FedGTP: Exploiting Inter-Client Spatial Dependency in Federated Graph-based Traffic Prediction

Linghua Yang
SKLSDE
Beihang University
sy2206212@buaa.edu.cn

Wantong Chen
SKLSDE
Beihang University
2306cwt@buaa.edu.cn

Xiaoxi He
FST
University of Macau
hexiaoxi@um.edu.mo

Shuyue Wei
SKLSDE
Beihang University
weishuyue@buaa.edu.cn

Yi Xu
SKLSDE
Beihang University
xuy@buaa.edu.cn

Zimu Zhou
SDSC, City University of Hong Kong
zimuzhou@cityu.edu.hk

Yongxin Tong
SKLSDE
Beihang University
yxtong@buaa.edu.cn

ABSTRACT

Graph-based methods have witnessed tremendous success in traffic prediction, largely attributed to their superior ability in capturing and modeling spatial dependencies. However, urban-scale traffic data are usually distributed among various owners, limited in sharing due to privacy restrictions. This fragmentation of data severely hinders interaction across clients, impeding the utilization of inter-client spatial dependencies. Existing studies have yet to address this non-trivial issue, thereby leading to sub-optimal performance. To fill this gap, we propose FedGTP, a new federated graph-based traffic prediction framework that promotes adaptive exploitation of inter-client spatial dependencies to recover close-to-optimal performance complying with privacy regulations like GDPR. We validate FedGTP via large-scale application-driven experiments on real-world datasets. Extensive baseline comparison, ablation study and case study demonstrate that FedGTP indeed surpasses existing methods through fully recovering inter-client spatial dependencies, achieving 21.08%, 13.48%, 19.90% decrease on RMSE, MAE and MAPE, respectively. Our code is available at https://github.com/LarryHawkingYoung/KDD2024_FedGTP.

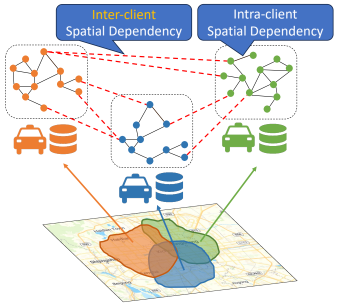
CCS CONCEPTS

• **Computing methodologies → Learning paradigms**.

KEYWORDS

Federated Learning; Traffic Prediction; Spatial-Temporal Graph Neural Network

ACM Reference Format:
Linghua Yang, Wantong Chen, Xiaoxi He, Shuyue Wei, Yi Xu, Zimu Zhou, and Yongxin Tong. 2024. FedGTP: Exploiting Inter-Client Spatial Dependency in Federated Graph-based Traffic Prediction. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24), August 25–29, 2024, Barcelona, Spain.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3637528.3671613



Inter-client Spatial Dependency
Intra-client Spatial Dependency

Figure 1: Inter-/intra-client spatial dependency.

1 INTRODUCTION

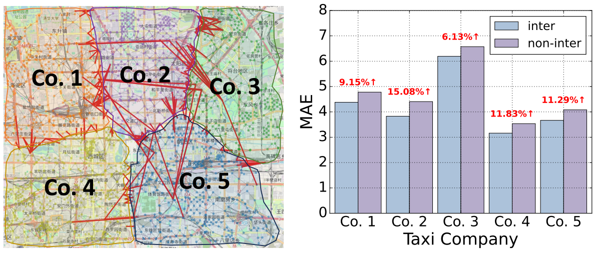
Traffic prediction is essential for optimizing urban mobility [26, 35], reducing congestion [17, 21], and enhancing road safety [42, 51, 56]. It forecasts traffic conditions by analyzing patterns derived from traffic data spanning both spatial and temporal dimensions. These data can be effectively modeled as spatiotemporal graphs, where nodes represent locations and edges represent spatial dependencies between them. Therefore, graph-based deep learning methods like Spatial-Temporal Graph Neural Networks (STGNNs) have emerged as the primary tool for traffic prediction, incorporating Graph Neural Networks (GNNs) to capture non-Euclidean spatial dependencies [44].

Spatial dependencies are crucial for accurate traffic prediction [14, 24, 48], yet their practical utilization poses significant challenges. This is because traffic data are often distributed among various owners (governments, companies, or individuals), *a.k.a.*, *clients*, with privacy regulations such as GDPR[1] restricting the free sharing of data. This leads to the fragmentation of spatial dependency information, dividing it into multiple sub-graphs, each owned by a distinct client. Accordingly, spatial dependencies in traffic data are now categorized into two types: *intra-client* and *inter-client* (see Fig. 1). Intra-client dependencies (individual sub-graphs) can still be captured effectively within each client's data. However, extracting inter-client dependencies (the connections between sub-graphs) becomes complicated due to the necessary cross-client information exchange under privacy constraints.

KDD '24, August 25–29, 2024, Barcelona, Spain.

ACM ISBN 979-8-4007-0490-1/24/08
https://doi.org/10.1145/3637528.3671613

[1] https://gdpr-info.eu



(a) **(b)**

Figure 2: A toy example on the importance of inter-client spatial dependencies: (a) shows the data distributions of five taxi companies, with red lines highlighting inter-client spatial dependencies; (b) plots the prediction error (MAE) increase after removing inter-client spatial dependencies.

Federated Learning (FL) [31, 49] offers a natural solution for distributed model training while respecting privacy concerns. There is also a growing interest to apply federated learning to STGNNs for federated graph-based traffic prediction [23, 27, 28, 32, 39, 43, 47, 52–54]. However, existing proposals can be classified into two categories by their **limitations**: (i) Those that largely *under-utilize* inter-client spatial dependencies under privacy constraints [23, 27, 43, 53], which *incur potential accuracy degradation*. A toy example (see Fig. 2) demonstrates the importance of inter-client spatial dependencies in federated graph-based traffic prediction. This example predicts the traffic flows of five taxi companies in a city and reveals that removing inter-client dependencies from training data results in up to 15% error increase for each taxi company. (ii) Those that treat the inter-client spatial dependencies as *public and predefined* [28, 32, 39, 47, 52, 54], which *compromise privacy*. Later our experiments also reveal that predefined dependencies inadequately capture complex relations, leading to poor performance. Thus, it's vital to fully exploit the inter-client spatial dependencies in federated graph-based traffic prediction.

In this paper, we introduce FedGTP (**Fed**erated **G**raph-based **T**raffic **P**rediction), a new federated graph learning framework designed to fully leverage inter-client spatial dependencies for traffic prediction. We base FedGTP on ASTGNNs [4, 20, 45], known for their state-of-the-art prediction accuracy through adaptive learning of spatial dependencies. However, learning inter-client spatial dependencies in a federated setup is non-trivial. This challenge arises because the spatial dependency learning in ASTGNNs requires access to raw data, which becomes impractical in federated learning without extensive secure operations. To address this, we reformulate spatial modeling and learning into components specific to intra- and inter-client interactions. At the heart of inter-client spatial dependency reconstruction, we introduce an adaptive polynomial-based activation decomposition mechanism, which minimizes the need for cross-client computations while preserving privacy. Finally, we integrate our reformulation into the core of federated learning, while enabling personalization, a beneficial feature when data distributions vary significantly among clients.

The main contributions of this paper are as follows:

- To the best of our knowledge, it is the first work that advocates and enables full utilization of inter-client spatial dependencies for federated graph-based traffic prediction.
- We propose a novel and unified framework *FedGTP*, which performs adaptive learning of spatial dependencies across clients, enabling profound exploitation of spatial relations and thus enhancing performance. This framework includes a polynomial-based privacy-preserving mechanism that reveals only aggregated intermediate results to server for secure summation, adhering to privacy regulations like GDPR.
- Extensive experiments on real-world datasets show that by fully recovering inter-client spatial dependencies, our solution largely outperforms prior methods by 21.08%, 13.48%, 19.90% for RMSE, MAE and MAPE, respectively.

2 RELATED WORK

2.1 Graph-based Traffic Prediction

Spatial-temporal graph neural networks (STGNNs) prevail in graph-based traffic prediction [44]. For *temporal modeling*, both recurrent [6, 14, 24] and convolutional neural networks [48, 50] are frequently used. Attention mechanisms are also used to extract dynamic temporal patterns in ASTGCN [15], GMAN [55], STG2Seq [2], and ASTGNN(p) [16]. Our focus is on the *spatial modeling*. Conventional STGNNs often assume a graph topology predefined by geographic distance [6, 24, 48, 50], or semantic similarity [2, 3, 14]. One alternative is to apply attention-based spatial models to learn edge weights from data [2, 6, 15, 16, 55]. However, these studies fail to capture the full spatial dependencies because the learned connections are still restricted by the predefined adjacency matrix. A major advancement is the development of adaptive spatial-temporal graph neural networks (ASTGNNs) [4, 20, 45]. For instance, Graph WaveNet [45] introduces a trainable AGCN layer to learn a normalized adaptive adjacency matrix. AGCRN [4] enhances the AGCN layer by discerning node-specific patterns. ASTGAT [20] adopts a network generator model to create an adaptive discrete graph and infer hidden correlations directly from the data. We ground our work upon ASTGNNs [4, 20, 45] since they can infer spatial dependencies from data and achieve the state-of-the-art performances.

2.2 Federated Graph Learning

To facilitate graph learning in scenarios where graph data are distributed and necessitate privacy protection, FL has been integrated into graph learning. Existing works in general FGL primarily focus on tackling challenges such as cross-client missing information, privacy leakage of graph structures, and data heterogeneity across clients [22, 29]. We specifically focus on the reconstruction of inter-client missing information, which is coherent to the spatial-temporal essence of traffic prediction.

Some studies strictly *protect privacy* of both temporal data on nodes and the spatial topology on edges. FLoS [43] and MFVST-GNN [27] aggregates local STGNNs via FedAvg [31] without considering inter-client spatial dependencies. FASTGNN [53] applies differential privacy to aggregate adjacency matrices of each subgraph, but it randomly generates inter-client spatial connections. FML-ST [23] constructs local spatial-temporal patterns on each client, which are aggregated into a global one assisted by the server. These methods largely under-utilize inter-client spatial dependencies with privacy constraints, yielding sub-optimal performance.

Another major part of studies reconstruct information based on traditional GNNs and treat the inter-client edges as *public and pre-defined.* For instance, CNFGNN [32] extracts the cross-node dependencies on a pre-defined graph topology at the server. FedSTN [52] extracts long-term and short-term spatial-temporal information separately on public road network. FedAGCN [39] and FCGCN [47] treat spatial topology as public and apply community detection algorithms to partition sub-graphs. In CTFL [54], the spatial graph is available to all clients. These methods partially compromise privacy, and the dependencies extracted solely from the public and predefined channels are insufficient and biased.

