Privacy-Utility Trade-offs in Neural Networks for Medical Population Graphs: Insights from Differential Privacy and Graph Structure

Tamara T. Mueller*
Technical University of Munich
tamara.mueller@tum.de

Maulik Chevli*
Technical University of Munich
maulikk.chevli@tum.de

Ameya Daigavane
Massachusetts Institute of Technology

Daniel Rueckert
Technical University of Munich
Imperial College London

Georgios Kaissis
Technical University of Munich
Helmholtz Zentrum Muenchen
g.kaissis@tum.de

Abstract

Differential privacy (DP) is the gold standard for protecting individuals' data while enabling deep learning. It is well-established and frequently used for applications in medicine and healthcare to protect sensitive patient data. When using graph deep learning on so-called population graphs, however, the application of DP becomes more challenging compared to grid-like data structures like images or tables. In this work, we initiate an empirical investigation of differentially private graph neural networks on population graphs in the medical domain by examining privacy-utility trade-offs under different graph learning methods on both real-world and synthetic datasets. We compare two state-of-the-art methods for differentially private graph deep learning and empirically audit privacy guarantees through node membership inference and link stealing attacks. We focus on the impact of the graph structure, one of the most important inherent challenges of medical population graphs. Our findings highlight the potential and challenges of this specific DP application area. Moreover, we find that the underlying graph structure constitutes a potential factor for larger performance gaps on one of the explored methods by showing a correlation between the graphs' homophily and the accuracy of the trained model.

1 Introduction

Graph Neural Networks (GNNs) are powerful tools to apply Deep Learning (DL) techniques to non-Euclidean data structures, such as graphs, meshes, or point clouds (1). They have been successfully employed across a range of data structures and demonstrated their versatility and usefulness in multiple domains, including medicine. The application of GNNs to medical data has been shown to improve performance of diagnostic systems (2; 3; 4; 5), structure medical data in more accurate ways (6; 7; 8), and to enable the modelling of medical knowledge in the form of knowledge graphs (9). The utilisation of GNNs has shown improved performance, even on datasets not exhibiting an intrinsic

*equal contribution

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

graph structure, e.g. 3D point clouds (10). The application of GNNs has therefore been expanded to datasets which require constructing a graph structure prior to learning. One such example are medical population graphs (2). Here, a cohort is represented by one (typically large) graph, in which nodes represent subjects and edges connect similar subjects. The construction of the graph's structure is an important step in this pipeline, since a graph of poor structural quality can substantially hinder graph learning (11; 12). This has been attributed to several different graph properties, one of them being *homophily* (13). Homophily is a measure for the ratio between identically and differently labelled neighbours in a graph. A high homophily indicates that the majority of nodes in all neighbourhoods in the graph share the same label as the node of interest.

Despite the aforementioned advantages of Artificial Intelligence (AI) methods, it has been shown that AI models are vulnerable to attacks designed to extract sensitive data and information (14). This is especially problematic when applying AI methods to highly sensitive data, such as in the medical domain. In order to combine the advances of AI with the protection of sensitive data, Differential Privacy (DP) has become the gold standard for training AI models, while providing formal privacy guarantees (15). Through an adapted training pipeline, DP enables the training of deep neural networks, while providing formal privacy guarantees. DP was originally conceived for tabular datasets, where individual data points can be treated separately. This is not the case in graph learning settings, where nodes are connected to each other and share information (16). This requires the design of new DP formulations for graph-structured data and GNNs as well as new methods to ensure privacy during GNN training. One of the main drawbacks of DP training is a resulting deterioration in model performance, resulting in the so-called privacy-utility trade-off.

Contributions In this work, we investigate the privacy-utility trade-offs of DP GNN training on medical population graphs. Our contributions are as follows: (1) To the best of our knowledge, our work demonstrates the first successful application of DP to multi-hop GNNs in medical population graphs, where we (2) specifically compare two different lines of methods: one introduced by Diagavane et al. (17) that is based on graph convolutional networks (18), which we call *DP-GCNs* throughout this work and one introduced by Sajadanesh et al. (19), which is called Graph Aggregation Perturbation (*GAP*)[2], (3) empirically investigate the success of Membership Inference Attacks (MIAs) and link stealing attacks at different levels of privacy protection with both methods, and (4) analyse the interplay between graph structure and the accuracy of differentially private GNNs, highlighting homophily as a key factor influencing model utility.

2 Background and Related Work

2.1 Differential Privacy

Differential Privacy (DP) (15) is a framework designed to enable the analysis of datasets while protecting the privacy of individual data owners. Intuitively, an algorithm is differentially private, if the output remains almost the same, indifferent of whether a single subject's data was part of the training set or not. Let an analyst $\mathcal{A}$ possess a database $\mathcal{D}$, which contains the sensitive data of individuals. A neighbouring database $\mathcal{D}'$ differs from $\mathcal{D}$ in exactly one entry. For graph data, the definition of this neighbouring database $\mathcal{D}'$ depends on the privacy notion of interest and will be discussed below. $\mathcal{A}$ executes a query function $f : X \to Y$ over a database.

Definition 2.1 (ε, δ)-Differential Privacy) *A randomised mechanism $\mathcal{M}$ satisfies (ε, δ)-DP if, for all adjacent database $\mathcal{D}$ and $\mathcal{D}'$ and all subsets $\mathcal{S} \subseteq$ Range$(\mathcal{M})$, the following statement holds:*

$$\mathbb{P}\left[\mathcal{M}(\mathcal{D}) \in \mathcal{S}\right] \leq e^{\varepsilon}\mathbb{P}\left[\mathcal{M}(\mathcal{D}') \in \mathcal{S}\right] + \delta. \tag{1}$$

We note that the statement is symmetric and also holds if D and D' are swapped.

Definition 2.2 (L_2 Sensitivity of f) *Let D, D' and $f : X \to Y$ be defined as above with d_X being an associated metric over the metric space X and d_Y the L_2-distance on the space Y of the outputs of f, then the L_2-sensitivity Δ of f is defined as:*

$$\Delta_f := \max_{D,D' \in X\, D \simeq D'} \frac{d_Y\left(f(D), f(D')\right)}{d_X\left(D, D'\right)}. \tag{2}$$

[2]We note that a recent pre-print (20) questions the DP guarantee and sensitivity analysis of this work. So we recommend taking the results of this method with care.

Definition 2.3 (Gaussian Mechanism) *A Gaussian mechanism $\mathcal{GM}$ operates on the output $y \in \mathbb{R}^n$ of f by adding noise drawn from a normal distribution with a variance calibrated to the L_2-sensitivity of f:*

$$\mathcal{GM}(y) = y + \xi, \tag{3}$$

where $\xi \sim \mathcal{N}(0, \sigma^2 \mathbb{I}^n)$ and $\mathbb{I}^n$ is the identity matrix with n diagonal elements.

DP Training of Neural Networks A commonly used method to train neural networks with DP is using DP-stochastic gradient descent (DP-SGD) (21), where the Gaussian Mechanism (GM) is used to privatise *per-sample-gradients* before updating the model parameters. Before that, the L_2-norm of each gradient is bounded to a specific threshold. We note that this method is applicable to all other (first-order) optimisation techniques, such as Adam.

2.2 Differential Privacy on Graphs

When transitioning from tabular datasets to graph structures, the notion of neighbouring datasets must be adapted. Three main notions of DP on graph-structured data exist: *node-level*, *edge-level*, and *graph-level* DP (22). In this work, we focus on *node-level* DP and compare it with guarantees provided by *edge-level* DP. Under *node-level* DP, two graphs $\mathcal{G}$ and $\mathcal{G}'$ are neighbouring if they differ in a single node and all its adjacent edges. Here, the sensitive information, such as features from medical images, is assumed to be stored in the node features of the graph as well as encoded in the graph structure. Under *edge-level* DP, two graphs $\mathcal{G}$ and $\mathcal{G}'$ are neighbouring if they differ in a single edge. Here, the sensitive information is assumed to only be stored in network connections.

So far, only few works have investigated DP training of GNNs for node classification tasks. Daigavane et al. (17) were among the first to introduce a privacy amplification by sub-sampling technique for multi-layer GNN training with DP-stochastic gradient descent (DP-SGD). In order to enable a sensitivity analysis for DP-SGD in multi-layer GNNs, the authors apply a graph neighbourhood sampling scheme. The number of k-hop neighbours is bounded to a maximum node degree. This ensures that the learned feature embeddings over the course of training are influenced by at most a bounded number of nodes. Furthermore, the standard privacy amplification by sub-sampling technique for DP-SGD is extended, such that a gradient can depend on multiple subjects in the dataset: First, a local k-hop neighbourhood of each node with a bounded number of neighbours is sampled. Next, a subset $\mathcal{B}_t$ of n sub-graphs is chosen uniformly at random from the set of sub-graphs that constitute the training set. On these sub-samples ("mini-batches"), standard DP-SGD is applied by clipping the per-sample gradients, adding noise, and using the noisy gradients for the update step. The noise is hereby calibrated to the sensitivity with respect to any individual node, which has been bounded via sub-sampling of the input graph. A different line of work by Sajadmanesh et al. (19) introduces a method for DP training of multi-layer GNNs via *aggregation perturbation*, i.e. by adding noise to the aggregation function of the GNN, which they call *GAP*. This method ensures edge-level DP and can be extended for node-level DP by bounding the degree of the graph and applying the same privacy-preserving aggregation mechanism calibrated to the bounding degree. GAP also uses the same sub-graphing technique introduced by (17) for bounding the maximum degree of the graph. Since GAP also uses MLPs in its architecture, standard DP-SGD (21) is used to train these MLPs.

