MetaNeighbor : a method to rapidly assess cell type identity using both functional and random gene sets

Megan Crow, Sara Ballouz, Manthan Shah, Jesse Gillis

2017-12-13

Contents

1 Introduction

The purpose of this method is to measure the similarity of cells across single cell RNA-sequencing (scRNA-seq) datasets by sampling from both random and functionally defined gene sets. MetaNeighbor works on the basis that cells of the same type should have more similar gene expression profiles than cells of different types. In other words, when we compare the expression profile between T cells and hepatocytes for a given gene set, we should see higher correlations among all T cells than we do between T cells and hepatocytes. This is illustrated in the schematic below:

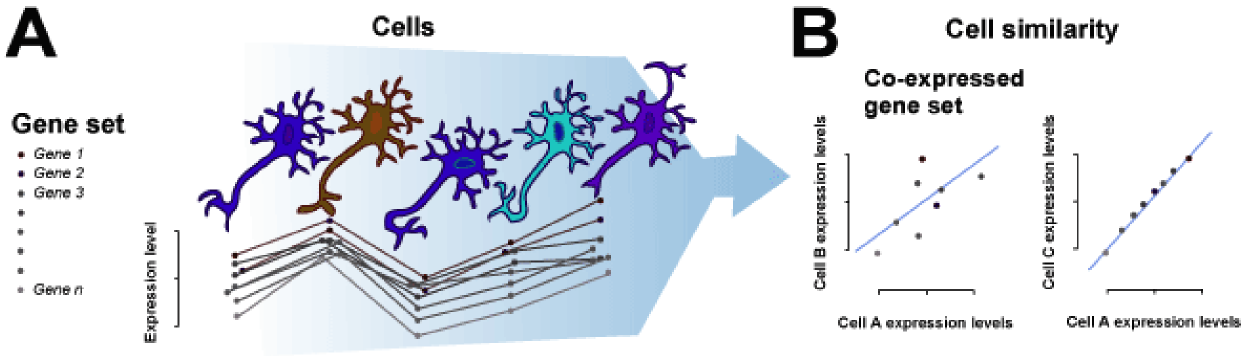


Figure 1. A. Relationship between gene set expression and cell type B. Cell similarity within and across cell types

In our approach, this is formalized through neighbor voting based on cell-cell similarities, which will be described in detail in the Methods section. In short , MetaNeighbor takes four inputs: a gene-by-sample expression matrix ("data"), a set of labels indicating each sample's dataset of origin ("experiment labels"), a set of labels indicating each sample's cell type ("cell type labels") and a set of genes ("gene sets"). The output is a performance vector ("AUROC scores"), which is the mean area under the receiver operator characteristic curve (AUROC) for the given task. This score reflects our ability to rank cells of the same known type higher than those of other types within a dataset, and can be interpreted as the probability that we will be correct about making a binary classification for a given cell (e.g. neuron vs. non-neuronal cell). An AUROC score of 0.5 means that we have performed as well as if we had randomly guessed the cell's identity.

This is a fully supervised analysis, and requires knowledge of the corresponding cell types across datasets. However, we have also used some heuristic measures to identify cell types across datasets when labels may be ambiguous or uninformative. We will walk through this unsupervised analysis in Part 2 of the vignette.

2 Data type requiremnets

For MetaNeighbor to run, the input data should be a SummarizedExperiment object (SEO) with the following considerations:

1. The gene-by-sample matrix should be an assay of SEO.
2. Gene sets of interest should be provided as a list of vectors
3. Additional data should have following vectors:
   i) sample_id : A character vector (length equal to the number of samples) containing a unique identifier for each sample
   ii) study_id : A character vector (length equal to the number of samples) that indicates the source of each sample (ex. "GSE60361" = Zeisel et al, "GSE71585" = Tasic et al, as in mn_data)
   iii) cell_type : A character vector (length equal to the number of samples) that indicates the cell type of each sample
4. cell_labels should be provided as a sample-by-cell label matrix. This should be a binary (0,1) matrix where 1 indicates cell type membership and 0 indicates non-membership

Additional requirements to be noted:

1. It is critical that genes within gene sets match the gene names of the expression matrix
2. Gene sets should contain more than one gene
3. The row names of the cell_labels object should match the column names of the expression matrix

3 System requirements/estimated run times

Because there is a ranking step, the code is not ideal for scaling in the R environment as is. The major factor is the total number of samples. Speed-ups are possible with parallelization and installing libraries such as MRAN (https://mran.revolutionanalytics.com/).

