Network Module Detection from Multi-Modal Node Features with a Greedy Decision Forest

Bastian Pfeifer

8/23/2021

Input

To perform Network Module Detection a network and node feature matrices are required. Here, we use a PPI-Network and multi-omics node features from gene expression and methylation data. In addition to that, a binary vector needs to be specified reflecting the outcome class. Here, it reflects the patient group survived (0) vs non-survived (1).

```
PPI      <- read.table("~/LinkedOmics/KIRC/KIDNEY_PPI.txt")
mRNA     <- read.table("~/LinkedOmics/KIRC/KIDNEY_mRNA_FEATURES.txt")
Methy    <- read.table("~/LinkedOmics/KIRC/KIDNEY_Methy_FEATURES.txt")
TARGET   <- read.table("~/LinkedOmics/KIRC/KIDNEY_SURVIVAL.txt")
```

Dimensions are:

```
dim(PPI)
```

```
## [1] 6926452       3
```

```
dim(mRNA)
```

```
## [1]   306 12029
```

```
dim(Methy)
```

```
## [1]   306 12029
```

```
dim(TARGET)
```

```
## [1]   1 306
```

The PPI Network has 6926452 edges and is organized as follows:

```
head(PPI,5)
```

```
##   protein1 protein2 combined_score
## 1     ARF5    CALM2            490
## 3     ARF5     ERN1            159
## 4     ARF5   CDKN2A            606
## 5     ARF5     P4HB            167
## 6     ARF5    STX10            267
```

The first two columns refer to the connected nodes. The last column indicate the confidence of the interaction between these nodes/protein.

The node feature matrices are organized as follows:

```
mRNA[1:5,1:5]
```

```
##                 RBL2   VDAC3   ACTN1 ATP2A1  SFRP1
## TCGA.3Z.A93Z 10.1967 10.8407 11.0698 3.0921 8.4911
## TCGA.6D.AA2E 10.4898 11.2592 11.4613 3.4214 5.9663
## TCGA.A3.3357 10.8225 11.4032 11.5370 3.0013 4.6062
## TCGA.A3.3358 11.6874 10.9420 12.8086 5.4678 5.1437
## TCGA.A3.3367 11.3013 11.0082 11.8861 4.9567 6.2678
```

where the rows are the patients and the columns are the genes/nodes. In this case, we analyze 12029 node feature values from 306 patients.

The second omic data set has the same structure.

```
Methy[1:5,1:5]
```

```
##                 RBL2   VDAC3   ACTN1  ATP2A1   SFRP1
## TCGA.3Z.A93Z -0.4897 -0.4686 -0.0063 -0.4851 -0.4156
## TCGA.6D.AA2E -0.4885 -0.4574  0.1481 -0.4799 -0.3343
## TCGA.A3.3357 -0.4854 -0.4721 -0.0540 -0.4859 -0.3052
## TCGA.A3.3358 -0.4838 -0.4339 -0.1005 -0.4778 -0.3213
## TCGA.A3.3367 -0.4890 -0.4684 -0.0493 -0.4830 -0.2719
```

Note, the rows of the multi-modal feature matrices should refer to the exact same patient.

Finally, we need the target vector specifying the survival (0) and non-survival (1) groups.

```
TARGET[1:5]
```

```
##    TCGA.3Z.A93Z TCGA.6D.AA2E TCGA.A3.3357 TCGA.A3.3358 TCGA.A3.3367
## 13            0            0            0            0            0
```

Creating a DFNET graph object

The DFNET package requires the following packages

```
library(DFNET)
require(igraph)
require(ranger)
require(pROC)
```

Lets create a DFNET_graph object and keep the edges with confidence values greater the 0.95-quantile

```
DFNET_graph  <- DFNET_generate_graph_Omics(PPI, list(mRNA, Methy), TARGET, 0.95)
```

```
summary(DFNET_graph)
```

```
##                Length Class  Mode
## graph            10   igraph list
## Feature_Matrix    2   -none- list
## gene.names     9033   -none- character
```

