REFOL: Resource-Efficient Federated Online Learning for Traffic Flow Forecasting

Qingxiang Liu, Sheng Sun, Yuxuan Liang, *Member, IEEE,* Xiaolong Xu, *Senior Member, IEEE,* Min Liu, *Senior Member, IEEE,* Muhammad Bilal, *Senior Member, IEEE,* Yuwei Wang, *Member, IEEE,* Xujing Li, and Yu Zheng, *Fellow, IEEE*

***Abstract*—Multiple federated learning (FL) methods are proposed for traffic flow forecasting (TFF) to avoid heavy-transmission and privacy-leaking concerns resulting from the disclosure of raw data in centralized methods. However, these FL methods adopt offline learning which may yield subpar performance, when concept drift occurs, i.e., distributions of historical and future data vary. Online learning can detect concept drift during model training, thus more applicable to TFF. Nevertheless, the existing federated online learning method for TFF fails to efficiently solve the concept drift problem and causes tremendous computing and communication overhead. Therefore, we propose a novel method named Resource-Efficient Federated Online Learning (REFOL) for TFF, which guarantees prediction performance in a communication-lightweight and computation-efficient way. Specifically, we design a data-driven client participation mechanism to detect the occurrence of concept drift and determine clients' participation necessity. Subsequently, we propose an adaptive online optimization strategy, which guarantees prediction performance and meanwhile avoids meaningless model updates. Then, a graph convolution-based model aggregation mechanism is designed, aiming to assess participants' contribution based on spatial correlation without importing extra communication and computing consumption on clients. Finally, we conduct extensive experiments on real-world datasets to demonstrate the superiority of REFOL in terms of prediction improvement and resource economization.**

***Index Terms*—Traffic flow forecasting, federated learning, concept drift, online learning, graph convolution.**

I. Introduction

Qingxiang Liu and Xujing Li are with Institute of Computing Technology, Chinese Academy of Sciences and also with University of Chinese Academy of Sciences, Beijing, China (e-mail: qingxiangliu737@gmail.com and lixujing19b@ict.ac.cn).

Sheng Sun and Yuwei Wang are with Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China (email: sunsheng@ict.ac.cn and ywwang@ict.ac.cn).

Yuxuan Liang is with Intelligent Transportation Thrust, Hong Kong University of Science and Technology (Guangzhou), Guangzhou, China (email: yuxliang@outlook.com).

Xiaolong Xu is with the School of Software, Nanjing University of Information Science and Technology, Nanjing 210044, China, also with the Jiangsu Collaborative Innovation Center of Atmospheric Environment and Equipment Technology (CICAEET), Nanjing University of Information Science and Technology, Nanjing 210044, China (e-mail: njuxlxu@gmail.com).

Min Liu is with Institute of Computing Technology, Chinese Academy of Sciences and also with Zhongguancun Lab, Beijing, China (email: liumin@ict.ac.cn).

Muhammad Bilal is with the School of Computing and Communications, Lancaster University, Bailrigg, Lancaster LA1 4WA, United Kingdom (e-mail: m.bilal@ieee.org).

Yu Zheng is with JD Intelligent Cities Research and JD iCity, JD Technology, Beijing, China (email: msyuzheng@outlook.com).

Min Liu is the corresponding author.

WITH the unprecedented growth of vehicles and unsatisfactory road planning, traffic congestion gets steadily worse and tremendously impacts the macro economy. It is forecast that drivers will expend $480 billion for time delay, fuel waste, and carbon emission caused by traffic congestion in the top 25 crowded cities of U.S. by 2027 [1]. For alleviating traffic jams, traffic flow forecasting (TFF) is regarded as the foremost component, based on which the administration is capable of foreseeing the burst flow and then taking measures accordingly [2]. Therefore, TFF has brought about extensive interests both in academia and industry, with great quantity of traffic nodes (i.e., sensors, loop detectors and radar-video integrated machines) deployed along roads generating enormous real-time traffic data [3] and large variety of innovative methods proposed to enhance performance gains in TFF.

Nowadays, most TFF methods rely on powerful deep neural networks to improve prediction performance, with Recurrent Neural Networks (RNN) [4] adopted to capture temporal patterns inside traffic flows and Convolutional Neural Networks (CNN) [5] as well as Graph Neural Networks (GNN) [6] used for evaluating spatial correlation among traffic nodes to boost performance. However, in these *centralized* methods, large volume of historical traffic data generated by traffic nodes should be transmitted to a central server for training prediction models, consequently resulting in considerable transmission overhead.

Federated learning (FL) emerges as a potent solution to tackle these problems above [7], where traffic nodes (termed *clients*) are orchestrated to collaboratively train a global prediction model without disclosure of raw data. Some recent researches have focused on forecasting traffic flow in FL architecture [8]–[10]. In [10], clients adopt encoder-decoder architecture of Gated Recurrent Units network (GRU) to model temporal patterns, and the central server uses GNN for updating encoders' hidden states to evaluate spatial correlation among traffic nodes. However, all of these proposed methods adopt the *offline* learning manner where prediction models are firstly trained based on historical traffic data and then deployed at traffic nodes for future traffic forecasting. When concept drift occurs, i.e., the distribution of historical traffic data and the newly-detected traffic data changes [11], directly applying these pre-trained models to the new traffic data may easily yield unsatisfactory prediction performance.

In contrast, *online* learning (OL) [12] makes it possible to detect concept drift upon newly-observed data arriving, merits of capturing fluctuating patterns and timely model updates, OL is more applicable for TFF to achieve much more attractive prediction performance. In [13], Liu *et al.* first propose a Federated Online Learning (FOL) method for traffic flow prediction by integrating OL into FL paradigm. Specifically, all clients need to update prediction models once they detect new traffic data, regardless of whether concept drift occurs, which results in large amount of computational cost for local optimization and communication cost for exchanging model parameters. Therefore, it is necessary to reasonably detect concept drift for avoiding resource waste derived from client-side ineffective model updates with little contribution to performance promotion. Besides, it is challenging to design a resource-efficient mechanism for evaluating the time-variant spatial correlation resulting from data fluctuation without frequent client-server parameter transmission and extra computational operations on clients, aiming to decrease resource consumption while guaranteeing prediction performance.

To this end, we propose a novel TFF method in FOL paradigm named Resource-Efficient Federated Online Learning (REFOL), aiming to reduce computational and communication cost at clients while guaranteeing prediction performance. Specifically, we first design a data-driven client participation mechanism, which enables each client to autonomously detect whether concept drift occurs by calculating data distribution divergence and further determine the necessity of local model updates. Furthermore, clients conduct adaptive online optimization locally to eliminate the influence of concept drift and guarantee performance gains. Finally, instead of simple averaging mechanism, we adopt a novel graph convolution-based model aggregation mechanism to efficiently evaluate the importance of local updated models depending on time-varying spatial correlation among traffic nodes and further yield the fresh global model with superior generalization ability.

The main contributions of this paper are summarized as follows:

- We propose a novel forecasting method called REFOL to precisely forecast traffic flow with spatio-temporal variation in a computation-efficient and communication-lightweight way under the federated online learning (FOL) paradigm.
- We design a data-driven client participation mechanism and an adaptive online optimization strategy, which makes clients detect concept drift and further determine requisite model updates at the least sacrifice of computing and communication resources.
- We design a novel graph convolution-based aggregation mechanism to ensure the generalization ability of global models, which can reasonably quantify participants' importance based on spatial correlation evaluation without importing extra resource consumption on traffic nodes.
- We validate the efficiency and effectiveness of REFOL by conducting comparison experiments with other FL and FOL prediction methods on two real-world datasets.

The remainder of this paper is organized as below. Section II presents the literature review on TFF, FL, and concept drift detection. In Section III, we formulate the TFF problem in FOL paradigm. Section IV elaborates the technical details of the proposed REFOL. Then, we analyze the performance of REFOL from experimental results in Section V. Finally, we conclude the paper in Section VI.

II. Related Work

A. *TFF*

Existing TFF methods can be divided into two categories, i.e, parametric and non-parametric methods. The most classical parametric methods are AutoRegressive Integrated Moving Average (ARIMA) [14] and its variants, e.g., Kohonen-ARIMA (KARIMA) [15] and subset ARIMA [16], where the parameters are interpretable. However, these methods suffer from subpar prediction performance when confronted with complicated traffic flow, since they are based on the assumption that the input sequences are stationary.

With the increasing number of traffic nodes and explosion of traffic data, these parametric methods get less effective and the non-parametric methods based on deep learning models have emerged. Due to the ability of capturing temporal patterns inside traffic sequences, RNN and its variants, e.g., Long-Short Term Memory network (LSTM) [17] and GRU [18] have been adopted in TFF. In [19], a novel model called Selected Stacked Gated Recurrent Units (SSGRU) is proposed to make traffic prediction in a road network. In [20], LSTM is used for short-term TFF, and hidden patterns contained in traffic data are analyzed to increase prediction accuracy. Furthermore, graph approaches such as GNN, Graph Convolution Network (GCN) and Graph Attention Network (GAT) are integrated to evaluate spatial correlations among traffic nodes. [21] proposed Diffusion Convolutional Recurrent Neural Network (DCRNN) and treated traffic flow as a diffusion process. [2] proposed the Attention based Spatial-Temporal Graph Convolutional Network (ASTGCN) considering dynamic changes of spatial-temporal correlation. Despite performance improvement, these methods adopt centralized training mode, which causes tremendous communication burden and poses a threat on privacy preserving.

B. *FL for TFF*

Due to the advantages of privacy protection and data localization, some researches focus on forecasting traffic flow in FL paradigm. Depending on what acts as the "client" in FL, these methods can be divided into three categories. In the first category, the cities work as clients which can train spatio-temporal prediction models with local datasets. Methods in this category aims to construct the prediction model upon cross-city datasets, thus promising effective prediction model for data-sparse cities by knowledge transfer [22]. Methods in the second category try to divide all traffic nodes into multiple organizations [8], [9], [23]–[27]. In [8], [9], [23]–[26], the inter-client dependencies are not considered, hardly guaranteeing the effective evaluation of the spatial correlation. While in [27], an extra intra-client spatial aggregation module is designed at clients to tackle such challenge. In the third category, the traffic nodes work as FL clients. In [10], traffic nodes are treated as clients and collaboratively update prediction model using local data. However, this method requires to transmit hidden states of encoder-GRU from clients to the central server for evaluating spatial correlation, inevitably increasing communication burden and prolonging the occupied time of model updates. Furthermore, all of them adopt offline learning manner and cannot meet the need of model re-training driven by data distribution fluctuation.

Liu *et al.* proposed an Online Spatio-Temporal Correlation-based FL (FedOSTC) method for traffic flow prediction in [13]. This is the first work on forecasting traffic flow in FOL paradigm. In FedOSTC, all clients are selected to perform local optimization once they receive new traffic data, in order to mitigate performance decreasing caused by concept drift. However, this method fails to consider practical model updating requirements on clients and results in unnecessarily tremendous communication and computational overhead.

C. Concept Drift Detection

Concept drift reflects the unforeseeable changes in underlying data distribution over time, and consequentially makes the existing prediction model unable to fit the upcoming data, resulting in poor prediction performance [11], [28]. It is of great importance to accurately detect concept drift, benefiting for timely model updating and efficiently preventing performance degradation. Concept drift detection methods can be classified into two categories, i.e, pre-prediction methods and after-prediction methods. The former devotes to quantifying the distribution divergence between the historical data and new data by certain divergence metrics before conducting prediction [29]–[31]. The latter first conducts prediction, and then evaluates the prediction error rate to judge whether concept drift occurs. The representative methods of this category are Drift Detection Drift (DDM) [32], Early DDM (EDDM) [33], Statistical Test of Equal Proportions Detection (STEPD) [34], *etc.*

III. Preliminaries and Problem Formulation

In this section, we present the introduction of TFF and concept drift detection. Then we formulate the problem of TFF in FOL paradigm. Finally, we present a vanilla FOL method for predicting traffic flow. For better understanding, we list primary symbols and their descriptions in Table I.

