Gromov-Wasserstein Discrepancy with Local Differential Privacy for Distributed Structural Graphs

Hongwei Jin[1] , **Xun Chen**[2*]

[1]University of Illinois at Chicago
[2]Samsung Research America, inc.
hjin25@uic.edu, xun.chen@samsung.com

Abstract

Learning the similarity between structured data, especially the graphs, is one of the essential problems. Besides the approach like graph kernels, Gromov-Wasserstein (GW) distance recently draws a big attention due to its flexibility to capture both topological and feature characteristics, as well as handling the permutation invariance. However, structured data are widely distributed for different data mining and machine learning applications. With the privacy concerns, accessing the decentralized data is limited to either individual clients or different silos. To tackle these issues, we propose a privacy-preserving framework to analyze the GW discrepancy of node embedding learned locally from graph neural networks in a federated flavor, and then explicitly place local differential privacy (LDP) based on Multi-bit Encoder to protect sensitive information. Our experiments show that, with strong privacy protection guaranteed by ε-LDP algorithm, the proposed framework not only preserves privacy in graph learning, but also presents a noised structural metric under GW distance, resulting in comparable and even better performance in classification and clustering tasks. Moreover, we reason the rationale behind the LDP-based GW distance analytically and empirically.

1 Introduction

A lot of applications in machine learning are extensively using either pairwise matrices or kernels to perform the task of prediction or regression. Examples like the RBF kernel and covariance matrices are commonly used in support vector machine classification in the Euclidean space. However, applications in the non-Euclidean domains, presented in graphs, such as social network [Fan *et al.*, 2019], recommendation systems [Wu *et al.*, 2020], fraud detection [Li *et al.*, 2020] and biochemical data [Coley *et al.*, 2019] involve the study the of relations among the featured objects. To measure the graph similarities, a variety of graph kernels [Vishwanathan *et al.*, 2010] are proposed but mainly focus on the topological
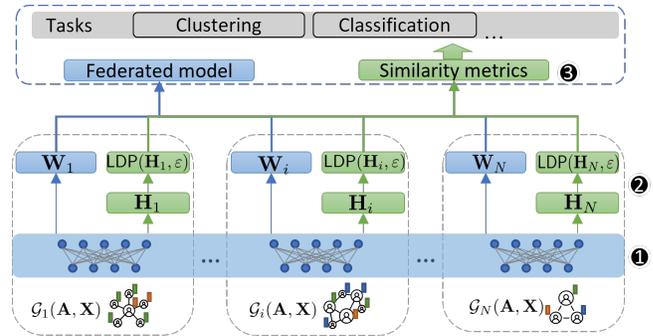

Figure 1: Framework

information. Graph neural networks (GNNs, e.g., [Hamilton *et al.*, 2017]) are one of the state-of-the-art models to learn node embedding. However, it requires additional readout layers to match the dimension for graph sizes, and it lacks the trackable way to capture the isometric issue on graphs. Recently, Gromov-Wasserstein (GW) distance [Mémoli, 2011; Peyré *et al.*, 2016] provides a geometry-based metric to measure the optimal transportation from one structural object to another. GW operates on the metric-measure spaces, i.e., a Borel probability measure on the set with its compact metric space. In other words, it operates on distances between pairs of points calculated within each domain and measures how the distance is compared to those in the other domain. Specifically, all-pair shortest path (APSP) is one of the common geodesic measures applied. Thus, it requires calculating the transportation between two domains by their intra-distance, turning the distance problem from a linear optimization to a quadratic one. And it extends both topological and features information (known as FGW [Titouan *et al.*, 2019]) so that the similarity problem cooperates with the node features nicely. One of the key advantages of applying GW distance is the capability of handling the isometric properties between structures. Thus, GW is highly related to graph matching problem, encoding structural information to compare graphs, and has long history been successfully adopted in image recognition [Peyré *et al.*, 2016], bio-domain adoption [Demetci *et al.*, 2020], and point-cloud data alignment [Mémoli and Sapiro, 2006].

All previous works have assumed that all structured data

---

are always available and accessible, which is often not true in reality. Structured data in many applications either contain sensitive and private information (such as payment transactions and social connections) or distribute across different clients commonly [Liao *et al.*, 2021]. On one hand, data owners are limited to sharing the original data due to the privacy regulations that are enforced globally, such as GDPR [Regulation, 2018] and CCPA [Bukaty, 2019]. On the other hand, in many applications, one has to deal with very large datasets that cannot be processed at once. Thus, a nested partitions of the input which are parameterized by a scale parameter is arising from the metric structure of the input data.

1.1 Problem Formulation

In this paper, we work on the problem of GW distance between distributed graphs while having privacy constraints. Two key assumptions are placed in the underlying such scenarios: 1) each graph holds its topological information, nodes with feature and labels locally; 2) server cannot access raw data globally while having to distinguish from graph to graph (anomaly detection, clustering, recommendation, etc). Accordingly, two main issues need to be addressed: first, to train the global model across different graphs while preserving the privacy of local data; second, to compare graphs even in the case of permutation invariant and size variant. In order to overcome the two issues, we take federated learning schema as our primary framework, along with a geometry-based metric named Gromov-Wasserstein discrepancy [Peyré *et al.*, 2016] on the server-side to learn similarity / dissimilarity across graphs. Gromov-Wasserstein distance [Mémoli, 2011] provides a metric to measure the optimal transportation from one structural object to another.