In our experiments, we compare GAP and DP-GCNs and evaluate and interpret their performance. It is important to note that GAP provides privacy guarantees at inference *without incurring any extra privacy cost* than what was spent training the model, unlike DP-GCN, where incorporating inference-time privacy requires an *additional privacy cost* and is not provided by default (19; 17). We still compare the performance of both methods under the same privacy budgets but highlight that the privacy guarantees differ in practice.

2.3 Empirical Auditing of DP Guarantees

In order to audit the privacy guarantees of AI models, specific attacks can be performed that empirically test the data leakage of a model. DP with sufficiently strong guarantees naturally protects against MIAs, which aim to infer whether a certain individual was part of the training set or not. There are a few works investigating MIAs on GNNs (23; 24; 25; 26). These and works on other privacy attacks on GNNs such as *link stealing* attacks (27; 28) and *inference attacks* (29; 30) highlight an increased vulnerability of GNNs compared to non-graph AI applications. Sensitive information can be located in the node features, or in the graph structure, or both. Furthermore, the message

passing between nodes makes a clear separation between individual nodes impossible. We adapt the MIA technique by (31) to GNNs for the purpose of empirically validating the privacy guarantees. Furthermore, we perform link-stealing attacks to evaluate the protection of edges in the graph using *LinkTeller* (27). In general, link stealing attacks aim to infer the graph structure from the GNN.

3 Methods

All experiments are performed under transductive learning, where all node features and edges are included in the forward pass, but only the training labels are used for backpropagation. We use two different DP graph learning methods: (a) DP-GNNs that were introduced by (17) and (b) GAP networks (19) that follow a different methodology using aggregation perturbation. As a baseline, we also train Multi-layer Perceptrons (MLPs) on the same datasets. For all trainings, we use the Adam optimiser. DP-GCNs (17) use a more classical GNN architecture and training pipeline and adapts the privacy amplification by sub-sampling technique to multi-hop GNNs, using DP-SGD. This requires the application of a sub-graphing mechanism to the graph structure. GAP on the other hand constructs a different architecture for GNN training, consisting of three modules: (1) an encoder module consisting of an MLP, (2) an aggregation module, which performs the message passing across the neighbourhoods, and (3) a classification module, consisting of one MLPs for each GNN layer as well as one from the encoder directly. We investigate their differing behaviour under attacks and evaluate their privacy-utility trade-offs in Section 4.

In order to evaluate our experiments under standardised conditions, we generated a synthetic binary classification dataset as a benchmark. This allows us to investigate the impact of different graph structures on the performance of DP population graphs under defined conditions. We use three medical datasets which are frequently used in the context of population graphs: The **TADPOLE** dataset studies Alzheimer's disease and functions as a benchmark dataset for population graphs (2; 3). The **ABIDE** dataset from the autism brain imaging data exchange (32) is a binary classification task to identify subjects suffering from autism. The ABIDE dataset is highly challenging and therefore lends itself to investigating the impact of the graph structure on our experiments. We also use an in-house **COVID** dataset as a realistic, noisy, and small medical dataset, with the task of predicting whether a COVID patient will require intensive care unit (ICU) treatment. All graph structures are generated using the k-nearest neighbour approach (33), where k is a hyperparameter that specifies how many neighbours each node has. The k most similar neighbours are connected to each other. We use $k = 5$ for all medical population graph datasets and $k = 10$ for the synthetic dataset. We also use k as the maximum bounding degree for all GAP experiments. Details about δ values used for training and the homophily of all datasets are summarised in the appendix in Table 8. We note that the graph construction is non-private and happens prior to training, which is in line with prior works on datasets with a pre-defined graph structure. We, therefore, consider the edges to contain private information as well –indicating which subjects are similar to each other–, which is why we use *node-level* DP for most of our experiments. This ensures the protection of both, the node features as well as the whole graph structure.

4 Experiments and Results

In this section, we summarise the results of DP-GCNs and GAP on population graph datasets. We (1) compare the performance of the different models, (2) empirically audit the data leakage via MIAs and link stealing attacks, and (3) investigate the impact of the graph structure on the model performance, using a synthetically generated dataset and the homophily metric for assessing the graph structure.

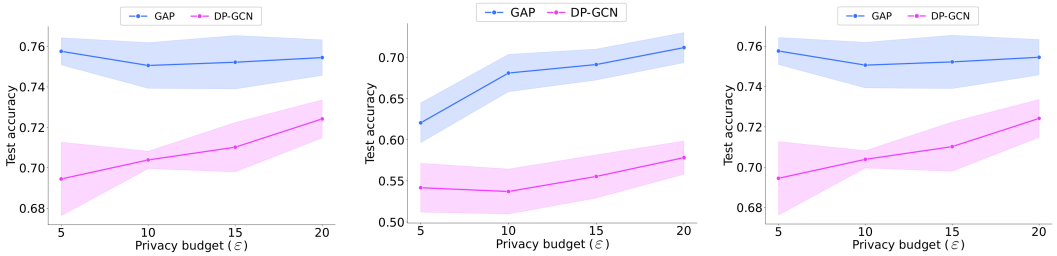
4.1 DP Training of GNNs on Population Graphs

We summarise the results of non-DP and DP training at different privacy budgets of both methods and under node-level DP in Table 1 and visualised in Figure 1. As anticipated by the privacy-utility trade-off, a stronger privacy guarantee results in lower model performance over all datasets and both methods. The impact of a decreasing ε is similar for both methods, but GAP outperforms DP-GCNs under all private and non-private settings on all three datasets. On the COVID dataset, both methods show a high standard deviation across different seed runs, indicating overall highly inconsistent performance. We attribute this to the very small size of the dataset.

DP training of GNNs relies on a sub-sampling of the graph structure, which we also evaluate separately from the non-DP and the DP cases in Table 1 ("Sub-graphing"). For two datasets, the model trained without DP, but employing sub-graph sampling and gradient clipping, outperforms the non-DP model trained without these techniques. We attribute this to the regularising effect of both aforementioned methods. Interestingly, GAP still achieves better performance than DP-GCNs with sub-graphing. An evaluation of the impact of the number of hops for GAP can be found in the appendix Table 9. We restrain the sub-graphing experiments to the DP-GCN models and assume similar behaviour for the GAP experiments.

Table 1: **Summary of Results** with DP-GCNs (17) and GAP (19) under different privacy budgets ε, using node-level DP. The results are averaged over 5 random seeds. For DP-GCN we also report the performance of sug-graphing only. The best performance is highlighted in **bold**.

	Method	Non-DP	Sub-graphing	DP $\varepsilon = 20$	DP $\varepsilon = 15$	DP $\varepsilon = 10$	DP $\varepsilon = 5$
TADPOLE	MLP	**79.84 ± 1.52**	-	77.25 ± 1.49	77.96 ± 1.23	76.94 ± 1.58	77.41 ± 2.02
	DP-GCN	72.73 ± 1.39	**76.09 ± 1.73**	72.42 ± 0.94	71.02 ± 1.22	70.39 ± 0.43	69.45 ± 1.82
	GAP	**78.82 ± 2.66**	-	75.45 ± 0.88	75.22 ± 1.32	75.06 ± 1.13	75.76 ± 0.67
ABIDE	MLP	71.09 ± 3.50	-	**71.54 ± 4.27**	72.00 ± 2.29	71.31 ± 1.46	67.43 ± 2.98
	DP-GCN	58.86 ± 0.81	**65.14 ± 2.37**	57.83 ± 2.02	55.54 ± 2.62	53.71 ± 2.73	54.17 ± 2.97
	GAP	**72.11 ± 1.17**	-	71.20 ± 1.83	69.14 ± 1.88	68.11 ± 2.27	62.06 ± 2.41
COVID	MLP	**73.85 ± 07.84**	-	70.77 ± 13.23	70.77 ± 13.23	70.77 ± 13.23	73.85 ± 13.41
	DP-GCN	**73.85 ± 10.32**	69.23 ± 10.88	56.92 ± 08.77	58.46 ± 06.88	61.54 ± 07.69	46.15 ± 17.20
	GAP	60.00 ± 08.97	-	**66.15 ± 11.51**	64.62 ± 11.51	64.62 ± 10.43	61.54 ± 09.73



(a) TADPOLE (b) ABIDE (c) COVID

Figure 1: Performance of DP-GCNs and GAP on the datasets (a) TADPOLE, (b) ABIDE, and (c) COVID under node-level DP.

