

CNS Summer Students 2024

Intermediate Project Results

Fan Huang

Million-level cell embedding Visualization

Start from visualization of two Lung cell datasets

hubmap id
HBM948.GXMD.986
HBM975.WQQQ.853

Cell number: 202k

Embedding space: 201905 × 60286

Special settings:

1. Utilize the correct matrix layer:

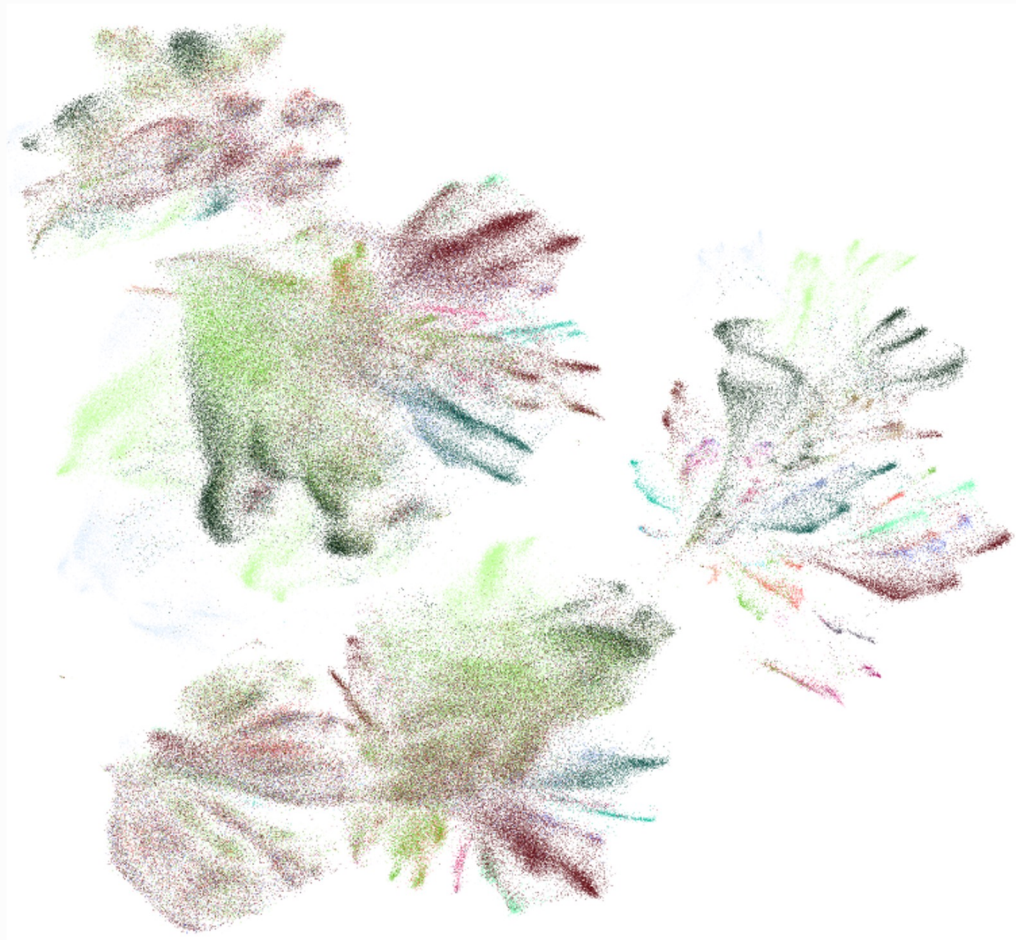
`'spliced_unspliced_sum'`

2. Adopt two steps of normalization before conducting the UMAP function.

`sc.pp.normalize_total(adata)`

`sc.pp.log1p(adata)`





Same normalization settings for ALL 10 lung datasets

cell type number is: 48


CL_label	
capillary endothelial cell	119435
type I pneumocyte	94120
type II pneumocyte	51142
effector memory CD8-positive, alpha-beta T cell	42042
alveolar type 1 fibroblast cell	25102
multi-ciliated epithelial cell:non-nasal	24346
CD4-positive helper T cell	20184
ionocyte	14056
endothelial cell of venule	11042
endothelial cell of artery	10208
lung pericyte	6133
alveolar capillary type 2 endothelial cell	4925
alveolar type 2 fibroblast cell	4324
respiratory basal cell:resting	3358
smooth muscle cell	3066
endothelial cell of lymphatic vessel:mature	2203
endothelial cell of venule:pulmonary	2065
monocyte	1983
CD1c-positive myeloid dendritic cell	1877
B cell	1624

Million-level cell embedding Visualization

Alternative Visualization Published work based on PCA + t-SNE

Transcriptomic diversity of cell types across the adult human brain

Kimberly Siletti,  Rebecca Hodge, Alejandro Mossi Albiach, Lijuan Hu, Ka Wai Lee, Peter Lönnerberg,