The proposed framework is illustrated in Figure 1. To summarize: 1) the synchronized vertical federated learning first updates local parameters $\mathbf{W}$ from the private structural data $\mathcal{G}(\mathbf{A}, \mathbf{X})$ to learn the local graph embedding $\mathbf{H}$; 2) then, instead of pulling the raw graph data from the client, the proposed framework takes the local graph embedding learned from the FL model, and local differential privacy (LDP) is applied explicitly, where we extend the multi-bit encoder from vector space to matrix scenario, protecting the sensitive information; 3) after getting the encoded graph embeddings, the framework calculates similarity metrics (GW distance) on the server and serves the downstream tasks, such as clustering, classification, and matching problems.

Overall, we highlight our core contributions as

- Beyond the conventional federated learning, we design the guaranteed ε-LDP on graph embeddings to explicitly protect the graph information. It is worth mentioning that LDP on topology is not applicable because of the vulnerability of GNN, and perturbation on either the node or edge leads to a non-smoothing function.

- Besides, we adopt the Gromov-Wasserstein discrepancy to be the learnable structural metric on the server, co-operating the LDP encoded graph embeddings from the client-side.

- Interestingly, we observe that the LDP on graph embeddings not only protects privacy but also helps to escape

saddle points in GW optimization. We interpret that this is largely due to the non-convexity of GW discrepancy.

- Having the GW discrepancy with noised input, we empirically evaluate the performance of graph classification and clustering problem.

To the best of our knowledge, this is the first work to learn the structural metric with local differential privacy under the distributed setting. And the framework is flexible to learn the node embedding and can be combined with a variety of GNNs for a broad range of different tasks, including graph classification, clustering, etc.

1.2 Notation

We take capital and lower letter in bold to represent matrix and vector, respectively. A graph $\mathcal{G}(\mathbf{A}, \mathbf{X})$ has adjacency matrix $\mathbf{A}$ as structural information and node feature $\mathbf{X}$ as feature information. Besides, $\mathcal{V}, \mathcal{E}$ are the node sets and edge sets from the graph, respectively. $\mathbf{H}$ is the retrieved node representation, each row of the matrix represents a single node on the graph. An upper script i means the i-th graph, while a lower script i means i-th row of the matrix.

2 Related Work

The ultimate goal of our proposed method is to protect the privacy of structured data while building the GW distance for similarities between graphs. Naturally, Federated Learning (FL) [McMahan *et al.*, 2017; Smith *et al.*, 2017] is an emerging collaborative learning framework that can train multiple clients locally while maintaining a global model on the server. Distinguished from the shared parameter server [Li *et al.*, 2014], one of the significant fruits of federated learning is privacy protection, where each client holds its raw data locally, only shared parameter/gradient will be exchanged with the server. Specifically, in the vertical FL setting, a model is shared among all the clients, and the client trains the model locally with its labels, while the server learns the global models by aggregating the local client models.

Recently, a set of different methods are proposed to build a GNN model with a federated setting. With the task of graph classification, [He *et al.*, 2021] design a federated learning system and benchmark for graph neural networks. The framework is built based on FedML [He *et al.*, 2020] research library, enabling to run a variety of GNN models effectively with non-IID graph dataset, which is a similar approach in the image domain. Interestingly, FL-AGCNS [Wang *et al.*, 2021] propose to take advantage of federated learning to accelerate in the task of neural architecture search of graph convolutional network. Specifically, they take federated SuperNet optimization to select random GCN architecture and take federated evolutionary optimization to perform evaluation suing GCN SuperNet.

However, all the work above trains a shared model with the task of supervised/semi-supervised learning, but do not address the issue of unlabelled graphs on clients. What's more, none of the work above discusses the graph differential privacy while learning the model. Unlike our proposed framework, a recent work of locally private graph neural network [Sajadmanesh and Gatica-Perez, 2020] build the feature and

label differential private on the clients at once while leaving the whole training process on the server-side instead.

Another approach to protecting privacy is differential privacy. More specifically, with distributed graphs, we mainly focus on the local differential privacy of each graph. Given a graph $\mathcal{G}(\mathbf{A}, \mathbf{X})$, the purpose of applying LDP is to protect the private information from topology $\mathbf{A}$ and node feature $\mathbf{X}$. A naive approach would be directly adding LDP noise to $\mathbf{A}$ and $\mathbf{X}$. However, we argue that such an approach has its limitations. For the topological LDP, assume we add random perturbation to the topology of the graph, i.e., add or delete edges on a graph. However, works like Nettack proposed in [Zügner *et al.*, 2018], found that even a small perturbation could lead to the misclassification of the nodes in the graph. Moreover, random perturbation could result in the same graph but is permutation invariant to the original graph, i.e., $\mathbf{A}' = \Pi(\mathbf{A})$, where $\mathbf{A}'$ is the perturbed graph [Niu *et al.*, 2020]. For the feature LDP, adding LDP on feature is a valid approach in the work [Sajadmanesh and Gatica-Perez, 2020], but it lacks protection for the structural information. Moreover, the study in graph diffusion Wasserstein distance [Barbe *et al.*, 2020] suggests that a noisy feature will let the fused GW depend mostly on the structure. Therefore, adding small random perturbation to either the topology or the feature is not a guaranteed approach to protect privacy in the federated learning setting.

