# Burst-Sensitive Traffic Forecast via Multi-Property Personalized Fusion in Federated Learning

Jingjing Xue, Sheng Sun, Min Liu, *Senior Member, IEEE*, Yuwei Wang, *Member, IEEE*, Xuying Meng, Jingyuan Wang, JunBo Zhang, *Senior Member, IEEE*, and Ke Xu, *Fellow, IEEE*

***Abstract*—For distributed network traffic prediction with data localization and privacy protection, Federated Learning (FL) enables collaborative training without raw data exchange across Base Stations (BSs). Nevertheless, traffic across BSs exhibit inherently heterogeneous trend burst and smooth fluctuation properties, but existing FL methods model single-scale series from only one view, which cannot simultaneously capture diverse trend and fluctuation properties, especially distinct burst distributions. In this paper, we propose *Personalized Federated Forecasting with Multi-property Self-fusion (P2FMS)*, which can represent multi-scale traffic properties from different views. With precise multi-property representations, a fusion-level prediction decision is learned for each client in a personalized manner to promptly sense traffic bursts and improve forecasting performance in non-IID settings. Specifically, P2FMS decomposes the traffic series into distinct time scales, based on which, we effectively extract closeness, period, and trend properties from different views. The closeness and period are embedded through global-view representations with spatial correlations, while non-stationary trends are individually fitted from the client-side view. Furthermore, a personalized combiner is designed to accurately quantify the proportion of general fluctuation raws (i.e., closeness and period) and specific trend property in predictions, which enables multi-property self-fusion for each client to accommodate heterogeneous traffic patterns and enhance prediction accuracy. Besides, an alternant training mechanism is introduced to optimize property representation and fusion modules with the convergence guarantee. Extensive experiments on real-world datasets show that P2FMS outperforms status quo methods in both prediction performance and convergence time.**

***Index Terms*—Network traffic forecast, federated learning (FL), heterogeneous data, multi-scale representation, personalized fusion.**

Received 9 May 2024; revised 5 December 2024; accepted 21 January 2025. Date of publication 5 February 2025; date of current version 5 June 2025. This work was supported in part by the National Key Research and Development Program of China under Grant 2021YFB2900102, in part by the National Natural Science Foundation of China under Grant 62472410, Grant 62072436, Grant 72222022, Grant 72171013, and Grant 7224210, and in part by the National Science Fund for Distinguished Young Scholars of China under Grant 62425201. Recommended for acceptance by J. Ko. *(Corresponding author: Min Liu.)*

Jingjing Xue is with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100045, China, and also with the University of Chinese Academy of Sciences, Beijing 101408, China (e-mail: xuejingjing20g@ict.ac.cn).

Sheng Sun, Yuwei Wang, and Xuying Meng are with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100045, China (e-mail: sunsheng@ict.ac.cn; ywwang@ict.ac.cn; mengxuying@ict.ac.cn).

Min Liu is with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100045, China, also with the University of Chinese Academy of Sciences, Beijing 101408, China, and also with Zhongguancun Lab, Beijing 100190, China (e-mail: liumin@ict.ac.cn).

Jingyuan Wang is with the School of Computer Science and Engineering, Beihang University, Beijing 100191, China, and also with the MIIT Key Laboratory of Data Intelligence and Management, Beihang University, Beijing 100191, China (e-mail: jywang@buaa.edu.cn).

JunBo Zhang is with the JD iCity, JD Intelligent Cities Research, JD Technology, Beijing 101111, China (e-mail: msjunbozhang@outlook.com).

Ke Xu is with the Department of Computer Science and Technology, Tsinghua University, Beijing 100190, China, and also with the Zhongguancun Laboratory, Beijing 100081, China (e-mail: xuke@tsinghua.edu.cn).

Digital Object Identifier 10.1109/TMC.2025.3538871

## I. Introduction

With the worldwide deployment of the fifth generation (5G) wireless communication networks and the proliferation of mobile devices (e.g., IoT devices and smartphones), modern network management is challenged not only by the considerable growth of traffic generated in edge networks [1], [2], but also by highly complex and dynamic traffic patterns [3]. Given these challenges, network traffic prediction plays a crucial role in mitigating network congestion and ensuring the Quality of Service (QoS). Precise traffic forecasting can proactively perceive traffic changes to optimize network operations, through which edge networks can realize efficient resource allocation and dynamic edge computing server deployment, further achieving intelligent self-management [4], [5], [6].

Most existing works on network traffic forecasting [7], [8], [9] adopt centralized architectures, which collect a vast amount of traffic data and position information to a server for model training. Direct data transfer is notoriously difficult due to limited network capacity and privacy concerns. On one hand, frequent data uploading quickly exhausts network capacity and negatively affects payload transmissions [10]. On the other hand, it also suffers from data access restrictions because traffic data are accessible for specific institutions, and these institutions are unwilling to share practical traffic with privacy concerns [11], [12]. In view of these issues, it is essential to develop distributed traffic prediction in edge networks.

As a distributed training paradigm, Federated Learning (FL) [13], [14] is extended to traffic forecasting for jointly training models across BSs (i.e., clients) without raw data transmission [10], [15], [16]. Despite the advantages in data localization and privacy protection, existing FL forecasting studies focus on learning a shared model [10], [17], [18] for all clients, which extracts general features from Independent and Identically Distributed (IID) data, failing to tackle the personalization challenge posed by heterogeneous traffic patterns. As shown in Fig. 1, the traffic series of different BSs exhibit diverse trend changes (e.g., surge or slumps) and periodic fluctuation laws [19], [20]. Here, we define a sudden rapid growth or drop in traffic as a

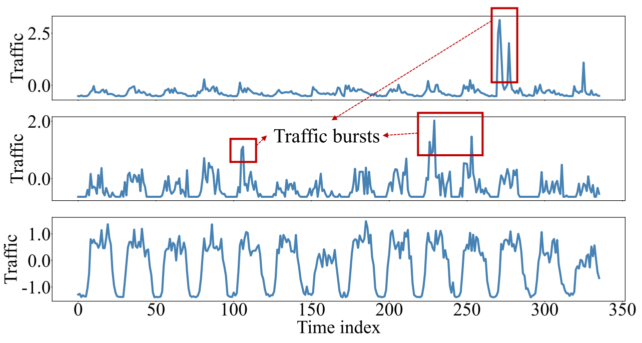
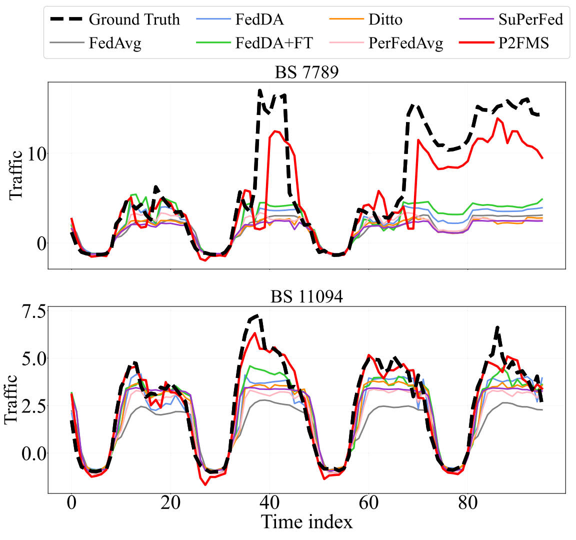


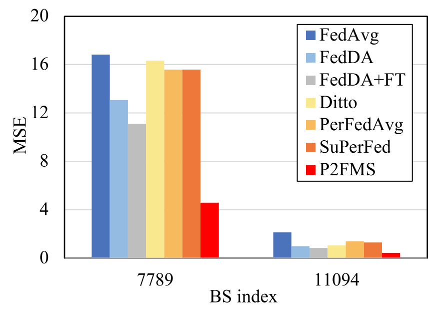
Fig. 1. Traffic data of different BSs in Trentino.

burst [21], [22]. In practical network management scenarios, sudden rapid growth is of great significance for congestion control and resource scheduling. We frame several bursts of sampled traffic series in Fig. 1, which are obviously higher than the maximum value of the previous time slots in the windows. We observe that traffic bursts occur at distinct time intervals on different BSs. These heterogeneous trend properties cause inconsistent local updates across clients, resulting in performance degradation of the shared model [23], [24], [25]. Empirically, we evaluate LSTM forecasting models in classical FedAvg [13] and FedDA [10] on Trentino traffic data [26]. As shown in Fig. 2(a), the shared deep learning models trained by FedAvg and FedDA only predict periodic non-linear fluctuations, but fail to capture diverse bursts, leading to larger forecasting errors (see Fig. 2(b)). An intuitive reason is that client-side unique burst features are averaged out by the global aggregation in FedAvg and FedDA. If we fine-tune the shared model of FedDA (i.e., FedDA+FT) on client-specific traffic data, the local burst forecasting performance can be slightly improved (as shown in Fig. 2(a)), but it is still not accurate enough.

Several pioneering studies explore personalized strategies in FL, such as meta-learning [27], [28], regularization [29], [30], and model mixing [31], [32], which customize an individual model for each client to counter heterogeneous data. They concentrate on general tasks (including natural language processing [33] and computer vision [30]), which extract features from text or images. Unlike text and image data, traffic series inherently have unique temporal properties, including smooth fluctuations (i.e., closeness and periods) and non-stationary trends (e.g., bursts). It has been verified that these fluctuation and trend properties are extremely valuable for effective forecasting [34], [35], [36]. Simply applications of these general Personalized FL (PFL) methods cannot precisely extract various properties of network traffic. As shown in Fig. 2, advanced PFL approaches PerFedAvg [28], Ditto [30] and SuPerFed [32] only depict smooth fluctuations but overlook heterogeneous traffic bursts, bringing larger forecasting errors. Since current PFL works only model traffic sequences at a single time scale, while various temporal properties are reflected in different time scales such that single-scale PFL models are challenging to simultaneously capture the short-term closeness and historically



(a) Ground truth and forecasting curves for two random BSs.



(b) MSE forecasting results.

Fig. 2. Prediction result comparisons of different FL methods.

occasional burst features at the long scale [34], failing to provide accurate traffic forecasting.

In light of the above observations, we propose Personalized Federated Forecasting with Multi-property Self-fusion (P2FMS) to precisely extract multi-scale temporal properties of local data from different views and individually integrate multiple property representations for each client, which achieves strong sensitivity to traffic bursts and significant performance gains in real non-IID scenarios. For accurate representation of fluctuation and trend properties, traffic data is decomposed into distinct time scales, and then two major components (shared fluctuation extractor and local trend catcher) are designed to model multi-scale input from different views. The first one embeds non-linear fluctuation properties correlated with spatial information from the server-side (i.e., global) view. Another individually captures unique trend property from the client-side (i.e., local) view. In view of distinct burst distributions across different clients, it is necessary to introduce individual trend fitting and personalized property fusion from the local view. With global-view fluctuation and local-view trend representations, we develop a personalized combiner to adaptively measure the occurrence probability of each property in a learnable manner. In this way, each client realizes heterogeneous multi-property self-fusion, tackling personalization demands for precise traffic forecasting in edge networks with non-IID data.

In summary, our main contributions are as follows:

- We propose Personalized Federated Forecasting with Multi-property Self-fusion (P2FMS) over edge network traffic in non-IID settings, which is the first PFL forecasting framework with multi-scale property representations, to the best of our knowledge.
- To accommodate heterogeneous traffic patterns, we develop a personalized fusion of multiple properties in a learnable manner, which accurately weighs the impact of global fluctuation laws and local trend changes on predictions for each client to boost forecasting performance.
- We introduce an alternant training mechanism to optimize the property representation and fusion modules of P2FMS, which is theoretically analyzed to reach the exponential convergence rate.
- We conduct extensive experiments on real-world datasets, which demonstrate that P2FMS achieves 2.37%–47.51% MSE performance gains and reduces 19.5%–82.6% convergence time compared to the state-of-the-art methods.

## II. Related Work

This work derives elements from network traffic forecasting and personalized FL frameworks. Here, we review some of the most relevant to us from the two research areas.

### A. Network Traffic Forecasting

To optimize network operations and realize intelligent management, traffic forecasting flexibly models the dynamics and proactively perceives traffic changes, which has aroused much attention in recent years. A variety of prediction techniques have been explored to improve traffic forecasting performance, which can be roughly categorized into two classifications: statistical methods and Deep Learning (DL) methods.

Statistical methods model traffic series with statistics theory and fit historical traffic data to get prediction functions. For example, AutoRegressive Integrated Moving Average (ARIMA) [37] and its variants [38] build mathematical models to characterize the linear changes of time series. Prior work [39] decomposes complicated traffic data into simple subcomponents, where each subcomponent is predicted individually, and the final result is the combination of all subcomponent predictions. Similar in spirit to series decomposition, Exponential Smoothing (ES) [40], [41] captures trends and seasonality of traffic data separately and then integrates them additively or multiplicatively as predictions, which exhibits a powerful ability to sense trends in certain settings. Although these statistical methods are highly interpretable and computation-efficient, they are too simple to capture complex non-linear variations, performing poorly for complicated traffic series.

Deep Learning (DL) methods learn forecasting models in a data-driven way, relying on the powerful non-linear expression capabilities of Deep Neural Networks (DNNs) to model non-linear fluctuations. Available solutions include Long Short-Term Memory (LSTM) [42], Convolutional Neural Networks (CNNs) [43], and other sophisticated DNNs [44]. These works depend on a single model structure. Recently, the combination of different models is explored to improve performance [45]. MVSTGN [9] integrates attention and convolution mechanisms into traffic representations. [46] and [47] learn multiple forecasting models in a serial or parallel manner and integrate the strengths of these models. In addition, [48] manages a set of prediction models based on the alternative feedback architecture and meta-learning scheme to dynamically select the best model according to recent prediction performance.

