Article

Leakage-Aware Defense Against Training-Trace Link Inference in Vertical Federated Graph Learning

Jian Zhang[1,*] and Jiaqi Liu[1]

[1] School of Computer and Software, Southwest Petroleum University, Chengdu, China
* Correspondence: Jian Zhang, School of Computer and Software, Southwest Petroleum University, Chengdu, China.

Abstract: Vertical federated graph learning (VFGL) shares intermediate training signals that can reveal graph links even when data are partitioned. We present LAD-VFGL, an empirical defense for link inference from a feature-side representation and its backward gradient. In a feature-partitioned two-branch VFGL implementation, a trace-based risk proxy combines prior protected gradients and current feature-side embeddings. An exponential moving average (EMA) of node-level risk controls Gaussian perturbation of both the forward embedding and its backward gradient; pair scores weight a decorrelation loss, while historical EMA traces preserve task utility. Across six public graph datasets and five random seeds, the gradient-channel attack area under the receiver operating characteristic curve (AUC) decreases from 0.7852 to 0.5128, while average node-classification accuracy remains 0.8758, only 0.28 percentage points below undefended training. At the evaluated operating points, LAD-VFGL attains lower attack AUC and higher average accuracy than fixed Gaussian noise. These findings are limited to the evaluated passive attack and do not constitute a formal privacy guarantee.

Keywords: vertical federated graph learning; link inference; privacy defense; graph neural networks; leakage-aware training

1. Introduction

Graph neural networks (GNNs) exploit relational structure to learn node representations and are widely used for citation, recommendation, fraud, and social-network data [1-4]. In cross-organization applications, node attributes may be feature-partitioned, while graph connectivity and supervision need not be visible to every participant. Vertical federated learning provides a collaboration mechanism without centralizing raw records [5,6], and VFGL extends this idea to graph-structured data [7,8].

Raw-data isolation alone does not guarantee link privacy [9]. Message passing makes gradients and embeddings topology dependent, and intermediate information exposed during training can therefore encode whether two nodes are connected. Gradient leakage is already well documented in federated learning [10-12]; more recently, Zari *et al.* showed that gradients can support effective link inference in VFGL [13]. This risk matters when edges represent friendships, transactions, or other sensitive relationships.

Uniform perturbation is a simple defense but treats all training signals as equally sensitive. Link leakage is pairwise and heterogeneous: a small subset of aligned gradients or highly correlated representations may carry much stronger structural evidence than other signals. A defense should therefore spend more protection on components that are easier to exploit and preserve low-risk task information whenever possible.

We propose LAD-VFGL to turn pairwise trace-based leakage cues into node-specific perturbation of forward representations and backward gradients, together with pair-weighted representation regularization. A consistency objective preserves downstream

Received: 03 September 2026
Revised: 26 October 2026
Accepted: 08 November 2026
Published: 12 November 2026

Copyright: © 2026 by the authors. Submitted for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

utility. Six datasets and five seeds demonstrate strong mitigation under the reported gradient-channel attack, with separate diagnostics for forward and output observations.

2. Problem Setting and Threat Model

2.1. Feature-Partitioned VFGL Implementation

We consider $G = (V, E)$ with adjacency A, node attributes X, and supervision Y. The implemented two-branch simulation partitions attributes by columns, $X = [X_G, X_X]$. The graph-aware graph convolutional network (GCN) processes (X_G, A), while the feature-only multilayer perceptron (MLP) processes X_X:

$$H_G = f_{\theta_G}(X_G, A), \qquad H_X = f_{\theta_X}(X_X). \tag{1}$$

The fusion head concatenates the branch outputs and predicts labels:

$$Z = [H_G \| \tilde{H}_X], \qquad \log P = f_{\theta_C}(Z). \tag{2}$$

Here $\tilde{H}_X = H_X$ without defense and denotes the perturbed feature-side embedding under LAD-VFGL. The graph-aware branch uses A and X_G; labels Y enter the classification loss, without implying a separately deployed label server. We emulate the functional split within one process; this does not constitute a networked secure communication deployment. The modeled training interface and passive observation boundary are illustrated in Figure 1.