2.1 Gromov-Wasserstein Discrepancy

Graph neural networks rely on training on structured data for various graph-related tasks. However, a common limitation is its difficulty in explaining the isomorphism in graph data. Identification of similarities between graphs is an essential problem in graph learning areas. In these areas, the graph isomorphism problem is known as the exact graph matching [Xu *et al.*, 2019b], which is not solvable in polynomial time. Two graphs are considered *isomorphic* if there is a mapping between the nodes of the graphs that preserves their adjacencies. The graph isomorphism testing is used in a wide range of applications, such as the identification of chemical compound, the generation of molecular graphs in chemical dataset [Titouan *et al.*, 2019], and the electronic design automation (EDA) with placement and routing operations [Chan *et al.*, 2000]. Graph Isomorphism Network (GIN) [Xu *et al.*, 2019c] is recently proposed to implement Weisfeiler-Lehman (WL) graph isomorphism test [Shervashidze *et al.*, 2011]. However, such kind of approach only deals with graphs of the same size, and hard to distinguish the difference between graphs with arbitrary sizes.

Recently, the optimal transport (OT) associated with their Gromov-Wasserstein (GW) discrepancy [Peyré *et al.*, 2016], which extends the Gromov-Wasserstein distance [Mémoli, 2011], has emerged as an effective transportation distance between structured data, alleviating the incomparability issue between different structures by aligning the *intra*-relational geometries. GW distance is isometric, meaning the unchanged similarity under rotation, translation, and permutation. Formally, [Peyré *et al.*, 2016] define the Gromov-Wasserstein discrepancy between two measured similarity matrices $(\mathbf{C}, \mathbf{p}) \in \mathbb{R}^{n \times n} \times \sum_n$ and $(\mathbf{D}, \mathbf{q}) \in \mathbb{R}^{m \times m} \times \sum_m$

as follows

$$\mathrm{GW}(\mathbf{C}, \mathbf{D}, \mathbf{p}, \mathbf{q}) = \min_{\mathbf{T} \in \mathcal{C}_{\mathbf{p},\mathbf{q}}} \sum_{i,j,k,l} \ell(\mathbf{C}_{i,k}, \mathbf{D}_{j,l}) \mathbf{T}_{i,j} \mathbf{T}_{k,l}, \quad (1)$$

where $\mathbf{C}$ and $\mathbf{D}$ are matrices representing either similarities or distances between nodes within the graph. $\ell(\cdot)$ is the loss function either in Euclidean norm or KL-divergence [Peyré *et al.*, 2016]. $\mathbf{p} \in \mathbb{R}_n^+ : \sum_i \mathbf{p}_i = 1$ is the simplex of histograms with n bins, and $\mathbf{T}$ is the coupling between the two spaces on which the similarity matrices are defined. Specifically,

$$\mathcal{C}_{\mathbf{p},\mathbf{q}} = \left\{ \mathbf{T} \in \mathbb{R}_+^{n \times m}, \mathbf{T}\mathbf{1} = \mathbf{p}, \mathbf{T}^\top \mathbf{1} = \mathbf{q} \right\}. \quad (2)$$

Here, we denote the domain $\mathcal{E} := \left\{ \mathbf{T} \in \mathbb{R}^{n \times m}, \mathbf{T}\mathbf{1} = \mathbf{p}, \mathbf{T}^\top \mathbf{1} = \mathbf{q} \right\}$ and $\mathcal{N} := \mathbb{R}_+^{n \times m}$. Furthermore, we can rewrite the problem in Koopmans-Beckmann form [Koopmans and Beckmann, 1957]:

$$\mathrm{GW}(\mathbf{C}, \mathbf{D}) = \frac{\|\mathbf{C}\|_F^2}{n^2} + \frac{\|\mathbf{D}\|_F^2}{m^2} - 2 \max_{\mathbf{T} \in \mathcal{C}_{\mathbf{p},\mathbf{q}}} \mathrm{tr}(\mathbf{C}\mathbf{T}\mathbf{D}\mathbf{T}^\top). \quad (3)$$

Note that the GW distance is non-convex and highly related to the quadratic assignment problem (QAP), which is NP-hard. Moreover, given the similarity matrices $\mathbf{C}$ and $\mathbf{D}$, we optimize with the quadratic trace form in (3) over the domain of $\mathcal{E} \cap \mathcal{N}$. The GW discrepancy problem can be solved iteratively by conditional gradient method [Peyré *et al.*, 2016] and the proximal point algorithm [Xu *et al.*, 2019a].

In the next section, we will address the details of our proposed ε-LDP on graph embeddings and cooperate them under GW metrics.

3 Framework

To highlight our framework, it encodes node representation from the client while building the structural metric on the server-side. To build the node representation, we train the model in the task of node classification, by using the node labels locally. A conventional federated learning framework is well suited in this setting. To the simplification, we take FedAvg [McMahan *et al.*, 2017] as the aggregation algorithm without further notice. Besides, we apply a multi-bit encoder to the node representation as LDP before transmit the graph information to the server. With the protected node representation, server calculates the geometric metric GW accordingly.