TABLE I
PRIMARY SYMBOLS AND DESCRIPTIONS IN SECTION III

Symbol	Description
r_n	The n-th traffic node.
N	Number of traffic nodes.
$e_{m,n}$	Adjacent edge between r_n and r_m.
$\mathcal{N}$	Traffic node set.
$\mathcal{E}$	Edge set among traffic nodes.
$\mathcal{G}$	Transportation network.
$s_{t,n}$	Traffic data detected by r_n at the t-th round.
$\mathcal{T}$	Number of rounds.
H	Historical horizon.
F	Forecasting horizon.
hw_n	Historical time window of r_n.
fw_n	Future time window of r_n.
$f(\cdot)$	Prediction model.
$S_{t,n}^H$	Inputting data of r_n at the t-th round.
$\hat{s}_{t,n}$	Predicted value of $s_{t,n}$.
$\hat{S}_{t,n}^F$	Predicted values of r_n at the t-th round.
$S_{t,n}^F$	True values of r_n at the t-th round.
$w_{t,n}$	Model parameters of r_n at the t-th round.
$\mathcal{L}(\cdot)$	Loss function.
w_n^*	Local optimal model of r_n.
rgt_n	Regret of r_n.
w^*	Global optimal model.

A. TFF

A traffic node is deployed at a certain road segment and responsible for monitoring the traffic speed of this segment. These traffic nodes compose a *transportation network*, which can be represented as a directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$. $\mathcal{N} = \{r_n \,|1 \leq n \leq N\}$ denotes the traffic node set, and r_n represents the n-th traffic node. $\mathcal{E} = \{e_{m,n} \,|1 \leq m,n \leq N\}$ denotes the edge set. If r_n is adjacent to r_m, $e_{m,n} \in \mathcal{E}$. Otherwise, $e_{m,n} \notin \mathcal{E}$. Let $\mathcal{S} = \{\mathcal{S}_1, \cdots, \mathcal{S}_t, \cdots, \mathcal{S}_\mathcal{T}\}$ denote the traffic data set, where $\mathcal{T}$ denotes the total number of time stamps and $\mathcal{S}_t = [s_{t,n}]_{1 \leq n \leq N}$. $s_{t,n} \in \mathbb{R}^1$ represents the traffic speed detected by r_n at the t-th time stamp. The TFF problem can be viewed as a process of predicting future traffic speed based on historical and current traffic speed. Without loss of generality, we consider to perform traffic prediction of F forecasting steps at the t-th time stamp based on the latest H historical steps, which can be formulated as $\{\mathcal{S}_{t-H+1}, \cdots, \mathcal{S}_t; \mathcal{G}\} \xrightarrow{\mathcal{F}} \left\{\hat{\mathcal{S}}_{t+1}, \cdots, \hat{\mathcal{S}}_{t+F}\right\}$. $\hat{\mathcal{S}}_{t+\delta} (1 \leq \delta \leq F)$ denotes the predicted value of $\mathcal{S}_{t+\delta}$ and $\mathcal{F}$ represents the prediction algorithm. The above process of forecasting traffic flow actually tackles the problem of modeling the spatio-temporal correlation, i.e., how to capture temporal dependence inside each traffic flow and evaluate spatial correlation among different traffic flows.

B. Federated Online Learning

In FOL paradigm, each traffic node is regarded as a *client*. Once detecting new traffic data, the clients adopt the locally-saved prediction models or request the fresh global model for prediction, and perform local update if necessary. The central server aggregates updated local models. At the t-th time stamp (i.e., t-th round in FOL paradigm), the prediction problem of r_n can be formulated as

$$\hat{S}_{t,n}^F = f\left(S_{t,n}^H; w_{t,n}\right), \tag{1}$$

where $f(\cdot)$ denotes the adopted prediction model and $w_{t,n}$ denotes the model parameters of r_n at the t-th round. $S_{t,n}^H = (s_{t-H+1,n}, s_{t-H+2,n}, ..., s_{t,n})$ stands for the input of the prediction model, and $\hat{S}_{t,n}^F = (\hat{s}_{t+1,n}, \hat{s}_{t+2,n}, ..., \hat{s}_{t+F,n})$ stands for the predicted value of $S_{t,n}^F = (s_{t+1,n}, s_{t+2,n}, \cdots, s_{t+F,n})$, where $\hat{s}_{t+1,n}$ denotes the predicted value of $s_{t+1,n}$.

Definition 1 (Local Optimal Model). *We assume that r_n performs prediction distributedly. After $\mathcal{T}$ rounds, all traffic data $\{s_{1,n}, s_{2,n}, \cdots, s_{\mathcal{T},n}\}$ are available to r_n. Therefore, the* *integrated training dataset* $\mathcal{TD}_n = \left\{\left(S^H_{t,n}, S^F_{t,n}\right) | 1 \leq t \leq \mathcal{T}\right\}$ *could be obtained.* r_n *could optimize the prediction model* $f(\cdot)$ *based on* $\mathcal{TD}_n$ *and would obtain the local optimal model, which is denoted as* w^*_n*. Formally,* w^*_n *is formulated as*

$$w^*_n = \arg\min_w \left\{\sum\nolimits_{t=1}^{\mathcal{T}} \mathcal{L}\left(f\left(S^H_{t,n}; w\right), S^F_{t,n}\right)\right\}, \quad (2)$$

where $\mathcal{L}(\cdot)$ *stands for the loss function to evaluate the discrepancy between the predicted and true values. It is intuitive that for the given dataset* $\mathcal{TD}_n$ *and the prediction model* $f(\cdot)$*, the local optimal model is fixed.*

We define rgt_n as the prediction regret of r_n, which is obtained via evaluating the prediction loss difference between the actual prediction model $w_{t,n}(1 \leq t \leq \mathcal{T})$ and the local optimal model w^*_n over all $\mathcal{T}$ rounds. The calculation of rgt_n with respect to w^*_n can be expressed as

$$\begin{aligned} rgt_n(w^*_n) = &\sum\nolimits_{t=1}^{\mathcal{T}} \mathcal{L}(f(S^H_{t,n}; w_{t,n}), S^F_{t,n}) \\ &- \sum\nolimits_{t=1}^{\mathcal{T}} \mathcal{L}(f(S^H_{t,n}; w^*_n), S^F_{t,n}). \end{aligned} \quad (3)$$

Definition 2 (**Global Optimal Model**)**.** *We denote* $\mathcal{TD} = \{\mathcal{TD}_n | 1 \leq n \leq N\}$ *as the integrated training datasets of all clients. If we input* $\mathcal{TD}$ *into* $f(\cdot)$ *and perform optimization, the global optimal model* w^* *can be obtained, which is formulated formally as*

$$w^* = \arg\min_w \left\{\sum\nolimits_{n=1}^{N} \sum\nolimits_{t=1}^{\mathcal{T}} \mathcal{L}(f(S^H_{t,n}; w), S^F_{t,n})\right\}. \quad (4)$$

For the given $\mathcal{TD}$ *and* $f(\cdot)$*, the global optimal model is also fixed.*

The objective of TFF in federated online learning paradigm is to minimize the prediction regret with respective to the global optimal model over all N traffic nodes, which is formulated as

$$\min \left\{\sum\nolimits_{n=1}^{N} rgt_n(w^*)\right\}. \quad (5)$$

$$\begin{aligned} \text{where} \quad rgt_n(w^*) = &\sum\nolimits_{t=1}^{\mathcal{T}} \mathcal{L}(f(S^H_{t,n}; w_{t,n}), S^F_{t,n}) \\ &- \sum\nolimits_{t=1}^{\mathcal{T}} \mathcal{L}(f(S^H_{t,n}; w^*), S^F_{t,n}). \end{aligned}$$

C. Vanilla Federated Online Learning Method

Federated Averaging (FedAvg) is a vanilla FL method which adopts offline learning [35]. We integrate OL into FedAvg for forecasting traffic flow, and the execution process is elaborated in Algorithm 1. Specifically, the central server first randomly selects a subset of available clients denoted as $\mathcal{N}_t$ to participate in this round. Each selected client concurrently makes prediction (Line 9) and updates model parameters via Online Gradient Descent (OGD) [12] for E epochs (Line 10-11). After finishing online optimization locally, these selected clients transmit updated model parameters to the server, which conducts averaging aggregation to generate the fresh global model (Line 6). The process continues until no new traffic data arrive.

Algorithm 1: FOL-vanilla

Input: Initialized model w_1, learning rate η.
Output: The global model $w_{\mathcal{T}+1}$.

1 **SERVEREXECUTE:**
2 **for** $t = 1, 2, \cdots, \mathcal{T}$ **do**
3 $\quad \mathcal{N}_t \leftarrow$ randomly select N_t clients.
4 $\quad$ **for** r_n *in* $\mathcal{N}_t$ **do**
5 $\quad\quad w_{t+1,n} \leftarrow$ `ClientExecute` (w_t, n)
6 $\quad w_{t+1} \leftarrow \frac{1}{N_t} \sum_{r_n \in \mathcal{N}_t} w_{t+1,n}$
7 **return** $w_{\mathcal{T}+1}$
8 **Function** `ClientExecute`(w, n)**:**
9 $\quad$ Make prediction via Eq. (1).
10 $\quad$ **for** $e = 1, 2, \cdots, E$ **do**
11 $\quad\quad w \leftarrow w - \eta \nabla \mathcal{L}(f(S^H_{t,n}; w), S^F_{t,n})$
12 $\quad$ **return** w

IV. METHODOLOGY

In this section, we elaborate the proposed TFF method, i.e, resource-efficient federated online learning (REFOL). The architecture of REFOL is shown in Fig. 1. Specifically, REFOL is composed of *data-driven client participation mechanism*, *adaptive online optimization*, and *graph convolution-based model aggregation*.

A. Data-driven Client Participation Mechanism

Intuitively, if the traffic flow distribution changes (i.e., *concept drift* occurs), the model parameters need to be updated to guarantee the prediction performance. It is a straightforward solution that in each round, the server selects all clients as participants to timely optimize their local prediction models based on input data. However, some clients do not need to perform optimization, if their historically-saved prediction models still work. Hence, this simple approach inevitably generates needless communication overhead for exchanging model parameters and compute cost for local optimization. To this end, we design a data-driven client participation mechanism which can skillfully detect the occurrence of concept drift based on traffic data distribution, and further offer clients the autonomy to decide whether to participate in local training.

Specifically, assuming the locally-saved model of r_n is generated at the u-th round based on traffic sequence $S^H_{u,n}$, denoted as $w_{u+1,n}$. At the t-th round $(u+1 \leq t \leq \mathcal{T})$, we set $hw_n = S^H_{u,n}$ and $fw_n = S^H_{t,n}$, as is shown in Fig. 2(a). Then, we utilize widely-used Kullback-Leibler Divergence (KLD) [36] to calculate the distribution divergence between $S^H_{t,n}$ and $S^H_{u,n}$, which can be denoted as $D_{KL}(S^H_{t,n}||S^H_{u,n})$. Higher KLD value indicates the distribution of $S^H_{t,n}$ is less similar to $S^H_{u,n}$, and vice verse.