4.2 Empirical Auditing via Attacks

In order to empirically evaluate the privacy leakage of our DP population graph models, we perform MIAs as well as link-stealing attacks on the different models.

4.2.1 Membership Inference Attacks

The dependencies between graph elements render GNNs more vulnerable to MIA (34). Moreover, in the transductive setting of graph learning, test node features are included in the forward pass, which further facilitates MIA (24). To empirically audit the privacy leakage of sensitive patient data from our GNN models, we employ the MIA implementation of Carlini et al. (31). The adversary/auditor in this membership inference scenario has full access to the trained model, its architecture, and the graph, including its ground-truth labels (31). We train several shadow models to estimate the models' output logit distributions and create a classifier that predicts whether a specific example was used as training data for the model.

We perform MIA on the GNNs trained on the TADPOLE dataset, as it is known that higher model accuracy improves MIA success (31) and the interest is to verify the privacy guarantees in the worst practical case possible. We report the true positive rate (TPR) at three fixed, low false positive rates (FPR) $(0.1\%, 0.5\%, 1\%)$ in Table 2. We also derive the maximum TPR (i.e. power) that is theoretically achievable for a given (ε, δ) setting through the duality between (ε, δ)-DP and hypothesis testing DP (35). We will refer to this maximum achievable TPR as the adversary's *supremum power*

$\mathcal{P}$. Visualisations of the log-scale receiver operating characteristic (ROC) curve of the MIA attacks performed on the DP-GCNs and GAP can be found in the appendix in Figure 4.

Table 2: Comparison of LiRA MIA (31) experiments on DP-GCN (17) and GAP (19) at different privacy budgets on the TADPOLE dataset

Training	TPR % @ **0.1% FPR**			TPR % @ **0.5% FPR**			TPR % @ **1% FPR**		
	$\mathcal{P}$	**DP-GCN**	**GAP**	$\mathcal{P}$	**DP-GCN**	**GAP**	$\mathcal{P}$	**DP-GCN**	**GAP**
Non-DP	-	0.92	1.84	-	0.92	2.01	-	2.30	5.18
DP ($\varepsilon = 20$)	100.0	0	0.17	100.0	0.69	1.55	100.0	2.30	2.23
DP ($\varepsilon = 15$)	100.0	0	0	100.0	0.46	1.03	100.0	0.69	1.89
DP ($\varepsilon = 10$)	100.0	0	0	100.0	0	1.37	100.0	0.23	2.23
DP ($\varepsilon = 5$)	14.85	0	0	74.22	0	0.17	100.0	0	0.34

For FPR-values < 0.001, MIA is unsuccessful for DP-GCNs. As the FPR tolerance is increased, all models trained with weaker privacy guarantees ($\varepsilon \in \{20, 15\}$) yield positive TPR when attacked, with TPR values approaching these of models trained without DP guarantees (*Non-DP* and *Sub-graphing*) in case of $\varepsilon = 20$. Interestingly, the model trained at $\varepsilon = 5$ successfully resists membership inference even at an FPR value of 0.01. Moreover, we observe that the GNN trained with clipped gradients is less vulnerable to membership inference than the GNNs trained without gradient clipping. This is in line with the findings in (31; 36) that clipping the gradients during training offers some (empirical) protection against MIAs. Furthermore, we note that MIAs on GAP are more successful, indicating more empirical data leakage. These results align with the findings in (19). We attribute these results mainly to the better performance of GAP compared to DP-GCN. It has been shown that more accurate models are more vulnerable to LiRA (31).

4.2.2 Link Stealing Attacks

We also perform link stealing attacks, using *LinkTeller* (27) on the different GNN architectures. The *LinkTeller* attack can be compared to performing a MIA on each edge and is guided by an estimated graph density. Since, in the context of medical population graphs, the graph is constructed with a specific number of neighbours (k), we assume the estimated density to match the actual degree of the graph. The results of the *LinkTeller* attacks on the TADPOLE dataset using GAP are summarised in Table 3. Since DP-GCNs do not provide privacy guarantees at inference time (17), all *LinkTeller* attacks achieved almost perfect precision and recall on these models. The results of the *LinkTeller* attacks on GAP models are summarised in Table 3. We evaluate these attacks using 1- and 2-hop GNNs and different maximum degrees, meaning that the corresponding graphs were constructed using different k for the k-NN method. As expected, with lower ε values, i.e. stronger privacy, the accuracy of the *LinkTeller* attack decreases. Moreover, both, 1- and 2-hop GNNs offer similar protection against the attacks and a lower-density graph yields better protection against *LinkTeller*.

Table 3: Results of *LinkTeller* attacks on the TADPOLE dataset using GAP in %.

Density	**Baseline**		**Privacy setting**	**1-hop-GNNs**		**2-hop-GNNs**	
	Precision	**Recall**		**Precision**	**Recall**	**Precision**	**Recall**
			non-DP	100	4.67	87.50	3.27
			$\varepsilon = 20$	0.98	0.93	0	0
k=5	0.64	0.64	$\varepsilon = 10$	0.98	0.93	0	0
			$\varepsilon = 5$	0.49	0.47	0	0
			$\varepsilon = 1$	0.49	0.47	0	0
			non-DP	100	90.87	51.51	46.81
			$\varepsilon = 20$	2.60	2.70	2.35	2.44
k=40	6.20	6.20	$\varepsilon = 10$	2.56	2.65	2.71	2.80
			$\varepsilon = 5$	2.61	2.70	2.55	2.65
			$\varepsilon = 1$	2.45	2.23	2.40	2.49

4.3 Impact of Graph Structure on Performance

The interaction between memorisation and generalisation in neural networks is of particular interest to the privacy community. Feldman (37) hypothesises that especially noisy and atypical data from the long tail of the data distribution requires memorisation. This increases the negative impact of DP training for those samples. To investigate the applicability of the long-tail hypothesis to graph

databases, we evaluate the impact of graph structure, measured in terms of homophily, on accuracy. Concretely, we hypothesise that graphs with low homophily are "noisier" and therefore suffer more from DP training. For example, at a homophily of 0.5 on a binary node classification dataset, on average, half of all neighbours have the same label and the other half the opposite label. Thus, when applying message-passing on such a graph, the node features will get averaged over an approximately equal number of nodes from both labels. This makes it nearly impossible to learn meaningful node feature embeddings, such that learning likely relies nearly exclusively on memorisation. We note that, in a binary classification task, homophily values are symmetric about 0.5. Interestingly, we observe a highly different impact of the graph structure on the performance of DP-GCNs and GAP under node-level DP. While DP-SGD GNNs show consistently worse performance with low homophily graphs, the GAP experiments under node-level DP indicate that the graph structure has almost no impact on the performance of the model, which we will investigate in more detail below.

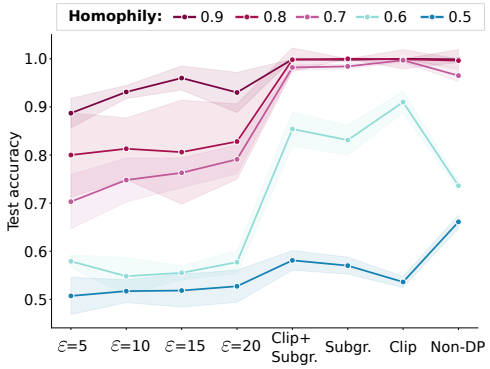
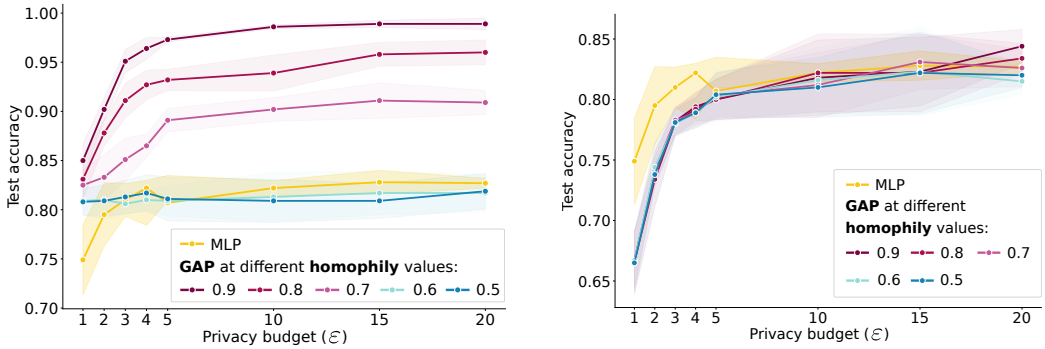


Figure 2: Impact of homophily on performance of **DP-GCN** (17) under node-level DP on the synthetic dataset. *Clip*: clipping only, *Subgr.*: sub-graphing only.



