Privacy-preserving design of graph neural networks with applications to vertical federated learning

Ruofan Wu[†], Mingyang Zhang[†], Lingjuan Lyu[§], Xiaolong Xu[†],
Xiuquan Hao[†], Xinyi Fu[†], Tengfei Liu[†], Tianyi Zhang[†], and Weiqiang Wang[†]

[†]Ant Group
[§]Sony AI
{ruofan.wrf, zhangmingyang.zmy, yiyin.xxl, haoxiuquan.hxq, fxy122992, aaron.ltf,
zty113091, weiqiang.wwq}@antgroup.com
lingjuan.lv@sony.com

Abstract

The paradigm of vertical federated learning (VFL), where institutions collaboratively train machine learning models via combining each other's local feature or label information, has achieved great success in applications to financial risk management (FRM). The surging developments of graph representation learning (GRL) have opened up new opportunities for FRM applications under FL via efficiently utilizing the graph-structured data generated from underlying transaction networks. Meanwhile, transaction information is often considered highly sensitive. To prevent data leakage during training, it is critical to develop FL protocols with *formal privacy guarantees*. In this paper, we present an end-to-end GRL framework in the VFL setting called VESPER, which is built upon a general privatization scheme termed *perturbed message passing (PMP)* that allows the privatization of many popular graph neural architectures. Based on PMP, we discuss the strengths and weaknesses of specific design choices of concrete graph neural architectures and provide solutions and improvements for both dense and sparse graphs. Extensive empirical evaluations over both public datasets and an industry dataset demonstrate that VESPER is capable of training high-performance GNN models over both sparse and dense graphs under reasonable privacy budgets.

1 Introduction

In recent years, there has been an increasing interest in adopting modern machine learning paradigms to the area of financial risk management (FRM) [31]. The most crucial task in operational risk scenarios like fraud detection is identifying risky identities based on the behavioral data collected from the operating financial platform [4, 24]. For institutions like commercial banks and online payment platforms, the most important source of behavior information is the *transaction records* between users, making *transaction networks* (with users as nodes and transactions as edges) a direct and appropriate data model. To exploit the potential of transaction networks in a machine learning context, recent approaches [26, 47] have been exploring the adoption of graph representation learning (GRL) [16] as a principled way of incorporating structural information contained in transaction networks into the learning process. The family of graph neural networks in the message passing form [13, 48] offers a powerful yet scalable solution to GRL, and has become the prevailing practice in industry-scale graph learning [52].

Despite its convincing performance, high-quality network data are not always available for financial institutions. F It is, therefore, of great interest for institutions to learn GRL models *collaboratively* while being coherent to regulatory strictures at the same time. The technique of federated learning

NeurIPS 2023 New Frontiers in Graph Learning Workshop (NeurIPS GLFrontiers 2023).

(FL) [20, 49] provides a recipe for such scenarios, with participating institutions (hereafter abbreviated as *parties*) exchanging intermediate results instead of raw data. Depending on the specific form of collaboration, FL protocols are generally divided into horizontal federated learning (HFL), where participants aggregate their locally trained models to obtain a strong global model, and vertical federated learning (VFL) where participants are able to align the identifiers of modeling entities and train a model that efficiently combines feature or label information that are distributed among different parties. VFL is particularly useful when training a (supervised) model is not possible based on information of a single party, i.e., each party holds only feature or label data, and has attracted significant attention in applications to FRM [28]. While ordinary FL paradigms avoid the transmission of local raw data, they typically lack a formal guarantee of privacy [20, Chapter 4]. Moreover, recent studies have reported successful attacks targeting individual privacy against FL protocols [54, 50, 19, 9, 8]. As transaction records are widely considered extremely sensitive personal information, it is thus critical to establish FL applications in FRM with rigorous privacy guarantees.

Differential privacy (DP) [11] is the state-of-the-art approach to address information disclosure that injects algorithm-specific random noise to fuse the participation of any individual. The adoption of DP as the privacy model for FL is now under active development, with most of the applications appearing in HFL over independently identically distributed (i.i.d.) data through the lens of optimization [20]. However, discussions on applying DP over VFL remain nascent [3, 53, 39]. The situation becomes even more complicated in VFL over graph-structured data, since the right notions of (differential) privacy on graphs are semantically different from that in the i.i.d. case [35, 22]. So far, as we have noticed, the only work that provides meaningful DP guarantee under VFL over graphs is the GAP model [39], which requires three stages of training. Meanwhile, a notable aspect of GRL is that the structure of the underlying graph, i.e., whether the graph is dense or sparse, might have a significant influence on the performance of the graph neural model especially when the aggregation process involves noisy perturbations. This phenomenon was overlooked in previous studies.

In this paper, we discuss private FL over graph-structured data under the task of node classification in the vertical setup with edge DP [35] chosen as the privacy model. We first develop a general privatization scheme termed *perturbed message passing (PMP)* that produces message-passing procedures over graphs that are guaranteed to satisfy edge DP constraints. Next, we discuss the influence of the underlying graph's degree profiles on the utility of specific design choices of PMP, using two representative graph aggregation schemes, namely GIN [48] and GCN [23], and develop further improvements of PMP that better handles sparse graphs under the GCN aggregation scheme. Finally, we integrate the developments of PMP and its variants into a VFL architecture called VESPER based on the SplitNN framework [14], and conducted extensive empirical evaluations over both public and industrial datasets covering dense and sparse graphs. We summarize our contributions as follows:

- We propose PMP, a general framework for designing differentially private message-passing procedures. PMP enables the privatization of many popular graph neural network architectures. The privacy guarantee of PMP is formally analyzed with new privacy amplification results under uniform neighborhood sampling.

- We discuss two representative design choices under the PMP framework, GIN and GCN, and discover the fact that the utility of the privatized GNN model may be affected by the *degree profile* of the input graph. To better accommodate varying graph structures, we develop the truncated message passing framework under the base model of GCN through properly tuning the hyperparameter that reduces noise scale at the cost of learning less structural information, which is beneficial when the input graph is *sparse*.

- We derive an end-to-end VFL learning framework operating over graph-structured data called VESPER, which is efficient in computation and communication. A thorough experimental study demonstrates that VESPER achieves better privacy-utility trade-off over previously proposed models and is capable of training high-performance GNN models over both sparse and dense graphs under reasonable privacy budgets.

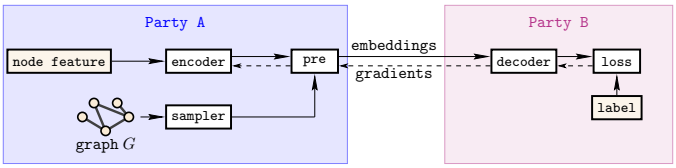

Figure 1: A concise pictorial description of the VESPER framework. We use solid arrows to depict the dataflow of forward computations and use dashed arrows to depict the dataflow of backward computations.

2 Methodology

2.1 Priliminaries

We focus on the node classification task over a static, undirected graph $G = (V, E)$ with node size $N = |V|$, node feature $X = \{x_v\}_{v \in V}$ and node labels $Y = \{y_v\}_{v \in V_T}$ where $V_T \subseteq V$ is the set of training nodes with $N_T = |V_T|$. Throughout this article, we will assume the graph of interest to be degree bounded, i.e.,

$$\max_{G} \max_{v \in G} d_v \leq D \tag{1}$$

for some $D > 1$. In this paper, we will be interested in the setup where the graph data G and label information are distributed over two distinct parties. Specifically, suppose there are two parties, A (Alice) and B (Bob), where A holds the graph data G as well as the node feature X and B holds the label collection Y, both indexed by node identifiers that are known to both sides (i.e., V_T is known to both party A and party B). We consider a representative federated learning paradigm that A and B collaboratively train a graph representation learning model via utilizing the panoply of graph neural networks [13], which could be regarded as a special case of vertical federated learning (VFL) [49]. Under VFL protocols, party A and party B iteratively exchange intermediate outputs depending on the specific training algorithm chosen. A main concern in VFL [20, Chapter 4] is, therefore, whether the exchanging process satisfies formal *privacy* guarantees. Before elaborating on privacy protection issues, we first state the threat model in our context.

Threat model We adopt the following threat model in this paper: In the training stage, label party B is curious about the adjacency information (i.e., the existence of some edges) in the data party A. The data party A is assumed to be benign, with both parties strictly obeying the chosen VFL protocol. [1] In other words, the goal of privacy protection is to prevent the *semi-honest* adversary (party B) from inferring the edge membership that is only known to party A.

Differential privacy [11] is now the *de facto* choice of privacy protection paradigm against membership inference adversaries. As an appropriate solution concept in the current setup, we introduce the edge-level differential privacy model (hereafter abbreviated as Edge DP).

Definition 1 (Edge-level differential privacy(Edge DP))**.** For a (randomized) graph-input mechanism $\mathcal{M}$ that maps graphs to some output space $\mathcal{S}$ and two non-negative numbers ϵ and δ, the mechanism is (ϵ, δ)-Edge DP if for any subset S (or more rigorously defined as Borel measurable sets) of the output space, the following holds uniformly for any two possible adjacent graphs (G, G'):

$$\mathbb{P}[\mathcal{M}(G) \in S] \leq e^{\epsilon} \mathbb{P}[\mathcal{M}(G') \in S] + \delta, \tag{2}$$

where we define two graphs G and G' as being adjacent if G could be edited into G' via adding or removing a single edge.

Regarding the capability of the adversary adopted in this paper, a VFL protocol satisfying Edge DP with a reasonable ϵ level implies that based on all the exchanged intermediate outputs between party A and party B, any membership inference algorithm may not be able to make any sophisticated guess about the existence of some specific edge in a probabilistic sense, thereby offering strong privacy protection. Most contemporary differentially private machine learning algorithms involve sequentially applying DP procedures to intermediate learning steps [1], with the privacy level of

[1] The assumption of a harmless party A might be relaxed to a curious onlooker that tries to infer party B's label information. We discuss related extensions in section D.

the entire training procedure obtained via composition theorems [11, 21]. In this paper, we choose the composition framework of analytical moment accountant (AMA) [44] that exploits the idea of Rényi DP [33], which we introduce below in our graph learning context:

Definition 2 (Edge-level Rényi -differential privacy(Edge RDP))**.** Sharing notations with definition 1, the mechanism $\mathcal{M}$ is $(\alpha, \epsilon(\alpha))$-Rényi differentially private with some $\alpha > 1$ and $\epsilon(\alpha) \geq 0$, if for any two possible adjacent graphs (G, G'), the α-Rényi divergence of the induced probability distribution of random variables $\mathcal{M}(G)$ and $\mathcal{M}(G')$ is bounded by $\epsilon(\alpha)$:

$$D_\alpha \left(\mathcal{M}(G) || \mathcal{M}(G')\right) \leq \epsilon(\alpha), \tag{3}$$

with the definition of α-Rényi divergence $D_\alpha \left(\cdot||\cdot\right)$ presented in appendix A.

To develop privacy-preserving learning algorithms under the AMA framework, we first design mechanisms that satisfy RDP guarantee in each step, then use standard composition results of RDP [33] to obtain the privacy level of the learning procedure. Finally, we apply the conversion rule in [2] to convert it back to (ϵ, δ)-DP for reporting.

Message passing GNNs with stochastic training The backbone of our privacy-preserving training framework is the graph neural network model in the message passing form [13]. We define the GNN of interest to be a map from the space of graphs to a node embedding matrix with embedding dimension d: $f : \mathcal{G} \mapsto \mathbb{R}^{N \times d}$, or $H := \{h_v\}_{v \in V} = f(G)$. For an L-layer GNN, let $h_v^{(0)} = g(x_v)$ be the input encoding of node v, which could be either x_v or some encoding based on x_v. We assume the following recursive update rule for $1 \leq l \leq L$ and $v \in V$:

$$h_v^{(l)} = \sigma\left(\widetilde{h}_v^{(l)}\right), \quad \widetilde{h}_v^{(l)} = \omega_v W_1^{(l)} h_v^{(l-1)} + \sum_{u \in N(v)} \beta_{uv} W_2^{(l)} h_u^{(l-1)}, \tag{4}$$

with $\boldsymbol{\omega} := \{\omega_v\}_{v \in V} \in \mathbb{R}^N$ and $\boldsymbol{\beta} := \{\beta_{uv}\}_{u,v \in V \times V} \in \mathbb{R}^{N \times N}$ be model-dependent coefficients, σ a parameter-free nonlinear function, and $\mathbf{W} = (W_1^{(1)}, \ldots, W_1^{(L)}, W_2^{(1)}, \ldots, W_2^{(L)})$ be the collection of learnable parameters. For any matrix W, we denote $\|W\|_{\text{op}}$ as the operator norm of the matrix (i.e., its largest singular value). In this paper, we assess two representative instantiations of the protocol (4) which are the GIN model [48] with with $\omega_v \equiv \beta_{uv} \equiv 1, \forall u, v \in V$ and the GCN model [23] with $\omega_v = \frac{1}{d_v + 1}$ and $\beta_{uv} = \frac{1}{\sqrt{d_u + 1}\sqrt{d_v + 1}}$. For simplicity we additionally let the nonlinearity be the ReLU function and set $W_1^{(l)} = W_2^{(l)} = W^{(l)}, 1 \leq l \leq L$.

