INfORM Execution Steps

Run INfORM

Install Dependencies

```
#Install CRAN dependencies
cran_pkgs <- c("V8", "RSQLite", "TopKLists", "doParallel", "foreach", "igraph", "plyr", "shiny",
"shinyjs", "shinyBS", "shinydashboard", "colourpicker", "DT", "R.utils", "treemap", "visNetwork",
"abind", "radarchart", "randomcoloR", "Rserve", "WriteXLS", “gplots”, “ggplot2”)
cran_pkgs.inst <- cran_pkgs[!(cran_pkgs %in% rownames(installed.packages()))]
if(length(cran_pkgs.inst)>0){
        print(paste0("Missing ", length(cran_pkgs.inst), " CRAN Packages:"))
        for(pkg in cran_pkgs.inst){
                print(paste0("Installing Package:'", pkg, "'..."))
                install.packages(pkg, repo="http://cran.rstudio.org", dependencies=TRUE)
                print("Installed!!!")
        }
}

#Install Bioconductor dependencies
if(!"GOSemSim" %in% rownames(installed.packages())){
        print("Installing GOSemSim from GitHub!")
        devtools::install_github("GuangchuangYu/GOSemSim")
}

source("http://bioconductor.org/biocLite.R")
bioc_pkgs <- c("org.Hs.eg.db", "org.Mm.eg.db", "GO.db", "AnnotationDbi", "GSEABase", "minet")
bioc_pkgs.inst <- bioc_pkgs[!(bioc_pkgs %in% rownames(installed.packages()))]
if(length(bioc_pkgs.inst)>0){
        source("http://bioconductor.org/biocLite.R")
        print(paste0("Missing ", length(bioc_pkgs.inst), " Bioconductor Packages:"))
        for(pkg in bioc_pkgs.inst){
                print(paste0("Installing Package:'", pkg, "'..."))
                biocLite(pkg, suppressUpdates=TRUE)
                print("Installed!!!")
        }
}
```

Launch from GitHub

Load 'shiny' library

```
library(shiny)
```

Using runGitHub

```
runGitHub("INfORM", "Greco-Lab", subdir="INfORM-app")
```

Using the archived file

```
runUrl("https://github.com/Greco-Lab/INfORM/archive/master.tar.gz", subdir="INfORM-app")

runUrl("https://github.com/Greco-Lab/INfORM/archive/master.zip", subdir="INfORM-app")
```

Launch locally

Clone the git repository

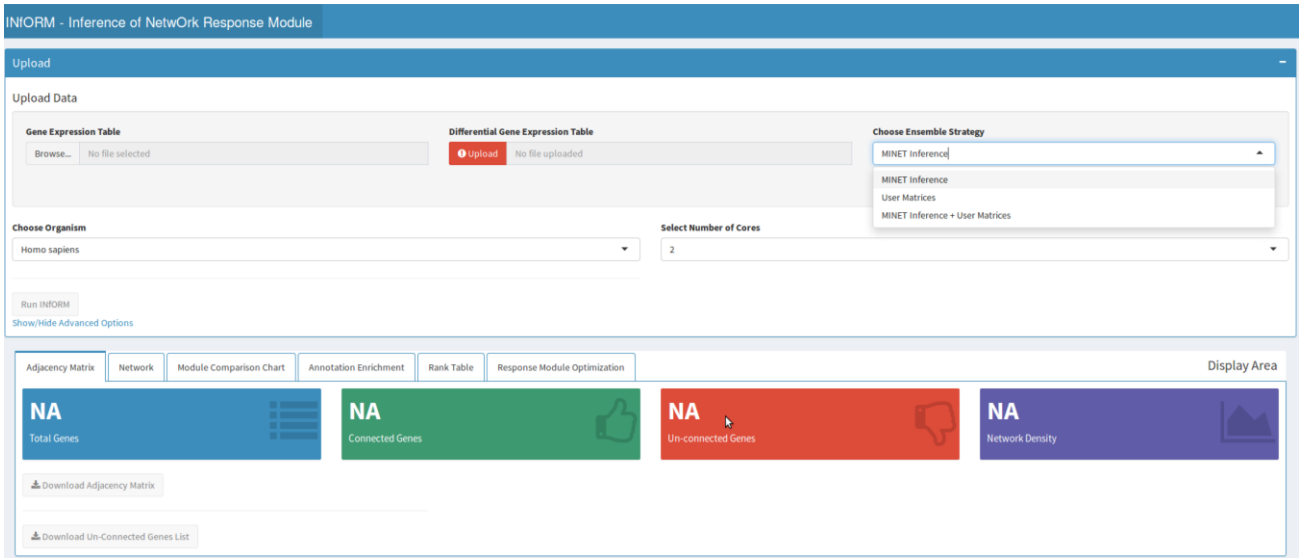
git clone https://github.com/Greco-Lab/INfORM INfORM_clone

