Mining & Learning on Graphs

Node Classification

Yu Wang, Ph.D.
Assistant Professor
Computer and Information Science
University of Oregon

CS 410/510 - Fall 2024

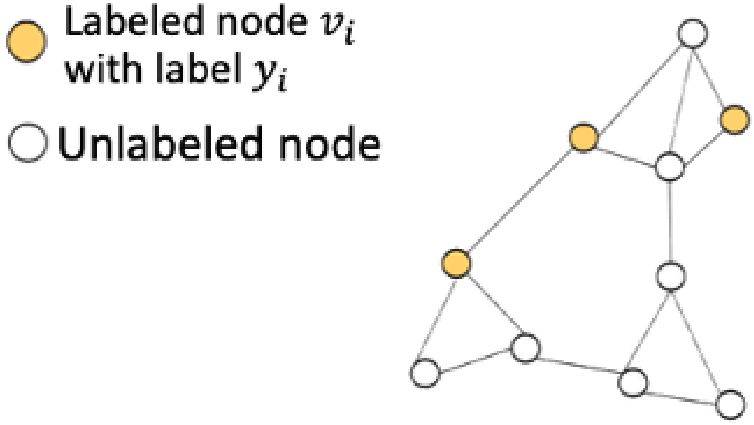
Node Classification - Summary

Data

Model

Loss

Optimization



Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X}) = (\mathbf{A}, \mathbf{X})$

Adjacency Matrix $\mathbf{A}$

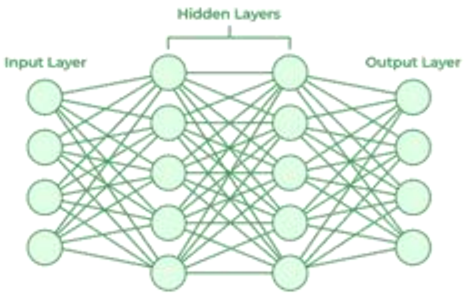
Node Feature Matrix $\mathbf{X}$

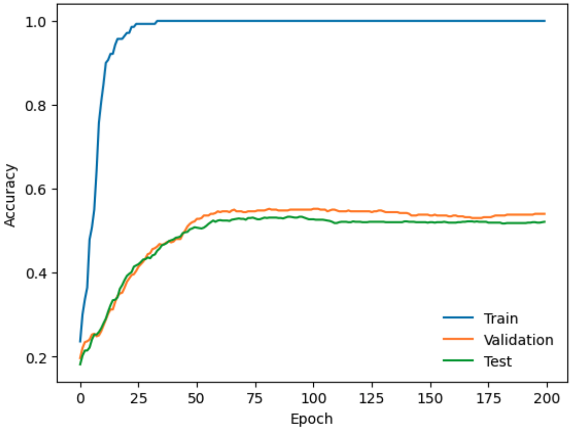
Labeled Data $\mathcal{D}_L = (\mathcal{V}_L, \mathcal{Y}_L)$

Unlabeled Data $\mathcal{D}_U = (\mathcal{V}_U)$

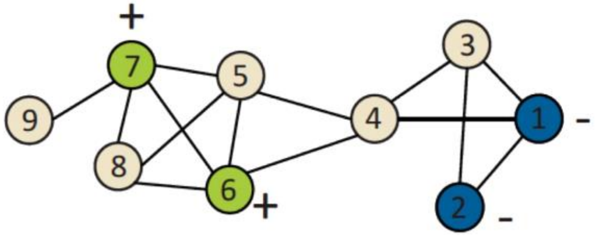
Node Classification - Summary

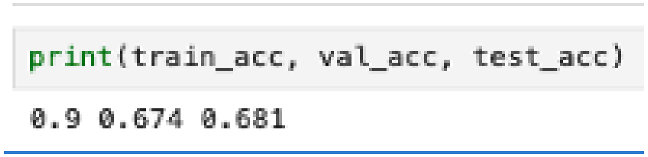
Data | **Model** | **Loss** | **Optimization**



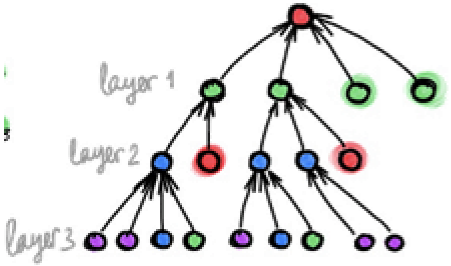


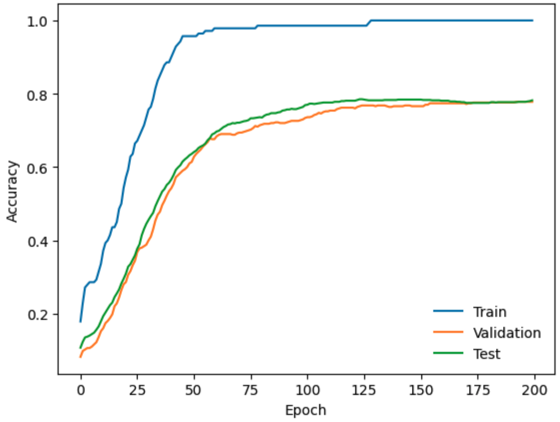
Test Performance:
0.531





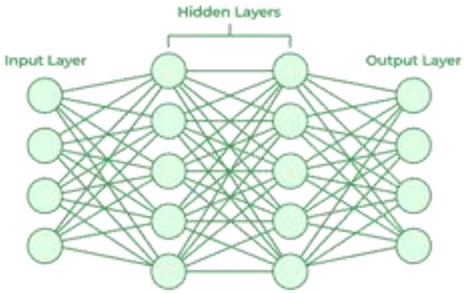
```
print(train_acc, val_acc, test_acc)
0.9 0.674 0.681
```

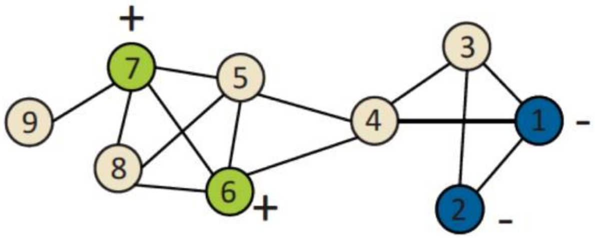


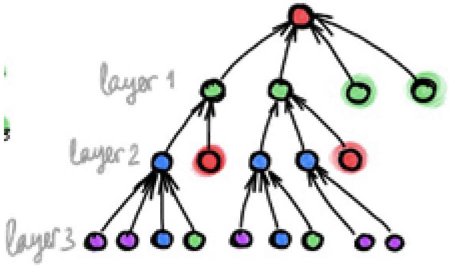


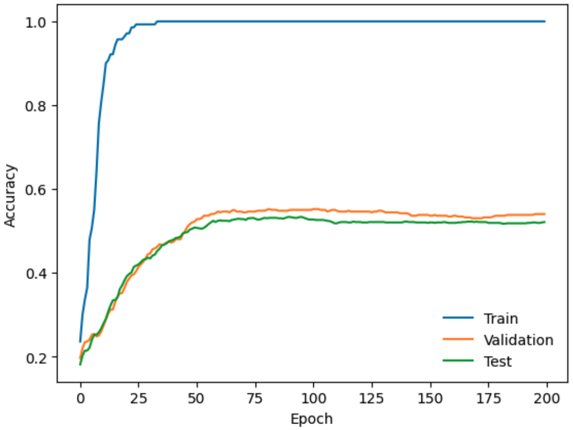
Test Performance:
0.788

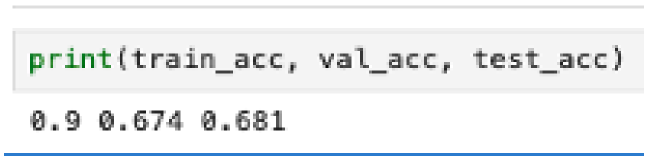
Node Classification – Dive Deep





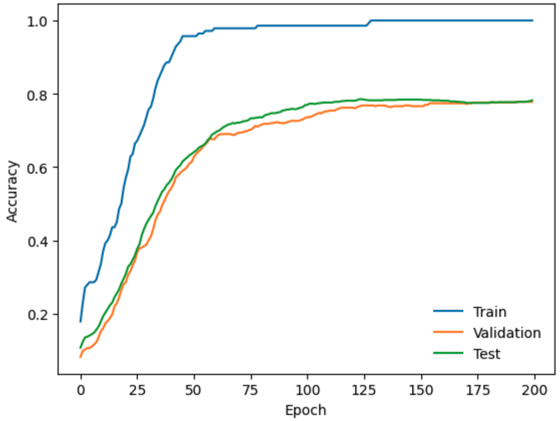






```
print(train_acc, val_acc, test_acc)
```

```
0.9 0.674 0.681
```



Why there is such a performance difference?