Laptop (OSX 10.10.4, 1.6 GHz, 8GB RAM, R 3.3.2, Rstudio 0.99.446)

Experiments	Cell types	Samples	Gene sets	Time (s)
2	1	100	10	0.1
2	10	100	10	0.5
2	10	100	100	1.7
2	10	100	1000	17.5

Experiments	Cell types	Samples	Gene sets	Time (s)
2	1	1000	10	9
10	1	1000	10	9
2	10	1000	10	12
2	10	1000	100	93
2	10	1000	1000	979
2	10	10000	10	3653

4 Methods

MetaNeighbor runs as follows: first, we build a network of rank correlations between all cells for a gene set. All values in the network are then re-ranked and standardized to lie between 0-1. Next, the neighbor voting predictor produces a weighted matrix of predicted labels by performing matrix multiplication between the network and the binary vector indicating cell type membership, then dividing each element by the null predictor (i.e., node degree) . That is, each cell is given a score equal to the fraction of its neighbors (including itself), which are part of a given cell type. For cross-validation, we permute through all possible combinations of leave-one-dataset-out cross- validation, and we report how well we can recover cells of the same type as area under the receiver operator characteristic curve (AUROC). This is repeated for all folds of cross-validation, and the mean AUROC across folds is reported.

4.1 Part 1: Supervised MetaNeighbor

4.1.1 Quick start

To run through the analysis and plot results, simply install the source package from here and load them into your R session:

```r
library("MetaNeighbor")
data(mn_data)
data(GOmouse)
AUROC_scores = MetaNeighbor(dat = mn_data,
                            experiment_labels = as.numeric(factor(mn_data$study_id)),
                            celltype_labels = mn_data@colData@metadata$cell_labels,
                            genesets = GOmouse,
                            bplot = TRUE)
```

4.1.2 More detail

We have provided sample data, as well as sample gene sets on our Github site. In this sample data, we have included the cortical interneurons from two public datasets,GSE60361 and GSE71585 (RPKM). A subset of ~3000 genes and 10 genesets have been included for demonstration. For this example, we will be testing how well we can identify the Sst Chodl subtype (Sst-Chodl from GSE71585 and Int1 from GSE60361) and the Smad3 subtype (Smad3 from GSE71585 and Int14 from GSE60361), relative to all other interneurons within their respective experiments.

MetaNeighbor requires a SummarizedExperimentObject as input, formatted as specified above.

There are two outputs of the method:

1. A matrix of AUROC scores representing the mean for each gene set tested for each celltype

2. A beanplot displaying density of AUROC scores for each cell type (by default the plot will be displayed and can be turned off by setting the argument)

4.1.2.1 Install data and R functions

To run through the analysis, simply install the source package from our Github site and load them into your R session:

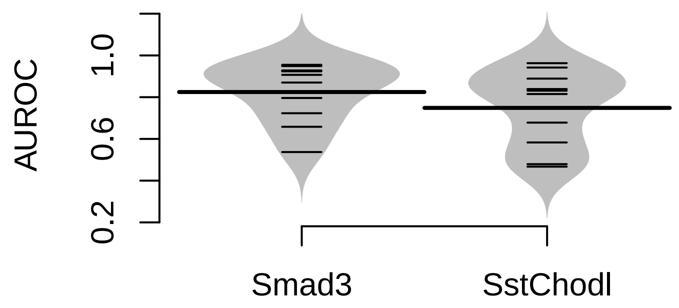
```
library("MetaNeighbor")
data(mn_data)
data(GOmouse)
```

