About

**GsRCL**

This method adopts a Gaussian noise-augmented single-cell RNA-seq contrastive learning approach, where it exploits the well-known Gaussian distribution to create views for self-supervised contrastive learning. It is designed for cell-type identification tasks, such as reference-query tasks, where we use the reference to annotate the unknown cells in the query. For simplicity, no pre-processing or genes selection is required, the query genes expression matrix should have raw counts or log-transformed counts (i.e. log(1 + count)). We break the task into several binary classification tasks. We cross reference the query genes against the reference genes. If more that 50% of the genes match, we obtain a set of probabilities for each query cell,where each probability is associated with a cell type in the selected reference. However, if the number of matching genes is less than or equal 50%, we suggest selecting a different reference.

**AF-RCL**

This method adopts an augmentation-free single-cell RNA-Seq contrastive learning approach, where it conducts supervised contrastive learning on the original cells without any augmentation. It is designed for cell-type identification tasks, such as reference-query tasks, where we use the reference to annotate the unknown cells in the query. For simplicity, no pre-processing or genes selection is required, the query genes expression matrix should have raw counts or log-transformed counts (i.e. log(1 + count)). The task is treated as a multi-class classification task. We cross reference the query genes against the reference genes. If more that 50% of the genes match, we obtain a set of probabilities for each query cell,where each probability is associated with a cell type in the selected reference. However, if the number of matching genes is less than or equal 50%, we suggest selecting a different reference.

Input

The below commands apply to both GsRCL and AF-RCL.

Step 1: Verify the input file

python verify\_hdf.py \

-i [the full path to the input file] \

-o [Default='.' - the full path to the output directory] \

-r [the name of the reference] \

-t [transpose the matrix if rows are genes and columns are cells] \

-mp [the full path to the reference directory] \

-mat [Optional - the key to the matrix object] \

-obs [Optional - the full tree to the observations object, i.e rows] \

-var [Optional - the full tree to the variables/features object, i.e columns] \

Step 2: Obtain predictions

python gsrcl\_predict.py \

-o [Default='.' - the full path to the output directory] \

-r [the name of the reference] \

-t [transpose the matrix if rows are genes and columns are cells] \

-mp [the full path to the reference directory] \

-p [Default=0.5 - select a p-value cut-off for putative new cell types] \

--log [log transform the input matrix if it contains raw counts] \

Examples

Here we provide some examples that apply to both GsRCL and AF-RCL, where the file *gsrcl\_predict.py* can be replaced with the file *afrcl\_predict.py*

**Example 1**

Log-transform the input matrix without transposing it and use the Quake\_Smart-seq2\_Limb\_Muscle reference to obtain predictions, where default values are used.

*python verify\_hdf.py -i [INPUT FILE] -r Quake\_Smart-seq2\_Limb\_Muscle -mp [PATH TO REFERENCE DIRECTORY] -t 0*

*python gsrcl\_predict.py -r Quake\_Smart-seq2\_Limb\_Muscle -mp [PATH TO REFERENCE DIRECTORY] --log 1 -t 0*

**Example 2**

Transpose the input matrix without log-transforming it and use the Adam reference to obtain predictions, where p-value is set to 0.7.

*python verify\_hdf.py -i [INPUT FILE] -r Adam -mp [PATH TO REFERENCE DIRECTORY] -t 1*

*python gsrcl\_predict.py -r Adam -mp [PATH TO REFERENCE DIRECTORY] -p 0.7 --log 0 -t 1*

**Example 3**

Same as Example 2 but set the output directory.

*python verify\_hdf.py -i [INPUT FILE] -o [PATH TO OUTPUT DIR] -r Adam -mp [PATH TO REFERENCE DIR] -t 1*

*python gsrcl\_predict.py -o [PATH TO OUTPUT DIR] -r Adam -mp [PATH TO REFERENCE DIRECTORY] -p 0.7 --log 0 -t 1*

**Example 4**

Same as Example 3 but with providing the keys to the h5 objects in the tree, where the required objects at depth 1 in the tree.

