Network Graph Analysis

Yacine MOKHTARI

January 2023

1 Introduction

The goal of this study is to analyze a real-world graph and to compare it to a random graph having the same number of nodes and average degree.

The graph I chose to work with is the *Twitch Social Networks[1]*. Although the streamers and viewers data have been anonymized, I wanted to see roughly the characteristics of one of the new social graphs of the internet.

Important note

I used the python library Plotly for interractive visualisation.
Data set, code and all the HTML pages leading to the interactive plots are available on my GitHub repository at this link : `https://github.com/YacineMOK/FR-Twitch__Network-Graph-Analysis`.

2 The minimal requirement

01 - Number of nodes and edges

Number of nodes, edges and connected components:

- Number of nodes: 6549.
- Number of edges: 112666.
- Number of connected components (sub-graphs): 1

Some additional numbers :

- Average degree: 34.4.
- Highest degree: 2040.
  - Nodes with deg $\geq \frac{2040}{2} = 1020 = 11$.
- Lowest degree: 1.
  - Nodes : 247

02 - Graph display

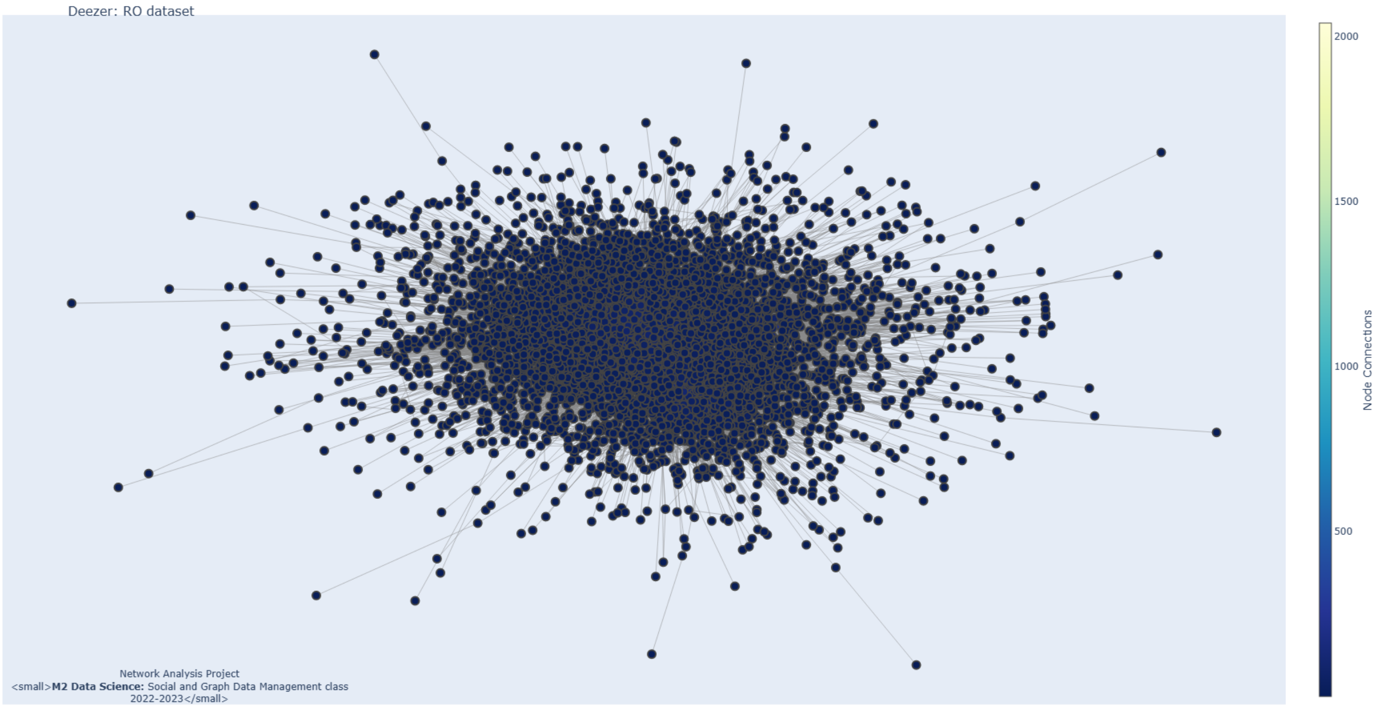
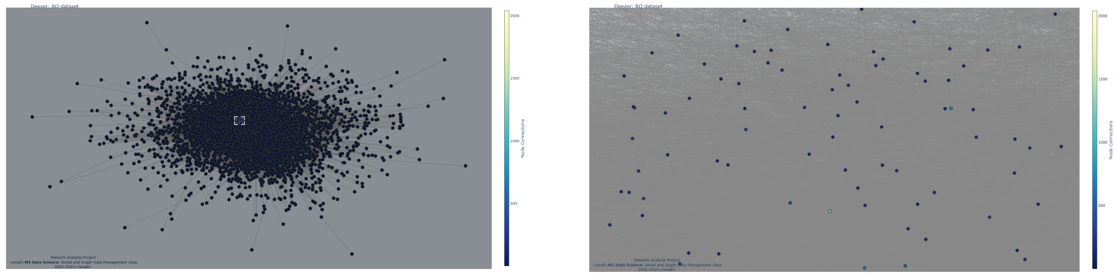


