Hypercluster

Release 0.0.2

Ruggleslab

Jan 24, 2020

CONTENTS

CHAPTER

ONE

HYPERCLUSTER PACKAGE

CHAPTER

TWO

HYPERCLUSTER SNAKEMAKE PIPELINE

2.1 Line-by-line explanation of config.yml

Table 1: Explanation for config.yml

config.yml parameter	Explanation	Example from scRNA-seq workflow
`input_data_folder`	Path to folder in which input data can be found. No / at the end.	`/input_data`
`input_data_files`	List of prefixes of data files. Exclude extension, .csv, .tsv and .txt allowed.	`['input_data1', 'input_data2']`
`gold_standard_file`	File name of gold_standard_file. Must have same pandas.read_csv kwargs as the corresponding input file. Must be in input_data_folder.	`{'input_data': 'gold_standard_file.txt'}`
`read_csv_kwargs`	Per input data file, keyword args to put into pandas.read_csv. **If specifying multiindex, also put the same in output_kwargs['labels']**	`{'test_input': {'index_col':[0]}}`
`output_folder`	Path to folder in which results will be written. No / at the end.	`/hypercluster_results`
`intermediates_folder`	Name of the folder within the output_folder to put intermediate results, such as labels and evaluations per condition. No need to change this usually.	`clustering_intermediates`
`clustering_results`	Name of the folder within the output_folder to put final results. No need to change this usually.	`clustering`

**Note: Formatting of lists and dictionaries can be in python syntax (like above) or yaml syntax, or a mixture, like below. **

2.2 config.yml example from scRNA-seq workflow

```yaml
input_data_folder: '.'
input_data_files:
  - sc_data
gold_standards:
  test_input: 'gold_standard.csv'
read_csv_kwargs:
  test_input: {'index_col':[0]}

output_folder: 'results'
intermediates_folder: 'clustering_intermediates'
clustering_results: 'clustering'

clusterer_kwargs: {}
generate_parameters_addtl_kwargs: {}

evaluations:
  - silhouette_score
  - calinski_harabasz_score
  - davies_bouldin_score
  - number_clustered
  - smallest_largest_clusters_ratio
  - smallest_cluster_ratio
eval_kwargs: {}

metric_to_choose_best: silhouette_score
metric_to_compare_labels: adjusted_rand_score
compare_samples: true

output_kwargs:
  evaluations:
    index_col: [0]
  labels:
    index_col: [0]
heatmap_kwargs: {}

optimization_parameters:
  HDBSCAN:
    min_cluster_size: &id002
    - 2
    - 3
    - 4
    - 5
  KMeans:
    n_clusters: &id001
    - 5
    - 6
    - 7
  MiniBatchKMeans:
    n_clusters: *id001
  OPTICS:
    min_samples: *id002
```

CHAPTER

THREE

INDICES AND TABLES

- genindex
- modindex
- search

CHAPTER

FOUR

INSTALLATION AND LOGISTICS

4.1 Installation

Available via pip:

```
pip install hypercluster
```

Or bioconda:

```
conda install hypercluster
 # or
conda install -c conda-forge -c bioconda hypercluster
```

If you are having problems installing with conda, try changing your channel priority. Priority of conda-forge > bioconda > defaults is recommended.

To check channel priority: `conda config --get channels`

It should look like:

```
--add channels 'defaults'   # lowest priority
--add channels 'bioconda'
--add channels 'conda-forge'   # highest priority
```

If it doesn't look like that, try:

```
conda config --add channels bioconda
conda config --add channels conda-forge
```

4.2 Quick reference for clustering and evaluation

Table 1: Clustering algorithms

Clusterer	Type
KMeans/MiniBatch KMeans	Partitioner
Affinity Propagation	Partitioner
Mean Shift	Partitioner
DBSCAN	Clusterer
OPTICS	Clusterer
Birch	Partitioner
OPTICS	Clusterer
HDBSCAN	Clusterer
NMF	Partitioner

Table 2: Evaluations

Metric	Type
adjusted_rand_score	Needs ground truth
adjusted_mutual_info_score	Needs ground truth
homogeneity_score	Needs ground truth
completeness_score	Needs ground truth
fowlkes_mallows_score	Needs ground truth
mutual_info_score	Needs ground truth
v_measure_score	Needs ground truth
silhouette_score	Inherent metric
calinski_harabasz_score	Inherent metric
davies_bouldin_score	Inherent metric
smallest_largest_clusters_ratio	Inherent metric
number_of_clusters	Inherent metric
smallest_cluster_size	Inherent metric
largest_cluster_size	Inherent metric

4.3 Quickstart and examples

4.3.1 With snakemake:

```
snakemake -s hypercluster.smk --configfile config.yml --config input_data_files=test_
↪data input_data_folder=.
```

4.3.2 With python:

```
import pandas as pd
from sklearn.datasets import make_blobs
import hypercluster

data, labels = make_blobs()
data = pd.DataFrame(data)
```

(continues on next page)

(continued from previous page)

```
labels = pd.Series(labels, index=data.index, name='labels')

# With a single clustering algorithm
clusterer = hypercluster.AutoClusterer()
clusterer.fit(data).evaluate(
  methods = hypercluster.constants.need_ground_truth+hypercluster.constants.inherent_
↪metrics,
  gold_standard = labels
  )

clusterer.visualize_evaluations()

# With a range of algorithms

clusterer = hypercluster.MultiAutoClusterer()
clusterer.fit(data).evaluate(
  methods = hypercluster.constants.need_ground_truth+hypercluster.constants.inherent_
↪metrics,
  gold_standard = labels
  )

clusterer.visualize_evaluations()
```

Example work flows for both python and snakemake are here

Source code is available here