Using Node Similarity to Achieve Local Differential Privacy in Graph Neural Networks

Project Presentations (CMPUT-622)

Saqib Ameen (saqib1)
Thirupathi Reddy Emireddy (emireddy)
Justin Stevens (jdsteven)

UNIVERSITY OF ALBERTA

Graph Neural Networks (GNN)

- Utilize graph structure
- Learns representation of relational data

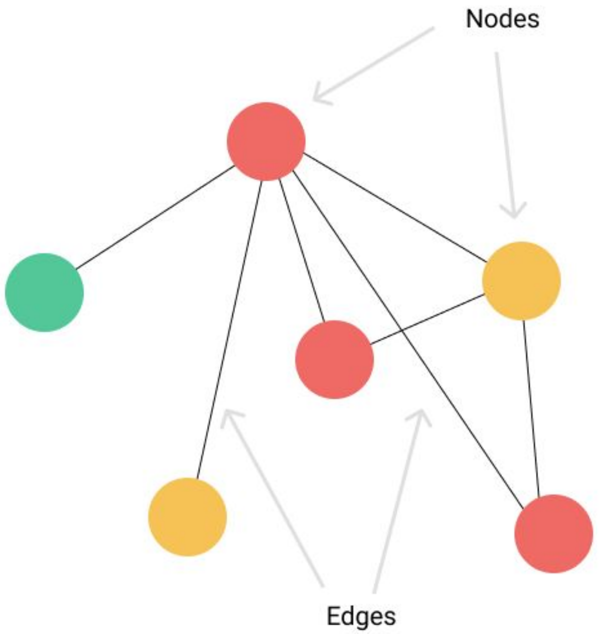
Graph Neural Networks (GNN)

- Utilize graph structure
- Learns representation of relational data
- Superior performance:
  - Molecular chemistry
  - Social Networks
  - Network analysis
  - Fraud Detection
  - Recommendation Systems

Image: https://alphafold.ebi.ac.uk/

Applications of GNN

- **Node classification (our task)**
- Edge/Link Prediction
- Graph Classification



Privacy in GNNs

- Recommendation engines
  - Facebook, Bumble, Ads

Privacy in GNNs

- Recommendation engines
  - Facebook, Bumble, Ads
- Node Data Privacy Settings
  - Personal Information Usage
  - Better Recommendations

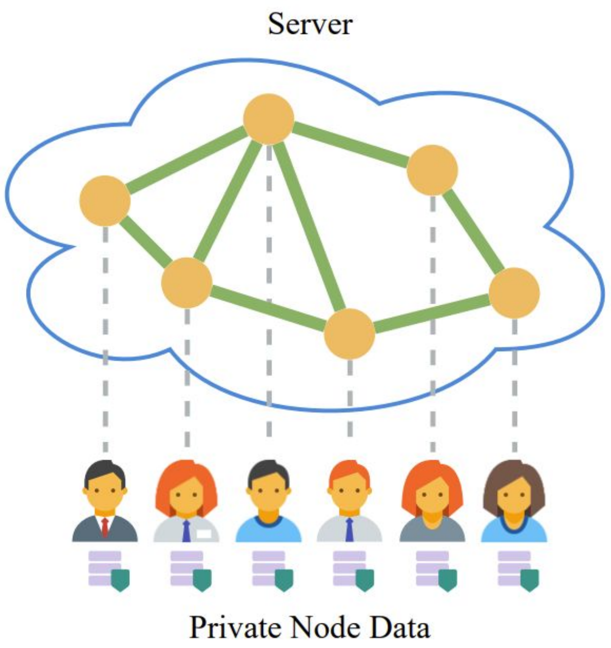


Image: Sajadmanesh, Sina & Gatica-Perez, Daniel. (2020). Locally Private Graph Neural Networks.

Privacy in GNNs

- Recommendation engines
  - Facebook, Bumble, Ads
- Node Data Privacy Settings
  - Personal Information Usage
  - Better Recommendations
- *Problem*
  - General Data Protection Regulation (GDPR)/ legal compliances
  - Data Privacy/ Anonymization

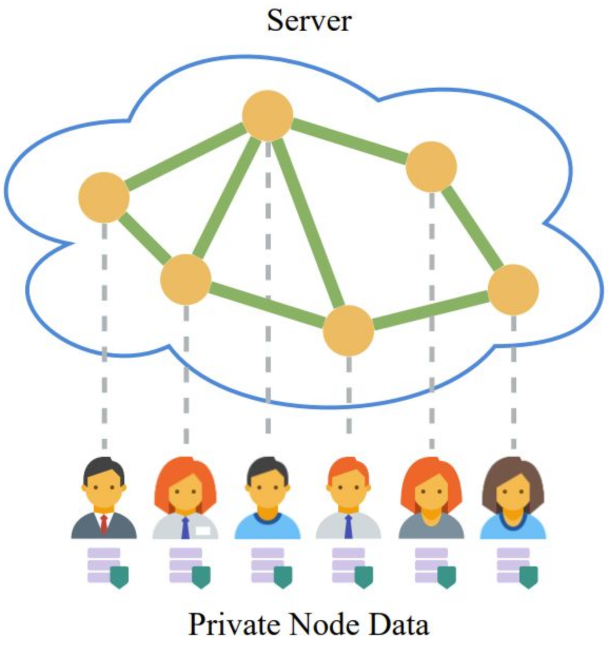


Image: Sajadmanesh, Sina & Gatica-Perez, Daniel. (2020). Locally Private Graph Neural Networks.

