Federated Learning-Based Distributed Data Completion in Sparse Mobile CrowdSensing

En Wang, *Member, IEEE*, Baoju Li, *Graduate Student Member, IEEE*, Wenbin Liu, *Member, IEEE*, Zengyi Han, *Member, IEEE*, Cong Wang, Ximing Li, and Jie Wu, *Fellow, IEEE*

***Abstract*—Sparse Mobile CrowdSensing (SMCS) is an emerging distributed data collection framework. As one of the core methods,data completion uses the collected data to fill in the missing data. However, this approach inevitably poses significant privacy risks, since the traditional data completion methods require users' time and location information. In this paper, we propose a federated learning-based distributed data completion framework, which employs matrix factorization (MF) for local completion model training and federated learning to aggregate parameters of the MF model. This enables the construction of a global completion model without requiring private data, thereby mitigating privacy concerns. To address the challenges posed by the sparsity and asynchrony of distributed data, we incorporate time-aware deep neural network architectures with an asynchronous mechanism to enable federated training under irregularly gathered local data. Experimental results demonstrate that the proposed model achieves high data completion accuracy while ensuring robust privacy protection.**

***Index Terms*—Privacy protection, distributed data, federated learning, sparse crowdsensing.**

I. Introduction

SPARSE Mobile CrowdSensing (SMCS) [1] has emerged as a powerful paradigm for urban data acquisition, addressing challenges of data sparsity in dynamic urban environments. By leveraging data contributed by distributed individuals or opportunistic sensing devices, this approach minimizes the need for costly, large-scale user recruitment and sensor deployments. SMCS has fueled the development of many diverse smart-city applications such as temperature monitoring [2] in the environmental monitoring system and the traffic flow sensing [3] in the intelligent transportation systems.

Received 12 May 2025; revised 19 November 2025; accepted 1 December 2025. Date of publication 4 December 2025; date of current version 6 April 2026. This work was supported in part by the National Key R&D Program of China under Grant 2022YFB3103700 and Grant 2022YFB3103702, in part by the National Natural Science Foundation of China under Grant 62472194 and Grant 62272193, and in part by Jilin Science and Technology Research Project under Grant 20250102220JC. Recommended for acceptance by F. Liu. *(Corresponding author: Wenbin Liu.)*

En Wang, Baoju Li, Wenbin Liu, Zengyi Han, Cong Wang, and Ximing Li are with the College of Computer Science and Technology, Jilin University, Changchun 130012, China, and also with the Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, Changchun 130012, China (e-mail: wangen@jlu.edu.cn; libj21@mails.jlu.edu.cn; liuwenbin@jlu.edu.cn; hzy@ieee.org; wcong22@mails.jlu.edu.cn; liximing86@gmail.com).

Jie Wu is with the China Telecom Cloud Computing Research Institute, Beijing 100088, China, and also with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122 USA (e-mail: jiewu@temple.edu).

Digital Object Identifier 10.1109/TMC.2025.3640170

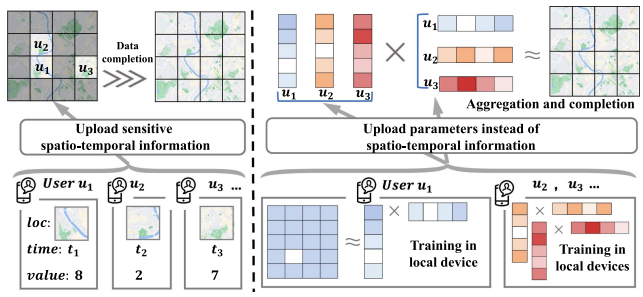

Fig. 1. A privacy-preserving framework for distributed data completion, where users should upload only model parameters instead of private data.

As one of the critical challenges in SMCS, data completion methods have seen rapid advancements in recent years. However, these methods often rely on direct access to users' private data, raising significant privacy concerns. Traditional data completion approaches require private data as an essential input. For example, interpolation-based methods [4] necessitate private data from each user, while Deep Neural Network (DNN)-based methods [5] rely on ground-truth data inputs for processing. The submission of private data makes it vulnerable to potential interception by malicious third parties, leading to privacy information leakages, particularly concerning when such data includes spatio-temporal information that can reveal individuals' routine behaviors. Consequently, it is an urgent need for data completion approaches that balance privacy preservation and completion accuracy.

An intuitive approach to balancing privacy protection and data completion is to integrate Federated Learning (FL) into the data completion process. Nevertheless, this seemingly straightforward solution presents several technical challenges.

1) How to resolve the inner contradiction between FL and data completion? As shown in Fig. 1, effective data completion requires access to private data to capture the inherent spatio-temporal correlations within the data. This necessity conflicts with the principles of FL, where local models are trained independently on user devices and then aggregated on a central server. Additionally, FL framework requires certain portions of the real data as input in the global model. Many data completion methods still raise a significant privacy concern when integrated with FL, as the global model would need to access the sensitive data to ensure that it can effectively complete the entire dataset.

2) How can we mitigate the loss of spatio-temporal correlations? Urban data always contains a large amount of

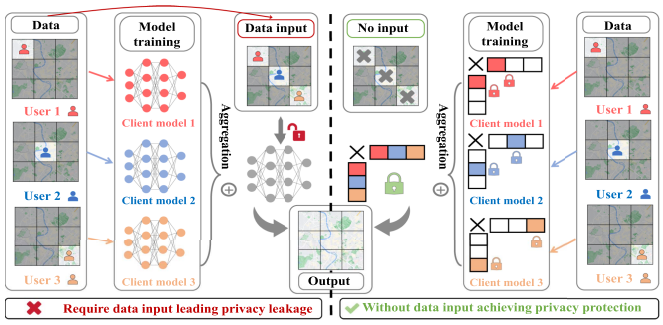

Fig. 2. Using MF for distributed data completion without requiring private data input.

spatio-temporal correlations. Applying FL to the data completion method might result in the loss of these critical correlations, since each user collects only a small amount of data covering specific points of interest. Given that urban data often involves numerous nonlinear spatio-temporal correlations, the limited localized and fragmented datasets further complicate the learning and completion processes, and make the framework fail to capture interconnected spatio-temporal correlations.

3) How to provide real-time completion results despite time schedule variations among users? In SMCS, data is often collected in real-time, forming time series with irregular intervals [6]. Existing FL-based models [7], [8], [9], which rely on fixed time intervals and synchronous processing—training local models and performing global aggregation after shared time interval. These kinds of synchronous training styles fail to capture temporal heterogeneity across users and are unsuitable for real-time, distributed settings.

In this paper, we propose an FL-based Asynchronous Matrix Factorization system (FLAMF) to address these challenges while preserving privacy and maintaining high inference accuracy. **1)** To resolve the inherent contradiction between FL and data completion, we use Matrix Factorization (MF) methods in the local model. Each user can perform local MF using its private data, where it learns a low-rank approximation of the data. This step allows each client to capture local patterns and correlations without sharing private data. **2)** To mitigate the loss of spatio-temporal correlations due to the limited data collected by individual users, we leverage the idea of recurrent neural networks to propose Time Continuous Deep MF (TIME-DMF) networks that explore the nonlinear relationships within sparse data and retain hidden correlations in the complete data map. **3)** To provide real-time completion results despite time schedule variations in data collection clients, we extend our design to a layer-wise asynchronous training process that can handle variable-length time intervals, enabling real-time results regardless of variations in input time.

Note that we use MF models because they are well-suited for data completion and their integration with FL. Unlike traditional methods that require partial real data as input for global model, MF relies solely on mathematical matrix computations to achieve data completion. As shown in Fig. 2, each client in FLAMF factorizes its local data into smaller matrices. These factorized matrices are then uploaded to the central server. The server aggregates these matrices to reconstruct the global data matrix without needing direct access to the private data. By leveraging the decentralized nature of FL and the mathematical principles of MF, FLAMF provides an elegant solution for both privacy-preserving and accurate data completion. The contributions of this paper are summarized as follows:

- We propose the problem of privacy preservation in integrating federated learning with distributed data completion and we formulate this problem into the time continuous MF in order to address it efficiently.
- We propose a distributed data completion model with robust private data protection, leveraging an FL framework specifically designed for distributed and sparse data. Furthermore, we integrate a Local Differential Privacy (LDP) mechanism during the user-side submission process, ensuring that private local data remains confidential throughout the communication.
- We enhance the local model by proposing TIME-DMF networks that can effectively preserve hidden correlations within sparse data. Additionally, we propose an asynchronous training process that can timely output a complete data map based on each independent local training and does not rely on additional private data.
- Extensive experiments and theoretical analyses were conducted to validate the effectiveness of our approach. We evaluated six distinct models across three datasets, demonstrating that the proposed algorithm achieves notable accuracy and convergence efficiency in data completion while ensuring robust data privacy and security within the FL paradigm.

The remainder of the article is organized as follows. We introduce some related works of data completion and privacy protection methods in Section II. Then the system model and problem definition are presented in Section III. We introduce our main method of FL-based distributed data completion in Section IV and present the corresponding theoretical analysis in Section V. The performance evaluations are presented in Section VI. At last, we conclude our work in Section VII.

II. Related Work

A. Data Completion

Data completion, also known as data inference [10], refers to the process of filling in missing values within a dataset. Early methods of data completion use regression models [11] where missing values are predicted based on other available variables. These performances of early data completion methods highly depend on the private data quality. Driven by the increasing complexity and volume of data in various fields, machine learning-based methods outperform traditional statistical techniques by capturing complex relationships within the data, especially the deep learning-based methods. Khan et al. [12] propose a data completion method that involves a Convolutional Neural Network to infer the missing values. They are imputed using a trained kernel that performs a weighted sum of neighboring elements. Other methods such as generative [13] and

predictive [14] approaches are also used in data completion, but all these ML-based models require input data to function effectively, which raises privacy concerns.

Distributed machine learning has been under the spotlight for decades. However, it remains a formidable challenge that the performance deteriorates significantly when data are heterogeneously distributed over different nodes in the network. Chen et al. [15] propose a distributed personalized imputation algorithm based on a Gaussian mixture model (GMM) to handle missing data in distributed scenarios. It constructs a general GMM with public and personalized components and develops the KT-dpEM algorithm for parameter estimation and imputation. However, it does not consider asynchronous aggregation and variable-length temporal inputs.

MF [16] is a fundamental technique for handling sparse data, and its theoretical foundation lies in the concept of low-rank approximation. Based on this principle, a variety of advanced MF algorithms have been developed. Classical approaches such as SVD [17], Probabilistic MF [18], and Non-negative MF [19] improve efficiency, robustness, and interpretability, and have been widely adopted in recommendation systems to exploit low-rank structures for data completion. More recently, Deep MF [20] extends MF to nonlinear settings by combining its interpretability with the expressive power of deep neural networks, making it particularly suitable for modeling complex spatio-temporal correlations.

Motivated by these developments, our work builds upon the MF foundation and integrates it into a federated learning framework. Specifically, we employ Time Continuous Deep MF to capture spatio-temporal correlations for data completion, while using federated aggregation to preserve privacy by avoiding direct sharing of private data.

B. Data Privacy Protection in Data Completion

Data privacy protection is a crucial concern in data completion because these methods often require access to sensitive and potentially identifiable information. The key methods in protecting data privacy during data completion [21] include differential privacy [22], federated learning [23], homomorphic encryption [24], and secure multi-party computation [25].

FL has become a key paradigm for privacy-preserving distributed learning. Recent surveys [26], [27] provide systematic reviews of FL from perspectives such as data partitioning, model architectures, privacy mechanisms, and communication strategies, while also highlighting open challenges including statistical heterogeneity and efficiency. In SMCS, FL has been adopted to enhance privacy and scalability. For example, CrowdFL [28] integrates FL with secure aggregation and incentive mechanisms for privacy-preserving sensing, while FedLens [29] explores federated crowdsensing in virtual tourism scenarios. These works demonstrate the feasibility of applying FL in collaborative sensing applications.