(a) Edge-level DP.

(b) Node-level DP.

Figure 3: Impact of homophily on performance of **GAP** under (a) edge-level DP and (b) node-level DP on the synthetic dataset.

DP-GCNs The results using a synthetic dataset with different levels of homophily using DP-GCNs are summarised in Table 4 and visualised in Figure 2. The generalisation gap of DP-GCNs is especially large on low-homophily graphs, indicating over-fitting in the non-DP setting (not shown in the Table). Model accuracy in the non-DP setting profits more from the regularising effects of clipping and sub-graphing in graphs with lower homophily (0.6) compared to high homophily (0.9). As expected, at the lowest homophily of 0.5, learning is severely compromised without DP, and the regularising effect of clipping and sub-graphing actually harms accuracy. Moreover, learning is nearly impossible with DP, corroborating that, in this setting, the accuracy benefit of non-DP learning is mostly due to memorisation. Under DP, graphs with high homophily (0.9) suffer a lower performance decrease compared to graphs with low homophily, likely due to the favourable graph structure for the learning task, i.e. not requiring strong memorisation.

Table 4: **Impact of homophily** on **DP-GCNs** using the synthetic dataset. All results refer to test accuracy (%), with the best ones highlighted in **bold**. Hom.: homophily.

Hom.	Non-DP	Clipping	Sub-graphing	Subg. + Clip.	DP ($\varepsilon = 20$)	DP ($\varepsilon = 15$)	DP ($\varepsilon = 10$)	DP ($\varepsilon = 5$)
0.9	99.90 ± 2.00	**100.0 ± 0.00**	99.80 ± 0.40	99.90 ± 0.00	93.00 ± 4.17	96.00 ± 2.49	93.10 ± 1.36	88.70 ± 3.06
0.8	99.62 ± 0.37	99.90 ± 0.00	**100.0 ± 0.00**	99.80 ± 2.45	82.80 ± 7.83	80.60 ± 10.8	81.30 ± 6.40	80.00 ± 8.76
0.7	96.50 ± 1.23	**99.70 ± 0.40**	98.42 ± 0.51	98.20 ± 0.50	79.10 ± 3.00	76.30 ± 3.10	74.80 ± 4.55	70.30 ± 5.64
0.6	73.60 ± 0.97	**91.00 ± 2.24**	83.10 ± 3.10	85.40 ± 3.54	57.72 ± 2.38	55.50 ± 1.45	54.80 ± 3.87	57.90 ± 1.32
0.5	**66.10 ± 1.361**	53.60 ± 1.16	57.00 ± 1.76	58.10 ± 2.03	52.71 ± 3.34	51.82 ± 3.43	51.70 ± 2.38	50.70 ± 3.87

GAP Corresponding results using GAP and both node-level and edge-level DP are summarised in Table 5 and visualised in Figure 3. We here investigate two different notions of DP: edge-level and node-level DP. The difference in overall model performance between node-level and edge-level DP on GAP is most likely due to the proportionally larger noise that is added under node-level DP compared to edge-level DP. Under edge-level DP (Figure 3a) we observe a similar behaviour to DP-GCNs, where high-homophily graphs outperform the MLP by a large margin while low-homophily graphs achieve lower results than the MLP. However, for node-level DP (Figure 3b), the graph structure (measured in homophily) shows almost no impact on the performance of GAP. All models achieve similar results. We attribute this to the architecture of GAP, which allows information content to flow through the MLPs of modules (1) to (3), without any impact from the graph structure. To corroborate this hypothesis, we conduct additional experiments investigating the impact of the individual modules using occlusion methods. Concretely, we occlude the input of the third module of GAP –the classification module– in three different ways. We (a) only use one MLP that takes the normalised node features as input, which are directly passed through from the encoder module (`ENC`), (b) use only the MLPs from the aggregated node features of the k-hop layers of the GNN (`NEIGHs`), or (c) the original GAP setup, which uses both the `ENC` and `NEIGHs` MLPs (`ORIG`). We note that the `ENC` module does not use the graph structure and therefore the notion of homophily does not apply to this part of the model. We perform the experiments under different homophily values on the respective occluded versions of GAP on the synthetic dataset and summarise the results in Table 6. The performance of the `ENC` module alone aligns strongly with the performance of the MLP, while the `NEIGHs` performance is highly dependent on the graph homophily. This is expected since only the `NEIGHs` module encodes the message passing across neighbours. It can therefore improve performance of the network on high-homophily graphs. An ablation study on the impact of the density of the graph can be found in the appendix in Tables 10 and 11.

Table 5: **Impact of homophily** on **GAP** under node-level and edge-level DP. The results are accuracy values on the test set in %. Hom.: homophily. GNNs outperforming the respective MLP are **bold**.

DP	Method	Hom.	Non-DP	$\varepsilon = 20$	$\varepsilon = 15$	$\varepsilon = 10$	$\varepsilon = 5$	$\varepsilon = 4$	$\varepsilon = 3$	$\varepsilon = 2$	$\varepsilon = 1$
	MLP	-	84.00 ± 1.10	82.70 ± 0.51	82.80 ± 1.21	82.20 ± 0.75	80.70 ± 2.29	82.20 ± 1.29	81.00 ± 1.67	79.50 ± 3.22	74.90 ± 3.57
node-level	GAP	0.9	**99.10 ± 0.20**	**84.40 ± 1.39**	82.30 ± 2.84	81.80 ± 3.12	80.00 ± 1.67	79.40 ± 1.20	78.20 ± 1.12	74.10 ± 1.16	66.70 ± 2.29
		0.8	**96.00 ± 0.71**	**83.40 ± 1.24**	82.20 ± 3.22	81.90 ± 3.15	80.00 ± 1.67	79.20 ± 1.29	78.10 ± 1.11	73.40 ± 2.31	66.60 ± 2.58
		0.7	**90.30 ± 1.96**	82.60 ± 1.53	83.10 ± 2.27	81.20 ± 2.68	80.30 ± 1.86	78.90 ± 1.28	78.30 ± 1.12	74.20 ± 1.50	66.50 ± 2.51
		0.6	83.90 ± 2.29	81.50 ± 1.05	82.30 ± 3.64	81.60 ± 2.71	80.20 ± 1.99	78.90 ± 1.20	78.00 ± 1.14	74.40 ± 1.56	66.70 ± 2.40
		0.5	81.60 ± 1.50	82.00 ± 1.00	82.20 ± 3.40	81.00 ± 2.47	80.40 ± 1.85	78.90 ± 1.20	78.10 ± 1.11	73.80 ± 2.60	66.50 ± 2.59
edge-level	GAP	0.9	**99.10 ± 0.20**	**98.90 ± 0.58**	**98.90 ± 0.37**	**98.60 ± 0.20**	**97.30 ± 0.24**	**96.40 ± 1.11**	**95.10 ± 1.20**	**90.20 ± 1.17**	**85.00 ± 1.76**
		0.8	**96.00 ± 0.71**	**96.00 ± 1.26**	**95.80 ± 1.21**	**93.90 ± 1.83**	**93.20 ± 1.08**	**92.70 ± 1.50**	**91.10 ± 1.77**	**87.80 ± 1.33**	**83.10 ± 2.01**
		0.7	**90.30 ± 1.96**	**90.90 ± 1.24**	**91.10 ± 1.80**	**90.20 ± 1.21**	**89.10 ± 1.20**	**86.50 ± 1.18**	**85.10 ± 2.20**	**83.30 ± 2.29**	**82.50 ± 1.58**
		0.6	83.90 ± 2.29	81.70 ± 1.91	81.70 ± 2.20	81.30 ± 2.50	80.90 ± 1.53	81.00 ± 1.05	80.60 ± 1.07	**81.00 ± 1.95**	**80.90 ± 1.46**
		0.5	81.60 ± 1.50	81.88 ± 1.81	80.90 ± 1.77	80.90 ± 2.20	**81.10 ± 2.24**	81.70 ± 1.86	**81.30 ± 1.69**	**80.90 ± 1.62**	**80.80 ± 1.36**

We perform similar experiments on TADPOLE and ABIDE, where we investigate the expressive power of the `ENC` and `NEIGHs` modules separately. The results are summarised in Table 7. We note that here again the `NEIGHs` module suffers most from DP. DP seems to have a stronger impact on the aggregation module than on the encoder and classification modules, even though all modules receive the same Gaussian mechanism and the same noise scale. We assume the results of these experiments to explain the difference in performance between node- and edge-level DP. Edge-level DP guarantees strictly weaker protection and therefore adds less noise, affecting the aggregation module less.

5 Discussion, Conclusion, and Future Work

In this work, we investigate the practicality and challenges of differential private (DP) graph neural networks (GNNs) applied to medical population graphs. Applying DP to GNNs requires a specialised

Table 6: GAP under node-level DP on the **synthetic dataset**. `ENC` uses only the MLP from the encoder, `NEIGHs` uses only the node features after neighbourhood aggregation, and `ORIG` the original setup. GNNs outperforming the MLP are **bold**.