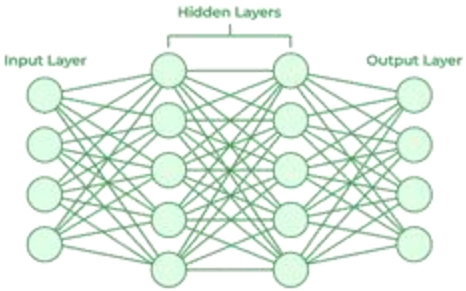
Because Graph Machine Learning Model is advanced, it is better, it uses graph-structure

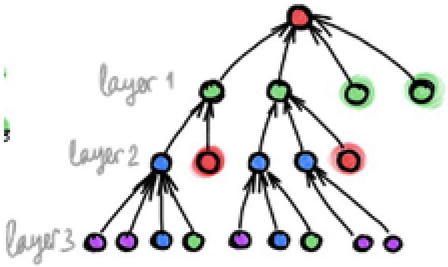
But still why?

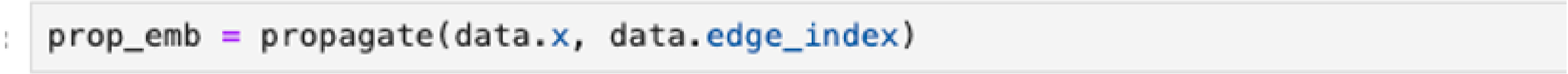
Node Classification – GNN Embedding Space

MLP: X, Y

GNN: X, A, Y



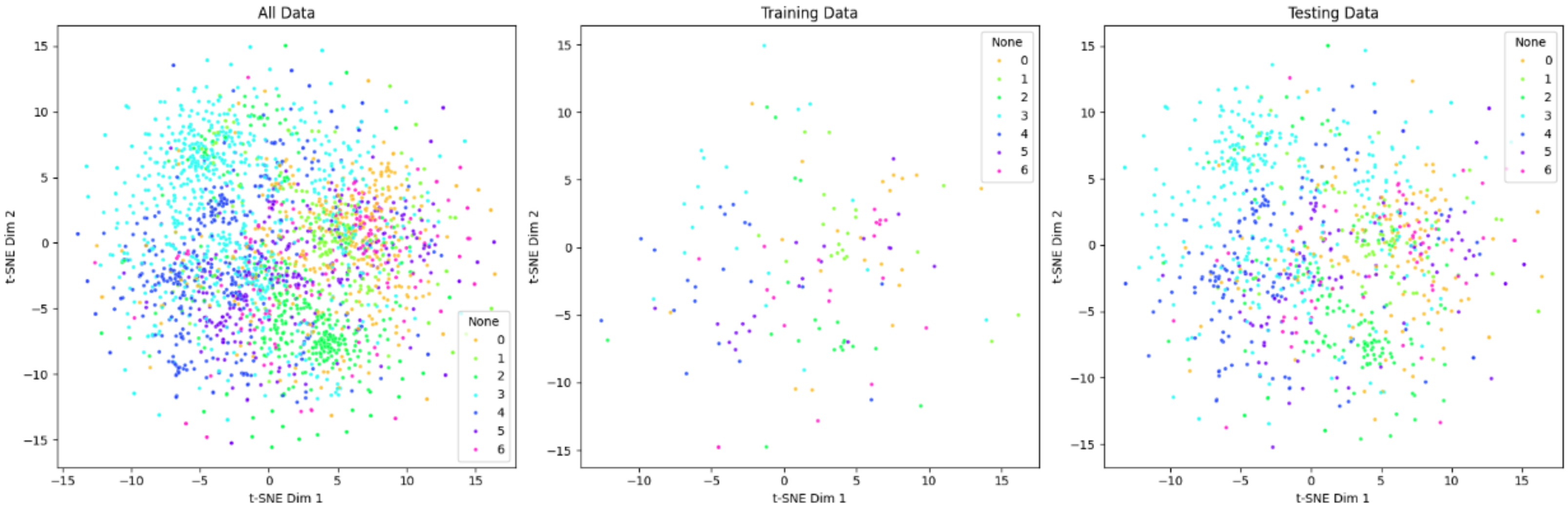




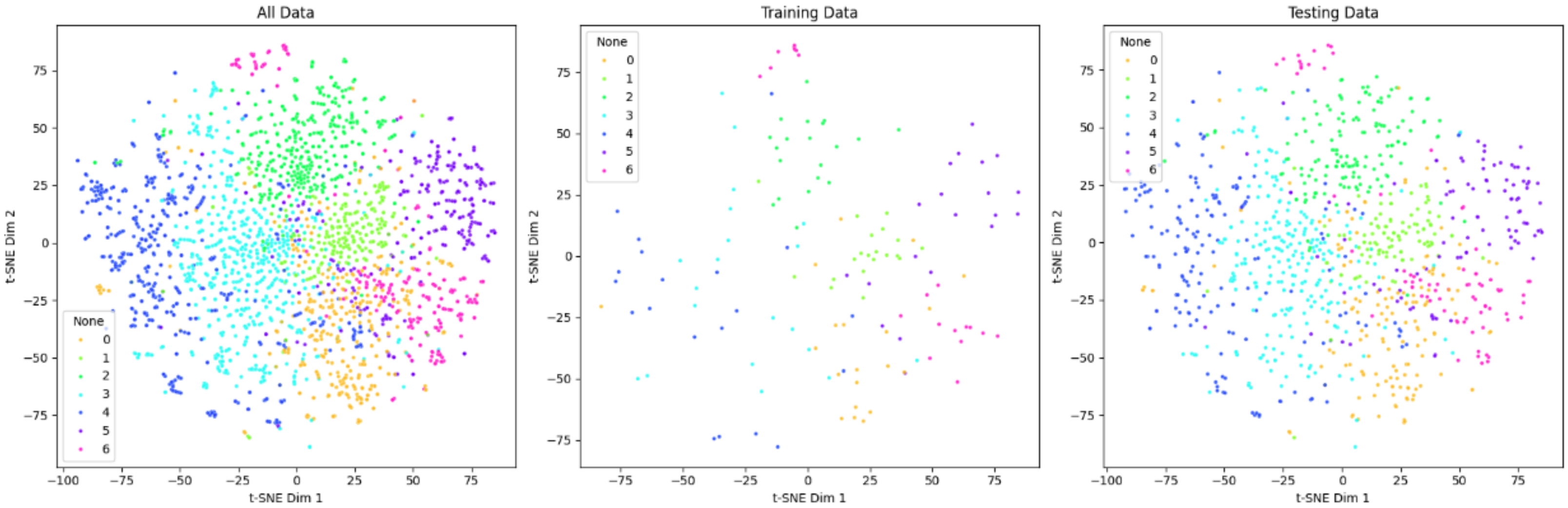
```
prop_emb = propagate(data.x, data.edge_index)
```

