Glioblastoma Gene Interaction Visualization Dashboard

LDATA 2010
Information Visualization

STUDENTS:

Solari Costa, Joaquín	NOMA: 26412100
Vergniaud, Agustín	NOMA: 00412100

2021 - 2022

Introduction

The aim of this report is to describe the software designed to analyze a dataset related with Glioblastoma cancer genes, which can be found at https://thebiogrid.org/project/5/glioblastoma.html. This dataset represents interactions between genes and proteins, together with the information collected about each gene.

This analysis and software development is useful for providing a network visualization for such an important topic as treating brain cancer. Finding relationships between different genes, expressed as nodes and edges, helps to the studies of possible arising illnesses.

Glioblastoma (GBM) is the most common malignant brain tumor that accounts for 15% of all primary brain tumors and 45% of primary malignant brain tumors. It has a poor prognosis of only 5% survival at 5 years with a median survival of 14-15 months.

A primary brain or spinal cord tumor is a tumor that starts in the brain or spinal cord. This year, an estimated 24,530 adults (13,840 men and 10,690 women) in the United States will be diagnosed with primary cancerous tumors of the brain and spinal cord. A person's likelihood of developing this type of tumor in their lifetime is less than 1%. Brain tumors account for 85% to 90% of all primary central nervous system (CNS) tumors.

Brain and other nervous system cancer is the 10th leading cause of death for men and women. It is estimated that 18,600 adults (10,500 men and 8,100 women) will die from primary cancerous brain and CNS tumors this year.

Understanding the Dataset

The dataset is composed of four files:

- *BIOGRID-PROJECT-glioblastoma_project-GENES.projectindex.txt*
- *BIOGRID-PROJECT-glioblastoma_project-INTERACTIONS.tab3.txt*
- *BIOGRID-PROJECT-glioblastoma_project-PTM.ptmtab.txt*
- *BIOGRID-PROJECT-glioblastoma_project-CHEMICALS.chemtab.txt*

The files contain the full information about the nodes and edges that are going to be analysed. The *GENES* file displays the information each node has in the *BIOGRID*[1] *and Entrez*[2] *Database*. The *INTERACTIONS* file displays the information of how nodes interact with each other *and their specific reactions.* The *PTM* file presents the information each node has in the *BIOGRID and Entrez Database* regarding *post-translational modifications*[3]. The *CHEMICAL* file shows the information each node has in the *BIOGRID and Entrez Database* regarding chemical treatments and its possible reactions to them.

Visualization Building

To create the software Python was implemented. The libraries that were used inside Python were Dash, NetworkX, Cytoscape.js, Plotly, Matplotlib, Numpy, Pandas and Json. The final result is a

[1] The Biological General Repository for Interaction Datasets (BioGRID) is an open access database that houses genetic and protein interactions curated from the primary biomedical literature for all major model organism species and humans.
[2] Entrez is a molecular biology database system that provides integrated access to nucleotide and protein sequence data, gene-centered and genomic mapping information, 3D structure data and PubMed MEDLINE.
[3] A post-translational modification (PTM) is a biochemical modification that occurs to one or more amino acids on a protein after the protein has been translated by a ribosome.

representation on html. Here, the user can display the dataset on a graph with different layouts and can collect information about each gene individually and about the relationship between them.

Several buttons and dropdowns are provided to the user to permit the selection of the features as desirable. As the location of these represent an integral aspect in the design of the application, they are located in tabs. As the network graph is the central part of the software, it is located in the center of the screen, while the buttons and dropdowns are located under it.

Features

The app provides different features to the user to interact and choose, useful for network graph analysis.

- **6 Different displayable layouts.**
- **Network analysis algorithms.**
- **Color scaling according to different network analysis algorithms.**
- **Clickable nodes and displayable information.**
- **Draggable nodes.**
- **Zoom in and out.**
- **Filter displayable nodes according to number of connections.**
- **Following and Follower arrows between nodes and edges.**
- **Shortest path shown between nodes when selected.**
- **Node source and target information in database.**
- **Histogram according to the weight assigned with the network algorithm chosen.**
- **Scatterplot according to the weight assigned with the network algorithm chosen.**
- **Real time visualization update as the user chooses preferences.**
- **JSON code information.**

Feature Analysis

6 Different displayable layouts

A dropdown is provided to the user to select a layout for visualizing the network graph.

Geometrical Layouts

The **grid** layout is an inexpensive layout that easily shows all nodes in the graph, so it is a natural default. It allows the user to visually verify that the graph has been correctly loaded.

