FedSkeleton: Secure Multi-Party Graph Skeleton Construction for Privacy-Preserving Federated Time-Series Forecasting

Anonymous submission

Abstract

In real-world time-series modelling, graph structures are widely adopted because they explicitly encode node topology and capture complex network dynamics. In practice, however, a complete graph is often partitioned across multiple parties; each party can access only its local sub-graph and, owing to privacy regulations, cannot share topology or data, creating pervasive data silos. Federated Graph Learning (FGL) offers a privacy-preserving collaborative-learning paradigm, yet current methods still face two key challenges: (1) they implicitly capture inter-edge information, making it difficult to accurately reconstruct the global structure and consequently degrading model performance; (2) explicitly exchanging inter-edge information may leak graph-topology privacy. To overcome these obstacles, we propose FedSkeleton, a privacy-preserving framework for time-series prediction that comprises a Skeleton Construction Module and a Dual-stream Forecasting Module, enabling global dependency capture without revealing the topology. Extensive experiments show that FedSkeleton consistently outperforms existing baselines and even surpasses centralised models with full-graph access. In addition, we conduct comprehensive security analysis, communication-cost evaluation and scalability experiments, demonstrating that FedSkeleton effectively resists common attacks, keeps communication overhead manageable and remains robust with respect to key hyper-parameters and the number of participating parties. Our code and datasets are available at the following link[1].

1 Introduction

Numerous real-world high-dimensional time series in real-world scenarios are formed by dynamically changing attributes of individual entities, whose evolution is typically driven by their interactions. These entities (as nodes) and their interactions (as edges) jointly constitute a graph structure, such as financial networks and electric networks. Therefore, the problem of forecasting these high-dimensional time series can be naturally converted into a nodal dynamics prediction problem on graphs, which is instrumental for a wide spectrum of real-world applications including anomaly detection (Miele, Bonacina, and Corsini 2022) and efficiency optimization (Shen et al. 2022). However, in practical applications, the data of different nodes is often held by different organizations (e.g., governmental agencies, and corporations). Due to privacy concerns, these organizations cannot disclose the attributes of their nodes or even the topological relationships among them. Under the circumstances, an important research question is how to efficiently implement federated learning between different organizations to forecast their nodal dynamics, while preserving the privacy of the graph data of each participating organization.

[1]https://anonymous.4open.science/status/FedSkeleton-BCFA

However, the problem of graph federated time-series forecasting is also a difficult task with several challenges unsolved. First, existing federated graph learning algorithms typically require the propagation and aggregation of node embeddings across the complete graph topology (Chen et al. 2021; Liu, Li, and Gu 2021). Regardless of the private information involved in node embeddings (e.g., node attributes), the graph topology itself is sensitive and potentially leads to privacy leaks. For example, in financial networks, edge information can disclose monetary transactions between nodes. Thus, achieving federated time-series forecasting without revealing node attributes as well as topological information is a critical challenge. Second, since nodal dynamics are inherently driven by interactions between nodes, the edges across different parties are a crucial factor that should be considered in the prediction task. Ignoring inter-party edges will lead to inevitable performance loss. How to incorporate inter-party interactions without revealing private graph data is the second challenge.

To solve these challenges, in this paper, we propose FedSkeleton, a federated graph learning framework to efficiently forecast time series on graphs while preserving the privacy of graph data of each participant. The core of FedSkeleton is an elaborately-designed secure multi-party graph skeleton construction algorithm that utilizes graph coarsening (Hashemi et al. 2024) and secret sharing (Blakley 1979) techniques to merge the attribute of K nodes into a supernode, guaranteeing k-anonymity (Sweeney 2002) while ensuring that the mapping between original nodes and supernodes is only known to the node owner. As a result, the private information of each node is obscured by aggregating K different nodes in the graph skeleton, which cannot be traced back to the original graphs. Therefore, the graph skeleton can encode both useful nodal dynamics and topological information of the original complete graph without leaking privacy, which addresses the first challenge. Further, based on the constructed graph skeleton, FedSkeleton employs a local-global dual-stream forecasting mechanism, where each participant trains a local Graph ODE model on their private graphs, while a global Graph ODE model is trained on the constructed graph skeleton at servers. By collaboratively integrating information from both local and global models during prediction, FedSkeleton effectively models inter-participant node interactions without compromising privacy.

Our contribution can be summarized threefold as follows:

- We propose a novel federated graph learning framework for time-series forecasting, which constructs graph skeletons through graph coarsening and secret sharing techniques to encode both useful nodal dynamics and topological information of the original complete graph without leaking privacy.
- We propose a local-global dual-stream forecasting mech-

anism that collaboratively integrates information from both local and global models during prediction, which can incorporate inter-participant node interactions and mitigate the performance decline caused by the non-disclosure of original graph data of different participants due to privacy concerns.

- Extensive experimental results on two real-world datasets demonstrating an average improvement of approximately 42.48% on average of FedSkeleton compared with baselines, demonstrating that FedSkeleton can effectively balance privacy protection and predictive performance while maintaining a reasonable communication overhead, and ensuring secure and efficient collaboration among parties without excessive data transmission.

2 Preliminaries

2.1 Secret Sharing

We adopt the classic n-out-of-n Shamir scheme (Shamir 1979), which splits a secret into n shares, one per participant, such that all n shares are required for reconstruction. The scheme is additively and Multiplicative homomorphic can be performed privately by first generating Beaver triples over the same shares (Beaver 1991).

As a concrete example, consider the 2-out-of-2 case. Alice splits her secret a into random shares (a_1, a_2) with $a = a_1 + a_2$; Bob does the same for b as (b_1, b_2). After exchanging a_2 and b_2, Alice holds (a_1, b_2) while Bob holds (b_1, a_2). They send the partial sums $(a_1 + b_2)$ and $(b_1 + a_2)$ to a combiner, who obtains $a+b = (a_1+b_2)+(b_1+a_2)$ but learns nothing about a or b. Thus the original secrets remain private throughout the protocol.

2.2 Inter-edges issue in Federated Graph Learning

Consider a power-grid network jointly formed by several utility companies. Let the global graph be $\mathcal{G} = (V, E)$ and the sub-graph owned by company i be $\mathcal{G}_i = (V_i, E_i)$ with node features $\mathbf{X}_i$ (historical generation time series). $\mathbf{X}_i$ and the local edges E_i are private to company i; in contrast, the geographic links between plants of different companies, $E_{ij} = E \cap (V_i \times V_j)$, are public *inter-edges*. Similar cross-organisation settings arise in banking transaction networks and mobile-operator base-station networks.

If these inter-edges are ignored during federated GNN training, each party learns on a disconnected sub-graph, and its local objective degrades to

$$F_i(\theta) = \mathcal{L}\big(\mathrm{GNN}(E_i, \mathbf{X}_i; \theta),\, y_i\big), \tag{1}$$

which misses inter-subgraph dependencies and ultimately harms global performance.

