Title: Assignment 2: Analyzing Facebook Large Page-Page Network

Author: "put your student id here"

Introduction:

In assignment 2, you will get familier with NetworkX (https://networkx.github.io/ (https://networkx.github.io/)) for network analysis (creating network, reporting network statistics, calculating three node centrality measures, and implement two link prediction methods). NetworkX has nice documentation including all graph algorithms it supports: https://networkx.github.io/documentation/stable/reference/index.html (https://networkx.github.io/documentation/stable/reference/index.html)

You can always add more code/markdown cells.

Dataset

The dataset we are using in this assignment is from Facebook. A detailed description of the dataset could be found at: https://snap.stanford.edu/data/facebook-large-page-page-network.html (https://snap.stanford.edu/data/facebook-large-page-page-network.html).

The whole network is a page-page graph of verified Facebook sites. Nodes represent official Facebook pages while the links are mutual likes between sites. Node features are extracted from the site descriptions that the page owners created to summarize the purpose of the site. This graph was collected through the Facebook Graph API in November 2017 and restricted to pages from 4 categories which are defined by Facebook. These categories are: politicians, governmental organizations, television shows and companies.

Important Note:

Don't forget to provide a summary of your findings/results for Task 2-4. Submit your code (executed ipynb file) to onq.

In [1]:
```
# import needed Python libraries, e.g., networkx
```

Task 1: load dataset and create network (10 points).

Download data from https://snap.stanford.edu/data/facebook-large-page-page-network.html (https://snap.stanford.edu/data/facebook-large-page-page-network.html). In this step, you need to load edge and node information from raw dataset. Usually, the edges and nodes are saved in csv files. To load and create network in networkx, please read the following tutorial:https://programminghistorian.org/en/lessons/exploring-and-analyzing-network-data-with-python (https://programminghistorian.org/en/lessons/exploring-and-analyzing-network-data-with-python)

In [ ]:

Task 2: network analysis, reporting basic statistics (10 points).

Basic network statistics include: 1) number of nodes in the network 2) number of edges in the network 3) average degree of node 4) radius of the network 5) diameter of the network 6) density of the network

ref: https://networkx.github.io/documentation/stable/auto_examples/basic/plot_properties.html#sphx-glr-auto-examples-basic-plot-properties-py (https://networkx.github.io/documentation/stable/auto_examples/basic/plot_properties.html#sphx-glr-auto-examples-basic-plot-properties-py)

Please also draw a figure representing the node degree distribution in the network. You can get degree distribution (count how frequent each degree value appear in the network) by:

```
degree_sequence = sorted([d for n, d in G.degree()], reverse=True)  # degree sequen
ce
degreeCount = collections.Counter(degree_sequence)
deg, cnt = zip(*degreeCount.items())
```

Plot the historgram distribution of degree, and then decide whether you need to create a log-log plot (using library matplotlib.pyplot) representing the degree distribution. ref. https://networkx.github.io/documentation/stable/auto_examples/drawing/plot_degree_histogram.html (https://networkx.github.io/documentation/stable/auto_examples/drawing/plot_degree_histogram.html)

In [ ]:

Findings for Task 2: Please summarize your findings from the above analysis, e.g., describe the characteristics of the network based on the network statistics.

Task 3: Node centrality analysis (30 points)

Pick three centrality metrics that you are interested to invesitgate and report at least three findings (e.g., what are the nodes with high centrality values, how the centrality values distributed in the network) Ref: https://networkx.github.io/documentation/stable/reference/algorithms/centrality.html (https://networkx.github.io/documentation/stable/reference/algorithms/centrality.html)

In [ ]:

Findings for Task 3: Please summarize your findings from the above analysis

Task 4: link prediction (40 points).

Pick one unsupervised and one supervised link prediction algorithm, implement the algorithms and compare the performance of two approaches. You must consider at least one ranking based evaluation metric and one classification based evalution metric.

Ref.https://github.com/lucashu1/link-prediction/blob/master/link-prediction-baselines.ipynb (https://github.com/lucashu1/link-prediction/blob/master/link-prediction-baselines.ipynb) https://github.com/lucashu1/link-prediction/blob/master/node2vec.ipynb (https://github.com/lucashu1/link-prediction/blob/master/node2vec.ipynb)

In [ ]:

Findings for Task 4: Please summarize the performance of two approaches.

Task 5: take-away message (10 points)

Please summary your findings from the above analysis (at least 2 findings):