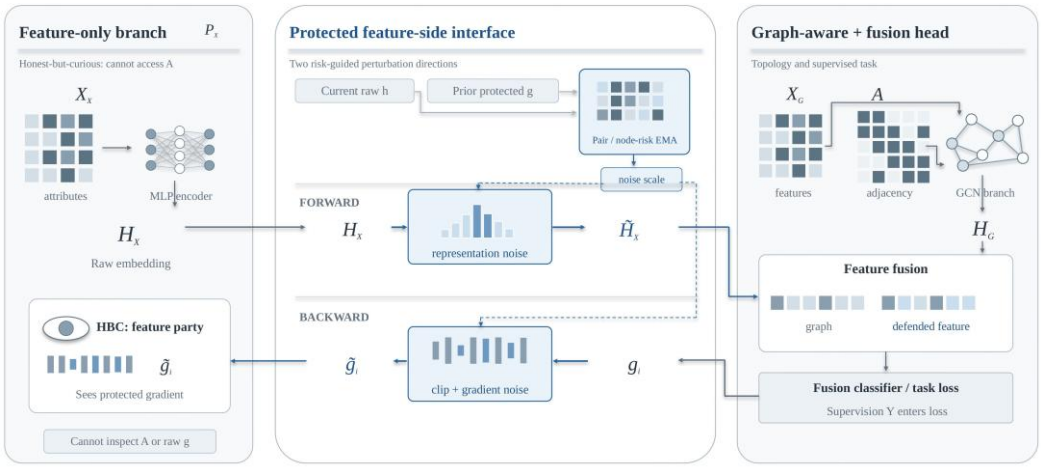


Figure 1. System and threat model of the functional VFGL split. The GCN processes (X_G, A) and the MLP processes X_X; $[H_G \| \tilde{H}_X]$ enters the fusion head. LAD perturbs the feature-side forward embedding and replaces the joint-loss interface gradient before its return to the feature-side encoder. Solid blue denotes protected signals and dashed blue denotes risk-to-noise control. This is a simulated interface, not a deployed secure communication protocol.

2.2. Observation and Attack Protocol

The principal passive adversary is the honest-but-curious feature-only party: it does not own A but can observe the feature-side embedding and the gradient *returned across its modeled backward interface*. At the modeled interface, a backward hook replaces the gradient before it reaches the feature-only branch; the attacker has no additional access path to the unprotected gradient. Risk computation and the hook are instrumented inside the single-process simulation, so this is a specified observation boundary, *not* a demonstrated cryptographic access-control guarantee. The gradient-channel attacker ranks node pairs by cosine similarity; forward-embedding and output-side probes are separately evaluated views, not necessarily a single adversary's joint access. Positive pairs are training-node edges and negatives are sampled training-node nonedges, used for evaluation but not LAD risk supervision. Epoch traces are recorded, yet the evaluator does not construct a jointly trained multi-round attacker. We report attack AUC, not differential privacy [13,14].

3. Leakage-Aware Defense

3.1. Risk Estimation from Available Traces

At epoch t, candidate pairs (i, j) are drawn uniformly from distinct training nodes, subject to a global pair cap. Risk is computed from the current raw feature-side embedding $h_i^{(t)}$ and the previously *protected* interface gradient $\tilde{g}_i^{(t-1)}$. Let $S_g^{(t-1)}(i,j)$ be rescaled cosine similarity of the *previous protected* gradients and $S_h^{(t)}(i,j)$ be rescaled cosine similarity of mean-centered *current* raw embeddings, both in $[0,1]$. The third cue $\Delta_{ij}^{(t)}$ is an exponential similarity derived from embedding distance; it does not require a separately learned probing network:

$$\Delta_{ij}^{(t)} = \exp\left(-\frac{\|h_i^{(t)} - h_j^{(t)}\|_2^2}{d\kappa}\right). \quad (3)$$

