Arnaud Massenet
arnaudyoh@gmail.com

Lab session # 5
ALTEGRAD 2019

29/12/19

Question 1

We attain the maximum number of triangles when the graph is complete, that is when all vertices are connected to each other.
Without any self-loops the maximum number of edges an undirected graph with n nodes can have is $\frac{n(n-1)}{2}$.
Without any self-loops the maximum number of triangle an undirected graph with n nodes can have is $\binom{n}{3}$.
This is intuitive since each vertex is connected to all the other ones, we are essentially counting the number of ways we can pick 3 vertices from the graph.

Question 2

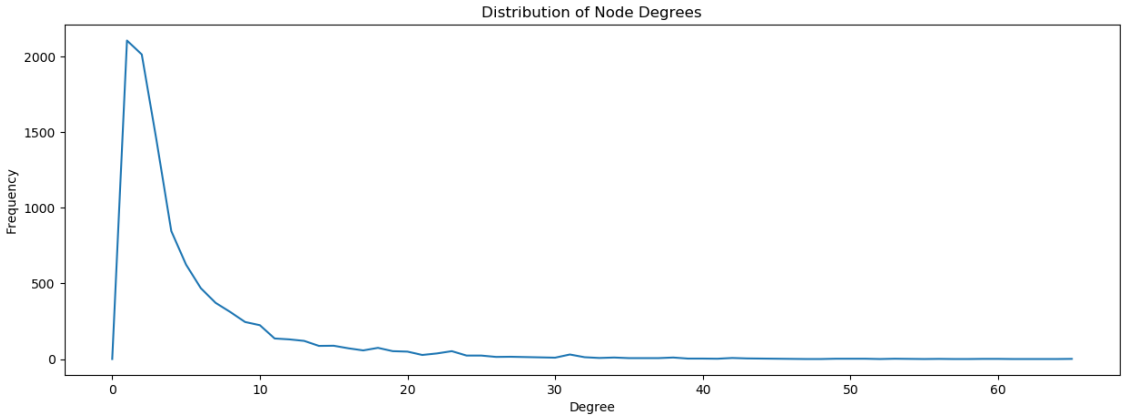

Figure 1: Node Degree Distribution

We can see that our distribution of degrees is skewed to the left, meaning that a large number of nodes have small degrees. Our degrees seems to follow a log-normal distribution.

Question 3

In spectral clustering, we are focusing on dividing graph elements. To do so, we are essentially trying to find multiple "min-cuts" in the graph where each cut defines a cluster. If we consider the graph where we have k subgraphs disconnected from each other representing the k clusters we want, then the smallest eigenvalues of the laplacian of this graph would be 0 and these eigenvalues would be linked to eigenvectors that carry information about the cuts we want to use for our clustering.

Since our Laplacian would have similar eigenvalues we can pick the smallest to get the eigenvectors that can give the best informations about the cuts to form the clustering.

In this spectral clustering problem, we are essentially solving the following problem:

$$\begin{aligned} \min_{x} \quad & x^T L x \\ \text{s.t.} \quad & x^T D x = 1 \end{aligned} \tag{1}$$

Where x is a vector, D is the diagonal degree matrix and L is our Laplacian.

Question 4

We compute the modularity of the graph clustering taken from this lab paper:

$$\begin{aligned} Q &= \sum_{c=1}^{3} \left[\frac{l_c}{10} - \left(\frac{d_c}{20} \right)^2 \right] \\ &= \left[\frac{1}{10} - \left(\frac{2}{20} \right)^2 + \frac{3}{10} - \left(\frac{7}{20} \right)^2 + \frac{5}{10} - \left(\frac{11}{20} \right)^2 \right] \quad = 0.465 \end{aligned} \tag{2}$$

Question 5

We can simply use the provided 3 node path graph and add a singleton node. Now we have one graph of 3 nodes and another one with 4 nodes, they are clearly not isomorphic. As there is no path from the singleton node to the other 3 nodes, the shortest path kernel's representation is the same but we do not have isomorphic graphs. Both graphs would get a shortest path

Question 6

We are able to achieve better accuracy with the shortest path kernel than the Graphlet kernel. We end up with 95% accuracy with the shortest path method while we achieve only 55% accuracy with the Graphlet method. Note that a random guess would get us 50% accuracy as we have a perfectly balanced dataset.

This means that the Graphlet method is not providing relevant information for the classification. We believe that these results might come from the fact that the graphlet kernel is based on graphlets of size 3. Since our dataset is made of paths and cycles it is easy to see that randomly picking subgraphs from any graphs with more than 3 nodes will always give use graphlets with 2 edges or less and almost always one or less edge. The main issue is that the representation of the cycle and path graphs with the graphlet kernel would end up being very similar.

Another issue is that the graphlet kernel captures the size of the graph. This is an issue since graphs of very different sizes will have very different representations. Indeed cycles with 4 nodes will have radically different representations than cycles with 100 nodes.

This issue is exacerbated by our point above. Indeed, on top of cycles and paths having similar representation, we expect the kernel representation between paths of different sizes to be easier to distinguish than cycle and paths of the same size. As such, we can hardly split the kernel space between cycles and paths. This would explain why our accuracy is close to the random guess.