Another line of work investigates federated MF for recommendation tasks. Recent studies have applied FL to collaborative filtering [30] or proposed verifiable and privacy-preserving protocols such as VPFedMF [31]. These methods mainly embed user–item interactions into latent vectors and apply MF to predict preferences. In contrast, our work targets spatio-temporal data completion, leveraging the low-rank property of SMCS matrices and exploiting non-linear spatio-temporal relationships via deep networks, rather than modeling preference embeddings. Li et al. [32] propose a federated MF framework that employs federated learning to ensure that the model can be trained without compromising user data privacy and involves adding noise to the gradients before they are shared with the central server. Chai et al. [33] present FedMF, a secure federated MF framework by combining federated learning with homomorphic encryption [34], allowing users to upload encrypted gradient information rather than private data. But these studies do not explore asynchronous FL training modes, focusing on traditional MF.

III. Problem Definition

A. Problem Description

SMCS has emerged as a vital framework for urban data collection, empowering a range of smart-city applications including environmental monitoring, intelligent transportation, and public health surveillance. In SMCS, data sparsity is a prominent challenge due to budget constraints and inconsistent user participation, which leads to a significant proportion of missing data and hinders accurate analysis and real-time decision-making.

Traditional data completion methods, such as interpolation-based and centralized deep learning approaches, require aggregating raw sensor readings from multiple users to reconstruct the missing values. Although effective in some specific settings, these approaches raise substantial privacy concerns. In privacy-sensitive applications, directly collecting and sharing spatio-temporal data can expose users' mobility patterns and personal routines, thereby increasing the risk of data breaches and unauthorized data interception.

To mitigate these privacy issues, FL has been proposed as a promising solution that enables privacy-preserving collaborative learning. By performing model training locally on users' devices and sharing only model parameters with a central server, FL ensures that raw data remains private. However, while FL addresses privacy concerns, it also imposes challenges on data completion due to the limited local data available for training.

For local data completion, MF models are well-suited due to their strong mathematical foundation and ability to leverage the low-rank structure commonly observed in real-world datasets. MF decomposes the observed local data matrix into lower-dimensional latent factors, which facilitates the reconstruction of missing entries efficiently. In the context of FL, this means that each local device can independently build an MF-based model without exposing raw data, while contributing to a global model through parameter sharing. To make it clear, in the MF-based model formulation, each client only uploads the decomposed latent factors—specifically, the learned low-rank matrices representing the model's input and parameters—rather than the original spatio-temporal data. This design ensures that even if a third party intercepts the transmitted data, they can at most reconstruct an approximate global data representation,

but cannot directly infer the private location or mobility patterns of any individual client. By preventing direct exposure of raw spatio-temporal observations, this kind of problem formulation significantly enhances privacy protection while maintaining effective data completion capabilities.

Moreover, the local data in SMCS is inherently spatio-temporal, possessing both temporal and spatial correlations. In real-world scenarios, observations arrive in a strict time-ordered fashion—forming a data stream where each X_t corresponds to sensor readings across multiple spatial locations at time t. This sequential nature of the data requires a model that can continuously learn from new observations while effectively capturing the inherent temporal dependencies and spatial relationships.

B. Problem Formulation

Although MF requires access to the private dataset for training in the FL framework, it only uploads parameters, ensuring that sensitive spatio-temporal information cannot be intercepted. It is still an effective method for solving distributed data completion due to its ability to handle missing data efficiently in distributed settings and its strong mathematical foundation [35]. Inspired by the gating mechanism in LSTM, we use TIME-DMF module to formulate the proposed problem because it builds upon MF's strengths by incorporating the assumption of low-rank structure, which is prevalent in real-world datasets. By leveraging TIME-DMF, we can exploit the hidden spatio-temperal correlations across distributed data sources to improve inference accuracy while maintaining computational efficiency.

Due to mobility constraints and limited budgets, urban sensing data in SMCS tends to cover specific subareas. As a result, the entire urban area is divided into N subareas. Each subarea is identified by its index n, helping in organizing the data collection process. The data providers actively gather data within their designated subareas using their mobile devices and record the sensory data at each time step t on the local devices. At each local client, the collected data forms a $T \times N$ sparse matrix of the entire urban map, where the unsensed data is recorded as 0 or other abnormal value if 0 is an unreasonable value. We aim to recover the unsensed data of each time step in the whole map through the sparse sensed data.

First, the recorded data in n locations at time step t is denoted as the vector $y_{1\times n}^{(t)'}$, comprising the local sparse matrix $Y'_{T\times N} = [y_{1\times N}^{(1)'\mathbf{T}}, y_{1\times N}^{(2)'\mathbf{T}}, \ldots, y_{1\times N}^{(T)'\mathbf{T}}]^{\mathbf{T}}$, and the corresponding ground truth matrix is denoted as $Y_{T\times N} = [y_{1\times N}^{(1)\mathbf{T}}, y_{1\times N}^{(2)\mathbf{T}}, \ldots, y_{1\times N}^{(T)\mathbf{T}}]^{\mathbf{T}}$. Each column in the sparse matrix Y'/Y is a temporal data record in the nth location. Next, we set a logical matrix $L_{T\times N}$, indicating whether the location n has been covered in the current time step t. If the location n has been sensed in time step t, $L[t][n] = 1$, and $L[t][n] = 0$ otherwise. Then we have:$Y' = Y \odot L$, where $\odot$ represents the Hadamard product. For each location j at time step i:

$$Y'[i][j] = Y[i][j]L[i][j]. \tag{1}$$

Then, based on the sensed data matrix Y', we need to find a data inference function like $f(\cdot)$ to infer the unsensed data matrix $\hat{Y}$, which is represented as follows:

$$f(Y' = \hat{Y}) \approx Y, \tag{2}$$

where $\hat{Y}$ is defined as the inference matrix. Correspondingly, we use $\varepsilon(\hat{Y}, Y) = \|\hat{Y} - Y\|_F$ to calculate the inference error, where $\|\cdot\|_F$ denotes the Frobenius norm of a matrix. By iteratively calculating the gradient descent updates to minimize the inference error, the inferred data matrix will converge.

Problem [nonlinear MF based data completion]: Given a set of sparse data collected from an urban area during a certain time interval. The dataset can be divided into m time steps and n subareas, forming the corresponding $N \times M$ sparse matrix. The elements y' in the sparse matrix represent the sensed data value, which are then used to infer the unsensed data value $\hat{y}$. The objective is to facilitate collaborative data completion functions $f(\cdot)$ utilizing distributed datasets from multiple participants, with the inference error minimized:

$$\min \sum_{i=1}^{n} \varepsilon(\hat{y}_i, y_i),$$
$$\text{s.t.} \quad f(y') = \hat{y}. \tag{3}$$

TABLE I
MAIN NOTIONS

Symbol	Meaning
T, t	Total time steps number, each time step index
N, n	Total subareas number, each subarea index
T	Matrix transposition operator
M	Clients number
$y_{1\times N}^{(n)}$	Sensed sparse data vector in time step n sparse matrix of sensed data covering
$Y'_{T\times N}$	All T time steps and N subareas
$L[t][n]$	Logical matrix reflecting whether subarea n is sensed in time step t
$\hat{Y}$, $\hat{y}$	Inference data matrix, inference data vector
Y, y	Ground truth data matrix, ground truth data vector
$F_{T\times r}$	Parameters matrix of inference algorithm
$X_{r\times N}$	Factorized matrix
$g(\cdot), W(\cdot), b(\cdot)$	Activation function, weight matrix and bias vector of the i-th layer

IV. METHODOLOGY

This section presents the components of our proposed method for data completion using the FL framework. We start with MF for nonlinear data completion, presenting the model setting to address nonlinear data relationships. Then we integrate the model into the FL framework and provide a detailed description of the local MF model and the asynchronous training process. Finally, we explore local privacy enhancement to further protect the privacy in data communication.

A. MF for Nonlinear Data Completion

MF is a fundamental technique for handling sparse data, grounded in the low-rank approximation assumption. The key idea is that a high-dimensional and sparse matrix can be approximated by the product of two much smaller matrices, i.e., $Y^{m\times n} \approx UV^T$, which maps explicit observations into a low-dimensional and dense "latent semantic space". The inner

product between two vectors geometrically reflects their directional similarity, making it a natural similarity measure. The effectiveness of MF relies on the assumption that real-world spatiotemporal data matrices are approximately low-rank: although they may appear complex and contain many missing entries, their variations are often driven by a few shared and fundamental correlations. Formally, a matrix is said to be low-rank if most of its information can be represented in a much lower-dimensional subspace, i.e., $\text{rank}(Y') = r \ll \min(m, n)$. In such cases, MF decomposes Y' into two factor matrices $U \in \mathbb{R}^{m\times r}$ and $V \in \mathbb{R}^{n\times r}$, such that $Y \approx Y' = UV^T$. The columns of U and V can be interpreted as latent features of the rows and columns of the original matrix. This low-rank property reflects the existence of hidden structures in the data: for example, in spatio-temporal sensing data, it reveals correlated patterns across space and time.

This kind of MF is confined to the assumption that Y' and U, V are linearly dependent. However, in most real-world situations, the sensory data exhibits complex spatio-temporal correlations with nonlinear structures. Therefore, we use a nonlinear function $f(\cdot)$ to represent the spatio-temporal correlations:

$$\hat{Y} = f(X), \tag{4}$$

and a deep learning model is used to represent this nonlinear function. By using deep neural networks, we aim to optimize the input X and $f(\cdot)$ simultaneously until we achieve an acceptable loss value. This approach, which treats both X and $f(\cdot)$ as parameters, enables us to directly output the inferred data $\hat{Y}$ for the entire map.

Specifically, we assume the deep learning model consists of K hidden layers. The transformation from the input layer to the output layer can be expressed using a combination of functions. Essentially, each layer performs a linear relationship on the output of the previous layer followed by the application of a nonlinear activation function. The i-th layer can be formalized using the following formula:

$$h^{(i)} = g^{(i)}(W^{(i)}h^{(i-1)} + b^{(i)}), \tag{5}$$

where $i = 1, 2, \ldots, K$ is the layer index, $g^{(i)}$ is the activation function, and $W^{(i)}$ and $b^{(i)}$ are the weight matrix and bias vector of the i-th layer, respectively.

The corresponding activation function set is:

$$g = \{\sigma_{(1)}(\cdot), \sigma_{(2)}(\cdot), \cdots, \sigma_{(K)}(\cdot)\}. \tag{6}$$

Given these settings, the nonlinear function $f(\cdot)$ can be expressed as:

$$\begin{aligned} f(X) = \sigma_K(W^{(K)}(\sigma_{K-1}(W^{(K-1)}, \cdots, \\ \sigma_1(W^{(1)}x + b^{(1)}) \cdots) + b^{(K-1)}) + b^{(K)}). \end{aligned} \tag{7}$$

And the loss function is set as follows:

$$\min \frac{1}{2n}\|(Y - f(X)) \odot L\|_F^2. \tag{8}$$

Since L is the logical matrix that indicates whether the spatio-temporal data is observed, by iteratively minimizing the loss value between the observed data and inferred data, the inferred data from the sparse matrix is acquired by (4).

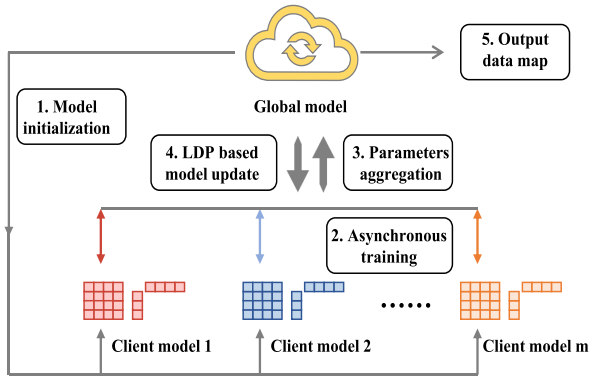

Fig. 3. FL-Based distributed data completion framework.

B. Federated Learning Based Data Completion

Using MF to infer unsensed data requires extensive spatio-temporal data, which cannot be collected by a single individual. Consequently, the prevalent method involves gathering sparse sensory data from multiple participants. Unfortunately, this approach raises privacy concerns. The data submission process may be vulnerable to interception by malicious third parties, potentially leading to privacy breaches since the sparse data includes spatio-temporal information that reflects an individual's routine behaviors. Some researchers address this by introducing random noise directly into the data submitted, which significantly compromises the accuracy of the final completion accuracy.

