cyTRON/JS pipeline

July 19, 2019

Introduction

cyTRON/JS is a web application which provides an interface to the R library TRONCO. In addition to what can be achieved through R, this tool offers an interactive visualization of the cancer progression models, making it possible to easily retrieve nodes information.
This is a tutorial which includes a case studies carried out through cyTRON/JS.

Case study: TCGA-prostate

This case study aims at reconstructing a cancer progression model starting from mutation data downloaded from TCGA.
All the data used in the following tutorial can be downloaded at `https://github.com/BIMIB-DISCo/cyTRON-js/tree/master/examples/TCGA-prostate`.
First, in order to proceed further in the analysis, the user needs to create an account or to sign-in from the home page. The only information required is a username and a password. The registration is necessary in order for the user to be able to access all the analysis he or she has carried out.

Input selection - 1 Figure 1 shows the page that is displayed after inserting the study name. There are three different possible formats for the input files: `MAF, GISTIC` and a custom Boolean file, which should contain a matrix indicating which mutation is present in each sample. For this example we are going to use the first two formats. In particular, the `MAF` file contains data about SNPs, Insertions and Deletions and the `GISTIC` file contains data about CNVs.

Input selection - 2 Figure 2 displays the second slide of the input selection. Here a user can decide to upload a list of driver genes to take into consideration during the analysis. This step can speed up the computation, as only genes contained in this list will be taken into consideration. The cancer type involved in the study may also be divided into different **subtypes**: to achieve this goal, a user can upload a cluster file, which should contain one or more columns that indicate the subtype corresponding to each sample.

Since TRONCO does not provide functions that identify tumor subtypes and driver genes, users should rely on external tools to produce this kind of files when carrying out their own analysis.
The files that need to be uploaded at this step are `gene_list.txt` and `clustering.csv`.

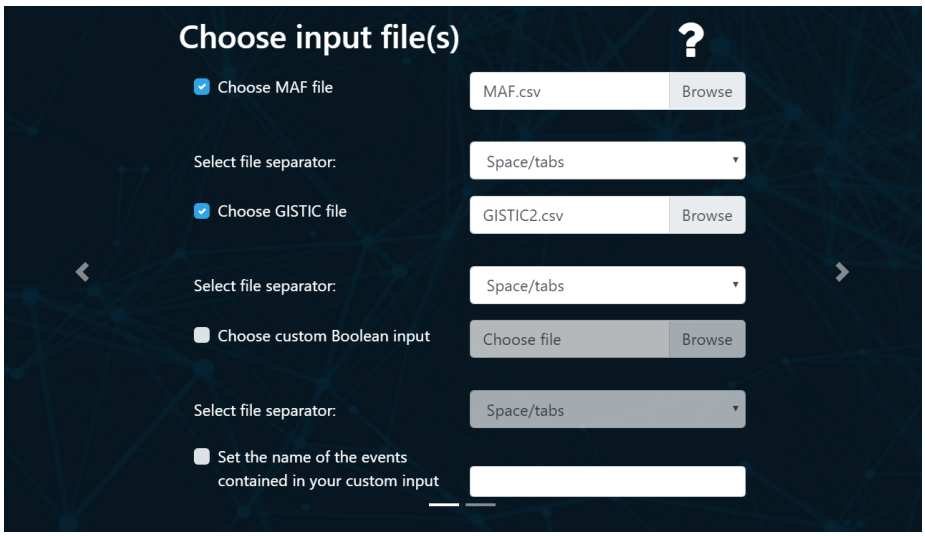


Figure 1: Input selection - 1

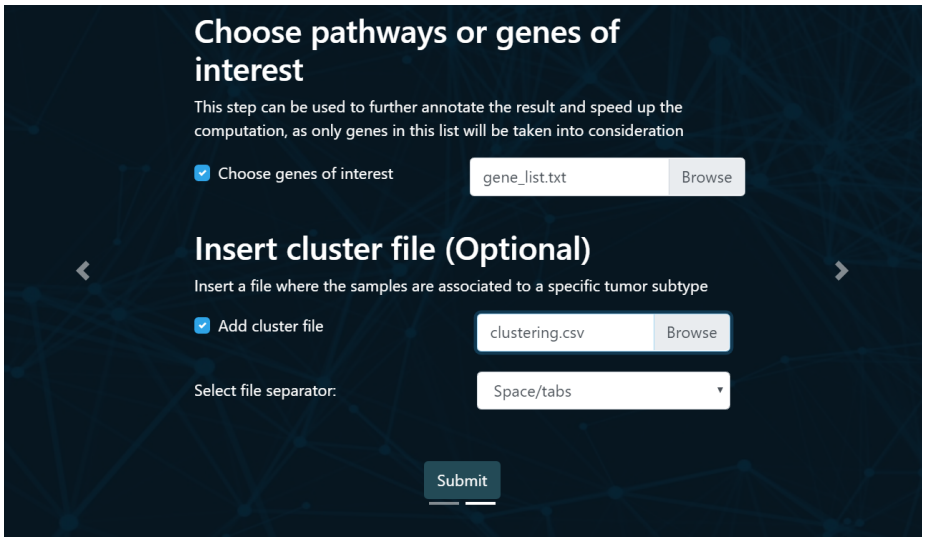


Figure 2: Input selection - 2

Cluster selection Figure 3 displays the page shown after the input files have been uploaded. Here the user has access to the names of the columns contained in the cluster file, in order to address the columns that contain the samples IDs and the cancer subtype associated to each sample.

