Locally Private Graph Neural Networks

Sina Sajadmanesh
Daniel Gatica-Perez

AI4Media Workshop on Explainability, Robustness and Privacy in AI
June 2, 2021

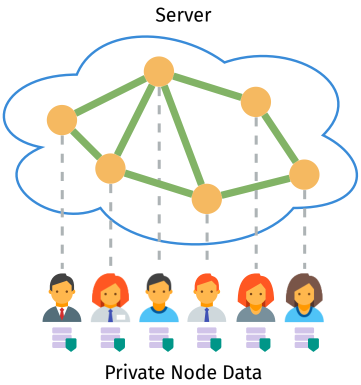
Graph learning with node data privacy

Setting:

- Graph topology is public to the server
- Node data (features and possibly labels) are private to nodes

Problem:

- How to learn a GNN without exposing private node data?



Local Differential Privacy

- Every data holder perturbs their data using a randomized mechanism
- The aggregator collects and aggregates perturbed data to estimate the target statistics

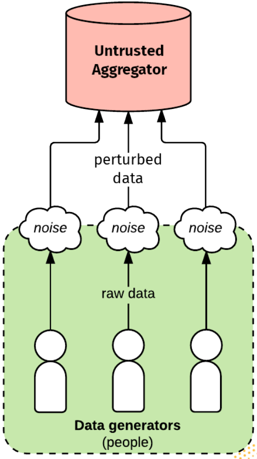
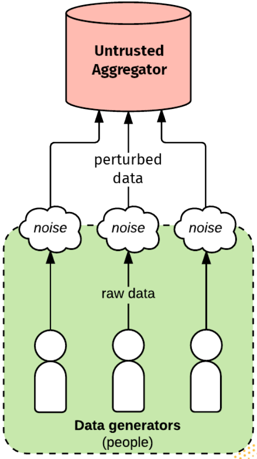


Image Credit: Bennett Cyphers

Local Differential Privacy

- Every data holder perturbs their data using a **randomized mechanism**
- The aggregator collects and **aggregates** perturbed data to **estimate** the target statistics



Definition

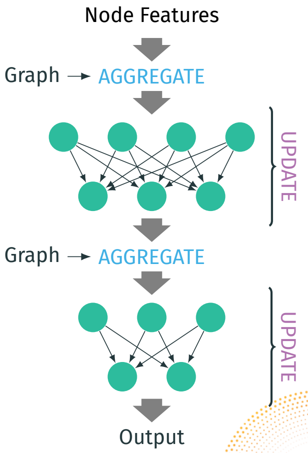
a randomized mechanism $\mathcal{M}$ satisfies ϵ-LDP if for all pairs of private data x_1 and x_2, and for all outputs x' of $\mathcal{M}$, we have:

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

Image Credit: Bennett Cyphers

GNNs are message-passing neural networks

- AGGREGATE: nodes aggregate their neighbors' representation vector
- UPDATE: a neural network generates new node representation from aggregated vectors



Why Local DP?

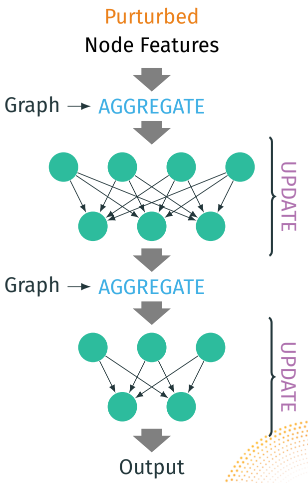
GNNs are message-passing neural networks

AGGREGATE: nodes aggregate their neighbors' representation vector

UPDATE: a neural network generates new node representation from aggregated vectors

Private neighborhood aggregation with LDP

- Node features are perturbed by injecting noise
- The neighborhood aggregation cancels out the noise



CHALLENGES

High-dimensional features

- The total privacy budget of a node scales with the number of features
  - Keeping the total privacy budget small$\rightarrow$Too much noise!

Challenges

High-dimensional features

- The total privacy budget of a node scales with the number of features
  - Keeping the total privacy budget small $\longrightarrow$ Too much noise!

