Node-Level Federated Learning with Adaptive Personalized Aggregation for Spatio-Temporal Traffic Prediction

Xiaoying Tu
East China Normal University
Shanghai, China
51275901130@stu.ecnu.edu.cn

Ying Lin
East China Normal University
Shanghai, China
linying_320@163.com

Xingjian Lu*
East China Normal University
Shanghai, China
xjlu@cs.ecnu.edu.cn

Yibing Wang
East China Normal University
Shanghai, China
51265901112@stu.ecnu.edu.cn

Bo Hu
East China Normal University
Shanghai, China
51265901115@stu.ecnu.edu.cn

ABSTRACT

Accurate and real-time traffic flow prediction is crucial for Intelligent Transportation Systems. Recent advances in federated learning and spatio-temporal modeling have improved accuracy and privacy protection. However, existing methods often rely on global topology for spatial features, neglecting topology protection, and typically train a generic global model without considering local personalized features, limiting prediction performance. This paper proposes ST-PFLA (Spatio-Temporal Traffic Flow Prediction via Personalized Federated Learning with Adaptive Aggregation), a framework designed for node-level scenarios where clients only have information about their respective connections, to improve prediction accuracy and training efficiency while safeguarding topology privacy. In ST-PFLA, clients conduct prediction by combining spatial and temporal features extracted by the attention mechanism and local datasets respectively. The method aggregates only encoders across clients, retaining decoders locally for personalization. Each client performs an additional local training round to generate a guide model, which is used to inform the calculation of aggregation weights. Experimental results on two public datasets show that ST-PFLA can significantly enhance prediction accuracy while safeguarding topology privacy at lower training costs.

KEYWORDS

Federated Learning, Traffic Flow Prediction, Spatio-Temporal Prediction

ACM Reference Format:
Xiaoying Tu, Ying Lin, Xingjian Lu*, Yibing Wang, and Bo Hu. 2026. Node-Level Federated Learning with Adaptive Personalized Aggregation for Spatio-Temporal Traffic Prediction. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 9 pages. https://doi.org/10.65109/UUWS7804

1 INTRODUCTION

Traffic flow prediction, as a critical component of Intelligent Transportation Systems (ITS), provides essential guidance for traffic management and route planning [25]. It plays a vital role in urban traffic control, autonomous driving, intelligent travel planning, and infrastructure development [24]. Given the inherent temporal and spatial correlations in traffic data, many studies focus on extracting and integrating spatio-temporal features to improve prediction accuracy. In light of growing privacy concerns in real-world applications, a number of studies advocate keeping traffic data decentralized [15, 21]. Leveraging Federated Learning (FL), multiple clients with local datasets can collaboratively train a global model without sharing raw data, thereby improving predictive performance while safeguarding privacy. However, existing FL-based spatio-temporal traffic flow prediction methods still face several critical challenges.

Firstly, many existing approaches rely on global sensor topologies and graph-based neural networks for spatial feature extraction [9, 20, 30], which incurs high computational and memory overhead. Moreover, the dynamic nature of sensor networks requires frequent topology updates and model retraining [40], leading to reduced stability and limited scalability in practical deployments.

Secondly, assuming full knowledge of the global topology raises significant privacy concerns. Since sensors are often owned by different organizations, exposing global topology information may reveal sensitive details such as sensor density, deployment patterns, and coverage, leading to privacy breaches and potential security risks [15, 39].

Finally, most existing methods [5, 36] adopt a single global model aggregated via simple averaging, ignoring client heterogeneity. In practice, traffic patterns vary considerably across regions, leading to non-IID data distributions across clients. Consequently, a single global model often fails to capture local variations and suffers from degraded generalization performance. Although personalized federated learning alleviates this issue to some extent, existing approaches still struggle with effective spatio-temporal modeling and adaptive aggregation, often relying on heuristic strategies or explicit topologies, while incurring substantial computational and communication overheads that hinder scalability in practical deployments.

To address the above challenges, we propose ST-PFLA (Spatio-Temporal traffic flow prediction based on Personalized Federated Learning with Adaptive aggregation), a novel framework designed

---

Corresponding author: Xingjian Lu.

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/UUWS7804

for node-level scenarios. ST-PFLA preserves topology privacy by restricting access to global structural information and is highly applicable in real-world traffic systems. Our goal is to improve prediction accuracy while protecting topology privacy, reducing the negative impact of naive averaging in federated aggregation, and minimizing both communication and computation overhead during training. To the best of our knowledge, this is the first work to apply personalized federated learning to node-level spatio-temporal traffic flow prediction, integrating adaptive encoder aggregation with topology privacy preservation to enhance personalization and training efficiency. The main contributions of this paper are summarized as follows:

- Building on the approach in [15], we introduce an attention-based spatial feature extraction module. Structural embeddings are initialized via a customized random walk algorithm and dynamically updated based on each client's local knowledge. Experimental results demonstrate that the proposed embeddings improve prediction accuracy, while the dynamic update mechanism yields additional performance gains.
- We propose a personalized FL strategy tailored for Non-IID scenarios, where only encoders are uploaded for aggregation, while decoders remain local for customization. Aggregation weights are computed based on gradient similarity between guide models, enabling adaptive and personalized model updates that better reflect client-specific characteristics.
- Extensive experiments on two real-world traffic datasets demonstrate that ST-PFLA achieves superior performance in terms of accuracy, efficiency, and privacy preservation, outperforming all baselines with significantly lower training costs.

2 RELATED WORK

2.1 Spatio-Temporal Traffic Flow Prediction

Traditional traffic flow prediction methods, such as ARIMA, Kalman filters, and SVMs, primarily capture temporal trends but overlook spatial correlations within road networks. Deep learning models, particularly LSTM, enhance temporal feature extraction by learning complex and nonlinear time dependencies through gated recurrent mechanisms. However, these models mainly operate on sequential data and thus struggle to capture spatial interactions across different locations, limiting their effectiveness in spatio-temporal contexts. To address this, recent work integrates spatial and temporal learning. Models like STGCN [32] and DCRNN [14] combine GCN-based spatial modeling with temporal components, while T-GCN [38] jointly captures spatio-temporal dependencies. However, these methods rely on static adjacency matrices, limiting adaptability. Advances such as Graph WaveNet [26] and STSGCN [23] introduce learnable graphs and synchronous mechanisms, and dynamic approaches like DMLSA [34] and LOGO [3] further enhance accuracy. Despite these developments, most approaches assume centralized data, limiting their applicability in distributed scenarios. Federated learning (FL) has emerged to address privacy and communication challenges, including frameworks such as FedAvg-based GRU, FASTGNN [33], and FedGCN [8], which integrate differential privacy and adaptive graph learning. CNFGNN [18] further encodes graph structures using GNNs and employs alternating optimization to reduce communication costs. More recently, FedGODE [1] combines federated learning with spatio-temporal graph ordinary differential equation networks to capture continuous-time traffic dynamics under privacy constraints. Nevertheless, these methods typically train a single global model and rely on complete topology knowledge, thus leaving personalization and topology privacy largely unaddressed — challenges our work aims to overcome.