Run by using runApp()

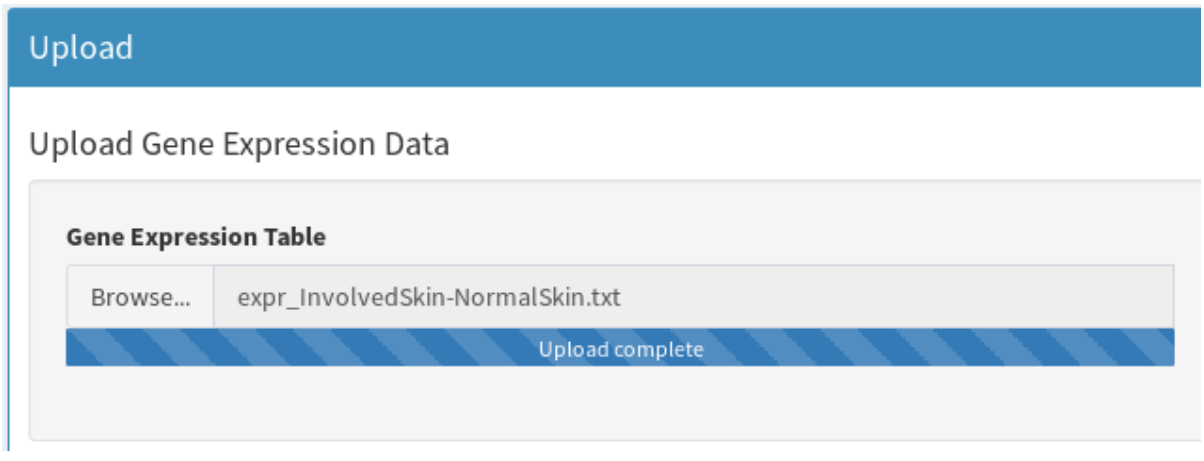
setwd("~/INfORM_clone")

runApp("INfORM-app/")

Upload

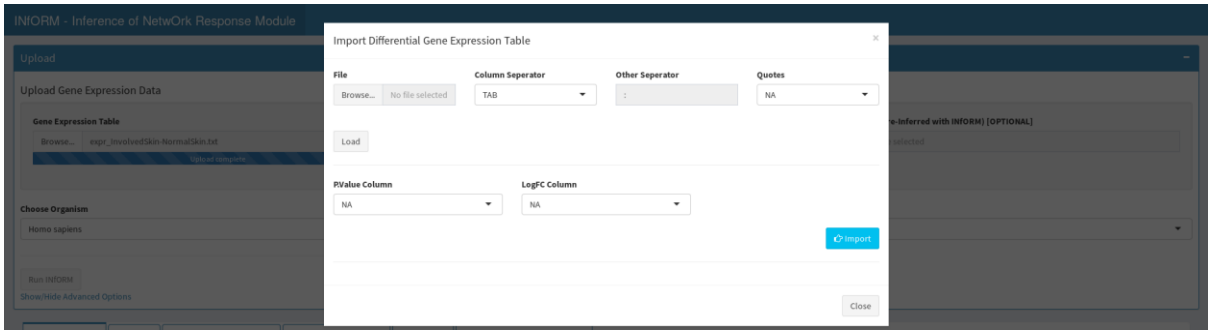


Load gene expression table.



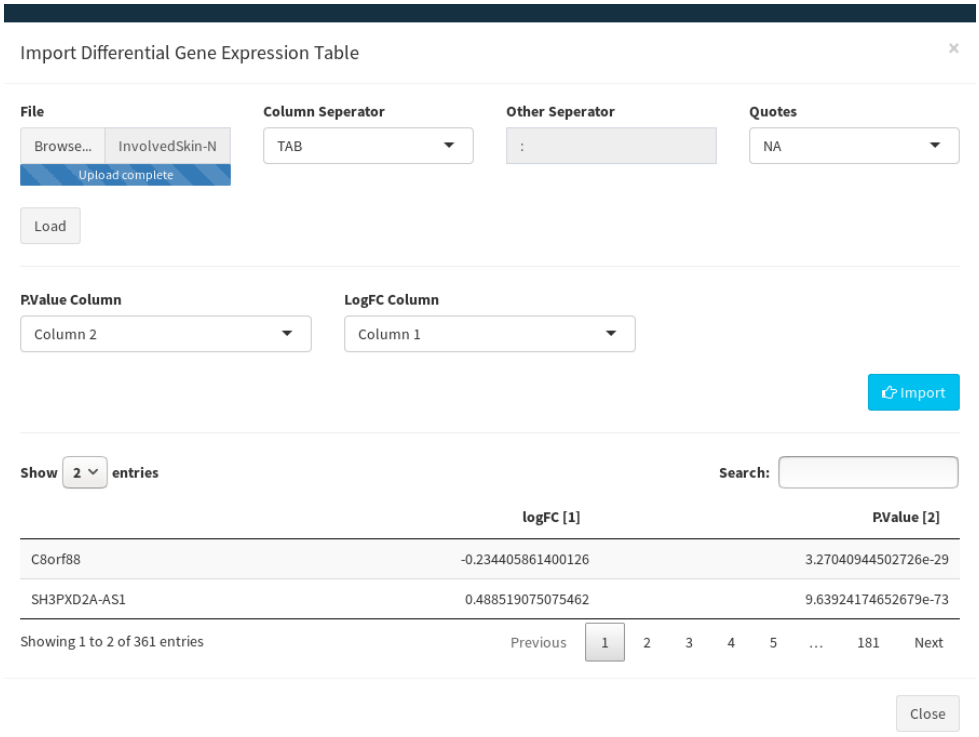
Gene expression table must have genes in rows and samples in a column while HGNC gene symbol must be provided as row names and sample names must be provided as column names. Genes in the expression should match the genes provided in the differential expression table in size and identity.

