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

Henggang Deng[1*], **Yuchao Tang**[1], **Wenjie Fu**[2], **Huandong Wang**[1†], **Kun Chen**[3†], **Tao Jiang**[2]
[1]Tsinghua University
[2]Huazhong University of Science and Technology
[3]Ant Group
denghenggang2003@163.com, wjfu99@outlook.com, wanghuandong@tsinghua.edu.cn, ck413941@antgroup.com

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) the graph topology itself contains sensitive structural information, which can lead to privacy leakage if directly shared during FGL; (2) cross-party edges are crucial for accurate modeling, yet exploiting them without compromising privacy remains a significant challenge. 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 models trained in a centralized setting with full-graph access in certain cases. 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.

Code — https://github.com/tsinghua-fib-lab/FedSkeleton

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.

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 the graph data of each participant. The core of FedSkeleton is an elaborately-designed secure multi-party graph skeleton construction algorithm that utilizes graph

[*]Work done while the author was an intern at Tsinghua University
[†]Corresponding Author (wanghuandong@tsinghua.edu.cn, ck413941@antgroup.com)

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 (ordinary differential equation) model on their private graphs, while a global Graph ODE model is trained on the constructed graph skeleton at the server. 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 mechanism 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 experiments on four datasets demonstrate that FedSkeleton achieves an average improvement of 42.48% compared with baselines. The result confirms FedSkeleton's capability to 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.

Preliminaries

Secret Sharing

We adopt the 2-out-of-2 secret sharing scheme from ABY2 (Patra et al. 2021) in the two-party (2PC) setting. It allow two parties ($\mathcal{P}_0$ and $\mathcal{P}_1$) to use secret shares to jointly compute a function on their private inputs. Based on the underlying algebraic structure of the computation, there are two main sharing schemes. The first is Arithmetic Secret Sharing, where computation is performed in a ring $\mathbb{Z}_{2^l}$. For each wire in the circuit with a value v, party $\mathcal{P}_i$ (where $i \in \{0,1\}$) holds an additive share, denoted $[v]_i$, such that $v = [v]_0 + [v]_1 \pmod{2^l}$. All linear gates (addition or subtraction) can be evaluated locally and non-interactively. To securely evaluate a multiplication gate, the protocol uses Beaver's circuit randomization technique (Beaver 1991), which requires generating additively shared random arithmetic triples during a setup phase. These triples are then used in the online phase to compute the shares of the product, which requires communication for each multiplication gate. The second scheme is Boolean Secret Sharing, namely the GMW protocol (Goldreich, Micali, and Wigderson 2019), which employs a function represented as a Boolean circuit (i.e., $l = 1$, operating in $\mathbb{Z}_2$). In this scheme, values are shared using XOR-based secret sharing, satisfying $v = [v]_0 \oplus [v]_1$. Similarly, linear gates (XOR gates) are non-interactive, while multiplication gates (AND gates) require pre-computing a Boolean multiplication triple. This is typically achieved using 1-out-of-2 Oblivious Transfer (OT), and can be optimized using 1-out-of-N OT extension (Ishai et al. 2003).

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). The data held by company i, specifically the feature matrix $\mathbf{X}_i$ and the local edges E_i, is retained locally and not shared with other companies or the server. In contrast, the links between plants of different companies $E_{ij} = E \cap (V_i \times V_j)$, which is also referred as *inter-edges*, are public to both company i and company j. Similar cross-organisation settings arise in banking transaction networks and mobile-operator base-station networks.

If these inter-edges are ignored during the training procedure of the federated graph neural network (GNN) model, 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.

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 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 satisfy $V_i \cap V_j = \emptyset$, for each $i \neq j$. 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 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 M FL parties.* ***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.*

Method

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 of the Appendix.

Graph Skeleton Construction

Local Skeleton Construction 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 $E_S \in \mathbb{R}^{\gamma N \times 2}$ 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:

$$P = \text{softmax}(\tilde{E}_S \tilde{E}^T) \in \{0,1\}^{\gamma N \times N}, \tag{4}$$

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

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:

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

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:

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

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 Merge After every party has produced its local skeleton (A_i^S, 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 of the Appendix.

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

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

(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

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

and realise every AND via Beaver triples over $\{0, 1\}$ (Algorithm 3 Secure Boolean MatMul in Appendix). 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 supernode list. We embed $A_{(i,j)}^s$ into the empty backbone B by

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

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.

