A Mobility Forecasting Framework with Vertical Federated Learning

Fatima Zahra Errounda
Department of Electrical and
Computer Engineering
Concordia University
Montreal, Canada
Email: f_errou@encs.concordia.ca

Yan Liu
Department of Electrical and
Computer Engineering
Concordia University
Montreal, Canada
Email: yan.liu@concordia.ca

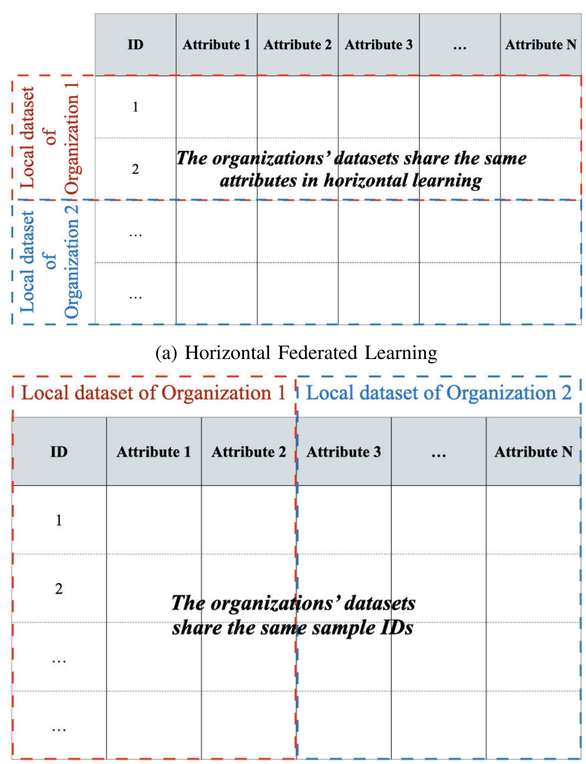
***Abstract*—With the prevalence of mobile devices and location-based services, forecasting human mobility has become a critical topic in ubiquitous computing. Existing forecasting approaches usually adopt frameworks with a centralized mobility data holder. However, mobility data typically pertains to independent organizations, introducing two learning challenges. First, since each organization only holds a location domain subset, none can tackle a forecasting model that covers the whole location domain. Second, distributed mobility data compromises the spatio-temporal correlation between locations hindering learning. Hence, reducing the forecasting accuracy. This work proposes a mobility vertical federated forecasting (MVFF) framework that allows the learning process to be jointly conducted over vertically partitioned data belonging to multiple organizations. MVFF enables the forecasting of mobility predictions covering a joint location domain. We evaluate MVFF's performance over two real-world datasets using different spatial and temporal neural network algorithms. Experimental results demonstrate that the two datasets' mean percentage error performance gains are up to 12% and 4% compared to the state-of-the-art, respectively.**

I. Introduction

With the prevalence of mobile devices and location-based services, forecasting human mobility has become an essential topic in ubiquitous computing. For example, mobility data may help alleviate traffic congestion [1], control traffic lights [2], and improve the efficiency of traffic operations [3]. Moreover, data on travel history can help evaluate the effectiveness of non-pharmaceutical interventions and forecast the spread of COVID-19 [4].

Currently, most mobility forecasting approaches ([5], [6], [7]) adopt a centralized learning strategy by transferring the entire mobility data to a data center to learn the prediction model. Alternatively, distributed learning methods ([8], [9]) focus on horizontal federated learning, where each organization has a subset of samples covering the entire mobility location domain. However, mobility data typically pertains to independent organizations with different locations regions. Therefore, the learning methods need to cater to vertically distributed data. Figure 1 demonstrates the difference in learning strategies depending on the data distribution method [10].

The first challenge we identify in mobility vertical federated forecasting is that since each organization only holds a location domain subset, none can tackle a forecasting model encompassing the whole location domain. Therefore, this raises the following research question:



(a) Horizontal Federated Learning

(b) Vertical Federated Learning

Fig. 1: Federated learning categorization: (a) horizontal data distribution results in local datasets that share the same attributes (b) vertical data distribution leads to datasets with distinct attributes but common sample IDs

> RQ1: How to achieve forecasting on the entire location domain when the individual locations belong in separate organizations' datasets?

978-1-6654-8810-5/22/$31.00 ©2022 IEEE
DOI 10.1109/COMPSAC54236.2022.00050

Vertically distributed mobility data compromises the spatio-temporal correlation between locations hindering learning. Hence, compromising the forecasting accuracy. Therefore, we define the second research question as follows:

> RQ2: How to reconstruct the spatio-temporal correlation between distributed locations to provide mobility count forecasting?

This work proposes a mobility vertical federated forecasting (MVFF) framework that allows the learning process to be jointly conducted over vertically partitioned data belonging to multiple organizations. MVFF enables the forecasting of mobility predictions covering a joint location domain. We summarize our **contributions** as follows:

1) Mobility Vertical Federated Forecasting (MVFF) framework is a cross-organization collaborative learning framework composed of three components: local models, a global model, and a learning collaborator. The local models independently embed the spatio-temporal correlation in each organization dataset. The global model combines the learning from the local models to incorporate the correlation between all locations in the location domain. The learning collaborator synchronizes the learning between the global and local models.

2) Temporal synchronization module: resides in the learning collaborator. Leverages a buffer that stores the local model parameters sent from the organizations with their corresponding timestamps. The timestamps correspond to the training period time performed by the organization.

3) Spatial synchronization module: resides in the learning collaborator. Leverages a map building that organizes the local model parameters to cover the location domain, then triggers the spatial correlation learning via the global model.

4) We evaluate MVFF's performance over two real-world datasets with four variables: two local (LSTM, GRU) and two global (CNN, GNN) models. We also perform a comparison with other state-of-the-art models. Experimental results demonstrate that the two datasets' mean percentage error performance gains are up to 12% and 4% compared to the state-of-the-art, respectively.

The rest of this paper is structured as follows. Section II introduces the fundamentals of federated learning and mobility forecasting models, while Section III gives the problem definition. Section IV describes the details of the framework, and Section V explains the experimental evaluation results. Section VI reviews the related work. Lastly, we conclude this paper in Section VII.

II. Preliminaries

In this section, we first introduce the definition of federated learning. Then, we briefly review the commonly used machine learning algorithms in mobility forecasting: Long Short Term Memory [11], Gated Recurrent Unit [12], Convolutional Neural Network [13], and Graph Neural Network [14].

