Locally Private Graph Neural Networks

Sina Sajadmanesh
Daniel Gatica-Perez

Idiap Research Institute
Swiss Federal Institute of Technology (EPFL)

Twitter Machine Learning Seminar
Jan 7, 2021

Problem Definition

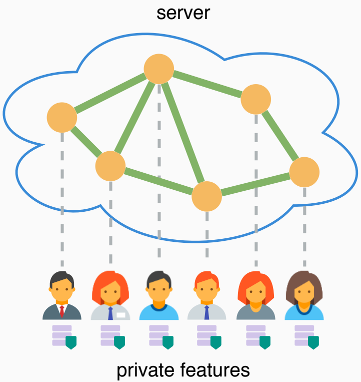
Privacy-Preserving GNN learning with node-level privacy

Setting:

- The server has access to a graph
- Each node has a private feature vector
- Node features are inaccessible by the server

Problem:

- How to learn a GNN without letting the private features leave the nodes?



Federated Learning

Can we use federated learning?

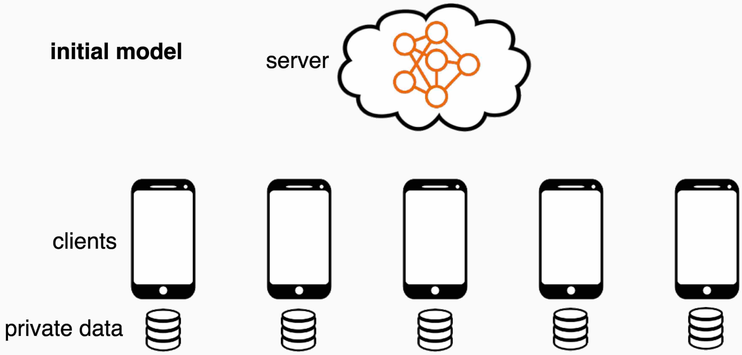


Image Source: "IEEE GLOBECOM 2020 Tutorial on Distributed Deep Learning" by Wojciech Samek and Deniz Gunduz.

Can we use federated learning?

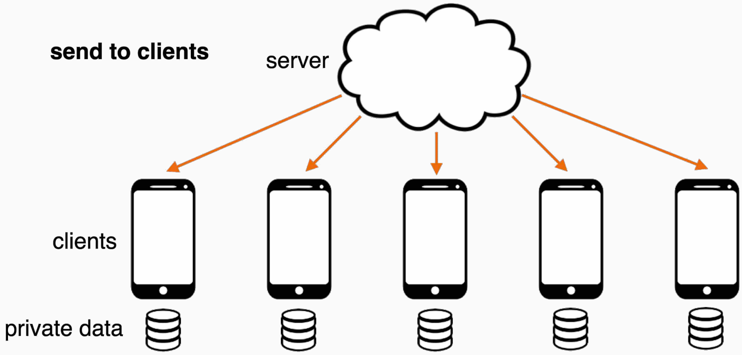


Image Source: “IEEE GLOBECOM 2020 Tutorial on Distributed Deep Learning” by Wojciech Samek and Deniz Gunduz.

Can we use federated learning?

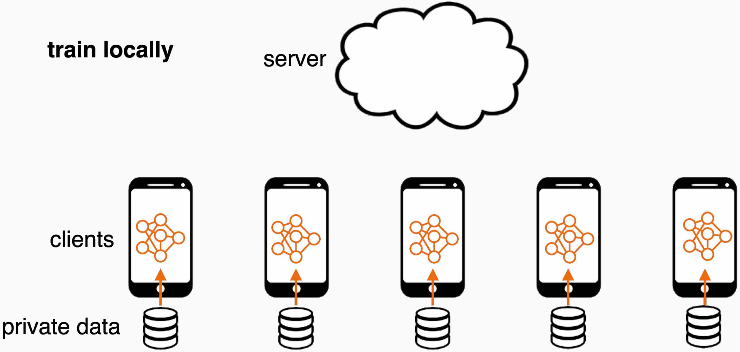


Image Source: "IEEE GLOBECOM 2020 Tutorial on Distributed Deep Learning" by Wojciech Samek and Deniz Gunduz.