The pairwise proxy, with weights summing to one, is

$$R_{ij}^{(t)} = \mathrm{clip}_{[0,1]}\left(\alpha S_g^{(t-1)}(i,j) + \beta S_h^{(t)}(i,j) + \gamma \Delta_{ij}^{(t)}\right). \quad (4)$$

When no previous protected-gradient cache exists (the first epoch), S_g is set to zero. Each sampled pair contributes to both endpoints. Let $\mathcal{S}_i^{(t)}$ denote the indices of nodes paired with i in the sampled candidate set; then

$$R_i^{(t)} = \frac{1}{\left|\mathcal{S}_i^{(t)}\right|} \sum_{j \in \mathcal{S}_i^{(t)}} R_{ij}^{(t)}, \quad \bar{R}_i^{(t)} = \rho \bar{R}_i^{(t-1)} + (1-\rho) R_i^{(t)}. \quad (5)$$

The EMA updates only sampled nodes; unsampled nodes retain their previous values. Risk scores are detached proxy measurements, not supervised predictions of A_{ij}.

3.2. Perturbation and Risk-Weighted Regularization

The smoothed node risk drives the noise scale

$$\sigma_i^{(t)} = \sigma_{\min} + (\sigma_{\max} - \sigma_{\min})\mathrm{sigmoid}\left(\frac{\bar{R}_i^{(t)} - \tau_r}{T_r}\right). \quad (6)$$

Noise is explicitly added to the *feature-side forward representation* before concatenation, and to its *joint-loss interface gradient* during backward propagation:

$$\begin{aligned} \tilde{h}_i^{(t)} &= h_i^{(t)} + \sigma_i^{(t)} \xi_i, \\ \tilde{g}_i^{(t)} &= \mathrm{clip}\left(g_i^{(t)}, C_g\right) + \sigma_i^{(t)} \epsilon_i, \end{aligned} \quad (7)$$

where $\xi_i, \epsilon_i \sim \mathcal{N}(0, I)$ are sampled independently for the two directions, although both use the same risk-dependent $\sigma_i^{(t)}$. The clipping norm C_g is fixed. Here $g_i^{(t)}$ is the *full joint-objective* gradient at the protected interface $\tilde{h}_i^{(t)}$; it is not only the classification-loss gradient. A backward hook replaces it with $\tilde{g}_i^{(t)}$ *before* propagation into the feature-only encoder. The graph-aware branch is not subjected to this hook.

Pairwise risk weights also regularize defended feature-side embeddings:

$$\begin{aligned} \omega_{ij} &= \frac{\exp(R_{ij}/T_d)}{\sum_{(u,v)\in\mathcal{B}} \exp(R_{uv}/T_d)}, \\ \mathcal{L}_{\mathrm{dec}} &= \sum_{(i,j)\in\mathcal{B}} \omega_{ij} \cos^2\left(\tilde{h}_i, \tilde{h}_j\right). \end{aligned} \quad (8)$$

This is a loss regularizer and does not modify any original edges.

3.3. Temporal Consistency and Training Sequence

We maintain separate EMA trace buffers $H^\star$ and $P^\star$, without introducing an independently parameterized teacher network. At epoch t the buffers are detached targets for

$$\mathcal{L}_{\mathrm{util}} = \mathrm{MSE}\left(\widetilde{H}_X, H^\star\right) + \zeta \mathrm{KL}(P^\star \,\|\, P). \quad (9)$$

