

netGO is an R/Shiny package for network-integrated pathway enrichment analysis. netGO provides user-interactive visualization of enrichment analysis results and related networks.

Currently, netGO supports analysis for four species ([Human](#), [Mouse](#), [Arabidopsis thaliana](#), and [Yeast](#))

These data are available from [netGO-Data](#) repository.

## Prerequisites

---

The R packages listed below are required to be installed before running netGO.(Alphabetical order)

*devtools, doParallel, doSNOW, DT, foreach, googleVis, htmlwidgets, shiny, shinyCyJS, shinyjs, V8*

- Most of the packages are available from [CRAN](#), but [shinyCyJS](#) should be installed from github.
- Linux user has to install V8 after installing the other packages.
- Note that netGO is not supported for CentOS 8, because V8 is not available in CentOS 8.

On Debian / Ubuntu : libv8-dev or libnode-dev.

On Fedora : v8-devel

[more information](#)

The user may want to use the following codes to install the required packages.

```
install.packages('devtools') # 2.2.1
library(devtools) # check Rcpp package is installed.
install_github('unistbig/shinyCyJS')
install.packages('doParallel') # 1.0.15
install.packages('doSNOW') # 1.0.18
install.packages('DT') # 0.11
install.packages('foreach') # 1.4.7
install.packages('googleVis') # 0.6.4
install.packages('htmlwidgets') # 1.5.1
install.packages('shiny') # 1.4.0
install.packages('shinyjs') # 1.0
install.packages('V8') # 2.3
```

## Running with an example data

---

Here are codes to run netGO for the breast tumor dataset (GEO [GSE3744](#).)

```
library(devtools)
install_github('unistbig/netGO') # install netGO library

library(netGO) # load netGO library
DownloadExampleData() # Download and load the breast tumor data
obj = netGO(genes = brca[1:30], genesets, network, genesetV)

# The user may also load the pre-calculated result using the following command
# load("brcaresult.RData")
```

For custom data analysis,

```
library(netGO)
userGenesetV = BuildGenesetV(genesets = userGenesets, network = userNetwork)
obj = netGO(genes = userGenes, genesets = userGenesets, network = userNetwork,
genesetV = userGenesetV)
```

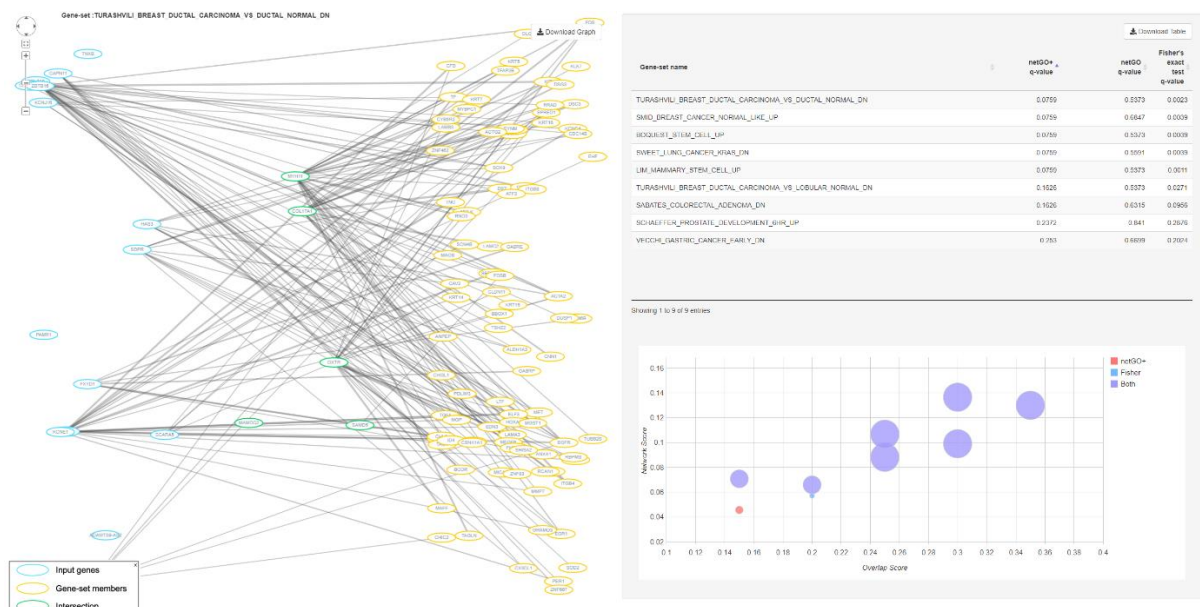
Running this example takes 5 to 25 minutes depending on the system used. The analysis results of netGO is shown below.

```
> head(obj)
```