3 PROBLEM STATEMENT

3.1 Graph Modeling of Traffic Data

Adhering to prior research on traffic prediction [2, 4, 10, 15, 18, 24, 45, 50], we depict traffic data as a sequence of graph signal frames $\{X^1, X^2, ..., X^T\}$. The graph signal $X^t \in \mathbb{R}^{N \times F}$ denotes the observations defined on $\mathcal{G}$ at the t-th time slot, where F is the feature channel. The graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, *a.k.a.* the spatial network[2] contains a node set $\mathcal{V}$ of size $|\mathcal{V}| = N$ and an edge set $\mathcal{E}$. The edges $\mathcal{E}$ are represented by an adjacency matrix $A \in \mathbb{R}^{N \times N}$, which is usually built upon *(i)* pre-defined graphs like geographic distance [6, 24, 48, 50]; *(ii)* domain knowledge based attributes similarity [2, 3, 14]; or *(iii)* spatial correlations learned form data [4, 20, 45]. Particularly, method *(iii)* is the state-of-the-art to define adjacency matrix, as it can learn spatial dependencies from data without prior or domain knowledge.

3.2 Graph-based Traffic Prediction

Consider graph $\mathcal{G}$ and T_{in} historical observations of graph signal $X^{(t-T_{in}+1):t}$. Traffic prediction at time slot t aims to learn a function $F(\cdot)$ which maps the historical observations into the future ones in the next T_{out} time slots:

$$X^{(t+1):(t+T_{out})} \leftarrow F(X^{(t-T_{in}+1):t}; \theta, \mathcal{G}) \tag{1}$$

where θ denotes all the learnable parameters. We use colons to denote a temporal sequence. The optimal model parameters θ^* are trained via the following objective.

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\, \mathcal{L}(F, \theta; \mathcal{D}) \tag{2}$$

where $\mathcal{L}$ represents the loss function of F with parameters θ on dataset $\mathcal{D} = \{\mathcal{G}, X^{1:T}\}$.

3.3 Federated Graph-based Traffic Prediction

Consider the client-server based federated learning, where M clients (data owners), denoted by $C = \{C_1, C_2, ..., C_M\}$, collaboratively train models under coordination of a central server [31, 37].

- *Data Partition.* Assume client C_i maintains a local traffic dataset $\mathcal{D}_i = \{\mathcal{G}_i, X_i^{1:T}\}$, where $\mathcal{G}_i = \{\mathcal{V}_i, \mathcal{E}_i\}$ is a local graph of N_i nodes. Note that $\mathcal{G} = \bigcup_{i=1}^{M} \mathcal{G}_i$ is the complete graph and $N = \sum_{i=1}^{M} N_i$ is the total number of nodes. Due to the partition of $\mathcal{G}$, the corresponding adjacency matrix $A \in \mathbb{R}^{N \times N}$ and graph signal $X^{1:T} \in \mathbb{R}^{T \times N \times F}$ are partitioned across clients as

$$A = \begin{bmatrix} A_{11} & \cdots & A_{1M} \\ \vdots & \ddots & \vdots \\ A_{M1} & \cdots & A_{MM} \end{bmatrix}, X^{1:T} = \begin{bmatrix} X_1^{1:T} \\ \vdots \\ X_M^{1:T} \end{bmatrix} \tag{3}$$

where $A_{ij} \in \mathbb{R}^{N_i \times N_j}$ is formed by rows and columns correspond to $\mathcal{V}_i$ and $\mathcal{V}_j$ in A. Similarly, $X_i^{1:T} \in \mathbb{R}^{T \times N_i \times F}$ is the local graph signal for the N_i nodes over T time slots.
- *Privacy Constraints.* As is common in federated learning, the server has no access to local datasets $\{\mathcal{D}_i\}$. Furthermore, each client C_i can share neither its own graph signals $X_i^{1:T}$ nor its local graph $\mathcal{G}_i$ with any other clients. For example, as a taxi company, the historical traffic data of its taxis, as well as the correlations between taxis in various regions, are considered sensitive business secrets not allowed to be disclosed or shared to others.
- *Training Objectives.* Since $\mathcal{D}_i$ embodies region-specific characteristics that can be non-IID (independent and identically distributed) across clients, we consider a personalized federated learning setting [12], where client-specific optimal parameters $\theta_1^*, ..., \theta_M^*$ are trained with the objective below

$$\{\theta_1^*, ..., \theta_M^*\} = \underset{\theta_1,...,\theta_M}{\operatorname{argmin}} \sum_{i=1}^{M} \frac{N_i}{N} \mathcal{L}(F_i, \theta_i, \mathcal{D}_i). \tag{4}$$

Due to the privacy constraints, it is challenging to make full use of the spatial dependencies (represented by A) for traffic prediction. While **intra-client spatial dependencies** (A_{ij} for $i = j$) can be readily derived from local dataset $\mathcal{D}_i$ [4, 20, 45], the extraction of **inter-client spatial dependencies** (A_{ij} for $i \neq j$) is far from straightforward due to the necessary data exchange across clients.

4 METHOD

This section presents FedGTP, a new federated graph-based traffic prediction framework. It is built upon centralized ASTGNNs (Sec. 4.1), with a novel spatial modeling formulation for the federated setting (Sec. 4.2). We then explain how to integrate the new spatial modeling with temporal modeling (Sec. 4.3), and introduce the overall system design and implementation of FedGTP (Sec. 4.4).

4.1 ASTGNNs for Federated Graph-based Traffic Prediction

4.1.1 Primer on ASTGNNs. As with other STGNNs [6, 14, 24, 24, 48, 50], ASTGNNs [4, 20, 45] also consist of a *spatial* and a *temporal* component. They adopt an Adaptive Graph Convolution Network (AGCN) [4] for spatial modeling and Gated Recurrent Units (GRU) [9] to capture temporal correlations. The core of ASTGNNs is the AGCN layer, which introduces adaptive learning into the conventional GCN layer [19]. Specifically, an AGCN layer contains a Data Adaptive Graph Generation (DAGG) module inferring spatial dependencies and a Node Adaptive Parameter Learning (NAPL) module capturing node-specific patterns:

$$\tilde{A} = I_N + \sigma\left(E \cdot E^\top\right), \tag{5}$$

$$H^{(l)} = \sigma\left(\tilde{A} \cdot H^{(l-1)} \cdot E \cdot W + E \cdot b\right), \tag{6}$$

[2] Following the conventions [2, 4, 7, 8, 10, 15, 18, 24, 44, 45, 48, 50], we only consider static homogeneous graphs with nodes and edges of the same type or class.

where $\tilde{A} \in \mathbb{R}^{N\times N}$ is the adaptive adjacency matrix, $I_N \in \mathbb{R}^{N\times N}$ is the identity matrix, each row of $E \in \mathbb{R}^{N\times d}$ presents the learnable node embedding, $H^{(l-1)} \in \mathbb{R}^{N\times F^{(l-1)}}$ and $H^{(l)} \in \mathbb{R}^{N\times F^{(l)}}$ are the input and output feature of the l-th layer, $W \in \mathbb{R}^{d\times F^{(l-1)}\times F^{(l)}}$ and $b \in \mathbb{R}^{d\times F^{(l)}}$ are the trainable weights and bias pool. $\sigma(\cdot)$ is the nonlinear activation.

The AGCN layer is then integrated into GRU by replacing all the linear layers in it with spatiotemporal feature fusion:

$$\begin{aligned} z^t &= \sigma\left(\tilde{A}\cdot[X^t \mid\mid h^{t-1}]\cdot E\cdot W_z + E\cdot b_z\right) \\ r^t &= \sigma\left(\tilde{A}\cdot[X^t \mid\mid h^{t-1}]\cdot E\cdot W_r + E\cdot b_r\right) \\ \tilde{h}^t &= tanh\left(\tilde{A}\cdot[X^t \mid\mid (r^t\odot h^{t-1})]\cdot E\cdot W_{\tilde{h}} + E\cdot b_{\tilde{h}}\right) \\ h^t &= z^t\odot h^{t-1} + (1-z^t)\odot \tilde{h}^t, \end{aligned} \tag{7}$$

where X^t and h^t are input and output at the t-th time slot. $||$ is the concatenation operation. $\odot$ is the Hadamard product. z, r, and $\tilde{h}$ are the update gate, reset gate and candidate state, respectively.

4.1.2 Challenges in Federated ASTGNNs. In the centralized setting, both the graph signals $X^{1:T}$ and graph structure $\tilde{A}$ are gathered in one place to train ASTGNNs. However, in the federated setting, the node embedding matrix E used to generate $\tilde{A}$ and the feature matrix H containing $X^{1:T}$ are partitioned into

$$E = \begin{bmatrix} E_1 \\ \vdots \\ E_M \end{bmatrix}, H = \begin{bmatrix} H_1 \\ \vdots \\ H_M \end{bmatrix} \tag{8}$$

and distributed across clients.

From the perspective of each single client C_i, the l-th local layer feature matrix can be expressed as

$$H_i^{(l)} = \sigma\left(\sum_{j=1}^{M}(\tilde{A}_{ij}\cdot H_j^{(l-1)})\cdot E_i\cdot W + E_i\cdot b\right). \tag{9}$$

Omitting layer id for brevity, we let $Z_i = \sum_{j=1}^{M}(\tilde{A}_{ij}\cdot H_j)$, which embodies **spatial dependencies** and can be divided into

$$Z_i = \tilde{A}_{ii}\cdot H_i + \sum_{j=1, j\neq i}^{M}(\tilde{A}_{ij}\cdot H_j). \tag{10}$$

Following the terminologies in Sec. 3, $\tilde{A}_{ii}\cdot H_i$ is the **intra-client** spatial dependencies with features available within C_i, while $\tilde{A}_{ij}\cdot H_j$ when $i\neq j$ is the **inter-client** ones that can only be calculated involving other C_j.

However, federated setting prohibits C_j from sharing its graph signals $X_j^{1:T}$ and graph structure $\mathcal{G}_j$ to C_i (see Sec. 3). Regarding ASTGNNs, the privacy restrictions extend to E_j and H_j. Since E_j multiplied by itself would reveal $\tilde{A}_{jj}$ (see Eq. (5)), which leaks local graph; and H_j contains $X_j^{1:T}$. Such constraints pose challenges to the establishment of spatial dependencies between clients.

To reconstruct the inter-client part of Z_i in federated setting, a naive solution based on Eq. (5) is shown in Fig. 3a. Each client first encrypts E_i and H_i, and uploads them to the server. Then the server performs matrix calculation in *ciphertext* using approaches like Homomorphic Encryption (HE) [40]. Finally, Z_i in ciphertext is sent to each client and decrypted into plaintext. This solution involves $O\left(N^2(d+F)\right)$ atomic operations in ciphertext, which can be hundreds to thousands of times slower than its plaintext counterpart [1], such a naive solution would cause tremendous computational costs. Thus, there is a need for more efficient alternatives to calculate the inter-client spatial dependencies.

