scanpy.pp.neighbors

scanpy.pp.neighbors(*adata*, *n_neighbors=15*, *n_pcs=None*, *use_rep=None*, *knn=True*, *random_state=0*, *method='umap'*, *metric='euclidean'*, *metric_kwds=mappingproxy({})*, *key_added=None*, *copy=False*)

Compute a neighborhood graph of observations [McInnes18].

The neighbor search efficiency of this heavily relies on UMAP [McInnes18], which also provides a method for estimating connectivities of data points - the connectivity of the manifold (`method=='umap'`). If `method=='gauss'`, connectivities are computed according to [Coifman05], in the adaption of [Haghverdi16].

Parameters:

adata : `AnnData`

Annotated data matrix.

n_neighbors : `int` (default: `15`)

The size of local neighborhood (in terms of number of neighboring data points) used for manifold approximation. Larger values result in more global views of the manifold, while smaller values result in more local data being preserved. In general values should be in the range 2 to 100. If `knn` is `True`, number of nearest neighbors to be searched. If `knn` is `False`, a Gaussian kernel width is set to the distance of the `n_neighbors` neighbor.

n_pcs : `Optional` [`int`] (default: `None`)

Use this many PCs. If `n_pcs==0` use `.X` if `use_rep is None`.

use_rep : `Optional` [`str`] (default: `None`)

Use the indicated representation. `'X'` or any key for `.obsm` is valid. If `None`, the representation is chosen automatically: For `.n_vars` < 50, `.X` is used, otherwise ‘X_pca’ is used. If ‘X_pca’ is not present, it’s computed with default parameters.

knn : `bool` (default: `True`)

If `True`, use a hard threshold to restrict the number of neighbors to `n_neighbors`, that is, consider a knn graph. Otherwise, use a Gaussian Kernel to assign low weights to neighbors more distant than the `n_neighbors` nearest neighbor.

random_state : `Union` [`None` , `int` , `RandomState`] (default: `0`)

A numpy random seed.

method : `Optional` [`Literal` [`'umap'` , `'gauss'` , `'rapids'`]] (default: `'umap'`)

Use ‘umap’ [McInnes18] or ‘gauss’ (Gauss kernel following [Coifman05] with adaptive width [Haghverdi16]) for computing connectivities. Use ‘rapids’ for the RAPIDS implementation of UMAP (experimental, GPU only).

metric : `Union` [`Literal` [`'cityblock'` , `'cosine'` , `'euclidean'` , `'l1'` , `'l2'` , `'manhattan'`], `Literal` [`'braycurtis'` , `'canberra'` , `'chebyshev'` , `'correlation'` , `'dice'` , `'hamming'` , `'jaccard'` , `'kulsinski'` , `'mahalanobis'` , `'minkowski'` , `'rogerstanimoto'` , `'russellrao'` , `'seuclidean'` , `'sokalmichener'` , `'sokalsneath'` , `'sqeuclidean'` , `'yule'`], `Callable` [[`ndarray` , `ndarray`], `float`]] (default: `'euclidean'`)

A known metric’s name or a callable that returns a distance.

metric_kwds : `Mapping` [`str` , `Any`] (default: `mappingproxy({})`)

Options for the metric.

key_added : `Optional` [`str`] (default: `None`)

If not specified, the neighbors data is stored in .uns[‘neighbors’], distances and connectivities are stored in .obsp[‘distances’] and .obsp[‘connectivities’] respectively. If specified, the neighbors data is added to .uns[key_added], distances are stored in .obsp[key_added+’_distances’] and connectivities in .obsp[key_added+’_connectivities’].

copy : `bool` (default: `False`)

Return a copy instead of writing to adata.

Return type: `Optional` [`AnnData`]

Returns:

: Depending on `copy` , updates or returns `adata` with the following:

See `key_added` parameter description for the storage path of connectivities and distances.

connectivities : sparse matrix of dtype `float32` .

Weighted adjacency matrix of the neighborhood graph of data points. Weights should be interpreted as connectivities.

distances : sparse matrix of dtype `float32`.

Instead of decaying weights, this stores distances for each pair of neighbors.

Notes

If `method='umap'`, it's highly recommended to install pynndescent `pip install pynndescent`. Installing `pynndescent` can significantly increase performance, and in later versions it will become a hard dependency.