Existing federated approaches for spatio-temporal traffic prediction often rely on global topologies and single global models, leading to high computational overhead, topology privacy risks, and limited adaptability under non-IID data. To address these issues, we propose ST-PFLA, a personalized federated framework that enables privacy-preserving spatial modeling without global topology knowledge and improves prediction accuracy through personalized learning.

2.2 Personalized Federated Learning

Traditional federated learning typically constructs a global model by aggregating updates from all participating clients. However, such an approach often performs poorly under highly heterogeneous (Non-IID) data, since it cannot adequately capture client-specific characteristics. To address this challenge, personalized federated learning has emerged, aiming to enhance local performance by generating client-specific models. Existing PFL approaches can be broadly categorized as follows: **a) Optimization-based methods.** Ditto [10] and PerAda [27] employ regularization terms to constrain the divergence between local and global models. **b) Transfer learning-based methods.** Methods such as pFedCSPC [19], FedAFK [31], and pFedAMF [29] treat the global model as a source model and transfer its knowledge to local models for personalization. **c) Model aggregation-based methods.** CPPer-FL [35] clusters clients for aggregation, while FedCAP [13], and FedASA [4] adjust aggregation weights to improve model performance. **d) Meta learning-based methods.** FedFGCR [37] employs meta-learning to learn better initial shared models for each client. **e) Parameter decoupling-based methods.** Approaches such as pFedHN [22], FedAS [28], FediOS [6], and FedRoD [2] decouple the shared and personalized parts of the model, balancing knowledge sharing and local adaptation.

Despite progress in personalized federated learning, existing methods are not well suited for spatio-temporal traffic prediction, where dynamic spatial dependencies and topology privacy are critical. To address these challenges, we propose ST-PFLA, a node-level personalized federated framework that balances global knowledge sharing and local specialization through adaptive encoder aggregation, while preserving topology privacy without requiring global adjacency information.

3 METHODOLOGY

3.1 Notations and Preliminary

By integrating spatio-temporal traffic flow prediction with personalized federated learning, our objective is to generate a personalized model for each client that adapts to the local data distribution, enabling effective prediction using spatio-temporal features at the

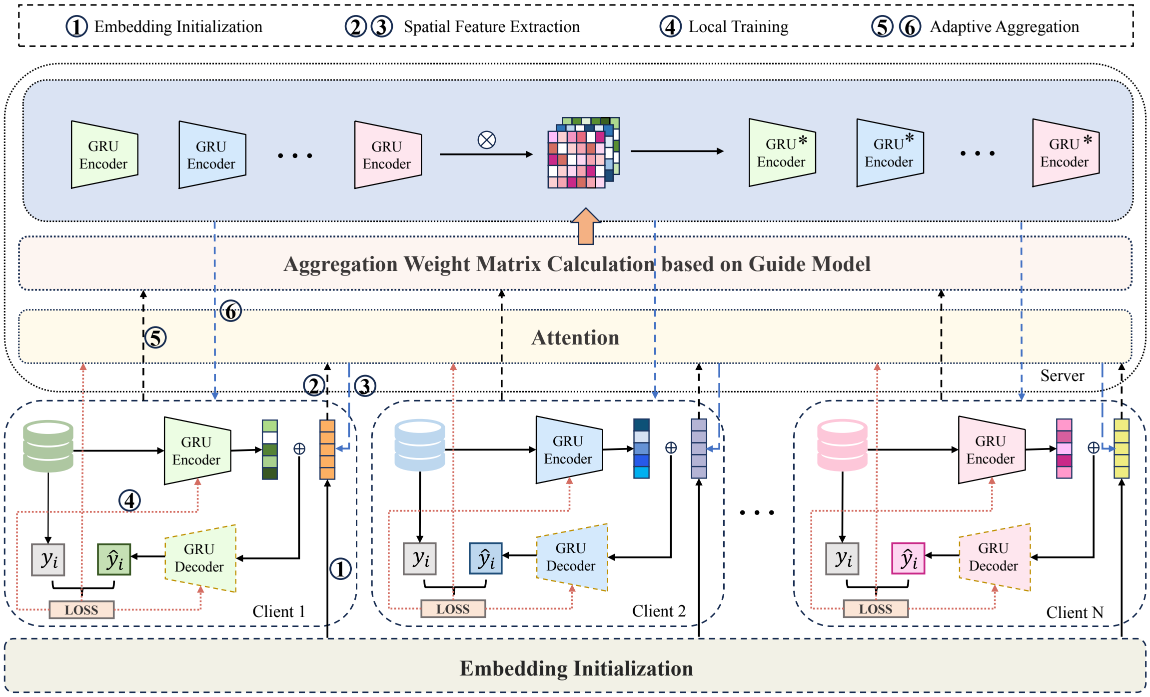


Figure 1: Overview of the ST-PFLA framework. Clients share noise-protected structural embeddings for topology privacy, while the server refines them via self-attention and performs adaptive encoder aggregation. Local decoders remain client-specific to enable personalized spatio-temporal prediction.

client side. Suppose there are N clients, each with a non-IID local dataset denoted as $D_i = \{(x_i^j, y_i^j)\}_{j=1}^{|D_i|}$, where $1 \le i \le N$ and $1 \le j \le |D_i|$. Here, $|D_i|$ represents the number of samples for client i and j is the sample index. The input data is represented as $x_i^j \in \mathbb{R}^{m \times P}$, where m and P denote the number of time steps and the feature dimension, respectively. The corresponding output is $y_i^j \in \mathbb{R}^{n \times P}$, where n indicates the number of future time steps to be predicted. The total amount of data across all clients is given by $|D| = \sum_{i=1}^{N} |D_i|$. The local model for each client i can be expressed as M_i: $\mathbb{R}^{m \times P} \to \mathbb{R}^{n \times P}$, which predicts traffic flow for the next n time steps based on the traffic data of the previous m time steps. The model M_i consists of two components: an encoder M_i^{enc} and a decoder M_i^{dec}, parameterized by θ_i^{enc} and θ_i^{dec}, respectively. Thus, the model parameters for client i are denoted as $\theta_i = \{\theta_i^{enc}, \theta_i^{dec}\}$. To capture spatial information of each client, we introduce structural embeddings E_i, which encode spatial characteristics without explicit global topology knowledge. These embeddings are dynamically updated during training via an attention mechanism. The spatio-temporal traffic flow prediction task executed by client i can therefore be formulated as:

$$\begin{aligned}\hat{y}_i &= M_i(\theta_i; x_i, E_i)\\ &= M_i^{dec}(\theta_i^{dec}; M_i^{enc}(\theta_i^{enc}; x_i), E_i)\end{aligned} \tag{1}$$

