northstar Documentation

Release "0.6.0"

Fabio Zanini

Jul 25, 2022

Contents

Single cell type annotation guided by cell atlases, with freedom to be queer.

CHAPTER 1

Brief description

northstar is a Python package to identify cell types within single cell transcriptomics datasets. It uses one or more cell atlases as a baseline and assigns each cell of your dataset to either a known cell type from the atlas(es) or to a novel cluster. northstar's superpower is that it learn from big data (atlases) but still allows queer cells to make their own cluster if they want to.

northstar was heavily developed during Pride Month.

CHAPTER 2

Installation

northstar is a pure Python package, so you can install it using pip:

```
pip install northstar
```

To automatically download and use our online atlas collection at https://northstaratlas.github.io/atlas_averages/, you will need to call:

```
pip install 'northstar[atlas-fetcher]'
```

2.1 Dependencies

- *numpy*
- *scipy*
- *pandas*
- *scikit-learn*
- *anndata*
- *python-igraph>=0.8.0*
- *leidenalg>=0.8.0*

It is recommended that you install python-igraph and leidenalg using *pip*. However, any installation (e.g. conda) that includes recent enough versions of both packages will work.

Optional deps to use our online atlases:

- *requests*
- *loompy*
- *scanpy* (reduces memory usage)
- *pynndescent* (only useful if you also use scanpy)

CHAPTER 3

Usage example

Also see our *Tutorial*.

```
import anndata
import northstar

# Choose an atlas
atlas_name = 'Darmanis_2015'

# Get an AnnData object with the new data to be annotated
new_dataset = anndata.read_loom('...')
# or any other format

# Initialize northstar classes
model = northstar.Averages(
        atlas=atlas,
        )

# Run the classifier
model.fit(new_dataset)

# Get the cluster memberships for the new cells
membership = model.membership
```

CHAPTER 4

Citation

If you use this software please cite the following paper:

Fabio Zanini*, Bojk A. Berghuis*, Robert C. Jones, Benedetta Nicolis di Robilant, Rachel Yuan Nong, Jeffrey Norton, Michael F. Clarke, Stephen R. Quake. **Northstar enables automatic classification of known and novel cell types from tumor samples.** Scientific Reports 10, Article number: 15251 (2020), DOI: https://doi.org/10.1038/s41598-020-71805-1

CHAPTER 5

License

northstar is released under the MIT license.

NOTE: The module *leidenalg*, which is a dependency of *northstar*, is released under the GLP3 license. You agree with those licensing terms if you use leidenalg within northstar.

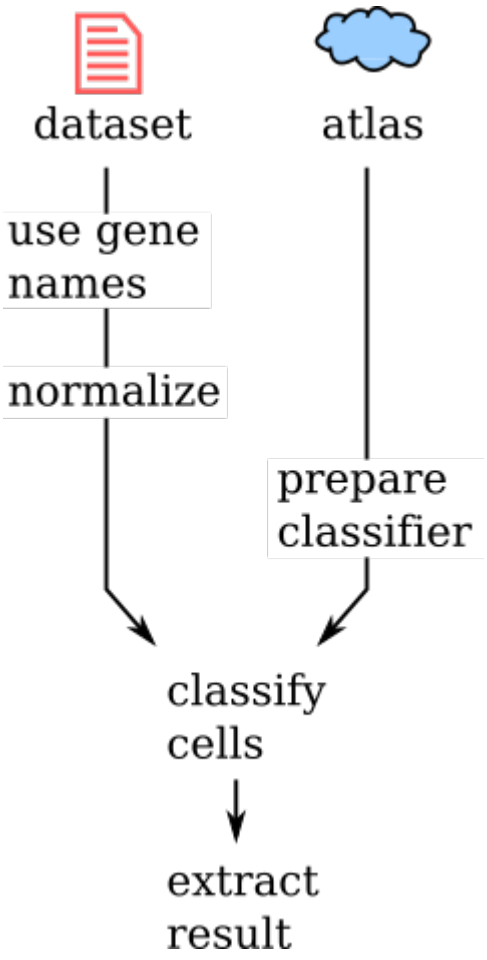
CHAPTER 6

Contents

6.1 Tutorial

Thank you for looking into *northstar*! This tutorial guides you through a typical use of the package to annotate your single cell dataset based on one or more cell atlases. At the end of the tutorial, you can look at some *Examples* and at the detailed *API* documentation.

