Subgraph Federated Learning with Missing Neighbor Generation

Ke Zhang1,4 , Carl Yang1□ , Xiaoxiao Li2 , Lichao Sun3 , Siu Ming Yiu4

摘要

由于图对现实世界对象及其交互的独特表示，因此已广泛用于数据挖掘和机器学习。随着现在图越来越大，经常看到它们的子图被单独收集并存储在多个本地系统中。因此，很自然地会考虑子图联合学习设置，其中每个本地系统都持有一个可能偏离整个图的分布的小子图。因此，子图联邦学习旨在协作训练一个强大且可泛化的图挖掘模型，而无需直接共享它们的图数据。在这项工作中，针对子图联邦学习的新颖而现实的设置，我们提出了两种主要技术：（1）FedSage，它训练基于 FedAvg 的 GraphSage 模型，以在多个局部子图上集成节点特征、链接结构和任务标签；(2) FedSage+，它沿着 FedSage 训练一个缺失的邻居生成器来处理跨局部子图的缺失链接。四个具有合成子图联邦学习设置的真实世界图数据集的实证结果证明了我们提出的技术的有效性和效率。同时，对其在全局图上的泛化能力提出了一致的理论启示。

关键词：图学习；联邦学习

1 引言

随着图数据越来越多，其子图通常被收集并存储在多个设备节点中，而每个设备节点中的子图与全图之间存在分布偏差 (bias)。这篇文章研究如何在节点之间无需进行数据通信的情景下训练一个具有足够性能及泛化能力的图学习模型，提出以下两种模型：

• FedSage：基于 FedAvg 训练 GraphSage 模型学习聚合学习多个局部子图的节点特征、边结构以及任务标签；

• FedSage+：设计 missing neighbor generator 学习生成多个局部子图之间可能缺失的边，从而促进 FedSage 训练。

2 相关工作

此部分对课题内容相关的工作进行简要的分类概括与描述，二级标题中的内容为示意，可按照行文内容进行增删与更改，若二级标题无法对描述内容进行概括，可自行增加三级标题，后面内容同样如此，引文的 bib 文件统一粘贴到 Ref 文件夹下的 Collection.bib 中并采用如下引用方式[1]。

2.1 情景导入

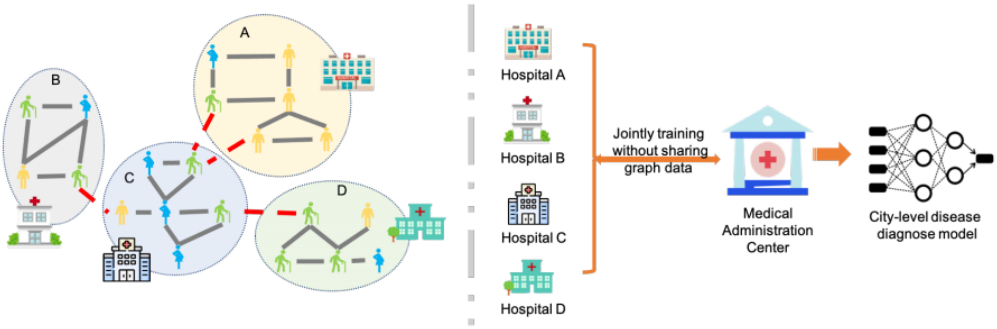

图 1: 情景示例图

在图 1 所示的健康系统中，各个医院各自存储医疗数据，当大规模流行病爆发时可通过图学习模型进行有效地传染预测。但各个医院的医疗数据通常并不共享，因此如何在不进行数据共享的情况下保证图学习模型在多子图下的有效性和泛化性值得研究。

2.2 存在的挑战

挑战一：如何联合学习多个局部子图？

全局图被分布到一组具有异构特征和结构分布的小子图中。在每个子图上训练单独的图挖掘模型可能无法捕获全局数据分布，并且也容易出现过度拟合。此外，目前还不清楚如何将多个图挖掘模型集成到一个普遍适用的模型中，以处理来自底层全局图的任何查询。

解决方案 1：FedSage：使用 FedAvg 训练 GraphSage。

挑战二：如何处理跨局部子图的缺失链接？

不同于 CV、NLP 等其他领域的分布式系统，图像和文本的数据样本是孤立、独立的，而图的数据样本是连通的、相关的。最重要的是，在子图联邦学习系统中，每个子图中的数据样本都可能与其他子图中的数据样本有联系。然而，这些连接承载着节点邻域的重要信息，并充当数据所有者之间的桥梁，但从未被任何数据所有者直接捕获。