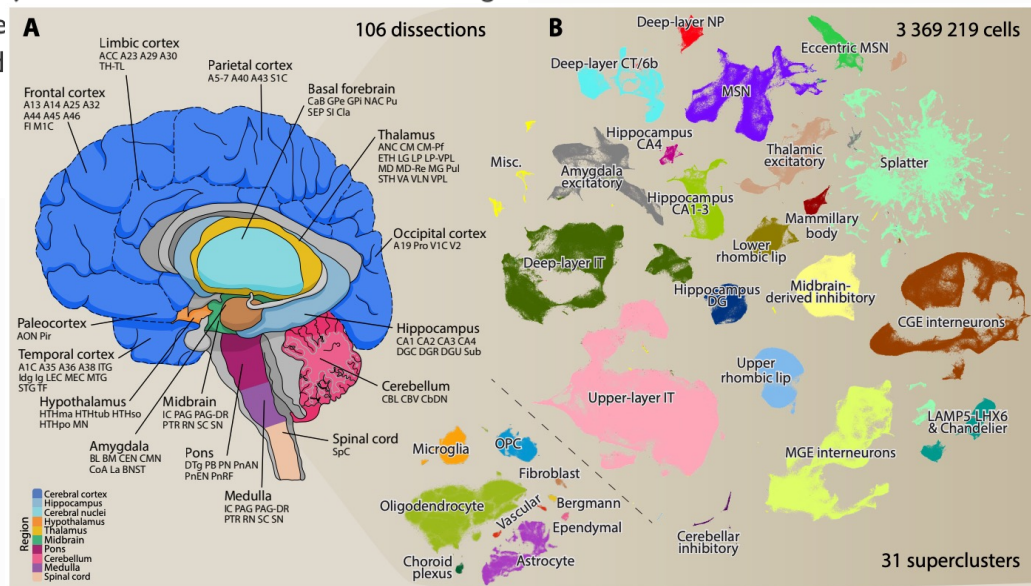
 Trygve Bakken, Song-Lin Ding, Michael Clark, Tamara Caspe
Julie Nyhus, Herman Tung, Anna Marie Yanny, Ernest Arenas, Ed

doi: <https://doi.org/10.1101/2022.10.12.511898>

Now published in *Science* doi: [10.1126/science.add7046](https://doi.org/10.1126/science.add7046)

Open-sourced github repository,
code for this figure:

<https://github.com/linnarsson-lab/adult-human-brain/blob/main/notebooks/Preprint/Figure1.ipynb>

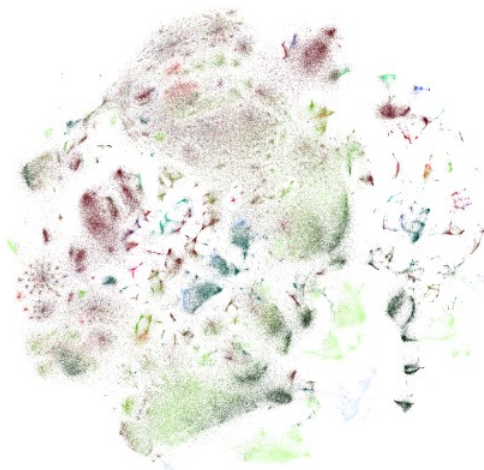


Million-level cell embedding Visualization

Alternative Visualization Published work based on PCA + t-SNE

We firstly adopt PCA for 200 dimensions, then use the t-SNE algorithm for final dimension reduction. We find the result is good, but the clusters are still not so perfectly separated.

