Federated Node Classification over Graphs with Latent Link-type Heterogeneity

Han Xie
Emory University
Atlanta, GA, United States
han.xie@emory.edu

Li Xiong
Emory University
Atlanta, GA, United States
lxiong@emory.edu

Carl Yang
Emory University
Atlanta, GA, United States
j.carlyang@emory.edu

ABSTRACT

Federated learning (FL) aims to train powerful and generalized global models without putting distributed data together, which has been shown effective in various domains of machine learning. The non-IIDness of data across local clients has been a major challenge for FL. In graphs, one specifically important perspective of non-IIDness is manifested in the link-type heterogeneity underlying homogeneous graphs– the seemingly uniform links captured in most real-world networks can carry different levels of homophily or semantics of relations, while the exact sets and distributions of such latent link-types can further differ across local clients. Through our preliminary data analysis, we are motivated to design a new graph FL framework that can simultaneously discover latent link-types and model message-passing w.r.t. the discovered link-types through the collaboration of distributed local clients. Specifically, we propose a framework FedLit that can dynamically detect the latent link-types during FL via an EM-based clustering algorithm and differentiate the message-passing through different types of links via multiple convolution channels. For experiments, we synthesize multiple realistic datasets of graphs with latent heterogeneous link-types from real-world data, and partition them with different levels of link-type heterogeneity. Comprehensive experimental results and in-depth analysis have demonstrated both superior performance and rational behaviors of our proposed techniques.

CCS CONCEPTS

• **Information systems** → **Data mining**; Social networks; **Clustering**; • **Computing methodologies** → *Neural networks*; *Cluster analysis*; **Distributed artificial intelligence**.

KEYWORDS

federated learning, graph mining, link-type heterogeneity, graph neural networks, clustering

ACM Reference Format:
Han Xie, Li Xiong, and Carl Yang. 2023. Federated Node Classification over Graphs with Latent Link-type Heterogeneity. In *Proceedings of the ACM Web Conference 2023 (WWW '23), May 1–5, 2023, Austin, TX, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3543507.3583471

WWW '23, May 1–5, 2023, Austin, TX, USA

ACM ISBN 978-1-4503-9416-1/23/04…$15.00
https://doi.org/10.1145/3543507.3583471

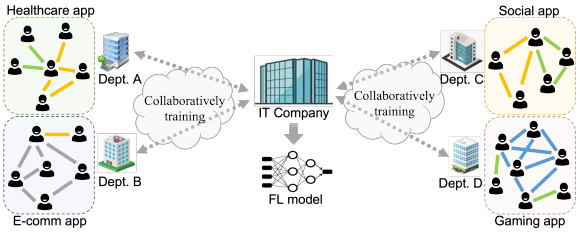


Figure 1: Toy example of an IT company. Links in different colors represent different latent relations. Green: friendship, yellow: kinship, gray: friends with similar purchasing preferences, blue: friends in the game.

1 INTRODUCTION

Federated Learning (FL) is a distributed learning paradigm in which multiple clients collaboratively train a global model without sharing their local data, in consideration of data privacy, commercial competition, resource limitation, and other regulations [5, 36]. FL has been applied to many areas including natural language processing (e.g., automatic text completion [14]) and computer vision (e.g., face recognition [13], medical images [34]). Recently, studies have also been conducted on applying FL to graph representation learning, such as FL on knowledge graphs [9], molecule classification [15], recommender systems [44], and social networks [21]. Different from applying FL on text or image data, graphs do not reside in the Euclidean space which bring a series of unique challenges for FL.

One specific challenge in FL over graph data is the data heterogeneity regarding graph links. Specifically, this can be brought by the types of links in graphs, which can be different and non-independent-identically-distributed (non-IID) across clients. Such link-types in a graph can be explicit or implicit. Explicit link-types are usually represented by a multi-view graph [3] or heterogeneous graph [35]. However, in often cases the real-world homogeneous graphs with a single explicit type of links can have different implicit link-types (relations). For example, users in a social network can be linked due to relations such as classmates, colleagues, friends, and families [47], but these relations are either unknown or private as impossible for the platforms to collect. In the FL setting, such latent link-types can vary a lot across local data owners, which makes learning a global unbiased graph model difficult.

Consider the scenario of FL with link-type heterogeneity based on the example of an IT company (see Figure 1). The data across different departments within the company can be biased due to the diverse functionalities of the departments. For example, departments developing social or healthcare applications are more likely to collect data where the major link-types are friendship or kinship,

which may not appear in the data collected by departments developing e-commerce or gaming applications whose major link-types are friends with similar interests. When the company develops a new service starting with a small number of users and connections or conducts a new analysis for which the most relevant link-types are unknown, it would be beneficial to collaboratively learn a graph model across all departments (clients).

In the centralized learning setting, several prior works have been done to study different link types in a single global graph. Among them, most studies focus on differentiating homophile and heterophile links [8, 10, 17] and inferring the different latent semantics of seemingly homogeneous links [38, 39, 47]. Works in the first category usually propose well-designed frameworks specifically for heterophile networks [52], while some works [10, 17] study the varying extent of homophily in graphs. Works in the second category incorporate auxiliary information such as user attributes and information flows for profiling the relations in graphs [47]. However, as we have illustrated before, data heterogeneity regarding link-types is a unique yet important challenge in FL over graph data, which has not been studied yet. In this work, we consider this unique challenge, which generalizes both the extent of homophily and semantics of relations on different links in the FL setting.

To further motivate the necessity of studying link-type heterogeneity for FL over graph data, we firstly conduct preliminary studies on real-world data to investigate *whether different link-types should be treated differently in both global and federated training scenarios* (see Section 3.3). Our preliminary results illustrate that (1) different link-types do preserve different levels of homophily (Table 2) and (2) the modeling of different link-types differently is beneficial in both global and federated learning scenarios (Table 3).

However, what are the necessary designs of an FL framework to properly differentiate and model the latent link-types across clients' local graphs? We decompose this into two sub-problems: First, *how to treat different link-types differently in the FL setting?* We adopt a multi-channel Graph Convolutional Network (GCN) architecture (mGCN) in which edges with different link-types will be modeled by different message-passing channels; on top of the FedAvg algorithm [27], we design a specialized aggregation algorithm for the split channels of mGCN that can aggregate model parameters of multiple link-type specific channels via a normalized averaging mechanism (Fed-mGCN, see Section 4.1). Second, *how to discover the link-types when they are latent and collaboratively learn link-type-aware models in the FL setting?* Motivated by the Expectation-Maximization (EM) algorithm [11] in k-means clustering [24], we design an edge clustering module that is jointly trained with the message-passing channels and can automatically detect the latent link-types; meanwhile, we design an FL algorithm in which the clustering modules participate in the communication in addition to model parameters to enable the collaboration of link-type-oriented edge clustering among clients. We name the proposed framework as dynamic latent **L**ink-**T**ype aware clustered **Fed**erated graph learning, a.k.a. FedLit.

Since we are the first to study FL with graph data from the novel yet important perspective of latent link-type heterogeneity, we prepare our own datasets by introducing multiple link-types extracted from real-world data of academic networks and electronic health records (EHR), and then distribute the pre-processed datasets into varying numbers of clients via different ways of partitioning the link-types. We conducted extensive experiments on four datasets (DBLP-dm, PubMed-diabetes, NELL, MIMIC3) with three data partitions (distinct, dominant, balanced), in both centralized and FL settings. The results show that basic GCN and Fed-GCN cannot work well with latent link-type heterogeneity, while our framework is able to significantly improve both GCN and Fed-GCN. Furthermore, we conduct in-depth analysis from the perspectives of clustering effectiveness, pre-defined number of link-types, and clients' behaviors during FL.

Our work contributes in the innovation of both problem formulation and technique development:

- This is the first work to study the latent link-type heterogeneity problem in the setting of FL on graphs, including latent link-type heterogeneity across links, and across clients.
- We design a dynamic latent **L**ink-**T**ype aware clustered **Fed**erated graph learning framework (FedLit) that can automatically and collaboratively detect the latent link-types underlying graphs and perform link-type-aware collaboration across clients.
- We build novel datasets with synthesized link-types from real-world data and conduct comprehensive experimental studies to demonstrate the superiority of our proposed framework and techniques.

