MLNS - Project
Communities Detection in undirected graph
Final Project Report

Gladys Roch / Aurélien Houdbert

April 2021

Abstract

In social networks (and any real-world network in general), nodes organize into densely-connected groups : network communities. In this project we chose to work on a Youtube network dataset provided by Stanford snap plat-form along with a paper presenting results of community detection on various graph networks [13]. The objective is to identify communities among the network using different algorithms and approaches and compare their performances.

1 Introduction

Graph networks are well adapted structures to represent many real world systems such as social networks. The notion of community refers to structure of nodes that tends to be organised into modules with specific properties. The community notion is not specific to social networks and have applications in many sectors such as biology where community detection has proven useful in protein-protein interaction, but also in politics, add targeting or even criminology. Community detection is a very well studied subject and lots of powerful algorithm have been developed. When it comes to social networks, the size of the graph quickly grows huge making most of classic community detection algorithms obsolete. To answer this issue, some algorithm have been developed especially to scale easily to large networks.

Community detection is an unsupervised task and we need good metrics to quantify the quality of the identified communities. Those metrics are often built on simple intuitions to illustrate and quantify expected behaviours and structure of communities. Of course there is no absolute truth when it comes to community detection and different metrics might lead to different optimization choice resulting in different partitions of the network. The metrics we proposed in our work are intuitive, yet efficient metrics, to characterise the quality of communities.

1.1 Data

The initial objective of our project, as presented in the project proposal, is to study the Youtube social Network (see characteristics is table 1.1). However, this graph is very large and requires a lot of computational power to manipulate, therefore to lower the computation burden, we selected a smaller graph on which to perform our analysis of the different communities-detection algorithms. We then applied to the Youtube graph 2 methods adapted to large graphs. The smaller graph represent circles of friends in Facebook network.

•Youtube graph

We use the Youtube-online-social-network database with ground-truth communities from Stanford Large Network Dataset Collection [1].

•Facebook graph

This facebook graph also comes from Stanford snap database. It is composed of circles of friends where edges represent friendship between two users. The dataset contains several graph of different size and structure.

[1] http://snap.stanford.edu/data/com-Youtube.html

Statistic	Youtube	Facebook
Nodes	1,134,890	324
Edges	2,987,624	2514
Average clustering coefficient	0.0808	0.522
Density	$4e^{-6}$	0.048
Diameter (longest shortest path)	20	11
Number of communities	8,385	not given

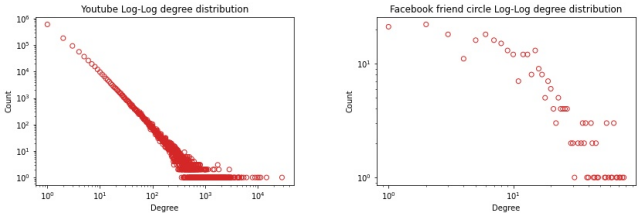

Figure 1: Youtube graph and Facebook friend circle log-log scale degree distribution. **Left** Youtube, **right** Facebook

For the Youtube network and Facebook graph, the degree distribution follows a power law distribution. In the case of Facebook graph it is a bit difficult to identify the parameters of this distribution because of the small size of the graph but in Youtube network, the power law distribution is of the form $p(k) = ck^{-\alpha}$ with $c = e^{11.16}$ and $\alpha = -1.58$.

1.2 Related work

Community detection is an already very covered subject. Originally approached with statistics [5], recent papers proposed deep learning approaches to deal with community detection [7, 2, 14]. Deep learning is more powerful than traditional machine learning for unsupervised learning tasks. For example in recent years, CNN based-approaches [12] have allowed to find communities in topologically incomplete networks (missing edges) and generative models [4] created more complete representations of community structures by leveraging node attributes in addition to network topology.

2 Problem definition

In this paper we will use the same notations in all following sections. Given G(V,E) an undirected graph, let $n = |V|$ be the number of nodes and $m = |E|$ the number of edges in the graph. Let also k_i be the degree of node i in G and A be the adjacency matrix of the graph. Given S a set of nodes in the graph G, let $n_S = |S|$ be the number of nodes in S, $m_S = |\{(u, v) \in E : u \in S, v \in S\}|$ the number of edges in S, $b_S = |\{(u, v) \in E : u \in S, v \notin S\}|$ the number of edges on the boundary of S.

