Simple MLP

Experiment to use a multilayer perceptron to predict the communities for each node. The test is conducted on a subsample of 200 communities.

Inputs:

Feature vector: Matrix containing the vector representation of all nodes
Each node is converted into a feature vector that contains:

1. embedding using Node2Vec algorithm dimension (embedding_dim , 1)
   - We use embedding_dim = 16 for our use case
2. Degree, int
3. Eigenvector centrality, float
4. Clustering Coefficient, float
5. Square clustering coefficient, float

Outputs:

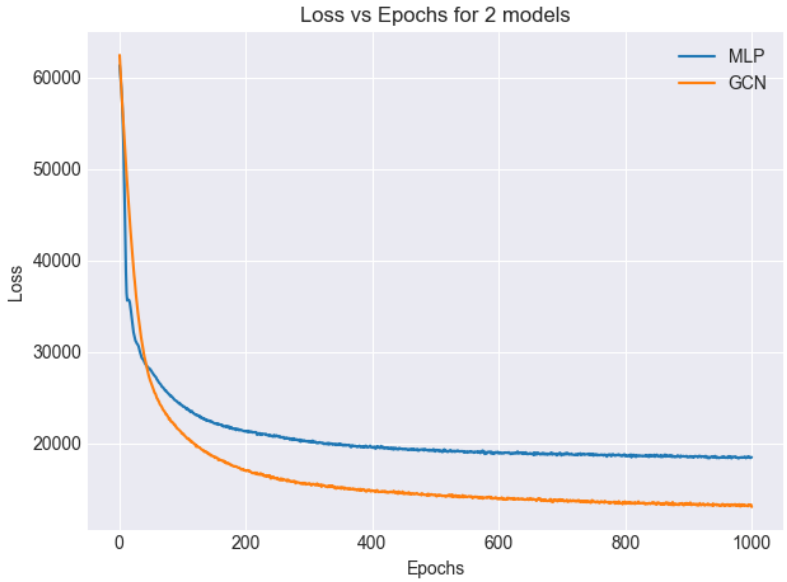
For each node we predict a (n_community,1) dimensional vector. Entry i is 1 if node is a part of the community i else 0.
The output matrix is sparse in nature as it contains (~98% 0s and remaining 1s)

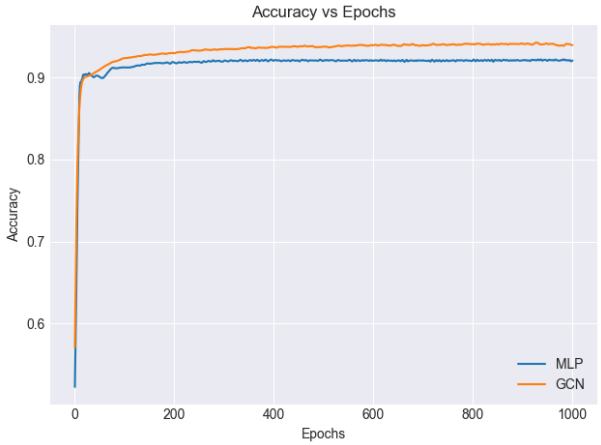
Goal

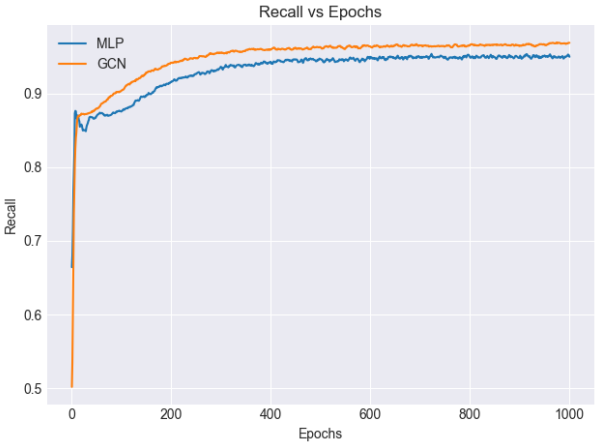
1. To quantify the ability of feature vectors (without edge information) to predict the output i.e community labels
2. To get motivation for using Graph Networks

Observations

1. An MLP model with 4296 parameters performs poorly as compared to a GCN model with similar number of parameters. It settles to an accuracy and recall of 86.63% and 63.35% respectively.
2. Throughout training there is an evident and strong tradeoff between accuracy and recall. This is highlighted in the graph given below.
3. This may also be caused by the model learning exact 0s which makes the loss infinite and breaks the learning process.