Dual-Stream Forecasting

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

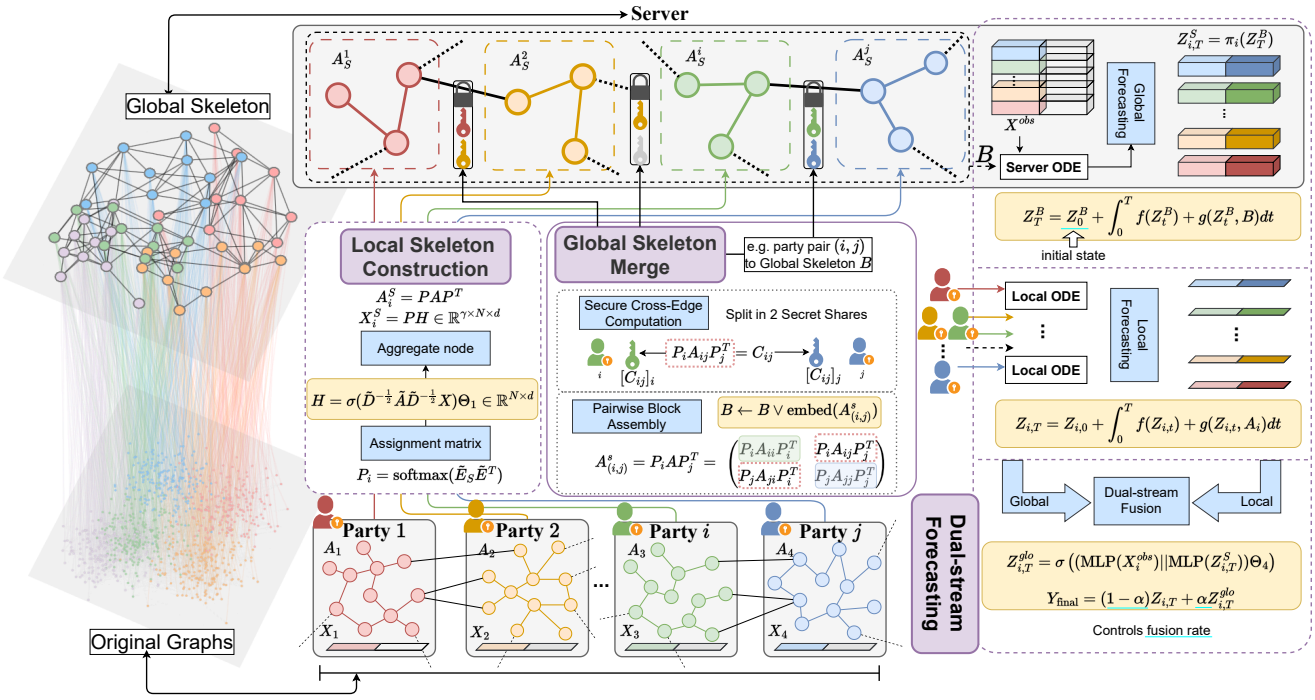

Figure 1: The overall architecture of FedSkeleton.

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.

(i) 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.

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

(ii) Globa Forecasting. The central server applies the same ODE framework to the global skeleton B, where super-node's aggregated time-series data X_i^S structures from multiple participants.

$$Z_T^B = Z_0^B + \int_0^T f(Z_t^B) + g(Z_t^B, B)dt.$$

The server's global forecast Z_T^B is a composite tensor containing the coarse-grained information of all participants. To distribute this back to the clients, we define a projection operator π_i. This operator extracts the segment relevant only to that client from Z_T^B based on the participant index i. This client-specific segment, $Z_{i,T}^S$, is defined as:

$$Z_{i,T}^S = \pi_i(Z_T^B).$$

$Z_{i,T}^S$ is then sent to participant i for subsequent fusion.

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:

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

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

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

Security Analysis

We analyze the security of FedSkeleton from three layers.

(i) Data layer. Raw time-series and their labels always remain on the party. Only the 2-out-of-2 boolean shares (Patra et al. 2021) enter the Secure Multi-Party Computation (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.

(ii) Topology layer. Each party compresses its local graph into k-anonymous (Sweeney 2002) super-nodes, then uses Secure Boolean MatMul (algorithm 3) to comupute cross-party edges and 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.

(iii) 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 party nor an honest-but-curious server can map super-node observations back to concrete nodes, making node-level membership or attribute inference infeasible.

the (algorithm 3), and the overall framework that depends on it, are privacy-preserving and correct in the two-party, semi-honest setting. The detailed definition and proof are provided in the Appendix.