3.1 ε-LDP with Multi-bit Encoder on Graph Embedding

Inspired by the locally differential privacy on features, we adopt the multi-bit encoder from [Sajadmanesh and Gatica-Perez, 2020], while their approach send the encoded feature and label directly to server and train a single model on the server-side. Differentiated from feature encoding, we take the encoder on the node representation, leveraging the privacy on both topology and feature learned from GNN. Algorithm 1 describes the LDP on node embedding in more details. We extend the one-bit encoder on a vector space to the multi-bit encoder in 2-D dimension while preserving the guaranteed privacy. Intuitively, the encoder first uniformly sample the m out of the dimension of $n \times h$ without replacement. Then for each row-wise dimension, we take the Bernoulli distribution to decide the mapped encoding after the mask. For

Algorithm 1 $\text{LDP}_G(\cdot, \varepsilon)$: Multi-Bit Encoder

---

Input: Node embedding $\mathbf{H} \in [\alpha, \beta]^{n \times h}$, ε
Output: Encoded embedding $\mathbf{H}^*$

1: Let $\mathcal{S}$ be a set of m values sampled uniformly from $[n] \times [h]$
2: **for** i in n **do**
3:   **for** j in h **do**
4:     $s_{ij} = 1$ if $\mathcal{S}_{ij} = 1$ else $s_{ij} = 0$
5:     $t_{ij} \sim \text{Bernoulli} \left(\frac{1}{e^{\varepsilon/(mn)}+1} + \frac{\mathbf{H}_{ij}-\alpha}{\beta-\alpha} \cdot \frac{e^{\varepsilon/(mn)}-1}{e^{\varepsilon/(mn)}+1} \right)$
6:     $\mathbf{H}^*_{ij} = s_{ij}(2t_{ij}-1)$
7:   **end for**
8: **end for**

---

each hidden feature dimension, the probability encoding is determined by the number of nodes in the graph (n) and the sampled mask (m). This is just one-pass encoding with the complexity of $\mathcal{O}(mn)$.

Theorem 1 *The multi-bit encoder presented in Algorithm 1 satisfies ε-local differential privacy for each graph representation.*

Theorem 1 provides the theoretically guaranteed privacy protection on hidden representation.

3.2 Constructing GW Distance

Besides the work of building the global model, we also explicitly enforce the server to construct a pairwise similarity matrix $\mathbf{C} \in \mathbb{R}^{n \times n}$. More specifically, $\mathbf{C}_{\mathbf{H}^{*(i)}} \in \mathbb{R}^{n^{(i)} \times n^{(i)}}$ represents intra-distance of the encoded node embedding from i-th client, namely $\mathbf{H}^{*(i)} \in \mathbb{R}^{n^{(i)} \times h}$, where $n^{(i)}$ is the number of node in i-th graph and h is the hidden dimension. Therefore, we can explicitly formulate the distance matrix between node u and v in $\mathbf{C}$ as

$$\mathbf{C}_{\mathbf{H}^*}(u, v) = d(\mathbf{H}^*_u, \mathbf{H}^*_v), \ \forall u, v \in \mathcal{V}, \tag{4}$$

where $d(\cdot, \cdot)$ is the distance function applied. Due to its Euclidean space over the reals of $\mathbf{H}^*$, we can simply take the Euclidean distance between node embeddings to build the intra-distance. Of course, it can be easily extended to other distance metrics, such as Cosine similarity and correlation. Note that, after taking the distance introduced from hidden representation, the $\mathbf{C}_{\mathbf{H}}$ is a metric space induced from the original graph (without ε-LDP encoding). Even though the unbalanced probability measure over nodes are also applicable [Titouan *et al.*, 2019], we will take $\mathbf{p}$ as a uniform distribution among the nodes, i.e., $\mathbf{p}_u = \frac{1}{|\mathcal{V}|}, \forall u \in \mathcal{V}$. Also, as $\mathbf{H}$ is the graph representation after federated learning, the embedding itself encodes both structural ($\mathbf{A}$) and feature ($\mathbf{X}$) information, which is different from the conventional calculating $\mathbf{C}$ where only geodesic distance is applied commonly.

Additionally, the framework requires the optimization of Gromov-Wasserstein discrepancy on the server, and the ε-LDP encoded graph embedding can be retrieved only once after a stable global model from FL learning is obtained. However, considering the unexpected drop-off of clients, the graph

Model	Metric Source	PROTEINS	MUTAG	IMDB-B
SPK	$\mathbf{A}$	n/a	82.95 ± 8.19	55.80 ± 2.93
GCN	$\mathbf{AX}$	69.03 ± 3.59	80.05 ± 3.21	n/a
SVM	$\text{GW}_{\mathbf{A}}/\text{FGW}_{\mathbf{AX}}$	74.55 ± 2.74	83.26 ± 10.3	63.8 ± 3.49
SVM	$\text{GW}_{\mathbf{H}(GCN)}$	72.31 ± 4.61	80.3 ± 9.75	53.69 ± 4.31
SVM	$\text{GW}_{\mathbf{H}^*(GCN)}$	71.19 ± 3.92	79.57 ± 6.73	54.01 ± 4.90
SVM	$\text{GW}_{\mathbf{H}(GIN)}$	75.53 ± 2.01	93.08 ± 3.35	74.4 ± 1.11
SVM	$\text{GW}_{\mathbf{H}^*(GIN)}$	76.89 ± 3.90	91.17 ± 4.21	73.6 ± 3.23

Table 1: Accuracy on graph classification

embedding can be retrieved periodically if necessary. Therefore, compared with the vanilla FL setting, sending the encoded graph embedding $\mathbf{H}^*$ will not introduce overhead in communication.