2 RELATED WORKS

2.1 Federated Learning over Graph Data

The confluence of the rapidly developing research on federated learning (FL) and emerging advanced graph neural networks (GNNs) promotes increased recent studies on FL over graph data. Prior works can be categorized into graph federated learning [2, 6, 7, 21, 29, 31] and federated graph learning [12, 15, 22, 23, 41–43, 48]. Works in the former category model the FL architecture (the relations between servers and local devices, or among devices) by graphs and exploits graph models such as GNNs to facilitate FL, which has the focus on the FL setting. Works in the latter category, including our work, focus on the unique challenges brought by graphs in the FL setting. Previous works of federated graph learning involve various topics, such as [28, 44, 49] which focus on the privacy issue; [5, 22, 50] which study the issue of isolated data island (cross-graph) and missing links; [4, 15, 37, 41, 45, 51] which study the data heterogeneity (non-IID) problems w.r.t. graph structures and node attributes; and more application-oriented FL such as recommendation [44], knowledge graphs [9], molecular graphs [16, 55], and financial crimes detection [36].

Although some existing works have studied the data heterogeneity (non-IID) problems w.r.t. graphs in the FL setting, the graph-specific non-IIDness across clients in FL is in fact rarely explored. [4] studies the heterogeneity and complementarity of graphs between clients for a global self-supervised FL framework, but it only considers the non-IIDness w.r.t. the numbers of nodes, edges, and labels, which are the sample-wise statistics of graphs and unrelated to the graph topology. [45] studies the graph-level tasks across datasets and domains, in which the data heterogeneity w.r.t. graphs is at the graph level and overlooks the local neighborhood information. [15] and [16] simulate the data non-IIDness by distributing

graphs from the same dataset to multiple clients using a Dirichlet distribution, where the non-IIDness is at the graph level and only based on sample size (number of graphs). [41] leverages the model-agnostic meta-learning (MAML) method to address the graph-level non-IIDness and inconsistency of label domains, and [37] decouples the learning of graph features and topology in FL to focus on the sharable structural information among clients; neither of them focuses on the graph heterogeneity regarding links.

2.2 Relation Learning without FL

As most GNNs have the assumption that similar nodes are more likely to be connected (homophily), they usually have difficulty in learning on heterophile graphs in which opposite nodes tend to connect. Although recently some works start to question whether the homophily is indeed necessary for GNNs [25, 26, 46], enormous research on heterophile graphs or graphs with varying heterophily is emerging. Regarding the technical designs, these works can be categorized [52] into extension of neighborhoods [17, 18, 30] and dedicated designs of GNN architectures [8, 10, 53]. Beyond these specifically designed GNN models for heterophile graphs, [10] trains the GNN with a Generalized PageRank algorithm regardless of the extent of homophily, and [17] claims that the real-world graphs are the composition of graphs with complete homophily/heterophily/randomness, and then chooses different hops of neighbors for aggregation w.r.t. different types of graphs.

Apart from the homophily and heterophily of graphs, edges in real-world graphs can carry abundant semantic information, which can be regarded as relations. Specifically, relational learning can be applied to social networks for modeling real-world social connections. For example, [47] learns the latent relations in social networks by leveraging the multi-modal semantic information, [38] incorporates relation learning to extract the latent social dimensions from the multi-dimensional connection in the social networks. [39] proposes a context-aware node embedding method to model the semantic relations between nodes. In our work, we include the semantic relations and the homophily/heterophily of graphs into a generalized concept named the latent link-types of graphs, and pioneer to study the unique challenges brought by the non-IID link-types across clients in the FL setting.

3 PROBLEM FORMULATION

3.1 Notations

We summarize all the notations that are used in this work in Table 1.

Table 1: Notations.

Notation	Representation
G	A homogeneous graph.
$V; u, v$	The set of nodes; u and v are nodes.
$E; e$	The set of edges; e is an edge.
$\mathbf{X}; \mathbf{y}$	Node attributes; node labels.
$\mathbf{h}; \mathbf{z}$	Node embedding; edge embedding.
$N; n$	The number of clients; the index of clients.
π	The number of oracle link-types.
k	The pre-set number of clusters (latent link-types).
$\mathbb{C}; c$	The set of link-types; c is a link-type.
$\mathbb{C}^{\mathrm{k}}$	The union set of link-types from all clients.
φ^c	The centroid in the edge embedding space of link-type c.
$\tilde{\varphi}^c$	The center of centroids $\{\varphi_n^c\}_{n\in N}$.
$\Theta; \theta^{\mathrm{s}}, \theta^c, \theta^{\mathrm{t}}$	The set of model parameters; parameters of different mGCN modules.
$r_{\mathrm{local}}; r$	The local training epoch; the index of communication rounds.

3.2 Preliminaries

3.2.1 Graph convolutional network. Graph convolutional network (GCN) [20] is a widely used type of GNN that learns the representations of nodes by iteratively aggregating the information propagated through the neighborhoods. Mathematically, given a graph $G = (V, E, \mathbf{X})$ with nodes V, edges E, and node attributes $\mathbf{X}$, an l-layer GCN can be formulated (in vector form) as

$$\mathbf{h}_u^{(l+1)} = \sigma\Big(\sum_{v\in\mathcal{N}_u} \frac{1}{\alpha_{uv}}\theta^{(l)}\mathbf{h}_v^{(l)}\Big), \tag{1}$$

where $\mathbf{h}_u^{(l+1)}$ is the representation of node u at layer $l+1$, v is a node from u's neighbors $\mathcal{N}_u$, and θ is the learnable parameters of a GCN. The normalization constant α_{uv} for an edge e_{uv} is from the symmetrically normalized adjacency matrix $\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$, where $\tilde{A} = A + I$ is the summation of the adjacency matrix and the identity matrix, and $\tilde{D}$ is the diagonal node degree matrix of the matrix $\tilde{A}$.

3.2.2 Federated averaging graph convolutional network (Fed-GCN). FedAvg [27] is the most basic FL algorithm, which is based on stochastic gradient descent and aggregates the models uploaded by clients through normalized averaging. The vanilla FedAvg algorithm is applied to grid-structured data. Considering our setting of FL on graphs, we term the algorithm of training GCNs with FedAvg as Fed-GCN. Usually, FedAvg normalizes the aggregation by the sample sizes of clients. In Fed-GCN, when applying the node-level downstream task, e.g., node classification, we formulate the aggregation of Fed-GCN as

$$\theta^{(r+1)} = \sum_{n=1}^{|N|} \frac{|V_n|}{|V|}\theta_n^{(r)}, \tag{2}$$

where $\theta^{(r+1)}$ is the aggregated model at the $(r+1)^{\mathrm{th}}$ communication round. N represents the set of clients, V_n is the node set of a graph on client n, and θ_n is the local uploaded model from client n. The aggregated model $\theta^{(r+1)}$ is then broadcast back to all clients.

3.3 Motivating Data Analysis

Given a graph $G = (V, E, \mathbf{X}, \mathbf{y})$ with π known link-types, where $\mathbf{y}$ represents node labels, we would like to investigate whether it is necessary and beneficial to treat different link-types separately. We utilize two metrics to quantitatively evaluate the link-types, and conduct preliminary experiments to compare the performance of node classification on datasets with different link-types, when using vanilla GCNs and multi-channel GCNs which differentiate the message passing through different link-types by channels in both centralized and FL settings.

We quantify various link-types from two perspectives, Edge Homophily and Attribute Homophily. Edge Homophily (EH) [54] measures the edge-level homophily ratio w.r.t. node labels, which is the fraction of edges in the whole graph that connect nodes belonging to the same class, formulated as below

$$EH = \frac{|e_{uv} \in E \wedge y_u = y_v|}{|E|}. \tag{3}$$

Edges with different link-types could have a particular tendency of connecting nodes with the same labels or different labels as

studied in [17]. Thus, if EH differs w.r.t. the generation rules underlying each link-type, it indicates that these links should be modeled separately. Beyond labels, we design another metric to evaluate the correlation between link-types and node attributes, termed Attribute Homophily (AH). AH measures the mean homophily on attributes of all node pairs in a graph, formulated as

$$AH = \frac{\sum_{e_{uv} \in E} \mathrm{S}_{\cos}(\mathbf{x}_u, \mathbf{x}_v)}{|E|}, \tag{4}$$

where $\mathrm{S}_{\cos}$ is the cosine similarity. Edges with different link-types may particularly focus on certain attributes, and connect nodes with strong correlations between these attributes. Thus, if AH differs w.r.t. the link-types, it also indicates the necessity to model the link-types separately.

