Scaling up reproducible research for single cell transcriptomics using MetaNeighbor (Anticipated results)

MetaNeighbor anticipated results

Because MetaNeighbor is non-parametric, there is no fine-tuning to be done for any of the procedures presented here. Over time, we have identified two sources of potential error: bad highly variable gene selection and coding or formatting errors, which can be easily diagnosed by looking at AUROC heatmaps. We expect AUROCs to correctly represent global relationships between cell types and generalize across studies.

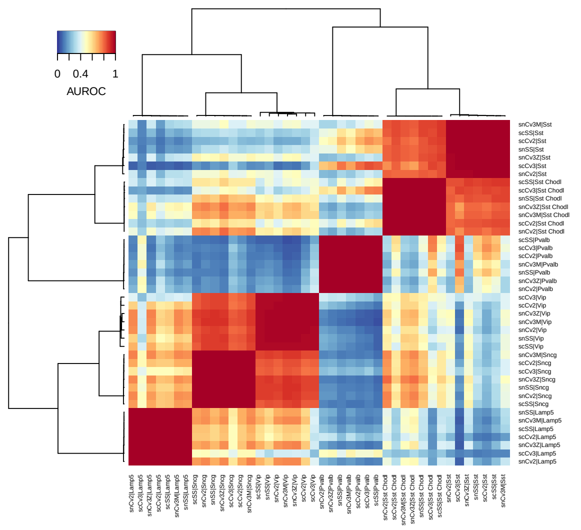
Bad gene set selection

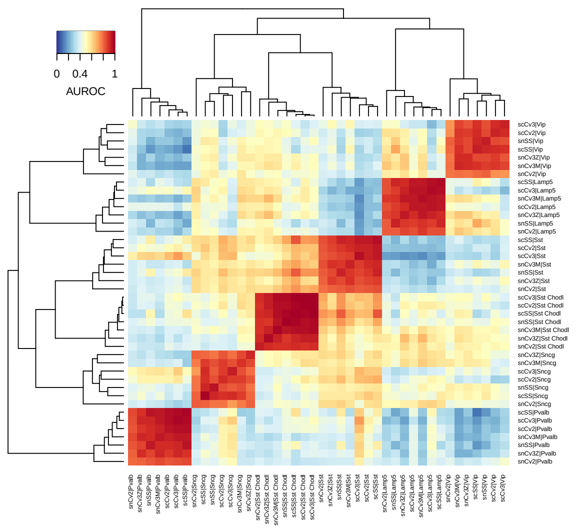
The most common problem is to forget to select a set of highly variable genes, which is expected to dampen the impact of technical variability on neighbor voting. First, we present an example of a correct analysis, where we load the BICCN GABAergic neurons, select highly variable genes, and compute cluster similarities (see Procedure 1 for more details).

```r
library(MetaNeighbor)
biccn_data = readRDS("biccn_gaba.rds")
biccn_hvgs = variableGenes(biccn_data, exp_labels = biccn_data$study_id)

# GOOD
aurocs = MetaNeighborUS(var_genes = biccn_hvgs,
                        dat = biccn_data,
                        study_id = biccn_data$study_id,
                        cell_type = biccn_data$joint_subclass_label,
                        fast_version = TRUE)
plotHeatmap(aurocs, cex = 0.5)

# BAD
aurocs = MetaNeighborUS(var_genes = sample(rownames(biccn_data), length(biccn_hvgs)),
                        dat = biccn_data,
                        study_id = biccn_data$study_id,
                        cell_type = biccn_data$joint_subclass_label,
                        fast_version = TRUE)
plotHeatmap(aurocs, cex = 0.5)
```





We recognize strong replicability structure, evidenced by the presence of dark red blocks. When we repeat the analysis with random genes, the replicability structure is still present, but we recognize two signatures of bad gene set selection: (a) AUROCs are low overall (shift to light red and orange), (b) within red blocks, there is a clear gradient structure. In our experience, there are 3 scenarios that lead to bad gene selection: errors in gene symbol conversion, errors when genes are stored as factors in R (that are implicitly converted to numerical values during indexing), forgetting to select highly variable genes altogether.