Privacy in GNNs (cont.)

- Goal
  - Privacy guarantee for private data
  - Efficient Learning

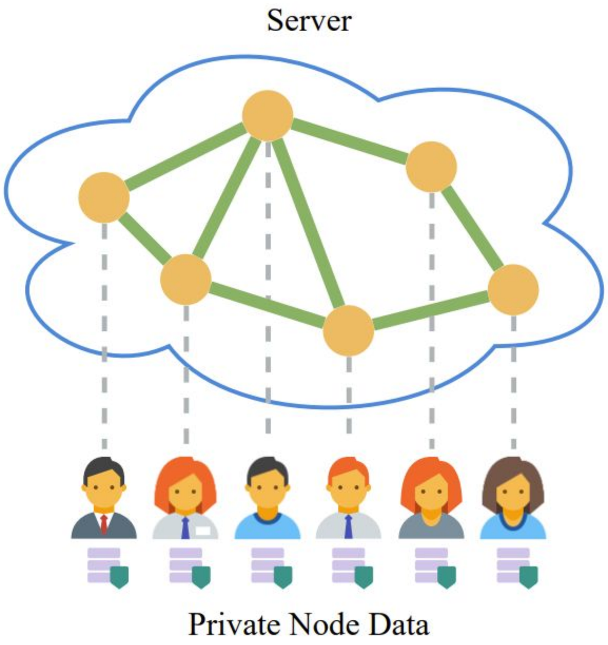


Image: Sajadmanesh, Sina & Gatica-Perez, Daniel. (2020). Locally Private Graph Neural Networks.

Graph Representation

- Consider a graph: $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathrm{X}, \mathrm{Y})$

  $\mathcal{V}$: Set of Nodes

  $\mathcal{E}$: Set of Edges

  X: Set of Nodes' Features

  Y: Set of Nodes' Labels

- In this graph, the nodes' features are indicated by colouring the node either orange or yellow.
  - In practice, nodes' features are vectors.

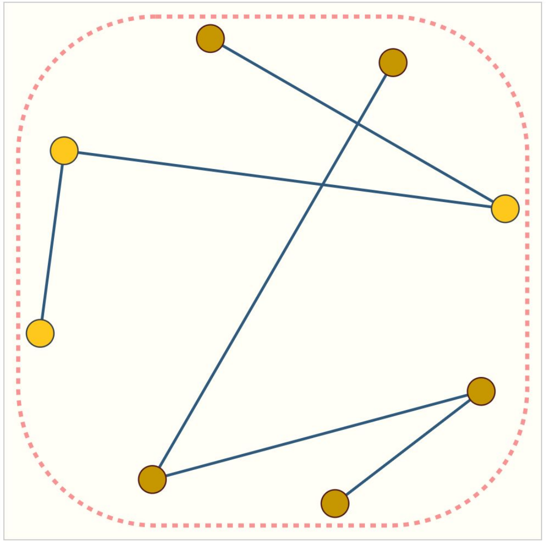
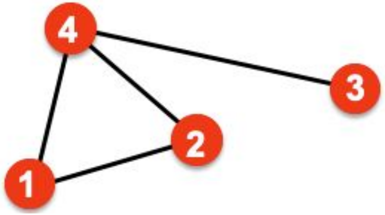


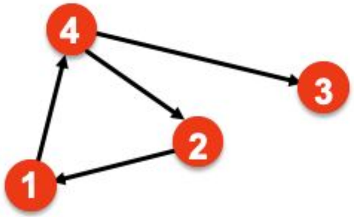
Image: https://distill.pub/2021/gnn-intro/

Adjacency Matrix

- One way graphs are represented is using the adjacency matrix A
    - $A_{ij} = 1$ if there is an edge between node i to j
    - $A_{ij} = 0$ otherwise