Figure 1: Network Graph

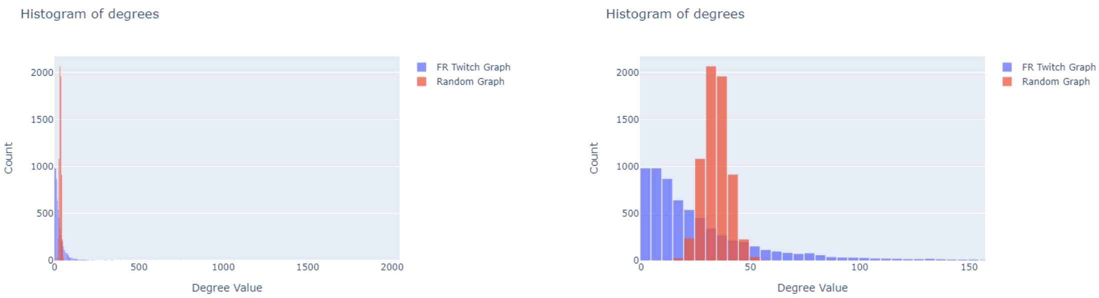


(a) Zoomed Area

(b) Showing nodes with some higher degreen and the many edges

Figure 2: Small fraction of the network graph

03 - Histogram of degrees



(a) Both the random and the FR-Twitch graphs

(b) Both: Zoomed

Figure 3: Histogram: Degree Distrubitions

04 - Degree Correlation

a. Degree Correlation Matrix

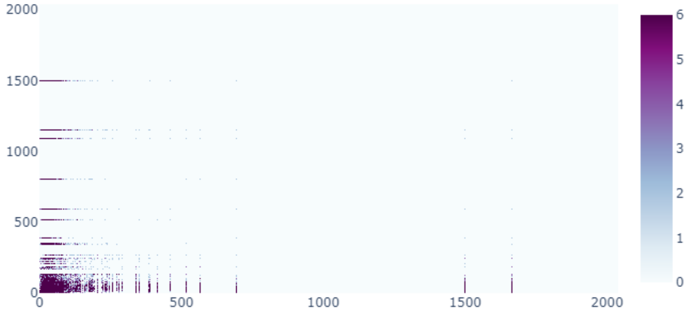


Figure 4: Degree correlation matrix (degree re-scaled from 1 to 6 to make it more visible and expressive

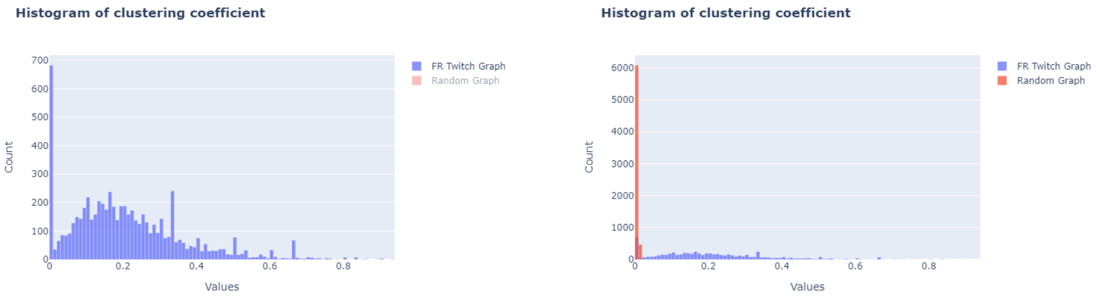
Using this simple code that I have implemented PUT PSEUDO CODE This is the degree correlation matrix of this network graph.
The only remark one can make is to say that low-degree nodes tend to connect with only low-degree nodes.