Applying message passing updates (4) may become computationally prohibitive for large input graphs, which are frequently encountered in industrial scenarios. To enable scalable GRL, the prevailing practice is to use graph sampling methods [15] and adopt **stochastic training of graph neural networks**. In this paper, we investigate the simple and effective sampling scheme of uniform neighborhood sampling [15, 7], with the maximum number of neighbors sampled in each layer to be the maximum degree D. Asides from their computational benefits, it has been observed [1, 32] that stochastic training with a low sampling ratio over large datasets is crucial to training high-utility differentially private machine learning models with reasonably small privacy budgets, which has also been recently verified in the case of differentially private graph learning [7, 39].

2.2 Perturbed message passing

A notable fact about the message-passing protocol (4) is that it uses the aggregation strategy of *weighted summation*, thereby allowing standard additive perturbation mechanisms like the Laplace mechanism or Gaussian mechanism that are prevailing in the design of differentially private algorithms [11]. Motivated by this fact, we propose a straightforward solution to privatize message-passing GNNs in a *layer-wise* fashion named *perturbed message passing (PMP)*, which adds layer-wide Gaussian noise with an additional normalization step that controls sensitivity. We present the pseudo-code of PMP with neighborhood sampling in algorithm 1. Next we discuss the privacy guarantee of algorithm 1. To state our main result, we first define the right notion of sensitivity in our context:

Definition 3 (Edge sensitivity)**.** Denote G' as the adjacent graph via removing the edge (u^*, v^*) from G, and let $\widetilde{h}_v$ and $\widetilde{h}'_v$ be the outputs of node v generated via some 1-layer GNN protocol under graph G and G' without nonlinearity, then we define the (ℓ_2-) *edge sensitivity* as:

$$\mathcal{S} = \max_{G, G'} \sqrt{\sum_{v \in V} \|\widetilde{h}_v - \widetilde{h}'_v\|_2^2}. \tag{5}$$

Algorithm 1 PMP with neighborhood sampling

Require: Graph $G = (V, E)$, input encodings $\{h_v^{(0)}\}_{v \in V}$, number of message passing rounds L, GNN spec $(\boldsymbol{\omega}, \boldsymbol{\beta}, \sigma)$, noise scale θ, GNN parameter $\mathbf{W}$, batch size B, maximum degree D.

1: Sample a random batch of root nodes $v_1, \ldots, v_B$.
2: Apply an L-layer neighborhood sampler with each layer sampling at most D nodes with roots $v_1, \ldots, v_B$, obtaining a batch of B subgraphs $(G_{v_1}^{(L)}, \ldots, G_{v_B}^{(L)})$.
3: Combine $(G_{v_1}^{(L)}, \ldots, G_{v_B}^{(L)})$ into a subgraph $G_B^{(L)}$. Additionally, overload the notation $N(v)$ for the neighborhood of node v with respect to $G_{v_B}^{(L)}$.
4: Set $h_v^{(0)} = \frac{h_v^{(0)}}{\left\| h_v^{(0)} \right\|_2}$ for $\forall v \in G_{v_B}^{(L)})$
5: **for** $l \in \{1, \ldots, L\}$ **do**
6: **for** $v \in G_{v_B}^{(L)}$ **do**
7: Compute the linear update $\widetilde{h}_v^{(l)} = \omega_v W_1^{(l)} h_v^{(l-1)} + \sum_{u \in N(v)} \beta_{uv} W_2^{(l)} h_u^{(l-1)}$.
8: Do additive perturbation, $h_v^{(l)} = \sigma(\widetilde{h}_v^{(l)} + N(0, \theta^2))$
9: Normalize $h_v^{(l} = \frac{h_v^{(l)}}{\left\| h_v^{(l} \right\|_2}$

return A list of all layers' embedding matrices $\mathbf{H}_L = (H^{(1)}, \ldots, H^{(L)})$, with $H^{(l)} = \{h_v^{(l)}\}_{v \in G_{v_B}^{(L)}}, 1 \leq l \leq L$.

The following theorem quantifies the privacy guarantee of algorithm 1:

Theorem 2.1 (RDP guarantee)**.** *Let $\mathbf{H}_L$ be the released outputs with input a minitach of B subgraphs produced by uniform neighborhood sampling for L layers with a maximum number of D neighbors sampled in each layer. Define $\epsilon(\alpha) := \frac{\alpha \sum_{l=1}^{L} \mathcal{S}_l^2}{2\theta^2}$, then $\mathbf{H}_L$ is $(\alpha, \epsilon_\gamma(\alpha)$-RDP for any $\alpha > 1$, where $\gamma = 1 - \frac{\binom{N_T - \frac{2(D^L - 1)}{D-1}}{B}}{\binom{N_T}{B}}$ and*

$$\epsilon_\gamma(\alpha) \leq \frac{1}{\alpha - 1} \log \left(1 + \gamma^2 \binom{\alpha}{2} \min \left(4 \left(e^{\epsilon(2)} - 1 \right), \epsilon(2) \min \left(2, \left(e^{\epsilon(\infty) - 1} \right)^2 \right) \right) + \sum_{j=3}^{\infty} \gamma^j \binom{\alpha}{j} e^{(j-1)\epsilon(j)} \min \left(2, \left(e^{\epsilon(\infty) - 1} \right)^j \right) \right) \tag{6}$$

Theorem 2.1 provides a principled way of analyzing the privacy of privatized GNN models using algorithm 1, which boils down to computing the edge sensitivity of the underlying message passing protocol. However, sensitivity computations are usually conducted in a *worst-case* manner, resulting in unnecessarily large noise levels and significant utility loss. Therefore, it is valuable to explore the utility of concrete PMP models and their relationships with the underlying input graph. To begin our expositions, we analyze the GIN model in the following section.

2.3 Analysis of GIN and the challenge of sparse graphs

We start with the following proposition:

Proposition 1. *Under the GIN model, the edge sensitivity is bounded from above by $\mathcal{S}_l^{GIN} \leq \sqrt{2} \|W^{(l)}\|_{op}$ for each $1 \leq l \leq L$.*

Advantage of layer-wise perturbations According to proposition 1, the edge sensitivity of GIN is independent of the input graph's maximum degree upper bound D, which is essentially a direct consequence of the fact that for a 1 layer message passing procedure, adding or removing one edge would affect up to two nodes' output embeddings. As a consequence, the privacy cost scales linearly with the number of message-passing layers in the Rényi DP framework, thereby offering a better privacy-utility trade-off than algorithms that do the do the perturbation only in the final layer [53], whose privacy cost may scale exponentially with D.

Effectiveness and challenges of summation pooling It has been observed in previous works [39] that aggregation perturbation with sum pooling works well on graphs with a large average degree. Intuitively, this phenomenon could be understood as keeping a high "signal-to-noise ratio (SNR)" during the aggregation process: For nodes with large degrees, the noise scale becomes relatively small with respect to the summation of incoming messages. Therefore if high-degree nodes are prevalent in the underlying graph, the utility loss during aggregation is reasonably controlled for most nodes. However, realistic graph data might not have large average degrees. For example, transaction networks in FRM scenarios are usually sparse, including many nodes with degrees smaller than 5 or even being singular (i.e., of degree 0). Consequently, the SNR of sparse networks makes it harder for summation pooling to maintain decent utility, which will be further verified in section 3.

2.4 Improvements of PMP in the GCN model

As discussed in the previous section, the degree profile of the input graph may affect the utility of PMP-privatized GNNs when the underlying aggregation follows the summation pooling scheme. It is therefore of interest to explore aggregation schemes that are more appropriate when the input graph is sparse. On first thought, we may expect aggregation schemes like mean pooling or GCN pooling to have smaller sensitivities. However, such sensitivity reduction does NOT hold in a worst-case analysis: Just think of nodes with degree 1, then it is not hard to check that mean pooling or GCN pooling behaves similarly to summation pooling. The primary issue with worst-case analysis is that the resulting sensitivity is determined by extremely *low-degree* nodes. Inspired by this phenomenon, we seek improvements by first deriving lower sensitivity with an extra requirement on a *degree lower bound*, and then relax the requirement via introducing a modified protocol. We start with the following observation:

Proposition 2. *Assume all the possible input graphs have a minimum degree larger or equal to D_{min}, or*

$$\min_{G} \min_{v \in G} d_v \geq D_{min} > 1. \tag{7}$$

Then for the GCN model, the edge sensitivity of the l-th layer $\mathcal{S}_l^{GCN}$ is bounded from above by a function $\eta_l(D_{min})$, defined as:

$$\eta_l(D_{min}) = \sqrt{2}\left(\frac{1 - 1/D_{min}}{2D_{min}} + \frac{1}{D_{min}(D_{min}+1)} + \frac{1}{D_{min}+1}\right)\|W^{(l)}\|_{op}. \tag{8}$$

Proposition 2 implies that the edge sensitivity of the GCN model shrinks significantly if the underlying graph has a reasonably large minimum degree, which will result in a significantly reduced noise scale that improves utility. However, the minimum degree assumption (7) is impractical since most of the realistic graph data have a large number of nodes with small degrees. To circumvent the impracticality of assumption (7) while still being able to reduce the noise scale in the GCN model, we propose a modification to the basic message passing algorithm 1 called *truncated message passing*. The idea of truncated message passing is to block all the incoming messages unless the receiver node's neighborhood is large than or equal to D_{min}, which is treated as a hyperparameter. For nodes with degrees lower than D_{min}, the output embedding is instead produced by an MLP with perturbation that does not involve any edge information. A detailed version is provided in algorithm 2 in appendix F. Consequently, it is straightforward to show that the differential privacy guarantee of the resulting algorithm operating on any graph matches the privacy level of perturbed GCN (produced by algorithm 1) operating only on graphs with minimum degree assumption.

How to choose D_{min}? To maintain the same privacy level under the truncated message passing algorithm, one may reduce the noise scale θ at the cost of raising the minimum degree hyperparameter D_{min}. On the one hand, reducing the noise scale significantly improves the utility of the message-passing procedure. On the other hand, raising D_{min} might prevent a non-ignorable proportion of nodes from learning structural information. Therefore, properly adjusting D_{min} may help achieve a better privacy-utility trade-off in the GCN model. In practice, one may choose D_{min} based on prior knowledge about the degree distribution of the underlying graph or via inspecting a private release of its degree distribution, which could be done efficiently using the Laplace mechanism [11].

2.5 VESPER: an end-to-end learning framework

In previous sections, we have established the PMP framework for differentially private graph representation learning. Now under the vertically federated learning setup described in section 2.1,

we propose an end-to-end architecture inspired by the SplitNN paradigm [14] based on the PMP framework, named **VE**rtically private **S**plit GNN with **PER**turbed message passing(**VESPER**). The VESPER architecture contains three main components: Encoder, Private representation extractor (PRE), and Decoder.

Encoder The encoder module maps input node features into a d-dimensional representation vector, with an ad-hoc choice being an MLP. Note that for node features with additional structural patterns (i.e., sequence data), we may use a more tailored encoder architecture as long as it does not involve edge information. The encoder model is physically stored in party A.

Private representation extractor The PRE module takes its input the node embeddings produced by the encoder and a batch of B subgraphs produced by a neighborhood sampler. The output representation of PRE is computed using some specific type of PMP mechanism such as PMP-GIN or PMP-GCN. The PRE module is physically stored in party A. The output of PRE is a tensor of shape $B \times d \times L$, with d and L being the dimension of graph representation and the number of message passing layers respectively. The outputs will be transmitted from party A to party B.

Decoder The decoder module is physically stored in party B, which decodes the received node embeddings produced by PRE into the final prediction of VESPER with its structure depending on the downstream tasks (i.e., classification, regression, ranking, etc.). We test two types of decoder architectures in our implementation of VESPER. The first one proceeds via concatenating the node embeddings of all layers followed by an MLP, which we call the CONCAT decoder. The second one treats the node embeddings as a sequence of L node embeddings and uses a GRU network to combine them, similar to the idea used in GNN architectures like GaAN [25] and GeniePath [30] which we term the GRU decoder.

The VFL training protocol closely resembles the SplitNN protocol [14], where in each step, forward computation results (i.e., the outputs of the PRE module) are transmitted from party A to party B. After party B finishes the forward computation using the decoded outputs and label information, party B first update its local decoder module via back-propagation, and then sends (partial) gradients that are intermediate results of the backward computation to party A for updating party A's local parameters (i.e., parameters of the encoder module and PRE module). A pictorial illustration of the VESPER architecture is presented in figure 1. We will discuss some practical issues in implementing VESPER in appendix E.1.

3 Experiments

In this section we present empirical evaluations of the VESPER framework via investigate its privacy-utility trade-off and resistance to empirical membership inference attacks. Due to limited space, a complete report will be postponed to appendix C.

3.1 Datasets

We use three large-scale graph datasets, with their summary statistics listed in table 2. Specifically, we use two public datasets ogbn-products and Reddit, with their detailed descriptions postponed to appendix C.1. We additionally used an industrial dataset called **the Finance dataset** which is generated from transaction records collected from one of the world's leading online payment systems. The underlying graph is generated by treating users as nodes, and two nodes are connected if at least one transaction occurred between corresponding users within a predefined time period. The business goal is to identify risky users which is cast into an algorithmic problem of node classification with a binary label. The node features are obtained via statistical summaries of corresponding users' behavior on the platform during a specific time period. The training and testing datasets are constructed under two distinct time windows with no overlap.

