Preprocessing

Reference Dataset

Goal

Reproduce part of the pipeline from the paper *"Immune cell type signature discovery and random forest classification for analysis of single cell gene expression datasets"* by Aybey et al., 2023, using the GitHub repository:

https://github.com/ba306/immune-cell-signature-discovery-classification-paper

This involves preparing both a **reference dataset** and a **query dataset**. We started with the **Hao PBMC reference dataset**, which will later be used for training the xgboost and lightgbm classifiers as was done with Random Forest.

Steps Completed: Preparing the Reference Dataset (Hao et al.)

1. Download the dataset

We manually downloaded the processed 10x Genomics data (barcodes, features, matrix) for sample `GSM5008737_RNA_3P` from the GEO accession GSE164378. The name of the file is `GSE164378_RAW.tar` . We also downloaded the metadata: `GSE164378_sc.meta.data_3P.csv.gz` .

This dataset corresponds to **CITE-seq PBMC data**, and was used in the paper as the **reference dataset**.

---

2. Organize the data

We renamed the downloaded files and put them inside the correct folder so that Seurat's `Read10X()` could detect:

```
./data/Hao/
├── barcodes.tsv.gz
├── features.tsv.gz
└── matrix.mtx.gz
```

3. Run the preprocessing script

After downloading and installing all the dependencies needed in RStudio (Seurat, dplyr), we executed the script `Benchmarking_preprocess_Hao_ref.R`.

Before running the script we made sure to comment out this line of code:

```
VlnPlot(tiss_immune, features = c("nFeature_RNA", "nCount_RNA", "percent.mt"), nc
```

It is memory hungry and will make the script to fail, probably needs more than 16 GM of memory to run. It also uses a non defined object, `tiss_immune` and will error anyway with:

```
Error in eval(ei, envir) : object 'tiss_immune' not found
```

The VlnPlot function, commonly used in Seurat for visualizing single-cell data, doesn't have a specific built-in memory size limit. However, the memory usage of VlnPlot is directly related to the size and complexity of the input data, such as the number of cells and features being plotted, and the plot's resolution. Very large datasets can lead to high memory consumption and potential performance issues.

After commenting out this line, the rest of the script:

- Reads the count matrix with `Read10X()`
- Loads metadata (cell type annotations) from the CSV file `GSE164378_sc.meta.data_3P.csv`
- Creates a Seurat object combining expression and metadata
- Calculates percent mitochondrial genes per cell
- Filters out cells annotated as "Doublet", "Eryth", and "Platelet"
- Normalizes the expression matrix
- Harmonizes detailed cell labels (`celltype.l2`) into medium-level categories such as:
    - `T CD4`, `T CD8`, `B`, `Mono`, `NK`, `DC`, `Other cells`

The resulting Seurat object contains:

- ~159k cells
- ~33k genes
- Mean ~2207 genes per cell

Finally, it saves the object in `.qs` format for fast future loading:

```
qsave(hao, paste0(dir_name, "pbmc_hao_ref_up.qs"))
```

Query Datasets

Zheng

Expected Folder Structure

```
./data/Zheng/
├── Bcells/
│   ├── barcodes.tsv
│   ├── features.tsv (or genes.tsv)
│   ├── matrix.mtx
├── Monocytes_CD14/
│   ├── ...
├── NK/
│   ├── ...
├── TCD4_memory/
│   ├── ...
├── ... (one folder per cell type)
```

Each subfolder must contain:

- `matrix.mtx` : Sparse matrix of expression
- `barcodes.tsv` : Cell barcodes
- `features.tsv` or `genes.tsv` : Gene names

These should be **10X Genomics-formatted files.**

Useful Resources

To locate and verify the availability of the Zheng dataset, we consulted the following sources during our search:

- https://www.10xgenomics.com/datasets?configure[hitsPerPage]=50&configure[maxValuesPerFacet]=1000&query=frozen pbmc
- https://www.nature.com/articles/ncomms14049
- https://github.com/10XGenomics/single-cell-3prime-paper
- https://www.ebi.ac.uk/ena/browser/view/PRJNA318252
- https://www.ncbi.nlm.nih.gov/sra/SRX1723938[accn]

Despite extensive searching, we only managed to locate a usable version of the Zheng dataset after coming across it by chance in this GitHub repository: https://github.com/clinicalml/sc-foundation-eval/tree/main/data

Kotliarov

We downloaded this file: `H1_day0_scranNorm_adtbatchNorm_dist_clustered_TSNE_labels.rds` by following this link: https://figshare.com/articles/dataset/CITE-seq_protein-mRNA_single_cell_data_from_high_and_low_vaccine_responders_to_reproduce_Figs_4-6_and_associated_Extended_Data_Figs_/11349761

and put it in this folder: `./data/Kotliarov/` in order to be able to run the script: `Benchmarking_preprocess_Kotliarov.R`

In more detailed steps, this script:

- **Loads** pre-normalized single-cell data from the RDS file `H1_day0_scranNorm_adtbatchNorm_dist_clustered_TSNE_labels.rds`
- **Extracts**:
  - The raw gene expression matrix from `@raw.data`
  - Cell metadata from `@meta.data`
- **Creates** a `Seurat` object with consistent cell IDs between expression and metadata
- **Computes** mitochondrial content percentage per cell (`percent.mt`)
- **Performs QC filtering**: keeps cells with:
  - More than 200 genes
  - Fewer than 2,500 genes
  - Less than 10% mitochondrial gene content
- **Annotates clusters** from the original paper using the `K1` and `K3` cluster columns:
  - `K1`: maps coarse clusters (e.g., `C0`, `C1`) to high-level immune cell types
  - `K3`: maps refined subclusters (e.g., `C3.0.1`, `C4.0.2`) to specific populations like “B switched” or “T CD8 NKT-like”
- **Constructs a harmonized cell type label** by:
  - Merging `K1` and `K3` information
  - Reclassifying ambiguous or overlapping labels
  - Unifying labels into consistent categories (`T CD4`, `T CD8`, `Mono`, `NK`, `DC`, `B`, `T unconv`, `HSC`)
- **Removes two misannotated cells** manually by barcode ID
- **Normalizes** the expression data using `NormalizeData()`
- **Saves** the final `Seurat` object to `.qs` format

The resulting Seurat object contains:

- ~52,849 cells
- ~32,738 genes
- Mean ~748 genes per cell

This preprocessed object is saved to: `"./data/Kotliarov/kotliarov_pbmc.qs"`