FedDGL: Federated Dynamic Graph Learning for Temporal Evolution and Data Heterogeneity

Zaipeng Xie ZAIPENGXIE@HHU.EDU.CN
Key Laboratory of Water Big Data Technology of Ministry of Water Resources, Hohai University, Nanjing, China

Likun Li LILIKUN@HHU.EDU.CN
College of Computer Science and Software Engineering, Hohai University, Nanjing, China.

Xiangbin Chen XBCHEN@HHU.EDU.CN
College of Artificial Intelligence and Automation, Hohai University, Nanjing, China.

Hao Yu HAOYU@HHU.EDU.CN
Qian Huang HUANGQIAN@HHU.EDU.CN
College of Computer Science and Software Engineering, Hohai University, Nanjing, China.

Editors: Vu Nguyen and Hsuan-Tien Lin

Abstract

Federated graph learning enhances federated learning by enabling privacy-preserving collaborative training on distributed graph data. While traditional methods are effective in managing data heterogeneity, they typically assume static graph structures, overlooking the dynamic nature of real-world graphs. Integrating federated graph learning with dynamic graph neural networks addresses this issue but often fails to retain previously acquired knowledge, limiting generalization for both global and personalized models. This paper proposes FedDGL, a novel framework designed to address temporal evolution , and data heterogeneity in federated dynamic graph learning. Unlike conventional approaches, FedDGL captures temporal dynamics through a global knowledge distillation technique and manages client heterogeneity using a global prototype-based regularization method. The framework employs contrastive learning to generate global prototypes, enhancing feature representation while utilizing a prototype similarity-based personalized aggregation strategy for effective adaptation to local and global data distributions. Experiments on multiple benchmark datasets show that FedDGL achieves significant performance improvements over state-of-the-art methods, with up to 9.02% and 8.77% gains in local and global testing, respectively, compared to FedAvg. These results highlight FedDGL's effectiveness in improving personalized and global model performance in dynamic, heterogeneous federated graph learning scenarios.

Keywords: Federated Graph Learning, Dynamic Graph Learning, Knowledge Distillation, Prototype Learning, Personalized Aggregation, Temporal Evolution

1. Introduction

Federated Graph Learning (FGL) (Zhang et al. (2021)) has emerged as a specialized area within federated learning (FL), attracting significant attention in recent years. FGL focuses on the collaborative training of graph-based models across multiple decentralized clients (Zhang et al. (2021); Pei et al. (2021); Yao et al. (2023); Zhang et al. (2024); Chen et al. (2024)). This computing paradigm leverages the structural information in graph data by

© 2024 Z. Xie, L. Li, X. Chen, H. Yu & Q. Huang.

aggregating information from neighboring nodes to enhance model performance. However, FGL faces significant challenges due to data heterogeneity in practical applications, where substantial differences exist in local graph data distributions across various clients (Zhang et al. (2021)). This heterogeneity can adversely affect the generalization performance of the global model, given the variations in node distributions, edge types, and feature spaces among clients (Tan et al. (2023)). Current research (Wan et al. (2024)) suggests that the inability to effectively address data heterogeneity may undermine the efficacy of federated learning.

Additionally, real-world graphs are often dynamic, with nodes and edges continuously evolving due to the nature of the underlying systems they represent. For instance, in social networks, users (nodes) regularly join and leave, while friendships and interactions (edges) change over time. In traffic networks (Han et al. (2023)), the connections (edges) between locations (nodes) vary throughout the day due to fluctuating traffic conditions. Similarly, in e-commerce recommendation systems (Yin and Yue (2023)), new products are added, and user preferences evolve, leading to changes in interactions. These examples underscore the necessity for models that can accommodate the temporal and structural changes in graph data, making Dynamic Graph Learning (DGL) (Pareja et al. (2020); Pang et al. (2021); Sun et al. (2022); Zhang et al. (2023b)) essential for accurately capturing and predicting the behavior of such systems. (Han et al. (2023); Yin and Yue (2023))

Various methods (Pareja et al. (2020); Pang et al. (2021); Yu et al. (2023)) have been developed to tackle dynamic graph learning (DGL) by predicting changes in graphs over time. EvolveGCN (Pareja et al. (2020)) leverages recurrent neural networks (RNNs) to dynamically update the parameters of graph convolutional networks (GCNs), efficiently managing frequent alterations in node sets. DM-GCN (Pang et al. (2021)) utilizes an attention mechanism with multiple attention heads to enhance the model's capacity to detect complex patterns. DyGFormer (Yu et al. (2023)) introduces a neighbor co-occurrence encoding scheme to capture correlations between nodes and employs a patching technique to process longer node histories effectively.

However, these DGL methods can be ineffective in a federated scenario due to several key challenges (Jiang et al. (2022)). Data heterogeneity across clients leads to inconsistencies in model aggregation, while the frequent updates required to capture dynamic changes significantly increase communication overhead and latency (Thakur et al. (2022)). Additionally, the detailed updates necessary for dynamic graphs can compromise data privacy (Wang et al. (2024)). The computational intensity of DGL methods poses challenges for clients with limited resources, and managing updates from numerous clients with complex graphs can be computationally demanding, affecting scalability. These issues highlight the need for specialized methods in federated DGL. Among all the challenges of federated DGL, two critical challenges persist: (1) accurately capturing temporal dynamics in graph snapshots to prevent knowledge loss in local models, and (2) effectively managing data heterogeneity across snapshots to enhance the global model's generalization capabilities across diverse distributions.

