Clustering Algorithm Analysis on Protein-Protein Interactomes

https://www.github.com/BatesJackson/COSC448_Final_Project

Abstract

Graphing protein-protein interaction networks (Interactomes) within different organisms has become a major field of study in bioinformatics. In these networks, a node represents individual proteins, while the edge represents an interaction between proteins. Researchers use clustering algorithms on such networks to discover highly connected groups of proteins that interact with one another. These highly connected groups are called protein complexes, and play essential roles in regulatory processes, cellular functions, and signaling cascades [5]. In this report, I will present a comparative assessment of three common clustering algorithms: MCODE, Girvan-Newman (GN) and Clauset-Newman-Moore (CNM). Using a graph similarity score, these algorithms will be compared throughout a series of network alterations to simulate the dynamic nature of protein interactions as shown in [1]. The goal of these comparisons is to investigate the robustness of each individual clustering algorithm, after each network perturbation has occurred. The network used for this report can be found here: https://www.ebi.ac.uk/intact/query/annot:%22dataset:biocreative%22?conversationContext=2

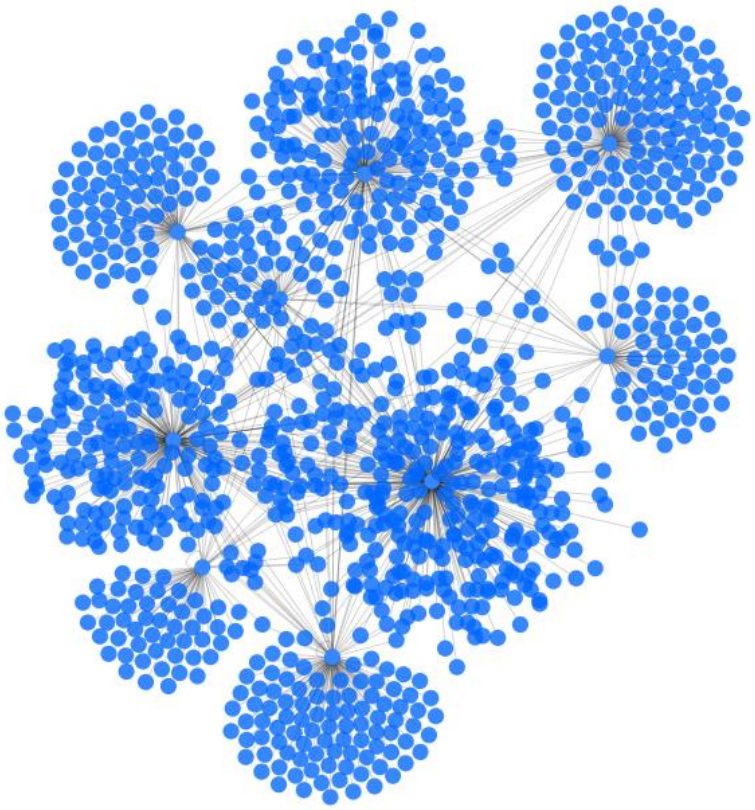


Figure 1. A visualization of the Network used for this report

Algorithms

Girvan-Newman: The following description of the Girvan-Newman algorithm is interpreted from [6].

The Girvan-Newman algorithm identifies clusters by progressively removing edges with the highest betweenness from the graph. The betweenness of an edge is measured by the number of all pairs shortest paths that run through the edge. Removing an edge with high betweenness increases the likelihood of exposing underlying clusters in the original graph.

GN runs in four steps "1. Calculate the betweenness for all edges in the network. 2. Remove the edge with the highest betweenness. 3. Recalculate betweenness for all edges affected by the removal. 4. Repeat from step 2 until no edges remain. [6]." The result of GN is a dendrogram which represents a hierarchical decomposition of the network. Since the algorithm must calculate the betweenness of the edges after each removal, the run time of GN for a graph with E edges and V vertices is O(E^2V).

MCODE: The following description of the MCODE algorithm is interpreted from [4].

Molecular Complex Detection – Is a clustering algorithm that detects potential protein complexes by grouping highly connected regions of a PPI network. MCODE runs in three stages, vertex weighting, complex prediction, and post-processing.

During the vertex weighting phase, each vertex in the graph is weighted by its local neighborhood density. Local neighborhood density is calculated using the highest k-core of the current vertex neighborhood. "A k-core is a graph of minimal degree k (graph G, for all v in G, deg(v) >= k) [4]." This weighting scheme causes the algorithm to prioritize densely grouped regions and will often leave many nodes with low local neighborhood density, out of the results.