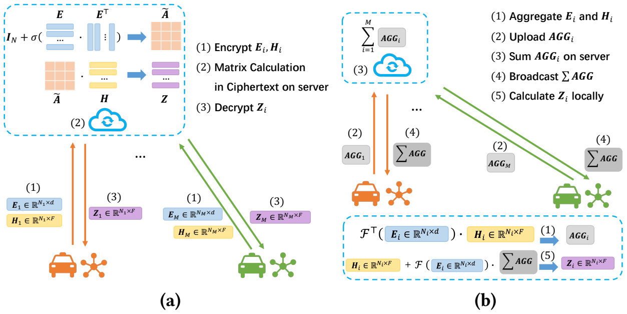

E
$E^\top$
$\tilde{A}$
$I_N + \sigma($
$)$
$\tilde{A}$
H
Z
(2)
(1) Encrypt E_i, H_i
(2) Matrix Calculation in Ciphertext on server
(3) Decrypt Z_i
(1)
$E_1 \in \mathbb{R}^{N_1\times d}$
$H_1 \in \mathbb{R}^{N_1\times F}$
(3)
$Z_1 \in \mathbb{R}^{N_1\times F}$
...
(1)
$E_M \in \mathbb{R}^{N_M\times d}$
$H_M \in \mathbb{R}^{N_M\times F}$
(3)
$Z_M \in \mathbb{R}^{N_M\times F}$
(a)
$\sum_{i=1}^{M}$ AGG_i
(3)
(1) Aggregate E_i and H_i
(2) Upload AGG_i
(3) Sum AGG_i on server
(4) Broadcast $\sum AGG$
(5) Calculate Z_i locally
(2)
AGG_1
(4)
$\sum AGG$
...
(2)
AGG_M
(4)
$\sum AGG$
$\mathcal{F}^\top($ $E_i \in \mathbb{R}^{N_i\times d}$ $)\cdot$ $H_i \in \mathbb{R}^{N_i\times F}$ (1) AGG_i
$H_i \in \mathbb{R}^{N_i\times F}$ $+ \mathcal{F}($ $E_i \in \mathbb{R}^{N_i\times d}$ $)\cdot$ $\sum AGG$ (5) $Z_i \in \mathbb{R}^{N_i\times F}$
(b)

Figure 3: Illustration of approaches for computing inter-client spatial dependency: (a) shows the naive encryption-based solution; (b) presents our proposed solution, which transfers ciphertext computation on server to plaintext computation in local as much as possible.

4.2 Learning Inter-Client Spatial Dependency

4.2.1 Reformulation of Spatial Modeling. To achieve inter-client spatial dependency reconstruction more computational-efficiently while preserving privacy, we reformulate the computation of spatial modeling. The idea is to maximize local calculations which can be performed in plaintext and reducing the time-consuming server-side ciphertext operations (see Fig. 3b). Specifically, we decompose $\tilde{A}_{ij}$ into the product of two distinct parts, one pertains exclusively to C_i and the other to C_j. However, a challenge arises from the non-linear nature of the activation in Eq. (5) (*i.e.* ReLU in ASTGNNs), which complicates privacy preservation. To address this, we propose an *activation decomposition mechanism* by applying an elaborated transform function, denoted as $\mathcal{F}(\cdot)$, to the node embedding matrix of each client, so as to retain the necessary non-linearity and allow effective restructuring of spatial modeling:

$$\tilde{A}_{ij} = \begin{cases} I_{N_i} + \mathcal{F}(E_i)\cdot\mathcal{F}^\top(E_i) & \text{if } i = j, \\ \mathcal{F}(E_i)\cdot\mathcal{F}^\top(E_j) & \text{if } i\neq j. \end{cases} \tag{11}$$

Bringing it into Eq. (10), we obtain the new form:

$$Z_i = H_i + \mathcal{F}(E_i)\cdot\sum_{j=1}^{M}\left(\mathcal{F}^\top(E_j)\cdot H_j\right). \tag{12}$$

According to Eq. (12), each client C_i first computes $AGG_i = \mathcal{F}^\top(E_i)\cdot H_i$ locally in plaintext. This aggregated intermediate result is then uploaded from each client to the server. The server conducts only a simple summation and broadcasts the result $\sum AGG$ to each client for the remaining local calculations. Since the intermediate results uploaded to the server are aggregated rather than raw data, it complies with privacy standards such as GDPR [38]. Next, we present two designs of the transform function $\mathcal{F}(\cdot)$.

4.2.2 Straight-forward Activation Decomposition. An intuitive option is to set $\mathcal{F}(\cdot)$ to $ReLU(\cdot)$. This simple mechanism is named

SprtReLU, as it reorders the *ReLU* activation to be applied *separately* on each node embedding matrix before proceeding with their multiplication. Based on this, we can rewrite Eq. (5) as:

$$\tilde{\boldsymbol{A}}_{(SprtReLU)} = \boldsymbol{I}_N + ReLU(\boldsymbol{E}) \cdot (ReLU(\boldsymbol{E}))^\top, \tag{13}$$

where for each adjacency block

$$\tilde{\boldsymbol{A}}_{(SprtReLU)ij} = \begin{cases} \boldsymbol{I}_{N_i} + ReLU(\boldsymbol{E}_i) \cdot (ReLU(\boldsymbol{E}_i))^\top & \text{if } i = j, \\ ReLU(\boldsymbol{E}_i) \cdot \left(ReLU(\boldsymbol{E}_j)\right)^\top & \text{if } i \neq j. \end{cases} \tag{14}$$

Therefore, the *SprtReLU*-version spatial dependencies

$$\boldsymbol{Z}_{(SprtReLU)i} = \boldsymbol{H}_i + ReLU(\boldsymbol{E}_i) \cdot \sum_{j=1}^{M} \left(\left(ReLU(\boldsymbol{E}_j)\right)^\top \cdot \boldsymbol{H}_j \right), \tag{15}$$

where $\boldsymbol{E}_i$ and $\boldsymbol{H}_i$ are local within C_i, then each term $\left(ReLU(\boldsymbol{E}_j)\right)^\top \cdot \boldsymbol{H}_j^{(l-1)}$ can be computed within the corresponding C_j. With the help of server, a secure summation is implemented to achieve inter-client spatial dependency reconstruction in federated context.

4.2.3 Polynomial-based Activation Decomposition. Although the *SprtReLU* mechanism reconstructs inter-client spatial dependencies at low computational costs, it suffers from performance degradation due to the loss of crucial inter-client information. This loss occurs because negative values in the embedding matrix are filtered out prematurely, hindering the model's capacity to learn complex nonlinear relationships. To this end, we introduce *AdptPoLU*, which approximates *ReLU* activation by *polynomials* with *adaptive* coefficients learned from data. This method alleviates the loss of inter-client information during activation decomposition.

It is well known that most non-linear activation functions can be represented with polynomials through Taylor expansion. Numerous studies [13, 30, 33, 36] have demonstrated the efficacy of low-order polynomials in approximating *ReLU* with minimal training error. Accordingly, we utilize a unified polynomial function to approximate the activation and retain non-linearity.

Consider a K-order polynomial function

$$\mathcal{P}_K(x) = \sum_{k=0}^{K} p_k x^k \tag{16}$$

with coefficient set $\{p_0, p_1, ..., p_K\}$ to be configured. We can rewrite Eq. (5) as:

$$\tilde{\boldsymbol{A}}^K_{(AdptPoLU)} = \boldsymbol{I}_N + \mathcal{P}_K(\boldsymbol{E} \cdot \boldsymbol{E}^\top) = \boldsymbol{I}_N + \sum_{k=0}^{K} \left(p_k (\boldsymbol{E} \cdot \boldsymbol{E}^\top)^k \right). \tag{17}$$

It is worth noting that the polynomials are applied on the matrix $(\boldsymbol{E} \cdot \boldsymbol{E}^\top)$ after multiplication in an element-wise manner. For each adjacency block between client C_i and C_j:

$$\tilde{\boldsymbol{A}}^K_{(AdptPoLU)ij} = \begin{cases} \boldsymbol{I}_{N_i} + \sum_{k=0}^{K} \left(p_k (\boldsymbol{E}_i \cdot \boldsymbol{E}_i^\top)^k \right) & \text{if } i = j, \\ \sum_{k=0}^{K} \left(p_k (\boldsymbol{E}_i \cdot \boldsymbol{E}_j^\top)^k \right) & \text{if } i \neq j. \end{cases} \tag{18}$$

To disentangle the node embedding matrices in each term $(\boldsymbol{E}_i \cdot \boldsymbol{E}_j^\top)^k$ for $k = 0, 1, ..., K$ in this K-order polynomial function, we propose a set of $K + 1$ transforms:

$$\mathcal{F}^K_{(AdptPoLU)} = \{f_k(\cdot)\}_{k=0}^{K}, \tag{19}$$

where the k-th transform $f_k(\cdot)$ for the k-th polynomial satisfies:

$$(\boldsymbol{E}_i \cdot \boldsymbol{E}_j^\top)^k = f_k(\boldsymbol{E}_i) \cdot f_k^\top(\boldsymbol{E}_j). \tag{20}$$

Next, we propose concrete formulations of these transforms.

Theorem 1. *There exists a transform function set*

$$\mathcal{F}^K_{(AdptPoLU)} = \{f_k(\cdot)\}_{k=0}^{K} = \{\otimes^k(\cdot)\}_{k=0}^{K} \tag{21}$$

with $K+1$ *transform functions satisfying Eq. (20) with no information loss. The operator* $\otimes^k(\cdot)$ *transforms a matrix from* $\mathbb{R}^{N \times d}$ *to* $\mathbb{R}^{N \times d^k}$ *by applying a* k*-th Cartesian power on each row.*

Detailed proof is shown in appendix A.1. Based on this theorem, we rewrite Eq. (17) as:

$$\tilde{\boldsymbol{A}}^K_{(AdptPoLU)} = \boldsymbol{I}_N + \sum_{k=0}^{K} \left(p_k \cdot \otimes^k \boldsymbol{E} \cdot \left(\otimes^k \boldsymbol{E} \right)^\top \right), \tag{22}$$

and rewrite Eq. (18) as:

$$\tilde{\boldsymbol{A}}^K_{(AdptPoLU)ij} = \begin{cases} \boldsymbol{I}_{N_i} + \sum_{k=0}^{K} \left(p_k \cdot \otimes^k \boldsymbol{E}_i \cdot \left(\otimes^k \boldsymbol{E}_i \right)^\top \right) & \text{if } i = j, \\ \sum_{k=0}^{K} \left(p_k \cdot \otimes^k \boldsymbol{E}_i \cdot \left(\otimes^k \boldsymbol{E}_j \right)^\top \right) & \text{if } i \neq j. \end{cases} \tag{23}$$

Therefore, the spatial dependencies with K-order *AdptPoLU*

$$\boldsymbol{Z}^K_{(AdptPoLU)i} = \boldsymbol{H}_i + \sum_{k=0}^{K} \left(p_k \cdot \otimes^k \boldsymbol{E}_i \cdot \sum_{j=1}^{M} \left((\otimes^k \boldsymbol{E}_j)^\top \cdot \boldsymbol{H}_j \right) \right). \tag{24}$$

The parameter K is critical to balance accuracy and computational complexity: a larger K enhances accuracy but increases computational demands, and vice versa. We can adjust K for different scenarios. Moreover, since the value distribution of model parameters, especially the node embeddings vary throughout training, a static set of $p_0, p_1, ..., p_K$ is sub-optimal. Instead, we dynamically learn the polynomial coefficients from data. Each client C_i is assigned a unique set of coefficients $\boldsymbol{P}_i = [p_{i,0}, p_{i,1}, ..., p_{i,K}]$. These coefficients are iteratively refined using gradient descent in parallel with the evolution of node embeddings.