This study proposes a novel Federated Dynamic Graph Learning (FedDGL) method to address the aforementioned two primary challenges. FedDGL captures local evolution relationships of graph snapshots by selecting sensitive nodes based on estimated loss differences and retaining previously learned information through knowledge distillation. To

capture global evolution relationships, FedDGL aggregates local prototypes from all clients to generate global prototypes that guide local training. Additionally, contrastive learning is utilized in FedDGL to enhance feature representations by improving intra-class compactness and inter-class separability. A prototype-based personalized aggregation method is proposed for personalized federated training. By integrating continual learning and prototype learning, clients can effectively capture the temporal evolution of graph snapshots, maintain predictive performance on global graph snapshots, and achieve improved personalized training, thereby mitigating these challenges. Our code is publicly available at https://github.com/zach82/FedDGL and our main contributions are summarized as follows:

- We propose a novel personalized Federated Dynamic Graph Learning (FedDGL) method to address the issues of temporal evolution in graph snapshots and data heterogeneity among clients in federated graph learning. FedDGL captures the temporal evolution of graph snapshots and manages data heterogeneity through a temporal evolution capture module and a prototype-based global regularization method, enhancing both the generalization performance of the global model and the personalization for each client.

- Our proposed FedDGL introduces global prototypes generated through contrastive learning to enhance intra-class compactness and inter-class separability of feature representations. It also employs a prototype-based personalized aggregation method, enabling each client's model to adapt to its local data distribution while maintaining alignment with the global data distribution.

- Experimental results demonstrate that FedDGL significantly outperforms state-of-the-art methods across multiple benchmark datasets, enhancing both personalized and global model performance. Its ability to improve global model generalization and effectively adapt to client-specific data through personalized aggregation underscores its potential as a robust solution for federated dynamic graph learning.

2. Related Work

Federated Graph Learning (FGL) (Zhang et al. (2021); Pei et al. (2021); Yao et al. (2023); Zhang et al. (2024); Chen et al. (2024)) combines the privacy-preserving features of federated learning with the capabilities of graph neural networks, enabling collaborative training on decentralized data. It is an effective solution for advanced graph-based models in various domains. For example, FedSage (Zhang et al. (2021)) addresses distributed subgraph systems, training a missing neighbor generator to restore edges between clients, while FedEgo (Zhang et al. (2024)) constructs K-hop ego graphs for server-side use. D-FedGNN (Pei et al. (2021)) supports client collaboration without a central server, and FedGL (Chen et al. (2024)) uploads predicted results and node embeddings. FedGCN (Yao et al. (2023)) exchanges average neighbor information between clients. However, these frameworks may pose privacy risks when node embeddings, subgraphs, etc., are shared during the federated learning process, potentially revealing nodes within a client's private dataset (Zhang et al. (2024)).

Data heterogeneity in FGL involves variations in graph structures, node features, and edge relationships across clients, resulting in non-IID data. This necessitates personalized

models that adapt to local distributions, enhancing performance and providing tailored recommendations. Addressing this heterogeneity is crucial for effective, privacy-preserving, and scalable federated learning systems. To tackle data heterogeneity, personalization in federated graph learning has gained attention (Cheng et al. (2023); Wang et al. (2024, 2023)). ProtoHAR (Cheng et al. (2023)) introduces prototypes into the FGL framework to decouple representation and classifier in heterogeneous environments, using global prototypes to correct activity feature representations and allow knowledge exchange between clients without compromising privacy. Other methods, such as clustering-based methods (Wang et al. (2024)), and mixing global and local models (Wang et al. (2023)), have proven effective in enhancing personalized models.

Dynamic Graph Neural Networks (Pareja et al. (2020); Pang et al. (2021); Zhang et al. (2023a)) are specialized neural networks designed to model and analyze evolving graph structures and features over time, capturing temporal dynamics and adapting to changes in the graph topology. For instance, EvolveGCN (Pareja et al. (2020)) integrates recurrent neural networks with graph convolutional networks to capture the temporal evolution of graph structures and node features. DM-GCN (Pang et al. (2021)) employs an attention mechanism, enhancing the model's ability to capture complex patterns by computing multiple attention heads in parallel. DRGCN (Zhang et al. (2023a)) addresses the over-smoothing problem in GCNs by introducing dynamic blocks that adaptively fetch information for each node from initial representations and evolving blocks that model the residual evolving pattern between layers. These advancements underscore the importance of developing robust and adaptive models to handle the complexities of dynamic graph data effectively.

Developing a training method for Federated Dynamic Graph Learning presents several unique challenges (Wang et al. (2024)). Synchronizing dynamic graph data across decentralized clients is complex due to varying rates of graph evolution (Jiang et al. (2022)), which complicates aligning temporal aspects for collaborative learning. Ensuring graph data privacy while handling sensitive temporal patterns is critical, necessitating advanced privacy-preserving techniques (Thakur et al. (2022)). The risk of model skew, the need for continuous learning mechanisms, robustness to temporal anomalies, and accommodating clients' resource constraints further complicate Federated Dynamic Graph Learning. These challenges necessitate innovative methods for effective and secure federated graph learning across evolving decentralized datasets.

3. Problem Formulation

This study addresses the problem of integrating federated learning with dynamic graph learning to effectively model and analyze time-evolving graph data distributed across multiple clients. The objective of personalized federated dynamic graph learning is to train personalized models locally, formulated as follows:

$$\min_{\{w_k^{(t)}\}} \sum_{k=1}^{K} \mathcal{L}_{CE,k}^{(t)}, \quad \text{where} \quad \mathcal{L}_{CE,k}^{(t)} = -\sum_{c=1}^{C} Y_k^{(t)} \log\left(f\left(\mathcal{V}_k^{(t)}, \tilde{\mathcal{A}}_k^{(t)}; w_k^{(t)}\right)\right) \tag{1}$$

Here, K denotes the total number of clients, each possessing local dynamic graph datasets $\mathcal{D}_k$. The parameter $w_k^{(t)}$ represents the model parameters for client k at time t, where

$t \in \{1, \ldots, T\}$ and T is the number of graph snapshots. $Y_k^{(t)}$ is the ground truth label for nodes in the set $\mathcal{V}_k^{(t)}$ of the client k at time t. The normalized adjacency matrix $\tilde{\mathcal{A}}_k^{(t)} = \mathcal{A}_k^{(t)} + I$ incorporates synchronous node features, with $\mathcal{A}_k^{(t)}$ representing these features at time t. C indicates the number of classes.