We will also denote $\mathcal{V} = \{S_1, S_2, S_3, ..., S_k\}$ a partition of the graph where $S_i, i \in \{1, 2, ..., k\}$ will be called communities. For a given partition $\mathcal{V}$ we can define $q = \frac{\sum_i^k m_{S_i}}{m}$ the relative number of internal edges and $\langle q \rangle = \frac{\sum_i^k \binom{n_{S_i}}{2}}{\binom{n}{2}}$ the relative number of expected internal edges.

3 Methodology

Community detection methods can be broadly categorized into two types; Agglomerative Methods and Divisive Methods. In Agglomerative methods, edges are added one by one to a graph which only contains nodes. Edges are added from the stronger edge to the weaker edge. Divisive methods follow the opposite of agglomerative methods. In there, edges are removed one by one from a complete graph.

Five popular community detection algorithms are:

- Girvan Newman
- Spectral Clustering
- Modularity Maximization
- Louvain / Surprise / Leiden Community Detection
- Walktrap Community Detection

The objective is to implement, study and compare these algorithms.

3.1 Girvan Newman

Girvan Newman [5] is a divisive method, It consists of hierarchical clustering based on the notion of edge betweenness centrality. Edge betweenness is the number of shortest paths passing through the edge. The algorithm starts by calculating the betweenness centrality of all edges in the graph, then the edge with the highest betweenness score is removed. The betweenness of all the edges affected by the removal is recalculated, the removal step is then repeated until no edges remain.

3.2 Spectral Clustering

This method uses information from the eigenvalues (spectrum) of the normalized Laplacien matrix of the graph. The Laplacien matrix (L) is built from the graph adjacency matrix (A) and the degree matrix (D): $L = D - A$. The normalized Laplacien matrix is then: $L_n = D^{-1}L$.

Spectral clustering [8] consist of applying a k-mean clustering algorithm onto the vectors $(y_i)_{1,...,nodes}$ representing

the rows of the matrix V of columns the k-first eigenvectors of L_n, $V = [v_1|v_2|...|v_k]$. The number of communities is set by the parameter k. The best value of k is chosen as the one that maximize the modularity of the resulting partition.

3.3 Modularity Maximization

The modularity is a widely use metric when it comes to network clustering and represent how components are connected. The modularity metric can be expressed as follow:

$$\begin{aligned} Q &= \frac{1}{2m}\sum_{ij}\left(A_{ij} - \frac{k_i k_j}{2m}\right)\delta(C_i, C_j) \\ &= \frac{1}{2m}\sum_{ij} B_{ij}\delta(C_i, C_j) \end{aligned}$$

where C_i denotes the community of node i and δ the Kronecker function and B the modularity matrix:

$$B_{ij} = A_{ij} - \frac{k_i k_j}{2m}$$

As for Spectral Clustering, Modularity maximization tries to optimize the modularity score and uses information from the eigenvalues and eigenvectors of B.

3.4 Louvain/Surprise/Leiden Community Detection

The Louvain method is an heuristic algorithm to detect communities in large networks. It maximizes a modularity score (section 3.3) for each community, where the modularity quantifies the quality of an assignment of nodes to communities. This means evaluating how much more densely connected the nodes within a community are, compared to how connected they would be in a random network. [6] It is widely seen as one of the best algorithms for detecting communities, but can lead to arbitrarily badly connected communities. The Leiden algorithm [11] improves on the Louvain algorithm and guarantees to find well-connected clusters. It is also faster. For these reasons we chose to focus on the Leiden algorithm.

Surprise is another metric that evaluates the quality of a partition of a network into communities. The Surprise communities detection algorithm [1] is similar to the Louvain algorithm except that it uses surprise instead of modularity. Surprise, in its asymptotic formulation [10] can be formulated as follow:

$$Q = m \times \mathcal{KL}(q \| \langle q \rangle)$$

where KL is the Kullback-Leibler divergence defined as:

$$\mathcal{KL}(x \| y) = x log\left(\frac{x}{y}\right) + (1-x) log\left(\frac{1-x}{1-y}\right)$$

3.5 Walktrap Community Detection

Walktrap is another approach for community detection based on random walks in which distance between vertices are measured through random walks in the network. The idea of the algorithm is that random walks on a graph/ network tend to get trapped into densely connected parts corresponding to communities, [9]. Walktrap uses the result of random walks to merge separate communities in a bottom-up manner. Quality of the partitions can be evaluated using any available quality criterion. It can be either modularity as in Louvain community detection or any other measure. We choose to use modularity for consistency with spectral clustering. The random walks considered needs to be long enough to produce the optimal number of communities.

4 Evaluation

4.1 Evaluation metrics

In this section we will define evaluation metrics using the notation introduced in section 2.