Load differential expression table.



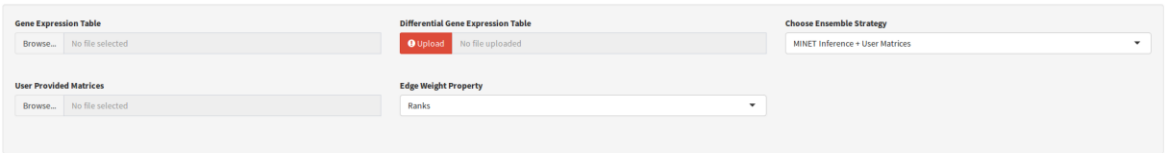
Differential expression table must have genes in rows and differential P.Value and Log2(FC) in columns, HGCNC gene symbol must be used as row names, and column names should be chosen appropriately to match their content.

Specify columns for P.Value and LogFC in the differential expression table.



While loading the differential expression table care must be taken to correctly specify the columns P.Value and Log2(FC) in the graphical interface.

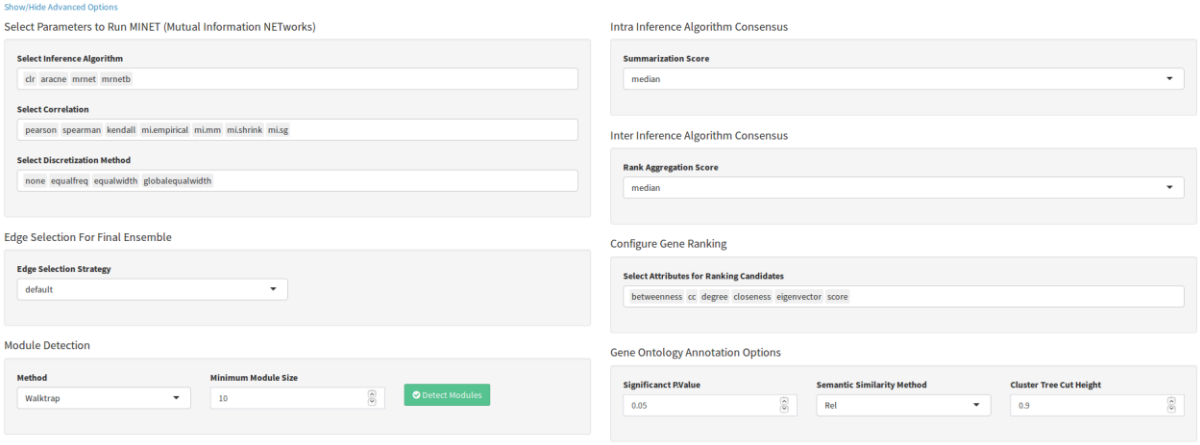
Choose ensemble strategy and upload pre-inferred co-expression matrices



INfORM allows three options for ensemble strategy. “MINET Inferences” option infers co-expression matrices by using the inference algorithms from MINET R package. “User Matrices” option allows the user to upload a set of co-expression matrices which can be inferred by using any inference algorithm and strategy preferred by the user. And “MINET Inference + User Matrices” option uses

the co-expression matrices inferred by using the inference algorithms in MINET R package in combination with the user provided co-expression matrices to create a consensus network.

