DriverNet package

Ali Bashashati, Reza Haffari, Jiarui Ding, Gavin Ha, Kenneth Liu, Jamie Rosner and Sohrab Shah

April 2, 2012

1 Overview

DriverNet is a package to predict functional important driver genes in cancer by integrating genome data (mutation and copy number variation data) and transcriptome data (gene expression data). The different kinds of data are combined by an influence graph, which is a gene-gene interaction network deduced from pathway data. A greedy algorithm is used to find the possible driver genes, which may mutated in a larger number of patients and these mutations will push the gene expression values of the connected genes to some extreme values.

Specifically, DriverNet formulates associations between mutations and expression levels using bipartite graph where nodes are i) the set of mutated genes M and ii) the set of genes exhibiting outlying gene expression G. An edge $E(g_i, g_j)$ between nodes in M and G are drawn under three conditions: gene $g_i \in M$ is mutated in patient p of the population; gene g_j shows outlying expression in patient p; and g_i and g_j are known to interact according to an influence graph. Our approach then uses a greedy optimization approach to explain the most number of nodes in G with the fewest number of nodes in M. The genes explaining the highest number of outlying expression events are nominated as putative driver genes. Finally, we apply statistical significance tests to these candidates based on null distributions informed by stochastic resampling.

2 Datasets

The package includes the matrices deduced from part of the TCGA Glioblastoma multiforme (GBM) data. For both `samplePatientMutationMatrix` and `samplePatientOutlierMatrix`, they are binary matrices and the row names are patients and the column names are genes.

```
> library(DriverNet)
> data(samplePatientMutationMatrix)
> data(samplePatientOutlierMatrix)
> data(sampleInfluenceGraph)
> data(sampleGeneNames)
>
```

3 Rank genes

`computeDrivers` is the main function, which uses a greedy algorithm to rank each mutated gene based on how many outliers gene the mutated gene can cover.

```
> # The main function to compute drivers
> driversList = computeDrivers(samplePatientMutationMatrix,
+               samplePatientOutlierMatrix,sampleInfluenceGraph,
+               outputFolder=NULL, printToConsole=FALSE)
> drivers(driversList)[1:10]

 [1] "EGFR"    "TP53"    "MTAP"    "PTEN"    "PIK3R1"
 [6] "CDKN2A"  "CYP27B1" "PIK3CA"  "PDGFRA"  "IDH1"
```

4 Compute p-values

The statistical significance of the driver genes are assessed using a randomization framework. The original datasets are permuted N times, and the algorithm is run on randomly generated datasets N times and results on real data are assessed to see if they are significantly different from the results on randomized datasets. This in an indirect way of perturbing the bipartite graph corresponding to the original problem. To generate the random datasets, we keep the contents of patient-mutation, M, and patient-outlier, G', matrices the same but replace the gene symbols with a randomly selected set of genes from the Ensmbl 54 protein-coding gene list. Using the same influence graph, the algorithm is run on the new patient-mutation, $M_1 ... M_N$, and patient-outlier, $G'_1 ... G'_N$, matrices.

Suppose D is the result of driver mutation discovery algorithm. D contains a ranked list of driver genes with their corresponding node coverage in the bipartite graph, $\mathcal{B}$. The statistical significance of a gene $g \in D$ with a corresponding node coverage, COV_g, is the fraction of times that we observe driver genes in our random data runs i, with node coverage more than COV_g.

```
> # random permute the gene labels to compute p-values
> randomDriversResult = computeRandomizedResult(
+   patMutMatrix=samplePatientMutationMatrix,
+   patOutMatrix=samplePatientOutlierMatrix,
+   influenceGraph=sampleInfluenceGraph,
+   geneNameList= sampleGeneNames, outputFolder=NULL,
+   printToConsole=FALSE,numberOfRandomTests=20, weight=FALSE,
+   purturbGraph=FALSE, purturbData=TRUE)
```

5 Summarize the results

Finlly, we provide a function to summarize the results.

```
> # Summarize the results
> res = resultSummary(driversList, randomDriversResult,
+       samplePatientMutationMatrix,sampleInfluenceGraph,
+       outputFolder=NULL, printToConsole=FALSE)
> res[1:2,]

     rank gene   mut  p-value total_events covered_events
EGFR "1"  "EGFR" "50" "0"     "16695"      "791"
TP53 "2"  "TP53" "38" "0"     "16695"      "659"
     node_degree no.of.cases
EGFR "396"       "50"
TP53 "416"       "38"
     connected.drivers
EGFR "1,2,4,5,6,8,9,13,14,15,16,17,18,19,20,21,23,26,27,28,31,32,33,34,36,38,39,40,44,45,47,
TP53 "1,2,4,5,6,8,9,12,13,14,17,18,19,20,21,23,26,27,28,30,33,34,36,38,39,40,42,43,44,45,47,
     case
EGFR "TCGA-02-0083-01,TCGA-02-0064-01,TCGA-06-0209-01,TCGA-06-0169-01,TCGA-02-0043-01,TCGA-0
TP53 "TCGA-02-0083-01,TCGA-02-0037-01,TCGA-02-0025-01,TCGA-02-0114-01,TCGA-06-0184-01,TCGA-0
```