A. Federated Learning

Machine learning aims to produce a prediction in response to a query [15]. To achieve this goal, machine learning [15] typically uses a set of training data that takes the form of a collection of features and labels. The inputs in the training data may be classical vectors or more complex objects, such as documents [16], images [13], DNA sequences [17], or graphs [14]. Federated learning [18] is a distributed machine learning paradigm that trains a high-quality centralized model while the training data remains distributed over many clients.

A client in federated learning computes an update of its model based on its local data and sends the updates to the server [18]. The server then collects the clients' model updates to compute a new global model [18]. Based on the distribution of features and labels, federated learning can be classified into horizontal and vertically federated learning [10].

In horizontal federated learning, clients share identical features and label space but differ in the samples' space [10]. For instance, a mobile service jointly trains a next-word prediction model using mobile phone users' data [19]. In this case, the mobile users share the same textual features; however, their locally stored text is different.

Conversely, clients in vertical federated learning typically only have a subset of the feature space that may be overlapping [10]. For example, consider a bank and e-commerce that aim to build a prediction model for product purchases based on user and buying history [20]. However, the bank holds the user's earnings and spending behaviour, while the e-commerce retains the user's purchasing history. Consequently, their feature spaces are very different despite the shared prediction model.

B. Mobility Forecasting

1) Long Short Term Memory: Long Short Term Memory (LSTM) is a class of recurrent neural networks, initially introduced by Hochreiter and Schmidhuber [11] in 1979, to correlate values from earlier stages for future use. This learning model propagates or forgets information over a long and recurrent training period to improve the prediction performance. LSTM is widely used in mobility forecasting, such as crowd prediction [5], [6], and traffic flow analysis [21].

2) Gated Recurrent Unit: Gated Recurrent Unit (GRU) is another variant of recurrent neural networks that handle time-series data. It was proposed by Cho et al. [22] in 2014. Similar to LSTM, GRU cells have a mechanism to judge whether a piece of information is valuable or not, however, without having a separate mechanism to memorize or forget it. GRU is widely used in understanding human mobility, such as next location prediction [23], crowd prediction [24], and traffic anomaly detection [25].

3) Convolutional Neural Networks: Convolutional Neural Network (CNN) [13] is a class of neural networks that mimics the connectivity pattern between neurons in the human visual cortex. CNNs extract localized spatial features and compose them to create highly expressive representations. Therefore, they are widely applied in the field of computer vision and image recognition fields ([26], [27]). CNNs are

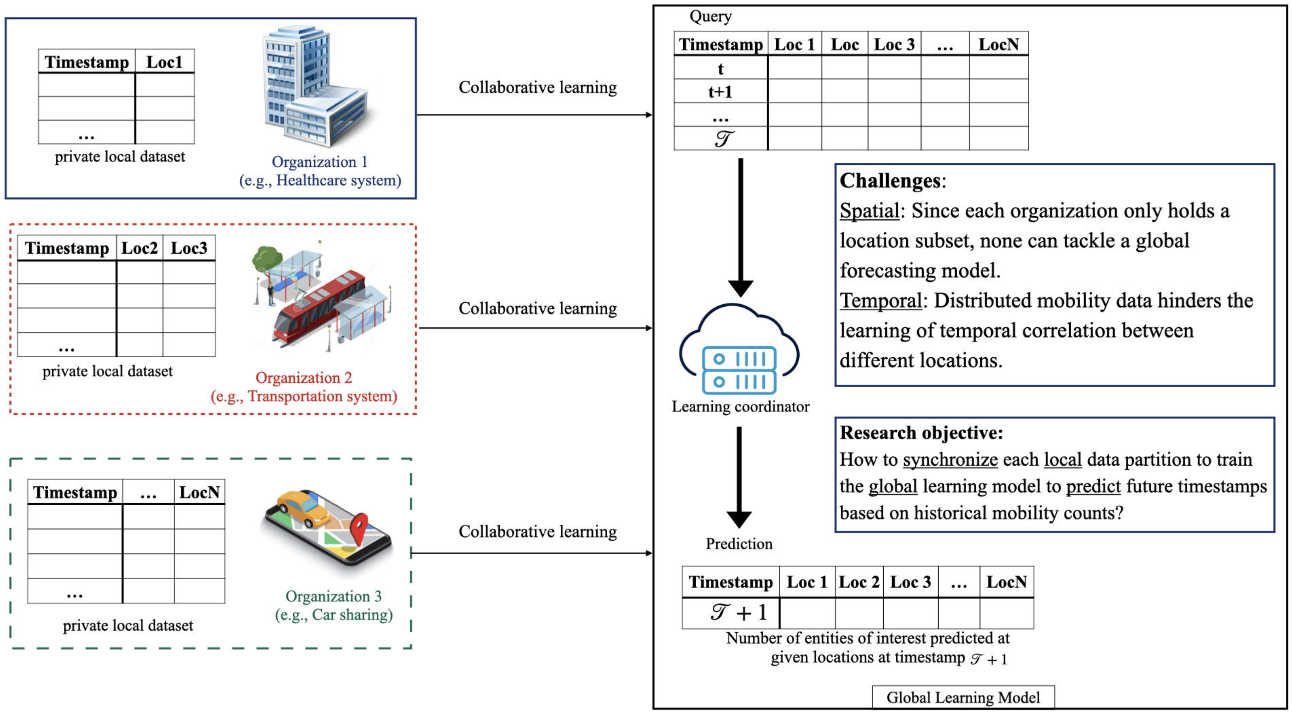


Fig. 2: System model: our main focus is to design a learning framework that coordinates the learning between multiple organizations to make mobility forecasting.

generally adopted in the mobility domain by converting human movement flow to images describing time and space relations. CNNs are used in a variety of mobility domains, such as indoor localization [28], intelligent transportation systems [29], and location-based marketing [30].

4) Graph Neural Network: Graph Neural Network (GNN) [14] is a category of neural networks designed to process data represented by a graph data structure. Unlike CNN, GNN can operate on graphs of arbitrary size and the complex topology. GNNs are used in several mobility domains, such as in capturing location transition patterns in location trajectory [31], modelling the spatiotemporal nature of the spread of infectious diseases [32], and predicting traffic flow [33].

III. Problem Statement

A. System Model

We consider a set of m organizations (or data owners) $\mathcal{O} = \{O_1, ..., O_m\}$ that jointly train a learning model without sharing their respective datasets, as illustrated in Fig.2. The learning coordinator, $\mathcal{C}$, is responsible for orchestrating the global learning objective across the organizations.