However, these DL methods focus on extracting general features from IID data and adopt centralized architectures, in which data collection causes privacy concerns and access permission issues. Several pioneering studies [10], [17], [18] explore FL frameworks for traffic prediction, which allow clients to collaboratively a shared model without raw data transmission. Unfortunately, traffic series across BSs are generally heterogeneous in practical scenarios, leading to performance degradation of the shared model. Until now, none of the federated prediction approaches for network traffic tackled the personalization challenge posed by heterogeneous traffic patterns across clients [17].

### B. Personalized Federated Learning

Non-IID data is one of the most significant challenges of FL [49], [50], which leads to apparent inconsistencies of local updates and further degrades global model performance [51], [52]. To mitigate the non-IID problem, many prior works have been devoted to FL personalization [53] in image or language datasets, which can be roughly divided into six directions: local Fine-Tuning (FT), meta-learning, Multi-Task Learning (MTL), model mixing, clustered FL, and model regularization.

Local FT [54] propose to retrain all or part parameters of the global model on local data before inference. Meta-learning [55] is known as a popular setting for fast model adaptation, which is beneficial for local model personalization. Recent researches [27], [28] combine meta-learning with FL to quickly adapt the shared model to local data [56]. Multi-Task Learning (MTL) [57] treats each client as a task and captures relationships between different tasks from non-IID data. In this way, we can customize an individual model for each task.

Moreover, a popular model mixing strategy is to mix global and local models [24], [25], [31], [32], [58], [59], where each client learns a personalized model through the combination of global and local models. The weights/proportions of this combination control the degree of personalization to provide a tradeoff between global generalization and local personalization. An extension of the mixing idea is explored in [60], where the personalized model is updated by the combination of global and local gradients. Besides, FedFomo [61] mixes multiple local models with similar targeted data distributions to learn a customized model for each client. For clustered FL with personalization, the main idea is to partition clients with similar data distributions into a cluster, and then each cluster can tailor an individual model [23] to fit the unique data distribution. Model regularization [29], [30] adds the regularization terms to constrain the gap between local models and their average (or the global model), which is beneficial for producing robust personalized models.

The above PFL methods focus on computer vision and natural language processing tasks with single-scale data modeling, which cannot simultaneously extract fluctuation and trend properties of traffic data at different time scales, resulting in poor forecasting performance in non-IID settings. To this end, we granularly consider heterogeneous distributions of multiple properties in traffic data across clients. On this basis, P2FMS is proposed to effectively represent multi-scale fluctuation and trend properties from different views, and further accurately quantify the proportion of these property representations in future predictions on each client, aiming at proactive burst perception and heterogeneous pattern accommodation.

## III. Background and Problem

Given $K$ BSs that form a client set $\mathcal{K}$, each client $k \in \mathcal{K}$ owns its traffic data $D^k = \{d_1^k, d_2^k, \ldots, d_T^k\}$ with a total of $T$ time slots, where $d_t^k$ is the traffic volume of client $k$ at time slot $t$. We focus on one-step-ahead forecasting, i.e., predicting traffic at the next time slot based on the traffic from the previous $\tau$ slots. The traffic forecasting task of client $k$ is defined on the training dataset $\boldsymbol{X}^k = \{\boldsymbol{x}_{t-\tau+1:t}^k \,|\, t = \tau, \ldots, T-1\}$, where

$$\boldsymbol{x}_{t-\tau+1:t}^k = [d_{t-\tau+1}^k, d_{t-\tau+2}^k, \ldots, d_t^k]. \tag{1}$$

For each client $k \in \mathcal{K}$, the forecasting model $f_k : \mathbb{R}^\tau \to \mathbb{R}$ maps the input sequence $\boldsymbol{x}_{t-\tau+1:t}^k \in \mathbb{R}^\tau$ to the prediction value $\hat{y}_{t+1}^k \in \mathbb{R}$, such that

$$\hat{y}_{t+1}^k = f_k\left(\boldsymbol{x}_{t-\tau+1:t}^k\,;\,\Phi^k\right), \tag{2}$$

where $\Phi^k$ denotes the parameters of the forecasting model $f_k$. The single-scale modeling used in conventional methods directly inputs local sequences with fixed and continuous $\tau$ time slots to a DNN model, which cannot simultaneously capture multi-scale closeness, period, and trend properties of client-side traffic data.

FL enables a group of distributed BSs (i.e., clients) in edge networks to collaboratively learn forecasting models without privacy-sensitive data exchange. Each client parallelly trains the local model on its own data and periodically uploads local updates to the server for global aggregation. The optimization objective of FL with $K$ clients can be expressed as:

$$\min_{\Phi^1,\ldots,\Phi^K \in \mathcal{F}} \frac{1}{K} \sum_{k=1}^{K} \mathcal{L}_k\left(\Phi^k\right), \tag{3}$$

where $\mathcal{F}$ denotes the feasible parameter space of forecasting models, and $\mathcal{L}_k$ is the loss function of client $k$. Conventional FL frameworks [10], [13] train a single shared model, i.e., $\Phi = \Phi^1 = \ldots = \Phi^K$, to minimize the average loss $(1/K)\sum_{k=1}^{K} \mathcal{L}_k(\Phi)$ over all clients. However, the traffic data across BSs are generally non-IID in practical applications, which degrades the performance of the shared model. To accommodate heterogeneous traffic distributions, we focus on searching for a personalized model with parameters $\Phi^k$ for each client $k$.

## IV. The P2FMS Framework

In this section, we first provide an overview of the P2FMS framework and then explain the core mechanisms of P2FMS in detail. P2FMS involves two key modules: *(1) multi-scale property representations* from different views and *(2) personalized fusion of multiple properties* for each client to effectively perceive trend bursts and adaptively learn heterogeneous distributions of various temporal properties.

### A. Framework Overview

Essentially, a complicated traffic series exhibits multiple temporal properties, including closeness dependence, regular periods, and non-stationary trends [34], [39], [62]. To simultaneously capture these properties of traffic series and cope with non-IID data across distributed clients, P2FMS allows each client to learn a customized model with a shared fluctuation extractor, a local trend catcher, and a personalized combiner, as shown in Fig. 3. Concretely, two major components, shared fluctuation extractor and local trend catcher, are designed to extract various property features from different views based on multi-scale traffic sequence inputs. From the global view, the shared fluctuation extractor models non-linear fluctuation laws (including closeness and periods) correlated with spatial information across clients, where we utilize the powerful non-linear expression capabilities of DNN models. From the local view, a local trend catcher should be sensitive to trend changes and adapt to the limited computation capabilities of edge devices. Statistical ES methods have demonstrated strong sensitivity for trend bursts [41], [63]. Besides, it is known that statistical ES methods are low-cost and have low computational complexity [40], which can effectively accommodate restricted computing resources. Following Occam's Razor [64], we introduce a simple statistical model as the local trend catcher to achieve trend fitting and improve learning efficiency. Furthermore, in view of heterogeneous traffic patterns across clients, we develop a personalized combiner to adaptively fuse fluctuation and trend properties into future predictions for each client.

On these basis, the client-side update would alternately perform two steps: *(i)* local personalized training for the multi-property combiner based on the statistically fitted trend catcher and *(ii)* shared module updating for the fluctuation extractor. In each training round $r$, the procedures of P2FMS are summarized as follows:

- Step 1. *Parameters sending:* The server randomly selects the client set $\mathcal{C}_r$ and sends the global parameters $\omega_{r-1}$ of the shared fluctuation extractor to each client $k \in \mathcal{C}_r$.
- Step 2. *Local personalized training:* Each client $k \in \mathcal{C}_r$ decomposes traffic data into different scales, freezes the received global parameters, and directly fits the local trend catcher by statistical calculations. Then, only the personalized combiner is trained for $E_l$ iterations based on local traffic data.
- Step 3. *Shared module updating:* The locally trained parameters of the personalized combiner are frozen. Client $k \in \mathcal{C}_r$ only updates the shared fluctuation extractor for $E$ iterations.

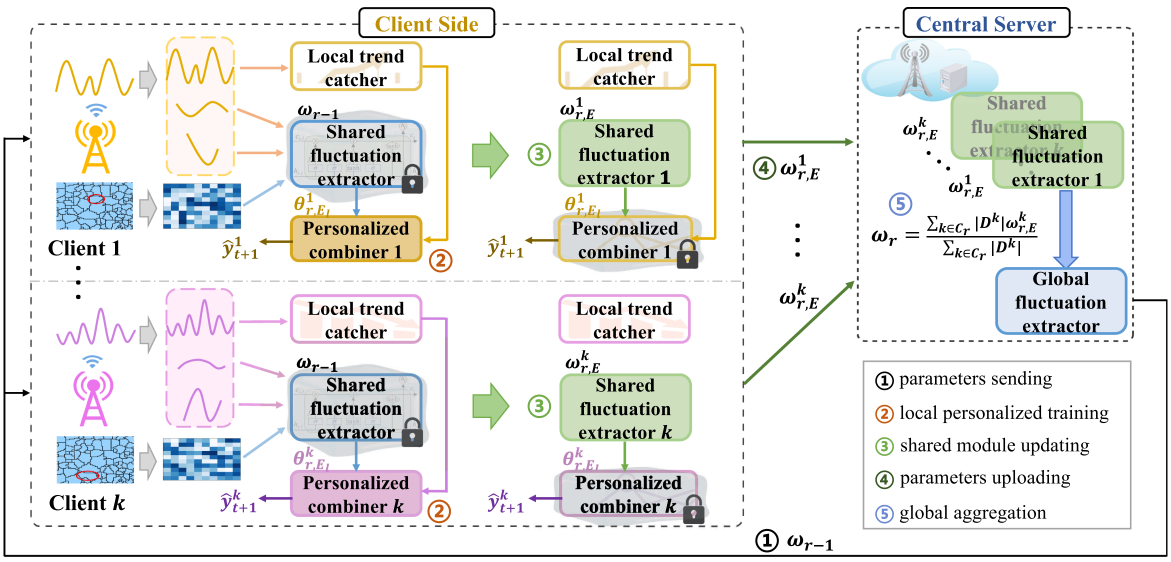


Fig. 3. The overview diagram of P2FMS framework. The numbers ①-⑤ represent the training procedures.

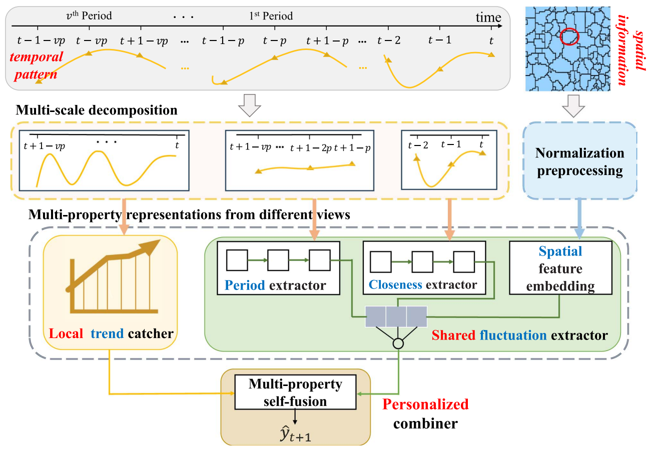
- Step 4. *Parameters uploading:* Each client $k \in \mathcal{C}_r$ only uploads the parameters of the shared fluctuation extractor to the server.
- Step 5. *Global aggregation:* The server aggregates all received parameters to calculate new global parameters $\omega_r$ for the shared fluctuation extractor.

The above steps are iteratively executed for $R$ rounds. In the following, we first elaborate on multi-scale property representations by the shared fluctuation extractor and local trend catcher (detailed in Section IV-B) as well as the personalized fusion of multiple properties (detailed in Section IV-C). Then, the alternant training mechanism for representation and fusion modules is described in Section IV-D.

## B. Multi-Scale Property Representations

Multi-scale property representations involve two major components from different views: shared fluctuation extractor and local trend catcher. Closeness dependence and regular periods are regarded as non-linear fluctuation properties, which involve different time scale inputs and are associated with the position distributions of BSs [34]. Hence, a shared fluctuation extractor integrates multi-scale temporal data with spatial information to learn a temporal-spatial fluctuation representation from the global view. While non-stationary trend property can reflect traffic changes, and a sharp uptrend indicates an impending traffic burst. Distributed BSs exhibit heterogeneous traffic bursts at distinct time intervals, but current personalized strategies - local FT, meta-learning, regularization, etc., still cannot effectively retain the unique knowledge of historical traffic bursts at the long time scale, leading to poor performance. Thus, we fit a trend catcher from the local view to improve the burst sensitivity and enhance forecasting accuracy.

*Fluctuation Property Representation by the Shared Fluctuation Extractor:* Closeness dependence and regular periods involve different-scale temporal attention, where closeness



$v^{th}$ Period
$1^{st}$ Period
time
temporal pattern
spatial information
Multi-scale decomposition
Normalization preprocessing
Multi-property representations from different views
Local trend catcher
Period extractor
Closeness extractor
Spatial feature embedding
Shared fluctuation extractor
Multi-property self-fusion
$\hat{y}_{t+1}$
Personalized combiner

Fig. 4. The structure of client-side customized model.

dependence focuses on short-term time slots, while the modeling of regular periods requires going back to several long-term periods. Intuitively, traffic fluctuation laws are related to the geographic locations of clients, where adjacent regions generally involve the same function and thus have similar fluctuation patterns [8]. Integrating position information into traffic forecasting can capture spatial correlations between different BSs to facilitate precise prediction. In P2FMS, we decompose the traffic series into different time scales and construct a shared fluctuation extractor composed of the closeness extractor $f^c$, period extractor $f^p$, and spatial feature representation module $f^s$ (as shown in Fig. 4) to model multi-scale temporal sequences and spatial information. Furthermore, a Fully Connected (FC) layer $f^\ell$ is used to integrate multi-scale temporal patterns with spatial correlations and further obtain the joint fluctuation representation.