We extract four potential relations as oracle link-types between papers (nodes)[1] from a publication dataset DBLP-DM (see Table 2). Besides the *reference*, *share authors*, and *share keywords*, the *same year* link-type is generated on purpose for introducing less-relevant edges, which are common in real scenarios. By selecting the common nodes shared by all link-types, we construct four subgraphs, each of which has one distinct link-type, and one mixed subgraph that contains all edges of four link-types, all subgraphs with *research topics* as node labels. According to Table 2, the characteristics of these link-types are indeed different. The *reference* link-type which should reflect the most direct relevance between papers shows the highest AH and EH, and the *same year* link-type which involves less-relevant connections between papers shows the least AH and EH. The *share authors* and *share keywords* link-types are in-between. Furthermore, we want to study how such different characteristics of link-types as reflected by AH and EH scores will affect the behaviors and utilities of graph learning models.

Table 2: Edge Homophily and Attribute Homophily on subgraphs of DBLP-DM with distinct or mixed link-types.

Link-type	*reference*	*share authors*	*share keywords*	*same year*	mixed
EH	0.2692	0.1847	0.2338	0.1150	0.1773
AH	0.2846	0.2394	0.2422	0.2044	0.2267

To experimentally validate the assumption that link-types can impact the performance of graph learning models, we apply vanilla GCNs [20], multi-channel GCNs (mGCNs) in which each channel is for the message passing through a link-type, and clustered multi-channel GCN (cGCN) which incorporates the clustering module of FedLit and can dynamically cluster edges (details in Section 4) on the five subgraphs. From Table 3, the performance of GCNs varies on subgraphs with different link-types. When EH and AH decrease, the power of GCN is impaired. Especially, for the subgraph with *same year* edges, the performance of all models downgrades greatly. It is also noticed that GCN cannot work well on the mixed graph with multiple link-types; however, mGCN can obviously improve from GCN by separating the message passing for different link-types. Additionally, cGCN can approach mGCN on the mixed graph without knowing the oracle link-types, and outperform mGCN on graphs with single link-types due to its ability to automatically uncover potential link-types in a data-driven fashion.

[1]The details of the link-types are discussed in Appendix A.1.

In the simplified illustrative simulation of the FL setting, each client owns a subgraph with a single link-type, and the collaboratively trained models are evaluated on the graph with mixed link-types. In the FL rows of Table 3, Fed-mGCN improves Fed-GCN by a large margin due to the separate handling of known different link-types. FedLit (Fed-cGCN), without knowing the oracle link-types, can still outperform Fed-GCN.

Table 3: Centralized and FL performance (accuracy) w.r.t. subgraphs of DBLP-DM with distinct or mixed link-types.

Link-type	reference	share authors	share keywords	same year	mixed
GCN	0.4066	0.3209	0.4159	0.1862	0.3403
mGCN	0.4025	0.3202	0.4155	0.1828	**0.4017**
cGCN	0.4177	0.3637	0.4315	0.2904	0.3797
Fed-GCN	0.3280				N/A
Fed-mGCN	**0.4125**				N/A
FedLit	0.3641				N/A

3.4 Problem Setup

In an FL system with a server and N local clients, each client n holds a graph $G_n = (V_n, E_n, \mathbf{X}_n, \mathbf{y}_n)$. Suppose there are π link-types in total underlying all clients' graphs $\{G_n\}_{n=1}^N$, the set of link-types of E_n on client n, denoted as $\mathbb{C}_n$, can have its size $|\mathbb{C}_n| \leq \pi$. An edge $e_{uv}^c \in E_n$ represents the edge between node u and v of graph G_n with the link-type $c \in \mathbb{C}_n$.

On each graph G_n, the downstream task is node classification which predicts the label $\hat{y}_u$ of node u using the function $f(\Theta_n; G_n) : \mathbf{X}_n \to \hat{\mathbf{y}}_n$, and Θ_n is the learnable parameters of the function $f(\cdot)$. A client n tries to find the optimized Θ_n^* by minimizing the local empirical risk

$$\mathcal{R}_n(\Theta_n; G_n) \coloneqq \ell(f(\Theta_n; G_n), \mathbf{y}_n) \tag{5}$$

during its local training, where $\ell(\cdot)$ is the loss function such as cross entropy. The goal of the federated node classification framework over graphs is then to find Θ^* by

$$\Theta^* = \arg\min \mathcal{R}(\Theta; \{G_n\}_{n=1}^N), \tag{6}$$

$$\mathcal{R}(\Theta; \{G_n\}_{n=1}^N) \coloneqq \mathbb{E}_{n \in N}[\mathcal{R}_n(\Theta_n; G_n)]. \tag{7}$$

4 PROPOSED FRAMEWORK: FEDLIT

This work aims to resolve the latent link-type heterogeneity problems in FL on graphs, where clients can hold graphs with different latent link-types, or with the same latent link-types falling in different distributions. To approach it, we propose a dynamic latent **L**ink-**T**ype-aware clustered **Fed**erated graph learning framework (FedLit) that can automatically detect the link-types by dynamically clustering the edges, and perform local link-type-wise information propagation and global link-type-wise collaborative training.

Figure 2 shows an overview of our proposed framework FedLit. It consists of two main parts: I. the FL algorithm for aggregating multi-channel GCNs for different link-types (Section 4.1); and II. the clustering mechanism for dynamically detecting the link-types of edges and clustering edges correspondingly, and the FL algorithm that is adapted to the collaboration among clients with the clustering mechanisms (Section 4.2).

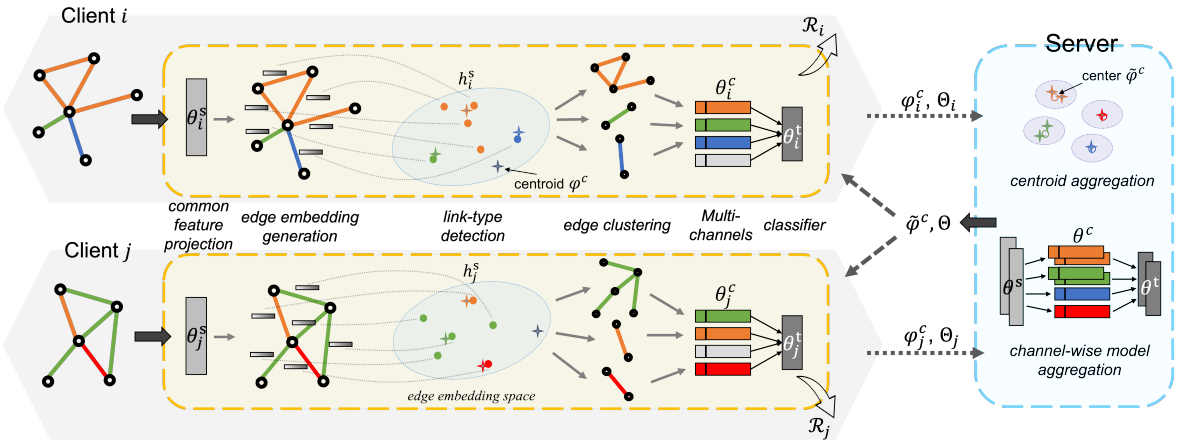


Figure 2: Overview of FedLit: On each client, a local model with the edge embedding and clustering modules is iteratively trained based on its local graph and communicated across all clients through the server to achieve federated training.

4.1 Fed-mGCN

Multi-channel GCN (mGCN). In order to distinguish the information propagation along different link-types, we employ multi-channel GCN (mGCN) at the local and global levels. In this work, we implement mGCN as R-GCNs [33], but other mGCNs such as [40] can also be adapted. Specifically, an mGCN preserves a common feature projection layer shared by all link-types, multiple link-type-specific graph convolution channels, and a task-specific classifier.

The common feature projection layer with parameters $\theta^{\mathrm{s}} = [\vartheta^{\mathrm{s}}_0, \vartheta^{\mathrm{s}}]$ takes the original node features of a graph G as the input, and outputs the node embedding for a node u by

$$\mathbf{h}^{\mathrm{s}}_u = \sigma(\vartheta^{\mathrm{s}}_{0,u} + \sum_{u \in V} \vartheta^{\mathrm{s}}_u \mathbf{x}_u). \tag{8}$$

On top of the common feature projection layer, the network is split into k (the pre-defined number of link-types) channels and each is responsible for a link-type c with parameters denoted as $\theta^c = [\vartheta^c_0, \vartheta^c; \vartheta^{c,(\cdot)}]$, where $[\vartheta^c_0, \vartheta^c]$ are the parameters of a link-type-specific feature projection layer, and $\vartheta^{c,(\cdot)}$ represent multiple graph convolution layers. The link-type-specific feature projection layer takes $\mathbf{h}^{\mathrm{s}}_u$ as the input for a node u who has connected edges of link-type c, and outputs

$$\mathbf{h}^c_u = \sigma(\vartheta^c_{0,u} + \sum_{u \in V} \vartheta^c_u \mathbf{h}^{\mathrm{s}}_u). \tag{9}$$

The embedding $\mathbf{h}^c_u$ is then used for l layers of graph convolution in the channel corresponding to link-type c, and the output will be

$$\mathbf{h}^{c,(l+1)}_u = \sigma(\sum_{v \in \mathcal{N}_{u,e^c_{uv}}} \frac{1}{\alpha_{uv}} \vartheta^{c,(l)} \mathbf{h}^{c,(l)}_v), \tag{10}$$

where $\mathbf{h}^{c,(0)}_v = \mathbf{h}^c_v$, and v is a node from u's neighbors $\mathcal{N}_{u,e^c_{uv}}$ connected by an edge of link-type c. α_{uv} is a normalization constant for the edge e_{uv}, which is computed from the symmetrically normalized adjacency matrix of graph G.