We denote $D_i = \{x_t^{D_i}\}_{t=1}^{d}$ where $x_t^{D_i}$ represents the t-th row of the organization O_i, and each column corresponds to a location. Each row, $x_{t,L_j}^{D_i}$, corresponds to the count of entities of interest at location L_j. Finally, n is the combined number of locations from all organizations.

After obtaining the trained learning model, $\mathcal{C}$ makes predictions $X_{\mathcal{T}+1} = \{x_{\mathcal{T}+1}^{L_1}, x_{\mathcal{T}+1}^{L_2}, ..., x_{\mathcal{T}+1}^{L_n}\}$ denoting the mobility counts at locations $\{L_1, ..., L_n\}$, at timestamp $\mathcal{T}+1$. Table I summarizes the frequently used notations.

Notation	Description
O_i	Organization i
D_i	Organization i's dataset
m	Number of organizations
$\mathcal{C}$	Learning coordinator
$x_t^{D_i}$	the t-th row of the organization O_i
n	Number of locations
$\mathcal{T}$	Mobility count observations length
$X_{\mathcal{T}+1}$	Predictions of mobility counts at timestamp $\mathcal{T}+1$
$\mathcal{I}$	Training interval length
M_i	Local learning model of organization O_i
G	Global learning model
ϕ	Location domain map

TABLE I: Used notations

B. Problem Formulation

To predict the mobility counts at timestamp $\mathcal{T}+1$, the learning coordinator $\mathcal{C}$ must synchronize the global learning between the organizations. In particular, the learning coordinator must maintain a correspondence between the mobility data in the consecutive timestamps at each organization. Moreover, the learning coordinator must model the correlation between all the locations in the location domain. Thus, we define the

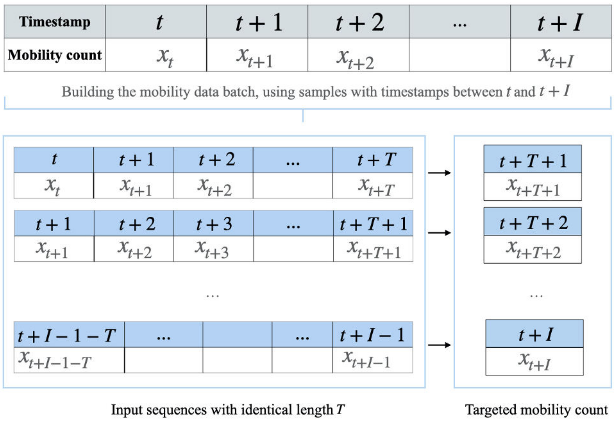


Fig. 3: Building the training batch: a sequence of $\mathcal{I}$ mobility counts with consecutive timestamps is restructured into sequences with the historical data length $\mathcal{T}$. The targeted mobility count represents the ground truth prediction.

problem as spatio-temporal coordination of the learning from the organizations $\mathcal{O}$ to predict the mobility count predictions $X_{\mathcal{T}+1}$ given historical mobility count observations of $\mathcal{T}$ timestamps $\{X_1, X_2, ..., X_{\mathcal{T}}\}$.

IV. Mobility Vertical Federated Forecasting

This section provides the details of our proposed MVFF framework. First, we present the overall framework. Then, we describe the temporal synchronization and the temporal synchronization modules.

A. Overall Learning Framework

We propose to perform vertical federated learning where the collaborator combines the parameters received from the organizations' models to train a global model that can forecast the location domain's mobility counts. The framework comprises three components: the local models, the global model, and the learning collaborator. Figure 4 illustrates the components and their interactions during the training process.

We consider that each organization, O_i, has a local model M_i that uses the organization's data for training. The m organizations support the learning collaborator to train a global model encompassing the location domain, ϕ. We assume that the local and global models do not share the same neural network architecture. This difference is a consequence of the vertical partitioning of the mobility dataset. Consequently, the learning objective of the local model is to conceptualize the spatio-temporal correlations between locations under the organization's jurisdiction only. In contrast, the global model's neural architecture covers the entire location domain. The temporal and spatial synchronization modules bridge the connection between local and global models.

Algorithm 1 gives the high-level steps of MVFF. The learning collaborator and organizations initialize all their models' parameters before the learning starts. We assume that the organizations communicate their local model's architecture and location topology to the learning collaborator at the initialization of the training. Each organization, O_i, selects a timestamp t and builds a mobility data batch, B_i, using samples with timestamps between t and $t+\mathcal{I}$ (line 3). These samples consist of sequences of consecutive mobility data with the length of the historical data $\mathcal{T}$ and the corresponding target counts, as illustrated in Fig.3. O_i then performs a training iteration of the local model with the built batch (line 4). Next, O_i sends the resulting model parameters with the corresponding timestamp, t, to the learning collaborator (line 5). Upon receiving the learning parameters, the learning collaborator uses them to build a training batch for the global model (lines 6-11). Finally, the learning collaborator computes the global model parameters and updates them (lines 12-13). This collaboration continues until the predefined number of iterations T is satisfied.

It is essential to notice that the learning collaborator $\mathcal{C}$ communicates the interval length $\mathcal{I}$ to the organizations at the initialization of the learning process. Moreover, when the collaborator finishes the global model training, it broadcasts the *Halt* command to all organizations. Meanwhile, the organizations continuously send their local model updates to the learning collaborator until they receive a *Halt* signal. Finally, the organizations can build their training batches with different starting timestamps. Hence, the learning collaborator is responsible for synchronizing the multiple parameters. We next describe the detailed procedures of the temporal and spatial synchronization modules, respectively.

B. Temporal Synchronization Module

The temporal discrepancy between mobility data is a direct consequence of the vertical partitioning of mobility data. Hence, the goal of the temporal synchronization module is to correlate between the organizations' learned parameters at the same timestamp. To achieve this, we utilize ordered buffers at the temporal synchronization module.

The temporal synchronization module creates one parameter buffer per organization. The buffers are initialized at the beginning of the training procedure. When the temporal synchronization module receives a model parameter and timestamp from an organization, it stores them in the dedicated buffer in ascending order of the timestamp value. When all the buffers have parameters corresponding to any given timestamp t, the temporal synchronization module sends them to the spatial synchronization module.