Considering the distributed sparse characteristic and local privacy protection, differential privacy federated learning emerges to be the optimal solution for collaborative data completion from distributed sparse sensory data without actual data communication. First, the training data in federated learning is distributed over a large number of local devices. Second, unlike traditional distributed learning systems, the central server in an FL system does not have direct access to private data, which helps protect local data privacy from malicious third parties. Third, in traditional FL, while the training data remains on the client's device, privacy concerns arise because sensitive information can potentially be inferred from the submitted model parameters. Therefore, by adding noise to the communications between clients and central server, FLAMF achieves a higher level of privacy protection.

1) System Overview: Fig. 3 presents the work flow of FLAMF, which includes a set $M = \{1, \ldots, m\}$ of clients. Each client $m \in M$ maintains a local dataset matrix D_m, which is only utilized to train a local MF model. Initially, the central server distributes a global model with randomized weights to all active clients. Each client then trains its nonlinear MF model in an asynchronous pattern. Subsequently, the clients apply differential privacy mechanisms, and then upload these perturbed parameters to the central server. In this work, we refer to this communication stage where local model parameters are transmitted to the server instead of private data as the up-link submission process. The central server aggregates these received parameters and directly outputs the entire data map.

Additionally, in practical data completion applications, sensory data is continuously collected due to the varying daily mobility routines of the users. As new sensory data is gathered, it is necessary to immediately update the local MF models and send them to the central server. This setup may lead to an asynchrony problem among the clients; that is, instead of synchronizing their training processes within predefined time intervals, each client's training process may occur independently at any time and does not require strict coordination across all participating clients. While this asynchrony problem allows for flexible and potentially more efficient data processing, it brings significant challenge to traditional FL-based data completion system.

2) Local MF Model: In this subsection, we aim to simultaneously optimize X and $f(\cdot)$ until we achieve an acceptable loss value. This approach, which treats both X and $f(\cdot)$ as parameters, enables us to directly output the inferred data for the entire map by (4).

The primary challenge in integrating the MF model into an FL system is the asynchrony among clients, since we are building an FL-based data completion system combining both distributed sensing and data inference stages within the client devices. The data-sensing process of each client cannot be simultaneous at a specific time step, resulting in an asynchronous training process that occurs in an online pattern. In other words, the local MF model begins training independently whenever new sparse data is collected, which makes it essential to maintain the temporal correlations among the sensory data. Therefore, the traditional DNNs are not capable of acting as a data completion method in the clients of an FL system, since they fail to preserve the temporal correlations among the data and aggregate in an asynchronous way. To overcome this challenge, we propose the time-continuous MF model to achieve time-continuous but asynchronous completion of sparse datasets. The TIME-DMF module draws inspiration from recurrent neural networks, specifically addressing the limitations of traditional LSTM networks in learning variable-length temporal dependencies, which is crucial for handling time-continuous but asynchronous distributed data completion.

FLAMF incorporates complex memory cells and time gates to effectively preserve temporal correlations and manage outputs related to time-dependent patterns. These mechanisms can simulate the effects of varying time intervals on the relationships between different time steps. The core of FLAMF is the memory cell, which plays an important role in retaining information over long input sequences. This ability to store and recall information is essential for modeling asynchronous training processes, where inputs from clients may arrive at irregular intervals. Thus, FLAMF ensures that the influence of earlier time steps is neither prematurely discarded nor overly dominant, allowing the model to adapt dynamically to new information while preserving temporal correlations.

Specifically, we present the collected sparse data in location n as a vector that contains spatio-temporal correlations: $y'_n = \{y'_{[1][n]}, y'_{[2][n]}, \ldots, y'_{[T][n]}\}^T$, and T time slots in N location sensory data forms a sparse matrix: $Y'_{T\times N} = \{y'_1, y'_2, \ldots, y'_N\}$. FLAMF adopts a multi-layer LSTM structure, where each time slot input is sequentially processed. At each time step, an LSTM unit maintains a cell state C_t, which acts as a memory mechanism to store long-term dependencies, and a hidden state h_t, which carries short-term dependencies and influences the final output. At each layer, these hidden state vectors influence the computation of the layer's output, which is then passed as input to the next layer. This hierarchical structure allows the model to progressively capture complex spatial and temporal dependencies. Each layer t contains three types of gates:

- *Forget Gate f_t:* Determines how much of the past cell state C_{t-1} should be retained or discarded. It ensures that irrelevant or outdated information is removed, preventing unnecessary accumulation of long-term dependencies.
- *Input Gate i_t:* Controls how much new information from the current input $y'_{[t][n]}$ should be added to the cell state. The candidate cell state $\tilde{C}_t$ is computed and combined with i_t to update the memory.
- *Output Gate o_t:* Decides how much information from the current cell state should be exposed as the hidden state h_t, which will be passed to the next time step and also influence the model's output.

To preserve temporal correlations, at each time step t, the temporal sequential data vector $y'_{[t][n]}$ is concatenated with the hidden state of the previous time step h_{t-1}. This combined vector $[h_{t-1}, y'_{[t][n]}]$ serves as the input for the current layer's LSTM unit. The above three gates are calculated as follows:

$$i_t = \sigma(W_t^i[h_{t-1}, y'_{[t][n]}] + b_t^i), \tag{9}$$

$$f_t = \sigma(W_t^f[h_{t-1}, y'_{[t][n]}] + b_t^f), \tag{10}$$

$$o_t = \sigma(W_t^o[h_{t-1}, y'_{[t][n]}] + b_t^o). \tag{11}$$

The candidate cell state $\tilde{C}_t$ and cell state update functions are denoted as follows:

$$\tilde{C}_t = \tanh(W_t^C[h_{t-1}, y'_{[t][n]}] + b_t^C), \tag{12}$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t. \tag{13}$$

Through LSTM's gating mechanism, particularly the output gate o_t, the model selectively filters and transforms this information into a latent representation h_t.

$$h_t = o_t \odot \tanh(C_t). \tag{14}$$

The output gate controls how much of the cell state's information should be exposed to the next layer or used as the final representation, ensuring that only the most relevant temporal features are retained for downstream processing.

C. Variable-Length Time Series Input

Traditional LSTM-based models assume that time-series data arrive at fixed, evenly spaced intervals, allowing the network to process updates in a predefined structured pattern. Under these conditions, information from past states propagates smoothly over time, making it easier to capture temporal dependencies. However, in real-world applications, time-series data is often collected at irregular intervals, meaning that data points do not

arrive at fixed time steps. Instead of evenly spaced timestamps like: $t = [0, 1, 2, 3, 4, \ldots]$, real-world data often follows an unpredictable pattern, such as: $t = [0, 0.5, 0.7, 1.4, 7.1, 7.3, \ldots]$. This irregularity introduces several fundamental challenges in modeling temporal dependencies, as traditional LSTM networks are not designed to handle varying time intervals between observations effectively.

One major challenge in SMCS is capturing temporal dependencies under irregular time intervals. While LSTM models perform well on evenly spaced data, they struggle when observations are uneven. In dense regions (e.g., urban hotspots), frequent updates may lead to redundant computation and overfitting to short-term patterns. In sparse regions, fixed-rate updates cause rapid information decay, weakening long-term memory. Traditional LSTM architectures update their cell state using gating mechanisms under the assumption that observations occur at regular intervals. These gates regulate how much past information is retained and how new observations are integrated, but they do not inherently account for varying time gaps between observations. As a result, the standard gating mechanism fails to properly weigh the significance of past observations, limiting the model's ability to effectively learn from irregularly sampled spatio-temporal data in SMCS environments.

To address the challenge posed by irregular time intervals in spatio-temporal data, we introduce TIME-DMF, an enhanced LSTM model that incorporates time-aware gating mechanisms. Unlike conventional LSTMs, which assume fixed time intervals between observations, TIME-DMF introduces two learnable time gates to dynamically adjust memory updates based on the actual time gaps between consecutive data points. The $T1_t$ gate, which regulates global memory updates, and the $T2_T$ gate, which governs local memory updates. These two gates are parameterized separately, allowing the model to adapt to different time intervals dynamically. This modification allows the model to avoid redundant updates in dense regions while ensuring that important information is retained during long intervals of missing data. By integrating this time-aware mechanism, TIME-DMF can effectively capture both short-term fluctuations and long-term dependencies in complex spatio-temporal data. The time gates are defined as follows:

$$T1_t = \sigma(x_m W_{x1} + \sigma_{\Delta t}((t - t_{last})W_{t1}) + b_1), \tag{15}$$

$$T2_t = \sigma(x_m W_{x2} + \sigma_{\Delta t}((t - t_{last})W_{t2}) + b_2), \tag{16}$$

where t and t_{last} are the time stamps of the input sparse sequence, and $W_{x1}, W_{x2}, W_{t1}, W_{t2}$ are learnable weight matrices.

Once the time gates have been computed, they determine how much past information should be retained or overwritten at each time step. The global memory state, responsible for capturing long-term dependencies, is updated using the $T1$ gate, while the local memory state, which preserves short-term correlations, is controlled by the $T2$ gate.

First, a dynamic weighting factor δ is computed based on both the historical hidden state h and the current input z: $\delta = \sigma(W_\delta[h_{t-1}, z_t] + b_\delta)$. This weighting factor δ is then used to compute an intermediate feature representation a_t: $a_t = \delta \odot h_{t-1} + (1 - \delta) \odot z_t$. This formulation dynamically determines whether the current time step should rely more on past information h_t or the newly observed input z_t, helping to adapt to irregular time intervals. Next, the global and local memory states, $\tilde{C}_t$ and $\hat{C}_t$, are updated based on the time gates $T1_t$ and $T2_t$:

$$\tilde{C}_t = \tilde{C}_{t-1} \odot T1_t + a_t \odot (1 - T1_t), \tag{17}$$

$$\hat{C}_t = \tilde{C}_{t-1} \odot T2_t + a_t \odot (1 - T2_t). \tag{18}$$

Finally, the output at time step t is calculated by the DMF model $f(\cdot)$ using local memory:

$$\hat{y}_t = f(\hat{C}_t) \tag{19}$$

This mechanism addresses the challenge of irregular time intervals by adaptively regulating memory updates, preventing excessive updates in densely sampled regions while preserving crucial information in sparse areas. By incorporating these learnable time-dependent gates, the model effectively captures long-range dependencies without unnecessary computations.

D. Asynchronous Training Process

In FL-based spatio-temporal data completion systems, a key challenge arises from client asynchrony. Unlike centralized training, where data is uniformly collected and processed, FL clients operate independently, collecting data according to personal schedules or regional activity patterns. As a result, the training data spans different and often non-overlapping time periods, leading to asynchronous model updates.

Some clients, such as those in active urban regions, frequently upload model updates due to continuous data collection, whereas others, operating in sparse regions or during off-peak hours, contribute less frequently. Furthermore, the time windows of data collected by different clients may fully overlap, partially overlap, or be entirely disjoint, complicating the aggregation process at the central server. If not handled properly, these inconsistencies can hinder model convergence, especially in time-sensitive tasks.

To handle the issue of misaligned temporal coverages across clients in federated settings, we design a timestamp-aligned asynchronous aggregation mechanism. Instead of waiting for all clients to synchronize, the central server proceeds in short center epochs, aggregating all client updates received within the current time window. At each center epoch, the server aggregates the parameters received up to that point, together with the active timestamps reported by those clients. For each client c_k the server computes timestamp weights $w_{k,t}$based on the coverage, and normalizes these into a single contribution weight W_k for aggregation. At a center epoch the server updates the global model by a weighted FedAvg:

$$w_{t+1} = \sum_{k \in \mathcal{A}_t} W_k w_k^{(t,loc)}, \tag{20}$$

where $\mathcal{A}_t$ denotes the active clients set in current center epoch. By explicitly accounting for temporal coverage and normalizing client weights, the timestamp-aligned asynchronous aggregation prevents frequent contributors from dominating, and ensures that

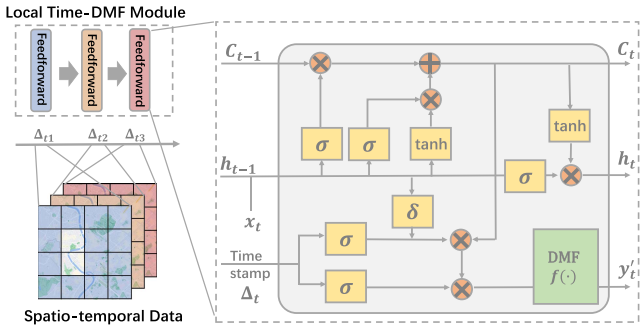

Fig. 4. Structure of TIME-DMF module.

clients covering isolated time intervals still contribute meaningfully. This both preserves the time structure learned by the LSTM in TIME-DMF and reduces the bias introduced by asynchronous, irregular participation.