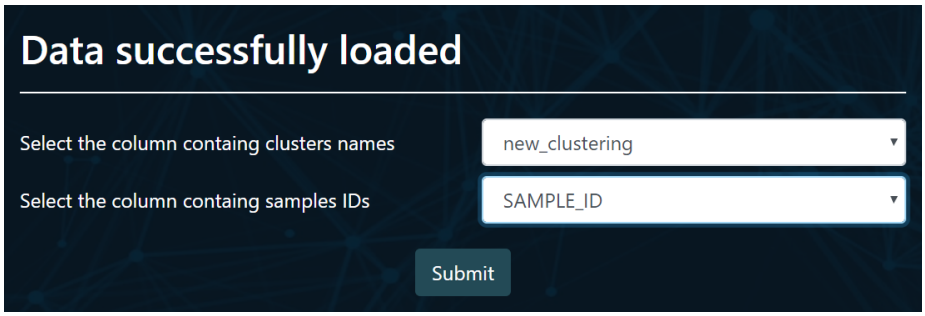
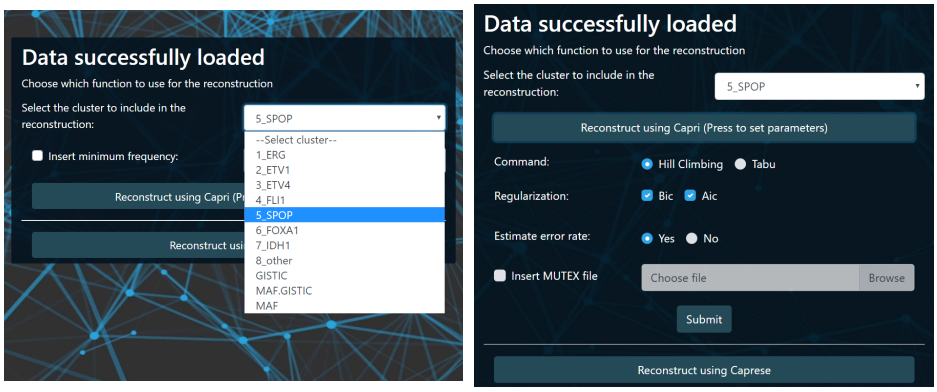


Figure 3: Clustering

Model reconstruction After the clustering step, the user will be redirected to a page for the model reconstruction. Figures 4a and 4b show the interface for the reconstruction: **first**, the user needs to select which one among the **clusters** will be involved in the reconstruction. **Second**, the user can decide which procedure to use. At the moment there are two **algorithms** available: `CAPRI`, which is indicated for ensemble level analysis, and `CAPRESE`, which is more indicated for individual level analysis.
Since the data used for this tutorial contain mutations from a cohort of patients, `CAPRI` is chosen. Thus, the following parameters need to be set:

- `Command`: it defines the heuristic search to be performed.
- `Regularization`: it selects the function to use for the likelihood estimation.
- `Error rate`: it enables or disables the estimation of the error rates given the reconstructed model.
- `Mutex`: this can be used to insert a file containing a list of mutually exclusive mutations, created through the MUTEX tool. For more information visit `https://github.com/PathwayAndDataAnalysis/mutex`.



(a) Cluster selection

(b) Parameters setting

Figure 4: Model reconstruction

Model plot Figure 5 shows the page displayed after the reconstruction has finished. Here the user can decide to either reconstruct another model or to **plot** one which has already been constructed. To plot a model, the user needs to indicate which confidence values have to be displayed for every edge, if the *prima facie* model needs to be displayed and whether disconnected nodes should be included in the visualization or not.

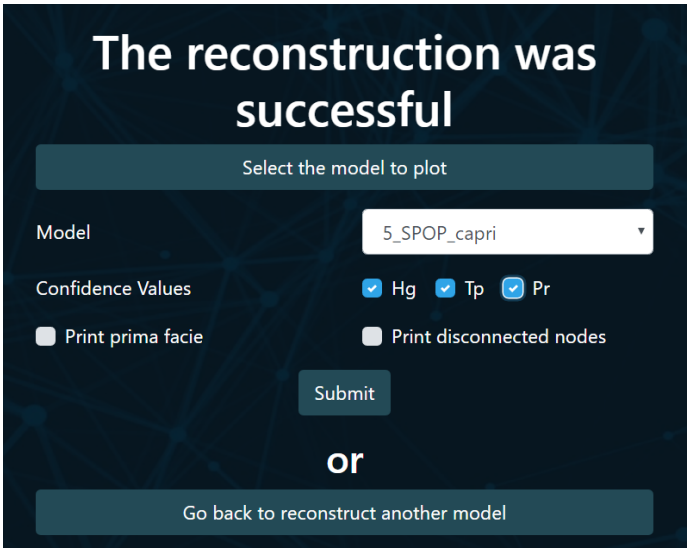


Figure 5: Model plot

Visualization After the previous step, a `.graphml` file containing the progression model is created, which is then displayed in the page shown in figure 6. Here a user can click on the model's nodes to get more information about a gene and can read confidence information for each edge.