C. Spatial Synchronization Module

Another consequence of the vertical partitioning of the mobility data is the loss of correlation between locations. For example, two spatially adjacent locations may belong to two different organizations. The mobility counts in the two locations may be mutually interdependent, such as in the case of crowd counts. However, it is challenging to assess this interdependence when the mobility counts of the two locations pertain to separate datasets. The goal of the spatial

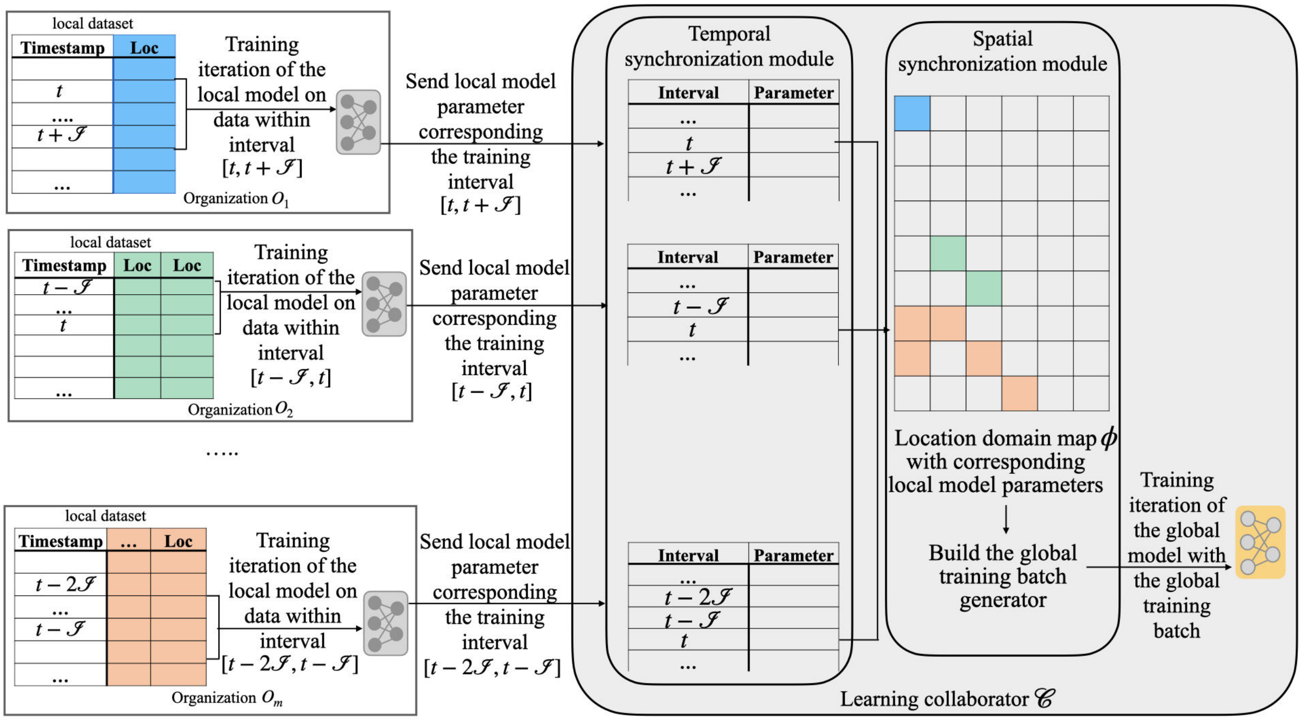


Fig. 4: Illustration of the training procedure in the MVFF framework. The global model trains mobility data generated using the temporally synchronized local models' training parameters and the location domain map. The different colours on the location map indicate different location features (mobility count) coming from the corresponding organizations.

synchronization module is to reconstruct the correlation between locations through the organizations' learned parameters.

The spatial synchronization module utilizes the location domain map ϕ and the local model parameters to generate a training batch for the global model. We assume that the synchronization module has a training mobility count dataset. Depending on the application domain, this dataset can be minimal and easy to obtain in practice, such as the publicly available datasets (Geolife [34], T-Drive [35], or Gowalla [36]).

Similar to the training batches built for training the organization's local model, the global training batch builds sequences with the historical data length $\mathcal{T}$ using the global training mobility count dataset. However, the targetted mobility counts corresponding to each sequence are generated using the local models' parameters. Since the spatial synchronization module has copies of the organization's models' neural architectures, it uses the model parameters received from the temporal synchronization module to compute the targeted mobility counts. As illustrated in Fig.4, the synchronization module coordinates each local model on the location domain map to the corresponding location. Then, it uses the local model to predict the targeted mobility counts of each sequence in the global training batch. Next, the spatial synchronization module uses the predictions as ground truth to train the global model. Finally, the spatial synchronization module triggers a global model training iteration over the global training batch.

V. Experimental Evaluation

In this section, we conduct extensive experimental analyses to validate the efficiency and utility of MVFF.

A. Datasets and Metrics

We evaluate our proposed model using two large-scale real-world datasets from New York City [37] and Yelp [38]. Each dataset contains mobility data as follows:

New York City Bike dataset [37]: It contains monthly trip details of bikes based on the bike stations deployed in New York City as described below:

{*TripId, Trip Duration, Start Time, Stop Time, Start Station ID, Start Station Name, Start Station Latitude, Start Station Longitude, End Station ID, End Station Name, End Station Latitude, End Station Longitude*}

We divide the city into 16x8 regions, each containing bike stations based on their latitude and longitude. We use the start and end stations and the start and stop times to generate a sequence of bike counts at each region, with a time granularity of 15 minutes. We collect the first four months of 2021: February 1st to Mai 31st, 2021. Therefore, the total number of timestamps in the dataset is 1.5 million.

Algorithm 1: Mobility vertical federated learning

Input: Mobility count observations length $\mathcal{T}$, temporal buffer size $\mathcal{I}$, location domain map ϕ, number of iterations T

Initialize: M_i parameters for every organization, global model parameters

//**Learning process at i-th organization**

while *not Halt* **do**
    for *timestamp t in D_i* **do**
        Build a data batch B_t with mobility counts in the interval $[t, \mathcal{I}]$
        Compute gradient $\Delta\theta_i$ of local model M_i on batch B_t
        Send (t,$\Delta\theta_i$)) to learning collaborator $\mathcal{C}$
    end
end

//**Learning process at collaborator $\mathcal{C}$**

for *each iteration j from $1, 2, ..., T$* **do**
    When $\mathcal{C}$ receives $\Delta\theta_i$ from O_i, the temporal synchronization module stores it to a buffer dedicated to the local model M_i
    The temporal synchronization module sorts the buffers and samples a timestamp t
    for *each location L_i in Φ* **do**
        The spatial synchronization module builds a mobility count batch B based on the corresponding local model's parameter $\Delta\theta_t$
    end
    The spatial synchronization module organizes the batch B following the location domain map structure ϕ
    Compute gradient $\Delta\theta_c$ of global model G on batch B
    Update global model G's parameter θ_c
end
Send *Halt* command to every organization

Yelp review dataset [38]: It contains 192k businesses from metropolitan areas in the United States with their location (latitude and longitude), and 6.68M timestamped users' reviews as below:

{*review id, user id, business id, review, date*}

We split the country into a uniform grid of 8×8 regions, each containing businesses based on their latitude and longitude. We use the reviews timestamps to generate a sequence of reviews counts at each region, with a time granularity of 6 hours. The review dates start on January 1st, 2018 and stop on December 31st, 2020. Overall, the sequence dataset contains 280k timestamps.