We define Q as the threshold of KLD values. If $D_{KL}(S^H_{t,n}||S^H_{u,n})$ doesn't reach Q, we draw the conclusion that $w_{u+1,n}$ is still competent on the current traffic flow $S^H_{t,n}$. Therefore, r_n does not need to download the up-to-date global model from the central server and then perform local model updates. In this case, hw_n keeps stable, while fw_n moves

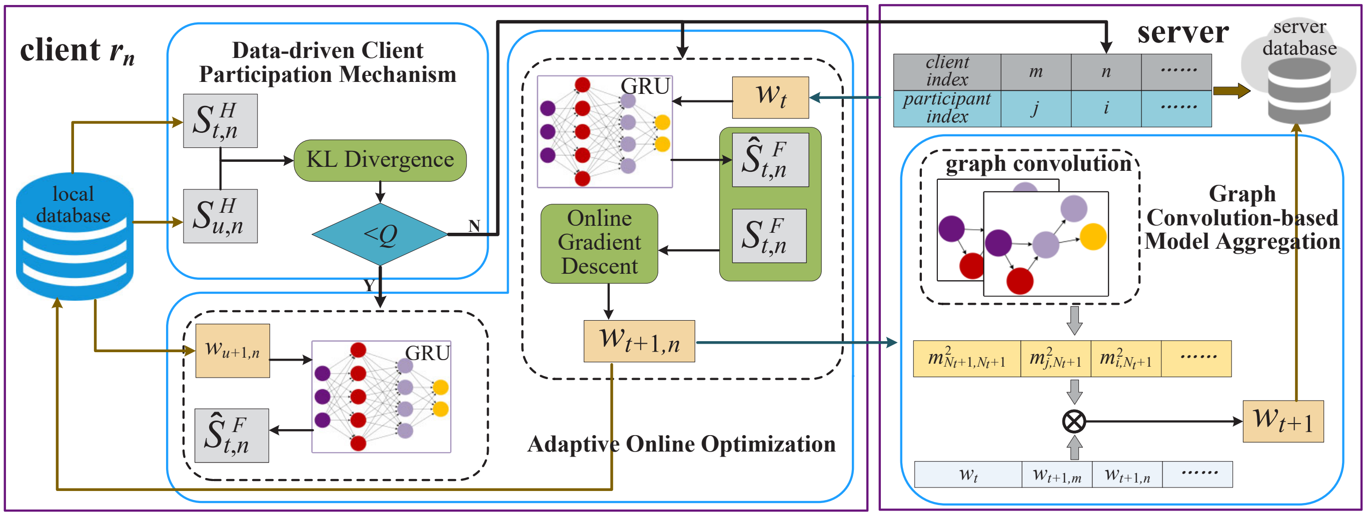
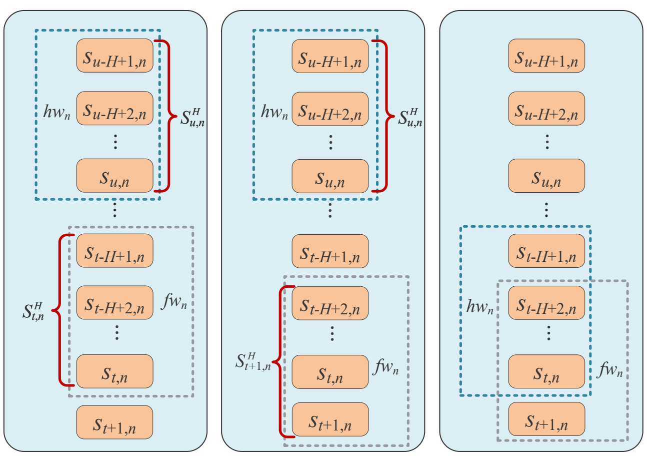


Fig. 1. The architecture of REFOL includes three modules, i.e., data-driven participation mechanism, adaptive online optimization, and graph convolution-based model aggregation. Each client determines whether to participate in this round of training based on data-driven client participation mechanism and further performs adaptive online optimization accordingly. The central server collects local optimized model parameters from participants and conducts graph convolution-based model aggregation.



(a) (b) (c)

Fig. 2. The changing process of hw_n and fw_n in concept drift detection.

one step backwards, as is shown in Fig. 2(b). Otherwise, we regard that concept drift occurs at the t-th round and $w_{u+1,n}$ is no longer applicable to $S^H_{t,n}$. Therefore, r_n should participate in the t-th round federated training (details in the next subsection). In this case, the historical window is updated as $hw_n = S^F_{t,n}$, as is shown in Fig. 2(c).

B. Adaptive Online Optimization

Each client firstly determines whether to participate in this round of training based on the subsection above. To improve prediction performance with least communication and computing overhead, we design an adaptive online optimization strategy, which is described from the following two cases.

The first case: When $D_{KL}(S^H_{t,n}||S^H_{u,n}) < Q$, r_n regards non-occurrence of concept drift and will not participate in the t-th round. It will utilize the previous local model to perform forecasting operation (i.e., $w_{t,n} = w_{u+1,n}$), which can avoid unnecessary communication overhead for downloading the up-to-date global model.

We reformulate $S^H_{t,n}$ as $S^H_{t,n} = (s_{a,n}, t - H + 1 \leq a \leq t)$. In this paper, we adopt the widely-used GRU model [37] as the prediction model. Concretely, the hidden state h^a_n can be obtained as follows:

$$u^a_n = \sigma\left(W^{(u)}_n s_{a,n} + U^{(u)}_n h^{a-1}_n\right), \tag{6}$$

$$r^a_n = \sigma(W^{(r)}_n s_{a,n} + U^{(r)}_n h^{a-1}_n), \tag{7}$$

$$h^{a'}_n = \tanh(W^{(h)}_n s_{a,n} + r^a_n \odot U^{(h)}_n h^{a-1}_n), \tag{8}$$

$$h^a_n = u^a_n \odot h^{a-1}_n + (1 - u^a_n) \odot h^{a'}_n, \tag{9}$$

where $W^{(u)}_n \in \mathbb{R}^{1\times hs}, U^{(u)}_n \in \mathbb{R}^{hs\times hs}, W^{(r)}_n \in \mathbb{R}^{1\times hs}, U^{(r)}_n \in \mathbb{R}^{hs\times hs}, W^{(h)}_n \in \mathbb{R}^{1\times hs}$ and $U^{(h)}_n \in \mathbb{R}^{hs\times hs}$ are the parameters of the GRU network. hs denotes hidden size of the GRU network. $\sigma(\cdot)$ represents the sigmoid function. u^a_n and r^a_n denote the update gate and reset gate respectively. The above-mentioned process is iterated for H times and then the hidden states $\{h^a_n | t - H + 1 \leq a \leq t\}$ are fed into a fully connected layer for yielding the final predicted value $\hat{S}^F_{t,n}$.

Since temporal patterns of local traffic data remain stable, r_n has no effects on changes in spatial correlation and fails to contribute to the fresh global model. Therefore, r_n will neither locally optimize $w_{t,n}$ nor upload $w_{t,n}$ to the central server for aggregation, thus further saving computing and communication resources.

The second case: When $D_{KL}(S^H_{t,n}||S^H_{u,n}) \geq Q$, r_n regards concept drift occurs and hence the locally-saved model $w_{u+1,n}$ is not applicable to $S^H_{t,n}$. Therefore, r_n will participate in the t-th round. We name r_n a *participant*. It first downloads the global model w_t as the prediction model, i.e., $w_{t,n} = w_t$, and then generates the predicted value $\hat{S}^F_{t,n}$ with the same operations as Eq. (6)-(9). After performing prediction, r_n starts to optimize its local model parameters via Online Gradient Descent (OGD) [12] for E epochs, which can be formulated as

$$w_{t,n}^{(e)} = w_{t,n}^{(e-1)} - \eta \nabla \mathcal{L}(f(S_{t,n}^{H}; w_{t,n}^{(e-1)}), S_{t,n}^{F}), \tag{10}$$

where $w_{t,n}^{(0)} = w_{t,n}$ and $1 \leq e \leq E$. After finishing local optimization, r_n saves $w_{t+1,n}$ ($w_{t+1,n} = w_{t,n}^{(E)}$) as its local model, and then uploads $w_{t+1,n}$ to the central server for aggregation.

The execution process of data-driven client participation mechanism and adaptive online optimization is summarized in Algorithm 2. If the KLD value $D_{KL}(\bar{S}_{t,n}^{H} || \bar{S}_{u,n}^{H}) < Q$, the locally-saved prediction model $w_{u+1,n}$ still works and r_n directly performs prediction (Line 2-4). Otherwise, r_n considers the occurrence of concept drift at the t-th time stamp and downloads the up-to-date global model w_t for prediction and then performs local optimization (Line 5-11). r_n saves the locally-optimized model $w_{t+1,n}$ as well as traffic sequence $S_{t,n}^{H}$, and pops out $w_{u+1,n}$ and $S_{u,n}^{H}$ from local database.

Algorithm 2: Data-driven Client Participation Mechanism and Adaptive Online Optimization

Input: Locally-saved model $w_{u+1,n}$, $S_{t,n}^{H}$, $S_{u,n}^{H}$, global model w_t, and learning rate η.
Output: Predicted value $\hat{S}_{t,n}^{F}$.

1 Calculate divergence $D_{KL}(S_{t,n}^{H} || S_{u,n}^{H})$.
2 **if** $D_{KL}(S_{t,n}^{H} || S_{u,n}^{H}) < Q$ **then**
3 $\quad w_{t,n} \leftarrow w_{u+1,n}$
4 $\quad r_n$ performs prediction via Eq. (6)-(9) and generates predicted value $\hat{S}_{t,n}^{F}$.
5 **else**
6 $\quad r_n$ downloads the global model w_t.
7 $\quad w_{t,n} \leftarrow w_t$
8 $\quad r_n$ performs prediction via Eq. (6)-(9) and generates predicted value $\hat{S}_{t,n}^{F}$.
9 $\quad w_{t,n}^{(0)} \leftarrow w_{t,n}$
10 $\quad$ **for** $e \leq E$ **do**
11 $\quad\quad$ Perform local optimization as Eq. (10).
12 $\quad w_{t+1,n} \leftarrow w_{t,n}^{(E)}$
13 $\quad$ Keep $w_{t+1,n}$ locally.
14 $\quad$ Upload $w_{t+1,n}$ to the aggregator.

C. Graph Convolution-based Model Aggregation

Generally, different participants exhibit diverse importance in TFF scenarios due to the coupling spatial correlation. Therefore, it is necessary to govern aggregation weights of participating nodes based on sophisticated spatial correlation for obtaining the updated global model with satisfactory generalization ability. However, existing FL methods evaluate participants' importance by importing an additional assessment procedure of spatial correlation, calling for frequent transmission and thus resulting in tremendous communication overhead. Thus, we design a novel graph convolution-based model aggregation mechanism, the execution process of which is shown in Fig. 3. It leverages graph convolution to online quantify time-variant spatial correlation among participants and correspondingly yields aggregation weights in a communication-lightweight and computation-efficient manner. The whole procedure is elaborated as follows.