4.2.4 Time Complexity Analysis. The time complexity of *SprtReLU* is $O\left(dF(M+N)\right)$, while for *AdptPoLU*, it is $O\left(d^K F(M+N)\right)$. Both expressions are presented in *plaintext*. Unlike the $O\left(N^2(d+F)\right)$ complexity in *cyphertext* of the naive encryption-based solution, our method achieves a linear complexity ($O(N)$) instead of quadratic complexity ($O(N^2)$) with respect to the number of nodes N. This is particularly significant because, in real-world applications, the number of nodes is often the primary scaling factor, and our approach demonstrates improved scalability in this regard.

4.3 Integration with Temporal Modeling

To integrate the restructured spatial modeling with temporal modeling into a cohesive federated ASTGNN model, we replace the linear layers in GRU with our reformulated AGCN layers (similar to Eq. (7)). We first redefine the spatial term $\boldsymbol{Z}_i$, treating it as a

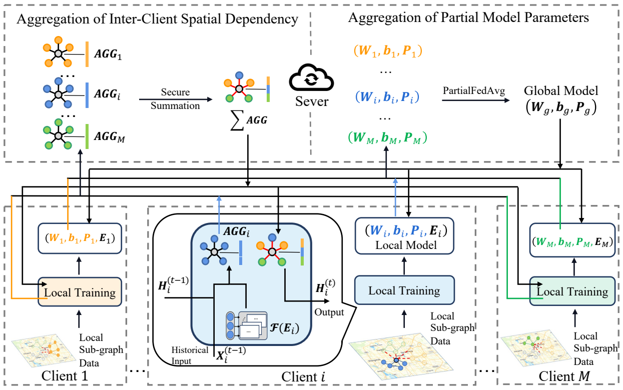


Figure 4: Overview of the FedGTP System.

function with input $\boldsymbol{H}$, which varies across different GRU gates:

$$Z_i(H) = \boldsymbol{H}_i + \sum_{k=0}^{K}\left(p_{i,k} \cdot f_k(\boldsymbol{E}_i) \cdot \sum_{j=1}^{M}\left(f_k^{\top}(\boldsymbol{E}_j) \cdot \boldsymbol{H}_j\right)\right)$$
$$= \boldsymbol{H}_i + \boldsymbol{P}_i \cdot \mathcal{F}_K(\boldsymbol{E}_i) \cdot \sum_{j=1}^{M}\left(\mathcal{F}_K^{\top}(\boldsymbol{E}_j) \cdot \boldsymbol{H}_j\right). \tag{25}$$

Then, we obtain our federated ASTGNN at t-th time slot:

$$\begin{aligned} z_i^t &= \sigma\left(Z_i\left([\boldsymbol{X}^t \,||\, \boldsymbol{h}^{t-1}]\right) \cdot \boldsymbol{E}_i \cdot \boldsymbol{W}_{i,z} + \boldsymbol{E}_i \cdot \boldsymbol{b}_{i,z}\right) \\ r_i^t &= \sigma\left(Z_i\left([\boldsymbol{X}^t \,||\, \boldsymbol{h}^{t-1}]\right) \cdot \boldsymbol{E}_i \cdot \boldsymbol{W}_{i,r} + \boldsymbol{E}_i \cdot \boldsymbol{b}_{i,r}\right) \\ \tilde{\boldsymbol{h}}_i^t &= tanh\left(Z_i\left([\boldsymbol{X}^t \,||\, (\boldsymbol{r}^t \odot \boldsymbol{h}^{t-1})]\right) \cdot \boldsymbol{E}_i \cdot \boldsymbol{W}_{i,\tilde{\boldsymbol{h}}} + \boldsymbol{E}_i \cdot \boldsymbol{b}_{i,\tilde{\boldsymbol{h}}}\right) \\ \boldsymbol{h}_i^t &= z_i^t \odot \boldsymbol{h}_i^{t-1} + (1 - z_i^t) \odot \tilde{\boldsymbol{h}}_i^t, \end{aligned} \tag{26}$$

In the above equations, $\boldsymbol{E}_i, \boldsymbol{P}_i, \boldsymbol{W}_{i,z}, \boldsymbol{W}_{i,r}, \boldsymbol{W}_{i,\hat{\boldsymbol{h}}}, \boldsymbol{b}_{i,z}, \boldsymbol{b}_{i,r}$ and $\boldsymbol{b}_{i,\hat{\boldsymbol{h}}}$ are the learnable parameters for client C_i, they can all be trained end-to-end with back-propagation through time.

4.4 System Implementation and Analysis

4.4.1 System Implementation. Previous sections have detailed our spatial modeling and its integration with temporal modeling in a federated context. Now we present the system design and implementation for FedGTP training, where M clients jointly train the models under the orchestration of a server. To adapt to real federated scenarios where each client has its own computing machine, we build up a system based on socket communication, allowing it to be deployed on distributed environments in real world. Fig. 4 illustrates the overview of FedGTP system, and Algorithm 1-3 details the process.

In local training round, each client C_i performs forward propagation in parallel on the spatiotemporal model by Eq. (26) with its local parameters. When inter-client spatial dependencies are required, the activation decomposition mechanism is activated according to Eq. (25). Each C_i first transforms $\boldsymbol{E}_i$ and aggregates it with $\boldsymbol{H}_i$ into $\boldsymbol{AGG}_i$ containing local spatial information, which is then uploaded to server. These aggregated intermediate results do not violate privacy standards such as GDPR [38]. For enhanced privacy protection, *e.g.* against inferences from intermediate results, we can employ established secure multi-party summation methods such as Homomorphic Encryption [34], Secret Sharing [41], Differential Privacy [11], and other MPC Protocols [5, 25]. While the specifics of these secure summation techniques are beyond our scope, they serve as adaptable modules, balancing security, accuracy, and efficiency according to the targeting scenarios.

Algorithm 1: FedGTP Framework

```
input  : Initial global model weights $(\boldsymbol{W}^{(0)}, \boldsymbol{b}^{(0)}, \boldsymbol{P}^{(0)})$;
         Initial personalized node embeddings $\{\boldsymbol{E}_i^{(0)}\}_{i=1}^{M}$;
         The number of global and local rounds $R_g, R_l$;
output : Trained model weights $(\boldsymbol{W}_i, \boldsymbol{b}_i, \boldsymbol{P}_i, \boldsymbol{E}_i)$ for each $C_i$;
Initialize global model weights with $(\boldsymbol{W}^{(0)}, \boldsymbol{b}^{(0)}, \boldsymbol{P}^{(0)})$;
for each client $C_i \in C$ in parallel do
    Initialize personalized node embeddings with $\boldsymbol{E}_i^{(0)}$;
for global round $r_g = 1, 2, ..., R_g$ do
    for each client $C_i \in C$ in parallel do
        Receives global model weights from server to
          update $\boldsymbol{W}_i, \boldsymbol{b}_i, \boldsymbol{P}_i$;
        for local round $r_l = 1, 2, ..., R_l$ do
            Forwards spatial-temporal modeling according
              to Eq. (26), during which performs inter-client
              spatial aggregation according to Eq. (25) (see
              Algorithm 2).
            Update $(\boldsymbol{W}_i, \boldsymbol{b}_i, \boldsymbol{P}_i, \boldsymbol{E}_i)$ through gradient descent.
        Sends $(\boldsymbol{W}_i, \boldsymbol{b}_i, \boldsymbol{P}_i)$ to server;
    Server performs $PartialFedAvg$ to update $(\boldsymbol{W}_g, \boldsymbol{b}_g, \boldsymbol{P}_g)$;
return $(\boldsymbol{W}_i, \boldsymbol{b}_i, \boldsymbol{P}_i, \boldsymbol{E}_i)$ for each $C_i$;
```

Algorithm 2: Inter-Client Spatial Aggregation

```
input  : Input state $\boldsymbol{H}_i^t$ on $C_i$;
output : Output state $Z_i^t$ aggregating inter-client spatial
         dependencies;
//Transform node embedding matrix
$\mathcal{F}_K(\boldsymbol{E}_i) \leftarrow \{f_k(\boldsymbol{E}_i)\}_{k=0}^{K}$;
//Aggregation of local spatial dependencies
$AGG_i \leftarrow \mathcal{F}_K^{\top}(\boldsymbol{E}_i) \cdot \boldsymbol{H}_i^t$;
Send $AGG_i$ to server;
Receive $\sum AGG \leftarrow SecureSummation$ from server;
//Aggregation of inter-client spatial dependencies
$Z_i^t = \boldsymbol{H}_i^t + \boldsymbol{P}_i \cdot \mathcal{F}_K(\boldsymbol{E}_i) \cdot \sum AGG$;
return $Z_i^t$;
```

Algorithm 3: Functions on Server

```
SecureSummation:
    Wait for each client to upload $\mathcal{F}_K^{\top}(\boldsymbol{E}_i) \cdot \boldsymbol{H}_i^t$;
    Sum and broadcast $\sum_{i=1}^{M}\left(\mathcal{F}_K^{\top}(\boldsymbol{E}_i) \cdot \boldsymbol{H}_i^t\right)$ to clients;
PartialFedAvg:
    Wait for each client to upload $(\boldsymbol{W}_i, \boldsymbol{b}_i, \boldsymbol{P}_i)$;
    $(\boldsymbol{W}_g, \boldsymbol{b}_g, \boldsymbol{P}_g) \leftarrow \sum_{i=1}^{M} \frac{N_i}{N} (\boldsymbol{W}_i, \boldsymbol{b}_i, \boldsymbol{P}_i)$;
```

The server further aggregates the uploaded information by *Secure-Summation* and broadcasts the cumulative result containing inter-client spatial dependencies to clients for subsequent computations. The learnable model parameters in C_i include weight pool $\boldsymbol{W}_i$, bias pool $\boldsymbol{b}_i$, polynomial coefficients $\boldsymbol{P}_i$ and node embedding matrix $\boldsymbol{E}_i$. Note that $\boldsymbol{E}_i$ contains local sub-graph-specific node embeddings for each C_i, and its size varies due to different node counts. To preserve the private region-specific characteristics in the embeddings and retain personalization, each client shares only $(\boldsymbol{W}_i, \boldsymbol{b}_i, \boldsymbol{P}_i)$ for partial aggregation (*PartialFedAvg*), while keeping $\boldsymbol{E}_i$ local.