4.1.2.2 Run MetaNeighbor and plot results

As MetaNeighbor runs, it outputs the name of the gene set that is being evaluated. When all gene sets have been tested, MetaNeighbor will return a gene set-by-cell type matrix of AUROC scores. A smoothed distribution of scores for each cell type will be plotted by default (turn off plotting by setting). Short horizontal lines inside the shape indicate AUROC values for individual gene sets, and the large horizontal line represents the mean.

```
AUROC_scores = MetaNeighbor(dat = mn_data,
                            experiment_labels = as.numeric(factor(mn_data$study_id)),
                            celltype_labels = mn_data@colData@metadata$cell_labels,
                            genesets = GOmouse,
                            bplot = TRUE)
```

```
## [1] "GO:0016853"
## [1] "GO:0005615"
## [1] "GO:0005768"
## [1] "GO:0007067"
## [1] "GO:0065003"
## [1] "GO:0042592"
## [1] "GO:0005929"
## [1] "GO:0008565"
## [1] "GO:0016829"
## [1] "GO:0022857"
```



```r
head(AUROC_scores)
```

```
##            SstChodl Smad3
## GO:0016853    0.678 0.658
## GO:0005615    0.963 0.949
## GO:0005768    0.815 0.870
## GO:0007067    0.583 0.723
## GO:0065003    0.839 0.928
## GO:0042592    0.889 0.955
```

AUROC scores greater than 0.5 indicate improvement over randomly guessing the identity of the cell type of interest.

4.2 Part 2: Unsupervised MetaNeighbor

4.2.1 Quick start

To run through the analysis and plot results, simply install the source package from our Github site and load them into your R session:

```r
library(MetaNeighbor)
data(mn_data)
var_genes = variableGenes(dat = mn_data, exp_labels = mn_data$study_id)
celltype_NV = MetaNeighborUS(var_genes = var_genes,
                             dat = mn_data,
                             study_id = mn_data$study_id,
                             cell_type = mn_data$cell_type)
top_hits = topHits(cell_NV = celltype_NV,
                   dat = mn_data,
                   study_id = mn_data$study_id,
                   cell_type = mn_data$cell_type,
                   threshold = 0.9)
top_hits
cols = rev(colorRampPalette(RColorBrewer::brewer.pal(11,"RdYlBu"))(100))
breaks = seq(0, 1, length=101)
gplots::heatmap.2(celltype_NV,
                  margins=c(8,8),
                  keysize=1,
                  key.xlab="AUROC",
                  key.title=NULL,
                  trace = "none",
                  density.info = "none",
                  col = cols,
                  breaks = breaks,
                  offsetRow=0.1,
                  offsetCol=0.1,
                  cexRow = 0.7,
                  cexCol = 0.7)
```

4.2.2 More detail

While ideally we would like to perform supervised analyses to investigate cell type identity, in some cases it is difficult to know how cell type labels compare across datasets. For this situation, we came up with a heuristic to allow researchers to make an educated guess about overlaps without requiring in-depth knowledge of marker genes. This was based on our observation that large, high variance gene sets tended to provide improved scores for known cell types.

Unsupervised Metaneighbor requires an input SEO as specified above, as well as a vector containing a set of variable genes ("var_genes"). The function will use the set of variable genes to create a cell-cell similarity network.

The output of the function is a cell type-by-cell type mean AUROC matrix, which is built by treating each pair of cell types as testing and training data for MetaNeighbor, then taking the average AUROC for each pair (NB AUROC scores across testing and training folds will not be identical because each test cell type is scored out of its own dataset, and differences in dataset heterogeneity influence scores). When comparing datasets that contained similar cell types, we found that cell types that were the best hit for one another ("reciprocal top hits"), and cell types with scores >0.9 tended to be good candidates for downstream tests of cell type identity using Supervised MetaNeighbor. This rule will not hold when experiments contain wholly different cell types (e.g., comparing brain to pancreas will likely yield some spurious overlaps), or when datasets are very unbalanced with respect to one another.

4.2.2.1 Install data and R functions

We have provided sample data and source code here. To begin the analysis, simply download and install the source package from the above link into your R session. You will also need to install two helper packages, gplots and RColorBrewer.

```r
library(MetaNeighbor)
data(mn_data)
```

4.2.2.2 Identify a highly variable gene set

To begin, we will use the function variableGenes, which picks the top quartile of variable genes across all but the top decile of expression bins for each dataset, then provides the intersect across datasets as the output.

```r
var_genes = variableGenes(dat = mn_data, exp_labels = mn_data$study_id)
head(var_genes)
```

```
## [1] "1110017D15Rik" "1190002N15Rik" "3110043O21Rik" "Aacs"
## [5] "Abcb10"        "Abcb6"
```

```r
length(var_genes)
```

```
## [1] 331
```