Node Classification –Input Space

MLP

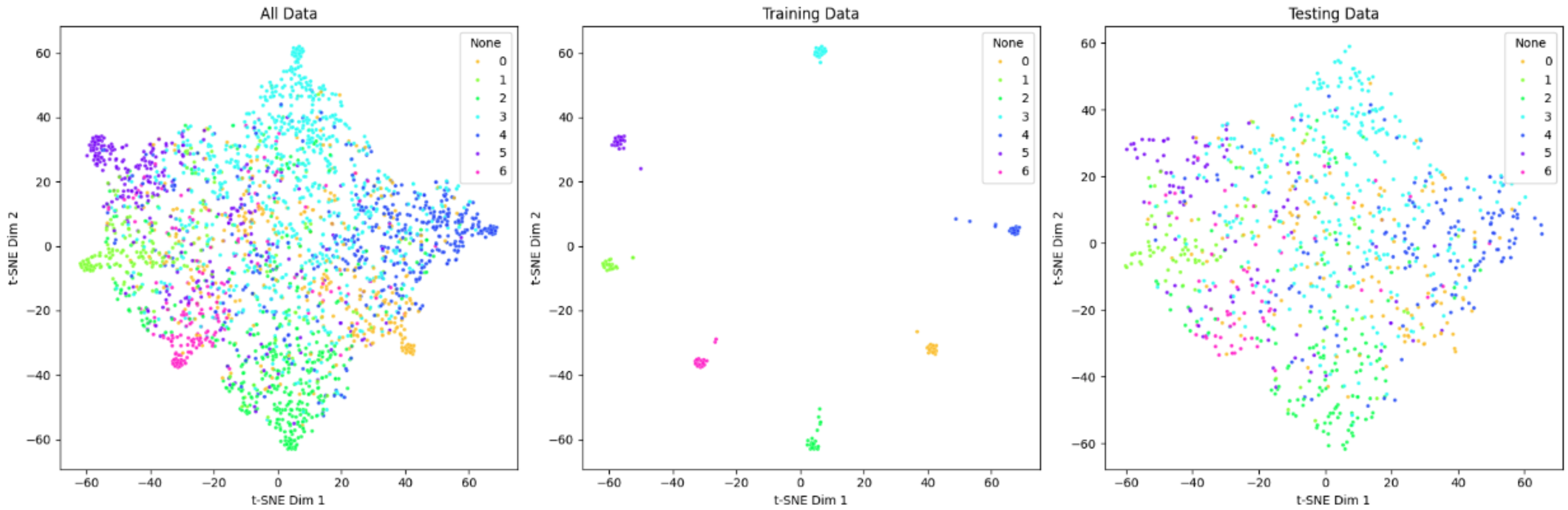


GNN



Node Classification –Embedding Space

MLP

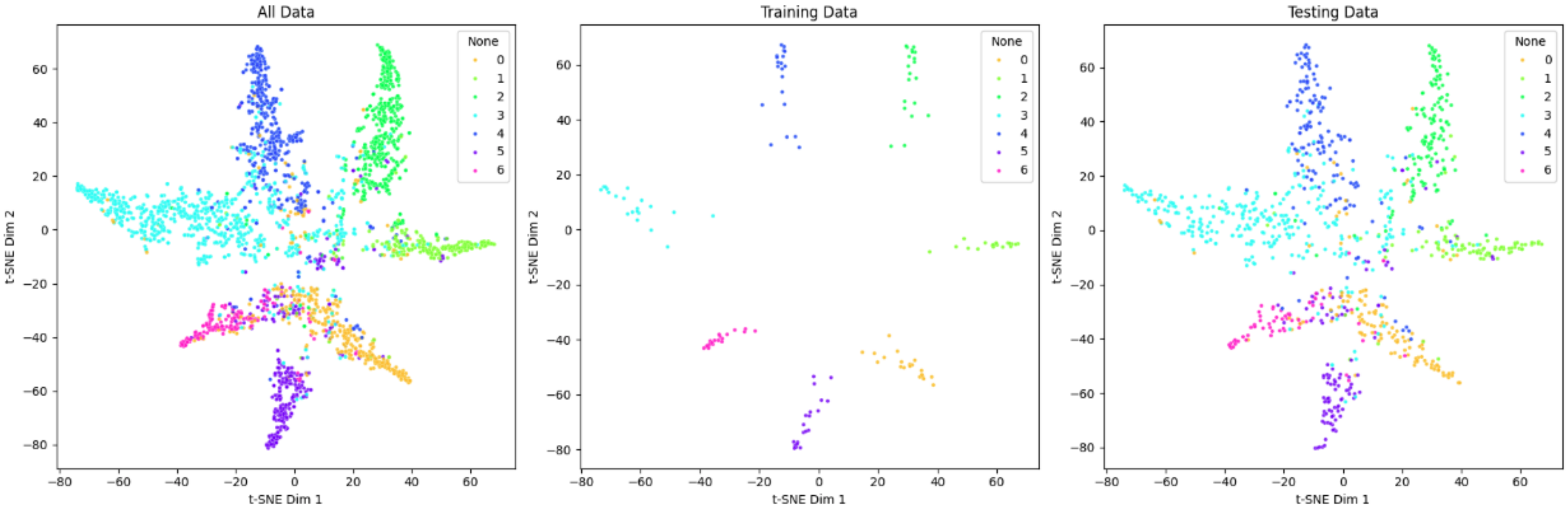


All Data

Training Data

Testing Data

GNN



All Data

Training Data

Testing Data

Node Classification – Advanced topics

1. **Homophily vs Heterophily**

2. **Number of Layers – Over-smoothing**

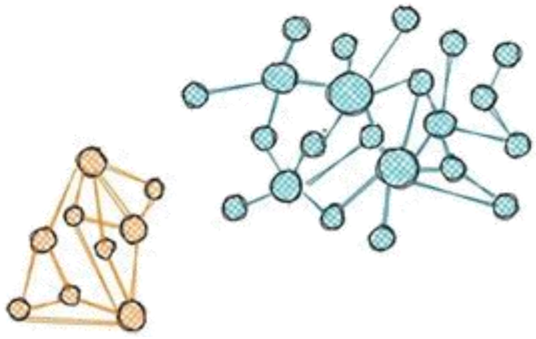
Node Classification – Homophily vs Heterophily

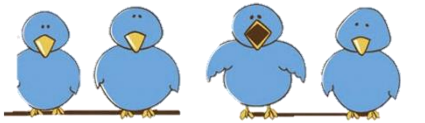
1. Homophily vs Heterophily

2. Number of Layers – Over-smoothing

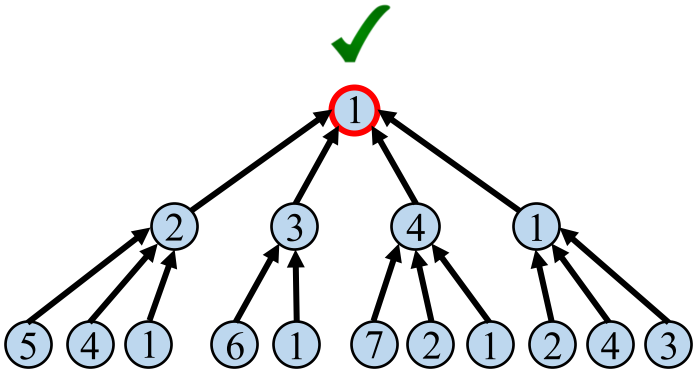
Node Classification – Homophily vs Heterophily

Homophily

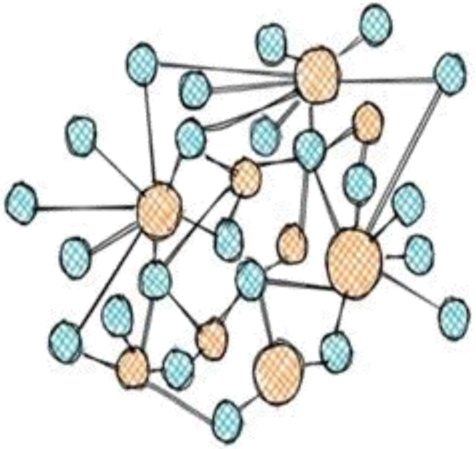


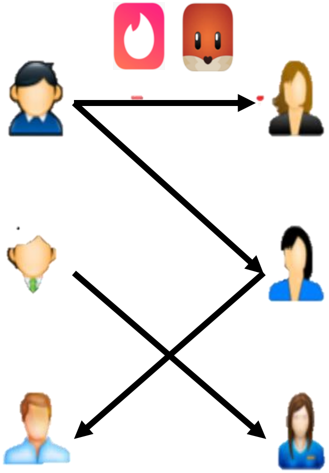


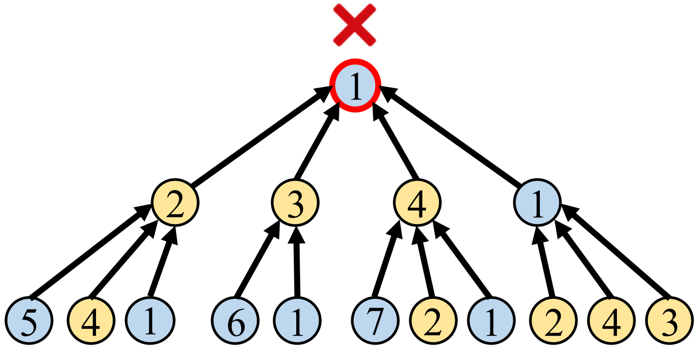
Birds of a feather flock together



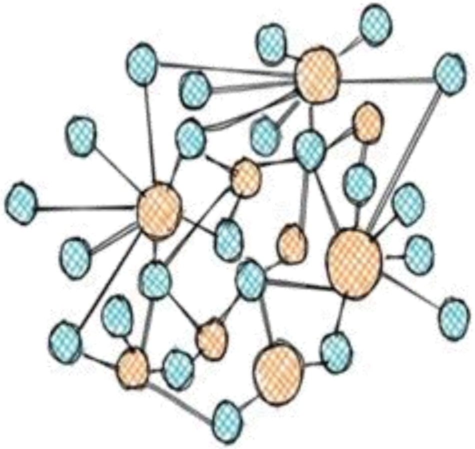
Heterophily





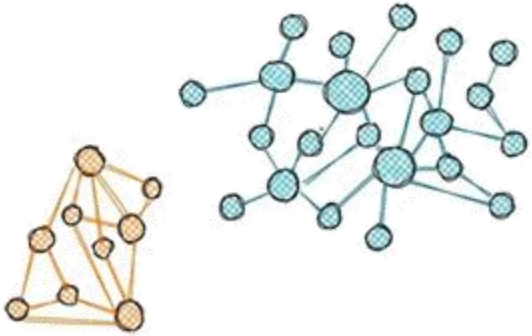


Node Classification – Homophily vs Heterophily



How to measure the homophily/heterophily

$$\phi_{\text{local}} = \frac{\text{\# Neigbors from the same class}}{\text{\# Neighbors}}$$