The trace buffers are initialized when first used. For each target/current pair $(Q^\star, Q) \in \{(H^\star, \widetilde{H}_X), (P^\star, P)\}$, the post-step update is

$$Q^\star \leftarrow m_T Q^\star + (1 - m_T)\mathrm{sg}(Q). \quad (10)$$

The temporal trace momentum m_T is independent of the node-risk EMA coefficient ρ in (5). The joint training objective is

$$\mathcal{L} = \mathcal{L}_{\mathrm{task}} + \mu\left(\mathcal{L}_{\mathrm{dec}} + \nu n^{-1} \textstyle\sum_i \mathrm{sg}\,(\sigma_i^2)\right) + \eta \mathcal{L}_{\mathrm{util}}, \quad (11)$$

where sg denotes stop-gradient. Consequently, the detached noise-size term does not contribute gradients to model updates. The protected interface gradient is $g_i = \nabla_{\tilde{h}_i}(\mathcal{L}_{\mathrm{task}} + \mu\mathcal{L}_{\mathrm{dec}} + \eta\mathcal{L}_{\mathrm{util}})$; *all three differentiable terms* are covered by the backward hook.

Risk is refreshed from current raw embeddings and the *prior* protected gradient *before* current forward perturbation, avoiding circular dependence. The risk EMA and post-step trace-target EMA follow separate update paths (Figure 2). At inference, accuracy is computed from clean, unperturbed branch outputs.

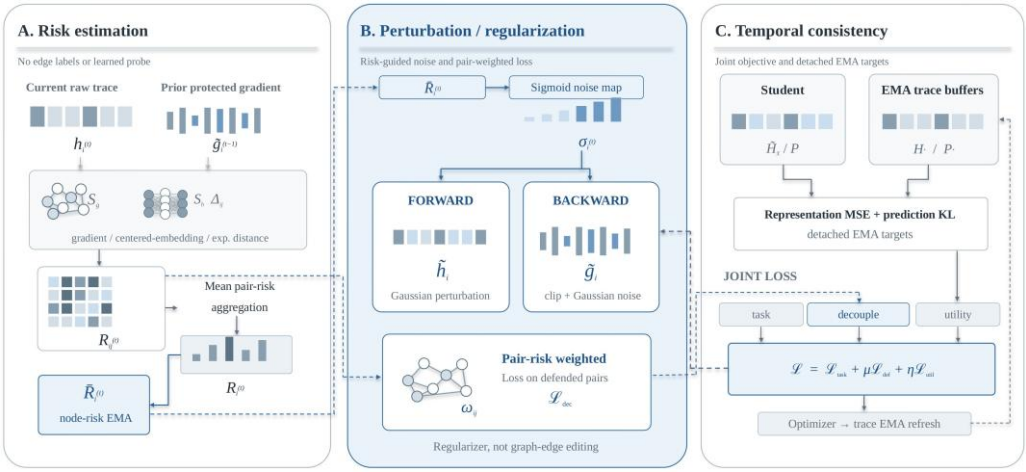


Figure 2. LAD-VFGL training workflow. The current unperturbed feature-side representation $h_i^{(t)}$ and the prior protected interface gradient $\tilde{g}_i^{(t-1)}$ produce the trace-based risk proxy. The pair score controls ω_{ij}, whereas node-risk EMA controls the shared noise scale $\sigma_i^{(t)}$ for independent forward and backward perturbations. The backward branch receives the *joint-objective* interface gradient. Detached representation/prediction EMA buffers provide utility targets; decorrelation is a loss, not graph-edge editing.

4. Experiments

4.1. Setup

We use six citation, product, and coauthor graph benchmarks: Cora, Citeseer, Pubmed, Amazon Photo, Amazon Computers, and Coauthor CS [15,16]. Their sizes and feature dimensions are listed in Table 1. The evaluated functional split divides node features between a two-layer graph-aware GCN and a two-layer feature-only MLP, followed by a fusion/classification head. Node-classification and passive link-inference results are averaged over five random seeds on all six datasets.

