. 13: Prediction accuracy under different sensed ratios.



(a) PM2.5 (b) Humidity (c) Traffic Speed (d) Traffic Flow

Fig. 14: Prediction accuracy under near future prediction cycles.

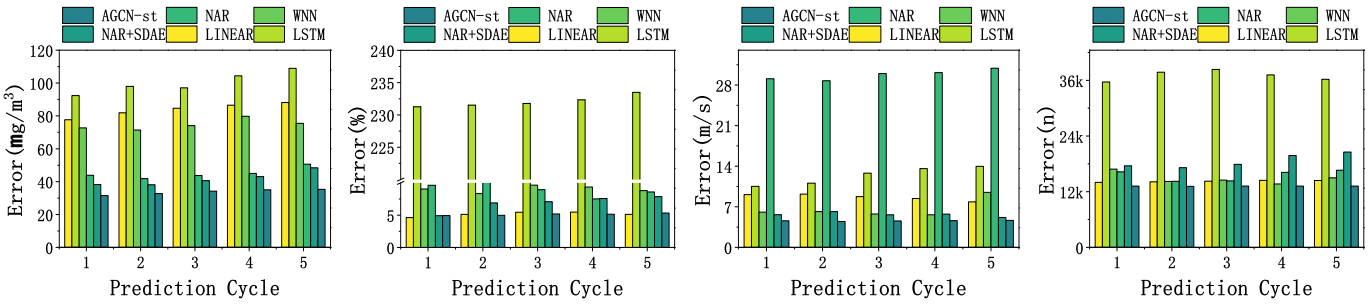
good performance at the data sets which fluctuate little and have linearity with the linear regressing. However, it is difficult to compare our method under the sensing data with complex spatiotemporal correlations. Similarly, although the NAR and NAR-SDAE got a fine result when the data sets are non-linear, they can not extract the correlations among the data contained in the graph structure, and there is no attention mechanism to capture these dynamic correlations, its performance is not as good as our method. These results prove that our method achieves the low error under the data sets of various characteristics.

7.4.3 *Spatiotemproal Attention*

Next, we evaluate the impact of the added attention mechanism on prediction in two data sets. As shown in Fig. 16, for PM2.5 data set, it is not only affected by spatiotemporal characteristics, wind direction, climate and other factors also account for a large proportion. Therefore, when only spatiotemporal attention is applied, the improvement of the algorithm is only about 4%. While for the traffic speed data set, it is mainly affected by spatiotemporal. When the attention mechanism is applied, the improvement of the algorithm is not small, up to nearly 20%. In short, the attention mechanism can effectively improve the accuracy of the algorithm, especially when the influence of spatiotemporal characteristics is very important.

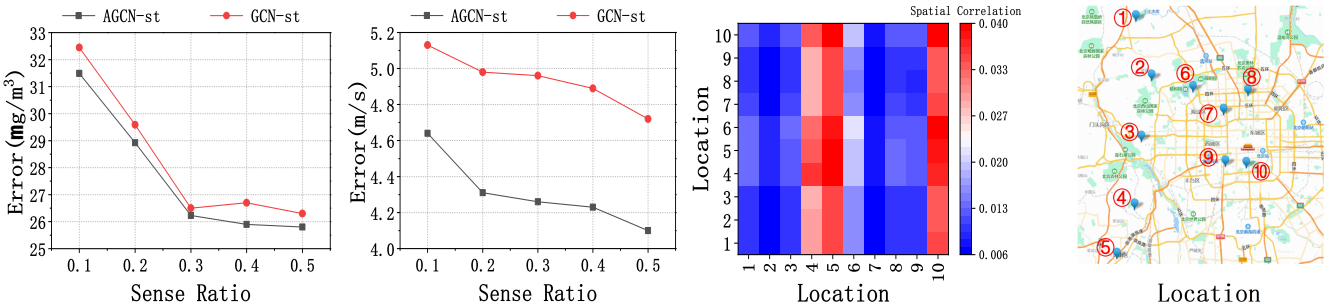
We also vividly show the attention matrix formed in the spatial dimension as shown in Fig. 16 (c, d). We present the real locations of ten subareas in the U-Air data set. The grid in row j of column i represents the spatial correlations between location i and location j. For example, in row 4, the colors of 5 and 10 are strong and close, while in the real map, 4 and 5 are close and 4 and 10 are green areas, so they have strong spatial correlations. And this also shows that our model is interpretable to some extent.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3195706