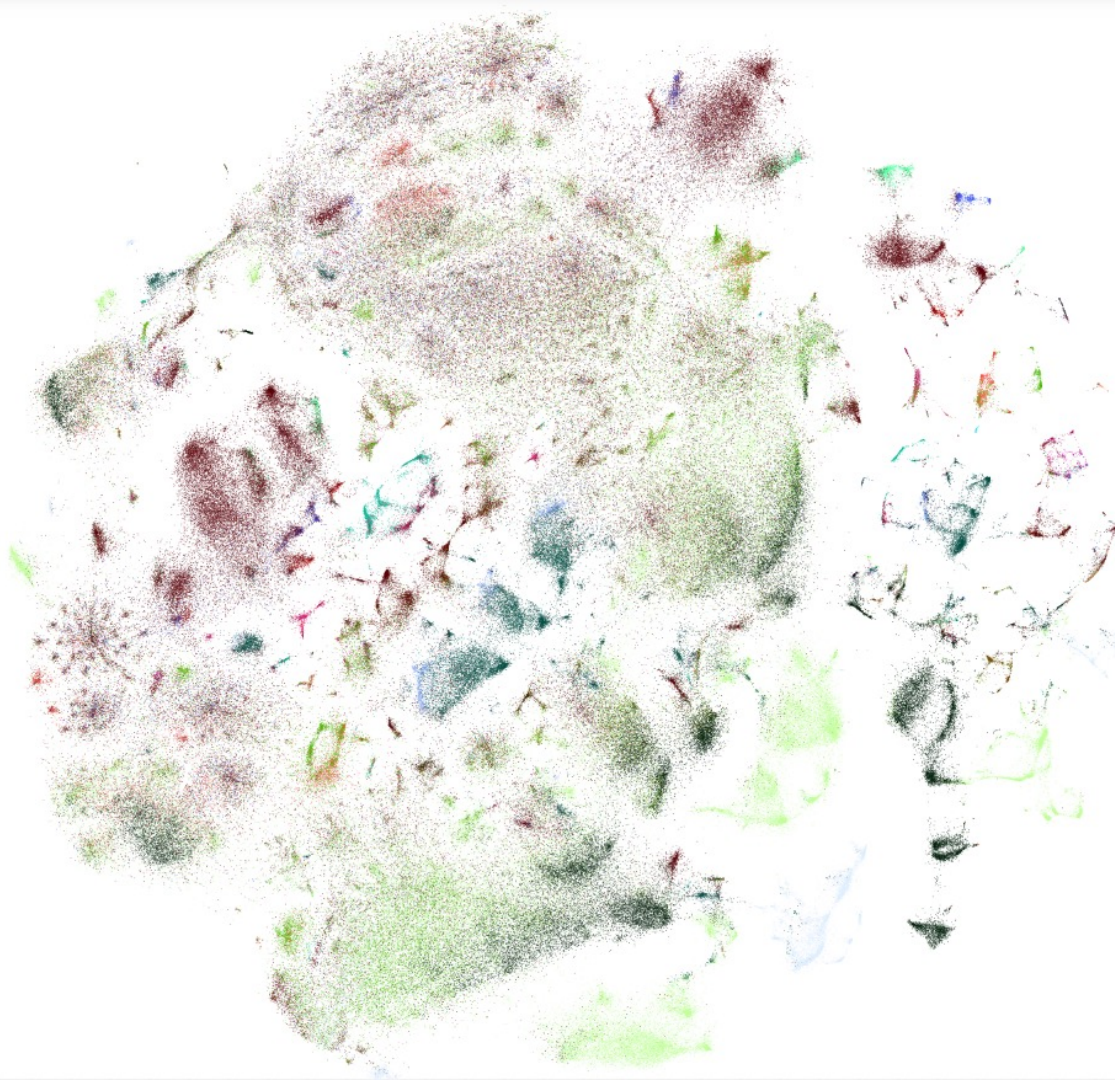
CL_label



- Alveolar Mφ proliferating
- B cell
- CD1c-positive myeloid dendritic cell
- CD4-positive helper T cell
- Interstitial Mφ perivascular
- Monocyte-derived Mφ
- Non-classical monocytes
- T cell:proliferating
- Transitional Club-AT2
- airway submucosal gland collecting duct epithelial cell
- alveolar capillary type 2 endothelial cell
- alveolar macrophage
- alveolar type 1 fibroblast cell
- alveolar type 2 fibroblast cell
- brush cell of tracheobronchial tree
- capillary endothelial cell

- club cell:nasal
- club cell:non-nasal
- deuterosomal cell
- effector memory CD8-positive, alpha-beta T cell
- endothelial cell of artery
- endothelial cell of lymphatic vessel:differentiating
- endothelial cell of lymphatic vessel:mature
- endothelial cell of venule
- endothelial cell of venule:pulmonary
- ionocyte
- lung pericyte
- mast cell
- monocyte
- mucus secreting cell of bronchus submucosal gland
- multi-ciliated epithelial cell:nasal
- multi-ciliated epithelial cell:non-nasal

- myofibroblast cell
- nasal mucosa goblet cell
- natural killer cell
- plasma cell
- plasmacytoid dendritic cell, human
- pulmonary interstitial fibroblast
- respiratory basal cell
- respiratory basal cell:resting
- serous secreting cell of bronchus submucosal gland
- serous secreting cell:activated
- serous secreting cell:nasal
- smooth muscle cell
- tracheobronchial goblet cell
- type I pneumocyte
- type II pneumocyte
- type II pneumocyte:proliferating



453k cell embedding vis:

- PCA, dimension to 200
- T-SNE, dimension to 2
(scanpy default T-SNE)
- Two-step normalizaiton

Million-level cell embedding Visualization

Alternative Visualization Published work based on PCA + t-SNE

We then try the openTSNE (paralleled version) to obtain more t-SNE algorithm parameter settings.