Can we use federated learning?

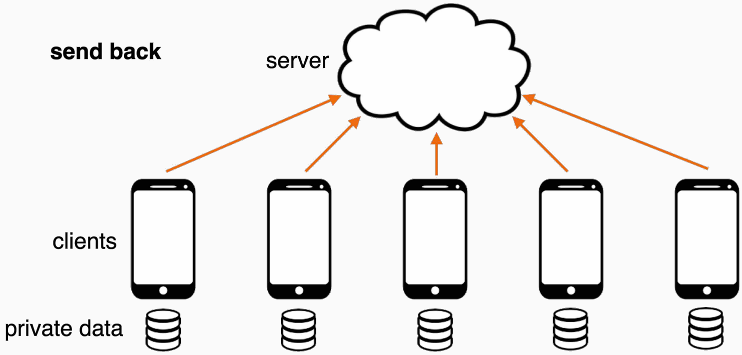


Image Source: "IEEE GLOBECOM 2020 Tutorial on Distributed Deep Learning" by Wojciech Samek and Deniz Gunduz.

Can we use federated learning?

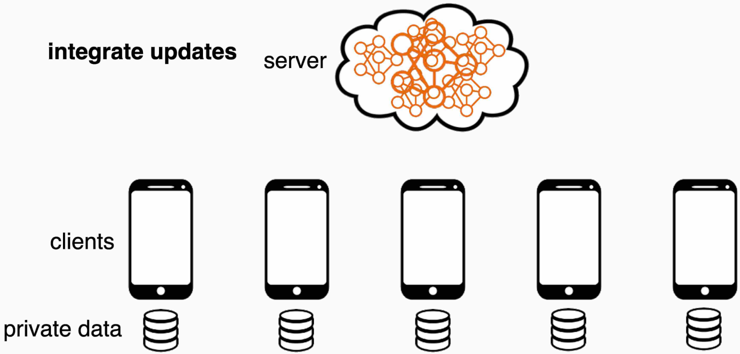
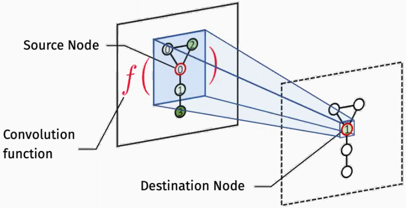


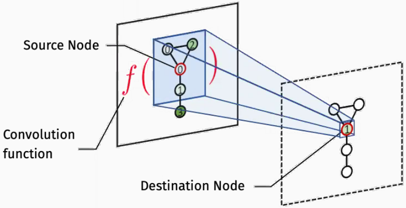
Image Source: "IEEE GLOBECOM 2020 Tutorial on Distributed Deep Learning" by Wojciech Samek and Deniz Gunduz.

Inside the GNN



Input: a representation vector $\mathbf{h}_v$ for each node v
(**initially node features**)
Output: a new representation vector $\mathbf{h}'_v$ for each node v

Image Source: “A gentle introduction to graph neural networks” by Andreas Loukas.



Input: a representation vector $\mathbf{h}_v$ for each node v (**initially node features**)

Output: a new representation vector $\mathbf{h}'_v$ for each node v

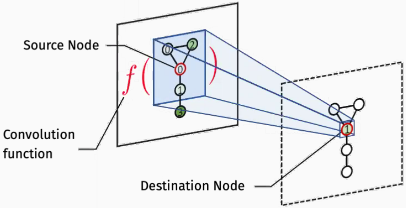
$$\mathbf{h}'_v = f\left(\{\mathbf{h}_u : u \in \mathcal{N}(v)\}\right)$$
$$= \text{UPDATE}\left(\text{AGGREGATE}\left(\{\mathbf{h}_u : u \in \mathcal{N}(v)\}\right)\right)$$