2.3 Problem Definition

In this work, we study a setting where the graph $\mathcal{G} = (V, E, X)$ is partitioned among m parties, with each party having access only to its own subset of the data. In particular, the data available to party i is represented by $\mathcal{G}_i = (V_i, E_i, \mathbf{X}_i)$, where $V_i \subset V$ denotes the set of nodes that belong internally to party i, and $\mathbf{X}_i = \{x_i(t) \mid t = 0, 1, 2, \ldots, T-1\}$ comprises the time-series data corresponding to those nodes. Since a node's time-series data is considered private, it is exclusively held by a single party. Moreover, we define the border nodes for party i as $V_i^* = \{v^* \mid (v^*, v) \in E,\ v^* \in V_i,\ v \notin V_i\}$. Thus, in this paper, we assume that different parties' internal nodes have $V_i \cap V_j = \emptyset,\ i \neq j$. E_i is the set of edges accessible by party i, including both inter edges $E_i^* = \{(v, v^*) \mid (v, v^*) \in E, v^* \in V_i^*, v \notin V_i\}$ and internal edges. The core task in our study is to build a FL framework each FL parties maintains a subgraph of the global graph and trains a model to predict future time-series with observed history time-series, which is defined as follows:

Definition 1 (Federated Time-Series Forecasting)**.** Given multiple sub-graphs $\{\mathcal{G}_i = (V_i, E_i, \mathbf{X}_i)\}_{i=1}^{M}$ owned by i'th FL party. **FedSkeleton** aims to coordinates all FL party collaborative training to forecast future time series $\hat{\mathbf{X}}_i = \{\hat{x}_i(t) \mid t = T, T+1, \ldots, T+\Delta t\}$ immediately following an observable historical time series it owns $\mathbf{X}_i^{obs} = \{\hat{x}_i(t) \mid t = T-\tau, T-\tau+1, \ldots, T-1\}$, meanwhile guarantees that both the internal node time series data and the graph topology remain private.

3 Method

3.1 Overall Framework

In this section, we present a detailed overview of the FedSkeleton framework, as depicted in Figure 1. The framework is composed of two main modules: the Dual-Stream Forecasting Module and the Skeleton Construction Module. The process of this framework is presented in Algorithm 1.

3.2 Graph Skeleton Construction

Local skeleton build Each FL participant independently constructs a local skeleton by first generating learnable node embeddings. The assignment matrix is then computed to determine the mapping from original nodes to supernodes, establishing a hierarchical structure. Finally, GCNs are applied to aggregate node features into supernodes, forming a coarsened representation that maintains the topological and dynamic properties of the original graph.

(i) Adaptive assignment matrix. To determine the number of supernodes, we utilize the reduction ratio γ. We initialize two sets of learnable embeddings: supernode embeddings $E_S \in \mathbb{R}^{N \times \gamma \times 2}$ and node embeddings $E \in \mathbb{R}^{N \times 2}$, where each embedding encapsulates the dynamic behavior of the associated original nodes. The assignment matrix is computed as follows:

$$P = \mathrm{softmax}(\tilde{E}_S \tilde{E}^T), \tag{4}$$

where the softmax function is applied row-wise. Here, $\tilde{E} = \mathrm{MLP}(E)$ and $\tilde{E}_S = \mathrm{MLP}(E_S)$.

To enforce sparsity and ensure that the assignment matrix approaches a one-hot structure, we minimize:

$$L_E = \frac{1}{N} \sum_{i=1}^{N} H(P_i),$$

where H represents the entropy function, and P_i denotes the i-th column of P.

Additionally, to retain the topological structure of the original graph, we introduce a regularization term:

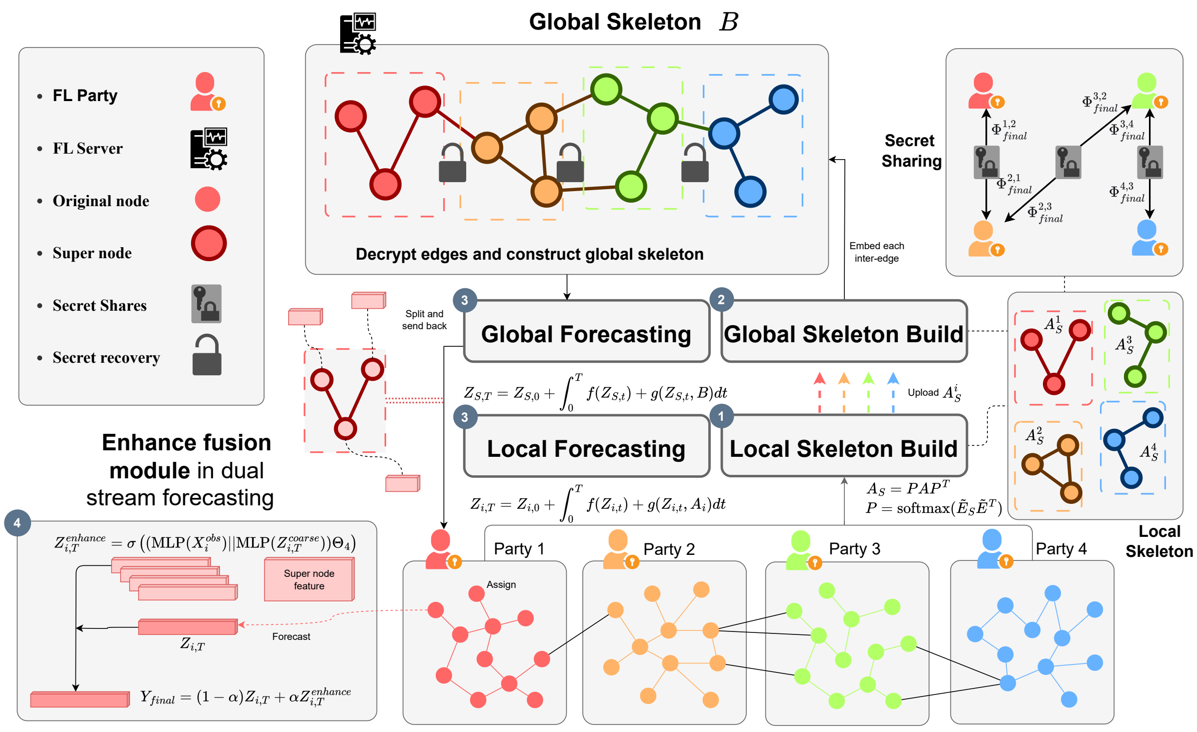
Global Skeleton B
FL Party
FL Server
Original node
Super node
Secret Shares
Secret recovery
Decrypt edges and construct global skeleton
Secret Sharing
Embed each inter-edge
Split and send back
Global Forecasting
Global Skeleton Build
Local Forecasting
Local Skeleton Build
$Z_{S,T} = Z_{S,0} + \int_0^T f(Z_{S,t}) + g(Z_{S,t}, B)dt$
$Z_{i,T} = Z_{i,0} + \int_0^T f(Z_{i,t}) + g(Z_{i,t}, A_i)dt$
Upload A_S^i
$A_S = PAP^T$
$P = \text{softmax}(\tilde{E}_S\tilde{E}^T)$
Enhance fusion module in dual stream forecasting
$Z_{i,T}^{enhance} = \sigma((\text{MLP}(X_i^{obs})||\text{MLP}(Z_{i,T}^{coarse}))\Theta_4)$
Super node feature
Forecast
$Z_{i,T}$
$Y_{final} = (1-\alpha)Z_{i,T} + \alpha Z_{i,T}^{enhance}$
Assign
Party 1
Party 2
Party 3
Party 4
Local Skeleton

Figure 1: The overall architecture of FedSkeleton.

$$L_R = ||A, P^T P||_F$$

where $||\cdot||_F$ denotes the Frobenius norm.

