Subgraph Federated Learning over Heterogeneous Graphs

Ke Zhang
cszhangk@connect.hku.hk
The University of Hong Kong
Hong Kong, China

Han Xie
hxie45@emory.edu
Emory University
Atlanta, U.S.A.

Zishan Gu
zg2409@columbia.edu
Columbia University
New York, U.S.A.

Xiaoxiao Li
xiaoxiao.li@ece.ubc.ca
University of British Columbia
Vancouver, Canada

Lichao Sun
lis221@lehigh.edu
Lehigh University
Bethlehem, U.S.A.

Siu Ming Yiu
smyiu@cs.hku.hk
The University of Hong Kong
Hong Kong, China

Yuan Yao
y.yao@nju.edu.cn
Nanjing University
Nanjing, China

Carl Yang*
j.carlyang@emory.edu
Emory University
Atlanta, Georgia, USA

ABSTRACT

Heterogeneous graphs containing multiple types of nodes and links are widely used to model complex real-world data mining applications. Nowadays, it is common that large and informative heterographs are separately collected and stored by multiple data owners. Therefore, it is natural to consider the *federated learning across distributed heterographs*, where each local owner holds a sub-heterograph that contains *private nodes* whose information cannot be shared with others and whose behaviors may be *biased* from the distribution of the global heterograph (the union of all sub-heterographs). Towards this innovative yet demanded setting, we propose two major techniques: (1) FedHG, which trains a type-aware GCN model using a sample-based normalization over FedAvg to integrate multi-types of node features, link structures, and task labels across sub-heterographs; (2) FedHG+, which jointly trains a type-aware missing neighbor generator with the type-aware GCN to deal with incomplete sub-heterogeneous neighborhoods. We theoretically analyze the effectiveness of both FedHG and FedHG+, regarding their expressiveness in capturing heterogeneous higher-order relations and neighborhood distributions, both extended with generalization analysis on the federated learning setting. Empirical results on two real-world heterograph datasets from different applications with synthesized distributed sub-heterographs demonstrate the effectiveness and efficiency of our proposed techniques.

CCS CONCEPTS

• **Human-centered computing → Collaborative and social computing**.

*Corresponding author.

FedGraph '22, October 21, 2022, Atlanta, GA, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/XXXXXXX.XXXXXXX

KEYWORDS

Federated Learning, Heterogeneous Graphs, Graph Mining

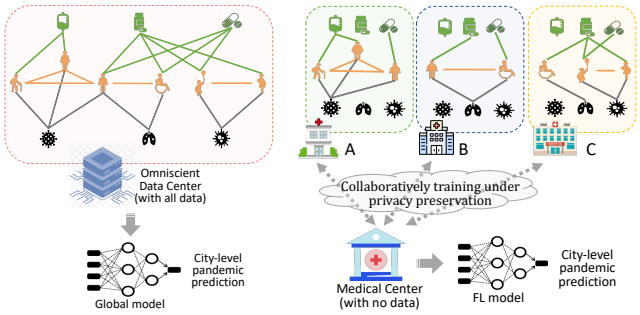
ACM Reference Format:
Ke Zhang, Han Xie, Zishan Gu, Xiaoxiao Li, Lichao Sun, Siu Ming Yiu, Yuan Yao, and Carl Yang. 2018. Subgraph Federated Learning over Heterogeneous Graphs. In *The First International Workshop on Federated Learning over Graph Data (FedGraph '22), October 21, 2022, Atlanta, GA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

In the real world, heterogeneous graphs (heterographs) containing rich types of nodes and links can well capture and abstract information from complex applications [39], such as citation prediction on bibliographical networks [26], patient profiling on clinical networks [27]) and recommender systems [2, 12, 19, 32]. Herein, we take the healthcare system as an instance, as shown in Fig. 1. For each hospital, it independently records its patients' profiles that contain various types of information, such as patients' demographics, laboratory testing data, medical treatments, and diagnosis history. By modeling patients, symptoms, medicines, and procedures as nodes of different types and linking every patient with his/her presented symptoms, received procedures, prescribed medicines, and other co-diagnosed patients, the hospital holds a heterograph.

Multiple service providers in the same domain, *e.g.*, hospitals, often separately collect local data possibly with certain selection biases compared to the entire global data. Similarly, in the healthcare system example, residents in a city can visit different hospitals based on their locations, specialties, personal preferences, and so on. Thus, each hospital possesses a local subset of the global clinical data. When each hospital constructs a local clinical heterograph, it can be regarded as a heterogeneous subgraph (sub-heterograph) of the entire heterograph constructed with all healthcare data generated in the city. Due to privacy protection regulations and interest conflicts, hospitals cannot directly share their sub-heterographs with others. However, given a city-level task such as pandemic prediction, the question arises as how to obtain a global heterograph model without actually putting the sub-heterographs together?

Federated learning (FL) [15] introduces a novel way of collaboratively learning a model across multiple data owners without



(a) Global Heterograph (b) Distributed Sub-Heterographs

Figure 1: A toy example of the real-world clinical heterograph. The medicine nodes (green) and symptom nodes (black) are public nodes, while the patient nodes (orange) are private nodes. Fig. 1(a) shows a global clinical heterograph of the city (which cannot be directly obtained in the FL setting), whereas Fig. 1(b) shows three sub-heterographs obtained by three hospitals (A, B, C), individually.

compromising local raw data. FL has achieved remarkable progress for traditional machine learning tasks, such as CV and NLP, especially when data are identically distributed [45]. Besides, FL also exhibits exciting potential in resolving learning tasks on relational data, *i.e.*, graphs [37, 42]. However, they primarily focus on homogeneous graphs (homographs), which cannot handle the diversity of node and link types. Extending the merit of [42] that studies FL over homographs, this paper formulates and addresses two novel and unique challenges in FL over distributed heterographs as follows.

Challenge 1: How to collaboratively learn a generalized heterograph mining model over distributed sub-heterographs? For nodes in a real-world distributed heterograph system, rather than being categorized into different types according to semantics, they can further be grouped according to the level of privacy they need. Specifically, we separate the heterogeneous types of nodes into public and private ones. Public nodes are those whose identities and features can be publicly accessed by every data owner in the system (*e.g.*, medicines and procedures on clinical heterographs). In contrast, private nodes have their identities and features privately collected and preserved by data owners (*e.g.*, patients and admission records in the healthcare system). Given nodes of different privacy levels, there are three sub-challenges here: (1) How to design a graph mining model effective at learning from a heterograph with multiple types of nodes; (2) How to collaboratively train this model across the distributed sub-heterographs under potential local biases; (3) How to avoid leaking private nodes' information in the collaboration process.

Solution 1: FedHG: Federated learning of a type-aware GCN. For sub-challenge 1, we first propose a type-aware GCN (T-GCN), which uses different encoders to accommodate the semantics and features of different types of nodes, and devises different message passing functions to capture the different types of links; then, we theoretically justify this design by showing its ability to approximate the message passing functions of any heterogeneous higher-order meta-paths.[1] For sub-challenge 2, we propose a sample-based normalization over FedAvg [21] to train T-GCN across distributed heterographs in a federated learning setting, and theoretically prove its effectiveness by showing an equivalence between model training with our framework and on the actual union of all distributed sub-heterographs. For sub-challenge 3, we argue about the privacy guarantee of our framework by showing that under the worst assumption where a malicious local data owner can reconstruct the features of a private node merely from the shared gradients, it still cannot confidently reveal the existence of the node in other data owners, due to the weight aggregation mechanism and our T-GCN's separate handling of features of different types of nodes. We term this distributed sub-heterographs FL framework as FedHG.

Challenge 2: How to deal with incomplete local neighborhoods during FedHG? Similar to other message-passing-based graph learning models, T-GCN mines the heterograph through convolving node features based on graph structures, *i.e.*, projecting and aggregating neighbors' information for nodes on the heterograph into the embedding vector of local neighborhoods. However, the same private nodes can have different neighbors in different local heterographs, and such information cannot be shared across the local data owners. For example, a patient can visit several hospitals, where each visit of the patient can include different examinations and diagnoses, which shall not be shared across hospitals. Therefore, for every sub-heterograph containing the patient, it only has a partial neighborhood of the patient compared to the complete neighborhood he/she shall present. As a consequence, when vanilla FedHG is applied, the system can only aggregate the knowledge learned from incomplete local neighborhoods on sub-heterographs. Such incompleteness degenerates the final performance.