The **circle** layout organises the nodes into a circle. By default, the nodes are placed clockwise from the 12 o'clock position, in the order that they are passed to the layout. It is useful to let the user see the data in a more organized way, as the layout is zoomed out and arranged into one only circle.

Hierarchical Layouts

Hierarchical layouts work well for trees and directed acyclic graphs (DAGs).

The **concentric** layout organises the nodes into concentric circles, based on the specified metric. The nodes with the highest metric values are placed in the innermost circle, and the metric values of the nodes descend for each outward circle. Each circle has nodes with metric values between a specified range, with the nodes within a circle sorted accordingly. This layout is useful for highlighting relative importance of the nodes as it sets the nodes with the most neighbours in the centre.

The **breadthfirst** is a hierarchical-typed layout that organises the nodes in levels. The data is displayed as a tree, that is very useful for the nodes in the top, that are the central ones in the graph node union.

Force-directed layouts

A ***force-directed*** graph drawing algorithm - also known as *spring-embedder* or *energy-based placement* algorithm - arranges graphs in an organic and aesthetically pleasing way. The resulting diagrams often expose the inherent symmetric and clustered structure of a graph and show a well-balanced distribution of nodes with few edge crossings.
The **algorithm** is based on a physical model. Nodes are represented as points in a plane that are electrically charged and apply repulsive forces against each other. Edges connect these points simulating a spring-force, attracting adjacent nodes. The model iteratively determines the resulting forces that act on the nodes and try to move the nodes closer to an equilibrium where all forces add up to zero, and the position of the nodes stays stable.
CoSE and CoSE Bilkent: They are both force-directed layouts that work on the bases of node attraction and repulsion. The **CoSE Bilkent** is a better implementation of **CoSE**, but also takes in account more processing time. These layouts are the most useful to evaluate the graph correctly as nodes are repelled from each other permitting a better visualization of its edges and connections.

Network analysis algorithms.

When displaying interactions in graphs, it is important not only to organize them in proper layouts but also to run algorithms that can deliver useful information. The ones chosen for the software are the following.
Degree centrality: refers to the number of edges attached to the node. In order to know the standardized score, it is needed to divide each score by n-1 (n = the number of nodes). With this basic algorithm the user can identify easily the most popular nodes.
Closeness centrality: is the average length of the shortest path between the node and all the other nodes in the graph. This measure can be interpreted as the time it takes for something flowing through the network to reach its destination. It can also be interpreted as how fast it will take for information to propagate from one node to all the others. Closeness measures in some way the accessibility of a node in the network.
Betweenness centrality: quantifies the number of times a node acts as a bridge along the shortest path between two other nodes. vertices that have a high probability to occur on a randomly chosen shortest path between two randomly chosen vertices have a high betweenness.
Clustering coefficient: the local clustering coefficient for a vertex is then given by a proportion of the number of links between the vertices within its neighbourhood divided by the number of links that could possibly exist between them. This algorithm quantifies how close its neighbours are to being a clique (complete graph). It is useful to determine whether a graph is a small-world network.
Google PageRank: this algorithm was originally invented to rank web pages according to their popularity in their search engine results. It has to calculated recursively and the formula would be the following one:

$$PR(u) = \sum_{v \in B_u} \frac{PR(v)}{L(v)},$$

where PR(u) is the PageRank for u, PR(v) is the PageRank of each page v that has a connection that goes from v to u, and finally L(v) is the total number of outgoing edges from page i (whether they are towards u or not). This powerful tool could help understand the user whether there are nodes that do not have too many interactions but are strongly connected to central nodes.

Color scaling according to different network analysis algorithms.

According to the different color scales offered to the user, the network analysis algorithms are processed and each node is given a weight. This gives the user the possibility to rank nodes by their color to know their properties.

The color scales selectable by the user are:

- **Wistia**
- **Jet**
- **Binary**
- **Greens**

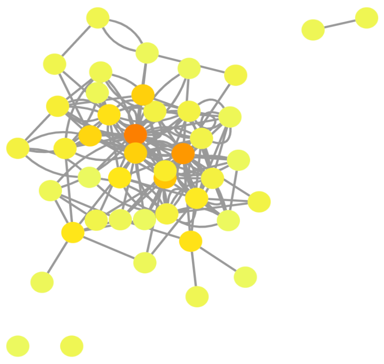

Image 1: Display of the filtered graph in Wistia scale, weighted by the Degree Centrality Algorithm.