$$A = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

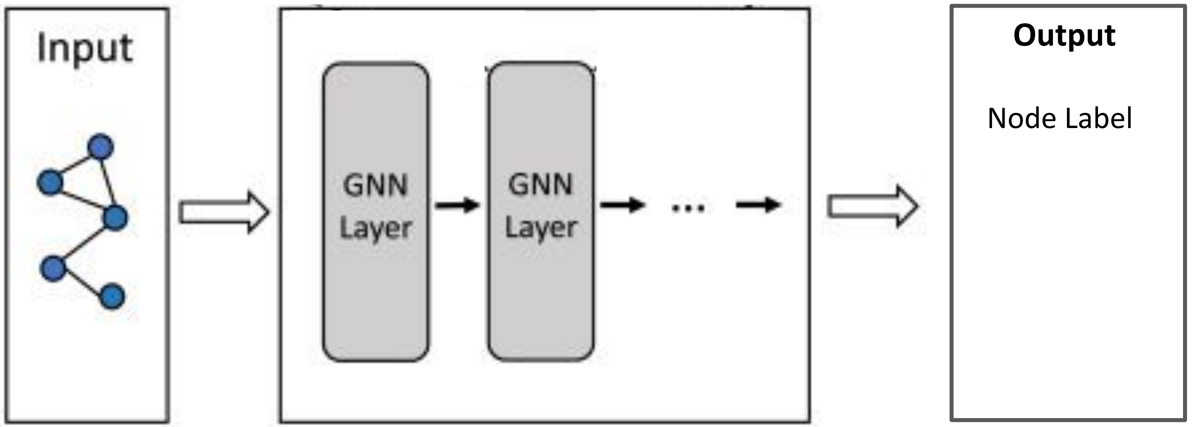


$$A = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

Images: http://web.stanford.edu/class/cs224w/slides/01-intro.pdf

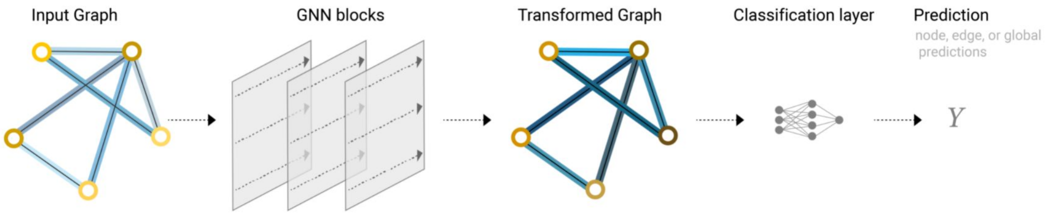
GNN

- GNN
  - Learns node representation using set of stacked graph convolution layers



GNN: Big Picture

- Given the input graph from before, we apply several GNN blocks (explained in the next slide) to obtain a new transformed graph with the same structure.
  - This new graph will have different features for each node.
- We then apply a classifier to obtain a prediction for every node.



An end-to-end prediction task with a GNN model.

Image: https://distill.pub/2021/gnn-intro/

GNN Block

- To obtain a transformation on the graph, we apply a GNN block a certain number of times. This block updates the information at a node by aggregating the information from all the neighbour’s nodes (known as *aggregation*).

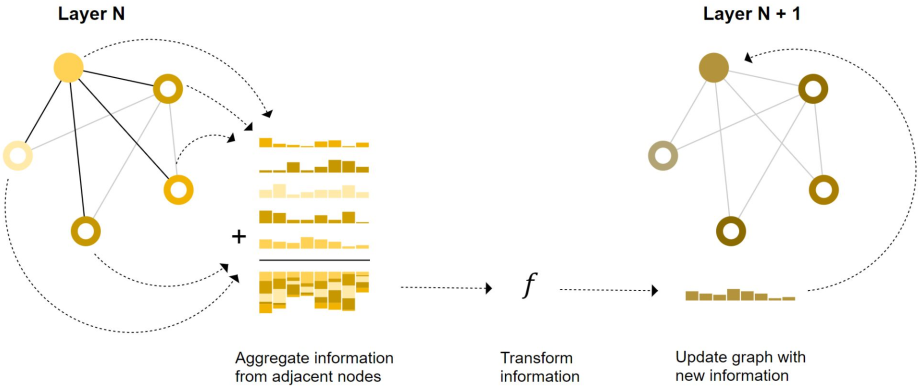


Image: https://distill.pub/2021/gnn-intro/

GNN (cont.)

- Initial embedding for node v is $h_v^0 = x_v$.
- New Embedding (h) for a node (v) at a layer l, in an M-layers GNN:

Neighboring nodes of v

$$\mathrm{h}_{\mathcal{N}(v)}^{l} = \textsc{Aggregate}_l\left(\{\mathrm{h}_u^{l-1}, \forall u \in \mathcal{N}(v)\}\right)$$

Aggregated embeddings from layer l-1 of neighboring nodes of v

Image: Sajadmanesh, Sina & Gatica-Perez, Daniel. (2020). Locally Private Graph Neural Networks.

GNN (cont.)

- Initial embedding for node v is $h_v^0 = x_v.$
- New Embedding (h) for a node (v) at a layer l, in an M-layers GNN:

Neighboring nodes of v

$$\mathrm{h}^l_{\mathcal{N}(v)} = \textsc{Aggregate}_l\left(\{\mathrm{h}_u^{l-1}, \forall u \in \mathcal{N}(v)\}\right)$$

$$\mathrm{h}_v^l = \textsc{Update}_l\left(\mathrm{h}^l_{\mathcal{N}(v)}\right)$$

Aggregated embeddings from layer l-1 of neighboring nodes of v

New Embedding for node v in layer l

Non-linear function (e.g. neural network)

Image: Sajadmanesh, Sina & Gatica-Perez, Daniel. (2020). Locally Private Graph Neural Networks.

Local Differential Privacy (LDP)

- Approach for collecting private data and computing statistical queries, such as mean, count, and histogram.
  - Used by Google, Microsoft, Apple
- Data holders send a perturbed version of their data using randomized mechanism $\mathcal{M}$
  - Not meaningful individually
  - Can approximate the target query when aggregated

LDP (cont.)