	gene-set	netGOQ	netGO+Q	FisherQ	OverlapScore	NetworkScore
1139	TURASHVILI_BREAST_DUCTAL_CARCINOMA_VS_DUCTAL_NORMAL_DN	0.5373404	0.07589241	0.002346190	0.25	0.08812947
2882	SMID_BREAST_CANCER_NORMAL_LIKE_UP	0.6846815	0.07589241	0.003880407	0.30	0.09902842
2932	BOQUEST_STEM_CELL_UP	0.5373404	0.07589241	0.003880407	0.25	0.10670026
3104	SWEET_LUNG_CANCER_KRAS_DN	0.5590548	0.07589241	0.003850778	0.30	0.13654055
3723	LIM_MAMMARY_STEM_CELL_UP	0.5373404	0.07589241	0.001110137	0.35	0.12997229
1141	TURASHVILI_BREAST_DUCTAL_CARCINOMA_VS_LOBULAR_NORMAL_DN	0.5373404	0.16262659	0.027094031	0.15	0.07073293

The analysis result can be visualized using the following codes:

```
netGOVis(obj, genes = brca[1:30], genesets, network, R = 50, Q = 0.25 ) #
visualize netGO's result
```



If user wants to access result without shinyweb-application, the following functions can be used to export the result as text files

```
# exportGraphTxt
table = exportGraphTxt(gene = brca[1:30], geneset =
genesets[['SMID_BREAST_CANCER_NORMAL_LIKE_UP']], network) # table
head(table)

# exportGraph
graph = exportGraph(brca[1:30], geneset =
genesets[['SMID_BREAST_CANCER_NORMAL_LIKE_UP']], network) # shinyCyJS graph object
shinyCyJS(graph)

# exportTable
table = exportTable(obj, R = 50, Q = 0.25) # table
head(table)

dtable = exportTable(obj, type='D', R = 50, Q = 0.25) # data.table
dtable
```

## Data

### Example Datasets ([netGO-Data repository](#))

#### Human

Data	genes	genesets	network	genesetV
Breast Tumor	brca.RData	c2gs.RData	networkString.RData networkHumannet.RData	genesetVString1,2.RData genesetVHumannet1,2.RData
P53	p53.RData	c2gs.RData	networkString.RData networkHumannet.RData	genesetVString1,2.RData genesetVHumannet1,2.RData
Diabetes	dg.RData	cpGenesets.RData	networkString.RData networkHumannet.RData	cpgenesetV1,2.RData

The user can download the breast tumor data using *DownloadExampleData* function(Recommended)

#### Arabidopsis thaliana

Data	genes	genesets	network	genesetV
ShadowResponse	Aragenes.RData	KEGGara.RData	networkAranet.RData	AragenesetV.RData

#### Mouse & Yeast ( gene-set and networks available )

Species	genesets	network
Mouse	KEGGmouse.Rdata	networkMousenet.Rdata
Yeast	KEGGyeast.Rdata	networkYeastnet.Rdata

## Data Formats

netGO requires the following four data types.

- *genes* : a character vector of input genes (e.g., differentially expressed genes).
- *genesets* : a named list of gene-sets consisting of groups of genes to be tested.
- *network* : a numeric matrix of network data. The network scores are normalized to the unit interval [0,1] by dividing each score by the maximum score
- *genesetV* : A numeric matrix of pre-calculated interaction data between gene and gene-sets.  
The dimension of matrix must be [{number of genes} , {number of gene-sets}]. It can be built by using *BuildGenesetV* function with network and genesets objects as the input arguments.

```
genesetV = BuildGenesetV(network, genesets)
```

## ○ Functions

---

### 1. netGO

netGO function tests the significance of the gene-sets for the input gene list and returns a data frame of gene-sets, their *p*-values, *q*-values derived from netGO+, Fisher's exact test and netGO (optional) as well as the scores for the network interaction and overlap.