*1) **Graph Construction**:* Suppose N_t clients participate in the t-th round according to the designed data-driven client participation mechanism. These participants execute the process of adaptive online optimization and subsequently upload the updated local models to the central server for generating the fresh global model w_{t+1}. They can also compose a directed graph, denoted as $\mathcal{G}_t = (\mathcal{N}_t, \mathcal{E}_t)$. $\mathcal{N}_t = \{r_{\mathrm{I}_{t,i}} | 1 \leq i \leq N_t\}$, where $\mathrm{I}_{t,i}$ represents the client index of the i-th participant at the t-th round and $\mathcal{N}_t \subset \mathcal{N}$. As is shown in Fig. 3(a), the purple nodes represent the participants. $\mathcal{E}_t = \{e_{\mathrm{I}_{t,i},\mathrm{I}_{t,j}} | e_{\mathrm{I}_{t,i},\mathrm{I}_{t,j}} \in \mathcal{E}\}$ represents the edge set of these participants. It is explicit that $\mathcal{E}_t \subset \mathcal{E}$ and $\mathcal{G}_t$ is a subgraph of $\mathcal{G}$. Let $A_t = [a_{i,j}]_{1 \leq i,j \leq N_t}$ denote the adjacent matrix. If $e_{\mathrm{I}_{t,i},\mathrm{I}_{t,j}} \in \mathcal{E}_t$, $a_{i,j} = 1$. Otherwise $a_{i,j} = 0$. Let $D_t = diag\left(d_{\mathrm{I}_{t,1}}, \cdots, d_{\mathrm{I}_{t,i}}, \cdots, d_{\mathrm{I}_{t,N_t}}\right)$ denote the indegree matrix, where $d_{\mathrm{I}_{t,i}}$ represents the indegree of $r_{\mathrm{I}_{t,i}}$.

We treat local models as participants' features and perform graph convolution on $\mathcal{G}_t$. If $\mathcal{G}_t$ is not a connected graph (as is shown in Fig. 3(a)), no participants will aggregate features (local models) from all the other ones after graph convolution. In order to obtain the up-to-date global model after graph convolution, we introduce a virtual participant $r_{\mathrm{I}_{t,N_t+1}}$ (the red node in Fig. 3(b)), and w_t is regarded as its feature. Furthermore, we define that all participants are adjacent to $r_{\mathrm{I}_{t,N_t+1}}$ and it is self-related. Accordingly, we have $\mathcal{N}_t' = \mathcal{N}_t \cup \{r_{\mathrm{I}_{t,N_t+1}}\}$ and $\mathcal{E}_t' = \mathcal{E}_t \cup \{e_{\mathrm{I}_{t,i},\mathrm{I}_{t,N_t+1}} | 1 \leq i \leq N_t + 1\}$ respectively. We can obtain the updated adjacent and indegree matrix $A_t' = [a_{i,j}]_{1 \leq i,j \leq N_t+1}$ and $D_t' = diag\left(d_{\mathrm{I}_{t,1}}, \cdots, d_{\mathrm{I}_{t,i}}, \cdots, d_{\mathrm{I}_{t,N_t}}, d_{\mathrm{I}_{t,N_t+1}}\right)$ respectively. Hereafter, $\mathcal{G}_t' = (\mathcal{N}_t', \mathcal{E}_t')$ is a connected graph and $r_{\mathrm{I}_{t,N_t+1}}$ is capable of aggregating all local models and generates the global model w_{t+1} after graph convolution.

*2) **Graph Convolution**:* Graph Convolution performs convolution directly on graphs [38], which makes each node aggregate the information from its adjacent ones. The conventional operation can be generally described as

$$Out = \sigma(D^{-\frac{1}{2}} A D^{-\frac{1}{2}} L P), \tag{11}$$

where D and A represent the indegree matrix and the adjacent matrix respectively. L and Out denote the input and the output of graph convolution. $\sigma(\cdot)$ represents the sigmoid function. P denotes the parameter matrix which needs iterative training. In this mechanism, to prevent disturbance on the global model and simultaneously bring the benefits of reducing computational burden and execution time, we omit the sigmoid function as well as additional parameter matrix. Concretely, the graph convolution operation on $\mathcal{G}_t'$ can be formulated as

$$\mathrm{W}_{\mathrm{t}}' = D_t'^{-\frac{1}{2}} A_t' D_t'^{-\frac{1}{2}} \mathrm{W}_{\mathrm{t}}, \tag{12}$$

where W_{t} and W_{t}' represent the set of raw and updated features respectively.

For simplicity, we define the graph convolution operation as M_t. $M_t = D_t'^{-\frac{1}{2}} A_t' D_t'^{-\frac{1}{2}}$ and $M_t = [m_{i,j}]_{1 \leq i,j \leq N_t+1}$. The

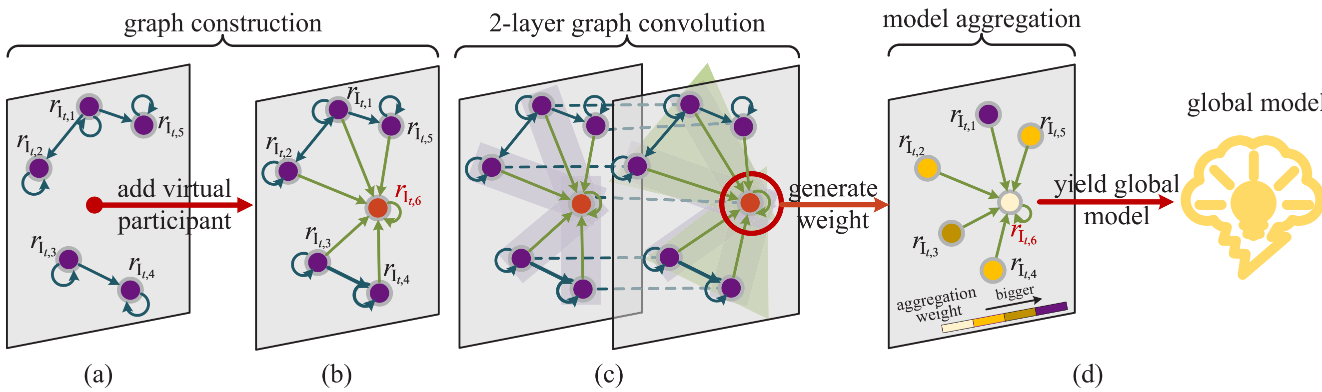


Fig. 3. The execution process of graph convolution-based model aggregation contains three parts, i.e., graph construction, 2-layer graph convolution , and model aggregation.

element $m_{i,j}$ represents the weight of $r_{\mathrm{I}_{t,i}}$ with respective to $r_{\mathrm{I}_{t,j}}$. According to the rules of matrix operations, we have

$$m_{i,j} = \begin{cases} \frac{1}{\sqrt{d_{\mathrm{I}_{t,i}}}} \frac{1}{\sqrt{d_{\mathrm{I}_{t,j}}}}, & \text{if } a_{i,j} = 1, \\ 0, & \text{otherwise.} \end{cases} \tag{13}$$

It is explicit that $m_{i,j}$ depends on $d_{\mathrm{I}_{t,i}}$, $d_{\mathrm{I}_{t,j}}$ and $a_{i,j}$. Note that we only need to focus on the computation of m_{i,N_t+1}, due to the objective of generating the fresh global model w_{t+1} at $r_{\mathrm{I}_{t,N_t+1}}$ after graph convolution. With $a_{i,N_t+1} = 1$, the difference of $m_{i,N_t+1}(1 \leq i \leq N_t+1)$ lies in the difference of $d_{\mathrm{I}_{t,i}}$. $d_{\mathrm{I}_{t,i}} > d_{\mathrm{I}_{t,j}}$ indicates that $r_{\mathrm{I}_{t,i}}$ has larger indegree, and therefore traffic data of $r_{\mathrm{I}_{t,i}}$ is influenced by more participants. Hence, $r_{\mathrm{I}_{t,i}}$ is less important in terms of spatial correlation compared with $r_{\mathrm{I}_{t,j}}$. It is reasonable that $m_{i,N_t+1} < m_{j,N_t+1}$. That is, $w_{t,\mathrm{I}_{t,i}}$ should be put a smaller weight than $w_{t,\mathrm{I}_{t,j}}$ when performing model aggregation. When $d_{\mathrm{I}_{t,i}} = d_{\mathrm{I}_{t,j}}$, $w_{t,\mathrm{I}_{t,i}}$ and $w_{t,\mathrm{I}_{t,j}}$ are thought to have equal importance in aggregation process.

However, the above strategy is unreasonable since the outdegrees of $r_{\mathrm{I}_{t,i}}$ and $r_{\mathrm{I}_{t,j}}$ are not considered. The outdegree of a certain participant can evaluate how many participants it affects, and thus the participant with larger outdegree should be paid more attention to when preforming model aggregation.

In fact, this problem is subject to the inherent characteristic of graph convolution, where the receptive fields depend on the layers of graph convolution. That is, after g-layer graph convolution, each participant can aggregate features from the ones whose distance to it is g steps. Since all participants are adjacent to $r_{\mathrm{I}_{t,N_t+1}}$, $r_{\mathrm{I}_{t,N_t+1}}$ is capable of aggregating all local updated models after 1-layer graph convolution. However, $r_{\mathrm{I}_{t,N_t+1}}$ fails to capture the adjacency among real participants, as is shown in Fig. 3(c). Therefore, we propose to leverage 2-layer graph convolution for $r_{\mathrm{I}_{t,N_t+1}}$ to obtain the adjacency among all participants.

Let V_t denote the 2-layer graph convolution operation and we have $V_t = M_t M_t$

We define $V_t = \left(D_t'^{-\frac{1}{2}} A_t' D_t'^{-\frac{1}{2}}\right)\left(D_t'^{-\frac{1}{2}} A_t' D_t'^{-\frac{1}{2}}\right)$ as two-layer graph convolution operation, and $V_t = [v_{i,j}]_{1 \leq i,j \leq N_t+1}$. Likewise, we only focus on the values of $v_{i,N_t+1}\,(1 \leq i \leq N_t+1)$ and have

$$v_{i,N_t+1} = \sum_{k=1}^{N_t+1} m_{i,k} m_{k,N_t+1}. \tag{14}$$

It is intuitive that the calculation of v_{i,N_t+1} involves all 2-step paths from $r_{\mathrm{I}_{t,i}}$ to $r_{\mathrm{I}_{t,N_t+1}}$. According to graph construction, the longest path from $r_{\mathrm{I}_{t,i}}(1 \leq i \leq N_t+1)$ to $r_{\mathrm{I}_{t,N_t+1}}$ is 2 steps. Therefore, 2-layer graph convolution is enough to obtain the inner spatial correlation among multiple participants. Besides, the execution of graph convolution excludes parameters to be trained iteratively and is just based on matrix multiplication operation on the server side, which is regarded as computation-lightweight. In addition, this method doesn't need participants to transmit extra parameters, thus evidently decreasing communication overhead.

*3) **Model Aggregation**:* We first perform normalization as

$$v_{i,N_t+1} = v_{i,N_t+1}/sum_t, (1 \leq i \leq N_t+1)\,, \tag{15}$$

where $sum_t = \sum_{i=1}^{N_t+1} v_{i,N_t+1}$. We deem that v_{i,N_t+1} can efficiently evaluate the importance of local models trained on $r_{\mathrm{I}_{t,i}}$ for model aggregation, and thus is directly regarded as the aggregation weight of $w_{t,\mathrm{I}_{t,i}}$. Therefore, the fresh global model w_{t+1} is ultimately yielded as

$$w_{t+1} = w_t v_{N_t+1,N_t+1} + \sum_{i=1}^{N_t} w_{t,\mathrm{I}_{t,i}} v_{i,N_t+1}. \tag{16}$$

D. Execution Process of REFOL

In this subsection, the whole procedure of REFOL is elaborated in Algorithm 3. At the beginning of each round, each client firstly detects concept drift by evaluating distribution difference of traffic data between the current and previous rounds using KLD, followed by comparing with threshold Q to determine whether to download the fresh global model or reuse the locally-saved model for online forecasting using Algorithm 2. After then the participants upload the updated models to the aggregator for aggregation. The aggregator calculates the aggregation weights of participants via 2-layer graph convolution, and then performs model aggregation to generate the fresh global model (Line 5-8).