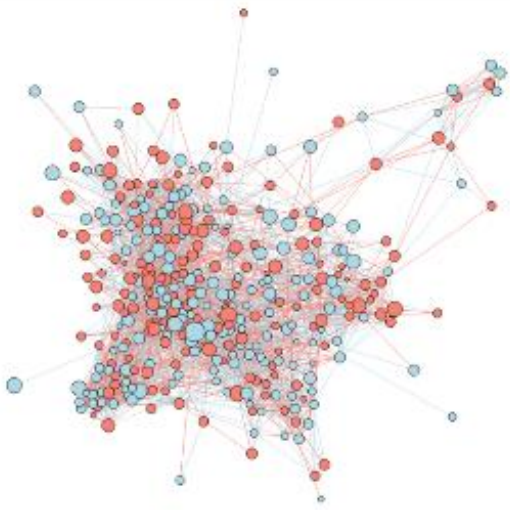
Advanced parameters

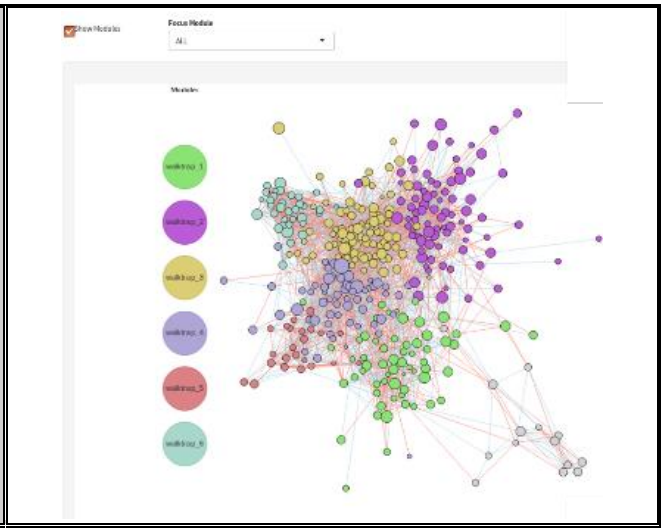
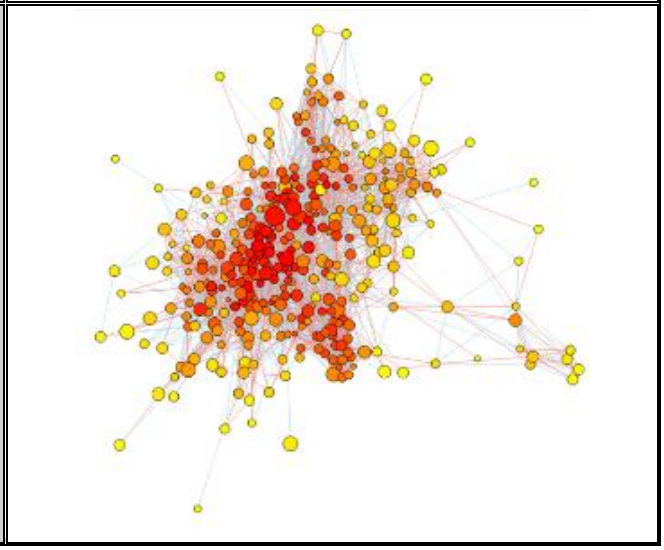


The suite of advanced parameters allows configuring different aspects for INfORM's analysis. The user can setup inference of co-expression matrices from MINET package. There are multiple options for summarization scores for intra inference algorithm consensus creation. Different options for rank aggregation scores are provided which are used in creating a consensus for inter inference algorithm. The user can choose the edge selection strategy to specify how many top ranked edges are selected in the ensemble network. Ranking of genes can be configured by selecting centrality scores and biological score to use for ranking. Module detection is configurable by selection of community detection method of choice and minimum module size. Biological significance by gene ontology is configured by gene ontology enrichment as well summarization options.

Visualization

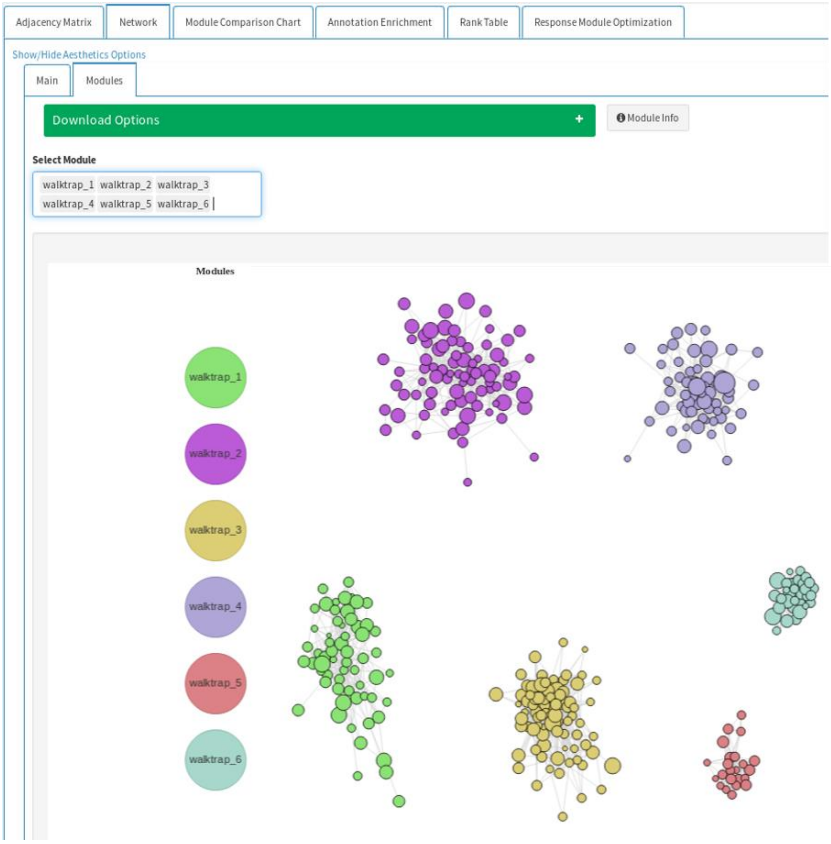
Inferred network.