A differentially private analysis of degree profiles While all three datasets are large in scale (i.e., with the number of nodes exceeding $100,000$), they differ significantly in their degree distributions. For a better illustration, we conduct a differentially private analysis of degree distribution (with $(0.1, 0)$-differential privacy) detailed in appendix C.2. According to the analysis, we find that both the ogbn-products and the Reddit contain a large portion of high-degree nodes (as illustrated by the spiking bar at the ≥ 50 category), while the Finance dataset exhibits a concentration on the lower-degree nodes. As discussed in section 2.2, it is expected that the Finance dataset is more challenging for (private) message passing under sum pooling.

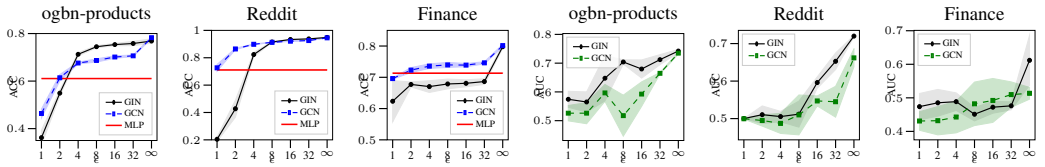

Figure 2: (a)-(c): Evaluation of privacy-utility trade-off regarding the VESPER framework, with mean $\pm$ std plotted according to 10 trials. The result of MLP is plotted as a reference line. Results below this line are practically problematic as it fails to exploit the graph information. (d)-(f): AUC (mean $\pm$ std over 10 trials) of membership inference attacks.

Table 1: Experimental results over three benchmark datasets using both non-private and private approaches, reported with format $\texttt{mean}_{\pm\texttt{std}}$, with `mean` and `std` (abbreviation for standard deviation) computed under 10 trials for each setting.

Non-private approaches												
Model	ogbn-products				Reddit				Finance			
MLP	$61.06_{\pm 0.08}$				$71.07_{\pm 0.25}$				$71.30_{\pm 0.17}$			
GIN	$76.84_{\pm 0.94}$				$94.85_{\pm 0.15}$				$79.78_{\pm 0.52}$			
GCN	$78.26_{\pm 0.36}$				$94.56_{\pm 0.10}$				$80.13_{\pm 0.41}$			
Private approaches												
Model	ogbn-products				Reddit				Finance			
	$\epsilon = 4$	$\epsilon = 8$	$\epsilon = 16$	$\epsilon = 32$	$\epsilon = 4$	$\epsilon = 8$	$\epsilon = 16$	$\epsilon = 32$	$\epsilon = 4$	$\epsilon = 8$	$\epsilon = 16$	$\epsilon = 32$
VFGNN	$26.94_{\pm 0.00}$	$40.96_{\pm 1.74}$	$56.27_{\pm 1.35}$	$69.57_{\pm 0.26}$	$19.40_{\pm 2.56}$	$31.20_{\pm 1.75}$	$43.67_{\pm 1.18}$	$86.72_{\pm 0.33}$	$53.87_{\pm 6.86}$	$56.17_{\pm 5.31}$	$54.62_{\pm 4.09}$	$52.12_{\pm 4.78}$
GAP	$59.20_{\pm 2.13}$	$65.02_{\pm 0.82}$	$66.20_{\pm 2.34}$	$67.14_{\pm 0.34}$	$76.84_{\pm 1.71}$	$86.59_{\pm 0.48}$	$88.57_{\pm 1.69}$	$89.65_{\pm 0.30}$	$52.02_{\pm 9.39}$	$48.66_{\pm 7.14}$	$59.04_{\pm 7.95}$	$67.54_{\pm 4.41}$
VESPER (GIN)	$71.27_{\pm 0.70}$	$74.46_{\pm 0.45}$	$75.36_{\pm 0.49}$	$75.82_{\pm 0.92}$	$82.34_{\pm 1.57}$	$91.52_{\pm 0.22}$	$93.34_{\pm 0.20}$	$93.77_{\pm 0.23}$	$67.03_{\pm 1.88}$	$67.87_{\pm 1.56}$	$68.08_{\pm 1.12}$	$68.64_{\pm 0.91}$
VESPER (GCN)	$67.60_{\pm 0.40}$	$68.68_{\pm 0.67}$	$70.13_{\pm 0.55}$	$70.62_{\pm 0.33}$	$89.85_{\pm 0.27}$	$91.28_{\pm 0.17}$	$92.11_{\pm 0.20}$	$92.57_{\pm 0.14}$	$73.57_{\pm 0.73}$	$73.96_{\pm 0.81}$	$73.90_{\pm 0.52}$	$74.62_{\pm 0.35}$

3.2 Baselines

We compare the proposed VESPER framework with three types of baselines, with each one being able to implement in the vertically federated setting. **MLP without edge information** we use MLP over node features directly is the most trivial solution to the learning task as it totally ignores edge information. **Non-private GNN counterparts** we compare with ordinary GCN and GIN models without privacy guarantees, or equivalently set the ϵ parameter in the VESPER framework to be infinity. **GNN models with privacy guarantees** we consider two alternative approaches to private GRL, namely the VFGNN model [53] and the GAP model [39]. We found the privacy analysis in the corresponding papers to be somewhat incoherent with the privacy model in our paper and we conducted new analysis of their privacy properties, detailed in appendix C.3.

3.3 Experimental setup

Due to limited space, we postpone the description of our training configurations to appendix C.4 and elaborate more on the **privacy configurations**: All the privacy reports are based on the (ϵ, δ)-differential privacy model, with δ being the reciprocal of the number of edges. To adequately inspect the privacy-utility trade-off, we aim to evaluate all the models with differential privacy guarantees under the total privacy costs (privacy budgets) $\epsilon \in \{1, 2, 4, 8, 16, 32\}$, with the privacy costs accounted during the entire training period. We treat the setting where $\epsilon \in \{1, 2\}$ as of *high privacy*, $\epsilon \in \{4, 8\}$ as of *moderate privacy*, and the rest as of *low privacy*. For VESPER and VFGNN, we add spectral normalization to each GNN layer. For the privacy accountant, we base our implementation upon AMA implementation available in the `dp-accounting` library and use an adjusted sampling probability according to theorem 2.1. For each required privacy level, we compute the minimum scale of Gaussian noise via conducting a binary search over the adjusted AMA, with associating spectral norms of weight matrices fixed at one in all layers.

Evaluation metrics We adopt classification accuracy (ACC) as the evaluation metric for the ogbn-products and Reddit datasets, and ROC-AUC score (AUC) as the evaluation metric for the Finance dataset.

3.4 Performance and privacy-utility trade-off

According to our empirical experience, obtaining reasonable performance in the *high privacy* regime is difficult, especially for baseline algorithms. Therefore, we report two sets of results: Firstly, we thoroughly investigate the privacy-utility trade-off regarding the proposed VESPER framework under both GIN and GCN aggregation schemes and plot the results in figure 2. Secondly, we report comparisons of VESPER against private and non-private baselines with only moderate to large privacy budgets and summarize the results in table 1. The results demonstrate that the proposed VESPER framework exhibits competitive privacy-utility trade-off under both GIN and GCN aggregators. Moreover, a comparison of GIN and GCN aggregator suggests that summation pooling excels when the underlying graph is dense (i.e., ogbn-products and Reddit), while introducing the truncated message passing mechanism helps achieving better results over sparse graphs (i.e., Finance). Finally, VESPER demonstrates a better privacy-utility trade-off compared to other private GNN baselines.

3.5 Protection against membership inference attacks

We launch a membership inference attack (MIA) [37] to empirically investigate the resilience of VESPER against practical privacy risks that targets the membership of nodes instead of edges, which is regarded as a stronger attack than edge MIA. We provide a detailed description of the attack setup in appendix C.7. The attack is conducted over trained models under privacy budgets $\epsilon \in \{1, 2, 4, 8, 16, 32, \infty\}$, where $\epsilon = \infty$ indicated no privacy protection is adopted. We use ROC-AUC (AUC) to evaluate the attack performance. We report the attack performances in Figure 4. From the results, we observe that when privacy protection is disabled ($\epsilon = \infty$), the attacks show non-negligible effectiveness, especially on obgn-products and Reddit datasets. Generally, with the privacy budget getting smaller (privacy getting stronger), the attack performances sharply decline. With an appropriate privacy budget, the attacks on all three datasets are successfully defended with AUC reduced to around 0.5 (random guess baseline).

Additional experiments We will report a series of ablation studies that assess the effect of maximum degree D, minimum degree D_{min} for PMP-GCN and batch size in appendix C.8.

4 Related Works

4.1 Graph representation learning in the federated setting

The majority of GRL research in the federated setting is based on the horizontal setup, with each party holding its own local graph data [45, 17, 38]. The adoption of VFL paradigms to GRL is relatively few, VFGNN [53] uses additive secret sharing to combine feature information held by different parties, followed by a straightforward adaptation of the SplitNN framework [14] with the underlying neural model being graph neural networks. In [5, 46], the authors discussed VFL setups where node features and graph topology belong to different parties. We refer to the recent survey [27] for a more detailed overview.

4.2 Graph representation learning with differential privacy guarantees

The most straightforward way to integrate DP techniques into GRL is via adopting private optimization algorithms like DP-SGD[1]. However, meaningful notions of differential privacy over graph data (i.e., the edge model [35] and node model [22]) are semantically different from that of i.i.d. data, and require refined privacy analysis which is sometimes ignored in the privacy analysis in previous works [53, 45, 36]. In [7], the authors analyzed the DP-SGD algorithm in the node DP model. The GAP model [39] proposed a three-stage training procedure and analyzed its privacy guarantee in both edge DP and node DP models. However, we noticed that the privacy analysis in [39] did not properly address the effect of sampling, resulting in an overly optimistic performance. Considering only edge DP, randomized response (RR) [46] that flips each entry of the underlying graph's adjacent matrix guarantees privacy (in a stronger *local* sense), but makes reasonable privacy-utility trade-off extremely hard to obtain in practice.

5 Conclusion and discussions

We present the VESPER framework as a differentially private solution to node classification in the VFL setup using graph representation learning techniques. The core algorithmic component of VESPER is the PMP scheme that allows efficient learning under both dense and sparse graph

data. We demonstrate the practicality and effectiveness of the proposed framework by establishing theoretical DP guarantees as well as investigating its ability for privacy protection and privacy-utility trade-off empirically. We will discuss possible extensions and future directions of the VESPER framework in appendix D.

References

[1] 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*, pages 308–318, 2016.

[2] B. Balle, G. Barthe, M. Gaboardi, J. Hsu, and T. Sato. Hypothesis testing interpretations and renyi differential privacy. In *International Conference on Artificial Intelligence and Statistics*, pages 2496–2506. PMLR, 2020.

[3] T. Chen, X. Jin, Y. Sun, and W. Yin. Vafl: a method of vertical asynchronous federated learning. *arXiv preprint arXiv:2007.06081*, 2020.

[4] Z. Chen, L. D. Van Khoa, E. N. Teoh, A. Nazir, E. K. Karuppiah, and K. S. Lam. Machine learning techniques for anti-money laundering (aml) solutions in suspicious transaction detection: a review. *Knowledge and Information Systems*, 57(2):245–285, 2018.

[5] T.-H. Cheung, W. Dai, and S. Li. Fedsgc: Federated simple graph convolution for node classification. In *IJCAI Workshops*, 2021.

[6] F. Chung and L. Lu. The diameter of sparse random graphs. *Advances in Applied Mathematics*, 26(4):257–279, 2001.

[7] A. Daigavane, G. Madan, A. Sinha, A. G. Thakurta, G. Aggarwal, and P. Jain. Node-level differentially private graph neural networks. *arXiv preprint arXiv:2111.15521*, 2021.

[8] T. Dang, O. Thakkar, S. Ramaswamy, R. Mathews, P. Chin, and F. Beaufays. Revealing and protecting labels in distributed training. *Advances in Neural Information Processing Systems*, 34:1727–1738, 2021.

[9] V. Duddu, A. Boutet, and V. Shejwalkar. Quantifying privacy leakage in graph embedding. In *MobiQuitous 2020-17th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, pages 76–85, 2020.

[10] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

[11] C. Dwork, A. Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[12] B. Ghazi, N. Golowich, R. Kumar, P. Manurangsi, and C. Zhang. Deep learning with label differential privacy. *Advances in Neural Information Processing Systems*, 34, 2021.

[13] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. Neural message passing for quantum chemistry. In D. Precup and Y. W. Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1263–1272, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[14] O. Gupta and R. Raskar. Distributed learning of deep neural network over multiple agents. *Journal of Network and Computer Applications*, 116:1–8, 2018.

[15] W. Hamilton, Z. Ying, and J. Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

[16] W. L. Hamilton. Graph representation learning. *Synthesis Lectures on Artifical Intelligence and Machine Learning*, 14(3):1–159, 2020.

[17] C. He, K. Balasubramanian, E. Ceyani, C. Yang, H. Xie, L. Sun, L. He, L. Yang, P. S. Yu, Y. Rong, et al. Fedgraphnn: A federated learning system and benchmark for graph neural networks. *arXiv preprint arXiv:2104.07145*, 2021.

[18] W. Hu, M. Fey, M. Zitnik, Y. Dong, H. Ren, B. Liu, M. Catasta, and J. Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133, 2020.