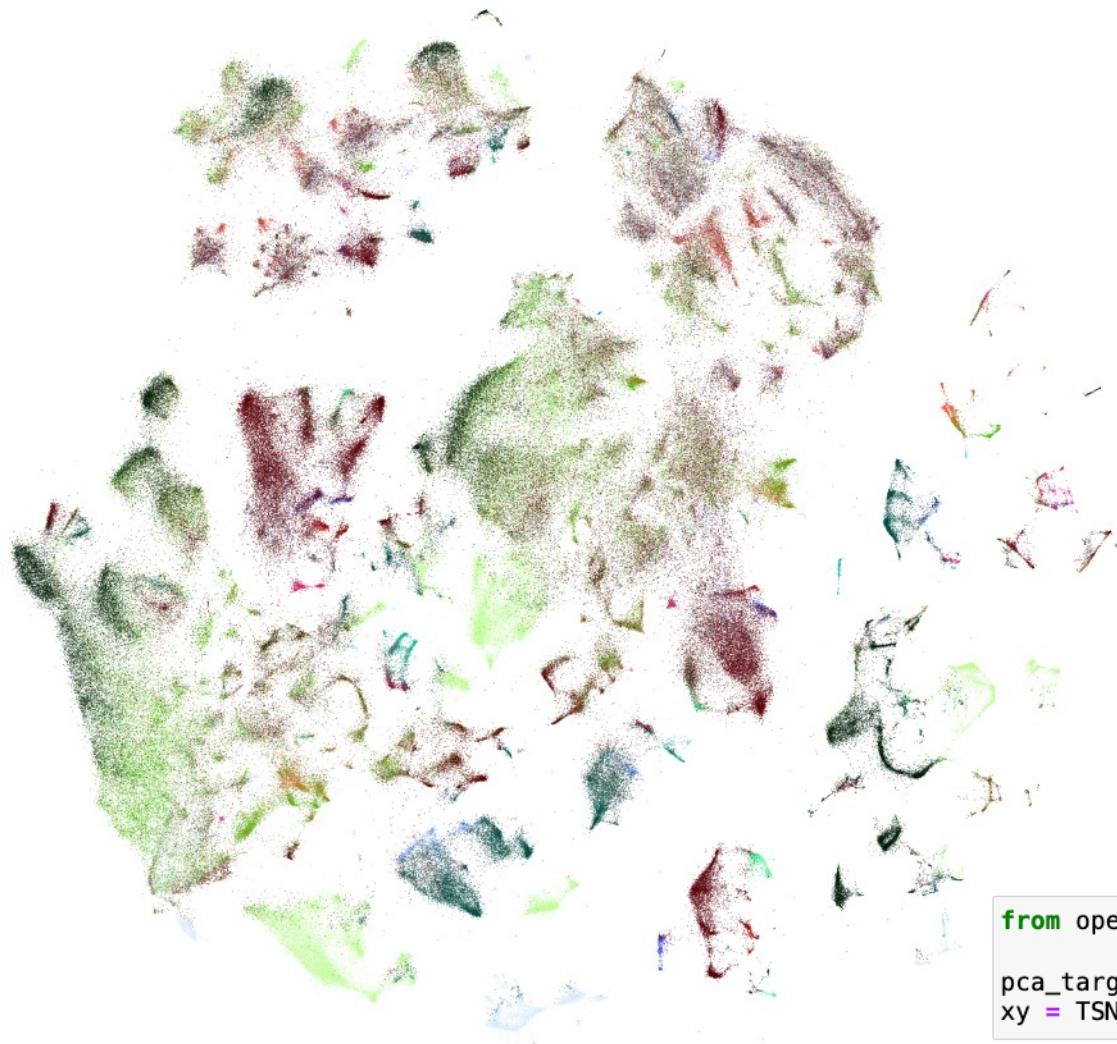
```
sc.tl.pca(adata_concat, n_comps=200)|
```

```
from openTSNE import TSNE

pca_target = adata_concat.obsm['X_pca']
xy = TSNE(perplexity=20, exaggeration=1.5).fit(pca_target)
```

```
from openTSNE import TSNE

pca_target = adata_concat.obsm['X_pca']
xy = TSNE(perplexity=20, exaggeration=2.8).fit(pca_target)
```

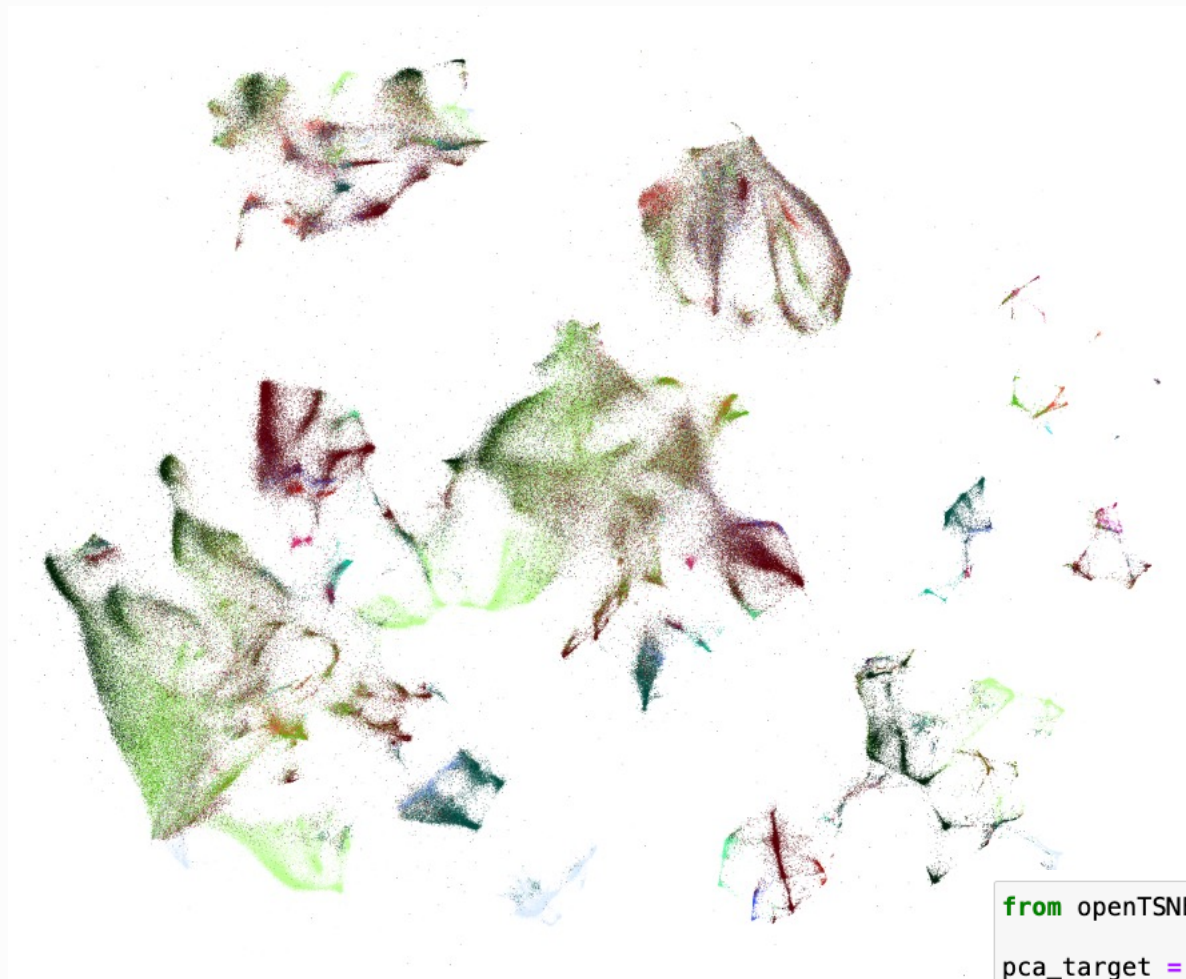
453k cell embedding vis:

- PCA, dimension to 200
- T-SNE, dimension to 2
(scanpy default T-SNE)
- Two-step normalizaiton

T-SNE: openTSNE package

```
from openTSNE import TSNE
```

```
pca_target = adata_concat.obsm['X_pca']  
xy = TSNE(perplexity=20, exaggeration=1.5).fit(pca_target)
```



453k cell embedding vis:

- PCA, dimension to 200
- T-SNE, dimension to 2 (scanpy default T-SNE)
- Two-step normalization

T-SNE: openTSNE package

```
from openTSNE import TSNE
```

```
pca_target = adata_concat.obsm['X_pca']
```

```
xy = TSNE(perplexity=20, exaggeration=2.8).fit(pca_target)
```

Million-level cell embedding Visualization


Alternative Visualization Published work based on PCA + t-SNE

We then try the openTSNE (paralleled version) to obtain more t-SNE algorithm parameter settings.

```
sc.tl.tsne(adata_concat, use_rep='X_pca')
```

```
from openTSNE import TSNE  
  
pca_target = adata_concat.obsm['X_pca']  
xy = TSNE(perplexity=20, exaggeration=1.5).fit(pca_target)
```

```
from openTSNE import TSNE  
  
pca_target = adata_concat.obsm['X_pca']  
xy = TSNE(perplexity=20, exaggeration=2.8).fit(pca_target)
```



As the **exaggeration** rate is higher, the clusters are more clearly separated in the visualization.

Million-level cell embedding Visualization

Alternative Visualization Published work based on PCA + t-SNE

To avoid the bias introduced by PCA re-processing dimension settings, we test 1000 dimension setting. Theoretically, the higher dimension number chosen should reflect better original embedding information.