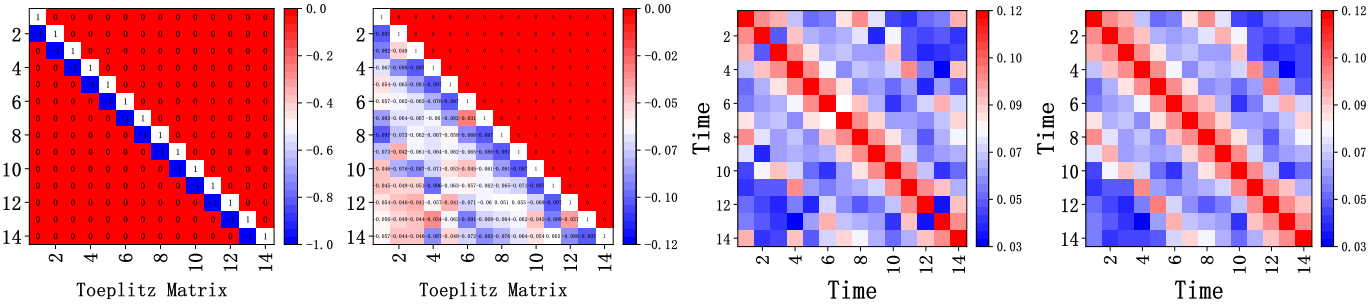
(a) PM2.5 (b) Humidity (c) Traffic Speed (d) Traffic Flow

Fig. 15: Prediction accuracy under near future prediction cycles.



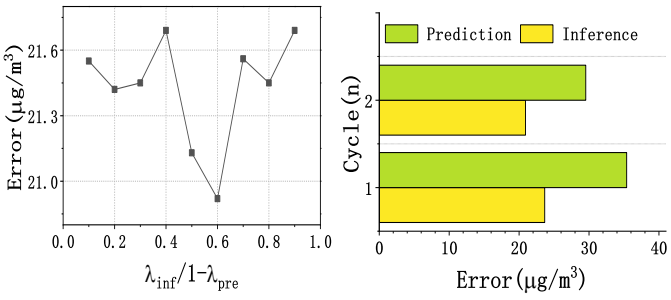
(a) PM2.5 (b) Traffic Speed (c) spatial attention (d) spatial location

Fig. 16: Attention mechanism over *U-Air* and *TrafficSpeed* and spatial attention over *U-Air*.



(a) Before Iteration (b) After Iteration (c) Before Iteration (d) After Iteration

Fig. 17: The spatiotemporal iteration process in the inference and prediction over *U-Air*.



(a) Iteration Ratio (b) Iteration Cycle

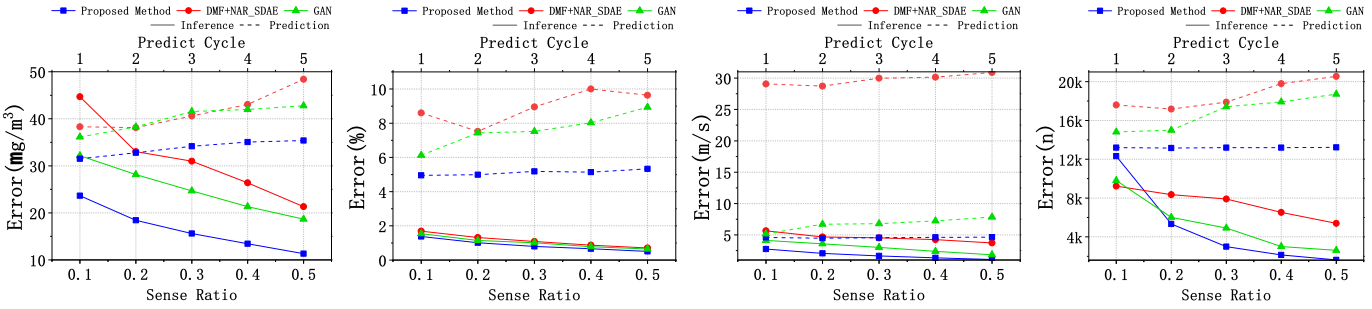
Fig. 18: Iteration ratio and cycle over *U-Air*.