The optimization objective of personalized federated learning is:

$$\min_{\Theta} \sum_{i=1}^{N} \frac{|D_i|}{|D|} \mathcal{L}_i(\Theta_i) \tag{2}$$

where $\mathcal{L}_i(\theta_i) = \frac{1}{|D_i|} \sum_{j=1}^{|D_i|} l(\theta_i; x_i^j, y_i^j, E_i)$. Here, $\Theta = \{\theta_i\}_{i=1}^{N}$ represents the set of personalized parameters for all clients. $\mathcal{L}_i(\theta_i)$ denotes the average loss over all samples on client i, while l evaluates the discrepancy between a single sample and its predicted value. In this work, l is defined as the Mean Squared Error (MSE) Loss: $l = (M_i(\theta_i; x_i^j, E_i) - y_i^j)^2$.

3.2 Proposed Method

This section provides an overview of the proposed ST-PFLA framework, organized into five subsections. We first introduce the framework and its six-step workflow, followed by detailed explanations of four core components: (1) Embedding Initialization, (2) Attention-Based Spatial Feature Extraction, (3) Adaptive Aggregation, and (4) Federated Training. These components collaboratively support personalized spatio-temporal prediction in federated settings, preserving topology privacy and enabling adaptive aggregation.

3.2.1 ST-PFLA Framework. Figure 1 illustrates the ST-PFLA framework, consisting of three key modules: Embedding Initialization, Attention-Based Spatial Feature Extraction, and Adaptive Aggregation with Guide Models. The workflow follows six sequential steps, denoted by circled numerals in the figure. Step ① represents embedding initialization, where each client generates its initial embedding using local topology and a customized random walk. Steps ② and ③ involve spatial feature extraction, where clients upload embeddings to the server and receive dynamically updated embeddings via self-attention. Step ④ involves local client training and gradient upload, bridging the spatial feature extraction and

Algorithm 1 ST-PFLA Framework Workflow

```
Input: Initial local models parameters $\{\theta_i^{(0)}\}_{i=1}^N$, initial Attention model parameters $\theta_{att}^{(0)}$, structural embedding extraction module $\mathcal{M}_{struct}$, learning rate $\eta$, total communication rounds $T$, local training epochs $K$.
Output: Personalized local models $\{\theta_i^{(T)}\}_{i=1}^N$.
Initialize structural embedding $E^{(0)}$ using $\mathcal{M}_{struct}$.
Server executes
for each communication round $t = 1, \dots, T$ do
    Receive embeddings $E^{(t-1)}$ from all clients.
    Calculate $E^{(t)}$ via (3).
    Distributes $E^{(t)}$ back to all clients.
    for each client $i \in \{1, \dots, N\}$ do
        $\widetilde{\theta}_{i,enc}^{(t)}, \nabla_{\theta_{i,\text{guide}}^{(t)}} \ell, \nabla_{E_i^{(t)}} \ell \leftarrow ClientUpdate(\theta_i^{(t-1)}; D_i, E_i^{(t)})$
    end for
    Update $\theta_{att}^{(t-1)}$ via (10).
    Calculate similarity matrix $C^{(t)}$ via (13).
    Calculate aggregation weights $A^{(t)}$ via (15).
    Aggregate local encoders via (16).
    Distribute $\theta_{i,enc}^{(t)}$ to all clients.
end for
return Personalized local models $\{\theta_i^{(T)}\}_{i=1}^N$.
ClientUpdate($\theta_i^{(t-1)}; D_i, E_i^{(t)}$)
Client $i$ uploads $E_i^{(t-1)}$ to the Server.
Client $i$ receives $E_i^{(t)}$ from the Server.
for each training epoch $e = 1, \dots, K$ do
    for minibatch $\xi_j \in D_i$ do
        Update model parameters $\theta_i^{(t-1)}$ via (11).
    end for
end for
for minibatch $\xi_j \in D_i$ do
    Update model parameters $\widetilde{\theta}_i^{(t)}$ via (12).
end for
return $\widetilde{\theta}_{i,enc}^{(t)}$, $\nabla_{\theta_{i,\text{guide}}^{(t)}} \ell$, and $\nabla_{E_i^{(t)}} \ell$. =0
```

aggregation processes. Finally, Steps ⑤ and ⑥ involve adaptive aggregation, where each client constructs a guide model to compute personalized aggregation weights for encoder updates. The detailed steps are as follows:

① **Initialization**: Each client's structural embedding is initialized using the spatial feature extraction module from [15].

② **Embedding Upload**: Clients upload their structural embeddings to the server. During the first upload, Gaussian noise is added to ensure topology privacy.

③ **Embedding Update**: The server employs a self-attention model to update all clients' embeddings and distributes the updated embeddings back to the clients.

④ **Local Training and Gradient Upload**: Clients perform local training, updating their encoders and decoders. They then upload the gradients corresponding to the embeddings to the server, which uses these gradients to update the attention model.

⑤ **Guide Model and Weight Calculation**: Each client performs an additional local training round to obtain a guide model. The gradients of the guide model and the updated encoder parameters are uploaded to the server. The server then computes aggregation weights based on the gradient similarity between guide models.

⑥ **Encoder Aggregation**: The server aggregates the encoders of all clients using the computed aggregation weights and distributes the updated encoders to the clients for the next round.

Through these six steps, the framework can generate personalized spatio-temporal prediction models tailored to each client's local data distribution. Detailed discussions on the three key modules mentioned above are provided in the following.

3.2.2 *Embedding Initialization.* In spatio-temporal traffic flow prediction, both temporal and spatial features are critical for accurate forecasting. Temporal features are extracted from local sequences, while spatial features are usually derived from topology information. Many existing methods assume complete knowledge of the global topology and use GNNs to learn structural embeddings; however, this assumption compromises topology privacy.

To address this, Lin et al.[15] proposed a customized random walk algorithm for spatial feature extraction in node-level scenarios where global topology is unavailable. Each client performs iterative random walks based on local connectivity to record sequences of node indices and corresponding embeddings. At each step, the next-hop node is probabilistically chosen based on neighbor weights until the sequence reaches a predefined length. Upon receiving the embedding sequences, the server applies the SkipGram model to update embeddings of all involved nodes. The updated results are then returned along the original communication paths. By repeating this process, clients gradually refine their structural embeddings while preserving privacy.