In the federated learning framework, each client k trains its local model on the dataset $\mathcal{D}_k$ and updates its parameters using Stochastic Gradient Descent:

$$w_k^{(t+1)} = w_k^{(t)} - \eta \cdot \nabla \mathcal{L}_{CE,k}^{(t)} \tag{2}$$

where η is the learning rate. The server aggregates the updates from the clients using the FedAvg (McMahan et al. (2017)) algorithm to construct the global model $w_g^{(t)}$.

4. Methodology

We propose a novel personalized federated dynamic graph learning (FedDGL) framework to capture temporal relationships and address data heterogeneity in FL. FedDGL simultaneously captures temporal evolutionary relationships in dynamic graph snapshots and addresses client data heterogeneity. Figure 1 illustrates our FedDGL framework, which includes a temporal evolution capture using global knowledge distillation for capturing temporal relationships. To manage heterogeneity, we introduce a global prototype-based regularization method, enhancing both the generalization of the global model and the personalization of each client's model. Additionally, FedDGL uses a contrastive learning-based global prototype representation to improve feature learning and differentiation, facilitating class-specific prototypes and enhancing inter-class dispersion. Finally, a prototype similarity-based personalized aggregation method ensures efficient, personalized training.

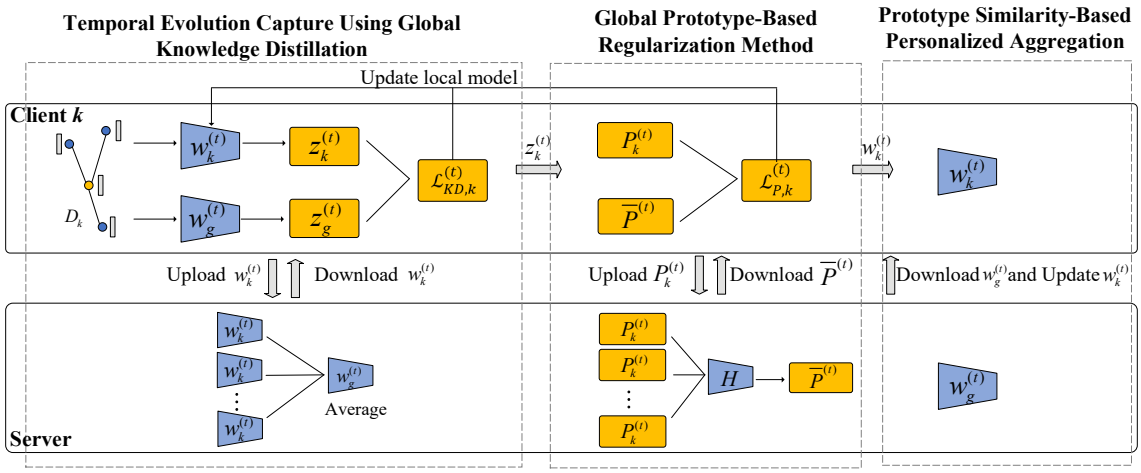

Figure 1: The overall architecture of FedDGL.

4.1. Temporal Evolution Capture Using Global Knowledge Distillation

In federated dynamic graph learning, the evolving graph structure changes the receptive field of nodes, impacting model performance over time. We propose a temporal evolution

capture method based on global knowledge distillation to effectively capture this temporal evolution. This module selects sensitive nodes by estimating loss differences and retains previously learned information through knowledge distillation, enhancing model adaptability in dynamic environments. Specifically, for the k-th client and the t-th snapshot, the optimization objective of this module is:

$$\mathcal{L}_{KD,k}^{(t)} = \arg\min_{w_k^{(t)}} \left(\mathcal{L} + \gamma \left(\mathbb{E}_{i \in \mathcal{S}_k} D_{KL} \left(z_{i,k}^{(t)} \| z_{i,g}^{(t-1)} \right) \right) \right) \tag{3}$$

where $\mathcal{L}$ is the loss on the true labels, S_k is the set of nodes for client k up to the t-th snapshot, $z_{i,k}^{(t)}$ is the feature of node i output by the model at time t for client k, $z_{i,g}^{(t-1)}$ is the feature of node i output by the global model at time $t-1$, and γ is a hyperparameter. By considering both the distillation loss and the ground truth loss, the model learns from previous snapshots and mitigates distribution shifts.

To optimize this objective, we first identify the set of nodes S_k for client k before the t-th snapshot. Active nodes are then marked at the current time step, distinguishing between nodes in the current snapshot and those from previous snapshots. This process ensures the model accurately tracks the state of each node over different time points.

$$V_{ii,k}^{(0)} = \begin{cases} 1 & \text{if } v_i \in \mathcal{V}_k^{(t)} \\ 0 & \text{if } v_i \in \mathcal{M}_k \end{cases} \tag{4}$$

where $\mathcal{M}_k$ is the memory buffer, representing the set of snapshot nodes that the k-th client has already learned. Next, the node set of the L-hop neighborhood is calculated as follows:

$$V_k^l = A_k^{(t)} V_k^{(l-1)}, \quad l = \{1, \ldots, L\} \tag{5}$$

Here, L is the depth of the model, and V_k^l represents the set of neighbor nodes at different hops for each node, effectively capturing the structural changes of the graph. Since the choice of L is crucial, we align L with the model depth to ensure balanced information aggregation and effective propagation.