Solution 2: FedHG+: Reconstructing local neighborhoods along with FedHG. To assist FedHG in aggregating more generalized graph information across sub-heterographs, we get inspired by the success of the missing neighbor generator (NeighGen) proposed in [42], and design a type-aware version of NeighGen, *i.e.*, T-NGen, whose goal is to enhance T-GCN's local graph convolution process via approximating the one executed on the global heterograph. Specifically, T-NGen mends the heterogeneous neighborhood of each node in a local sub-heterograph by generating neighbors of similar nodes in other local sub-heterographs. We theoretically prove that our T-NGen module can capture the local neighborhood distributions when trained on a single heterograph, and training it with a similar sample-based normalization technique on top of FedAvg can allow it to approximate the neighborhood distribution on the union of all sub-heterographs. We further argue about the privacy guarantee of the framework with the additional T-NGen module. We show that even if a malicious local data owner can predict the neighbors of a private node in other data owners, it cannot confidently know to which node these neighbors are linked due to the protected features of private nodes in FedHG. We term this improved framework as FedHG+.

[1] Meta-path is the de facto tool in heterographs [29], which has been leveraged to model complex message passing mechanisms by various recent studies on GNNs for heterographs [20, 34, 39, 41]

We conduct experiments on two distributed heterograph systems realistically synthesized with different numbers of data owners from real-world benchmark datasets of two different application scenarios, to empirically verify the utility of our proposed methods. We observe that both our proposed models exceed locally trained classifiers in all tested scenarios.

2 RELATED WORKS

Mining heterographs. Heterographs have been widely used to model complex real-world applications, with earlier works largely relying on pre-defined meta-paths for the characterization of semantic-rich relations and distances among nodes [26, 28, 30, 40]. Recent representation learning on heterographs started with meta-path guided random walks and proximity preservation [8, 9], followed by various designs of heterogeneous graph neural networks [11, 20, 35, 46]. Despite the exterior designs of models, the underlying need for mining heterographs has hardly changed– the comprehensive modeling of complex semantics along heterogeneous meta-graphs and within heterogeneous neighborhoods. However, to the best of our knowledge, none of the existing works on heterographs have studied the emerging setting of federated learning.

Federated learning on graphs. Recently, there has been a rapidly increasing amount of work studying federated learning (FL) on graphs. Assuming nodes are fully aligned across data owners, [22, 43] have studied the vertical FL setting where node features and structures vary across local devices. In the more common setting of horizontal FL setting, [10] proposed an open-source benchmark system for federated GNNs, which adapts to various FL algorithms, GNN models, and data distributions; [37] proposed a clustered FL framework that can be applied on graphs from different domains for graph-level classification; [33] focused on semi-supervised node classification over distributed subgraphs– these works all tried to address the data heterogeneity problem in homogeneous graphs, but ignored the possibility of inter-graph links across local devices. Some recent works [1, 3, 42] studied the distributed graph setting under the consideration of cross-graph links, but they have only studied it on homogeneous graphs. [36] studied FL over the bipartite user-item graph for recommendation and [4] studied FL over knowledge graphs for their completion. However, these graphs are still different from the general heterographs with multi-types of nodes and links as we consider in this work.

3 FL OVER HETEROGRAPHS

3.1 Problem Formulation

Notations. We denote a global heterograph as $H = \{V^p, V^s, E, \varphi, \psi, X^p, X^s\}$. $V^p \cup V^s$ includes all nodes on H, where each node v is associated with a node type $\varphi(v)$ and attributed with a feature vector $x_v \in X^p \cup X^s$ with dimension $d_{\varphi(v)}$. V^p is the set of public nodes, whereas V^s is the set of private nodes. E denotes the set of all links on H. Each $e \in E$ is associated with an edge type $\psi(e)$, which is determined by the types of nodes on its two ends. Note that in this work, we consider heterographs without multiple types of links between two specific types of nodes, but our methods extend trivially beyond this constraint.

In the FL setting, we have the central server S, and M data owners $\{D_i | i \in [M]\}$ with distributed subgraphs $\{H_i | i \in [M]\}$. Slightly different from H, we denote $H_i = \{V^p, V_i^s, E_i, \varphi, \psi, X^p, X_i^s\}$ as the sub-heterograph of H owned by D_i, for $i \in [M]$. While the global graph H conceptually exists, no entity is able to aggregate all sub-heterographs to really get H. Every data owner has a copy of the same set of public nodes V^p with shared identities and features. For private nodes in V_i^s, the corresponding owner D_i privately preserves their identities and features X_i^s.

Problem setup. According to the system described above, the global graph H has its all nodes distributed in M sub-heterographs. Note that we divide nodes into public nodes and private nodes according to their privacy levels, and we have $V^p \cap V^s = \emptyset$. For private nodes, we have $V^s = \bigcup_{i=1}^{M} V_i^s$, but $|V_i^s \cap V_j^s| \geq 0$, for $i, j \in [M]$ and $i \neq j$. That is, one private node can appear in multiple local heterographs, with different node features, but the data owners are unaware of this– both the identifies and features of private nodes in other heterographs are kept private.

We consider the downstream task of classifying private nodes on H, which is one of the most common tasks in a distributed heterograph system (*e.g.*, profiling of patients or authors). For a set of private nodes $V^t \subset V^s$ which are to be classified, each node $v \in V^t$ is labeled with $y_v \in Y$, where y_v is a d_y-dimensional one-hot vector. For a typical message-passing-based graph learning model, predicting a node's label requires an ego-graph of the node drawn from the heterograph (*i.e.*, an ego-heterograph). Therefore, querying a node $v \in V^t$ on graph H is equal to querying its ego-heterograph, which we denote as $H(v)$, and the query distribution on H is $\mathcal{D}_H$, *i.e.*, $(H(v), y_v) \sim \mathcal{D}_H$.

We formulate our goal of federated node classification on heterographs as follows.

Goal. The system exploits an FL framework to collaboratively learn on isolated sub-heterographs, $\{H_i\}_{i\in[M]}$, across M data owners, without sharing the information of private nodes, to obtain a global node classifier F. The learnable weights θ in F are optimized on queried ego-heterographs following the distribution of ones drawn from the global heterograph H.We formulate the problem as finding θ^* that minimizes the risk $\mathcal{R}$ on H by aggregating local risks as

$$\theta^* = \arg\min \mathcal{R}(F(\theta|H)) = \frac{1}{M}\sum_{i}^{M} \mathcal{R}_i(F_i(\theta|H_i))), \tag{1}$$

where $\mathcal{R}_i$ is the local empirical risk defined as

$$\mathcal{R}_i(F_i(\theta|H_i)) \coloneqq \mathbb{E}_{(H_i(v),y_v)\sim\mathcal{D}_{H_i}}[\ell(F_i(\theta; H_i(v)), y_v)]. \tag{2}$$

3.2 FedHG

To collaboratively learn a heterograph mining model across the distributed heterograph system with proper privacy protection, we design a type-aware GCN (T-GCN) stemming from Relational-Graph Convolution Network (RGCN) [25] and modify the vanilla FedAvg framework to achieve better model generalization.

RGCN is defined as an L-layer simple propagation model for calculating the forward-passing update of a node v on a multi-relational graph, *e.g.*, a knowledge graph. For $l \in [L]$, the l-th layer message passing with RGCN is as follows

$$h_v^l = \sigma\left(\sum_{r\in\psi}\sum_{u\in\mathcal{N}^r(v)} c_{v,r} W_r^l h_u^{l-1} + W_0^l h_v^{l-1}\right), \tag{3}$$

where $h_v^l \in \mathbb{R}^{d_l}$ is the hidden state of node v in the l-th layer with d_l the dimension of this layer. $\sigma(\cdot)$ is the activation function, $\mathcal{N}^r(v)$

denotes the set of nodes in relation r with v. $c_{v,r}$ is a normalization constant. W_r^l and W_0^l are the weight matrices for relation r and self-connection in the l-th layer, respectively.

Originally designed for knowledge graph completion, RGCN ignores the diversity of node types, which can cause the misalignment of the input spaces when nodes have different features. For example, recall the clinical heterograph, where patient nodes' features have different semantics and dimensions from medicine nodes' features.

To properly incorporate the node feature information into the message-passing process, we propose a type-aware GCN (T-GCN) on heterographs with node-type-aware convolutions.

T-GCN. As shown in Fig. 2, our T-GCN contains two components, *i.e.*, (1) type-aware encoders: a set of type-aware feedforward neural networks (FNNs) to accommodate the semantics and features of various types of nodes, and (2) type-aware message-passing functions: a set of type-aware graph convolution networks (GCNs) to model different types of neighbors during the heterograph convolution.

For a node v of type $\varphi(v)$, T-GCN first aligns its features via the type-aware encoder as $h_v^0 = W_{\varphi(v)} x_v$, where $W_{\varphi(v)}$ is the learnable weights for the encoder.