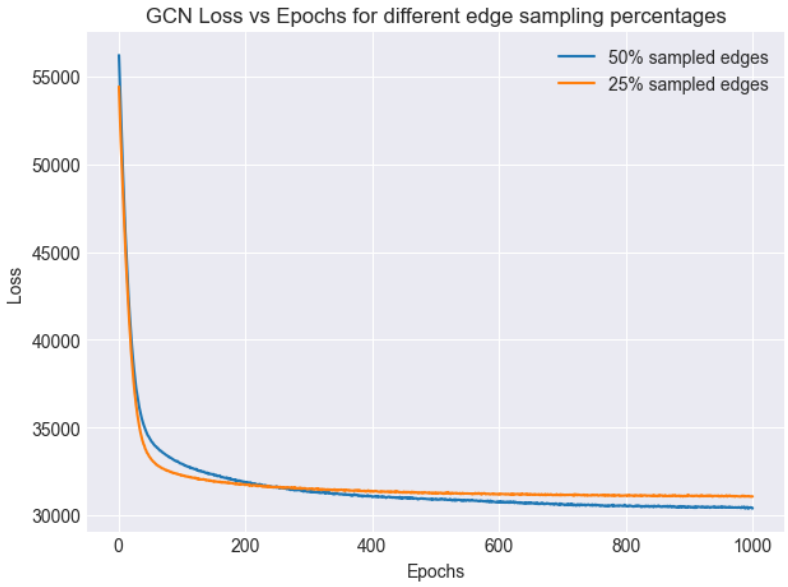


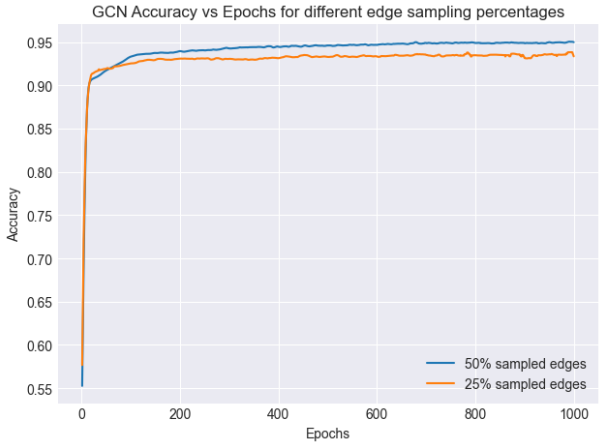
GCN

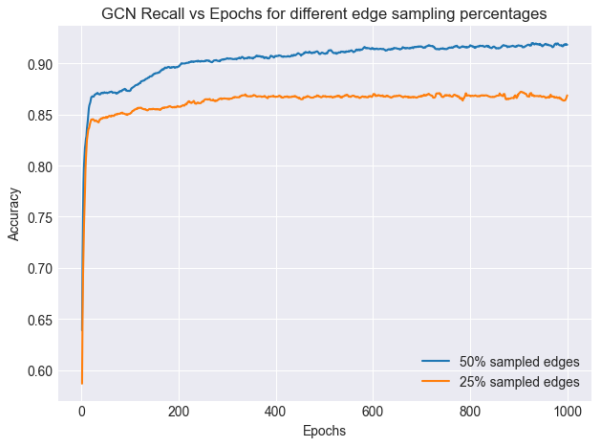
The results obtained from the MLP motivates us to use a graph network that leverage edge information Experiment to use a Graph Convolutional Network to predict the communities for each node. The test is conducted on a subsample of 200 communities with 50% edges sampled randomly.

Edge sampling

Edge sampling is done to predict a nodes communities based on its incomplete interaction with the graph. This is representative of the real world use case:
Consider the example of a social networking platform like Facebook. When a new user(node) joins the graph, it doesn't immediately form all its edges. So initially we won't have all its edge information to predict its communities.
To make our model give a quality predictions when all the node edges are not available we test the it on datasets having different percentages of retained edges







Inputs to GCN:
Along with the Matrix containing the feature vectors for each node as used in the MLP, the GCN also takes in
1. edge_index: adjacency list of edge
For message passing and creating the local neighbourhood networks

Goal
1. To predict the communities with high accuracy and recall (this is important because the output matrix is sparse in nature)
2. To gauge that Graph Convolutional Networks leverage edge information appropriately and perform better than an MLP
3. To check the models performance on different levels of edge sampling