Subsequently, the set of nodes $V_{C,k}$ that do not belong to the L-hop neighborhood of the current node is filtered out. These nodes will be used for subsequent loss evaluation and candidate node selection:

$$V_{C,k} = \left\{ v_i \mid \sum_{j=1}^{N} \sum_{l=1}^{L} V_{ij,k}^{(l)} = 0, i = 1, \ldots, N \right\} \tag{6}$$

where N is the number of nodes. By calculating the loss $\mathcal{L}_k^{(t-1)}$ of the node set $V_{C,k}$ in the previous model, we can evaluate the loss situation of these nodes in the previous model:

$$\mathcal{L}_k^{(t-1)} = \ell \left(f \left(\mathcal{V}_{C,k}, \tilde{\mathcal{A}}_k^{(t-1)}; w_k^{(t-1)} \right) \right) \tag{7}$$

Based on the threshold q, we select the set of sensitive candidate nodes V_k' from $V_{C,k}$ as those with losses below the threshold in the previous model. This ensures the model focuses on nodes that are sensitive to new inputs:

$$V_k' = \left\{ v_i \in V_{C,k} \mid \mathcal{L}_{v_i,k}^{(t-1)} < q \cdot \mathcal{L}_k^{(t-1)} \right\} \tag{8}$$

After identifying these sensitive candidate nodes, we evaluate their loss values under the current input to calculate their losses under the new graph snapshot for subsequent influence evaluation:

$$\mathcal{L}_k^{(t)} = \ell\left(f\left(\mathcal{V}_k', \tilde{\mathcal{A}}_k^{(t)}; w_k^{(t-1)}\right)\right) \tag{9}$$

By comparing the differences between $\mathcal{L}_k^{(t)}$ and $\mathcal{L}_k^{(t-1)}$, the impact of the candidate nodes is calculated as:

$$\pi_k = \left\{\mathcal{L}_{v_{i,k}}^{(t)} - \mathcal{L}_{v_{i,k}}^{(t-1)} \mid v_i \in \mathcal{V}_k'\right\} \tag{10}$$

Next, the top k most influential candidate nodes are selected from π_k, denoted as $S_k = \{v_i \mid v_i \sim \text{top}k(\pi_k)\}$. This ensures the model focuses on significantly influential new input nodes. To retain important information learned from previous time steps, we can calculate the distillation loss. Using knowledge distillation, the model retains previously learned knowledge while normalizing node features to ensure the stability and comparability of feature vectors during training. The distillation loss is calculated as follows:

$$\mathcal{L}_{KD,k}^{(t)} = -\sum_{c=1}^{C} Y^{S_k} \log f(S_k; w_k^{(t)}) + \gamma \sum_{i \in S_k} \left\| z_{i,k}^{(t)} - z_{i,g}^{(t-1)} \right\|_2 \tag{11}$$

where the first term on the right represents the cross-entropy loss, penalizing incorrect predictions under distributional shifts during knowledge distillation. C is the number of classes in $\mathcal{D}_k$, Y^{S_k} is the true label of S_k, $f(\cdot)$ denotes the classifier prediction, and γ is the distillation coefficient. Term $z_{i,k}^{(t)}$ is the feature of node i output by the model of client k at time t, and $z_{i,g}^{(t-1)}$ is the feature of node i output by the global model $w_g^{(t-1)}$ at time $t-1$.

This process helps capture the evolution relationships of dynamic graph snapshots and retains previously learned information to enhance the performance and adaptability of the local model.

4.2. Global Prototype-Based Regularization Method

We propose a global prototype-based regularization method to address the diversity of data distributions across various clients. This method enhances the generalization of the global model while preserving the personalization of each client's model. We implement a contrastive learning method to train the global prototypes, which enhances the differentiation between learned features and various global class prototypes. By synthesizing knowledge from participating clients, our regularization method constructs a more accurate and generalized model, improving adaptability and performance across diverse data distributions. Our temporal evolution capture method, essential for understanding dynamic graph relationships, is thus complemented and strengthened by this method.

At each training round t, each client k calculates the class prototypes $P_k^{(t)}$ for its local data and uploads these prototypes to the server:

$$P_{k,c}^{(t)} = \frac{1}{\left|v_{k,c}^{(t)}\right|} \sum_{x \in v_{k,c}^{(t)}} h(x; w_k^{(t)}) \tag{12}$$

where $h(\cdot)$ represents the intermediate features extracted from the model output, and $\mathcal{V}_{k,c}^{(t)}$ denotes the nodes of class c owned by client k. Upon receiving the local prototypes, the server initializes a trainable vector $\bar{P}^{(t)}$ for each class c. Using a neural network model H parameterized by ϕ, the server trains $\bar{P}^{(t)}$ to obtain the global prototype:

$$\bar{P}^{(t+1)} = H(\bar{P}^{(t)}; \phi) \tag{13}$$

To learn effective prototypes, the global prototype for class c needs to achieve two objectives: (1) align closely with the client prototypes of class c to preserve semantic information, and (2) maintain a significant distance from the client prototypes of other classes to enhance separability. Contrastive learning inherently supports these goals by promoting both tightness within the same class and separability between different classes. Therefore, the optimization objective of this method is as follows:

$$\min_{\bar{P}^{(t)}} \mathcal{L}_{\text{cons},c}^{(t)}, \text{ where } \mathcal{L}_{\text{cons},c}^{(t)} = -\sum_{k=1}^{K} \log \frac{d(k,c,c,t)}{d(k,c,c,t) + \sum_{c' \neq c} d(k,c,c',t)} \tag{14}$$

where $d(k,c,c',t) = e^{-\cos(p_{k,c}^{(t)}, \bar{p}_{c'}^{(t)})}$. Our proposed method ensures that each class's prototype has a clear margin in the feature space while maintaining compactness within the same class. By enhancing the separability and quality of global prototypes, our proposed method can significantly improve the overall performance of client models.

To constrain local client training, the cosine similarity between the global and local prototypes is calculated to measure their differences. This difference is incorporated into the local training loss function as a regularization term, defined as:

$$\mathcal{L}_{P,k}^{(t)} = \frac{1}{|C_k|} \sum_{c \in \mathcal{C}_k} \cos(P_{k,c}^{(t)}, \bar{P}_c^{(t)}) \tag{15}$$

where C_k is the number of categories owned by client k, $\mathcal{L}_{P,k}^{(t)}$ can guide the training of each client, reducing the performance differences caused by data heterogeneity between clients, thereby improving the overall performance of the entire system.