For $l \in [L]$, the l-th layer message passing with T-GCN for node v is as follows

$$h_v^l = \sigma\left(\sum_{u \in \mathcal{N}(v)} c_{\mathcal{N}(v),\varphi(u)} W^l_{\varphi(u),\varphi(v)} h_u^{l-1} + W_0^l h_v^{l-1} \right), \tag{4}$$

where $\mathcal{N}(v)$ is v's heterogeneous neighborhood, $c_{\mathcal{N}(v),\varphi(u)}$ is a normalization constant for v's $\varphi(u)$-type neighbors within $\mathcal{N}(v)$, and $W^l_{\varphi(u),\varphi(v)}$ is the learnable weights for the message passing function on the link between $\varphi(u)$ and $\varphi(v)$ types of nodes.

We theoretically analyze the ability of T-GCN to approximate the message passing functions of any heterogeneous higher-order meta-paths in Theorem 3.1. Its proof follows the one of Theorem 2.1 in [18], and is provided in Appendix A.

Theorem 3.1 (Modeling Meta-paths with A Composition of R Functions). *For a heterograph H defined in Section 3.1 with R types of relations, we assume there is an oracle function $\hat{O}$ that takes in a target node v's meta-paths information $\mathcal{M}_v \in \mathbb{R}^d$ on H, and outputs the v's ground-truth label $y_v \in \mathbb{R}^d$. When $\mathcal{M}_v$ is absolutely continuous with respect to the Lebesgue measure, for any given approximation error ε and R functions $\{F_r | r \in [R]\}$, there exists a composition function $Comp(\cdot|\{F_r|r \in [R]\}) : \mathbb{R}^d \to \mathbb{R}^d$, which is viewed as the gradient function of an FNN $u(\cdot) : \mathbb{R}^d \to \mathbb{R}$ with ReLU activation, of depth $L = [\log_2 n]$ and width $N = 2L$, where $n = O(\frac{1}{\varepsilon^d})$. For the 1-Wasserstein distance measurement $W_1(\cdot,\cdot)$, we have*

$$\mathbb{E}_{\mathcal{M}_v \sim H}[W_1(\hat{O}(\mathcal{M}_v), Comp(\mathcal{M}_v|\{F_r|r \in [R]\})] < \varepsilon.$$

Normalized FedAvg. To collaboratively learn the proposed T-GCN across distributed sub-heterographs with proper privacy protection, we propose an FL framework with sample-based normalization over FedAvg.

For a queried node $v \in V^t$, a globally shared $(L+1)$-layer T-GCN classifier F integrates v and its L-hop multiple types of neighborhood on graph H to conduct prediction with learnable parameters $W = \{W_\varphi\} \cup \{W^l_{\varphi,\varphi}\}_{l=1}^L$.

With F outputting the inference label $\widetilde{y}_v = \text{Softmax}(h_v^L)$ from h_v^L computed with Eq. 4, we defined the supervised loss function $\ell(W|\cdot)$ as the cross-entropy function shown below

$$\ell(W|H(v), y_v) = -\left[y_v \log \widetilde{y}_v + (1 - y_v) \log (1 - \widetilde{y}_v)\right]. \tag{5}$$

In our FL setting, sub-heterographs are independently collected with potential selection bias. The non.i.i.d. distributions of training data sampled from sub-heterographs ($\mathcal{D}_{H_i}$) can bring non-trivial degeneration for the FL model's performance when evaluated with queries from the global distribution ($\mathcal{D}_H$) [16]. Thus, we design sample-based normalization over FedAvg to reweigh the contribution of individual nodes to the training process based on the number of samples in each sub-heterograph.

Specifically, we introduce a locally computed sample-based normalization term into the model updating process. During each epoch e, every D_i locally computes the normalized gradients as

$$\nabla \tilde{\ell}(W|H_i) = \sum_{y \in Y} \left(\frac{|V_i^t|}{|V^t|} \nabla \ell(W|\{(H_i(v), y)|v \in V_i^e, y_v = y\}) \right), \tag{6}$$

where $V_i^e \subseteq V_i^t$ contains the sampled training nodes for epoch e.

Then D_i updates the model with $W_i \leftarrow W - \eta \nabla \tilde{\ell}(W|H_i)$, where η is the learning rate, and sends W_i to the central server S. After collecting the latest $\{W_i|i \in [M]\}$, S sets W as their average and broadcasts W to all data owners, which finishes one round of training on F. After e_c epochs, the entire system retrieves F as the outcome global classifier, which is not limited to or biased towards the queries in any specific data owner.

We also theoretically analyze FedHG's effectiveness in obtaining a classifier that generalizes as the one trained on the union of all sub-heterographs in Theorem 3.2. Its proof follows the one of Theorem 3 in [38] and is omitted due to the space limitation.

Theorem 3.2 (Generalization Bound of FedHG). *For FedHG with M data owners communicating with the central server to retrieve a classifier parameterized by $\widehat{W} = \mathbb{E}[W_i]$, we denote that each data owner D_i, where $i \in [M]$, has an independent training set $S_i = \{(H_i(v), y_v)|v \in V_i^t\}$. With denoting the optimal global weights as $\widehat{W}^*$, which is retrieved from training on the ego-heterographs sampled from the global heterograph, and assuming the expectation of $S = \{S_i\}_{i \in [M]}$ as μ , FedHG's generalization error, i.e.,* $\text{gen}(\mu; FedHG)$, *is given by*

$$-\mathbb{E}[\frac{1}{|V^T|} \sum_{i \in [M]} \sum_{v \in V_i^T} \psi_+^{*-1}\left(I\left(S_{i,v}; \widehat{W}^*\right)\right)] \leq \text{gen}(\mu; FedHG)$$
$$\leq \mathbb{E}[\frac{1}{|V^T|} \sum_{i \in [M]} \sum_{v \in V_i^T} \psi_-^{*-1}\left(I\left(S_{i,v}; \widehat{W}^*\right)\right)],$$

where $S_{i,v} = (H_i(v), y_v) \in S_i$; ψ_+^{*-1}, ψ_+^{*-1} are defined in Appendix B.

Privacy discussions. Under the worst-case assumption where a malicious local data owner (attacker) can reconstruct the original features of a private node v merely from the shared model weights during FL, the attacker has no reference to map these features to a specific private node in other sub-heterographs, because the original features of private nodes are never shared and can be rather different across data owners. Moreover, the normalized weight aggregation mechanism also helps to further prevent the attacker from inferring the membership of v to any specific sub-heterograph.

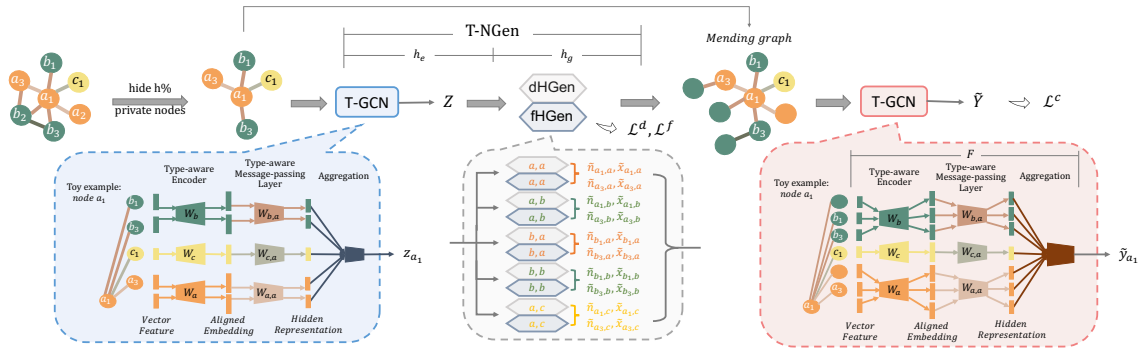


Figure 2: Joint training of T-NGen with T-GCN. This figure shows an example of training the joint model with a one-hop ego-heterograph of an orange-type node a_1. Within the figure, different colors denote different node types. The unnumbered nodes are the generated neighbors outputted by T-NGen.

4 MISSING NEIGHBOR GENERATION

Private nodes appearing in multiple sub-heterographs can have different local neighborhoods, each of which thus only has incomplete information. Aggregating such biased ego-heterographs can degenerate the federated classifier's performance for queries drawn from the global heterograph. To obtain a generalized global classifier with FedHG, we propose a novel type-aware missing neighbor generator, T-NGen, to correct the local neighborhood distributions of each private node by generating its potential missing neighbors.

4.1 T-NGen

Similar to the purpose of NeighGen [42], for a queried private node drawn from a specific sub-heterograph, we design T-NGen to generate its possible neighbors in all local sub-heterographs. Note that T-NGen differs from NeighGen in the necessity to deal with multiple types of neighbors under privacy concerns. By using a properly trained T-NGen model, the local data owner can mend the incomplete neighborhood of each private node by predicting its missing neighbors in other data owners and adding them as virtual neighbors. After data owners mend their incomplete local neighborhoods with generated missing neighbors, they can obtain the mended sub-heterographs with neighborhood distributions following the ones on the global heterograph. In this way, the heterograph convolution process on each mended sub-heterograph is similar to the one executed on the global heterograph.

