G Adversarial Learning on Graph

The adversarial learning model for graph embedding [5] is illustrated as follows. Let $\mathcal{N}(v_r)$ be the node set directly connected to v_r. We denote the underlying true connectivity distribution of node v_r as the conditional probability $p(v|v_r)$, which captures the preference of v_r to connect with other nodes $v \in V$. In other words, the neighbor set $\mathcal{N}(v_r)$ can be interpreted as a set of observed nodes drawn from $p(v|v_r)$. The adversarial learning for the graph $\mathcal{G}$ is conducted by the following two modules:

Generator G: Through optimizing the generator parameters θ_G, this module aims to approximate the underlying true connectivity distribution and generate (or select) the most likely nodes $v \in V$ that are relevant to v_r. Specifically, the ***fake***[1] (i.e., estimated) connectivity distribution of node v_r is calculated as:

$$p'(v|v_r) = G(v|v_r;\theta_G) = \frac{\exp\left(\mathbf{g}_v^\top \mathbf{g}_{v_r}\right)}{\sum_{v \neq v_r} \exp\left(\mathbf{g}_v^\top \mathbf{g}_{v_r}\right)}, \tag{1}$$

where $\mathbf{g}_v, \mathbf{g}_{v_r} \in \mathbb{R}^k$ are the k-dimensional vectors of nodes v and v_r, respectively, and θ_G is the union of all $\mathbf{g}_v$'s. To update θ_G in each iteration, a set of node pairs (v, v_r), not necessarily directly connected, is sampled according to $p'(v|v_r)$. The key purpose of generator G is to deceive the discriminator D, and thus its loss function L_G is determined as follows:

$$L_G = \min \sum_{r=1}^{|V|} \mathbb{E}_{v \sim G(\cdot|v_r;\theta_G)} \left[\log\left(1 - D\left(v_r, v \mid \theta_D\right)\right)\right], \tag{2}$$

where the discriminant function $D(\cdot)$ estimates the probability that a given node pairs (v, v_r) are considered ***real***, i.e., directly connected.

Discriminator D: This module tries to distinguish between real node pairs and fake node pairs synthesized by the generator G. Accordingly, the discriminator estimates the probability that an edge exists between v_r and v, denoted as:

$$D(v_r, v|\theta_D) = \sigma(\mathbf{d}_v^\top \mathbf{d}_{v_r}) = \frac{1}{1 + \exp(-\mathbf{d}_v^\top \mathbf{d}_{v_r})}, \tag{3}$$

where $\mathbf{d}_v, \mathbf{d}_{v_r} \in \mathbb{R}^k$ are the k-dimensional vectors corresponding to the v-th and v_r-th rows of discriminator parameters θ_D, respectively. $\sigma(\cdot)$ represents the sigmoid function of the inner product of these two vectors. Given the sets of real and fake node pairs, the loss function of D can be derived as:

$$\begin{aligned} L_D = &\max \sum_{r=1}^{|V|} \left(\mathbb{E}_{v \sim p(\cdot|v_r)} \left[\log D\left(v, v_r \mid \theta_D\right)\right]\right. \\ &\left.+ \mathbb{E}_{v \sim G(\cdot|v_r;\theta_G)} \left[\log\left(1 - D\left(v_r, v \mid \theta_D\right)\right)\right]\right). \end{aligned} \tag{4}$$

In summary, the generator G and discriminator D operate as two adversarial components: the generator G aims to fit the true connectivity distribution $p(v|v_r)$, generating candidate nodes v that resemble the real neighbors of v_r to deceive the discriminator D. In contrast, the discriminator D seeks to distinguish whether a given node is a true neighbor of v_r or one generated by G. Formally, D and G are engaged in a two-player minimax game with the following loss function:

$$\begin{aligned} \min_{\theta_G} \max_{\theta_D} L(G, D) = &\sum_{r=1}^{|V|} \left(\mathbb{E}_{v \sim p(\cdot|v_r)} \left[\log D\left(v, v_r \mid \theta_D\right)\right]\right. \\ &\left.+ \mathbb{E}_{v \sim G(\cdot|v_r;\theta_G)} \left[\log\left(1 - D\left(v_r, v \mid \theta_D\right)\right)\right]\right). \end{aligned} \tag{5}$$

Based on Eq. (5), the parameters θ_D and θ_G are updated by alternately maximizing and minimizing the loss function $L(G, D)$. Competition between G and D results in mutual improvement until G becomes indistinguishable from the true connectivity distribution.