7.5 Evaluation Results: Iterative Update

For the iteration module, we need to test the degree of iteration to see how much proportion is more appropriate to correct the spatiotemporal matrix of the inference part. We set the sense ratio as 0.1, the prediction cycle is 5 and set the optimal spatiotemporal ratio, after testing, we find that the ratio of 0.6 is the best correction ratio for U-Air data set. After the spatiotemporal iteration process, the spatiotemporal correlations matrix in the inference and prediction will be updated. As show in Fig. 17(a, b[7]), the spatiotemporal constraint matrix becomes more flexible. It no longer strictly constrains the data similarity of two adjacent rows, but constrains the data relationship among rows according to the spatiotemporal attentions matrix from the prediction. Similarly, as show in Fig. 17(c, d) because the updated constraint matrix can provide more accurate inferred data, the new attention matrix can capture more detailed and accurate temporal correlation. Therefore, the spatiotemporal information of the inference part and the prediction part has been corrected with higher accuracy,

7. The constraints of data correlations between any two rows need to be calculated only once, so the constraint matrix is the lower triangular matrix.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3195706



(a) PM2.5 (b) Humidity (c) Traffic Speed (d) Traffic Flow

Fig. 19: Inference and prediction accuracy under different frameworks.

and the experiment performance has also been improved as shown in Fig. 18. Regarding the spatiotemporal weight ratio within the updated matrix, we did not conduct a more detailed test, which is not our focus, and it will also make the whole system too cumbersome. Anyway, the iterative update module does improve the accuracy of the two parts.

7.6 Evaluation Results: Inference-prediction Framework

To demonstrate the superiority of the inference-prediction framework, which is connected based on spatiotemporal correlations, we conduct the experiments to evaluate the performance of inference and prediction under different inference-prediction frameworks. As shown in Fig. 19, the proposed method is our framework which has a complete inference algorithm with spatiotemporal constraints and prediction algorithm with spatiotemporal attentions, and we utilize the spatiotemporal correlations to connect the two together. The GAN method [31] is an inference-prediction framework based on the Generative Adversarial Network, but it does not have the connection between the inference and prediction and the spatiotemporal correlations. The DMF+NAR_SDAE method [32] is also an inference-prediction framework based on the neural networks without connection and it utilized the spatiotemporal information in the prediction. As shown in Fig .19, we evaluated the performance of the inference and prediction algorithm under different frameworks. The solid line in the figure indicates the inference error of each method at the 0.1-0.5 sense ratio, and the dashed line indicates the prediction error of each method at the 1-5 prediction cycles. In the inference experiment, our method achieves the best results under each metric, while GAN and DMF have worse experimental results than our method due to the lack of spatiotemporal constraints. In the prediction experiment, our method achieves the lowest error under each metric as well. The GAN-based method has difficulty in extracting sufficient spatiotemporal correlations compared with the graph neural network, so its results are worse than ours. Although the NAR+SDAE method can utilize spatiotemporal information, it cannot capture spatiotemporal correlations dynamically without an attention mechanism. Therefore, its prediction results are not as good as ours and it even performs poorly on some datasets with large spatiotemporal fluctuations. Another point worth noting is that the added iterative update mechanism also contributes to the improvement of our inference and prediction effectiveness. It is able to obtain more accurate spatiotemporal information to infer and predict better, which cannot be done by the other two methods.

TABLE 2: Running time for main methods

	PM2.5	Hum.	TraSpeed.	TraFlow.
BGMC-st	0.17s	0.21s	0.36s	0.22s
AGCN-st	0.21s	0.30s	0.54s	0.24s

7.7 Running time

Finally, we list the running times of the main methods in the table 2. Our experiment platform is equipped with Inter(R) Core(TM) CPU i5-10400F @ 2.90GHz and 16.00 GB RAM, and we use the Mxnet to do the urban prediction experiments. The table shows the time of inferring a complete matrix for each sensing cycle and training once in all data sets. The BGMC-st costs 0.17-0.36s to infer data and the AGCN-st costs 0.21-0.54s to predict data. The running time is totally acceptable in practical applications.