解决方案 2：FedSage+：在 FedSage 的基础上，引入 NeighGen，生成缺失的邻居。

3 问题描述

联邦学习系统包括一个中央服务器 (central server)S 及 M 个数据端 (data owner)，中央服务器只保留一个图学习模型并不存储数据。

给定全局图 $G=\{V,E,X\}$，$G_i=\{V_i,E_i,X_i\}$ 为数据端 D_i 所保存的子图，其中 V 为节点集合，E 为边集合，X 为节点特征，$i\in[M]$.

假设 $V=V_1\cup\cdots\cup V_M$，各个数据端没有重叠节点，即 $\forall i,j\in[M],V_i\cap V_j=\oslash$；数据端 D_i 无法直接获取 D_j 中的节点 $u\in V_j$.

图 G 中任意节点 $v\in V$ 具有特征 $x\in X$ 及用于下游任务的标签 $y_v\in Y$(如节点分类)，数据端 D_i 只能获取 V_i 中的节点特征与标签。

对于图中节点 v，其对应 $ego-graph$ 为 $G(v)$，且 $(G(v),y_v)\ D_G$.

目标：设计一个联邦学习框架，在无需数据共享情况下从所有独立数据端中学习一个全局节点分类器 F，优化目标为：

$$\phi^* = arg\ \min R\left(F\left(\phi\right)\right) = \frac{1}{M}\sum_i^M R_i\left(F_i\left(\phi\right)\right)$$

其中 R_i 为数据端 D_i 的局部经验风险：

$$R_i\left(F_i\left(\phi\right)\right) := E_{(G_i,Y_i)\ D_{G_i}}\left[l\left(F_i\left(\phi;G_i\right),Y_i\right)\right]$$

其中 l 则为下游任务对应的损失函数:

$$l := \frac{1}{|V_i|}\sum_{v\in V_i} l\left(\phi;G_i\left(v\right),y_v\right)$$

4 挑战一解决方案——FedSage

4.1 基本设置

联邦学习框架为 FedAvg 框架，节点分类器 F 为 K 层 GraphSage 模型 (全局共享更新，但在各个数据端独立训练)，参数为 $\phi = \left\{\phi^k\right\}_{k=1}^K$，$\phi^k$ 为数据端 D_i 的局部参数。

在子图 G_i 上的聚合公式为：

$$h_v^k = \sigma\left(\phi^k \cdot \left(h_v^{k-1}||Agg\left(\left\{h_v^{k-1},\forall u\in N_{G_i}\left(v\right)\right\}\right)\right)\right)$$

节点分类损失函数为 Softmax：

$$\mathcal{L}^c = l\left(\phi|G_i\left(v\right),y_v\right) = CE\left(y_v,y_v\right) = -\left[y_v\log y_v + \left(1-y_v\right)\log\left(1-y_v\right)\right]$$

4.2 算法流程

训练轮数为 e_c，对于第 t 轮：

1. 个数据端 D_i 进行梯度下降更新：$\phi_i \leftarrow \phi - \eta\nabla l\left(\phi|\left\{G_i\left(v\right),y_v\right\}|v\in V_i^t\right)$，其中 $V_i^t \subseteq V_i$，y_v 为 v 的真实标签，η 为学习率；

2. 中央服务器收集所有数据端更新后的参数 $\{\phi_i|i\in[M]\}$;

3. 中央服务器对所有数据端的参数取平均得到 ϕ;

4. 中央服务器广播 θ_c 到所有数据端，数据端更新参数为 ϕ。

5 挑战二解决方案——FedSage+

FedSage 没有考虑各个子图之间潜在的边，分类器 F 无法学习到真实的全局数据分布。

解决方法: 让模型自己学习预测潜存的边——Missing Neighbor Generator (NeighGen)