No overlap between datasets

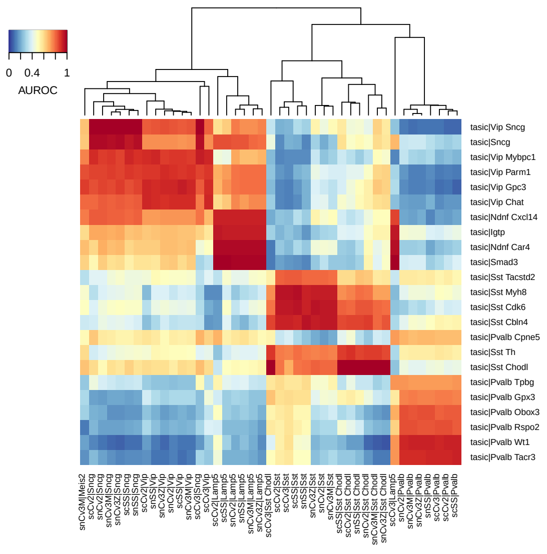
The second problem occurs when there is no overlap between datasets. We illustate this problem with the data from Procedure 2, where we expect all cell types from the Tasic dataset to be present in the pre-trained BICCN model. According to our expectations, all cell types have strong hits with BICCN clusters and we see a hierarchical structure that is consistent with prior biological knowledge: lighter red blocks corresponding to MGE and CGE-derived inhibitory neurons. We compare with the same block of code, where we mistakenly keep non-neurons from the BICCN taxonomy instead of inhibitory neurons. The lack of biological overlap can be deduced from 3 factors: (a) low AUROC values overall, (b) almost no strong hits (contrary to expectations), (c) lack of expected hierarchical structure (MGE and CGE derived interneurons).

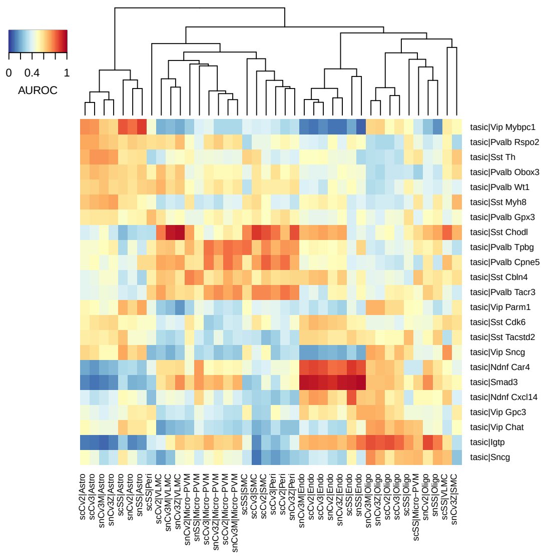
```
library(scRNAseq)
tasic = TasicBrainData(ensembl = FALSE)
tasic$study_id = "tasic"

biccn_subclasses = read.table("pretrained_biccn_subclasses.txt", check.names = FALSE)
global_aurocs = MetaNeighborUS(
  trained_model = biccn_subclasses, dat = tasic,
  study_id = tasic$study_id, cell_type = tasic$primary_type,
  fast_version = TRUE
)
gabaergic_tasic = splitTestClusters(global_aurocs, k = 4)[[2]]

# GOOD
gabaergic_biccn = splitTrainClusters(global_aurocs[gabaergic_tasic,], k = 4)[[4]]
keep_cell = makeClusterName(tasic$study_id, tasic$primary_type) %in% gabaergic_tasic
tasic_subdata = tasic[, keep_cell]
aurocs = MetaNeighborUS(
  trained_model = biccn_subclasses[, gabaergic_biccn],
  dat = tasic_subdata, study_id = tasic_subdata$study_id,
  cell_type = tasic_subdata$primary_type, fast_version = TRUE
)
plotHeatmapPretrained(aurocs, cex = 0.7)

# BAD: non-neurons instead of GABAergic neurons
gabaergic_biccn = splitTrainClusters(global_aurocs, k = 5)[[1]]
keep_cell = makeClusterName(tasic$study_id, tasic$primary_type) %in% gabaergic_tasic
tasic_subdata = tasic[, keep_cell]
aurocs = MetaNeighborUS(
  trained_model = biccn_subclasses[, gabaergic_biccn],
  dat = tasic_subdata, study_id = tasic_subdata$study_id,
  cell_type = tasic_subdata$primary_type, fast_version = TRUE
)
plotHeatmapPretrained(aurocs, cex = 0.7)
```



