ARVIN : Identifying Risk Noncoding Variants Using Disease-relevant Gene Regulatory Networks

Long Gao, Yasin Uzun, Kai Tan

October 31, 2017

Contents

1 Introduction

Identifying causalnoncoding variants remains a daunting task. Because noncoding variants exert their effectsin the context of a gene regulatory network (GRN), we hypothesize that explicit use of disease-relevant GRNcan significantly improve the inference accuracy of noncoding risk variants. We describe Annotation of Regulatory Variants using Integrated Networks (ARVIN), a general computational framework for predictingcausal noncoding variants. For each disease, ARVIN first constructs a GRN using multi-dimensional omics data oncell/tissue-type relevant to the disease. ARVIN then uses a set of novel regulatory network-basedfeatures, combined with sequence-based features to make predictions.This user guide contains the information necessary to run ARVIN.

2 Network construction

2.1 Enhancer prediction

ARVIN uses enhancer regions for mapping SNPs to enhancers. The data file must be in the following tab separated format:
#chromosome enhancer_center enhancer_probability
chr6 138231000 0.86
chr1 160807000 0.79
chr1 160808600 0.88
You can use CSI-ANN software for predicting the enhancers. For this purpose, you can download CSI-ANN

from our lab web page http://tanlab4generegulation.org/CSIANNWebpage.html . CSI-ANN uses histone modification data as input to predict enhancer regions in genome-wide. You can use the output of CSI-ANN (which is as shown above) as input for ARVIN. ### 2.2 Enhancer-promoter interaction prediction ARVIN uses enhancer-promoter interaction for mapping SNPS to genes via enhancers. The enhancer-promoter interaction data must be in tab separated format as follows:
#Chr Start End Target Score
chr9 22124001 22126001 ENST00000452276 0.93
chr2 242792001 242794001 ENST00000485966 0.792
You can use IM-PET software for predicting enhancer-promoter interactions. For this purpose, you can download IM-PET from http://tanlab4generegulation.org/IM-PET.html . IM-PET uses enhancer predictions (computed using CSI-ANN) and gene expression data to compute enhancer-promoter interactions. You can use the output of IM-PET (which is as shown above) as input for ARVIN.

2.2 Obtain gene-gene interaction network

Gene interaction network can be obtained from multiple sources including protein-protein interaction networks and functional interaction networks.

2.3 Network scoring

To make different types of scores comparable, we used a min-max normalization to normalize scores within each category.

2.4 Network input file format

There are two types of network input files users need to prepare. One is the node attribute file and the other is network/edge attribute file. The node attribute file has 3 columns. The first column denotes snp id or gene id, and the second column denotes the score of this snp or gene. The third column specifiy if this node is a snp or gene. In the network file, there are 4 columns. The first two columns list two nodes of a given edge. The third collumn has the normalized score for this edge. The forth column specifies edge type indicating if this interaction is between snps and genes or genes.

```
edgeFile <- "example_input/EdgeFile.txt"
nodeFile <- "example_input/NodeFile.txt"
edge_set <- read.table(edgeFile, sep="\t")
node_set   <- read.table(nodeFile, sep="\t")
colnames(node_set) <- c("Node", "Node score", "Node type")
colnames(edge_set) <- c("First node", "Second node", "Edge score", "Edge type")
edge_set[98:103,]
```

```
##     First node Second node Edge score Edge type
## 98    CR004576        3043  0.7854000        EP
## 99    CR034843        3043  0.3298000        EP
## 100   CR040152        3043  0.6225000        EP
## 101       2237        5111  0.9965896        FI
## 102        506         509  0.9948976        FI
## 103        498         506  0.9827500        FI
```

```
node_set[98:103,]
```

```
##         Node Node score Node type
## 98  CR004576 0.79200000      eSNP
## 99  CR034843 0.80740000      eSNP
```

```
## 100 CR040152 0.76540000      eSNP
## 101       19 0.70045366      Gene
## 102       20 0.14164698      Gene
## 103       24 0.07013084      Gene
```