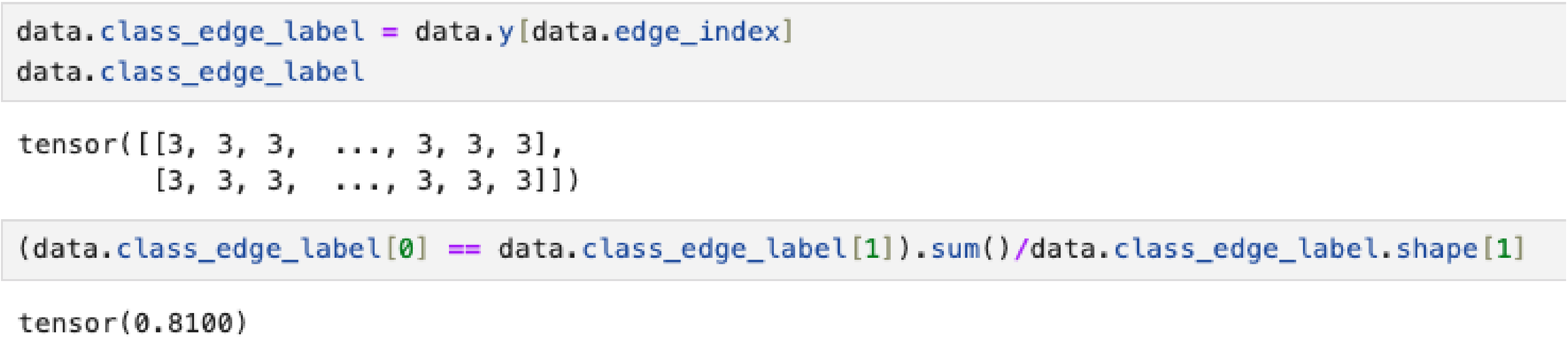


$$\phi_{\text{global}}^{1} = \sum_{n=1}^{N} \phi_{\text{local}} / N$$

$$\phi_{\text{global}}^{2} = \frac{\text{\#Edges between nodes of the same class}}{\# \, Edges}$$

Node Classification – Homophily vs Heterophily

How to measure the homophily/heterophily



```
data.class_edge_label = data.y[data.edge_index]
data.class_edge_label
```

```
tensor([[3, 3, 3,  ..., 3, 3, 3],
        [3, 3, 3,  ..., 3, 3, 3]])
```

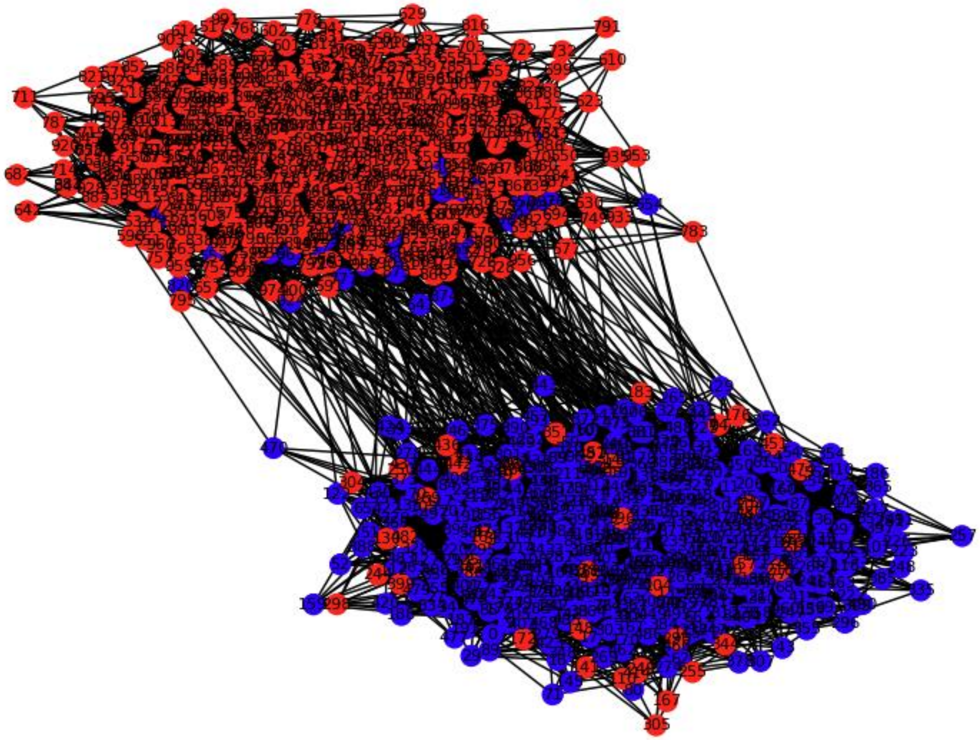
```
(data.class_edge_label[0] == data.class_edge_label[1]).sum()/data.class_edge_label.shape[1]
```

```
tensor(0.8100)
```

Very high homophily

Node Classification – Homophily vs Heterophily

How is the performance changing when homophily/heterophily changes?



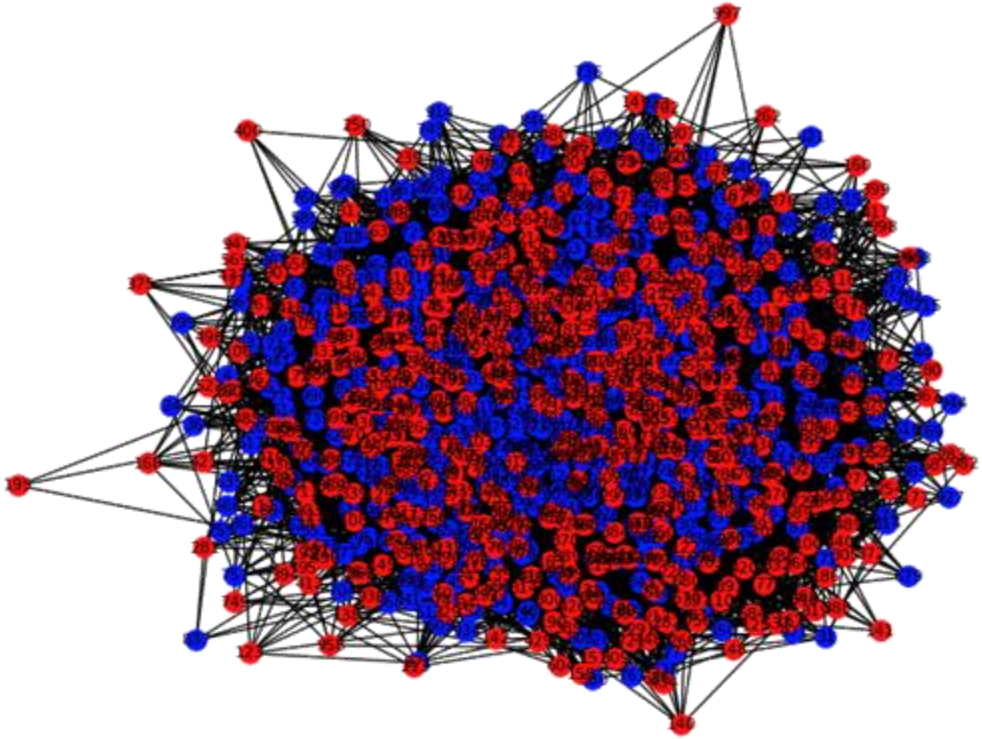
Stochastic Block Model

$$\begin{bmatrix} 0.03 & 0.001 \\ 0.001 & 0.03 \end{bmatrix}$$

Homophily: 0.9679660362794288

Node Classification – Homophily vs Heterophily

How is the performance changing when homophily/heterophily changes?