From a short-term time scale, the input of the closeness extractor is $\boldsymbol{x}^{k,c} = [d_{t-n+1}^k, d_{t-n+2}^k, \ldots, d_t^k]$, where $n$ denotes the window size of closeness dependence. From a periodic time scale, the traffic volumes sampled from several period histories can be denoted as

$$\boldsymbol{x}^{k,p} = [d^k_{t-vp+1}, d^k_{t-(v-1)p+1}, \ldots, d^k_{t-p+1}],$$

which is the input sequence of the period extractor with preset period $p$. Here, the period-dependent window size is set to $v$. We learn the closeness extractor $f^c$ and period extractor $f^p$ to represent the temporal patterns, where the input sequences are mapped by

$$\hat{\boldsymbol{y}}^{k,c}_{t+1} = f^c\left(\boldsymbol{x}^{k,c}; \omega_c\right), \quad \hat{\boldsymbol{y}}^{k,p}_{t+1} = f^p\left(\boldsymbol{x}^{k,p}; \omega_p\right). \tag{4}$$

Here, $\omega_c$ and $\omega_p$ are the parameters of the closeness extractor $f^c$ and period extractor $f^p$, respectively.

The spatial feature representation module $f^s$ is used to characterize the geographical information of clients. It is well known that geographically close clients generally show similar fluctuation patterns [1], [65]. In this way, geographical features are able to assist fluctuation forecasting. For an arbitrary client $k$, its location information (including longitude and latitude) is normalized to $\boldsymbol{g}_k$ and then embedded by $\boldsymbol{x}^k_s = f^s(g_k; \omega_s)$, where $\omega_s$ denotes the parameters of $f^s$. Then, we concatenate the spatial representation vector $\boldsymbol{x}^k_s$ with the outputs of the closeness extractor and period extractor. Finally, a FC layer $f^\ell$ with parameters $\omega_\ell$ is adopted to generate the joint fluctuation representation result $\hat{y}^{k,g}_{t+1}$. For each client $k$, the fluctuation extractor is regarded as the global shared module $f^G$, calculated by

$$\begin{aligned}\hat{y}^{k,g}_{t+1} &= f^G\left(\boldsymbol{x}^k_{t-\tau+1:t}, \boldsymbol{g}_k; \omega\right)\\ &= f^\ell\left(f^c\left(\boldsymbol{x}^{k,c}; \omega_c\right), f^p\left(\boldsymbol{x}^{k,p}; \omega_p\right),\right.\\ &\qquad \left. f^s(\boldsymbol{g}_k; \omega_s);\ \omega_\ell\right),\end{aligned} \tag{5}$$

where the shared parameters $\omega = \{\omega_c, \omega_p\, \omega_s, \omega_\ell\}$, and $\boldsymbol{x}^{k,c}$, $\boldsymbol{x}^{k,p}$ are the subsequences of $\boldsymbol{x}^k_{t-\tau+1:t}$ with $\tau = v \times p$, which are decomposed from different time scales.

*Trend Property Representation by the Local Trend Catcher:* Traffic data in each client involves unique non-stationary trends, and strong upward trends may bring traffic bursts, which are notoriously challenging to capture in existing FL and PFL frameworks [10], [30], [32]. In this work, we introduce a local trend catcher to model the non-stationary trends of client-side traffic series. Each client $k$ fits a local trend catcher $f^e_k$ by the statistical calculation of Exponential Smoothing (ES) [41], [63], [66]. The core idea of ES is that the prediction is the weighted averages of past data $\boldsymbol{x}^k_{t-\tau+1:t}$, and the weights decrease exponentially as the time gap increases. In order to depict trend changes, we exploit ES with the damped trend [40], in which the forecasting equation consists of the level term and the trend term. Given the input $\boldsymbol{x}^{k,e} = \boldsymbol{x}^k_{t-\tau+1:t} = [d^k_{t-\tau+1}, d^k_{t-\tau+2}, \ldots, d^k_t]$ with $\tau = v \times p$, the local trend catcher fits $f^e_k : \mathbb{R}^\tau \to \mathbb{R}$ to calculate the prediction $\hat{y}^{k,e}_{t+1}$ through

$$\hat{y}^{k,e}_{t+1} = f^e_k\left(\boldsymbol{x}^{k,e}\right) = h_t + \phi m_t, \tag{6}$$

where $h_t$ denotes the level term computed by $h_t = \gamma d^k_t + (1-\gamma)(h_{t-1} + \phi m_{t-1})$. Here, $0 < \gamma < 1$ is the smoothing parameter for the level term. $\phi m_t$ is the damped trend term and $m_t = \rho(h_t - h_{t-1}) + (1-\rho)\phi m_{t-1}$, where $0 < \rho, \phi < 1$ are smoothing and damping parameters for the trend, respectively. Before local personalized training, the smooth parameters $\gamma, \rho$, and damping parameter $\phi$ are preset. Note that, the local trend catcher can be directly fitted through the statistical calculations on client-side traffic data with lower costs, which does not need to be trained and transmitted.

## C. Personalized Multi-Property Fusion

Multiple properties exhibit a unique distribution in the traffic series of each client, which poses a challenge for the customized fusion of multi-property representations. To tackle this challenge, we develop a personalized combiner on each client for self-weighting the impact of joint fluctuation properties and non-stationary trend property on future predictions, which adaptively integrates the representation results of two components (i.e., shared fluctuation extractor and local trend catcher) in a learnable manner.

For each client $k$, the output result $\hat{y}^k_{t+1}$ of the personalized combiner $f^L_k$ can be viewed as the weighted fusion of the representation $\hat{y}^{k,g}_{t+1}$ on the shared fluctuation extractor $f^G$ and the prediction $\hat{y}^{k,e}_{t+1}$ on the local trend catcher $f^e_k$ such that

$$\hat{y}^k_{t+1} = f^L_k\left(\hat{y}^{k,g}_{t+1},\ \hat{y}^{k,e}_{t+1};\ \theta^k\right), \tag{7}$$

where $\hat{y}^{k,g}_{t+1}$ is obtained by (5), and $\hat{y}^{k,e}_{t+1}$ is computed by (6). Besides, $\theta^k$ denotes the parameter set of the personalized combiner $f^L_k$ in client $k$. More importantly, considering the heterogeneous patterns of fluctuation and trend properties in traffic data across different clients, the personalized combiner $f^L_k$ is only visible for client $k$, meaning that parameters $\theta^k$ of $f^L_k$ are not transmitted to the server.

To sum up, the ensemble of the shared fluctuation extractor and local trend catcher simultaneously captures non-linear fluctuation properties and non-stationary trend property at multiple time scales from different views, which is able to sensitively sense traffic bursts and identify complex traffic patterns in edge networks. The shared fluctuation extractor can be trained with the full quantity of temporal data and spatial information of all clients, while the local trend catcher must be series-wise (or client-wise). While the personalized combiner can not only merge information extracted from the dataset level involving the traffic data of all clients and the single series level, but also uniquely depict the proportion of fluctuation and trend properties in future targets for each client, thereby realizing heterogeneous multi-property self-fusion in each client and coping with personalization demands in real non-IID scenarios.

## D. Client-Side Alternant Training Mechanism

In the client-side model shown in Fig. 4, the fluctuation extractor is regarded as the global shared module denoted by $f^G(\cdot\,; \omega)$. On the contrary, the local trend extractor fits an individual function $f^e_k(\cdot)$ for each client, which can be viewed as a personalized structure. Moreover, the personalized combiner is also a client-specific personalization module $f_k^L(\cdot;\theta^k)$. In this way, a local customized model $f_k(\cdot;\Phi^k)$ is expressed as

$$f_k\left(\boldsymbol{x}^k_{t-\tau+1:t};\Phi^k\right) = f_k^L\left(f^G(\boldsymbol{x}^k_{t-\tau+1:t};\omega),\right.$$
$$\left. f_k^e\left(\boldsymbol{x}^k_{t-\tau+1:t}\right);\theta^k\right),$$

and the global objective of P2FMS can be rewritten as

$$\min_{\omega\in\Omega}\frac{1}{K}\sum_{k=1}^{K}\min_{\theta^k\in\Theta}\mathcal{L}_k\left(\omega,\theta^k\right), \tag{8}$$

where $\Omega$ denotes the parameter space of all feasible shared fluctuation extractors and $\Theta$ is the parameter space of all feasible personalized combiners. Each client $k$ aims to learn the optimal customized model $f_k^*(\cdot;\Phi_*^k)$ with optimal parameters $\Phi_*^k=\{\omega^*,\theta_*^k\}$ to approximate the prediction function, so as to achieve precise traffic forecasting.

P2FMS solves the optimization problem (8) by leveraging distributed traffic data and computing power across all clients. Theoretically, the optimization problem (8) is of min-min type with two groups of variables, i.e., $\omega$ and $\theta^k$. Let us introduce the inner function

$$h_k(\omega)=\min_{\theta^k\ \in\Theta}\mathcal{L}_k(\omega,\theta^k) \tag{9}$$

and rewrite (8) as

$$\min_{\omega\in\Omega}\frac{1}{K}\sum_{k=1}^{K}h_k(\omega). \tag{10}$$

The iterative optimization for (10) needs to solve the inner function (9) in each iteration [67], [68], while the local error of (9) would lead to the inaccurate oracle of (10), which requires alternant optimization of both. Therefore, the client should first freeze $\omega$ and learn $\theta^k$ for the optimization of the inner function (9), which is carried out in the step of local personalized training. After that, the client freezes $\theta^k$ and optimizes $\omega$ to solve (10) in the shared module updating.

The specific alternant training mechanism is as follows. In round $r$, the server $\mathcal{S}$ randomly selects the client set $\mathcal{C}_r$ and sends current global parameters $\omega_{r-1}$ to each client $k\in\mathcal{C}_r$. After receiving $\omega_{r-1}$, client $k\in\mathcal{C}_r$ imports $\omega_{r-1}$ into the shared fluctuation extractor and enters the local personalized training step. In this step, the shared fluctuation extractor is frozen, while the local trend catcher of client $k$ can be directly fitted through (6) without training. Based on fine-grained multi-property representations, client $k$ individually trains the personalized combiner on its local traffic data for $E_l$ iterations. For each iteration $l=1,\ldots,E_l$, client $k$ updates the parameters $\theta^k_{r,l-1}$ of its personalized combiner as follows:

$$\theta^k_{r,l}=\mathbf{SGD}\left(\mathcal{L}_k\left(\omega_{r-1},\theta^k_{r,l-1}\right),\theta^k_{r,l-1},\alpha_l\right), \tag{11}$$

where $\mathbf{SGD}(\mathcal{L},\theta,\alpha)$ denotes an update of $\theta$ based on the gradient of the loss function $\mathcal{L}$ on $\theta$ and the learning rate $\alpha$. After that, client $k$ obtains its customized model $f_{k,r}(\cdot;\Phi_r^k)$ with parameters $\Phi_r^k=\{\omega_{r-1},\theta^k_{r,E_l}\}$ in round $r$.

**Algorithm 1:** P2FMS.

```
Input: client selection fraction $\epsilon$, the number of iterations for
    local personalized training $E_l$,
    the number of iterations for shared module updating $E$,
    the number of rounds $R$
Initialize: global parameters $\omega_0$,
    local saved parameters $\theta_s^1,\ldots,\theta_s^K$,
    number of selected clients $c\leftarrow\max(\epsilon\cdot K,\,1)$
Aggregate:                          // Run on the server
 1: for each round $r=1,2,\ldots,R$ do
 2:    $\mathcal{C}_r\leftarrow$ (random set of $c$ clients)
 3:    Send global parameters $\omega_{r-1}$ to client $k\in\mathcal{C}_r$
 4:    for each client $k\in\mathcal{C}_r$ in parallel do
 5:       $\omega^k_{r,E}\leftarrow$ ClientUpdate($\omega_{r-1}$)
 6:    end for
 7:    $\omega_r=\frac{\sum_{k\in\mathcal{C}_r}|D^k|\omega^k_{r,E}}{\sum_{k\in\mathcal{C}_r}|D^k|}$          // global aggregation
 8: end for

ClientUpdate ($\omega_{r-1}$):              // Done by clients
 9: Initialize $\theta^k_{r,0}\leftarrow\theta^k_s$ and $\omega^k_{r,0}\leftarrow\omega_{r-1}$
10: Fit the local trend catcher
11: for each iteration $l=1,\ldots,E_l$ do
12:    Learn $\theta^k_{r,l}$ by (11) // local personalized training
13: end for
14: Save $\theta^k_s\leftarrow\theta^k_{r,E_l}$
15: for each iteration $\ell=1,\ldots,E$ do
16:    Update $\omega^k_{r,\ell}$ by (12) // shared module updating
17: end for
18: return $\omega^k_{r,E}$ to the server $\mathcal{S}$
```

Subsequently, the client-side training enters the step of shared module updating, where client $k\in\mathcal{C}_r$ freezes the trained personalized combiner with parameters $\theta^k_{r,E_l}$ and updates the shared fluctuation extractor for $E$ iterations such that for $\ell=1,\ldots,E$,

$$\omega^k_{r,\ell}=\mathbf{SGD}\left(\mathcal{L}_k\left(\omega^k_{r,\ell-1},\theta^k_{r,E_l}\right),\omega^k_{r,\ell-1},\eta_\ell\right), \tag{12}$$

where $\eta_\ell$ is the learning rate of the $\ell$-th iteration. Once the shared module updating accomplishing, the selected client $k\in\mathcal{C}_r$ uploads the shared parameters $\omega^k_{r,E}$ to the server $\mathcal{S}$.

When all clients in $\mathcal{C}_r$ complete the transmission of shared parameters, the server $\mathcal{S}$ performs the average aggregation to obtain new global parameters:

$$\omega_r=\frac{\sum_{k\in\mathcal{C}_r}|D^k|\omega^k_{r,E}}{\sum_{k\in\mathcal{C}_r}|D^k|}, \tag{13}$$

where $|D^k|$ denotes the number of training samples in client $k$ and $\omega_r$ represents the global parameters in round $r$. The whole process is summarized in Algorithm 1.