- To prevent aggregator from inferring the original value, $\mathcal{M}$ must satisfy:

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

Multi-bit Mechanism

- Multi-bit Encoder
  - Runs at user-side
  - Randomized feature selection and perturbation
  - Introduces bias
- Multi-bit Rectifier
  - Runs at server-side
  - Debiases
- Not a denoising step

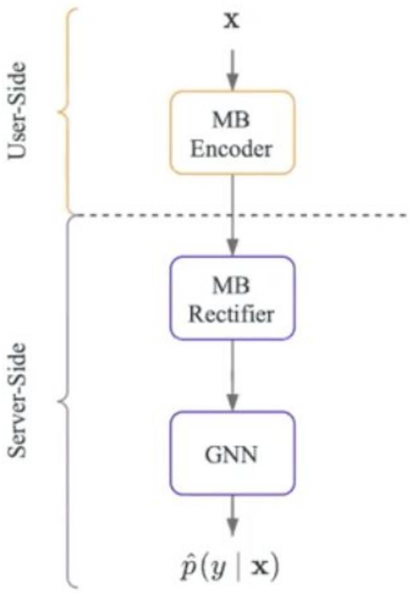


Image: Sajadmanesh, Sina & Gatica-Perez, Daniel. (2020). Locally Private Graph Neural Networks.

Noisy Labels

- To further improve the privacy of the model, a randomized response mechanism is applied to the labels.
- Let ϵ_y be the privacy budget for the labels. Then if c is the number of classes, we run a randomized response as
- $$p(y'|y) = \begin{cases} \frac{e^{\epsilon_y}}{e^{\epsilon_y}+c-1}, & \text{if } y' = y \\ \frac{1}{e^{\epsilon_y}+c-1}, & \text{otherwise} \end{cases}$$

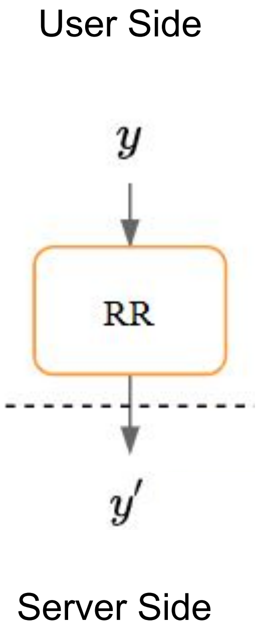


Image: Sajadmanesh, Sina & Gatica-Perez, Daniel. (2020). Locally Private Graph Neural Networks.

KProp Denoising

- Motivation
  - Noise $\longrightarrow$ Less Accuracy
  - For every node v, in the initial layer of GNN:

Linear Aggregation approx. error $\propto \frac{1}{\sqrt{N(v)}}$

- To reduce first layer approximation error, we perform the KProp algorithm.
  - K consecutive linear $\textsc{Aggregate}$ functions

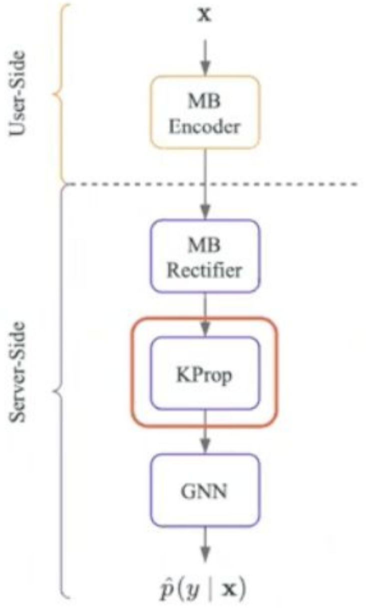
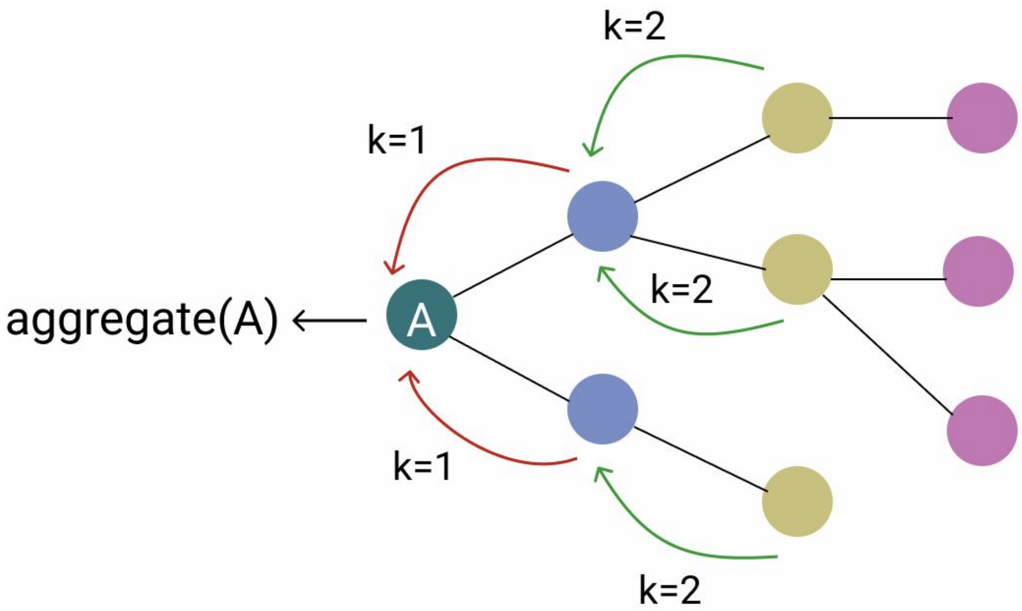


Image: Sajadmanesh, Sina & Gatica-Perez, Daniel. (2020). Locally Private Graph Neural Networks.