3 Prepare features for risk variants prediction

3.1 Network-based features

For most of network features such as the centrality, we wrapped up functions from "igraph" package to calculate their values. We also implemented our module identification algorithm to find modules containing snps we are intereted in. To calculate all network based features, users can simply call NetFeature(). SNPs in the same enhancer usually have same topological features. However, we can further distinguish them using their TF motif breaking scores.

```
Nodes <- as.character(edge_set[,2])
Net <- makeNet(edgeFile, nodeFile)
topoFeature <-NetFeature(Net, nodeFile, edgeFile)
```

```
## [1] "Time for running pairwise jac index: "
## [1] "Finish module merging!"
```

```
head(topoFeature)
```

```
##           Module_Score Betweenness    Closeness     Pagerank
## CR080767      2.602336      194156 2.353081e-05 0.0012016878
## CR083996      1.842770       39753 2.302741e-05 0.0003457232
## CR0911347     2.043153      211025 2.377987e-05 0.0006015015
## CR0911356     2.300549       18571 2.278298e-05 0.0002193292
## CR095246      2.202802      453964 2.380770e-05 0.0034255243
## CR095443      1.597560       74821 2.307164e-05 0.0006546970
##           Weighted_Degree SNP_Disrutption
## CR080767        24.734249       0.2655087
## CR083996         6.388578       0.3721239
## CR0911347        8.799405       0.5728534
## CR0911356        4.594794       0.9082078
## CR095246        85.214685       0.2016819
## CR095443        11.806411       0.8983897
```

3.1.1 Betweenness centrality

Betweenness is a centrality measure of a vertex within a graph. Betweenness centrality quantifies the number of times a node acts as a bridge along the shortest path between two other nodes.

```
bet_vals <- BetFeature(Net, edge_data)
head(bet_vals)
```

```
##  CR080767  CR083996 CR0911347 CR0911356  CR095246  CR095443
##    194156     39753    211025     18571    453964     74821
```

3.1.2 Closeness centrality

Closeness is a measure of the degree to which an individual is near all other individuals in a network. It is the inverse of the sum of the shortest distances between each node and every other node in the network. Closeness is the reciprocal of farness.

```r
close_vals <- CloseFeature(Net, edge_data)
head(close_vals)
```

```
##     CR080767     CR083996    CR0911347    CR0911356     CR095246
## 2.353081e-05 2.302741e-05 2.377987e-05 2.278298e-05 2.380770e-05
##     CR095443
## 2.307164e-05
```

3.1.3 Pagerank centrality

PageRank (PR) is an algorithm used by Google Search to rank websites in their search engine results. PageRank is a way of measuring the importance of website pages. In the biological networks, we can also use this algorithm to measure the importance of genes/nodes.

```r
page_vals <- PageFeature(Net, edge_data)
head(page_vals)
```

```
##     CR080767     CR083996    CR0911347    CR0911356     CR095246
## 0.0012016878 0.0003457232 0.0006015015 0.0002193292 0.0034255243
##     CR095443
## 0.0006546970
```

3.1.4 Weighted degree

The weighted degree of a node is like the degree. It's based on the number of edge for a node, but ponderated by the weigtht of each edge. It's doing the sum of the weight of the edges.

```r
wd_vals <- WDFeature(Adj_List, edge_data)
head(wd_vals)
```

```
##  CR080767  CR083996 CR0911347 CR0911356  CR095246  CR095443
## 24.734249  6.388578  8.799405  4.594794 85.214685 11.806411
```

3.1.5 Module score

Gene modules downstream of an eSNP. Our overall hypothesis is that a causal eSNP contributes to disease risk by directly causing expression changes in genes of diseaserelevant pathways. Thus, in addition to the direct target gene of the eSNP, other genes in the same pathway can also provide discriminative information. With the weighted GRN, our goal is to identify "heavy" gene modules in the network that connects a given eSNP to a set of genes

```r
mod_vals <- ModuleFeature(Adj_List, E_adj, eSNP_seeds, V_weight, Nodes)
```

```
## [1] "Time for running pairwise jac index: "
## [1] "Finish module merging!"
```

```r
head(mod_vals)
```

```
##  CR080767  CR083996 CR0911347 CR0911356  CR095246  CR095443
##  2.602336  1.842770  2.043153  2.300549  2.202802  1.597560
```

3.2 GWAVA features

ARVIN uses sequence features for the input SNPs generated by GWAVA. GWAVA is an open-source software developed by Sanger Institute. You can either upload the SNPs to GWAVA web page and get the output or download the source and run locally. For running GWAVA online navigate to https:

//www.sanger.ac.uk/sanger/StatGen_Gwava, upload the list of input SNPs and get the features in csv format, which will be input for ARVIN. If you prefer to run it locally, you need to dowload the source code from ftp://ftp.sanger.ac.uk/pub/resources/software/gwava/v1.0/src/ and annotation data from ftp://ftp.sanger.ac.uk/pub/resources/software/gwava/v1.0/source_data/ . Then you can run it local by running gwava_annotate.py and generate the features, which will be input for ARVIN.

```
gwava <- read.table("example_input/gwava_matrix.txt", header=T, sep="\t")
gwava[1:8,168:174]
```

```
##             avg_daf in_cpg seq_A seq_C seq_G seq_T repeat.
## CR080767  0.0015376      1     0     1     0     0       0
## CR083996  0.0016738      0     0     1     0     0       0
## CR0911347 0.0021400      0     0     0     1     0       0
## CR0911356 0.0013821      0     0     0     0     1       1
## CR095246  0.0006147      0     0     1     0     0       0
## CR095443  0.0006415      0     0     0     1     0       0
## CR109943  0.0019969      0     0     0     0     1       1
## CR1210014 0.0017969      0     1     0     0     0       0
```