Method	Hom.	Non-DP	DP $\varepsilon = 20$	DP $\varepsilon = 15$	DP $\varepsilon = 10$	DP $\varepsilon = 5$	DP $\varepsilon = 4$	DP $\varepsilon = 3$	DP $\varepsilon = 2$	DP $\varepsilon = 1$
MLP	-	84.00 ± 1.10	82.70 ± 0.51	82.80 ± 1.21	82.20 ± 0.75	80.70 ± 2.29	82.20 ± 1.29	81.00 ± 1.67	79.50 ± 3.22	74.90 ± 3.57
`ENC`	-	83.00 ± 1.45	82.30 ± 2.25	81.30 ± 1.89	81.60 ± 2.01	80.40 ± 2.54	79.10 ± 2.56	76.50 ± 1.82	74.70 ± 2.58	69.30 ± 3.20
NEIGHs	0.9	**98.50 ± 0.55**	68.50 ± 1.79	63.10 ± 1.43	58.60 ± 2.80	52.00 ± 3.94	51.00 ± 4.17	52.50 ± 3.19	50.00 ± 2.26	49.60 ± 2.87
	0.8	**94.90 ± 1.39**	63.40 ± 2.40	60.00 ± 3.33	54.70 ± 2.99	51.10 ± 4.82	50.10 ± 3.72	52.20 ± 3.44	49.80 ± 2.54	49.60 ± 2.87
	0.7	**85.90 ± 1.46**	58.10 ± 1.93	54.30 ± 1.36	51.80 ± 2.40	50.10 ± 3.64	51.40 ± 3.60	51.50 ± 2.92	51.30 ± 2.69	49.70 ± 2.71
	0.6	66.90 ± 1.53	52.70 ± 3.74	52.00 ± 2.74	50.30 ± 1.86	50.50 ± 2.97	51.20 ± 3.36	51.20 ± 3.17	50.50 ± 2.10	49.50 ± 2.51
	0.5	48.20 ± 2.57	50.50 ± 2.93	49.90 ± 2.24	49.40 ± 2.40	50.20 ± 3.20	50.10 ± 2.71	51.30 ± 2.94	50.20 ± 2.34	49.30 ± 2.79
ORIG	0.9	**99.10 ± 0.20**	**84.40 ± 1.39**	82.30 ± 2.84	81.80 ± 3.12	80.00 ± 1.67	79.40 ± 1.20	78.20 ± 1.12	74.10 ± 1.16	66.70 ± 2.29
	0.8	**96.00 ± 0.71**	**83.40 ± 1.24**	82.20 ± 3.22	81.90 ± 3.15	80.00 ± 1.67	79.20 ± 1.29	78.10 ± 1.11	73.40 ± 2.31	66.60 ± 2.58
	0.7	**90.30 ± 1.96**	82.60 ± 1.53	83.10 ± 2.27	81.20 ± 2.68	80.30 ± 1.86	78.90 ± 1.28	78.30 ± 1.12	74.20 ± 1.50	66.50 ± 2.51
	0.6	83.90 ± 2.29	81.50 ± 1.05	82.30 ± 3.64	81.60 ± 2.71	80.20 ± 1.99	78.90 ± 1.20	78.00 ± 1.14	74.40 ± 1.56	66.70 ± 2.40
	0.5	81.60 ± 1.50	82.00 ± 1.00	82.20 ± 3.40	81.00 ± 2.47	80.40 ± 1.85	78.90 ± 1.20	78.10 ± 1.11	73.80 ± 2.60	66.50 ± 2.59

Table 7: Results of GAP under node-level DP with occlusions. `ENC` uses only the MLP, `NEIGHs` uses only the node features after neighbourhood aggregation, and `ORIG` the original GAP architecture.

Data	Method	Non-DP	DP $\varepsilon = 20$	DP $\varepsilon = 15$	DP $\varepsilon = 10$	DP $\varepsilon = 5$
TADPOLE	MLP	79.84 ± 1.52	77.25 ± 1.49	77.96 ± 1.23	76.94 ± 1.58	77.41 ± 2.02
	`ENC`	81.73 ± 1.50	76.71 ± 0.91	75.14 ± 0.73	75.76 ± 0.94	74.75 ± 1.66
	NEIGHs	75.37 ± 1.06	42.12 ± 1.93	38.67 ± 1.23	36.94 ± 1.67	36.24 ± 0.81
	ORIG	78.82 ± 2.66	75.45 ± 0.88	75.22 ± 1.32	75.06 ± 1.13	75.76 ± 0.67
ABIDE	MLP	71.09 ± 3.50	71.54 ± 4.27	72.00 ± 2.29	71.31 ± 1.46	67.43 ± 2.98
	`ENC`	71.77 ± 1.93	70.86 ± 1.25	69.37 ± 0.58	68.11 ± 2.15	62.86 ± 1.70
	NEIGHs	53.14 ± 2.40	54.06 ± 0.86	52.69 ± 1.32	52.47 ± 1.82	53.37 ± 1.00
	ORIG	72.11 ± 1.17	71.20 ± 1.83	69.14 ± 1.88	68.11 ± 2.27	62.06 ± 2.41

application of DP concepts like privacy amplification techniques and DP-SGD methods (17). We evaluate privacy-utility trade-offs of DP GNNs trained on medical population graphs of two state-of-the-art graph deep learning methods with DP: (1) DP-GCNs by Daigavane et al. (17) and (2) DP GNNs with aggregation perturbation (GAP) (19). While the first method adapts GCNs (18) to graph learning with DP, GAP designs a new model architecture. We investigate these methodological differences and their implications on population graph studies. On all utilised datasets, GAP outperformed DP-GCNs, from which we conclude that GAP seems to be a more suitable method for DP training on population graphs. This might be due to the comparably low homophily of the generated population graphs, which shows less impact on GAP than on DP-GCNs. GAP also performs more consistently across different homophily values. We attribute this to the architecture of GAP, which has three different modules that allow for information passing through the model without using the neighbourhood aggregation. We further investigate this correlation between graph structure and model performance and find that models trained on graphs with low homophily (indicating noisy neighbourhoods with different labels) are more impacted by DP than models trained on high homophily graphs when using DP-GCNs. This finding and its possible connection to the long-tail hypothesis (37) is a promising direction for future work to potentially improve DP methods for GNNs by improving the underlying graph structure. Even though our current experiments are on medical population graphs, we believe our results to be more generally applicable to DP graph deep learning for node classification tasks.

We note that there are other methods to perform node-level predictions with GNNs under DP, which are interesting directions for future work. One example are private aggregation of teacher ensemble (PATE) (38) methods, which have also been applied for graph learning (39). However, these methods require a suitable public dataset, which is difficult to obtain in medical settings. Recently, another method for DP training of GNNs has been introduced (40). The authors propose new decoupled graph convolutions and show strong performance on low-homophily graphs. In addition, homophily is not the only measure for the "quality" of a graph structure. Further metrics such as cross-class neighbourhood similarity (41), neighbourhood entropy (42), or the normalised smoothness value (11) should be evaluated, which may shed more light on the impact of graph structure on the performance of differntially private graph neural networks.

References

[1] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: going beyond euclidean data," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, 2017.

[2] S. Parisot, S. I. Ktena, E. Ferrante, M. Lee, R. G. Moreno, B. Glocker, and D. Rueckert, "Spectral graph convolutions for population-based disease prediction," in *International conference on medical image computing and computer-assisted intervention*, pp. 177–185, Springer, 2017.

[3] L. Cosmo, A. Kazi, S.-A. Ahmadi, N. Navab, and M. Bronstein, "Latent-graph learning for disease prediction," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 643–653, Springer, 2020.

[4] A. Kazi, S. Shekarforoush, S. Arvind Krishna, H. Burwinkel, G. Vivar, K. Kortüm, S.-A. Ahmadi, S. Albarqouni, and N. Navab, "Inceptiongcn: receptive field aware graph convolutional network for disease prediction," in *Information Processing in Medical Imaging: 26th International Conference, IPMI 2019, Hong Kong, China, June 2–7, 2019, Proceedings 26*, pp. 73–85, Springer, 2019.

[5] A. Kazi, L. Cosmo, S.-A. Ahmadi, N. Navab, and M. M. Bronstein, "Differentiable graph module (dgm) for graph convolutional networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 2, pp. 1606–1617, 2022.

[6] L. Chen, T. Hatsukami, J.-N. Hwang, and C. Yuan, "Automated intracranial artery labeling using a graph neural network and hierarchical refinement," in *Medical Image Computing and Computer Assisted Intervention–MICCAI 2020: 23rd International Conference, Lima, Peru, October 4–8, 2020, Proceedings, Part VI 23*, pp. 76–85, Springer, 2020.