(ii) Node Aggregation. To model the temporal dynamics of original nodes, we employ graph convolutional networks (GCNs). The aggregated representation of the supernodes is computed as:

$$H = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}X)\Theta_1 \in \mathbb{R}^{N\times d}, X_S = PH \in \mathbb{R}^{\gamma\times N\times d}, \tag{5}$$

where $\tilde{D} = \sum_j \tilde{A}_{ij}$, $\tilde{A} = A + I$, and Θ is a learnable parameter.

The topological relationship between supernodes, also known as the local skeleton, is given by:

$$A_S = PAP^T,$$

Global skeleton build After every party has produced its local skeleton (A_S^i, P_i), the server must stitch these pieces together so that super-nodes belonging to different parties are linked whenever an inter-edge exists. We decompose this procedure into three reusable sub-modules, mirroring the style of the Local Skeleton Build. The process of this module is presented in Algorithm 2.

(i) Pairwise block formulation. For each unordered party pair (i, j) let $A_{ij} = E^*_{ij}$, $A_{ii} = A_S^i$, $A_{jj} = A_S^j$ and construct the block-diagonal assignment matrix $P = \text{diag}(P_i, P_j)$. The desired pairwise backbone is

$$A_s^{(i,j)} = PAP^\top = \begin{pmatrix} P_i A_{ii} P_i^\top & P_i A_{ij} P_j^\top \\ P_j A_{ji} P_i^\top & P_j A_{jj} P_j^\top \end{pmatrix}. \tag{8}$$

(ii) Secure off-diagonal computation. The diagonal blocks $C_{11} = P_i A_{ii} P_i^\top$ and $C_{22} = P_j A_{jj} P_j^\top$ are computed *locally*. The off-diagonal block $C_{12} = P_i A_{ij} P_j^\top$ must be obtained without revealing P_i, P_j or A_{ij}. We perform a Boolean-semiring matrix multiplication with

$$C_{12}[u, v] = \bigvee_k [(P_i)_{u,k} \wedge (A_{ij})_{k,v}], \tag{9}$$

and realise every AND via Beaver triples over $\{0, 1\}$ (Algorithm 3). Because $A_{ji} = A_{ij}^\top$, the second off-diagonal block is reused as $C_{21} = C_{12}^\top$.

(iii) Block embedding and global merge. Let $\mathcal{I}_i$ (resp. $\mathcal{I}_j$) be the index range of party i (resp. j) in the global super-node list. We embed $A_s^{(i,j)}$ into the empty backbone B by

$$B[\mathcal{I}_i \cup \mathcal{I}_j, \mathcal{I}_i \cup \mathcal{I}_j] \leftarrow B[\cdot] \vee A_s^{(i,j)}. \tag{10}$$

A Boolean OR ensures that repeated embeddings (from different pairs) only add connectivity, never delete it.

After all pairs are processed, $B \in \{0, 1\}^{N^* \times N^*}$ is a sparsified yet connectivity-complete *global skeleton*, it is ready to guide downstream Dual-Stream Forecasting.

3.3 Dual-Stream Forecasting

ODE prediction We now define the forward ODE function for the latent dynamics $\frac{dZ}{dt}$ of the skeleton. Taking into account that the evolution of each node x_i is influenced by its self-dynamics and coupling dynamics, the parameterized time derivative consists of two terms: one denoting the self-dynamics function $f(Z)$ and another representing the interaction with neighbors $g(Z, A)$. Thus, the dynamic equation for each node is given by:

$$\frac{dZ}{dt} = f(Z) + g(Z, A), \tag{6}$$

where f is an MLP and g is a GNN responsible for information propagation:

$$g(Z, A) = \sigma(A\sigma(Z\Theta_3)\Theta_2). \tag{7}$$

Given the ODE function, the predicted trajectory of the skeleton dynamics can be solved by any ODE solver as an initial value problem:

$$Z_T = Z_0 + \int_0^T f(Z_t) + g(Z_t, A)dt, \tag{8}$$

where the initial state $Z_0 = MLP(X) \in \mathbb{R}^{N\times 1}$. This allows us to predict the state of super-nodes at any continuous time point T.

Local and global forecasting Both local forecasting (executed independently by each participant) and global forecasting (conducted on the global skeleton at the central server) follow the same ODE-based modeling framework described above. However, they differ in their scope and information flow: **Local Forecasting**: Each participant applies the ODE model to its own subgraph, using a local adjacency matrix A^i and local node features X^i. This process captures the fine-grained temporal evolution of local structures.

$$Z_{i,T} = Z_{i,0} + \int_0^T f(Z_{i,t}) + g(Z_{i,t}, A_i)dt.$$

Global Forecasting: The central server applies the same ODE framework to the global skeleton$B$, where super-node's aggregated time-series data X_S^i structures from multiple participants.

$$Z_{S,T} = Z_{S,0} + \int_0^T f(Z_{S,t}) + g(Z_{S,t}, B)dt.$$

Enhance Fusion Module: Upon completion of the global forecast, the predicted global skeleton is partitioned, and the relevant segments are sent back to the participants. Each participant then enhances its local prediction by integrating enhanced results obtained from the coarse global information. The fusion is achieved through the following formulae:

$$Z_{i,T}^{enhance} = \sigma\left((\text{MLP}(X_i^{obs})||\text{MLP}(Z_{i,T}^{coarse}))\Theta_4\right),$$

$$Y_{\text{final}} = (1-\alpha)Z_{i,T} + \alpha Z_{i,T}^{enhance}.$$

By adjusting fusion rate α, participants can control the level of fusion applied to their final forecast.

3.4 Security Analysis

We show that the BoolMatMult is privacy-preserving and correct in the two-party, semi-honest, honest-majority setting. The detailed definition and proof is in appendix.

Resistance to attack **Data layer.** Raw time-series and their labels always remain on the client. Only the n-out-of-n XOR shares enter the MPC pipeline. Because any single share is information-theoretically independent of the underlying value and the server never accesses plaintext or gradients, an adversary cannot reconstruct samples or labels, nor perform membership inference or GAN-based data–synthesis attacks.