- UPDATE is a neural network (e.g., MLP)
- AGGREGATE is a permutation invariant function, e.g., sum, mean, max, or:

GCN: $SUM\left(\frac{\mathbf{h}_u}{\sqrt{|\mathcal{N}(u)|\cdot|\mathcal{N}(v)|}} : u \in \mathcal{N}(v)\right)$

GraphSAGE: $CONCAT\left(\mathbf{h}_v\,,\ MEAN\left(\{\mathbf{h}_u : u \in \mathcal{N}(v)\}\right)\right)$

Image Source: “A gentle introduction to graph neural networks” by Andreas Loukas.



Input: a representation vector $\mathbf{h}_v$ for each node v
(initially node features)
Output: a new representation vector $\mathbf{h}'_v$ for each node v

$$\mathbf{h}'_v = f\left(\{\mathbf{h}_u : u \in \mathcal{N}(v)\}\right)$$
$$= \text{UPDATE}\left(\text{AGGREGATE}\left(\{\mathbf{h}_u : u \in \mathcal{N}(v)\}\right)\right)$$

- UPDATE is a neural network (e.g., MLP)
- AGGREGATE is a permutation invariant function, e.g., sum, mean, max, or:

GCN: $SUM\left(\frac{\mathbf{h}_u}{\sqrt{|\mathcal{N}(u)|\cdot|\mathcal{N}(v)|}} : u \in \mathcal{N}(v)\right)$

GraphSAGE: $CONCAT\left(\mathbf{h}_v\ ,\ MEAN\left(\{\mathbf{h}_u : u \in \mathcal{N}(v)\}\right)\right)$

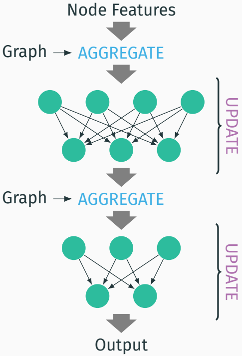
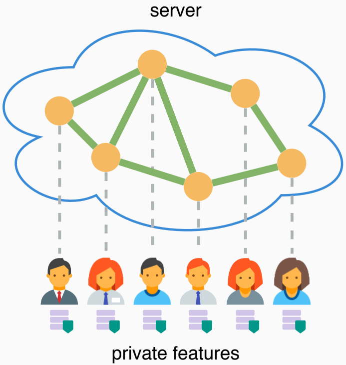


Image Source: "A gentle introduction to graph neural networks" by Andreas Loukas.

What's the problem with federated learning?

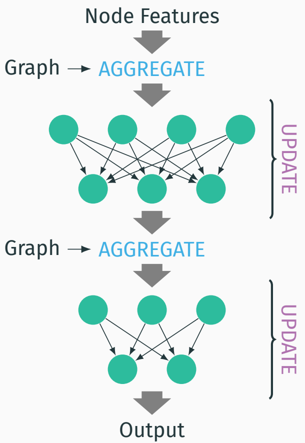
- AGGREGATE must be computed at **node side**
- Nodes require the private features of their neighbors
  - If sent in plain text → privacy violation!
  - If sent using SMC → massive communication!



Our Approach

Let's keep the model on the server

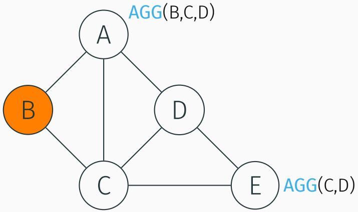
- Private node features are only needed in the first layer of the GNN
- We only need to compute the first layer's AGGREGATE function privately!

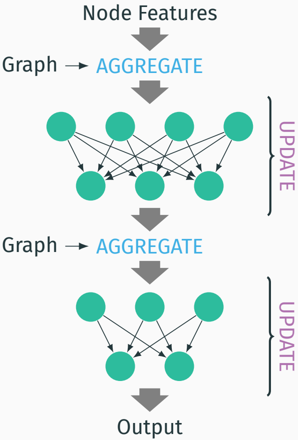


Our Approach

Let's keep the model on the server

- Private node features are only needed in the first layer of the GNN
- We only need to compute the first layer's AGGREGATE function privately!
- But an exact computation of AGGREGATE is vulnerable to differencing attack!