H Details of Theorem

Theorem 1. *By constraining both the number and length of paths generated via random walks on the BFS-trees to N and L, respectively, the gradient sensitivity Δ_g of the discriminator can be reduced from BC to $\frac{N^{L+1}-1}{N-1}C$.* [2] *Thus, the noisy gradient $\tilde{\nabla} L_D$ of discriminator within a mini-batch $\mathcal{B}_t$ is denoted as:*

$$\tilde{\nabla} L_D = \frac{1}{|\mathcal{B}_t|}\Big(\sum_{v \in \mathcal{B}_t} Clip_C\big(\frac{\partial L_D}{\partial \mathbf{d}_v}\big) + \mathcal{N}\left(0, \Delta_g^2 \sigma^2 \mathbf{I}\right)\Big), \tag{6}$$

where the gradient sensitivity $\Delta_g = \frac{N^{L+1}-1}{N-1}C$.

Theorem 2. *Given the number of training set N_{tr}, number of epochs n^{epoch}, number of discriminators' iterations n^{iter}, batch size B_d, maximum path length L, and maximum path number N, over $T = n^{epoch} n^{iter}$ iterations, Algorithm 2 satisfies node-level $(\alpha, 2T\gamma)$-RDP, where $\gamma = \frac{1}{\alpha-1}\ln\left(\sum_{i=0}^{R_{N,L}} \beta_i \left(\exp\frac{\alpha(\alpha-1)i^2}{2\sigma^2 R_{N,L}^2}\right)\right)$, $R_{N,L} = \frac{N^{L+1}-1}{N-1}$ and $\beta_i = \binom{R_{N,L}}{i}\binom{N_{tr}-R_{N,L}}{B_d-i}/\binom{N_{tr}}{B_d}$. Please refer to **App. K** for the proof.*

[1] The term "Fake" indicates that although a node v selected by the generator is relevant to v_r, there is no actual edge between them.

[2] Empirical results in Section 5 demonstrate that our ASGL achieves satisfactory performance even with a relatively small receptive field. Specifically, when setting $N = 3$ and $L = 4$, that is, $\frac{N^{L+1}-1}{N-1} = 121 < B = 256$, the ASGL method still performs good model utility.

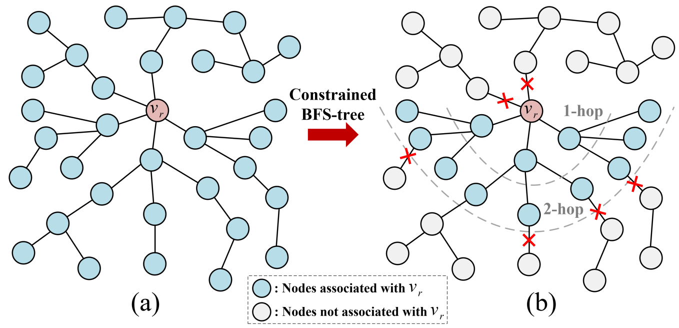

Figure 4: The receptive field of node v_r within a batch is illustrated in two cases: (a) An unconstrained BFS tree, and the receptive field size of v_r is $B = |V_B| = 34$; (b) A constrained BFS tree with path length $L = 2$, path amount $N = 3$ of each node, and the receptive field size of v_r is $\sum_{l=0}^{L} N^l = 13$.