The model parameters are updated by considering the current input classification loss $\mathcal{L}_{CE,k}^{(t)}$, distillation loss $\mathcal{L}_{KD,k}^{(t)}$, and global regularization term $\mathcal{L}_{P,k}^{(t)}$, resulting in the final loss function $\mathcal{L}_k^{(t)}$:

$$\mathcal{L}_k^{(t)} = \mathcal{L}_{CE,k}^{(t)} + \mathcal{L}_{KD,k}^{(t)} + \mathcal{L}_{P,k}^{(t)} \tag{16}$$

During local model training, gradient descent is used to minimize the total loss to improve model performance:

$$w_k^{(t+1)} = w_k^{(t)} - \eta \cdot \nabla \mathcal{L}_k^{(t)} \tag{17}$$

The loss function $\mathcal{L}_k^{(t)}$ captures temporal evolution in dynamic graphs and manages client data heterogeneity through global prototype-based regularization, thereby reducing dependency on prototype quality. This dual consideration enhances the generalization and stability

of the global model. Moreover, contrastive learning in prototype generation improves intra-class compactness and inter-class separability, further mitigating the impact of insufficient prototype learning and boosting the model's discriminative capability.

4.3. Prototype Similarity-Based Personalized Aggregation

We propose a personalized aggregation method based on prototype similarity to achieve personalized training. During model updates, each client uses a mixture of local and global weights to leverage the advantages of the global model. First, the residual similarity between the local prototype $P_k^{(t)}$ of the k-th client and the global prototype $\bar{P}^{(t)}$ is calculated to quantify the similarity between the local and global models, as shown in the following formula:

$$\cos_k^{(t)} = \frac{P_k^{(t)} \cdot \bar{P}^{(t)}}{\|P_k^{(t)}\| \|\bar{P}^{(t)}\|} \tag{18}$$

Next, the similarities of all clients are normalized to obtain personalized weights λ_k, which are then used to update the local model:

$$w_k^{(t+1)} = \lambda_k w_k^{(t)} + (1-\lambda_k)\frac{1}{K}\sum_{k=1}^{K} w_k^{(t)}, \ \ where \ \lambda_k = \frac{\cos_k^{(t)}}{\sum_{k=1}^{K} \cos_k^{(t)}} \tag{19}$$

Algorithm 1 FedDGL

Require: At time step $t > 1$: Local dataset $\mathcal{D}_k$; Local model $w_k^{(t)}$; Round number E
Ensure: Personalized models $w_k^{(t)}$, $k \in \{1, \ldots, K\}$

```
Server initializes global client personalized models w_k^(t)
Distribute personalized client models w_k^(t) to each client
for e ∈ {1, ··· , E} do
    for client k ∈ {1, ··· , K} in parallel do
        Calculates candidate node set S_k
        Obtains the local feature z_k^(t) and global node feature z_g^(t−1)
        for each local epoch do
            for batch ∈ D_k do
                Calculate L_CE,k^(t), L_KD,k^(t), L_P,k^(t) by Eq. (1), Eq. (11), Eq. (15)
                Update model parameters w_k^(t) by Eq. (17)
            end for
        end for
        Calculates local prototype P_k^(t) by Eq. (12)
    end for
    Update global prototype P̄^(t) by Eq. (14)
    w_k^(t+1) ← λ_k w_k^(t) + (1 − λ_k) (1/K) Σ_{k=1}^K w_k^(t)
end for
return w_k^(t+1), k ∈ {k, ..., K}
```

The personalized aggregation method based on prototype similarity dynamically adjusts the weights of local models for efficient personalized training. This ensures each client's model adapts to its data distribution while effectively accommodating the global data distribution. Algorithm 1 illustrates the overall process of the proposed FedDGL method. In our proposed framework, the computational complexity is primarily determined by the size of the model and the volume of local data processed by each client. Efficient management of these factors helps maintain scalability, even as the model or dataset grows. The communication complexity scales linearly with the number of participating clients and the quantity of data transmitted during training rounds. This linear growth ensures that the framework remains efficient and practical for large-scale deployments, minimizing the impact of increasing client numbers and data volume on overall system performance.

5. Experiment

5.1. Experimental Setup

5.1.1. Datasets and Baselines

We evaluate our method on Brain, DBLP-3, DBLP-5 (Tang et al. (2008)), and Reddit (Kumar et al. (2018)) datasets. The Brain[1] dataset comprises 5,000 nodes, 1,955,488 edges, 20 attributes, and 10 classes, spanning 12 timesteps, where nodes represent small cubes of brain tissue. The DBLP-3 and DBLP-5 datasets[2] are extracted from the bibliography website DBLP, the DBLP-3 and DBLP-5 datasets include 4,257 and 6,606 nodes, respectively, representing authors in different research areas, with 3 classes in DBLP-3 and 5 classes in DBLP-5. Both datasets contain 100 attributes and cover 10 timesteps. The Reddit[3] dataset consists of 8,291 nodes, 264,050 edges, 20 attributes, and 4 classes, covering 10 timesteps, where nodes represent posts and edges connect posts with similar keywords.

We benchmark our FedDGL against nine state-of-the-art federated learning methods, divided into federated static and dynamic graph learning algorithms. The static algorithms include FedAvg (McMahan et al. (2017)), FedSage (Zhang et al. (2021)), FedProto (Tan et al. (2022)), FedEgo (Zhang et al. (2024)), FedGCN (Yao et al. (2023)), and D-FedGNN (Pei et al. (2021)), all utilizing the static GCN (Kipf and Welling (2017)) as their base model. The dynamic algorithms include FedAvgDyn, FedSageDyn, and FedProtoDyn, based on the dynamic GNN model EvolveGCN (Pareja et al. (2020)).

5.1.2. Tasks

We conduct experiments on dynamic graph classification tasks, where the graph structure evolves over T dynamic graphs in the dataset. To simulate a realistic federated environment, a central server collaborates with K clients, each holding a portion of the entire dynamic graph. We adopt two data partitioning strategies proposed by (Zhang et al. (2024)) for federated dynamic graph classification tasks: label-skew and community clustering.

The label-skew strategy partitions data based on label distribution, ensuring each client's data contains different proportions of labels. This simulates non-IID data scenarios common

1. https://neurodata.io/project/ocp/
2. https://www.aminer.cn/billboard/citation
3. https://snap.stanford.edu/data/soc-RedditHyperlinks.html

in real-world federated learning. The community clustering strategy uses community detection algorithms to partition the graph, grouping densely connected nodes. This preserves the graph's structural characteristics within each client's data, improving the model's ability to learn from local graph structures.

