NodeDrop: Graph Membership Inference Attack Mitigation

Presented by:
M. Shahid Modi - 661999357

Project Details

- Regularization reduces overfitting by reducing member-specific information learned by a model.
- Regularization emerged as the first line of defense against membership inference attacks.
- Graph networks have a large degree imbalance between high degree and low degree nodes.
- Idea: Regularization by randomly dropping out specific nodes in each epoch?


WE INTERRUPT THIS PROGRAM FOR A
COMMERCIAL BREAK
BROUGHT TO YOU BY BIG DATA

What is a GNN?

Let's talk graphs!

- A graph is just another way to represent data.
- It consists of Nodes (vertices) and Edges.
- G(V,E) is a graph G with:
    - set V of vertices
    - set E of edges.

- Graphs are stored and utilized as edgelists or adjacency matrices.

- Graphs have certain traits that give them advantages over Euclidean data.

- This includes concepts like neighborhood and degree.

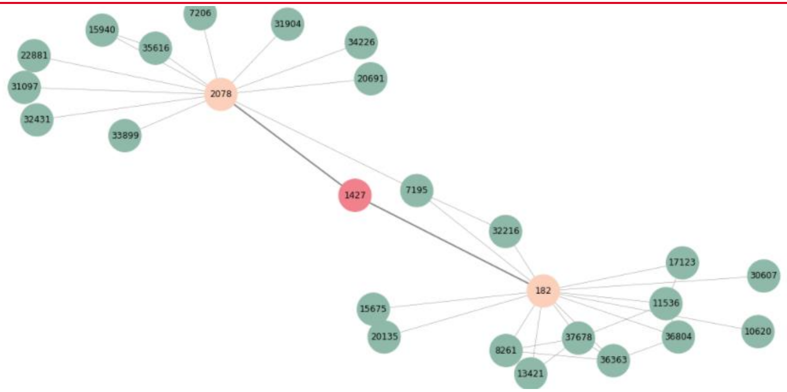
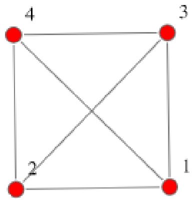


Figure 1: A Graph! Yellow nodes are especially **high degree.**



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

Figure 2: Another Graph! This time with **A, its adjacency matrix.**

What is a GNN?

There are graph datasets. For example, CORA!

- The Cora dataset consists of 2708 scientific publications (nodes) classified into one of seven classes.

- The citation network consists of 5429 links (edges).

- Wait, classes?

- Yes! CORA is an attributed graph. Each node has a label and an attribute array.

- Specifically, each node is described by a 0/1-valued word vector indicating the absence/ presence of the corresponding word from a 1433-word dictionary.

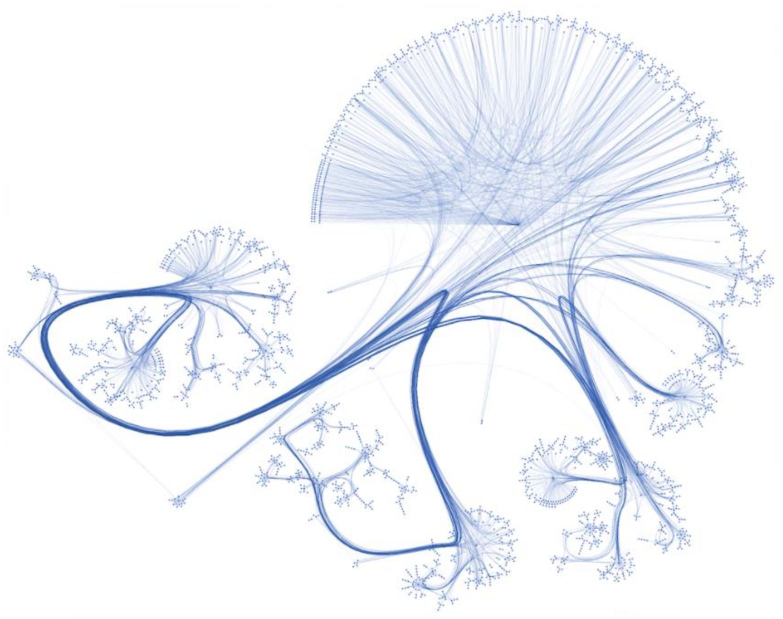


Figure 3: A Deep-Sea Monster! Actually, it's the CORA dataset.

What is a GNN?

