Alexandre Zajac
zajac.alexandre@gmail.com

Lab session # 5
ALTEGRAD 2020

12/18/20

1 Question 1

Here we consider a graph G with 2 connected components, where each of the component is a Erdős-Rényi random graph with $n = 50$ nodes and $p = 0.1$. Such graphs contain nodes connected randomly, such that each edge is included in the graph with probability p independent from every other edge. In particular, the distribution of the degree of any vertex is binomial:

$$P(deg(v) = k) = \binom{n-1}{k} p^k (1-p)^{n-1-k} \tag{1}$$

On average, each connected component of our graph will have $\binom{n}{2}p$ edges, so around 122 edges. But, the edges are distributed randomly, and in a 1960 paper [1], Erdős and Rényi described that if $n * p = c > 1$, then a graph in $G(n, p)$ will almost surely have a unique giant component containing a positive fraction of the vertices. No other component will contain more than $O(log(n))$ vertices.

For our case, it means the connected components will probably have a giant cluster and a large number of small clusters of nodes with few edges. In this context, and since the edges are random, the embeddings inside and outside of the connected components will be **represented comparably**, since the nodes inside a single connected component are expected to be **sparse**. If the graph would have been denser ($p \to 1$), there would have been a much closer relationship between the nodes in a connected component compared to the ones between the 2 components.

2 Question 2

The Random Walk algorithm is a stochastic process, and its performance depends on the number of walks (n) to perform for each node and the length of the walk (L). When both n and L are big, it's expected for the random walk to perform better. For v_1 and v_2, the paths generated by the random walks will be more and more similar as we increase both n and L, but **the embeddings will not be necessarily close to each other** in the embedding space.

The random walk is not directly designed to detect similar structures, but rather to detect close relationships (equivalent of semantics for NLP). For example, if the group of nodes close to v_1 might represent romantic movies, and the one close to v_2 action movies, the **groups might have the same structure**, but romantic and action movies **aren't specially close** one to another.

3 Question 3

Here we consider a graph G as C_4, a graph on 4 nodes containing a single cycle through all nodes. We consider X being an all-ones 4 x 1 matrix. We define the $relu()$ function as

$$relu(x) = x^+ = max(0, x) \tag{2}$$

And to compute Z^1 we have:

$$Z_1 = relu(\hat{A} * Z^0 * W^2) \; and \; Z_0 = relu(\hat{A} * X * W^1) \tag{3}$$

We also have for $\hat{A}$:

$$\hat{A} = \tilde{D}^{-\frac{1}{2}} * \tilde{A} * \tilde{D}^{-\frac{1}{2}} \tag{4}$$

Note that here we first perform matrix multiplication with the W matrices, and apply the $relu()$ function element-wise on the matrices.

Let's first calculate the $\hat{A}$ normalized adjacency matrix. For a cyclic C_4 graph, we have:

$$A = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}; \tilde{A} = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}; \tilde{D} = I_4 * 3; \tilde{D}^{-\frac{1}{2}} = I_4 * \frac{\sqrt{3}}{3} \tag{5}$$

We then get for Z_0:

$$X * W_1 = \begin{bmatrix} 0.8 & -0.5 \\ 0.8 & -0.5 \\ 0.8 & -0.5 \\ 0.8 & -0.5 \end{bmatrix} ; Z_0 = \begin{bmatrix} 0.8 & 0 \\ 0.8 & 0 \\ 0.8 & 0 \\ 0.8 & 0 \end{bmatrix} \tag{6}$$

We finally get for Z_1:

$$Z_1 = \begin{bmatrix} 0.72 & 0 & 0.32 \\ 0.72 & 0 & 0.32 \\ 0.72 & 0 & 0.32 \\ 0.72 & 0 & 0.32 \end{bmatrix} \tag{7}$$

With these trainable parameters W_1 and W_2 for the Z_1 matrix, we can obverse that all the nodes have the same representation. This is expected since we have an unweighted not directed cyclic graph with only 4 nodes, so their representation is identical in this small context.

4 Question 4

When setting all the values of the features to a constant equal to 1, we obtain a classification accuracy of $\approx 28,57\%$, which is pretty low compared the 100% obtained with the previous initialization scheme. In the previous case, the feature matrix was I_n meaning that each node is assigned to a **different feature unit vector**. In the latter case, we set all the feature vectors of all the nodes to the **same all-ones feature vector**, and the model is not able to learn and extract meaningful relationships for classification from this poor initialization. We can conclude that initialization influences greatly the performance of the GNN classification.

References

[1] Paul Erdős and Alfréd Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5(1):17–60, 1960.