Topology layer. Each client compresses its local graph into k-anonymous super-nodes, then uses BoolMatMult to reveal only whether cross-party edges exist; the server further binarises the result. Coarse aggregation removes fine-grained structure, and binarisation erases edge-count and degree information, blocking graph-reconstruction and degree-based attribute-inference attacks. k-anonymous proof in appendix.

Mapping layer. The node-to-super-node assignment matrix P_i is never sent in plaintext; it appears only as masked shares inside the MPC and is dynamically updated during training. Consequently, neither a corrupted client nor an honest-but-curious server can map super-node observations back to concrete nodes, making node-level membership or attribute inference infeasible.

3.5 Communication Overhead

FedSkeleton's traffic naturally splits into a cold-start phase and an online training loop. During cold start each client pair (P_i, P_j) executes the Boolean-semiring protocol once, sending one masked matrix each $e \in \{0,1\}^{(\gamma|V_i|)\times|V_j|}$ from P_i and $f \in \{0,1\}^{(\gamma|V_j|)\times\gamma_j}$ from P_j-for an aggregate P2P cost of $\sum_i$ bits; every client then uploads its internal adjacency A_i^S and the cross-block shares to the server, which assembles and broadcasts the global binary backbone $B \in \{0,1\}^{\Gamma\times\Gamma}$ and $\Gamma = \sum_{i=1}^m \bigl(\gamma|V_i|\bigr)$ once. In each subsequent training round the node-to-supernode mapping P_i may change, so clients must re-run that same cross-block protocol and exchange fresh e, f matrices of identical size; meanwhile each client uploads $\gamma|V_i|d$ floats of super-node features X_i^S, receives the same-sized coarse forecast slice $Z_{i,T}^{\text{coarse}}$ and returns two scalar losses. All payloads scale linearly with the super-node count $\gamma|V_i|$, where $\gamma|V_i| \ll |V_i|$ and is far smaller than the full GNN parameter size $|\theta|$. Consequently, even with per-round client-to-client exchanges, the P2P load remains only a fraction of the gradients or node embeddings transferred in conventional graph FL, and the client–server round trip is just $2\gamma|V_i|d$ floats. Except for the one-off $O(\Gamma^2)$ server broadcast, cumulative training traffic grows only linearly with γ, yielding one- to two-order-of-magnitude savings over classical graph-federated learning while preserving structural privacy and accuracy. The theoretical communication-overhead comparison is provided in Appendix

4 Experiments

4.1 Datasets

We evaluate FedSkeleton on four time-series graph datasets-three real-world and one synthetic-covering power-grid operation, public-health surveillance and generic network dynamics. complete data sources, preprocessing and sliding-window settings are deferred to Appendix A(National Renewable Energy Laboratory 2006; Kim et al. 2018). Solar contains 137 nodes representing Alabama photovoltaic plants with 10-minute power-output readings and is split into 2 parties; ChileNet models a Chilean generator–substation grid with 218 nodes and 3 default parties; Syn is a 2,000-node Barabási–Albert synthetic network governed by Hindmarsh–Rose dynamics, evaluated under 5-, 20- and 50-party partitions; Covid links 3,142 U.S. counties through adjacency edges, provides 500-day case counts, and treats each state as a party for a total of 45.

4.2 Compared Algorithm

We compare FedSkeleton with three state-of-the-art algorithms: (1) **STGNCDE** (Choi et al. 2022) integrates graph convolutions and controlled differential equations for the prediction of time series in graph-structured data. (2) **MTGODE** (Jin et al. 2023) combines spatial dependencies through graph convolutions and temporal dynamics through ODEs for multi-step forecasting. (3) **FourierGNN** (Yi et al. 2024) combines Fourier Transform and GNN to capture spatio-temporal dependencies in the frequency domain. (4) **T-PATCHGNN** (Zhang et al. 2024) splits irregular multivariate series into transformable patches and forecasts with a Transformer–GNN hybrid. (5) **NDCN** (Zang and Wang 2020) views GNN layers as continuous diffusion ODEs, integrating node dynamics for sequence prediction. (6) **FedGC** (Fu et al. 2025) condenses graphs federatively by gradient-matching parties while safeguarding membership privacy. (7) **CNFGNN** (Meng, Rambhatla, and Liu 2021a) decouples on-device temporal encoding and server-side spatial GNN to model decentralized spatio-temporal data.

4.3 Experimental Setup

We evaluate FedSkeleton under two complementary learning regimes. In the centralized setting, a single server holds the complete graph and all node features. In the federated setting, both topology and features remain partitioned across clients and never leave their silos. The full implementation details provided in the Appendix.

4.4 Metrics

To evaluate the model's performance, we use four key metrics: Mean Squared Error (MSE), Normalized Mean Squared Error (NMSE), Mean Absolute Error (MAE), and Root Mean Squared Error (RMSE). Specifically, for all metrics, a smaller value indicates a better performance. Their detailed description can be found in Appendix.

4.5 Main Results

Centralized performance Table 1 shows that FedSkeleton consistently outperforms every baseline on both Solar and ChileNet. On Solar, FedSkeleton attains the lowest MSE (18.13) and RMSE (4.20), cutting error by roughly 11% and 7% relative to the second-best FourierGNN (20.37 / 4.51) and by 40–72% against MTGODE, STGNCDE and the newly added NDCN. Although FourierGNN edges ahead in NMSE and MAE, FedSkeleton still delivers the best overall accuracy. The advantage widens on ChileNet: FedSkeleton leads all four metrics, with an NMSE of 0.0360-23% lower than FourierGNN, 55% lower than MTGODE, 67% lower than NDCN, and 85% lower than STGNCDE. Its MSE drops to 429.63, improving on FourierGNN by 14% and on MTGODE by 49%, while MAE (5.84) and RMSE (19.43) are likewise the best recorded. Crucially, every baseline is trained on the full graph topology, whereas FedSkeleton relies only on party-local subgraphs plus a privacy-preserving global skeleton. Despite lacking complete topology, the framework leverages secure cross-edge information to surpass centrally trained models, underscoring FedSkeleton's strength for time-series forecasting under federated, privacy-constrained settings.

Federated performance In TableTable 2 showsUnder a fully federated setting, FedSkeleton delivers either the best or second-best scores on all four datasets and clearly outperforms the other three baselines overall. On Solar, it reduces MSE to 18.13 and RMSE to 4.20, improvements of 28.0% and 12.7% over the next-best T-PatchGNN. On ChileNet, its NMSE is only 0.036, corresponding to 90.0% of T-PatchGNN and 34.6% of CNFGNN, while FedGC-limited by structural information loss-shows errors more than an order of magnitude higher. On the synthetic Syn dataset, FedSkeleton achieves NMSE 5×10^{-4} and MSE 3×10^{-3}, a further 4–5$\times$ reduction relative to CNFGNN, whereas FedGC lags by two orders of magnitude. Even on the highly imbalanced Covid dataset, FedSkeleton remains ahead with an MSE of 4.50×10^{6}, just 0.14% of T-PatchGNN and 0.38% of CNFGNN, demonstrating robustness to heterogeneous graph sizes. In summary, by leveraging a privacy-preserving global skeleton to recover cross-party topology, FedSkeleton consistently lowers forecasting error and maintains superiority across all evaluation metrics.