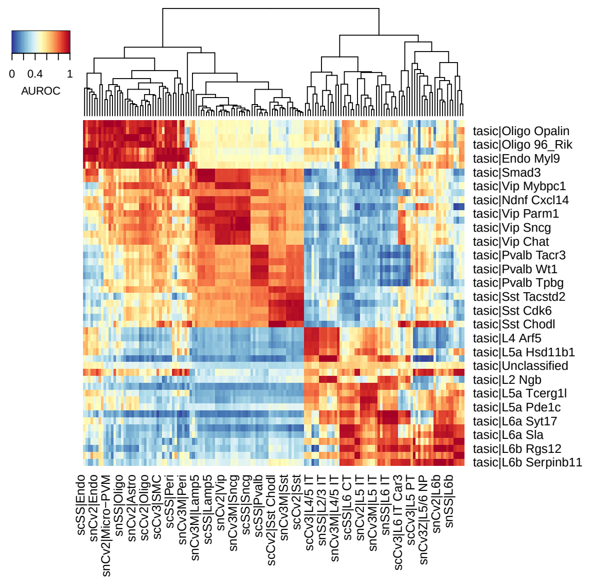


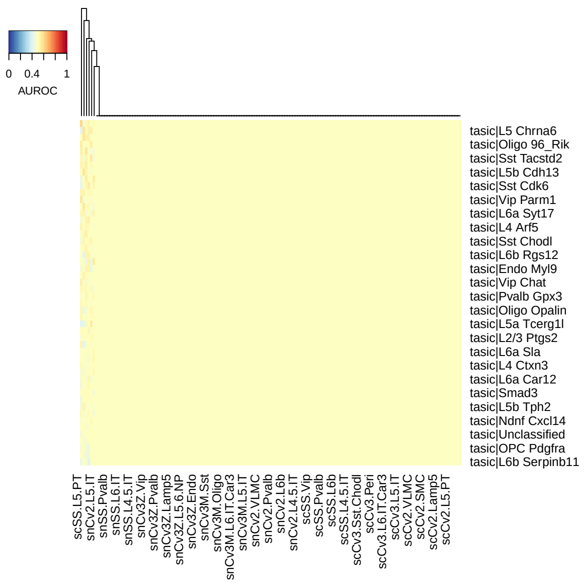
Pretrained MetaNeighbor: bad name formatting

The third problem we have encountered is a mistake that occurs when loading pre-trained model and forgetting to specify "check.names = FALSE", which is essential to preserve correct formatting of cell type names. Below, we present an example of correct code based on data from Procedure 2. We obtain the expected replicability structure, with evidence of strong hits across all cell types (see Procedure 2 for further details and analyses). When we forget "check.names = FALSE", MetaNeighbor is unable to correctly recognize dataset names and cell type names in the pre-trained model, the similarity computations become meaningless, leading to AUROC values that are essentially 0.5. This problem is easy to diagnose and fix, but can be very confusing when it occurs.

```
# GOOD
biccn_subclasses = read.table("pretrained_biccn_subclasses.txt", check.names = FALSE)
aurocs = MetaNeighborUS(
  trained_model = biccn_subclasses, dat = tasic,
  study_id = tasic$study_id, cell_type = tasic$primary_type,
  fast_version = TRUE
)
plotHeatmapPretrained(aurocs)

# BAD
biccn_subclasses = read.table("pretrained_biccn_subclasses.txt")
aurocs = MetaNeighborUS(
  trained_model = biccn_subclasses, dat = tasic,
  study_id = tasic$study_id, cell_type = tasic$primary_type,
  fast_version = TRUE
)
plotHeatmapPretrained(aurocs)
```





Generalizable quantification of cell type similarities

In their computation, MetaNeighbor's AUROCs are directly related to Spearman correlations. More precisely, all computations are based on average Spearman correlations between cells from two cell types, but include an additional prediction that alleviate batch effects, while keeping an interpretability power that is comparable to correlations (where AUROC = 0 maps to correlation = -1, AUROC = 0.5 maps to correlation = 0, AUROC = 1 maps to correlation = 1).