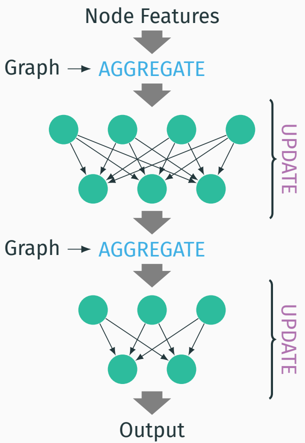


Our Approach

Let's keep the model on the server

- Private node features are only needed in the first layer of the GNN
- We only need to compute the first layer's AGGREGATE function privately!
- But an exact computation of AGGREGATE is vulnerable to differencing attack!

Idea: privately estimate the AGGREGATE function using Local Differential Privacy!



Local Differential Privacy

- An **untrusted data aggregator** wishes to compute an aggregate function over a **private dataset**
- Data holder i perturbs his data x_i using a **randomized mechanism** $\mathcal{M}$, returning $x_i' = \mathcal{M}(x_i)$ to the aggregator
- The aggregator computes the target statistic using an **estimator function**

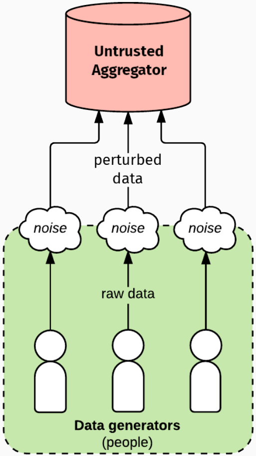
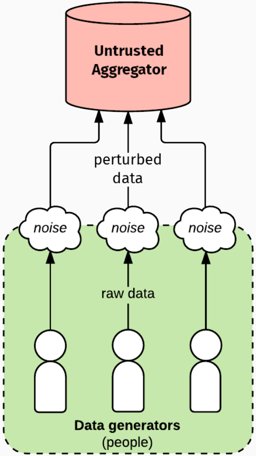


Image Credit: Bennett Cyphers

Local Differential Privacy

- An **untrusted data aggregator** wishes to compute an aggregate function over a **private dataset**
- Data holder i perturbs his data x_i using a **randomized mechanism** $\mathcal{M}$, returning $x_i' = \mathcal{M}(x_i)$ to the aggregator
- The aggregator computes the target statistic using an **estimator function**



Definition

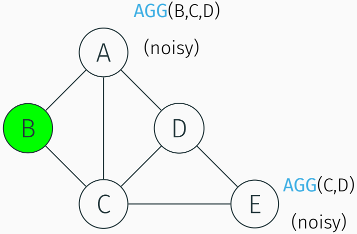
a randomized mechanism $\mathcal{M}$ satisfies ϵ-LDP if for all pairs of private data x_1 and x_2, and for all outputs x' of $\mathcal{M}$, we have:

$$\Pr[\mathcal{M}(x_1) = x'] \leq e^{\epsilon} \Pr[\mathcal{M}(x_2) = x']$$

Image Credit: Bennett Cyphers

Private neighborhood aggregation with LDP

- Node features are perturbed by **injecting noise**
  - The LDP mechanism should be **unbiased**, i.e., $\mathbb{E}[\mathcal{M}(x)] = x$
- The neighborhood aggregation will **cancel out** the injected noise
  - AGGREGATE should be a **weighted summation**, as in GCN, GIN, ...