[19] X. Jin, P.-Y. Chen, C.-Y. Hsu, C.-M. Yu, and T. Chen. Cafe: Catastrophic data leakage in vertical federated learning. *Advances in Neural Information Processing Systems*, 34:994–1006, 2021.

[20] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, R. G. L. D'Oliveira, H. Eichner, S. E. Rouayheb, D. Evans, J. Gardner, Z. Garrett, A. Gascón, B. Ghazi, P. B. Gibbons, M. Gruteser, Z. Harchaoui, C. He, L. He, Z. Huo, B. Hutchinson, J. Hsu, M. Jaggi, T. Javidi, G. Joshi, M. Khodak, J. Konecný, A. Korolova, F. Koushanfar, S. Koyejo, T. Lepoint, Y. Liu, P. Mittal, M. Mohri, R. Nock, A. Özgür, R. Pagh, H. Qi, D. Ramage, R. Raskar, M. Raykova, D. Song, W. Song, S. U. Stich, Z. Sun, A. T. Suresh, F. Tramèr, P. Vepakomma, J. Wang, L. Xiong, Z. Xu, Q. Yang, F. X. Yu, H. Yu, and S. Zhao. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021.

[21] P. Kairouz, S. Oh, and P. Viswanath. The composition theorem for differential privacy. In *International conference on machine learning*, pages 1376–1385. PMLR, 2015.

[22] S. P. Kasiviswanathan, K. Nissim, S. Raskhodnikova, and A. Smith. Analyzing graphs with node differential privacy. In *Proceedings of the 10th Theory of Cryptography Conference on Theory of Cryptography*, TCC'13, page 457–476, Berlin, Heidelberg, 2013. Springer-Verlag.

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

[24] D. V. Kute, B. Pradhan, N. Shukla, and A. Alamri. Deep learning and explainable artificial intelligence techniques applied for detecting money laundering–a critical review. *IEEE Access*, 9:82300–82317, 2021.

[25] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*, 2015.

[26] C. Liu, L. Sun, X. Ao, J. Feng, Q. He, and H. Yang. Intention-aware heterogeneous graph attention networks for fraud transactions detection. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 3280–3288, 2021.

[27] R. Liu and H. Yu. Federated graph neural networks: Overview, techniques and challenges. *arXiv preprint arXiv:2202.07256*, 2022.

[28] Y. Liu, Y. Kang, T. Zou, Y. Pu, Y. He, X. Ye, Y. Ouyang, Y.-Q. Zhang, and Q. Yang. Vertical federated learning. *arXiv preprint arXiv:2211.12814*, 2022.

[29] Y. Liu, X. Zhang, Y. Kang, L. Li, T. Chen, M. Hong, and Q. Yang. Fedbcd: A communication-efficient collaborative learning framework for distributed features. *IEEE Transactions on Signal Processing*, 70:4277–4290, 2022.

[30] Z. Liu, C. Chen, L. Li, J. Zhou, X. Li, L. Song, and Y. Qi. Geniepath: Graph neural networks with adaptive receptive paths. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4424–4431, 2019.

[31] A. Mashrur, W. Luo, N. A. Zaidi, and A. Robles-Kelly. Machine learning for financial risk management: A survey. *IEEE Access*, 8:203203–203223, 2020.

[32] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang. Learning differentially private recurrent language models. *arXiv preprint arXiv:1710.06963*, 2017.

[33] I. Mironov. Rényi differential privacy. In *2017 IEEE 30th computer security foundations symposium (CSF)*, pages 263–275. IEEE, 2017.

[34] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

[35] K. Nissim, S. Raskhodnikova, and A. Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 75–84, 2007.

[36] I. E. Olatunji, T. Funke, and M. Khosla. Releasing graph neural networks with differential privacy guarantees. *arXiv preprint arXiv:2109.08907*, 2021.

[37] I. E. Olatunji, W. Nejdl, and M. Khosla. Membership inference attack on graph neural networks. In *2021 Third IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*, pages 11–20. IEEE, 2021.

[38] M. Ramezani, W. Cong, M. Mahdavi, M. T. Kandemir, and A. Sivasubramaniam. Learn locally, correct globally: A distributed algorithm for training graph neural networks. *arXiv preprint arXiv:2111.08202*, 2021.

[39] S. Sajadmanesh, A. S. Shamsabadi, A. Bellet, and D. Gatica-Perez. Gap: Differentially private graph neural networks with aggregation perturbation. *arXiv preprint arXiv:2203.00949*, 2022.

[40] L. Song and P. Mittal. Systematic evaluation of privacy risks of machine learning models. In *USENIX Security Symposium*, volume 1, page 4, 2021.

[41] A. T. Suresh, X. Y. Felix, S. Kumar, and H. B. McMahan. Distributed mean estimation with limited communication. In *International conference on machine learning*, pages 3329–3337. PMLR, 2017.

[42] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

[43] M. Y. Wang. Deep graph library: Towards efficient and scalable deep learning on graphs. In *ICLR workshop on representation learning on graphs and manifolds*, 2019.

[44] Y.-X. Wang, B. Balle, and S. P. Kasiviswanathan. Subsampled rényi differential privacy and analytical moments accountant. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1226–1235. PMLR, 2019.

[45] C. Wu, F. Wu, Y. Cao, Y. Huang, and X. Xie. Fedgnn: Federated graph neural network for privacy-preserving recommendation. *arXiv preprint arXiv:2102.04925*, 2021.

[46] F. Wu, Y. Long, C. Zhang, and B. Li. Linkteller: Recovering private edges from graph neural networks via influence analysis. *arXiv preprint arXiv:2108.06504*, 2021.

[47] R. Wu, B. Ma, H. Jin, W. Zhao, W. Wang, and T. Zhang. Grande : a neural model over directed multigraphs with application to anti-money laundering. *IEEE International Conference on Data Mining*, 2022.

[48] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

[49] Q. Yang, Y. Liu, T. Chen, and Y. Tong. Federated machine learning: Concept and applications. *ACM Trans. Intell. Syst. Technol.*, 10(2), jan 2019.

[50] H. Yin, A. Mallya, A. Vahdat, J. M. Alvarez, J. Kautz, and P. Molchanov. See through gradients: Image batch recovery via gradinversion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16337–16346, 2021.

[51] C. Ying, T. Cai, S. Luo, S. Zheng, G. Ke, D. He, Y. Shen, and T.-Y. Liu. Do transformers really perform badly for graph representation? *Advances in Neural Information Processing Systems*, 34:28877–28888, 2021.

[52] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 974–983, 2018.

[53] J. Zhou, C. Chen, L. Zheng, H. Wu, J. Wu, X. Zheng, B. Wu, Z. Liu, and L. Wang. Vertically federated graph neural network for privacy-preserving node classification. *arXiv preprint arXiv:2005.11903*, 2020.

[54] L. Zhu, Z. Liu, and S. Han. Deep leakage from gradients. *Advances in neural information processing systems*, 32, 2019.

A Some standard tools for Rényi differential privacy

Rényi divergence the Rényi divergence between distributions of random variables X and Y given by

$$D_\alpha\left(X||Y\right) = \frac{1}{\alpha-1}\log \mathbb{E}_{y\sim\mathbb{P}_Y}\left[\left(\frac{d\mathbb{P}_X}{d\mathbb{P}_Y}(y)\right)^\alpha\right]. \tag{9}$$

Here we use $\frac{d\mathbb{P}_X}{d\mathbb{P}_Y}(\cdot)$ to denote the density ratio between X and Y (or more formally the Radon-Nikodym derivative of the induced probability measure $\mathbb{P}_X$ with respect to $\mathbb{P}_Y$). We state here a

couple of useful results in implementing and proving algorithms with Rényi differential privacy. The results will be stated under the context of graph algorithms in the edge DP model. The first lemma is the composition theorem of RDP:

Lemma 1 (Composition of Rényi DP [33]). *Let $\mathcal{M}_1$ be a graph-input mechanism that satisfies (α, ϵ_1)-RDP, and $\mathcal{M}_2$ be a graph-input mechanism that is allowed to further depend on the output of $\mathcal{M}_1$ satisfying (α, ϵ_2)-RDP, then the composed mechanism $(\mathcal{M}_1 \circ \mathcal{M}_2)(G) = \mathcal{M}_2(\mathcal{M}_1(G), G)$ satisfies $(\alpha, \epsilon_1 + \epsilon_2)$-RDP.*

The second lemma is the conversion rule of RDP to the approximate (ϵ, δ)-DP:

Lemma 2 (Conversion of RDP to (ϵ, δ)-DP, [2]). *Let mechanism $\mathcal{M}$ satisfy (α, ϵ)-RDP, then it is (ϵ', δ)-DP for*

$$\epsilon' = \epsilon - \frac{\log(\delta\alpha)}{\alpha - 1} + \log(1 - \frac{1}{\alpha}) \tag{10}$$

with any $\delta > 0$.

B Missing proofs

Proof of theorem 2.1. The proof contains two steps: In the first step, we prove that without neighborhood sampling, the algorithm is $(\alpha, \epsilon(\alpha))$-RDP. Then in the second step, we construct an algorithm that is *less or equally private* than the procedure 1 and could be directly analyzed by [44, Theorem 9] such that the privacy guarantee of the algorithm is the one stated in the theorem.

Step 1: We ignore neighborhood sampling and consider the first layer. By [33, proposition 3], the collection of perturbed embeddings $\{\check{h}_v^{(1)}\}_{v \in V}$, with $\check{h}_v^{(1)} = \widetilde{h}_v^{(1)} + N(0, \theta^2 I_d)$, is $\left(\alpha, \frac{\alpha \mathcal{S}_1^2}{2\theta^2}\right)$-Rényi differentially private for any $\alpha > 1$. Since the nonlinear transform does not involve edge information and is therefore treated as a post-processing mechanism [11], it follows that the collection of transformed embeddings $\{h_v^{(1)}\}_{v \in V}$, with $h_v^{(1)} = \sigma\left(\check{h}_v^{(1)}\right)$, is also $\left(\alpha, \frac{\alpha \mathcal{S}_1^2}{2\theta^2}\right)$-Rényi differentially private for any $\alpha > 1$. Now we view the operation in a single layer as a base mechanism, an L-layer perturbed message passing procedure could thus be viewed as composing the base mechanism for L times. Then it follows by the composition theorem of Rényi differential privacy [33, Proposition 1] that the non-sampling version is $(\alpha, \epsilon(\alpha))$-RDP.

Step 2: First we introduce some additional notations: Denote $G_v^{(L)}$ as the L-layer rooted subgraph with root node $v \in V$ produced by a neighborhood sampler. Then each training batch consists of B randomly chosen subgraphs $(G_{v_1}^{(L)}, \ldots, G_{v_B}^{(L)})$ with root nodes $(v_1, \ldots, v_B)$, further denote $G_B^{(L)}$ as the graph generated via combining $(G_{v_1}^{(L)}, \ldots, G_{v_B}^{(L)})$ with node set $V_B^{(L)}$ and edge set $E_B^{(L)}$. Let N_e be the maximum number of possible subgraphs that any specific edge might affect after an L-layer message passing procedure, then we may bound the probability of the event that any specific edge $e \in E$ is contained in $G_B^{(L)}$

$$\max_{e \in E} \mathbb{P}\left[e \in E_B^{(L)}\right] \leq 1 - \frac{\binom{N_T - N_e}{B}}{\binom{N_T}{B}} \tag{11}$$

Since the maximum degree is bounded from above by D, we further bound the above probability by bounding N_e

$$N_e \leq 2 \sum_{l=0}^{L-1} D^l = \frac{2(D^L - 1)}{D - 1},$$

yielding

$$\max_{e \in E} \mathbb{P}\left[e \in E_B^{(L)}\right] \leq \gamma := 1 - \frac{\binom{N_T - \frac{2(D^L - 1)}{D-1}}{B}}{\binom{N_T}{B}}$$

Next, we construct an algorithm $\mathcal{A}$ as follows: For a batch of size B, the algorithm first randomly samples B nodes, then independently samples $\lfloor \gamma |E| \rfloor$ edges to form a subgraph $G_B^{\mathcal{A}}$. Then it returns

the result via running a non-sampled version of algorithm 1 over $G_B^{\mathcal{A}}$. Here note the fact that for any edge e, the probability that e is contained in $G_B^{(L)}$ is no greater than the probability that it is contained in $G_B^{\mathcal{A}}$. Therefore, algorithm $\mathcal{A}$ is less or equally private than the procedure 1.

Since the privacy guarantee of algorithm $\mathcal{A}$ can be directly analyzed by [44, Theorem 9], yielding a Rényi differential privacy guarantee of $(\alpha, \epsilon_\gamma(\alpha))$ with $\epsilon_\gamma(\alpha)$ defined in (6). The result of the theorem follows. □

In the proofs of propotision 1 and 2, **we will prove for an arbitrary weight matrix** W and the result trivially applies to the weight matrices in each layer of the message passing procedure.