## V. Further Analysis

### A. Theoretical Analysis

In this section, we theoretically prove the convergence of P2FMS under the mini-batch Stochastic Gradient Descent (SGD) optimization. For each batch of training data $\mathbf{x}^k$, SGD calculates $\nabla q_k(\omega;\mathbf{x}^k)$ to estimate the gradient of the loss function on $\omega$ (i.e., $\nabla_\omega \mathcal{L}_k$) in the step of shared module updating. Similarly, $\nabla q_k(\theta^k;\mathbf{x}^k)$ is used to estimate the gradient of the loss function on $\theta^k$ (i.e., $\nabla_{\theta^k}\mathcal{L}_k$) in the step of local personalized training. Considering the global $\omega$, we define the global average loss of $\omega$ as $\mathcal{L}(\omega) = 1/K\sum_{k=1}^K \mathcal{L}_k(\omega,\theta_*^k)$, where $\theta_*^k = \arg\min_{\theta^k\in\Theta}\mathcal{L}_k(\omega,\theta)$ for any $\omega\in\Omega$. To facilitate analysis, we make the following assumptions.

*Assumption 1:* There exists $\omega^*\in\Omega$ and $\theta_*^k\in\Theta$ such that the per sample loss $\mathcal{L}_k(\omega^*,\theta_*^k)=0$ for all $k\in\{1,\ldots,K\}$.

*Assumption 2:* The gradients $\nabla q_k(\theta^k;\mathbf{x}^k)$ and $\nabla q_k(\omega;\mathbf{x}^k)$ are 1-Hölder continuous. There exist $L_\theta, L_\omega > 0$ such that

$$\|\nabla q_k(\theta^k;\mathbf{x}^k) - \nabla q_k(\tilde{\theta}^k;\mathbf{x}^k)\|_2 \le L_\theta \|\theta^k - \tilde{\theta}^k\|_2$$

and

$$\|\nabla q_k(\omega;\mathbf{x}^k) - \nabla q_k(\tilde{\omega};\mathbf{x}^k)\|_2 \le L_\omega \|\omega - \tilde{\omega}\|_2$$

for any $k\in\{1,\ldots,K\}$, $\theta^k,\tilde{\theta}^k\in\Theta$, $\omega,\tilde{\omega}\in\Omega$, $\mathbf{x}^k\subseteq \boldsymbol{X}^k$.

*Assumption 3:* $\mathcal{L}_1(\omega,\theta^1),\ldots,\mathcal{L}_K(\omega,\theta^K)$ satisfy the PL inequality with the parameter $\mu>0$, i.e.,

$$\|\nabla_{\theta^k}\mathcal{L}_k(\omega,\theta^k)\|_2^2 \ge \mu\left(\mathcal{L}_k(\omega,\theta^k) - \mathcal{L}_k(\omega,\theta_*^k)\right)$$

and

$$\|\nabla_\omega \mathcal{L}_k(\omega,\theta_*^k)\|_2^2 \ge \mu\left(\mathcal{L}_k(\omega,\theta_*^k) - \mathcal{L}_k(\omega^*,\theta_*^k)\right)$$

for any $k\in\{1,\ldots,K\}$, $\theta^k\in\Theta$, $\omega\in\Omega$. Here, $\theta_*^k = \arg\min_{\theta^k\in\Theta}\mathcal{L}_k(\omega,\theta)$ for any $\omega\in\Omega$ and $\omega^* = \arg\min_{\omega\in\Omega}\frac{1}{K}\sum_{k=1}^K\mathcal{L}_k(\omega,\theta_*^k)$.

*Assumption 4:* The gradient estimates are unbiased, i.e., $\mathbb{E}[\nabla q_k(\theta^k;\mathbf{x}^k)] = \nabla_{\theta^k}\mathcal{L}_k(\omega,\theta^k)$ and $\mathbb{E}[\nabla q_k(\omega;\mathbf{x}^k)] = \nabla_\omega\mathcal{L}_k(\omega,\theta^k)$ for any $\theta^k\in\Theta$, $\omega\in\Omega$, $\mathbf{x}^k\subseteq\boldsymbol{X}^k$.

The above assumptions are standard assumptions made by prior works [69], [70], [71]. For instance, [69], [70] introduce Assumptions 1 to facilitate analysis. [70], [71] assume the continuity and PL inequality like Assumptions 2 and 3 to derive the convergence in centralized training. Furthermore, [72] also introduces these assumptions to prove the convergence of FedAvg in non-convex objectives. Based on these commonly-used assumptions, we derive the convergence of P2FMS.

*Lemma 1:* Under Assumptions 2-4, we choose a fixed learning rate $\alpha_l = \bar{\alpha}$ with $\bar{\alpha}\le\frac{\mu}{2L_\theta^2}$ and define $\varepsilon_r^k = \mathbb{E}[\mathcal{L}_k(\omega_r,\theta_s^k) - \mathcal{L}_k(\omega_r,\theta_*^k)]$, where $\theta_s^k$ denotes the local saved parameters mentioned in Algorithm 1. For any round $r\in\{1,\ldots,R\}$ and any iteration $l\in\{1,\ldots,E_l\}$, it follows

$$\mathbb{E}\left[\mathcal{L}_k(\omega_{r-1},\theta_{r,l}^k) - \mathcal{L}_k(\omega_{r-1},\theta_*^k)\right] \le \left(1-\frac{\mu\bar{\alpha}}{2}\right)^l \varepsilon_{r-1}^k. \quad (14)$$

The detailed proof of Lemma 1 is presented in Appendix A, available online. It establishes the upper bound of the local deviation for inner function (9). Next, we derive the average loss of the shared module with the global parameters $\omega_r$ in Lemma 2.

*Lemma 2:* Under Assumptions 1-4, we set a fixed learning rate $\eta_\ell = \eta$ with $\eta\le\min\{\frac{2}{\mu},\sqrt{\frac{\mu}{16L_\omega^3E^3}},\frac{\mu}{2L_\omega^2},\frac{\mu}{8L_\omega^2(1+1/K)},\frac{L_\omega^2}{2L_\omega^3(1+1/K)}\}$ in the step of shared module updating. For any round $r\in\{1,\ldots,R\}$, the following inequality holds

$$\mathbb{E}\left[\frac{1}{K}\sum_{k=1}^K\mathcal{L}_k(\omega_r,\theta_*^k)\right] - \mathcal{L}^* \le \left(1-\frac{\eta\mu}{8}\right)^r\mathcal{L}_0, \quad (15)$$

where $\mathcal{L}^* = \frac{1}{K}\sum_{k=1}^K\mathcal{L}_k(\omega^*,\theta_*^k)$. With Assumption 1, it follows $\mathcal{L}^*=0$. The $\mathcal{L}_0$ is denoted as $\mathcal{L}_0 = \frac{1}{K}\sum_{k=1}^K\mathcal{L}_k(\omega_0,\theta_*^k)$.

The detailed proof of Lemma 2 is presented in Appendix B, available online. Combining Lemmas 1 and 2 yields the convergence of P2FMS, as presented in Theorem 1.

*Theorem 1:* Suppose that Assumptions 1-4 are true, we set a fixed learning rate $\eta_\ell = \eta$ with $\eta\le\min\{\frac{2}{\mu},\sqrt{\frac{\mu}{16L_\omega^3E^3}},\frac{\mu}{2L_\omega^2},\frac{\mu}{8L_\omega^2(1+1/K)},\frac{L_\omega^2}{2L_\omega^3(1+1/K)}\}$ for shared module updating of (12). Based on Lemmas 1 and 2, the following upper bound holds

$$\mathbb{E}\left[\frac{1}{K}\sum_{k=1}^K\mathcal{L}_k(\omega_r,\theta_{r,E_l}^k)\right] - \mathcal{L}^* \le \left(1-\frac{\eta\mu}{8}\right)^r\mathcal{L}_0 + \delta_r \quad (16)$$

with

$$\delta_r = \left(1-\frac{\mu\bar{\alpha}}{2}\right)^{E_l}\frac{1}{K}\sum_{k=1}^K\varepsilon_r^k, \quad (17)$$

where $\varepsilon_r^k = \mathbb{E}[\mathcal{L}_k(\omega_r,\theta_s^k) - \mathcal{L}_k(\omega_r,\theta_*^k)]$ is introduced in Lemma 1. $\mathcal{L}_0 = \frac{1}{K}\sum_{k=1}^K\mathcal{L}_k(\omega_0,\theta_*^k)$ and $\mathcal{L}^* = \frac{1}{K}\sum_{k=1}^K\mathcal{L}_k(\omega^*,\theta_*^k)$ are mentioned in Lemma 2. Besides, $\mu$ is the constant parameter of the PL inequality in Assumption 3. $\bar{\alpha}\le\frac{\mu}{2L_\theta^2}$ denotes a fixed learning rate of the local personalized training, and each client locally trains the personalized combiner for $E_l$ iterations, as mentioned in Section IV-D.

Hence, Theorem 1 indicates that P2FMS can reach the exponential convergence rate.

## B. Cost Analysis

We first consider the client-side computation cost of P2FMS. The statistical calculation of ES in the local trend catcher is known for its low complexity and low cost [40]. While the model complexity of the personalized combiner is much smaller than that of the shared fluctuation extractor. Empirically, the model complexity of the shared fluctuation extractor with the single-layer LSTM networks is denoted as $4(na+n^2+n)+4(va+v^2+v)$, where $a$ is the number of hidden units in the LSTM. The model complexity of the personalized combiner with two fully-connected layers is $3b$, where $b$ is the number of hidden neurons in the fully-connected layer. In our work, $a\gg b$ and then $4(na+n^2+n)+4(va+v^2+v)\gg 3b$, i.e., the model complexity of the personalized combiner is much less than that of the shared fluctuation extractor and even can be ignored. In this way, the computation cost of client-side personalization operations is low enough compared to shared module updating. In conclusion, P2FMS does not significantly increase the local computing burden.

In terms of communication cost, the parameters of the local trend catcher and personalized combiner do not need to be uploaded to the server. Only parameters of the shared fluctuation extractor are exchanged between the server and clients such that

P2FMS has a similar communication overhead as conventional FL frameworks (e.g., FedAvg and FedDA).

## C. Privacy Analysis

For data privacy protection, P2FMS only transmits the parameters of the shared fluctuation extractor without exchanging client-side traffic data. Therefore, it does not introduce extra privacy concerns compared to conventional FL frameworks. Besides, conventional FL generally sends their full model parameters to the server, such as FedAvg [13], which still does not provide comprehensive privacy protection. For example, attackers may infer their training data through the full model parameters uploaded by clients (i.e., inference attacks) [73]. Prior works [74] have demonstrated that model stacking can significantly alleviate membership inference attacks, where the stacked model consists of three base models, divided into upper and lower levels. The lower level involves two models, which are trained separately on the training data. The upper level consists of a single model whose input is the concatenation of the outputs of the two models in the lower level. Our proposed P2FMS perfectly matches the two-level design of model stacking, where the shared fluctuation extractor and local trend catcher constitute the lower level, while the personalized combiner denotes the upper level. Besides, a lot of works [75], [76], [77] have pointed out that overfitting is a major factor in the success of inference attacks. The design of the stacked model can avoid overfitting, thereby reducing the amount of deterministic data information memorized by ML models to defense inference attacks [76], [78]. Actually, this stacked model is internally complex, from which it is difficult for attackers to infer clear data information. Given the above observations, the stacked forecasting model in P2FMS has the ability to defend against inference attacks. In addition, the parameters of the personalized combiner are not transmitted to the server such that attackers can only obtain partial parameter updates of local models at most. The amount of data information memorized in transmitted local updates also reduces accordingly, thereby alleviating inference attacks.

Furthermore, P2FMS is orthogonal to existing FL attack defense techniques (e.g., differential privacy [79], Homomorphic Encryption [80], and robust aggregation [81]), which can be directly applied with P2FMS. For instance, we could add noise to transmitted parameters via differential privacy, and encrypt local updates to further improve the defense capability.

## VI. Experimental Evaluation

### A. Experimental Settings

*Datasets:* We conduct extensive experiments over two realistic telecommunication datasets published by Telecom Italia [26]. They contain traffic series of cell phones (i.e., Short Message Service (SMS), Call and Internet service sent or received by users) in the city of Milan and the Province of Trentino. Referring to [10], we allocate 100 distributed clients, where each client logs the traffic data for two months. We choose the traffic data of the first 40 days as the training set, the traffic data of the next 7 days as the validation set, and the remaining 14 days as the test set.

*Data preprocessing:* The 10-minute interval of original datasets makes the traffic data sparse and is unsuitable for practical network management. Therefore, we aggregate traffic hourly [62]. Referring to [10], we use Z-score normalization to process temporal data and use the Min-Max normalization to scale the location information into the range $[0, 1]$.

*Evaluation metrics:* We adopt two general regression metrics, Mean Squared Error (MSE) and Mean Absolute Error (MAE), to evaluate forecasting performance. A lower MSE or MAE indicates a better forecasting performance.

*Hyperparameter settings:* For model and hyperparameter settings, we mainly refer to [10]. Specifically, we set the window size of closeness dependence to $n = 3$, the period-dependent window size to $v = 3$, and the periodicity to $p = 24$. In this setting, the length of the input sequence of the local trend catcher is $\tau = v \times p = 72$. Due to the limited computation capability in edge networks, the closeness extractor and period extractor are implemented by relatively lightweight LSTM networks [10]. For the traffic in Milan, we utilize single-layer LSTM networks with the number of hidden units $a = 64$ as the closeness extractor and period extractor, followed by two fully-connected layers with the number of hidden neurons $b = 2$ as the personalized combiner. For data in Trentino, the model architectures of the closeness extractor and period extractor contain two LSTM layers. The total number of rounds is $R = 60$. In client-side training, each client executes local personalized training for two epochs and updates the shared module for three epochs. We set the local batch size to be 20 such that the batch number of local training data is $B = 39$ by default, which is also correlated with the local training sample size. Our code is available at https://github.com/sunnyxuejj/P2FMS.

### B. Baselines

We compare against a variety of network traffic forecasting frameworks that do not transmit raw data to protect privacy, including local/centralized architectures, conventional FL, and personalized FL.