Our solution: Multi-bit mechanism for multidimensional perturbation

- **Multi-bit Encoder:** perturb a random subset of node features and compress the output
- **Multi-bit Rectifier:** uncompress and de-bias encoded features

Challenges

Small neighborhoods

- Lots of the nodes have too few neighbors
  - Noise won't cancel out if the neighborhood size is small

Small neighborhoods

- Lots of the nodes have too few neighbors
    - Noise won't cancel out if the neighborhood size is small

Our solution: KProp linear convolution

- Expands the neighborhood to the nodes that are up to K-hops away
- Applies K consecutive AGGREGATE
- Can be prepended to any GNN architecture as a feature denoising mechanism

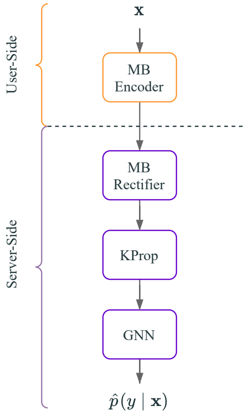
Locally Private GNN Architecture

User-Side:

1. Perturb node features using MB encoder
2. Send encoded features to server

Server-Side:

3. De-bias encoded features with MB rectifier
4. De-noise rectifier's output using KProp
5. Train GNN on denoised features



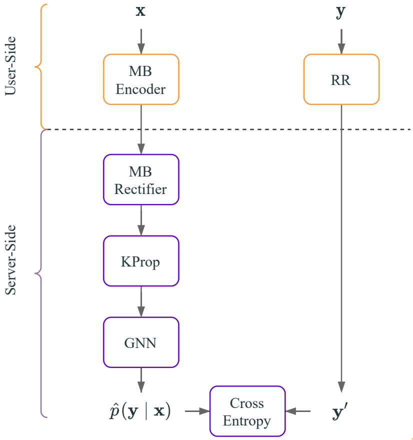
Randomized Response for label differential privacy

- True label $\mathbf{y}$
- Perturbed label $\mathbf{y}'$
- Number of classes c
- DP privacy budget ϵ

$$p(\mathbf{y}' \mid \mathbf{y}) = \begin{cases} \frac{e^\epsilon}{e^\epsilon + c - 1}, & \text{if } \mathbf{y}' = \mathbf{y} \\ \frac{1}{e^\epsilon + c - 1}, & \text{otherwise} \end{cases}$$

Trivial method: directly train GNN with noisy labels

- GNN severely overfits the noisy labels
- Poor generalization performance

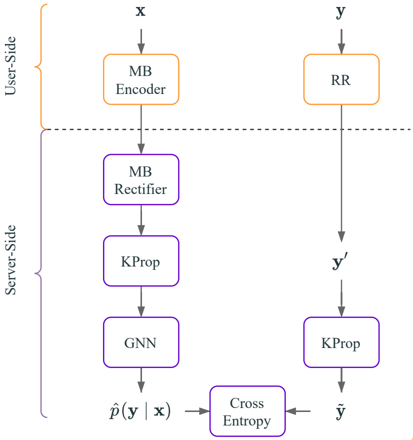


Trivial method: directly train GNN with noisy labels

- GNN severely overfits the noisy labels
- Poor generalization performance

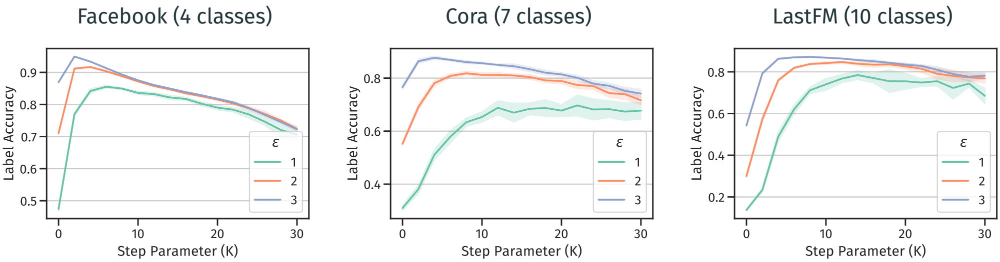
Key Idea: use KProp to denoise labels!