*python verify\_hdf.py -i [INPUT FILE] -o [PATH TO OUTPUT DIR] -r Adam -mp [PATH TO REFERENCE DIR] -t 1 -mat exprs -obs obs\_names -var var\_names*

*python gsrcl\_predict.py -o [PATH TO OUTPUT DIR] -r Adam -mp [PATH TO REFERENCE DIRECTORY] -p 0.7 --log 0 -t 1*

A computer screen shot of a black screen

Description automatically generated

**Example 5**

Same as Example 4 but the required objects at depth 2 in the tree. The keys should be space delimited.

*python verify\_hdf.py -i [INPUT FILE] -o [PATH TO OUTPUT DIR] -r Adam -mp [PATH TO REFERENCE DIR] -t 1 -mat X -obs obs barcode -var var feature\_name*

*python gsrcl\_predict.py -o [PATH TO OUTPUT DIR] -r Adam -mp [PATH TO REFERENCE DIRECTORY] -p 0.7 --log 0 -t 1*

A computer screen shot of a black screen

Description automatically generated

**Example 6**

Same as Example 4 but the obs object at depth 2 and the var object at depth 3 in the tree. The keys should be space delimited.

*python verify\_hdf.py -i [INPUT FILE] -o [PATH TO OUTPUT DIR] -r Adam -mp [PATH TO REFERENCE DIR] -t 1 -mat matrix -obs matrix barcodes -var matrix features id*

*python gsrcl\_predict.py -o [PATH TO OUTPUT DIR] -r Adam -mp [PATH TO REFERENCE DIRECTORY] -p 0.7 --log 0 -t 1*

A computer screen shot of a computer code

Description automatically generated

Output

There are two output files, the first is **probabilities.csv** that shows a set of probabilities for each query cell, where each probability is associated with a cell type in the selected reference. Each query cell is annotated based on the cell-type with the highest probability, however, if the highest probability is less than or equal to the p-value, the cell is annotated as “Unassigned” to consider new cell types. The second file is **tsne.svg** that shows a scatter plot of a 2D projection of the query matrix using t-SNE. It also illustrates the annotations in the first file.

The scripts creates four intermediary files with type .npy that should be deleted after the output is presented.

Reference file structure

Given the below file structure, the -mp argument for GsRCL should be set as -mp …/GsRCL and for AF-RCL should be set as -mp …/AFRCL

A close-up of a form

Description automatically generated

Sample h5 files for testing

**File name: sample1.h5 (Unknown organ)**

Case 1: Try to read the file without setting the optional arguments.

*python verify\_hdf.py -i …/sample1.py -r Quake\_Smart-seq2\_Limb\_Muscle -mp [PATH TO REFERENCE DIRECTORY] -t 0*

Output:

A black screen with white text

Description automatically generated

Case 2: Based on the printed tree, provide space delimited keys for the -obs and -var arguments.

*python verify\_hdf.py -i …/sample1.py -r Quake\_Smart-seq2\_Limb\_Muscle -mp [PATH TO REFERENCE DIRECTORY] -t 0 -obs matrix barcodes -var matrix features name*

Output:

*A computer screen with white text

Description automatically generated*

Case 3: Based on the error message, the name object has duplicates, hence we replace it with the id object

*python verify\_hdf.py -i …/sample1.py -r Quake\_Smart-seq2\_Limb\_Muscle -mp [PATH TO REFERENCE DIRECTORY] -t 0 -obs matrix barcodes -var matrix features id*

Output:

**

None of the query genes match the selected reference, where the reference’s organ is Mouse Limb Muscle.

**File name: sample2.h5 (Mouse Diaphragm)**

Case 1: Try to read the file without setting the optional arguments.