KProp Example (k=2)



Private Data and Label Privacy

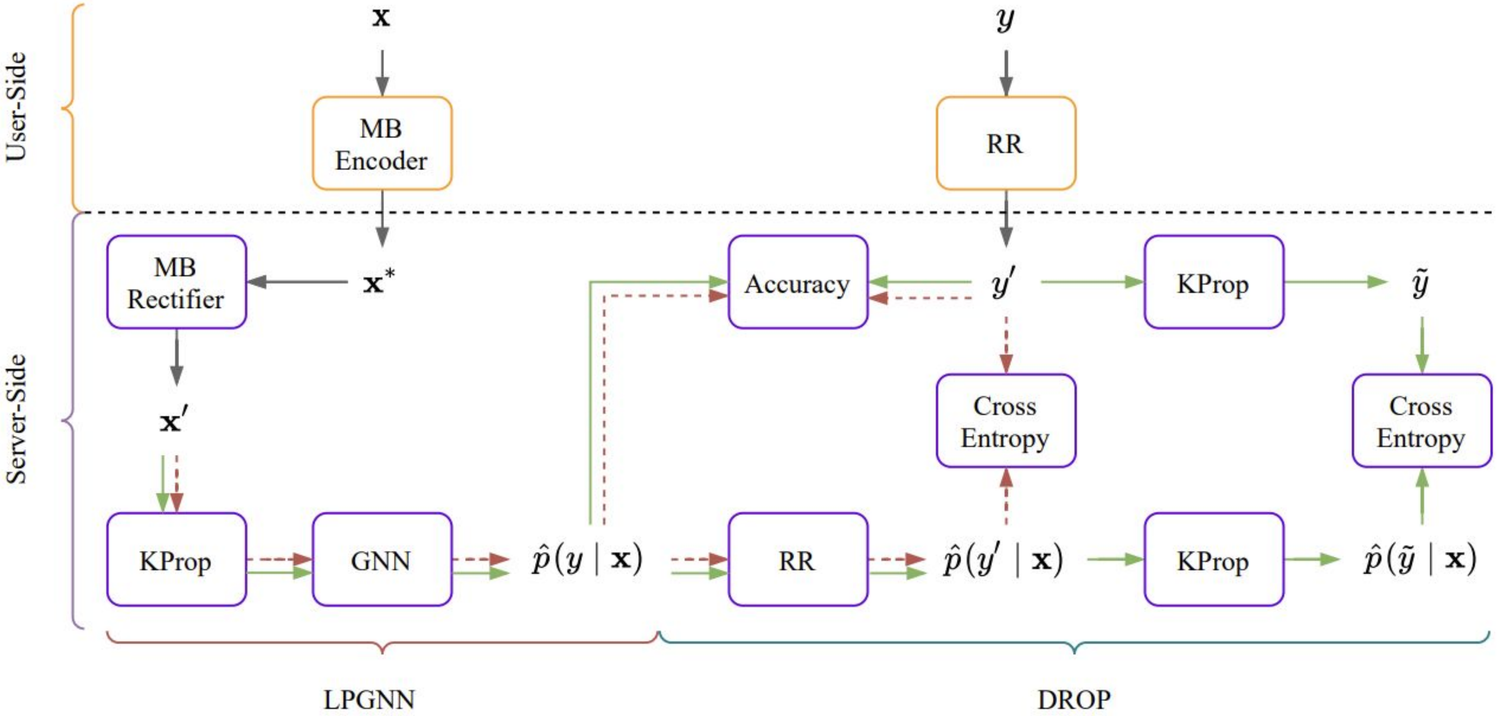
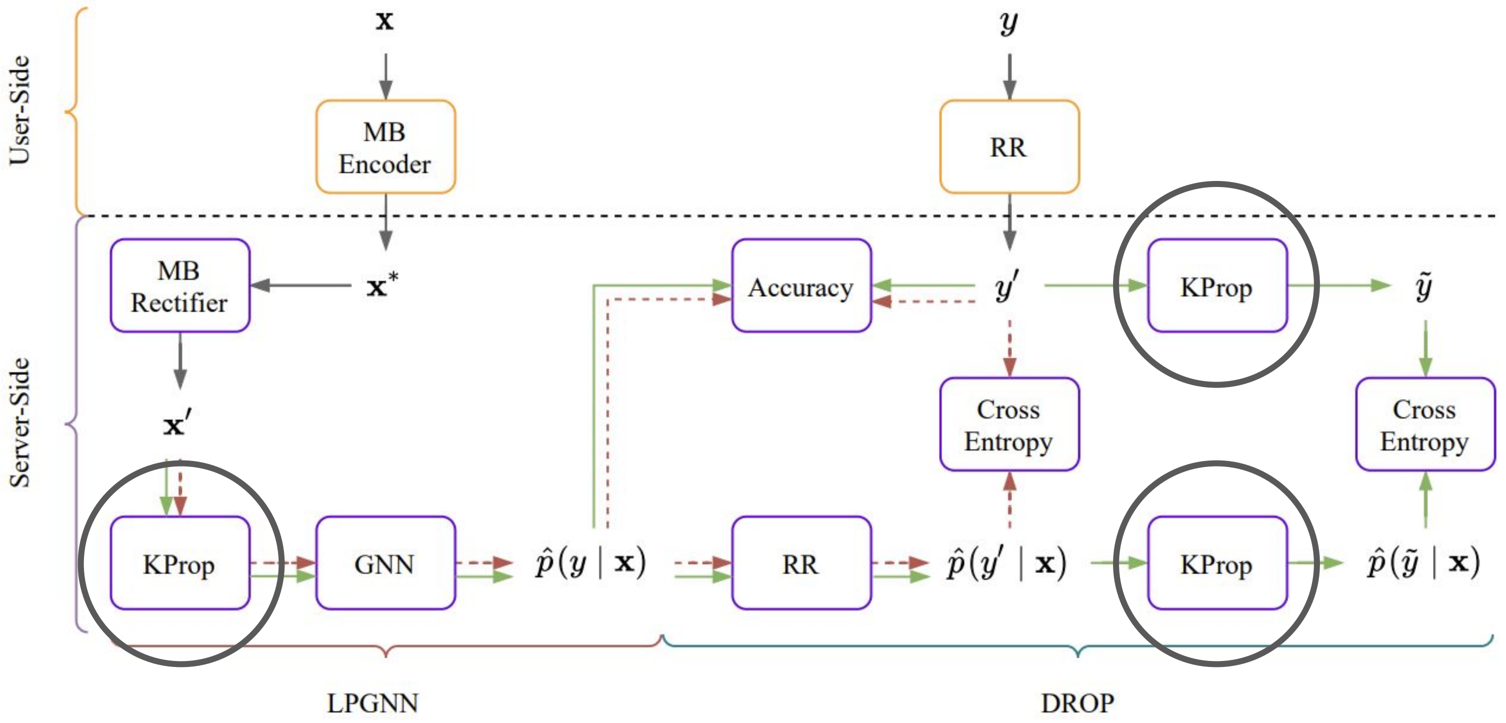