The second stage of MCODE is the process of predicting clusters. This step starts at the highest weighted vertex in the network and recursively moves outward clustering all vertices whose weight scores above a predetermined threshold. When no more vertices can be added to the cluster, the process is repeated, starting with the highest scored vertex that does not belong to a cluster. The cutoff threshold is calculated by (1.0 – Node Score Cutoff) x (Start node score). The node score cutoff is an adjustable parameter that can be used to increase or decrease the size of the resulting clusters. A larger node score cutoff creates larger clusters.

The third stage of MCODE is post-processing. This stage contains two optional features, haircut and fluff. When haircut is active, it ensures that all resulting clusters are at least 2-core. It does this by removing all vertices with one connecting edge from each cluster. Most versions of MCODE will have haircut on by default. The fluff feature adds neighboring vertices to the cluster based on the neighborhood density of and the fluff score.

MCODE has a time complexity of O(VEH^3) where V is the number of Vertices, E is the number of edges and H is the size of the average vertex neighborhood. However, once each vertex is weighted, the algorithm can run in linear time.

Clauset- Newman-Moore: The following description of the CNM algorithm is interpreted from [7].

The Clauset-Newman-Moore algorithm was developed to combat the slower run times of different community detection algorithms. CNM is based on "the greedy optimization of the quantity known as modularity [7]." Modularity is the "measure of the quality of a particular division [8]." For information on how modularity is calculated see [8].

CNM begins by initializing every node in the network as its own community and calculating the modularity for every possible combination of communities. The combination that creates the largest modularity is combined to create a new community. The algorithm then repeats these steps until there is only one community remaining. The results of the CNM algorithm is a dendrogram. However, unlike GN, the dendrogram created from CNM is built from the bottom up. Starting with each individual node, building to a single network.

The runtime of Clauset-Newman-Moore is O(EdlogV) where d is the depth of the resulting dendrogram. This results in an algorithm that can produce results with networks consisting of millions of vertices, in reasonable runtimes.

Analysis

In order to evaluate the robustness of each algorithm, I created various altered networks that were generated by adding or removing a percentage of edges on the original network. For this report, I created six different states of altered graphs. These states are: adding 1% of edges, adding 5% of edges, adding 10% of edges, removing 1% of edges, removing 5%, of edges, removing 10% of edges. Each altered state was generated 10 times to reduce randomness in the results. This resulted in 61 total graphs including the original network.

To score each algorithm, they had to be measured against themselves. All three algorithms were first ran on the original graph. The largest cluster from these results were then individually compared to the largest clusters from all 10 graphs in each state using a similarity scoring system. The average for each state was calculated and used to evaluate each algorithms robustness through each state. To calculate the similarity score, I multiplied the intersection of the two clusters by two, then divided by the sum of each cluster's length. Therefore, if C1 = [1,2,3] and C2 = [1,2,3] the similarity score would equal 1.0. The result ranges between 1.0 (perfect match) and 0 (no matching nodes).

MCODE:

As shown in Figure 2, the results of MCODE varied quite significantly with any alterations present. However, the algorithm proves much more robust when adding edges then when removing edges. When adding 1%, 5%, and 10%, the similarity scores were 0.86, 0.72, and 0.53 respectively, compared to 0.67, 0.61 and 0.25 when removing edges. These differences are likely due to MCODES nature of finding densely connected clusters within the graph. If you remove an edge from a densely connected cluster, there is a chance that two nodes immediately do not meet the k-core requirement, thus the results of MCODE are more susceptible to change when removing edges. This behaviour seems to be shown by the large jump between removing 5 and 10 percent.

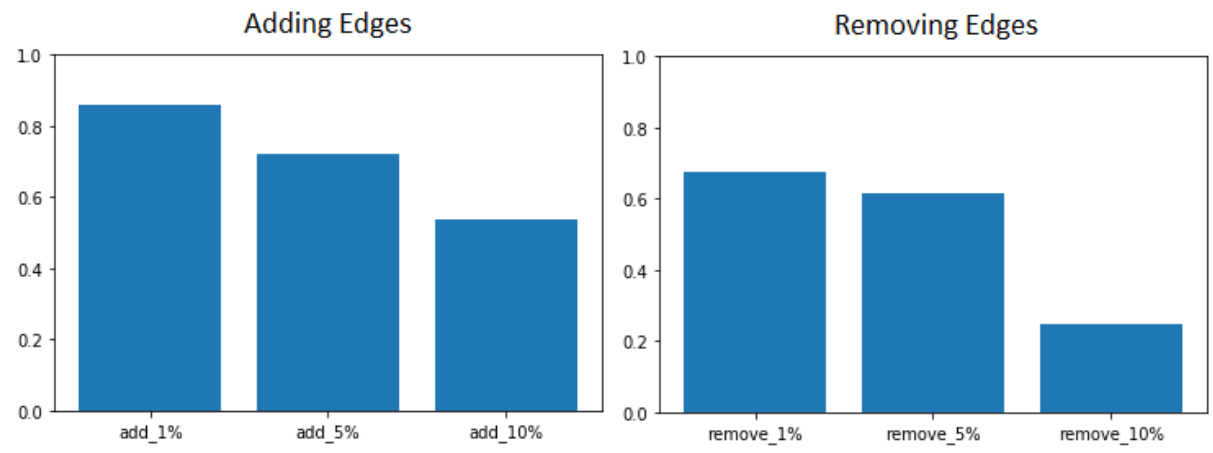


Figure 2

Girvan-Newman:

As shown in Figure 3, for the Girvan-Newman algorithm, adding edges scores are 0.98, 0.94, 0.91, and the removing edges scores 0.79, 0.78, 0.76. GN displays a much more robust behaviour when adding edges compared to removing edges. However, unlike MCODE, there is not a drastic change in similarity when increasing the amount of edges added or removed. It is unclear to me why the Girvan-Newman algorithm behaves this way. However, my guess would be that removing edges would lead to more edges with higher betweenness, which could potentially lead to the different structure of communities.

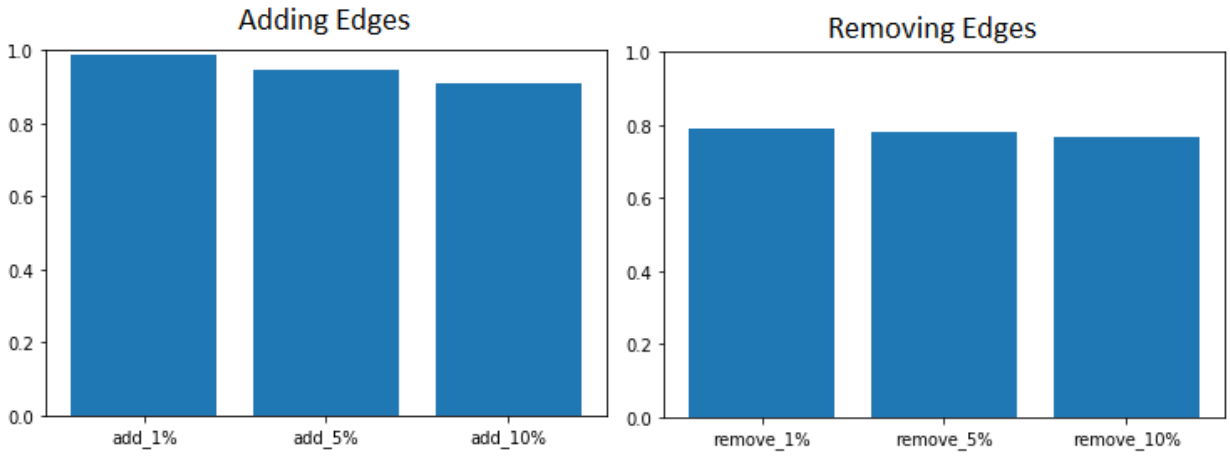


Figure 3

Clauset-Newman-Moore:

As shown in Figure 4, the CNM similarity scores for adding and removing edges was 0.99, 0.95, 0.92 and 0.99, 0.97, 0.93 respectively. The results of this algorithm show that it is the most robust out of the three algorithms over the measured states, and for the first time, there is a slight improvement when removing than adding edges. Again, it is unclear to me why the CNM algorithm seems to have little reaction to altered graphs.