6.1.1 Flowchart



6.1.2 Short version

```
import anndata
import northstar

# Load new dataset
dataset = anndata.read_loom('GBM_data.loom', sparse=False)

# Set gene names (if needed)
dataset.var_names = dataset.var['GeneName']

# Normalize (if needed)
dataset.X = 1e6 * (dataset.X.T / dataset.X.sum(axis=1)).T

# Choose atlas
atlas = 'Darmanis_2015_nofetal'

# Prepare classifier
model = northstar.Subsample(
    atlas=atlas,
    )
```

(continues on next page)

(continued from previous page)

```
# Run classifier
model.fit()

# Extract result
cell_types = model.membership
```

6.1.3 Intro: atlas landmarks

To transfer cell types from an atlas, you need an average or small subsample of cells for each cell type within the atlas. We call these **atlas landmarks**. To keep things simple, in this tutorial we use precomputed landmarks, in particlar the brain atlas from Darmanis_2015. Custom atlases are supported (see below).

6.1.4 Prepare your single cell dataset

Then we need to prepare the new dataset to annotate. For this tutorial, we will use the glioblastoma data from Darmanis et al. (2017) which is made available for this tutorial as a loom file at this address: download that file into the current folder with the name *GBM_data.loom*.

After the download is done, read it with anndata:

```
dataset = anndata.read_loom('GBM_data.loom', sparse=False)
```

Let's make sure gene names are used as columns of the AnnData table:

```
dataset.var_names = dataset.var['GeneName']
```

Note: *northstar* will take the intersection of your features names and the atlas features to assign cell types. Most atlases use gene names instead of EnsemblIDs or other names, so make sure you do the same. Remember human genes all ALL CAPS but mouse genes are Capitalized only.

We need to normalize the dataset:

- **log**: *northstar* will take the logarithm of the counts when necessary. If your data is already logged, undo the transformation (by exponentiating *and* subtracting any pseudocounts) before using northstar.
- **normalization**: *northstar* will look for overdispersed features in the new dataset prior to normalization. It is therefore highly recommended to normalize your new data (e.g. by counts per million reads or counts per 10,000 reads).

```
dataset.X = 1e6 * (dataset.X.T / dataset.X.sum(axis=1)).T
```

Note: Forgetting to format the data according to the two rules above can lead to gross misclassification.

6.1.5 Choose an annotated atlas

You can choose one of the available atlas landmarks by name, e.g. *Darmanis_2015* is an early atlas of the human brain, and *Darmanis_2015_nofetal* excludes fetal cells (our tutorial glioblastoma data are all adult tumors).

Optional: exploring atlas landmarks

northstar provides a class to explore our precompiled landmarks:

```
import northstar
af = northstar.AtlasFetcher()
```

To list available atlases, just type:

```
af.list_atlases()
```

and to download one of them, for instance:

```
myatlas = af.fetch_atlas('Darmanis_2015_nofetal', kind='subsample')
```

Note: If you just use the name (string) of a precompiled atlas landmark in the classifier (see below), the landmark will be automatically downloaded for you.

Alternative: custom atlas

You can also use a custom atlas. In that case, the atlas should be in an *AnnData* object (with rows as cells, genes as columns):

- If you plan to use the *Subsample* class, the *AnnData* must have an *obs* column called

CellType that describes for each cell its cell type. - If you plan to use the *Averages* class, the *AnnData* must have an *obs* column called *NumberOfCells* that is used to weight each cell type in the PCA. A value of 20 for all cell types is typical.

northstar provides a function to subsample an existing annotated dataset to small cell numbers within each cell type, ready for further use with the *Subsample* class. You data must be in an *AnnData* object. You can call it by:

```
import northstar
myatlas = northstar.subsample_atlas(mydataset)
```

The default metadata column used for subsampling each cell type evenly is *CellType*. If your dataset uses a different column, you can just set the *cell_type_column* argument in this function.