4.4.2 Communication Cost Analysis. The communication cost of FedGTP is $O\left(|\Theta| \cdot M \cdot R_g + (M \cdot d^K \cdot F) \cdot R_l\right)$, which consists of two parts. The first part, $O\left(|\Theta| \cdot M \cdot R_g\right)$, arises from the aggregation of model parameters, a common aspect of all current federated graph learning methods, where $|\Theta|$ denotes the model size. Since we keep $\boldsymbol{E}_i$ personalized for each client, $|\Theta|$ is reduced. The second part, $O\left((M \cdot d^K \cdot F) \cdot R_l\right)$, is due to the aggregation of inter-client spatial dependency. Unlike state-of-the-art baselines which are proportional [32] to or quadratic [27, 53] to N, FedGTP incurs a cost linear with M, thanks to the Activation Decomposition Mechanism. Since each client manages hundreds to thousands of nodes locally, M is orders of magnitude smaller than N. Therefore our solution results in much lower communication costs than these baselines in the real-world cross-silo scenarios where N is in the tens of thousands, while M is usually around 10.

5 EXPERIMENTS

5.1 Experimental Setup

5.1.1 Datasets. We test on the following real-world traffic datasets: METR-LA, PeMS Data and TaxiFlow-BJ. *METR-LA* contains traffic speed collected from the highway of the Los Angeles County road network over 4 months. *PeMS Data*[3] comprises detector data of traffic speed, flow and occupancy from the Caltrans Performance Measurement System (PeMS), and different subsets of PeMS Data have been used in previous studies [4, 15, 32, 53]. *TaxiFlow-BJ* is a dataset assembled by collecting data from five taxi companies in Beijing, covering the period from June 1st to August 31st, 2013. We preprocess the raw taxi trajectory data and map it to the road network, which is obtained from OpenStreetMap[4]. Then, we identify the area with the highest concentration of mappings, which encompasses 1905 road segments. Based on the matching results, we can obtain traffic flow data for different road segments of each company, which can then be used to conduct federated traffic predictions among five clients. In graph modeling of traffic data, each node represents a sensor in METR-LA and PeMS Data, whereas in TaxiFlow-BJ, each node corresponds to a road segment. More details of the datasets are shown in Tab. 3.

5.1.2 Default Configuration and Environment. In the default hyper-parameter configurations, the hidden feature dimension $F = 64$ with 2 hidden layers. The embedding dimension $d = 2$, and the polynomial coefficient $K = 4$. The learning rate $\eta = 0.003$, and the batch size is 64. The models undergo 200 global epochs and 2 local epochs. Additionally, both the validation ratio and the testing ratio are set to 0.2. The performance metrics include Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE). A lower value of these metrics indicates a better prediction performance. All experiments are implemented with PyTorch 1.13.1 and conducted on Intel(R) Xeon(R) Gold 6230R CPU @ 2.10GHz and four NVIDIA A100-PCIE-40GB GPUs with CUDA 11.6.

5.2 Baseline Comparison

5.2.1 Baselines and Settings. To evaluate the overall effectiveness of our work, we align and compare FedGTP with state-of-the-art baselines in distinct federated settings where they show best performance. Due to the lack of available source code for some of these baselines (except for CNFGNN), we rely on the results reported in papers when our reproduced performance is sub-optimal. These compared baselines can be classified into two categories:

Baselines overlooking inter-client spatial dependencies. This category of baselines largely under-utilize inter-client spatial dependencies due to privacy constraints, including:

- FASTGNN [53]: It introduces a federated attention-based STGNN and constructs a random spatial connection among clients. The attention-based STGNN consists of a two-layer GRU, with layer dimensions of 64 and 256 respectively, along with a graph attention network. Following the setting in [53], we conduct traffic speed prediction on the PeMSD7 with $M = 4$, $T_{in} = 12$ and $T_{out} = 9$.
- MFVSTGNN [27]: It introduces a FL-based traffic forecasting model that utilizes a Variational Graph AutoEncoder (VGAE) to enhance intra-client dependencies and employs STGNNs for prediction. We compare our results with those reported in [27], which are the best (using Graph WaveNet for prediction) on the PEMS-BAY and METR-LA, with $M = 8$, $T_{in} = 12$, and $T_{out} = 12$.
- FLoS [43]: It constructs a FL framework with opportunistic client selection for traffic flow prediction. The local model uses GRU and GCN to capture spatial-temporal dependencies, and the hidden dimension is set to 32. The results are based on the PeMSD4 with $M = 5$, $T_{in} = 24$, and $T_{out} = 12$.

Baselines considering inter-client spatial dependencies. This category of baselines treat inter-client spatial dependencies as public and predefined, including:

- CNFGNN [32]: It uses a GRU-based model on each node (client) to extract the temporal features with local data, and performs GNN with a pre-defined graph on the server to capture inter-node spatial dependencies. The model on each node has 1 layer GRU with the hidden dimension of 64, while the model on the server is a 2-layer GNN. Following the same setting as in [32], we predict the traffic speed on PEMS-BAY and METR-LA with $T_{in} = 12$ and $T_{out} = 12$.
- FCGCN [47]: It proposes a framework that combines a two-layer GCN with FL. Both GCN layers include a ReLU activation. To ensure a fair comparison, we compare our approach with FCGCN in a setting where all clients participate in the FL aggregation process. The results are employed to predict

[3] http://pems.dot.ca.gov/
[4] https://www.openstreetmap.org/

Table 1: Comparison of performance on the traffic prediction task between FedGTP and baselines which do not consider inter-client spatial dependencies.

baseline	federated setting	task	method	RMSE	MAE	MAPE(%)
FASTGNN [53]	PeMSD7 (4 clients, 12->9)[5]	speed	FASTGNN	5.83	3.50	8.36
			FedGTP	**4.73**	**2.60**	**5.88**
MFVSTGNN [27]	PEMS-BAY (8 clients, 12->12)	speed	MFVSTGNN	3.91	1.93	4.48
			FedGTP	**3.88**	**1.79**	**3.54**
	METR-LA (8 clients, 12->12)	speed	MFVSTGNN	4.45	3.35	9.42
			FedGTP	**4.41**	**3.23**	**8.75**
FLoS [43]	PeMSD4 (5 clients, 24->12)	flow	FLoS	42.84	28.64	-[6]
			FedGTP	**41.33**	**25.62**	-

Table 2: Comparison of performance on the traffic prediction task between FedGTP and and baselines which consider inter-client spatial dependencies as public and predefined.

baseline	federated setting	task	method	RMSE	MAE	MAPE(%)
CNFGNN [32]	PEMS-BAY (325 clients, 12->12)	speed	CNFGNN	3.7090	2.3528	4.82
			FedGTP	**3.6440**	**1.6813**	**3.35**
	METR-LA (207 clients, 12->12)	speed	CNFGNN	11.4137	7.5161	36.26
			FedGTP	**10.3978**	**4.2883**	**24.83**
FCGCN [47]	PeMSD4 (28 clients, 6->1)	flow	FCGCN	29.6775	18.6483	22.57
			FedGTP	**26.6993**	**17.9049**	**13.99**
		speed	FCGCN	1.8777	1.0008	1.84
			FedGTP	**1.6800**	**0.9493**	**1.72**
		occ	FCGCN	0.1181	0.0066	18.92
			FedGTP	**0.0126**	**0.0064**	**16.03**
	PeMSD8 (14 clients, 6->1)	flow	FCGCN	22.4601	14.7723	11.81
			FedGTP	**20.4897**	**13.7754**	**9.09**
		speed	FCGCN	1.5570	0.8228	1.54
			FedGTP	**1.4221**	**0.7616**	**1.35**
		occ	FCGCN	0.0598	0.0058	12.91
			FedGTP	**0.0110**	**0.0054**	**10.05**
CTFL [54]	PeMSD4 (8 clients, 12->9)	speed	CTFL-STGCN	4.87	-	4.84
			CTFL-MTGNN	4.91	-	4.79
			FedGTP	**3.42**	-	**3.78**
	PeMSD7 (8 clients, 12->9)	speed	CTFL-STGCN	5.25	-	7.08
			CTFL-MTGNN	5.23	-	7.08
			FedGTP	**4.89**	-	**6.88**

traffic flow, speed, and occupancy on the PeMS04 ($M = 28$) and PeMS08 ($M = 14$) with $T_{in} = 6$ and $T_{out} = 1$.

- CTFL [54]: It designs a clustering-based FL framework for STGNNs to forecast traffic speed in the federated scenario. We utilize the results reported in [54], which include two STGNNs, namely STGCN[50] and MTGNN[46]. The experiments were conducted on PeMSD4 and PeMSD7 with $M = 8$, $T_{in} = 12$ and $T_{out} = 9$.

5.2.2 Results and Analysis. Tab. 1 and Tab. 2 present the comparison of prediction performance between FedGTP and the two categories of baselines. From Tab. 1 we can observe that our proposed FedGTP exhibits superior performance to the counterparts [27, 43, 53] that also protect spatial privacy, owing to the ability of FedGTP to fully recover and exploit the disrupted inter-client spatial dependencies. Moreover, as shown in Tab. 2, FedGTP consistently outperforms its competitors [32, 47, 54] that utilize inter-client spatial dependencies. This is primarily because these baselines rely on predefined spatial graphs, whereas FedGTP adaptively uncovers and leverages more profound inter-client spatial dependencies. Additionally, FedGTP provides stronger privacy protection compared to these baselines, as they treat the spatial graph as public. The overall results indicate that our proposed FedGTP has improved over existing works on average by 21.08%, 13.48%, 19.90% decrease on RMSE, MAE and MAPE, respectively.

5.3 Ablation Studies

5.3.1 Impact of Inter-Client Spatial Dependency Reconstruction. To quantitatively investigate the impact of inter-client spatial dependency reconstruction on the performance of FedGTP, we randomly eliminate these dependencies reconstructed involving each client and evaluate the prediction accuracy across a spectrum of elimination rates at $\{0.0, 0.5, 1.0\}$. Here, $\rho = 0.0$ represents the full utilization of inter-client spatial dependencies, whereas $\rho = 0.5$ indicates a partial, 50% elimination, and $\rho = 1.0$ corresponds to the complete elimination of these reconstructed dependencies. Due to space limitation, we present in Fig. 5a the normalized error in form of the three metrics on only PeMSD7 (results on other datasets are similar, shown in A.2). We can observe that the errors rise by around 8% upon the elimination of half the inter-client spatial dependencies, with a surge of up to 16.47% in error when these dependencies are completely removed. From the above results, we can conclude that the full reconstruction of inter-client spatial dependencies indeed boosts the accuracy of traffic prediction.