As shown in Fig. 4, Client A collects data from 7:00 to 7:20, Client B from 7:10 to 7:30, and Client C from 7:40 to 8:00. Each client utilizes sensed private data to refine data completion. After training asynchronously, these clients submit their model parameters at 7:20, 7:30, and 8:00, respectively. Upon receiving updates, the server performs timestamp alignment to determine each client's active timestamps and computes a weight for each client. For the overlap between 7:00 and 7:30, contributions from Client A and Client B are combined using timestamp-aligned weights if they are in the same center epoch. For the non-overlapping intervals (e.g., 7:40–8:00 from Client C), the server also assigns weights to preserve their unique temporal contributions rather than discarding them. Compared with synchronous training, this short-epoch asynchronous strategy accelerates convergence by avoiding idle waiting for straggler clients, while simultaneously incorporating information from temporally isolated clients to further enhance convergence.

E. Local Privacy Enhancement

Since clients transmit both the input matrix X_m and model parameters $f(\cdot)$, an attacker could potentially reconstruct the model output, but they still cannot recover the exact sparse local data. Nevertheless, to further strengthen privacy guarantees, we apply LDP by perturbing client parameters with zero-mean noise before transmission.

Definition 1: (Local Differential Privacy) [36]. A disturbing function $M : X \to R$ satisfies ϵ-local differential privacy (ϵ-LDP), where $\epsilon \geq 0$, if and only if for any input v and v', we have:

$$\forall u \in R(M) : Pr[M(v) = u] \leq e^{\epsilon} Pr[M(v') = u], \quad (21)$$

where $R(M)$ denotes the set of all possible outputs of the disturbing function M.

Based on the above local differential privacy definition, we employ the Laplace Mechanism in the up-link submission process of FL system.

Definition 2: (Laplace Mechanism) [37]. For a numerical query function Q on set v, the Laplace Mechanism M disturbs the query result by:

$$M(v) = Q(v) + Lap\left(\frac{\delta Q}{\epsilon}\right), \quad (22)$$

where $Lap(\frac{\delta Q}{\epsilon})$ denotes the random noise extracted from the Laplace distribution with a scale factor $\frac{\Delta Q}{\epsilon}$.

We quantify it using the standard ϵ-differential privacy metric, where a smaller ϵ indicates stronger privacy protection but comes with higher noise and potentially reduced accuracy. Federated learning itself already prevents direct leakage of personal information. The introduction of LDP further strengthens protection in the up-link communication stage. The injected noise is carefully controlled so that the completion accuracy remains stable, and during aggregation across multiple clients, part of the noise effect is naturally canceled out.

V. Theoretical Analysis

This section provides a theoretical analysis of FLAMF. We first examine its local privacy guarantees via LDP. Then, we analyze spatio-temporal heterogeneity across clients—starting with a spatial data heterogeneity perspective, followed by a convergence analysis under temporal asynchrony. Finally, we assess the computational complexity based on matrix operations.

A. Local Privacy Protection

Theorem 1: (Local Differential Privacy): By introducing Laplace Mechanism $Lap(0, \frac{\Delta Q}{\epsilon})$ in the up-link submission stage, FLAMF satisfies ϵ-local differential privacy.

Proof: Since we add Laplace noise into the up-link submission stage, the Laplace probability density function $Lap((\mu, b)$ is given by two parameters:

$$Lap(x; \mu, b) = \frac{1}{2b} e^{-\frac{|x-\mu|}{b}}. \quad (23)$$

As we introduce $Lap(0, \frac{\Delta Q}{\epsilon})$ into the up-link submission by (22). Given a query function $Q : v \to R$, the query difference is denoted as:

$$\Delta Q = \max_{v,v'} |Q(v) - Q(v')|, \quad (24)$$

where v and v' are adjacent submission sets. Since the Laplace mechanism adds Laplace noise $Lap(0, \frac{\Delta Q}{\epsilon})$ to $Q(v)$, then:

$$Q(v) + Lap\left(0, \frac{\Delta Q}{\epsilon}\right) = Lap(Q(v), \frac{\Delta Q}{\epsilon}). \quad (25)$$

Considering two adjacent sets v and v', and any query output set S, we can derive the following inequality using the symmetry and exponential decay properties of the Laplace distribution:

$$\frac{Pr(Q(v) + Lap(0, \frac{\Delta Q}{\epsilon}) \in S)}{Pr(Q(v') + Lap(0, \frac{\Delta Q}{\epsilon}) \in S)} = e^{\frac{\epsilon |Q(v) - Q(v')|}{\Delta Q}}. \quad (26)$$

Then we have $|Q(v) - Q(v')| \leq \Delta Q$ based on (24). Finally, we derive:

$$\frac{Pr(Q(v) + Lap(0, \frac{\Delta Q}{\epsilon}) \in S)}{Pr(Q(v') + Lap(0, \frac{\Delta Q}{\epsilon}) \in S)} \leq e^{\epsilon}, \quad (27)$$

which proves the Theorem 1. □

B. Heterogeneity Balancing Analysis

We next analyze the spatio-temporal data heterogeneity across clients. Let $Y \in \mathbb{R}^{m\times n}$ be the ground-truth matrix and $L_k \in \{0,1\}^{m\times n}$ be the binary indicator matrix of observed entries for client k, with $\Omega_k = \{(i,j) : L_k(i,j) = 1\}$ and $n_k = |\Omega_k| > 0$.

Theorem 2: (Lipschitz continuity of the local loss): For any prediction matrix $Z \in \mathbb{R}^{m\times n}$ define the loss of client k:

$$F_k(Z) \;=\; \frac{1}{2n_k}\big\|(Y - Z)\odot L_k\big\|_F^2, \tag{28}$$

where $\odot$ denotes the Hadamard product and $\|\cdot\|_F$ the Frobenius norm. Assume there exists $M > 0$ such that $|Y_{ij}| \le M$ and $|Z_{ij}| \le M$ for all $(i,j) \in \Omega_k$ and for all considered Z. Then for any $Z_1, Z_2 \in \mathbb{R}^{m\times n}$ we have

$$\big|F_k(Z_1) - F_k(Z_2)\big| \le L_k \big\|(Z_1 - Z_2)\odot L_k\big\|_F, \tag{29}$$

with $L_k = 2M/\sqrt{n_k}$.

Proof: By definition,

$$\begin{aligned} &F_k(Z_1) - F_k(Z_2)\\ &= \frac{1}{2n_k}\sum_{(i,j)\in\Omega_k}\left[(Y_{ij} - Z_{1,ij})^2 - (Y_{ij} - Z_{2,ij})^2\right]\\ &= \frac{1}{2n_k}\sum_{(i,j)\in\Omega_k}(Z_{2,ij} - Z_{1,ij})\,(2Y_{ij} - Z_{1,ij} - Z_{2,ij})\\ &= \frac{1}{2n_k}\big\langle (Z_1 - Z_2)\odot L_k,\; (2Y - Z_1 - Z_2)\odot L_k\big\rangle, \end{aligned} \tag{30}$$

where $\langle\cdot,\cdot\rangle$ denotes the Frobenius inner product.

Applying the Cauchy–Schwarz inequality yields:

$$\begin{aligned} &\big|F_k(Z_1) - F_k(Z_2)\big|\\ &\le \frac{1}{2n_k}\big\|(Z_1 - Z_2)\odot L_k\big\|_F\,\big\|(2Y - Z_1 - Z_2)\odot L_k\big\|_F. \end{aligned} \tag{31}$$

Under the condition that $|Y_{ij}|$ and $|Z_{l,ij}|$ are bounded by M for $l = 1, 2$, we have $|2Y_{ij} - Z_{1,ij} - Z_{2,ij}| \le 4M$ and therefore

$$\big\|(2Y - Z_1 - Z_2)\odot L_k\big\|_F \le 4M\sqrt{n_k}. \tag{32}$$

Combining the above estimates gives

$$\begin{aligned} \big|F_k(Z_1) - F_k(Z_2)\big| &\le \frac{4M\sqrt{n_k}}{2n_k}\,\big\|(Z_1 - Z_2)\odot L_k\big\|_F\\ &= \frac{2M}{\sqrt{n_k}}\,\big\|(Z_1 - Z_2)\odot L_k\big\|_F, \end{aligned} \tag{33}$$

which proves the claim with $L_k = 2M/\sqrt{n_k}$. □

Theorem 2 suggests each local loss function $F_k(\theta)$ is L-Lipschitz continuous, and with a bounded parameter space Θ, for any clients i, j, there exists a constant $\delta > 0$ such that: $|F_i(\theta) - F_j(\theta)| \le \delta, \quad \forall\theta \in \Theta$. Let θ_i^t be the model trained on client i after local updates at round t, and based on FedAvg:

$$\theta^{t+1} = \sum_{i=1}^{K} W_i\theta_i^t. \tag{34}$$

Then from the convexity of squared error and linearity of expectation:

$$F(\theta^{t+1}) = F\left(\sum_{i=1}^{K} W_i\theta_i^t\right) \le \sum_{i=1}^{K} W_i F_i(\theta_i^t) + \mathcal{O}(\delta), \tag{35}$$

where the residual $\mathcal{O}(\delta)$ comes from the bounded discrepancy between local objectives.

This indicates that despite the presence of spatial heterogeneity in client data, the weighted averaging in FedAvg can still approximate the minimizer of the global objective within a bounded error.

C. Convergence Analysis

Let $f(w) = \frac{1}{K}\sum_{k=1}^{K} f_k(w)$ be the global objective. Consider the asynchronous update of active clients set $\mathcal{A}_t$ in time step t:

$$w_{t+1} = w_t - \eta\sum_{k\in\mathcal{A}_t} W_k^{(t)} g_k^{(t)}, \tag{36}$$

with step size $0 < \eta \le \eta_0$ for some constant η_0. To theoretically analyze the convergence of timestamp-aligned asynchronous aggregation, we begin by defining the following six assumption.

Definition 3: (L-smoothness). Each f_k is L-smooth:

$$f_k(y) \le f_k(x) + \langle\nabla f_k(x), y - x\rangle + \frac{L}{2}\|y - x\|^2, \quad \forall x, y. \tag{37}$$

Definition 4: (Unbiased bounded variance). For all k, t,

$$\mathbb{E}[g_k^{(t)} \mid w_{t_k}] = \nabla f_k(w_{t_k}), \qquad \mathbb{E}\|g_k^{(t)} - \nabla f_k(w_{t_k})\|^2 \le \sigma^2. \tag{38}$$

Definition 5: (Bounded heterogeneity). For all w,

$$\frac{1}{K}\sum_{k=1}^{K}\|\nabla f_k(w) - \nabla f(w)\|^2 \le \zeta^2. \tag{39}$$

Definition 6: (Bounded staleness). There exists $S_{\max} \ge 0$ such that for every processed update $t - t_k \le S_{\max}$.

Definition 7: (Weight normalization and boundedness). For every t, $\sum_{k\in\mathcal{A}_t} W_k^{(t)} = 1$ and $0 \le W_k^{(t)} \le A_{\max}$.

Definition 8: (Bounded second moment). There exists nonnegative G such that for all k, t, $\mathbb{E}\|g_k^{(t)}\|^2 \le G^2$.

Our proposed LSTM based TIME-DMF model and the corresponding timestamp-aligned asynchronous aggregation naturally satisfies these definitions.

Theorem 3: (Convergence of timestamp-aligned asynchronous aggregation): There exist constants $C_0, C > 0$ such that the timestamp-aligned asynchronous aggregation asymptotically converges to a stationary point:

$$\begin{aligned} \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(w_t)\|^2 \;&\le\; C_0\,(f(w_0) - f^\star)/\eta T\\ &+ C\left(\eta A_{\max}\sigma^2 + A_{\max}\zeta^2 + \eta^2 S_{\max}^2 G^2\right), \end{aligned} \tag{40}$$

where $f^\star = \inf_w f(w)$.