4.6 Parameter Analysis

Impact of compression ratio In FedSkeleton, the compression ratio γ controls the resolution of the global skeleton's structure. Higher values of γ preserve more node-level details, while lower values have coarser structure. This section investigates the relationship between γ and predictive accuracy. To evaluate the effect of γ on performance, we conduct a parameter study using seven different values: [0.05, 0.08, 0.1, 0.2, 0.3, 0.4, 0.5]. For clarity, we present the performance trends of the model with varying γ values at specific epochs: [10, 30, 40, 50], where all participants in a given training session use the same γ. Figure 2 presents the results for the Solar dataset. As curves in (a) and (b), When the number of training epochs is small, the impact of γ on performance is not significant. As training progresses, higher values of γ lead to better predictive accuracy, while γ values below 0.1 result in a noticeable decline in performance. However, when γ exceeds 0.2, there is no significant improvement in prediction performance.

Impact of fusion rate In this section, we explore the impact of different fusion rates on the FedSkeleton framework, conducting experiments on two datasets, Solar and Chile. Each FL party uses the same fusion rate by default, and we examine the effect of varying α from 0 to 1 with a step size of 0.1.

Figure 3 upper row illustrates the impact of α on the Solar dataset, with prediction loss trends measured using MAE and MSE at different epochs (10, 50, and 80). As observed, when α is set to a low value (closer to 0), the prediction relies primarily on local information, resulting in higher loss values, particularly in long-term predictions. As α increases, incorporating more global information, the prediction error

Table 1: Average performance comparison on Solar and ChileNet (smaller is better). **Bold** = best, underline = second best.

	Solar				ChileNet			
	NMSE	MSE	MAE	RMSE	NMSE	MSE	MAE	RMSE
NDCN	0.6328	65.3593	4.3139	8.0845	0.1085	1191.1644	13.2282	34.5132
MTGODE	0.1920	30.0970	3.0341	5.4860	0.0800	840.8540	9.4250	28.9970
STGNCDE	0.2750	42.8610	4.4991	6.5426	0.2420	2549.2980	18.2940	50.4770
FourierGNN	**0.1370**	20.3730	**2.2860**	4.5140	0.0470	501.8770	7.0110	22.4020
Our model	0.1970	**18.1340**	2.3520	**4.1980**	**0.0360**	**429.6340**	**5.8420**	**19.4330**

Table 2: Average performance comparison on four real-world datasets (smaller is better). **Bold**=best, underline=second best.

	Solar				ChileNet			
	NMSE	MSE	MAE	RMSE	NMSE	MSE	MAE	RMSE
t-PatchGNN	**0.1493**	25.1150	**2.3190**	4.8112	0.0400	452.1530	7.3579	20.1395
FedGC	0.7735	117.0280	9.4185	10.736	0.4287	4145.1114	40.5153	63.933
CNFGNN	0.7053	104.6111	5.3617	9.977	0.1041	1174.9940	15.6223	33.131
Our model	0.1970	**18.1340**	2.3520	**4.1980**	**0.0360**	**429.6340**	**5.8420**	**19.433**
	Syn				**Covid**			
	NMSE	MSE	MAE	RMSE	NMSE	MSE	MAE	RMSE
t-PatchGNN	0.0047	0.0301	0.1187	0.1730	0.0213	3.158e+9	2.212e+3	3.167e+3
FedGC	0.3967	2.5369	1.1290	1.5858	0.8142	2.082e+9	1.551e+4	3.448e+4
CNFGNN	0.0022	0.0139	0.0911	0.1178	0.3069	1.182e+9	7.881e+3	1.712e+4
Our model	**0.0005**	**0.0030**	**0.0348**	**0.0550**	**0.0040**	**4.501e+6**	**7.066e+2**	**2.122e+3**

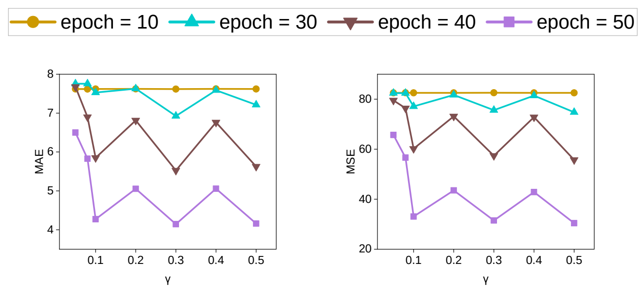

Figure 2: Prediction loss trends measured by MAE and MSE over different values of γ on the Solar dataset, with results recorded at epochs 10, 30, 40, and 50.

gradually decreases. However, when α approaches 1, the MSE and MAE both start increasing again, especially at later epochs. This suggests that excessive reliance on global forecasts may introduce noise or excessive coarsening, leading to performance degradation. The best performance is achieved for moderate values of α (around 0.3 to 0.6), where local fine-grained details and global structural information are optimally fused.

Lower row presents the results on the Chile dataset, (b) omits the MSE when $\alpha = 0$. showing a similar trend in the influence of α. When α is too low, the MAE and MSE values remain relatively high, as local-only forecasting lacks sufficient global structural information to generalize well. However, unlike the Solar dataset, the Chile dataset exhibits more variation in performance as α increases, with noticeable fluctuations at higher values of α. This indicates that the balance between local and global forecasting plays a more dynamic role depending on the dataset characteristics. Overall, for both datasets, moderate values of α yield the most stable and accurate predictions, confirming the necessity of balancing local and global fusion in the FedSkeleton framework.

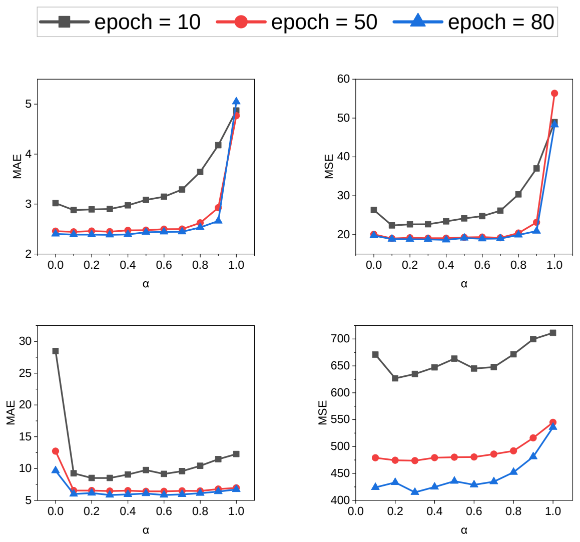

Figure 3: Prediction loss trends measured by MAE and MSE over different values of α on the Solar (top row) and Chile (bottom row) datasets, with results recorded at epochs 10, 50, and 80.