By looking at the color of the nodes, the user can see the node that has the most connections, in relation to its degree centrality weight.

Clickable nodes and displayable information. Draggable nodes. Zoom in and out.

The user can interact with the nodes in many ways. Firstly, when a node is clicked, the software highlights the edges and the nodes connected to the one clicked. The second feature is that all the information related to the clicked node appears in the *Information* tab. Nodes are also draggable and movable to isolate them from each other. The last feature related is that the user can zoom in and out to see in better detail the connections.

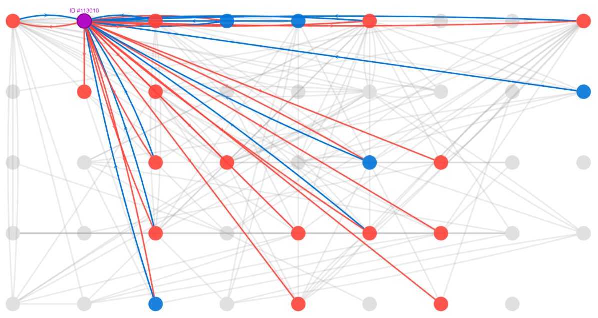

Image 2: Display of the graph when the node 113010 is selected.

Filter displayable nodes according to number of connections.

As the dataset is very large, the user may find that displaying a graph with almost two hundred nodes and more than thirty six edges does not show useful information. Consequently, a way to filter nodes was added. The user simply has to write the minimum number of edges that a node should have to be shown in the graph. This makes it possible to filter according to the desired information to visualize and also to optimize the performance of the application.

Following and Follower arrows between nodes and edges.

Whenever a node is clicked and consequently selected, its connected edges will appear with arrows indicating whether the connection is incoming or outgoing to that node. The utility of this being possible to analyse the dual connections in the Database between genes when selected.

Node source and target information in database.

In the *Information* tab the user can find all the information collected in the database, related to a selected node. This information includes all the available information from the nodes clicked (source and targets) in the BioGrid Database as well as its chemical properties, number of interactions or post translational modifications.

Histogram according to the weight assigned with the network algorithm chosen.

In the last tab, “Histogram”, the user can find an histogram of the values for each node computed by the algorithm chosen. By hovering the mouse over the histogram bars, it is possible to see information such as the bar range and the value of each bar.

This functionality permits the user to evaluate the frequencies of most nodes in the network graph. This is valuable information because, for example, in the graph below, we see most of the Glioblastoma Genes are accessible for the other Genes in the Database. In the histogram we can see clearly that most of the nodes round the weight of 0.6 in the closeness centrality algorithm.

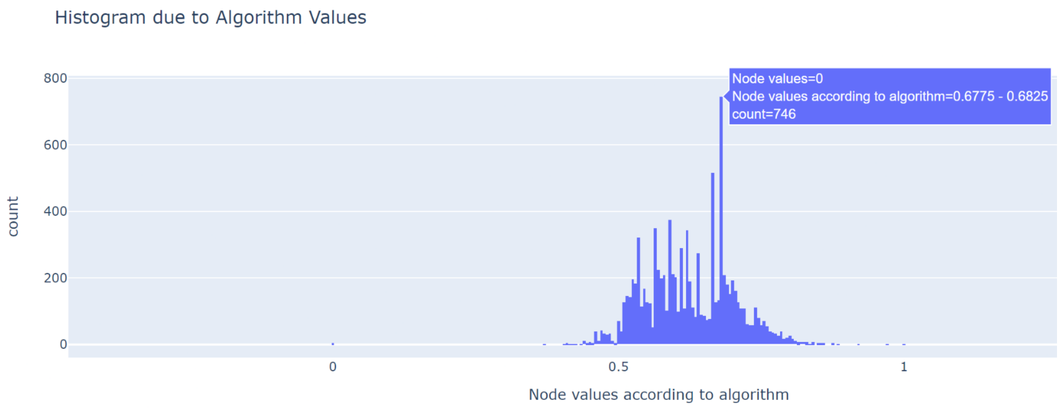

Image 3: Histogram for the closeness centrality method.

Scatterplot according to the weight assigned with the network algorithm chosen.

In the same tab, the user can also find a scatterplot to show practical information like the number of neighbours for each node in relation to the value computed by the algorithm chosen to make the analysis.

This is useful for the user as it is possible to see the behaviour of the network graph varying one parameter as the algorithm weight is and a stable one as the number of neighbours is.