DEFAULT	

COLORED BY MODULES	
COLORED BY RANK	

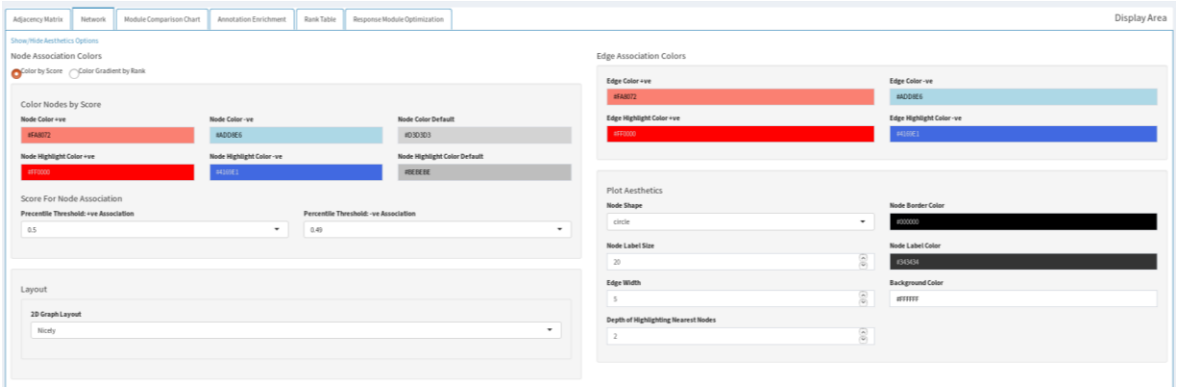
The inferred consensus gene network is represented as a vizNetwork where circles represent genes, and the lines represent the connection between the genes. In the default setting the nodes are colored by differential expression score, and the edges are colored to show the positive or negative correlation between the genes which was computed by using *cor()* function on the gene expression table. The second option is to color the nodes by the rank of the gene in the network which is represented as a color gradient. The third option is to add the module information to the network and differentially color the nodes as per their membership in the modules.

Module specific network view for one or more selected module(s).



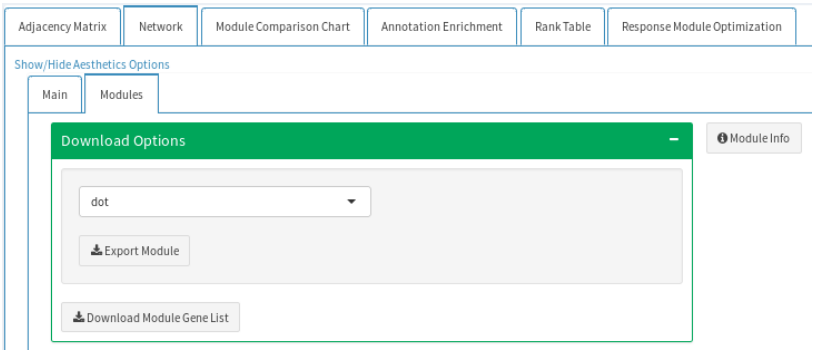
A separate network view area provided where the user can choose to see the specific module(s). The user can select one or more module, while the module colors and network layout specifications are same as the whole network view.

Aesthetics options can be used to change network display.