#### Input arguments

- genes: a character vector of input genes (e.g., differentially expressed genes).

- **genesets**: a list of gene-sets consisting of groups of genes.
- **network**: A numeric matrix of network data. The network scores are normalized to the unit interval [0,1]. 1 represents strong interaction and 0 for no interaction

	A	B	C
A	0	0.1	0.76
B	0.1	0	0.324
C	0.76	0.324	0

- **genesetV**: a numeric matrix of pre-calculated interaction data between genes and gene-sets.  
This object can be built with *BuildGenesetV* function.

	Gene-set1	Gene-set2	Gene-set3
A	0.837	1.647	0.074
B	0	1.75	0.113
C	0.464	0.486	2.442

- **alpha** (optional): a numeric parameter ( $\geq 1$ ; the default is 20) that weights the contribution of network connections in enrichment analysis.
- **beta** (optional): a numeric parameter ( $\in [0,1]$ ; the default is 0.5) that balances the weights between the relative and absolute network scores.

$$P(T \rightarrow A) = \underbrace{\frac{|T \cap A|}{|T|}}_{\text{Overlap score}} + \underbrace{\frac{\alpha}{|T|} \sum_{x \in T-A} AI(x, A)^\beta \cdot RI(x, A)^{1-\beta}}_{\text{Network score}}$$

$$AI(x, A) = \frac{1}{|A|} \sum_{y \in A} I(x, y)$$

$$RI(x, A) = \frac{1}{|N(x)|} \sum_{y \in A} I(x, y)$$

- nperm (optional): a numeric parameter to determine the bin size (number of genes) to be used during resampling. The default is NULL which assigns approximately 2000 genes to each bin
- pvalue (optional): a boolean parameter to determine whether to return Q-values only ( FALSE ) or both P-values and Q-values (TRUE)
- plus (optional): a boolean parameter to determine whether to run both netGO and netGO+ (plus = FALSE) or netGO+ only ( plus = TRUE, default )
- verbose (optional) : a boolean parameter whether to show more process of netGO as follows.

```
> obj = netGO(brca[1:30], genesets, network, genesetV)
Fisher Pvalue Calculation finished
2000 genes in each category
Indexing genes
Build category
Parallel functions load
netGO skipped
netGO+ Calculation start

Progress - each = means 5%
=====
netGO+ Calculation finished
```

**Notice** the input genes should be represented in **gene symbols** when using the default networks and gene-sets (STRING and MSigDB). Other types of gene names are also allowed if the corresponding customized data (networks and gene-set data) are used.

---

## 2. netGOVis

netGOVis function visualizes the analysis results on the web browser (google chrome is recommended).

The resulting graphs (svg format) and table are downloadable from the web browser.

