Research Proposal

Zheng Huang

November 2022

1 Introduction

In recent years, graph mining has been widely used to understand rich information hidden in graphs. Graph Neural Networks (GNNs) have emerged as state-of-the-art for graph mining and show impressive performance in various real-world applications, e.g., healthcare [1, 2], bioinformatics [3, 4] and recommender systems [5, 6], to name a few.

To deal with the need for a huge number of graph data and the regularization of user privacy [7] when training GNN models, Federated Graph Machine Learning (FGML) [8, 9] is proposed. FGML is a distributed machine learning scheme that trains a graph model across different participating devices/silos without sharing private data. While providing a promising paradigm, FGML faces several challenges in realistic scenarios. For example, residents of a city may go to different banks for various reasons. Their personal data, such as demographics, income conditions, and transaction activities (interactions) can be collected by bank branches. When approving loan requests, a central bank administration (or central server) can utilize a graph model trained on the entire bank network to evaluate better if a loan applicant is benign. However, due to conflicts of interest, a bank may reluctant to share its user networks with others. Thus, the first challenge is that the local subgraphs collected from different devices/silos are limited and may miss critical information. What's worse, a graph model trained on the corrupted network plus the bias introduced by devices/silos selection [10] could lead to undesired discrimination against users from certain demographic subgroups (e.g., age, gender, and race) and would mistakenly reject a loan of a user that belongs to an underprivileged group. As a result, how to alleviate the bias in FGML is another challenge.

As an attempt to address the challenges, we propose an FGML framework named **EGRESS** (f**E**derated **G**ene**R**ativ**E** GNN with **S**tructural debia**S**ing). The overall structure is shown in Fig.1. Firstly, to deal with the challenge of missing information (e.g., node and feature) across local subgraphs, we collaboratively train a **Generator Module** to generate missing neighbors as well as features for nodes on each subgraph and form an augmented local subgraph. Specifically, each subgraph first holds out some nodes and trains the generator based on the held-out neighbors. The gradients of the generator are then aggregated with ones on other subgraphs in a federated fashion. The output augmented graph is later fed into a local GNN model trained with FedAvg [8]. In addition, to tackle the second challenge, we propose a **Dibiasing Module** where we use a debiasing strategy with the help of GNNExplainer [11]. To be more specific, we employ a Bias Explainer to identify edges that maximally account for the exhibited node-level bias given a node's computation graph in a device/silo. Similarly, another Fairness Explainer can be defined by identifying the edges whose presence can maximally alleviate the node-level bias. We employ a sensitive bias metric based on Wasserstein distance [12] in the probabilistic outcome of GNN prediction since the probabilistic space can better preserve the exhibited bias [13, 14].

2 Problem Setup

Preliminaries. We mainly focus on cross-silo FGML in this proposal. Let $D = \{d_1, d_2, ..., d_m\}$ be