5.3.2 Impact of Activation Decomposition. To evaluate the effectiveness of our proposed activation decomposition mechanism, including *SprtReLU* and *AdptPoLU*, we run experiments on PeMSD7 (results on other datasets are similar, shown in A.2) with *ctr* and *sgl* as two controls. The term *ctr* denotes centralized training with all sub-graphs from clients joined together. The term *sgl* represents each single client training solely based on its local data. From the results in Fig. 5(b)-(d), we obtain the following observations:

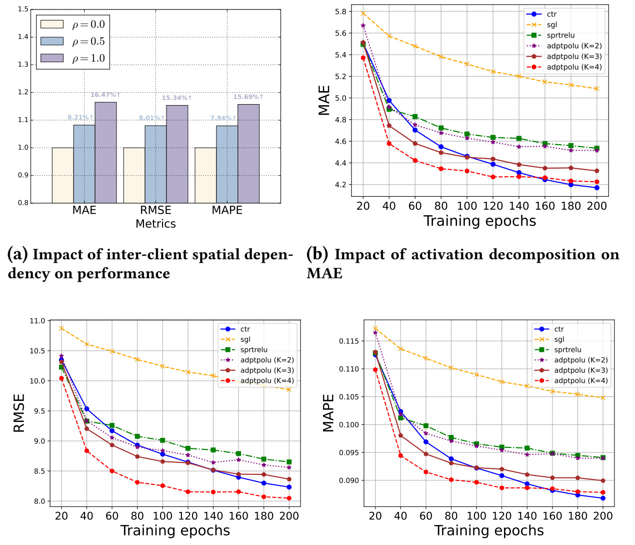
- The *SprtReLU* mechanism can help improve prediction accuracy compared to *sgl* by utilizing inter-client spatial dependency under privacy constraints. However, there is still a non-negligible gap compared to *ctr* due to information loss.
- The *AdptPoLU* mechanism can compensate the performance gap between *SprtReLU* and *ctr* by using adaptive polynomial approximation. *AdptPoLU* has the similar performance with *SprtReLU* when $K = 2$, and performs almost close to *ctr*, which we regard as optimal, when K rises to 4.
- The *AdptPoLU* mechanism will accelerate the convergence of training process when K rises, and sometimes it may even outperform *ctr* in RMSE (see Fig. 5c). We attribute it to the adaptive coefficients facilitating the training process.

5.4 Case Study

To develop an in-depth understanding of inter-client spatial dependency rooted in practical scenarios, we conduct a case study with the TaxiFlow-BJ dataset, which contains traffic data of five distinct taxi companies (each serving as a client), to carry out federated traffic flow prediction.

[5] The notation "T_{in}->T_{out}" is used to indicate the prediction of T_{out} time slots based on the previous T_{in} slots.

[6] The absence of values for certain baselines in the table is due to the inability to replicate these models and the lack of reported metrics in the original publications. The same applies to Tab. 2.



(a) Impact of inter-client spatial dependency on performance

(b) Impact of activation decomposition on MAE

(c) Impact of activation decomposition on RMSE

(d) Impact of activation decomposition on MAPE

Figure 5: Ablation results on impact of inter-client spatial dependency (a) and activation decomposition (b)-(d).

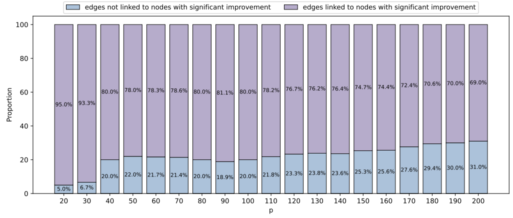


Figure 6: The strong correlation between performance improvement and the learned dependencies.

Firstly, we compare our FedGTP incorporating inter-client dependency against that without it, discovering that the former yields a decrease in MAE by around 3%. To delve deeper, we explore the relationship between the MAE decrease for each node and the adaptively learned weights of inter-client edges connecting it, which are determined through the product of node embeddings. Specifically, we select the top 20% nodes with the largest decrease in MAE as the most benefited ones. Regarding the extensive number of edges, we focus on the top p edges with the largest weights, which denote the key inter-client spatial dependencies. Fig. 6 illustrates the proportion of edges linked to the benefited nodes against those not linked, with p ranging from 20 to 200. We can observe that over 69% of these critical edges are linked to nodes exhibiting remarkable performance enhancement. This proportion reaches 95% especially for the top 20 edges, far exceeding the average probability level of 36%. This indicates that the adaptively learned inter-client connections play a vital role in reducing prediction errors, especially through effectively excavating complex relationships between nodes, thereby enhancing overall prediction accuracy.

Furthermore, we present a concrete example to demonstrate how adaptively learned inter-client spatial dependencies contribute to performance improvement. We focus our gaze on one of the significantly benefited nodes and its associated key dependencies. As depicted in Fig. 7, this node is identified as a road segment near a school, besides, all nodes linked through these dependencies are also road segments in the school surroundings. This observation complies to the intuition that the same functional areas possess similar spatiotemporal characteristics, and it is logical to infer that such semantic dependencies can enhance prediction performance. This example also underlines that even though the data of these interacted road segments are dispersed across multiple clients with no direct access, our method can still overcome this barrier to fully utilize and learn the inter-client spatial dependencies.

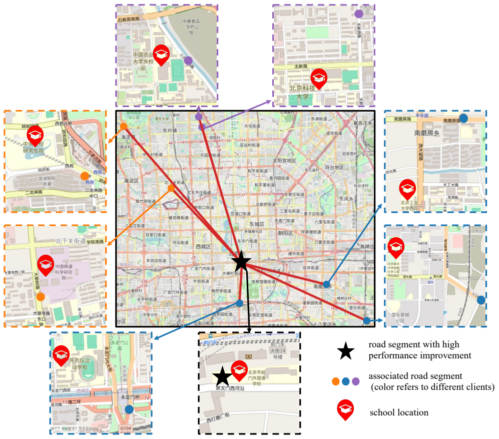

road segment with high performance improvement
associated road segment (color refers to different clients)
school location

Figure 7: Case study of a concrete node with significant performance improvement.

6 CONCLUSION

In this paper, we propose FedGTP, a pioneering framework designed for federated graph-based traffic prediction that fully exploits inter-client spatial dependencies with privacy preservation. FedGTP progresses by adaptive learning of inter-client spatial dependencies, enabling deeper exploration of spatial relationships and thus boosting prediction performance. For privacy protection, we then introduce an innovative polynomial-based activation decomposition mechanism that ensures compliance with privacy regulations like GDPR during the reconstruction of inter-client spatial dependencies. Extensive experiments on real-world traffic datasets have been conducted to validate our approach. The results demonstrate that FedGTP significantly outperforms existing baselines in prediction performance, achieving near-optimal results while adhering to privacy constraints, thus confirming its effectiveness. Future work will aim at improving FedGTP's communication efficiency to address challenges arising from asynchronous scenarios for better accommodation in large-scale applications.

ACKNOWLEDGMENTS

We thank the reviewers for their constructive comments. This work is partially supported by National Science Foundation of China (NSFC) under Grant Nos. U21A20516 and 62336003, Beijing Natural Science Foundation under Grant No. Z230001, Beihang University Basic Research Funding No. YWF-22-L-531, Chow Sang Sang Group Research Fund No. 9229139, Didi Collaborative Research Program NO2231122-00047, Yongxin Tong is the corresponding author.

REFERENCES

[1] Abbas Acar, Hidayet Aksu, A. Selcuk Uluagac, and Mauro Conti. 2018. A survey on homomorphic encryption schemes: Theory and implementation. *ACM Comput. Surv.* 51, 4 (2018), 1–35.

[2] Lei Bai, Lina Yao, Salil S. Kanhere, Xianzhi Wang, et al. 2019. STG2Seq: Spatial-Temporal Graph to Sequence Model for Multi-step Passenger Demand Forecasting. In *IJCAI*. ijcai.org, 1981–1987.

[3] Lei Bai, Lina Yao, Salil S. Kanhere, Zheng Yang, et al. 2019. Passenger Demand Forecasting with Multi-Task Convolutional Recurrent Neural Networks. In *PAKDD*, Vol. 11440. Springer, 29–42.

[4] Lei Bai, Lina Yao, Can Li, Xianzhi Wang, et al. 2020. Adaptive Graph Convolutional Recurrent Network for Traffic Forecasting. In *NeurIPS*.

[5] Donald Beaver. 1991. Efficient Multiparty Protocols Using Circuit Randomization. In *CRYPTO*, Vol. 576. Springer, 420–432.

[6] Weiqi Chen, Ling Chen, Yu Xie, Wei Cao, et al. 2020. Multi-Range Attentive Bicomponent Graph Convolutional Network for Traffic Forecasting. In *AAAI*. AAAI Press, 3529–3536.

[7] Yuzhou Chen, Ignacio Segovia-Dominguez, Baris Coskunuzer, and Yulia R. Gel. 2022. TAMP-S2GCNets: Coupling Time-Aware Multipersistence Knowledge Representation with Spatio-Supra Graph Convolutional Networks for Time-Series Forecasting. In *ICLR*. OpenReview.net.

[8] Yuzhou Chen, Ignacio Segovia-Dominguez, and Yulia R. Gel. 2021. Z-GCNETs: Time Zigzags at Graph Convolutional Networks for Time Series Forecasting. In *ICML*, Vol. 139. PMLR, 1684–1694.

[9] Kyunghyun Cho, Bart Merrienboer, Caglar Gulcehre, and Dzmitry Bahdanau others. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *EMNLP*. ACL, 1724.

[10] Jeongwhan Choi, Hwangyong Choi, Jeehyun Hwang, and Noseong Park. 2022. Graph Neural Controlled Differential Equations for Traffic Forecasting. In *AAAI*. AAAI Press, 6367–6374.

[11] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. 2006. Calibrating Noise to Sensitivity in Private Data Analysis. In *TCC*, Vol. 3876. Springer, 265–284.

[12] Alireza Fallah, Aryan Mokhtari, and Asuman E. Ozdaglar. 2020. Personalized Federated Learning with Theoretical Guarantees: A Model-Agnostic Meta-Learning Approach. In *NeurIPS*.

[13] Karthik Garimella, Nandan Kumar Jha, and Brandon Reagen. 2021. Sisyphus: A Cautionary Tale of Using Low-Degree Polynomial Activations in Privacy-Preserving Deep Learning. *CoRR* abs/2107.12342 (2021).

[14] Xu Geng, Yaguang Li, Leye Wang, Lingyu Zhang, et al. 2019. Spatiotemporal Multi-Graph Convolution Network for Ride-Hailing Demand Forecasting. In *AAAI*. AAAI Press, 3656–3663.

[15] Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, et al. 2019. Attention Based Spatial-Temporal Graph Convolutional Networks for Traffic Flow Forecasting. In *AAAI*. AAAI Press, 922–929.

[16] Shengnan Guo, Youfang Lin, Huaiyu Wan, Xiucheng Li, et al. 2022. Learning Dynamics and Heterogeneity of Spatial-Temporal Graph Data for Traffic Forecasting. *IEEE Trans. Knowl. Data Eng.* 34, 11 (2022), 5415–5428.

[17] Jizhou Huang, Zhengjie Huang, Xiaomin Fang, Shikun Feng, et al. 2022. DuETA: Traffic Congestion Propagation Pattern Modeling via Efficient Graph Learning for ETA Prediction at Baidu Maps. In *CIKM*. ACM, 3172–3181.