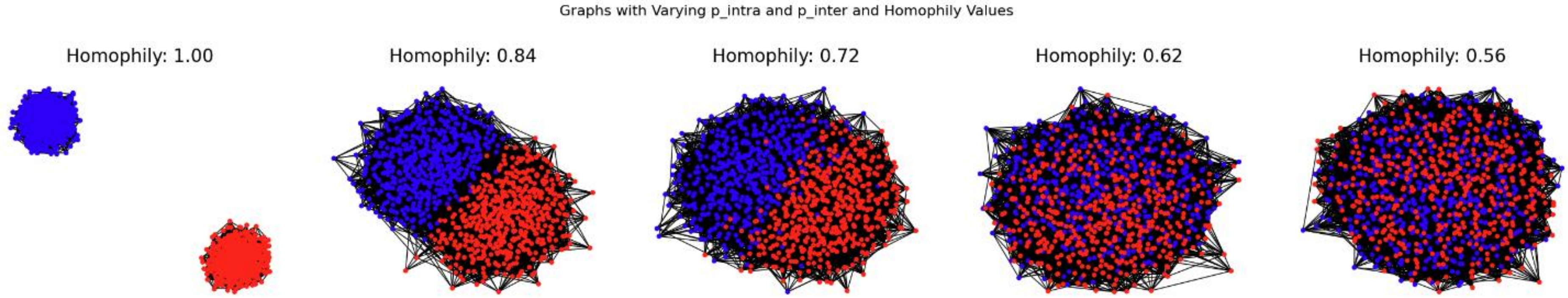


Stochastic Block Model

$$\begin{bmatrix} 0.001 & 0.03 \\ 0.03 & 0.001 \end{bmatrix}$$

Homophily: 0.0350115001277792

Node Classification – Homophily vs Heterophily

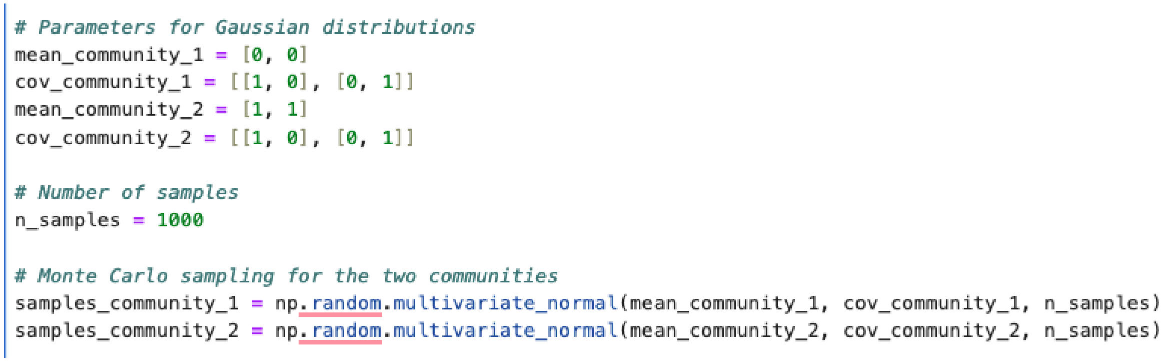


Now we have our structure, but how about feature?

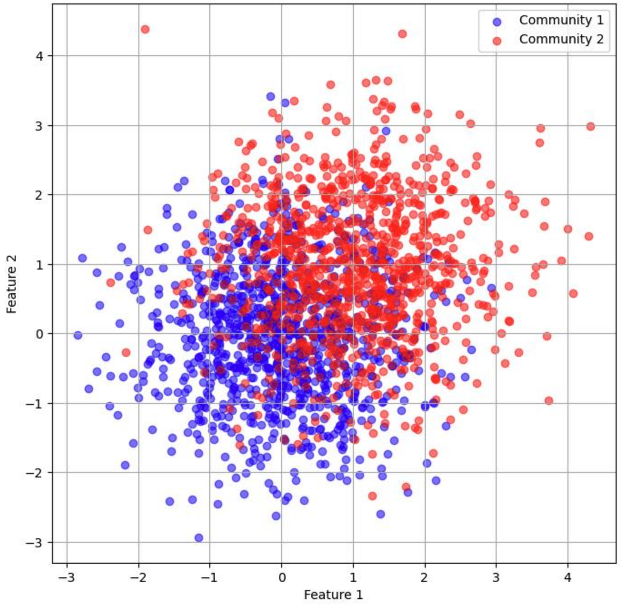
Multi-variant Gaussian Distribution

Node Classification – Homophily vs Heterophily

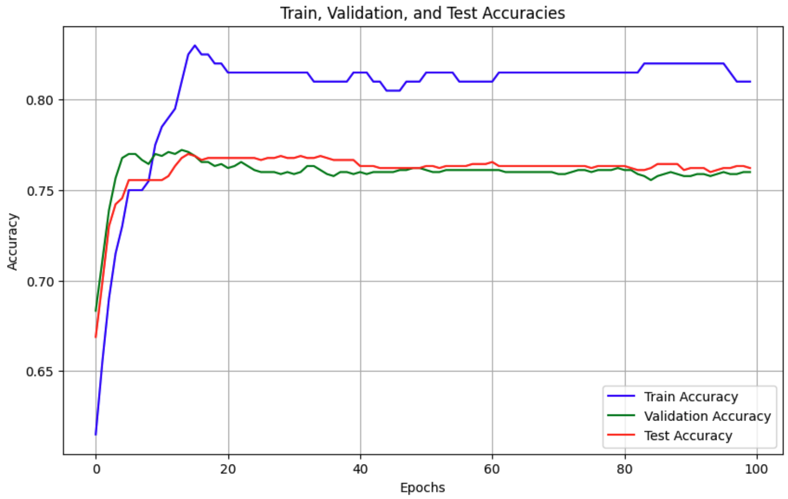
2D Gaussian Distribution



```
# Parameters for Gaussian distributions
mean_community_1 = [0, 0]
cov_community_1 = [[1, 0], [0, 1]]
mean_community_2 = [1, 1]
cov_community_2 = [[1, 0], [0, 1]]

# Number of samples
n_samples = 1000

# Monte Carlo sampling for the two communities
samples_community_1 = np.random.multivariate_normal(mean_community_1, cov_community_1, n_samples)
samples_community_2 = np.random.multivariate_normal(mean_community_2, cov_community_2, n_samples)
```

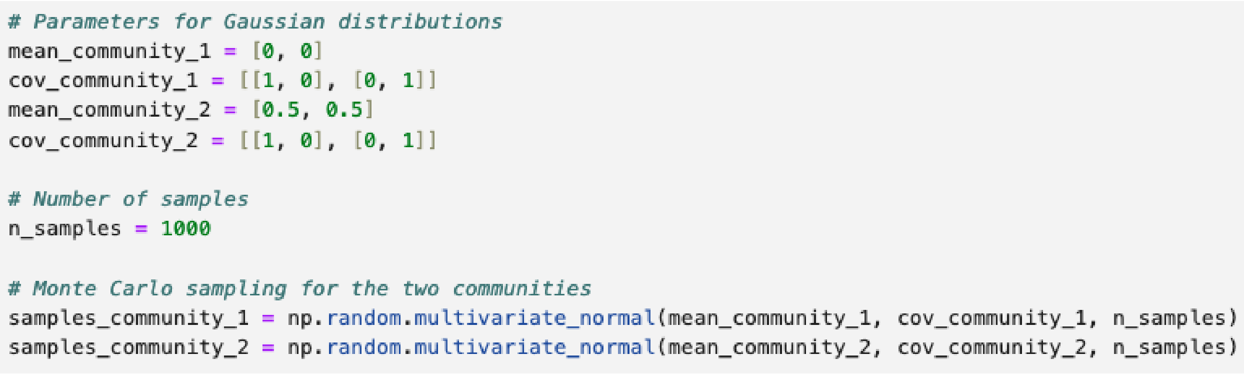


If you use MLP, what performance would you get?