We select two metrics for utility assessment, root mean square error (RMSE) and weighted mean absolute percentage error (WMAPE). We compute the two metrics as follows: let x_t be actual mobility count at any time instance t and x'_t be the corresponding predicted count, where t varies from 1 to n, consequently:

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(x_t - x'_t)^2} \tag{1}$$

$$WMAPE = \frac{\sum_{t=1}^{n}|x_t - x'_t|}{\sum_{t=1}^{n}|x_t|} \tag{2}$$

WMAPE is less sensitive to outliers than RMSE since WMAPE focuses on absolute errors ($|x_t - x'_t|$). However, both metrics do not include the sign of the prediction error. Therefore, adding the average error (AE) metric is necessary to see if the forecasting techniques underestimate or overestimate the mobility count. The average error (AE) will be reported for each case as follows:

$$AE = \frac{1}{n}\sum_{t=1}^{n}(x_t - x'_t) \tag{3}$$

Dataset	Min	Max	Mean	Median	Standard deviation
New York City Bike	0	50.70	3.96	6.24	6.94
Yelp review	0	6.73	0.41	0.72	0.89

TABLE II: Basic analysis of the used datasets

B. Framework Implementation with Model Variations

We study the accuracy of our vertical federated learning framework using different learning models. Since the framework consists of local and global models, we implement them as different neural networks that are commonly utilized for mobility forecasting to form the following variations:

GRU+CNN: The local models in this variation implement the GRU architecture with one unit, followed by a linear layer that maps the features to predictions. We interpret the location grids as images where longitude and latitude are equivalent to width and height, and timestamps perform as channels. Therefore, the global model has CNN architecture that comprises a three-dimensional convolutional layer to reveal the activity patterns between timestamps. Stride and padding for convolutional layers were selected to preserve the original size of the grid.

LSTM+CNN: The local models in this variation implement the LSTM architecture with one LSTM cell followed by a linear layer that maps the features to predictions. The global model follows the same architecture as the CNN in the GRU+CNN variation.

GRU+GNN: In this variation, the local model follows the same architecture as the GRU in the GRU+CNN variation. The global model implements the Graph Attention Networks architecture [39], utilizing the attention mechanism to obtain the correlations between locations in the grid. We implement an attention layer for each cell that factors the spatial and temporal mobility data from the cell's neighbours on the grid. The adjacency matrix encodes the neighbourhood relationship between the cells in the location grid.

LSTM+GNN: In this variation, the local model follows the same architecture as the LSTM+CNN variation, while the global model follows the same architecture as the GRU+CNN variation.

C. Comparison Frameworks

We compare our model with the following federated frameworks for time-series forecasting to investigate utility improvement:

FedGRU [40]: This framework uses the FederatedAveraging [41] algorithm to build a global GRU model based on model parameters from different organizations. Each organization uses its local data to perform a gradient descent optimization on its local model. Then the central cloud performs a weighted average aggregation of the local model updates uploaded by the clients. Consequently, the framework uses the same network architecture in the global and local models.

Federated LSTM [42]: This framework uses LSTM on edge devices that capture houses' electrical loads to predict future ones. The studied group of houses has similar geographical and infrastructural properties. The edges train the local LSTM models using a dataset composed of sliding windows with predetermined look-back steps. The initial global LSTM model is pre-trained using publicly available data or initialized randomly. Then, the cloud server aggregates the local model updates to compute the global model using the FederatedAveraging algorithm [41].

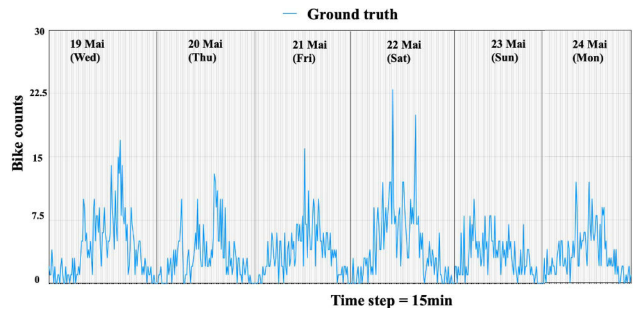
D. Experiment design and results

We set the MVFF parameters described in Algorithm 1 as follows. We fix the mobility count observations length $\mathcal{T}$ to 6 and forecast 1 step ahead. The temporal buffer size $\mathcal{I}$ for experiments 1 and 2 is set as 24hours, and we vary this parameter in experiment 3. The location domain map ϕ for each dataset corresponds to a grid of 16x8 and 8x8 for New York City Bike and Yelp review, respectively. We consider the highest possible vertical distribution. Hence, the number of organizations corresponds to the number of cells in the grid of each dataset.

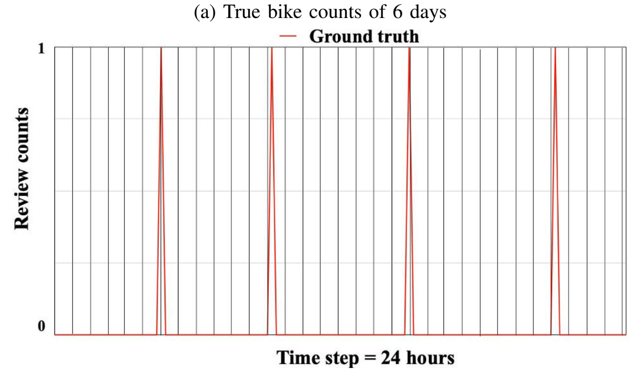
The training dataset for each organization's local model comprises the mobility counts of the corresponding grid cell. We build the training data for the global model by randomly sampling 10% of the local datasets and testing data with 10% of each organization's local dataset.

We implement MVFF and the comparison frameworks using Pytorch [43] and Colab platform. We use Adam [44] with a learning rate of 0.001 for the local and global models. Our code is available at https://github.com/FatimaErrounda/MVFF.

Experiment 1 (RQ1): Assesses the utility of the variations with the comparison frameworks.