Since a node u can be connected by multiple edges of different link-types, the embedding of u is aggregated by its output embeddings generated from different channels, as long as there exists an edge of link-type c connecting node u,

$$\tilde{\mathbf{h}}^c_u = \underset{c \in \mathbb{C}, \exists e^c_{u\cdot} \in E}{\mathrm{agg}} (\mathbf{h}^{c,(l+1)}_u), \tag{11}$$

where $\mathrm{agg}(\cdot)$ can be either summation or averaging.

The last layer of an mGCN is a task-specific classifier that trains parameter $\theta^{\mathrm{t}} = [\vartheta^{\mathrm{t}}_0, \vartheta^{\mathrm{t}}]$ with a supervised downstream task such as node classification. The final prediction of node u is calculated as

$$\mathbf{h}^{\mathrm{t}}_u = \sigma(\vartheta^{\mathrm{t}}_0 + \sum_{u \in V} \vartheta^{\mathrm{t}}_u \tilde{\mathbf{h}}^c_u), \tag{12}$$

$$\hat{y}_u = \mathrm{softmax}(\mathbf{h}^{\mathrm{t}}_u). \tag{13}$$

We calculate the empirical risk using cross-entropy loss as

$$\mathcal{R}(\Theta; G) = -\sum_{u \in V} y_u \log \hat{y}_u, \tag{14}$$

where Θ is the set of parameters θ^{s}, θ^c, and θ^{t} from common feature projection, link-type-specific channel, and task-specific classifier modules, respectively.

Federated learning of mGCNs. In the FL setting, local clients train the mGCN models on their local graphs for r_{local} epochs, and then upload their models Θ_n to the server for communication. Different from federated learning of vanilla GCNs, the FL of mGCNs (Fed-mGCN) requires a dedicated design for model aggregation. Regarding the common feature projection and task-specific classifier modules, Fed-mGCN exploits the FedAvg algorithm [27] and aggregates parameters $\{\theta^{\mathrm{s}}_n\}_{n \in N}$ and $\{\theta^{\mathrm{t}}_n\}_{n \in N}$ of all clients' models using normalized averaging, respectively, by

$$\theta^{\mathrm{s},(r+1)} = \sum_{n=1}^{|N|} \frac{|V_n|}{|V|} \theta^{\mathrm{s},(r)}_n, \ \theta^{\mathrm{t},(r+1)} = \sum_{n=1}^{|N|} \frac{|V_n|}{|V|} \theta^{\mathrm{t},(r)}_n, \tag{15}$$

where $|V|$ is the total number of node samples of all clients.

However, for separate channels in mGCNs, the aggregation is in a manner that only channels learning information propagation with the same link-type will be aggregated together. Thus, the channels of mGCNs w.r.t. link-type c from all clients are aggregated by

$$\theta^{c,(r+1)} = \sum_{n=1}^{|\{n\}_{\exists c \in \mathbb{C}_n}|} \frac{|V^c_n|}{|V^c|} \theta^{c,(r)}_n, \tag{16}$$

where $\{n\}_{\exists c \in \mathbb{C}_n}$ is the set of clients that link-type c exists in their graphs, and V_n^c is the set of nodes with connected edges of c. If $|\mathbb{C}_n| < k$, indicating client n lacks certain link-types, only the channels w.r.t. link-type $c \in \mathbb{C}_n$ of client n's model will contribute to the global model. Besides, the normalized aggregation allows clients with more edges of link-type c to dominate the contribution of learning the optimal corresponding channels of a global model.

4.2 FedLit: a dynamic latent link-type-aware clustered FL framework

The dynamic link-type-aware clustering. Fed-mGCN can realize the link-type-wise information propagation and FL w.r.t. explicit link-types. However, regarding implicit link-types, how could we detect the latent link-types of edges for link-type-specific channel assignments? Conceptually, the detecting and clustering can occur either offline or online. The offline manner would finish the link-type detecting and edge clustering before transmitting the data into a GNN model for a downstream task. However, this way is practically infeasible since the link-types are implicit underlying graphs, and what attribute and structure information is essential for clustering edges to promote the performance of a certain task is hard to know. Hereby, we propose an online clustering method along with the model training for a downstream task, which can dynamically detect the latent link-types of edges and cluster edges.

Specifically, we represent edges by concatenating their two-end nodes' embeddings $\mathbf{h}^{\mathrm{s}}$ (Equation 8) from the common feature projection module of an mGCN, $\mathbf{z}_{uv} = \mathbf{h}_u^{\mathrm{s}} \oplus \mathbf{h}_v^{\mathrm{s}}$.

The edge clustering is motivated by the k-means clustering [24], which employs the expectation–maximization (EM) algorithm [11] and iteratively finds the centroids and minimizes the within-cluster distances. Given a pre-defined number of link-types k, the initial k centroids $\{\varphi^c\}_{c=1}^k$ are selected on the edge embedding space $\mathcal{Z}$ using k-means++ initialization [1]. An edge e_{uv} is then assigned to the cluster of link-type c if the centroid φ^c is closet to its embedding,

$$\Gamma(e_{uv}) \rightarrow c : \arg\min_c \mathrm{S}_{\cos}(\mathbf{z}_{uv}, \varphi^c), \tag{17}$$

where Γ is the cluster assignment (M step). Within each cluster, the centroid φ^c can be updated by an E step,

$$\varphi^c = \frac{1}{|\{e_{uv}^c\}|} \sum_{e_{uv}^c} \mathbf{z}_{uv}. \tag{18}$$

Between communication rounds, the E-M steps can be run multiple times per local training epoch or once every few local epochs.

Federated learning with dynamic clustering. Once a centroid φ^c is initialized, it fixes its mapping to a channel of the local model. Edges clustered as link-type c will be transmitted into the channel associated with φ^c for propagating information to their connected nodes (Equation 9-11), and the local model training is supervised by a task (Equation 12-14). As the centroids reside in the edge embedding space which is generated from the task-supervised model training, the link-type detection and edge clustering procedures are aware of the downstream task need.

A local client n preserves a local model and k centroids $\{\varphi_n^c\}_{c \in \mathbb{C}^{\mathrm{k}}}$. During the communication of FL, client n transmits its model parameters Θ_n and the set of updated centroids $\{\varphi_n^c\}_{c \in \mathbb{C}_n}$ to the server for collaborative learning.

On the server side, the server first separates its received $N * k$ centroids into k groups $\{\phi_1, \phi_2, \ldots, \phi_k\}$, each with N centroids from distinct clients, so that a group ϕ_c is corresponding to a link-type c. The objective is to find a grouping that minimizes the within-group summation of distances,

$$\phi : \arg\min_{\phi} \sum_{i=1}^{k} \sum_{\varphi \in \phi_c} \mathrm{S}_{\cos}(\varphi, \tilde{\varphi}_c), \tag{19}$$

with the constraint that all φ in ϕ_c should be from different clients. $\tilde{\varphi}_c$ is the center of group ϕ_c,

$$\tilde{\varphi}_c = \frac{1}{|\phi_c|} \sum_{\varphi \in \phi_c} \varphi. \tag{20}$$

We employ a heuristic approach to address the grouping problem. Except for the first communication round in which the server randomly selects a centroid from each client to form a group ϕ and computes the center $\tilde{\varphi}$, in each successive communication round, we compute the similarity between elements in $\{\varphi\}^{N*k}$ and those in $\{\tilde{\varphi}\}^k$. The client's centroid φ_n with the highest similarity to a center $\tilde{\varphi}_c$ will be assigned to the group ϕ_c first, after which all other centroids from client n cannot be assigned to ϕ_c. We then find the next most similar centroid-center pair excluding all pairs of $\tilde{\varphi}_c$ and elements in $\{\varphi_n\}$, and iterate until all centroids are grouped.