[7] J. C. Paetzold, J. McGinnis, S. Shit, I. Ezhov, P. Büschl, C. Prabhakar, M. I. Todorov, A. Sekuboyina, G. Kaissis, A. Ertürk, *et al.*, "Whole brain vessel graphs: a dataset and benchmark for graph learning and neuroscience (vesselgraph)," *arXiv preprint arXiv:2108.13233*, 2021.

[8] J. M. Wolterink, T. Leiner, and I. Išgum, "Graph convolutional networks for coronary artery segmentation in cardiac ct angiography," in *Graph Learning in Medical Imaging: First International Workshop, GLMI 2019, Held in Conjunction with MICCAI 2019, Shenzhen, China, October 17, 2019, Proceedings 1*, pp. 62–69, Springer, 2019.

[9] Z. Ye, Y. J. Kumar, G. O. Sing, F. Song, and J. Wang, "A comprehensive survey of graph neural networks for knowledge graphs," *IEEE Access*, vol. 10, pp. 75729–75741, 2022.

[10] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *Acm Trans. On Graphics (tog)*, vol. 38, no. 5, pp. 1–12, 2019.

[11] S. Luan, C. Hua, Q. Lu, J. Zhu, X.-W. Chang, and D. Precup, "When do we need gnn for node classification?," *arXiv:2210.16979*, 2022.

[12] J. Zhu, Y. Yan, L. Zhao, M. Heimann, L. Akoglu, and D. Koutra, "Beyond homophily in graph neural networks: Current limitations and effective designs," *Advances in Neural Information Processing Systems*, vol. 33, pp. 7793–7804, 2020.

[13] H. Pei, B. Wei, K. C.-C. Chang, Y. Lei, and B. Yang, "Geom-gcn: Geometric graph convolutional networks," *arXiv preprint arXiv:2002.05287*, 2020.

[14] G. A. Kaissis, M. R. Makowski, D. Rückert, and R. F. Braren, "Secure, privacy-preserving and federated machine learning in medical imaging," *Nature Machine Intelligence*, vol. 2, no. 6, pp. 305–311, 2020.

[15] C. Dwork, A. Roth, *et al.*, "The algorithmic foundations of differential privacy.," *Found. Trends Theor. Comput. Sci.*, vol. 9, no. 3-4, pp. 211–407, 2014.

[16] T. T. Mueller, D. Usynin, J. C. Paetzold, D. Rueckert, and G. Kaissis, "Differential privacy guarantees for analytics and machine learning on graphs: A survey of results," *Journal of Privacy and Confidentiality (Accepted)*, 2023.

[17] 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.

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

[19] S. Sajadmanesh, A. S. Shamsabadi, A. Bellet, and D. Gatica-Perez, "Gap: Differentially private graph neural networks with aggregation perturbation," in *USENIX Security 2023-32nd USENIX Security Symposium*, 2023.

[20] Z. Xiang, T. Wang, and D. Wang, "Preserving node-level privacy in graph neural networks," *arXiv preprint arXiv:2311.06888*, 2023.

[21] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pp. 308–318, 2016.

[22] T. T. Mueller, D. Usynin, J. C. Paetzold, D. Rueckert, and G. Kaissis, "Sok: Differential privacy on graph-structured data," *arXiv preprint arXiv:2203.09205*, 2022.

[23] B. Wu, X. Yang, S. Pan, and X. Yuan, "Adapting membership inference attacks to gnn for graph classification: approaches and implications," in *2021 IEEE International Conference on Data Mining (ICDM)*, pp. 1421–1426, IEEE, 2021.

[24] X. He, R. Wen, Y. Wu, M. Backes, Y. Shen, and Y. Zhang, "Node-level membership inference attacks against graph neural networks," *arXiv preprint arXiv:2102.05429*, 2021.

[25] 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*, pp. 76–85, 2020.

[26] 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)*, pp. 11–20, IEEE, 2021.

[27] F. Wu, Y. Long, C. Zhang, and B. Li, "Linkteller: Recovering private edges from graph neural networks via influence analysis," in *2022 IEEE Symposium on Security and Privacy (SP)*, pp. 2005–2024, IEEE, 2022.

[28] X. He, J. Jia, M. Backes, N. Z. Gong, and Y. Zhang, "Stealing links from graph neural networks.," in *USENIX Security Symposium*, pp. 2669–2686, 2021.

[29] Z. Zhang, M. Chen, M. Backes, Y. Shen, and Y. Zhang, "Inference attacks against graph neural networks," in *31st USENIX Security Symposium (USENIX Security 22)*, pp. 4543–4560, 2022.

[30] Z. Zhang, Q. Liu, Z. Huang, H. Wang, C. Lu, C. Liu, and E. Chen, "Graphmi: Extracting private graph data from graph neural networks," *arXiv preprint arXiv:2106.02820*, 2021.

[31] N. Carlini, S. Chien, M. Nasr, S. Song, A. Terzis, and F. Tramèr, "Membership inference attacks from first principles," *2022 IEEE Symposium on Security and Privacy (SP)*, pp. 1897–1914, 2021.

[32] A. Di Martino, C.-G. Yan, Q. Li, E. Denio, F. X. Castellanos, K. Alaerts, J. S. Anderson, M. Assaf, S. Y. Bookheimer, M. Dapretto, *et al.*, "The autism brain imaging data exchange: towards a large-scale evaluation of the intrinsic brain architecture in autism," *Molecular psychiatry*, vol. 19, no. 6, pp. 659–667, 2014.

[33] S. Lu, Z. Zhu, J. M. Gorriz, S.-H. Wang, and Y.-D. Zhang, "NAGNN: classification of covid-19 based on neighboring aware representation from deep graph neural network," *Int. Journal of Intelligent Systems*, vol. 37, no. 2, pp. 1572–1598, 2022.

[34] C. Liu, S. Chakraborty, and P. Mittal, "Dependence makes you vulnberable: Differential privacy under dependent tuples.," in *NDSS*, vol. 16, pp. 21–24, 2016.

[35] J. Dong, A. Roth, and W. Su, "Gaussian differential privacy," *Journal of the Royal Statistical Society*, 2021.

[36] M. Nasr, S. Songi, A. Thakurta, N. Papernot, and N. Carlin, "Adversary instantiation: Lower bounds for differentially private machine learning," in *2021 IEEE Symposium on Security and Privacy (SP)*, pp. 866–882, 2021.

[37] V. Feldman, "Does learning require memorization? a short tale about a long tail," in *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pp. 954–959, 2020.

[38] N. Papernot, M. Abadi, U. Erlingsson, I. Goodfellow, and K. Talwar, "Semi-supervised knowledge transfer for deep learning from private training data," *arXiv preprint arXiv:1610.05755*, 2016.

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

[40] E. Chien, W.-N. Chen, C. Pan, P. Li, A. Özgür, and O. Milenkovic, "Differentially private decoupled graph convolutions for multigranular topology protection," *arXiv preprint arXiv:2307.06422*, 2023.

[41] Y. Ma, X. Liu, N. Shah, and J. Tang, "Is homophily a necessity for graph neural networks?," *arXiv preprint arXiv:2106.06134*, 2021.

[42] Y. Xie, S. Li, C. Yang, R. C.-W. Wong, and J. Han, "When do gnns work: Understanding and improving neighborhood aggregation," in *IJCAI'20: Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence,{IJCAI} 2020*, vol. 2020, 2020.

A Additional Information on the Datasets

A summary of the different homophily values of all datasets, the utilised δ values, as well as the k used for graph construction are summarised in Table 8.

Table 8: Summary of the homophily of the utilised graph structures as well as the δ and k values for all datasets.

Dataset	**Nr. nodes**	**Homophily**	δ	k
TADPOLE	1 277	0.7392	$1.31 \cdot 10^{-4}$	5
COVID	65	0.7569	$2.78 \cdot 10^{-3}$	5
ABIDE	871	0.6009	$1.92 \cdot 10^{-4}$	5
Synthetic	1 000	varying	$1.79 \cdot 10^{-4}$	10

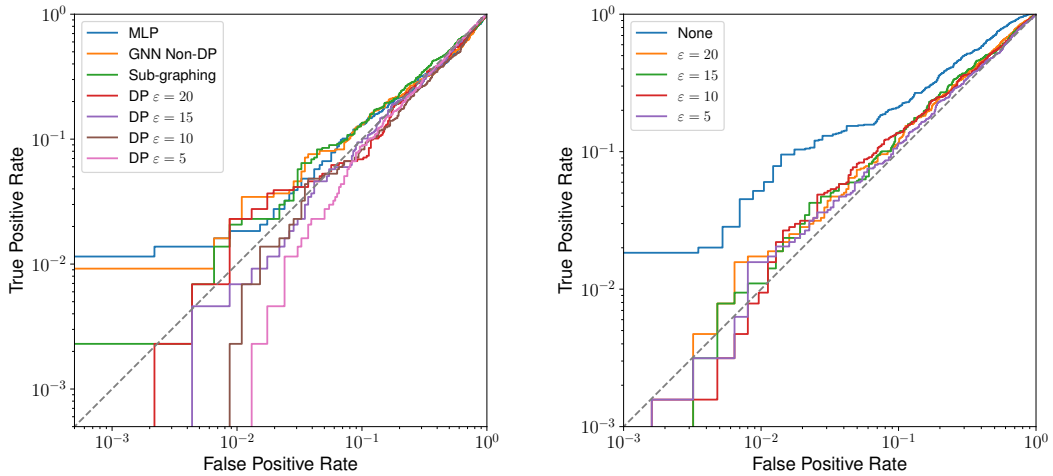
B Additional Experiments and Results

We perform some additional experiments to further investigate privacy-utility trade-offs of DP population graphs in the medical domain. We, for example, investigate the impact of different number of hops on the performance of GAP on the datasets TADPOLE, ABIDE, and COVID. The results are summarised in Table 9.