(a) True bike counts of 6 days



(b) True review counts of 40 days: Nov 1st, 2020 to October 11th, 2020

Fig. 5: Mobility counts (ground truth) for Bike New York and Yelp review datasets.

Configuration: Report the utility metrics (Equations 1, 2, and 3) based on the variations' global models' forecasting over the testing data. Capture the utility metrics for the comparison frameworks using the same testing data. For simplicity, we average the metric values over all the cells in the dataset grid and round the values to two decimal places.

Results: Table III and Table IV show the results for the New York City Bike and Yelp review datasets, respectively. The four variations deliver smaller metric values than the comparison frameworks. This improvement is due to the inferred location correlation achieved by the spatial synchronization module. Moreover, we notice that RMSE and AE values for Yelp are smaller than the New York City Bike. This difference can be attributed to the variability of the mobility counts per dataset. We calculate the mean and standard deviations for the New York City Bike and Yelp review datasets. Table II supports our analysis; as illustrated, the variability of the New York City Bike dataset is higher than Yelp. Moreover, Fig.5 shows a portion of the two test datasets to better illustrate the variability in the mobility counts. GNN-based variations give the smallest metric values.

Experiment 2 (RQ1): Assesses the framework's forecasting compared to the ground truth mobility data.

Configuration: Since the location domain includes multiple locations, we select one cell per dataset to display the ground truth and the predictions made by the four variations and the comparison frameworks.

Model	RMSE	WMAPE	AE
LSTM+CNN	12.23	1.07	7.81
GRU+CNN	15.62	0.99	7.86
LSTM+GNN	6.88	0.58	2.08
GRU+GNN	6.79	0.58	1.93
FedGRU [40]	17.14	1.00	8.90
Federated LSTM [42]	17.24	1.18	8.95

TABLE III: The metrics results on Bike New York dataset

	RMSE	WMAPE	AE
LSTM+CNN	1.10	1.14	0.63
GRU+CNN	1.07	1.17	0.76
LSTM+GNN	0.95	1.11	0.18
GRU+GNN	0.96	1.12	0.19
FedGRU [40]	1.22	1.51	0.66
Federated LSTM [42]	1.24	1.14	0.68

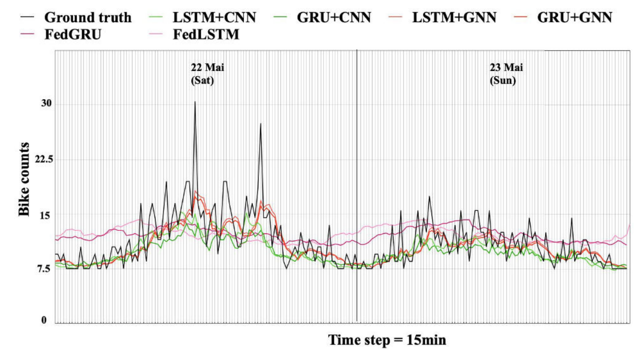
TABLE IV: The metrics results on Yelp dataset

Results: As shown in Fig.6, the four variations forecast closely follow the trend of the oscillations of the ground truth, with the GNN-based variations as the nearest. The predicted counts produced by the comparison frameworks noticeably exceed the ground truth at some timestamps. None of the models could predict the sharp peaks in the New York Bike dataset. However, the GNN-based variations can predict the peaks with fewer counts than occurred. This improvement is due to better capturing of the spatial features of mobility counts of the GNN model.

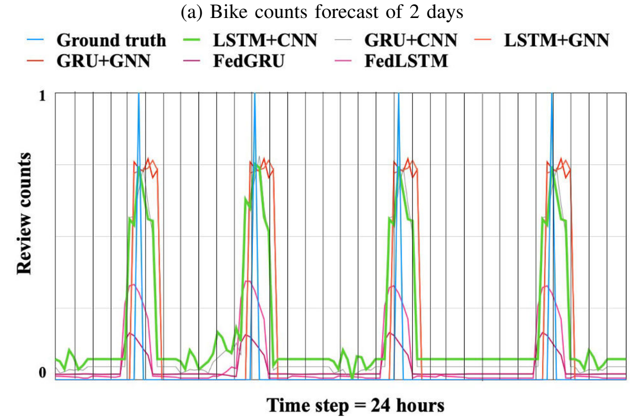
Experiment 3 (RQ2): Assesses the impact of the interval length parameter in reconstructing the spatio-temporal correlation between locations.
Configuration: Vary the temporal buffer size $\mathcal{I}$ and capture the utility metrics for the 4 variations.
Results: Figure 7 demonstrates that the framework's performance is orthogonal to the temporal buffer size. This result is the consequence of the sequential aspect of creating the local training batch for each organization. As explained in Section IV, training the local models on a given interval leads to the generations of sequences with the length of the historical mobility data. Therefore, a more extended temporal buffer size means that generated sequences number increase. Nevertheless, the correlation between locations is encoded in the sequences' counts, not their numbers. However, the temporal buffer size change might affect the communication rounds between the local models and the learning collaborator, which results in computation overhead.

E. Discussion

A noteworthy aspect of vertical federated learning is rebuilding the link between the distributed features. The typical approach uses identifiers shared by all the vertical partitions to reconstruct training samples for the global models ([45],



(a) Bike counts forecast of 2 days



(b) Yelp review forecast of 40 days

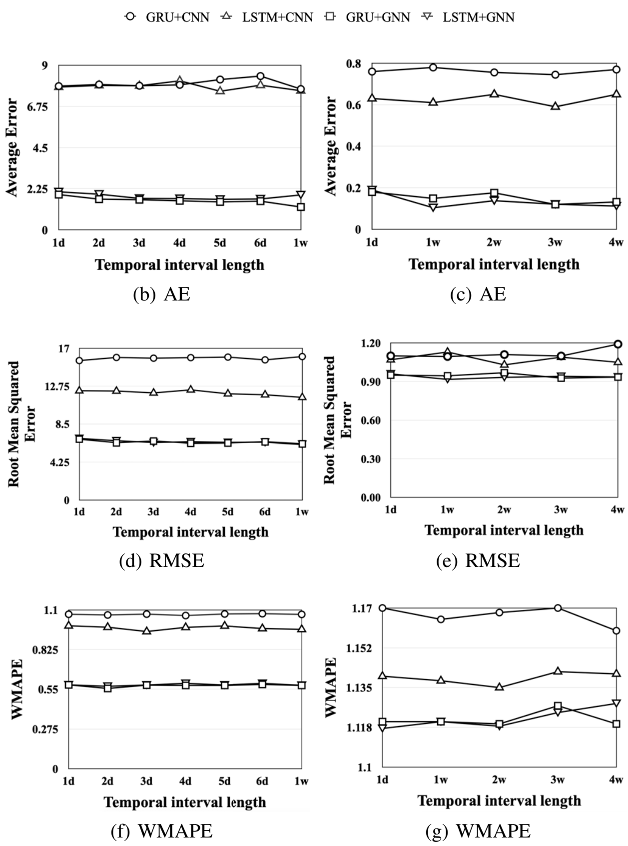
Fig. 6: Mobility forecast for Bike New York and Yelp review datasets.