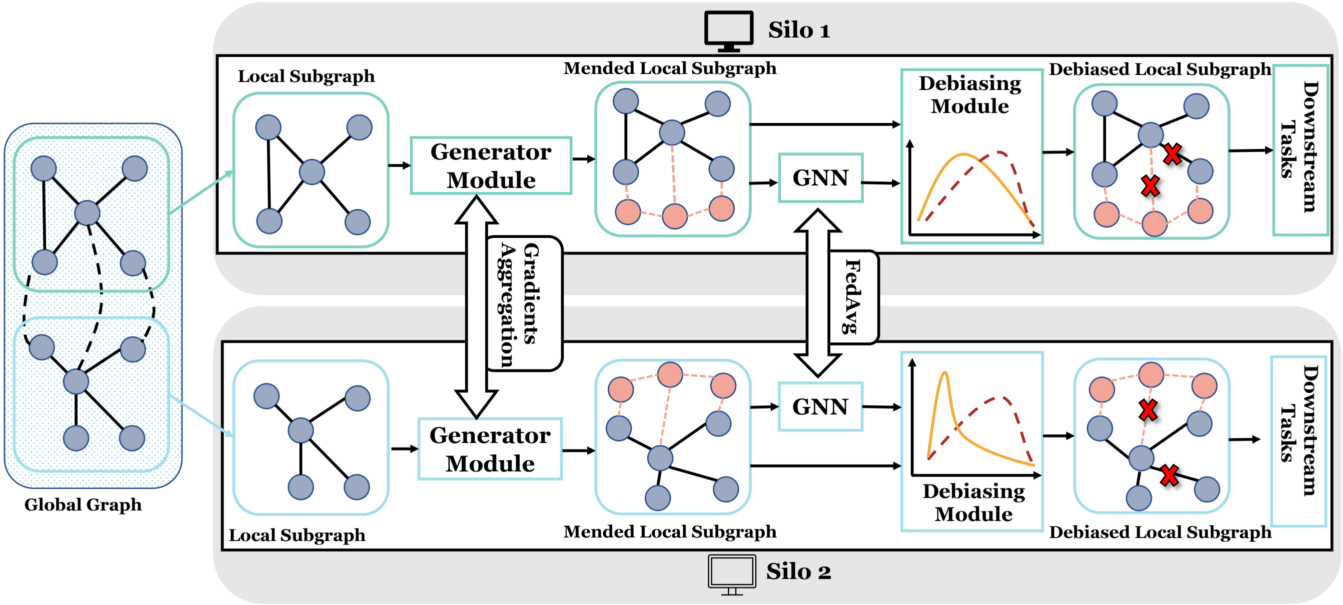

Figure 1: Overall Structure of EGRESS

a set of M data owners. We denote a global graph as $G = \{V, E, X\}$, where $\{V\}$ is the node set, $\{X\}$ is the corresponding node feature set, and E is the edge set. Under the FGML system, we have a central server S to conduct Gradients Aggregation as well as FedAvg, and M data owners with their corresponding local subgraph $G_i = \{V_i, E_i, X_i\}, \forall i \in [D]$. In addition, any data owner D_i cannot directly retrieve data from another data owner D_j. We assume $V = V_1 \cup V_2 \cup ... \cup V_M$. Besides, we have missing edges existing in reality but not stored in the whole system. Specifically, for an edge $e_{v,u} \in E$, where $v \in V_i, u \in V_j$ and $e_{v,u} \notin (E_i \cup E_j)$ (shown as the dash line in Global Graph in Fig. 1).

Goal. We mainly focus on node classification. The goal of the proposed framework is to collaboratively learn on isolated subgraphs in all data owners, without raw graph data sharing, to obtain a global graph mining model (e.g., GNN) [15]. For the global graph $G = \{V, E, X\}$, every node $v \in V$ has its feature $x_v \in X$ and label $y_v \in Y$ which is a d_y-dimensional one-hot vector.

3 Proposed Method

In this section, we first present the methodology to train the **Generator Module**, which can generate mended subgraphs under the FGML system. Secondly, we demonstrate how to address the bias introduced by the FGML system and mended subgraphs with the **Debiasing Module**.

3.1 Generator Module

To deal with the challenge of missing node and feature across local subgraphs, we proposed a Generator Module. For each silo, we have a Generator Module including a multi-layer GNN encoder H^e and a generative model H^g.

H^e. The output of our GNN encoder is the embeddings for nodes, $Z_i = \{z_v | z_v \in \mathbb{R}^{dz}, v \in V_i\}$.

H^g. For each silo, we feed the output of the GNN encoder Z_i into the generative model. It contains a MLP, H_1^g, that aims to predict the numbers of missing neighbors for each node $\widetilde{N} = \{\widetilde{n}_v | \widetilde{n}_v \in \mathbb{N}, v \in V_i\}$, and another MLP, H_2^g, that generates a set of $\widetilde{N}$ feature vectors $\widetilde{X}_i = \{\widetilde{x}_v | x \in \mathbb{R}^{\widetilde{n}_v \times d_x}, \widetilde{n}_v \in \widetilde{N}, v \in V_i\}$.