We compare NoDef, Dropout, representation masking (RepMask), fixed Gaussian gradient noise (UniNoise), and LAD-VFGL. We report gradient-channel attack AUC for the feature-only branch, evaluate representation/output channels separately, and measure classification accuracy, macro-averaged F1 (Macro-F1), and training time. These are reported configurations, not a matched-privacy parameter sweep or a jointly optimized multi-round attack.

Table 1. Datasets used in the experiments.

Dataset	Nodes	Edges	Feat.	Classes
Cora	2,708	5,429	1,433	7
Citeseer	3,327	4,732	3,703	6
Pubmed	19,717	44,338	500	3
Amazon Photo	7,650	119,081	745	8
Amazon Computers	13,752	245,861	767	10
Coauthor CS	18,333	81,894	6,805	15

Table 2. Gradient-channel link-inference AUC (mean ± standard deviation over five seeds). Reduction denotes the relative decrease from NoDef to LAD-VFGL.

Dataset	NoDef	Dropout	RepMask	UniNoise	LAD-VFGL	Reduction
Cora	0.7907 ± 0.0100	0.7908 ± 0.0106	0.7903 ± 0.0126	0.5258 ± 0.0045	0.5138 ± 0.0020	35.02%
Citeseer	0.7620 ± 0.0062	0.7604 ± 0.0041	0.7581 ± 0.0072	0.5251 ± 0.0033	0.5119 ± 0.0026	32.82%
Pubmed	0.7389 ± 0.0107	0.7337 ± 0.0083	0.7342 ± 0.0093	0.5256 ± 0.0025	0.5097 ± 0.0018	31.02%
Amazon Photo	0.8204 ± 0.0171	0.8245 ± 0.0135	0.8214 ± 0.0187	0.5232 ± 0.0032	0.5162 ± 0.0031	37.08%
Amazon Computers	0.7625 ± 0.0130	0.7569 ± 0.0193	0.7578 ± 0.0175	0.5250 ± 0.0033	0.5124 ± 0.0024	32.80%
Coauthor CS	0.8368 ± 0.0097	0.8329 ± 0.0130	0.8345 ± 0.0124	0.5239 ± 0.0026	0.5128 ± 0.0021	38.72%
Average	0.7852	0.7832	0.7827	0.5248	0.5128	34.69%

Table 3. Node-classification accuracy under different defenses, averaged over five seeds.

Dataset	NoDef	Dropout	RepMask	UniNoise	LAD-VFGL
Cora	0.8501	0.8483	0.8538	0.8634	0.8568
Citeseer	0.7429	0.7363	0.7318	0.7324	0.7408
Pubmed	0.8970	0.8955	0.8971	0.8747	0.8905
Amazon Photo	0.9380	0.9396	0.9400	0.9310	0.9361
Amazon Computers	0.8979	0.8994	0.9006	0.8666	0.8899
Coauthor CS	0.9454	0.9455	0.9456	0.9323	0.9407
Average	0.8786	0.8774	0.8782	0.8667	0.8758

Table 4. Average attack AUC by observation channel.

Observation channel	NoDef	LAD-VFGL
Gradient	0.7852	0.5128
Forward representation	0.7761	0.5528
Output server	0.8773	0.8731

Table 5. Ablation results averaged over six datasets (lower AUC and higher accuracy are better).

Variant	Grad. AUC ↓	Accuracy ↑
Full LAD-VFGL	0.5128	0.8758
NoRisk	0.5489	0.8710
NoLAD	0.7116	0.8790
NoUtility	0.5140	0.8639
Noise+Utility	0.5298	0.8676
Decouple+Utility	0.6845	0.8837

4.2. Privacy and Utility

Dropout and RepMask yield little improvement in gradient-channel attack AUC, whereas LAD-VFGL reduces the average from 0.7852 (NoDef) to 0.5128, a 34.69% relative decrease (Table 2). UniNoise reaches 0.5248, with LAD-VFGL attaining lower AUC on all

six datasets at the reported settings. A noise-budget sweep would be needed to compare the defenses at *matched* privacy or utility.

