# The stability and robustness of the Github network of collaborators

**Jan Kenda**[a] **and Andraž De Luisa**[b]

[a]University of Ljubljana, Faculty of Computer and Information Science, Ljubljana, Slovenia, jk3977@student.uni-lj.si ; [b]University of Ljubljana, Faculty of Computer and Information Science, Ljubljana, Slovenia, ad9366@student.uni-lj.si

This manuscript was compiled on June 22, 2020

**Social networks maintain a fragile equilibrium that can be easily disturbed by removing key individuals called linchpins. This is no less true in the network of Github collaborators (1). We propose a measure of robustness for social networks, that how vulnerable a network is when it comes to losing a percentage of its key individuals. The measure is based on the stability of the community structure of the social network, specifically by finding individuals which we deem most important to maintaining the current social structure. We first evaluate the performance of the proposed methods on Lancichinetti–Fortunato–Radicchi benchmark graphs (2) and then on the Github network itself.**

networks | communities | robustness | node centrality

Sharing ideas, problems and solutions is a fundamental reason why humans are as advanced as they are today. This is true in all fields of science: biology, physics, mathematics and potentially even more so in computer science. With the advancements in technology and specifically the internet in the last couple of decades, cooperation between researchers and innovators has never been easier. This is in large part due to Git and other version control systems, which allow both researchers and software developers to interact and collaborate on projects with ease, in turn leading to faster learning, innovation and development of scientific and technological advancements.

But this network has a big drawback: it is a social network. And as a social network, it has its linchpins - key members of a society which hold everything together. Without them the network might fall apart. These are the people who introduce others. The keys that connect people into a community. And this is what we are interested in.

If we are able to find a community structure in the GitHub network of collaborators (3), we can then try to measure how "well connected" each community is. Using this measure as a criterion, we can then start looking for linchpins - in this case nodes, which if removed, would destabilize the community structure of the network as much as possible. In the real world social network this means removing individuals that we think contribute most to the development of new ideas and projects. Not necessarily provide solutions themselves, but by connecting others and creating (or just being the core of) communities, contribute to further technological advancements the most.

In this paper we analyse the ability of several node centrality measures to highlight these crucial nodes for the robustness of the community structure in social networks.

## Related work

A lot of work has been done in detecting communities and evaluating their stability. A quantification of the community structure robustness and the method for its computation is proposed by Karrer et al. (4). A somehow similar approach to statistical validation of such partitions of networks into communities has been analysed by Carissimo et al. (5). Both are based on random perturbations of edges in the network, analysing the induced changes in the community structure. Our approach differs from them, as instead on the edges we are more concerned on the importance of single nodes in the communities.

A node centrality measure for classification of nodes with respect to their importance in the community structure, based on the spectrum of the adjacency matrix, has been recently proposed by Wang et al. (6).

## Methods and Data

For this project we use the data gathered from SNAP (1), which contains a social network of collaborators on Github up until August 2019. The nodes of the network are Github users who starred popular machine learning and web development repositories (only repositories with at least 10 stars are taken into account). The edges of the network are their follower relationships. It is an undirected graph.

We start by generating Lancichinetti–Fortunato–Radicchi benchmark graphs with planted communities. These graphs serve as testing networks for our project. First we find the communities within the network using various community detection algorithms, such as Louvain (7), Infomap (8) and Label propagation (9).

Since the Github network is a social network, the communities are not as well-defined and clearly seperated as is the case with Lancichinetti–Fortunato–Radicchi benchmark graphs. Because of this, we require algorithms which allow for overlapping communities, such as Demon (10), (11), which uses a local-first approach to community discovery, or if overlaps are significant, a $k$-clique algorithm based on (12), which works by locating all cliques in the network and then identifying communities based on the analysis of the clique-clique overlap matrix.

The next step is measuring how well-connected a community actually is, and how stable is the whole community structure. The most basic method would be to simply measure the number of connections between nodes within a community and compare it with the number of connections that lead outside of it. A more advanced method is to measure the conductance (13) of communities of the network. The stability of the community structure can be estimated from the changes

[1] Both authors contributed equally to this work.

in the number of communities detected, their average size and the number of nodes contained in a community.

Once we have this, we start looking for key nodes that, if removed, would most disturb the community structure of the network. We do this by employing various methods of node importance. We use different methods of node centrality, namely degree centrality, closeness, bridgeness, betweenness, pagerank and eigenvector centrality. We also consider if, instead of removing nodes, we could just remove edges (e.g. sever the collaboration between two users). For this we calculate the edge betweenness centrality measure for all edges.

Using the above measures as criteria, we start testing by removing the nodes (or edges) with the highest measure and again calculating the conductance of communities, their number and average sizes, or even testing if some communities fall apart into connected components due to the removal of an individual. As such we aim to find the best measure, or even a combination of measures, to identify key linchpin individuals, without which the community structure of a social network would change drastically.