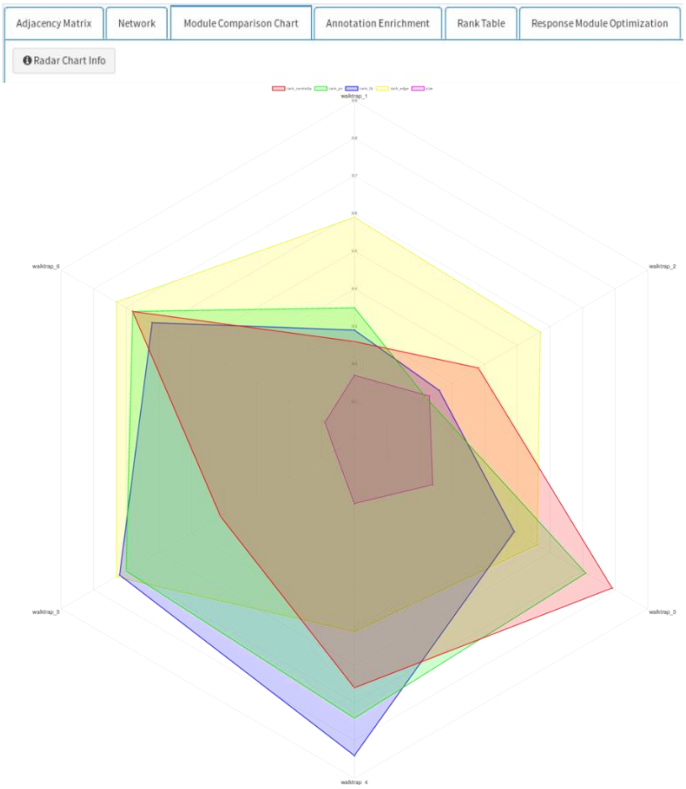
There is a gamut of options to specify node and edge color schemes, network layout, node shapes, node label size, node label color, node border color, edge width, background color, and the vizNetwork property to highlight nearby nodes on mouse over.

Export the whole network or the separate modules in any igraph supported format.



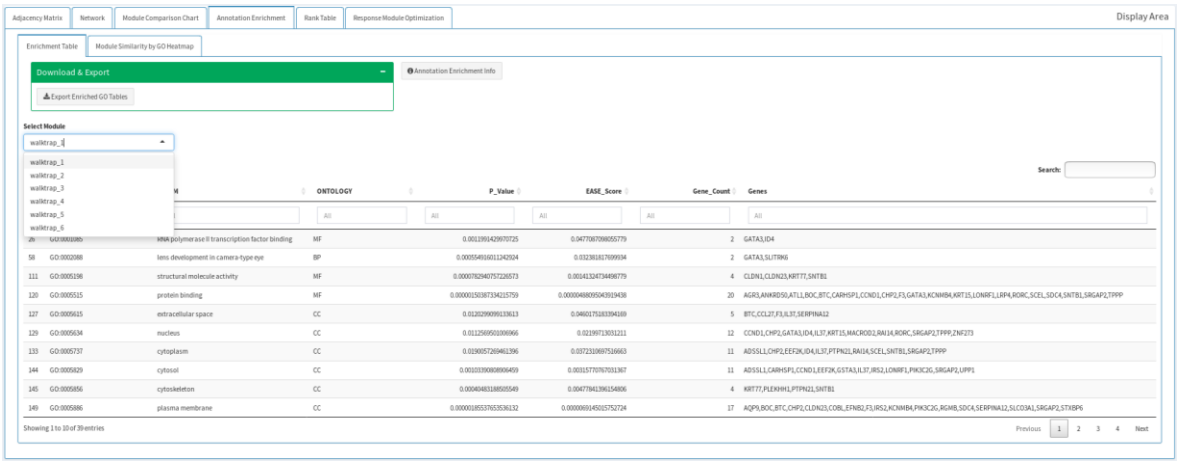
The whole network from the Main network tab and the module specific network from the Modules network can be exported in network formats supported by igraph package. Also, the user can choose to list of genes present in the network if they wish.

Visualize module annotation as a radar chart.



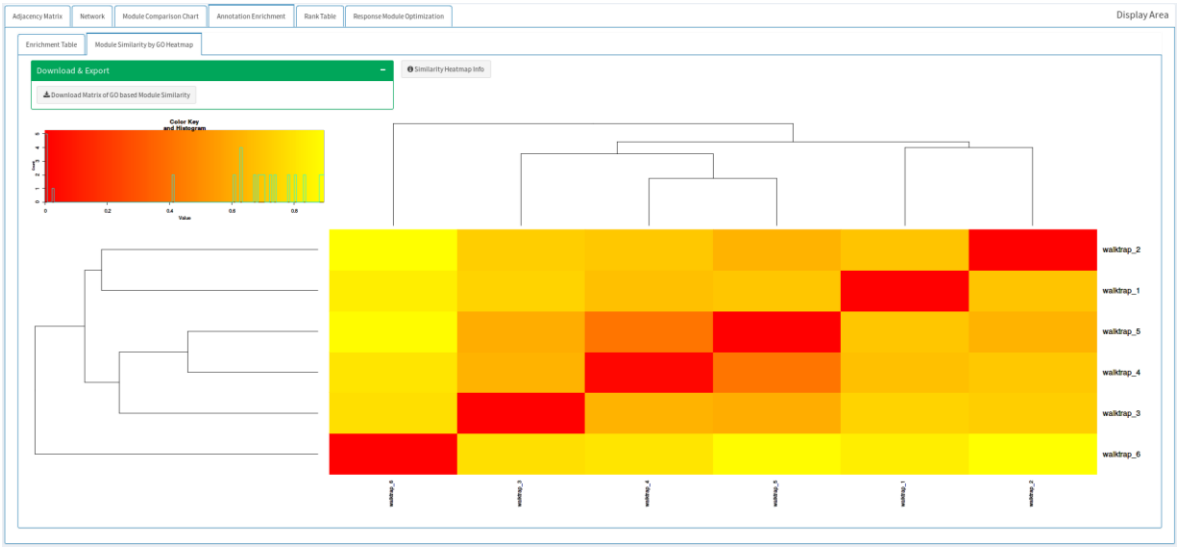
The identified modules can be evaluated from the Module Comparison Chart which is a radar chart where radii represent the modules and the areas drawn within the radii represent the different score metrics. The metrics for evaluation are node ranks (centrality, differential P.Value, differential Log2(FC)), edge rank and module size. The user can export the ranked gene table for the whole network and the modules as a spreadsheet.