We evaluate the effectiveness of FedDGL on two tasks: local and global. The local task assesses the model's performance on datasets reflecting the clients' local data distribution, providing insights into its personalized learning capability. The global task uses a test dataset reflecting the aggregated data distribution of all clients, measuring the model's ability to generalize and manage local knowledge biases on a larger scale.

5.1.3. Implementation Details

Our FedDGL framework is implemented in PyTorch and deployed across 10 clients, each participating in every communication round with one training epoch per round. A subset of nodes is excluded from the global dataset, and the remaining nodes are distributed among clients as local datasets.

To simulate label-skew scenarios, nodes are divided by labels. Each client randomly selects two primary node labels, making up 80% of its local dataset, with the remaining 20% from non-primary labels. For each client, 10% of the nodes are used for testing, 20% for validation, and the rest for training. In the community clustering scenario, we use the Louvain method (Blondel et al. (2008)) to extract community clusters, assigning nodes from one cluster as the global test set. Each node includes its two-hop neighbors, with a maximum of six neighbors for self-graph sampling.

5.2. Label-skew Scenarios Analysis

5.2.1. Personalization Ability

Table 1 presents the local test F1 scores, demonstrating the personalization capabilities of each method in the label-skew scenario. The local test results are significantly higher than the global test results, primarily because the training and test data distributions are identical. Additionally, federated dynamic graph algorithms generally outperform federated static graph algorithms. FedDGL outperforms all algorithms across all datasets. For example, on the Brain dataset, FedDGL outperforms the state-of-the-art FedProtoDyn by 2.40% and exceeds FedAvg by 9.02%. FedGCN performs particularly poorly on the DBLP-3 and Brain datasets, with scores of 57.48% and 47.65%, respectively, likely due to its insufficient generalization capability when handling complex non-IID data. Conversely, D-FedGNN performs the worst on the Reddit dataset, with a score of 26.02%, potentially due to the high complexity of the Reddit dataset and its inability to effectively capture relationships between graph structures and node features, resulting in poor model performance.

5.2.2. Generalization Ability

The comparison of F1 scores in Table 1 also reveals the superior generalization ability of FedDGL compared to other methods. By leveraging past knowledge, clients achieve a 7.48%-8.77% performance improvement over FedAvg. This demonstrates FedDGL's ability to facilitate client collaboration and address the non-IID problem in dynamic graph data.

Table 1: F1 scores for node classification on local (L) and global (G) tasks across four datasets under label-skew scenarios.

Method	Test Average Acc							
	DBLP-5 (L)	DBLP-5 (G)	DBLP-3 (L)	DBLP-3 (G)	Brain (L)	Brain (G)	Reddit (L)	Reddit (G)
FedAvg	0.7807	0.6446	0.5834	0.4958	0.5040	0.3895	0.2529	0.1984
FedSage	0.7949	0.6753	0.6028	0.5224	0.5262	0.4186	0.2597	0.2064
FedProto	0.8018	0.6897	0.6263	0.5491	0.5375	0.4415	0.2749	0.2241
FedEgo	0.8076	0.6891	0.6202	0.5066	0.5378	0.4152	0.2665	0.2246
FedGCN	0.7640	0.6329	0.5748	0.4941	0.4765	0.3550	0.2477	0.1887
D-FedGNN	0.8228	0.6787	0.6009	0.5188	0.5494	0.4399	0.2603	0.2033
FedAvgDyn	0.7914	0.6733	0.5982	0.5381	0.5457	0.4248	0.2794	0.2353
FedSageDyn	0.8133	0.6856	0.6236	0.5439	0.5547	0.4316	0.3016	0.2506
FedProtoDyn	0.8347	0.7099	0.6316	0.5648	0.5701	0.4415	0.3017	0.2487
FedDGL(Ours)	**0.8431**	**0.7243**	**0.6416**	**0.5836**	**0.5942**	**0.4706**	**0.3204**	**0.2733**

It is worth noting that FedProtoDyn also demonstrates good performance in global tests, indicating that combining prototype learning with dynamic graph neural networks can effectively enhance generalization. In contrast, FedGCN and D-FedGNN perform poorly, likely due to insufficient generalization capabilities when handling complex non-IID data.

5.3. Community Clustering Scenarios Analysis

5.3.1. Personalization Ability

Table 2 shows the F1 scores for node classification on local (L) and global (G) tasks across four datasets under Community Clustering scenarios. The significant decline in performance of all algorithms in the label-skew scenario suggests that the local structures captured by the community detection algorithm may be challenging for GNNs to train and learn from. Federated dynamic graph algorithms continue to perform well, generally surpassing federated static graph algorithms. FedDGL excels in local tests and significantly outperforms FedAvg. Although slightly inferior to FedProtoDyn in DBLP-3 and Reddit, FedDGL still achieves the second-highest scores. Notably, FedGCN exhibits high performance in local evaluations but demonstrates suboptimal results in global assessments, presumably due to its inability to model inter-graph node connections, resulting in a localized focus on individual datasets.

Table 2: F1 scores for node classification on local (L) and global (G) tasks across four datasets under Community Clustering scenarios. FedDGL achieves the best results on local tasks in DBLP-5 and Brain and is slightly weaker than FedProtoDyn on DBLP-3 and Reddit.