The graph below reflects the accessibility of each node in the network (closeness centrality), which permits us to see that when the number of neighbours is bigger, nodes are more likely to have a bigger weight value.

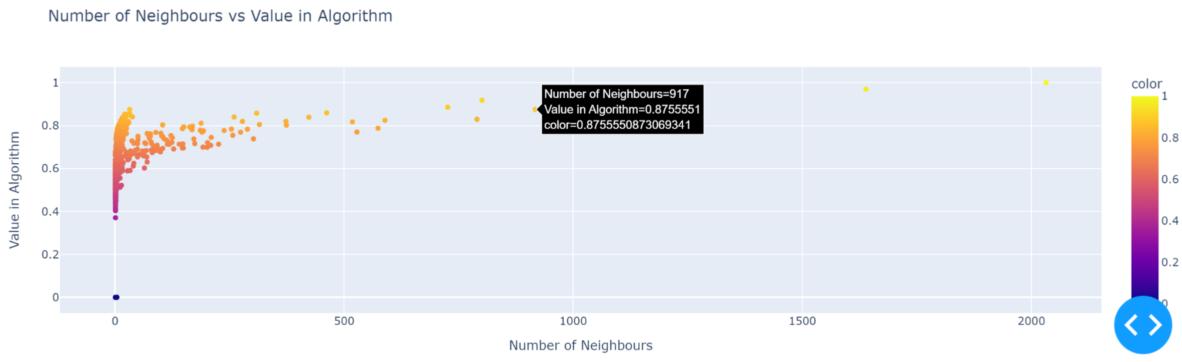

Image 4: Scatterplot for the closeness centrality method.

Real time visualization update as the user chooses preferences.

Finally, it is important to remark that all the changes that the user is doing while manipulating the software are updated in real time so it is possible to analyze the differences between the setups, layouts and algorithms selected by the user.

JSON code information.

To permit the user see the information down the software, a JSON tab is provided. In this tab, real-time changing information is shown related to the code.

User interactive experience and future possible implementations.

The software permits the user to select information and analyze it according to its personal desire to make conclusions about the Gene Glioblastoma Dataset. It is possible to visualize the network graph according to Size, Color, Layout, Algorithm and Connection Filter preferences. Also, with a click, they can obtain all the information related to a specific node and the shortest connection paths with its targets.

For future implementations and improvements, a useful gadget would be to permit the user to modify the attributes in the Force Directed Layouts used, such as Force Atlas is in Gephi. Also, it would be very useful to have a Data Laboratory in which it is possible to view the Data table as well as the adjacency matrices used to implement the algorithms. Having a Data Laboratory would also permit the user to add more nodes and edges manually to the Database from the software. In this way it is possible for the person to update the dataset manually according to Gene discoveries or conclusions.

Bibliography

Algorithms:
https://moodle.uclouvain.be/pluginfile.php/183431/mod_resource/content/2/lecture_aalto.pdf
https://moodle.uclouvain.be/pluginfile.php/183432/mod_resource/content/2/mcguffin-2012-simpleNetVis.pdf
https://towardsdatascience.com/large-graph-visualization-tools-and-approaches-2b8758a1cd59
https://www.researchgate.net/publication/260653607_Simple_algorithms_for_network_visualization_A_tutorial

Layouts:
https://manual.cytoscape.org/en/stable/Navigation_and_Layout.html
https://blog.js.cytoscape.org/2020/05/11/layouts/

Dash Cytoscape:
https://dash.plotly.com/cytoscape
https://morioh.com/p/edaf8226d82f
https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.800.333&rep=rep1&type=pdf
https://medium.com/plotly/exploring-and-investigating-network-relationships-with-plotlys-dash-and-dash-cytoscape-ec625ef63c59

Dash:
https://dash.plotly.com/
https://dash.plotly.com/dash-core-components/input
https://realpython.com/python-dash/
https://pythonprogramming.net/data-visualization-application-dash-python-tutorial-introduction/
https://github.com/plotly/dash
https://github.com/ucg8j/awesome-dash

NetworkX:
https://www.geeksforgeeks.org/network-centrality-measures-in-a-graph-using-networkx-python/
https://networkx.org/documentation/stable/tutorial.html
https://medium.com/swlh/a-tutorial-on-networkx-network-analysis-in-python-part-i-43c1d35830b6
https://www.datacamp.com/community/tutorials/networkx-python-graph-tutorial