*Local and centralized architectures:* In local architectures, each client independently fits a Trend Catcher (TC) or trains a Fluctuation Extractor (FE) based on its data for local forecasting. For centralized architectures, the traffic data from all clients are transferred to the server. With the collected data, the server can centrally train a prediction model with two LSTM layers [10] (i.e., the same representation model structure as our shared fluctuation extractor). Besides, we evaluate the Graph Convolutional Network (GCN) [82] to integrate spatial and temporal correlations of traffic data in centralized architectures (i.e., Central GCN). The state-of-the-art AdpSTGCN [83] is also considered, which improves GCN-based models through adaptive graph convolution and dynamic gathering of multiple feature graphs in centralized architectures.

*Conventional FL:* For conventional FL frameworks with a shared model, we evaluate *FedAvg* [13], *FedAtt* [84], *FedProx* [85], and *FedDA* [10]. FedAvg is a typical FL framework for general data with average parameter aggregation. FedAtt explores an attention-based aggregation strategy in the server,

TABLE I
TEST RESULTS OF DIFFERENT METHODS

| Methods | M-SMS | | M-Call | | M-Internet | | T-SMS | | T-Call | | T-Internet | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| Local TC | 0.6953 | 0.5136 | 0.3463 | 0.3963 | 0.5200 | 0.5264 | 3.8800 | 0.9376 | 1.6124 | 0.5719 | 10.1427 | 1.5517 |
| Local FE | 0.3831 | 0.3312 | 0.0954 | 0.1821 | 0.1351 | 0.2485 | 3.3707 | 0.7826 | 1.1029 | 0.4017 | 7.0469 | 0.9692 |
| Central LSTM | 0.3002 | 0.3106 | 0.0889 | 0.1779 | 0.1304 | 0.2390 | 1.6263 | 0.5749 | 0.7393 | 0.3374 | 5.0348 | 0.7453 |
| Central GCN | 0.2857 | 0.3184 | 0.0861 | 0.1782 | 0.1306 | 0.2447 | 1.5344 | 0.6596 | 0.6873 | 0.3608 | 3.6230 | 0.7368 |
| AdpSTGCN | 0.2767 | 0.2987 | 0.0797 | 0.1693 | 0.1242 | 0.2345 | 1.5197 | 0.5747 | 0.7309 | 0.3416 | 3.3933 | 0.7167 |
| FedAvg | 0.3235 | 0.3084 | 0.0900 | 0.1727 | 0.1312 | 0.2419 | 2.4492 | 0.6329 | 1.3108 | 0.4091 | 5.5967 | 0.8006 |
| FedAtt | 0.3013 | 0.3076 | 0.0905 | 0.1752 | 0.1315 | 0.2369 | 2.3400 | 0.6344 | 1.1701 | 0.3841 | 5.6914 | 0.7998 |
| FedProx | 0.3167 | 0.3084 | 0.0906 | 0.1737 | 0.1293 | 0.2400 | 2.8174 | 0.6657 | 1.3103 | 0.4081 | 6.6608 | 0.8674 |
| FedDA | 0.3242 | 0.3111 | 0.0912 | 0.1754 | 0.1291 | 0.2420 | 2.2306 | 0.6282 | 1.0779 | 0.3789 | 5.7224 | 0.7757 |
| FedAvg+FT | 0.3376 | 0.3156 | 0.0890 | 0.1749 | 0.1242 | 0.2359 | 2.4430 | 0.6642 | 0.9889 | 0.3690 | 4.9263 | 0.7943 |
| FedAtt+FT | 0.3360 | 0.3137 | 0.0883 | 0.1752 | 0.1238 | 0.2362 | 2.7035 | 0.6857 | 0.9030 | 0.3556 | 5.2831 | 0.8095 |
| FedProx+FT | 0.3391 | 0.3139 | 0.0862 | 0.1732 | 0.1223 | 0.2351 | 2.6422 | 0.6775 | 0.9960 | 0.3669 | 5.5199 | 0.8254 |
| FedDA+FT | 0.3094 | 0.3072 | 0.0864 | 0.1725 | 0.1224 | 0.2357 | 2.5753 | 0.6827 | 1.0688 | 0.3652 | 6.1347 | 0.8166 |
| Fed-MTL | 0.3645 | 0.3265 | 0.0948 | 0.1879 | 0.1393 | 0.2585 | 2.7007 | 0.6979 | 1.1587 | 0.4100 | 6.4387 | 0.9354 |
| PerFedAvg | 0.3103 | 0.3058 | 0.0888 | 0.1719 | 0.1316 | 0.2425 | 2.1775 | 0.6087 | 1.1886 | 0.3885 | 5.5834 | 0.7900 |
| APFL | 0.3366 | 0.3178 | 0.0897 | 0.1820 | 0.1313 | 0.2493 | 2.4521 | 0.6431 | 1.1520 | 0.3956 | 5.1650 | 0.7684 |
| L2GD | 0.3564 | 0.3300 | 0.0982 | 0.1851 | 0.1387 | 0.2575 | 2.4894 | 0.6691 | 1.0157 | 0.4064 | 5.9384 | 0.9011 |
| FedRep | 0.3708 | 0.3698 | 0.0950 | 0.1863 | 0.1275 | 0.2386 | 2.6054 | 0.6837 | 0.9866 | 0.3705 | 5.5221 | 0.7858 |
| Ditto | 0.2987 | 0.3087 | 0.0895 | 0.1723 | 0.1253 | 0.2403 | 2.3131 | 0.6328 | 1.1266 | 0.3823 | 5.1878 | 0.7651 |
| SuPerFed | 0.3468 | 0.3321 | 0.1098 | 0.2097 | 0.1316 | 0.2539 | 2.8600 | 0.7184 | 1.0007 | 0.4050 | 5.7834 | 0.8853 |
| **P2FMS** | **0.2748** | **0.2966** | **0.0793** | **0.1641** | **0.1194** | **0.2341** | **1.4921** | **0.5688** | **0.4629** | **0.3340** | **2.5859** | **0.7058** |

A lower MSE or MAE indicates a better prediction.

and FedProx introduces a regularization term into local training loss. FedDA first extends FL to network traffic forecasting and proposes augmented data-based clustering and dual attention-based aggregation.

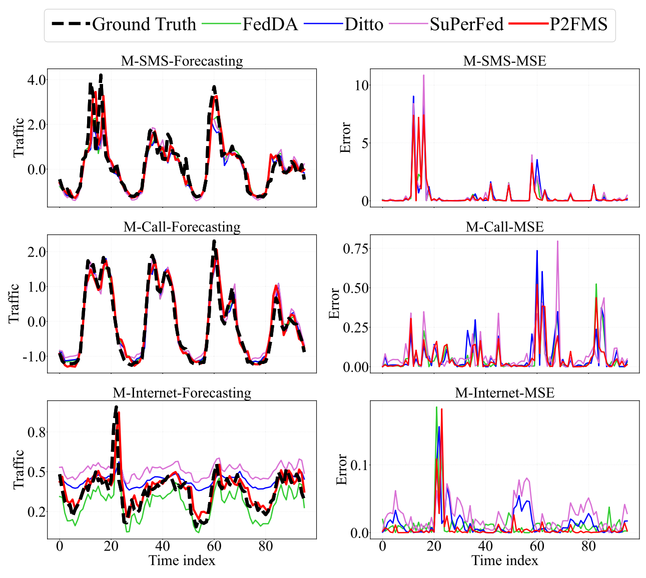
*Personalized FL:* Inspired by [54], we compare with the *local fine-tuning (FT)*, where each client trains total or several top layers of the shared model on local data before inference. Among other personalized FL approaches, *Fed-MTL* [57] applies a multi-task learning framework to address statistical heterogeneity challenges. *PerFedAvg* [28] explores the personalized variant of FedAvg within Model-Agnostic Meta-Learning (MAML) [55]. *APFL* [31] adaptively learns a customized model for each client by leveraging the correlation between local and global models. *L2GD* [24] seeks an explicit trade-off between the global and the local model for each client. *FedRep* [25] leverages all of the data stored across clients to learn a global high-dimensional representation and then enables each client to train a personalized low-dimensional head. *Ditto* [30] adds a lightweight personalization module for regularizing personalized models towards their average to learn accurate and robust models. *SuPerFed* [32] introduces an explicit connected subspace containing low-loss solutions into model mixture-based personalized FL.

## C. Performance Evaluation

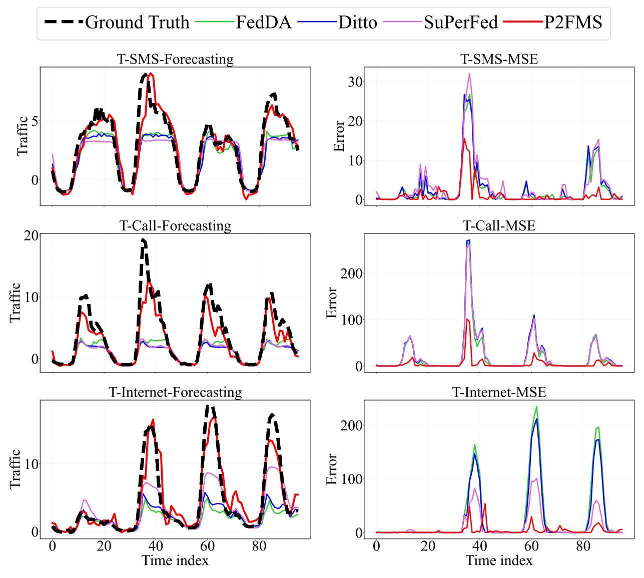
*Overall forecasting performance:* The forecasting performance results of different methods in the test sets are shown in Table I, where M and T denote Milan and Trentino. As shown in Table I, P2FMS consistently provides superior forecasting performance across all datasets compared to status quo methods. First, local TC and FE demonstrate poor performance with higher MSE and MAE. The main reasons are that a local TC fitted by ES cannot capture complex non-linear patterns, and a local FE trained on the traffic data of a single client shows larger errors, indicating DNN with only local knowledge is not enough to accurately model fluctuation features.

Centralized architectures collect client-side data on the server for model training, which can extract more knowledge from all data to improve the representation capabilities of DNN models. Thus, central LSTM significantly outperforms local FE, as shown in Table I. Furthermore, central GCN and AdpSTGCN model spatial correlations of traffic data to boost forecasting performance. However, privacy concerns and network congestion caused by data transmission are unavoidable in centralized architectures. Moreover, these centralized methods train a model with strong generalization, failing to tackle the personalization challenge posed by heterogeneous traffic patterns. P2FMS adaptively integrates the general fluctuation representations from the global view and unique trend fittings from the local view to achieve personalized forecasts. Compared to the state-of-the-art AdpSTGCN, P2FMS consistently demonstrates lower forecasting errors and provides up to 36.67% MSE performance gains in Call traffic data of Trentino. Besides, P2FMS does not involve any raw data exchange, avoiding data privacy and network congestion issues.

As for conventional FL, P2FMS achieves 7.51%–11.18% MSE performance gains for SMS, Call, and Internet traffic in Milan compared to the state-of-the-art FL network forecasting framework, FedDA. For Trentino, the MSE performance improves by 30.13%–54.81% for the three types of services. Furthermore, P2FMS outperforms state-of-the-art personalized FL

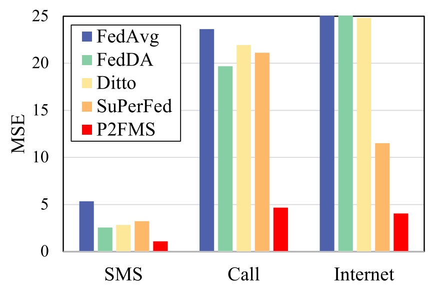


(a) Forecasting results and corresponding squared error of Milan.

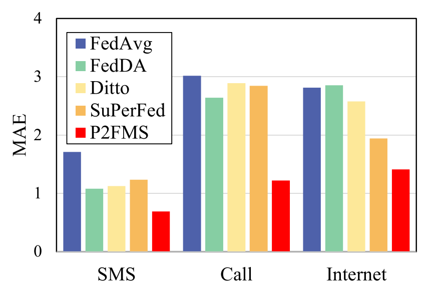


(b) Forecasting results and corresponding squared error of Trentino.

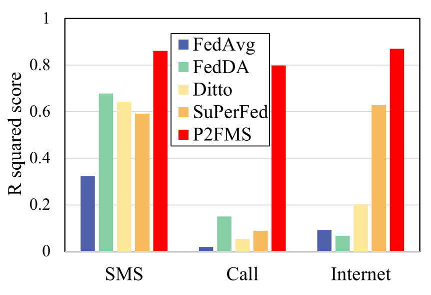
Fig. 5. Comparison between ground truth and forecasting results and the corresponding squared error of each time slot.



(a) MSE results of burst cases.



(b) MAE results of burst cases.



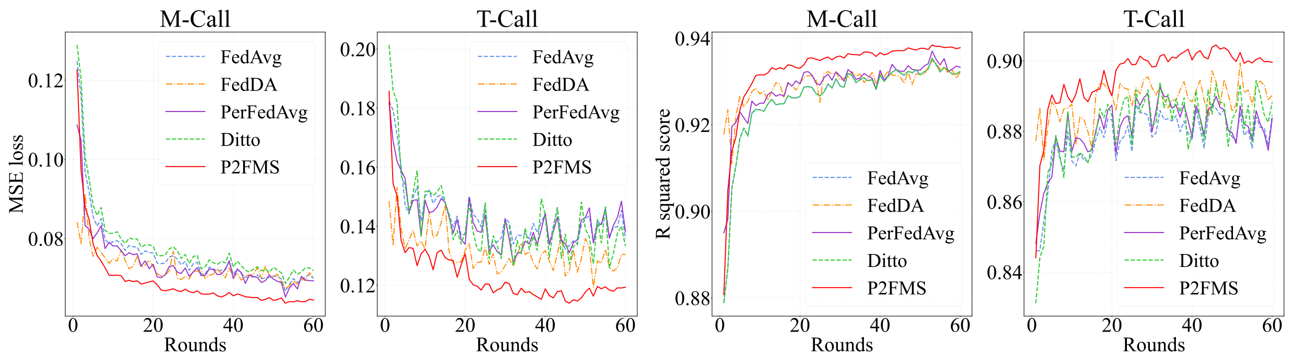
(c) R-squared scores of burst cases.