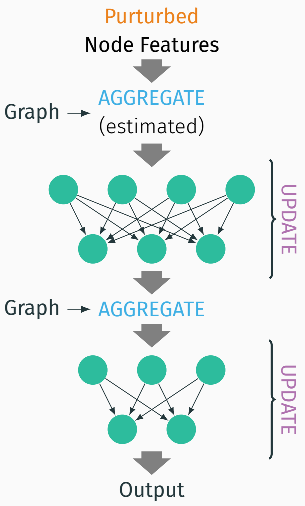
Locally Private GNN (LPGNN)

Node-Side:

1. Perturb the private feature vector $\mathbf{x}_i$ using the LDP mechanism $\mathbf{x}'_i = \mathcal{M}(\mathbf{x}_i)$ and send $\mathbf{x}'_i$ to the server

Server-Side:

1. Estimate the first layer's AGGREGATE function for every node using the perturbed feature vectors
2. Proceed with forward and backward propagation as usual
3. Return to step 2 if stopping criteria has not met



Challenge #1: High-dimensional features

- The total privacy budget of a node scales with the number of features
  - Give every single feature a small privacy budget → **Too much privacy leakage!**
  - Keep the total privacy budget of a node small → **Too much noise!**

Multi-bit mechanism: multidimensional feature perturbation

- Randomly **sample m/d features** without replacement
- Perturb each sampled feature with ϵ/m privacy budget using **1-bit mechanism**
- Map the output of the 1-bit mechanism to either -1 or 1
- For the rest of the features (not sampled) return 0

Theorem 3.1

The multi-bit mechanism **satisfies ϵ-LDP** for each node.

Proposition 3.5

The **optimal value** of the sampling parameter m in the multi-bit mechanism is:
$m^\star = \max(1, \min(d, \lfloor \frac{\epsilon}{2.18} \rfloor))$

Challenge #2: Small-size neighborhood

- Lots of the nodes have too few neighbors (Power-Law degree distribution)
- The neighborhood aggregator cannot cancel out the noise if the neighborhood size is small

KProp convolution layer: neighborhood expansion method

- Expands the neighborhood to the nodes that are up to K-hops away
- Applies K consecutive AGGREGATE function
- Applies the UPDATE function after the K-th AGGREGATE
- Trade-off between the aggregation estimation error and over-smoothing

Experiments: Settings

Learning Task

- Node Classification

Comparison methods

- GCN+Raw: A standard two-layer GCN trained on raw features (non-private)
- LPGNN: A two layer GNN (KProp as the first, GCN as the second layer) trained on perturbed features using the multi-bit mechanism (locally-private)
- GCN+Rnd: Similar to GCN+Raw, but trained on random features (fully-private)
- GCN+Ohd: Similar to GCN+Raw, but trained on "one-hot degree" features (fully-private)

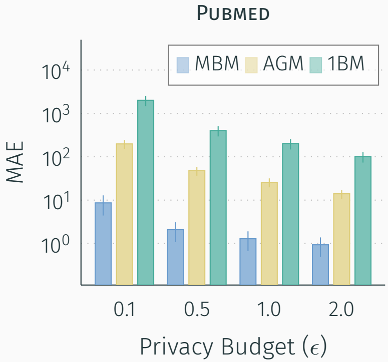
Experiments: Datasets

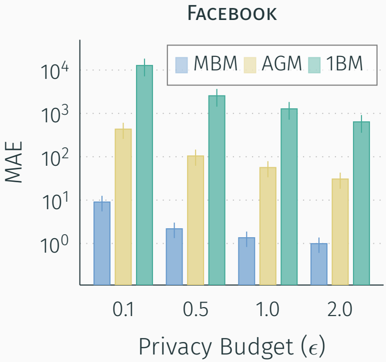
Dataset	#Classes	#Features	Avg. Degree
Cora	7	1,433	3.90
Citeseer	6	3,703	2.74
Pubmed	3	500	4.50
Facebook	4	4,714	15.21
Github	2	4,005	15.33
LastFM	18	7,842	7.29