With the grouping $\{\phi_c\}_{c=1}^k$, the local centroids φ_n^c of client n belonging to the group ϕ_c will be updated by $\tilde{\varphi}^c$ in a communication. Meanwhile, as the grouping maintains the alignments between local centroids and global centers, the correspondence among channels of local models, and between them and channels of the global model in the server is also known. Therefore, the aggregation of channels of local models follows the grouping $\{\phi_c\}_{c=1}^k$ where channels corresponding to $\varphi \in \phi_c$ will be aggregated and updated using Equation 16. The aggregation of common feature projection and task-specific classifier modules is the same as Equation 15.

After aggregation, the server broadcasts the updated model and centers to clients following the grouping. Herewith, the local cluster assignments are not only co-trained with its local model for downstream tasks, but also involved in the communication of FL for collaborating with other clients to discover consistent edge clusters and latent link-types globally.

4.3 Discussion on FedLit

Different from traditional FL which can be categorized into horizontal and vertical according to (Euclidean) data partitioning, FL on graphs can be more complicated due to the introduction of graph topology. We provide more discussion on horizontal v.s. vertical FL w.r.t. our specific scenario in Appendix B.1. As we focus on the utilities of FL on graphs, the privacy and efficiency problems are not fully explored in this work. Related discussions on the limitation of this work w.r.t. privacy and efficiency, as well as w.r.t. incorporating the EM-based clustering, are included in Appendix B.2.

5 EXPERIMENTS

5.1 Experimental Settings

Data Since this is the first work to study latent link-types of graphs in the FL setting, we pre-process four raw datasets and construct

Table 4: Accuracy on publication and medical datasets with three data partitions. Bold represents the best results in centralized and FL settings, respectively. Grey background represents the best results among all methods excluding mGCN and Fed-mGCN.

Dataset	DBLP-DM			PUBMED-DIABETES			NELL			MIMIC3		
GCN	0.3378 (±0.0022)			0.8468 (±0.0024)			0.3758 (±0.0210)			0.3603 (±0.0021)		
cGCN [$k = \pi$]	0.3712 (±0.0042)			0.8781 (±0.0055)			0.4983 (±0.0198)			0.3655 (±0.0039)		
cGCN [$k > \pi$]	0.3884 (±0.0048)			**0.8795** (±0.0028)			0.5024 (±0.0069)			0.3715 (±0.0030)		
mGCN	**0.4148** (±0.0026)			0.8778 (±0.0023)			**0.5162** (±0.0129)			**0.3973** (±0.0028)		
	distinct	dominant	balanced	distinct	dominant	balanced	distinct	dominant	balanced	distinct	dominant	balanced
Fed-GCN	0.3033 (±0.0028)	0.3088 (±0.0025)	0.3226 (±0.0036)	0.7776 (±0.0046)	0.8008 (±0.0043)	0.8284 (±0.0029)	0.2345 (±0.0184)	0.2260 (±0.0016)	0.3491 (±0.0107)	0.3399 (±0.0055)	0.3388 (±0.0026)	0.3584 (±0.0032)
Fed-mGCN	**0.3930** (±0.0016)	**0.3779** (±0.0050)	**0.4046** (±0.0015)	0.8758 (±0.0025)	0.8657 (±0.0058)	0.8745 (±0.0022)	0.4447 (±0.0178)	0.4050 (±0.0055)	**0.5178** (±0.0065)	**0.3830** (±0.0015)	0.3474 (±0.0020)	0.3762 (±0.0011)
FEDLIT [$k = \pi$]	0.3238 (±0.0049)	0.3371 (±0.0066)	0.3400 (±0.0083)	**0.8776** (±0.0044)	0.8779 (±0.0028)	0.8771 (±0.0070)	0.4158 (±0.0071)	0.3970 (±0.0067)	0.4750 (±0.0152)	0.3552 (±0.0017)	0.3541 (±0.0037)	0.3750 (±0.0048)
FEDLIT [$k > \pi$]	0.3533 (±0.0040)	0.3701 (±0.0080)	0.3669 (±0.0062)	0.8759 (±0.0092)	**0.8820** (±0.0047)	**0.8811** (±0.0023)	**0.4715** (±0.0111)	**0.4243** (±0.0253)	0.5091 (±0.0060)	0.3685 (±0.0018)	**0.3593** (±0.0038)	**0.3765** (±0.0046)

four graphs with multiple oracle link-types, including two publication datasets, DBLP-DM[2] and PUBMED-DIABETES[3], and two electronic health record (EHR) datasets, NELL[4] and MIMIC3[5]. For each raw dataset, we sample data, generate node features and labels, and construct links using different link generation rules. The oracle link-types and statistics of constructed graphs are in Appendix A.

Data partitioning. We consider three ways of data partitioning for a graph with multiple link-types, in which each simulates a different extent of link-type heterogeneity across clients. a) Distinct. It simulates the situation in FL where clients preserve graphs with different link-types, and there exists link-type heterogeneity w.r.t. the set of link-types across clients. We extremize the situation by making a client hold only one link-type. Each client randomly chooses a link-type and receives a subgraph with the link-type sampled from the full graph. b) Dominant. It simulates a more natural situation in FL where clients preserve graphs with all link-types but have a dominant link-type of their interests. In this situation, there exists the link-type heterogeneity w.r.t. link-type distribution across clients. Each client receives a subgraph sampled from the full graph which contains most edges of the randomly-selected dominant link-type and smaller portions of edges of other link-types. c) Balanced. It simulates an unnatural situation in FL where clients preserve graphs with minimum link-type heterogeneity. In this situation, the sampled graphs on clients have the same distribution of edges with different link-types.

Compared baselines. We design baselines in the centralized setting, including a) GCN which trains a vanilla GCN [20] with feature projection layers and a classifier, b) mGCN which trains an mGCN on a graph with oracle link-types, and c) cGCN which adopts the clustering mechanism of FEDLIT into an mGCN and trains it on a graph without access to its oracle link-types; and in the FL setting, including d) Fed-GCN which aggregates the local GCNs by the FedAvg algorithm, and e) Fed-mGCN which aggregates local mGCNs in a channel-wise manner using oracle link-types.

Hyper-parameter settings. The architecture of all GNN models includes two feature projection layers, two graph convolutional layers, and a classifier layer. We use Adam [19] optimizer with a learning rate of 0.01 for the publication datasets, and with a learning rate of 0.001 for the EHR datasets. The numbers of training epochs and communication rounds are 100. In FL settings, the number of clients is 10, and the local training epoch r_{local} is set as 1. We perform a 10-fold cross-validation for each experiment. All experiments are run on a server with eight 48GB NVIDIA Quadro RTX 8000 GPUs. All code and data are provided in this GitHub repository[6].

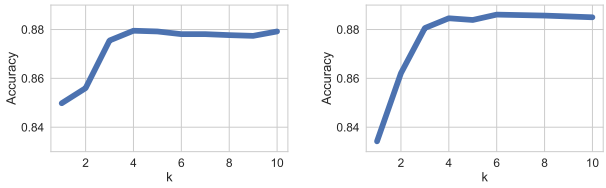
5.2 Experimental Results at Global Level

Table 4 displays the comprehensive results of evaluating all methods on a global graph in the centralized and FL settings. In general, FEDLIT can outperform all FL baselines in all cases except for Fed-mGCN, due to its access to oracle link-types. In the centralized setting, cGCN leveraging our designed clustering mechanism can get very close to mGCN which has the access to oracle link-types. In the FL setting, our framework FEDLIT achieves comparable results to the corresponding centralized cGCN, and consistently beats the Fed-GCN model. Conceptually, the baseline Fed-mGCN should perform best in the FL setting. However, the oracle link-types are not always the optimal ground truth (the links can still be heterogeneous even with the same oracle types), and the intentionally introduced noisy edges with the link-type that is less relevant to the tasks may deteriorate the rigid framework of mGCN. Thus, it can be observed from the table that half of the results of FEDLIT actually surpass the results of Fed-mGCN.

Although the "oracles" of link-types may not be ideal, we still use them as the ground-truth link-types and evaluate our designed clustering module with the number of channels equal to ($k = \pi$) or greater than ($k > \pi$) the number of oracle link-types π. It is observed that the results of cGCN and FEDLIT with $[k > \pi]$ are always better than the results with $[k = \pi]$, except for the distinct setting on PUBMED-DIABETES where FEDLIT with $[k = \pi]$ slightly outperforms that with $[k > \pi]$. This observation suggests that simply setting the number of clusters k to be larger than the real number of link-types can often guarantee satisfactory performance, probably due to a margin provided for tolerating inaccurate clustering.