Method	Test Average Acc							
	DBLP-5 (L)	DBLP-5 (G)	DBLP-3 (L)	DBLP-3 (G)	Brain (L)	Brain (G)	Reddit (L)	Reddit (G)
FedAvg	0.5914	0.6141	0.5173	0.4925	0.3898	0.3896	0.2318	0.1995
FedSage	0.5973	0.6234	0.5137	0.4965	0.4054	0.4024	0.2285	0.1989
FedProto	0.5969	0.6367	0.5069	0.4994	0.4084	0.4148	0.2426	0.2121
FedEgo	0.6106	0.6349	0.5278	0.5172	0.3832	0.3881	0.2535	0.2158
FedGCN	0.6010	0.5997	0.5145	0.4737	0.4040	0.3775	0.2414	0.1803
D-FedGNN	0.6137	0.6239	0.5166	0.5047	0.4200	0.4411	0.2361	0.2032
FedAvgDyn	0.6140	0.6334	0.5278	0.4967	0.3976	0.4038	0.2529	0.2068
FedSageDyn	0.6223	0.6458	0.5330	0.5145	0.4082	0.4213	0.2563	0.2119
FedProtoDyn	0.6283	0.6455	**0.5494**	0.5296	0.4196	0.4302	**0.2882**	0.2316
FedDGL(Ours)	**0.6354**	**0.6649**	0.5469	**0.5360**	**0.4307**	**0.4512**	0.2757	**0.2320**

5.3.2. Generalization Ability

The global test results in the community clustering scenario, shown in Table 2, demonstrate that FedDGL continues to achieve state-of-the-art performance, exceeding FedAvg by 3.25%-6.15% and surpassing FedProtoDyn by 0.63%-1.90%. Notably, the performance of baselines aligns with the conclusions drawn from the label-skew scenario.

5.4. Ablation Study

To further verify the effectiveness of each module in FedDGL, we perform three ablation studies: (1) FedDGL w/o KD: To investigate the effectiveness of the temporal evolution capture method based on global knowledge distillation, we conducted experiments without the knowledge distillation component. (2) FedDGL w/o GP: To examine the effectiveness of constraining local training using global prototypes, we conducted experiments without the global prototype component. (3) FedDGL w/o PA: To explore the effectiveness of the personalized aggregation method based on prototype similarity, we conducted experiments without the personalized aggregation component. The experimental results are shown in Figure 2, and the detailed results are reported in Table 3.

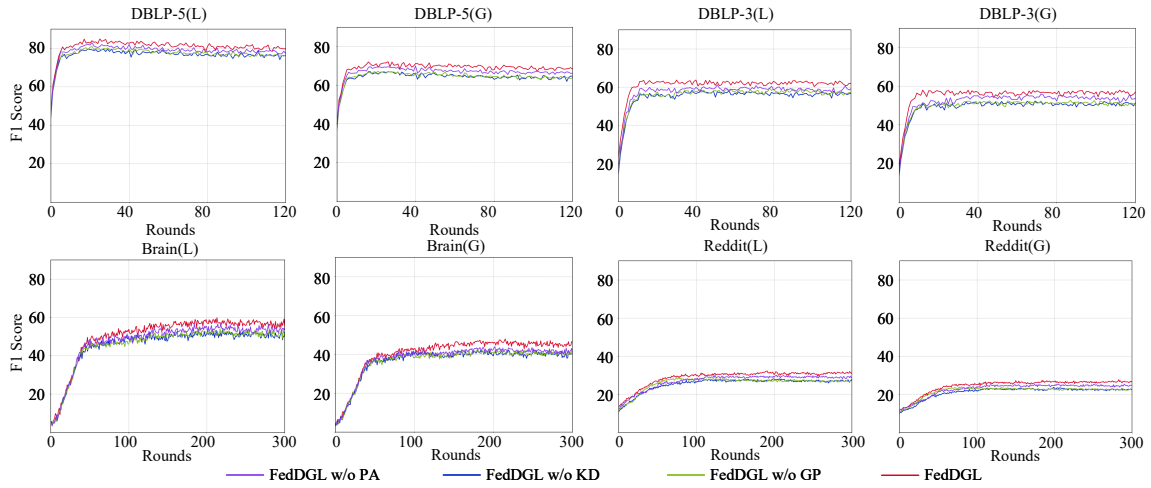

Figure 2: Comparison between FedDGL, FedDGL w/o KD, FedDGL w/o GP, and FedDGL w/o PA, where (L) represents local tasks and (G) represents global tasks.

Table 3 summarizes the F1 scores for node classification of local and global tasks across four datasets under label-skew scenarios. The poor performance of FedDGL w/o KD suggests that capturing past node information is crucial for enhancing model memorization. Without the temporal evolution capture module, clients tend to forget learned information, negatively affecting both local and global generalization. Similarly, the poor performance of FedDGL w/o GP highlights the effectiveness of the regularization method based on global prototypes. Without this regularization, client training deviates from the global data distribution, impairing generalization. Additionally, personalized aggregation plays a critical role in performance improvement. Replacing this module with federated averaging reduces train-

Table 3: F1 scores for node classification on local and global tasks across four datasets under label-skew scenarios. FedDGL consistently surpasses FedDGL w/o KD, FedDGL w/o GP, and FedDGL w/o PA on local tasks and excels on global tasks.

	Local				Global			
Methods	DBLP-5	DBLP-3	Brain	Reddit	DBLP-5	DBLP-3	Brain	Reddit
FedDGL w/o KD	0.7952	0.5824	0.5394	0.2814	0.6696	0.5227	0.4232	0.2365
FedDGL w/o GP	0.8094	0.5935	0.5473	0.2865	0.6716	0.5260	0.4295	0.2398
FedDGL w/o PA	0.8291	0.6116	0.5658	0.3007	0.6990	0.5584	0.4402	0.2536
FedDGL	**0.8430**	**0.6416**	**0.5941**	**0.3204**	**0.7242**	**0.5836**	**0.4706**	**0.2732**

ing effectiveness, as local models may fail to adequately capture diverse data characteristics from different clients, leading to performance degradation.

6. Conclusion

This paper presented FedDGL, a novel framework for federated dynamic graph learning that addresses temporal evolution and data heterogeneity challenges. FedDGL integrates global knowledge distillation, prototype-based regularization, and contrastive learning to enhance both personalized and global model performance. Experimental results across multiple datasets demonstrate FedDGL's significant improvements over state-of-the-art methods, with up to 9.02% and 8.77% gains in local and global testing, respectively, compared to FedAvg. These findings highlight FedDGL's effectiveness in managing federated dynamic graph learning environments. Future work could explore FedDGL's scalability to larger graph structures and its application in specific domains. Overall, FedDGL represents a substantial advancement in federated learning for dynamic graph data, offering a promising solution for real-world scenarios involving evolving graph structures and distributed data.