Algorithm 3: REFOL

Input: Initialized model parameter w_1, learning rate η, threshold Q, client set $\mathcal{N}$.
Output: The global model $w_{\mathcal{T}+1}$.

SERVEREXECUTE:
for $t = 1, 2, \cdots, \mathcal{T}$ **do**
  for $r_n \in \mathcal{N}$ *in parallel* **do**
    `ClientExecute` (n, t)
  Receive local updated models from the participants.
  Perform 2-layer graph convolution via Eq. (13)(14).
  Normalize aggregation weights via Eq. (15).
  Perform aggregation via Eq. (16) and generate w_{t+1}.
return $w_{\mathcal{T}+1}$
Function `ClientExecute` (n, t)**:**
  Execute local processes as Algorithm 2.

V. EXPERIMENTS

In this section, comprehensive experiments are conducted to validate the high efficiency of our proposed REFOL. Firstly, we present a brief introduction of the system configurations including datasets, metrics, experiment settings, and baselines. Then we analyze the performance comparisons of REFOL and baselines. Finally, the ablation study is conducted and the effects of varying parameter settings on prediction performance are further explored.

A. System Configuration

*1) **Datasets and Metrics**:* The experiment datasets are generated from two real-world datasets: PEMS-BAY and METR-LA [21]. PEMS-BAY includes the vehicular speed from 325 sensors in the Bay Area from 2017/1/1 to 2017/5/31. METR-LA is comprised of traffic speed collected by 207 sensors in the highway of Los Angeles County from 2012/3/1 to 2012/6/30. The adjacent matrices of sensors are constructed as per [21]. Two widely-used metrics for regression tasks, Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) are adopted to assess the prediction performance. RMSE and MAE are calculated respectively as follows:

$$\text{RMSE} = \frac{\sum_{n=1}^{N}\sum_{t=1}^{\mathcal{T}}\sqrt{\frac{\sum_{\tau=1}^{F}(s_{t+\tau,n}-\hat{s}_{t+\tau,n})^2}{F}}}{N\mathcal{T}}, \quad (17)$$

$$\text{MAE} = \frac{\sum_{n=1}^{N}\sum_{t=1}^{\mathcal{T}}\frac{\sum_{\tau=1}^{F}|s_{t+\tau,n}-\hat{s}_{t+\tau,n}|}{F}}{N\mathcal{T}}. \quad (18)$$

*2) **Experimental Setting**:* All experiments are conducted on a server with Intel(R) Xeon(R) Gold 5218 CPU @ 2.30GHz and two NVIDIA Geforce RTX 3090 Founders Edition GPUs. We implement REFOL with PyTorch, where a GRU network with 1 layer with 128 hidden cells is used as the prediction model. The learning rate is set to 0.001 and the local epoch E is set to 5. The KLD threshold Q is set to 0.0003 (how to determine such threshold is analyzed in Subsection V-F). The historical horizon H is set to 12. We adopt MSE as the loss function. We divide the traffic flow series into samples by sliding window strategy, as per [10]. For batch learning methods, all samples are split to train, validation, and test datasets with the ratio of 7:1:2. For online learning, we average the prediction errors of the samples belonging to the test dataset in the batch learning methods to achieve fair comparison.

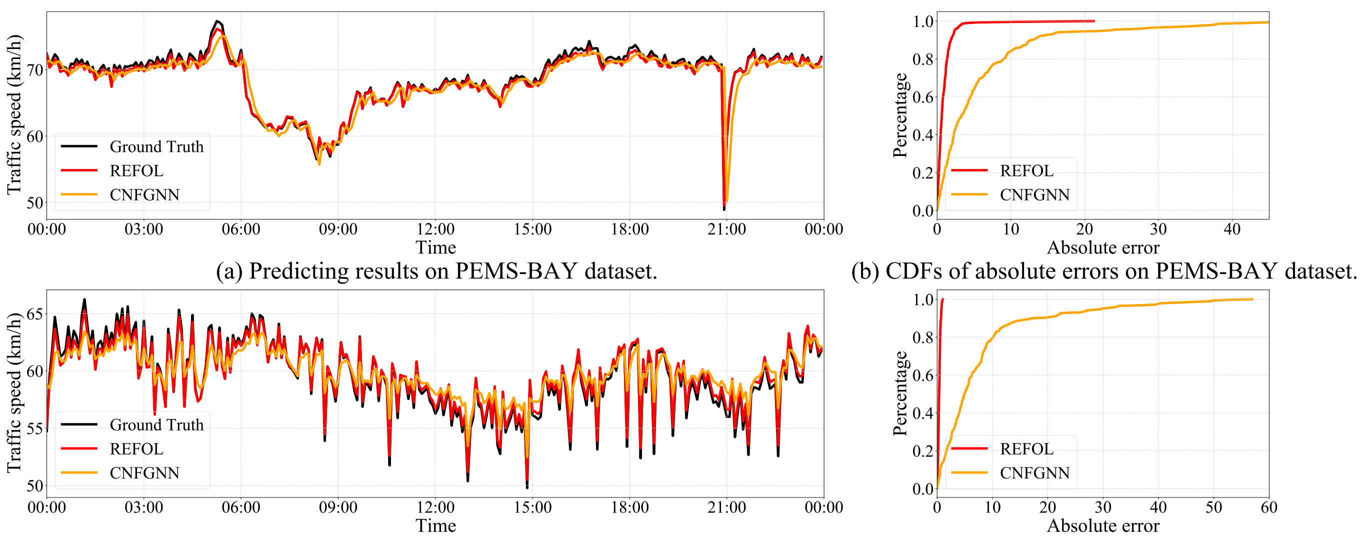
B. Prediction Performance Comparison

In this subsection, we compare the prediction performance of REFOL with that of 12 baselines, with 5 centralized methods (SVR [39], GRU [18], DCRNN [21], STGCN [40], and MegaCRN [41]) and 7 FL methods (FedAvg [35], FedGRU [8], FCGCN [23], FASTGNN [9], CNFGNN [10], FedGTP [27], and pFedCTP [22]). Note that all of these baselines are trained in offline learning and the configurations of baselines comply with the corresponding literature, unless stated otherwise. The brief introduction of these baselines is presented as follows.

- SVR (Support Vector Regression) [39]: Each client independently predicts traffic flow using SVR and doesn't interact with the central server.
- GRU [18]: The central server conducts prediction using GRU based on the uploaded raw traffic data.
- DCRNN [21]: It is a centralized prediction method, which treats traffic flow as a diffusion process and also evaluates spatio-temporal correlation among clients.
- STGCN [40]: It is centralized method, named spatio-temporal graph convolutional network, where the graph convolution and gated temporal convolution are adopted for fast training and accurate performance.
- MegaCRN [41]: It is a centralized method, where meta-graph convolutional recurrent network is proposed to tackle the spatio-temporal heterogeneity.
- FedAvg [35]: It is a conventional framework, where the central server aggregates the locally-updated model parameters via averaging mechanism.
- FedGRU [8]: Traffic nodes are divided into clusters and train local GRU models. The server conducts aggregation by averaging mechanism.
- FCGCN [23]: Traffic nodes are divided into local road networks, each of which is treated as a client in the FL paradigm and trains a GCN model. The server aggregates the GCN model parameters by averaging mechanism.
- FASTGNN [9]: Like FedGRU, each cluster works as FL client which adopts GAT to assess the spatial correlation and GRU for prediction. The server aggregates the adjacency matrix and model parameters across clusters.
- CNFGNN [10]: It is a FL method, which adopts encoder-decoder architecture of GRU and GNN to predict traffic flow based on evaluating spatio-temporal correlation.
- FedGTP [27]: It is a novel federated graph-based traffic prediction framework for better capturing intra-client spatial dependencies.
- pFedCTP [22]: It is a personalized federated learning method for cross-city traffic prediction, which aims to construct personalized prediction models for data-sparse cities by knowledge transfer.

TABLE II
COMPARISON IN PREDICTION PERFORMANCE ON TWO DATASETS WITH DIFFERENT FORECASTING STEPS

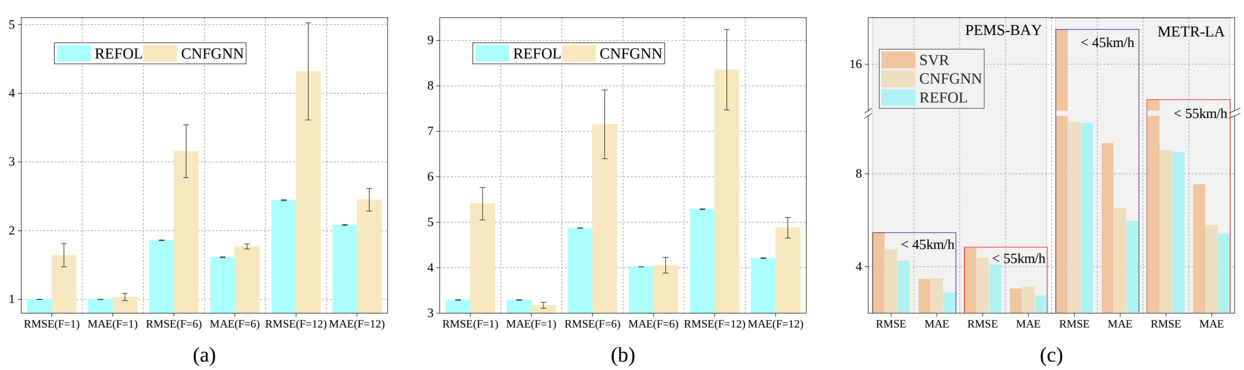
Methods	PEMS-BAY						METR-LA					
	5min ($F=1$)		30min ($F=6$)		1h ($F=12$)		5min ($F=1$)		30min ($F=6$)		1h ($F=12$)	
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
SVR	2.01	1.09	3.74	1.79	4.80	2.28	6.43	3.46	8.41	4.34	9.61	5.00
GRU	2.02	0.99	3.93	1.77	5.50	2.42	5.56	**2.91**	8.13	4.17	9.41	4.90
DCRNN	1.65	0.93	4.79	2.09	6.19	2.71	4.30	3.57	7.09	4.61	8.71	5.33
STGCN	1.46	**0.86**	3.62	1.73	4.43	2.26	5.92	3.82	8.60	4.62	10.66	5.92
MegaCRN	1.61	0.91	4.42	1.94	5.33	2.33	4.24	3.54	6.77	4.51	8.11	5.09
FedAvg	1.80	1.02	3.72	1.78	5.25	2.41	5.84	3.32	8.16	4.45	9.52	5.22
FedGRU	1.82	0.97	4.72	2.27	6.52	3.37	5.71	2.99	9.23	5.00	11.03	6.56
FCGCN	9.25	4.82	9.45	4.97	9.36	5.04	15.37	9.69	15.45	10.04	15.16	9.83
FASTGNN	5.32	2.82	6.06	3.20	6.84	3.76	11.73	7.27	12.98	8.23	12.82	8.16
CNFGNN	1.82	1.04	3.60	1.73	5.13	2.64	5.82	3.25	8.03	4.24	9.38	5.12
FedGTP	1.75	1.00	4.59	2.06	6.10	2.74	6.11	2.96	9.34	4.40	11.77	5.66
pFedCTP	5.29	2.84	6.49	3.40	7.43	3.92	8.88	5.21	10.56	6.02	12.08	7.15
REFOL	**1.00**	1.00	**1.86**	**1.61**	**2.44**	**2.08**	**3.29**	3.29	**4.87**	**4.02**	**5.29**	**4.21**



(a) Predicting results on PEMS-BAY dataset. (b) CDFs of absolute errors on PEMS-BAY dataset.