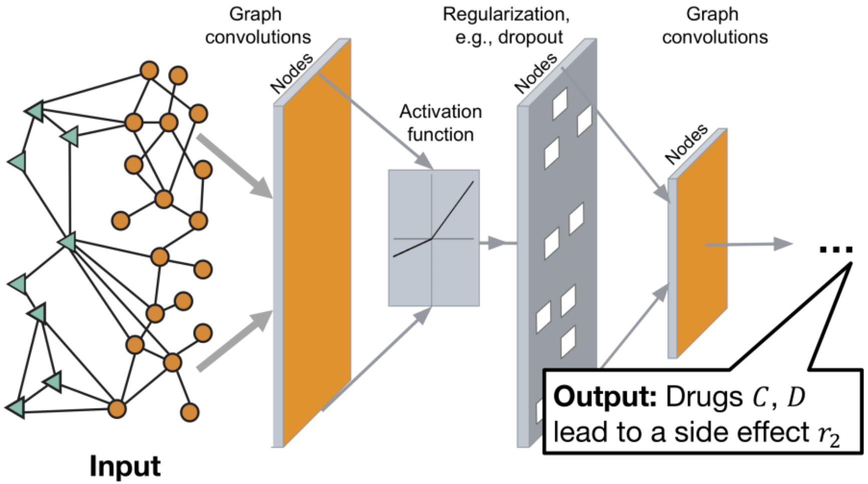


Figure 4: A GCN. Hey, that looks like a neural network but with graphs?

So what IS a GNN?

- Simply put, a Graph Neural Network is a Neural Network for Graphs.
- There are different types, such as Graph Convolutional Networks (Figure 4), GraphSAGE and others.
- They have an input layer, hidden layers, and output layers.
- Each layer has an aggregation function followed by an update function.


AND NOW BACK TO
OUR REGULARLY
SCHEDULED
PROGRAMMING

Theory…

Dropping Low Degree Nodes Might Hide Them From Attacks.

- It is possible to conduct Membership Inference attacks on GNNs.
- Experimental analysis shows that low degree nodes are more vulnerable to MI exposure.
- What if we regularized our GNN by randomly dropping low degree nodes before each round of training?

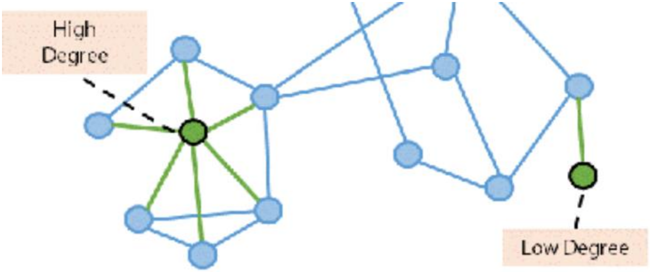


Figure 5: High and Low Degree Nodes

GraphSAGE – The GCN used for experimentation.

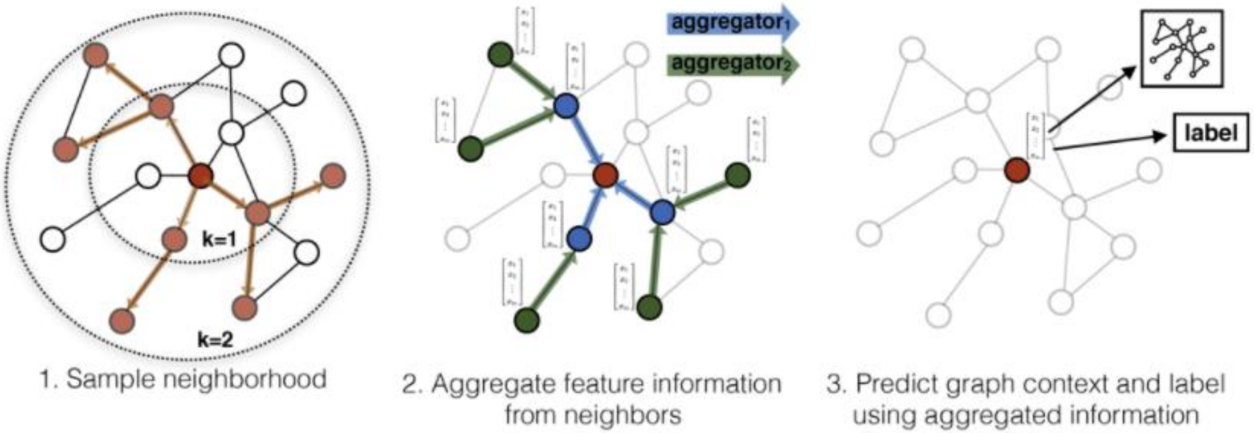


Figure 6: The GraphSAGE idea.

Algorithm - NodeDrop

Initialization:

- ➢ deg_list[ ]: Degree of each training node by index (predetermined)
- ➢ train_set, val_set, test_set lists: Node lists by index

Repeat for each epoch:

- → Set drop_list = randomly generated list of n training nodes having degree below threshold c.
- → Mask out all nodes in drop_list and their edges from train_set.
- → Train model for one epoch.

Algorithm

Tunable Parameters (n, c):

- ➢ n is the number of nodes that are dropped randomly before each round.
- ➢ c is the degree threshold, only nodes with degree below c are considered droppable.
- ➢ Goal: Find ideal combination of n,c values such that number of high degree and low degree nodes per round is equalized.