Proof: By L-smoothness of f and the asynchronous update,

$$f(w_{t+1}) \le f(w_t)+\langle \nabla f(w_t), w_{t+1}-w_t\rangle+\frac{L}{2}\|w_{t+1}-w_t\|^2$$
$$= f(w_t) - \eta\Big\langle \nabla f(w_t), \sum_{k\in\mathcal{A}_t} W_k^{(t)} g_k^{(t)}\Big\rangle$$
$$+ \frac{L\eta^2}{2}\Big\| \sum_{k\in\mathcal{A}_t} W_k^{(t)} g_k^{(t)}\Big\|^2. \tag{41}$$

Take total expectation and using Definition 4 we get

$$\mathbb{E}[f(w_{t+1})] \le \mathbb{E}[f(w_t)] + \frac{L\eta^2}{2}\,\mathbb{E}\Big\| \sum_{k\in\mathcal{A}_t} W_k^{(t)} g_k^{(t)}\Big\|^2$$
$$- \eta\,\mathbb{E}\Big\langle \nabla f(w_t), \sum_{k\in\mathcal{A}_t} W_k^{(t)} \nabla f_k(w_{t_k})\Big\rangle. \tag{42}$$

To handle the mismatch between the global gradient and the aggregated local gradients, we add and subtract $\nabla f(w_t)$ inside the inner product. This decomposition isolates the ideal descent term $|\nabla f(w_t)|^2$ from two error components: the staleness bias (due to clients computing gradients on stale models w_{t_k} rather than w_t) and the heterogeneity bias (arising from differences between local and global gradients even at the same w_t).

$$b_t := \sum_{k\in\mathcal{A}_t} W_k^{(t)} \left(\nabla f_k(w_{t_k}) - \nabla f_k(w_t)\right),$$
$$h_t := \sum_{k\in\mathcal{A}_t} W_k^{(t)} \left(\nabla f_k(w_t) - \nabla f(w_t)\right). \tag{43}$$

Then the inner product in (42) derives:

$$\Big\langle \nabla f(w_t), \sum_{k} W_k^{(t)} \nabla f_k(w_{t_k})\Big\rangle$$
$$= \|\nabla f(w_t)\|^2 + \langle \nabla f(w_t), b_t + h_t\rangle. \tag{44}$$

Hence from the descent inequality,

$$\mathbb{E}[f(w_{t+1})] \le \mathbb{E}[f(w_t)] - \eta\mathbb{E}\|\nabla f(w_t)\|^2$$
$$- \eta\mathbb{E}\langle \nabla f(w_t), b_t + h_t\rangle + \frac{L\eta^2}{2}\,\mathbb{E}\Big\| \sum_{k} W_k^{(t)} g_k^{(t)}\Big\|^2. \tag{45}$$

Apply Young's inequality to the cross term: for any $\alpha > 0$,

$$-\eta\langle \nabla f(w_t), b_t + h_t\rangle \le \frac{\eta}{2\alpha}\|\nabla f(w_t)\|^2 + \frac{\eta\alpha}{2}\|b_t + h_t\|^2. \tag{46}$$

Choose $\alpha = 1$ (other choices only change constants). Then

$$-\eta\langle \nabla f(w_t), b_t + h_t\rangle \le \frac{\eta}{2}\|\nabla f(w_t)\|^2 + \frac{\eta}{2}\|b_t + h_t\|^2. \tag{47}$$

Substitute into (45) to obtain

$$\mathbb{E}[f(w_{t+1})] \le \mathbb{E}[f(w_t)] - \frac{\eta}{2}\mathbb{E}\|\nabla f(w_t)\|^2$$
$$+ \frac{\eta}{2}\mathbb{E}\|b_t + h_t\|^2 + \frac{L\eta^2}{2}\,\mathbb{E}\Big\| \sum_{k} W_k^{(t)} g_k^{(t)}\Big\|^2. \tag{48}$$

Next, we bound the staleness term b_t. Using Lipschitzness of each ∇f_k,

$$\|b_t\| \le L\sum_k W_k^{(t)}\|w_{t_k} - w_t\|. \tag{49}$$

For each active client time step s, we have $t - t_k \le S_{\max}$ and

$$\|w_{t_k} - w_t\| \le \sum_{s=t_k}^{t-1} \|w_{s+1} - w_s\|$$
$$= \eta \sum_{s=t_k}^{t-1} \Big\| \sum_{j\in\mathcal{A}_s} a_j^{(s)} g_j^{(s)}\Big\|. \tag{50}$$

Therefore

$$\|b_t\| \le L\eta \sum_{s=t-S_{\max}}^{t-1} \Big\| \sum_{j\in\mathcal{A}_s} a_j^{(s)} g_j^{(s)}\Big\| \tag{51}$$

and by Cauchy–Schwarz,

$$\|b_t\|^2 \le L^2\eta^2 S_{\max} \sum_{s=t-S_{\max}}^{t-1} \Big\| \sum_{j\in\mathcal{A}_s} a_j^{(s)} g_j^{(s)}\Big\|^2. \tag{52}$$

Taking expectation and use Definition 8, we have:

$$\Big\| \sum_j a_j^{(s)} g_j^{(s)}\Big\|^2 \le \left(\sum_j a_j^{(s)}\right)^2 \max_j \|g_j^{(s)}\|^2. \tag{53}$$
$$\mathbb{E}\|b_t\|^2 \le L^2\eta^2 S_{\max}^2 G^2. \tag{54}$$

Then we bound heterogeneity term h_t. Using Definition 7, we have

$$\|h_t\|^2 = \Big\| \sum_k W_k^{(t)} (\nabla f_k(w_t) - \nabla f(w_t))\Big\|^2$$
$$\le \left(\sum_k (W_k^{(t)})^2\right)\left(\sum_k \|\nabla f_k(w_t) - \nabla f(w_t)\|^2\right). \tag{55}$$

Taking expectation and using Definition 5 yields:

$$\mathbb{E}\|h_t\|^2 \le A_{\max}\sum_{k=1}^{K} \mathbb{E}\|\nabla f_k(w_t) - \nabla f(w_t)\|^2 \le A_{\max}\,\zeta^2. \tag{56}$$

At last we need to bound the aggregated second moment.

$$\sum_k W_k^{(t)} g_k^{(t)} = \sum_k W_k^{(t)} \nabla f_k(w_{t_k})$$
$$+ \sum_k W_k^{(t)} \left(g_k^{(t)} - \nabla f_k(w_{t_k})\right). \tag{57}$$

Thus by Cauchy–Schwarz, we derive (57):

$$\mathbb{E}\Big\| \sum_k W_k^{(t)} g_k^{(t)}\Big\|^2 \le 2\mathbb{E}\Big\| \sum_k W_k^{(t)} \nabla f_k(w_{t_k})\Big\|^2$$
$$+ 2\mathbb{E}\Big\| \sum_k W_k^{(t)} \left(g_k^{(t)} - \nabla f_k(w_{t_k})\right)\Big\|^2 \le 6\mathbb{E}\|\nabla f(w_t)\|^2$$
$$+ 6\mathbb{E}\|h_t\|^2 + 6\mathbb{E}\|b_t\|^2 + 2\sum_k (W_k^{(t)})^2\sigma^2, \tag{58}$$

where we used (43) to derive $\sum_k W_k^{(t)} \nabla f_k(w_{t_k}) = \nabla f(w_t) + h_t + b_t$. Using $\sum_k (W_k^{(t)})^2 \leq A_{\max}$, we obtain

$$\mathbb{E}\Big\| \sum_k W_k^{(t)} g_k^{(t)} \Big\|^2 \leq 6\mathbb{E}\|\nabla f(w_t)\|^2 + 6\mathbb{E}\|h_t\|^2 + 6\mathbb{E}\|b_t\|^2 + 2A_{\max}\sigma^2. \quad (59)$$

Finally, we combine these bounds into descent. From (48), (56) and (59),

$$\mathbb{E}[f(w_{t+1})] \leq \mathbb{E}[f(w_t)] - \frac{\eta}{2}\mathbb{E}\|\nabla f(w_t)\|^2 + \frac{\eta}{2}\left(2A_{\max}\zeta^2 + 2L^2\eta^2 S_{\max}^2 G^2\right) + \frac{L\eta^2}{2}\left(6\mathbb{E}\|\nabla f(w_t)\|^2 + 6A_{\max}\zeta^2 + 6L^2\eta^2 S_{\max}^2 G^2 + 2A_{\max}\sigma^2\right). \quad (60)$$

Collect terms in $\mathbb{E}\|\nabla f(w_t)\|^2$:

$$\mathbb{E}[f(w_{t+1})] \leq \mathbb{E}[f(w_t)] - \left(\frac{\eta}{2} - 3L\eta^2\right)\mathbb{E}\|\nabla f(w_t)\|^2 + C_1\left(\eta A_{\max}\zeta^2 + \eta^3 L^2 S_{\max}^2 G^2 + \eta^2 A_{\max}\sigma^2\right), \quad (61)$$

for some absolute constant C_1. Choose η small enough so that $\frac{\eta}{2} - 3L\eta^2 \geq \frac{\eta}{4}$,

$$\mathbb{E}[f(w_{t+1})] \leq \mathbb{E}[f(w_t)] - \frac{\eta}{4}\mathbb{E}\|\nabla f(w_t)\|^2 + C_2\left(\eta A_{\max}\zeta^2 + \eta^2 A_{\max}\sigma^2 + \eta^3 S_{\max}^2 G^2\right). \quad (62)$$

Rearrange and sum from $t = 0$ to $T - 1$:

$$\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(w_t)\|^2 \leq 4\left(\mathbb{E}[f(w_0)] - \mathbb{E}[f(w_T)]\right)/\eta + 4C_2\sum_{t=0}^{T-1}\left(A_{\max}\zeta^2 + \eta A_{\max}\sigma^2 + \eta^2 S_{\max}^2 G^2\right). \quad (63)$$

Using $f(w_T) \geq f^\star$ and dividing by T yields

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(w_t)\|^2 \leq 4\left(f(w_0) - f^\star\right)/\eta T + 4C_2\left(A_{\max}\zeta^2 + \eta A_{\max}\sigma^2 + \eta^2 S_{\max}^2 G^2\right). \quad (64)$$

Rename constants $C_0 = 4$ and $C = 4C_2$:

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(w_t)\|^2 \leq C_0\left(f(w_0) - f^\star\right)/\eta T + C\left(\eta A_{\max}\sigma^2 + A_{\max}\zeta^2 + \eta^2 S_{\max}^2 G^2\right). \quad (65)$$

□

D. Computational Complexity Analysis

We present the effectiveness of the proposed model from the perspective of computational complexity.

Theorem 4: (Computation Complexity): Given the active clients number M in a specific aggregation time T, each client trains collected data x_t the computation complexity of our proposed FLAMF: $\Lambda(\sum f(x_t^m)) = O(\frac{1}{M})$.

Proof: The entire central aggregation time period T is divided into several time steps $t \in (0, T]$, and time t involves an input data slice x_t collected by local user. Suppose that the concatenated input $[h_{t-1}, y'_{[t][n]}] \in \mathbb{R}^{n\times(t+h)}$ and with the weight matrices $W \in \mathbb{R}^{d\times n}$, the computation complexity of computing $W[h_{t-1}, y'_{[t][n]}]$ can be measured as:

$$\Lambda(W[h_{t-1}, y'_{[t][n]}]) = O(d \times n \times (t+h)). \quad (66)$$

For each time step t, the operations for the gates and cells state involve matrix multiplications and element-wise operations. For each gate from (9) to (12), the computation complexity of matrix operations is: $O(dnt + dnh)$. And the element-wise operations have a cost of $O(d)$. Therefore, the total computation complexity of the four gates in one time step is $O(4dnt + 4dnh + 4d) = O(dnt + dnh)$. The L-layer FLAMF that operates across all time steps T has a computation complexity:

$$O(L \times T \times (dnt + dnh)). \quad (67)$$

We can uniformly divide the total time step T into K time slices $t = \frac{T}{K}$, and the dimension of the hidden state is proportional to the input data dimension $h = \alpha t = \alpha\frac{T}{K}$. Thus the total computation complexity of FLAMF is presented as:

$$O\left(LTdn \times \left(\frac{T}{K} + \alpha\frac{T}{K}\right)\right) = O\left(\frac{1}{K}\right), \quad (68)$$

which is solely determined by the number of time step K. As the number of clients increases, the data collection frequency rises, resulting in more densely packed time steps. Since FLAMF uses FedAvg to aggregate the global model, the computational complexity FLAMF is:

$$\frac{1}{M}\sum_{1}^{M}(\Lambda(W[h_{t-1}, y'_{[t][n]}])) = O(\frac{1}{M}), \quad (69)$$

which proofs Theorem 3. As the number of clients increases, the data collection frequency rises, resulting in more densely packed time steps. □

For the MF component, the per-iteration complexity is proportional to the number of observed entries, $O(|\Omega|d)$, where $|\Omega|$ denotes the number of nonzero entries and d the latent dimension. In real-world SMCS scenarios, each client only contributes data for a small subset of locations and time slices. This means that the local input matrices are of relatively low dimension, and the effective N and T values per client are modest. The sparsity of user data thus greatly reduces the actual computational cost compared with the theoretical worst case. For instance, in the Sensor-Scope dataset, with $N = 55$ locations, $T = 312$ time slices, and sparsity $\rho = 10\%$, each client processes approximately $|\Omega| \approx 1{,}700$ entries, resulting in a modest computational load.