## Results

Because of the size of the network (37700 nodes and 289003 edges), we choose to use degree centrality, eigenvector centrality and pagerank as our measures of node importance. We use the Demon algorithm to discover overlapping communities, first on the base network and again after each iteration of node removals. We discard all communities smaller than three nodes.

For each measure of node importance, we start with the base network and then remove 10 by 10 most important nodes (0.026 % of the entire network each step) and at each step run the Demon algorithm to detect communities. All three measures yielded similar results. It is apparent that there is a small number of very key individual nodes that, when removed, cause the entire community structure to change drastically. The effect can be seen in figures 1, 2, 3 and 4.

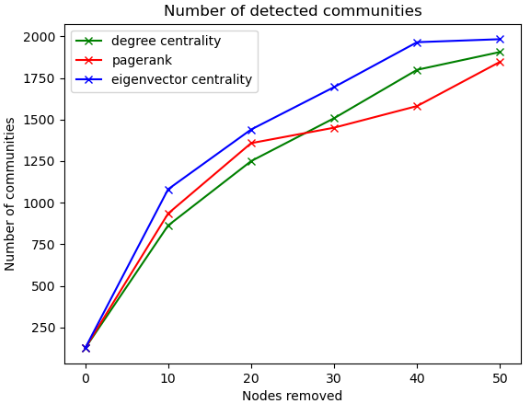

**Fig. 1.** The number of detected communites based on how many nodes we remove from the network. As can be seen from the graph, as we remove key nodes, communities start to fall apart. This is true for all measures of node importance. From the fact that the rise in the number of communities with the first removal (from $125$ to $1081$ when using eigenvector centrality) is so steep, we can assume there are a few extremely important individuals, who connect several smaller communities into fewer, larger ones.

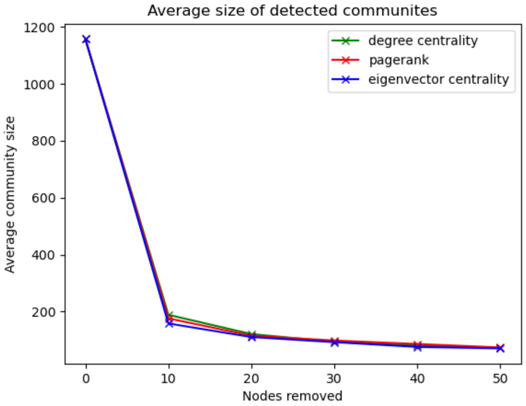

**Fig. 2.** The average size of detected communities based on how many nodes we remove from the network. All three measures yield extremely similar results, varying only slightly at $10$ removed nodes. The steep drop in the average size is apparent at the $10$ nodes removed mark.

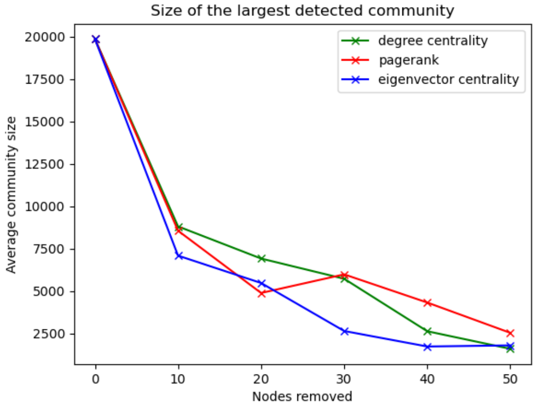

**Fig. 3.** The size of the largest detected community based on how many nodes we remove from the network. From this figure it is apparent that when using different measures of node importance we target different nodes for removal. The end results still seem to converge.

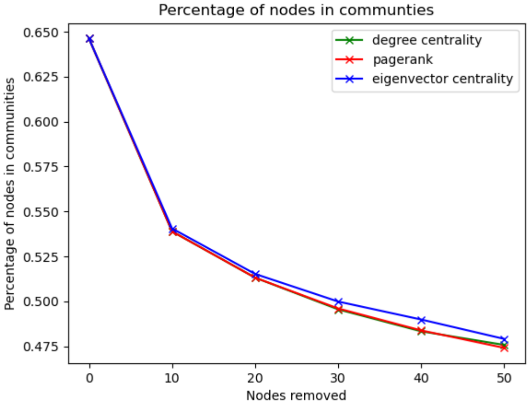

**Fig. 4.** The percentage of all nodes that are found in communties based on how many nodes we remove from the network. All three measures again yielded very similar results. In the base network, nearly two thirds of all nodes are in communities. After the removal of only $50$ nodes though, less than half of all nodes remain in communities.