Regarding the different data partitioning in the FL setting, Fed-GCN often performs better in the balanced setting than in the distinct and dominant settings, which is natural because the balanced setting is closest to the IID case which is easiest for the vanilla FedAvg algorithm in Fed-GCN. Meanwhile, it further proves that FL with vanilla GCNs would fail when facing significant link-type heterogeneity. However, with mGCNs, i.e., Fed-mGCN and FEDLIT,

[2]DBLP-DM: https://dblp.uni-trier.de/
[3]PUBMED-DIABETES: https://linqs.org/datasets/#pubmed-diabetes
[4]NELL: https://www.nursing.emory.edu/pages/project-nell
[5]MIMIC3: https://physionet.org/content/mimiciii-demo/1.4/
[6]https://github.com/Oxfordblue7/FedLIT



(a) cGCN (b) FedLit

Figure 3: Hyper-parameter analysis on PubMed-diabetes.

the distinct and dominant settings have the chance to surpass the balanced setting, which indicates that our proposed frameworks are able to address the link-type heterogeneity problem effectively. Additionally, the larger the extent of link-type heterogeneity is (i.e., distinct > dominant > balanced), the effects of FedLit are more compelling, as demonstrated by the larger gains over Fed-GCN.

5.3 Experimental Results at Local Level

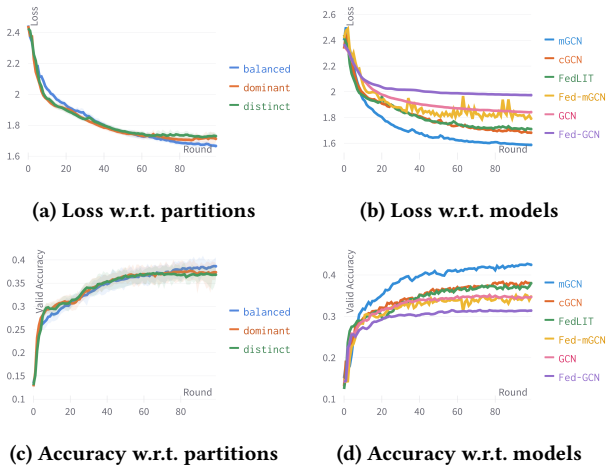
As noisy edges with link-type less relevant to the downstream tasks are introduced into the graphs, it is motivating to investigate whether the clients with the noisiest edges would benefit from FL using our framework. Table 5 displays local results of three clients (i.e., n7, n8, n9) with the same link-type (i.e., *same year*). Overall, FedLit can significantly improve the performance of local clients compared to training GCNs on the clients themselves, by a large margin of 13% in distinct and 8% in dominant settings. However, Fed-GCN has no effect on helping the lagging clients. Fed-mGCN can improve the local clients in the dominant setting because of channel-wise collaboration. While in the distinct setting, the performance of Fed-mGCN is similar to the local training because channels without input edges will not participate in the collaboration. The experimental results indicate that FedLit can improve the lagging clients with less-relevant information to the task.

Table 5: Accuracy on local clients with subgraphs of DBLP-dm from two data partitions. Bold represents the best results and grey background indicates results from our framework.

Partition	distinct			dominant		
Client	n7	n8	n9	n7	n8	n9
GCN	0.1828 (±0.0023)	0.1856 (±0.0050)	0.1862 (±0.0023)	0.2745 (±0.0029)	0.2708 (±0.0027)	0.2759 (±0.0022)
Fed-GCN	0.1771 (±0.0013)	0.1770 (±0.0003)	0.1801 (±0.0011)	0.2721 (±0.0014)	0.2746 (±0.0015)	0.2784 (±0.0016)
Fed-mGCN	0.1839 (±0.0029)	0.1842 (±0.0042)	0.1850 (±0.0035)	0.3462 (±0.0046)	0.3392 (±0.0031)	0.3408 (±0.0018)
FedLit	**0.3126** (±0.0072)	**0.3115** (±0.0057)	**0.3170** (±0.0054)	**0.3579** (±0.0041)	**0.3590** (±0.0051)	**0.3565** (±0.0031)

6 IN-DEPTH ANALYSES

Hyper-parameter analysis The number of link-type k should be pre-defined as a hyper-parameter and can be tuned to achieve better performance. To study the effect of k and find the rule-of-thumb for choosing a reasonable k, we analyze the performance of cGCN and FedLit with four oracle link-types when varying k. As shown in Figure 3, increasing k till a certain value will benefit the performance of cGCN and FedLit, but continuing increasing k will not further improve the performance as the performance will converge. The reason can be that when k is larger than the actual need of the number of channels, multiple channels would be used to model



(a) Loss w.r.t. partitions (b) Loss w.r.t. models

(c) Accuracy w.r.t. partitions (d) Accuracy w.r.t. models

Figure 4: Convergence analysis on DBLP-dm

one link-type, which should neither facilitate nor deteriorate the performance, but requires more time and computational resources. With the observation, we can conclude the rules for choosing a reasonable k as: 1) choosing the exact best k is not necessary since the performance will converge as k grows large; 2) one can choose a relatively larger k that is affordable under the considerations of computational resources and runtime budgets.

Convergence analysis To analyze the convergence of FedLit, we visualize the training curves and validation accuracy w.r.t. three ways of data partitioning and w.r.t. all models in Figure 4. It can be seen that the validation accuracy has the correct correspondence to the training curves among all models, where the order from large loss to small loss is aligned with the order from low performance to high performance. From the training curves in Figure 4b, we conclude that our proposed framework can converge at similar speeds as all other baselines. By observing the validation accuracy in Figure 4d, FedLit can achieve comparable performance as cGCN in the centralized setting, and always outperform the baselines of GCN and Fed-GCN. Assuming the condition that the oracle link-types are accurate, our framework FedLit would naturally have a performance lower than mGCN and Fed-mGCN.

More in-depth analyses, regarding the clustering effectiveness and client behaviors in FL are presented in Appendix C.

7 CONCLUSION

This work focuses on FL on graphs with latent link-type heterogeneity. Specifically, it studies the problems that the latent link-types including the extent of homophily and semantic relations vary among local clients, and the distributions of latent link-types are non-independent and identical. To resolve the problems, we propose a dynamic latent link-type-aware clustered federated graph learning framework (FedLit) that automatically detects the latent link-types in graphs and performs link-type-aware FL. The comprehensive experimental results and in-depth analysis demonstrate the effectiveness of FedLit. Moreover, we discuss the limitations of FedLit, such as the incapability of generating empty clusters when necessary and the lack of optimization on privacy and efficiency, which can be further studied in future work.

ACKNOWLEDGMENTS

This research was partially supported by National Science Foundation (NSF) CNS-2124104, CNS-2125530, National Institute of Health (NIH) R01ES033241, R01LM013712, as well as internal funds and GPU servers provided by the Computer Science Department of Emory University.