(c) Predicting results on METR-LA dataset. (d) CDFs of absolute errors on METR-LA dataset.

Fig. 4. Ground truth values and forecasting values of CNFGNN and REFOL.



(a) (b) (c)

Fig. 5. (a): Error bars on PEMS-BAY. (b): Error bars on METR-LA. (c): Prediction performance in the conditions of traffic jams.

The RMSE and MAE results of different methods are summarized in Table II. We compare the predicting errors of the 12 methods with different forecasting steps, i.e., $F = 1$, $F = 6$ and $F = 12$. RMSE and MAE values of REFOL on two datasets are equal in the case of $F = 1$. It is because that when $F = 1$, we have

$$\sqrt{\frac{\sum_{\tau=1}^{F}\left(s_{t+\tau,n}-\hat{s}_{t+\tau,n}\right)^2}{F}} = \left|s_{t+1,n}-\hat{s}_{t+1,n}\right|. \quad (19)$$

We have the key observations from Table II that the proposed REFOL performs best among the baselines on both datasets. Specifically, compared with state-of-the-art FL method CNFGNN, REFOL can offer RMSE gains of 45.05%, 48.33%, and 52.44% on PEMS-BAY dataset when $F = 1$, $F = 6$, and $F = 12$ respectively. Likewise, for METR-LA dataset, REFOL obtains RMSE gains of 43.43% ($F = 1$), 39.35% ($F = 6$), and 43.60% ($F = 12$) respectively.

We compare predicted values of different methods. The predicted results and cumulative distribution functions (CDFs) of absolute errors are shown in Fig. 4. As is illustrated in Fig. 4(a) and (c), the predicted values of REFOL have the same distribution with the ground truth on both datasets. In terms of prediction errors, REFOL yields much smaller absolute errors, compared with CNFGNN. As shown in Fig. 4 (b), on PEMS-BAY dataset, 67% of the absolute errors in REFOL are smaller than 1, while 20% in CNFGNN. As shown in Fig. 4 (d), the comparisons on METR-LA dataset are more remarkable, with 100% and 14% for REFOL and CNFGNN respectively. Hence, we can conclude that REFOL yields superior prediction performance, compared with state-of-the-art FL method and the other offline methods.

Moreover, we conduct experiments on CNFGNN and REFOL for 5 times and report the mean and standard deviation values of prediction errors, as shown in Fig. 5 (a) and (b). We can observe that the standard deviation values do not exceed 0.01 and the mean values decrease consistently in all forecasting scenarios. It indicates that REFOL can significantly and stably outperform CNFGNN, which agrees with the findings in Table II. Given than traffic control centers may be more interested in accurate prediction during congested time periods, we investigate the prediction performance in traffic congestion. When the traffic speed is lower than 45 km/h or 55 km /h, we deem that traffic congestion occurs. We respectively calculate the prediction errors when true traffic speed values do not exceed such criteria. As shown in Fig. 5 (c), REFOL can always achieve the superior performance, especially on PEMS-BAY dataset, with 10.5% (4.6%) RMSE reduction compared with CNFGNN when 45 km/h (55 km/h) is adopted as the criterion. The performance gains result from the effectiveness of evaluating spatio-temporal correlation in a fine-grained way.

C. Comparison of Computational and Communication Cost at Clients

In this subsection, we compare the communication and computational cost of all clients in REFOL with the other two FOL prediction methods. These FOL methods are introduced as follows.

- FedOSTC [13]: All clients are selected as participants at each round. Each client trains an encoder-decoder architecture of GRU locally. The server maintains a GAT for evaluating the spatial correlation and aggregates local models via averaging strategy.
- FOL-vanilla: Different from REFOL, the server randomly selects a certain number of clients as participants at each round. For fair comparison, the participant number is equal to the averaged participant number over $\mathcal{T}$ rounds in REFOL.

*1) **Computational Cost Analysis**:* Adaptive online optimization contains two functions, i.e., prediction and local optimization. In fact, these two functions can be fulfilled by the processes of $forward\ propagation$ and $backward\ propagation$ in the context of machine learning. Hence, we calculate the computational cost of the two processes. The forward propagation of GRU is shown in Eq. (6)-(9) and we ignore the computational cost generated by Eq. (7). According to [42], the computational cost of Eq. (6)-(8) can be calculated as $(1 + hs) \times hs \times 3 \times 2(\text{FLOPs})$. In addition, the computational cost of the fully connected layer is $hs \times 2(\text{FLOPs})$. Therefore, the computational cost of one forward propagation in GRU is

$$(1 + hs) \times hs \times 3 \times 2 + hs \times 2(\text{FLOPs}). \quad (20)$$

According to [43], the computational cost of backward propagation is twice as much as that of the forward propagation. Therefore, the computational cost of one backward propagation is

$$2 \times ((1 + hs) \times hs \times 3 \times 2 + hs \times 2)\,(\text{FLOPs}). \quad (21)$$

In FedOSTC, the encoder and decoder are all GRU networks. Therefore, we can calculate the computational cost accordingly as above.

Furthermore, we consider the computational cost in calculating KLD values. The process can be split into three operations as

- r_n calculates the summation of $s_{t-i,n}(0 \le i \le H-1)$ and $s_{u-i,n}(0 \le i \le H-1)$, and the corresponding computational cost is $H \times 2(\text{FLOPs})$.
- r_n divides $s_{t-i,n}(0 \le i \le H-1)$ and $s_{u-i,n}(0 \le i \le H-1)$ by $\sum_{i=0}^{H-1} s_{t-i,n}$ and $\sum_{i=0}^{H-1} s_{u-i,n}$ respectively. The computational cost is $H \times 2(\text{FLOPs})$.
- r_n finally calculates the KLD value, which is composed by operations of division, logarithm, multiplication, and addition. The generated computational cost is $H \times 3(\text{FLOPs})$.

Therefore, the computational cost of calculating KLD value is $H \times 7(\text{FLOPs})$.

*2) **Communication Cost Analysis**:* Then we dive into how to calculate the communication cost of all clients in different methods. For FOL-vanilla and REFOL, the participants just need to transmit model parameters of the prediction models (i.e., GRU network). Based on Eq. (6)-(9), the model parameter amount of GRU layer can be calculated as$(3 \times hs + 3 \times hs \times hs + 3 \times hs)$. In addition, the parameter amount of the fully connected layer is $(hs + 1)$. Therefore, the parameter amount of prediction model can be calculated as

$$3 \times hs \times (hs + 2) + hs + 1. \quad (22)$$

TABLE III
COMPARISONS IN COMPUTATIONAL AND COMMUNICATION COST AT CLIENTS

Methods	PEMS-BAY							
	5min (F=1)		30min (F=6)		1h (F=12)		Computational Cost/(GFLOPs)	Communication Cost/(GB)
	RMSE	MAE	RMSE	MAE	RMSE	MAE		
FedOSTC	0.96	0.96	1.74	1.51	1.91	1.62	533.45	303.41
FOL-vanilla	6.55	6.55	6.73	6.47	6.87	6.41	125.00	37.72
REFOL	1.00	1.00	1.86	1.61	2.44	2.08	125.03	37.72
Methods	METR-LA							
	5min (F=1)		30min (F=6)		1h (F=12)		Computational Cost/(GFLOPs)	Communication Cost/(GB)
	RMSE	MAE	RMSE	MAE	RMSE	MAE		
FedOSTC	2.90	2.90	4.35	3.66	4.98	4.10	533.45	303.41
FOL-vanilla	11.72	11.72	12.22	11.69	12.55	11.81	310.65	113.71
REFOL	3.29	3.29	4.87	4.02	5.29	4.21	310.68	113.71

Since the encoder and decoder module in FedOSTC are all GRU networks, the model parameter amount can also be calculated accordingly. Furthermore, the additional parameters should be exchanged between clients with the aggregator in FedOSTC, for updating clients' hidden states and optimizing GNN parameters, which can be calculated as follows [10]:

$$hs + [3 \times 2hs \times (2hs + 2) + (2hs + 1)]. \tag{23}$$

*3) **Comparison Analysis**:* The experimental results on prediction performance, computational cost, and communication cost at clients are presented in Table III. It is illustrated that the FOL-vanilla performs worst in terms of perdiction accuracy, in spite of the equal participants number as REFOL. In FedOSTC, all clients participate in each federated training round. Therefore, the computational and communication costs keep constant on two datasets. While in REFOL, each client need to determine whether to participate in federated round based on traffic distribution, the cost reductions on different datasets vary, resulting variant computational and communication costs on PEMS-BAY and METR-LA datasets.

Although FedOSTC achieves the best prediction accuracy, compared with FOL-vanilla and REFOL, the yielded computational cost and communication cost are much higher than the other two methods, mainly because all clients are selected as participants at each training round and extra parameters should be transmitted for updating hidden states and for optimizing GAT's parameters. Compared with FOL-vanilla, REFOL has an extra lightweight process of calculating KLD values at clients. Therefore, the total computational cost of clients in REFOL is little higher than that in FOL-vanilla. But more importantly, REFOL can guarantee prediction performance, with a small drop from the best-performed FedOSTC, while significantly decreasing the computational and communication costs by $\mathbf{76.56}\%$ and $\mathbf{87.57}\%$ ($\mathbf{41.76}\%$ and $\mathbf{62.52}\%$) on PEMS-BAY dataset (METR-LA dataset) respectively. This improvement results from the designed data-driven client participation mechanism to avoid redundant model updates and graph convolution-based model aggregation to integrate the process of evaluating spatial correlation into model aggregation.

D. Ablation Study

In this subsection, we conduct ablation study to verify the effectiveness of different components in REFOL. Firstly, we transform the REFOL method into the following variants.

- REFOL-D: Each client online performs prediction distributedly. When concept drift occurs, clients independently optimize their locally-owned models.
- REFOL-V1: The central server randomly selects participants at each round and aggregates local models based on our designed aggregation mechanism. We calculate the average number of participants per round in REFOL as the selected number in REFOL-V1.
- REFOL-V2: It is the same with our designed REFOL except that the server aggregates local models based on the averaging strategy.

We conduct extensive experiments on PEMS-BAY dataset to compare the prediction performance of different variants and the predicting results are summarized in Table IV.

It is plain that REFOL performs best among all the variants with different forecasting steps. Compared with the three variants, the averaged (RMSE, MAE) gains of REFOL are (1.95, 1.95), (1.92, 1.90), and (1.78, 1.72) with $F = 1, 6$, and 12 respectively. Therefore, the smaller the forecasting step is, the more performance gains REFOL will offer.

Specifically, compared with REFOL-D, clients can download the global model from the central server in REFOL, when detecting the occurrence of concept drift. The performance gains in REFOL demonstrate that the global model is superior to local models and the FOL paradigm is more effective than the distributed learning paradigm. Compared with REFOL-V1, REFOL can offer performance gains of 74% in terms of RMSE, which indicates the high efficiency of data-driven client participation mechanism. In REFOL-V2, the central server utilizes averaging aggregation strategy to yield the updated global model without evaluating participating importance of clients. Consequently, global model generalization in REFOL-V2 decreases and the predicting performance declines compared with REFOL, indicating the effectiveness of graph convolution-based aggregation mechanism.