The DFNET_graph object is a list and consists of three slots. The first slot is the PPI network internally converted to a igraph object.

```
DFNET_graph$graph
```

```
## IGRAPH ddf83b3 U--- 9032 322206 --
## + edges from ddf83b3:
##  [1] 365--3411 365--1618 365--5723 365--8823 365--2469 365--4046 365--3793
##  [8] 365--6680 365--8829 365--6551 365--7310 365--3929 365--2550 365--5588
## [15] 365--1359 365--7638 120-- 365 140-- 365 365--1158 365--2171 365-- 589
## [22] 365--7900 365--6420 365--1297 365--8699 365--5182 365--4736 365--5805
## [29] 365--7215 365--1538 365--8643 365--6355 365--4688 365--6619 365--1073
## [36] 365--2961 365--2038 239-- 365 365--4766 278-- 365 365--4778 365--1284
## [43] 365--3950 365--1262 365--2285 102-- 365 365--6932 365--5808 365-- 449
## [50] 365--6536 365--1022 365--3422 365--8567 365--9011 365--6928 365--2717
## [57] 365-- 757 365--5728 365--2213 170-- 365 365--2813 365--2968 365--2889
## + ... omitted several edges
```

The second slot is a list of feature matrices. In our case gene expression and methylation data from the same set of patients.

```
DFNET_graph$Feature_Matrix[[1]][1:5,1:5]
```

```
##                 AN_1    AN_2    AN_3   AN_4   AN_5
## TCGA.3Z.A93Z 10.1967 10.8407 11.0698 3.0921 8.4911
## TCGA.6D.AA2E 10.4898 11.2592 11.4613 3.4214 5.9663
## TCGA.A3.3357 10.8225 11.4032 11.5370 3.0013 4.6062
## TCGA.A3.3358 11.6874 10.9420 12.8086 5.4678 5.1437
## TCGA.A3.3367 11.3013 11.0082 11.8861 4.9567 6.2678
```

```
DFNET_graph$Feature_Matrix[[2]][1:5,1:5]
```

```
##                 BN_1    BN_2    BN_3    BN_4    BN_5
## TCGA.3Z.A93Z -0.4897 -0.4686 -0.0063 -0.4851 -0.4156
## TCGA.6D.AA2E -0.4885 -0.4574  0.1481 -0.4799 -0.3343
## TCGA.A3.3357 -0.4854 -0.4721 -0.0540 -0.4859 -0.3052
## TCGA.A3.3358 -0.4838 -0.4339 -0.1005 -0.4778 -0.3213
## TCGA.A3.3367 -0.4890 -0.4684 -0.0493 -0.4830 -0.2719
```

The third slot contains the node/gene names.

```
head(DFNET_graph$gene.names,5)
```

```
## [1] "RBL2"   "VDAC3"  "ACTN1"  "ATP2A1" "SFRP1"
```

A DFNET_graph object can thus be easily created, also without using the “DFNET_generate_graph_Omics” function. Note, the colnames of the feature matrices need to be as specified! A prefix letter followed with a “N” and than simply the node identifier. Node identifier should match the identifier used for the igraph network.

```
head(as_edgelist(DFNET_graph$graph, names = TRUE))
```

```
##      [,1] [,2]
## [1,]  365 3411
## [2,]  365 1618
## [3,]  365 5723
## [4,]  365 8823
## [5,]  365 2469
## [6,]  365 4046
```

As seen from the above table “AN_365” and “AN_3411” are connected. The same is true for the second modality “BN_365” and “BN_3411”.

DFNET for Subnetwork Detection

The main function for network module detection expects the number of trees (ntrees), the number of greedy iteration (niter), and the initial size of the module as an input. The ntrees parameter specifies the number of random works initialized. The niter parameter sets the total number of greedy iterations, and the init.mtry defines the depth of the random work, and thus the initial size of the modules.

```
DFNET_object <- DFNET(DFNET_graph, ntrees=100, niter=100, init.mtry=10)
```

```
summary(DFNET_object)
```

```
##                    Length Class  Mode
## DFNET_graph          3    -none- list
## DFNET_trees        600    -none- list
## DFNET_MODULES      100    -none- list
## DFNET_MODULES_AUC 100    -none- numeric
```