Fig. 6. Performance evaluations on burst cases of Trentino.

approaches over all datasets, achieving the lowest MSE and the lowest MAE. Specifically, compared with the best-performing method in baselines, P2FMS provides 2.37%–8.00% MSE performance gains on Milan traffic data. The MSE performance of Trentino traffic data is improved by 28.43%–47.51%. Prediction performance improvements of P2FMS on Trentino traffic are more obvious than that on Milan data. Since the traffic data in Trentino has a higher degree of non-IID and contains more complex burst distributions, while our proposed P2FMS is sensitive to unique traffic bursts of each client, as shown in Fig. 5. Fig. 5 plots the comparison of the ground truth and prediction values from test sets, where the corresponding squared error of each time slot is also provided for quantitative comparison. In Fig. 5(a) and (b), the three subplots on the left present the curves of the ground truth and prediction values for several clients over the last 4 days, and the three subplots on the right show the corresponding forecasting errors in different time slots. We notice that P2FMS generally performs better than baselines, especially when the traffic bursts and the curve has spikes.

*Burst case performance:* As shown in Fig. 5(b), we sample burst cases from each type of traffic in Trentino. We observe that the baseline methods only predict stationary periodic patterns. In contrast, P2FMS not only accurately extracts smooth fluctuation laws, but also promptly captures and responds to the uptrend to forecast traffic bursts. Thus, for these burst cases, the predictions of P2FMS are generally closer to the ground truth than those of other baselines, significantly reducing the forecasting MSE in almost all time slots. Compared to the traffic series of Milan in Fig. 5(a), the advantages of P2FMS are more apparent in Trentino traffic data involving more traffic bursts. Quantitatively, we report the forecasting MSE, MAE, and R-squared score of the three burst cases in Fig. 6. Motivated by [10], the R-squared score is used to characterize the prediction accuracy. It can be seen that P2FMS consistently provides the lowest prediction errors and highest forecasting accuracy, demonstrating the superior burst sensitivity of P2FMS.

*Convergence performance:* To evaluate the convergence of P2FMS, we report the average forecasting errors and R-squared scores of validation sets on Call traffic data varying with rounds in Fig. 7. Note that FedDA introduces a data augmentation strategy, and the augmented dataset is transferred to the server to centrally pre-train a quasi-global model before federated training. Starting with the global model pre-trained on the augmented data, FedDA performs better than other methods with



(a) Average forecasting errors (MSE) vary with different numbers of rounds. (b) R-squared scores vary with different numbers of rounds.

Fig. 7. Forecasting performance versus rounds on Call traffic data in Milan and Trentino.

TABLE II
TEST RESULTS OF ABLATION EXPERIMENTS

| Methods | M-SMS | | M-Call | | M-Internet | | T-SMS | | T-Call | | T-Internet | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| Local CFT | 0.3612 | 0.3216 | 0.0875 | 0.1756 | 0.1367 | 0.2505 | 2.9356 | 0.7352 | 0.8194 | 0.3690 | 5.7380 | 0.8968 |
| Local TC | 0.6953 | 0.5136 | 0.3463 | 0.3963 | 0.5200 | 0.5264 | 3.8800 | 0.9376 | 1.6124 | 0.5719 | 10.1427 | 1.5517 |
| P2FL | 0.3708 | 0.3698 | 0.0950 | 0.1863 | 0.1275 | 0.2386 | 2.6054 | 0.6837 | 0.9866 | 0.3705 | 5.5221 | 0.7858 |
| FedCFT | 0.2918 | 0.3011 | 0.0816 | 0.1672 | 0.1298 | 0.2391 | 1.8540 | 0.5899 | 0.8142 | 0.3538 | 3.6780 | 0.7447 |
| **P2FMS** | **0.2748** | **0.2966** | **0.0793** | **0.1641** | **0.1194** | **0.2341** | **1.5585** | **0.5756** | **0.5437** | **0.3428** | **2.5859** | **0.7058** |

randomly initialized global parameters in the first five early rounds of federated training, as shown in Fig. 7. However, it converges to higher forecasting error and lower accuracy in later rounds. Besides, augmented data transmission still has the risk of data privacy leakage and exhausts network capacity. Global model pre-training also takes a lot of time, as mentioned in Section VI-F. As the federated training rounds proceed, P2FMS quickly converges to the lowest MSE loss and highest prediction accuracy. For example, after training 60 rounds on the Call traffic data of Trentino, the MSE validation performance in the last ten rounds of P2FMS is 0.1187, while FedDA and Ditto offer the average MSE loss of 0.1278 and 0.1359, respectively. The average R-squared score of P2FMS is 0.9001, while FedDA and Ditto provide the average R-squared scores of 0.8917 and 0.8854.

## D. Ablation Studies

We validate the effectiveness of different modules in the P2FMS framework by several ablation experiments. First, we evaluate the Combination of Fluctuation extractor and Trend catcher in the Local training architecture (i.e., Local CFT). As shown in Table II, Local CFT provides worse performance than P2FMS. Thus, we can infer that the knowledge learned from a single client is insufficient to support precise traffic forecasting, further demonstrating the necessity of federated training across clients.

*Ablation evaluation of local TC:* To validate the role of the local Trend Catcher (TC), we evaluate the average prediction MSE and MAE across clients with only local TC. As shown in Table II, Local TC provides the worst performance with the highest MSE and MAE. For qualitative analysis, we sample two burst cases from Internet traffic in Trentino to compare the prediction curves of the local TC with the ground truth curves in Fig. 8. It can be seen that the use of local TC alone overreacts to uptrends or downtrends and fails to accurately grasp bursts, resulting in large forecasting deviations.

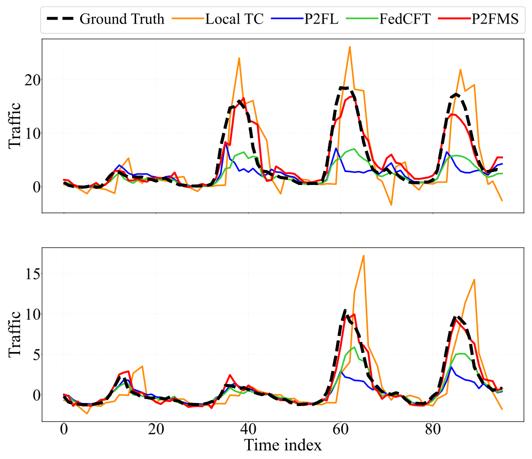


Fig. 8. Comparison between ground truth and forecasting results for ablation experiments.

*Ablation evaluation of shared FE:* To verify the impact of the FL-trained shared Fluctuation Extractor (FE), we directly train a DNN forecasting model with the shared fluctuation representation and personalized prediction head for each client in our framework (called P2FL) without involving locally additional trend-aware operations. Specifically, the personalized prediction

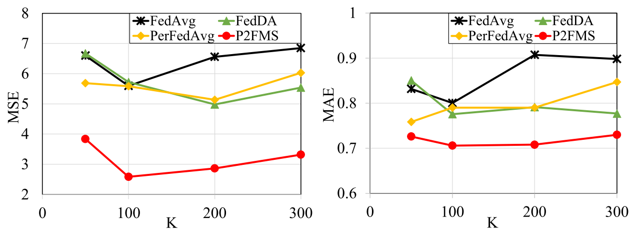


Fig. 9. MSE and MAE results under different client numbers.

TABLE III
INFLUENCE OF $E_l$ AND $E$ ON FORECASTING MSE

| | $E_l = 1B$ | $E_l = 2B$ | $E_l = 3B$ | $E_l = 4B$ |
|---|---|---|---|---|
| $E = 1B$ | 4.4059 | 4.1430 | 3.5330 | 3.3394 |
| $E = 2B$ | 2.9593 | 3.4279 | 3.0028 | **3.2559** |
| $E = 3B$ | 3.2698 | **2.5859** | **2.6816** | 3.5174 |
| $E = 4B$ | **2.9433** | 3.2404 | 3.5304 | 3.8417 |

head is the last layer of the DNN forecasting model, which is regarded as the client-specific personalization module trained in the step of local personalized training, as referred to [25]. The results in Table II reveal that P2FL always performs worse than P2FMS in all datasets. Qualitatively, we observe from Fig. 8 that P2FL only focuses on periodic fluctuation laws and fails to cope with unstable traffic spikes, bringing poor performance. In summary, both have different emphases: Local TC is extremely sensitive to trends, while shared FE is good at extracting stationary patterns. When the two are used alone, the prediction performance is poor due to single-angle property modeling. The adaptive combination of the Local TC and shared FE would integrate the advantages of both sides and make up for the shortcomings.

*Ablation evaluation of personalized combiner:* Without personalized self-fusion of multiple properties, we use a global shared combiner to integrate the fluctuation extractor and local trend catcher in federated settings (i.e., FedCFT), where the parameters of the shared combiner are transmitted to the server for global aggregation. FedCFT offers lower forecasting errors than other baselines (including the FL methods in Table I), which implies that the multi-scale representations of fluctuation and trend properties are beneficial for boosting forecasting performance. Furthermore, P2FMS consistently outperforms FedCFT in terms of forecasting MSE and MAE across all datasets, quantitatively demonstrating the effectiveness of our personalized combiner over heterogeneous traffic data. Qualitatively, the global combiner of FedCFT weights fluctuations and trends from a global view, which cannot individually capture unique traffic bursts of each client in heterogeneous settings, as shown in Fig. 8. While the personalized combiner in P2FMS adaptively weighs the impact of periodic fluctuation laws and local trend changes on future predictions for each client to accurately grasp heterogeneous traffic bursts across clients, bringing superior predictions closer to the ground truth.

## E. Hyperparameters Test

We further investigate the impact of different hyperparameters in our proposed P2FMS, including the number of total clients $K$, local personalized training epochs $E_l$ and shared updating epochs $E$. Besides, we also evaluate P2FMS under the unbalanced data setting.

*Effect of total client number:* We explore the scalability of P2FMS with different numbers of total clients. Fig. 9 shows the prediction MSE and MAE results of P2FMS and other baselines with the number of clients $K \in \{50, 100, 200, 300\}$ on the Internet traffic dataset of Trentino, from which we observe that P2FMS consistently maintains the lowest forecasting errors under different numbers of total clients. FedAvg generally performs worst, and the forecasting error of FedAvg increases as more clients participate. Compared to three baselines, P2FMS can reach more than 32% MSE performance improvements under various client number settings.

*Effect of training epochs:* To evaluate the influence of local personalized training iteration $E_l$ and shared updating iteration $E$ on forecasting performance, we evaluate P2FMS on the Internet traffic of Trentino with $E \in \{1B, 2B, 3B, 4B\}$ and $E_l \in \{1B, 2B, 3B, 4B\}$, where $B = 39$ denotes the batch number of training data on each client, as mentioned in Section VI-A. The MSE results of different settings are shown in Table III. Note that the italiced value is the lowest one in each column, meaning the best performance with the optimal $E$ under a specific $E_l$. We notice that the settings of $E$ and $E_l$ demonstrate significant impacts on forecasting errors of P2FMS. When $E_l = 1$, with the increase of $E$, the prediction MSE exhibits a declining trend. However, if $E_l > 1$, the forecasting MSE first decreases rapidly and then slowly increases as $E$ rises. This phenomenon is consistent with our theoretical derivation. According to Inequality (27) in the Appendix, available online, for fixed local personalized parameters $\theta^k, k \in \mathcal{K}$, the average prediction loss of the shared module is determined by two terms: $(1 - \frac{\eta\mu}{4})^E$ and $2\eta^3 \tilde{} L_\omega^3 (E-1)^3$. The first term decreases as $E$ increases. The second term, on the contrary, increases with the rising of $E$. Hence, there exists a critical point to balance the two items so that the forecasting MSE first decreases and then increases. For different local personalized parameters caused by distinct $E_l$, the critical point would be different. Moreover, considering the nature of model training, the personalized combiners are simple structures with few parameters (as mentioned in Section V-B) and local data is limited, which are prone to overfitting. Proper selection of $E_l$ is necessary to avoid overfitting and improve test performance. In terms of $E$, a large $E$ will lead to severe local model inconsistency on heterogeneous data across clients, which damages the generalization of the shared FE. Therefore, the choice of $E$ has to trade off the challenges of local underfitting and model inconsistency. Taking these factors into consideration, we adopt the grid search method for training epoch settings and find the optimal settings with $E_l = 2B$ and $E = 3B$, as shown in Table III.