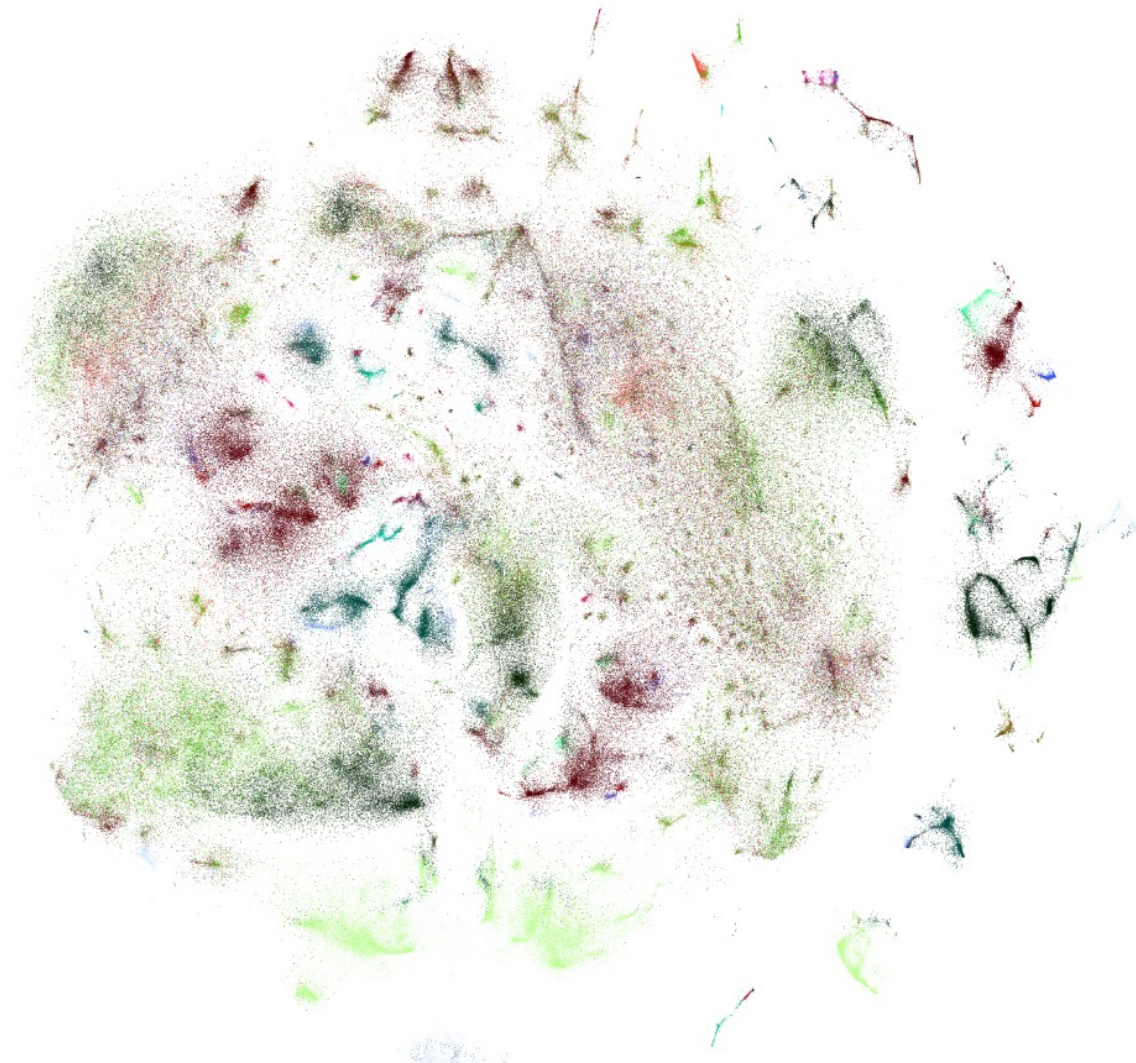
```
sc.tl.pca(adata_concat, n_comps=1000)
```

```
from openTSNE import TSNE

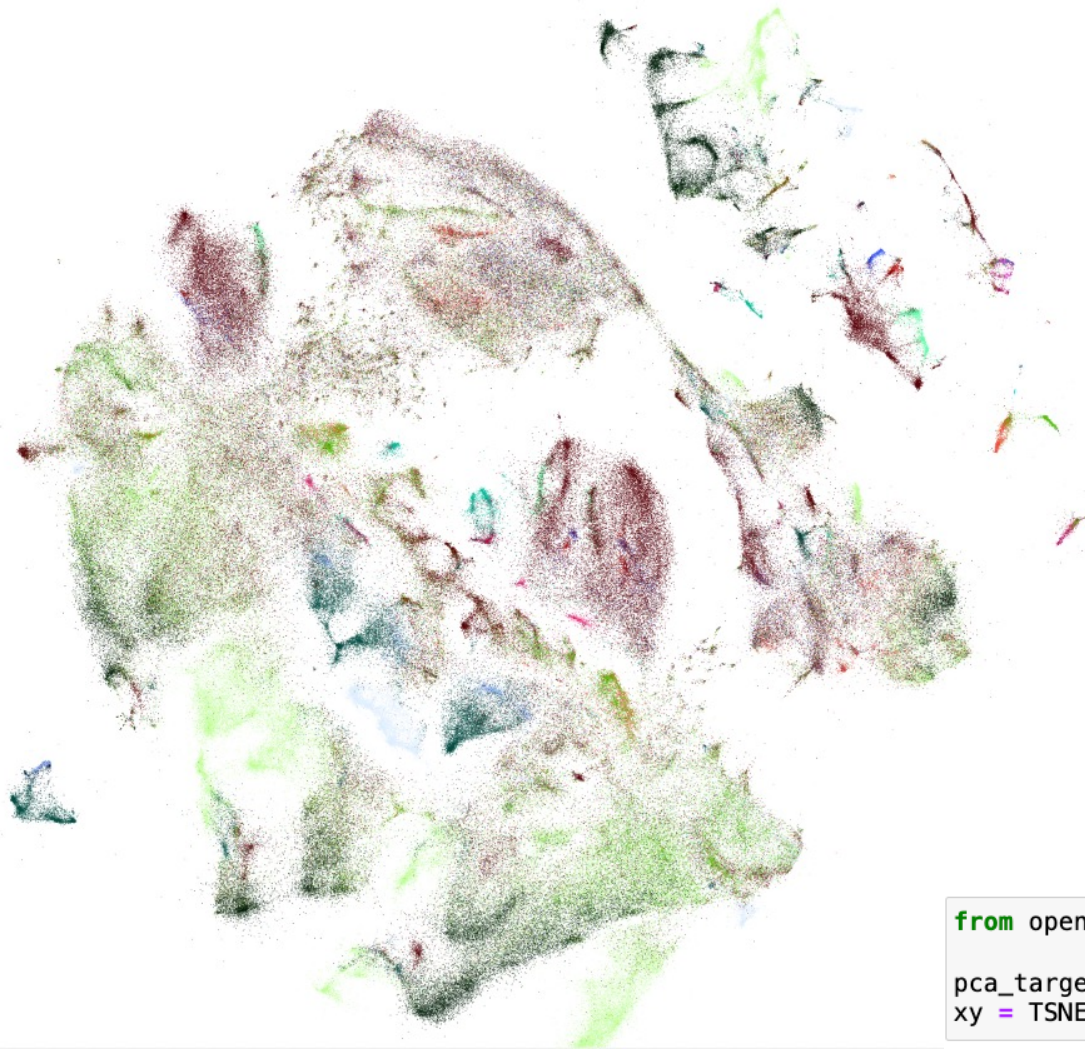
pca_target = adata_concat.obsm['X_pca']
xy = TSNE(perplexity=20, exaggeration=1.5).fit(pca_target)
```

```
from openTSNE import TSNE

pca_target = adata_concat.obsm['X_pca']
xy = TSNE(perplexity=20, exaggeration=2.8).fit(pca_target)
```