4 Experiments

Dataset. We will perform two tasks, one with graph classification under federated learning via GW, where we take the benchmark datasets from TUDataset [Morris *et al.*, 2020]. Another task is to take the graph clustering with subgraph generated from the single big graph, where we take the citation benchmark dataset from Planetoid [Yang *et al.*, 2016].

Graph classification. Regarding the task of graph classification under a centralized setting, we adopt an SVM using the indefinite kernel matrix $e^{-\gamma \text{FGW}}$ as a distance between graphs [Titouan *et al.*, 2019], where the training, validation, and testing sets are split in the ratio of $7 : 2 : 1$. For all methods using SVM, we cross validate the parameter $C \in \{10^{-7}, 10^{-6}, \cdots, 10^7\}$ and $\gamma \in 2^{-10}, 2^{-9}, \cdots, 2^{10}$. For the decentralized setting, we split the graphs into 10 clients from the Dirichlet distribution [Wang *et al.*, 2020], and each client has 20% and 10% for validation and testing, respectively. We report the classification accuracy in Table 1. Both PROTEINS and MUTAG has node features, where we adopt the FGW as the base metric. While IMDB-B has no features on nodes, we only take the GW (topology only) as the base metric. SPK [Borgwardt and Kriegel, 2005] is the shortest path kernel method on raw data. GCN indicates a two-layer graph convolutional layer with hidden dimension of 16, followed by an average pooling method. GIN represent a five graph isomorphic layers with hidden dimension of 64, followed by a two linear perception layers.

Comparing the results from raw data (metric based on $\mathbf{A}, \mathbf{X}$), we can see the kernel based SVM on GW discrepancy performs better than the base GCN and SPK model. For the FL setting, we defined the shared model with GCN and GIN. $\mathbf{H}$ is the graph embedding without privacy protection, and $\mathbf{H}^*$ is ε-LDP graph embedding. Without the further notice, we take the optimal $m^* = \max(1, \min(d, \lfloor \frac{\varepsilon}{2.18} \rfloor))$ from [Sajadmanesh and Gatica-Perez, 2020], where d is the number of classes from nodes. We extract the node embedding after the softmax layer, and leave $\alpha = 0, \beta = 1$. The default value is $\varepsilon = \frac{1}{|\mathcal{V}|}$ which varies from each graph. Intuitively, this basically means the cumulative privacy budget will be normalized on the number of nodes n.

To see the effectiveness of FL, GCN model under FL perform a bit worse than SVM on raw data. However, FL under GIN (a strong model) improves almost 10% when compared

with vanilla SVM with raw data. What's more, to check how LDP works (using the $\mathbf{H}^*$ graph embedding) in the classification tasks, we can find that the ε-LDP didn't hurt the overall performance.

Graph clustering. Our framework fits unsupervised learning with federated learning. And it is quite common to see that labels for each node are trivial, but not for the subgraphs (communities). Regarding the data with a single graph, we split the big graph into small subgraphs by taking multi-hop neighbors for simplification. Moreover, for such kind of dataset, we might not have the ground-truth label of the split subgraph. Therefore, we take the label of the central node as its pseudo-label. Note that we will see the overlap of nodes in different clients, which is a typical scenario in various applications, such Internet-of-Things (IoT) devices, community connections, and recommendation systems. A shared 2-layer GCN model with node classification task is deployed in the federated setting, where each client builds the node embedding. We retrieve the node embedding after a fixed number of 200 rounds of updating the shared model.

Figure 2 shows the t-SNE plot of KMeans clustering of 500 1-hop subgraphs from citeseer dataset with different ε. KMeans under GW distance is different from Euclidean distance, the centroid of each cluster is represented by a metric matrix. To simplify, we fix the dimension of the centroid by the average number of nodes within the cluster. Note that $\varepsilon = 0$ means no LDP introduced to the graph embedding and we assume the number of clusters is equal to the number of labels from nodes. Intuitively, data in the citation network has the label for each paper, and the induced subgraph is the citations referred for the central node. To show the impact of GW distance in clustering, we compare the first and second plots in Figure 2. It shows that even without the LDP encoder, the KMeans clustering with GW distance performs better than the naive labeling from central node. The property of GW distance makes it a better similarity metrics, especially with a pre-learned graph embeddings. To see the impact of LDP in clustering, compared with the raw graph embedding, the encoded $\mathbf{H}^*$ with small ε won't hurt too much in the clusters. However, a larger privacy strength ($\varepsilon = 1$) introduces more deviates of unlabelled subgraphs. This is consistent with the observation in classification problem as well. Additional results with 2-hop subgraphs and other datasets are provided in Appendix. In the next subsection, we will have a close look at privacy strength ε in the GW distance calculation.