Although this method captures structural characteristics, the embeddings stay static during training. To overcome this limitation, we use the module from [15] for initialization and introduce an attention-based mechanism to dynamically update embeddings during training.

3.2.3 *Attention-Based Spatial Feature Extraction.* To improve the expressiveness of structural embeddings, we introduce a dynamic update mechanism based on a self-attention model deployed at the server. In each communication round, the server collects embeddings from all clients and processes them with the self-attention model. The updated embeddings are then returned to the respective clients. To prevent the server from inferring the global topology through cross-client embedding similarity, we add Gaussian noise to embeddings before their initial upload. This hides the original embeddings while preserving their utility for training. The structural embedding for client i can be updated as:

$$E_i^{(t)} = f_{self-attention}(\theta_{att}^{(t-1)}; E_i^{(t-1)}) \tag{3}$$

where $E_i^{(t)}$ is the embedding for client i at epoch t, $f_{self-attention}(\cdot)$ is the self-attention model on the server, and θ_{att} denotes the model parameters. The processing steps of the function $f_{self-attention}(\cdot)$ are as follows. First, all clients' structural embeddings undergo linear transformations to obtain queries, keys, and values:

$$Q^{(t)} = W_Q E^{(t-1)} \tag{4}$$

$$K^{(t)} = W_K E^{(t-1)} \tag{5}$$

$$V^{(t)} = W_V E^{(t-1)} \tag{6}$$

Then, attention scores $\alpha_{ij}^{(t)}$ are computed by measuring the similarity between queries and keys and normalized with a Softmax function:

$$\alpha_{ij}^{(t)} = \frac{\exp\left(\frac{Q_i^{(t)} K_j^{(t)T}}{\sqrt{d_k}}\right)}{\sum_{p=1}^{N} \exp\left(\frac{Q_i^{(t)} K_p^{(t)T}}{\sqrt{d_k}}\right)} \tag{7}$$

where d_k is the dimension of the key vectors and serves as a scaling factor to prevent gradient vanishing or explosion. $\alpha_{ij}^{(t)}$ represents the attention weight of client i's query on client j. After obtaining the attention scores for client i, a weighted sum of all clients' values is calculated to get client i's new structural embedding. This embedding then passes through a linear transformation layer followed by a nonlinear activation function to enhance feature expressiveness, yielding the final updated structural embedding:

$$\tilde{E}_i^{(t)} = \sum_{j=1}^{N} \alpha_{ij}^{(t)} V_j^{(t)} \tag{8}$$

$$E_i^{(t)} = \sigma\left(W_O \tilde{E}_i^{(t)} + b_O\right) \tag{9}$$

where $\tilde{E}_i^{(t)}$ is the intermediate embedding, W_O and b_O are the output layer's weight matrix and bias term respectively, and $\sigma(\cdot)$ denotes the nonlinear activation function $ReLU(\cdot)$.

The attention model on the server is dynamically updated based on the local knowledge of the clients. During each communication round, clients use the updated E_i for local training and compute $\nabla_{E_i} l$ using the chain rule. This gradient is then uploaded to the server to update the self-attention model:

$$\theta_{att}^{(t)} = \theta_{att}^{(t-1)} - \eta \nabla_{\theta_{att}^{(t-1)}} l \tag{10}$$

where η is the learning rate, and $\nabla_{\theta_{att}^{(t-1)}} l$ represents the parameter gradient which is calculated using $\nabla_{E_i} l$.

3.2.4 *Adaptive Aggregation Based on Guide Model.*

Our goal is to generate a personalized model for each client that aligns with its local data distribution. The local model on each client consists of two components: an encoder and a decoder. The encoder is responsible for extracting temporal features from the input data, while the decoder leverages both spatial and temporal features to generate predictions. Intuitively, the knowledge learned by the encoder is more general and transferable, whereas the knowledge learned by the decoder needs to be tailored to the local data distribution, making it more specific. We hypothesize that sharing the encoder across clients allows it to benefit from a larger pool of samples, thereby enhancing its feature extraction capabilities. In contrast, sharing the decoder may weaken its alignment with the local data distribution, ultimately degrading prediction performance. To address this, we adopt a personalized federated learning approach that aggregates only the encoder while keeping the decoder localized to each client. This strategy strikes a balance between leveraging global knowledge and preserving local adaptability.

Considering that the knowledge learned by each client's encoder is also related to its local data distribution, and inspired by [16], we propose an adaptive aggregation mechanism. This mechanism generates aggregation weights based on the similarity between the guide models of clients. By having each client perform an additional training round after completing local training, the resulting guide model incorporates updated local knowledge for the next round, thereby effectively improving the efficiency of model aggregation. The guide model is obtained via two consecutive gradient descent steps:

$$\widetilde{\theta}_i^{(t)} = \theta_i^{(t-1)} - \eta \nabla_{\theta_i^{(t-1)}} l \tag{11}$$

$$\theta_{i,guide}^{(t)} = \widetilde{\theta}_i^{(t)} - \eta \nabla_{\widetilde{\theta}_i^{(t)}} l \tag{12}$$

where $\widetilde{\theta}_i^{(t)}$ represents the model obtained by client i after local training in round t, $\theta_i^{(t-1)}$ is the local model of client i from round $t-1$, and $\theta_{i,guide}^{(t)}$ denotes the guide model of client i in round t. After this, the client uploads both $\widetilde{\theta}_{i,enc}^{(t)}$ and $\nabla_{\widetilde{\theta}_i^{(t)}} l$ to the server. The server then calculates the similarity matrix $C^{(t)}$ using the gradients, where each element $C_{ij}^{(t)}$ represents the similarity between client i and client j:

$$C_{ij}^{(t)} = \frac{\nabla_{\widetilde{\theta}_i^{(t)}} l \cdot \nabla_{\widetilde{\theta}_j^{(t)}} l}{\left\|\nabla_{\widetilde{\theta}_i^{(t)}}\right\| \left\|\nabla_{\widetilde{\theta}_j^{(t)}}\right\|} \tag{13}$$

Next, we apply linear shifting and normalization to $C^{(t)}$ to obtain the weight matrix $A^{(t)}$, where each element $A_{ij}^{(t)}$ represents the aggregation weight of client j for client i:

$$\widetilde{A}_{ij}^{(t)} = \frac{C_{ij}^{(t)} - \min_{1\le i,j\le N} C_{ij}^{(t)}}{\max_{1\le i,j\le N} C_{ij}^{(t)} - \min_{1\le i,j\le N} C_{ij}^{(t)}} \tag{14}$$

$$A_{ij}^{(t)} = \frac{\widetilde{A}_{ij}^{(t)}}{\sum_{j=1}^{N} \widetilde{A}_{ij}^{(t)}} \tag{15}$$

Finally, we use $A^{(t)}$ to perform personalized aggregation of the $\widetilde{\theta}_{enc}^{(t)}$ uploaded by all clients:

$$\theta_{enc}^{(t)} = \widetilde{\theta}_{enc}^{(t)} A^{(t)} \tag{16}$$

where $\theta_{i,enc}^{(t)} = \sum_{j=1}^{N} A_{ij}^{(t)} \times \theta_{j,enc}^{(t)}$.