- 453k cell embedding vis:
- PCA, dimension to **1000**
 - T-SNE, dimension to 2
(scanpy default T-SNE)
 - Two-step normalizaiton



453k cell embedding vis:

- PCA, dimension to **1000**
- T-SNE, dimension to 2
(scanpy default T-SNE)
- Two-step normalizaiton

T-SNE: openTSNE package

```
from openTSNE import TSNE
```

```
pca_target = adata_concat.obsm['X_pca']  
xy = TSNE(perplexity=20, exaggeration=1.5).fit(pca_target)
```



453k cell embedding vis:

- PCA, dimension to **1000**
- T-SNE, dimension to 2
(scanpy default T-SNE)
- Two-step normalizaiton

T-SNE: openTSNE package

```
from openTSNE import TSNE
```

```
pca_target = adata_concat.obsm['X_pca']
```

```
xy = TSNE(perplexity=20, exaggeration=2.8).fit(pca_target)
```

Million-level cell embedding Visualization

Alternative Visualization Published work based on PCA + t-SNE

For 1000 PCA setting, to better understand the most prevalent cell types, we filtered out the cell categories that is less than 1k in the whole embedding. 453k → 448k

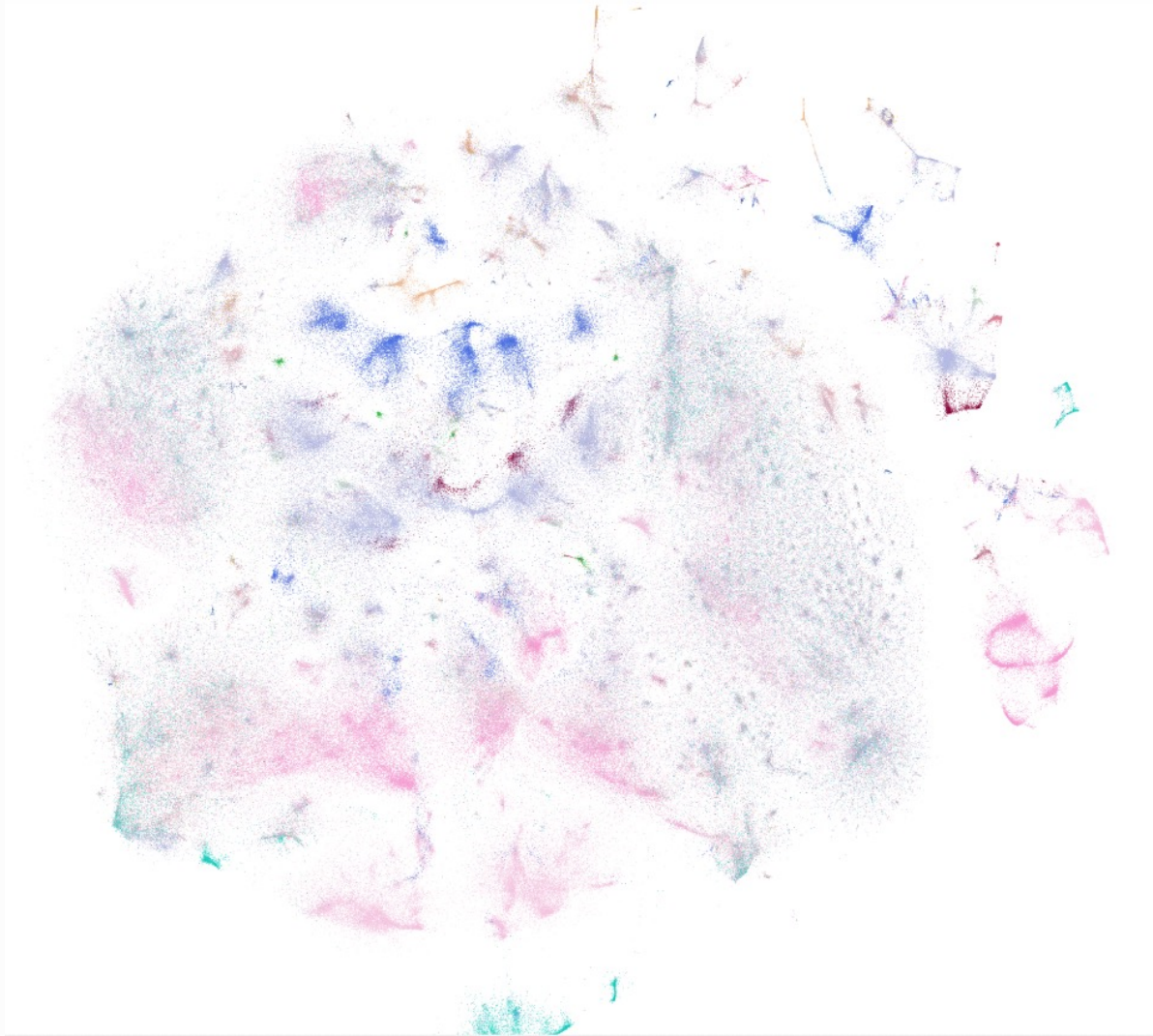
```
sc.tl.pca(adata_concat, n_comps=1000)
```

```
from openTSNE import TSNE

pca_target = adata_concat.obsm['X_pca']
xy = TSNE(perplexity=20, exaggeration=1.5).fit(pca_target)
```

```
from openTSNE import TSNE

pca_target = adata_concat.obsm['X_pca']
xy = TSNE(perplexity=20, exaggeration=2.8).fit(pca_target)
```