[46]). In the case of mobility count data, timestamps can be considered as the shared identifiers. However, even gathering the mobility counts in one place would not solve the forecasting problem since we still need to rebuild the temporal ordering between samples to predict future timestamps. Unlike MVFF, the comparison frameworks uniformly aggregate the learning from each vertical partition to build a forecasting model. However, as demonstrated in the experimental evaluation, this aggregation approach performs less than the MVFF framework. This improvement is due to the temporal and spatial synchronization modules that reconstruct the spatio-temporal correlation between timestamps and locations.

VI. Related Work

A. Vertical Federated Learning

The global model training is dependent on the concatenation of local models in the vertical federated learning frameworks. This property requires the update of local models to be collaborative. One approach to train the global is entity resolution consisting of finding the correspondence between the rows of the datasets [46]. However, entity resolution techniques are error-prone [46]. Therefore, the authors in [46] consider two organizations that aim to jointly perform logistic regression in the cross-feature space. They study the effect of entity resolution on learning performance. This work is not



(b) AE (c) AE

(d) RMSE (e) RMSE

(f) WMAPE (g) WMAPE

Fig. 7: Metrics vs temporal interval length for the Bike New York and Yelp review datasets

applicable to our mobility forecasting because it is limited to logistic regression.

Privacy is another aspect of vertical federated learning since the participating organizations need to prevent data leakage. The work in [45] proposes algorithms for privacy-preserving two-party logistic regression for vertically partitioned data. Secure gradient descent is used to train the logistic regression model, and the intermediate calculated parameters are exchanged between the two participating organizations. The proposed algorithms do not suit our mobility forecasting since they are limited to only two participants.

B. Federated Mobility Forecasting Learning Models

The authors in [40] propose a GRU-based learning framework (FedGRU) for traffic flow prediction. First, the cloud server distributes a global model copy to all organizations. Each organization trains its copy on local data and shares the encrypted parameters with the cloud server. Once the parameters are aggregated to build a new global model, the cloud server distributes the new global model to each organization. This process repeats until the learning goal is achieved. Although we can apply this framework to our vertically partitioned data, the resulting trained model would lack the spatial correlation between locations belonging to different organizations.

Another work [9] proposes a wireless traffic prediction framework with a model that is trained collaboratively by multiple organizations. The framework groups the multiple organizations into different clusters based on geo-location information and traffic patterns to tackle the data heterogeneity. Then, a global model is trained and shared among the organizations' clusters. This framework is also applicable to our vertically partitioned data. However, the resulting model might lack the correlation between locations within the same cluster.

VII. Conclusion

This paper proposes a vertical federated learning framework for mobility data forecasting (MVFF). Our MVFF allows the learning process to be jointly conducted over multiple organizations targetting a common location domain to produce accurate mobility predictions. Using a local learning model, each organization extracts the embedded spatio-temporal correlation between its locations. A global model synchronizes between the local models to incorporate the correlation between all the organizations' locations. We investigate the performance of MVFF under four different variations of local and global models. We compare the MVFF's performance to two other federated frameworks on the real-life datasets: New York Bike and Yelp reviews, achieving better performances. In most real situations, the organizations may cover irregular mobility time periods, leading to the disparity data spread (non-independent and non-identically distributed data). Moreover, we assume that communication with all organizations is reliable, leading to continuous model updates. However, differences in communication reliability capabilities are inevitable among organizations. Therefore, we plan to investigate the integration of asynchronous federated learning strategies in the future.

References

[1] G. Kothai, E. Poovammal, G. Dhiman, K. Ramana, A. Sharma, M. A. AlZain, G. S. Gaba, and M. Masud, "A new hybrid deep learning algorithm for prediction of wide traffic congestion in smart cities," *Wireless Communications and Mobile Computing*, vol. 2021, 2021.

[2] H. Wei, G. Zheng, H. Yao, and Z. Li, "Intellilight: A reinforcement learning approach for intelligent traffic light control," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 2496–2505.

[3] A. S. S. Rao and J. A. Vazquez, "Identification of covid-19 can be quicker through artificial intelligence framework using a mobile phone–based survey when cities and towns are under quarantine," *Infection Control & Hospital Epidemiology*, vol. 41, no. 7, pp. 826–830, 2020.

[4] C. Ilin, S. Annan-Phan, X. H. Tai, S. Mehra, S. Hsiang, and J. E. Blumenstock, "Public mobility data enables covid-19 forecasting and management at local and global scales," *Scientific reports*, vol. 11, no. 1, pp. 1–11, 2021.

[5] U. Singh, J.-F. Determe, F. Horlin, and P. De Doncker, "Crowd forecasting based on wifi sensors and lstm neural networks," *IEEE transactions on instrumentation and measurement*, vol. 69, no. 9, pp. 6121–6131, 2020.

[6] W. Li, W. Tao, J. Qiu, X. Liu, X. Zhou, and Z. Pan, "Densely connected convolutional networks with attention lstm for crowd flows prediction," *IEEE Access*, vol. 7, pp. 140 488–140 498, 2019.

[7] S. Guo, Y. Lin, S. Li, Z. Chen, and H. Wan, “Deep spatial–temporal 3d convolutional neural networks for traffic data forecasting,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3913–3926, 2019.

[8] J. Feng, C. Rong, F. Sun, D. Guo, and Y. Li, “Pmf: A privacy-preserving human mobility prediction framework via federated learning,” *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 4, no. 1, pp. 1–21, 2020.

[9] C. Zhang, S. Dang, B. Shihada, and M.-S. Alouini, “Dual attention-based federated learning for wireless traffic prediction,” in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 2021, pp. 1–10.

[10] Q. Yang, Y. Liu, T. Chen, and Y. Tong, “Federated machine learning: Concept and applications,” *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, pp. 1–19, 2019.

[11] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation,” *arXiv preprint arXiv:1406.1078*, 2014.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.

[14] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, “The graph neural network model,” *IEEE transactions on neural networks*, vol. 20, no. 1, pp. 61–80, 2008.