As for the communication cost, only the model parameters are transmitted. The parameter count of the TIME-DMF model can be estimated by summing all trainable modules: the latent matrix $Z \in \mathbb{R}^{z\times T}$, the RNN-related linear mappings, the temporal modulation modules T_1 and T_2, and the DMF projection layers. Each linear layer contributes in × out weights plus out

TABLE II
STATISTICS OF THREE EVALUATION DATASETS

	Sensor-Scope	U-Air	Traffic Volume
City	Lausanne(Switzerland)	Beijing(China)	Sydney(Australia)
Data	Humidity	PM2.5	Traffic volume
Subarea	57subareas each with 50*30 m^2	36 subareas each with 1000*1000 m^2	100 traffic count stations selected in Sydney as subareas
Cycle & Duration	0.5h&7d	1h&11d	1h&7d
Mean	84.52±6.32%	79.11±81.21	492.27±473.84

biases. Under the experimental settings, this yields a total parameter count on the order of 10^4 , i.e., 9.8 K–14.7 K depending on the dataset, which translates to 0.037–0.056 MB of model size. Therefore, one communication round (upload+download) requires less than 0.12 MB per client, which is only tens of kilobytes per submission. In practice, FLAMF converges within limited communication rounds. This efficiency ensures the overall communication overhead is approximately 0.6 MB, 1.2 MB, and 1.8 MB, respectively.

VI. EXPERIMENTS

A. Datasets

We conduct experiments across three datasets. We show the main information about the datasets in Table II and the detailed description as follows:

- The *Sensor-Scope*[1] dataset, which includes measurements of temperature, humidity, and other variables, is collected regularly through numerous static sensors deployed across the EPFL campus.
- The *U-Air* [38] is an air quality dataset, which includes PM2.5, PM10, and other measurements. It comprises latitude and longitude coordinates for Points of Interest (POIs) as well as Air Quality Index (AQI) records.
- The *Traffic Volume* [2] is a data map of traffic flow information, which collects data through sensors deployed at traffic collection stations in NSW, Australia. It monitors the traffic flow, such as the number or the type of vehicles at over 60 stations.

B. Experimental Settings

In the experiments, our dataset is randomly masked according to the sparsity rate, and the retained data is regarded as the collected data. We define the "sparsity rate" as the proportion of valid (non-zero) data points in the dataset. In this experiment, the sparsity rate is set between 0.1 and 0.3. Specially, for the centralized MF, the sparsity rate is up to 0.5 due to the fairness consideration. Then, in the federated learning framework experiments, we set the number of clients to range from 2 to 5.For model training in both the proposed FLAMF and the baseline methods, we use Adam [39] as the optimizer with an initial learning rate of 0.001. In FLAMF, the hidden state dimension is set to $d_h = 128$, with a batch size equal to the total number of location grid cells. In each client, the training proceeds for 30 epochs and then upload the parameters to the server for aggregation. We consider such 5 aggregation as one center epoch. All experiments are conducted using Intel Core i7-10700 CPU with 2.9 GHz. We use Training Loss and Validation Loss as our evaluation metrics, where the Training Loss is the average training loss among clients and the Validation Loss is the MSE with the ground truth after one center epoch. To account for statistical fluctuations and ensure the robustness of our results, all experimental outcomes are averaged over 10 independent runs.

[1]http://sensorscope.epfl.ch/network_code

[2]https://www.rms.nsw.gov.au/about/corporate-publications/statistics/traffic-volumes/aadt-map/

C. Compare Algorithms

MF with deep neural networks (DMF) [40]: A centralized MF model that directly gathers the training data from data providers. The method combines MF and DNNs, aiming to effectively recover the missing data by exploiting the nonlinear relationships inherent in the data.

FL-based MF with DNNs (FL-DMF): A distributed training system where clients employ DNNs to perform MF to protect local data privacy. After several training rounds, clients submit their model parameters to the central server, and the server aggregates these parameters to a global MF model.

FL-based MF with RNNs (FL-RMF): Improved based on the previous algorithm by replacing DNNs with RNNs. This kind of network structure has the capability of maintaining a memory of previous inputs. This enhancement increases the ability to discover temporal relationships. This method uses a synchronous data updating approach.

Transformer-based model (TRANS): A self-attention architecture that captures long-range temporal dependencies and global correlations across spatial dimensions [41].

Temporal Convolutional Network (TCN): A dilated causal convolutional network that efficiently models sequential patterns with flexible receptive fields [42].

Matrix Factorization (FedMF): A classical latent vector embedding method that approximates the incomplete sensing matrix by low-rank decomposition in a federated manner [33].

D. Performance Evaluation Across FL Methods

We first compare the performance of FLAMF with other MF-based methods. We set the number of clients to 5, and the sparsity rate is 0.1. As shown in Figs. 6 and 7, our method outperforms FL-RMF and FL-DMF in terms of convergence speed and loss.

It is worth clarifying that FL-DMF and FL-RMF are the baseline models specifically designed in this study for fair comparison. FL-DMF extends the DMF into the federated learning framework, allowing us to evaluate how DMF performs in a distributed training environment. To further examine the role of temporal modeling, we construct FL-RMF by replacing the deep

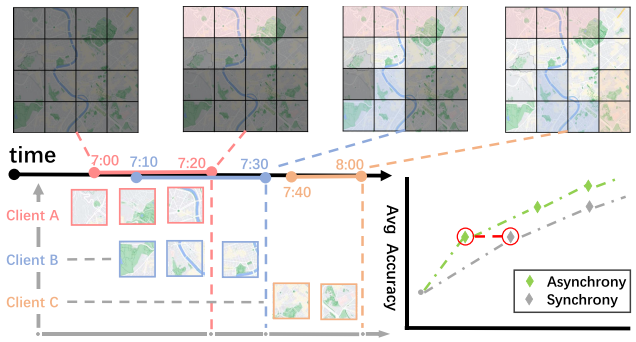

Fig. 5. An example of asynchronous training showing faster convergence and improved stability compared to synchronous updates.

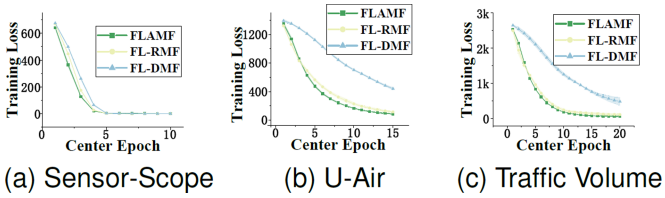

Fig. 6. Training loss across FL-based methods, showing that FLAMF achieves faster convergence, especially in more complex spatio-temporal scenarios.

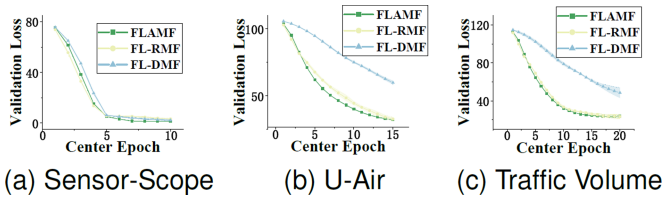

Fig. 7. Validation loss across FL-based methods, showing that FLAMF achieves faster convergence and lower final loss.

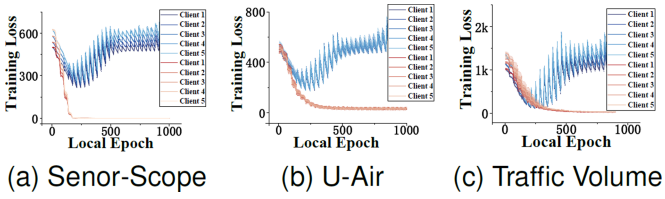

Fig. 8. Training loss of five clients under TRANS and FLAMF methods, demonstrating that FLAMF achieves more stable convergence.

neural network component in FL-DMF with RNNs. This design enables us to contrast the capability of short-term temporal memory with our proposed asynchronous TIME-DMF model. However, we find that FL-RMF is limited in handling asynchronous aggregation, as it cannot effectively capture long-term temporal dependencies.

We further compare FLAMF with three representative baselines: TRANS, TCN, and FedMF. All three baselines rely on a synchronous aggregation mode, and for fair comparison we report the training trajectories across local epochs. The results in Figs. 8 to 10 show that FLAMF converges faster and more

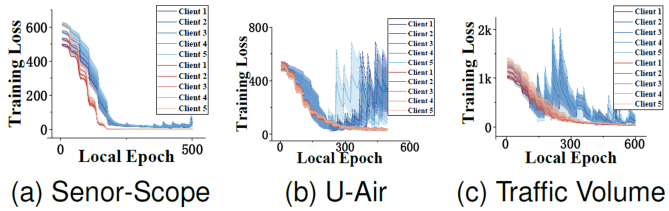

Fig. 9. Training loss of five clients under TCN and FLAMF methods, demonstrating that FLAMF achieves more stable convergence.

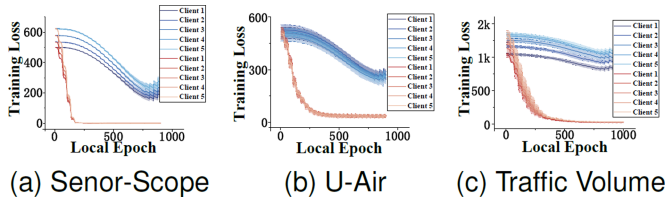

Fig. 10. Training loss of five clients under FedMF and FLAMF methods, demonstrating that FLAMF achieves more stable convergence.

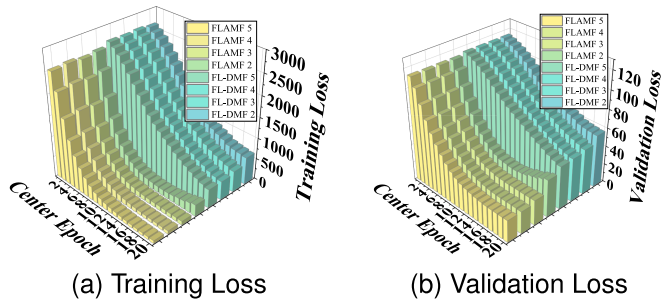

Fig. 11. Performance evaluations on clients number, showing that FLAMF achieves faster convergence with more clients.

stably than Transformer and TCN, which suffer from asynchronous updates and misaligned sequences. Compared with FedMF, FLAMF achieves significantly higher accuracy by effectively capturing nonlinear temporal dependencies and handling variable-length inputs. These advantages are particularly evident under high sparsity, where FLAMF consistently outperforms all competitors.

Nonlinear correlations in datasets often require frequent updates to capture complex relationships. It is important to note that the Traffic Volume and UAir datasets exhibit more nonlinear correlations compared to the Sensor-Scope dataset. The results indicate that FLAMF and FL-RMF can effectively handle it, especially when dealing with sequential, time-dependent, and complex urban datasets. After a certain central epoch, the loss may increase as the noise introduced by LDP is amplified during aggregation. This reflects a trade-off between privacy preservation and data accuracy, as the addition of noise enhances privacy but can affect model performance.