- Apply KProp on one-hot encoded noisy labels
- Pick the label with highest value



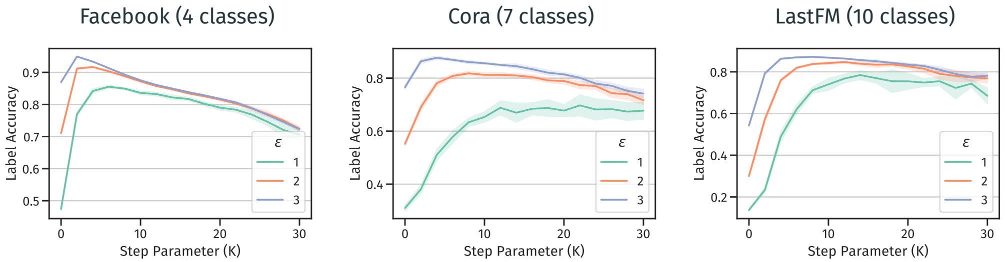
Effect of KProp on label accuracy

- Accuracy between true label $\mathbf{y}$ and recovered label $\tilde{\mathbf{y}}$



Effect of KProp on label accuracy

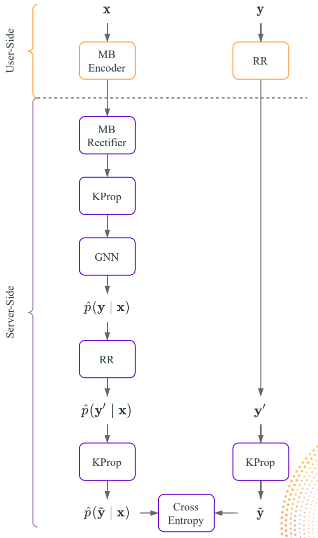
- Accuracy between true label **y** and recovered label **ỹ**



How to find best performing K without clean validation data?

Prevent absorbing noise in $\tilde{\mathbf{y}}$

- $\mathbf{y}$ is perturbed by RR and is given KProp to get $\tilde{\mathbf{y}}$
- Apply the **same process** on $\hat{p}\,(\mathbf{y} \mid \mathbf{x})$ to obtain $\hat{p}\,(\tilde{\mathbf{y}} \mid \mathbf{x})$
- Train $\hat{p}\,(\tilde{\mathbf{y}} \mid \mathbf{x})$ with $\tilde{\mathbf{y}}$



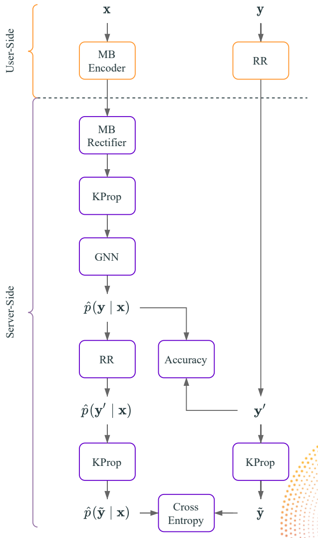
Label Denoising with Propagation

Prevent absorbing noise in $\tilde{\mathbf{y}}$

- $\mathbf{y}$ is perturbed by RR and is given KProp to get $\tilde{\mathbf{y}}$
- Apply the **same process** on $\hat{p}(\mathbf{y} \mid \mathbf{x})$ to obtain $\hat{p}(\tilde{\mathbf{y}} \mid \mathbf{x})$
- Train $\hat{p}(\tilde{\mathbf{y}} \mid \mathbf{x})$ with $\tilde{\mathbf{y}}$

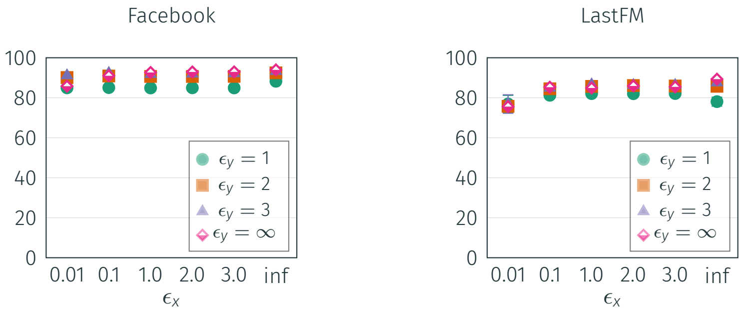
Prevent absorbing noise in $\mathbf{y}'$

- RR gives an **upperbound** on label accuracy:
  $Acc^* = p(\mathbf{y}' = \mathbf{y}) = \frac{e^\epsilon}{e^\epsilon + c - 1}$
- Stop training when GNN's accuracy on $\mathbf{y}'$ goes beyond Acc^*