b. Pearson Degree Correlation

Using the `netwokx`'s `degree_pearson_correlation_coefficient` function, we found that the (Pearson) degree correlation is -0.17.

In this case, we have $r = -0.170$ which means that this graph is an Assortative Network[1]. This matches our previous analysis and remarks using the generated degree correlation matrix.

05 - Histogram of clustering coefficient



(a) FR-Twitch Graph

(b) Both

Figure 5: Histogram: Clustering Coefficient

With no surprise, these two distributions have nothing in common. While the random graph's distribution has a similar shape to a Poisson distribution with $\lambda = 1$, the FR-Twitch dataset seems to have a higher value for λ with a bump around 0.

06 - Histogram of distances

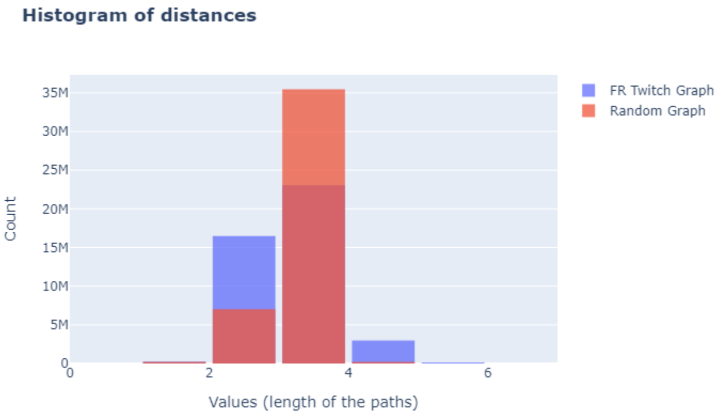


Figure 6: Histogram: Distances

Unlike all of the previous plots, this histogram shows a high similarity for the two distributions (with lower deviation for the network graph)

[1] **Assortative networks :** is a type of network graph in which nodes tend to connect to other nodes of a similar degree.

2.1 Extra requirements

09 - Community detection

Using the `greedy_modularity_communities()` function, and here are the results:

- Number of communities detection: 23.
- The average size of each community: 284.74.
- Size deviation : 715.65 (very high).
- Maximum size: 2821
- Minimum size: 2

I failed to display the different communities with Plotly...

08 - Triangles

Expected number of triangles for random graphs:
For three distinct nodes $i, j, k \in V$, the probability that the three edges $(i, j), (j, k)$ and (i, k) exist is p^3. Let $T_{i,j,k}$ a random variable equal to 1 if the triangle on those three nodes exists in the graph G, or 0 otherwise. (Hence, $\mathbb{P}(T_{i,j,k} = 1) = p^3$).

Then, the expected number of triangles in graph G can be computed like this:

$$\begin{aligned}\mathbb{E}\left[\sum_{i,j,k} T_{i,j,k}\right] &= \sum_{i,j,k} \mathbb{E}\left[T_{i,j,k}\right] \\ &= \sum_{i,j,k \in V} \mathbb{P}\left(T_{i,j,k} = 1\right) \\ &= \binom{n}{3} p^3\end{aligned}$$

In this case, with $n =$and $p =$, we have:

- The number of triangles of our dataset is 422694[2].
- The number of triangles of our dataset is around 6788.79[3].

Some remarks:

- The original value (from the karate graph) has a way higher value of 422K.
- The expected value is not even half of it, but this was expected!

References

[1] B. Rozemberczki, C. Allen, and R. Sarkar. Multi-scale attributed node embedding, 2019.

[2] Using the `networkx`'s `triangles()` function

[3] Using the estimation method