8 Conclusion and Discussion

8.1 Conclusion

In this paper, we turn attention from inferring the current unsensed data to predicting the future full map from the sparse sensed data. We propose an urban inference and prediction framework in Sparse MCS, which consists of three parts: data inference, data prediction, and iterative update. Firstly, we propose a Bipartite-Graph-based Matrix Completion algorithm with spatiotemporal constraints to recover the full sensing map from the historical data. Then we present a GCN model with spatiotemporal attentions to predict the near future data. Finally, we use the spatiotemporal correlations matrix in both to update iteratively, in order to enhance the correlations to improve the performances of data inference and prediction at the same time. Extensive evaluations have been conducted on two types of typical urban sensing tasks with four real-world data sets, including the monitoring of urban environment (PM2.5, Humidity) and urban traffic (Traffic speed, Traffic flow). The results show that our proposed algorithms can achieve a high inference and prediction accuracy with the sparse sensed data.

8.2 Discussion

However, as the data density increases, the advantages of our methods are no longer as obvious as when the data is sparse. The reason is that we use the data inference method first and then predict the future full map, which is designed for the sparse mobile crowdsensing and has one more step than the general prediction algorithms. In addition, our sparse performance is that there are missing values in each round of collection. In the face of large-area missing data caused by no data collected in the whole round, our method improvement is limited. And if the data set is greatly affected by factors other than spatiotemporal factors, our method improvement is not obvious enough in the future. We should focus on the inference of spatiotemporal fault data and the prediction under the influence of multiple factors, and further expand the prediction range in the future.

ACKNOWLEDGEMENTS

This work is supported in part by National Key R&D Program of China under Grant Nos. 2021ZD0112501 and 2021ZD0112502, and National Natural Science Foundation of China under Grant Nos. 62102161, 61772230 and 61972450, and CCF-Baidu Open Fund (No.2021PP15002000), and NSF grants CNS 1824440, CNS 1828363, CNS 1757533, CNS 1629746, CNS 1651947, and CNS 1564128.

REFERENCES

[1] W. Liu, Y. Yang, E. Wang, and J. Wu, "Fine-grained urban prediction via sparse mobile crowdsensing," in *2020 IEEE 17th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, 2020, pp. 265–273.

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

[3] Y. Liu, K. Liu, and M. Li, "Passive diagnosis for wireless sensor networks," *IEEE/ACM Transactions on Networking*, vol. 18, no. 4, pp. 1132–1144, 2010.

[4] D. Lin, Q. Wang, W. Min, J. Xu, and Z. Zhang, "A survey on energy-efficient strategies in static wireless sensor networks," *ACM Trans. Sen. Netw.*, vol. 17, no. 1, oct 2020. [Online]. Available: https://doi.org/10.1145/3414315

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

[6] H. Chen, B. Guo, Z. Yu, and Q. Han, "Crowdtracking: Real-time vehicle tracking through mobile crowdsensing," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 7570–7583, Oct 2019.

[7] S. B. Azmy, N. Zorba, and H. S. Hassanein, "Quality estimation for scarce scenarios within mobile crowdsensing systems," *IEEE Internet of Things Journal*, vol. 7, no. 11, pp. 10 955–10 968, 2020.

[8] W. Liu, Y. Yang, E. Wang, and J. Wu, "Dynamic user recruitment with truthful pricing for mobile crowdsensing," in *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, July 2020, pp. 1113–1122.

[9] L. Wang, D. Zhang, Y. Wang, C. Chen, X. Han, and A. M'hamed, "Sparse mobile crowdsensing: challenges and opportunities," *IEEE Communications Magazine*, vol. 54, no. 7, pp. 161–167, 2016.

[10] 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 Transactions on Intelligent Systems and Technology*, vol. 9, no. 2, pp. 1–28, 2017.

[11] W. Liu, L. Wang, E. Wang, Y. Yang, D. Zeghlache, and D. Zhang, "Reinforcement learning-based cell selection in sparse mobile crowdsensing," *Computer Networks*, vol. 161, pp. 102–114, 2019.

[12] K. Xie, X. Li, X. Wang, G. Xie, J. Wen, and D. Zhang, "Active sparse mobile crowd sensing based on matrix completion," in *Proceedings of the 2019 International Conference on Management of Data*, ser. SIGMOD '19. New York, NY, USA: ACM, 2019, pp. 195–210.

[13] W. Liu, Y. Yang, E. Wang, and J. Wu, "User recruitment for enhancing data inference accuracy in sparse mobile crowdsensing," *IEEE Internet of Things Journal*, pp. 1–1, 2019.

[14] S. Bose, C. Das, S. Dutta, and S. Chattopadhyay, "A novel interpolation based missing value estimation method to predict missing values in microarray gene expression data," in *2012 International Conference on Communications, Devices and Intelligent Systems (CODIS)*, 2012, pp. 318–321.