Communication Overhead

FedSkeleton's traffic naturally splits into a cold-start phase and an online training loop. During cold start each party pair $(\mathcal{P}_i, \mathcal{P}_j)$ executes the Boolean-semiring protocol once, sending one masked matrix each $e \in \{0,1\}^{\gamma|V_i|\times|V_j|}$ from $\mathcal{P}_i$ and $f \in \{0,1\}^{\gamma|V_j|\times\gamma_j}$ from $\mathcal{P}_j$; every party then uploads its internal adjacency A_i^S and the cross-block shares to the server, which assembles and broadcasts the global binary backbone $B \in \{0,1\}^{\gamma N\times\gamma N}$. In each subsequent training round the node-to-supernode mapping P_i may change, so parties must re-run that same cross-block protocol and exchange fresh e, f matrices of identical size; meanwhile each party uploads $\gamma|V_i|d$ floats of super-node features X_i^S, receives the same-sized coarse forecast slice $Z_{i,T}$ 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 party-to-party exchanges, the P2P load remains only a fraction of the gradients or node embeddings transferred in conventional graph FL, and the party–server round trip is just $2\gamma|V_i|d$ floats. Except for the one-off $O(\gamma N^2)$ server broadcast, cumulative training traffic grows only linearly with γ while preserving structural privacy and accuracy. The theoretical communication-overhead comparison is provided in Appendix.

Experiments

Datasets

We evaluate FedSkeleton on four time-series graph datasets (National Renewable Energy Laboratory 2006; Kim et al. 2018), which include three real-world datasets and one synthetic dataset, covering power-grid operation, public-health surveillance, and generic network dynamics. Complete data sources, preprocessing and sliding-window settings are presented in Appendix A. Specifically, 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.

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 2021) decouples on-device temporal encoding and server-side spatial GNN to model decentralized spatio-temporal data.

Experimental Setup

We categorize baselines into two learning regimes. In the centralized regime, a single server has the complete graph and all node features and trains each baseline end-to-end. In the federated regime, topology and features remain isolated within each party, and FL-enabled baselines follow their official protocols. FedSkeleton is never trained on the full graph; in all results it runs strictly under our federated protocol, where each party keeps its own subgraph and the server only receives the binary skeleton produced by our secure construction, without raw features or edge counts. A fully decentralized variant is obtained by setting, eliminating cross-party information exchange. See implementation details in the Appendix.

	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 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
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**

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

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 As Table 2 shows, under 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 second-best algorithm T-PatchGNN. On ChileNet, its NMSE is only 0.036, corresponding to 90.0% of T-PatchGNN and 34.6% of CNFGNN. 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. 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.

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 the curves shown in Figure 2(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.

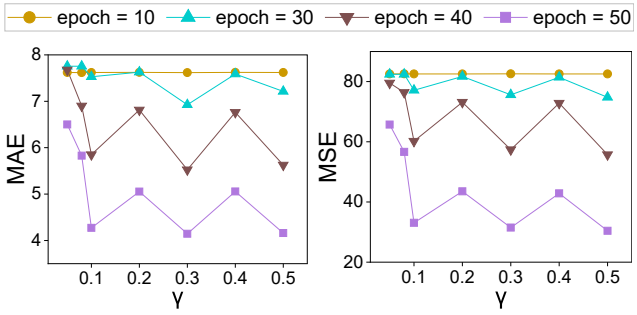

Figure 2: Prediction loss trends measured by MAE and MSE over different values of γ on the Solar dataset, with results recorded at epochs 10, 30, 40, and 50.

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.

The upper row of Figure 3 shows 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 we can observe, 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, the model can incorporate more global information, and the prediction error 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.

Both the Solar and Chile datasets show a similar trend regarding the influence of α. When α is too low (e.g., $\alpha = 0$), the model performs local-only forecasting, resulting in relatively high MAE and MSE values due to the lack of sufficient global structural information to generalize well. However, as α increases, the Chile dataset exhibits more performance fluctuation compared to the Solar dataset. 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.