4.1.1 Modularity

Unlike the other metrics, the modularity Q is an evaluation metric that quantifies the quality of the whole partition and not only of a single community. Intuitively, the structure of communities is better if the number of internal edges exceeds the expected number. The metric was defined in section 3.3

4.1.2 Separability

Separability of a community is nothing else than the ratio of the number of internal edges (m_S) and external edges (edges pointing outside from the community, b_S). Intuitively, the separability quantifies how "separate" a community is from the rest of the graph. Indeed, a good community is expected to have a large number of internal edges and few connections with the rest of the graph.

$$Sep(S) = \frac{m_S}{b_S}$$

4.1.3 Clustering coefficient

The clustering coefficient quantifies what portion of a node's neighbors are connected. The clustering coefficient of a community S is the average of all node's clustering coefficient in S. Intuitively, in a good community, nodes with common neighbors are very susceptible to be connected together.

$$C(S) = \frac{1}{n_S} \sum_{i \in S} C_i, \text{ where } C_i = \frac{2e_i}{k_i(k_i - 1)}$$

where e_i is the number of edges between the neighbors of node i.

4.1.4 Density

The density quantifies how dense a community is. Intuitively, a community should be "dense" with numerous edges between nodes. The density metric is the ratio of the number of edges and the total number of possible edges:

$$D(S) = \frac{2m_S}{n_S(n_S - 1)}$$

4.2 Results

In this project we worked with two different datasets:

- Facebook
- Youtube

The facebook network is actually a very small graph. It represents friendships between users in a small friend circle. We used this network to test and compare classic clustering algorithm that are not scalable to large graphs. Youtube network is the graph we were originally interested in. It is a large network that needs adapted algorithm, designed specifically to perform well on large graphs in a reasonable amount of time.

4.2.1 Facebook

The facebook friend circle network has 324 nodes and 2514 edges. It is a small network on which we tested the different algorithms described in section 3.

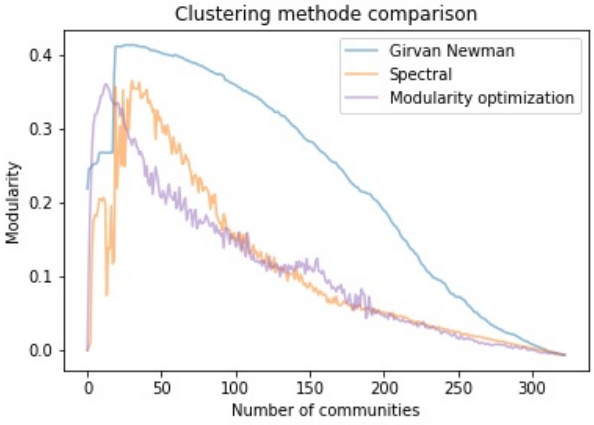

Figure 2: Facebook graph clustering method comparison.

Girvan Newman [5], Spectral clustering and Modularity maximization algorithms are powerful clustering techniques but they require a prior on the number of clusters to perform segmentation. In figure 2, we plotted the modularity as a function of the number of communities. The best modularity score is reached by the Girvan Newman algorithm [5]. Girvan Newman is a bit more computationally expensive to compute as it requires to compute the betweeness centrality measure for all edges in the graph. Spectral clustering and Modularity maximization are quite fast to execute on such small graph and should be scalable to medium size graph with sparse matrix eigen-vectors decomposition.

Table 2 shows results obtained for all algorithm in {Girvan Newman, Spectral Clustering, Modularity Maximization, Louvain, Leiden, Walktrap} for which we computed the mean and standard deviation of all metrics described in section 4.1. From this table we can observe that Louvain and Leiden method obtained the best metric scores and both detected 10 communities. Except Modularity maximization which detected 13 communities, the other algorithms identified a lot more clusters (between 26 and 31). Leiden seems the best overall method and obtained the best modularity score of 0.46 and the best separability score of 8.44. It also obtained reasonable clustering coefficient and density but Modularity maximization and Spectral Clustering actually obtained better results on those metrics. What is surprising concerning those results is that the modularity maximization algorithm obtained the worst modularity score while we expected to see the modularity of this method close to leiden or louvain as modularity is the metric it tries to optimize. We can also note that Grivan and Walktrap give close results in terms of modularity, separability and density despite Girvan having more communities.

Algorithm	N	Modularity	Separability	Clustering coefficient	Density
Girvan Newman	31	0.41	3.48 (7.75)	0.39 (0.35)	0.39 (0.4)
Spectral Clustering	31	0.37	2.19 (6.96)	0.37 (0.27)	**0.76** (0.25)
Modularity maximization	13	0.36	1.84 (3.1)	**0.59** (0.14)	0.58 (0.26)
Louvain	10	0.45	8.3 (12.53)	0.57 (0.15)	0.41 (0.27)
Leiden	10	**0.46**	**8.44** (12.45)	0.57 (0.16)	0.41 (0.27)
Walktrap	26	0.40	3.76 (7.93)	0.52 (0.31)	0.34 (0.31)

Table 2: Evaluation of the communities detected by each algorithm (Facebook graph)

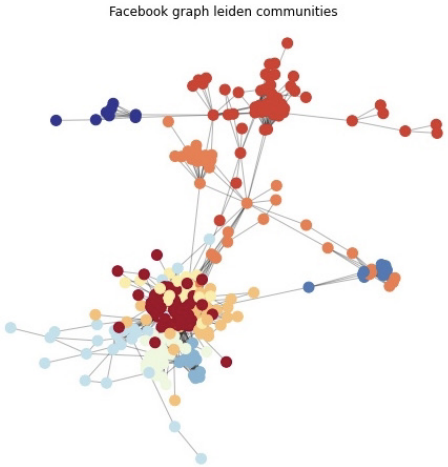

Figure 3: Facebook communities using Leiden optimization algorithm

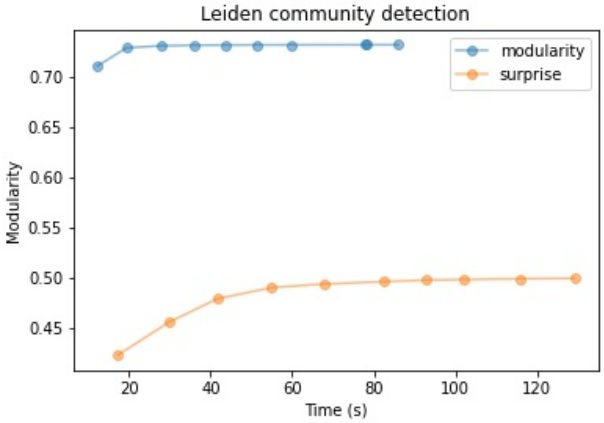

Figure 4: Leiden algorithm applied on Youtube network with modularity and surprise optimization. The graphic shows the evolution of modularity with the number of iteration and the computing time. Each point on the curves corresponds to the next iteration of the algorithm

Figure 3 shows the graph segmented by Leiden algorithm. Visually we can clearly identify some clusters (light red, orange and dark blue) that the algorithm properly detected but it also seem that the method has difficulties clustering the big node nest at the bottom left. Facebook network is a bit special. Indeed we can easily image that some communities would overlap when focusing on a friend circle which is not allowed by leiden method and that might be a reason to explain the issue encountered clustering this part of the network.

4.2.2 Youtube

The youtube social graph is a large network composed of $1,134,890$ nodes and $2,987,624$ edges. Classic clustering algorithm such as Girvan Newman, Spectral Clustering or Modularity maximization cannot be used in this case and we had to turn to more advanced clustering methods (section 3).

To cluster our graph into communities, we used the Leiden algorithm which performs iterations over the graph and tries to maximize the modularity metric. Figure 4 compares the evolution of the modularity with the number of iterations for the Modularity Maximization and the surprise optimization. For both algorithms we can observe that the modularity increases during the first 3 iterations and begins to plateau after the 4^{th} one.

Metric	Leiden	Louvain	Surprise
N	3592	7365	109521
Modularity	**0.73**	0.72	0.46
Separability	**6.34** (6.26)	4.41 (4.8)	0.63 (1.92)
Clustering coefficient	**0.11** (0.19)	0.08 (0.17)	0.06 (0.18)
Density	0.32 (0.18)	**0.41** (0.2)	0.14 (0.29)

Table 3: Leiden, Louvain and Surprise obtained metrics on youtube graph

We tested the Louvain, Leiden and Surprise clustrering algorithm on the youtube graph and reported the results in table 3. Here again Leiden obtained the best scores on

every metric except for the density. Louvain detected twice more clusters than Leiden but it is still reasonnable. Surprise on the other hand detected over 100,000 communities which is a bit much and resulting in poor quality metrics compared to the two other methods.