Table 9: GAP results across different number of hops on the dataset TADPOLE, ABIDE, and COVID at different privacy budgets.

	Hops	**Non-DP**	$\varepsilon = 20$	$\varepsilon = 15$	$\varepsilon = 10$	$\varepsilon = 5$	$\varepsilon = 4$	$\varepsilon = 3$	$\varepsilon = 2$	$\varepsilon = 1$
TADPOLE	1	79.84 ± 0.91	74.98 ± 2.47	74.12 ± 2.69	76.00 ± 0.97	74.67 ± 0.91	74.43 ± 0.63	74.51 ± 0.50	71.14 ± 2.01	67.37 ± 1.37
	2	78.82 ± 2.66	75.45 ± 0.88	75.22 ± 1.32	75.06 ± 1.13	75.76 ± 0.67	75.29 ± 0.66	73.18 ± 1.90	69.80 ± 1.79	66.43 ± 1.13
	3	79.45 ± 2.17	76.78 ± 0.72	76.55 ± 1.52	75.92 ± 1.01	75.37 ± 1.78	74.59 ± 2.04	72.16 ± 1.66	69.57 ± 1.13	66.59 ± 2.28
ABIDE	1	72.69 ± 1.32	70.74 ± 4.00	70.17 ± 2.12	68.69 ± 3.32	62.86 ± 2.96	61.37 ± 2.25	61.03 ± 2.54	57.14 ± 3.49	53.03 ± 4.05
	2	72.11 ± 1.17	71.20 ± 1.83	69.14 ± 1.88	68.11 ± 2.27	62.06 ± 2.41	61.94 ± 2.67	59.77 ± 3.52	57.71 ± 1.62	53.26 ± 1.89
	3	72.46 ± 1.89	72.00 ± 2.82	68.91 ± 2.95	67.77 ± 3.08	63.43 ± 3.00	62.86 ± 2.50	60.91 ± 3.24	58.17 ± 3.64	52.80 ± 1.06
COVID	1	60.00 ± 11.31	52.31 ± 11.31	53.85 ± 12.87	47.69 ± 8.97	47.69 ± 8.97	49.23 ± 13.41	49.23 ± 7.84	46.15 ± 10.88	56.92 ± 6.15
	2	61.54 ± 6.88	66.15 ± 11.51	64.62 ± 11.51	64.62 ± 10.43	61.54 ± 9.73	64.62 ± 10.43	66.15 ± 12.50	61.54 ± 14.60	56.92 ± 14.27
	3	63.08 ± 12.31	58.46 ± 15.07	58.46 ± 12.50	58.46 ± 15.84	44.62 ± 22.51	58.46 ± 7.84	53.85 ± 10.88	52.31 ± 12.31	52.31 ± 16.43

In Figure 4, we visualise the log-scale receiver operating characteristic (ROC) curve of the MIA attacks performed on DP-GCN and GAP on the TADPOLE dataset.



(a) DP-GCN (b) GAP

Figure 4: Empirical ROC curves for MIA on the TADPOLE dataset.

The GAP method consists of three modules: (1) the encoder module, (2) the aggregation module, and (3) the classification module. In order to investigate the impact of the aggregation module on the overall performance of GAP, we occlude parts of the GAP model and evaluate the individual

performances. We split the classification module into three versions: (a) using the original architecture (ORIG), (b) only using the MLP from the encoder module (ENC), and (c) using the MLPs from the neighbourhood aggregation (NEIGHs). We here summarise some additional experiments using this setup. Table 10 shows the performance of the individual modules with at different homophily values on the synthetic dataset, using two different maximum degrees: 10 and 200. This degree refers to both the k for graph construction and the bounding degree for sub-graphing. It is noteworthy that the NEGHs module performs better at a high maximum degree than on a low maximum degree. We furthermore summarise similar experiments at a fixed homophily of 0.9 on the synthetic dataset with several different maximum degrees in Table 11, which shows the same trend that the performance of the NEIGHs module performs better with an increasing maximum degree.

Table 10: Breakdown of ENC and NEIGHs modules of the classifier module of GAP under node-level DP along with the homophily value on the synthetic dataset.

Max degree	Method	Hom.	Non-DP	DP $\varepsilon = 20$	DP $\varepsilon = 15$	DP $\varepsilon = 10$	DP $\varepsilon = 5$	DP $\varepsilon = 4$	DP $\varepsilon = 3$	DP $\varepsilon = 2$	DP $\varepsilon = 1$
	MLP	-	84.00 ± 1.10	82.70 ± 0.51	82.80 ± 1.21	82.20 ± 0.75	80.70 ± 2.29	82.20 ± 1.29	81.00 ± 1.67	79.50 ± 3.22	74.90 ± 3.57
10	ENC	-	83.00 ± 1.45	82.30 ± 2.25	81.30 ± 1.89	81.60 ± 2.01	80.40 ± 2.54	79.10 ± 2.56	76.50 ± 1.82	74.70 ± 2.58	69.30 ± 3.20
	NEIGHs	0.9	98.50 ± 0.55	68.50 ± 1.79	63.10 ± 1.43	58.60 ± 2.80	52.00 ± 3.94	51.00 ± 4.17	52.50 ± 3.19	50.00 ± 2.26	49.60 ± 2.87
		0.8	94.90 ± 1.39	63.40 ± 2.40	60.00 ± 3.33	54.70 ± 2.99	51.10 ± 4.82	50.10 ± 3.72	52.20 ± 3.44	49.80 ± 2.54	49.60 ± 2.87
		0.7	85.90 ± 1.46	58.10 ± 1.93	54.30 ± 1.36	51.80 ± 2.40	50.10 ± 3.64	51.40 ± 3.60	51.50 ± 2.92	51.30 ± 2.69	49.70 ± 2.71
		0.6	66.90 ± 1.53	52.70 ± 3.74	52.00 ± 2.74	50.30 ± 1.86	50.50 ± 2.97	51.20 ± 3.36	51.20 ± 3.17	50.50 ± 2.10	49.50 ± 2.51
		0.5	48.20 ± 2.57	50.50 ± 2.93	49.90 ± 2.24	49.40 ± 2.40	50.20 ± 3.20	50.10 ± 2.71	51.30 ± 2.94	50.20 ± 2.34	49.30 ± 2.79
	ORIG	0.9	99.10 ± 0.20	84.40 ± 1.39	82.30 ± 2.84	81.80 ± 3.12	80.00 ± 1.67	79.40 ± 1.20	78.20 ± 1.12	74.10 ± 1.16	66.70 ± 2.29
		0.8	96.00 ± 0.71	83.40 ± 1.24	82.20 ± 3.22	81.90 ± 3.15	80.00 ± 1.67	79.20 ± 1.29	78.10 ± 1.11	73.40 ± 2.31	66.60 ± 2.58
		0.7	90.30 ± 1.96	82.60 ± 1.53	83.10 ± 2.27	81.20 ± 2.68	80.30 ± 1.86	78.90 ± 1.28	78.30 ± 1.12	74.20 ± 1.50	66.50 ± 2.51
		0.6	83.90 ± 2.29	81.50 ± 1.05	82.30 ± 3.64	81.60 ± 2.71	80.20 ± 1.99	78.90 ± 1.20	78.00 ± 1.14	74.40 ± 1.56	66.70 ± 2.40
		0.5	81.60 ± 1.50	82.00 ± 1.00	82.20 ± 3.40	81.00 ± 2.47	80.40 ± 1.85	78.90 ± 1.20	78.10 ± 1.11	73.80 ± 2.60	66.50 ± 2.59
200	ENC	-	83.00 ± 1.45	83.10 ± 2.27	82.40 ± 2.20	80.90 ± 3.01	81.00 ± 2.97	80.80 ± 2.25	78.80 ± 2.48	75.70 ± 1.81	69.90 ± 1.62
	NEIGHs	0.9	100.00 ± 0.00	96.70 ± 1.75	91.60 ± 1.24	84.50 ± 2.07	65.30 ± 2.08	59.60 ± 2.11	54.70 ± 4.18	50.50 ± 2.30	50.90 ± 2.22
		0.8	100.00 ± 0.00	91.10 ± 2.30	84.90 ± 2.48	75.50 ± 3.32	60.80 ± 5.16	53.80 ± 2.86	51.80 ± 1.91	51.30 ± 2.18	51.20 ± 1.78
		0.7	99.80 ± 0.40	79.40 ± 2.40	72.50 ± 3.59	64.60 ± 3.75	54.00 ± 2.19	51.90 ± 2.89	50.60 ± 3.53	51.80 ± 2.69	50.80 ± 1.86
		0.6	98.70 ± 1.08	63.50 ± 3.52	57.10 ± 2.87	53.70 ± 2.06	50.80 ± 3.12	50.80 ± 2.42	51.10 ± 3.01	51.20 ± 2.06	51.80 ± 1.50
		0.5	49.30 ± 4.27	48.90 ± 2.78	49.50 ± 3.18	50.30 ± 3.04	49.30 ± 2.91	50.30 ± 4.06	49.60 ± 3.17	51.20 ± 2.34	51.60 ± 1.74
	ORIG	0.9	99.90 ± 0.20	98.20 ± 0.68	94.80 ± 1.44	87.00 ± 3.13	80.40 ± 2.11	78.50 ± 1.34	79.20 ± 1.57	76.30 ± 2.62	67.50 ± 5.63
		0.8	100.00 ± 0.00	94.00 ± 1.00	89.20 ± 1.81	83.90 ± 3.02	80.70 ± 2.38	78.60 ± 1.16	79.50 ± 1.48	76.20 ± 2.68	66.90 ± 5.72
		0.7	99.70 ± 0.40	86.20 ± 1.96	84.40 ± 2.03	82.40 ± 2.56	80.60 ± 2.48	78.60 ± 1.24	79.60 ± 1.32	78.60 ± 2.65	67.50 ± 6.16
		0.6	98.80 ± 0.51	82.90 ± 1.16	82.20 ± 1.57	81.20 ± 1.86	80.30 ± 2.11	79.00 ± 1.41	79.60 ± 2.15	76.30 ± 2.44	67.20 ± 5.60
		0.5	80.90 ± 2.09	82.50 ± 1.30	82.90 ± 1.56	81.80 ± 2.11	80.80 ± 1.94	79.00 ± 1.38	79.20 ± 1.69	76.10 ± 2.75	67.50 ± 5.98