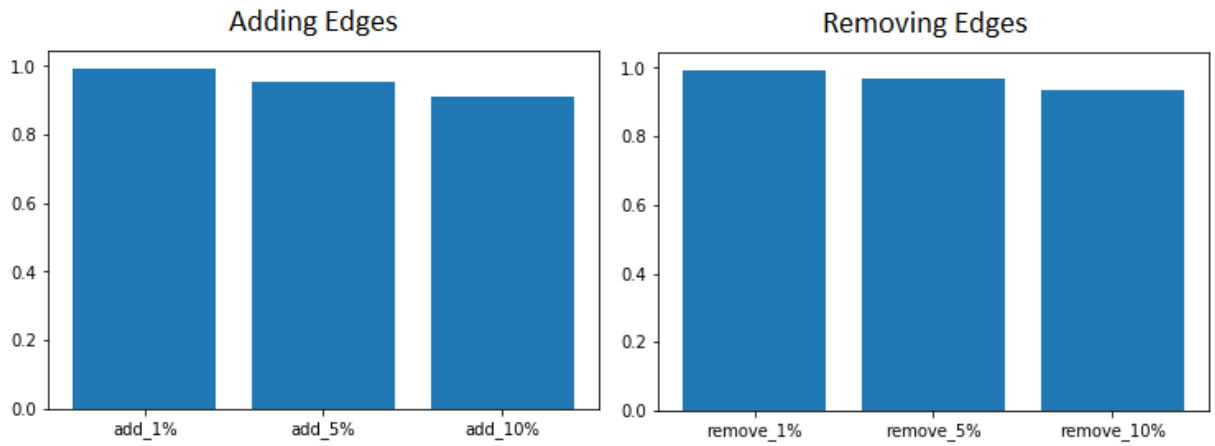
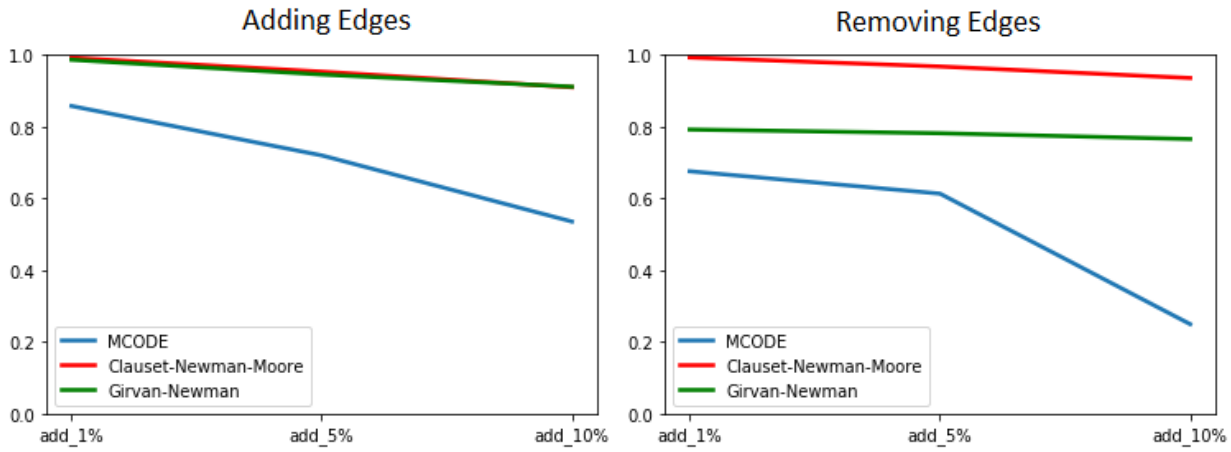


Figure 4

Conclusion

The goal of this report was to test and assess the robustness of different clustering algorithms that may be used in the analysis of protein-protein interaction networks. The clustering algorithms under assessment were MCODE, Girvan-Newman and Clauset-Newman-Moore. Each algorithm was scored by comparing its results from the original network graph, to its results on various altered network graphs. Below in Figure 5 is the comparison of all algorithms. As visualized, MCODE did not perform well compared to the other two algorithms. GN and CNM were very similar in the addition tests, but CNM was nearly 20% better than GN through all states in the removal test. One should keep in mind that this report does not test the accuracy of protein complex prediction. So while CNM is the most stable when exposed to a perturbed network, it may not be clustering the proteins into their correct complexes.



References

[1] R. Stoney, D. L. Robertson, G. Nenadic, and J.-M. Schwartz, “Mapping biological process relationships and disease perturbations within a pathway network,” *npj Systems Biology and Applications*, vol. 4, no. 1, Jun. 2018.

[2] S. P, “Importance of Studying the Interactome,” *News*, 24-Aug-2018. [Online]. Available: https://www.news-medical.net/life-sciences/Importance-of-Studying-the-Interactome.aspx. [Accessed: 29-Nov-2019].

[3] J. Snider, M. Kotlyar, P. Saraon, Z. Yao, I. Jurisica, and I. Stagljar, “Fundamentals of protein interaction network mapping,” *Molecular Systems Biology*, vol. 11, no. 12, Dec. 2015.

[4] Bader, G.D., Hogue, C.W. “An automated method for finding molecular complexes in large protein interaction networks.” *BMC Bioinformatics* **4,** 2 (2003) doi:10.1186/1471-2105-4-2

[5] Z. He, "Protein complex identification from AP-MS data", *Data Mining for Bioinformatics Applications*, pp. 61-68, 2015. Available: 10.1016/b978-0-08-100100-4.00007-7.

[6] M. Girvan and M. E. J. Newman, “Community structure in social and biological networks,” *Proceedings of the National Academy of Sciences*, vol. 99, no. 12, pp. 7821–7826, Nov. 2002.

[7] A. Clauset, M. E. J. Newman, and C. Moore, “Finding community structure in very large networks,” *Physical Review E*, vol. 70, no. 6, Jun. 2004.

[8] M. Newman and M. Girvan, "Finding and evaluating community structure in networks", *Physical Review E*, vol. 69, no. 2, 2004. Available: 10.1103/physreve.69.026113.