Get GO terms overrepresented in each module.



GO (Gene Ontology) terms overrepresented in each module can be viewed in the Annotation Enrichment tab, the user can choose a specific module to view the GO terms table in the graphical interface where the user can order the table by specific columns and search in the whole table or a specific column of interest. This table structure can be exported for all the identified modules altogether as a spreadsheet file.

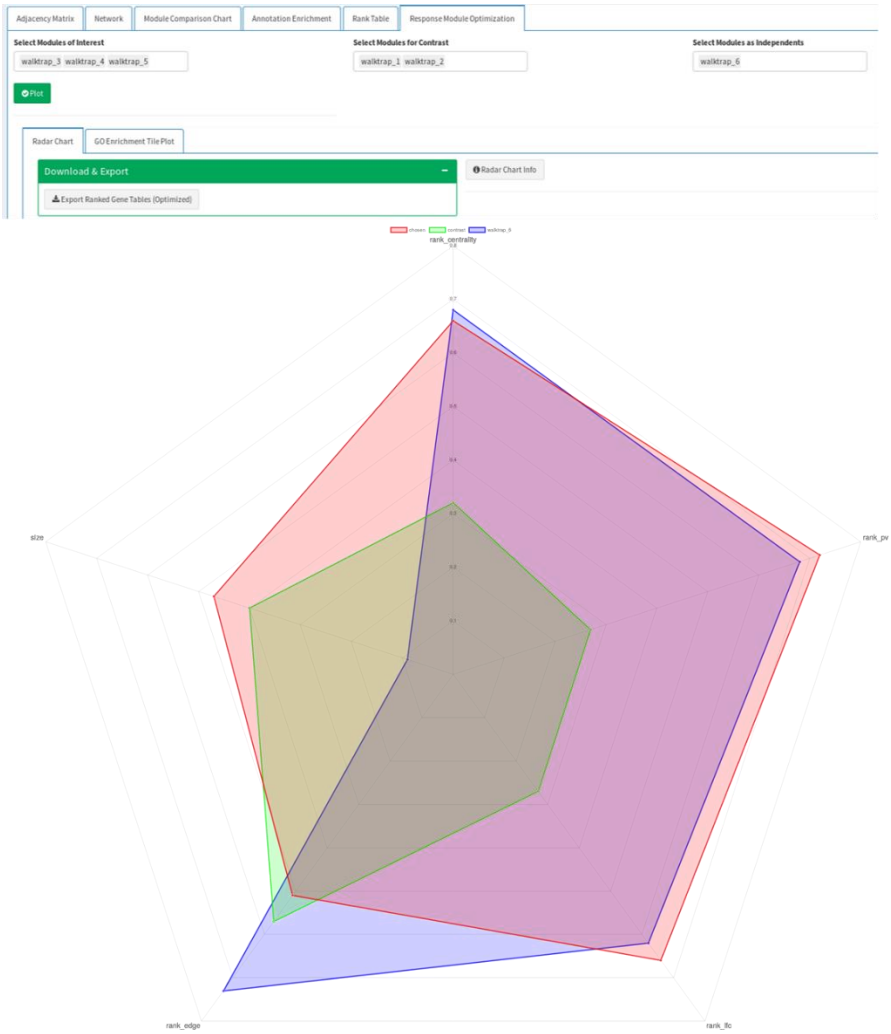
Representation of similarity between sets of GO terms overrepresented in modules as a heatmap.



Within the Annotation Enrichment tab, the user can also view a heatmap representing the similarity between the modules by GO terms representing all pairs of modules. The color gradient ranges from red to yellow where read means similar and yellow means dissimilar. On the left and top of the heatmap is a dendrogram displaying the clustering view of the similarity matrix. The user can export the similarity matrix as a text file.