3.2.5 *Federated Training.*

Algorithm 1 outlines the federated training procedure of the ST-PFLA framework, which consists of two main components: the server-side process (Lines 2–16) and the client-side process (Lines 17–28), corresponding to the six-step workflow described earlier. The Server procedure handles global operations such as embedding aggregation, attention model updates, and personalized encoder aggregation, while the Client procedure focuses on local model training, gradient computation, and interaction with the server.

More specifically, the server begins by updating the received client embeddings using the self-attention model and then distributes the updated embeddings back to the corresponding clients

(Lines 4–6). Upon receiving the updated embeddings, each client performs local model training in parallel based on its traffic flow dataset, optimizing both encoder and decoder parameters while computing gradients associated with the embeddings (Line 8). After completing the local training (Lines 20–24), clients conduct additional training to generate a guide model with more advanced knowledge (Lines 25–27), and subsequently upload their encoder parameters, guide model gradients, and embedding-related gradients to the server (Line 28). Once the server receives these gradients, it first updates the self-attention model using the embedding-related gradients to enhance its ability to refine structural embeddings (Line 10). Then, it measures the similarity between guide model gradients across clients and calculates personalized aggregation weights $A^{(t)}$ based on this similarity (Lines 11–12). These weights are used during the encoder aggregation phase to perform personalized weighted aggregation, effectively balancing global knowledge sharing with personalized feature retention and thereby improving the predictive performance (Line 13). This server–client interaction is iteratively performed over each communication round ($t = 1, \dots, T$), eventually converging to personalized local models $\{\theta_i^{(T)}\}_{i=1}^{N}$ for clients.

4 EXPERIMENTS

4.1 Experimental Setup

4.1.1 Datasets. To evaluate ST-PFLA under a rigorous node-level federated setting, we conduct experiments on two widely-used real-world traffic datasets, METR-LA and PEMS-BAY. We define each client as an individual traffic sensor node (i.e., one loop detector or sensor), which locally stores its own spatio-temporal traffic observations. Since the sensor deployments and collection periods are fixed in these real-world datasets, data are naturally partitioned by sensor and cannot be arbitrarily reassigned; thus, each sensor forms a standalone client with a fixed local dataset:

- **METR-LA**: This dataset contains traffic speed measurements collected by 207 loop detectors on Los Angeles County highways over a 4-month period, from March 1, 2012, to June 30, 2012.
- **PEMS-BAY**: This dataset consists of traffic speed data recorded by 325 sensors in the Bay Area, spanning from January 1, 2017, to May 31, 2017.

4.1.2 Baselines. To comprehensively evaluate the effectiveness of our method, we select the following baselines: **a) Local**: All clients train exclusively on their own local datasets. **b) Centralized Methods**: DCRNN[14], LSTM[7], and ST-PFLA (Centralized), trained on centralized datasets using DCRNN, LSTM, and GRU encoder-decoder architectures, respectively. **c) Federated Methods**: FedAvg[17], FedProx[11], and CNFGNN[18]. These methods aim to train a single global model shared across all clients. FedAvg performs simple averaging to aggregate local models, FedProx extends FedAvg by incorporating a regularization term to tackle client heterogeneity, and CNFGNN combines GNN and GRU to capture spatio-temporal dependencies for prediction. **d) Personalized Federated Methods**: Ditto[10], FedPHP[12] and ST-TPFL[15], which focus on generating personalized models for individual clients. Ditto introduces a hyperparameter λ to balance global and local objectives, FedPHP employs historical model ensembles for next-round training instead of direct overwriting, and ST-TPFL leverages static structural embeddings and performs personalized aggregation based on the similarity between these embeddings.

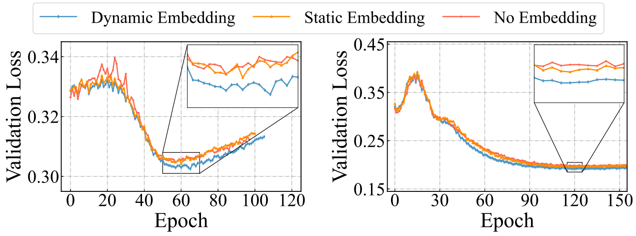
Table 1: Experimental results of baselines on two datasets.

Method	METR-LA		PEMS-BAY	
	RMSE	MAE	RMSE	MAE
Local	11.656	5.510	3.994	1.795
DCRNN	12.087	6.050	4.198	1.986
LSTM	11.729	5.983	4.129	1.919
ST-PFLA(C)	11.774	5.853	4.108	1.885
FedAvg	12.126	6.200	4.407	1.997
FedProx	12.129	6.318	5.182	2.849
CNFGNN	11.706	5.949	3.910	1.804
Ditto	11.642	5.574	3.936	1.875
FedPHP	11.789	5.865	3.970	1.898
ST-TPFL	11.885	5.972	4.282	1.951
ST-PFLA	**11.581**	**5.514**	**3.871**	**1.795**

4.1.3 Implementation Details. Traffic speed records are aggregated using a 5-minute window, and the aggregated data is divided into multiple sequences, each containing 24 time intervals. The prediction task involves using traffic speed data from the past 12 time intervals to forecast the speeds for the subsequent 12 time intervals. All baselines except DCRNN and LSTM employ a GRU-based encoder-decoder architecture as the local model for traffic flow prediction. The learning rate is set to 1×10^{-3}, and the Adam optimizer. In ST-PFLA (Centralized), the model comprises a single GRU layer with 100 hidden units, while other methods use a single GRU layer with 64 hidden units. For federated methods, the number of local training rounds is set to 1. The structural embedding dimension for each client is 64 in methods such as CNFGNN and ST-PFLA. Models are trained on an NVIDIA GeForce RTX 3090 GPU, equipped with an Intel Core i7 processor and 24GB of RAM. The results are averaged over five independent runs. Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) are used as evaluation metrics to measure prediction performance.