*Impact of unbalanced data:* In practical applications, it is inevitable that the sample size of each client is unbalanced. Therefore, we evaluate P2FMS and several baselines on SMS traffic under the unbalanced non-IID data setting, where the

TABLE IV
PREDICTION PERFORMANCE ON UNBALANCED DATA

| Methods | M-SMS | | T-SMS | |
|---|---|---|---|---|
| | MSE | MAE | MSE | MAE |
| FedAvg | 0.3185 | 0.3111 | 2.5649 | 0.6497 |
| FedDA | 0.3306 | 0.3127 | 2.0591 | 0.6189 |
| PerFedAvg | 0.3169 | 0.3083 | 2.7036 | 0.6638 |
| **P2FMS** | **0.2973** | **0.3044** | **1.5990** | **0.5935** |

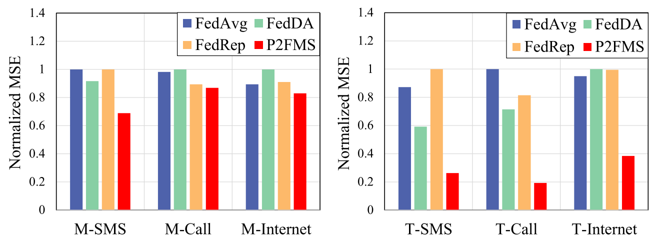
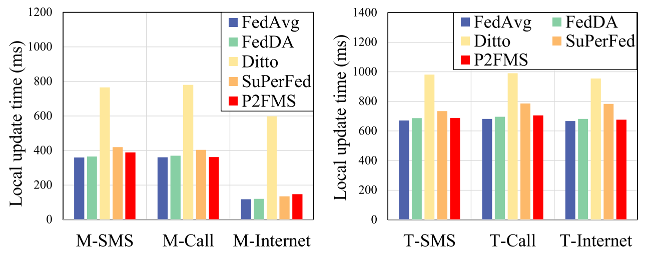
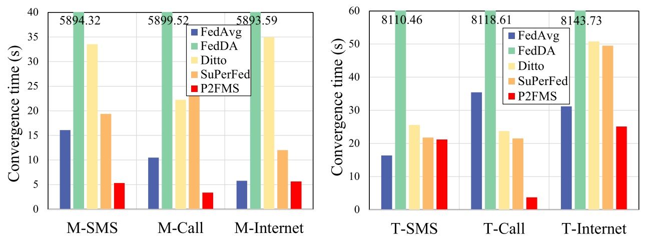


Fig. 10. Generalization evaluations for new clients.



(a) Comparison of local training time in a round.



(b) Comparison of convergence time.

Fig. 11. Running time analysis.

number of training samples on each client is randomly chosen from $\{a * 24 - 3 | a = 11, 12, \ldots, 37\}$, and different clients possesses different numbers of training samples. The experimental results are shown in Table IV. We observe that P2FMS always outperforms baselines, demonstrating the effectiveness of P2FMS on unbalanced data. In particular, P2FMS improves MSE performance by up to 40.85% and achieves up to 10.60% MAE gains under the unbalanced data setting.

## *F. Generalization and Cost Evaluation*

*Generalization for new clients:* We evaluate the strength of P2FMS in generalization for new clients. We compare against FedAvg, FedDA, and FedRep, where the FL-trained global model of FedAvg and FedDA is tuned for 5 epochs on the local data of a new client. In FedRep, the last layer of the model architecture is trained on the local data of the new client, where the newly trained last layer is viewed as the local head and can be combined with previously learned shared representation to form a customized model for the new client. For P2FMS, we freeze the FL-trained shared fluctuation extractor, then locally fit a local trend catcher and train a personalized combiner for the new client. As shown in Fig. 10, P2FMS has the best forecasting performance for new clients across all datasets compared to other baselines, and this strength is more pronounced in the Trentino data that involves more traffic bursts. It reveals that P2FMS offers better generalization for new clients than existing FL methods.

*Computation and communication cost:* As for local computation costs, Fig. 11(a) reports local update time in a round for Milan (the left subfigure) and Trentino (the right subfigure). We notice that P2FMS generally provides shorter local training time than advanced PFL baselines Ditto and SuPerFed, but slightly increases the local training time compared to conventional FedAvg framework. The local update time of P2FMS is 376.57 ms shorter than Ditto, while the maximum time increase is 29.74 ms in SMS traffic of Milan compared to FedAvg. Furthermore, we observe from Fig. 7(b) that P2FMS can generally achieve higher accuracy than baselines under the same number of rounds, which indicates that P2FMS can consume fewer communication rounds to reach a pre-defined target accuracy, beneficial for communication efficiency. For example, P2FMS obtains the R-squared score of 0.83 after 9 rounds for Call traffic in Milan, while FedDA needs 20 rounds and FedAvg needs 31 rounds.

To simulate wireless links, we utilize the Verizon 5G Cellular network with an average download speed of 135.3 Megabits per second (Mbps) and an average upload speed of 19.9 Mbps [86]. On this basis, we calculate the convergence time which is the sum of local training time and parameter transmission time before the target accuracy is reached, where the target accuracies are 0.82, 0.93, and 0.83 for SMS, Call, and Internet traffic in Milan, respectively. For Trentino, they are 0.75, 0.89, and 0.78 for three types of service. The results are shown in Fig. 11(b). Since FedDA includes augmented data transmission, client clustering, and global model pre-training before FL training across clients, its convergence time is much longer than that of other baselines. Although P2FMS increases the local update time by 29.74 ms compared to FedAvg in SMS traffic of Milan, it significantly reduces the number of communication rounds required to reach target accuracy, thus making the total convergence time 12.07 s shorter than FedAvg. In addition, the total convergence time of P2FMS is generally shorter than other baselines, which brings 19.5%–82.6% time savings compared to the state-of-the-art methods over all datasets, indicating the time efficiency of P2FMS.

## VII. DISCUSSION

We discuss the adaptability and prospective extensions of P2FMS. We can easily expand P2FMS to accommodate various neural networks (NNs), like Graph Convolutional Network (GCN) [82], Transformer [87], and iTransformer [88], by simply adopting different NNs as model structures for the shared

TABLE V
RESULTS ON THE BACKBONE OF TRANSFORMER

| Methods | M-SMS | | T-SMS | |
|---|---|---|---|---|
| | MSE | MAE | MSE | MAE |
| FedAvg + Trans | 0.0800 | 0.0990 | 1.4613 | 0.2528 |
| FedDA + Trans | 0.0633 | 0.0865 | 1.3732 | 0.2498 |
| PerFedAvg + Trans | 0.0787 | 0.0959 | 1.4432 | 0.2562 |
| **P2FMS + Trans** | **0.0548** | **0.0735** | **0.9455** | **0.2453** |

fluctuation extractor. This work emphasizes the PFL framework with multi-scale property representations from different views and personalized fusion of multiple properties rather than the detailed model structure design, where the model structures of the shared fluctuation extractor and personalized combiner can be adjusted according to practical scenarios. Referring to [10], a relatively lightweight model should be adopted in edge networks with limited computing power. In this work, we consider edge network scenarios and use a lightweight LSTM as the main structure of the shared fluctuation extractor. However, if participating clients involve data centers with strong computing power, complicated models could be adopted for the shared fluctuation extractor.

We take Transformer [87] as an example to demonstrate the adaptability of P2FMS to other complicated models. We add experiments using Transformer as the main structure of the shared fluctuation extractor. The MSE and MAE results on SMS data are shown in Table V. We observe that a complicated Transformer can achieve better performance than a lightweight LSTM. More importantly, P2FMS still outperforms baselines with Transformer, which demonstrates that P2FMS still maintains its performance advantages on complicated models.

## VIII. Conclusion

In this paper, we first integrate the heterogeneous multi-scale properties of client-side traffic data into PFL and propose a novel prediction framework, Personalized Federated Forecasting with Multi-property Self-fusion (P2FMS). By multi-scale property representations from different views and personalized multi-property fusion, P2FMS can sensitively capture traffic bursts and improve forecasting performance over heterogeneous traffic data. In P2FMS, each client decomposes the traffic series into different time scales. Based on multi-scale temporal sequences and spatial information of clients, we represent fluctuation properties from the global view and fit unique non-stationary trend property from the local view. A personalized combiner is designed to accurately weigh the impact of fluctuation laws and trend changes on predictions for each client, which can accommodate heterogeneous traffic patterns across BSs in real edge network scenarios. Extensive experiments show that P2FMS offers up to 47.51% MSE performance gain with better generalization and shorter convergence time compared to state-of-the-art PFL methods. We believe P2FMS represents a significant step towards the realization of distributed traffic forecasting in edge networks.

## References

[1] C. Zhang, H. Zhang, J. Qiao, D. Yuan, and M. Zhang, "Deep transfer learning for intelligent cellular traffic prediction based on cross-domain big data," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1389–1401, Jun. 20191389–1401, Jun. 2019. .

[2] S. Hui et al., "Knowledge enhanced GAN for IoT traffic generation," in *Proc. ACM Web Conf.*, 2022, pp. 3336–3346.

[3] X. Wang et al., "A survey on deep learning for cellular traffic prediction," *Intell. Comput.*, vol. 3, 2024, Art. no. 0054.

[4] X. Chen et al., "One for all: Traffic prediction at heterogeneous 5G edge with data-efficient transfer learning," in *Proc. IEEE Glob. Commun. Conf.*, 2021, pp. 01–06.

[5] T. Zheng and L. Baochun, "Poisoning attacks on deep learning based wireless traffic prediction," in *Proc. 2022 IEEE Conf. Comput. Commun.*, 2022, pp. 660–669.

[6] Q. Wu, K. He, X. Chen, S. Yu, and J. Zhang, "Deep transfer learning across cities for mobile traffic prediction," *IEEE/ACM Trans. Netw.*, vol. 30, no. 3, pp. 1255–1267, Jun. 2022.

[7] F. Sun et al., "Mobile data traffic prediction by exploiting time-evolving user mobility patterns," *IEEE Trans. Mobile Comput.*, vol. 21, no. 12, pp. 4456–4470, Dec. 2022.

[8] K. He, X. Chen, Q. Wu, S. Yu, and Z. Zhou, "Graph attention spatial-temporal network with collaborative global-local learning for citywide mobile traffic prediction," *IEEE Trans. Mobile Comput.*, vol. 21, no. 4, pp. 1244–1256, Apr. 2022.

[9] Y. Yao, B. Gu, Z. Su, and M. Guizani, "MVSTGN: A multi-view spatial-temporal graph network for cellular traffic prediction," *IEEE Trans. Mobile Comput.*, vol. 22, no. 5, pp. 2837–2849, May 2023.

[10] C. Zhang, S. Dang, B. Shihada, and M.-S. Alouini, "Dual attention-based federated learning for wireless traffic prediction," in *Proc. 2021 IEEE Conf. Comput. Commun.*, 2021, pp. 1–10.

[11] M. Marwah and M. Arlitt, "Deep learning for network traffic data," in *Proc. 28th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2022, pp. 4804–4805.

[12] C. Meng, S. Rambhatla, and Y. Liu, "Cross-node federated graph neural network for spatio-temporal data modeling," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2021, pp. 1202–1211.

[13] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, 2017, pp. 1273–1282.

[14] Q. Li et al., "A survey on federated learning systems: Vision, hype and reality for data privacy and protection," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 4, pp. 3347–3366, Apr. 2023.

[15] V. Perifanis et al., "Towards energy-aware federated traffic prediction for cellular networks," in *Proc. Int. Conf. Fog Mobile Edge Comput.*, 2023, pp. 93–100.

[16] D. Kim, S. Shin, J. Jeong, and J. Lee, "Joint edge server selection and data set management for federated-learning-enabled mobile traffic prediction," *IEEE Internet Things J.*, vol. 11, no. 3, pp. 4971–4986, Feb. 2024.

[17] V. Perifanis, N. Pavlidis, R.-A. Koutsiamanis, and P. S. Efraimidis, "Federated learning for 5G base station traffic forecasting," *Comput. Netw.*, vol. 235, 2023, Art. no. 109950.

[18] L. Li, Y. Zhao, J. Wang, and C. Zhang, "Wireless traffic prediction based on a gradient similarity federated aggregation algorithm," *Appl. Sci.*, vol. 13, no. 6, 2023, Art. no. 4036.

[19] C. Guang, G. Jian, and D. Wei, "A time-series decomposed model of network traffic," in *Proc. Adv. Natural Comput.*, 2005, pp. 338–345.

[20] S. S. Rangapuram, M. W. Seeger, J. Gasthaus, L. Stella, Y. Wang, and T. Januschowski, "Deep state space models for time series forecasting," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 7796–7805.

[21] Q. Yu et al., "Network traffic overload prediction with temporal graph attention convolutional networks," in *Proc. IEEE Int. Conf. Commun. Workshops*, 2022, pp. 885–890.

[22] S. Li, J. Song, L. Xu, Y. Hu, W. Luo, and X. Zhou, "Network traffic prediction based on the feature of newly-generated network flows," in *Proc. IFIP Netw. Conf.*, 2022, pp. 1–8.

[23] L. Yang, J. Huang, W. Lin, and J. Cao, "Personalized federated learning on non-IID data via group-based meta-learning," *ACM Trans. Knowl. Discov. Data*, vol. 17, no. 4, pp. 1–20, 2023.

[24] F. Hanzely and P. Richtárik, "Federated learning of a mixture of global and local models," 2020, *arXiv:2002.05516*.

[25] L. Collins, H. Hassani, A. Mokhtari, and S. Shakkottai, "Exploiting shared representations for personalized federated learning," in *Proc. 38th Int. Conf. Mach. Learn.*, 2021, pp. 2089–2099.

[26] G. Barlacchi et al., "A multi-source dataset of urban life in the city of Milan and the province of Trentino," *Sci. Data*, vol. 2, 2015, Art. no. 150055.

[27] Y. Jiang, J. Konečný, K. Rush, and S. Kannan, "Improving federated learning personalization via model agnostic meta learning," 2019, *arXiv:1909.12488*.

[28] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 3557–3568.

[29] C. T. Dinh, N. Tran, and J. Nguyen, "Personalized federated learning with moreau envelopes," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 21394–21405.

[30] T. Li, S. Hu, A. Beirami, and V. Smith, "Ditto: Fair and robust federated learning through personalization," in *Proc. 38th Int. Conf. Mach. Learn.*, 2021, pp. 6357–6368.

[31] Y. Deng, M. M. Kamani, and M. Mahdavi, "Adaptive personalized federated learning," 2020, *arXiv:2003.13461*.

[32] S.-J. Hahn, M. Jeong, and J. Lee, "Connecting low-loss subspace for personalized federated learning," in *Proc. 28th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2022, pp. 505–515.

[33] K. Wang et al., "Federated evaluation of on-device personalization," 2019, *arXiv:1910.10252*.

[34] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, 1655–1661.

[35] D. Li, F. Jiang, M. Chen, and T. Qian, "Multi-step-ahead wind speed forecasting based on a hybrid decomposition method and temporal convolutional networks," *Energy*, vol. 238, 2022, Art. no. 121981.

[36] A. Mahmud and A. Mohammed, "A survey on deep learning for time-series forecasting," 2021, *arXiv:2004.13408*.