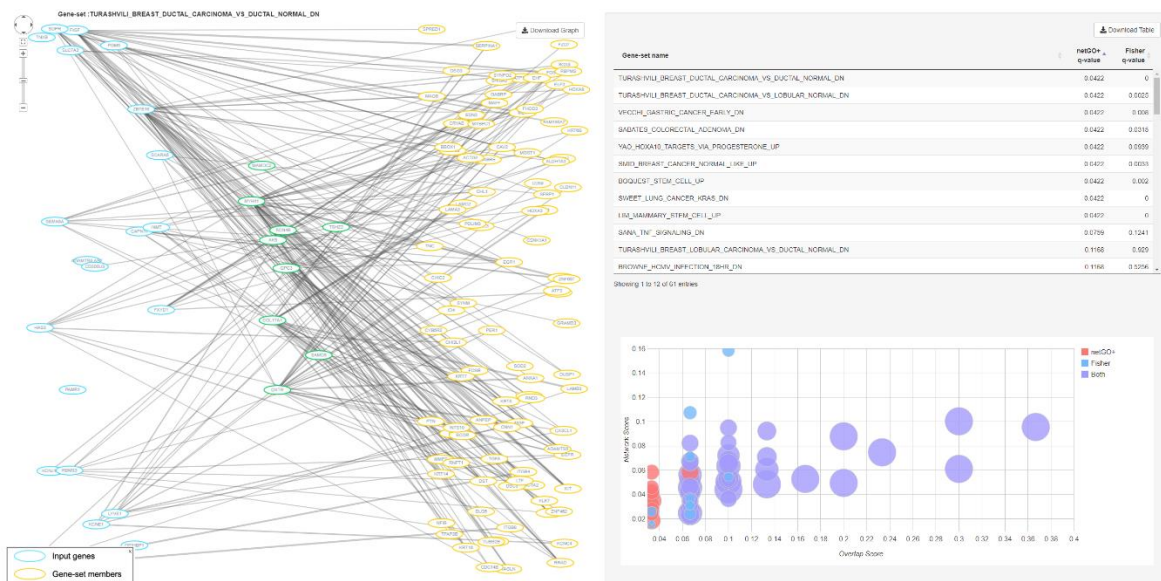
### Input arguments

- obj: the data frame of analysis results obtained by running **netGO** function. It consists of multiple columns including
  1. gene-set name and p, q-values evaluated using netGO (optional), netGO+, and Fisher's exact test as well as the scores for the overlap and networks.
- genes, genesets, network: the same as those in the *netGO* function.
- R (optional): gene-set rank threshold, The default is 50 (Top 50 gene-sets in either method will be shown).
- Q (optional): Gene-set Q-value threshold, The default is 0.25. (gene-sets with Q-value  $\leq 0.25$  will be used)

After running the netGO function, the user may see the following logs in the R console.

```
> netGOVis(obj, brca[1:30], genesets, network)
Listening on http://127.0.0.1:6042
```

and user's default web browser (**netGO was built based on chrome environment**) will return the following interactive visualization:



### 3. BuildGenesetV

BuildGenesetV function will build genesetV object using the given *network* and *genesets*.

genesetV is pre-calculated interaction files used to reduce the running time of netGO.

#### Input arguments

- genesets, network: the same as those in the *netGO* function.

### 4. DownloadExampleData

This function will download example data in the user's working directory and load the data ( breast tumor, [GSE3744](#) ) in user's R environment.

Note that, if objects exist in the working directory, this function will not download the data again, so we recommend removing and downloading them again if netGO package is updated.

#### Input arguments

- none
- R object named *brca*, *genesets*, *genesetV*, *network*, *obj* will be loaded.



---

## 5. exportGraph

exportGraph function will export network data from the netGO analysis result as graph object that can be accessed using shinyCyJS function

### Input arguments

- genes, network : the same as those in the *netGO* function.
- geneset : a character vector of gene symbols (e.g., member of genesets object in *netGO*).

for example,

```
geneset = genesets[['SMID_BREAST_CANCER_NORMAL_LIKE_UP']]
graph = exportGraph(brca[1:30], geneset =
genesets[['SMID_BREAST_CANCER_NORMAL_LIKE_UP']], network) # shinyCyJS graph object
shinyCyJS(graph)
```

---

## 6. exportGraphTxt

exportGraphTxt function will export network data from the netGO analysis result as table format.

### Input arguments

- genes, network, geneset : the same as those in the *exportGraph* function.

For example,

```
table = exportGraphTxt(brca[1:30], geneset, network)
head(table)
```

the exported data are shown as

geneA	geneB	strength	type
A	B	0.1	Inter

geneA	geneB	strength	type
C	D	0.82	Inner

'Inter' means geneB belongs to the intersection of *genes* and *genesets*. 'Inner' means geneB belongs to the differenced set *genesets* – *genes*.

## 7. exportTable

exportTable will export the result object of netGO as table or data.table.