[18] Rongzhou Huang, Chuyin Huang, Yubao Liu, Genan Dai, et al. 2020. LSGCN: Long Short-Term Traffic Prediction with Graph Convolutional Networks. In *IJCAI*. ijcai.org, 2355–2361.

[19] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*. OpenReview.net.

[20] Xiangyuan Kong, Jian Zhang, Xiang Wei, Weiwei Xing, et al. 2022. Adaptive spatial-temporal graph attention networks for traffic flow forecasting. *Appl. Intell.* 52, 4 (2022), 4300–4316.

[21] Tian Lan, Ziyue Li, Zhishuai Li, Lei Bai, et al. 2023. MM-DAG: Multi-task DAG Learning for Multi-modal Data - with Application for Traffic Congestion Analysis. In *KDD*. ACM, 1188–1199.

[22] Can Li and Wei Liu. 2023. Multimodal Transport Demand Forecasting via Federated Learning. *IEEE Trans. Intell. Transp. Syst.* (2023).

[23] Wenzhu Li and Shuang Wang. 2022. Federated meta-learning for spatial-temporal prediction. *Neural Comput. Appl.* 34, 13 (2022), 10355–10374.

[24] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. 2018. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In *ICLR*. OpenReview.net.

[25] Yehuda Lindell. 2020. Secure Multiparty Computation (MPC). *IACR Cryptol. ePrint Arch.* (2020), 300.

[26] Fan Liu, Weijia Zhang, and Hao Liu. 2023. Robust Spatiotemporal Traffic Forecasting with Reinforced Dynamic Adversarial Training. In *KDD*. ACM, 1417–1428.

[27] Lei Liu, Yuxing Tian, Chinmay Chakraborty, Jie Feng, et al. 2023. Multilevel Federated Learning-Based Intelligent Traffic Flow Forecasting for Transportation Network Management. *IEEE Trans. Netw. Serv. Manag.* 20, 2 (2023), 1446–1458.

[28] Qingxiang Liu, Sheng Sun, Min Liu, Yuwei Wang, et al. 2023. Online Spatio-Temporal Correlation-Based Federated Learning for Traffic Flow Forecasting. *CoRR* abs/2302.08658 (2023).

[29] Rui Liu and Han Yu. 2024. Federated Graph Neural Networks: Overview, Techniques and Challenges. *IEEE Trans. Neural Networks Learn. Syst.* (2024).

[30] Qian Lou, Yilin Shen, Hongxia Jin, and Lei Jiang. 2021. SAFENet: A Secure, Accurate and Fast Neural Network Inference. In *ICLR*. OpenReview.net.

[31] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, et al. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *AISTATS*. 1273–1282.

[32] Chuizheng Meng, Sirisha Rambhatla, and Yan Liu. 2021. Cross-Node Federated Graph Neural Network for Spatio-Temporal Data Modeling. In *KDD*. ACM, 1202–1211.

[33] Pratyush Mishra, Ryan Lehmkuhl, Akshayaram Srinivasan, Wenting Zheng, et al. 2020. Delphi: A Cryptographic Inference Service for Neural Networks. In *USENIX Security Symposium*. USENIX Association, 2505–2522.

[34] Pascal Paillier. 1999. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In *EUROCRYPT*, Vol. 1592. Springer, 223–238.

[35] Zheyi Pan, Yuxuan Liang, Weifeng Wang, Yong Yu, et al. 2019. Urban Traffic Prediction from Spatio-Temporal Data Using Deep Meta Learning. In *KDD*. ACM, 1720–1730.

[36] Hongwu Peng, Shaoyi Huang, Tong Zhou, Yukui Luo, et al. 2023. Autorep: Automatic relu replacement for fast private network inference. IEEE, 5155–5165.

[37] Le Trieu Phong, Yoshinori Aono, Takuya Hayashi, Lihua Wang, et al. 2017. Privacy-Preserving Deep Learning via Additively Homomorphic Encryption. *IEEE Trans. Inf. Forensics Secur.* 13, 5 (2017), 1333–1345.

[38] Emanuela Podda. 2021. Shedding light on the legal approach to aggregate data under the GDPR & the FFDR. In *conference of European statisticians Expert Meeting on Statistical Data Confidentiality*.

[39] Tao Qi, Lingqiang Chen, Guanghui Li, Yijing Li, et al. 2023. FedAGCN: A traffic flow prediction framework based on federated learning and Asynchronous Graph Convolutional Network. *Appl. Soft Comput.* 138 (2023), 110175.

[40] Ronald L Rivest, Len Adleman, Michael L Dertouzos, et al. 1978. On data banks and privacy homomorphisms. *Found. of Sec. Comp.* 4, 11 (1978), 169–180.

[41] Adi Shamir. 1979. How to Share a Secret. *Commun. ACM* 22, 11 (1979), 612–613.

[42] Beibei Wang, Youfang Lin, Shengnan Guo, and Huaiyu Wan. 2021. GSNet: learning spatial-temporal correlations from geographical and semantic aspects for traffic accident risk forecasting. In *AAAI*. AAAI Press, 4402–4409.

[43] Hanqiu Wang, Rongqing Zhang, Xiang Cheng, and Liuqing Yang. 2022. Federated Spatio-Temporal Traffic Flow Prediction Based on Graph Convolutional Network. In *WCSP*. IEEE, 221–225.

[44] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, et al. 2021. A Comprehensive Survey on Graph Neural Networks. *IEEE Trans. Neural Networks Learn. Syst.* 32, 1 (2021), 4–24.

[45] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, et al. 2019. Graph WaveNet for Deep Spatial-Temporal Graph Modeling. In *IJCAI*. ijcai.org, 1907–1913.

[46] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, et al. 2020. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *KDD*. ACM, 753–763.

[47] Mengran Xia, Dawei Jin, and Jingyu Chen. 2023. Short-Term Traffic Flow Prediction Based on Graph Convolutional Networks and Federated Learning. *IEEE Trans. Intell. Transp. Syst.* 24, 1 (2023), 1191–1203.

[48] Sijie Yan, Yuanjun Xiong, and Dahua Lin. 2018. Spatial Temporal Graph Convolutional Networks for Skeleton-Based Action Recognition. In *AAAI*. AAAI Press, 7444–7452.

[49] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. 2019. Federated Machine Learning: Concept and Applications. *ACM Trans. Intell. Syst. Technol.* 10, 2 (2019), 12:1–12:19.

[50] Bing Yu, Haoteng Yin, and Zhanxing Zhu. 2018. Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting. In *IJCAI*. ijcai.org, 3634–3640.

[51] Le Yu, Bowen Du, Xiao Hu, Leilei Sun, and other. 2021. Deep spatio-temporal graph convolutional network for traffic accident prediction. *Neurocomputing* 423 (2021), 135–147.

[52] Xiaoming Yuan, Jiahui Chen, Jiayu Yang, Ning Zhang, et al. 2023. FedSTN: Graph Representation Driven Federated Learning for Edge Computing Enabled Urban Traffic Flow Prediction. *IEEE Trans. Intell. Transp. Syst.* 24, 8 (2023), 8738–8748.

[53] Chenhan Zhang, Shuyu Zhang, JQ James, and Shui Yu. 2021. FASTGNN: A topological information protected federated learning approach for traffic speed forecasting. *IEEE Trans. Ind. Informatics* 17, 12 (2021), 8464–8474.

[54] Chenhan Zhang, Shiyao Zhang, Shui Yu, and James Yu. 2022. Graph-Based Traffic Forecasting via Communication-Efficient Federated Learning. In *WCNC*. IEEE, 2041–2046.

[55] Chuanpan Zheng, Xiaoliang Fan, Cheng Wang, and Jianzhong Qi. 2020. GMAN: A Graph Multi-Attention Network for Traffic Prediction. In *AAAI*. AAAI Press, 1234–1241.

[56] Zhengyang Zhou, Yang Wang, Xike Xie, Lianliang Chen, et al. 2020. RiskOracle: A minute-level citywide traffic accident forecasting framework. In *AAAI*. AAAI Press, 1258–1265.

A APPENDIX

A.1 Proof of Theorem 1

Proof. Without loss of generality, we focus on an arbitrary element $e_{r,s}$ on the r-th row and s-th column of the matrix $(\boldsymbol{E}_i \cdot \boldsymbol{E}_j^\top)$ after multiplication, where $1 \leq r \leq N_i$ and $1 \leq s \leq N_j$. It is resulted from the inner product of the r-th row vector of $\boldsymbol{E}_i$ and the s-th row vector of $\boldsymbol{E}_j$:

$$e_{r,s} = \left\langle \boldsymbol{e}_{i,r} \, , \boldsymbol{e}_{j,s} \right\rangle ,$$

the k-th power of which is

$$\begin{aligned} e_{r,s}^k &= \left(\begin{bmatrix} e_{i,r}^1 & \cdots & e_{i,r}^d \end{bmatrix} \cdot \begin{bmatrix} e_{j,s}^1 \\ \vdots \\ e_{j,s}^d \end{bmatrix} \right)^k \\ &= \left(e_{i,r}^1 e_{j,s}^1 + e_{i,r}^2 e_{j,s}^2 + \ldots + e_{i,r}^d e_{j,s}^d \right)^k \\ &= \left\langle (\boldsymbol{e}_{i,r} \otimes \boldsymbol{e}_{i,r} \otimes \cdots \otimes \boldsymbol{e}_{i,r}) \, , (\boldsymbol{e}_{j,s} \otimes \boldsymbol{e}_{j,s} \otimes \cdots \otimes \boldsymbol{e}_{j,s}) \right\rangle \\ &= \left\langle \otimes^k \boldsymbol{e}_{i,r}, \otimes^k \boldsymbol{e}_{j,s} \right\rangle , \end{aligned}$$

where $\otimes$ denotes the Cartesian product between vectors, and $\langle \cdot, \cdot \rangle$ indicates the inner product between vectors.

Therefore,

$$(\boldsymbol{E}_i \cdot \boldsymbol{E}_j^\top)^k = (\otimes^k \boldsymbol{E}_i) \cdot (\otimes^k \boldsymbol{E}_j)^\top ,$$

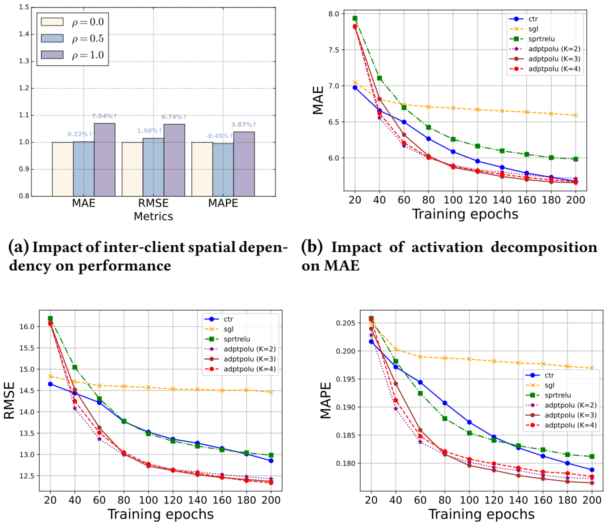
and we can set $f_k(\boldsymbol{E}_i) = \otimes^k \boldsymbol{E}_i$. □

A.2 Additional Experiment Results

Table 3: Statistics of datasets.