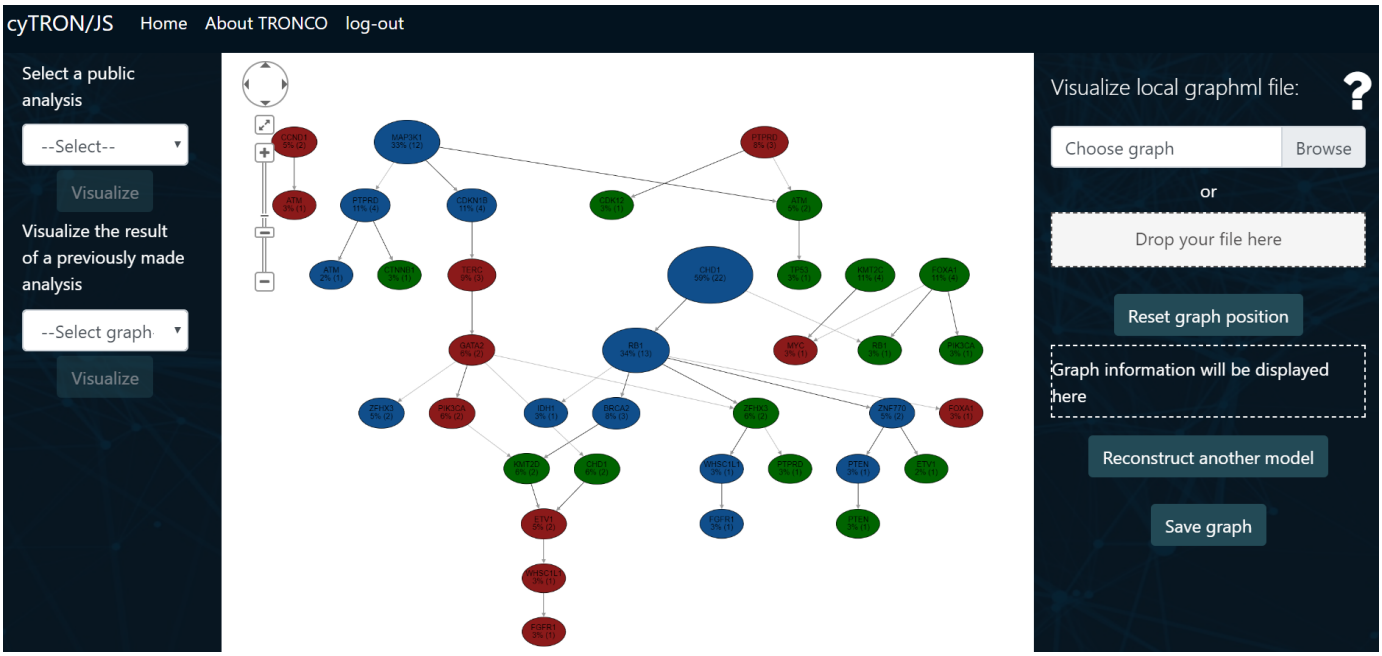


Figure 6: Model visualization

Exploiting the interactive view

Here we present an example of how cyTRON/JS can serve as a means to identify associations among genes. Indeed, by clicking on a node of the graph users can read additional information about the corresponding gene and they can exploit this information to find which genes are involved in the same processes.
The example is based on the ERG subtype of the prostate cancer, whose graph can be reconstructed using the CAPRI algorithm following the steps reported in the previous section.
The observations that stood out while exploring the progression model are the following:

- There is an oriented edge which connects RB1 to TP53. The protein encoded by the former serves as a negative regulator for cell cycle, and the one encoded by the latter has a role in inducing cell cycle arrest, apoptosis and senescence.

- Another oriented edge connects RB1 to PTEN. The latter encodes a protein which is a negative regulator of the AKT/PKB signaling pathway.

Therefore through cytRON/JS interface it was possible to identify two connected nodes that represent genes which both have a role in cell cycle.
Even though these might be trivial associations, this is an example of how this tool can be exploited to better understand the role of genes that are connected in the graph, possibly finding interesting associations.
Given the fact that some researchers may want to access cyTRON/JS only to explore some already constructed model, the web application contains a section which allows users to access without authenticating, in order to visualize one of their local `graphml` files.