Proof of proposition 1. We inherit the notation from definition 3 that G' is the adjacent graph via removing the edge (u^*, v^*) from G. Write the summation pooling update rule as

$$\widetilde{h}_v \leftarrow \sum_{u \in N(v)} W h_u, \quad \forall v \in V \tag{12}$$

Note that the only two node embeddings that get affected by the removal is h_{v^*} and h_{u^*}. For node v^*, it follows that

$$\left\| \widetilde{h}_{v^*} - \widetilde{h}'_{v^*} \right\| = \|W h_{u^*}\| \le \|W\|_{\text{op}} \|h_{u^*}\| = \|W\|_{\text{op}}. \tag{13}$$

Where the last equality follows since the input representations are ℓ_2-normalized. The same argument leads to

$$\left\| \widetilde{h}_{u^*} - \widetilde{h}'_{u^*} \right\| \le \|W\|_{\text{op}}. \tag{14}$$

Then we arrive at

$$\begin{aligned} &\sqrt{\sum_{v \in V} \|\widetilde{h}_v - \widetilde{h}'_v\|_2^2} \\ =&\sqrt{\left\| \widetilde{h}_{u^*} - \widetilde{h}'_{u^*} \right\|^2 + \left\| \widetilde{h}_{v^*} - \widetilde{h}'_{v^*} \right\|^2} \\ \le&\sqrt{2} \|W\|_{\text{op}}. \end{aligned} \tag{15}$$

□

Proof of proposition 2. Recall the update rule of GCN [23]

$$\widetilde{h}_v \leftarrow \frac{W h_v}{d_v + 1} + \sum_{u \in N(v)} \frac{W h_u}{\sqrt{d_v + 1}\sqrt{d_u + 1}} \tag{16}$$

Following similar arguments in the proof of proposition 1, we first bound the difference between $\widetilde{h}_{v^*}$ and $\widetilde{h}'_{v^*}$ in ℓ_2 norm. First we inspect

$$\begin{aligned} &\widetilde{h}_{v^*} - \widetilde{h}'_{v^*} \\ =&- \frac{W h_{v^*}}{d_{v^*}(d_{v^*} + 1)} + \frac{W h_{u^*}}{\sqrt{d_{u^*} + 1}\sqrt{d_{v^*} + 1}} \\ &+ \sum_{u \in N(v^*) \setminus \{u^*\}} \left(\frac{W h_u}{\sqrt{d_v + 1}\sqrt{d_u + 1}} - \frac{W h_u}{\sqrt{d_v}\sqrt{d_u + 1}} \right) \\ :=&\mathcal{T}_1 + \mathcal{T}_2 + \mathcal{T}_3 \end{aligned} \tag{17}$$

Bounding $\mathcal{T}_1$ and $\mathcal{T}_2$ are straightforward

$$\mathcal{T}_1 \le \frac{\|W\|_{\text{op}}}{D_{\min}(D_{\min} + 1)} \tag{18}$$

$$\mathcal{T}_2 \le \frac{\|W\|_{\text{op}}}{D_{\min} + 1} \tag{19}$$

Where we use the minimum degree assumption (7). To bound $\mathcal{T}_3$, we use the inequality

$$\forall x > 0, \quad \frac{1}{x} - \frac{1}{x+1} \leq \frac{1}{2x^{3/2}} \tag{20}$$

Now we proceed as follows:

$$\begin{aligned}
&\|\mathcal{T}_3\| && (21)\\
\leq &\sum_{u \in N(v^*) \setminus \{u^*\}} \left\| \frac{Wh_u}{\sqrt{d_v+1}\sqrt{d_u+1}} - \frac{Wh_u}{\sqrt{d_v}\sqrt{d_u+1}} \right\| && (22)\\
\leq &\sum_{u \in N(v^*) \setminus \{u^*\}} \frac{\|W\|_{\text{op}}}{\sqrt{d_u+1}} \left(\frac{1}{\sqrt{d_v}} - \frac{1}{\sqrt{d_v+1}} \right) && (23)\\
\leq &\sum_{u \in N(v^*) \setminus \{u^*\}} \frac{\|W\|_{\text{op}}}{2\sqrt{d_u+1}d_v^{3/2}} && \text{By inequality (20)}\\
\leq &\sum_{u \in N(v^*) \setminus \{u^*\}} \frac{\|W\|_{\text{op}}}{2\sqrt{D_{\min}+1}d_v^{3/2}} && \text{By assumption (7)}\\
= &\frac{\|W\|_{\text{op}}(d_v-1)}{2\sqrt{D_{\min}+1}d_v^{3/2}} && (24)
\end{aligned}$$

To further bound (24), observe that the function

$$f(x) = \frac{x-1}{x^{3/2}}, \qquad x > 1 \tag{25}$$

attains its maximum at $x = 3$, and becomes monotonically decreasing as $x \geq 3$. Since $d_v \geq D_{\min}$, it suffices to check the case for $D_{\min} = 2$ and $D_{\min} \geq 3$ separately. For $D_{\min} = 2$, we have

$$\|\mathcal{T}_3\| \leq \frac{\|W\|_{\text{op}}(3-1)}{2\sqrt{2+1}3^{3/2}} < \frac{\|W\|_{\text{op}}(2-1)}{2\sqrt{2}2^{3/2}} = \frac{\|W\|_{\text{op}}(D_{\min}-1)}{2\sqrt{D_{\min}}D_{\min}^{3/2}} \tag{26}$$

For $D_{\min} \geq 3$, we have

$$\|\mathcal{T}_3\| \leq \frac{\|W\|_{\text{op}}(D_{\min}-1)}{2\sqrt{D_{\min}+1}D_{\min}^{3/2}} < \frac{\|W\|_{\text{op}}(D_{\min}-1)}{2\sqrt{D_{\min}}D_{\min}^{3/2}} \tag{27}$$

Combining (26) and (27) we get

$$\|\mathcal{T}_3\| \leq \frac{\|W\|_{\text{op}}(1-1/D_{\min})}{2D_{\min}}. \tag{28}$$

Finally, combine (18), (19) and (28), and then use the argument in (15) yield the result. □

C A complete report of empirical evaluations

C.1 Datasets

The ogbn-products dataset is an undirected and unweighted graph that represents an Amazon product co-purchasing network [18]. Nodes represent products sold on Amazon, and edges between two products indicate that the products are purchased together. The node features are generated as dimensionality-reduced bag-of-words of the product descriptions. The learning task is to predict the category of a product in a multi-class classification setup with 47 classes. We took the dataset and train/validation/test splitting from the official implementation available in the ogb library.

The Reddit dataset is a graph dataset from Reddit posts made in September, 2014. Nodes represent Reddit posts; two posts are connected if the same user comments on both. The node features are generated by combining the word embeddings of the corresponding post's metadata, as described in [15]. The learning task is to predict which community different Reddit posts belong to, with 41 classes. We use the training/validation/testing splitting from [15].

The Finance dataset This dataset is generated from transaction records collected from one of the world's leading online payment systems. The underlying graph is generated by treating users as

Table 2: Summary statistics of the evaluation datasets

	ogbn-products	Reddit	Alipay
# Nodes	2449029	232965	1132511
# Edges	123718280	114615892	2447370
# Training nodes	196615	153431	848963
# Node features	100	602	155
# Classes	47	41	2

nodes, and two nodes are connected if at least one transaction occurred between corresponding users within a predefined time period. The business goal is to identify risky users which is cast into an algorithmic problem of node classification with a binary label. The node features are obtained via statistical summaries of corresponding users' behavior on the platform during a specific time period. The training and testing datasets are constructed under two distinct time windows with no overlap.

We list the summary statistics in table 2

C.2 A differentially private analysis of degree distributions

While all three datasets are large in scale (i.e., with the number of nodes exceeding $100,000$), they differ significantly in their degree distributions. Specifically, the average node degree is much higher in the ogbn-products (≈ 50) and Reddit dataset (≈ 490) than that in the Finance dataset (≈ 2.2). Following literature in random graph theory [6], we might consider ogbn-products and Reddit as dense graphs (with average degree $\gg \log(N)$) and Finance as a sparse graph (with average degree $\ll \log(N)$). For a better illustration, we conduct a differentially private analysis of degree distribution (with $(0.1, 0)$-differential privacy). Since we are basically interested in graphs with bounded degrees (and enforcing the property using neighborhood sampling), during the computation of degree distributions, we group all nodes with degrees over 50 to a single category (i.e., with degrees greater than or equal to 50). As a result, the final histogram represents counts of nodes under degree $\{0, 1, \ldots, 49, \geq 50\}$. The analysis is based on the trivial fact that the addition and removal of any single edge would change the degree of 2 nodes by exactly 1, therefore, the ℓ_1 sensitivity [11] of the degree distribution histogram query is exactly 2. By standard Laplacian mechanism [10, 11], we add to each count an independent Laplacian noise with scale $\frac{2}{\epsilon}$ with $\epsilon = 0.1$. The resulting private histograms are shown in figure 3 with counts reported at a logarithmic scale. According

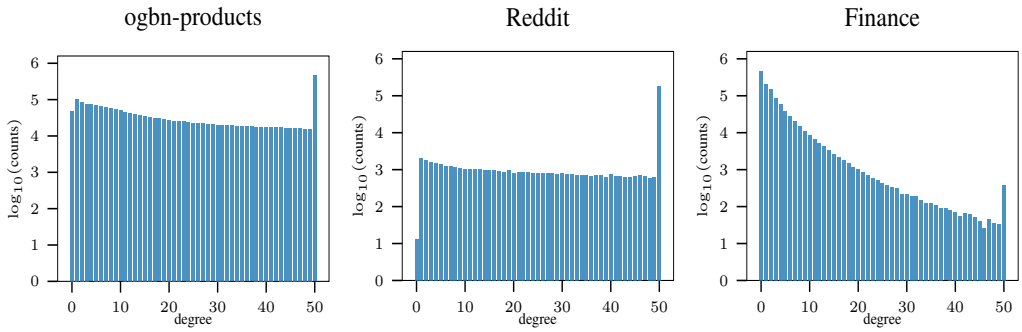

Figure 3: $(0.1, 0)$-differentially private histograms of three benchmark datasets

to the histograms, we find that both the ogbn-products and the Reddit contain a large portion of high-degree nodes (as illustrated by the spiking bar at the ≥ 50 category), while the Finance dataset exhibits a concentration on the lower-degree nodes. In particular, around half of the nodes in the Finance dataset are singular nodes without any neighbors. According to the discussion in section 2.2, it is expected that the Finance dataset is more challenging for (private) message passing under sum pooling.

C.3 Baselines

MLP without edge information Using an MLP over node features directly is the most trivial solution to the learning task as it totally ignores edge information. Equivalently, this corresponds to removing the PRE module in the VESPER framework. This baseline is of critical importance in evaluating the privacy-utility trade-off since in practice we require the model trained with graph information to significantly outperform MLPs.

Non-private GNN counterparts We compare with ordinary GCN and GIN models without privacy guarantees, or equivalently set the ϵ parameter in the VESPER framework to be infinity. Ideally, the performance of these models should serve as performance upper bounds for corresponding VESPER models.

GNN models with privacy guarantees We consider two alternative approaches to private GRL, namely the VFGNN model [53] and the GAP model [39]. Both models add Gaussian noise to node embeddings and are implementable in the vertically federated setting. The privacy guarantees stated in [53] are not directly applicable to the edge privacy setup. Thus we provide an independent privacy analysis in the edge privacy model, similar to theorem 2.1. For the GAP model [39], the effect of sampling was not properly addressed in the original paper, and we instead use a corrected version by noting that the aggregation perturbation mechanism is equivalent to PMP-GIN without learnable parameters and use theorem 2.1 to analyze it.

C.4 Experimental setup

Training configurations Across all models (i.e., MLP or GNN-related baselines), we used a hidden dimension of $d = 128$ for the ogbn-products dataset, $d = 512$ for the Reddit dataset and $d = 256$ for the Finance dataset. We use the Adam optimizer with a learning rate 0.001 across all the tasks. We trained each model for 5 epochs under the ogbn-products and Reddit dataset and 2 epochs under the Finance dataset. For the GNN-related approaches, according to the private degree histogram analysis, we tune the maximum degree with range $\{20, 50\}$ for ogbn-products and Reddit datasets and $\{10, 20\}$ for the Finance dataset. For VESPER, we tested different decoder architectures as described in section 2.5. For VESPER using the GCN aggregator, we tune the minimum degree hyperparameter over $\{10, 20, 40\}$ for ogbn-products and Reddit and $\{3, 5\}$ for Finance. We tested the number of message passing rounds with $L \in \{1, 2, 3\}$. We found that $L = 2$ works best in general across all datasets for VESPER and GAP, and $L = 1$ works best for VFGNN. We use the `DGL` framework [43] for the implementation of GNN algorithms.

privacy configurations: All the privacy reports are based on the (ϵ, δ)-differential privacy model, with δ being the reciprocal of the number of edges. To adequately inspect the privacy-utility trade-off, we aim to evaluate all the models with differential privacy guarantees under the total privacy costs (privacy budgets) $\epsilon \in \{1, 2, 4, 8, 16, 32\}$, with the privacy costs accounted during the entire training period. We treat the setting where $\epsilon \in \{1, 2\}$ as of *high privacy*, $\epsilon \in \{4, 8\}$ as of *moderate privacy*, and the rest as of *low privacy*. For VESPER and VFGNN, we add spectral normalization to each GNN layer. For the privacy accountant, we base our implementation upon AMA implementation available in the `dp-accounting` library and use an adjusted sampling probability according to theorem 2.1. For each required privacy level, we compute the minimum scale of Gaussian noise via conducting a binary search over the adjusted AMA, with associating spectral norms of weight matrices fixed at one in all layers.