453k cell embedding vis:

- PCA, dimension to **1000**
- T-SNE, dimension to 2
(scanpy default T-SNE)
- Two-step normalizaiton
- Filtered cells that less than 1k of its type



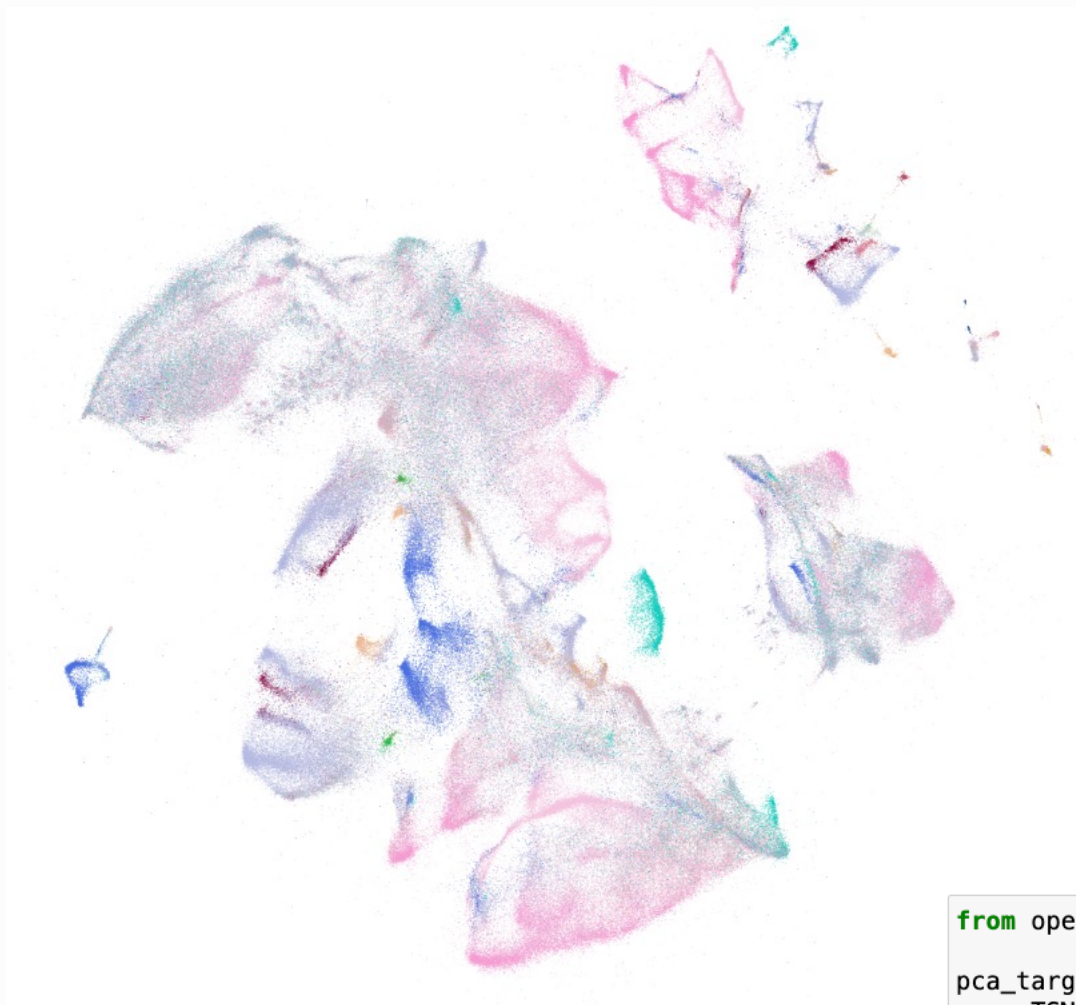
453k cell embedding vis:

- PCA, dimension to **1000**
- T-SNE, dimension to 2 (scanpy default T-SNE)
- Two-step normalization
- Filtered cells that less than 1k of its type

T-SNE: openTSNE package

```
from openTSNE import TSNE
```

```
pca_target = adata_concat.obsm['X_pca']  
xy = TSNE(perplexity=20, exaggeration=1.5).fit(pca_target)
```

453k cell embedding vis:

- PCA, dimension to **1000**
- T-SNE, dimension to 2 (scanpy default T-SNE)
- Two-step normalization
- Filtered cells that less than 1k of its type

T-SNE: openTSNE package

```
from openTSNE import TSNE
```

```
pca_target = adata_concat.obsm['X_pca']
```

```
xy = TSNE(perplexity=20, exaggeration=2.8).fit(pca_target)
```