### Input arguments

- obj, R, Q : the same as those in the *netGOVis* function.

for example,

```
table = exportTable(obj, R = 50, Q = 0.25) # table
head(table)

dtable = exportTable(obj, type='D', R = 50, Q = 0.25) # data.table
dtable
```

The exported data have the format as follows:

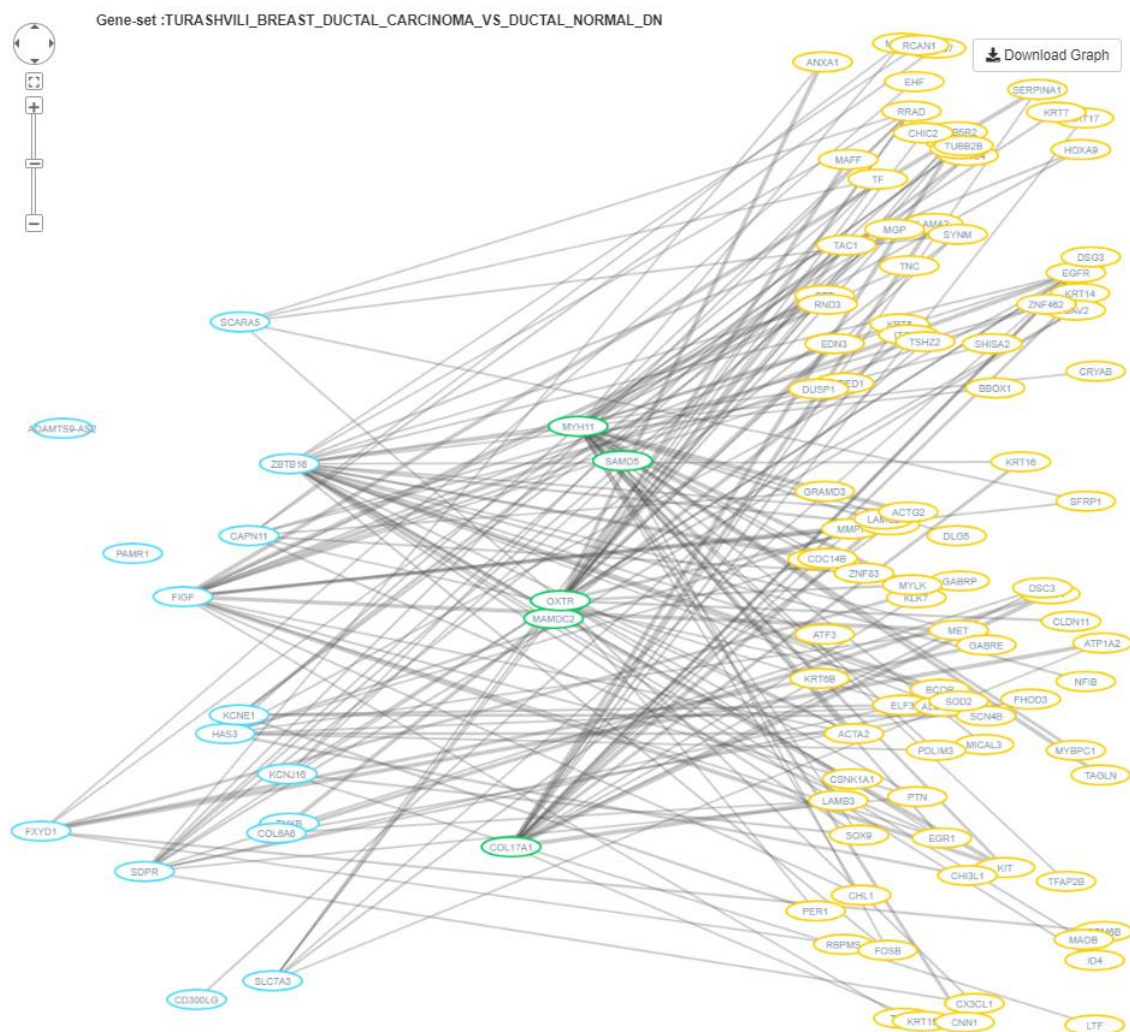
geneset name	netGO+ q-value	Fisher q-value
genesetA	0.11	0.2

## Visualization and exploration of netGO analysis results

The netGO analysis results are visualized through three panels: interaction networks, list of significant gene-sets, and the bubble chart.

### Interaction Network

- The network panel displays the input genes, selected gene-set, and the network connections between the two.
- ■ Sky blue nodes represent input genes (e.g., differentially expressed genes)
- ■ Yellow nodes represent genes in the selected gene-set
- ■ Green nodes represent the intersection of input genes and the gene-set.
- The edge width represents the strength of interaction between two nodes.
- Genes without edges will not be displayed.
- The gene-set can be selected by clicking on the gene-set name on the upper-right panel.
- The user can download the graph image as SVG format.