Dataset		Period(m/d/y)	#Intervals	#Nodes	Max	Min	Mean	Median	Std
METR-LA		03/01/12-06/30/12	34272	207	70	0	53.719	62.4444	20.2614
PEMS-BAY		01/01/17-05/31/17	52116	325	85.1	0	62.6196	65.3	9.5944
PeMSD4	Flow	01/01/18-02/28/18	16992	307	919	0	211.7008	180	158.0684
	Occupancy				0.7716	0	0.0528	0.0443	0.0495
	Speed				85.2	3	63.4706	65.6	8.3557
PeMSD8	Flow	07/01/16-08/31/16	17856	170	1147	0	230.6807	215	146.217
	Occupancy				0.8955	0	0.0651	0.0601	0.0459
	Speed				82.3	3	63.763	64.9	6.652
PeMSD7		05/01/12-06/30/12	12672	228	82.6	3	58.8892	64.1	13.4833

Fig. 8-Fig. 15 are results of ablation studies on more datasets.



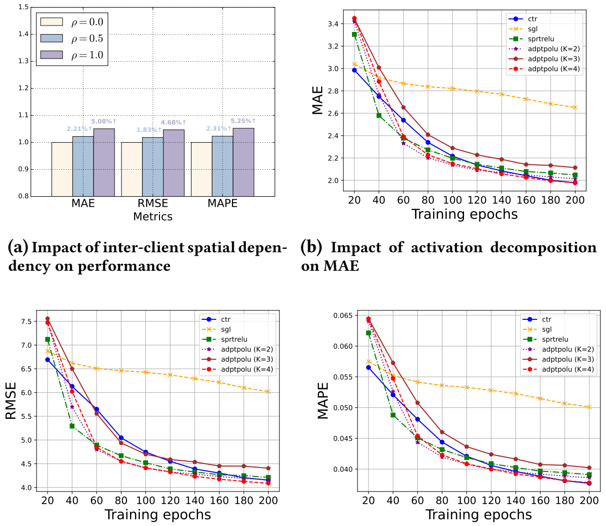
(a) Impact of inter-client spatial dependency on performance

(b) Impact of activation decomposition on MAE

(c) Impact of activation decomposition on RMSE

(d) Impact of activation decomposition on MAPE

Figure 8: Ablation results on impact of inter-client dependency (a) and activation decomposition (b)-(d) on METR-LA.



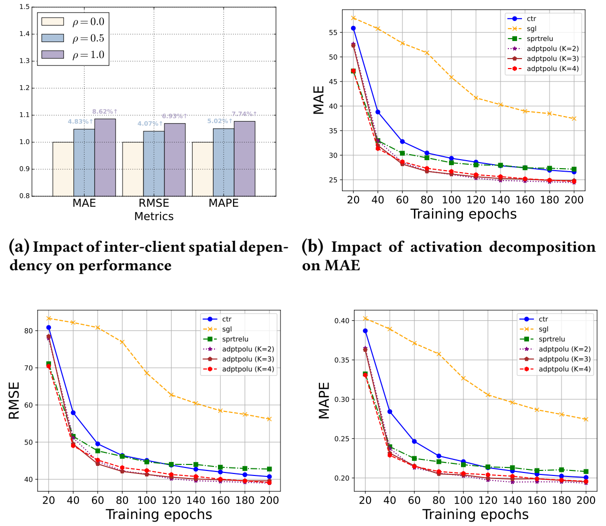
(a) Impact of inter-client spatial dependency on performance

(b) Impact of activation decomposition on MAE

(c) Impact of activation decomposition on RMSE

(d) Impact of activation decomposition on MAPE

Figure 9: Ablation Study on inter-client spatial dependency (a) and activation decomposition (b)-(d) on PEMS-BAY.



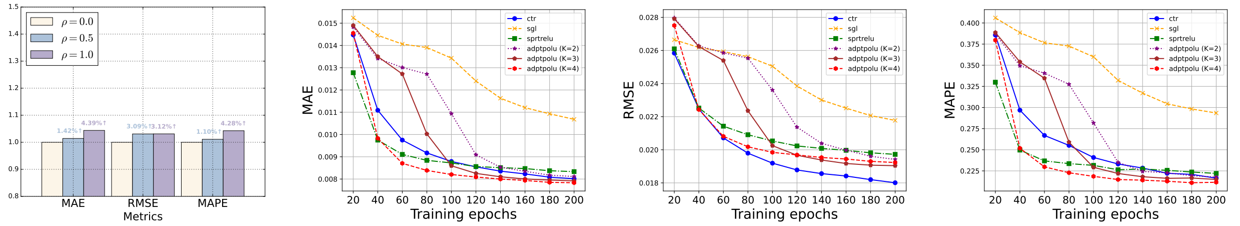
(a) Impact of inter-client spatial dependency on performance

(b) Impact of activation decomposition on MAE

(c) Impact of activation decomposition on RMSE

(d) Impact of activation decomposition on MAPE

Figure 10: Ablation Study on inter-client spatial dependency (a) and activation decomposition (b)-(d) on PeMSD4-FLOW.



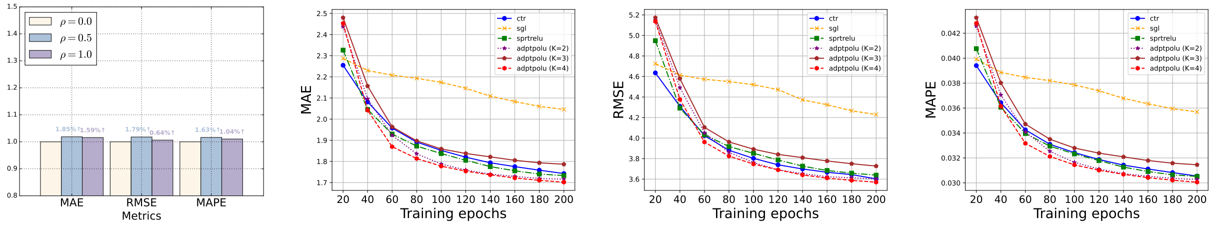
(a) Impact of inter-client spatial dependency on performance

(b) Impact of activation decomposition on MAE

(c) Impact of activation decomposition on RMSE

(d) Impact of activation decomposition on MAPE

Figure 11: Ablation Study on inter-client spatial dependency (a) and activation decomposition (b)-(d) on PeMSD4-OCCUP.



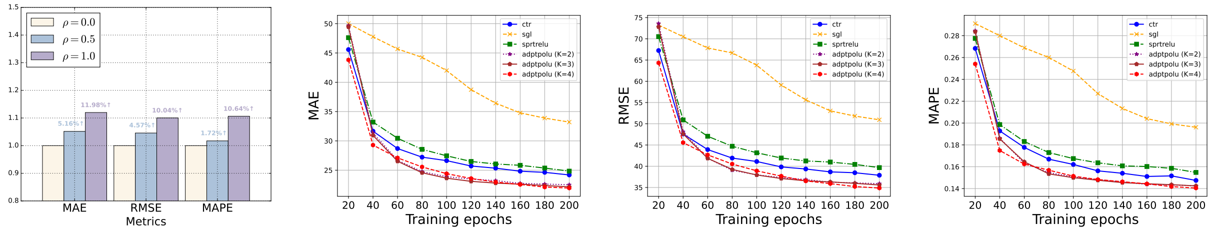
(a) Impact of inter-client spatial dependency on performance

(b) Impact of activation decomposition on MAE

(c) Impact of activation decomposition on RMSE

(d) Impact of activation decomposition on MAPE

Figure 12: Ablation Study on inter-client spatial dependency (a) and activation decomposition (b)-(d) on PeMSD4-SPEED.



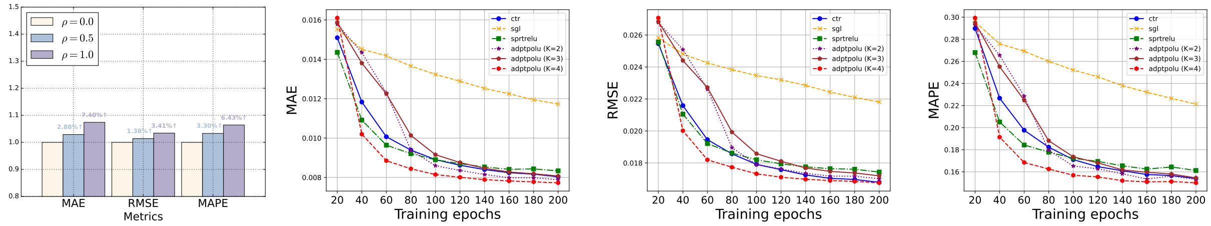
(a) Impact of inter-client spatial dependency on performance

(b) Impact of activation decomposition on MAE

(c) Impact of activation decomposition on RMSE

(d) Impact of activation decomposition on MAPE

Figure 13: Ablation Study on inter-client spatial dependency (a) and activation decomposition (b)-(d) on PeMSD8-FLOW.



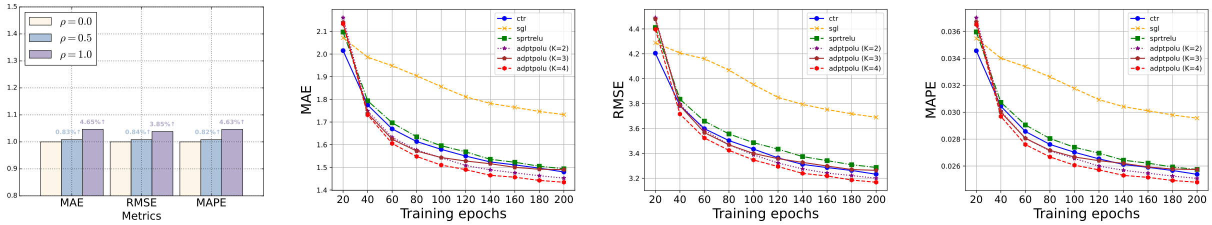
(a) Impact of inter-client spatial dependency on performance

(b) Impact of activation decomposition on MAE

(c) Impact of activation decomposition on RMSE

(d) Impact of activation decomposition on MAPE

Figure 14: Ablation Study on inter-client spatial dependency (a) and activation decomposition (b)-(d) on PeMSD8-OCCUP.



(a) Impact of inter-client spatial dependency on performance

(b) Impact of activation decomposition on MAE

(c) Impact of activation decomposition on RMSE

(d) Impact of activation decomposition on MAPE

Figure 15: Ablation Study on inter-client spatial dependency (a) and activation decomposition (b)-(d) on PeMSD8-SPEED.