Global topology updating To quantify how the global skeleton update frequency affects accuracy, we keep all other hyper-parameters fixed on the Syn-5 partition and vary the trigger ratio of Global Skeleton Build. Result present in Table 3. FedSkeleton-all rebuilds the backbone in every communication round, whereas FedSkeleton-$\{0.8, 0.5, 0.4, 0.2\}$ refresh it in only 80%, 50%, 40%, and 20% of the rounds, reusing the previous adjacency otherwise. Full updates yield the best scores (NMSE $= 0.0005$, MSE $= 0.0030$). Reducing the ratio to 0.8 causes a modest rise (roughly $1.8\times$ in MSE); at 0.5 and 0.4 the increments grow further and both MAE and RMSE deteriorate markedly, indicating that stale cross-party topology induces drift that local models cannot correct. The extreme 0.2 setting, which effectively approximates local training in most rounds, exhibits larger and unpredictable fluctuations. Overall, promptly synchronising the global structure substantially boosts forecasting accuracy, while an update ratio of at least 80% offers a pragmatic trade-off between communication cost and performance in bandwidth-constrained scenarios.

Table 3: Impact of Global Skeleton Update Ratio on Forecasting Errors (Syn-5)

	Syn-5			
	NMSE	MSE	MAE	RMSE
FedSkleton-all	0.0005	0.0030	0.0348	0.0550
FedSkleton-0.8	0.0009	0.0057	0.0384	0.0758
FedSkleton-0.5	0.0011	0.0070	0.0433	0.0835
FedSkleton-0.4	0.0011	0.0070	0.0433	0.0835
FedSkleton-0.2	0.0011	0.0030	0.0348	0.0550

Impact of federation scale To assess robustness under federation growth, we partition the synthetic dataset (Syn) into three federation sizes-5, 20, and 50 clients-keeping the per-client data volume and all training hyper-parameters identical to the main experiments. As the number of clients increases, the total amount of data rises while communication and aggregation costs also grow. We report four error metrics and plot the two most indicative ones-mean squared error (MSE) and mean absolute error (MAE)-to visualize scaling trends.

Figure 4 reports performance when the synthetic Syn dataset is split across 5, 20 and 50 clients. This shows that FedSkeleton is the most scale-robust: its MAE increases only from 0.0348 to 0.0572 and its MSE from 0.0030 to 0.0090-both less than a two-fold rise-while remaining the lowest throughout, CNFGNN improves slightly as federation size grows (MAE $0.0911 \rightarrow 0.0811$; MSE $0.0139 \rightarrow 0.0135$) yet still lags FedSkeleton by roughly 40–140% in every setting. T-PatchGNN is more scale-sensitive, with MAE dropping from 0.1187 to 0.0611 and MSE from 0.0301 to 0.0090, but it stays an order of magnitude worse than FedSkeleton. FedGC's MSE hovers around 6.2 at both 20 and 50 clients-nearly three orders of magnitude higher than FedSkeleton-indicating that its compression strategy struggles to preserve structural information under highly heterogeneous splits. Overall, FedSkeleton sustains the best error levels even as communication and aggregation costs rise, demonstrating superior scalability and robustness.

4.7 Additional Experiment

To quantify how graph compression affects privacy, we randomly choose target nodes on the Solar dataset and let the server act as a latent attacker that is given partial raw data. The attacker trains a model to infer the targets' time–series while FedSkeleton operates with seven compression ratios $\gamma \in \{0.05, 0.08, 0.1, 0.2, 0.3, 0.4, 0.5\}$. Figure 5 plots the attacker's MAE and MSE throughout training. The curves

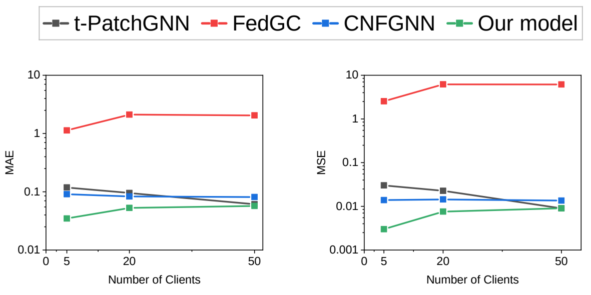

Figure 4: Prediction loss trends measured by MAE (left) and MSE (right) as the synthetic Syn dataset is partitioned across 5, 20, and 50 clients.

show that when $\gamma < 0.08$, both errors remain highest, indicating the strongest privacy because many primitive nodes are merged into the same super-node. For $\gamma \geq 0.2$ the errors drop noticeably, meaning the attacker can more easily recover single-node information. In short, very small compression ratios offer clear privacy gains, while improvements saturate once γ exceeds 0.1. Full experimental details and the exact attack-model definition are provided in the appendix.

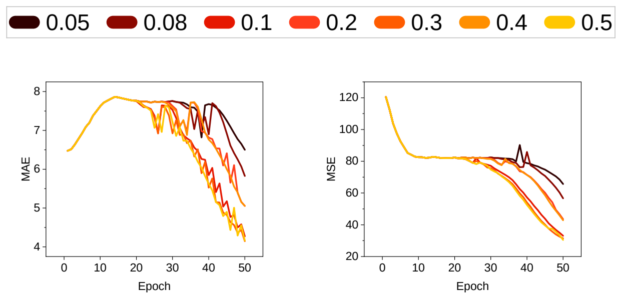

Figure 5: Attack loss trends measured by MAE and MSE over training epochs on the Solar dataset for different values of γ.

5 Conclusion

In this paper, we propose a privacy-preserving federated graph learning framework, named FedSkeleton, for collaborative time-series prediction tasks. FedSkeleton can strike a better balance between forecasting utility and data privacy with the enhancement of the two curated modules. Specifically, we first curate a skeleton construction method to aggregate cross-party features in a privacy-preserving manner. Subsequently, we introduce a dual-stream forecast mechanism for mitigating the performance decline raised by the privacy-preserving scheme. Extensive experiments on two real-world datasets demonstrate that FedSkeleton significantly outperforms state-of-the-art techniques in federated graph learning for time-series forecasting. Future research can explore integrating different local predictive models with the global skeleton to further investigate its capability in capturing and understanding complex network topologies. Additionally, further studies can focus on the security and robustness of supernodes, exploring adversarial strategies in federated learning.

References

Bai, L.; Yao, L.; Li, C.; Wang, X.; and Wang, C. 2020. Adaptive graph convolutional recurrent network for traffic forecasting. *Advances in neural information processing systems*, 33: 17804–17815.

Beaver, D. 1991. Efficient multiparty protocols using circuit randomization. In *Annual international cryptology conference*, 420–432. Springer.

Blakley, G. R. 1979. Safeguarding cryptographic keys. In *Managing requirements knowledge, international workshop on*, 313–313. IEEE Computer Society.

Cao, D.; Wang, Y.; Duan, J.; Zhang, C.; Zhu, X.; Huang, C.; Tong, Y.; Xu, B.; Bai, J.; Tong, J.; et al. 2020. Spectral temporal graph neural network for multivariate time-series forecasting. *Advances in neural information processing systems*, 33: 17766–17778.

