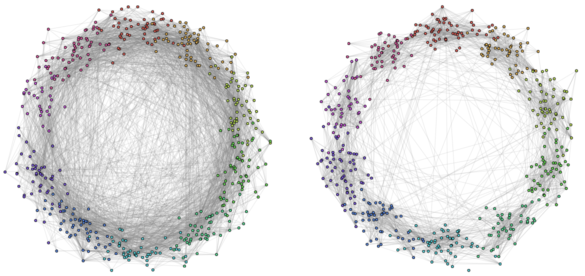
A. Synthetic Dataset

For generating synthetic data, we split nodes to equal size classes $1, 2, ..., k$. At each iteration, we add new node v with a random class to the graph. The probability of connecting v to existing node u is the following:

- if $c_v = c_u$: $p_{v \rightarrow u} = d_u \times \mathit{homophily}$
- if $c_v \neq c_u$: $p_{v \rightarrow u} = d_u \times \mathit{homophily} \times w_{|c_v - c_v|}$

Where c_v is class of node v and $w_{|c_v - c_v|}$ is weight of connecting two different classes with *distance* $|c_v - c_u|$. By *distance* we mean shortest distance of two classes on a circle, starting from class 1 to k respectively. The weight is exponentially decreasing with increasing of *distance*. For example if we have 6 classes and $w_{|1-4|} = 1$, then $w_{|1-3|} = w_{|1-5|} = 2$ and $w_{|1-2|} = w_{|1-6|} = 4$. We normalize w such that $\sum w_i = 1$. We use $k = 10$.

The final probability of connecting v to u is normalized over all existing nodes: $p_{v \rightarrow u} = \frac{p_{v \rightarrow u}}{\sum_i p_{v \rightarrow i}}$. Since $\sum w_i = 1$, based on (Karimi et al., 2017), sampling edges proportional to $p_{v \rightarrow u}$ at each iteration, would give us a graph with desired homophily and Barabasi-Albert degree distribution. Figure 6, shows two graphs with same number of nodes and edges, but with different homophily coefficients. As expected, edges of 6a are between different classes, while the edges of 6b are within classes.



(a) Homophily = 0.1

(b) Homophily = 0.9

Figure 6: Synthetic data generated for (a) $\mathit{homophily} = 0.1$ and (b) $\mathit{homophily} = 0.9$. Both graphs have 500 nodes, 2798 edges and 10 equal size classes. Classes are specified with different color of node.

The feature values of the nodes are sampled from 2D Gaussians. In particular, each node class has its own Gaussian. The means of the Gaussians can be best described in polar coordinates: Each mean has radius 300 and angle $\frac{2\pi}{10} \times$ (class id). The covariance matrix of each class is $3500 \times \mathrm{diag}([7, 2])$, that is rotated by angle $\frac{2\pi}{10} \times$ (class id). The synthetic datasets are available for download via github.

B. Future Work

Because our work has general applicability to a large number of graph tasks, we believe there is significant room to fine tune our model. In the spirit of Hamilton et al. (2017), we can extend our methods to large graphs (e.g. with millions of edges) that might not fit in computer memory. Such scalability would be vital for the practical usage of these methods. In this work, we utilize a weighted softmax output layer that acts as a regularization method on the final feature output of the model. We believe that different kinds of output layers may be better suited for different tasks. An easy extension of our work is to analyze how our model performs with a fully connected output layer. Similarly, to apply our model to graph classification, one can average all node representations at the output layer (e.g. using a set2set model) to feed into classification layers for the full graph. While we analyze the ability of our model to learn delta operators in this work, we think that it would be fascinating to visualize how each delta operator actually operates in the graph. Specifically, we want to observe which community edges the model is capturing. In future experiments, we hope to explore methods of visualizing these features.