3.3 FunSeq features

ARVIN also uses sequence features generated by FunSeq. FunSeq can also be run online or binaries can be downloaded to run locally. For running FunSeq online, navigate to http://funseq.gersteinlab.org/analysis and upload the list of SNPs that you want to analyze. In the web page, it is noted that the input SNPs can be uploaded in bed format, SNP coordinates followed by reference and alternate alleles; but we discovered that it fails to process bed input. In order to have it run, you the first two seperators need to be two spaces and last two separators need to be tabs, as follows:
chr16 · · 4526757 · · 4526758 G A
chr14 · · 52733136 · · 52733137 C A
where each dot (·) represents a space. Then, FunSeq will generate the features by selecting "bed" as the output format., which will be used as input by ARVIN.

If you prefer to run FunSeq locally, you can download FunSeq binaries from http://funseq.gersteinlab.org/static/funseq-0.1.tar.gz and extract it into your local. You will also need to download FunSeq annotation data from http://funseq.gersteinlab.org/static/data/data.tar.gz , extract it into directory that you saved the binaries. Then you can run FunSeq binary file by setting the output format to bed.

```
funseq <- read.table("example_input/funseq_matrix.txt", header=T, sep="\t")
head(funseq)
```

```
##           is_annotated_in_encode is_sensitive is_ultrasensitive
## CR080767                       1            0                 0
## CR083996                       1            0                 0
## CR0911347                      1            0                 0
## CR0911356                      1            0                 0
## CR095246                       1            0                 0
## CR095443                       1            0                 0
##           is_motif_breaking target_gene_known taget_gene_is_hub
## CR080767                  0                 0                 0
## CR083996                  0                 1                 0
## CR0911347                 0                 0                 0
## CR0911356                 0                 0                 0
## CR095246                  0                 0                 0
## CR095443                  0                 0                 0
```

4 Build a classifier for prioritizing risk varints

4.1 Train a random forest classifier

Combining network, GWAVA features and FunSeq features, a random forest model can be trained to predict risk SNPs.

```
#combine 3 types of features
group <- as.character(read.table("example_input/snp_labels.txt")[,1])
features <- data.frame(topoFeature, gwava, funseq, group)
RFmodel <- trainMod(features)#train a random forest classifier
```

```
## [1] "The random forest model has been trained!"
```

4.2 Predict causal disease variants

To run the first module of ARVIN to process the features for the SNPs, navigate into the ARVIN directory where the executable shell script process_features.sh is stored and execute it as follows: ./process_features.sh input_snps_file.bed csi_ann_output_file.txt im_pet_output_file.bed gwava_features_file.csv funseq_feautes_file.bed output_directory

This script will generate three output files, to be used as input in the second step.

1. disruption_p.txt : This file contains strongest transcription factor binding disruption caused by the SNPs being analyzed in tab separated format.
   snp ref alt TF disruption_p disruption_q log_disruption_q
   rs4784227 C T CUX1 0.0518 0.0518 1.28567024025477
   rs11568821 C T RUNX3 0.0215 0.0215 1.66756154008439

2. snp_target_gene.txt: This file lists the SNP-gene interactions in tab separated format.
   snp_id gene_symbol entrez_gene_id interaction_score
   rs200820567 ADAM7 8756 0.00450758418
   rs339331 MIR624 693209 0.058555728
   rs1542725 C1RL 51279 0.258880096

3. features_gwava_funseq.csv: This file contains the GWAVA and FunSeq features for the input SNPs in comma separated format.
   snp_id,chr,end,start,ATF3,BATF,BCL11A,BCL3,BCLAF1,...
   rs10757278,chr9,22124477,22124476,0.0,0.0,0.0,0.0,...
   rs10811656,chr9,22124472,22124471,0.0,0.0,0.0,0.0,...

```
prob <- predMod(features[,-dim(features)[2]], RFmodel)#estimate the probablity score using trained mode
head(prob)#display the probablity score as positive or negative snps
```

```
##             neg   pos
## CR080767  0.398 0.602
## CR083996  0.300 0.700
## CR0911347 0.348 0.652
## CR0911356 0.224 0.776
## CR095246  0.448 0.552
## CR095443  0.348 0.652
```