In this case, we return a set of 331 highly variable genes. AUROC scores depend on both gene set size and variance. If the size of the returned set is small (<200 genes) the suggested AUROC cut-off of >0.9 may not be applicable, and it may be helpful to increase the gene set size. This may be done by taking a majority rule on genes included in the highly variable sets of each dataset in the analysis (i.e., include a gene if it is highly variable in >50% of datasets) . This strategy is likely to be required with an increasing number of datasets included. However, we note that if few genes are returned when using a small number of datasets (2-3), this may indicate that the datasets have different cell type compositions, or have very different gene coverage. Under these circumstances, we do not recommend the use of unsupervised MetaNeighbor.

4.2.2.3 Run Unsupervised MetaNeighbor

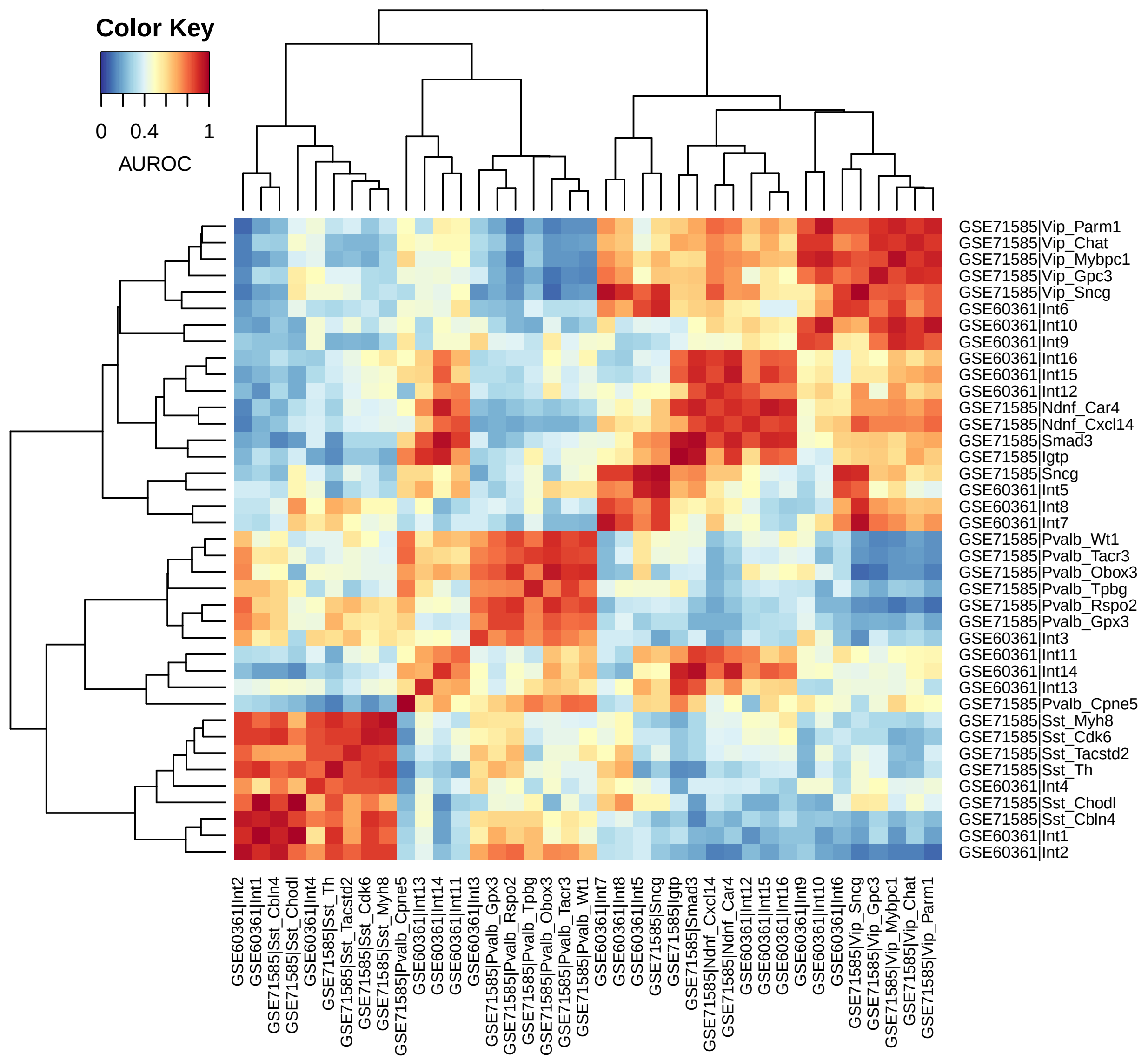
Once we have a set of highly variable genes, we can simply run an unsupervised version of MetaNeighbor using the function:

```r
celltype_NV = MetaNeighborUS(var_genes = var_genes,
                             dat = mn_data,
                             study_id = mn_data$study_id,
                             cell_type = mn_data$cell_type)
```