5.1 Missing Neighbor Generator (NeighGen)

5.1.1 NeighGen 结构

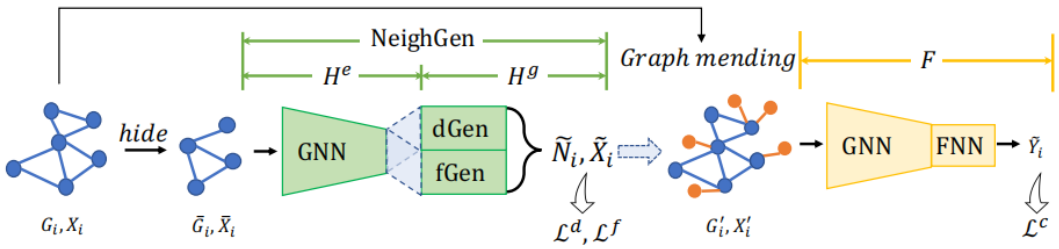

图 2: NeighGen 结构

如图 2 所示，NeighGen 包含两部分：encoder H^e 和 generator H^g:

H^e：K 层 GraphSage 模型，参数量为 θ^e，以 $\bar{G}_i$ 作为输入。

H^g：包含 dGen 和 fGen 两部分，均为全连接神经网络 (FNN)：

dGen: 线性回归模型，参数为 θ_d，用于预测节点的缺失邻居数 $\tilde{n}_i$;

fGem: 特征生成模型，参数为 θ_f，用于生成 $\tilde{n}_i$ 个特征向量，记为集合 $\tilde{X}_i$，生成过程如下：

$\tilde{n}_i = \sigma\left(\left(\theta^d\right)^T \cdot n_v\right), \tilde{x}_v = R\left(\sigma\left(\left(\theta^f\right)^T \cdot \left(z_v + N\left(0,1\right)\right)\right), \tilde{n}_v\right)$

其中 $N\left(0,1\right)$ 为高斯噪声，R 为随机采样器，目的是为单个节点生成多样的邻居特征。

5.1.2 图模拟修复 (Graph mending simulation)

假设：每个数据端只与部分其它数据端的部分节点存在缺失的边。

方法：在每个子图 G_i 中随机删除 h% 的节点 $V_i^h \subset V_i$ 以及与 V_i^h 中节点相连的边 E_i^h，生成新的子图 $\bar{G}_i = \left(\bar{V}_i, \bar{E}_i, \bar{X}_i\right)$。以破坏后的图 $\bar{G}_i$ 作为输入，删除的节点 V_i^h 和 E_i^h 作为 ground-truth 训练 NeighGen，优化目标为：

$\mathcal{L}^n = \lambda^d \mathcal{L}^d + \lambda^f \mathcal{L}^f$

5.1.3 邻居生成 (Neighbor Generation)

NeighGen 根据破坏后的图 $\bar{G}_i$ 及对应 ground-truth 训练，学习生成邻居，输出修补后的图 G_i'。

5.2 GraphSage+NeighGen 局部联合学习——LocSage+

当定义好恢复缺失边的 NeighGen 模型后，在单个子图上训练 NeighGen 和 GraphSage 以进行节点分类任务。GraphSage 以 NeighGen 修复后的图 G_i' 作为输入，联合学习的优化损失函数为上述各部分损失的加权和：

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

5.3 GraphSage+NeighGen 联邦学习—FedSage+

由于 LocSage+ 只在单个子图上训练，所学习的信息受限于单个子图并具有偏差，并且只能生成当前子图内的边，无法有效生成子图之间缺失的边，如何解决这一问题？在这一点上，作者提出使用联邦学习训练 NeighGen。

NeighGen 目的是为每个子图预测缺失的多条边帮助训练全局分类器 F，而全局共享的 NeighGen 无法处理不同子图上的多样性。因此作者为每个数据端 D_i 训练一个 $NeighGen_i$。为了预测子图间缺失的边，在 $fGen_i$ 中添加一个跨子图 (cross-subgraph) 的特征重构损失 $\mathcal{L}_i^f$.