Micro-F1 score of different methods for node classification

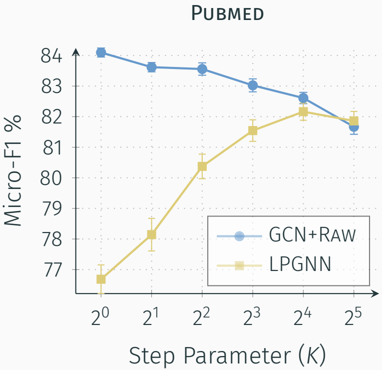
Dataset	GCN +Raw	LPGNN $\epsilon = 0.1$	LPGNN $\epsilon = 0.5$	LPGNN $\epsilon = 1.0$	LPGNN $\epsilon = 2.0$	GCN +Rnd	GCN +Ohd
Cora	85.0 ± 0.5	84.6 ± 0.5	84.6 ± 0.6	84.6 ± 0.6	84.6 ± 0.6	78.1 ± 1.3	58.4 ± 0.7
Citeseer	73.7 ± 0.5	68.6 ± 0.8	68.4 ± 0.7	68.6 ± 0.9	68.6 ± 0.8	58.3 ± 4.1	38.5 ± 0.9
Pubmed	87.0 ± 0.2	82.1 ± 0.2	82.2 ± 0.3	82.2 ± 0.3	82.2 ± 0.3	56.5 ± 2.2	62.4 ± 0.9
Facebook	94.8 ± 0.1	94.0 ± 0.1	94.0 ± 0.2	94.0 ± 0.2	94.0 ± 0.2	40.6 ± 1.2	79.2 ± 0.3
Github	86.7 ± 0.2	85.9 ± 0.1	85.9 ± 0.2	85.9 ± 0.2	85.9 ± 0.1	74.4 ± 0.1	84.0 ± 0.1
LastFM	87.7 ± 0.4	86.1 ± 0.3	86.1 ± 0.3	86.1 ± 0.2	86.1 ± 0.3	25.2 ± 7.1	70.6 ± 0.5

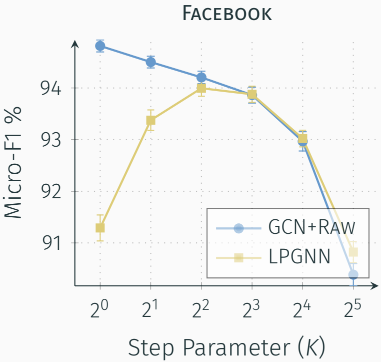
Average L1 distance between the true and estimated neighborhood aggregation obtained by the multi-bit (MBM), 1-bit (1BM), and the Analytic Gaussian (AGM) mechanisms



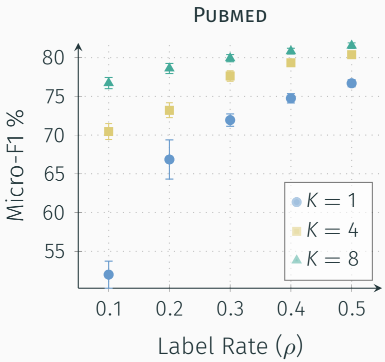


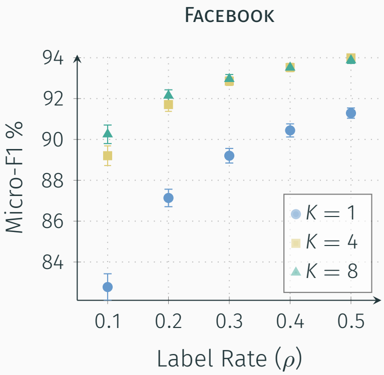
Effect of the KProp step parameter (K) on the performance of LPGNN ($\epsilon = 1$)





Effect of the label rate on the performance of LPGNN ($\epsilon = 1$)





Summary

- Proposed a privacy-preserving GNN based on local differential privacy
- Demonstrated promising results in terms of accuracy-privacy trade-off
- Works better on graphs with higher average degree

Future Work

- Protect privacy of graph topology
- Relationship to adversarial robustness
- Expressive power of private graph networks

Thank You!

Questions?

@sajadmanesh
sajadmanesh@idiap.ch
https://arxiv.org/pdf/2006.05535.pdf