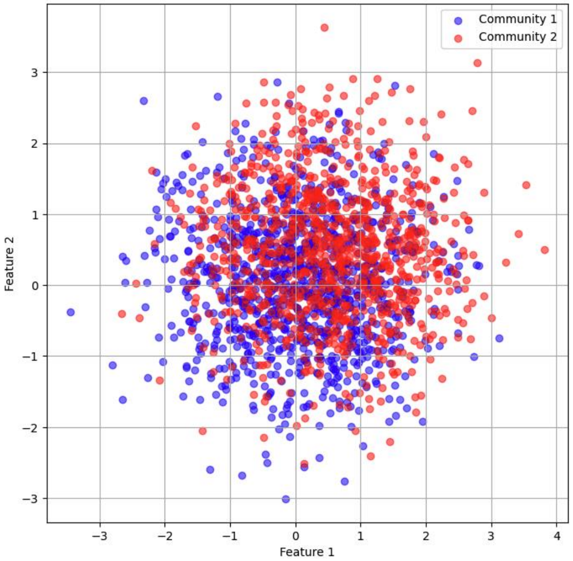


Node Classification – Homophily vs Heterophily

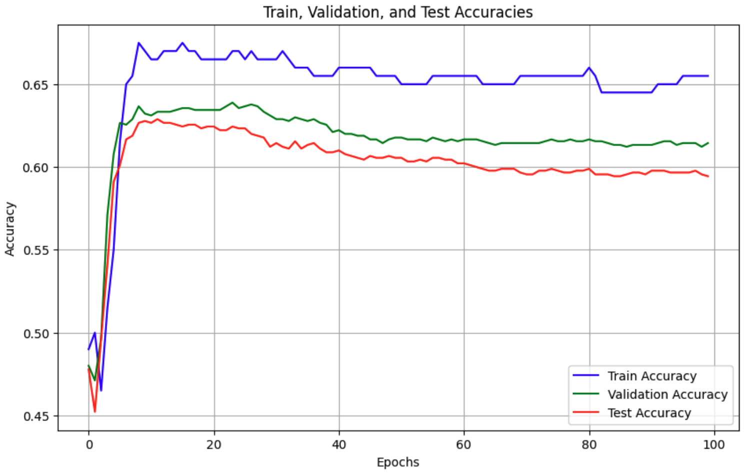
2D Gaussian Distribution



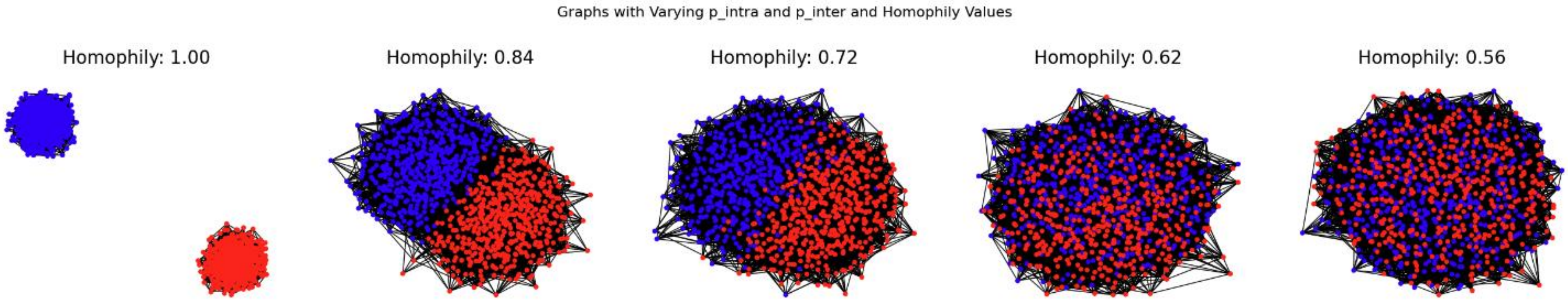
```
# Parameters for Gaussian distributions
mean_community_1 = [0, 0]
cov_community_1 = [[1, 0], [0, 1]]
mean_community_2 = [0.5, 0.5]
cov_community_2 = [[1, 0], [0, 1]]

# Number of samples
n_samples = 1000

# Monte Carlo sampling for the two communities
samples_community_1 = np.random.multivariate_normal(mean_community_1, cov_community_1, n_samples)
samples_community_2 = np.random.multivariate_normal(mean_community_2, cov_community_2, n_samples)
```



If you use MLP, what performance would you get?