E. Evaluations on Clients Number and Sparsity Rates

Fig. 11 presents the evaluation results of the number of clients from 2 to 5 in FLAMF and FL-DMF. We can observe that

TABLE III
LDP INTRODUCES SLIGHT NOISE-INDUCED LOSS INCREASE BUT MAINTAINS EFFICIENT CONVERGENCE, ESPECIALLY WITH MORE CLIENTS

	Traffic Volume						Sensor Scope						U-Air					
	5 clients		4 clients		3 clients		5 clients		4 clients		3 clients		5 clients		4 clients		3 clients	
	LDP	No	LDP	No	LDP	No	LDP	No	LDP	No	LDP	No	LDP	No	LDP	No	LDP	No
Epoch 11	29.4	29.6	29.7	29.4	30.5	29.71	1.3	1.3	3.4	2.6	3.3	3.0	37.3	37.7	37.9	37.4	39.3	37.7
Epoch 12	27.6	27.6	27.8	27.5	29.0	27.8	1.4	1.3	3.1	2.2	3.2	2.6	35.3	35.7	35.9	35.4	37.2	35.7
Epoch 13	26.3	26.1	26.2	26.0	27.7	26.5	1.4	1.3	2.9	2.0	3.0	2.3	33.8	34.2	34.4	33.9	35.5	34.2
Epoch 14	25.2	25.0	25.2	25.0	27.5	25.5	**1.3**	1.3	**2.8**	1.8	2.9	2.1	32.7	33.0	33.1	32.8	34.4	33.0
Epoch 15	24.4	24.2	24.7	24.3	27.0	24.8	1.4	1.3	2.8	1.6	**2.8**	1.9	32.1	32.1	32.2	32.0	33.7	32.1
Epoch 16	23.9	23.7	24.4	23.8	27.1	24.2	1.4	1.3	2.7	1.6	2.8	1.7	31.4	31.4	31.6	31.2	33.0	31.4
Epoch 17	**23.8**	23.4	**24.4**	23.5	27.1	23.9	1.4	1.3	2.8	1.5	2.8	1.6	30.9	30.8	31.1	30.6	33.0	30.9
Epoch 18	23.8	23.3	24.5	23.3	**27.0**	23.7	1.4	1.3	2.7	1.5	2.9	1.5	30.6	30.5	30.8	30.2	**32.4**	30.5
Epoch 19	23.8	23.3	24.4	23.1	27.0	23.6	1.5	1.3	2.7	1.4	2.9	1.4	30.5	30.3	**30.8**	29.9	32.8	30.3
Epoch 20	24.0	23.3	24.1	22.9	27.1	23.6	1.5	1.3	2.7	1.4	3.0	1.4	**30.4**	30.1	30.8	29.8	33.7	30.3

TABLE IV
RESULTS OF INFERENCE TIME: TIME-DMF ACHIEVES PRACTICAL EFFICIENCY WITH ACCEPTABLE LATENCY

	DMF	RMF	TIME-DMF	TRANS	TCN	MF
Inference time	0.092 ± 0.010	0.853 ± 0.136	2.581 ± 0.243	1.832 ± 0.187	1.731 ± 4.442	0.125 ± 0.036

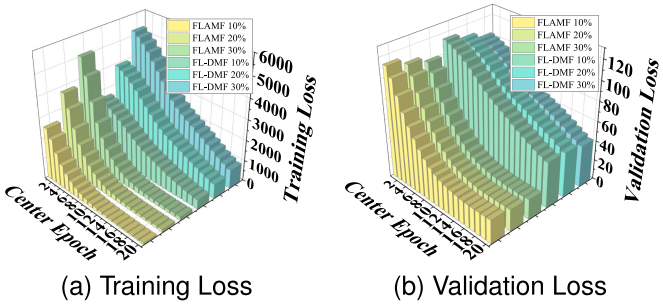

Fig. 12. Performance evaluations on sparsity rates, showing that FLAMF is robust to missing data.

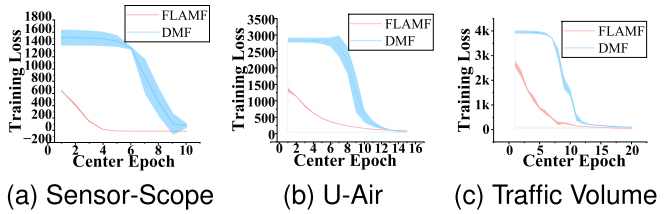

Fig. 13. Training loss between FLAMF and DMF, showing faster convergence of the FL method.

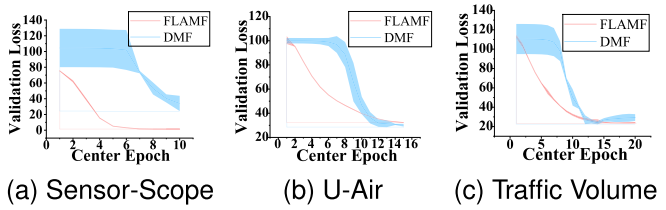

Fig. 14. Validation loss between FLAMF and DMF, showing the stability of FLAMF.

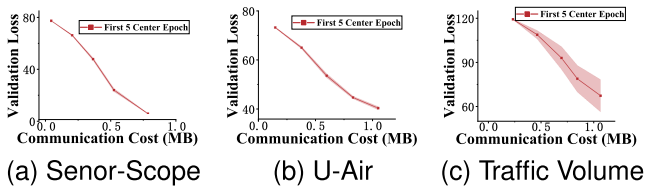

Fig. 15. Communication cost analysis in the first 5 center epochs.

the number of clients indeed affects the convergence speed, as FLAMF benefits from diverse client contributions, while synchronous models such as FL-DMF face significant delays due to the need for client synchronization. For FLAMF, while maintaining consistently fast convergence, the reduction in the number of clients does not significantly increase the loss value, as it preserves hidden nonlinear correlations and efficiently utilizes contributions from available clients.

Fig. 12 presents the evaluation results of the sparsity rates from 0.1 to 0.3 in FLAMF and FL-DMF. It is worth noting that the sparsity rates subtly affect the convergence efficiency for FLAMF but rather heavily disturb that for FL-DMF. FLAMF benefits from asynchronous training, which incorporates updates as they arrive and adapt continuously and preserve hidden temporal correlations.

F. Evaluations of Computation and Communication Overhead

For computational cost, we compared the inference time of different data completion models (Table IV). The results show that the inference overhead of our model is within a practical range, and the communication volume remains acceptable while the model converges steadily.

For communication overhead, we conducted a simulation to record the actual communication volume during the first five center epochs. As shown in Fig. 15, the curves track the performance over the first five center epochs. It can be observed that for all three datasets, the total communication volume remains at a low level of approximately 1 MB during this phase, while the validation loss decreases rapidly. This demonstrates that our method achieves fast model convergence with low communication overhead.

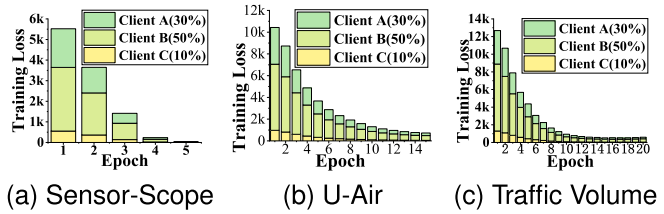

Fig. 16. Evaluation of the imbalance in clients' data, showing that performance remains stable.

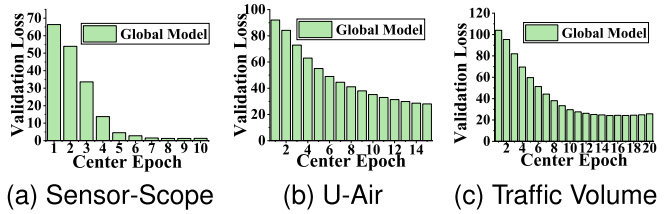

Fig. 17. Evaluation of the imbalance in center server, showing it does not significantly affect performance.

G. Evaluation of the Imbalance in Clients' Data

We evaluate the impact of client data imbalance on distributed completion. In our setup, three clients (A: 30%, B: 50%, C: 10%) participated. Figs. 16 and 17 show the training loss and the validation loss. Results indicate that imbalance does not significantly affect performance, as FL aggregation normalizes client contributions. Interestingly, Client B exhibits the highest training loss due to its larger dataset, since client loss is computed as a summation of per-sample terms (8). These findings highlight the robustness of FLAMF in handling imbalance of distributed data completion.

H. Ablation Experiments

1) Performance Evaluation Between FL Method and non-FL Method: We first compare the performance of FLAMF with the centralized approach DMF. We set the number of clients to 5 and, for fairness, the sparsity rates to 0.1 and 0.5, respectively. As shown in Figs. 13 and 14, our FL-based method significantly outperforms DMF in terms of convergence speed. Additionally, since FLAMF incorporates LDP to ensure communication privacy, its loss is slightly higher than that of DMF, which is generally acceptable. These evaluation results indicate that FLAMF, in addition to preserving privacy, is inherently distributed and resilient to data imbalance as the number of clients or devices increases.

2) Analysis of the Asynchronous Training in FLAMF: To further examine the effect of asynchronous aggregation, we compare our proposed TIME-DMF trained under synchronous FL and asynchronous FL (FLAMF). Fig. 18 reports the validation loss with respect to the number of central epochs on the datasets.The synchronous variant exhibits noticeably slower convergence in terms of validation loss. In contrast, the timestamp-aligned asynchronous mechanism enables more stable convergence. The validation loss curves under central

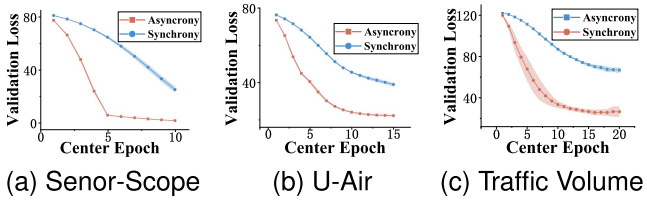

Fig. 18. Validation loss comparison, showing that asynchronous training achieves faster convergence and improved stability.

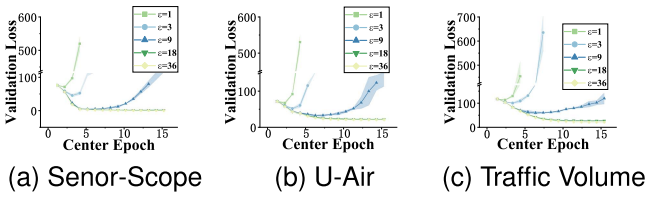

Fig. 19. Validation loss of different privacy budgets, showing the trade-off between privacy and convergence.

epochs demonstrate the benefits of adopting an asynchronous training process.

3) Performance Evaluation of the LDP Mechanism: Table III records the ablation experiments on the use of LDP, showing the changes in validation loss from 10 to 20 central epochs. The column under "NO" reports the performance results without incorporating LDP, and the underlined values indicate the final converged validation loss. First, we find that with LDP enabled, the validation loss tends to increase slightly after reaching convergence due to the noise introduced. In contrast, without LDP, the validation loss continues to decrease smoothly. Second, the results indicate that the convergence speed is marginally affected by the LDP, suggesting that the system maintains reasonable efficiency under privacy constraints. Furthermore, as the number of clients increases, the model converges faster and achieves a lower final validation loss (even when LDP is applied) implying that larger federated networks can offset the noise impact through richer aggregated information. Lastly, the convergence speed without LDP is consistently slightly faster across different client configurations. These findings collectively validate the practicality of LDP in distributed spatio-temporal learning.

In addition, we conducted experiments with different privacy budgets ϵ. The results are presented in Fig. 19. With ϵ=18 as the default setting, the model achieves stable convergence and high completion accuracy. When ϵ is larger than 18, convergence in the early rounds is slightly faster, but the validation error after ten central epochs shows little difference. As ϵ decreases, stronger privacy protection is obtained at the cost of increased noise; convergence becomes slower, and in the extreme case of ϵ=1, the excessive perturbation prevents the model from converging. These results demonstrate that while too small a privacy budget may hinder training, choosing a moderate value such as ϵ=18 achieves a good balance between privacy and utility.