I Details of Lemma

The following lemmas are used for proving Theorem 1:

Lemma 2 (Receptive field of a node)**.** *As shown in Fig. 4(b), we define the* ***receptive field*** *of a node as the region (i.e., the set of nodes) over which it can exert influence. Accordingly, for a* ***subgraph*** *constructed from paths sampled on constrained BFS-trees (Fig. 4(b)), the maximum receptive field size of v_r is given by $R_{N,L} = \sum_{l=0}^{L} N^l = \frac{N^{L+1}-1}{N-1} \leq B$.*

Lemma 3. *Let S_{tr} denote the training set of subgraphs constructed from constrained BFS-tree paths, and $S(v) \subset S_{tr}$ denote the subgraph subset that contains the node v. Since $R_{N,L}$ represents the upper bound on the number of occurrences of any node in S_{tr}, it follows that $|S(v)| \leq R_{N,L}$. The proof of Lemma 3 is illustrated in* ***App. J****.*

J Proof of Lemma 3

Proof. We proceed by induction [1] on the path length L of the BFS-tree.

Base case: When $L = 0$, each sampled subgraph $S(v)$ contains exactly the training node $v \in V_{tr}$ itself. Thus, every node appears in one subgraph, trivially satisfying the bound $|S(v)| = R_{N,0} = 1$.

Inductive hypothesis: Assume that for some fixed $L \geq 0$, any $v \in V_{tr}$ appears in at most $R_{N,L}$ subgraphs constructed from constrained BFS-tree paths. Let $S^L(v)$ denote a subgraph set with L path length. Thus, the hypothesis is $|S^L(v)| \leq R_{N,L}$ for any v.

Inductive step: We further show that the above hypothesis also holds for $L + 1$ path length: Let $T_{u'}$ represent the L-length BFS-tree rooted at u'. If $T_{u'} \in S^{L+1}(v)$, there must exit node u such that $u \in T_{u'}$ and $T_u \in S^L(v)$. According to the setting of Algorithm 1, the number of such nodes u is at most N. By the hypothesis, there are at most $R_{N,L} - 1$ such $u' \neq v$ such that $T_{u'} \in S^{L+1}(v)$. Based on these upper bounds, we can derive the upper bound matching the inductive hypothesis for $L + 1$:

$$\left|S^{L+1}(v)\right| \leq N \cdot (R_{N,L} - 1) + 1 = \frac{N^{L+2} - 1}{N - 1} = R_{N,L+1}. \tag{7}$$

By induction, the Lemma 3 holds for all $L \geq 0$.

K Proof of Theorem 2

The following lemmas are used for proving Theorem 2:

Lemma 4 (Adaptation of Lemma 5 from [4])**.** *Let $\mathcal{N}(\mu, \sigma^2)$ represent the Gaussian distribution with mean μ and standard deviation σ^2, it holds that:*

$$\mathcal{D}_\alpha \left(\mathcal{N}\left(\mu, \sigma^2\right) \| \mathcal{N}\left(0, \sigma^2\right)\right) = \frac{\alpha\mu^2}{2\sigma^2} \tag{8}$$

Lemma 5 (Adaptation of Lemma 25 from [2])**.** *Assume $\mu_0, ..., \mu_n$ and $\eta_0, ..., \eta_n$ are probability distributions over some domain Z such that their Rényi divergences satisfy: $\mathcal{D}_\alpha(\mu_0||\eta_0) \leq \epsilon_0, ..., \mathcal{D}_\alpha(\mu_n||\eta_n) \leq \epsilon_n$ for some given $\epsilon_0, ..., \epsilon_n$. Let ρ be a probability distribution over $\{0, ..., n\}$. Denoted by μ_ρ (η_ρ, respectively) the probability distribution on Z obtained by sampling i from ρ and then randomly sampling from μ_i and η_i, we have:*

$$\mathcal{D}_\alpha \left(\mu_\rho \| \eta_\rho\right) \leq \ln \mathbb{E}_{i\sim\rho} \left[e^{\varepsilon_i(\alpha-1)}\right] = \frac{1}{\alpha - 1} \ln \sum_{i=0}^{n} \rho_i e^{\varepsilon_i(\alpha-1)} \tag{9}$$