Chen, F.; Li, P.; Miyazaki, T.; and Wu, C. 2021. Fedgraph: Federated graph learning with intelligent sampling. *IEEE Transactions on Parallel and Distributed Systems*, 33(8): 1775–1786.

Chen, F.; Long, G.; Wu, Z.; Zhou, T.; and Jiang, J. 2022. Personalized federated learning with graph. *arXiv preprint arXiv:2203.00829*.

Chen, H.-Y.; and Chao, W.-L. ???? On Bridging Generic and Personalized Federated Learning for Image Classification. In *International Conference on Learning Representations*.

Chen, Y.; Segovia, I.; and Gel, Y. R. 2021. Z-GCNETs: Time zigzags at graph convolutional networks for time series forecasting. In *International Conference on Machine Learning*, 1684–1694. PMLR.

Choi, J.; Choi, H.; Hwang, J.; and Park, N. 2022. Graph neural controlled differential equations for traffic forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, 6367–6374.

Fang, Z.; Long, Q.; Song, G.; and Xie, K. 2021. Spatial-temporal graph ode networks for traffic flow forecasting. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, 364–373.

Fu, W.; Wang, H.; Gao, C.; Liu, G.; Li, Y.; and Jiang, T. 2024. Privacy-Preserving Individual-Level COVID-19 Infection Prediction via Federated Graph Learning. *ACM Transactions on Information Systems*, 42(3): 1–29.

Fu, X.; Gao, Y.; Yang, B.; Wu, Y.; Qian, H.; Sun, Q.; and Li, X. 2025. Bi-directional multi-scale graph dataset condensation via information bottleneck. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 16674–16681.

Fu, X.; Zhang, B.; Dong, Y.; Chen, C.; and Li, J. 2022. Federated Graph Machine Learning: A Survey of Concepts, Techniques, and Applications. arXiv:2207.11812.

Guo, S.; Lin, Y.; Feng, N.; Song, C.; and Wan, H. 2019. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, 922–929.

Hashemi, M.; Gong, S.; Ni, J.; Fan, W.; Prakash, B. A.; and Jin, W. 2024. A comprehensive survey on graph reduction: Sparsification, coarsening, and condensation. *arXiv preprint arXiv:2402.03358*.

He, C.; Ceyani, E.; Balasubramanian, K.; Annavaram, M.; and Avestimehr, S. 2022. SpreadGNN: Decentralized Multi-Task Federated Learning for Graph Neural Networks on Molecular Data. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(6): 6865–6873.

Huang, W.; Liu, J.; Li, T.; Ji, S.; Wang, D.; and Huang, T. 2022. FedCKE: Cross-Domain Knowledge Graph Embedding in Federated Learning. *IEEE Transactions on Big Data*, 1–12.

Jin, M.; Zheng, Y.; Li, Y.-F.; Chen, S.; Yang, B.; and Pan, S. 2023. Multivariate Time Series Forecasting With Dynamic Graph Neural ODEs. *IEEE Transactions on Knowledge and Data Engineering*, 35(9): 9168–9180.

Kim, H.; Olave-Rojas, D.; Álvarez-Miranda, E.; and Son, S.-W. 2018. In-depth data on the network structure and hourly activity of the central Chilean power grid. *Scientific data*, 5(1): 1–10.

Lei, R.; Wang, P.; Zhao, J.; Lan, L.; Tao, J.; Deng, C.; Feng, J.; Wang, X.; and Guan, X. 2023. Federated learning over coupled graphs. *IEEE Transactions on Parallel and Distributed Systems*, 34(4): 1159–1172.

Li, Y.; Yu, R.; Shahabi, C.; and Liu, Y. 2017. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv preprint arXiv:1707.01926*.

Liu, T.; Li, P.; and Gu, Y. 2021. Glint: Decentralized federated graph learning with traffic throttling and flow scheduling. In *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*, 1–10. IEEE.

Liu, Y.; Garg, S.; Nie, J.; Zhang, Y.; Xiong, Z.; Kang, J.; and Hossain, M. S. 2020. Deep anomaly detection for time-series data in industrial IoT: A communication-efficient on-device federated learning approach. *IEEE Internet of Things Journal*, 8(8): 6348–6358.

Liu, Y.; Liu, Q.; Zhang, J.-W.; Feng, H.; Wang, Z.; Zhou, Z.; and Chen, W. 2022. Multivariate time-series forecasting with temporal polynomial graph neural networks. *Advances in neural information processing systems*, 35: 19414–19426.

Lu, S.; Zhang, Y.; Wang, Y.; and Mack, C. 2019. Learn electronic health records by fully decentralized federated learning. *arXiv preprint arXiv:1912.01792*.

McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; and y Arcas, B. A. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, 1273–1282. PMLR.

Mei, G.; Guo, Z.; Liu, S.; and Pan, L. 2019. SGNN: A Graph Neural Network Based Federated Learning Approach by Hiding Structure. In *2019 IEEE International Conference on Big Data (Big Data)*, 2560–2568.

Meng, C.; Rambhatla, S.; and Liu, Y. 2021a. Cross-node federated graph neural network for spatio-temporal data modeling. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, 1202–1211.

Meng, C.; Rambhatla, S.; and Liu, Y. 2021b. Cross-Node Federated Graph Neural Network for Spatio-Temporal Data Modeling. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, KDD '21, 1202–1211. New York, NY, USA: Association for Computing Machinery. ISBN 978-1-4503-8332-5.

Miele, E. S.; Bonacina, F.; and Corsini, A. 2022. Deep anomaly detection in horizontal axis wind turbines using graph convolutional autoencoders for multivariate time series. *Energy and AI*, 8: 100145.

National Renewable Energy Laboratory. 2006. Solar Power Data. Accessed: 2025-02-05.

Pan, Z.; Liang, Y.; Wang, W.; Yu, Y.; Zheng, Y.; and Zhang, J. 2019. Urban traffic prediction from spatio-temporal data using deep meta learning. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 1720–1730.

Shamir, A. 1979. How to share a secret. *Communications of the ACM*, 22(11): 612–613.

Shang, C.; Chen, J.; and Bi, J. 2021. Discrete graph structure learning for forecasting multiple time series. *arXiv preprint arXiv:2101.06861*.

Shen, Y.; Zhang, J.; Song, S.; and Letaief, K. B. 2022. Graph neural networks for wireless communications: From theory to practice. *IEEE Transactions on Wireless Communications*, 22(5): 3554–3569.

Sweeney, L. 2002. k-anonymity: A model for protecting privacy. *International journal of uncertainty, fuzziness and knowledge-based systems*, 10(05): 557–570.

Truong, H. T.; Ta, B. P.; Le, Q. A.; Nguyen, D. M.; Le, C. T.; Nguyen, H. X.; Do, H. T.; Nguyen, H. T.; and Tran, K. P. 2022. Light-weight federated learning-based anomaly detection for time-series data in industrial control systems. *Computers in Industry*, 140: 103692.