5.4 FedSage+ 算法流程

Algorithm 1 FedSage+: Subgraph federated learning with missing neighbor generation

Notations. Data owners set $\{D_1, \ldots, D_M\}$, server S, epochs for jointly training NeighGen e_g, epochs for FedSage e_c, learning rate for FedSage η.
For $t = 1 \to e_g$, we iteratively run **procedure A**, **procedure C**, **procedure D**, and **procedure E**
Every $D_i \in \{D_1, \ldots, D_M\}$ retrieves G'_i from FL trained NeighGen$_i$
S initializes and broadcasts ϕ
For $t = 1 \to e_c$, we iteratively run **procedure B** and **procedure F**

On the server side:
procedure A SERVEREXCECUTIONFORGEN(t) ▷ FL of NeighGen on epoch t
  Collect $(Z_i^t, H_i^g) \leftarrow$ LOCALREQUEST(D_i, t) from every data owner D_i, where $i \in [M]$
  Send $\{(Z_j^t, H_j^g) | j \in [M] \setminus \{i\}\}$ to every data owner D_i, where $i \in [M]$
  for $D_i \in \{D_1, \ldots, D_M\}$ **in parallel do**
    $\{\nabla \mathcal{L}_{i,1}^f, \ldots, \nabla \mathcal{L}_{i,M}^f\} \setminus \{\nabla \mathcal{L}_{i,i}^f\} \leftarrow$ FEEDFORWARD$(D_i, \{(Z_j^t, H_j^g) | j \in [M] \setminus \{i\}\})$
  for $D_i \in \{D_1, \ldots, D_M\}$ **in parallel do**
    Aggregate gradients as $\nabla \mathcal{L}_{i,J}^f \leftarrow \sum_{j \in [M] \setminus \{i\}} \nabla \mathcal{L}_{i,j}^f$
    Send $\nabla \mathcal{L}_{i,J}^f$ to D_i for UPDATENEIGHGEN$(D_i, \nabla \mathcal{L}_{i,J}^f)$
procedure B SERVEREXCECUTIONFORC(t) ▷ FedSage for mended subgraphs on epoch t
  Collect $\phi_i \leftarrow$ LOCALUPDATEC(D_i, ϕ, t) from all data owners
  Broadcast $\phi \leftarrow \frac{1}{M} \sum_{i \in [M]} \phi_i$

On the data owners side:
procedure C LOCALREQUEST(D_i, t) ▷ Run on D_i
  Sample a $V_i^t \in \bar{V}_i^t$ and get $Z_i^t \leftarrow \{H_i^e(\bar{G}_i(v)) | v \in V_i^t\}$
  Send Z_i^t, H_i^g to Server
procedure D FEEDFORWARD$(D_i, \{(Z_j^t, H_j^g) | j \in [M] \setminus \{i\}\})$ ▷ Run on D_i
  for $j \in [M] \setminus \{i\}$ **do**
    $\mathcal{L}_{j,i}^f \leftarrow \frac{1}{|Z_j^t|} \sum_{z_v \in Z_j^t} \sum_{p \in [|H_j^g(z_v)|]} \left(\min_{u \in V_i} (||H_j^g(z_v)^p - x_u||_2^2)\right)$ ▷ A part of Eq. (6)
  Compute and send $\{\nabla \mathcal{L}_{1,i}^f, \ldots, \nabla \mathcal{L}_{M,i}^f\} \setminus \{\nabla \mathcal{L}_{i,i}^f\}$ to Sever
procedure E UPDATENEIGHGEN$(D_i, \nabla \mathcal{L}_{i,J}^f)$ ▷ Run on D_i
  Train NeighGen$_i$ by optimizing Eq. (6).
procedure F LOCALUPDATEC(D_i, ϕ, t) ▷ Run on D_i
  Sample a $V_i^t \subseteq V_i$
  $\phi_i \leftarrow \phi - \eta \nabla l(\phi | \{(G'_i(v), y_v) | v \in V_i^t\})$
  Send ϕ_i to Sever