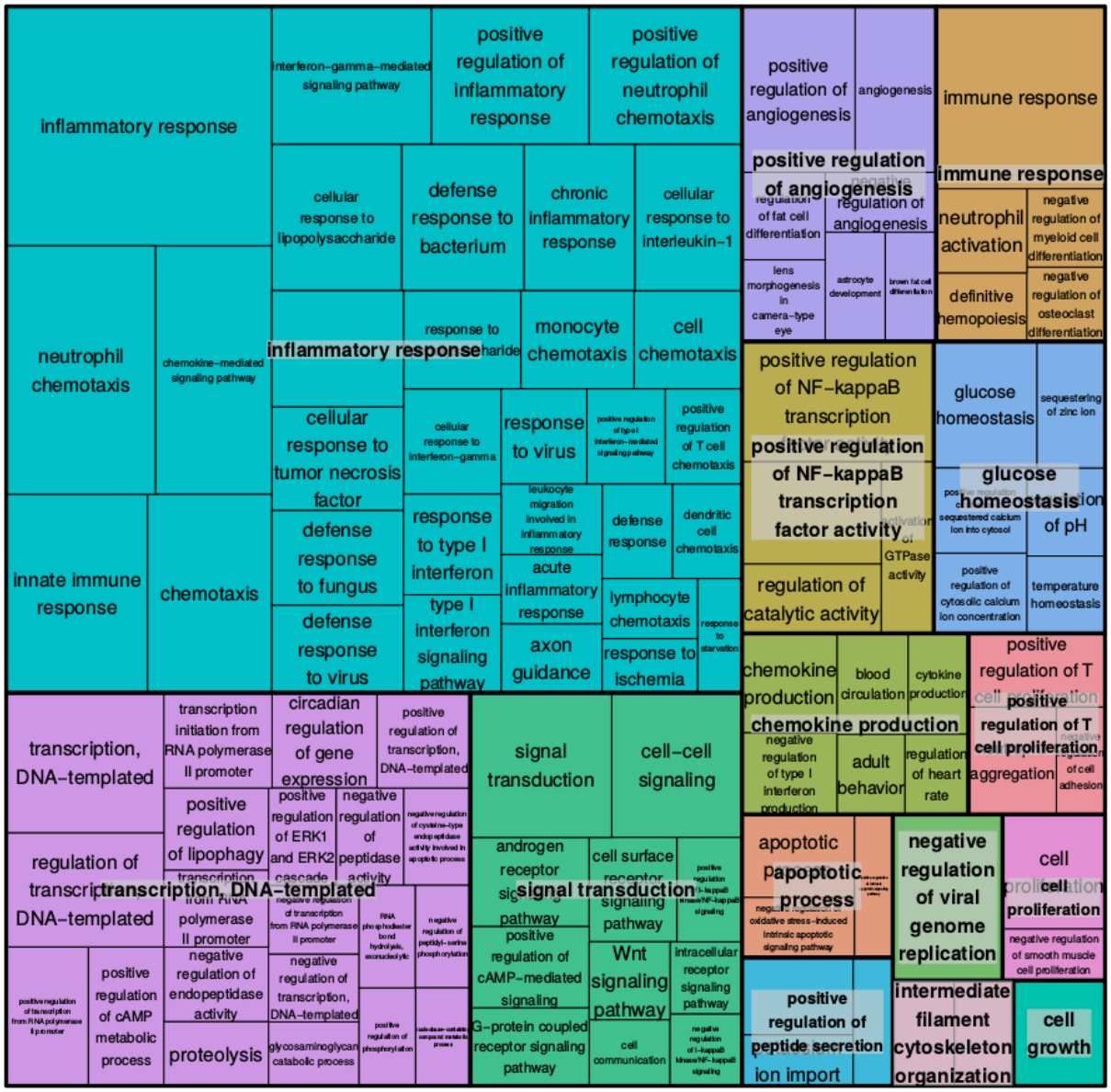
Create response module.

Select and merge modules based on GO similarity to create the final response module.



INfORM allows the user to merge multiple modules based on their GO similarity and create a response module. This can be performed from the Response Module Optimization tab where user can select module as Modules of Interest, the gene sets and GO representation of these modules will be merged this new set will be reassigned new scores which will be used to plot a radar chart where radii represent the metrics node ranks (centrality, differential P.Value, differential Log2(FC)), edge ranks, and module size, while the areas within the radii represent the merged set. To refrain from getting compartmentalized we advise the user to choose the remaining modules as contrast set or to view them individually, the addition of contrast set allows to understand the impact of module optimization strategy while considering all modules identified from the gene network. The user can export the ranked gene table for the whole network and the restructured modules as a spreadsheet.

Representation of the clustered and summarized GO terms from the final response module as a tile plot.



GO terms representing the optimized response module are further summarized by using the semantic similarity and are clustered to reduce redundancy and highlight the most significant GO terms. The clustered GO terms are displayed as a tile plot where each GO term is represented as a rectangular tile, and the size of the tile is a measure of the overrepresentation, clusters are marked with thicker borders and have differential colors, the most significant GO term from each cluster is used to label the cluster. This tile plot can be exported as a PDF file.