Model structure. Technically, as shown in Fig. 2, T-NGen consists of two modules, which can be regarded as the heterogeneous extensions of NeighGen in [42], *i.e.*, a type-aware encoder h_e and a type-aware generator h_g as follows.

h_e: A T-GCN model, *i.e.*, an (L+1)-layer T-GCN encoder, with parameters θ^e. For node $v \in V_i^s$ on the input graph H_i, h_e computes node embeddings $Z_i = \{z_v = h_v^L \in \mathbb{R}^{d_z} | v \in V_i^s\}$ w.r.t. Eq. (4).

h_g: A generative model reconstructing multiple types of missing neighbors for the input heterograph based on the node embedding of the center node. h_g contains two submodules of dHGen and fHGen, where dHGen is a type-aware linear regression model parameterized by $\theta^d_{\varphi_1,\varphi_2}$ and fHGen is a type-aware feature generator parameterized by $\theta^f_{\varphi_1,\varphi_2}$, where φ_1 and φ_2 are the types of the center node and its neighbor.

Specifically, for a queried private node v, dHGen and fHGen respectively predicts the numbers and corresponding features of its various types of missing neighbors. Specifically, when T-NGen generates v's φ_2 type neighbors, dHGen predicts $\widetilde{n}_{v,\varphi_2}$, and fHGen generates a set of neighbor features $\widetilde{x}_{v,\varphi_2} \in \mathbb{R}^{\widetilde{n}_{v,\varphi_2} \times d_{\varphi_2}}$.

Both dHGen and fHGen are constructed as type-aware FNNs, while fHGen is further equipped with a Gaussian noise generator $\mathbf{N}(0,1)$ that generates d_z-dimensional noise vectors and a random sampler R. For node $v \in V_i^s$, fHGen is variational, which generates the missing neighbors' features for v after inserting noises into the embedding z_v for each possible missing node type φ_2, while R ensures fHGen to output the features of a specific number of φ_2-type neighbors by sampling $\widetilde{n}_{v,\varphi_2}$ feature vectors from the feature generator's output. Mathematically, to retrieve the φ_2-type neighbors for a node v of type φ_1, we have

$$\begin{aligned} &\widetilde{n}_{v,\varphi_2} = \sigma(\theta^d_{\varphi_1,\varphi_2} \cdot z_v), \widetilde{x}_{v,\varphi_2} = R\left(f(v,\varphi_2), \widetilde{n}_{v,\varphi_2}\right), \\ &\text{where } f(v,\varphi_2) = \sigma\left(\theta^f_{\varphi_1,\varphi_2} \cdot (z_v + \mathbf{N}(0,1))\right). \end{aligned} \tag{7}$$

Expectations for T-NGen lie in two aspects: (1) generation-wise: generating realistic structures and feature distributions, and (2) generalization-wise: enabling local data owners to achieve a similar graph convolution process as the one on the global heterograph. To satisfy the described expectations, we introduce our training methods for T-NGen in the following.

4.2 Local training of T-NGen

To fulfill the generation-wise expectation for T-NGen, we start from learning on a single sub-heterograph. To obtain neighborhoods' structures and feature distributions for the supervised training of T-NGen, we design a heterograph mending simulation based on [42] to first locally simulate the incomplete neighborhoods by separately impairing private and public neighborhoods. Specifically, we first hide h% of the local private nodes and their links to simulate the private neighbors that are not locally recorded. Then, for the remaining private nodes, we randomly hide h% of their links to public nodes. We denote the set of hidden nodes as $V_i^h \subset V_i^s$, and the set hidden links as $E_i^h \subset E_i$. We have the impaired local sub-heterograph as $\bar{H}_i = \{V_i^p, \bar{V}_i^s, \bar{E}_i, \varphi, \psi, X^p, \bar{X}_i^s\}$, where $\bar{V}_i^s = V_i^s \setminus V_i^h$, $\bar{E}_i = E_i \setminus E_i^h$, and $\bar{X}_i^s = X_i^s \setminus X_i^h$.

Accordingly, based on the local ground-truth missing nodes V_i^h and links E_i^h, the training of T-NGen on the impaired local graph $\bar{H}_i$ boils down to jointly training dHGen and fHGen as follows

$$\mathcal{L}^n = \lambda^d \mathcal{L}^d + \lambda^f \mathcal{L}^f = \frac{1}{|\bar{V}_i^s|} \sum_{v \in \bar{V}_i^s} \left(\sum_{\varphi_2 \in \varphi} c_{v,\varphi_2} (\lambda^d \mathcal{L}^d_{v,\varphi_2} + \lambda^f \mathcal{L}^f_{v,\varphi_2}) \right), \tag{8}$$

where λ^d and λ^f are two tunable hyper-parameters, c_{v,φ_2} is a normalization term. Herein, $\mathcal{L}^d$ forces dHGen to learn the structural information on local H_i, whereas $\mathcal{L}^f$ encodes realistic local node feature distributions into fHGen. Specifically, for a φ_1-type node $v \in \bar{V}_i^s$, the calculations of dHGen's loss $\mathcal{L}^d$ and fHGen's loss $\mathcal{L}^f$ on v's predicted φ_2-type missing neighbors are

$$\mathcal{L}^d_{v,\varphi_2} = L_1^S(\widetilde{n}_{v,\varphi_2} - n_{v,\varphi_2}), \tag{9}$$

$$\mathcal{L}^f_{v,\varphi_2} = \sum_{k \in [\max(\widetilde{n}_{v,\varphi_2}, n_{v,\varphi_2})]} \min_{u \in \mathcal{N}_i^{\varphi_2}(v) \cap V_i^h} (||f(v,\varphi_2)^{[k]} - x_u||_2^2), \tag{10}$$

where L_1^S is the smooth L1 loss, $f(v,\varphi_2)^{[k]} \in \mathbb{R}^{d_{\varphi_2}}$ is the k-th predicted feature in $f(v,\varphi_2)$. $\mathcal{N}_i^{\varphi_2}(v) \cap V_i^h$ contains v's φ_2-type neighbors hidden into V_i^h. $\mathcal{N}_i^{\varphi_2}(v) \cap V_i^h$, which can be retrieved from V_i^h and E_i^h, provides ground-truth for locally training T-NGen.

Now we theoretically analyze the ability of T-NGen in capturing the missing neighbor distribution when trained on a local heterograph in Theorem 4.1. The proof follows the one of Theorem 2.1 in [18], and it is omitted due to the space limitation.

Theorem 4.1 (Local r-type Neighborhood Generation). *In a heterograph H defined in Section 3.1, for a relation r with the node type of the predicted end with feature dimension d, and an input node embedding $z \sim Z$ in dimension d, if the local r-type neighborhood distribution P_r lies in a bounded space of $\mathbb{R}^d$, for an FNN $u_r(\cdot): \mathbb{R}^d \to \mathbb{R}$, which satisfies the assumptions for the FNN in Theorem 3.1, the 1-Wasserstein distance $W_1(P_r, \nabla u_r(Z)) < \varepsilon$ holds for any $\varepsilon > 0$. $\nabla u_r(Z)$ is the r-type neighborhood distribution that is generated from the embedding space Z through the mapping $\nabla u_r(\cdot)$.*

4.3 FedHG+

To satisfy the second generalization expectation for T-NGen, we propose an FL framework, FedHG+, to enable data owners to securely transfer diverse neighborhoods' information across the system. Intuitively, perceiving neighborhood information across entities fosters the T-NGen's ability to generate more diverse neighbors.

Jointly train T-NGen with T-GCN. To improve the quality of T-NGen's generated nodes to better accommodate the downstream task, we embed downstream task labels into T-NGen by jointly training it with a T-GCN classifier.

After getting the predicted missing neighbors from T-NGen, the data owner mends the local sub-heterograph and then feeds the mended sub-heterograph into a T-GCN classifier. The classifier provides a supervised loss $\mathcal{L}^c$ for the joint model, which is calculated following Eq. (5) by substituting the H_i with the mended sub-heterograph H_i'. Combining Eq. (8) with $\mathcal{L}^c$, we train the joint model by minimizing the following objective function

$$\mathcal{L}^n = \lambda^d \mathcal{L}^d + \lambda^f \mathcal{L}^f + \lambda^c \mathcal{L}^c, \tag{11}$$

where λ^c is an additional hyper-parameter.

Federated learning of the joint model. We introduce FedHG+ as the federated training process for the joint model of T-NGen and T-GCN. It is worth noting that in consideration of enhancing a T-NGen's capability in transferring features learned from the system to a local sub-heterograph, so as to serve the downstream task better, in the FL process, each data owner separately updates its exclusive T-NGen model. Without loss of generality, we use D_i as an example to describe the FedHG+ process.

