Supplemental Material: A Robust Detection and Correction Framework for GNN-based Vertical Federated Learning

Zhicheng Yang[1,2], Xiaoliang Fan[1,2(✉)], Zheng Wang[1,2], Zihui Wang[1,2], and Cheng Wang[1,2]

[1] Fujian Key Laboratory of Sensing and Computing for Smart Cities, School of Informatics, Xiamen University, 361005, P.R. China.
[2] Key Laboratory of Multimedia Trusted Perception and Efficient Computing, Ministry of Education of China, Xiamen University, 361005, P.R. China.
{zcyang, zwang, wangziwei}@stu.xmu.edu.cn
{fanxiaoliang, cwang}@xmu.edu.cn

A Notation Table

Table 1. The definitions of symbols

symbol	definition
$G = (\mathcal{V}, \mathcal{E}, \mathbf{X})$	the original graph with nodes set $\mathcal{V}$, edges set $\mathcal{E}$ and feature matrix $\mathbf{X}$
A	the adjacency matrix
$\mathbf{x}$	the sample / node feature
y	the sample / node label
N	the number of nodes in graph
M	the number of participants
f	the local GNN model in GNN-based Vertical Federated Learning
f_0	the global classifier model
Agg	the aggregation function of the global model
$\mathbf{h}$	the feature embedding
i	the index of participants
j, v	the index of nodes
m	the index of malicious participant
$\mathcal{D}_{val} = (\mathcal{V}_{val}, Y_{val})$	validation dataset with N_{val} nodes and labels

B Detailed Experiment Setting

Datasets. We use three benchmark datasets, i.e., Cora, Cora_ML[3] and Pubmed [4]. We use the same dataset partition for training, validation, and testing as described in the prior work[2]. Following [1], the node features were divided equally among each participant, and each participant retains the complete graph structure to prevent the fragmentation of numerous isolated nodes.

Evaluation. To mitigate the impact of randomness, we let different participants act as the malicious participant in turn and run each experiment with 3 random seeds. To evaluate the effectiveness of identifying malicious participants, we measure the average detection rate of malicious participants. To evaluate the robustness of each method, we compute the average model accuracy when there exist malicious participants.