[15] T. Liu, Y. Zhu, Y. Yang, and F. Ye, "Alc2: When active learning meets compressive crowdsensing for urban air pollution monitoring," *IEEE Internet of Things Journal*, 2019.

[16] S. He and K. G. Shin, "Steering crowdsourced signal map construction via bayesian compressive sensing," in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, April 2018, pp. 1016–1024.

[17] W. Liu, Y. Yang, E. Wang, H. Wang, Z. Wang, and J. Wu, "Dynamic online user recruitment with (non-) submodular utility in mobile crowdsensing," *IEEE/ACM Transactions on Networking*, pp. 1–14, 2021.

[18] W. Liu, Y. Yang, E. Wang, and J. Wu, "User recruitment for enhancing data inference accuracy in sparse mobile crowdsensing," *IEEE Internet of Things Journal*, vol. 7, no. 3, pp. 1802–1814, 2020.

[19] Y. Yang, W. Liu, E. Wang, and J. Wu, "A prediction-based user selection framework for heterogeneous mobile crowdsensing," *IEEE Transactions on Mobile Computing*, vol. 18, no. 11, pp. 2460–2473, 2019.

[20] 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 *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, 2010, pp. 105–116.

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

[22] I. Lana, I. Olabarrieta, M. Velez, and J. D. Ser, "On the imputation of missing data for road traffic forecasting: New insights and novel techniques," *Transportation Research Part C Emerging Technologies*, vol. 90, no. MAY, pp. 18–33, 2018.

[23] A. Liu, C. Li, W. Yue, and X. Zhou, "Real-time traffic prediction: A novel imputation optimization algorithm with missing data," *IEEE*, 2019.

[24] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal arima process: Theoretical basis and empirical results," *Journal of Transportation Engineering*, vol. 129, no. 6, pp. 664–672, 2003.

[25] E. Zivot and J. Wang, *Vector Autoregressive Models for Multivariate Time Series*. Modeling Financial Time Series with S-Plus®, 2003.

[26] H. V. Lint and C. V. Hinsbergen, "Short-term traffic and travel time prediction models," *Transportation Research E-Circular*, 2012.

[27] Y.-S. Jeong, Y.-J. Byon, M. M. Castro-Neto, and S. M. Easa, "Supervised weighting-online learning algorithm for short-term traffic flow prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 4, pp. 1700–1707, 2013.

[28] J. Zhang, Z. Yu, D. Qi, R. Li, and T. Li, "Predicting citywide crowd flows using deep spatio-temporal residual networks," *Artificial Intelligence*, vol. 259, 2017.

[29] H. Yao, F. Wu, J. Ke, X. Tang, Y. Jia, S. Lu, P. Gong, J. Ye, and Z. Li, "Deep multi-view spatial-temporal network for taxi demand prediction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[30] R. Chan, M. Y. Lim, and R. Parthiban, "A neural network approach for traffic prediction and routing with missing data imputation for intelligent transportation system," *Expert Systems with Applications*, vol. 171, no. 2, p. 114573, 2021.

[31] M. Gupta and R. Beheshti, "Time-series imputation and prediction with bi-directional generative adversarial networks," 2020.

[32] 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 Transactions on Industrial Informatics*, vol. 17, no. 9, pp. 6170–6181, 2021.

[33] M. Roughan, Y. Zhang, W. Willinger, and L. Qiu, "Spatio-temporal compressive sensing and internet traffic matrices," *IEEE/ACM Transactions on Networking*, vol. 20, no. 3, pp. 662–676, 2012.
[34] P. Lancaster and M. Tismenetsky, "The theory of matrices: With applications," 1985.
[35] Haupt, Jarvis, Bajwa, Waheed, U., Raz, Gil, Nowak, and Robert, "Toeplitz compressed sensing matrices with applications to sparse channel estimation." *IEEE Transactions on Information Theory*, vol. 56, no. 11, pp. 5862–5875, 2010.
[36] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazirbas, V. Golkov, P. v. d. Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 2758–2766.
[37] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *International Conference on Learning Representations*, 2018.
[38] A. E. Brouwer and A. E. Brouwer, *Spectra of Graphs*. Springer New York, 2012.
[39] L. M. Surhone, M. T. Tennoe, and S. F. Henssonow, "Spectral graph theory," *Published for the Conference Board of the mathematical sciences by the American Mathematical Society*, 2010.
[40] R. Bracewell, "The fourier transform and its applications," *American Journal of Physics*, vol. 34, 2002.
[41] M. Simonovsky and N. Komodakis, "Dynamic edge-conditioned filters in convolutional neural networks on graphs," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
[42] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 922–929, 2019.
[43] X. Xiang, M. Zhai, R. Zhang, N. Lv, and A. E. Saddik, "Optical flow estimation using spatial-channel combinational attention-based pyramid networks," in *2019 IEEE International Conference on Image Processing (ICIP)*, 2019, pp. 1272–1276.
[44] Y. Li, C. Wu, L. Li, Y. Liu, and J. Zhu, "Caption generation from road images for traffic scene modeling," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–12, 2021.