FedSage+[42] provides an insight into how to federally train the missing neighbor generator. Yet it requires data owners to send their node embeddings and models across the system, which can leak nodes' information [44]. As nodes to be reconstructed with missing neighbors in distributed heterograph system are private, in this work, for T-NGen, we design a secure FL process that does not require the sharing of embeddings so as to reduce the exposure of private nodes.

Technically, there are four steps in an epoch of FedHG+. Firstly, D_i sends its joint model, *i.e.*, T-NGen$_i$ and the T-GCN classifier, to all data owners in the system. Next, D_i gets the gradient $\nabla \mathcal{L}_i^n$ computed with Eq. (11). Note that the joint classifier's gradient $\nabla \mathcal{L}_j^c$ is normalized by applying Eq. (6). Simultaneously, for $j \in [M] \setminus \{i\}$ in parallel, data owner D_j fixes h_e and gets the gradient $\nabla \mathcal{L}_j^n$ computed with Eq. (11), whose $\bar{V}_i^s$ and H_i' are substituted by $\bar{V}_j^s$ and H_j' respectively. Similarly, $\nabla \mathcal{L}_j^c$ is normalized by applying Eq. (6) on H_j'. Then, data owners send the gradients back to D_i. Finally, D_i updates joint model's learnable parameters with $\theta_i \leftarrow \theta_i - \sum_{j \in [M]} \lambda_j \nabla \mathcal{L}_j^n$ to retrieve the latest T-NGen$_i$ and T-GCN of this epoch, where λ_j's are tunable hyper-parameters.

After retrieving the federally trained T-NGen models across all data owners, every data owner mends its respective sub-heterograph by leveraging T-NGen on g% of its private nodes. On obtaining mended sub-heterographs, data owners perform FedHG according to the description in Section 3.2 and retrieve the shared generalized T-GCN classifier. The respective generalization bound is given in Theorem 4.2, whose proof is in Appendix C.

Theorem 4.2 (Generalization Bound of FedHG+). *For the system defined in Section 3.2, the FL process of training the joint model for a data owner via FedHG+ requires communication among M data owners. We denote each data owner $D_i (i \in [M])$ has an independent training set retrieved from the mended graph H_i' as $S_i' = \{(H_i'(v), y_v) | v \in V_i^s\}$, where $\mathcal{H}_i'(v)$ is the local ego-heterograph of node v drawn from the mended sub-heterograph H_i', and y_v is v's label. Similarly, we denote the training set retrieved from the original sub-heterographs (with incomplete neighborhoods) as $S_i = \{(H_i(v), y_v) | v \in V_i^s\}$. With denoting optimal global weights as $\widehat{W}_i^*$ retrieved from training on the ego-heterographs sampled from the global heterograph (with complete neighborhoods), FedHG+'s generalization error, i.e.,* $\text{gen}\left(\mathbb{E}_{S_{i,v}'}; FedHG+\right)$*, is given by*

$$-\mathbb{E}[\frac{1}{|V^s|} \sum_{i \in [M]} \sum_{v \in V_i^s} \psi_+^{*-1}(I(S_{i,v}'; \widehat{W}_i^*))] \le \text{gen}(\mathbb{E}_{S_{i,v}'}; FedHG+)$$
$$\le \mathbb{E}[\frac{1}{|V^s|} \sum_{i \in [M]} \sum_{v \in V_i^s} \psi_-^{*-1}(I(S_{i,v}'; \widehat{W}_i^*))],$$

and we have $\psi_-^{*-1}(I(S_{i,v}'; \widehat{W}_i^*)) \le \psi_-^{*-1}(I(S_{i,v}; \widehat{W}_i^*))$, namely the upper bound is tighter than that of Theorem 3.2.

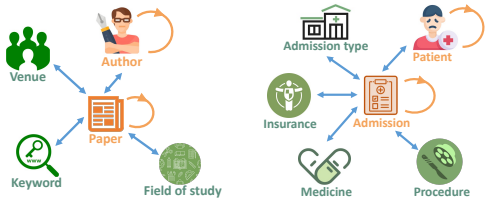
Privacy discussions. We consider a malicious local data owner who can use its T-NGen model to generate the neighborhood structures and features of any private node. Yet it can only do this for its own private nodes, because it has no access to the original features and the incomplete neighborhoods of others' private nodes. Due to the weight aggregation mechanism, it is difficult for the malicious data owner to infer a generated neighbor's original data owner.

5 EXPERIMENTS

We conduct comprehensive experiments on two real-world heterographs constructed from benchmark datasets in two application scenarios and compare the models toward the practical node classification task. The further in-depth analysis illustrates the advantages of our proposed techniques.

5.1 Experimental settings

We choose a large real-world bibliographical dataset DBLP [31] and a widely used clinical dataset MIMIC-III [14] to simulate the distributed heterograph system.



(a) DBLP (b) MIMIC-III

Figure 3: Schemas of two real-world heterographs. Green nodes are public nodes, and orange nodes are private nodes.

DBLP: We construct a heterograph of authors, papers, venues, fields of study, and keywords from DBLP. The schema of this bibliographic heterograph is shown in Fig. 3(a). We construct sub-heterographs using the venue information– we chose M largest venues with the most number of papers, and construct each sub-heterograph based on all papers published in the corresponding venue. For each public node, we use the average GloVe embedding (300-dim) [23] of all words in its name as the node feature. For each private node, to simulate local features, each data owner computes an average over the node features of its all 2-hop public neighbors on the local heterograph. We categorize authors into five classes according to the number of their total citations in the entire dataset and use the downstream task of citation classification, which is a common and challenging task on the bibliographic graphs [5, 13, 24].

MIMIC-III: For MIMIC-III, we construct a heterograph of patients, admission records, admission types, medicines, procedures, and insurance types. The schema of this clinical heterograph is shown in Fig. 3(b). We construct the sub-heterographs w.r.t. insurance types– since the entire dataset only contains five insurance types, we use $M = 3, 5$ for this dataset. After selecting M insurance types, we construct each sub-heterograph based on all admissions with the corresponding insurance type. All public nodes are one-hot encoded. As for private nodes, patients' features contain their demographic information, and features of the admission records are the lab test results. We categorize patients into five classes according to the average length of their staying in ICU. The downstream task is thus to predict the severeness of the patients' diseases, with the ICU time as labels [6, 7].

Table 1: Statistics of the datasets and the synthesized distributed heterograph system with different numbers of data owners. $\mathcal{T}$ is the number of node types; #C denotes the number of classes in the downstream task; M is the simulated number of data owners; $|V^p \cup V^s|$ and $|E|$ are the total number of nodes and links in all data owners, respectively; $|V^p \cup V_i^s|$ and $|E_i|$ are the averaged number of nodes and links in each data owner (the same nodes or links can appear in multiple data owners); ΔE denotes the average number of missing neighbors in each data owner. We further show the averaged missing ratio of local neighbors.

Data-($\|\mathcal{T}\|$,#C)	DBLP-(5,5)			MIMIC-III-(6,5)	
M	3	5	10	3	5
$\|V^p \cup V^s\|$	369,531	546,309	883,519	43,747	44,792
$\|E\|$	5,388,638	8501,978	13,103,128	2,332,254	2,393,110
$\|V^p \cup V_i^s\|$	134,665	124,220	105,402	16,801	10,888
$\|E_i\|$	1,859,946	1803,856	1485,254	751,826	461,873
ΔE_i	302,166	494,570	588,221	27,406	18,017
	(13.98%)	(21.52%)	(28.37%)	(3.52%)	(3.75%)

For T-GCN model, in the DBLP dataset, we implement it with 5 layers, while in the MIMIC-III dataset, we implement it with 3 layers. For T-NGen model, we implement its h_e as the same T-GCN model (5 layers for the DBLP and 3 layers for the MIMIC-III), dHGen as a set of type-aware 3-layer FNNs, and fHGen as the combination of a set of type-aware 3-layer FNNs and a Gaussian random sampler. We train models on DBLP using batch size 256, and setting training epochs to 50. In MIMIC-III, we use batch size 32, and set training epochs to 50. For both datasets, we sample 4 instances for each type of the node-pair combination at each layer. The training-validation-testing ratio is 50%-10%-40%.

Based on our observations in hyper-parameter studies, for the graph hiding portion h and the graph generating ratio g, we set $h\% \in [20\%, 80\%]$ (varying across datasets and number of clients) and fix $g\% = 20\%$. All loss weights λs are simply set to 1. Optimization is done with RAdam [17] with a learning rate of 0.001. We implement FedHG and FedHG+ in Python and execute all experiments on a server with 8 NVIDIA GeForce GTX 1080 Ti GPUs. All code and data are released on Github. [2]

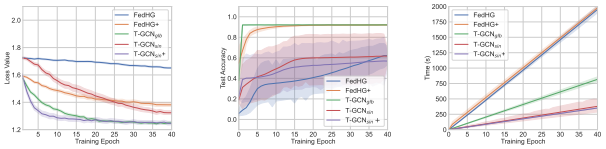
Since we are the first to study the novel yet the important setting of sub-heterograph federated learning, there are no existing baselines. We conduct a comprehensive ablation evaluation by comparing FedHG and FedHG+ with three natural baseline models, i.e., 1) T-GCN$_{glb}$: the T-GCN model trained on the original global heterograph without data partitioning (as an upper bound for any FL framework without considering T-NGen), 2) T-GCN$_{sin}$: one T-GCN model trained solely on each sub-heterograph, 3) T-GCN$_{sin}$+: the T-GCN plus T-NGen model jointly trained solely on each subgraph. The metric used in our experiments is the node classification

[2] https://github.com/zkzhangke/FedHGN

Table 2: Node classification results on two datasets with varying numbers of clients. Averaged accuracy and the corresponding std are provided.

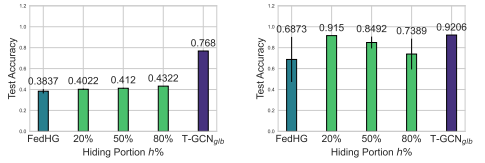
	DBLP			MIMIC-III	
Model	M=3	M=5	M=10	M=3	M=5
T-GCN$_{sin}$	0.3336	0.3360	0.3296	0.7002	0.5782
	± 0.0103	± 0.0306	± 0.0525	±0.3643	± 0.3912
T-GCN$_{sin}$+	0.3325	0.3432	0.3215	0.7690	0.4743
	±0.0024	± 0.0402	± 0.0763	± 0.313	±0.3965
FedHG	0.3336	0.3837	0.3356	0.7401	0.6873
	±0.0003	± 0.0214	± 0.0019	± 0.2165	±0.2654
FedHG+	**0.3343**	**0.4322**	**0.3673**	**0.8054**	**0.8492**
	± 0.0006	± 0.0142	± 0.0051	± 0.0954	±0.0565
T-GCN$_{glb}$	0.6419	0.7680	0.7041	0.9201	0.9206
	± 0.0010	±0.0014	± 0.0011	± 0.0002	± 0.0004

accuracy on the queries sampled from the testing set on the global heterograph. For globally (or federally) trained models of T-GCN, FedHG, and FedHG+, we report the average accuracy over five random repetitions, while for locally trained models of T-GCN$_{sin}$ and T-GCN$_{sin}$+, the scores are further averaged across M local models.

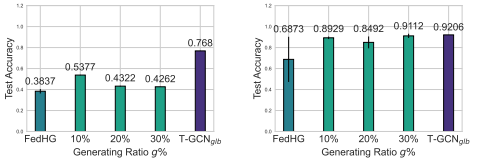


(a) Training Loss (b) Testing Accuracy (c) Training Time

Figure 4: Training curves of compared frameworks.



(a) DBLP with $g\% = 20\%$. (b) MIMIC-III with $g\% = 20\%$.



(c) DBLP with $h\% = 80\%$. (d) MIMIC-III with $h\% = 50\%$.

Figure 5: Hyper-parameter studies on T-NGen.

5.2 Experimental results and analysis

Overall performance analysis. Comprehensive experimental results shown in Table 2 empirically verify the non-trivial elevations brought by FedHG and FedHG+ in federated node classification. Primarily, we can observe from the results that FedHG+ improves T-GCN$_{sin}$ by an average of 10.22% across all settings on two datasets, which demonstrates its superior utility in this novel and important setting. Meanwhile, it significantly dismisses the average accuracy drop brought by the incomplete neighborhood problem– FedHG+ narrows the absolute accuracy reduction of FedHG, when compared with the upper-bound model T-GCN$_{glb}$, by at most 16.19%.

The notable gaps between a locally obtained classifier and a federally trained classifier, *i.e.*, by comparing T-GCN$_{sin}$ or T-GCN$_{sin}$+ with FedHG or FedHG+, prove the benefits brought by the collaboration across local data owners. When comparing FedHG+ with FedHG, the considerable elevation brought by T-NGen corroborates the assumed degeneration brought by incomplete neighborhoods and validates the effectiveness of our innovatively designed T-NGen module. Notably, when each sub-heterograph has a relatively larger amount of missing neighbors (*e.g.*, MIMIC-III with five data owners in Table 2), FedHG+ significantly exhibits its robustness in resisting the information loss compared to FedHG. It is worth pointing out that observation of the comparatively smaller gaps between T-GCN$_{sin}$ and T-GCN$_{sin}$+ indicate that our T-NGen is uniquely crucial in the sub-heterograph FL setting.

In-depth model analysis. Take MIMIC-III with five data owners as an example, we visualize the training loss, testing accuracy, and training time along 40 epochs in obtaining the node classifier with all compared frameworks in Fig. 4. From subfigures (a) and (b), we observe that FedHG+ can consistently achieve convergence with rapidly improved testing accuracy (larger loss value due to additional objectives but rather close performance towards T-GCN$_{glb}$), while FedHG struggles to converge with acceptable accuracy. This again asserts our assumption on the non-negligible degeneration that can be caused by locally incomplete neighborhoods. The locally trained models of T-GCN$_{sin}$ and T-GCN$_{sin}$+ are easier to converge, perhaps due to their less complex local heterographs, but they do not achieve good performance after convergence, because simply averaging multiple biased models do not lead to a good one. The locally trained T-NGen seems to help the convergence of local T-GCN, but it does not help improve the performance either, again indicating the unique advantage of our model designs in the federated learning setting. Finally, regarding the training time, the inclusion of T-NGen does not incur significantly more training time for FedHG+ compared with FedHG. Due to the additional communications and computations in FL, both FedHG and FedHG+ consume observably more training time compared to T-GCN$_{glb}$, but the overhead is tolerable given their unique benefit in avoiding direct data sharing or centralizing.

Hyper-parameter studies. We compare the downstream task performance under different hiding portions h% and generating ratio g% with on two datasets both with the five data owner setting. Results are shown in Fig. 5, where Fig. 5(a) and 5(b) show results when g% is fixed, and Fig. 5(c) and 5(b) show results with h% fixed.

Both h% and g% affect the local neighborhood completion process. As observed from Fig. 5(a) and 5(b), choosing a proper h%, which controls the learning of neighborhood distribution through the local neighborhood incompleteness simulation, can constantly elevate the final testing accuracy. Notably, when h% is set to a value close to the actual amount of missing neighbors (c.f. Table 1 last row), the model obtained from FedHG+ achieves the best performance.

As the hyper-parameter determines the number of nodes to be mended to local sub-heterographs, $g\%$ is crucial in controlling the model expressiveness and computation overhead trade-off for a local data owner. Referring to Fig. 5(c) and 5(d), we can observe that choosing a relatively small $g\%$ can assist local data owners in achieving satisfying downstream task performance without overly extending the local sub-heterographs' sizes.

6 CONCLUSION

In this work, we consider the innovative yet demanded setting of federated learning across distributed heterographs. We propose FedHG to apply FL across distributed heterographs without compromising the information of private nodes, and design FedHG+ to overcome the local neighborhood incompleteness problem. Empirical results and theoretical analysis corroborate the effectiveness of our proposed techniques. Important future directions might include the experimental analysis of more different datasets, communication compression techniques to reduce FL overhead, and further rigorous analysis on privacy and model robustness.

REFERENCES

[1] Chuan Chen, Weibo Hu, Ziyue Xu, and Zibin Zheng. 2021. FedGL: Federated Graph Learning Framework with Global Self-Supervision. arXiv:2105.03170 [cs.LG]

[2] Chong Chen, Weizhi Ma, Min Zhang, Zhaowei Wang, Xiuqiang He, Chenyang Wang, Yiqun Liu, and Shaoping Ma. 2021. Graph Heterogeneous Multi-Relational Recommendation. In *AAAI*.

[3] Fahao Chen, Peng Li, Toshiaki Miyazaki, and Celimuge Wu. 2022. FedGraph: Federated Graph Learning With Intelligent Sampling. *TPDS* (2022).

[4] Mingyang Chen, Wen Zhang, Zonggang Yuan, Yantao Jia, and Huajun Chen. 2021. Fede: Embedding knowledge graphs in federated setting. In *IJCKG*. 80–88.

[5] Daniel Cummings and Marcel Nassar. 2020. Structured citation trend prediction using graph neural networks. In *ICASSP*.