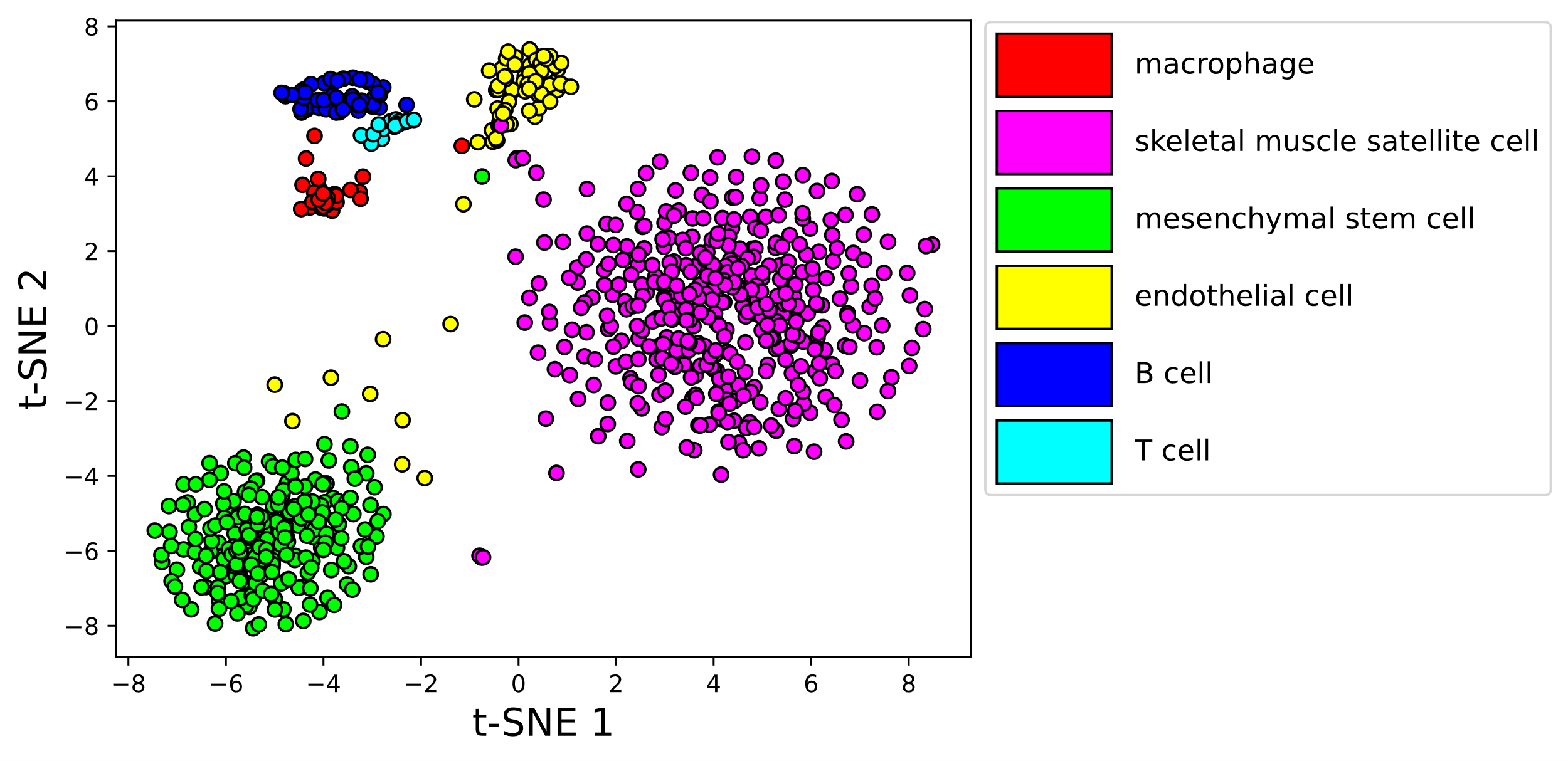
*python verify\_hdf.py -i …/sample2.py -r Quake\_Smart-seq2\_Limb\_Muscle -mp [PATH TO REFERENCE DIRECTORY] -t 0*

Output:

The file verified successfully.

Case2: Try to obtain the probabilities without setting the p-value.

Output:



Case 3: Try to obtain the probabilities but with setting the p-value to 0.9 following scPred.

Output:

