Comparison of Network Analytics and Significance Analysis on Spotify Artist Feature Collaboration Network

Learning From Networks - Mid-term report

Fabio Cociancich, Luca Fantin, Alessandro Lincetto

Master Degree in Computer Engineering - University of Padova

I. Random graph implementation

We have implemented a script generating random graphs using the `powerlaw_cluster_graph` method from the `networkx` library [1]. It creates random graphs following the Holme-Kim model [2] with a power-law degree distribution and clustering, resembling real-world networks. The number of nodes in the random graphs is specified by a parameter, and the structure of the graph is controlled by two more parameters: the number of edges created for each new node added to the graph, and the probability of adding triangles involving the newly added nodes. The code generates multiple random graphs, calculates several metrics (number of connected components, degree / closeness / betweenness / PageRank / eigenvector centrality, clustering coefficients for nodes, global/average/approximate global clustering coefficient for the whole graph) and stores the results in a CSV file for further analysis.

II. Subgraph creation and analysis

In our implementation, subgraphs are being computed with `networkx` and `networkit` packages. Until now we have computed subgraphs considering only the nodes which represent artists of a certaing musical genre and only the existing edges between those nodes. The code computes the same metrics computed for random graphs. It also stores the subgraph in a suitable file in order to be analyzed with the SILVAN algorithm [3], which is implemented in a separate script.

III. Statistical hypothesis testing

Our statistical hypothesis testing procedure will comprise of two steps. Any statistical test assumes that the considered population has a known distribution, most often a Gaussian one. Because of the specificity of our definition of random graph, instead of analytically computing the distribution *a priori* of the centrality metrics of our model, the first step will compute how similar this distribution is compared to a Gaussian distribution starting from the features computed on the actual generated graphs, through a *normality test*. We will use Shapiro-Wilk test [4], since it is considered the most powerful normality test available [5]. The second step will actually determine how likely it is for the features computed on the real graph to have been drawn from the same distributions as the random graph. The available tests will be determined by the output of the first step.

IV. Experiments

We have determined the feasibility of performing our experiments on the CAPRI cluster [6]. The project repository can be cloned and the necessary Python packages can be installed on our own profile, making it easy to efficiently translate the code development done on our own machines into this testing environment. For any computation, we will submit our jobs to the cluster via the SLURM work scheduler. This allows us to exploit the full computational power of CAPRI while also keeping track of execution time and resource usage.

We performed some early tests, executing the code for random graph generation and analysis

on the cluster. We have seen that computing all exact metrics on random graphs of the same size as the real-world graph, requires a large amount of time. Thus, we intend to computer all metrics on the real graph only and then perform less expensive computation on the random graphs, either by using only approximate algorithms or by generating graphs with the same size as the genre-specific subgraphs.

Contributions

Out of the work presented in this report, Fabio Cociancich wrote the Python scripts to extract the subgraphs based on genre and artists and the scripts to compute the metrics of a graph. Alessandro Lincetto wrote the script to generate random graphs and compute their metrics, and Luca Fantin investigated how to use the CAPRI cluster and the statistical tests to use, as well as performing an early refactoring of our code. In addition, each of the members wrote the section related to their work, except for the refactoring. The percentage of work done can thus be estimated in 35%, 20% and 45% respectively.

References

[1] *NetworkX, graph generators*. URL: https://networkx.org/documentation/stable/reference/generators.html.

[2] Petter Holme and Beom Jun Kim. "Growing scale-free networks with tunable clustering". In: *Physical Review E* 65.2 (Jan. 2002). ISSN: 1095-3787. DOI: 10.1103/physreve.65.026107. URL: http://dx.doi.org/10.1103/PhysRevE.65.026107.

[3] Leonardo Pellegrina and Fabio Vandin. *SILVAN: Estimating Betweenness Centralities with Progressive Sampling and Non-uniform Rademacher Bounds*. 2022. arXiv: 2106.03462 [cs.DS]. URL: https://arxiv.org/abs/2106.03462.

[4] S. S. Shapiro and M. B. Wilk. "An Analysis of Variance Test for Normality (Complete Samples)". In: *Biometrika* 52.3/4 (1965), pp. 591–611. ISSN: 00063444, 14643510. URL: http://www.jstor.org/stable/2333709 (visited on 12/17/2024).

[5] Nornadiah Mohd Razali and Bee Yap. "Power Comparisons of Shapiro-Wilk, Kolmogorov-Smirnov, Lilliefors and Anderson-Darling Tests". In: *J. Stat. Model. Analytics* 2 (Jan. 2011).

[6] University of Padova Strategic Research Infrastructure Grant 2017. *CAPRI: Calcolo ad Alte Prestazioni per la Ricerca e l'Innovazione*. https://capri.dei.unipd.it/.