→ I chose to go with average degree as the threshold. Average degree of the CORA dataset is ~4.

Baselines

From [1]:

- Using the Cora graph citations dataset.
- Achieved a 0.754 attack accuracy on Cora using GraphSage.
- Their random edge addition defense method decreased attack accuracy proportional to decrease in model accuracy.
- The hit to utility was too high. At 20x edge additions, model accuracy dropped below ~0.65 and attack accuracy below ~0.6.
- I also used some experimental setup from this source.

Attack! Pipeline

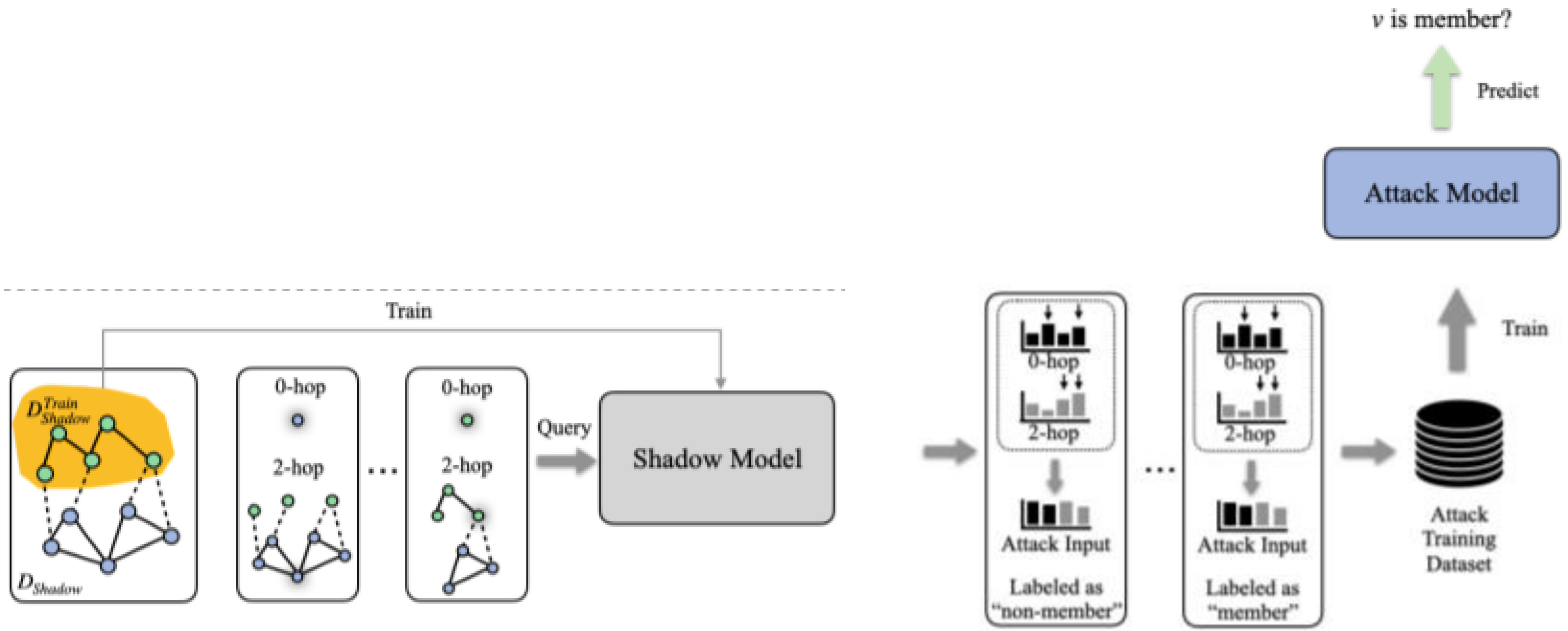


Figure 7: The attack pipeline used in [0]. I only tested 0 hop.

My Outcomes:

Important outcomes:

- Significant boost to defense: Attack accuracy was ~0.60 using same type of attack as [1].
- Low hit to accuracy: 0.1458 without drop to 0.1666 with drop (~0.02 hit to accuracy).
- High number of false negatives?

Some reflections…

Important Related Works

[1] Baselines for attack effectiveness from: He, Xinlei, et al. "Node-level membership inference attacks against graph neural networks." arXiv preprint arXiv:2102.05429 (2021).

[2] Attack training technique and Additional Baselines: Olatunji, Iyiola E., Wolfgang Nejdl, and Megha Khosla. "Membership inference attack on graph neural networks." arXiv preprint arXiv:2101.06570 (2021).

[3] Regularization outcomes on non-graph models: Li, Jiacheng, Ninghui Li, and Bruno Ribeiro, "Membership inference attacks and defenses in classification models." Proceedings of the Eleventh ACM Conference on Data and Application Security and Privacy (2021)

[4] Important reference: Hu, Hongsheng, et al. "Membership inference attacks on machine learning: A survey." arXiv preprint arXiv:2103.07853 (2021).