Impact of Federation Scale To assess robustness under federation growth, we partition the synthetic dataset (Syn) into three federation sizes, i.e., 5, 20, and 50 parties, keeping the per-party data volume and all training hyper-parameters identical to the main experiments. As the number of parties 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 metrics, i.e., MSE and MAE, to visualize scaling trends. Figure 4 reports performance when the synthetic Syn dataset is split across 5, 20 and 50 parties. This shows that FedSkeleton is the most scale-robust: its MAE increases only from 0.0348 to 0.0572 and its MSE increases from 0.0030 to 0.0090, both showing only a moderate increase while remaining the

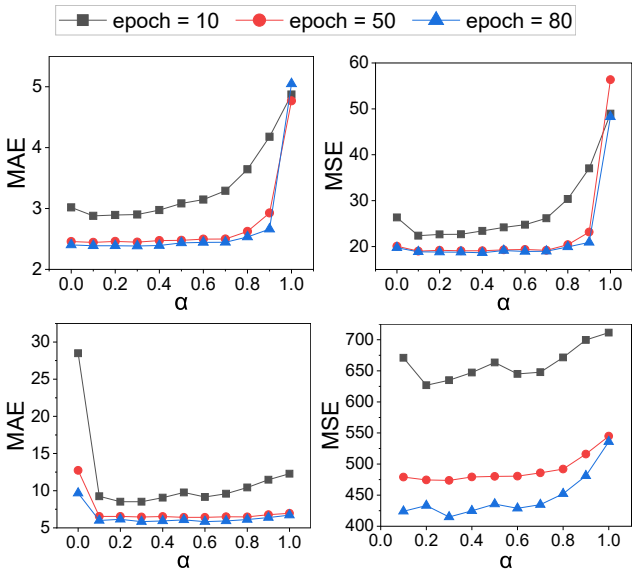

Figure 3: Prediction loss trends measured by MAE and MSE over different values of α on the Solar (top row) and Chile (bottom row) datasets, with results recorded at epochs 10, 50, and 80.

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%. 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. Under both 20 and 50 party settings, FedGC's MSE result (around 6.2) shows a significant gap compared to FedSkeleton. This appears to suggest that when data splits are highly heterogeneous, the effectiveness of FedGC's compression strategy in preserving structural information may be somewhat limited. Overall, FedSkeleton sustains the best error levels even as communication and aggregation costs rise, demonstrating superior scalability and robustness.

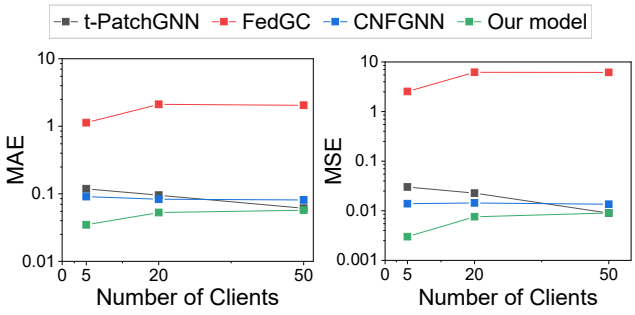

Figure 4: Prediction loss trends measured by MAE (left) and MSE (right) as the synthetic Syn dataset is partitioned across 5, 20, and 50 parties.

Additional Experiments

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 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.

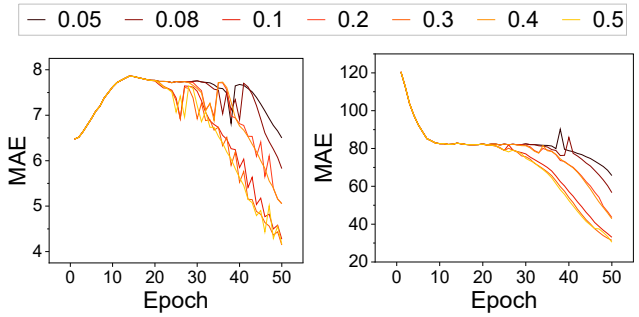

Figure 5: Attack loss trends measured by MAE and MSE over training epochs on the Solar dataset for different values of γ.

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 four datasets demonstrate that FedSkeleton exhibits competitive performance 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.

Acknowledgments

This work was supported by Ant Group through CCF-Ant Research Fund (CCF-AFSG RF20240208).

References

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.

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.

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.

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.

Goldreich, O.; Micali, S.; and Wigderson, A. 2019. How to play any mental game, or a completeness theorem for protocols with honest majority. In *Providing sound foundations for cryptography: on the work of Shafi Goldwasser and Silvio Micali*, 307–328.

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*.

Ishai, Y.; Kilian, J.; Nissim, K.; and Petrank, E. 2003. Extending oblivious transfers efficiently. In *Annual International Cryptology Conference*, 145–161. Springer.

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.

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.

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

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.

Patra, A.; Schneider, T.; Suresh, A.; and Yalame, H. 2021. {ABY2. 0}: Improved {Mixed-Protocol} secure {Two-Party} computation. In *30th USENIX Security Symposium (USENIX Security 21)*, 2165–2182.

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.

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.

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*.