Image: Sajadmanesh, Sina & Gatica-Perez, Daniel. (2020). Locally Private Graph Neural Networks.

Private Data and Label Privacy



Research Question

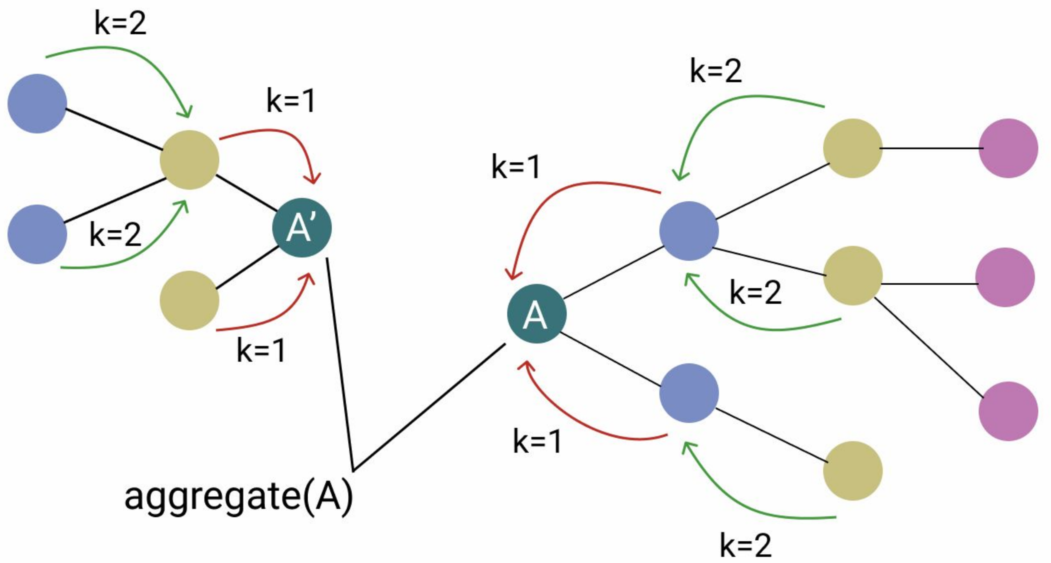
- KProp expands neighbourhood of a node to **k**-hops away
    - **Issue:** Nodes **k**-hops away might be uncorrelated with the given node as the value of **k** increases. We might lose accuracy.
- *Can aggregation of similar nodes' neighborhood with smaller* ***k*** *help us improve the denoising capabilities and provide better accuracy?*

SimGat

- Proposed algorithm: SIM(ILARITY) (AGGRE)GAT(OR)
- Exploits node similarity
- SimGat:
    - Identify similar nodes S_v for a given node v
    - Run k'-prop, where k' < k to avoid dissimilar nodes
    - Aggregate the features from neighbourhoods of similar nodes and the neighbours of the node itself.
- Goal: Denoise the aggregation data to obtain high accuracy

SimGAT Example

- Consider the node A’ that is similar to node A and use its neighbours for node A aggregation



Node Similarity

- Three similarity metrics considered:
    - Multi-hop Similarity (GraRep[1])
    - DeepWalk[2]
    - node2vec[3]

[1] Cao et al. 2015. GraRep: Learning Graph Representations with Global Structural Information
[2] Perozzi et al. 2014. DeepWalk: Online Learning of Social Representations.
[3] Grover et al. 2016. node2vec: Scalable Feature Learning for Networks.