Proof of Theorem 2. Consider any minibatch $\mathcal{B}_t$ randomly sampled from the training subgraph set S_{tr} of Algorithm 2 at iteration t. For a subset $S(v^*) \subset S_{tr}$ containing node v^*, its size is bounded by $R_{N,L}$ (Lemma 3). Define the random variable β as $|S(v^*) \cap \mathcal{B}_t|$, and its distribution follows the hypergeometric distribution Hypergeometric($|S_{tr}|, R_{N,L}, |\mathcal{B}_t|$) [3]:

$$\beta_i = P[\beta = i] \xrightarrow[|S_{tr}|=N_{tr}]{|\mathcal{B}_t|=B_d} \frac{\binom{R_{N,L}}{i}\binom{N_{tr}-R_{N,L}}{B_d-i}}{\binom{N_{tr}}{B_d}}. \tag{10}$$

Next, consider the training of the discriminators (Lines 12–18 and 24–30 in Algorithm 2). Let $\mathcal{G}$ and $\mathcal{G}'$ be two adjacent graphs differing only in the presence of node v^* and its associated signed edges. Based on the gradient perturbation applied in Lines 15 and 27 of Algorithm 2, we have:

$$\begin{aligned} \tilde{g}_t &= g_t + \mathcal{N}\left(0, \sigma^2\Delta_g^2\mathbf{I}\right) = \sum_{v\in\mathcal{B}_t} \mathrm{Clip}_C(\frac{\partial L_D}{\partial \mathbf{d}_v}) + \mathcal{N}\left(0, \sigma^2\Delta_g^2\mathbf{I}\right) \\ \tilde{g}'_t &= g'_t + \mathcal{N}\left(0, \sigma^2\Delta_g^2\mathbf{I}\right) = \sum_{v'\in\mathcal{B}'_{tr}} \mathrm{Clip}_C(\frac{\partial L_D}{\partial \mathbf{d}_{v'}}) + \mathcal{N}\left(0, \sigma^2\Delta_g^2\mathbf{I}\right), \end{aligned} \tag{11}$$

where $\Delta_g = R_{N,L}C = \frac{N^{L+1}-1}{N-1}C$ (Theorem 1). $\tilde{g}_t$ and $\tilde{g}'_t$ denote the noisy gradients of $\mathcal{G}$ and $\mathcal{G}'$, respectively. When $\beta = i$, their Rényi divergences can be upper bounded as:

$$\begin{aligned} &\mathcal{D}_\alpha\left(\tilde{g}_{t,i} \| \tilde{g}'_{t,i}\right) \\ &= \mathcal{D}_\alpha\left(g_{t,i} + \mathcal{N}\left(0, \sigma^2\Delta_g^2\mathbf{I}\right) \| g'_{t,i} + \mathcal{N}\left(0, \sigma^2\Delta_g^2\mathbf{I}\right)\right) \\ &= \mathcal{D}_\alpha\left(\mathcal{N}\left(g_{t,i}, \sigma^2\Delta_g^2\mathbf{I}\right) \| \mathcal{N}\left(g'_{t,i}, \sigma^2\Delta_g^2\mathbf{I}\right)\right) \\ &\overset{(a)}{=} \mathcal{D}_\alpha\left(\mathcal{N}\left(\left(g_{t,i} - g'_{t,i}\right), \sigma^2\Delta_g^2\mathbf{I}\right) \| \mathcal{N}\left(0, \sigma^2\Delta_g^2\mathbf{I}\right)\right) \\ &\overset{(b)}{\leq} \sup_{\|\Delta_i\|_2 \leq iC} \mathcal{D}\left(\mathcal{N}\left(\Delta_i, \sigma^2\Delta_g^2\mathbf{I}\right) \| \mathcal{N}\left(0, \sigma^2\Delta_g^2\mathbf{I}\right)\right) \\ &\overset{(c)}{=} \sup_{\|\Delta_i\|_2 \leq iC} \frac{\alpha\|\Delta_i\|_2^2}{2\Delta_g^2\sigma^2} = \frac{\alpha i^2}{2R_{N,L}^2\sigma^2}, \end{aligned} \tag{12}$$

where $\Delta_i = g_{t,i} - g'_{t,i}$. (a) leverages the property that Rényi divergence remains unchanged under invertible transformations [4], while (b) and (c) are derived from Theorem 1 and Lemma 4, respectively. Based on Lemma 5 , we derive that:

$$\begin{aligned} \mathcal{D}_\alpha\left(\tilde{g}_t \| \tilde{g}'_t\right) &\leq \ln \mathbb{E}_{i\sim\beta}\left[\exp\left(\frac{\alpha i^2(\alpha-1)}{2R_{N,L}^2\sigma^2}\right)\right] \\ &= \frac{1}{\alpha-1}\ln\left(\sum_{i=0}^{R_{N,L}} \beta_i \exp\left(\frac{\alpha i^2(\alpha-1)}{2R_{N,L}^2\sigma^2}\right)\right) = \gamma. \end{aligned} \tag{13}$$

Here, β_i is illustrated in Eq. (10). Based on the composition property of DP, after $T = n^{epoch} \cdot n^{iter}$ interations, the discriminators satisfy node-level $(\alpha, 2T\gamma)$-RDP. Moreover, owing to the post-processing property of DP, the generators G^+ and G^- inherit the same privacy guarantee as the discriminators. Therefore, Algorithm 2 obeys node-level $(\alpha, 2T\gamma)$-RDP, and the proof of Theorem 2 is completed.

Table 7: Summary of average SSI with different ϵ and datasets for node clustering tasks. (BOLD: Best)

ϵ	Dataset	SGCN	SDGNN	SiGAT	LSNE	GAP	ASGL
1	Bitcoin-Alpha	0.4819	0.4378	0.4877	0.4977	0.4988	**0.5091**
	Bitcoin-OTC	0.4505	0.4677	0.5025	0.4970	0.5008	**0.5160**
	Slashdot	0.4715	0.5011	0.5025	0.5052	0.5005	**0.5107**
	WikiRfA	0.4788	0.4988	0.4968	0.4890	0.5003	**0.5126**
	Epinions	0.5001	0.4965	0.5022	0.5013	0.6095	**0.6106**
2	Bitcoin-Alpha	0.4910	0.4733	0.4969	0.4985	0.5032	**0.5402**
	Bitcoin-OTC	0.4733	0.4968	0.5075	0.4986	0.5729	**0.6810**
	Slashdot	0.4888	0.4864	0.4871	0.5134	0.5132	**0.5494**
	WikiRfA	0.4934	0.5054	0.5117	0.4996	0.5032	**0.5577**
	Epinions	0.5068	0.5116	0.5086	0.5463	0.6263	**0.6732**
4	Bitcoin-Alpha	0.5019	0.4948	0.5112	0.5049	0.6204	**0.6707**
	Bitcoin-OTC	0.5005	0.5325	0.5612	0.5465	0.6953	**0.7713**
	Slashdot	0.5003	0.5685	0.5545	0.5671	0.5444	**0.5994**
	WikiRfA	0.5005	0.5142	0.5538	0.5476	0.5644	**0.5977**
	Epinions	0.5148	0.5389	0.5386	0.6255	0.6747	**0.6787**

References

[1] Ameya Daigavane, Gagan Madan, Aditya Sinha, Abhradeep Guha Thakurta, Gaurav Aggarwal, and Prateek Jain. 2022. Node-Level Differentially Private Graph Neural Networks. In *ICLR 2022 Workshop on PAIR^2Struct*.

[2] Vitaly Feldman, Ilya Mironov, Kunal Talwar, and Abhradeep Thakurta. 2018. Privacy amplification by iteration. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 521–532.

[3] Catherine Forbes, Merran Evans, Nicholas Hastings, and Brian Peacock. 2011. *Statistical distributions*. John Wiley & Sons.

[4] Renxuan Hou, Qingqing Ye, Xun Ran, Sen Zhang, and Haibo Hu. 2025. PrivIM: Differentially Private Graph Neural Networks for Influence Maximization. In *2025 IEEE 41st International Conference on Data Engineering (ICDE)*. 3467–3479.

[5] Hongwei Wang, Jia Wang, Jialin Wang, Miao Zhao, Weinan Zhang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. Graphgan: Graph representation learning with generative adversarial nets. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.