VII. Conclusion

We focus on the issue of privacy leakage in distributed data completion. We present that integrating data completion

methods into the FL framework raises several challenges, including private data availability, limited distributed data volume and real-time update. We propose FLAMF that integrates an enhanced LSTM-based DMF model to capture nonlinearity in the data. By leveraging the decentralized nature of FL, the framework enables asynchronous client updates and supports direct output of completion results at the central server with privacy guarantees. Finally, several theoretical analyses and extensive experimental evaluations show that FLAMF outperforms other FL-based methods as well as non-FL methods.

REFERENCES

[1] E. Wang et al., "A new data completion perspective on sparse crowdsensing: Spatiotemporal evolutionary inference approach," *IEEE Trans. Mobile Comput.*, vol. 24, no. 3, pp. 1357–1371, Mar. 2025.

[2] A. Badugu, K. Arunab, A. Mathew, and P. Sarwesh, "Spatial and temporal analysis of urban heat island effect over Tiruchirappalli city using geospatial techniques," *Geodesy Geodynamics*, vol. 14, no. 3, pp. 275–291, 2023.

[3] H. Dui, S. Zhang, M. Liu, X. Dong, and G. Bai, "IoT-enabled real-time traffic monitoring and control management for intelligent transportation systems," *IEEE Internet Things J.*, vol. 11, no. 9, pp. 15842–15854, May 2024.

[4] J. Wang, W. Sun, L. Huang, and J. Zhang, "Accurate and computationally efficient interpolation-based method for two-dimensional harmonic retrieval," *Digit. Signal Process.*, vol. 78, pp. 108–120, 2018.

[5] Y. Mao, J. Zhang, H. Qi, and L. Wang, "DNN-MVL: DNN-multi-view-learning-based recover block missing data in a dam safety monitoring system," *Sensors*, vol. 19, no. 13, 2019, Art. no. 2895.

[6] M. B. Karo, B. P. Miller, and O. A. Al-Kamari, "Leveraging data utilization and predictive analytics: Driving innovation and enhancing decision making through ethical governance," *Int. Trans. Educ. Technol.*, vol. 2, no. 2, pp. 152–162, 2024.

[7] M. Peng, L. Zhang, X. Sun, Y. Cen, and X. Zhao, "A synchronous long time-series completion method using 3-D fully convolutional neural networks," *IEEE Geosci. Remote Sens. Lett.*, vol. 19, 2022, Art. no. 8007605.

[8] C. Liu, K. Xie, T. Wu, C. Ma, and T. Ma, "Distributed neural tensor completion for network monitoring data recovery," *Inf. Sci.*, vol. 662, 2024, Art. no. 120259. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025524001725

[9] E. Wang, M. Zhang, B. Yang, Y. Xu, Z. Song, and Y. Yang, "Few-shot data completion for new tasks in sparse crowdsensing," in *Proc. IEEE Conf. Comput. Commun.*, 2024, pp. 1831–1840.

[10] D. B. Rubin, "Inference and missing data," *Biometrika*, vol. 63, no. 3, pp. 581–592, 1976, doi: 10.1093/biomet/63.3.581.

[11] J. Langhammer and J. Česák, "Applicability of a nu-support vector regression model for the completion of missing data in hydrological time series," *Water*, vol. 8, no. 12, pp. 1–25, 2016. [Online]. Available: https://www.mdpi.com/2073-4441/8/12/560

[12] H. Khan, X. Wang, and H. Liu, "Handling missing data through deep convolutional neural network," *Inf. Sci.*, vol. 595, pp. 278–293, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025522001931

[13] L. Han, K. Zheng, L. Zhao, X. Wang, and H. Wen, "Content-aware traffic data completion in ITS based on generative adversarial nets," *IEEE Trans. Veh. Technol.*, vol. 69, no. 10, pp. 11950–11962, 2020.

[14] R. L. Lopes and A. M. Jorge, "Assessment of predictive learning methods for the completion of gaps in well log data," *J. Petroleum Sci. Eng.*, vol. 162, pp. 873–886, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0920410517309014

[15] S. Chen and Y. Liu, "Distributed personalized imputation based on gaussian mixture model for missing data," *Neural Comput. Appl.*, vol. 36, no. 23, pp. 14237–14250, 2024.

[16] N. Srebro, J. Rennie, and T. Jaakkola, "Maximum-margin matrix factorization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2004, pp. 1–8. [Online]. Available: https://proceedings.neurips.cc/paperfiles/paper/2004/file/e0688d13958a19e087e123148555e4b4-Paper.pdf

[17] A. P. Singh and G. J. Gordon, "Relational learning via collective matrix factorization," in *Proc. Int. Conf. Knowl. Discov. Data Mining*, 2008, pp. 650–658, doi: 10.1145/1401890.1401969.

[18] R. Salakhutdinov and A. Mnih, "Probabilistic matrix factorization," in *Proc. 21st Int. Conf. Neural Inf. Process. Syst.*, NY, USA, Curran Associates Inc., 2007, pp. 1257–1264.

[19] X. Shen, X. Zhang, L. Lan, Q. Liao, and Z. Luo, "Another robust NMF: Rethinking the hyperbolic tangent function and locality constraint," *IEEE Access*, vol. 7, pp. 31089–31102, 2019.

[20] J. Fan and J. Cheng, "Matrix completion by deep matrix factorization," *Neural Netw.*, vol. 98, pp. 34–41, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0893608017302502

[21] S.-R. Oh, Y.-D. Seo, E. Lee, and Y.-G. Kim, "A comprehensive survey on security and privacy for electronic health data," *Int. J. Environ. Res. Public Health*, vol. 18, no. 18, pp. 1–47, 2021. [Online]. Available: https://www.mdpi.com/1660-4601/18/18/9668

[22] Z. Wei, Z. Li, X. Mao, and J. Wang, "Applying differential privacy to tensor completion," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2022, pp. 3923–3927.

[23] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Artif. Intell. Statist.*, 2017, pp. 1273–1282.

[24] X.-Y. Liu, Z. S. Li, and X. Wang, "Homomorphic matrix completion," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, pp. 12197–12210. [Online]. Available: [https://proceedings.neurips.cc/paperfiles/paper/2022/file/4f550cb7b30b59553e50cd08a9dbf068-Paper-Conference.pdf

[25] O. Goldreich, "Secure multi-party computation," *Manuscript. Preliminary Version*, vol. 78, no. 110, pp. 1–108, 1998.

[26] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, "A survey on federated learning," *Knowl.-Based Syst.*, vol. 216, 2021, Art. no. 106775.

[27] Z. Liu, J. Guo, W. Yang, J. Fan, K.-Y. Lam, and J. Zhao, "Privacy-preserving aggregation in federated learning: A survey," *IEEE Trans. Big Data*, early access, Jul. 15, 2022, doi: 10.1109/TBDATA.2022.3190835.

[28] B. Zhao, X. Liu, W.-N. Chen, and R. H. Deng, "CrowdFL: Privacy-preserving mobile crowdsensing system via federated learning," *IEEE Trans. Mobile Comput.*, vol. 22, no. 8, pp. 4607–4619, Aug. 2023.

[29] D. De, "FedLens: Federated learning-based privacy-preserving mobile crowdsensing for virtual tourism," *Innovations Syst. Softw. Eng.*, vol. 20, no. 2, pp. 137–150, 2024.

[30] A. Saiapin et al., "Federated privacy-preserving collaborative filtering for on-device next app prediction," *User Model. User-Adapted Interaction*, vol. 34, no. 4, pp. 1369–1398, 2024.

[31] X. Wan, Y. Zheng, Q. Li, A. Fu, M. Su, and Y. Gao, "Towards privacy-preserving and verifiable federated matrix factorization," *Knowl.-Based Syst.*, vol. 250, 2022, Art. no. 109193.

[32] Z. Li, B. Ding, C. Zhang, N. Li, and J. Zhou, "Federated matrix factorization with privacy guarantee," *Proc. VLDB Endowment*, vol. 15, no. 4, pp. 900–913, 2021. [Online]. Available: https://par.nsf.gov/biblio/10322933

[33] D. Chai, L. Wang, K. Chen, and Q. Yang, "Secure federated matrix factorization," *IEEE Intell. Syst.*, vol. 36, no. 5, pp. 11–20, Sep/Oct. 2021.

[34] A. Berlioz, A. Friedman, M. A. Kaafar, R. Boreli, and S. Berkovsky, "Applying differential privacy to matrix factorization," in *Proc. 9th ACM Conf. Recommender Syst.*, 2015, pp. 107–114, doi: 10.1145/2792838.2800173.

[35] P. Paatero and U. Tapper, "Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values," *Environmetrics*, vol. 5, no. 2, pp. 111–126, 1994. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/env.3170050203

[36] N. Wang et al., "Collecting and analyzing multidimensional data with local differential privacy," in *Proc. IEEE Int. Conf. Data Eng.*, 2019, pp. 638–649.

[37] N. Phan, X. Wu, H. Hu, and D. Dou, "Adaptive laplace mechanism: Differential privacy preservation in deep learning," in *Proc. IEEE Int. Conf. Data Mining*, 2017, pp. 385–394.

[38] Y. Zheng, F. Liu, and H.-P. Hsieh, "U-air: When urban air quality inference meets Big Data," in *Proc. 19th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2013, pp. 1436–1444.

[39] K. D. B. J. Adam et al., "A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[40] E. Wang et al., "Deep learning-enabled sparse industrial crowdsensing and prediction," *IEEE Trans. Ind. Informat.*, vol. 17, no. 9, pp. 6170–6181, Sep. 2021.

[41] W. Yuan, C. Dai, H. Ding, and Q. Tang, "A patch-based multiple imputation transformer model for time series data completion," in *Proc. 10th Int. Conf. Big Data Inf. Analytics*, 2024, pp. 120–124.

[42] D. Xu, H. Peng, Y. Tang, and H. Guo, "Hierarchical spatio-temporal graph convolutional neural networks for traffic data imputation," *Inf. Fusion*, vol. 106, 2024, Art. no. 102292.

En Wang (Member, IEEE) received the BE degree in software engineering from Jilin University, Changchun, in 2011, and the ME and PhD degrees in computer science and technology from Jilin University, Changchun, in 2013 and 2016, respectively. He was also a joint PhD student with the Department of Computer and Information Science, Temple University. He is currently a professor with the Department of Computer Science and Technology, Jilin University, Changchun. His current research focuses on mobile computing, crowd intelligence, and data mining.

Baoju Li (Graduate Student Member, IEEE) received the BE degree in measurement and control technology and instrument from the Harbin Institute of Technology, Harbin, in 2013. He is currently working toward the PhD degree with Jilin University. He enrolled in a combined master's and PhD degrees program in computer science and technology with Jilin University, Changchun, China, in 2019. His current research focuses on mobile crowdsensing, online incentive mechanism algorithms, and federated learning.

Wenbin Liu (Member, IEEE) received the BS degree in physics and the PhD degree in computer science and technology from Jilin University, China, in 2012 and 2020, respectively. He is currently an associate professor with the College of Computer Science and Technology, Jilin University. He is currently also a postdoctoral researcher with China Telecom. His research interests include mobile crowdsensing, spatio-temporal crowdsourcing, and ubiquitous computing.

Zengyi Han (Member, IEEE) received the PhD degree in information science and technology from the University of Tokyo, Japan, in 2023. He is currently a lecturer with the College of Artificial Intelligence, Dalian Maritime University, Dalian, China. He has authored papers in *IEEE Transactions on Mobile Computing*, IEEE PerCom, IEEE ICDCS, etc. His main research interests include pervasive computing and mobile computing.

Cong Wang received the BE degree in software engineering from Jilin University, Changchun, China, in 2022. Currently, he is working toward the ME degree in software engineering with Jilin University. His current research focuses on mobile crowdsensing, federated learning.

Ximing Li received the PhD degree from Jilin University, China. He is currently a full-time professor with Jilin University, China. His research interests include machine learning, weakly-supervised learning, and natural language processing applications. He has published more than 100 papers at the competitive venues, including ICML, NeurIPS, ICLR, ACL, SIGIR, WWW, CVPR, IJCAI, AAAI, etc.

Jie Wu (Fellow, IEEE) is the director with the Center for Networked Computing and Laura H.Carnell professor with Temple University. He also serves as the director of international affairs with the 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 with the National Science Foundation and was a distinguished professor with 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. He was an IEEE Computer Society distinguished visitor and ACM distinguished speaker. He is a China Computer Federation (CCF) distinguished speaker. Currently, he is on leaving working as a scientist with China Telecom.