[6] Tahani A Daghistani, Radwa Elshawi, Sherif Sakr, Amjad M Ahmed, Abdullah Al-Thwayee, and Mouaz H Al-Mallah. 2019. Predictors of in-hospital length of stay among cardiac patients: a machine learning approach. *International journal of cardiology* 288 (2019), 140–147.

[7] Tingting Dan, Yang Li, Ziwei Zhu, Xijie Chen, Wuxiu Quan, Yu Hu, Guihua Tao, Lei Zhu, Jijin Zhu, Yuyan Jin, et al. 2020. Machine Learning to Predict ICU Admission, ICU Mortality and Survivors' Length of Stay among COVID-19 Patients: Toward Optimal Allocation of ICU Resources. In *BIBM*.

[8] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *KDD*.

[9] Tao-yang Fu, Wang-Chien Lee, and Zhen Lei. 2017. Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning. In *CIKM*.

[10] Chaoyang He, Keshav Balasubramanian, Emir Ceyani, Carl Yang, Han Xie, Lichao Sun, Lifang He, Liangwei Yang, Philip S. Yu, Yu Rong, Peilin Zhao, Junzhou Huang, Murali Annavaram, and Salman Avestimehr. 2021. FedGraphNN: A Federated Learning System and Benchmark for Graph Neural Networks. arXiv:2104.07145 [cs.LG]

[11] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. 2020. Heterogeneous graph transformer. In *WWW*.

[12] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and S Yu Philip. 2021. A survey on knowledge graphs: Representation, acquisition, and applications. *TNNLS* (2021).

[13] Song Jiang, Bernard Koch, and Yizhou Sun. 2021. HINTS: Citation Time Series Prediction for New Publications via Dynamic Heterogeneous Information Network Embedding. In *WWW*.

[14] Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. MIMIC-III, a freely accessible critical care database. *Scientific data* 3, 1 (2016), 1–9.

[15] Qinbin Li, Zeyi Wen, Zhaomin Wu, Sixu Hu, Naibo Wang, Yuan Li, Xu Liu, and Bingsheng He. 2021. A Survey on Federated Learning Systems: Vision, Hype and Reality for Data Privacy and Protection. (2021).

[16] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. 2020. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine* 37 (2020), 50–60.

[17] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. 2020. On the Variance of the Adaptive Learning Rate and Beyond. In *ICLR*.

[18] Yulong Lu and Jianfeng Lu. 2020. A universal approximation theorem of deep neural networks for expressing probability distributions. In *NeurIPS*.

[19] Linhao Luo, Yixiang Fang, Xin Cao, Xiaofeng Zhang, and Wenjie Zhang. 2021. Detecting Communities from Heterogeneous Graphs: A Context Path-based Graph Neural Network Model. In *CIKM*.

[20] Qingsong Lv, Ming Ding, Qiang Liu, Yuxiang Chen, Wenzheng Feng, Siming He, Chang Zhou, Jianguo Jiang, Yuxiao Dong, and Jie Tang. 2021. Are we really making much progress? Revisiting, benchmarking and refining heterogeneous graph neural networks. In *KDD*.

[21] 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*.

[22] Guangxu Mei, Ziyu Guo, Shijun Liu, and Li Pan. 2019. SGNN: A Graph Neural Network Based Federated Learning Approach by Hiding Structure. In *ICBD*.

[23] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.

[24] Barbara Plank and Reinard van Dalen. 2019. CiteTracked: A Longitudinal Dataset of Peer Reviews and Citations.. In *BIRNDL@ SIGIR*.

[25] 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*.

[26] Chuan Shi, Yitong Li, Jiawei Zhang, Yizhou Sun, and S Yu Philip. 2016. A survey of heterogeneous information network analysis. *TKDE* (2016).

[27] Elena Sügis, Jerome Dauvillier, Anna Leontjeva, Priit Adler, Valerie Hindie, Thomas Moncion, Vincent Collura, Rachel Daudin, Yann Loe-Mie, Yann Herault, et al. 2019. HENA, heterogeneous network-based data set for Alzheimer's disease. *Scientific data* 6 (2019), 1–18.

[28] Yizhou Sun and Jiawei Han. 2012. Mining heterogeneous information networks: principles and methodologies. *Synthesis Lectures on KDD* 3 (2012), 1–159.

[29] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S Yu, and Tianyi Wu. 2011. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *VLDB* 4 (2011), 992–1003.

[30] Yizhou Sun, Jiawei Han, Peixiang Zhao, Zhijun Yin, Hong Cheng, and Tianyi Wu. 2009. Rankclus: integrating clustering with ranking for heterogeneous information network analysis. In *EDBT*.

[31] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. ArnetMiner: Extraction and Mining of Academic Social Networks. In *KDD*.

[32] Marc Vidal, Michael E Cusick, and Albert-László Barabási. 2011. Interactome networks and human disease. *Cell* 144 (2011), 986–998.

[33] Binghui Wang, Ang Li, Hai Li, and Yiran Chen. 2020. GraphFL: A Federated Learning Framework for Semi-Supervised Node Classification on Graphs. arXiv:2012.04187 [cs.LG]

[34] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous graph attention network. In *WWW*.

[35] Xiao Wang, Nian Liu, Hui Han, and Chuan Shi. 2021. Self-supervised heterogeneous graph neural network with co-contrastive learning. In *KDD*.

[36] Chuhan Wu, Fangzhao Wu, Yang Cao, Yongfeng Huang, and Xing Xie. 2021. FedGNN: Federated Graph Neural Network for Privacy-Preserving Recommendation. arXiv:2102.04925 [cs.IR]

[37] Han Xie, Jing Ma, Li Xiong, and Carl Yang. 2021. Federated graph classification over non-iid graphs. In *NeurIPS*.

[38] Semih Yagli, Alex Dytso, and H Vincent Poor. 2020. Information-theoretic bounds on the generalization error and privacy leakage in federated learning. In *SPAWC*.

[39] Carl Yang, Yuxin Xiao, Yu Zhang, Yizhou Sun, and Jiawei Han. 2020. Heterogeneous network representation learning: A unified framework with survey and benchmark. *TKDE* (2020).

[40] Xiao Yu, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, and Jiawei Han. 2014. Personalized entity recommendation: A heterogeneous information network approach. In *WSDM*.

[41] Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. 2019. Graph transformer networks. In *NeurIPS*.

[42] Ke Zhang, Carl Yang, Xiaoxiao Li, Lichao Sun, and Siu Ming Yiu. 2021. Subgraph federated learning with missing neighbor generation. In *NeurIPS*.

[43] Jun Zhou, Chaochao Chen, Longfei Zheng, Huiwen Wu, Jia Wu, Xiaolin Zheng, Bingzhe Wu, Ziqi Liu, and Li Wang. 2021. Vertically Federated Graph Neural Network for Privacy-Preserving Node Classification. arXiv:2005.11903 [cs.LG]

[44] Ligeng Zhu, Zhijian Liu, and Song Han. 2019. Deep Leakage from Gradients. In *NeurIPS*.

[45] Xinghua Zhu, Jianzong Wang, Zhenhou Hong, and Jing Xiao. 2020. Empirical studies of institutional federated learning for natural language processing. In *EMNLP*.

[46] Zhihua Zhu, Xinxin Fan, Xiaokai Chu, and Jingping Bi. 2020. HGCN: A Heterogeneous Graph Convolutional Network-Based Deep Learning Model Toward Collective Classification. In *KDD*.

A. PROOF FOR THEOREM 3.1

Theorem 3.1 (Modeling Meta-paths with A Composition of R Functions). *For a heterograph H defined in Section 3.1 with R types of relations, we assume there is an oracle function $\hat{O}$ that takes in a target node v's meta-paths information $\mathcal{M}_v \in \mathbb{R}^d$ on H, and outputs the v's ground-truth label $y_v \in \mathbb{R}^d$. When $\mathcal{M}_v$ is absolutely continuous with respect to the Lebesgue measure, for any given approximation error ε and R functions $\{F_r | r \in [R]\}$, there exists a composition function $Comp(\cdot|\{F_r|r \in [R]\}) : \mathbb{R}^d \to \mathbb{R}^d$, which is viewed as the gradient function of an FNN $u(\cdot) : \mathbb{R}^d \to \mathbb{R}$ with ReLU activation, of depth $L = [\log_2 n]$ and width $N = 2L$, where $n = O(\frac{1}{\varepsilon^d})$. For the 1-Wasserstein distance measurement $W_1(\cdot,\cdot)$, we have* $\mathbb{E}_{\mathcal{M}_v \sim H}[W_1(\hat{O}(\mathcal{M}_v), Comp(\mathcal{M}_v|\{F_r|r \in [R]\})] < \varepsilon$.

Proof Theorem 3.1 can be obtained by properly revising Theorem 2.1 in [18], which we state as follows.