Node Classification – Homophily vs Heterophily


Graphs with Varying p_intra and p_inter and Homophily Values

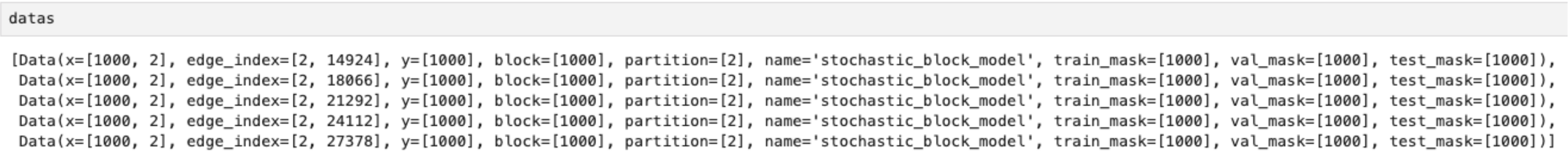
Homophily: 1.00

Homophily: 0.84

Homophily: 0.72

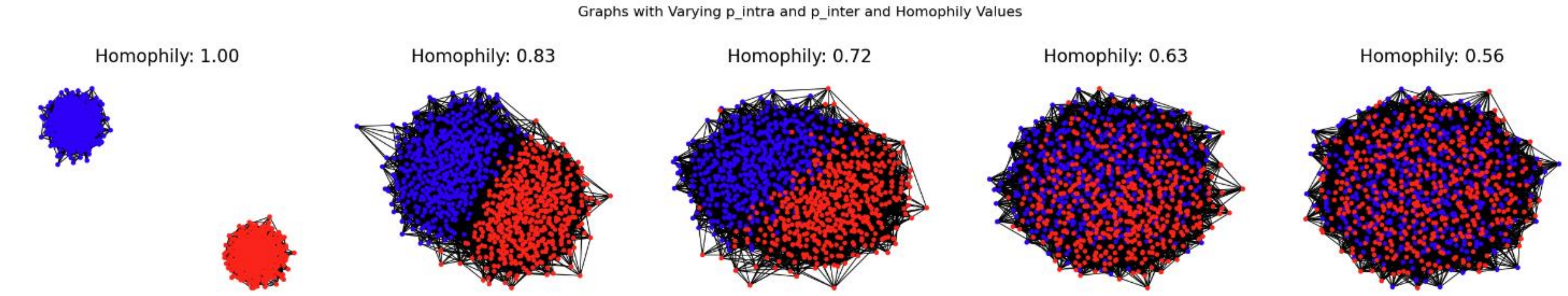
Homophily: 0.62

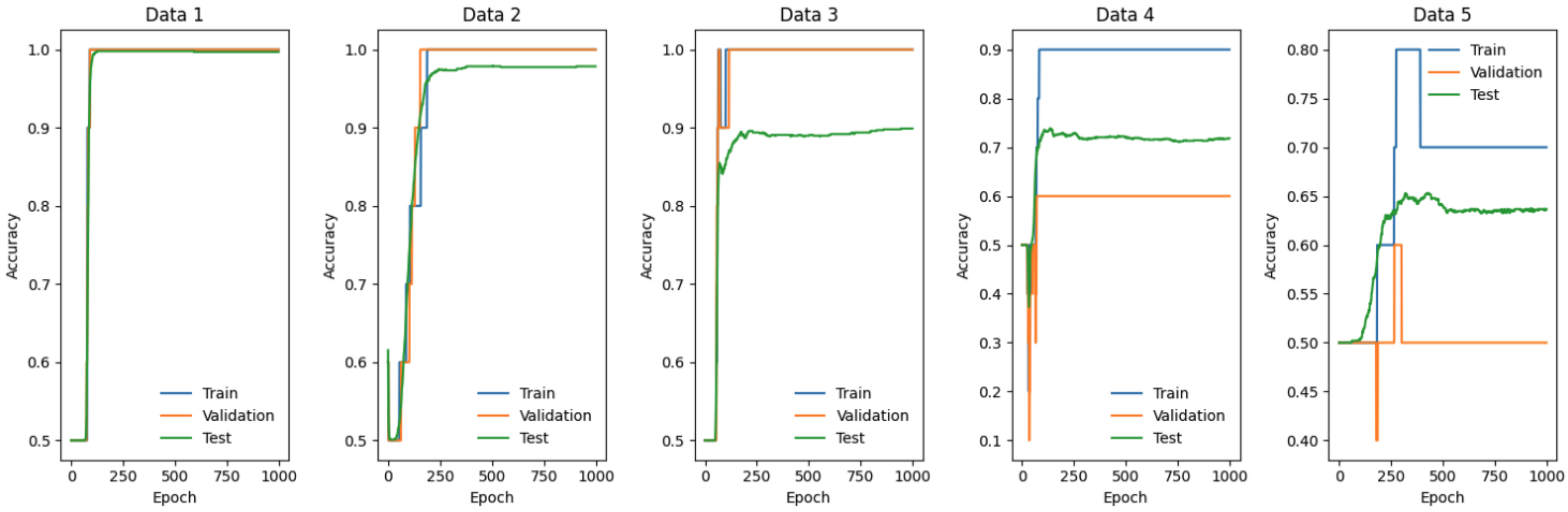
Homophily: 0.56


```
datas
```

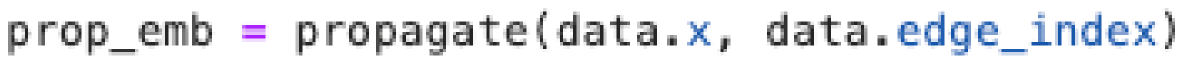
```
[Data(x=[1000, 2], edge_index=[2, 14924], y=[1000], block=[1000], partition=[2], name='stochastic_block_model', train_mask=[1000], val_mask=[1000], test_mask=[1000]),
 Data(x=[1000, 2], edge_index=[2, 18066], y=[1000], block=[1000], partition=[2], name='stochastic_block_model', train_mask=[1000], val_mask=[1000], test_mask=[1000]),
 Data(x=[1000, 2], edge_index=[2, 21292], y=[1000], block=[1000], partition=[2], name='stochastic_block_model', train_mask=[1000], val_mask=[1000], test_mask=[1000]),
 Data(x=[1000, 2], edge_index=[2, 24112], y=[1000], block=[1000], partition=[2], name='stochastic_block_model', train_mask=[1000], val_mask=[1000], test_mask=[1000]),
 Data(x=[1000, 2], edge_index=[2, 27378], y=[1000], block=[1000], partition=[2], name='stochastic_block_model', train_mask=[1000], val_mask=[1000], test_mask=[1000])]
```

Node Classification – Homophily vs Heterophily

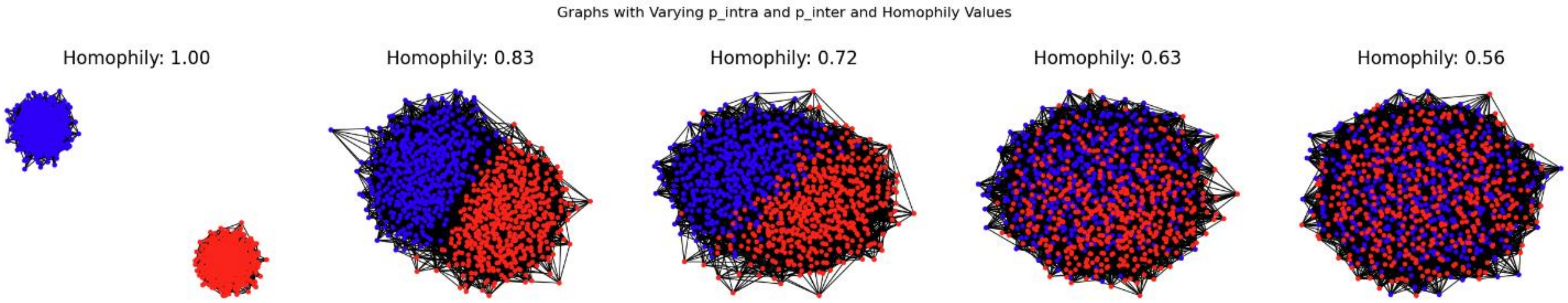

Graphs with Varying p_intra and p_inter and Homophily Values

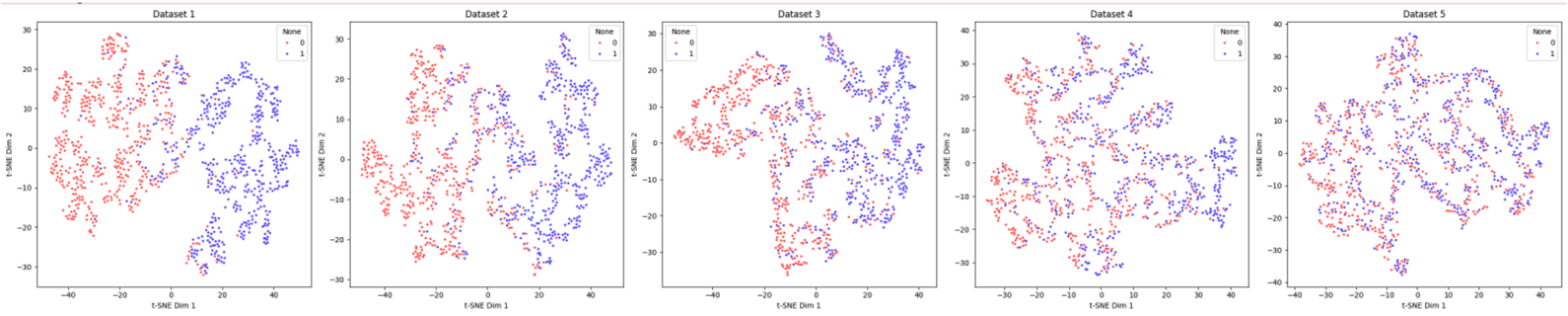


Node Classification – Homophily vs Heterophily

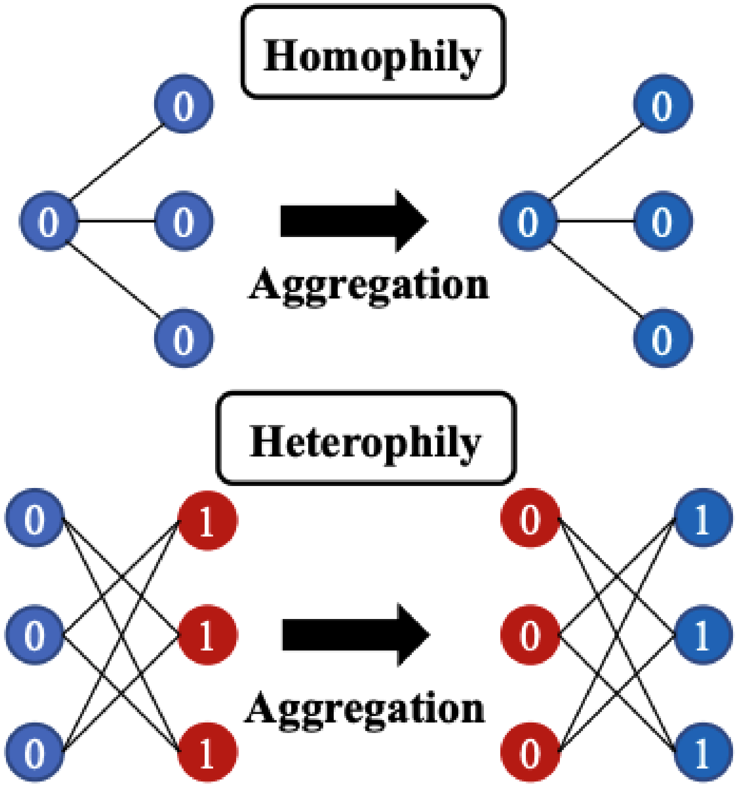


```
prop_emb = propagate(data.x, data.edge_index)
```





Node Classification – Homophily vs Heterophily



Can we classify correctly?

Is Homophily a Necessity for Graph Neural Networks?

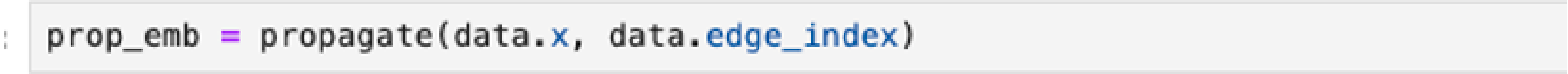
Yao Ma, Xiaorui Liu, Neil Shah, Jiliang Tang

Node Classification – Over-smoothing

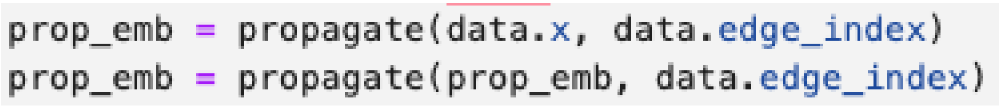
1. **Homophily vs Heterophily**

2. **Number of Layers – Over-smoothing**

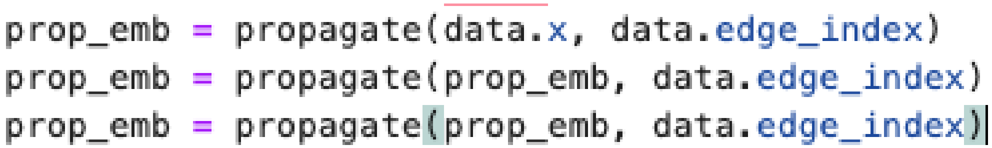
Node Classification – Over-smoothing



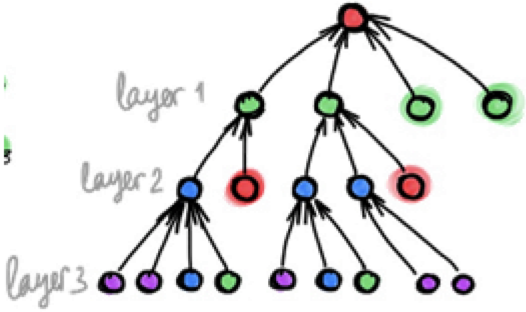
```
prop_emb = propagate(data.x, data.edge_index)
```



```
prop_emb = propagate(data.x, data.edge_index)
prop_emb = propagate(prop_emb, data.edge_index)
```