Wang, X.; Ma, Y.; Wang, Y.; Jin, W.; Wang, X.; Tang, J.; Jia, C.; and Yu, J. 2020. Traffic flow prediction via spatial temporal graph neural network. In *Proceedings of the web conference 2020*, 1082–1092.

Wu, C.; Wu, F.; Lyu, L.; Qi, T.; Huang, Y.; and Xie, X. 2022. A Federated Graph Neural Network Framework for Privacy-Preserving Personalization. *Nature Communications*, 13(1): 3091.

Wu, Z.; Pan, S.; Long, G.; Jiang, J.; Chang, X.; and Zhang, C. 2020. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, 753–763.

Yan, B. 2024. Federated Graph Condensation with Information Bottleneck Principles. *arXiv preprint arXiv:2405.03911*.

Yang, L.; Tan, B.; Zheng, V. W.; Chen, K.; and Yang, Q. 2020. Federated recommendation systems. *Federated Learning: Privacy and Incentive*, 225–239.

Yao, Y.; Jin, W.; Ravi, S.; and Joe-Wong, C. 2024. FedGCN: Convergence-communication tradeoffs in federated training of graph convolutional networks. *Advances in neural information processing systems*, 36.

Yi, J.; Wu, F.; Wu, C.; Liu, R.; Sun, G.; and Xie, X. 2021. Efficient-FedRec: Efficient Federated Learning Framework for Privacy-Preserving News Recommendation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2814–2824.

Yi, K.; Zhang, Q.; Fan, W.; He, H.; Hu, L.; Wang, P.; An, N.; Cao, L.; and Niu, Z. 2024. FourierGNN: Rethinking multivariate time series forecasting from a pure graph perspective. *Advances in Neural Information Processing Systems*, 36.

Yu, B.; Yin, H.; and Zhu, Z. 2017. Spatio-temporal Graph Convolutional Neural Network: A Deep Learning Framework for Traffic Forecasting. *CoRR*, abs/1709.04875.

Zang, C.; and Wang, F. 2020. Neural dynamics on complex networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, 892–902.

Zhang, W.; Yin, C.; Liu, H.; Zhou, X.; and Xiong, H. 2024. Irregular multivariate time series forecasting: A transformable patching graph neural networks approach. In *Forty-first International Conference on Machine Learning*.

Reproducibility Checklist

Instructions for Authors:

This document outlines key aspects for assessing reproducibility. Please provide your input by editing this `.tex` file directly.

For each question (that applies), replace the “Type your response here” text with your answer.

Example: If a question appears as

```
\question{Proofs of all novel claims
are included} {(yes/partial/no)}
Type your response here
```

you would change it to:

```
\question{Proofs of all novel claims
are included} {(yes/partial/no)}
yes
```

Please make sure to:

- Replace ONLY the “Type your response here” text and nothing else.
- Use one of the options listed for that question (e.g., **yes**, **no**, **partial**, or **NA**).
- **Not** modify any other part of the `\question` command or any other lines in this document.

You can `\input` this .tex file right before `\end{document}` of your main file or compile it as a stand-alone document. Check the instructions on your conference’s website to see if you will be asked to provide this checklist with your paper or separately.

1. General Paper Structure

1.1. Includes a conceptual outline and/or pseudocode description of AI methods introduced (yes/partial/no/NA) yes

1.2. Clearly delineates statements that are opinions, hypothesis, and speculation from objective facts and results (yes/no) yes

1.3. Provides well-marked pedagogical references for less-familiar readers to gain background necessary to replicate the paper (yes/no) yes

2. Theoretical Contributions

2.1. Does this paper make theoretical contributions? (yes/no) no

If yes, please address the following points:

2.2. All assumptions and restrictions are stated clearly and formally (yes/partial/no) yes

2.3. All novel claims are stated formally (e.g., in theorem statements) (yes/partial/no) yes

2.4. Proofs of all novel claims are included (yes/partial/no) yes

2.5. Proof sketches or intuitions are given for complex and/or novel results (yes/partial/no) yes

2.6. Appropriate citations to theoretical tools used are given (yes/partial/no) yes

2.7. All theoretical claims are demonstrated empirically to hold (yes/partial/no/NA) yes

2.8. All experimental code used to eliminate or disprove claims is included (yes/no/NA) yes

3. Dataset Usage

3.1. Does this paper rely on one or more datasets? (yes/no) yes

If yes, please address the following points:

3.2. A motivation is given for why the experiments are conducted on the selected datasets (yes/partial/no/NA) yes

3.3. All novel datasets introduced in this paper are included in a data appendix (yes/partial/no/NA) yes

3.4. All novel datasets introduced in this paper will be made publicly available upon publication of the paper with a license that allows free usage for research purposes (yes/partial/no/NA) yes

3.5. All datasets drawn from the existing literature (potentially including authors' own previously published work) are accompanied by appropriate citations (yes/no/NA) yes

3.6. All datasets drawn from the existing literature (potentially including authors' own previously published work) are publicly available (yes/partial/no/NA) yes

3.7. All datasets that are not publicly available are described in detail, with explanation why publicly available alternatives are not scientifically satisficing (yes/partial/no/NA) NA

4. Computational Experiments

4.1. Does this paper include computational experiments? (yes/no) yes

If yes, please address the following points:

4.2. This paper states the number and range of values tried per (hyper-) parameter during development of the paper, along with the criterion used for selecting the final parameter setting (yes/partial/no/NA) yes

4.3. Any code required for pre-processing data is included in the appendix (yes/partial/no) yes

4.4. All source code required for conducting and analyzing the experiments is included in a code appendix (yes/partial/no) yes

4.5. All source code required for conducting and analyzing the experiments will be made publicly available upon publication of the paper with a license that allows free usage for research purposes (yes/-partial/no) yes

4.6. All source code implementing new methods have comments detailing the implementation, with references to the paper where each step comes from (yes/-partial/no) yes

4.7. If an algorithm depends on randomness, then the method used for setting seeds is described in a way sufficient to allow replication of results (yes/partial/no/NA) yes

4.8. This paper specifies the computing infrastructure used for running experiments (hardware and software), including GPU/CPU models; amount of memory; operating system; names and versions of relevant software libraries and frameworks (yes/partial/no) yes

4.9. This paper formally describes evaluation metrics used and explains the motivation for choosing these metrics (yes/partial/no) yes

4.10. This paper states the number of algorithm runs used to compute each reported result (yes/no) yes

4.11. Analysis of experiments goes beyond single-dimensional summaries of performance (e.g., average; median) to include measures of variation, confidence, or other distributional information (yes/no) no, due to the high computational cost of the experiments, all experimental results were obtained using a fixed random seed.

4.12. The significance of any improvement or decrease in performance is judged using appropriate statistical tests (e.g., Wilcoxon signed-rank) (yes/partial/no) no, the performance is significant enough

4.13. This paper lists all final (hyper-)parameters used for each model/algorithm in the paper's experiments (yes/partial/no/NA) yes