Lemma .1 (Theorem 2.1 in [18]). *Let P and Q be the target and the source distributions respectively, both defined on $\mathbb{R}^d$. Assume that $\mathbb{E}_{x\sim P}||x||^3$ is bounded and Q is absolutely continuous with respect to the Lebesgue measure. It holds that for any given approximation error ε, setting $n = O(\frac{1}{\varepsilon^d})$, there is a fully connected and feed-forward deep neural network $u(\cdot)$ of depth $L = [\log_2 n]$ and width $N = 2L$, with d inputs and a single output and with ReLU activation such that for 1-Wasserstein distance measurement, $W_1(P, \nabla u(Q)) < \varepsilon$ holds. Here, $\nabla u(\cdot)$ is the function $\mathbb{R}^d \to \mathbb{R}^d$ induced by the gradient of u while $\nabla u(Q)$ is the distribution that is generated from the distribution Q through the mapping $\nabla u(\cdot)$.*

With setting $P = \hat{O}(\mathcal{M}_v)$ and $Q = \mathcal{M}_v$, where $\mathcal{M}_v$ is absolutely continuous with respect to the Lebesgue measure. Obviously, $\hat{O}(\mathcal{M}_v)$, as a set of label vectors, is absolute a bounded 3-order moment. Thus, by setting the $Comp(\cdot|\{F_r|r \in [R]\}) : \mathbb{R}^d \to \mathbb{R}^d$ as the gradient function of the feed-forward deep neural network $u(\cdot)$ in Lemma A.1, Theorem 3.1 is proved.

For a better empirical performance, our case adopts R type-aware FNNs for the R F_r functions in Theorem 3.1 and composes these FNNs by T-GCN. We argue that the design of T-GCN can approximate any higher-order meta-paths' message passing functions.

B. DEFINITION OF ψ IN THEOREM 3.2

Definition B.1 (Revisiting the Lemma 1 in [38]). *Let $(X, Y) \sim P_{XY}$, $(\bar{X}, \bar{Y}) \sim P_X P_Y$, and $\Lambda_{f(\bar{X},\bar{Y})}(\lambda) = \log \mathbb{E}\left[e^{\lambda(f(\bar{X},\bar{Y}) - \mathbb{E}[f(\bar{X},\bar{Y})])}\right]$ denote the cumulative generating function of $f(\bar{X}, \bar{Y})$. For $b_+ \in (0, \infty]$, if we can find a convex function $\psi_+ : [0, b_+) \to \mathbb{R}$ with $\psi_+(0) = \psi'_+(0) = 0$ satisfying $\Lambda_{f(\bar{X},\bar{Y})}(\lambda) \le \psi_+(\lambda)$ for $\lambda \in [0, b_+)$, then*

$$-\psi_+^{*-1}(I(X;Y)) \le \mathbb{E}[f(\bar{X}, \bar{Y})] - \mathbb{E}[f(X, Y)],$$

where ψ_+^{*-1} denotes the inverse of the Legendre dual of ψ_+.

Similarly, for $b_- \in (0, \infty]$, if we can find a convex function $\psi_- : [0, b_-) \to \mathbb{R}$ with $\psi_-(0) = \psi'_-(0) = 0$ satisfying $\Lambda_{f(\bar{X},\bar{Y})}(\lambda) \le \psi_-(-\lambda)$ for $\lambda \in (-b_-, 0]$, then

$$\mathbb{E}[f(\bar{X}, \bar{Y})] - \mathbb{E}[f(X, Y)] \le \psi_-^{*-1}(I(X;Y)).$$

C. PROOF FOR THEOREM 4.2

Theorem 4.2 (Generalization Bound of FedHG+). *For the system defined in Section 3.2, the FL process of training the joint model for a data owner via FedHG+ requires the communication among M data owners. We denote each data owner $D_i (i \in [M])$ has an independent training set retrieved from the mended graph H'_i as $S'_i = \{(H'_i(v), y_v) | v \in V_i^s\}$, where $\mathcal{H}'_i(v)$ is the local ego-heterograph of node v drawn from the mended sub-heterograph H'_i, and y_v is v's label. Similarly, we denote the training set retrieved from the original sub-heterographs (with incomplete neighborhoods) as $S_i = \{(H_i(v), y_v) | v \in V_i^s\}$. With denoting optimal global weights as $\widehat{W}_i^*$ retrieved from training on the ego-heterographs sampled from the global heterograph (with complete neighborhoods), FedHG+'s generalization error, i.e.,* $\text{gen}\left(\mathbb{E}_{S'_{i,v}}; FedHG+\right)$, *is given by*

$$-\mathbb{E}[\frac{1}{|V^s|}\sum_{i\in[M]}\sum_{v\in V_i^s}\psi_+^{*-1}\left(I\left(S'_{i,v}; \widehat{W}_i^*\right)\right)] \le \text{gen}\left(\mathbb{E}_{S'_{i,v}}; FedHG+\right)$$
$$\le \mathbb{E}[\frac{1}{|V^s|}\sum_{i\in[M]}\sum_{v\in V_i^s}\psi_-^{*-1}\left(I\left(S'_{i,v}; \widehat{W}_i^*\right)\right)],$$

and we have $\psi_-^{*-1}\left(I\left(S'_{i,v}; \widehat{W}_i^*\right)\right) \le \psi_-^{*-1}\left(I\left(S_{i,v}; \widehat{W}_i^*\right)\right)$, namely the upper bound is tighter than that of Theorem 3.2.

Proof We first need to justify that FedHG+ is an instance of the described FL process in Theorem 3 of [38]. FedHG+ that sets the number of users as M, chooses all M users at each round of training, fixes the non-negative loss function as the loss defined in Eq. (11), and instantiates the fusion function as a sample-based normalized FedAvg, is obviously an instance of FL defined in [38].

Therefore, with respectively substituting the general FL framework to FedHG in Theorem 3 of [38], we have the generalization bound for FedHG+ in Theorem 4.2 proved.

Next, we prove the tighter upper bound of FedHG+, when compared to FedHG. We first denote the training set containing the same training nodes while retrieved from the global heterograph (with complete neighbors) as $S_i^+ = \{(H_i^+(v), y_v) | v \in V_i^s\}$. And then we denote all training data retrieved from the global complete heterograph as $S^+ = \{S_i^+\}_{i\in[M]}$.

Given the sampling probability of a node $v \in V_i^s$ in a local sub-heterograph H_i as $P(S_{i,v})$, we have $P(S_{i,v}) = P(S'_{i,v}) = P(S^+_{i,v})$. Then we have the mutual information (MI) between $P(S_{i,v})$ and the generalized model's weights as $\widehat{W}^*$ as $I(P(S_{i,v}), \widehat{W}^*)$. By denoting the joint distribution of $S_{i,v}$ and $\widehat{W}^*$ as $P(S_{i,v}, \widehat{W}^*)$, we have $I(P(S_{i,v}, \widehat{W}^*) \propto P(S_{i,v}, \widehat{W}^*) \log P(S_{i,v}, \widehat{W}^*)$.

As $\widehat{W}^*$ is obtained by $\widehat{W}^* \leftarrow \arg\min_W \ell(\cdot; S^+)$, we have $I(P(S^+_{i,v}, \widehat{W}^*) \ge I(P(S_{i,v}, \widehat{W}^*)$. By mending the original impaired sub-heterograph with generated neighbors during FedHG+, similarly, we have $I(P(S'_{i,v}, \widehat{W}^*) \ge I(P(S_{i,v}, \widehat{W}^*)$.

Referring to the Definition E.1 that ψ_-^{*-1} is defined as the inverse of the Legendre dual of a convex function ψ_-. With $\psi_- : [0, b_-) \to \mathbb{R}$ having $\psi_-(0) = \psi'_-(0) = 0$ and satisfying $\Lambda_{f(\bar{X},\bar{Y})}(\lambda) \le \psi_-(-\lambda)$ for $\lambda \in (-b_-, 0]$, we have $\psi_-^{*-1}\left(I\left(S'_{i,v}; \widehat{W}_i^*\right)\right) \le \psi_-^{*-1}\left(I\left(S_{i,v}; \widehat{W}_i^*\right)\right)$.

Denoting $I\left(S_{i,v}; \widehat{W}_i^*\right) = \mathbb{I}$ and $I\left(S'_{i,v}; \widehat{W}_i^*\right) = \mathbb{I}'$ we have

$$\mathbb{E}[\frac{1}{|V^s|}\sum_{i\in[M]}\sum_{v\in V_i^s}\psi_-^{*-1}(\mathbb{I})] \le \mathbb{E}[\frac{1}{|V^s|}\sum_{i\in[M]}\sum_{v\in V_i^s}\psi_-^{*-1}(\mathbb{I}')].$$

Hence we conclude our proof.