## Significant gene-sets

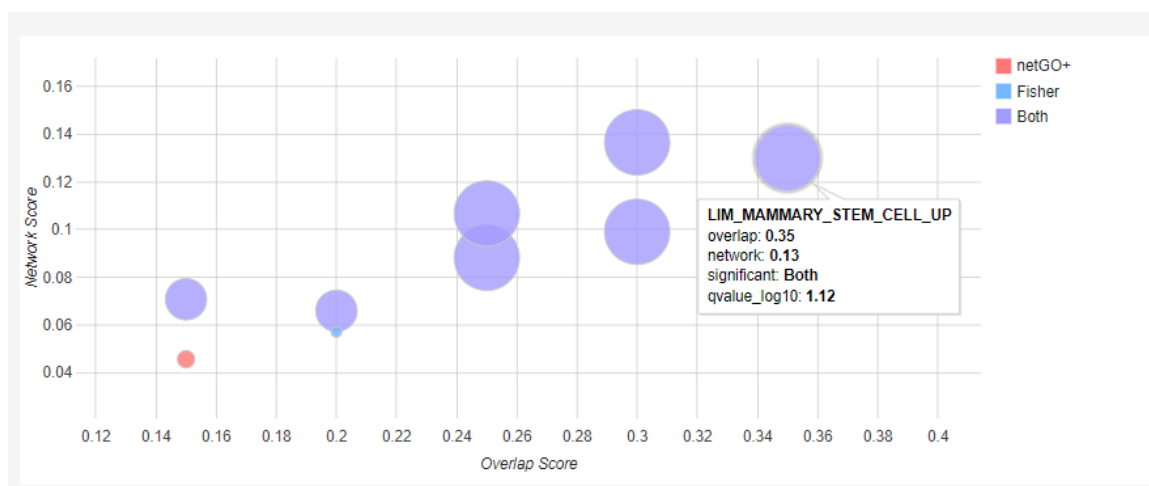
- This panel contains the list of significant gene-sets as well as their Q-values ( or P-values ) evaluated from netGO, netGO+ and Fisher's exact test. It is

downloadable by clicking the 'Download Table' button in the upper right corner of the table

Download Table			
Gene-set name	netGO+ q-value	netGO q-value	Fisher's exact test q-value
TURASHVILI_BREAST_DUCTAL_CARCINOMA_VS_DUCTAL_NORMAL_DN	0.0759	0.5373	0.0023
SMID_BREAST_CANCER_NORMAL_LIKE_UP	0.0759	0.6847	0.0039
BOQUEST_STEM_CELL_UP	0.0759	0.5373	0.0039
SWEET_LUNG_CANCER_KRAS_DN	0.0759	0.5591	0.0039
LIM_MAMMARY_STEM_CELL_UP	0.0759	0.5373	0.0011
TURASHVILI_BREAST_DUCTAL_CARCINOMA_VS_LOBULAR_NORMAL_DN	0.1626	0.5373	0.0271
SABATES_COLORECTAL_ADENOMA_DN	0.1626	0.6315	0.0956
SCHAEFFER_PROSTATE_DEVELOPMENT_6HR_UP	0.2372	0.841	0.2876
VECCHI_GASTRIC_CANCER_EARLY_DN	0.253	0.6699	0.2024


## Bubble chart

- This module plots the bubble chart of significant gene-sets for the netGO+ results.
- The overlap (x-axis) and network (y-axis) scores of the significant gene-sets are represented.
- The size of bubbles represents the significance level of each gene-set in -log10 scale (Qvalue).
- Hovering/Click on each bubble will show corresponding statistical values.



## 😊 Contact

- Comments / suggestions and questions will be greatly appreciated,

-  Jinhwan Kim [@jhk0530](https://github.com/jhk0530) [kjh0530@unist.ac.kr](mailto:kjh0530@unist.ac.kr)
- prof. Dougu Nam [dougnam@unist.ac.kr](mailto:dougnam@unist.ac.kr)

## License

---

This project is [MIT](#) licensed