4.1 Privacy Analysis for LDP on Embedding

Knowing that the ε is a key factor in privacy protection and downstream tasks, we invest the impact of privacy budget ε in LDP under classification. Following the Figure 3, in order to eliminate the influence of hyperparameter in GNN models, we report the accuracy using KNN based on $\mathrm{GW}_{\mathbf{H}^*}$ by cross-validating on ε on MUTAG dataset. As we can see on both settings of $k = 5, 10$, the model is relative stable with just a small perturbation, i.e., $\varepsilon < 1$, and with the increased privacy protection strength, a larger ε will hurt the overall performance. More interestingly, introducing the ε-LDP not only helps to protect the privacy, but also potentially helps the performance in downstream tasks with a proper ε.

Furthermore, we look into details of changes in similarity metric GW distance with respect to the ε. For the task of clustering, we extract five subgraphs (check Appendix for the subgraph structures) from Citeseer dataset, and shows the results of GW discrepancy when LDP is applied with different ε in Figure 4. We consider two privacy budgets $\varepsilon = 0.1, 1$ to be compared. GW_H represents the distance without LDP, i.e., $\varepsilon = 0$, with the baseline statistics of GW discrepancy to be 2.38 ± 2.22. GW_{H^*} represents the distance under encoded LDP with three different privacy budget $\varepsilon = \frac{1}{|\mathcal{V}|}, 0.1, 1$, and $2.42 \pm 2.33, 2.43 \pm 2.35, 3.25 \pm 3.09$ are the distances statistics, respectively. It can be seen from Figure 4 that, with a proper privacy budget, we can see little difference on pairwise distance based on GW. In other words, from the perspective of applications, applying LDP on node embedding has little impact on the clustering / classification tasks. It is also true from the algorithmic view that GW distance is quantitatively stable [Mémoli, 2011] in the sense of the structure invariant. Moreover, in the distributed training, graphs are completely in non-i.i.d. distribution, suggesting that the ε can vary from client to client. Therefore, we have our default ε set to be $1/|\mathcal{V}|$, which also matches our design of algorithm 1, sampling based on number of nodes.

4.2 Rationale behind GW with ε-LDP Encoder

We step further to reason why the LDP on $\mathbf{H}$ won't sacrifice the GW distance. Due to the non-convexity of GW discrepancy, which involves a quadratic problem, the absolute error in distance from different pairs cannot have a guaranteed boundary to tell the closeness between them. Despite the challenge of finding the exact the global minima, an effective way is to escape saddle points in the non-convex manifold, which prevents the optimal solution to be stuck at the local minima.

Among many of the works targeting the escape saddle points, perturb gradient descent (PGD) [Jin *et al.*, 2017] is one of practical simple method to solve the problem. The meta-algorithm is described as

$$\mathbf{x}_t \leftarrow \mathbf{x}_t + \boldsymbol{\xi}_t, \qquad \mathbf{x}_{t+1} \leftarrow \mathbf{x}_t - \eta \nabla f(\mathbf{x}_t) \qquad (5)$$

where $\boldsymbol{\xi}_t \sim \mathcal{B}(\varepsilon)$ is a small perturbation on the input space within a small ball. Noting that, the $\mathbf{H}^* = \mathbf{H} + \mathcal{O}(\varepsilon)$ and $\mathbf{C}_{u,v} = d(\mathbf{H}_u^*, \mathbf{H}_v^*) = d(\mathbf{H}_u, \mathbf{H}_v) + \mathcal{O}(\varepsilon)$, when d is a Euclidean distance function. Interestingly, we can take the LDP as small perturbation on the node embedding, which is ultimately the noisy input to the GW optimization problem.

Since it is very challenging to characterize the impact of noise in optimization, we empirically check the average output of GW with the increasing ε. Figure 5 indicates the average GW discrepancy calculated by increasing the privacy budget ε on the dataset Citeseer. We fix the initial point and other hyperparameters (threshold, max iter and etc.) in optimization, the only differences are ε. The solid blue curve indicates the average GW discrepancy (optimal function value) and the dashed orange line indicate the baseline without any perturbations. Although the increase of the privacy budget ε eventually leads to a higher mean and variance of GW discrepancy which is expected, a slight perturbation from LDP (when ε is small) helps to get a lower GW discrepancy
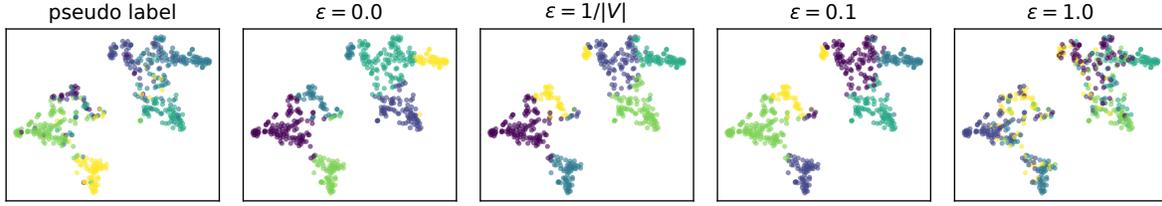
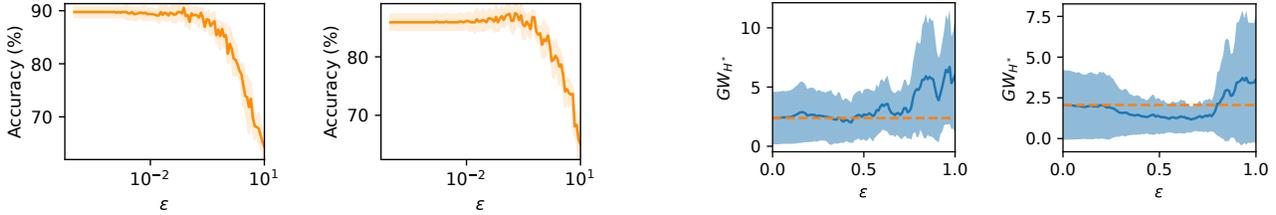Figure 2: Clustering on subgraphs based on GW. Pseudo label comes from the central node.