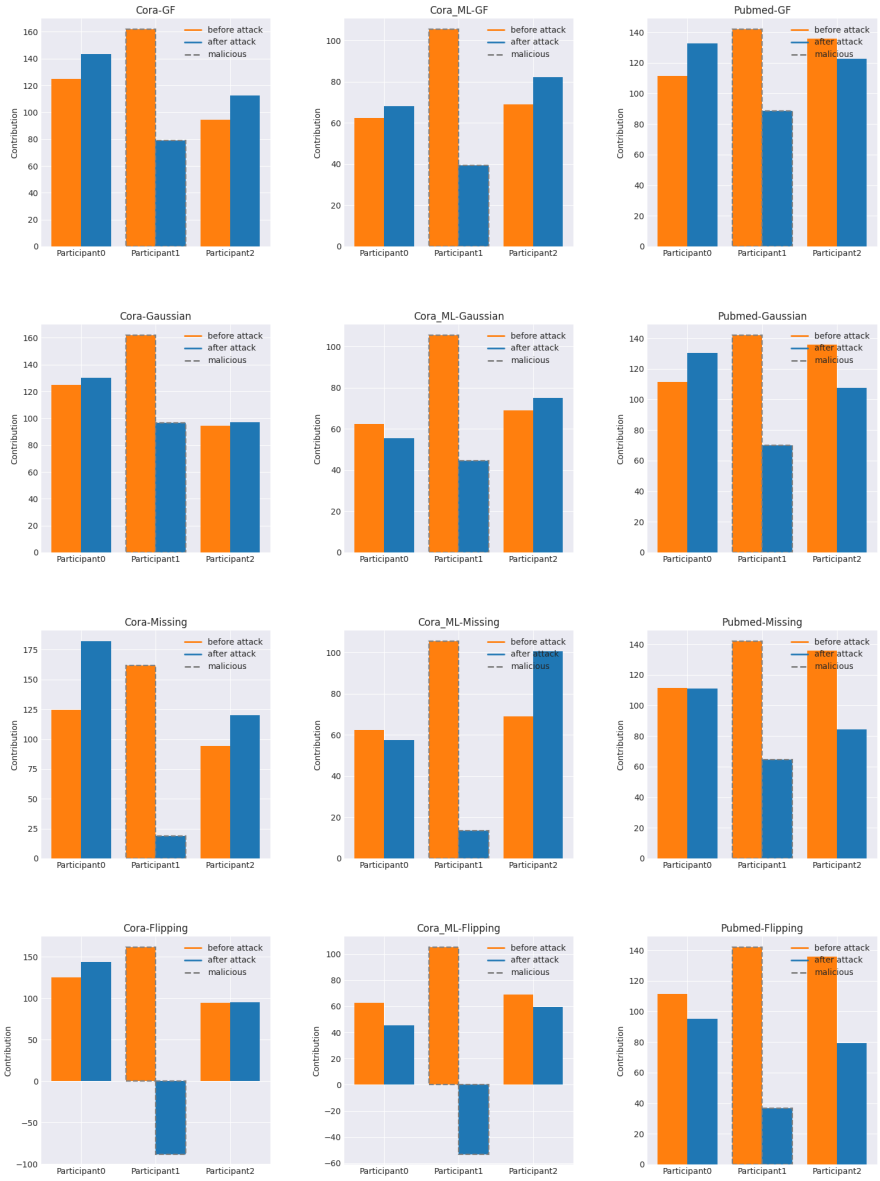
Parameter Settings. We use GCN[2] and SGC[5] as the local GNN model. For each local GNN model, a two-layer GNN model is applied to extract the local node embeddings, whose dimension is 16 and the number of hidden units is 32. Besides, the activation function is ReLU. The GVFL is trained for 200 epochs using Adam with learning rate of 0.01. Considering the stealthiness of the attack, the attack budget Δ is fixed to 1.

Implementation. All our experiments are run on a 64 GBRAM Ubuntu 18.04.6 server with Intel(R) Xeon(R) CPU E5-2630 v4 @ 2.20GHz and 4 NVidia(R) 2080Ti GPUs. All code is implemented in PyTorch 1.12.0.

C Visualization

After demonstrating the effectiveness of our algorithm, we conducted a deeper exploration of detection in GVFL. In order to explain the reasons behind the effectiveness of our defense, we plotted the contribution of each participant before and after the attack in the case of three participants using three graph datasets. The results are illustrated in Figure 1. From the figure, we made the following observations:

- **The contribution of the compromised participant to the global model is significantly reduced**: While each participant's contribution to the global model may vary in the absence of an adversary, the presence of the malicious participant consistently leads to a significant decrease in the contribution of the compromised participant, often reaching the minimum level. This phenomenon is evident across various attacks. For example, under the GF attack with Cora dataset, participant1 initially contributes 161 to the model, but after being compromised, their contribution drops to only 79.5, which is the lowest among the three participants.
- **Different attacks have different effects on reducing participant contributions**: The impact on reducing the contribution of the malicious participant varies depending on the attack type. The more effective the attack, the more significant the decrease in the contribution of the malicious participant, making it easier for RDC-GVFL to identify them. As an example, in the GCN setting, the Flipping attack is the most harmful to the model, resulting in the contribution of the participant1 becoming negative immediately after being compromised. Conversely, the Gaussian attack is less effective, resulting in only a minor change in the participant1's contribution.



(a) Cora (b) Cora-ML (c) Pubmed

Fig. 1. Visualization of participant contribution in GVFL before and after various adversarial attacks on participant1, where the number of participant is 3.

As concluded from our experiments, RDC-GVFL demonstrates the ability to identify the malicious participant in the majority of cases, with a higher de-

tection rate for more effective attacks. Therefore, these observations align with our experimental results. Based on these findings, we believe that the occasional failure of RDC-GVFL to identify a malicious participant may be attributed to the attack not being sufficiently effective.