Evaluation metrics We adopt classification accuracy (ACC) as the evaluation metric for the ogbn-products and Reddit datasets, and ROC-AUC score (AUC) as the evaluation metric for the Finance dataset.

C.5 A privacy analysis for VFGNN [53]

The VFGNN model [53] adds Gaussian noise to the normalized output of an L layer message passing. It is trivial to check that under the edge differential privacy model with noise scale θ, VFGNN is $\left(\alpha, \frac{\alpha \mathcal{S}_L^2}{2\theta^2}\right)$-Rényi differentially private for any $\alpha > 1$, with edge sensitivity $\mathcal{S}_L$ slightly generalized (c.f. definition 3) with h and h' being the output of an L-layer message passing procedure. Without loss of generality, we assume the norm of node embeddings before perturbation to be C. To compute $\mathcal{S}_L$, first note that the change of any node embedding under an edge addition or removal operation is bounded by $2C$, it remains to bound the number of node embeddings that may get affected upon

an edge addition or removal operation. Through similar arguments in the proof of theorem 2.1, we bound this count from above by $2\sum_{l=0}^{L-1} D^l$. We conclude the analysis in the following proposition:

Proposition 3 (Rényi DP guarantee for VFGNN)**.** *The output of an L-layer VFGNN model with normalization constant C satisfies* $\left(\alpha, \dfrac{4\alpha C^2\left(\sum_{l=0}^{L-1} D^l\right)}{\theta^2}\right)$*-Rényi differential privacy for any $\alpha > 1$.*

C.6 A complete report of performance comparisons

In this section, we present a complete report of empirical performance containing both the concatenation decoder (denote via using the "-C" postfix) and the GRU decoder (denote via using the "-G" postfix) listed in table 3. We summarize our experimental findings as follows:

Table 3: Experimental results over three benchmark datasets using both non-private and private approaches, reported with format $\texttt{mean}_{\pm\texttt{std}}$, with `mean` and `std` (abbreviation for standard deviation) computed under 10 trials for each setting.

Non-private approaches												
Model	ogbn-products				Reddit				Finance			
MLP	$61.06_{\pm 0.08}$				$71.07_{\pm 0.25}$				$71.30_{\pm 0.17}$			
GIN-C	$76.84_{\pm 0.94}$				$94.85_{\pm 0.15}$				$79.76_{\pm 0.59}$			
GIN-G	$76.10_{\pm 0.67}$				$94.38_{\pm 0.16}$				$79.78_{\pm 0.52}$			
GCN-C	$78.26_{\pm 0.36}$				$94.56_{\pm 0.10}$				$79.70_{\pm 0.60}$			
GCN-G	$75.80_{\pm 0.65}$				$94.37_{\pm 0.13}$				$80.13_{\pm 0.41}$			
Private approaches												
Model	ogbn-products				Reddit				Finance			
	$\epsilon = 4$	$\epsilon = 8$	$\epsilon = 16$	$\epsilon = 32$	$\epsilon = 4$	$\epsilon = 8$	$\epsilon = 16$	$\epsilon = 32$	$\epsilon = 4$	$\epsilon = 8$	$\epsilon = 16$	$\epsilon = 32$
VFGNN	$26.94_{\pm 0.00}$	$40.96_{\pm 1.74}$	$56.27_{\pm 1.35}$	$69.57_{\pm 0.26}$	$19.40_{\pm 2.56}$	$31.20_{\pm 1.75}$	$43.67_{\pm 1.18}$	$86.72_{\pm 0.33}$	$53.87_{\pm 6.86}$	$56.17_{\pm 5.31}$	$54.62_{\pm 4.09}$	$52.12_{\pm 4.78}$
GAP	$59.20_{\pm 2.13}$	$65.02_{\pm 0.82}$	$66.20_{\pm 2.34}$	$67.14_{\pm 0.34}$	$76.84_{\pm 1.71}$	$86.59_{\pm 0.48}$	$88.57_{\pm 1.69}$	$89.65_{\pm 0.30}$	$52.02_{\pm 9.39}$	$48.66_{\pm 7.14}$	$59.04_{\pm 7.95}$	$67.54_{\pm 4.41}$
VESPER (GIN-C)	$71.27_{\pm 0.70}$	$74.46_{\pm 0.45}$	$75.18_{\pm 0.52}$	$75.82_{\pm 0.92}$	$82.34_{\pm 1.57}$	$91.52_{\pm 0.22}$	$93.34_{\pm 0.20}$	$93.77_{\pm 0.23}$	$67.03_{\pm 1.88}$	$67.87_{\pm 1.56}$	$68.08_{\pm 1.12}$	$68.64_{\pm 0.91}$
VESPER (GIN-G)	$69.55_{\pm 0.61}$	$73.45_{\pm 0.30}$	$75.36_{\pm 0.49}$	$75.60_{\pm 0.51}$	$75.48_{\pm 2.61}$	$89.60_{\pm 0.44}$	$92.01_{\pm 0.48}$	$92.95_{\pm 0.41}$	$58.90_{\pm 0.67}$	$64.43_{\pm 0.23}$	$66.05_{\pm 0.17}$	$67.10_{\pm 2.15}$
VESPER (GCN-C)	$66.73_{\pm 0.40}$	$67.53_{\pm 0.79}$	$70.13_{\pm 0.55}$	$70.47_{\pm 0.41}$	$89.85_{\pm 0.27}$	$91.28_{\pm 0.17}$	$92.11_{\pm 0.20}$	$92.57_{\pm 0.14}$	$70.44_{\pm 0.89}$	$71.39_{\pm 0.91}$	$72.56_{\pm 0.57}$	$72.72_{\pm 1.11}$
VESPER (GCN-G)	$67.60_{\pm 0.40}$	$68.68_{\pm 0.67}$	$70.02_{\pm 0.48}$	$70.62_{\pm 0.33}$	$89.54_{\pm 0.11}$	$91.02_{\pm 0.31}$	$91.47_{\pm 0.20}$	$92.41_{\pm 0.19}$	$73.57_{\pm 0.73}$	$73.96_{\pm 0.81}$	$73.90_{\pm 0.52}$	$74.62_{\pm 0.35}$

Privacy-utility trade-off Overall, the VESPER framework exhibits competitive privacy-utility trade-off under both GIN and GCN aggregators. Specifically, VESPER using GCN aggregator achieves better performance than the non-private MLP baseline across all three datasets under a decent privacy protection level with $\epsilon = 4$.. The results suggest that the model has the capability of privately learning the structural information brought by the underlying graph. Moreover, if we are allowed to relax the privacy requirement via adopting bigger privacy budgets (i.e., $\epsilon \in \{16, 32\}$), VESPER might obtain high-performance models that closely match the performance of non-private versions. This phenomenon is particularly evident when using the GIN aggregator under ogbn-products and Reddit datasets, where the utility loss is cut to be around 1 percent or fewer in the low-privacy regime.

Sparse v.s. dense graphs On one hand, VESPER using the GIN aggregator performs better than the GCN counterpart in terms of privacy-utility trade-off in moderate to high-privacy regime when the underlying graph is dense, i.e., on ogbn-products and Reddit datasets, which is likely due to the fact that GIN aggregates from the full neighborhood (with high SNR when the underlying graph is dense as discussed in section 2.2), while GCN requires truncating a significant fraction of neighborhood to control the noise level. On the other hand, for sparse graphs like the Finance dataset, the performance of the GIN aggregator deteriorates significantly (i.e., failing to match the non-private MLP baseline) due to the lower SNR in the underlying graph. In contrast, using a GCN aggregator equipped with truncated message passing allows finer noise level control, leading to much better results on sparse graphs. Meanwhile, the reduced noise level when applying GCN with truncated message passing demonstrates a significant advantage when the privacy requirements are more stringent. In particular, the performance of the GCN aggregator surpasses GIN in the high-privacy regime across all three datasets. On the Reddit dataset, GCN with truncated message passing achieves much better performance than MLP with $\epsilon = 2$, with GIN totally losing its performance at the same privacy level.

Comparison against baselines VESPER demonstrates a better privacy-utility trade-off compared to other private GNN baselines. The advantage over GAP is attributed to the end-to-end nature of the VESPER framework and better handling of message-passing mechanisms. The advantage over VFGNN is attributed to the tighter sensitivity control provided by the layer-wise perturbation strategy, as shown in the relative advantage of GAP to VFGNN.

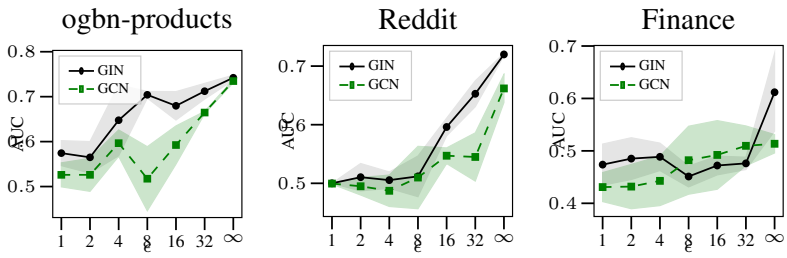

Figure 4: AUC (mean ± std over 10 trials) of membership inference attacks.

Comparison of decoders The result shows that concatenation decoder performs better over ogbn-products and Reddit datasets, while the GRU decoder performs better over Alipay.

C.7 Protection against membership inference attacks

We conduct membership inference attacks (MIAs) to empirically assess the resilience of our model against practical privacy risks. Note that although our method provides edge-privacy protection, we adopted the node MIA [37] in this experiment due to two considerations. First, no generic and appropriate edge MIA is relevant to the GNN application in this paper. Therefore, the node MIA is a more realistic threat in our scenarios. Second, the node MIA can be considered as a strengthened variant of edge MIA where the adversary obtains extra node information. Therefore, the model with certain node-membership privacy will guarantee stronger edge-membership privacy.

Attack settings Following [37, 39], we adopted the TFTS (train on subgraph, test on the full graph) setting of node MIA. Namely, the GNN model is trained on a subgraph, and the attack is reduced to a binary classification problem that distinguishes between nodes inside and outside the training subgraph. We consider an attacker with the following knowledge:

- API access to the trained model, which returns a posterior distribution of node classes.
- A shadow dataset consists of 1000 nodes per class sampled randomly from the full graph.
- Architecture, hyperparameters of the target model.

For ogbn-products and Reddit datasets, we followed the attack procedures in [37]. We first train a shadow model with the same architecture and hyperparameters as the target model using the shadow dataset. Then, we construct the attack training dataset by querying the shadow model. Finally, we train a 3-layer MLP as the attack model. For Finance dataset, since there are only two node classes, we adopted the entropy-based MIA as suggested in [40] instead of shadow model training. Specifically, we computed the Shannon entropy of the node class distribution output by the target model. The nodes with smaller entropy (larger classification confidence) tend to be in the training subgraph.

We respectively set the privacy budget $\epsilon \in \{1, 2, 4, 8, 16, 32, \infty\}$, where $\epsilon = \infty$ indicated no privacy protection is adopted. We use ROC-AUC (AUC) to evaluate the attack performance.

Results We report the attack performances in Figure 4. From the results, we observe that when privacy protection is disabled ($\epsilon = \infty$), the attacks show non-negligible effectiveness, especially on obgn-products and Reddit datasets. Generally, with the privacy budget getting smaller (privacy getting stronger), the attack performances sharply decline. With an appropriate privacy budget, the attacks on all three datasets are successfully defended with AUC reduced to around 0.5 (random guess baseline). In conclusion, the above observations demonstrate that our method effectively mitigates the risks of privacy attacks with reasonable privacy budgets.

C.8 A complete report of ablation study

In this section, we investigate the effects of several critical hyperparameters in the VESPER framework via assessing their corresponding privacy-utility trade-off. All the results reported in this section will be based on a VESPER instantiation with GCN aggregator and concatenation decoder.

On the effect of maximum degree D In GNN training with neighborhood sampling, a larger D might retain more structural information of the underlying graphs, at the same time weakening the

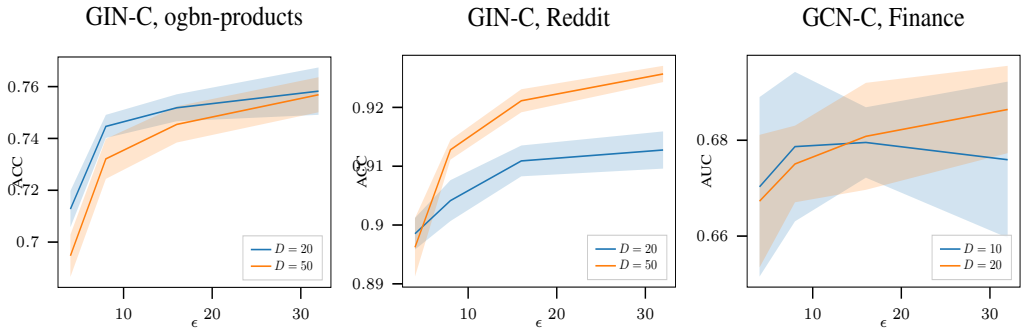

Figure 5: Performance (mean $\pm$ std over 10 trials) of VESPER under varying max degree D, using GIN aggregator and CONCAT decoder.