4.2 Experimental Results

4.2.1 Performance Comparison. We conduct extensive experiments on two datasets to evaluate our method alongside all baselines, with the results presented in Table 1. Bolded values denote the best results for each metric. It can be observed that our method outperforms others in all metrics. This superiority is attributed to the effective integration of temporal and spatial features for prediction, combined with a personalized aggregation strategy based on guide models, which generates client-specific models tailored to their local data distributions. For centralized and federated approaches that generate a single global model, most results are even outperformed by Local. This is due to the significant heterogeneity among client datasets. Training on the aggregated data or creating a universal model for all clients leads to conflicting knowledge during the training process, ultimately weakening the prediction performance. Methods like CNFGNN, ST-TPFL and ST-PFLA incorporate both spatial and temporal features for prediction. However, CNFGNN trains a generic global model for prediction, failing to address the heterogeneity of the data distribution among clients. ST-TPFL utilizes static structural embeddings and generates personalized aggregation weights based on these fixed embeddings, employing a relatively simple aggregation strategy. Consequently, their performance is inferior to that of ST-PFLA. For federated methods that rely solely on temporal features, Ditto and FedPHP outperform FedAvg and FedProx because they employ personalized federated learning strategies to mitigate the adverse effects of data inconsistency. Moreover, among all the methods, only ST-TPFL and ST-PFLA consider topology privacy protection. Notably, ST-PFLA achieves superior predictive performance by leveraging dynamic spatial feature extraction and adaptive personalized aggregation.



(a) METR-LA (b) PEMS-BAY

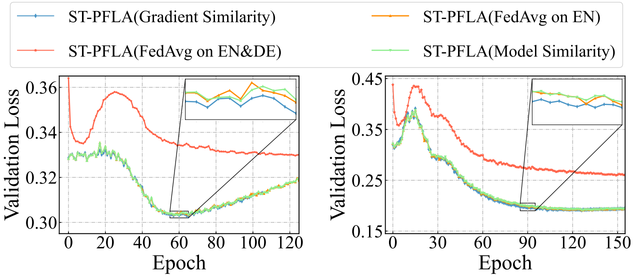
Figure 2: Ablation study of the Spatial Feature Extraction Module in ST-PFLA. Validation loss comparison of Dynamic, Static, and No Embedding settings on two datasets across training rounds.

4.2.2 Effect of the Spatial Feature Extraction Module. In the proposed ST-PFLA framework, we initialize the structural embeddings for all clients using a customized random walk algorithm proposed in [15]. During each training round, these structural embeddings are dynamically updated through a self-attention mechanism, enabling the model to effectively capture and leverage the spatial information of the clients. To validate whether the proposed feature extraction module improves model performance, we conduct experiments under three settings: Dynamic Embedding, Static Embedding, and No Embedding. Dynamic Embedding refers to the ST-PFLA approach, which incorporates a mechanism for dynamically updating embeddings. In Static Embedding, the initialized embeddings remain unchanged throughout the training process. No Embedding excludes structural embeddings entirely, meaning no spatial information is utilized during training. The validation loss trends across training rounds for these three methods are shown in Figure 2. As illustrated in the figure, the Dynamic Embedding approach achieves significantly lower validation loss compared to both Static Embedding and No Embedding, demonstrating superior predictive performance. While the performance of Static Embedding and No Embedding is relatively similar, Static Embedding consistently performs slightly better. This observation underscores the importance of leveraging spatial information in spatio-temporal traffic flow prediction tasks.

4.2.3 Effect of the Adaptive Aggregation Module. To address the issue of knowledge conflicts during the aggregation of heterogeneous clients, we propose an adaptive aggregation strategy. In each communication round, aggregation weights are computed using the guide models from all clients. The encoder is then personalized through this adaptive aggregation, while the decoder is kept local and excluded from the upload process. To validate the effect of the Adaptive Aggregation Module, we conduct experiments on two datasets and record the validation loss for different methods, as shown in Figure 3. For clarity, the key characteristics of each method are annotated in parentheses. ST-PFLA (Gradient Similarity), our proposed method, aggregates only the encoder while keeping the decoder local and calculates personalized aggregation weights based on gradient similarity among guide models. ST-PFLA (Model Similarity) uses model parameter similarity instead of gradient similarity to determine the aggregation weights. ST-PFLA (FedAvg on EN) performs simple averaging on the encoder during aggregation without computing personalized weights, while ST-PFLA (FedAvg on EN&DE) averages both the encoder and decoder during aggregation.

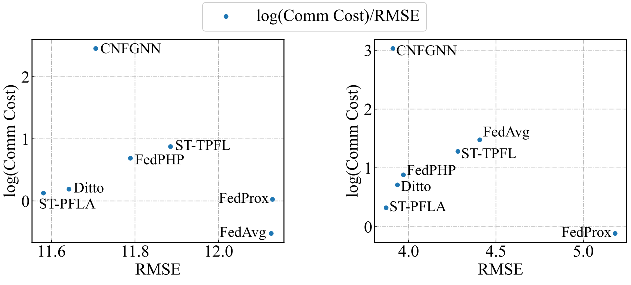
From Figure 3, it can be observed that methods decoupling the encoder and decoder during aggregation significantly improve both convergence speed and prediction accuracy compared to ST-PFLA (FedAvg on EN&DE). Among the three decoupled methods, the approach based on gradient similarity for computing personalized weights achieves the best performance, outperforming both the averaging-based approach and the model similarity-based strategy. This is because the gradient vector represents the direction of model updates on local data, and the cosine similarity between gradient vectors effectively captures the alignment of learning patterns across different clients during a training round. When two clients exhibit highly similar gradient directions, it indicates that they are learning effectively from similar data distributions. In such cases, assigning higher aggregation weights promotes knowledge sharing and reinforces common patterns. Conversely, a large divergence in gradient directions suggests significant heterogeneity in local data characteristics, necessitating weight adjustment to mitigate potential knowledge conflicts. Compared to the other two methods, this gradient-based dynamic weighting strategy better captures inter-client feature correlations. As a result, it enables better model generalization under Non-IID settings by adaptively balancing shared and personalized learning.

4.2.4 Communication Cost. In federated learning, communication overhead plays a crucial role in determining system efficiency and practicality. Specifically, it depends on the size of model parameters exchanged between clients and the server, as well as the number of communication rounds. In edge deployments, limited bandwidth and energy resources make minimizing communication costs particularly important. To evaluate the trade-off between communication overhead and prediction accuracy, we compare all federated methods and visualize the results in a scatter plot (Figure 4). The vertical axis shows communication overhead on a logarithmic scale, while the horizontal axis denotes the final RMSE. Each point corresponds



(a) METR-LA (b) PEMS-BAY

Figure 3: Ablation study of the Adaptive Aggregation Module in ST-PFLA. Validation loss comparison of aggregation strategies—Gradient Similarity, Model Similarity, FedAvg (EN), and FedAvg (EN&DE)—on two datasets.