REFERENCES

[1] David Arthur and Sergei Vassilvitskii. 2007. K-Means++: The Advantages of Careful Seeding. In *SODA*.
[2] Debora Caldarola, Massimiliano Mancini, Fabio Galasso, Marco Ciccone, Emanuele Rodolà, and Barbara Caputo. 2021. Cluster-driven Graph Federated Learning over Multiple Domains. In *CVPRW*.
[3] Chen Chen, Jingrui He, Nadya Bliss, and Hanghang Tong. 2017. Towards optimal connectivity on multi-layered networks. *TKDE* 29, 10 (2017), 2332–2346.
[4] Chuan Chen, Weibo Hu, Ziyue Xu, and Zibin Zheng. 2021. FedGL: federated graph learning framework with global self-supervision. *arXiv preprint arXiv:2105.03170* (2021).
[5] Chaochao Chen, Jun Zhou, Longfei Zheng, Huiwen Wu, Lingjuan Lyu, Jia Wu, Bingzhe Wu, Ziqi Liu, Li Wang, and Xiaolin Zheng. 2022. Vertically federated graph neural network for privacy-preserving node classification. In *IJCAI*.
[6] Fengwen Chen, Guodong Long, Zonghan Wu, Tianyi Zhou, and Jing Jiang. 2022. Personalized federated learning with graph. In *IJCAI*.
[7] Jiayi Chen and Aidong Zhang. 2022. FedMSplit: Correlation-Adaptive Federated Multi-Task Learning across Multimodal Split Networks. In *KDD*.
[8] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. 2020. Simple and deep graph convolutional networks. In *ICML*.
[9] Mingyang Chen, Wen Zhang, Zonggang Yuan, Yantao Jia, and Huajun Chen. 2021. FedE: Embedding Knowledge Graphs in Federated Setting. In *IJCKG*.
[10] Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. 2020. Adaptive Universal Generalized PageRank Graph Neural Network. In *ICLR*.
[11] A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum Likelihood from Incomplete Data Via the EM Algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)* 39, 1 (1977), 1–22.
[12] Zishan Gu, Ke Zhang, Guangji Bai, Liang Chen, Liang Zhao, and Carl Yang. 2023. Dynamic Activation of Clients and Parameters for Federated Learning over Heterogeneous Graphs. In *ICDE*.
[13] Xu Han, Haoran Yu, and Haisong Gu. 2019. Visual inspection with federated learning. In *ICIAR*.
[14] Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. 2018. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604* (2018).
[15] Chaoyang He, Keshav Balasubramanian, Emir Ceyani, Carl Yang, Han Xie, Lichao Sun, Lifang He, Liangwei Yang, Philip S Yu, Yu Rong, et al. 2021. Fedgraphnn: A federated learning system and benchmark for graph neural networks. *arXiv preprint arXiv:2104.07145* (2021).
[16] Chaoyang He, Emir Ceyani, Keshav Balasubramanian, Murali Annavaram, and Salman Avestimehr. 2022. Spreadgnn: Decentralized multi-task federated learning for graph neural networks on molecular data. In *AAAI*.
[17] Di Jin, Zhizhi Yu, Cuiying Huo, Rui Wang, Xiao Wang, Dongxiao He, and Jiawei Han. 2021. Universal Graph Convolutional Networks. In *NeurIPS*.
[18] Wei Jin, Tyler Derr, Yiqi Wang, Yao Ma, Zitao Liu, and Jiliang Tang. 2021. Node Similarity Preserving Graph Convolutional Networks. In *WSDM*.
[19] Diederik P. Kingma and Jimmy Ba. 2017. Adam: A Method for Stochastic Optimization. In *ICLR*.
[20] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*.
[21] Anusha Lalitha, Osman Cihan Kilinc, Tara Javidi, and Farinaz Koushanfar. 2019. Peer-to-peer federated learning on graphs. *arXiv preprint arXiv:1901.11173* (2019).
[22] Runze Lei, Pinghui Wang, Junzhou Zhao, Lin Lan, Jing Tao, Chao Deng, Junlan Feng, Xidian Wang, and Xiaohong Guan. 2023. Federated Learning over Coupled Graphs. *TPDS* (2023).
[23] Rui Liu and Han Yu. 2022. Federated Graph Neural Networks: Overview, Techniques and Challenges. *arXiv preprint arXiv:2202.07256* (2022).
[24] S. Lloyd. 1982. Least squares quantization in PCM. *IEEE Transactions on Information Theory* 28, 2 (1982), 129–137.
[25] Sitao Luan, Chenqing Hua, Qincheng Lu, Jiaqi Zhu, Mingde Zhao, Shuyuan Zhang, Xiao-Wen Chang, and Doina Precup. 2021. Is Heterophily A Real Nightmare For Graph Neural Networks To Do Node Classification? *arXiv preprint arXiv:2109.05641* (2021).
[26] Yao Ma, Xiaorui Liu, Neil Shah, and Jiliang Tang. 2022. Is Homophily a Necessity for Graph Neural Networks?. In *ICLR*.
[27] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*.
[28] Guangxu Mei, Ziyu Guo, Shijun Liu, and Li Pan. 2019. Sgnn: A graph neural network based federated learning approach by hiding structure. In *BigData*.
[29] Chuizheng Meng, Sirisha Rambhatla, and Yan Liu. 2021. Cross-Node Federated Graph Neural Network for Spatio-Temporal Data Modeling. In *KDD*.
[30] Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. 2020. Geom-GCN: Geometric Graph Convolutional Networks. In *ICLR*.
[31] Elsa Rizk and Ali H Sayed. 2021. A graph federated architecture with privacy preserving learning. In *SPAWC*.
[32] Peter J Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics* 20 (1987), 53–65.
[33] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *ESWC*.
[34] Santiago Silva, Boris A Gutman, Eduardo Romero, Paul M Thompson, Andre Altmann, and Marco Lorenzi. 2019. Federated learning in distributed medical databases: Meta-analysis of large-scale subcortical brain data. In *ISBI*.
[35] Yizhou Sun and Jiawei Han. 2013. Mining heterogeneous information networks: a structural analysis approach. *Acm Sigkdd Explorations Newsletter* 14, 2 (2013), 20–28.
[36] Toyotaro Suzumura, Yi Zhou, Natahalie Baracaldo, Guangnan Ye, Keith Houck, Ryo Kawahara, Ali Anwar, Lucia Larise Stavarache, Yuji Watanabe, Pablo Loyola, et al. 2019. Towards federated graph learning for collaborative financial crimes detection. *arXiv preprint arXiv:1909.12946* (2019).
[37] Yue Tan, Yixin Liu, Guodong Long, Jing Jiang, Qinghua Lu, and Chengqi Zhang. 2022. Federated Learning on Non-IID Graphs via Structural Knowledge Sharing. *arXiv preprint arXiv:2211.13009* (2022).
[38] Lei Tang and Huan Liu. 2009. Relational learning via latent social dimensions. In *KDD*.
[39] Cunchao Tu, Han Liu, Zhiyuan Liu, and Maosong Sun. 2017. Cane: Context-aware network embedding for relation modeling. In *ACL*.
[40] Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha Talukdar. 2020. Composition-based multi-relational graph convolutional networks. In *ICLR*.
[41] Binghui Wang, Ang Li, Meng Pang, Hai Li, and Yiran Chen. 2022. Graphfl: A federated learning framework for semi-supervised node classification on graphs. In *ICDM*.
[42] Chunnan Wang, Bozhou Chen, Geng Li, and Hongzhi Wang. 2021. FL-AGCNS: federated learning framework for automatic graph convolutional network search. *arXiv preprint arXiv:2104.04141* (2021).
[43] Zhen Wang, Weirui Kuang, Yuexiang Xie, Liuyi Yao, Yaliang Li, Bolin Ding, and Jingren Zhou. 2022. FederatedScope-GNN: Towards a Unified, Comprehensive and Efficient Package for Federated Graph Learning. In *KDD*.
[44] Chuhan Wu, Fangzhao Wu, Yang Cao, Yongfeng Huang, and Xing Xie. 2021. Fedgnn: Federated graph neural network for privacy-preserving recommendation. In *ICMLW*.
[45] Han Xie, Jing Ma, Li Xiong, and Carl Yang. 2021. Federated graph classification over non-iid graphs. In *NeurIPS*.
[46] Yujun Yan, Milad Hashemi, Kevin Swersky, Yaoqing Yang, and Danai Koutra. 2022. Two sides of the same coin: Heterophily and oversmoothing in graph convolutional neural networks. In *ICDM*.
[47] Carl Yang, Jieyu Zhang, Haonan Wang, Sha Li, Myungwan Kim, Matt Walker, Yiou Xiao, and Jiawei Han. 2020. Relation learning on social networks with multi-modal graph edge variational autoencoders. In *WSDM*.
[48] Huanding Zhang, Tao Shen, Fei Wu, Mingyang Yin, Hongxia Yang, and Chao Wu. 2021. Federated graph learning–a position paper. *arXiv preprint arXiv:2105.11099* (2021).
[49] Kai Zhang, Yu Wang, Hongyi Wang, Lifu Huang, Carl Yang, and Lichao Sun. 2022. Efficient Federated Learning on Knowledge Graphs via Privacy-preserving Relation Embedding Aggregation. In *ACLW*.
[50] Ke Zhang, Carl Yang, Xiaoxiao Li, Lichao Sun, and Siu Ming Yiu. 2021. Subgraph Federated Learning with Missing Neighbor Generation. In *NeurIPS*.
[51] Longfei Zheng, Jun Zhou, Chaochao Chen, Bingzhe Wu, Li Wang, and Benyu Zhang. 2021. Asfgnn: Automated separated-federated graph neural network. *Peer-to-Peer Networking and Applications* 14, 3 (2021), 1692–1704.
[52] Xin Zheng, Yixin Liu, Shirui Pan, Miao Zhang, Di Jin, and Philip S Yu. 2022. Graph neural networks for graphs with heterophily: A survey. *arXiv preprint arXiv:2202.07082* (2022).
[53] Jiong Zhu, Ryan A Rossi, Anup Rao, Tung Mai, Nedim Lipka, Nesreen K Ahmed, and Danai Koutra. 2021. Graph neural networks with heterophily. In *AAAI*.
[54] Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. 2020. Beyond Homophily in Graph Neural Networks: Current Limitations and Effective Designs. In *NeurIPS*.
[55] Wei Zhu, Jiebo Luo, and Andrew D. White. 2022. Federated learning of molecular properties with graph neural networks in a heterogeneous setting. *Patterns* 3, 6 (2022), 100521.