D Complete Defense Performance

Table 2 shows each defense performance against various attacks in GVFL where the local model adopted is SGC.

Table 2. The accuracy (%) of each defense under five adversarial attack scenarios on multiple datasets with a varying number of participants.

Local Model	Dataset	M	Attack	Defense			
				Unsecured	Krum	Ours w/o Cor.	Ours
SGC	Cora	2	Oracle	**77.90**	60.53	65.63	71.03
			GF	61.37	43.80	63.08	**68.58**
			Gaussian	37.20	45.78	63.08	**68.58**
			Missing	63.10	37.31	63.08	**68.58**
			Flipping	21.93	31.11	63.08	**68.58**
		3	Oracle	**77.73**	73.13	73.13	74.60
			GF	65.17	60.25	70.33	**72.58**
			Gaussian	52.97	70.33	70.33	**72.58**
			Missing	70.33	62.56	70.33	**72.58**
			Flipping	41.17	64.53	70.33	**72.58**
		4	Oracle	**77.53**	73.86	73.86	74.73
			GF	68.57	63.90	72.01	**72.68**
			Gaussian	54.97	72.01	72.01	**72.68**
			Missing	72.03	67.04	72.01	**72.68**
			Flipping	50.70	52.93	72.01	**72.68**
	Cora_ML	2	Oracle	**83.30**	73.10	73.10	78.64
			GF	65.33	55.37	72.19	**78.64**
			Gaussian	35.73	48.44	72.19	**78.64**
			Missing	72.20	43.90	72.19	**78.64**
			Flipping	24.03	37.02	72.19	**78.64**
		3	Oracle	**83.30**	72.71	77.43	80.75
			GF	71.57	59.84	76.00	**80.30**
			Gaussian	35.57	76.00	76.00	**80.30**
			Missing	76.00	59.21	76.00	**80.30**
			Flipping	42.10	46.74	76.00	**80.30**
		4	Oracle	**83.27**	77.09	77.98	81.04
			GF	74.37	75.17	77.95	**81.27**
			Gaussian	35.87	77.95	77.95	**81.27**
			Missing	77.97	66.01	77.95	**81.27**
			Flipping	53.53	50.56	77.95	**81.27**
	Pubmed	2	Oracle	**75.80**	67.56	67.56	68.23
			GF	35.30	47.30	64.35	**68.80**
			Gaussian	43.13	53.08	64.35	**68.80**
			Missing	64.37	50.55	64.35	**68.80**
			Flipping	34.77	38.13	64.35	**68.80**
		3	Oracle	**76.17**	68.93	70.06	72.13
			GF	36.97	62.71	68.32	**70.91**
			Gaussian	43.10	68.32	68.32	**70.91**
			Missing	68.33	59.97	68.32	**70.91**
			Flipping	51.80	44.70	68.32	**70.91**
		4	Oracle	**75.63**	64.83	70.80	71.16
			GF	42.47	57.27	67.47	**70.60**
			Gaussian	43.30	67.47	67.47	**70.60**
			Missing	67.40	54.42	67.47	**70.60**
			Flipping	55.67	43.70	67.47	**70.60**

References

1. Chen, J., Huang, G., Zheng, H., Yu, S., Jiang, W., Cui, C.: Graph-fraudster: Adversarial attacks on graph neural network-based vertical federated learning. IEEE Transactions on Computational Social Systems (2022)
2. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
3. McCallum, A.K., Nigam, K., Rennie, J., Seymore, K.: Automating the construction of internet portals with machine learning. Information Retrieval **3**, 127–163 (2000)
4. Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., Eliassi-Rad, T.: Collective classification in network data. AI magazine **29**(3), 93–93 (2008)
5. Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., Weinberger, K.: Simplifying graph convolutional networks. In: International conference on machine learning. pp. 6861–6871. PMLR (2019)