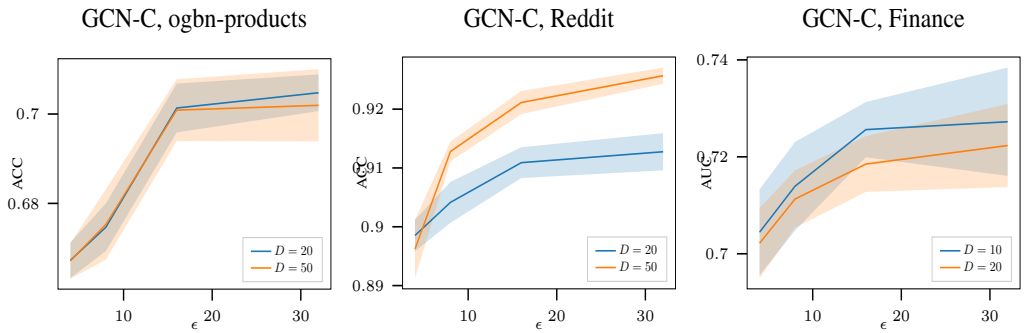

Figure 6: Performance (mean $\pm$ std over 10 trials) of VESPER under varying max degree D, using GCN aggregator and CONCAT decoder.

privacy amplification effect, resulting in a higher noise level. We evaluate the effect of D under the range $\{20, 50\}$ for ogbn-products and Reddit dataset and $\{10, 20\}$ for the Finance dataset. We plot privacy-utility trade-off curves in figure 5, 6, 8, 7. The results imply the trade-off that larger D may not always be beneficial in private GRL, especially for sparse graphs, where efficient control of noise level becomes more important than retaining structural information.

On the effect of minimum degree D_{min} for PMP-GCN As discussed in section 2.4, the D_{min} parameter trades off the amount of structural information involved during message passing and the noise scale during perturbation. We evaluate the effect of D_{min} under the range $\{10, 20, 40\}$ for ogbn-products and Reddit datasets and $\{3, 5\}$ for the Finance dataset and plot the resulting curves

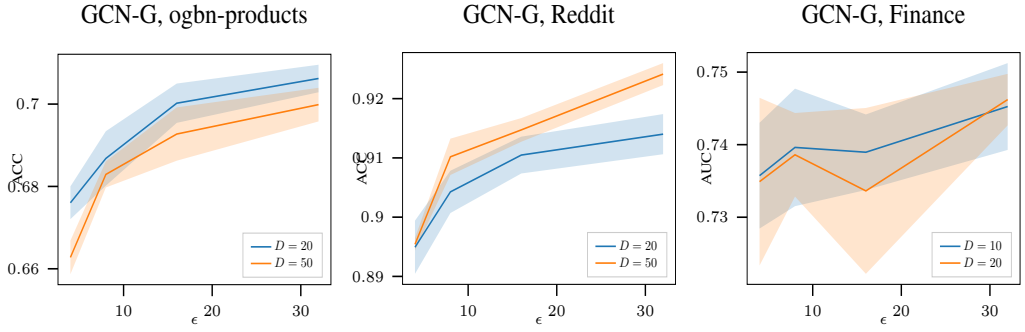

Figure 7: Performance (mean $\pm$ std over 10 trials) of VESPER under varying max degree D, using GCN aggregator and GRU decoder.

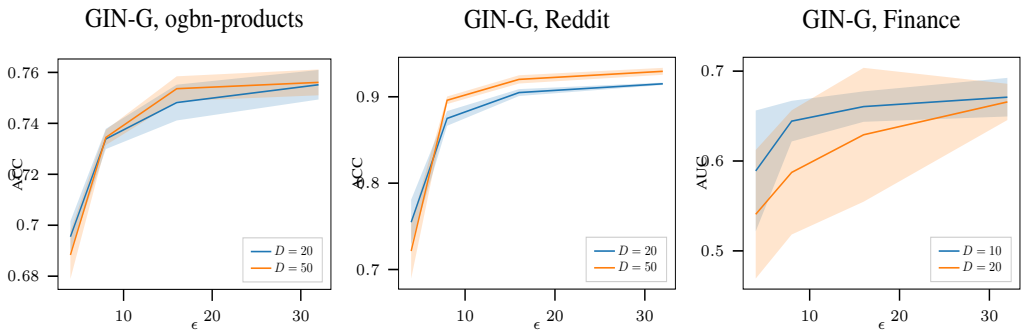

Figure 8: Performance (mean $\pm$ std over 10 trials) of VESPER under varying max degree D, using GIN aggregator and GRU decoder.

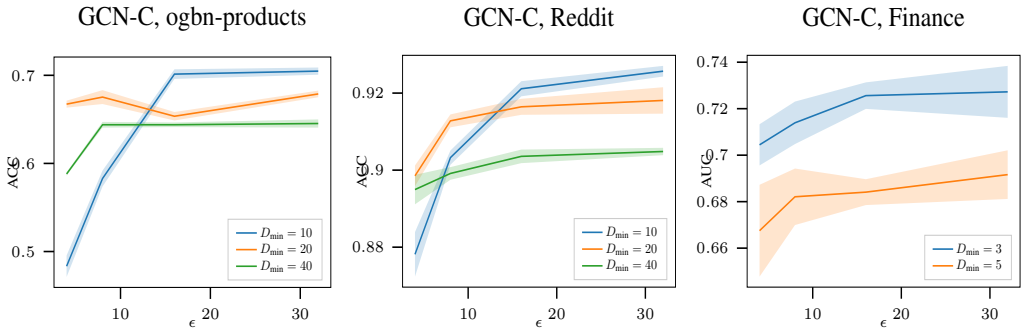

Figure 9: Performance (mean $\pm$ std over 10 trials) of VESPER under varying minmimum degree $D_{\min}$, using GCN aggregator and CONCAT decoder.

in figure 9, 10. We observe two interesting phenomena. First, adopting larger $D_{\min}$ makes the noise scale less sensitive with respect to privacy constraints, resulting in a flatter privacy-utility curve. On the two dense graph datasets, this shows the potential benefits of using a larger $D_{\min}$ when the required privacy level is more stringent. Second, an *crossing effect* is observed on ogbn-products and Reddit datasets, suggesting that a lower $D_{\min}$ is beneficial in the low-privacy regime, where the noise scale is efficiently controlled and the incorporation of more structural information becomes effective.

On the effect of batch size Finally, we assess the effect of varying batch sizes. According to composition results [1, 33, 44], consider running a fixed amount of epochs under a given sample and privacy constraint, choosing a smaller batch size in general leads to a smaller per-step noise scale.

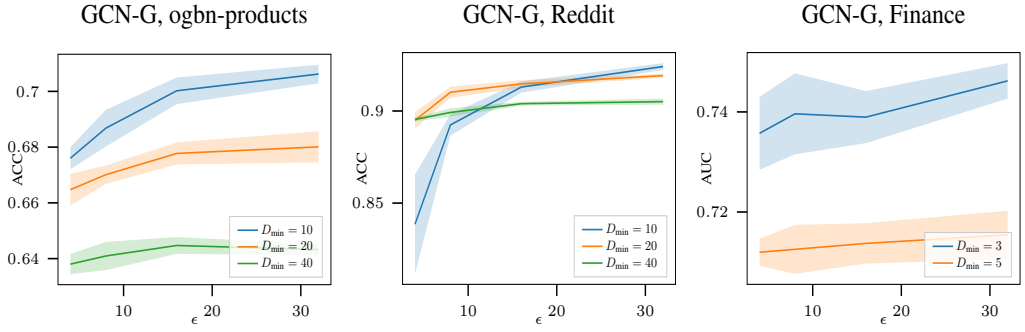

Figure 10: Performance (mean $\pm$ std over 10 trials) of VESPER under varying minmimum degree $D_{\min}$, using GCN aggregator and GRU decoder.

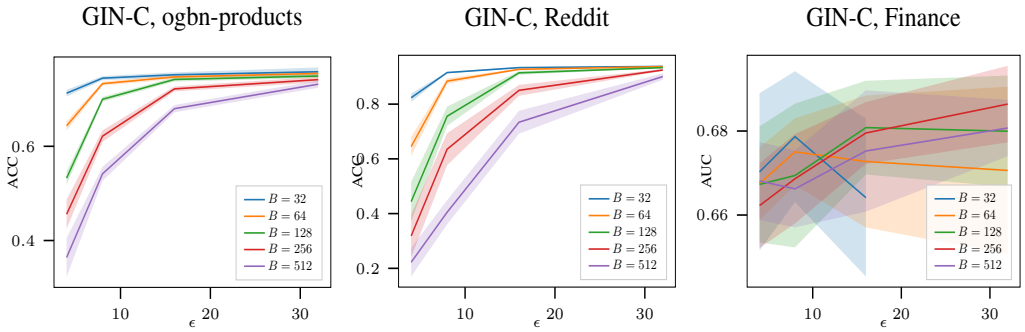

Figure 11: Performance (mean $\pm$ std over 10 trials) of VESPER under varying batch size B, using GIN aggregator and CONCAT decoder.

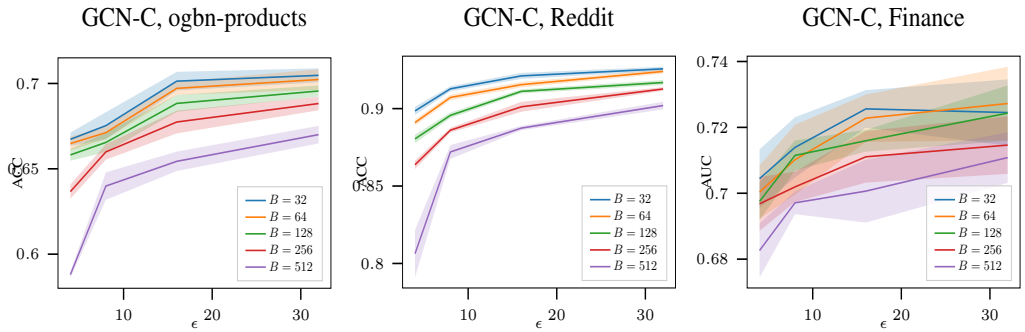

Figure 12: Performance (mean $\pm$ std over 10 trials) of VESPER under varying batch size B, using GCN aggregator and CONCAT decoder.

However, an overly small batch size may cause the stochastic gradients to be too noisy for good performance. We evaluate the effect of batch size under the range $\{32, 64, 128, 256, 512\}$ across all three datasets and plot the resulting curves in figure 11, 12, 13, 14. According to the results, we find that the reduction in noise scale caused by choosing smaller batch sizes may produces better performance on ogbn-products and Reddit dataset, while on the Finance dataset changing batch size does not produce a statistically significant difference in privacy-utility trade-offs, as indicated by the overlapping region in the figure.

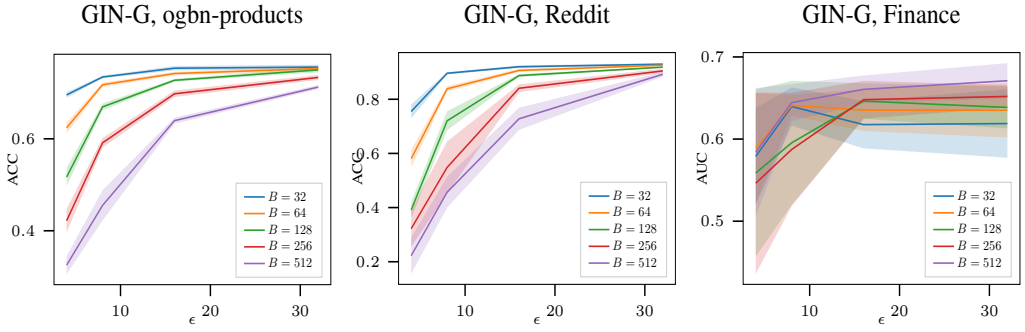

Figure 13: Performance (mean $\pm$ std over 10 trials) of VESPER under varying batch size B, using GIN aggregator and GRU decoder.

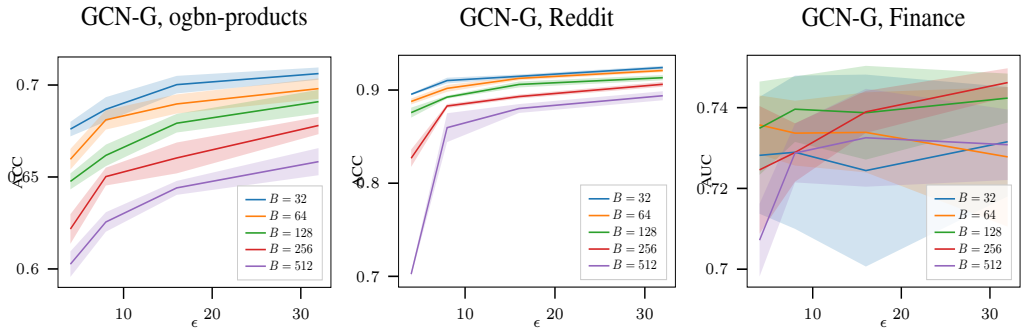

Figure 14: Performance (mean $\pm$ std over 10 trials) of VESPER under varying batch size B, using GCN aggregator and GRU decoder.

D Discussions

In this section, we discuss two extensions of the proposed framework regarding the threat model and the privacy model.