Remember that the metadata column *CellType* is required anyway to use northstar. So you should set your cell type information into that column before or after subsampling:

```
myatlas.obs['CellType'] = myatlas.obs[my_other_column]
```

6.1.6 Prepare the classifier

Let's use the *Subsample* class:

```
import northstar

model = northstar.Subsample(
    atlas='Darmanis_2015_nofetal',
    )
```

6.1.7 Classify your cells

This is where the actual computations happen:

```
model.fit()
```

Advanced: understanding the single steps

If you are curious about the steps within *northstar*, you can call in your Jupyter notebook or ipython console:

```
model.fit??
```

and check out the steps one by one. Most users will not need this.

6.1.8 Extract the result

The result of the cell type assignment can be extracted by the following command:

```
cell_types = model.membership
```

This is a numpy array with the same length and order as your cells.

Note: You can also run the classifier and exctract the result all at once using *model.fit_transform()*.

6.1.9 Downstream analysis

northstar's main job is done with the cell type classification. Here some common downstream steps.

Optional: embbedding

Embeddings in two dimensions are useful to characterize single cell data. Northstar merges the atlas subsample/averages and the new dataset into the same PC space, and it's easy to get an embedding of your data "into" the atlas:

```
embedding = model.embed(method='umap')
```

Available embeddings are *tsne*, *umap*, and *pca*.

Optional: closest atlas cell type

Sometimes you get novel clusters that do not match any atlas cell type. To start identifying those clusters, you can ask *northstar* what known atlas cell type they are most similar to. Here's the code to do that:

```
closest_cell_types = model.estimate_closest_atlas_cell_type()
```

The output is a *pandas.Series* with the novel clusters as index and the closest atlas cell types as values.

Optional: custom data harmonization

northstar divides the cell classification task in two steps:

1. Create a similarity graph that contains both the atlas and the new data
2. Cluster that graph with awareness of the atlas annotations.

For advanced users, it is possible to use a custom approach to step 1 and only use *northstar* for the atlas-aware clustering step 2. In this scenario, the similarity graph might be constructed using external data harmonization algorithms, such as scVI, BBKNN, Seurat3, or whatever else.

northstar offers the class *ClusterWithAnnotations* for this purpose:

```
import northstar
model = northstar.ClusterWithAnnotations(graph, cell_types_atlas)
cell_types_newcells = model.fit_transform()
```

where *graph* must be a *igraph.Graph* instance from python-igraph or a dense or sparse boolean square matrix representing the adjacency matrix of the graph (i.e. it has nonzeros on on *graph[i , j]* if cell *i* and cell *j* are neighbors).

6.1.10 Next steps

Browse the *Examples* and *API* pages for more information.

6.1.11 Conclusion

We hope *northstar* helps you understand your tissue sample and do not hesitate to open an issue on github if you have trouble. If *northstar* was useful for a publication, please consider citing us on bioRxiv.

6.2 Examples

Northstar has two main classes to use averages or subsamples of cell atlases:

- Averages with precomputed atlas
- Subsample with precomputed atlas

You can use a custom atlas:

- Averages with custom atlas
- Subsample with custom atlas

You can also harmonize your atlas and target dataset (to be annotated) with another tool and then use northstar for clustering only. The advantage is that northstar's clustering algorithm is aware of the atlas annotations, therefore it is guaranteed to neither split not merge atlas cell types:

- External data harmonization

You can also use northstar just as an API interface to our precompiled list of annotated atlases. This can be used to download averages and subsamples (we call them **atlas landmarks**) and use them to do whatever you want (e.g. classify using another tool, harmonize, look up marker genes, etc):

- Fetch a precompiled atlas landmark

Another use of northstar is for its ability to compress large datasets into cell type averages or subsamples. We call this operation **creating an atlas landmark**:

- Compress an atlas

Finally, you might want to play with northstar's internals and take inspiration to build your very own classifer, data harmonizer, clustering algorithm, feature selection tool, or whatever else. This is probably only interesting for **advanced users**:

- Tinkering with northstar's internals

6.3 API

6.3.1 northstar.Averages

6.3.2 northstar.Subsample

6.3.3 northstar.ClusterWithAnnotations

6.3.4 northstar.AtlasFetcher

6.3.5 northstar.average_atlas

6.3.6 northstar.subsample_atlas

CHAPTER 7

Indices and tables

- genindex
- modindex
- search