En Wang received his B.E. degree in software engineering from Jilin University, Changchun in 2011, and his M.E. degree and Ph.D. in computer science and technology from Jilin University, Changchun in 2013 and 2016. He is currently a professor in the Department of Computer Science and Technology at Jilin University, Changchun. His current research focuses on the efficient utilization of network resources, scheduling and drop strategy in terms of buffer-management, energy-efficient communication between human-carried devices, and mobile crowdsensing.

Weiting Liu received his bachelor's degree in software engineering from Jilin University, Changchun, China, in 2020. Currently, he is studying for the master's degree in computer science and technology from Jilin University, Changchun, China. His current research focuses on mobile crowdsensing, spatiotemporal data processing.

Wenbin Liu received the B.S. degree in physics and the Ph.D. degree in computer science and technology from Jilin University, China, in 2012 and 2020, where he is currently a Postdoctoral Researcher in Dingxin Scholar Program with the College of Computer Science and Technology. He was also a visiting Ph.D. student in the Wireless Networks and Multimedia Services Department, Telecom SudParis/Institut Mines-Telecom, France. His research interests include Mobile CrowdSensing, Mobile Computing, and Ubiquitous Computing.

Yongjian Yang received his B.E. degree in automatization from Jilin University of Technology, Changchun, Jilin, China in 1983; his M.E. degree in computer communication from Beijing University of Post and Telecommunications, Beijing, China in 1991; and his Ph.D. in software and theory of computer from Jilin University, Changchun, Jilin, China in 2005. He is currently a professor and a PhD supervisor at Jilin University, Director of Key lab under the Ministry of Information Industry, and Standing Director of the Communication Academy. His research interests include: network intelligence management, wireless mobile communication and services, and wireless mobile communication.

Bo Yang is currently a professor in the College of Computer Science and Technology, Jilin University. He is also the director of the Key Laboratory of Symbolic Computation and Knowledge Engineering, Ministry of Education, China. His current research interests are in the areas of data mining, complex network analysis, self-organized and self-adaptive multi-agent systems, with applications to knowledge engineering and intelligent health informatics.

Jie Wu is the Director of the Center for Networked Computing and Laura H.Carnell professor at Temple University. He also serves as the Director of International Affairs at College of Science and Technology. He served as Chair of Department of Computer and Information Sciences from the summer of 2009 to the summer of 2016 and Associate Vice Provost for International Affairs from the fall of 2015 to the summer of 2017. Prior to joining Temple University, he was a program director at the National Science Foundation and was a distinguished professor at Florida Atlantic University. His current research interests include mobile computing and wireless networks, routing protocols, cloud and green computing, network trust and security, and social network applications. Dr. Wu regularly publishes in scholarly journals, conference proceedings, and books. He serves on several editorial boards, including IEEE Transactions on Mobile Computing, IEEE Transactions on Service Computing, Journal of Parallel and Distributed Computing, and Journal of Computer Science and Technology. Dr. Wu was general co-chair for IEEE MASS 2006, IEEE IPDPS 2008, IEEE ICDCS 2013, ACM MobiHoc 2014, ICPP 2016, and IEEE CNS 2016, as well as program co-chair for IEEE INFOCOM 2011 and CCF CNCC 2013. He was an IEEE Computer Society Distinguished Visitor, ACM Distinguished Speaker, and chair for the IEEE Technical Committee on Distributed Processing (TCDP). Dr. Wu is a CCF Distinguished Speaker and a Fellow of the IEEE. He is the recipient of the 2011 China Computer Federation (CCF) Overseas Outstanding Achievement Award.