References

Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, 2008. ISSN 1742-5468.

Chuan Chen, Ziyue Xu, Weibo Hu, Zibin Zheng, and Jie Zhang. FedGL: Federated graph learning framework with global self-supervision. *Information Sciences*, 657:119976, 2024.

Dongzhou Cheng, Lei Zhang, Can Bu, Xing Wang, Hao Wu, and Aiguo Song. ProtoHAR: Prototype guided personalized federated learning for human activity recognition. *IEEE Journal of Biomedical and Health Informatics*, 27(8):3900–3911, 2023.

Seunghoon Han, Sungsoo Kim, and Sungsu Lim. Improved dynamic coupled graph convolutional recurrent networks for traffic forecasting. In *IEEE International Conference on Big Data, BigData 2023, Sorrento, Italy, December 15-18*, pages 6159–6161, 2023.

Meng Jiang, Taeho Jung, Ryan Karl, and Tong Zhao. Federated dynamic graph neural networks with secure aggregation for video-based distributed surveillance. *ACM Transactions on Intelligent Systems and Technology*, 13(4):56:1–56:23, 2022.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, Conference Track Proceedings*, 2017.

Srijan Kumar, William L Hamilton, Jure Leskovec, and Dan Jurafsky. Community interaction and conflict on the web. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, pages 933–943, 2018.

Brendan McMahan, Eider Moore, Daniel Ramage, et al. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS, 20-22 April, 2017 Fort Lauderdale, FL, USA*, volume 54, pages 1273–1282, 2017.

Shiguan Pang, Yun Xue, Zehao Yan, Weihao Huang, and Jinhui Feng. Dynamic and multichannel graph convolutional networks for aspect-based sentiment analysis. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP, Online Event, August 1-6*, volume ACL/IJCNLP 2021, pages 2627–2636, 2021.

Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao B. Schardl, and Charles E. Leiserson. EvolveGCN: Evolving graph convolutional networks for dynamic graphs. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI, New York, NY, USA, February 7-12*, pages 5363–5370, 2020.

Yang Pei, Renxin Mao, Yang Liu, Chao-Shu Chen, Shifeng Xu, and Feng Qiang. Decentralized federated graph neural networks. In *International Workshop on Federated and Transfer Learning for Data Sparsity and Confidentiality in Conjunction with IJCAI*, 2021.

Mengzhu Sun, Xi Zhang, Jiaqi Zheng, and Guixiang Ma. DDGCN: Dual dynamic graph convolutional networks for rumor detection on social media. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI, February 22 - March 1*, pages 4611–4619, 2022.

Yue Tan, Guodong Long, Lu Liu, Tianyi Zhou, Qinghua Lu, Jing Jiang, and Chengqi Zhang. FedProto: Federated prototype learning across heterogeneous clients. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI, February 22 - March 1*, pages 8432–8440, 2022.

Yue Tan, Yixin Liu, Guodong Long, et al. Federated learning on non-IID graphs via structural knowledge sharing. In *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI, Washington, DC, USA, February 7-14*, pages 9953–9961, 2023.

Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. ArnetMiner: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, August 24-27*, pages 990–998, 2008.

Anshul Thakur, Pulkit Sharma, and David A. Clifton. Dynamic neural graphs based federated reptile for semi-supervised multi-tasking in healthcare applications. *IEEE Journal of Biomedical and Health Informatics*, 26(4):1761–1772, 2022.

Guancheng Wan, Wenke Huang, and Mang Ye. Federated graph learning under domain shift with generalizable prototypes. In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI, February 20-27, Vancouver, Canada*, pages 15429–15437, 2024.

Jiaqi Wang, Yuzhong Chen, Yuhang Wu, Mahashweta Das, Hao Yang, and Fenglong Ma. Rethinking personalized federated learning with clustering-based dynamic graph propagation. In *Advances in Knowledge Discovery and Data Mining. PAKDD 2024*, pages 155–167, 2024.

Wenzhong Wang, Zaipeng Xie, Bingzhe Yu, Zhihao Qu, Yufeng Zhang, and Hongli Cao. Federated learning with common representation learning criterion and personalized predictor. In *IEEE International Conference on Systems, Man, and Cybernetics, SMC 2023, Honolulu, Oahu, HI, USA, October 1-4*, pages 2069–2074, 2023.

Yuhang Yao, Weizhao Jin, Srivatsan Ravi, and Carlee Joe-Wong. FedGCN: Convergence-communication tradeoffs in federated training of graph convolutional networks. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems, New Orleans, LA, USA, December 10 - 16*, 2023.

Zidu Yin and Kun Yue. Temporal resonant graph network for representation learning on dynamic graphs. *Applied Intelligence*, 53(7):7466–7483, 2023.

Le Yu, Leilei Sun, Bowen Du, and Weifeng Lv. Towards better dynamic graph learning: New architecture and unified library. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems, New Orleans, LA, USA, December 10 - 16*, 2023.

Ke Zhang, Carl Yang, Xiaoxiao Li, Lichao Sun, and Siu-Ming Yiu. Subgraph federated learning with missing neighbor generation. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems, December 6-14, virtual*, pages 6671–6682, 2021.

Lei Zhang, Xiaodong Yan, Jianshan He, Ruopeng Li, and Wei Chu. DRGCN: dynamic evolving initial residual for deep graph convolutional networks. In *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Washington, DC, USA, February 7-14*, pages 11254–11261, 2023a.

Mengqi Zhang, Shu Wu, Xueli Yu, Qiang Liu, and Liang Wang. Dynamic graph neural networks for sequential recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 35(5):4741–4753, 2023b.

Taolin Zhang, Chengyuan Mai, Yaomin Chang, Chuan Chen, Lin Shu, and Zibin Zheng. FedEgo: Privacy-preserving personalized federated graph learning with ego-graphs. *ACM Transactions on Knowledge Discovery from Data*, 18(2):40:1–40:27, 2024.