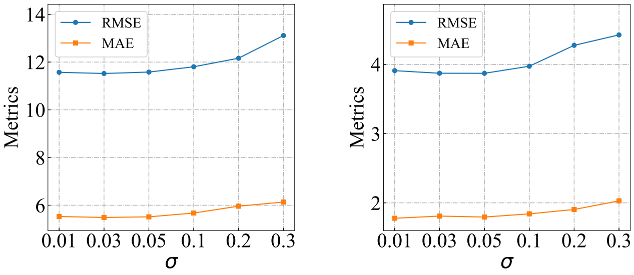
(a) METR-LA (b) PEMS-BAY

Figure 4: Scatter plot of federated methods illustrating the trade-off between communication overhead and RMSE on two datasets.

to a federated method, where lower values on both axes indicate better performance; points closer to the bottom-left corner reflect higher overall effectiveness. From the scatter plots of communication overhead versus RMSE across two datasets, it is evident that our method is positioned at the bottom-left corner, demonstrating superior performance in balancing these two critical metrics. In contrast, CNFGNN achieves relatively low prediction errors, yet its frequent transmission of graph neural network parameters leads to a substantial increase in communication cost. While FedProx and FedAvg converge relatively quickly with lower communication overhead, their final prediction accuracy lags behind significantly. These methods fail to achieve an effective trade-off, resulting in less competitive performance compared to our approach. The reduced communication overhead further highlights the efficiency of ST-PFLA.

4.2.5 *Parameter Analysis.*

In ST-PFLA, to ensure the protection of global topology privacy, a certain level of Gaussian noise is added to the structural embeddings when clients upload them for the first time. This prevents the server from inferring the global topology based on the similarity between the original embeddings. To investigate the impact of such noise on the final prediction performance, we conduct experiments under different levels of noise, setting the standard deviation of the Gaussian noise $\sigma = \{0.01, 0.03, 0.05, 0.1, 0.2, 0.3\}$ covering scenarios from minimal perturbation to strong interference. A larger σ corresponds to stronger noise. Figure 5 shows the results on two datasets under different noise levels, with two lines representing the evaluation metrics. As shown in the figure, when σ is in the range of 0.01 to 0.05, the model performance remains relatively stable and is only mildly affected by the noise. This stability arises from the dynamic self-attention mechanism's capacity to suppress low-level noise: even with minor perturbations in structural embeddings, the mechanism adaptively adjusts attention weights to dynamically capture real-time node correlations, effectively mitigating the impact of noise on spatial feature modeling. However, as σ increases to the range of 0.1 to 0.3, a significant drop in prediction accuracy is observed as noise severely degrades embedding semantics, making it difficult for self-attention to capture spatial dependencies. Excessive noise can severely degrade the quality of the structural embeddings and harm the model's predictive capability. To strike a balance between privacy protection and prediction accuracy, we set σ to 0.05 in our experiments.



(a) METR-LA (b) PEMS-BAY

Figure 5: Parameter Analysis of Gaussian Noise Standard Deviation σ.

5 CONCLUSION

This paper introduces ST-PFLA, a novel federated learning framework for spatio-temporal traffic flow prediction in node-level scenarios. Designed to preserve data and topology privacy, ST-PFLA enhances prediction accuracy while minimizing communication overhead. Specifically, structural embeddings are initialized using a tailored random walk algorithm and dynamically refined through a self-attention mechanism. The model is decoupled during training, where only the encoder is aggregated, leaving the decoder local to each client. To address heterogeneity among clients, we propose an adaptive aggregation strategy based on guide models, leveraging gradient similarity to compute personalized aggregation weights. Extensive experiments on two benchmark datasets confirm the effectiveness and efficiency of ST-PFLA, showcasing its ability to balance privacy preservation, prediction accuracy and communication efficiency.

ACKNOWLEDGMENTS

This work was supported by Natural Science Foundation of Shanghai (No. 25ZR1401100).

REFERENCES

[1] Rasha Al-Huthaifi, Tianrui Li, Zaid Al-Huda, Wei Huang, Zhipeng Luo, and Peng Xie. 2024. FedGODE: Secure traffic flow prediction based on federated learning and graph ordinary differential equation networks. *Knowledge-Based Systems* 299 (2024), 112029.

[2] Hong-You Chen and Wei-Lun Chao. 2021. On bridging generic and personalized federated learning for image classification. *arXiv preprint arXiv:2107.00778* (2021).

[3] Haiyang Chi, Yuhuan Lu, Can Xie, Wei Ke, and Bidong Chen. 2025. Spatio-temporal attention based collaborative local–global learning for traffic flow prediction. *Engineering Applications of Artificial Intelligence* 139 (2025), 109575.

[4] Dongshang Deng, Xuangou Wu, Tao Zhang, Xiangyun Tang, Hongyang Du, Jiawen Kang, Jiqiang Liu, and Dusit Niyato. 2024. Fedasa: A personalized federated learning with adaptive model aggregation for heterogeneous mobile edge computing. *IEEE Transactions on Mobile Computing* (2024).

[5] Jian Feng, Cailing Du, and Qi Mu. 2024. Traffic flow prediction based on federated learning and spatio-temporal graph neural networks. *ISPRS International Journal of Geo-Information* 13, 6 (2024), 210.

[6] Lingzhi Gao, Zexi Li, Xinyi Shang, Yang Lu, and Chao Wu. 2025. Fedios: Decoupling orthogonal subspaces for personalization in feature-skew federated learning. *Machine Learning* 114, 10 (2025), 228.

[7] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[8] Na Hu, Wei Liang, Dafang Zhang, Kun Xie, Kuanching Li, and Albert Y Zomaya. 2024. FedGCN: A federated graph convolutional network for privacy-preserving traffic prediction. *IEEE Transactions on Sustainable Computing* 9, 6 (2024), 925–935.

[9] Duc Kieu, Tung Kieu, Peng Han, Bin Yang, Christian S Jensen, and Bac Le. 2024. TEAM: Topological Evolution-aware Framework for Traffic Forecasting–Extended Version. *arXiv preprint arXiv:2410.19192* (2024).

[10] Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. 2021. Ditto: Fair and robust federated learning through personalization. In *International Conference on Machine Learning*. PMLR, 6357–6368.

[11] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems* 2 (2020), 429–450.

[12] Xin-Chun Li, De-Chuan Zhan, Yunfeng Shao, Bingshuai Li, and Shaoming Song. 2021. Fedphp: Federated personalization with inherited private models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 587–602.

[13] Youpeng Li, Xinda Wang, Fuxun Yu, Lichao Sun, Wenbin Zhang, and Xuyu Wang. 2024. FedCAP: Robust Federated Learning via Customized Aggregation and Personalization. In *2024 Annual Computer Security Applications Conference (ACSAC)*. IEEE, 747–760.