The DFNET_object contains four slots. The first slot "DFNET_graph" is the igraph object storing the network topology. The second slot "DFNET_trees" contains the Decision Trees of the Decision Forest. The third slot "DFNET_MODULES" stores the detected Network Modules, and the "DFNET_MODULES_AUC" consists of the corresponding AUC values of the modules. The accuracy of the Decision Forest Classifier can be calculated as

```
DFNET_acc    <- DFNET_accuracy(DFNET_graph, DFNET_object)
```

```
DFNET_acc
```

```
## [1] 0.7667751
```

DFNET Edge Importance Scores

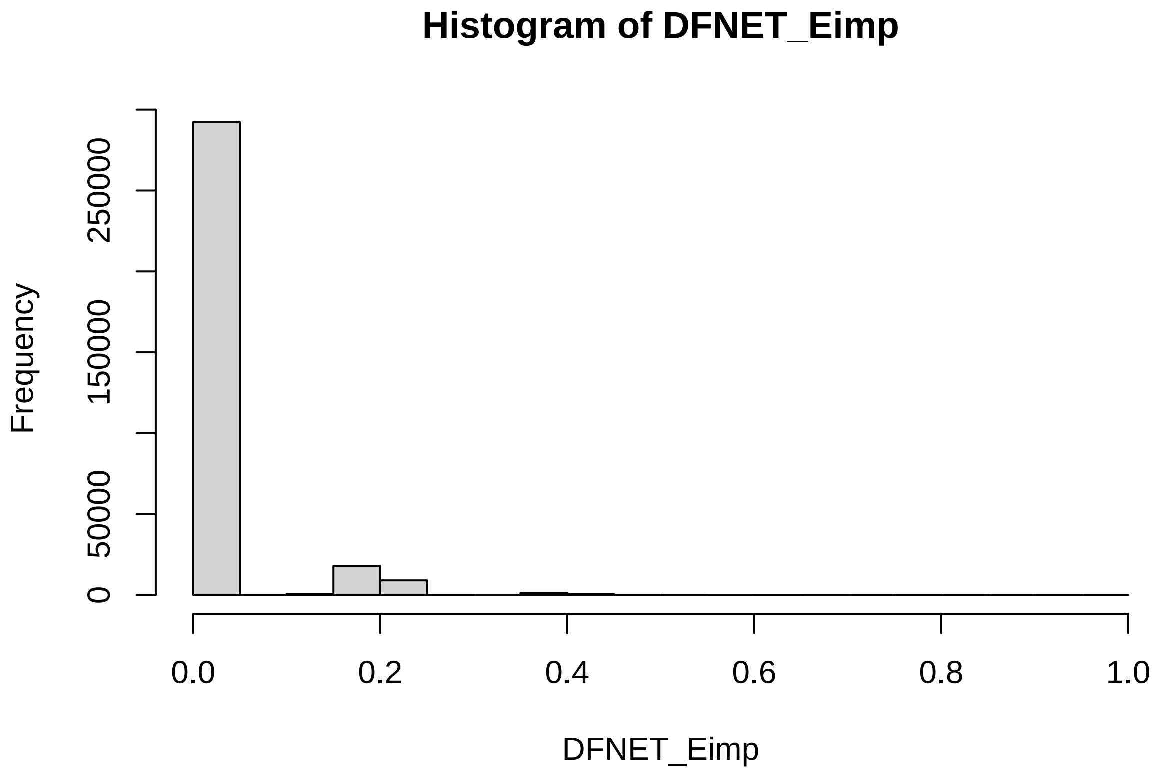
Edge Importance Scores can be calculated with the following function

```
DFNET_Eimp    <- DFNET_Edge_Importance(DFNET_graph, DFNET_object)
```

```
length(DFNET_Eimp)
```

```
## [1] 322206
```

```
hist(DFNET_Eimp)
```