Figure 3: Cross validation on ε with KNN model (left: $k = 5$, right: $k = 10$).

Figure 5: Averaged GW discrepancy (optimal function value) with the increase of ε. Left: graphs induced from 1-hop neighbors. Right: graphs induced from 2-hop neighbors.

further downstream tasks. With the guaranteed ε-LDP encoding on graph embedding, our experiments show that the privacy budget not only helps in the distributed privacy protection learning, but also won't hurt the performance of the downstream tasks due to the nonconvexity of GW distance. Considering the flexibility of the framework, an interesting extension is to see the performance on large heterogeneous graphs, where the subgraphs represent small communities in social networks, fraud actions in transactions networks, and browsing logs in recommendation systems.
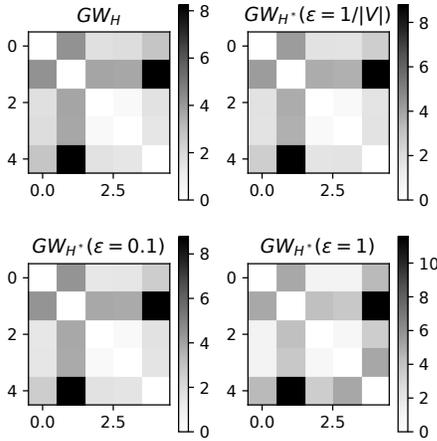
Acknowledgements

Figure 4: A demonstration of LDP on node embedding for GW metric.

(smaller optimal function value from blue curve), especially when the graph is more complicated (2-hop neighbors).

The empirical result matches with our understanding of non-convex optimization with perturbed method. However, a rigorous analysis of relation between LDP and non-convex still remains unclear, especially in the connection between privacy budget and the constrained perturbation domain. We leave this problem for our future work.

5 Conclusion

In this paper, we propose a novel framework to protect the distributed structural graphs by introducing the local differential privacy to graph embeddings and handle the permutation invariant issue by taking the Gromov-Wasserstein distance for

References

[Barbe *et al.*, 2020] Amélie Barbe, Marc Sebban, Paulo Gonçalves, Pierre Borgnat, and Rémi Gribonval. Graph diffusion wasserstein distances. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 577–592. Springer, 2020.

[Borgwardt and Kriegel, 2005] Karsten M Borgwardt and Hans-Peter Kriegel. Shortest-path kernels on graphs. In *ICDM*, pages 8–pp. IEEE, 2005.

[Bukaty, 2019] Preston Bukaty. *The California Consumer Privacy Act (CCPA): An Implementation Guide*. IT Governance Publishing, 2019.

[Chan *et al.*, 2000] Tony F Chan, Jason Cong, Tianming Kong, and Joseph R Shinnerl. Multilevel optimization for large-scale circuit placement. In *ICCAD*, pages 171–176. IEEE, 2000.

[Coley *et al.*, 2019] Connor W Coley, Wengong Jin, Luke Rogers, Timothy F Jamison, Tommi S Jaakkola, William H Green, Regina Barzilay, and Klavs F Jensen. A graph-convolutional neural network model for the prediction of chemical reactivity. *Chemical science*, 10(2):370–377, 2019.

[Demetci *et al.*, 2020] Pinar Demetci, Rebecca Santorella, Björn Sandstede, William Stafford Noble, and Ritambhara Singh. Gromov-wasserstein optimal transport to align single-cell multiomics data. *bioRxiv preprint bioRxiv:2020.04.28.066787*, 2020.

[Fan *et al.*, 2019] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. Graph neural networks for social recommendation. In *The World Wide Web Conference*, pages 417–426, 2019.

[Hamilton *et al.*, 2017] William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, pages 1025–1035, 2017.

[He *et al.*, 2020] Chaoyang He, Songze Li, Jinhyun So, Xiao Zeng, Mi Zhang, Hongyi Wang, Xiaoyang Wang, Praneeth Vepakomma, Abhishek Singh, Hang Qiu, et al. Fedml: A research library and benchmark for federated machine learning. *arXiv preprint arXiv:2007.13518*, 2020.

[He *et al.*, 2021] Chaoyang He, Keshav Balasubramanian, Emir Ceyani, Yu Rong, Peilin Zhao, Junzhou Huang, Murali Annavaram, and Salman Avestimehr. Fedgraphnn: A federated learning system and benchmark for graph neural networks. *arXiv preprint arXiv:2104.07145*, 2021.