The number of detected communities seems to be steadily increasing, but it cannot rise forever, so we continue with the nodes removal to check its behaviour. This time we remove 50 by 50 nodes, up to 377 removed nodes (which is exactly 1 % of the entire network). The results are shown in figure 5. The other three measures of network robustness (average size of communities: figure 6, size of the largest community: figure 7 and percetage of remaining nodes in communities: figure 8) all keep declining at a steady pace.

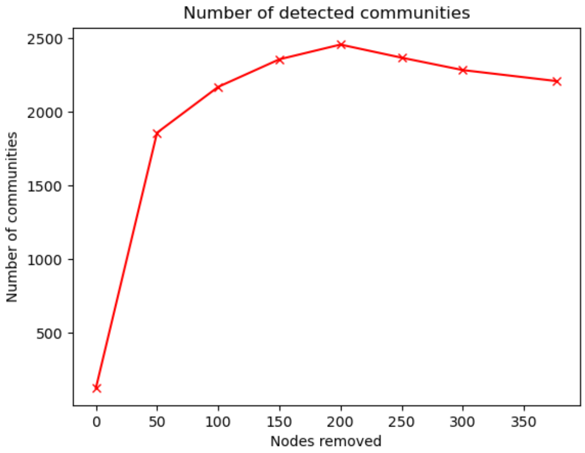

**Fig. 5.** The number of detected communites based on how many nodes we remove from the network. As we can see, the number of detected communities starts to slowly decline after removing more than $200$ nodes from the network. This can be attributed to some communities breaking into parts that are too small to be detected as communities.

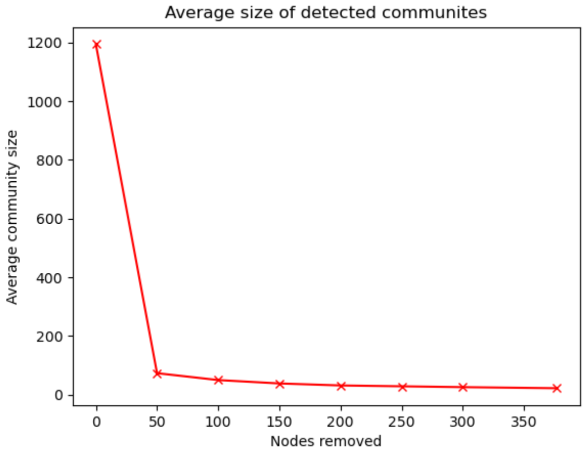

**Fig. 6.** The average size of detected communities based on how many nodes we remove from the network. It continues to decline at a steady pace. While not apparent from the graph, the average size at $377$ removed nodes (avg size of $22.5$) is less than half of that at $100$ removed nodes (avg size of $50.3$).

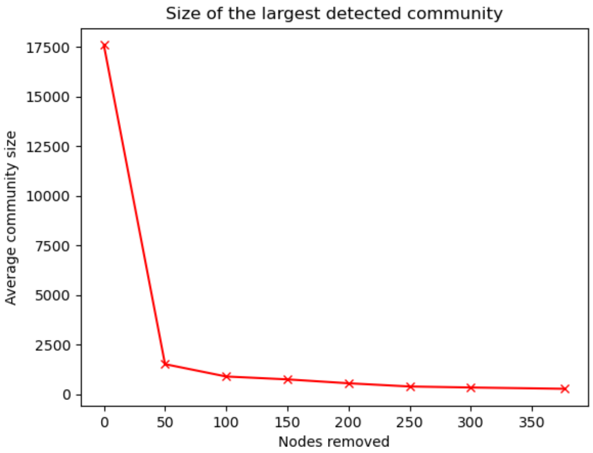

**Fig. 7.** The size of the largest detected community based on how many nodes we remove from the network. It is steadily decreasing also after the initial drop. At $1$ % removed nodes, the largest community contains just $276$ nodes, less than a third than the largest community after the removal of 100 nodes (with 896 nodes).

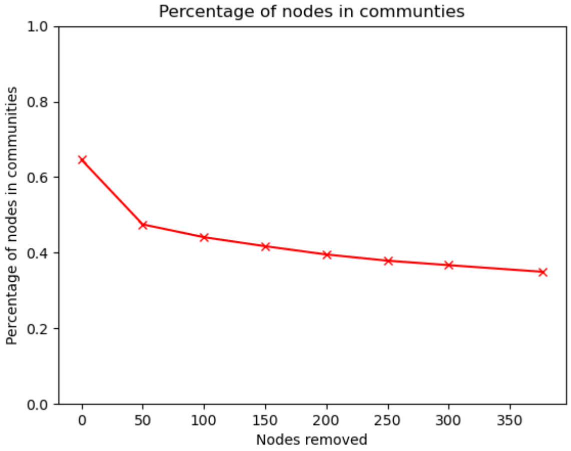

**Fig. 8.** The percentage of all nodes that are found in communties based on how many nodes we remove from the network. At $1$ % removed nodes, only $34.9$ % of all nodes remaining in communities. This is almost half of the original ($64.6$ %).