DFNET Detected Modules

The detected modules can be retrieved via the "DFNET_modules" function

```
DFNET_mod    <- DFNET_modules(DFNET_graph, DFNET_object, DFNET_Eimp)
```

```
head(DFNET_mod)
```

```
##                                                           Module EDGE_IMP
## 2         1532 4653 4917 5294 5495 6128 6297 6635 6784 8430 9013 2.182502
## 1 307 436 4276 4653 5301 5470 5583 6019 6379 6384 7723 8323 8883 2.200262
## 3                         504 2813 4009 4880 5441 5972 7096 8890 1.964621
## 4           11 3374 3771 4099 4292 4601 5952 7011 7371 7582 8592 1.684959
## 5                        1869 4295 4379 5770 6038 6979 7453 8680 1.636016
## 6         1068 1305 1749 4185 4420 4668 4973 5240 6133 7660 8807 1.591245
##         AUC      IMP
## 2 0.7016790 2.884181
## 1 0.6684211 2.868683
## 3 0.6827068 2.647328
## 4 0.6961575 2.381117
## 5 0.7231853 2.359201
## 6 0.7532468 2.344492
```

The modules are ranked by their importance (last column). Note, node ids are shown, but the actual node names can be retrieved from "DFNET_graph$gene.names".

Lets have a closer look at the top ranked module

```r
Nodes        <- as.numeric(strsplit(DFNET_mod[1,1]," ")[[1]])
DFNET_graph$gene.names[Nodes]
```

```
##  [1] "SMNDC1"  "CTNNBL1" "CPSF4"   "POLR2L"  "LSM6"    "HNRNPH1" "EIF4A3"
##  [8] "MAGOHB"  "RBM5"    "HNRNPK"  "CSTF2T"
```

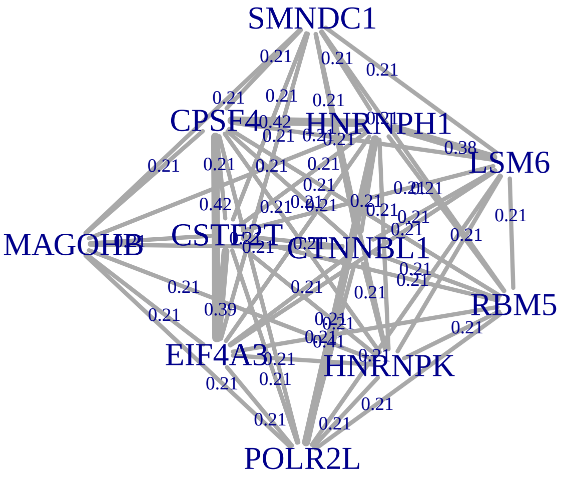
The module is reflected by the following edges and nodes

```r
Top_Module <- DFNET_get_module(Nodes, DFNET_graph, DFNET_Eimp)
head(Top_Module)
```

```
##      GENE    GENE          EDGE_IMP
## 1  SMNDC1 CTNNBL1 0.212290409880912
## 2  SMNDC1   CPSF4 0.212290409880912
## 3 CTNNBL1   CPSF4 0.212290409880912
## 4 CTNNBL1  POLR2L 0.212290409880912
## 5   CPSF4  POLR2L 0.212290409880912
## 6  SMNDC1    LSM6 0.212290409880912
```

We can visualize this subgraph using the function “DFNET_plot_module”.

```r
require(igraph)
DFNET_plot_module(Nodes, DFNET_graph, DFNET_Eimp)
```



DFNET Node Feature importance scores

The feature importances of the nodes of that module can be calculated as

```
DFNET_Fimp   <- DFNET_calc_feature_importance(Nodes, DFNET_object, DFNET_graph)
DFNET_Fimp
```

```
##           SMNDC1    CTNNBL1      CPSF4     POLR2L       LSM6    HNRNPH1
## omic1 0.01243056 0.01299474 0.11582528 0.02890176 0.01923387 0.08894694
## omic2 0.00000000 0.00000000 0.07120735 0.03436464 0.07839865 0.02436282
##           EIF4A3     MAGOHB       RBM5     HNRNPK     CSTF2T
## omic1 0.06299942 0.04710311 0.01071508 0.01244777 0.01919737
## omic2 0.06120851 0.04662184 0.00757414 0.00000000 0.03246349
```