Objective Function. To train the Generator Module, we first randomly hold out $h\%$ of nodes $V_i^h \subset V$ and corresponding edges and features in the local subgraph G_i, denoted as $\overline{G}_i = \{\overline{V}_i, \overline{E}_i, \overline{X}_i\}$, where $\overline{V}_i$, $\overline{E}_i$ and $\overline{X}_i$ is the impaired node set, edge set, and feature set, respectively. Thus H_1^g can be optimized by

$$L_g^1 = \frac{1}{|\overline{V}_i|} \sum_{v \in \overline{V}_i} L_1^S(\widetilde{n}_v - n_v),$$

where L_1^S is the smooth L1 distance [16]. Then, H_2^g can be jointly optimized by Gradients Aggregation [17]. Specifically, for node $v \in \overline{V}_i$, we ensure that the generated features in silo i and j should be close to the feature of v's hided neighbor. Thus, mathematically, H_2^g can be trained with

$$\begin{aligned} L_g^2 =& \frac{1}{|\overline{V}_i|} \sum_{v \in \overline{V}_i} \sum_{p \in [\widetilde{n}_v]} \Big(\min_{u \in \mathbf{N}_{\mathbf{G_i}}(v) \cap V_i^h} (||\widetilde{x}_v^p - x_u||) \\ &+ \alpha \sum_{j \in [M] \backslash i} \min_{u \in V_j} (||H_2^g(z_v)^p - x_u||) \Big), \end{aligned}$$

where $\mathbf{N}_{\mathbf{G_i}}(\mathbf{v})$ is the neighbor of node v in the local graph G_i and the feature of node $u \in \mathbf{N}_{\mathbf{G_i}}(\mathbf{v}) \cap \mathbf{V_i^h}$ provides ground truth for the generated feature in the first term $min_{u \in \mathbf{N}_{\mathbf{G_i}}(v) \cap V_i^h}(||\widetilde{x}_v^p - x_u||)$. In addition, the model gradients of loss term $\sum_{j \in [M]/i} min_{u \in V_j}(||H_2^g(z_v)^p - x_u||)$ can be calculated from silo D_j. Finally, the gradients are weighted by α and aggregated by silo D_i.

Based on the Generator Module, we feed the mended graph into a GNN model that can be trained by FedAvg in FGML. Thus, the Generator Module and a GNN classifier can be jointly trained by

$$L = \lambda_1 L_c + \lambda_2 L_g^1 + \lambda_3 L_g^2,$$

where L_c is a cross-entropy loss of the GNN model for node classification, and $\lambda_1, \lambda_2, \lambda_3$ are hyperparameters.

Table 1: Dataset Statistics.

	Global Graph	Silo1	Silo2	Silo3
#V	1000	367	254	379
#E	24970	7150	4911	7649

3.2 Debiasing Module

To alleviate the bias introduced by the generative model [18] and the Federated Learning system [19], also to get a more fair result, we utilize a node-level Debiasing Module that includes a Bias Explainer and Fairness Explainer inspired by GNNExplainer. The output of this module is the debiased local subgraph as shown in Fig.1. Please note that for simplicity we ignore the silo notation for the following sections.