[37] H.-W. Kim, J.-H. Lee, Y.-H. Choi, Y.-U. Chung, and H. Lee, "Dynamic bandwidth provisioning using arima-based traffic forecasting for mobile WiMAX," *Comput. Commun.*, vol. 34, no. 1, pp. 99–106, 2011.

[38] Y. Yu, J. Wang, M. Song, and J. Song, "Network traffic prediction and result analysis based on seasonal ARIMA and correlation coefficient," in *Proc. Int. Conf. Intell. Syst. Des. Eng. Appl.*, 2010, pp. 980–983.

[39] G. Cheng, J. Gong, and W. Ding, "A time-series decomposed model of network traffic," in *Proc. 1st Int. Conf. Adv. Natural Comput.*, 2005, pp. 338–345.

[40] T. Quang, H. Li, and T. Quang, "Cellular network traffic prediction using exponential smoothing methods," *J. Inf. Commun. Technol.*, vol. 18, pp. 1–18, 2019.

[41] T. T. Quang, L. Hao, and Q. K. Trinh, "A comprehensive research on exponential smoothing methods in modeling and forecasting cellular traffic," *Concurrency Comput., Pract. Experience*, vol. 32, 2020, Art. no. e5602.

[42] H. D. Trinh, L. Giupponi, and P. Dini, "Mobile traffic prediction from raw data using LSTM networks," in *Proc. IEEE 29th Annu. Int. Symp. Pers. Indoor Mobile Radio Commun.*, 2018, pp. 1827–1832.

[43] S. Zhan et al., "Cell traffic prediction based on convolutional neural network for software-defined ultra-dense visible light communication networks," *Secur. Commun. Netw.*, vol. 2021, pp. 1–10, 2021.

[44] Z. Chen, E. Jiaze, X. Zhang, H. Sheng, and X. Cheng, "Multi-task time series forecasting with shared attention," in *Proc. IEEE 2020 Int. Conf. Data Mining Workshops*, 2020, pp. 917–925.

[45] W. Jiang, "Cellular traffic prediction with machine learning: A survey," *Expert Syst. Appl.*, vol. 201, pp. 117–163, 2022.

[46] C.-C. Kao, C.-W. Chang, C.-P. Cho, and J.-Y. Shun, "Deep learning and ensemble learning for traffic load prediction in real network," in *Proc. IEEE Eurasia Conf. IOT Commun. Eng.*, 2020, pp. 36–39.

[47] Y. Gao, M. Zhang, J. Chen, J. Han, D. Li, and R. Qiu, "Accurate load prediction algorithms assisted with machine learning for network traffic," in *Proc. 2021 Int. Wireless Commun. Mobile Comput.*, 2021, pp. 1683–1688.

[48] Q. He, A. Moayyedi, G. Dán, G. P. Koudouridis, and P. Tengkvist, "A meta-learning scheme for adaptive short-term network traffic prediction," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 10, pp. 2271–2283, Oct. 2020.

[49] P. Kairouz, et al. "Advances and open problems in federated learning," 2019, *arXiv:1912.04977*.

[50] Y. Deng et al., "TailorFL: Dual-personalized federated learning under system and data heterogeneity," in *Proc. 20th ACM Conf. Embedded Netw. Sensor Syst.*, 2023, pp. 592–606.

[51] X. Li, M. Liu, S. Sun, Y. Wang, H. Jiang, and X. Jiang, "FedTrip: A resource-efficient federated learning method with triplet regularization," in *Proc. IEEE Int. Parallel Distrib. Process. Symp.*, 2023, pp. 809–819.

[52] Z. Wang, H. Xu, Y. Xu, Z. Jiang, J. Liu, and S. Chen, "FAST: Enhancing federated learning through adaptive data sampling and local training," *IEEE Trans. Parallel Distrib. Syst.*, vol. 35, no. 02, pp. 221–236, Feb. 2024.

[53] A. Z. Tan, H. Yu, L. Cui, and Q. Yang, "Towards personalized federated learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 12, pp. 9587–9603, Dec. 2023.

[54] T. Yu, E. Bagdasaryan, and V. Shmatikov, "Salvaging federated learning by local adaptation," 2020, *arXiv:2002.04758*.

[55] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 1126–1135.

[56] S. Yue, J. Ren, J. Xin, D. Zhang, Y. Zhang, and W. Zhuang, "Efficient federated meta-learning over multi-access wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 5, pp. 1556–1570, May 2022.

[57] V. Smith, C.-K. Chiang, M. Sanjabi, and A. Talwalkar, "Federated multi-task learning," 2017, *arXiv:1705.10467*.

[58] P. P. Liang, T. Liu, L. Ziyin, R. Salakhutdinov, and L.-P. Morency, "Think locally, act globally: Federated learning with local and global representations," 2020, *arXiv:2001.01523*.

[59] J. Zhang et al., "FedALA: Adaptive local aggregation for personalized federated learning," in *Proc. AAAI Conf. Artif. Intell.*, 2023, pp. 11237–11244.

[60] X. Wu, J. Niu, X. Liu, T. Ren, Z. Huang, and Z. Li, "pFedGF: Enabling personalized federated learning via gradient fusion," in *Proc. IEEE Int. Parallel Distrib. Process. Symp.*, 2022, pp. 639–649.

[61] M. Zhang, K. Sapra, S. Fidler, S. Yeung, and J. M. Alvarez, "Personalized federated learning with first order model optimization," 2020, *arXiv:2012.08565*.

[62] C. Zhang, H. Zhang, D. Yuan, and M. Zhang, "Citywide cellular traffic prediction based on densely connected convolutional neural networks," *IEEE Commun. Lett.*, vol. 22, no. 8, pp. 1656–1659, Aug. 2018.

[63] C. C. Holt, "Forecasting seasonals and trends by exponentially weighted moving averages," *Int. J. Forecasting*, vol. 20, no. 1, pp. 5–10, 2004.

[64] D. J. C. MacKay, *Information Theory, Inference & Learning Algorithms*. New York, NY, USA: Cambridge University Press, 2002.

[65] C. Zhang and P. Patras, "Long-term mobile traffic forecasting using deep spatio-temporal neural networks," in *Proc. 18th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2018, pp. 231–240.

[66] P. R. Winters, "Forecasting sales by exponentially weighted moving averages," *Manage. Sci.*, vol. 6, no. 3, pp. 324–342, 1960.

[67] E. Gladin, M. Alkousa, and A. Gasnikov, "Solving convex min-min problems with smoothness and strong convexity in one group of variables and low dimension in the other," *Autom Remote Control*, vol 82, pp. 1679–1691, 2021.

[68] E. Gladin et al., "Solving smooth min-min and min-max problems by mixed oracle algorithms," in *Proc. Math. Optim. Theory Operations Res. Recent Trends*, 2021, pp. 19–40.

[69] R. Bassily, M. Belkin, and S. Ma, "On exponential convergence of SGD in non-convex over-parametrized learning," 2018, *arXiv:1811.02564*.

[70] Y. Lei, T. Hu, G. Li, and K. Tang, "Stochastic gradient descent for nonconvex learning without bounded gradient assumptions," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 10, pp. 4394–4400, Oct. 2020.

[71] C. Liu, L. Zhu, and M. Belkin, "Loss landscapes and optimization in over-parameterized non-linear systems and neural networks," *Appl. Comput. Harmon. Anal.*, vol. 59, pp. 85–116, 2022.

[72] S. P. Maralappanavar, P. Khanduri, and B. N. Bharath, "Linear convergence of decentralized fedavg for PL objectives: The interpolation regime," *Trans. Mach. Learn. Res.*, 2025.

[73] V. Mothukuri, R. M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha, and G. Srivastava, "A survey on security and privacy of federated learning," *Future Gener. Comput. Syst.*, vol. 115, pp. 619–640, 2021.

[74] A. Salem, Y. Zhang, M. Humbert, M. Fritz, and M. Backes, "ML-Leaks: Model and data independent membership inference attacks and defenses on machine learning models," in *Proc. 26th Annu. Netw. Distrib. Syst. Secur. Symp.*, 2019.

[75] S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha, "Privacy risk in machine learning: Analyzing the connection to overfitting," in *Proc. IEEE 31st Comput. Secur. Found. Symp.*, 2018, pp. 268–282.

[76] K. Leino and M. Fredrikson, "Stolen memories: Leveraging model memorization for calibrated white-box membership inference," in *Proc. 29th USENIX Conf. Secur. Symp.*, 2020, pp. 1605–1622.

[77] L. Hu et al., "Defenses to membership inference attacks: A survey," *ACM Comput. Surv.*, vol. 56, no. 4, pp. 1–34 2023.

[78] X. Luo and X. Zhu, "Exploiting defenses against GAN-based feature inference attacks in federated learning," 2020, *arXiv:2004.12571*.

[79] K. Wei et al., "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Trans. Inf. Forensics Secur.*, vol. 15, pp. 3454–3469, 2020.

[80] C. Zhang, S. Li, J. Xia, W. Wang, F. Yan, and Y. Liu, “BatchCrypt: Efficient homomorphic encryption for cross-silo federated learning,” in *Proc. USENIX Annu. Tech. Conf.*, 2020, pp. 493–506.

[81] J. So, B. Güler, and A. S. Avestimehr, “Turbo-Aggregate: Breaking the quadratic aggregation barrier in secure federated learning,” *IEEE J. Sel. Areas Inf. Theory*, vol. 2, no. 1, pp. 479–489, Mar. 2021.

[82] B. Yu, H. Yin, and Z. Zhu, “Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting,” in *Proc. 27th Int. Joint Conf. Artif. Intell.*, 2018, pp. 3634–3640.

[83] X. zhang, X. Chen, H. Tang, Y. Wu, H. Shen, and J. Li, “AdpST-GCN: Adaptive spatial–temporal graph convolutional network for traffic forecasting,”*Knowl.-Based Syst.*, vol. 301, 2024, Art. no. 112295.

[84] S. Ji, S. Pan, G. Long, X. Li, J. Jiang, and Z. Huang, “Learning private neural language modeling with attentive aggregation,” in *Proc. Int. Joint Conf. Neural Netw.*, 2019, pp. 1–8.

[85] T. Li et al., “Federated optimization in heterogeneous networks,” in *Proc. Mach. Learn. Syst.*, vol. 2, pp. 429–450, 2020.

[86] opensignal.com, “USA mobile network experience reportJan. 2024,” 2024. [Online]. Available: https://www.opensignal.com/reports/2024/01/usa/mobile-network-experience

[87] A. Vaswani et al., “Attention is all you need,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6000–6010.

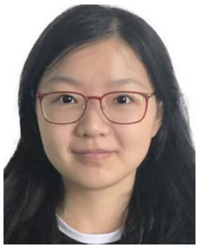
[88] Y. Liu et al., “iTransformer: Inverted transformers are effective for time series forecasting,” in *Proc. Int. Conf. Learn. Representations*, 2024.

**Jingjing Xue** received the BS degree from the School of Computer & Communication Engineering, University of Science and Technology Beijing, China, in 2020. Since 2020, She is currently working toward the PhD degree with the Networking Technology Research Centre, Institute of Computing Technology, Chinese Academy of Sciences. Her current research interests include federated learning and edge intelligence.

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

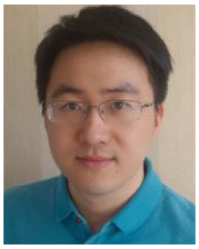
**Min Liu** (Senior Member, IEEE) received the BS and MS degrees in computer science from Xi’an Jiaotong University, China, in 1999 and 2002, respectively and the PhD degree in computer science from the Graduate University of the Chinese Academy of Sciences in 2008. She is currently a professor with the Networking Technology Research Centre, Institute of Computing Technology, Chinese Academy of Sciences. Her current research interests include mobile computing and edge intelligence.

**Yuwei Wang** (Member, IEEE) received the PhD degree in computer science from the University of Chinese Academy of Sciences, Beijing, China. He is currently a senior engineer (equivalent to associate professor) with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing. His current research interests include federated learning, mobile edge computing, and next-generation network architecture.



**Xuying Meng** received the BS degree from Wuhan University in 2013 and the PhD degree from the University of Chinese Academy of Sciences in 2018. She is currently an associate professor with the Institute of Computing Technology, Chinese Academy of Sciences. She has published innovative works in top conference proceedings. Her current research interests include data mining and security protection of network services.

**Jingyuan wang** received the PhD degree from the Department of Computer Science and Technology, Tsinghua University. He is currently a professor with the School of Computer Science and Engineering, and School of Economics and Management, Beihang University. He is the head of Beihang Interest Group on SmartCity (BIGSCity). His general area of research is data mining and machine learning, with special interests in smart cities and spatiotemporal data analytics. He is also the recipient of NSFC Distinguished Young Scholars.



**Junbo Zhang** (Senior Member, IEEE) is a senior researcher of JD Intelligent Cities Research. He is leading the Urban AI Product Department of JD iCity with JD Technology, as well as AI Lab of JD Intelligent Cities Research. Prior to that, he worked with Micorsoft Research Asia from 2015 to 2018. He has published more than 60 research papers in spatio-temporal data mining and AI, urban computing, deep learning, and federated learning. He serves as an associate editor of *ACM Transactions on Intelligent Systems and Technology*. He received a number of honors, including the Second Prize of the Natural Science Award of the Ministry of Education (2021), the 22nd and 23 rd China Patent Excellence Award (2021, 2022). He is an ACM/CCF senior member.

**Ke Xue** (Fellow, IEEE) received the PhD degree from the Department of Computer Science and Technology, Tsinghua University, Beijing, China. He serves as a full professor with Tsinghua University. He has published more than 200 technical papers and holds 11 U.S. patents in the research areas of next-generation internet, blockchain systems, the Internet of Things, and network security. He is a member of ACM. He served as the Steering Committee chair for IEEE/ACM IWQoS. He has guestedited several special issues in IEEE and Springer journals. He is the editor of *IEEE Internet of Things Journal*.