[14] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. 2017. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv preprint arXiv:1707.01926* (2017).

[15] Ying Lin, Xingjian Lu, Yibing Wang, Yuhui Jiang, and Wei Mao. 2024. ST-TPFL: Towards Spatio-Temporal Traffic Flow Prediction Based on Topology Protected Federated Learning. In *Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint International Conference on Web and Big Data*. Springer, 437–451.

[16] Jiahao Liu, Jiang Wu, Jinyu Chen, Miao Hu, Yipeng Zhou, and Di Wu. 2023. Feddwa: Personalized federated learning with dynamic weight adjustment. *arXiv preprint arXiv:2305.06124* (2023).

[17] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*. PMLR, 1273–1282.

[18] Chuizheng Meng, Sirisha Rambhatla, and Yan Liu. 2021. Cross-node federated graph neural network for spatio-temporal data modeling. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*. ACM, 1202–1211.

[19] Lei Meng, Zhuang Qi, Lei Wu, Xiaoyu Du, Zhaochuan Li, Lizhen Cui, and Xiangxu Meng. 2024. Improving global generalization and local personalization for federated learning. *IEEE Transactions on Neural Networks and Learning Systems* (2024).

[20] Lin Pan, Qianqian Ren, Zilong Li, and Xingfeng Lv. 2025. Rethinking spatial-temporal contrastive learning for Urban traffic flow forecasting: multi-level augmentation framework. *Complex & Intelligent Systems* 11, 1 (2025), 96.

[21] Youyang Qu, Shui Yu, Longxiang Gao, Keshav Sood, and Yong Xiang. 2024. Blockchained dual-asynchronous federated learning services for digital twin empowered edge-cloud continuum. *IEEE Transactions on Services Computing* 17, 3 (2024), 836–849.

[22] Aviv Shamsian, Aviv Navon, Ethan Fetaya, and Gal Chechik. 2021. Personalized federated learning using hypernetworks. In *International Conference on Machine Learning*. PMLR, 9489–9502.

[23] Chao Song, Youfang Lin, Shengnan Guo, and Huaiyu Wan. 2020. Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 914–921.

[24] Hanqiu Wang, Rongqing Zhang, Xiang Cheng, and Liuqing Yang. 2022. Federated spatio-temporal traffic flow prediction based on graph convolutional network. In *2022 14th International Conference on Wireless Communications and Signal Processing (WCSP)*. IEEE, 221–225.

[25] Hanqiu Wang, Rongqing Zhang, Xiang Cheng, and Liuqing Yang. 2022. Hierarchical traffic flow prediction based on spatial-temporal graph convolutional network. *IEEE Transactions on Intelligent Transportation Systems* 23, 9 (2022), 16137–16147.

[26] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. Graph wavenet for deep spatial-temporal graph modeling. *arXiv preprint arXiv:1906.00121* (2019).

[27] Chulin Xie, De-An Huang, Wenda Chu, Daguang Xu, Chaowei Xiao, Bo Li, and Anima Anandkumar. 2024. Perada: Parameter-efficient federated learning personalization with generalization guarantees. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 23838–23848.

[28] Xiyuan Yang, Wenke Huang, and Mang Ye. 2024. Fedas: Bridging inconsistency in personalized federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 11986–11995.

[29] Dezhong Yao, Ziquan Zhu, Tongtong Liu, Zhiqiang Xu, and Hai Jin. 2024. Rethinking personalized federated learning from knowledge perspective. In *Proceedings of the 53rd International Conference on Parallel Processing*. 991–1000.

[30] Xiangyu Yao, Xinglin Piao, Qitan Shao, Yongli Hu, Baocai Yin, and Yong Zhang. 2025. SHKD: A framework for traffic prediction based on Sub-Hypergraph and Knowledge Distillation. *Knowledge-Based Systems* 312 (2025), 113163.

[31] Keting Yin and Jiayi Mao. 2024. Personalized federated learning with adaptive feature aggregation and knowledge transfer. *arXiv preprint arXiv:2410.15073* (2024).

[32] Bing Yu, Haoteng Yin, and Zhanxing Zhu. 2017. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875* (2017).

[33] Chenhan Zhang, Shuyu Zhang, James J. Q. Yu, and Shui Yu. 2021. FASTGNN: A Topological Information Protected Federated Learning Approach for Traffic Speed Forecasting. *IEEE Transactions on Industrial Informatics* 17, 12 (2021), 8464–8474. https://doi.org/10.1109/TII.2021.3055283

[34] Hui Zhang, Kun Ding, Jietao Xie, Weidong Xiao, and Yuxiang Xie. 2025. Flow prediction via adaptive dynamic graph with spatio-temporal correlations. *Expert Systems with Applications* 261 (2025), 125474.

[35] Ran Zhang, Fangqi Liu, Jiang Liu, Mingzhe Chen, Qinqin Tang, Tao Huang, and F Richard Yu. 2024. Cpper-fl: Clustered parallel training for efficient personalized federated learning. *IEEE Transactions on Mobile Computing* 23, 10 (2024), 9424–9436.

[36] Rongqing Zhang, Jingxin Mao, Hanqiu Wang, Bing Li, Xiang Cheng, and Liuqing Yang. 2024. A survey on federated learning in intelligent transportation systems. *IEEE Transactions on Intelligent Vehicles* (2024).

[37] Xiangjie Zhang, Chuanjiang Li, Changkun Han, Li Shaobo, Yixiong Feng, Haoyu Wang, Zuo Cui, and Konstantinos Gryllias. 2024. A personalized federated meta-learning method for intelligent and privacy-preserving fault diagnosis. *Advanced Engineering Informatics* 62 (2024), 102781.

[38] Ling Zhao, Yujiao Song, Chao Zhang, Yu Liu, Pu Wang, Tao Lin, Min Deng, and Haifeng Li. 2019. T-GCN: A temporal graph convolutional network for traffic prediction. *IEEE Transactions on Intelligent Transportation Systems* 21, 9 (2019), 3848–3858.

[39] Yang Zhao, Jun Zhao, Mengmeng Yang, Teng Wang, Ning Wang, Lingjuan Lyu, Dusit Niyato, and Kwok-Yan Lam. 2020. Local differential privacy-based federated learning for internet of things. *IEEE Internet of Things Journal* 8, 11 (2020), 8836–8853.

[40] Yinlin Zhu, Xunkai Li, Zhengyu Wu, Di Wu, Miao Hu, and Rong-Hua Li. 2024. Fedtad: Topology-aware data-free knowledge distillation for subgraph federated learning. *arXiv preprint arXiv:2404.14061* (2024).