A DATA DETAILS

A.1 The Oracle Link-types

The publication datasets. The oracle link-types of DBLP-DM are defined as: i) references between papers, ii) papers sharing author(s), iii) papers sharing more than one keyword, and iv) papers published in the same year. PUBMED-DIABETES has the same oracle link-types i), ii), and iv) as DBLP-DM, and a link-type iii) that papers appear among the top-k similar papers of others.

The EHR datasets. For the EHR datasets (NELL and MIMIC3), their oracle link-types include: i) admissions of the same patients, ii) the overlap ratios of procedures between two admissions above a certain threshold, iii) the overlap ratios of prescription between two admissions above a certain threshold, and iv) the time frames of stay between two admissions are overlapped.

A.2 Data Statistics

Table 6: Statistics of processed publication and EHR datasets.

Dataset	#Node	#Feature	#Class	#Total Edge	#Oracle Edge			
					i)	ii)	iii)	iv)
DBLP-DM	46,582	200	12	7,097,924	206,219	1,285,315	2,818,120	2,788,270
PUBMED-DIABETES	13,778	200	3	588,529	20,035	118,441	74,971	375,082
NELL	41,671	2,792	5	39,250,315	91,229	6,499,135	24,529,421	8,130,530
MIMIC3	58,495	6,671	6	30,603,469	23,068	27,244,566	2,413,231	922,604

B FURTHER DISCUSSION

B.1 Horizontal v.s. Vertical FL

Traditional FL applied to data in the Euclidean space can be categorized into horizontal FL and vertical FL from the perspective of data partitioning. Horizontal FL is object-based where the data among clients share the same feature space but different data samples (objects). Vertical FL is feature-based where the data among clients share the same objects but the feature spaces of them vary. In the setting of FL on graphs, depending on various downstream tasks at node-, link-, or graph-level, the definition of objects and features can be ambiguous. For example, a simple way is to define the nodes as objects and their attribute information as features in FL. In this way, if nodes in graphs across clients have the same identity but different feature spaces regarding attributes, it is considered as vertical FL on graphs. This setting can be seen in scenarios where departments in an organization can hold the same set of users but are in charge of attribute collection focusing on disparate aspects. On the contrary, if nodes across graphs have the same attribute space but are not completely aligned w.r.t. identities, then we can consider it as a horizontal FL on graphs. A real scenario can be in the healthcare system, where patient groups across institutes are usually not totally overlapped, but the feature space w.r.t. a medical diagnosis follows similar conventions.

However, the situation of FL on graphs is actually more complicated than traditional FL, due to the newly introduced perspective of graph topology. Take our scenario as an example, which considers both nodes and links distributed across local clients, the structure information, i.e., links, can be either regarded as a kind of node attribute, or as a new type of object. Since 1) our scenario does require the totally overlapped node sets across clients, and 2) our objective is studying the latent link-type heterogeneity, our setting is closer to horizontal FL, where links are considered as objects together with nodes.

B.2 Limitation

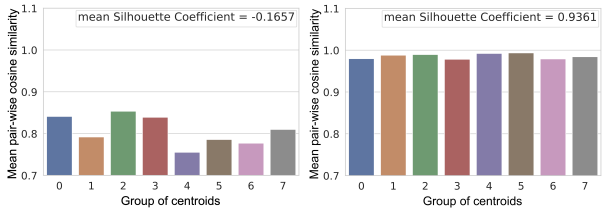
Non-empty clusters. One limitation of our framework is that edge clustering can hardly generate empty clusters when necessary, because of the large edge embedding space. Although in most clustering situations one wants to avoid empty clusters, the empty clusters can be useful in our scenarios if there are indeed some link-types missing on a client. Thus, our framework may cluster some edges wrongly in such scenarios. The good side of this problem is that the wrongly clustered edges usually only occupy a small portion of the full set of edges and therefore the overall performance will not be strongly impeded.

Data Privacy. In our setting of FL on graphs, the global task which requires collaboration among clients does not involve any direct data transmission between the server and clients or among clients themselves. All the information transmitted is represented by models/gradients that are updated by local training on clients, which prevents the data leakage problem in FEDLIT. However, we do not focus on the more advanced attacks in this work. In the future, we can dive deeper into the potential privacy problems of FEDLIT in the context of link heterogeneity, such as privacy protection against membership and backdoor attacks.

Efficiency. In this work, we focus less on the efficiency consideration, and the current training pipeline for FEDLIT takes observably longer time than Fed-GCN, although the good side is that such longer training time is still linear regarding the size of graphs and training epochs, and FEDLIT usually does not need significantly more epochs to converge. In the future, we could further improve FEDLIT in the consideration of efficiency by involving some edge sampling techniques to approximate the edge clusters, and some model compression or model distillation techniques regarding the local model aggregations.

C MORE IN-DEPTH ANALYSIS

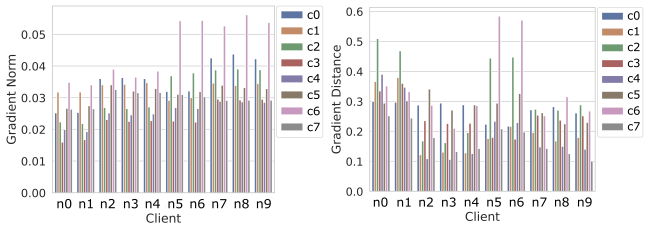
Clustering analysis. We delve into the edge clustering w.r.t link-types of FEDLIT to investigate whether the communication among local centroids is effective. To address the question, we focus on the groups of centroids corresponding to different link-types on the server and evaluate the within-group similarity of the centroids. If the edge clustering module w.r.t. link-types effectively communicates, the within-group similarity of centroids should increase during FL. The grouped bar plot in Figure 5 displays the mean pair-wise cosine similarity between centroids within groups, in which each bar represents a group corresponding to an edge cluster assignment. According to the bar plots at the first and last communication rounds, it is obvious that the mean pair-wise cosine similarity of each group increases after training of FEDLIT. Especially, the mean pair-wise cosine similarity of all groups approaches 1.0 at round 100. In addition, we also measure the Silhouette coefficients [32] for each group, evaluating the goodness of a cluster assignment and averaging them over all groups to get the mean Silhouette coefficient. We observe a high mean Silhouette coefficient over groups which is very close to 1 at round 100. These results indicate that the edge clustering w.r.t. link-types on clients in our proposed framework FEDLIT are effectively communicated and trained across clients.



(a) round 1 **(b) round 100**

Figure 5: Clustering analysis results on DBLP-DM.

Client behavior analysis. We delve into the behaviors of multiple channels of cGCN on local clients during FL with FedLit, to investigate whether the channels can be effectively learned for different link-types. We consider two points: whether the behaviors of multiple channels are different, and whether the clients with similar link-type distribution show similar behaviors w.r.t. the channels. To measure these behaviors, we compute two metrics, (a) gradient norm, and (b) gradient distance. The gradient norm of a channel is calculated by averaging its norms of gradients over the communication rounds. The gradient distance of a channel is calculated by averaging the cosine distances between its gradients and the server aggregated gradients over the communication rounds. Figure 6 visualizes the gradient norm and gradient distance of each channel (c*) in local models on all clients (n*). The grouped bars for each client can be regarded as a distribution of the channel's behaviors during FL. Thus, we can infer which clients preserve similar link-type distributions. According to the data partitioning, clients n0 and n1 have the dominant oracle link-type i); clients n2, n3, and n4 have the dominant oracle link-type ii); clients n5 and n6 have the dominant oracle link-type iii); and client n7, n8, and n9 have the dominant oracle link-type iv). In both Figure 6a and 6b, it can be observed that clients with the same dominant link-type show a similar distribution of gradient norm and gradient distance. In addition, the gradient distance measures the channel-wise contribution of clients to the global model. From Figure 6b, we can conclude that the c6 channel of models on clients n5 and n6, and the c2 channel of models on clients n0, n1, n5, and n6, contribute the most to the global model.



(a) Gradient Norm **(b) Gradient Distance**

Figure 6: FL analysis results on DBLP-DM

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009