图 3: FedSage+ 算法伪代码

FedSage+ 算法流程如图 3 所示。

6 复现过程

6.1 数据集与实验设置比

采用四个数据集：Cora、Citeseer、PubMed、MSAcademic；

使用 Louvain 算法将每个数据集划分为 3、5、10 个簇三种情况，每个簇大小相似；

训练-验证-测试集划分比例为 60

GraphSage 层数为 2，采用平均聚合，batchsize 设置为 64，每一层节点采样数为 5；

子图的节点删除比例 $h\% \in [3.4\%, 27.8\%]$，λ 参数均设置为 1；

Adam 优化器，学习率为 0.001，训练轮数为 50。

6.2 与已有开源代码对比

该论文在 GitHub 平台有开源的第三方代码，我主要复现的过程是先将该代码进行理解并跑通，然后进行优化，最后得出优化后的实验结果，再与原本的实验结果进行对比，

6.3 实验结果对比

	Cora			Citesser		
Model	M=3	M=5	M=10	M=3	M=5	M=10
LocSage	0.5762 (±0.0302)	0.4431 (±0.0847)	0.2798 (±0.0080)	0.6789 (±0.054)	0.5612 (±0.086)	0.4240 (±0.0859)
LocSage+	0.5644 (±0.0219)	0.4533 (±0.047)	0.2851 (±0.0080)	0.6848 (±0.0517)	0.5676 (±0.0714)	0.4323 (±0.0715)
FedSage	0.8656 (±0.0043)	0.8645 (±0.0050)	0.8626 (±0.0103)	0.7241 (±0.0022)	0.7226 ±0.0066)	0.7158 (±0.0053)
FedSage+	**0.8686** (±0.0054)	**0.8648** (±0.0051)	**0.8632** (±0.0034)	**0.7454** (±0.0038)	**0.7440** (±0.0025)	**0.7392** (±0.0041)
GlobSage	0.8701 (±0.0042)			0.7561 (±0.0031)		
	PubMed			MSAcademic		
Model	M=3	M=5	M=10	M=3	M=5	M=10
LocSage	0.8447 (±0.0047)	0.8039 (±0.0337)	0.7148 (±0.0951)	0.8188 (±0.0331)	0.7426 (±0.0790)	0.5918 (±0.1005)
LocSage+	0.8481 (±0.0041)	0.8046 (±0.0318)	0.7039 (±0.0925)	0.8393 (±0.0330)	0.7480 (±0.0810)	0.5927 (±0.1094)
FedSage	0.8708 (±0.0014)	0.8696 (±0.0035)	0.8692 (±0.0010)	0.9327 (±0.0005)	0.9391 (±0.0007)	0.9262 (±0.0009)
FedSage+	**0.8775** (±0.0012)	**0.8755** (±0.0047)	**0.8749** (±0.0013)	**0.9359** (±0.0005)	**0.9414** (±0.0006)	**0.9314** (±0.0009)
GlobSage	0.8776(±0.0011)			0.9681(±0.0006)		

图 4: 论文结果

		Cora				Citesser	
Model	M=3	5	10		3	5	10
LocSage	0.5494333	0.41012	0.30706		0.6342667	0.54046	0.41019
LocSage+	0.6157	0.40478	0.30625		0.6292	0.54556	0.40428
FedSage	0.869	0.8665	0.8394		0.7564	0.7346	0.7646
FedSage+	0.8644	0.8702	0.8422		0.7526	0.7368	0.7661
GlobSage		0.8638667				0.7386	
		PubMed				MSAcademic	
Model	M=3	5	10		3	5	10
LocSage	0.8274333	0.79176	0.71743		0.79506667	0.75341	0.59051
LocSage+	0.83086667	0.78828	0.72257		0.8120333	0.7352	0.58173
FedSage	0.8797	0.8794	0.8917		0.9237	0.936667	0.91312
FedSage+	0.8871	0.8834	0.8931		0.9278	0.9375667	0.931
GlobSage		0.8675				0.9327	

图 5: 复现的结果

图 4 是论文中的结果，图 5 是我跑通代码后优化的结果，其中颜色为红色的数据是提升较大的地方，但大部分结果都在原文的所预期望的范围以内。

7 总结与展望

在本次课程研究中，前期我通过阅读相关的基础知识，了解了图联邦学习的相关内容，然后选择一篇优秀的论文（出自 NeurIPS 2021）进行精读。在此过程中，我一开始是计算从零开始复现，但尝试了几周后，我仍完成不过，主要原因在于个人并没有写过联邦学习和图神经网络的相关代码，因此

后期我选择在他人的基础上进行复现，但这并不是轻松的工作，我需要在没有注释的情况下读懂他人的代码，然后思考可以优化的地方，尽管最后的优化结果并不理想，但我认为我在此过程中同样学习到很多知识，最起码对如何写图联邦学习的代码有了一些了解，在未来的工作中，我认为可以进一步对代码进行优化，还可以尝试不同的数据集测验泛化效果。

参考文献

[1] ZHANG K, YANG C, LI X, et al. Subgraph Federated Learning with Missing Neighbor Generation [J/OL]., 2021. https://arxiv.org/abs/2106.13430. DOI: 10.48550/ARXIV.2106.13430.