4.2.2.4 Plot results

Results can be plotted as follows:

```r
cols = rev(colorRampPalette(RColorBrewer::brewer.pal(11,"RdYlBu"))(100))
breaks = seq(0, 1, length=101)
gplots::heatmap.2(celltype_NV,
                  margins=c(8,8),
                  keysize=1,
                  key.xlab="AUROC",
                  key.title=NULL,
                  trace = "none",
                  density.info = "none",
                  col = cols,
                  breaks = breaks,
                  offsetRow=0.1,
                  offsetCol=0.1,
                  cexRow = 0.7,
                  cexCol = 0.7)
```



This plot shows the AUROC scores between each testing and training pair. Red indicates a higher score and blue indicates a lower score. Note that the diagonal is not equal to one. This is because of the scoring system: cell types that are ‘promiscuous’ (i.e., have broad similarity to many types) will tend to have higher node degrees in the network. Differences in degree across cell types will affect scores as predictions are standardized by this factor. Within-dataset scores are shown for visualization purposes only, and should not be used for replicability inference.

4.2.2.5 Identify reciprocal top hits and high scoring cell type pairs

To find reciprocal top hits and those with AUROC>0.9 we use the following code:

```
top_hits = topHits(cell_NV = celltype_NV,
                   dat = mn_data,
                   study_id = mn_data$study_id,
                   cell_type = mn_data$cell_type,
                   threshold = 0.9)
top_hits
```

```
##     Study_ID|Celltype_1  Study_ID|Celltype_2 Mean_AUROC         Match_type
## 1         GSE60361|Int1   GSE71585|Sst_Chodl       0.99 Reciprocal_top_hit
## 2         GSE60361|Int7    GSE71585|Vip_Sncg       0.97 Reciprocal_top_hit
## 3        GSE60361|Int14       GSE71585|Smad3       0.97 Reciprocal_top_hit
## 4         GSE60361|Int5        GSE71585|Sncg       0.96 Reciprocal_top_hit
## 5        GSE60361|Int10   GSE71585|Vip_Parm1       0.95 Reciprocal_top_hit
## 6         GSE60361|Int2   GSE71585|Sst_Cbln4       0.95 Reciprocal_top_hit
## 7        GSE60361|Int15   GSE71585|Ndnf_Car4       0.94 Reciprocal_top_hit
## 8        GSE60361|Int10  GSE71585|Vip_Mybpc1       0.93          Above_0.9
## 9        GSE60361|Int14        GSE71585|Igtp       0.92          Above_0.9
## 10        GSE71585|Sncg        GSE60361|Int6       0.92          Above_0.9
## 11  GSE71585|Ndnf_Car4       GSE60361|Int16       0.92          Above_0.9
## 12       GSE60361|Int12 GSE71585|Ndnf_Cxcl14       0.91 Reciprocal_top_hit
## 13 GSE71585|Vip_Mybpc1        GSE60361|Int9       0.91          Above_0.9
## 14   GSE71585|Vip_Sncg        GSE60361|Int8       0.91          Above_0.9
## 15        GSE60361|Int3 GSE71585|Pvalb_Obox3       0.81 Reciprocal_top_hit
```

These top hits can then be used for supervised analysis, making putative cell type labels for each unique grouping (see Part 1).

For any assistance reproducing analyses please contact mcrow@cshl.edu or jgillis@cshl.edu