[15] M. I. Jordan and T. M. Mitchell, “Machine learning: Trends, perspectives, and prospects,” *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[16] A. Khan, B. Baharudin, L. H. Lee, and K. Khan, “A review of machine learning algorithms for text-documents classification,” *Journal of advances in information technology*, vol. 1, no. 1, pp. 4–20, 2010.

[17] A. B. Sahakyan, V. S. Chambers, G. Marsico, T. Santner, M. Di Antonio, and S. Balasubramanian, “Machine learning model for sequence-driven dna g-quadruplex formation,” *Scientific reports*, vol. 7, no. 1, pp. 1–11, 2017.

[18] J. Konečnỳ, H. B. McMahan, D. Ramage, and P. Richtárik, “Federated optimization: Distributed machine learning for on-device intelligence,” *arXiv preprint arXiv:1610.02527*, 2016.

[19] A. Hard, K. Rao, R. Mathews, S. Ramaswamy, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage, “Federated learning for mobile keyboard prediction,” *arXiv preprint arXiv:1811.03604*, 2018.

[20] T. Chen, X. Jin, Y. Sun, and W. Yin, “Vafl: a method of vertical asynchronous federated learning,” *arXiv preprint arXiv:2007.06081*, 2020.

[21] Y. Liu, J. James, J. Kang, D. Niyato, and S. Zhang, “Privacy-preserving traffic flow prediction: A federated learning approach,” *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7751–7763, 2020.

[22] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, “On the properties of neural machine translation: Encoder-decoder approaches,” *arXiv preprint arXiv:1409.1259*, 2014.

[23] H. Lin, G. Liu, F. Li, and Y. Zuo, “Where to go? predicting next location in iot environment,” *Frontiers of Computer Science*, vol. 15, no. 1, pp. 1–13, 2021.

[24] Q. Zhou, J.-J. Gu, C. Ling, W.-B. Li, Y. Zhuang, and J. Wang, “Exploiting multiple correlations among urban regions for crowd flow prediction,” *Journal of Computer Science and Technology*, vol. 35, pp. 338–352, 2020.

[25] X. Tao, Y. Peng, F. Zhao, S. Wang, and Z. Liu, “An improved parallel network traffic anomaly detection method based on bagging and gru,” in *International Conference on Wireless Algorithms, Systems, and Applications*. Springer, 2020, pp. 420–431.

[26] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, “Face recognition: A convolutional neural-network approach,” *IEEE transactions on neural networks*, vol. 8, no. 1, pp. 98–113, 1997.

[27] C. Qin, J. Schlemper, J. Caballero, A. N. Price, J. V. Hajnal, and D. Rueckert, “Convolutional recurrent neural networks for dynamic mr image reconstruction,” *IEEE transactions on medical imaging*, vol. 38, no. 1, pp. 280–290, 2018.

[28] X. Song, X. Fan, C. Xiang, Q. Ye, L. Liu, Z. Wang, X. He, N. Yang, and G. Fang, “A novel convolutional neural network based indoor localization framework with wifi fingerprinting,” *IEEE Access*, vol. 7, pp. 110 698–110 709, 2019.

[29] G. Liu, Z. Yin, Y. Jia, and Y. Xie, “Passenger flow estimation based on convolutional neural network in public transportation system,” *Knowledge-Based Systems*, vol. 123, pp. 102–115, 2017.

[30] B. Zhu, W. Tang, X. Mao, and W. Yang, “Location-based hybrid deep learning model for purchase prediction,” in *2020 5th International Conference on Computational Intelligence and Applications (ICCIA)*. IEEE, 2020, pp. 161–165.

[31] H. Sun, C. Yang, L. Deng, F. Zhou, F. Huang, and K. Zheng, “Periodicmove: Shift-aware human mobility recovery with graph neural network,” in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2021, pp. 1734–1743.

[32] P. Roy, S. Sarkar, S. Biswas, F. Chen, Z. Chen, N. Ramakrishnan, and C.-T. Lu, “Deep diffusion-based forecasting of covid-19 by incorporating network-level mobility information,” in *Proceedings of the 2021 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, 2021, pp. 168–175.

[33] X. Wang, Y. Ma, Y. Wang, W. Jin, X. Wang, J. Tang, C. Jia, and J. Yu, “Traffic flow prediction via spatial temporal graph neural network,” in *Proceedings of The Web Conference 2020*, 2020, pp. 1082–1092.

[34] Y. Zheng, X. Xie, W.-Y. Ma *et al.*, “Geolife: A collaborative social networking service among user, location and trajectory.” *IEEE Data Eng. Bull.*, vol. 33, no. 2, pp. 32–39, 2010.

[35] J. Yuan, Y. Zheng, C. Zhang, W. Xie, X. Xie, G. Sun, and Y. Huang, “T-drive: driving directions based on taxi trajectories,” in *Proceedings of the 18th SIGSPATIAL International conference on advances in geographic information systems*, 2010, pp. 99–108.

[36] E. Cho, S. A. Myers, and J. Leskovec, “Friendship and mobility: user movement in location-based social networks,” in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2011, pp. 1082–1090.

[37] L. Gordon-Koven and N. Levenson, “Citi bike takes new york,” *Rudin Center for Transportation Management and Policy*, 2014.

[38] Yelp, “Yelp public dataset,” 2019. [Online]. Available: https://www.yelp.com/dataset

[39] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, “Graph attention networks,” in *International Conference on Learning Representations*, 2018.

[40] Y. Liu, S. Zhang, C. Zhang, and J. James, “Fedgru: Privacy-preserving traffic flow prediction via federated learning,” in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2020, pp. 1–6.

[41] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communication-efficient learning of deep networks from decentralized data,” in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[42] A. Taïk and S. Cherkaoui, “Electrical load forecasting using edge computing and federated learning,” in *ICC 2020-2020 IEEE International Conference on Communications (ICC)*. IEEE, 2020, pp. 1–6.

[43] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, “Pytorch: An imperative style, high-performance deep learning library,” *Advances in neural information processing systems*, vol. 32, pp. 8026–8037, 2019.

[44] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[45] S. Hardy, W. Henecka, H. Ivey-Law, R. Nock, G. Patrini, G. Smith, and B. Thorne, “Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption,” *arXiv preprint arXiv:1711.10677*, 2017.

[46] R. Nock, S. Hardy, W. Henecka, H. Ivey-Law, G. Patrini, G. Smith, and B. Thorne, “Entity resolution and federated learning get a federated resolution,” *arXiv preprint arXiv:1803.04035*, 2018.