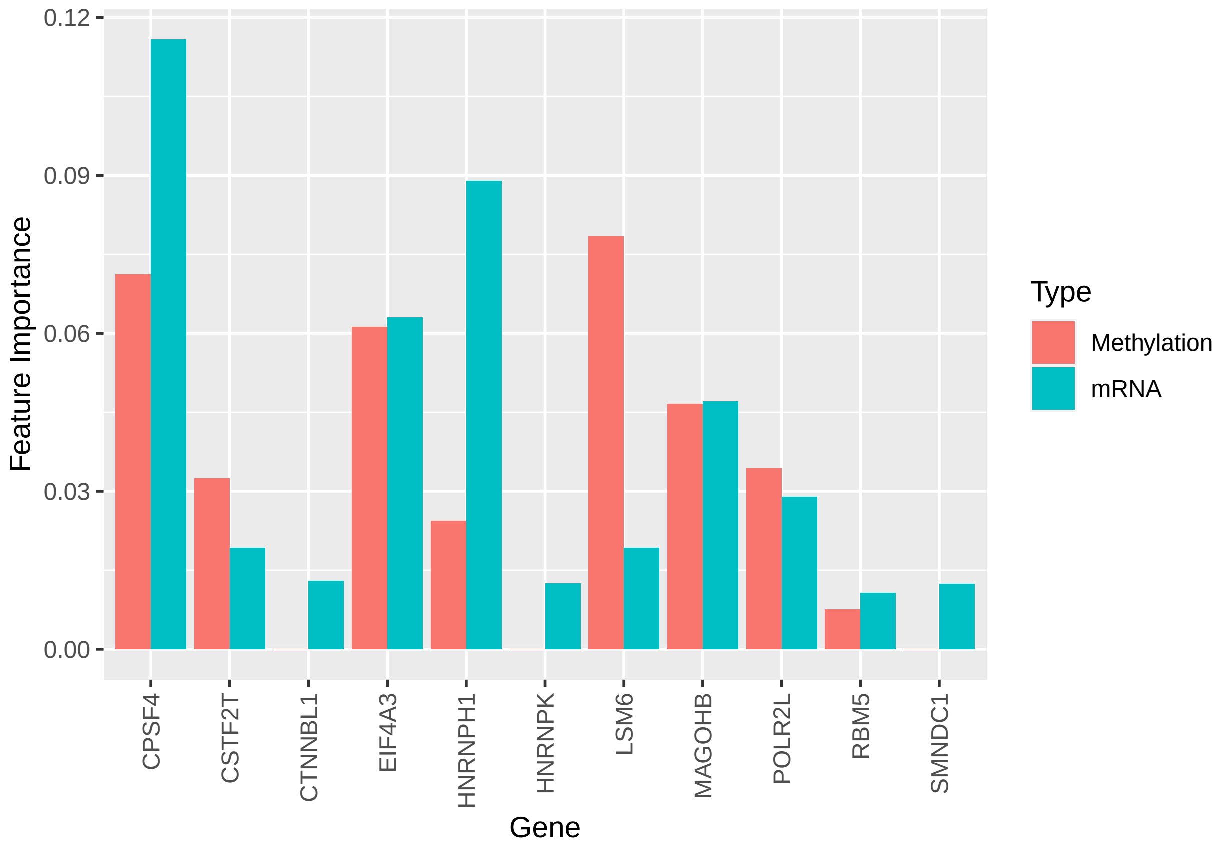
```
## GGPLOT
library(ggplot2)
library(reshape)

RES1 <- cbind(colnames(DFNET_Fimp),DFNET_Fimp[1,])
RES2 <- cbind(colnames(DFNET_Fimp),DFNET_Fimp[2,])
RES1 <- cbind(RES1,"mRNA")
RES2 <- cbind(RES2,"Methylation")

RES <- rbind(RES1,RES2)
rownames(RES) <- NULL
colnames(RES) <- c("Gene","IMP","Type")
RES     <- as.data.frame(RES)
RES$IMP <- as.numeric(RES$IMP)

p <- ggplot(RES, aes(fill=Type, y=IMP, x=Gene)) +
    geom_bar(position="dodge", stat="identity") +
    ylab("Feature Importance") +
    theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))

plot(p)
```


0.12
0.09
0.06
0.03
0.00
Feature Importance
CPSF4
CSTF2T
CTNNBL1
EIF4A3
HNRNPH1
HNRNPK
LSM6
MAGOHB
POLR2L
RBM5
SMNDC1
Gene
Type
Methylation
mRNA