Observations
1. An GCN model with 4392 parameters performs well
2. Improved training
3. Performance on edge sampling

Loss function
A custom weighted Binary Cross Entropy loss function is used to make the learning process efficient.

$$BCE = -\frac{1}{N}\sum_{i=0}^{N} y_i \cdot log(\hat{y}_i) + (1 - y_i) \cdot log(1 - \hat{y}_i)$$

Standard BCE loss is used for multi-label classification. The first term in the above penalises 1s predicted as 0s and the second term penalises 0s predicted as 1s.
The output matrix is sparse and contains a lot more 0s. Hence, the loss is heavily influenced by the second term.
When gradient descent is performed this loss function, 0s are predicted correctly causing high accuracy and low recall.

To fix this, we add a simple modification by adding weights to these terms

$$BCE_{sparse} = (-\Sigma\, w_{true} \cdot y_i \cdot \log(\hat{y}_i) + w_{false} \cdot (1 - y_i) \cdot \log(1 - \hat{y}_i))$$

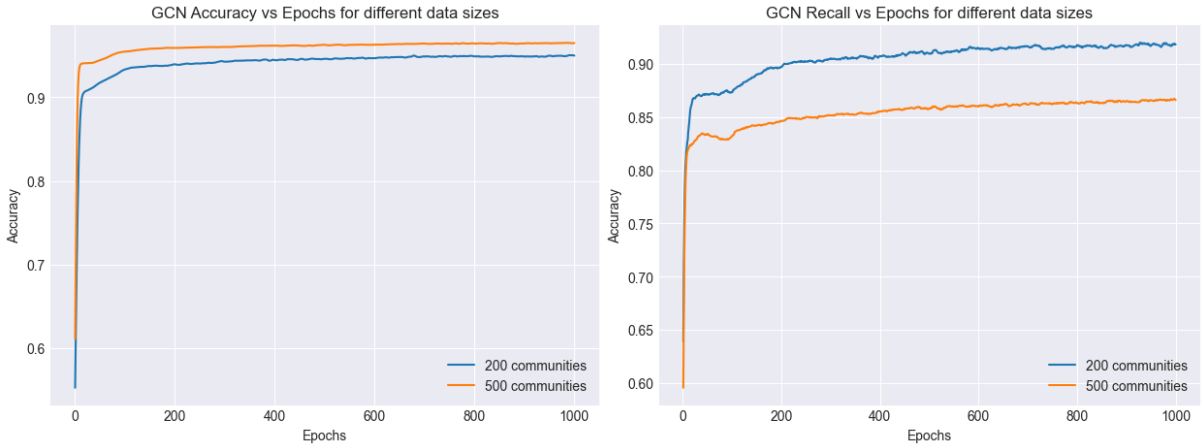
By setting Wtrue >> Wfalse, we can make the model learn how to predict 1s as well as 0s. Experiments have shown that Wtrue = 1.2 and Wfalse = 0.01 makes the model reach high accuracy and recall of 95.04% and 91.98% respectively(200 communities , 50% edge sampling).

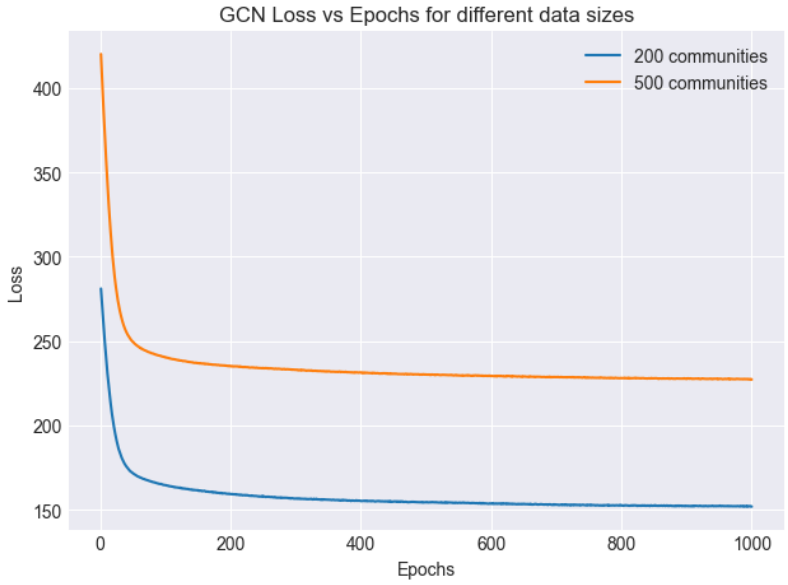
These hyper parameters can be tuned to achieve specific goals like getting higher accuracy or recall at the cost of the other.