D.1 Beyond one-sided adversary

In this paper, we are mainly interested in the threat model with only one malicious party B which possesses only label data. Such a one-sided threat model might be further extended to a more complicated setup where party B owns not only label data, but also its own node features and graph structure and allows party A to be a semi-honest adversary to infer party B's label information and graph structure. We provide a straightforward solution to this extended scenario. Specifically, party B needs to protect the edge privacy of its local graph as well as the precise label data from being reconstructed by Party A. For the former privacy requirement, party B may apply the PMP framework to its local graph representation learning procedure. For protecting label data, we can simply adopt the randomized response technique for *label differential privacy* [12] during the loss computation step, and deploy three accountants with one accounts with the privacy budget corresponding to party A that is identical to the one used in this paper, and the other two accountant tracking the cumulative privacy cost incurred by PMP and randomized response mechanisms conducted by party B, which is trivial to analyze [11]. Moreover, extending the above scenario to more than two parties is also technically feasible using similar algorithmic procedures. In this paper, we did not examine such kinds of scenarios empirically since there are no publicly available graph VFL datasets that provide natural feature/graph splits between multiple parties.

D.2 On extensions to node-level DP

The node-level differential privacy (node DP) model [22] is a strictly stronger notion of privacy than the edge-level DP model regarding graph-input queries. In particular, node DP is analogously defined as in definition 1 under the approximate (ϵ, δ)-DP model with the *adjacency relation* modified in the sense that two graphs G, G' are node-level adjacent if G could be edited into G' via adding or removing one node as well as its adjacent edges. According to its original proposal, node DP targets the *protection of node memberships*, which is somewhat subtle under the VFL context since both party A and party B know the participating nodes' identities throughout the VFL process. Nevertheless, it is still possible to use additive noise perturbation to guarantee that the node embeddings are probabilistically similar with or without the participation of some specific nodes during the message-passing procedure. Formally, we establish the node DP guarantee of PMP without neighborhood sampling in the following theorem:

Theorem D.1 (RDP guarantee of PMP under node DP, non-sampling version)**.** *For graphs with bounded maximum degree D, the released output of the entire graph $\mathbf{H}_L$ in algorithm 1 without neighborhood sampling is $\left(\alpha, \frac{\alpha \sum_{l=1}^{L}(1+\sqrt{D}\mathcal{S}_l)^2}{2\theta^2}\right)$-Rényi differentially private for any $\alpha > 1$.*

Proof. From the proof of theorem 2.1, it suffices to show that the node sensitivity defined as

$$\mathcal{S}_{\mathrm{n}} = \max_{G,G'} \sqrt{\sum_{v \in V \setminus \{u^*\}} \|h_v - h'_v\|_2^2 + \|h_{u^*}\|_2^2} \tag{29}$$

is bounded from above by $1 + \sqrt{D}\mathcal{S}_l$ in the l-th layer. Note that removing a node v^* and all its adjacent edges affects two parts of node embeddings: h_{v^*} and $\{h_u, u \in N(v^*)\|$, leaving the rest embeddings untouched. Therefore we bound both using corresponding upper bounds:

$$\begin{aligned}
\mathcal{S}_{\mathrm{n}} &\leq \sqrt{\max_{G,G'} \sum_{v \in N(v^*)} \|h_v - h'_v\|_2^2 + \max_{G} \|h_{u^*}\|_2^2} \\
&\leq \sqrt{\sum_{v \in N(v^*)} \max_{G,G'} \|h_v - h'_v\|_2^2 + 1} \\
&\leq 1 + \sqrt{D}\mathcal{S}_l
\end{aligned}$$

□

Theorem D.1 implies that, under general PMP mechanisms, the node-level privacy guarantee becomes much weaker than that of edge-level by a factor of the order $O(\sqrt{D})$. Moreover, in stochastic training paradigms, the sampling amplification phenomenon is also weaker than theorem 1 [7, Theorem 1]. In our experiments, we find node DP guarantee to be overly stringent which produces meaningless results under moderate privacy budgets. Therefore we report only edge DP results in this paper. However, as illustrated in section we empirically investigated the protection of applying edge-level private mechanisms against node-level membership inference adversaries and the results are confirmatory, this serves as empirical evidence that edge-level privacy might be adequate for reasonable privacy protection rather than sticking to node-level DP definitions.

D.3 Beyond GIN and GCN aggregators

In this section, we discuss possible extensions of the PMP framework into other aggregation schemes. Throughout the discussion, we adopt the ReLU function as the default nonlinearity and focus mainly on the aggregation step.

D.3.1 On max-pooling aggregation of SAGE [15]

Technically, the case of max-pooling does not directly fit into the message passing form in (4), due to the fact that the max-pooling operation is applied along each coordinate, or:

$$\left[\widetilde{h}_v^{(l)}\right]_i = \max\left(\left[W_1^{(l)} h_v^{(l-1)}\right]_i, \left\{\left[W_2^{(l)} h_u^{(l-1)}\right]_i\right\}_{u \in N(v)}\right), \tag{30}$$

where we use the notation $[a]_i$ to denote the i-th coordinate of some vector a. Nonetheless, we may still analyze the associating edge sensitivity directly, i.e. via carefully inspecting the geometry of the max-pooling operation in high-dimensional spaces. However, it is straightforward to check that max-pooling causes high edge sensitivity, which is no smaller than that of summation pooling in the worse case, resulting in relatively large noise scales. Meanwhile, the "signal" brought by the aggregation does not scale with neighborhood size. Therefore it is intuitively clear that using the global sensitivity framework to privatize SAGE in this max-pooling form would lead to poor privacy-utility trade-offs. It is worth mentioning that chances are that adopting more elegant techniques like the smooth sensitivity paradigm [35] may allow meaningful privatization of the max-pooling aggregator, which is beyond the scope of the current paper and delegated to futrue explorations.

D.3.2 On attentive pooling of GAT [42]

Next we consider the renowned GAT model [42] with updating rule:

$$\widetilde{h}_v \leftarrow \sum_{u \in N(v) \cup \{v\}} \beta_{uv} W h_u, \tag{31}$$

with the attention coefficients defined as

$$\beta_{uv} = \frac{e^{k_\phi(u,v)}}{\sum_{u' \in N(v) \cup \{v\}} e^{k_\phi(u',v)}}. \tag{32}$$

under the *attention kernel* k_ϕ. In the original implementation of GAT, the authors used additive attention kernels. Later extensions use alternatives such as the multiplicative kernel in graph transformer architectures[51]. The protocol of attentive aggregation is also a special case of (4), which may be understood as an interpolation between mean-pooling (where all the attention coefficients are equal) and max-pooling (where one of the attention kernels being extremely large in value that dominates the rest). As a consequence, the noise scale required under the edge sensitivity calculation paradigm (i.e., theorem **??**) will be between that obtained by mean-pooling and max-pooling, depending on the *range* of the attention kernel. In particular, if the attention kernel has an unbounded range, i.e. the entire real line. Then the resulting edge sensitivty is almost the same as the one obtained by max-pooling and is thus impractical. Hence, to reduce the noise scale required for privatization, we need to use *bounded* attention via effectively controlling the output range of the attention kernel, i.e., via applying bounded range nonlinearities like Tanh. The analysis could be done in the same manner as that of GCN with extra hyperparameters controlling the upper and lower bounds of attention coefficients. We leave related developments to future works.

E Some further remarks

E.1 Practical considerations in implementing VESPER

According to proposition 1 and 2, precise tracking of privacy budgets under PMP requires computing the operator norm of each layer's weight matrix $\{\|W^{(l)}\|_{\text{op}}\}_{1 \le l \le L}$ which is computationally demanding. In practice, we instead add a spectral normalization operation [34] to each layer's weight matrix so that we may approximately control all the operator norms throughout the training process to be around 1. [2]

E.2 Complexity analysis of VESPER

The computational complexity of vesper is of the same order as that in a standard GRL pipeline with neighborhood sampling. The communication complexity of VESPER is dominated by the data volume of (forward) embedding and (backward) gradients that get transmitted during each VFL step, which are both of the order $O(BLdK)$, with B being the batch size and K being the number of bits required to represent a scalar number. Note that the communication complexity may be further optimized via quantization techniques [41] or asynchronous optimization tricks [29]. We leave such explorations to future research.

F Omitted algorithm descriptions

In this section we present detailed descriptions of two algorithmic procedures, the first one is the truncated message passing algorithm for PMP-GCN, which for simplicity we present in a non-sampling fashion in algorithm 2. The second one is a fully-detailed description of the training procedure of the VESPER framework, illustrated pictorially in figure 1. We use different colors to differentiate (local) computations that are performed by different parties. Additionally, we abbreviate the forward computation of three algorithmic components of VESPER by Encode, PRE and Decode respectively.

[2] Technically, most of the current spectral normalization algorithms do not offer strict control over spectral norms but are instead carried out using approximations like power iteration. We observe in our experiments that the approximation error brought by inexact normalization is pretty benign, i.e., the total privacy budget accounted using exact normalization and using a single power iteration differs in their absolute value by less than 0.1.

Algorithm 2 PMP-GCN with truncated message passing

Require: Graph $G = (V, E)$, input encodings $\{h_v^{(0)}\}_{v \in V}$, number of message passing rounds L, minimum degree D_{min}, GNN parameter $\mathbf{W}$, MLP parameter $\{W_{tr}^{(l)}, b_{tr}^{(l)}\}_{1 \le l \le L}$

Normalize each $h_v^{(0)}$ into unit ℓ_2 norm.
for $l \in \{1, \ldots, L\}$ **do**
    for $v \in V$ **do**
        Compute the linear update

$$\widetilde{h}_v^{(l)} = \frac{W^{(l)} h_v^{(l-1)}}{d_v + 1} + \sum_{u \in N(v)} \frac{W^{(l)} h_u^{(l-1)}}{\sqrt{d_v + 1}\sqrt{d_u + 1}}. \tag{33}$$

        if $d_v \ge D_{\text{min}}$ **then** let $\widehat{h}_v^{(l)} = \widetilde{h}_v^{(l)}$.
        else let $\widehat{h}_v^{(l)} = W_{tr}^{(l)} h_v^{(l-1)} + b_{tr}^{(l)}$.
        Add Gaussian noise and apply nonlinearity

$$h_v^{(l)} = \sigma(\widetilde{h}_v^{(l)} + z_v),\ z_v \sim N(0, \theta^2 I_d). \tag{34}$$

        Normalize $h_v^{(l)} = \frac{h_v^{(l)}}{\left\| h_v^{(l)} \right\|_2}$.
return A list of all layers' embedding matrices $\mathbf{H}_L = (H^{(1)}, \ldots, H^{(L)})$, with $H^{(l)} = \{h_v^{(l)}\}_{v \in V}, 1 \le l \le L$.

Algorithm 3 Algorithmic description of VESPER

Require: Graph $G = (V, E)$, node features X, node label Y, batch size B, number of training steps T. Architectural specifics {Encoder, PRE, Decoder}, with the parameters of Encoder and PRE grouped together with notation $\mathbf{W}_A$ and the parameters of Decoder denoted as $\mathbf{W}_B$. Number of message passing layers L, max degree D.

Initialize parameters $\mathbf{W}_A^{(0)}$, $\mathbf{W}_B^{(0)}$
for $t = 1, \ldots, T$ **do**
  Sample a random batch of root nodes $\mathcal{B}_t = \{v_1^t, \ldots, v_B^t\}$.
  `/*Computations by party A*/`
  Use neighborhood sampler as stated in algorithm 1 to obtain
  the combined subgraph $G_{\mathcal{B}_t}^{(L)} = (V_{\mathcal{B}_t}^{(L)}, E_{\mathcal{B}_t}^{(L)})$.
  Encode node features

$$h_v = \text{Encode}(X_v), v \in V_{\mathcal{B}_t}^{(L)} \tag{35}$$

  Do message passing using the selected PRE mechanism

$$\mathbf{H} = \text{PRE}(G_{\mathcal{B}_t}^{(L)}, H) \tag{36}$$

  Pick the node embeddings to transmit $\mathbf{H}_{\mathcal{B}_t} = \{\mathbf{H}_v\}_{v \in \mathcal{B}_t}$
  and send to party B
  `/*Computations by party B*/`
  Decode node embeddings and compute learning objective

$$\mathcal{L} = \frac{1}{B} \sum_{v \in \mathcal{B}_t} \ell\left(y_v, \text{Decode}(\mathbf{H}_v)\right) \tag{37}$$

  Compute gradients w.r.t. decoder $\dfrac{\partial \mathcal{L}}{\partial \mathbf{W}_B^{(t)}}$ and
  update into $\mathbf{W}_B^{(t+1)}$ using selected optimizer.
  Send individual gradients w.r.t. node embedding collections

$$\frac{\partial \mathcal{L}}{\partial \mathbf{H}_v}, v \in \mathcal{B}_t \tag{38}$$

  back to party A.
  `/*Computations by party A*/`
  Party A compute gradients w.r.t. all its local parameters
  (encoder and PRE)

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}_A^{(t)}} = \frac{1}{B} \sum_{v \in \mathcal{B}_t} \frac{\partial \mathcal{L}}{\partial \mathbf{H}_v} \frac{\partial \mathbf{H}_v}{\partial \mathbf{W}_A^{(t)}} \tag{39}$$

  and use the selected optimizer to update $\mathbf{W}_A^{(t)}$ into $\mathbf{W}_A^{(t+1)}$
return The parameters at the final iteration $\mathbf{W}_A^{(T)}$, $\mathbf{W}_B^{(T)}$