In figure 5. we observed the evolution of the quality metrics with the community size. The first thing we can observe is the distribution of density that is clearly a power law distribution $d(k) = ck^{-\alpha}$ with $c = e^{3.3}$ and $\alpha = -0.4$. This behavior was expected mainly because of the density definition (section 4.1.4). Concerning the clustering coefficient, it seems that the it is almost independent from the cluster size. Looking at the separability, on the left size of the graph, we can observe a quite abrupt positive slope at the beginning but drops quickly after community size reach about a 100 nodes. The distribution seems to follow a sort of skewed distribution with a heavy tail on the right.

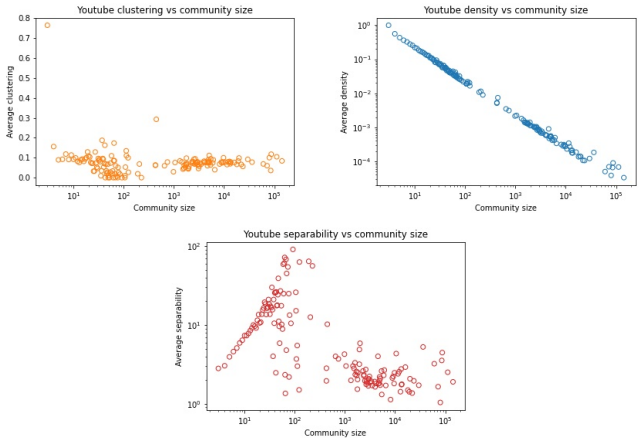

Figure 5: Youtube graph partitioned by Leiden algorithm into 3592 communities. The metrics are aggregated by community size. **Top-left** Clustering coefficient with x axis log-scaled, **Top-right** Density in log-log scale, **Bottom** Separability in log-log scale.

5 Conclusion

In this project we compared performances of several famous clustering algorithm that we tested on two different networks of different sizes and structure evaluating them on several quality metrics. We observed that the Leiden algorithm is by far the fastest algorithm and obtained the best results on both networks and especially reached a modularity score of 0.73 with 3592 communities on the Youtube graph. More classic algorithms such as Spectral Clustering, Girvan Newman or Modularity Maximization perform well on small size graphs but are hardly scalable to large networks.

References

[1] Marín I Aldecoa R. "Deciphering Network Community Structure by Surprise." In: (2011). URL: https://doi.org/10.1371/journal.pone.0024195.

[2] Sandro Cavallari et al. "Learning Community Embedding with Community Detection and Node Embedding on Graphs." In: (2017). URL: https://sentic.net/community-embedding.pdf.

[3] Javier Del Ser et al. "Community detection in graphs based on surprise maximization using firefly heuristics". In: (July 2016), pp. 2233–2239. DOI: 10.1109/CEC.2016.7744064.

[4] J. Hoffmann C. Huang F. Sun M. Qu and J. Tang. "vGraph: A generative model for joint community detection and node representation learning". In: (2019). URL: https://arxiv.org/abs/1906.07159.

[5] M. Girvan and M. E. J. Newman. "Community structure in social and biological networks." In: (2002). URL: https://www.pnas.org/content/pnas/99/12/7821.full.pdf.

[6] Mahantesh Halappanavar Hao Lu and Ananth Kalyanaraman. "Parallel Heuristics for Scalable Community Detection". In: (2014). URL: https://arxiv.org/pdf/1410.1237.pdf.

[7] Fanzhen Liua et al. "Deep Learning for Community Detection: Progress, Challenges and Opportunities." In: (2020). URL: https://arxiv.org/abs/2005.08225.

[8] Ulrike von Luxburg. "A Tutorial on Spectral Clustering". In: (2007). URL: https://arxiv.org/pdf/0711.0189.pdf.

[9] Pascal Pons and Matthieu Latapy. "Computing communities in large networks using random walks". In: (2016).

[10] V. A. Traag, R. Aldecoa, and J.-C. Delvenne. "Detecting communities using asymptotical surprise". In: (2015). URL: https://arxiv.org/pdf/1503.00445.pdf.

[11] L. Waltman V. A. Traag and N. J. van Eck. "From Louvain to Leiden: guaranteeing well-connected communities". In: (2019). URL: https://arxiv.org/pdf/1810.08473.pdf.

[12] X. Ying X. Xin C. Wang and B. Wang. "Deep community detection in topologically incomplete networks". In: (2017). URL: https://doi.org/10.1016/j.physa.2016.11.029.

[13] Jaewon Yang and Jure Leskovec. "Defining and Evaluating Network Communities based on Ground-truth." In: (2012). URL: https://arxiv.org/abs/1205.6233.

[14] Liang Yang et al. “Modularity Based Community Detection with Deep Learning.” In: (2016). URL: https://www.ijcai.org/Proceedings/16/Papers/321.pdf.