TABLE IV
ABLATION TESTS ON PEMS-BAY DATASET

Methods	Participation determination	Aggregation strategy	5min (F=1)		30min (F=6)		1h (F=12)	
			RMSE	MAE	RMSE	MAE	RMSE	MAE
REFOL-D	*data-driven*	✗	1.22	1.22	2.53	2.26	2.99	2.60
REFOL-V1	*random selection-based*	*graph convolution-based*	6.62	6.62	6.72	6.46	6.93	6.48
REFOL-V2	*data-driven*	*averaging*	1.00	1.00	2.10	1.82	2.73	2.32
REFOL	*data-driven*	*graph convolution-based*	1.00	1.00	1.86	1.61	2.44	2.08

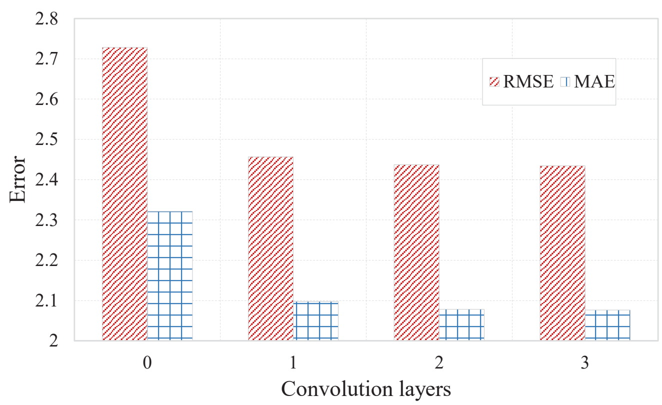


Fig. 6. The number of graph convolution layers versus prediction performance on PEMS-BAY dataset.

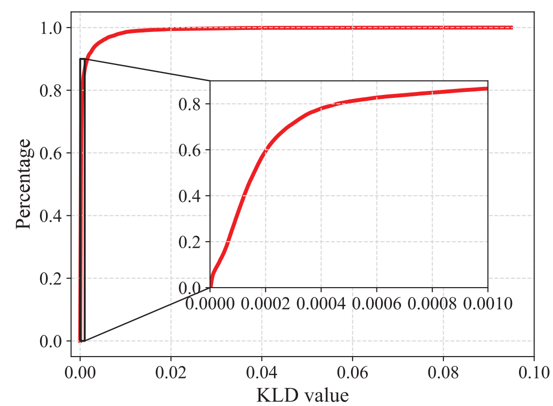


Fig. 7. Distribution of KLD values on PEMS-BAY dataset.

E. Effect of Graph Convolution Layers on Prediction Performance

The number of graph convolution layers determines how well the central server can evaluate spatial dependence among participants and ultimately affects the prediction performance. Hence, in this subsection, we explore the effects of graph convolution layers on prediction performance. The values of RMSE and MAE with different numbers of convolution layers on PEMS-BAY dataset are shown in Fig. 6. When the number of convolution layers is 0, the central server adopts the conventional averaging aggregation mechanism to generate the fresh global model. It is intuitive that our proposed graph convolution-based model aggregation mechanism offers much more performance gains compared with the averaging aggregation mechanism, which demonstrates the effectiveness of our proposed aggregation mechanism. Furthermore, with 1-layer graph convolution, the central server fails to evaluate spatial correlation comprehensively, thus yielding subpar prediction performance. With 2-layer graph convolution, the spatial correlation among participants can be well evaluated. When the layer number is greater than 2, the increased layers of graph convolution will pose more computational pressure on the central server but offer few performance improvement. As shown in Fig. 6, the performance difference with 2- and 3-layer graph convolution is negligible. Overall, the experimental results demonstrate the superiority of our proposed graph convolution-based model aggregation mechanism. This reason is that the novel mechanism embraces the assessment of spatial correlation among participants, which can increase global model generalization and further improve prediction performance.

F. Influence of Q on Prediction Performance

In this subsection, we analyze how determine the value of Q and explore the influence of Q on prediction performance. We first calculate the KLD values between two neighbour traffic sequences for each client, and then calculate the cumulative distribution of these values. The results on PEMS-BAY dataset are shown in Fig. 7. It is plain that most KLD values are less than 0.001 and especially range from 0 to 0.0006. Then, we conduct experiments with 7 different settings of Q (from 0 to 0.0006). Fig. 8 (a) and (b) show how RMSE and participation proportion change with Q varying. We can observe that when Q increases, the concept drift detection criterion is less strict and fewer client participate in federated training (participation proportion declines with Q increasing), leading to poorer performance (RMSE increases with Q increasing). Specifically, the prediction performance of REFOL is consistently superior to that of the baselines, until Q increases to 0.0003. When $Q = 0.0003$, the participation proportion is 28%. Therefore, the computational cost for local optimization and communication cost for exchanging model parameters can be decreased by up to 72%. Therefore, to strike a balance between prediction performance and resource consumption, Q is finally determined as 0.0003.

Furthermore, we explore the efficiency of data-driven client participation mechanism. We randomly choose experimental results of 3 clients with $F = 1$ and $Q = 0.0003$ on PEMS-BAY dataset. Fig. 8 (c) and (d) show the raw traffic flows and

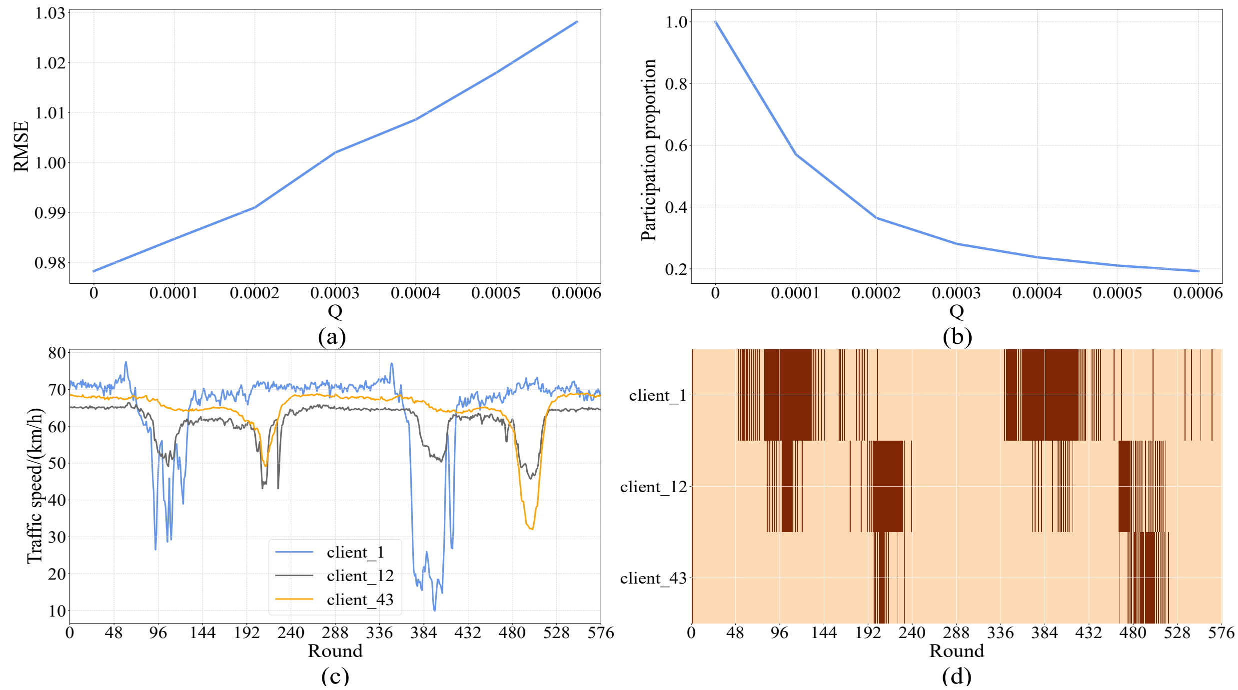


Fig. 8. (a) Prediction RMSE versus Q, (b) participation proportion versus Q, (c) traffic flows of three random clients, and (d) participation of these three clients.

the participation of these three clients respectively. In Fig. 8 (d), when the client downloads the global model at a certain round, the corresponding region is marked as red. It is intuitive that the participation of clients has much to do with the raw raw traffic flows. Specifically, in the range of the 48-th to 144-th round, there is fluctuation in the traffic flow of r_1 and the KLD values are over the given KLD threshold. Therefore, r_1 participates and requests the global model from the central server at most rounds in this range. However, the traffic flow of r_{43} is smooth enough, and r_{43} does not detect concept drift and hence reuses the local model for prediction. Therefore, we argue that our proposed data-driven client participation mechanism is efficient in detecting concept drift, and thus enables clients to reasonably determine whether to participate in model updates at each round.

VI. Conclusion and Discussion

In this paper, we investigate the TFF problem and propose a novel FOL method named REFOL, aiming at improving prediction performance without bringing out unnecessary computational and communication overhead. We design a data-driven client participation mechanism to detect the occurrence of concept drift by evaluating distribution divergence of traffic data. Accordingly, each client decides whether to participate in model updates and further performs adaptive online optimization at each round, which can not only guarantee the prediction performance but also avoid unnecessary computing and communication overhead for insignificant model optimization. Furthermore, we build the immediate evaluation of time-varying spatial correlation in the aggregation process and propose a graph convolution-based model aggregation mechanism, which gets rid of client-side extra resource waste for evaluating spatial dependence like existing FL methods. Finally, comprehensive experiments are conducted on PEMS-BAY and METR-LA datasets to validate the superiority of REFOL in terms of prediction improvement and resource saving.

To increase the transparency and reliability of our REFOL, we conduct further discussion as follows.

- **Universality.** REFOL can effectively evaluate the concept drift (i.e., traffic distribution shift) and therefore avoid extra model optimization, which guarantees the prediction performance and simultaneously reduces the computational and communication cost. REFOL can apply to other spatio-temporal forecasting tasks, such as cellular traffic forecasting, weather forecasting, and retail sales forecasting, etc. In different tasks, the KLD threshold Q can adjusted to satisfy specific requirements in prediction accuracy and resource consumption.
- **Scalability.** The proposed REFOL can adapt to real-world traffic forecasting scenarios with hundreds or thousands of traffic nodes. Given the complex network conditions in real world, the process of exchanging model parameters between nodes and the central server may be blocked. Therefore, we can supplement waiting time stamps for the central server and nodes. If the time stamp at node expires, it will reuse the locally-saved model parameters. If the time stamp at the central server expires, it will aggregate the punctual model parameters. Moreover, considering that distant nodes have little or no correlation, the topological graph can be split into multiple subgraphs to mitigate the computation load of the central server. We will follow the idea to refine REFOL to accommodate to large-scale traffic nodes in the future works.

- **Limitations and Future Works.** We assume that all traffic nodes are furnished with similar computing and communication resources. However, traffic nodes may be equipped with imbalanced resources in real world. In such asynchronous FL paradigm, a simple strategy of adapting REFOL may be setting time stamps at the central server and traffic nodes. However, how to guarantee the prediction performance and efficiency is non-trivial. Hence, in the future works, we will dive into asynchronous FL methods for TFF.

Acknowledgments

This work is supported by the National Key Research and Development Program of China (2021YFB2900102) and the National Natural Science Foundation of China (No. 62072436 and No. 62202449).

References

[1] T. Afrin and N. Yodo, "A survey of road traffic congestion measures towards a sustainable and resilient transportation system," *Sustainability*, vol. 12, no. 11, p. 4660, 2020.

[2] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 922–929.

[3] A. Miglani and N. Kumar, "Deep learning models for traffic flow prediction in autonomous vehicles: A review, solutions, and challenges," *Vehicular Communications*, vol. 20, p. 100184, 2019.

[4] F. Zhao, G.-Q. Zeng, and K.-D. Lu, "Enlstm-wpeo: Short-term traffic flow prediction by ensemble lstm, nnct weight integration, and population extremal optimization," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 1, pp. 101–113, 2020.

[5] Y. Liang, K. Ouyang, L. Jing, S. Ruan, Y. Liu, J. Zhang, D. S. Rosenblum, and Y. Zheng, "Urbanfm: Inferring fine-grained urban flows," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp. 3132–3142.