Lead author last name *et al.* PNAS | **June 22, 2020** | vol. XXX | no. XX | **3**

|  | Eigenv | Degree | Area | Abs diff |
|---|---|---|---|---|
| Eigenv | 1 | 0.9811 | -0.6318 | -0.2168 |
| Degree | 0.9811 | 1 | -0.7200 | -0.2435 |
| Area | -0.6318 | -0.7200 | 1 | 0.5160 |
| Abs diff | -0.2168 | -0.2435 | 0.5160 | 1 |

**Table 1. Correlations between centrality measures**

| Id | Value | Account |
|---|---|---|
| 22881 | 44.369 | ajsb85 |
| 25477 | 36.950 | mcanthony |
| 2281 | 31.650 | nikitavoloboev |
| 14242 | 30.559 | sibelius |
| 19253 | 29.473 | JonnyBanana |

**Table 2. Top nodes with respect to "area between ncp curves"**

| Id | Value | Account |
|---|---|---|
| 31890 | -509.66 | dalinhuang99 |
| 18163 | -96.021 | nelsonic |
| 36652 | -39.096 | rfthusn |
| 14954 | -22.323 | rasbt |
| 9051 | -19.213 | getify |

**Table 4. Worst nodes with respect to "area between ncp curves"**

| Id | Value | Account |
|---|---|---|
| 31890 | -2.0034 | dalinhuang99 |
| 30235 | -1.3018 | philsturgeon |
| 33029 | -1.0853 | igrigorik |
| 18945 | -0.9490 | TheLarkInn |
| 36628 | -0.5055 | SuriyaaKudoIsc |

**Table 5. Worst nodes with respect to "maximum difference between ncp curves"**

## Discussion

Of the three measures of node importance that we used, all three performed similarly. We believe that betweenness or bridgeness centralities might perform slightly better, but due to their time complexity, we decided against using them on a network of this size.

The github network is connected. What we would like to test further is whether it would be better to seek out and remove articulation points instead of using measures of node importance, thus breaking the network into connected components. It would also be interesting to test if we can achieve similar community decomposition *without* removing any articulation points, i.e. keeping the network connected. For this purpose we will need to further explore other node centrality measures.

We also believe that other measures of community quality are needed to evaluate the robustness of the network's partition into communities, hence we will also measure the average conductances, modularities and out-degree fractions.

1. Rozemberczki B, Allen C, Sarkar R (2019) Multi-scale attributed node embedding.
2. Lancichinetti A, Fortunato S, Radicchi F (2008) Benchmark graphs for testing community detection algorithms. *Phys. Rev. E* 78(4):046110.
3. Rozemberczki B, Allen C, Sarkar R (2019) Multi-scale attributed node embedding.
4. Karrer B, Levina E, Newman MEJ (2008) Robustness of community structure in networks. *Physical Review E* 77(4).
5. Carissimo A, Cutillo L, Feis ID (2018) Validation of community robustness.
6. Wang Y, Di Z, Fan Y (2011) Identifying and characterizing nodes important to community structure using the spectrum of the graph. *PloS one* 6:e27418.
7. Blondel VD, Guillaume JL, Lambiotte R, Lefebvre E (2008) Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment* 2008(10):P10008.
8. Edler D, Eriksson A, Rosvall M (year?) The mapequation software package.
9. Raghavan UN, Albert R, Kumara S (2007) Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E* 76(3).
10. Coscia M, Rossetti G, Giannotti F, Pedreschi D (2012) Demon: A local-first discovery method for overlapping communities in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12. (Association for Computing Machinery, New York, NY, USA), p. 615–623.
11. Coscia M, Rossetti G, Giannotti F, Pedreschi D (2014) Uncovering hierarchical and overlapping communities with a local-first approach. *ACM Trans. Knowl. Discov. Data* 9(1).
12. Palla G, Derényi I, Farkas I, Vicsek T (2005) Uncovering the overlapping community structure of complex networks in nature and society. *Nature* 435:814–818.
13. Leskovec J, Lang KJ, Mahoney MW (2010) Empirical comparison of algorithms for network community detection.

| Id | Value | Account |
|---|---|---|
| 36652 | 0.8556 | rfthusn |
| 18163 | 0.8544 | nelsonic |
| 3712 | 0.8541 | isaacs |
| 27803 | 0.7729 | nfultz |
| 36289 | 0.5655 | Trinea |

**Table 3. Top nodes with respect to "maximum difference between ncp curves"**