Dataset size

We also test the performance on 2 dataset sizes:

1. 200 communities and their respective nodes
2. 500 communities and their respective nodes



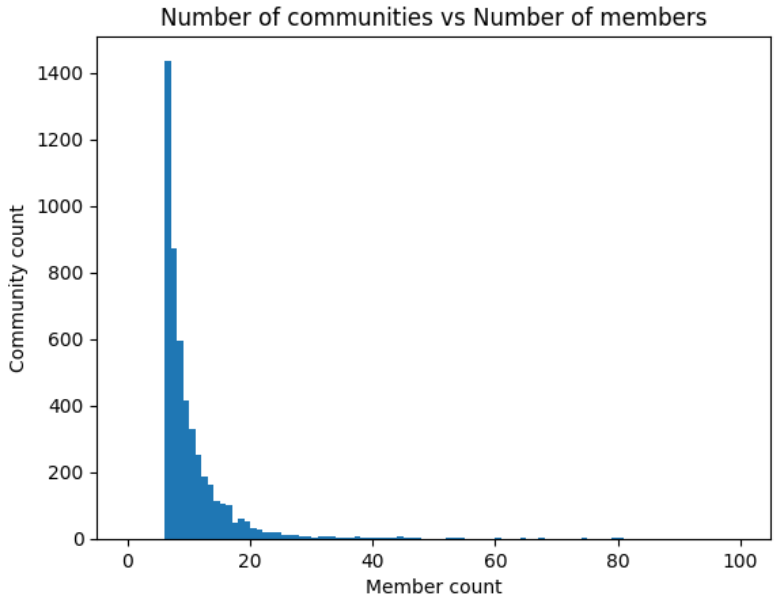


Observations

1. Accuracy is higher than recall.
   - We can to use a statistical method to determine the wtrue and wfalse for the loss function which makes their relative weights a function of the sparsity of the output matrix.
2. Loss functions have similar profiles
3. We also achieve high accuracy and recall

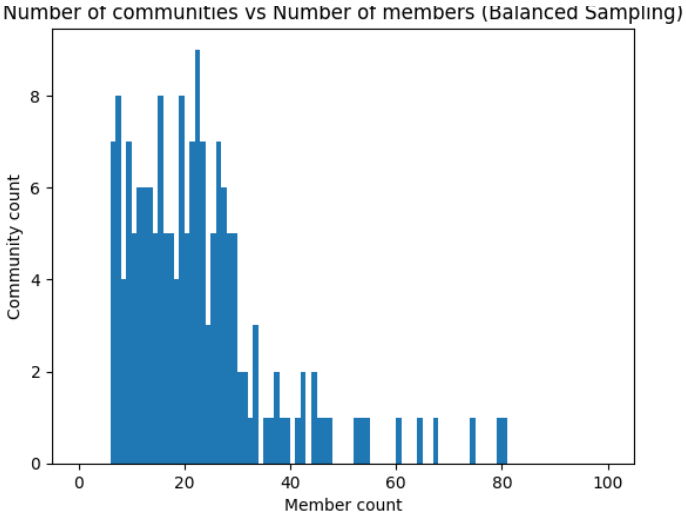
Sampling

The histogram shows the number of communities vs member count. The frequency diminishes exponentially as member count increases. In other words, the number of communities with 6 members is very high while the number of communities with 20 or more member is very low.
To make our models generalise well to communities with larger member counts we test use frequency based sampling to while sampling communities. The probability of a community being selected in the train/test/val subset is inversely proportional to the number of communities with that number. This way we can create a balanced dataset.



We evaluate the models training on the two types of sampling.

1. Balanced sampling: community selection probability is inversely proportional to the number of communities with the same number of members.
   - This method creates a distribution closer to uniform over different member counts. I.e. number of communities with lower number of members is comparable to that with higher number of members
2. Random sampling: Select communities randomly
   - This will cause the distribution to be the same like the one shown in the above figure



Observations

1. Accuracy and recall for random sampling is higher than balanced sampling
   - This is because the most communities have a lesser number of members in random sampling
   - This makes it easier for the model to make better predictions
   - Balanced sampling reduces the training accuracy and recall however we expect the model trained on balanced classes to perform better at making predictions for bigger communities
2. Loss profiles in both cases are similar