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

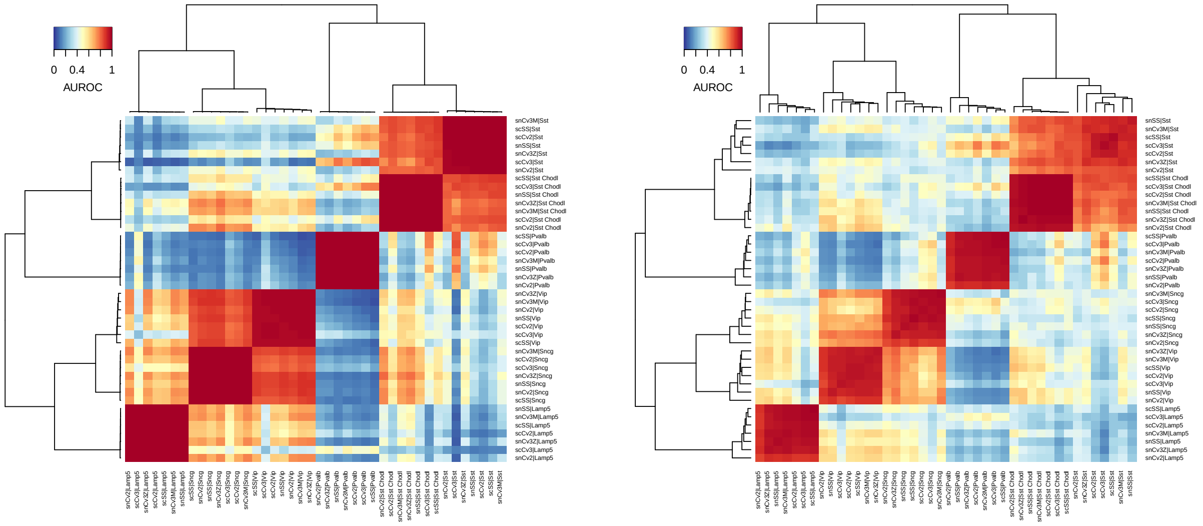
MetaNeighbor anticipated results

Because MetaNeighbor is non-parametric, there is no fine-tuning to be done for any of the protocols 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.

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 Protocol 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.

Pretrained MetaNeighbor: bad name formatting

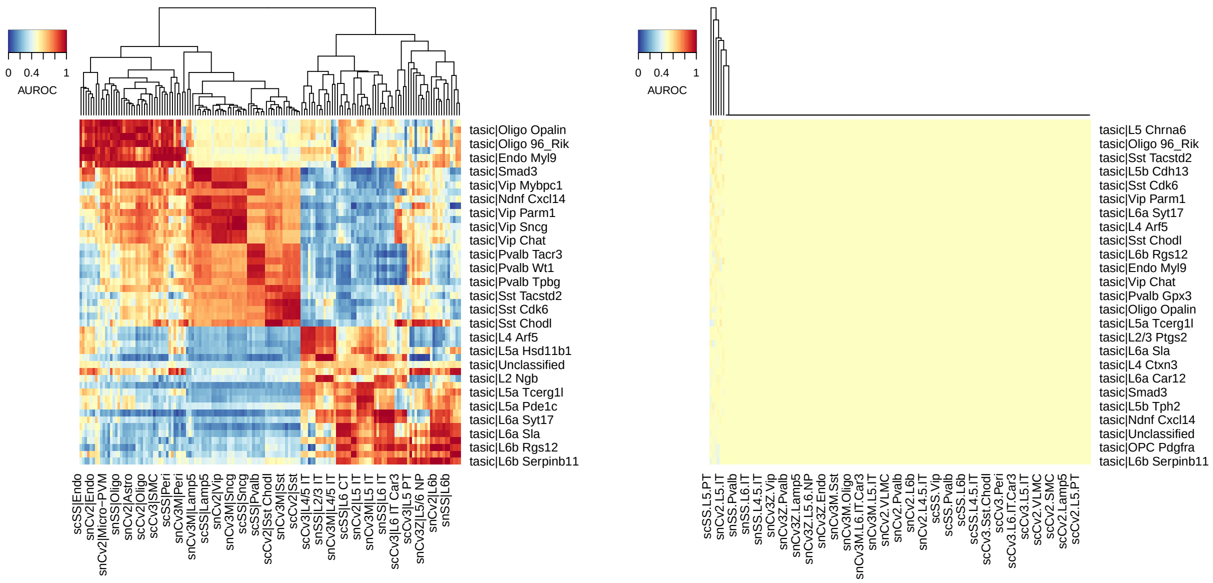
The second problem we have encountered is a mistake that occurs when loading pre-trained model and forgetting to specify "check.names = FALSE", which is essential to preserve correct formatting of cell type names. Here is an example of correct code based on data from Protocol 2.

```
library(scRNAseq)
tasic = TasicBrainData(ensembl = FALSE)
```

```
## snapshotDate(): 2020-04-27

## see ?scRNAseq and browseVignettes('scRNAseq') for documentation

## loading from cache

## see ?scRNAseq and browseVignettes('scRNAseq') for documentation

## loading from cache
```

```
tasic$study_id = "tasic"

# 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)
```