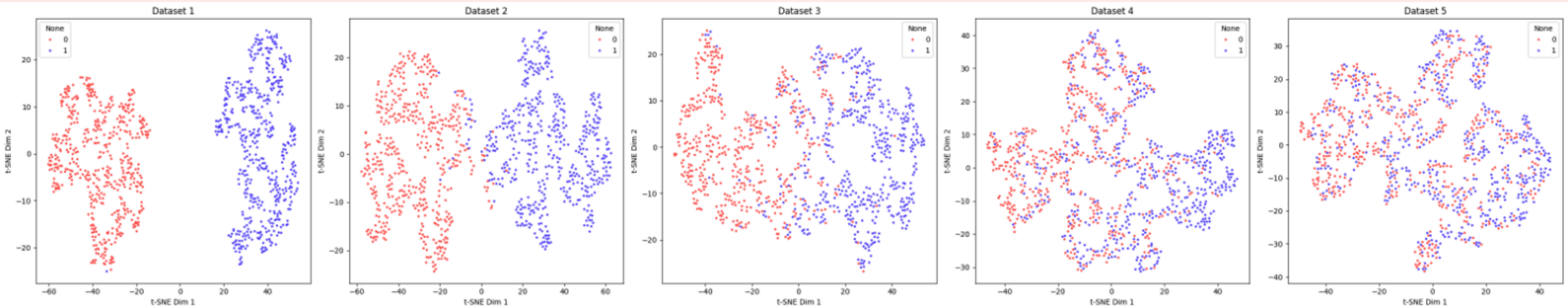


```
prop_emb = propagate(data.x, data.edge_index)
prop_emb = propagate(prop_emb, data.edge_index)
prop_emb = propagate(prop_emb, data.edge_index)
```

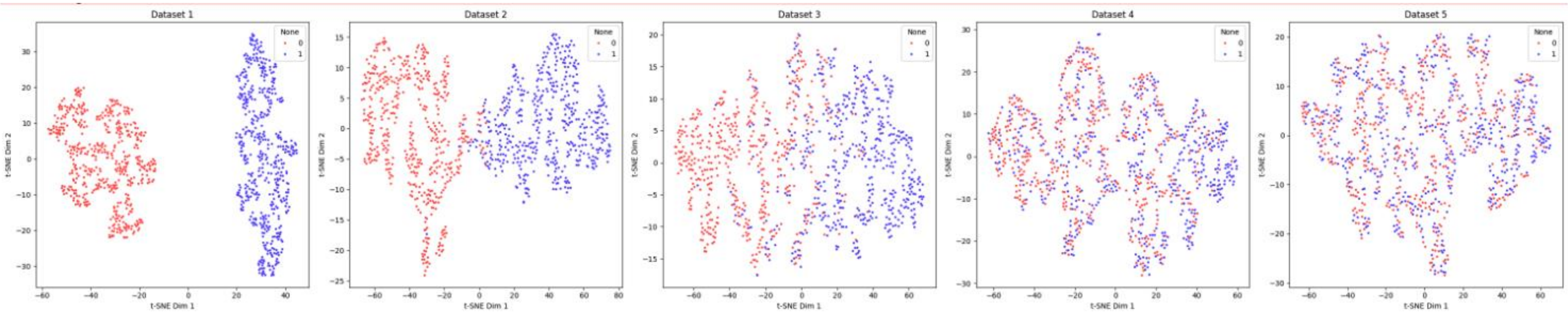


Node Classification – Over-smoothing

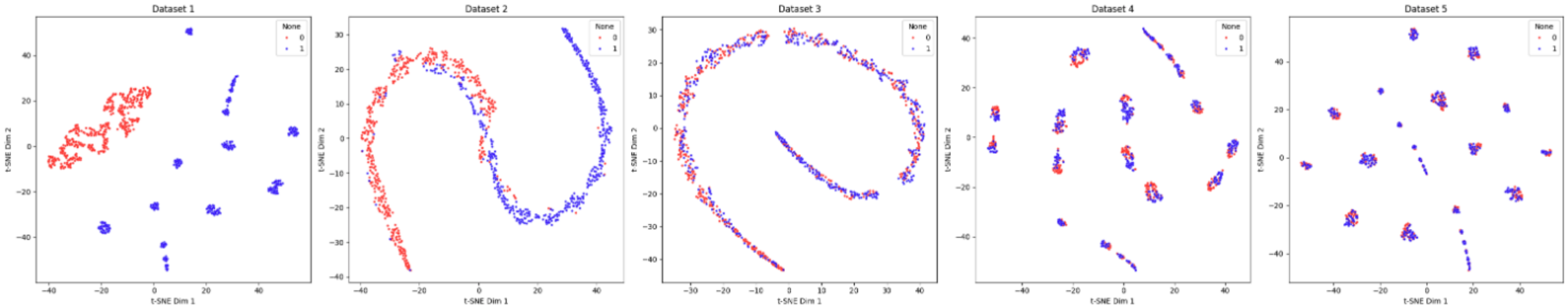
Layer 1



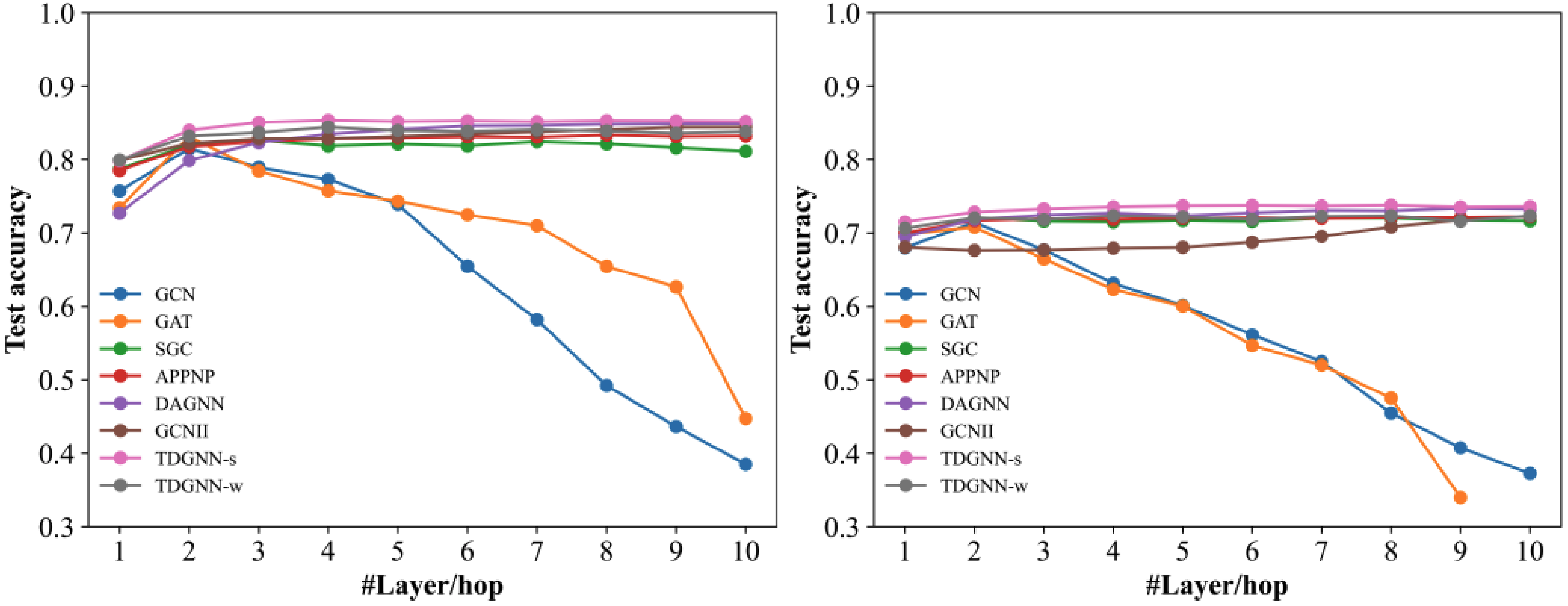
Layer 2



Layer 6



Node Classification – Over-smoothing



(a) Cora

(b) Citeseer

Any Question?