Objective Function. We use $\hat{Y}$ to represent the outcome of the GNN model in a silo, and $\widetilde{y}_i$ to represent the probabilistic outcome of the GNN model with fixed parameters based on the computation graph G_c^i for node v_i. We replace $\hat{y}_i$ in the GNN outcome with $\widetilde{y}_i$, and the outcome set can then be denoted as $\widetilde{Y}$, e.g., $\widetilde{Y} = \hat{Y} \backslash \{\hat{y}_i\} \cup \{\widetilde{y}_i\}$. We then split $\widetilde{Y}$ into two sets based on the sensitive group (e.g., gender) as $\widetilde{Y}_0$ and $\widetilde{Y}_1$, also the corresponding distribution is $P(\widetilde{Y}_0)$ and $P(\widetilde{Y}_1)$. If $\hat{y}_i$ is replaced, then the distribution distance between the two sensitive subgroups will also be changed accordingly. Thus, the Biased Explainer aims to derive $\widetilde{y}_i$ that can maximize the distribution distance between $P(\widetilde{Y}_0)$ and $P(\widetilde{Y}_1)$. This can be achieved by training a mask to identify edges $\widetilde{E}_i$ in the computation graph of node v_i, and $\widetilde{E}_i$ are supposed to lead to a larger distribution distance. Thus, the objective of the Biasd Explainer based on Wasserstein-1 distance (W1) as

$$\max_{\widetilde{E}_i} W1(P(\widetilde{Y}_0), P(\widetilde{Y}_1)),$$

where $\widetilde{E}_i$ is the edge set given by Bias Explainer. We adopt a common-used approximation strategy [20, 14] to optimize Wasserstein-1 distance with SGD.

Table 2: Preliminary Experimental Results.

	Accuracy	**Wasserstein Distance**	**Wasserstein Distance'**	**Statistical Party**	**Equal Opportunity**
Global Graph	0.7040	0.1487	0.1386	0.0309	0.0106
Silo1	0.6892	0.3901	0.3822	0.0481	0.0659
Silo2	0.6965	0.2221	0.2199	0.0491	0.0849
Silo3	0.6940	0.2867	0.2842	0.0338	0.0447

Similarly, the goal of Fairness Explainer is to train another mask to identify an edge set $\widetilde{E}_i'$ in the computation graph of node v_i, and $\widetilde{E}_i'$ can minimize the distribution distance. Thus, by following the similar method of Bias Explainer, $\widetilde{y}_i'$ is derived from the Fairness Explainer based on its computation graph $G_c^{i\prime}$, and $P(\widetilde{Y}_0')$ and $P(\widetilde{Y}_1')$ denote the distribution of two sensitive groups by replacing $\hat{y}_i$ with $\widetilde{y}_i'$, ($\widetilde{Y}' = \hat{Y}\backslash\{\hat{y}_i\} \cup \{\widetilde{y}_i'\}$). Therefore, we demonstrate the goal of the Fairness Explainer as

$$\underset{\widetilde{E}_i'}{min}\, W1(P(\widetilde{Y}_0'), P(\widetilde{Y}_1')),$$

Based on the above methods, the two Explainers can be jointly optimized with

$$L_e = W1(P(\widetilde{Y}_0'), P(\widetilde{Y}_1')) - W1(P(\widetilde{Y}_0), P(\widetilde{Y}_1)).$$

Apart from that, the two explanations given by Bias and eneccbvffrvuvgvtlbencbnrkvibbbekvednklbhclh Fairness Explainers should incorporate the critical information of the original prediction $\hat{Y}_i$. This constraint can be achieved by maximizing the mutual information [11] between the corresponding computation graph $G_c^i/G_c^{i\prime}$ and $\hat{Y}_i$:

$$L_m = -\mathbb{E}_{\hat{Y}_i|G_c^i}[logP_\Theta(\hat{Y}_i|G_c^i)] - \mathbb{E}_{\hat{Y}_i|G_c^{i\prime}}[logP_\Theta(\hat{Y}_i|G_c^{i\prime})].$$

Thus, the two explainers can be jointly trained by

$$L = \lambda_4 L_e + \lambda_5 L_m + \lambda_6 L_r,$$

where $\lambda_4, \lambda_5, \lambda_6$ are hyperparameters and L_r is the regularization term to ensure the sparsity of the masks of the two explainers.

4 Experiments

In this section, we demonstrate the experiments of our proposed framework on the graph classification task. We first introduce settings, and then the experimental results.