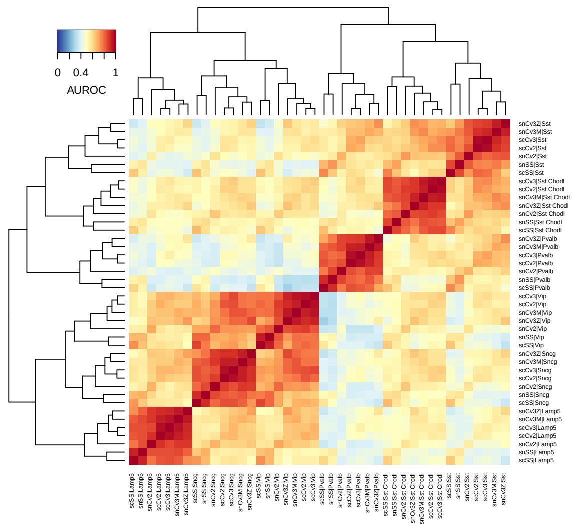
To appreciate how the additional prediction step enables us to obtain "batch free correlations", we compare MetaNeighbor's output with a more naive similarity output, where we compute the Spearman correlation between cell type centroids. Centroid correlations display two desirable patterns: centroids cluster primarily by cell type (then by dataset), we see global hierarchical structure (MGE-derived interneurons vs CGE-derived interneurons). However, batch effects are clearly visible throughout the heatmap. For example, within each cell type, Chromium-based datasets tend to cluster on one side, SmartSeq-based datasets on the other side. In contrast, for an equivalent computation time, all the "good" patterns are made pristinely clear with MetaNeighbor AUROCs. Technical substructure has been lost: technologies mix well within cell-types, homogenous cell groupings look uniform, and biological relationships between cell types are correctly displayed.

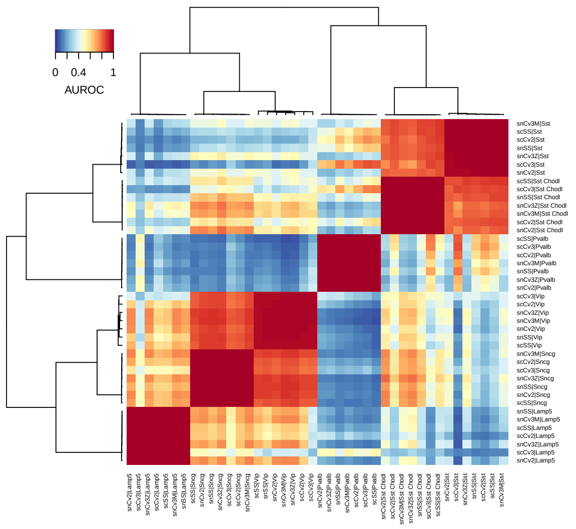
```
cell_types = as.factor(
    makeClusterName(biccn_data$study_id, biccn_data$joint_subclass_label)
)
normalization_factor = Matrix::colSums(assay(biccn_data)) / 1000000
```

```r
cpm = assay(biccn_data)
cpm@x = cpm@x / rep.int(normalization_factor, diff(cpm@p))
cpm = as.matrix(cpm[biccn_hvgs,])

centroids = sapply(levels(cell_types), function(ct) {
    matrixStats::rowMeans2(log2(cpm+1), cols = cell_types == ct)
})

centroid_cor = cor(centroids, method = "spearman")
aurocs = MetaNeighborUS(var_genes = biccn_hvgs,
                        dat = biccn_data,
                        study_id = biccn_data$study_id,
                        cell_type = biccn_data$joint_subclass_label,
                        fast_version = TRUE)
plotHeatmap((1+centroid_cor)/2, cex = 0.5)
plotHeatmap(aurocs, cex = 0.5)
```





Compared to correlations, AUROCs have one additional "parameter": the outgroup used for the prediction task. In Procedure 1, we illustrated how the outgroup can be controled and interpreted. A deviation from AUROC = 1 can thus be interpreted as a combination of two factors: lack of similarity between cell types and choice of outgroup (difficulty of prediction task). If the outgroup is well controled, AUROC values will generalize across studies and fundamentally indicate the quality of the clustering. For a given cell type in a given background (for example Sst cells in an unbiased sample of MOp interneurons), the similarity to Sst cells in another dataset (or any other interneuron type for that matter) should be in the range of similarity observed within the BICCN.

As a robust alternative to centroid correlations, MetaNeighbor AUROCs can be applied to simple pre-processing tasks, such as identifying and selecting cell types that overlap between datasets before applying a merging framework. However, beyond the purely applicative viewpoint, we believe that MetaNeighbor-style AUROCs are a stepping stone toward a generalizable formalization of cell type similarity.