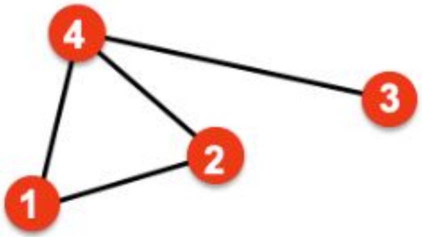
Multi-hop Similarity

1. Calculate the transition probability matrix **P**:

$$P_{ij} = \frac{T_{ij}}{\sum_{c} T_{ic}} \qquad \text{where} \qquad T = \sum_{m=1}^{n} A^m$$

1. A is adjacency matrix, n is number of hops
2. We define the similarity matrix **S** based on similarity threshold $\boldsymbol{\delta}$:

$$S_{ij} = \begin{cases} 1, & \text{if } P_{ij} > \delta \\ 0, & \text{otherwise} \end{cases}$$



$$A = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

Random Walk

- Start from node u and run a random walk over the graph using a search strategy R.
- *Similarity:* Probability that u and v occur on the same random walk is used as a measure of their similarity.
- DeepWalk and node2vec are different strategies (R) of doing Random Walk.

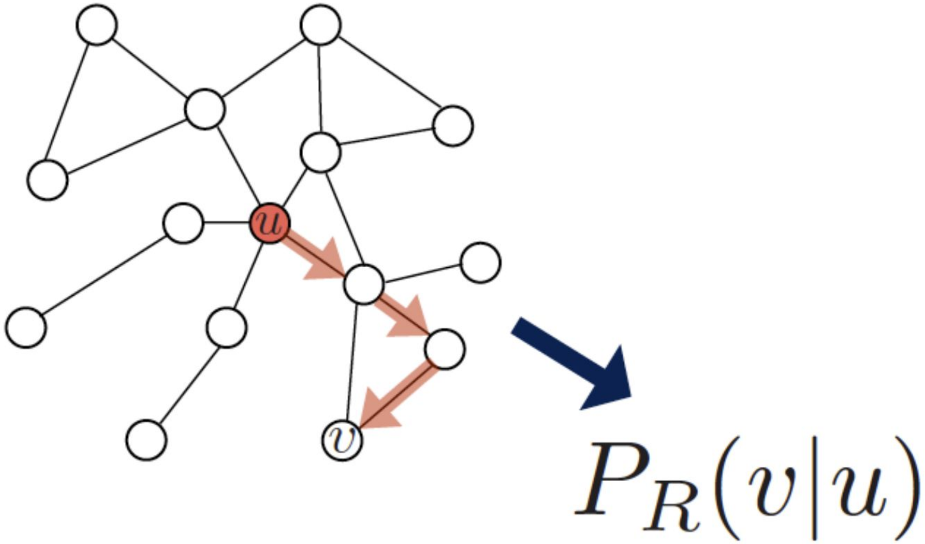
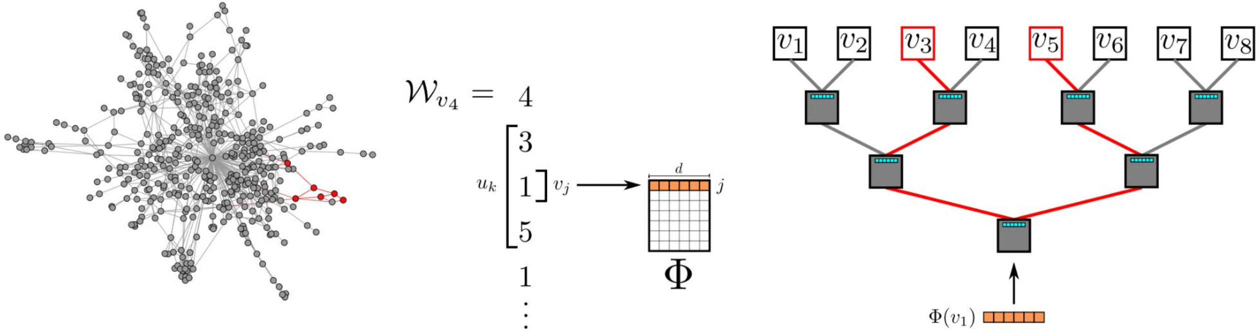


Image: https://web.stanford.edu/class/cs224w/slides/03-nodeemb.pdf

DeepWalk

- Walks as sentence metaphor
- Use *skip gram model* to identify similar nodes during random walks
- Uses *Hierarchical Softmax* function to optimize the computation
- Unbiased random walks (treats all nodes equally to pick the next)



(a) Random walk generation. (b) Representation mapping. (c) Hierarchical Softmax.

Image: Perozzi et al. 2014. DeepWalk: Online Learning of Social Representations.

node2vec

- Biased random walk that balances staying close to the node (parameter p) and moving away (parameter q).
- In the graph on the right, let's suppose the random walk strategy has moved from s_1 to w.
  - With probability proportional to 1/p, move back to s_1.
  - With probability proportional to 1, move to a node 1-hop away from s_1.
  - With probability proportional to 1/q, move to a node 2-hops away from s_1.