[6] C. Zheng, X. Fan, C. Wang, and J. Qi, "Gman: A graph multi-attention network for traffic prediction," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 01, 2020, pp. 1234–1241.

[7] 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.

[8] 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.

[9] C. Zhang, S. Zhang, J. James, and S. Yu, "Fastgnn: A topological information protected federated learning approach for traffic speed forecasting," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 12, pp. 8464–8474, 2021.

[10] C. Meng, S. Rambhatla, and Y. Liu, "Cross-node federated graph neural network for spatio-temporal data modeling," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 1202–1211.

[11] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, "Learning under concept drift: A review," *IEEE transactions on knowledge and data engineering*, vol. 31, no. 12, pp. 2346–2363, 2018.

[12] S. C. Hoi, D. Sahoo, J. Lu, and P. Zhao, "Online learning: A comprehensive survey," *Neurocomputing*, vol. 459, pp. 249–289, 2021.

[13] Q. Liu, S. Sun, M. Liu, Y. Wang, and B. Gao, "Online spatio-temporal correlation-based federated learning for traffic flow forecasting," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 10, pp. 13 027–13 039, 2024.

[14] W. W. Wei, "Time series analysis," in *The Oxford Handbook of Quantitative Methods in Psychology: Vol. 2*, 2006.

[15] M. Van Der Voort, M. Dougherty, and S. Watson, "Combining kohonen maps with arima time series models to forecast traffic flow," *Transportation Research Part C: Emerging Technologies*, vol. 4, no. 5, pp. 307–318, 1996.

[16] S. Lee and D. B. Fambro, "Application of subset autoregressive integrated moving average model for short-term freeway traffic volume forecasting," *Transportation research record*, vol. 1678, no. 1, pp. 179–188, 1999.

[17] B. Yang, S. Sun, J. Li, X. Lin, and Y. Tian, "Traffic flow prediction using lstm with feature enhancement," *Neurocomputing*, vol. 332, pp. 320–327, 2019.

[18] G. Dai, C. Ma, and X. Xu, "Short-term traffic flow prediction method for urban road sections based on space–time analysis and gru," *IEEE Access*, vol. 7, pp. 143 025–143 035, 2019.

[19] P. Sun, A. Boukerche, and Y. Tao, "Ssgru: A novel hybrid stacked gru-based traffic volume prediction approach in a road network," *Computer Communications*, vol. 160, pp. 502–511, 2020.

[20] Y.-L. Hsueh and Y.-R. Yang, "A short-term traffic speed prediction model based on lstm networks," *International journal of intelligent transportation systems research*, vol. 19, no. 3, pp. 510–524, 2021.

[21] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=SJiHXGWAZ

[22] Y. Zhang, H. Lu, N. Liu, Y. Xu, Q. Li, and L. Cui, "Personalized federated learning for cross-city traffic prediction," in *33rd International Joint Conference on Artificial Intelligence, IJCAI 2024*. International Joint Conferences on Artificial Intelligence, 2024, pp. 5526–5534.

[23] M. Xia, D. Jin, and J. Chen, "Short-term traffic flow prediction based on graph convolutional networks and federated learning," *IEEE Transactions on Intelligent Transportation Systems*, 2022.

[24] L. Liu, Y. Tian, C. Chakraborty, J. Feng, Q. Pei, L. Zhen, and K. Yu, "Multilevel federated learning-based intelligent traffic flow forecasting for transportation network management," *IEEE Transactions on Network and Service Management*, vol. 20, no. 2, pp. 1446–1458, 2023.

[25] G. Lou, Y. Liu, T. Zhang, and X. Zheng, "Stfl: A spatial-temporal federated learning framework for graph neural networks," in *AAAI Conference on Artificial Intelligence Workshop on Deep Learning on Graphs: Methods and Applications*, 2022.

[26] X. Yuan, J. Chen, J. Yang, N. Zhang, T. Yang, T. Han, and A. Taherkordi, "Fedstn: Graph representation driven federated learning for edge computing enabled urban traffic flow prediction," *IEEE Transactions on Intelligent Transportation Systems*, 2022.

[27] L. Yang, W. Chen, X. He, S. Wei, Y. Xu, Z. Zhou, and Y. Tong, "Fedgtp: Exploiting inter-client spatial dependency in federated graph-based traffic prediction," in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024, pp. 6105–6116.

[28] N. Lu, J. Lu, G. Zhang, and R. L. De Mantaras, "A concept drift-tolerant case-base editing technique," *Artificial Intelligence*, vol. 230, pp. 108–133, 2016.

[29] D. Kifer, S. Ben-David, and J. Gehrke, "Detecting change in data streams," in *VLDB*, vol. 4. Toronto, Canada, 2004, pp. 180–191.

[30] F. Gu, G. Zhang, J. Lu, and C.-T. Lin, "Concept drift detection based on equal density estimation," in *2016 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2016, pp. 24–30.

[31] L. Bu, D. Zhao, and C. Alippi, "An incremental change detection test based on density difference estimation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 10, pp. 2714–2726, 2017.

[32] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," in *Advances in Artificial Intelligence–SBIA 2004: 17th Brazilian Symposium on Artificial Intelligence, Sao Luis, Maranhao, Brazil, September 29-Ocotber 1, 2004. Proceedings 17*. Springer, 2004, pp. 286–295.

[33] M. Baena-Garcıa, J. del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavalda, and R. Morales-Bueno, "Early drift detection method," in *Fourth international workshop on knowledge discovery from data streams*, vol. 6. Citeseer, 2006, pp. 77–86.

[34] K. Nishida and K. Yamauchi, "Detecting concept drift using statistical testing," in *Discovery science*, vol. 4755. Springer, 2007, pp. 264–269.

[35] 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.

[36] J. M. Joyce, "Kullback-leibler divergence," in *International encyclopedia of statistical science*. Springer, 2011, pp. 720–722.

[37] 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," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, Oct. 2014.

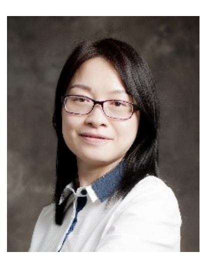
[38] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations, 2017*.
[39] H. Feng, Y. Shu, S. Wang, and M. Ma, "Svm-based models for predicting wlan traffic," in *2006 IEEE international conference on communications*, vol. 2. IEEE, 2006, pp. 597–602.
[40] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting," in *27th International Joint Conference on Artificial Intelligence, IJCAI 2018*, 2018, pp. 3634–3640.
[41] R. Jiang, Z. Wang, J. Yong, P. Jeph, Q. Chen, Y. Kobayashi, X. Song, S. Fukushima, and T. Suzumura, "Spatio-temporal meta-graph learning for traffic forecasting," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 37, no. 7, 2023, pp. 8078–8086.
[42] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz, "Pruning convolutional neural networks for resource efficient inference," in *5th International Conference on Learning Representations, ICLR 2017-Conference Track Proceedings*, 2019.
[43] M. Hobbhahn and J. Sevilla, "What's the backward-forward flop ratio for neural networks?" 2021, accessed: 2023-4-18. [Online]. Available: https://epochai.org/blog/backward-forward-FLOP-ratio

Qingxiang Liu is currently pursuing the Ph.D. degree with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. He has authored or coauthored several papers at IEEE Transactions on Intelligent Transportation Systems, Future Generation Computer Systems and so on. His current research interests include federated learning , time series forecasting and spatio-temporal mining.

Sheng Sun received her B.S. and Ph.D degrees in computer science from Beihang University, China, and the University of Chinese Academy of Sciences, China, respectively. She is currently an assistant professor at the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. Her current research interests include federated learning, mobile computing and edge intelligence.

Yuxuan Liang is currently a tenure-track Assistant Professor at Hong Kong University of Science and Technology (Guangzhou), working on the research, development, and innovation of spatio-temporal data mining and AI, with a broad range of applications in urban computing. Prior to that, he obtained his PhD degree at the School of Computing, National University of Singapore. He has published over 70 papers in refereed journals (e.g., TPAMI, AI, TKDE) and conferences (e.g., KDD, NeurIPS, ICML, ICLR, WWW). These publications have attracted 3,700 citations with h-index of 30. He has served as a program committee member for prestigious conferences, such as KDD, ICML, ICLR, NeurIPS, WWW, CVPR, and ICCV. He has served as organizer or co-chair of Workshop on Urban Computing (Urbcomp@KDD-23) and Workshop on AI for Time Series (AI4TS@IJCAI-24). He has received The 23rd China Patent Excellence Award and SDSC Dissertation Research Fellowship 2020.

Xiaolong Xu (Senior Member, IEEE) received the Ph.D. degree in computer science and technology from Nanjing University, China, in 2016. He is currently a Full Professor with the School of Software, Nanjing University of Information Science and Technology. He received the Best Paper Award from the IEEE CBD 2016, IEEE CyberTech2021, CENET2021, IEEE iThings2022 and EAI CloudComp2022, and the Outstanding Paper Award from IEEE SmartCity2021. He also received the Annual Best Paper Award from Elsevier JNCA. He has been selected as the Highly Cited Researcher of Clarivate 2021 and 2022. His research interests include edge intelligence and service computing.



Min Liu (Senior Member, IEEE) received her B.S. and M.S. degrees in computer science from Xi'an Jiaotong University, China, in 1999 and 2002, respectively. She got her Ph.D in computer science from the Graduate University of the Chinese Academy of Sciences in 2008. She is currently a professor at the Networking Technology Research Centre, Institute of Computing Technology, Chinese Academy of Sciences. Her current research interests include mobile computing and edge intelligence.

Muhammad Bilal (Senior Member, IEEE) received the Ph.D. degree in information and communication network engineering from the School of Electronics and Telecommunications Research Institute (ETRI), Korea University of Science and Technology, Daejeon, South Korea, in 2017. From 2018 to 2023, he was an Assistant Professor with the Division of Computer and Electronic Systems Engineering, Hankuk University of Foreign Studies, Yongin, South Korea. In 2023, he joined Lancaster University, Lancaster LA1 4YW, United Kingdom, where he is currently working as a Senior Lecturer with School of Computing and Communications. His research interests include design and analysis of network protocols, network architecture, network security, the IoT, named data networking, blockchain, cryptology, and future Internet.

Yuwei Wang received his Ph.D. degree in computer science from the University of Chinese Academy of Sciences, Beijing, China. He is currently a Senior Engineer (equivalent to Associate Professor) at the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. His current research interests include federated learning, mobile edge computing, and next-generation network architecture.

Xujing Li is a Ph.D. candidate with Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. His primary research focuses on Efficient Federated Learning and Distributed Learning Optimization. He received his bachelor's degree at Central South University, Changsha, China in 2019.

Yu Zheng (Fellow, IEEE) is the Vice President of JD.COM and head JD Intelligent Cities Research. Before Joining JD.COM, he was a senior research manager at Microsoft Research. He currently serves as the Editor-in-Chief of ACM Transactions on Intelligent Systems and Technology and has served as the program co-chair of ICDE 2014 (Industrial Track), CIKM 2017 (Industrial Track) and IJCAI 2019 (industrial track). He is also a keynote speaker of AAAI 2019, KDD 2019 Plenary Keynote Panel and IJCAI 2019 Industrial Days. His monograph, entitled Urban Computing, has been used as the first text book in this field. In 2013, he was named one of the Top Innovators under 35 by MIT Technology Review (TR35) and featured by Time Magazine for his research on urban computing. In 2016, Zheng was named an ACM Distinguished Scientist and elevated to an IEEE Fellow in 2020 for his contributions to spatio-temporal data mining and urban computing.