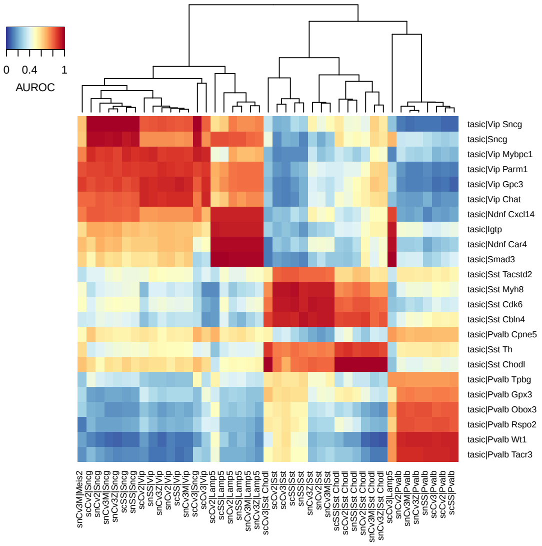


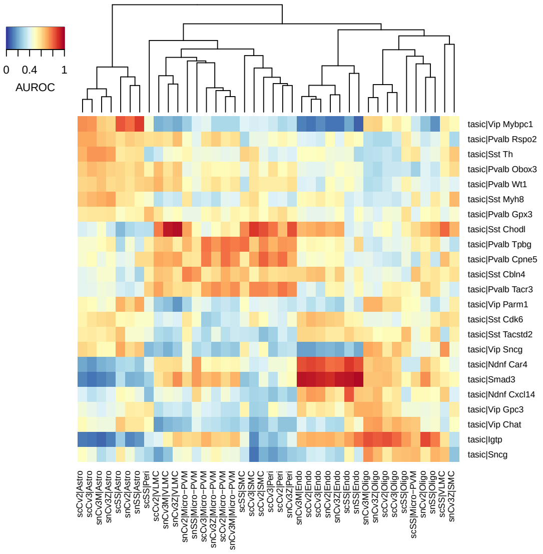
We obtain the expected replicability structure, with evidence of strong hits across all cell types (see Protocol 2 for further details and analyses). When we forget "check.names = FALSE", MetaNeighbor is unable to correctly recognize dataset names and cell type names in the pre-trained model, the similarity 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.

No overlap between datasets

The final problem occurs when there is no overlap between datasets. We illustarte this problem with the data from Protocol 2, where we expect all cell types from the Tasic dataset to be present in the pre-trained BICCN model.

```
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)
```





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).