4.1 Experimental Settings

Dataset. We mainly focus on the German Credit dataset. Nodes and edges represent the bank clients and the connections between client accounts, respectively. To simulate the scenario of FGML, we cut the graph of German Credit into three silos (from silo1 to silo3) with Louvain algorithm [21]. The statistics of the dataset are shown in Table 1, where $\#V$ means the number of nodes and $\#E$ means the number of edges.

Evaluation Metrics. We use four metrics in the experiments. **Accuracy** can be used to gauge the model performance. The node-level bias can be measured by **Wasserstein-1 distance**, the lower, the better. **Statistical Party** [22] and **Equal Opportunity** [23] are two traditional fairness metrics.

4.2 Experimental Results

The preliminary experimental results are demonstrated in Table 2.

- The row of the Global Graph means the result is calculated on the graph that is not split into different parts. It sets the upper bound of the experiment. Regarding accuracy, our proposed framework achieves comparable results.
- With respect to the node-level bias, the column **Wasserstein Distance** denotes the results acquired before the Debiasing Module. Compared

with the result of Global Graph, the FGML system and the generative model definitely introduces node-level bias, which proves the necessity of our Debiasing Module.

- **Wasserstein Distance'** shows the node-level bias after the Debiasing Module. The reduction of node-level bias can be observed in Global Graph and all silos. This indicates that our proposed Wasserstein distance-based objective functions effectively help to identify edges that introduce bias.

- **Statistical Party** and **Equal Opportunity** also demonstrate the node-level bias after our Debiasing Module. The comparable results can be observed with respect to **Statistical Party**. However, we still have room to improve on the metric of **Equal Opportunity**.

5 Future Works

In this section, I present my thoughts on future works inspired by the experiments.

- **Framework.** The framework can be further refactored in the future. The Generator Module is originally proposed to deal with problems in the medical area, thus, we could redesign the framework to make it fit our scenario. In addition, this framework contains codes based on both Pytorch and TensorFlow. We are not sure if the fuse of these two libraries can cause the degradation of performance. Thus, it is necessary to modify the framework in a unified way as it benefits both code sharing and future experiments.

- **Data Heterogeneity.** When I conducted the experiments, I observed that the performance as well as the node-level bias highly depend on the data on each silo (e.g., node and edge number). If the data gap is huge, for example, one silo contains 100 nodes but another 500, the results will degrade so much. More research on dealing with heterogeneous data while assuring fairness is needed.

- **Generative Model.** To deal with the data heterogeneity, from my perspective, we can resort to generative models. Also the appearance of diffusion models [24, 25] on CV area provides us a powerful way to utilize probabilistic generative models. There might be potential in the FGML system. Also, some augmentation methods like LAGNN [26] could be used in the Federated Learning scenario.

- **Trustworthy FGML.** In our framework, the debiasing strategy is not trained with a Federated Learning fashion, since in my opinion, we want to acquire a mask tailored for each silo. It makes no sense to aggregate bias information from other silos. However, there might be other ways to maintain fairness in FGML, e.g., try other fairness notions apart from group fairness.

6 Conclusion

In a nutshell, we propose a framework called **EGRASS** to utilize a Generator Module in a distributed subgraph system. The Debias Module is then used to alleviate the node-level bias introduced by a generative model and the federated system. Experimental results evidence the necessity of our proposed framework and point out some future directions we can pursue.

References

[1] Z. Wang, R. Wen, X. Chen, S. Cao, S.-L. Huang, B. Qian, and Y. Zheng, “Online disease diagnosis with inductive heterogeneous graph convolutional networks,” in *Proceedings of the Web Conference 2021*, pp. 3349–3358, 2021.

[2] K. Zhang, C. Yang, X. Li, L. Sun, and S. M. Yiu, “Subgraph federated learning with missing neighbor generation,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 6671–6682, 2021.