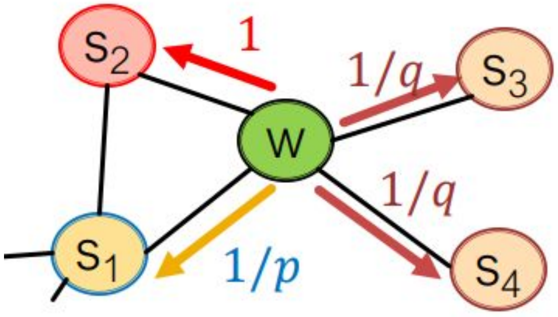


Image: https://web.stanford.edu/class/cs224w/slides/03-nodeemb.pdf

Datasets

Dataset	#Classes	#Nodes	#Edges	#Features	Avg. Deg.
Cora	7	2,708	5,278	1,433	3.90
PubMed	3	19,717	44,324	500	4.50
Facebook	4	22,470	170,912	4,714	15.21
LastFM	10	7,083	25,814	7,842	7.29

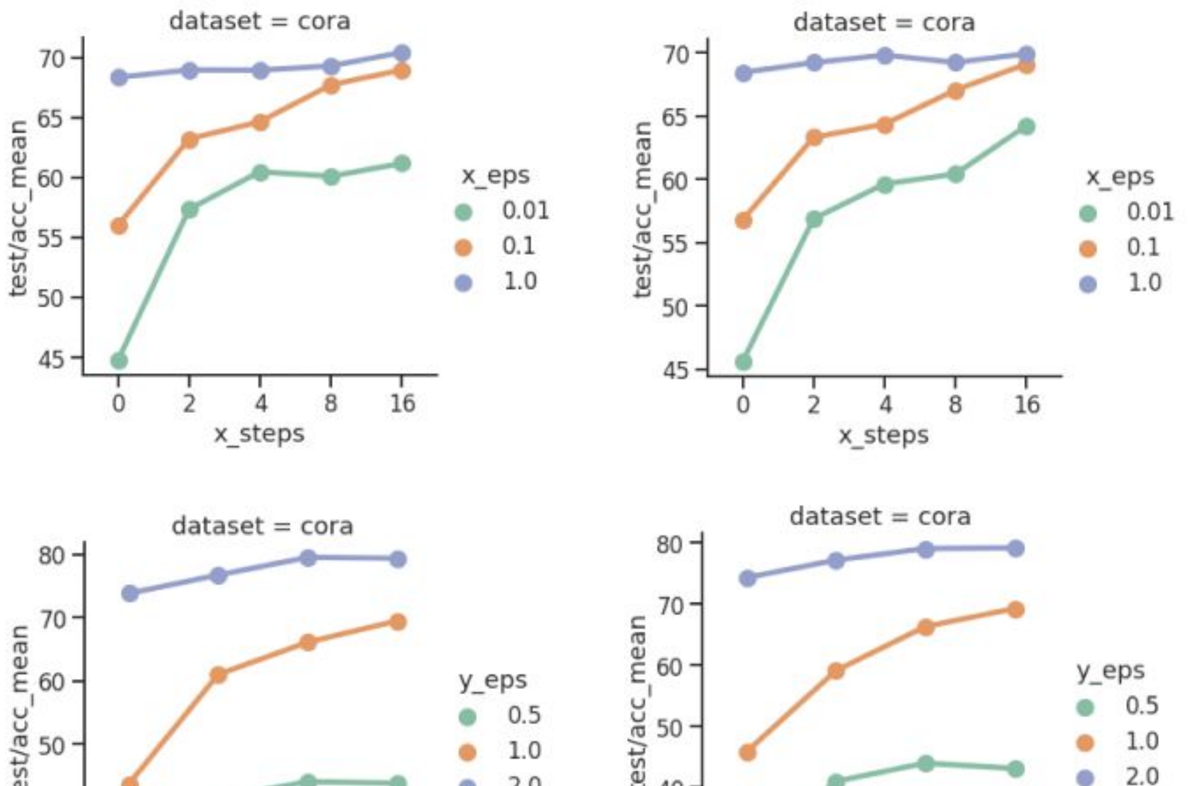
Source: Sajadmanesh, Sina & Gatica-Perez, Daniel. (2020). Locally Private Graph Neural Networks.

Evaluation

- Compare the accuracy of KProp (pre-existing work) to SimGAT (our proposed architecture) with each of the three different similarity methods on the datasets.
  - We use a 50% training, 25% validation, 25% testing split of the data.
- We keep the rest of the GNN model the same as the one in KProp paper [1].
- We'll vary the privacy budget for features, ε_x , for the experiments.

[1] Sajadmanesh, Sina & Gatica-Perez, Daniel. (2020). Locally Private Graph Neural Networks.

Preliminary Results



Future Experiments

- In the previous experiments, we ran this with k’=k-1. We plan to run it with different and much smaller values of k’ and different hyperparameter to reduce k’ while maintaining accuracy with SimGAT.
- We will run results on the other 3 datasets for a full comparison.

Summary

- GNN are powerful and widely used in a wide array of
- Using LDP to preserve user data privacy.
- Improved denoising of first layer of GNN to improve accuracy.
- Our New method utilizes similarity of nodes in the graph.
  - GraRep, DeepWalk, node2vec
- Results comparison with the state-of-the-art mechanism KProp

Questions?