The privacy gain comes with a modest average accuracy reduction: LAD-VFGL retains 0.8758, only 0.28 percentage points below NoDef and 0.91 points above UniNoise (Table 3). Its mean Macro-F1 is 0.8440, compared with 0.8535 for NoDef and 0.7979 for UniNoise. The advantage is not uniform: on Cora, UniNoise achieves higher accuracy (0.8634 versus 0.8568). The evaluated privacy–utility operating points are visualized in Figure 3.

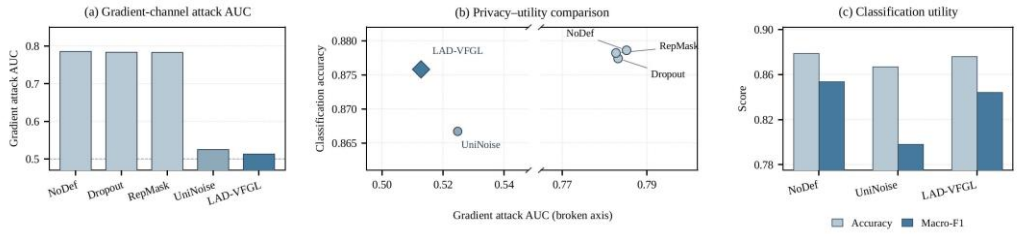


Figure 3. Privacy–utility results: (a) gradient-channel attack AUC; (b) AUC versus accuracy (broken horizontal axis); and (c) accuracy and Macro-F1. Lower AUC and higher utility are preferred; the plotted settings do not define a Pareto frontier.

The strongest reduction occurs for gradient observations, the feature party's primary attack view (Table 4). Forward-representation AUC also decreases, but output-side AUC changes only from 0.8773 to 0.8731. The evidence therefore supports training-trace leakage mitigation rather than protection against unprotected prediction outputs.

4.3. Ablation, Sensitivity, and Cost

The ablations include both joint and individual component changes (Table 5). NoLAD disables perturbation, decoupling, and utility consistency together, whereas Noise+Utility retains consistency. Thus, the effect of adaptivity versus matched fixed-strength noise remains unresolved.

The sensitivity results reveal a consistent privacy–utility trade-off (Figure 4). As μ increases, gradient attack AUC falls from 0.548 to 0.510 while accuracy declines from 0.878 to 0.864; most privacy gains occur by $\mu = 0.7$.

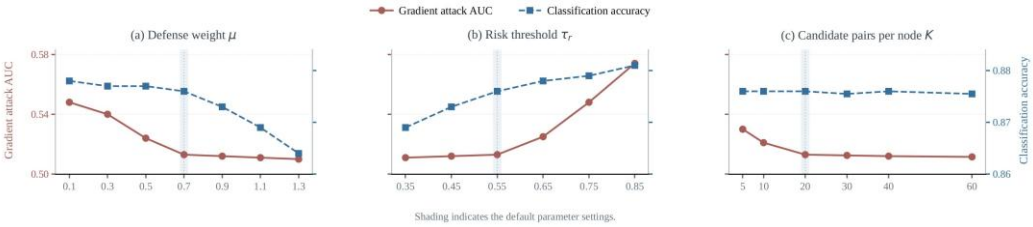


Figure 4. Parameter sensitivity: (a) defense weight μ (7 settings), (b) risk threshold τ_r (6), and (c) candidate-pair budget K (6). Solid red: gradient attack AUC; dashed blue: accuracy. Shading marks defaults.

Raising τ_r from 0.35 to 0.85 increases AUC from 0.511 to 0.574 and accuracy from 0.869 to 0.881. Increasing the pair budget K from 5 to 20 lowers AUC from 0.530 to 0.513; at $K = 60$, AUC reaches 0.5115 with accuracy near 0.876. The additional gains diminish as more candidate pairs are sampled.