[3] K. Zhang, Y. Wang, H. Wang, L. Huang, C. Yang, and L. Sun, “Efficient federated learn-

ing on knowledge graphs via privacy-preserving relation embedding aggregation," *arXiv preprint arXiv:2203.09553*, 2022.

[4] X.-M. Zhang, L. Liang, L. Liu, and M.-J. Tang, "Graph neural networks and their current applications in bioinformatics," *Frontiers in genetics*, vol. 12, 2021.

[5] H. Chen, L. Wang, Y. Lin, C.-C. M. Yeh, F. Wang, and H. Yang, "Structured graph convolutional networks with stochastic masks for recommender systems," in *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 614–623, 2021.

[6] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, and D. Yin, "Graph neural networks for social recommendation," in *The world wide web conference*, pp. 417–426, 2019.

[7] P. Voigt and A. Von dem Bussche, "The eu general data protection regulation (gdpr)," *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, vol. 10, no. 3152676, pp. 10–5555, 2017.

[8] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*, pp. 1273–1282, PMLR, 2017.

[9] X. Fu, B. Zhang, Y. Dong, C. Chen, and J. Li, "Federated graph machine learning: A survey of concepts, techniques, and applications," *arXiv preprint arXiv:2207.11812*, 2022.

[10] A. Abay, Y. Zhou, N. Baracaldo, S. Rajamoni, E. Chuba, and H. Ludwig, "Mitigating bias in federated learning," *arXiv preprint arXiv:2012.02447*, 2020.

[11] Z. Ying, D. Bourgeois, J. You, M. Zitnik, and J. Leskovec, "Gnnexplainer: Generating explanations for graph neural networks," *Advances in neural information processing systems*, vol. 32, 2019.

[12] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International conference on machine learning*, pp. 214–223, PMLR, 2017.

[13] E. Dai and S. Wang, "Say no to the discrimination: Learning fair graph neural networks with limited sensitive attribute information," in *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pp. 680–688, 2021.

[14] Y. Dong, S. Wang, Y. Wang, T. Derr, and J. Li, "On structural explanation of bias in graph neural networks," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 316–326, 2022.

[15] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[16] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448, 2015.

[17] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for federated learning on user-held data," *arXiv preprint arXiv:1611.04482*, 2016.

[18] A. Grover, J. Song, A. Kapoor, K. Tran, A. Agarwal, E. J. Horvitz, and S. Ermon, "Bias correction of learned generative models using likelihood-free importance weighting," *Advances in neural information processing systems*, vol. 32, 2019.

[19] Y. H. Ezzeldin, S. Yan, C. He, E. Ferrara, and S. Avestimehr, "Fairfed: Enabling group fairness in federated learning," *arXiv preprint arXiv:2110.00857*, 2021.

[20] M. Cuturi and A. Doucet, "Fast computation of wasserstein barycenters," in *International conference on machine learning*, pp. 685–693, PMLR, 2014.

[21] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of statistical mechanics: theory and experiment*, vol. 2008, no. 10, p. P10008, 2008.

[22] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel, "Fairness through awareness," in *Proceedings of the 3rd innovations in theoretical computer science conference*, pp. 214–226, 2012.

[23] M. Hardt, E. Price, and N. Srebro, "Equality of opportunity in supervised learning," *Advances in neural information processing systems*, vol. 29, 2016.

[24] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in Neural Information Processing Systems*, vol. 33, pp. 6840–6851, 2020.

[25] L. Yang, Z. Zhang, Y. Song, S. Hong, R. Xu, Y. Zhao, Y. Shao, W. Zhang, B. Cui, and M.-H. Yang, "Diffusion models: A comprehensive survey of methods and applications," *arXiv preprint arXiv:2209.00796*, 2022.

[26] S. Liu, R. Ying, H. Dong, L. Li, T. Xu, Y. Rong, P. Zhao, J. Huang, and D. Wu, "Local augmentation for graph neural networks," in *International Conference on Machine Learning*, pp. 14054–14072, PMLR, 2022.