Table 11: Results of node-level DP with GAP for graph structures with different density values and a separate investigation of ENC and NEIGHs modules of GAP at a homophily of 0.9. ENC uses only the MLP without neighbourhood aggregation, NEIGHs uses only the node features after neighbourhood aggregation and ORIG uses the original setup of GAP (using both ENC and NEIGHs).

Max Degree	Method	Non-DP	DP $\varepsilon = 20$	DP $\varepsilon = 15$	DP $\varepsilon = 10$	DP $\varepsilon = 5$	DP $\varepsilon = 4$	DP $\varepsilon = 3$	DP $\varepsilon = 2$	DP $\varepsilon = 1$
	ENC	83.00 ± 1.45	82.30 ± 2.25	81.30 ± 1.89	81.60 ± 2.01	80.40 ± 2.54	79.10 ± 2.56	76.50 ± 1.82	74.70 ± 2.58	69.30 ± 3.20
10	NEIGHs	98.50 ± 0.55	68.50 ± 1.79	63.10 ± 1.43	58.60 ± 2.80	52.00 ± 3.94	51.80 ± 4.17	52.50 ± 3.19	50.00 ± 2.26	49.60 ± 2.87
	ORIG	99.10 ± 0.20	84.40 ± 1.39	82.30 ± 2.84	81.80 ± 3.12	80.00 ± 1.67	79.40 ± 1.20	78.20 ± 1.12	74.10 ± 1.16	66.70 ± 2.29
20	NEIGHs	100.00 ± 0.00	75.30 ± 2.99	68.90 ± 2.35	63.20 ± 2.71	55.40 ± 3.77	52.10 ± 3.99	51.80 ± 2.48	51.10 ± 2.89	50.60 ± 1.66
	ORIG	98.80 ± 0.24	86.90 ± 1.59	83.60 ± 2.42	81.90 ± 3.09	80.60 ± 1.39	79.30 ± 0.98	78.10 ± 1.11	74.40 ± 1.36	66.70 ± 2.54
30	NEIGHs	99.90 ± 0.20	81.70 ± 2.01	73.90 ± 2.73	67.80 ± 2.29	57.50 ± 3.03	54.00 ± 4.06	51.80 ± 3.56	51.10 ± 3.06	49.70 ± 2.77
	ORIG	99.90 ± 0.20	89.50 ± 2.28	85.30 ± 1.63	81.30 ± 2.84	80.30 ± 1.54	79.10 ± 1.16	78.00 ± 1.26	74.30 ± 1.44	65.90 ± 2.71
40	NEIGHs	99.80 ± 0.24	85.90 ± 1.16	78.20 ± 1.44	70.80 ± 2.50	59.30 ± 3.06	55.80 ± 3.04	53.70 ± 4.51	51.30 ± 3.03	50.60 ± 1.77
	ORIG	99.90 ± 0.20	92.50 ± 2.70	87.60 ± 2.52	82.50 ± 1.48	80.30 ± 1.94	79.20 ± 1.25	78.20 ± 0.98	74.20 ± 1.47	66.40 ± 2.67
50	NEIGHs	100.0 ± 0.00	89.50 ± 1.87	80.70 ± 3.39	73.00 ± 2.17	61.20 ± 3.14	57.70 ± 3.20	55.00 ± 3.92	52.50 ± 2.07	49.40 ± 2.65
	ORIG	100.0 ± 0.00	94.00 ± 2.21	88.70 ± 3.30	83.10 ± 1.53	80.30 ± 1.94	79.50 ± 1.22	77.90 ± 1.11	74.20 ± 1.44	65.40 ± 2.63
60	NEIGHs	100.00 ± 0.00	91.70 ± 2,58	84.70 ± 2.87	74.70 ± 3.04	62.00 ± 2.28	58.50 ± 3.61	55.30 ± 3.37	52.60 ± 2.46	48.80 ± 1.69
	ORIG	100.00 ± 0.00	94.80 ± 2.06	91.00 ± 2.51	84.80 ± 3.04	80.30 ± 1.75	79.60 ± 0.97	77.90 ± 1.11	74.30 ± 1.44	66.10 ± 2.42
70	NEIGHs	100.00 ± 0.00	94.30 ± 1.29	86.60 ± 2.46	76.90 ± 2.37	62.60 ± 2.50	59.50 ± 2.66	55.00 ± 3.61	52.50 ± 2.02	48.70 ± 1.60
	ORIG	100.00 ± 0.00	96.80 ± 0.81	91.50 ± 2.49	85.50 ± 2.95	80.80 ± 1.57	78.90 ± 1.20	78.00 ± 1.05	74.20 ± 1.44	65.70 ± 2.42
80	NEIGHs	100.00 ± 0.00	95.10 ± 1.74	88.00 ± 2.74	78.70 ± 3.34	64.50 ± 2.49	59.90 ± 2.67	55.50 ± 3.94	52.90 ± 2.24	49.50 ± 2.77
	ORIG	100.00 ± 0.00	96.50 ± 1.00	93.20 ± 1.21	86.30 ± 2.77	80.50 ± 1.64	79.10 ± 1.02	77.70 ± 1.29	74.30 ± 1.44	65.70 ± 2.11
90	NEIGHs	100.00 ± 0.00	95.40 ± 1.16	90.20 ± 1.94	81.00 ± 1.64	65.50 ± 1.87	61.10 ± 2.85	57.70 ± 3.01	53.30 ± 2.14	49.10 ± 2.76
	ORIG	99.90 ± 0.2	97.70 ± 1.08	93.90 ± 1.85	87.70 ± 3.11	80.50 ± 1.64	79.60 ± 0.86	77.70 ± 1.29	74.30 ± 1.44	66.70 ± 3.59
100	NEIGHs	100 ± 0.00	97.50 ± 1.14	92.00 ± 1.92	82.10 ± 2.15	66.90 ± 1.85	61.40 ± 2.67	58.10 ± 2.31	53.10 ± 2.13	48.80 ± 2.23
	ORIG	99.80 ± 0.40	98.50 ± 0.84	95.20 ± 1.63	88.30 ± 3.06	80.50 ± 1.41	79.80 ± 0.93	77.90 ± 1.11	74.30 ± 1.44	66.50 ± 3.56
200	NEIGHs	100.00 ± 0.00	96.70 ± 1.75	91.60 ± 1.24	84.50 ± 2.07	65.30 ± 2.08	59.60 ± 2.11	54.70 ± 4.18	50.50 ± 2.30	50.90 ± 2.22
	ORIG	99.90 ± 0.20	98.20 ± 0.68	94.80 ± 1.44	87.00 ± 3.13	80.40 ± 2.11	78.50 ± 1.34	79.20 ± 1.57	76.30 ± 2.62	67.50 ± 5.63