Mean training overhead is 16.2% (Cora: 21.5%; Citeseer: 26.8%; Pubmed: 15.2%; Amazon Photo: 11.7%; Amazon Computers: 13.4%; Coauthor CS: 8.7%). Risk estimation and decoupling account for most of this cost.

5. Conclusion

We evaluated LAD-VFGL in feature-partitioned VFGL. Prior protected gradients and current embeddings guide forward/backward perturbation and pair-weighted regularization, while EMA trace targets support utility. Across six datasets, gradient-channel attack AUC falls from 0.7852 to 0.5128 with a 0.28 percentage-point accuracy loss, supporting the method against the evaluated passive attack. These results do not establish robustness to adaptive multi-round attacks, matched-budget optimality, or protection of released predictions.

References

1. T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in Proceedings of the International Conference on Learning Representations (ICLR), 2017.
2. W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in Advances in Neural Information Processing Systems, vol. 30, pp. 1024–1034, 2017.
3. C. He, K. Balasubramanian, E. Ceyani, Y. Rong, P. Zhao, J. Huang, M. Annavaram, and S. Avestimehr, "FedGraphNN: A federated learning system and benchmark for graph neural networks," in Proceedings of the 1st MLSys Workshop on Graph Neural Networks and Systems (GNNSys), 2021.
4. C. Wu, F. Wu, Y. Cao, L. Lyu, Y. Huang, and X. Xie, "FedGNN: Federated graph neural network for privacy-preserving recommendation," presented at the International Workshop on Federated Learning for User Privacy and Data Confidentiality (FL-ICML), 2021.
5. H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Agüera y Arcas, "Communication-efficient learning of deep networks from decentralized data," in Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS), vol. 54, pp. 1273–1282, 2017.
6. Q. Zhang, B. Gu, C. Deng, and H. Huang, "Secure bilevel asynchronous vertical federated learning with backward updating," in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, no. 12, pp. 10896–10904, 2021.
7. R. Liu, P. Xing, Z. Deng, A. Li, C. Guan, and H. Yu, "Federated graph neural networks: Overview, techniques, and challenges," IEEE Transactions on Neural Networks and Learning Systems, vol. 36, no. 3, pp. 4279–4295, 2025.
8. Z. Liu, L. Yang, Z. Fan, H. Peng, and P. S. Yu, "Federated social recommendation with graph neural network," ACM Transactions on Intelligent Systems and Technology, vol. 13, no. 4, Art. no. 55, pp. 1–24, 2022.
9. G. Mei, Z. Guo, S. Liu, and L. Pan, "SGNN: A graph neural network based federated learning approach by hiding structure," in 2019 IEEE International Conference on Big Data (Big Data), pp. 2560–2568, 2019.
10. L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," in Advances in Neural Information Processing Systems, vol. 32, pp. 14774–14784, 2019.
11. J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients: How easy is it to break privacy in federated learning?" in Advances in Neural Information Processing Systems, vol. 33, pp. 16937–16947, 2020.
12. M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning," in 2019 IEEE Symposium on Security and Privacy (SP), pp. 739–753, 2019.
13. O. Zari, C. Xu, J. Parra-Arnau, A. Ünsal, and M. Önen, "Link inference attacks in vertical federated graph learning," in 2024 Annual Computer Security Applications Conference (ACSAC), pp. 761–777, 2024.
14. M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, pp. 308–318, 2016.
15. P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad, "Collective classification in network data," AI Magazine, vol. 29, no. 3, pp. 93–106, 2008.
16. O. Shchur, M. Mumme, A. Bojchevski, and S. Günnemann, "Pitfalls of graph neural network evaluation," in Relational Representation Learning Workshop, Advances in Neural Information Processing Systems (NeurIPS), 2018.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of Publisher and/or the editor(s). Publisher and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.