[Jin *et al.*, 2017] Chi Jin, Rong Ge, Praneeth Netrapalli, Sham M Kakade, and Michael I Jordan. How to escape saddle points efficiently. In *ICML*, pages 1724–1732. PMLR, 2017.

[Koopmans and Beckmann, 1957] Tjalling C Koopmans and Martin Beckmann. Assignment problems and the location of economic activities. *Econometrica: journal of the Econometric Society*, pages 53–76, 1957.

[Li *et al.*, 2014] Mu Li, David G Andersen, Jun Woo Park, Alexander J Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J Shekita, and Bor-Yiing Su. Scaling distributed machine learning with the parameter server. In *OSDI*, pages 583–598, 2014.

[Li *et al.*, 2020] Xiangfeng Li, Shenghua Liu, Zifeng Li, Xiaotian Han, Chuan Shi, Bryan Hooi, He Huang, and Xueqi Cheng. Flowscope: Spotting money laundering based on graphs. In *AISTATS*, volume 34, pages 4731–4738, 2020.

[Liao *et al.*, 2021] Peiyuan Liao, Han Zhao, Keyulu Xu, Tommi Jaakkola, Geoffrey J Gordon, Stefanie Jegelka, and Ruslan Salakhutdinov. Information obfuscation of graph neural networks. In *International Conference on Machine Learning*, pages 6600–6610. PMLR, 2021.

[McMahan *et al.*, 2017] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, pages 1273–1282. PMLR, 2017.

[Mémoli and Sapiro, 2006] Facundo Mémoli and Guillermo Sapiro. Computing with point cloud data. In *Statistics and analysis of shapes*, pages 201–229. Springer, 2006.

[Mémoli, 2011] Facundo Mémoli. Gromov–wasserstein distances and the metric approach to object matching. *Foundations of computational mathematics*, 11(4):417–487, 2011.

[Morris *et al.*, 2020] Christopher Morris, Nils M Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. Tudataset: A collection of benchmark datasets for learning with graphs. *arXiv preprint arXiv:2007.08663*, 2020.

[Niu *et al.*, 2020] Chenhao Niu, Yang Song, Jiaming Song, Shengjia Zhao, Aditya Grover, and Stefano Ermon. Permutation invariant graph generation via score-based generative modeling. In *AISTATS*, pages 4474–4484. PMLR, 2020.

[Peyré *et al.*, 2016] Gabriel Peyré, Marco Cuturi, and Justin Solomon. Gromov-wasserstein averaging of kernel and distance matrices. In *ICML*, pages 2664–2672. PMLR, 2016.

[Regulation, 2018] General Data Protection Regulation. General data protection regulation (gdpr). *Intersoft Consulting*, 24(1), 2018.

[Sajadmanesh and Gatica-Perez, 2020] Sina Sajadmanesh and Daniel Gatica-Perez. Locally private graph neural networks. *arXiv preprint arXiv:2006.05535*, 2020.

[Shervashidze *et al.*, 2011] Nino Shervashidze, Pascal Schweitzer, Erik Jan Van Leeuwen, Kurt Mehlhorn, and Karsten M Borgwardt. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(9), 2011.

[Smith *et al.*, 2017] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet Talwalkar. Federated multi-task learning. In *AISTATS*, pages 4427–4437, 2017.

[Titouan *et al.*, 2019] Vayer Titouan, Nicolas Courty, Romain Tavenard, and Rémi Flamary. Optimal transport for structured data with application on graphs. In *ICML*, pages 6275–6284. PMLR, 2019.

[Vishwanathan *et al.*, 2010] Vichy Vishwanathan, Nicol Schraudolph, Risi Kondor, and Karsten Borgwardt. Graph kernels. *JMLR*, 11:1201–1242, 2010.

[Wang *et al.*, 2020] Binghui Wang, Ang Li, Hai Li, and Yiran Chen. Graphfl: A federated learning framework for semi-supervised node classification on graphs. *arXiv preprint arXiv:2012.04187*, 2020.

[Wang *et al.*, 2021] Chunnan Wang, Bozhou Chen, Geng Li, and Hongzhi Wang. Fl-agcns: Federated learning framework for automatic graph convolutional network search. *arXiv preprint arXiv:2104.04141*, 2021.

[Wu *et al.*, 2020] Shiwen Wu, Fei Sun, Wentao Zhang, and Bin Cui. Graph neural networks in recommender systems: a survey. *arXiv preprint arXiv:2011.02260*, 2020.

[Xu *et al.*, 2019a] Hongteng Xu, Dixin Luo, and Lawrence Carin. Scalable gromov-wasserstein learning for graph partitioning and matching. In *NeurIPS*, pages 3052–3062, 2019.

[Xu *et al.*, 2019b] Hongteng Xu, Dixin Luo, Hongyuan Zha, and Lawrence Carin Duke. Gromov-wasserstein learning for graph matching and node embedding. In *ICML*, pages 6932–6941, 2019.

[Xu *et al.*, 2019c] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations (ICLR)*, 2019.

[Yang *et al.*, 2016] Zhilin Yang, William Cohen, and Ruslan Salakhudinov. Revisiting semi-supervised learning with graph embeddings. In *ICML*, pages 40–48. PMLR, 2016.

[Zügner *et al.*, 2018] Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. Adversarial attacks on neural networks for graph data. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2847–2856, 2018.