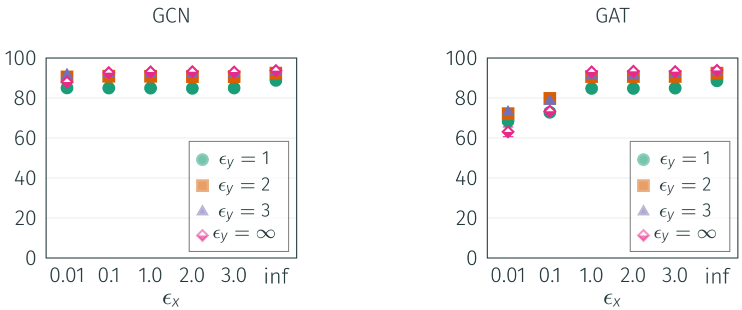
LPGNN's performance under varying feature and label privacy budgets

- Base GNN: GraphSAGE



Comparison of base GNN architectures

- Dataset: Facebook



Comparison of different LDP mechanisms

- Base GNN: GraphSAGE
- $\epsilon_y = \infty$

Dataset	Mechanism	$\epsilon_x = 0.01$	$\epsilon_x = 0.1$	$\epsilon_x = 1$	$\epsilon_x = 2$
Cora	1B	45.8 ± 3.3	62.3 ± 1.5	59.9 ± 2.7	58.5 ± 2.9
	LP	43.2 ± 3.1	57.8 ± 2.3	61.9 ± 3.1	58.1 ± 2.1
	AG	59.7 ± 2.3	62.7 ± 2.8	67.5 ± 3.0	77.2 ± 1.9
	MB	$\mathbf{68.0 \pm 2.9}$	$\mathbf{64.6 \pm 3.2}$	$\mathbf{83.9 \pm 0.4}$	$\mathbf{84.0 \pm 0.3}$
Facebook	1B	57.0 ± 3.4	76.3 ± 1.6	86.1 ± 0.6	84.0 ± 1.3
	LP	54.2 ± 2.9	72.5 ± 2.1	85.4 ± 0.4	84.8 ± 1.6
	AG	78.2 ± 1.4	85.6 ± 0.7	92.0 ± 0.1	92.4 ± 0.2
	MB	$\mathbf{85.8 \pm 0.4}$	$\mathbf{91.0 \pm 0.4}$	$\mathbf{92.7 \pm 0.1}$	$\mathbf{92.9 \pm 0.1}$

Comparison of different learning algorithms

- Base GNN: GraphSAGE
- $\epsilon_x = 1$

Dataset	ϵ_y	Cross Entropy	Forward Correction	Drop
CORA	0.5	18.6 ± 1.3	18.6 ± 2.5	$\mathbf{42.9 \pm 1.5}$
	1.0	25.5 ± 1.7	37.1 ± 2.5	$\mathbf{69.3 \pm 1.2}$
	2.0	52.9 ± 2.1	75.1 ± 1.0	$\mathbf{78.4 \pm 0.7}$
FACEBOOK	0.5	50.9 ± 4.2	68.9 ± 1.3	$\mathbf{75.1 \pm 0.6}$
	1.0	55.2 ± 1.3	73.8 ± 1.1	$\mathbf{84.9 \pm 0.2}$
	2.0	81.6 ± 1.2	88.9 ± 0.2	$\mathbf{90.7 \pm 0.1}$
LASTFM	0.5	21.1 ± 4.6	44.9 ± 5.3	$\mathbf{70.0 \pm 3.0}$
	1.0	28.4 ± 2.5	58.5 ± 3.6	$\mathbf{82.1 \pm 1.0}$
	2.0	56.8 ± 2.8	79.2 ± 1.3	$\mathbf{85.7 \pm 0.7}$

Conclusion

Summary

- Proposed a privacy-preserving GNN based on local differential privacy
  - Multi-bit mechanism for high-dimensional feature perturbation
  - KProp for feature and label denoising
  - Drop algorithm for learning with noisy labels
- Demonstrated promising results in terms of accuracy-privacy trade-off

Future Work

- Protect privacy of graph topology

Thank You!

sajadmanesh

sajadmanesh@idiap.ch