dask-ndmeasure Documentation

Release 0.1.0+0.g1981658.dirty

John Kirkham

Aug 07, 2017

Contents

Contents:

CHAPTER 1

dask-ndmeasure

A library for computing N-D measurements on labeled Dask Arrays

- Free software: BSD 3-Clause
- Documentation: https://dask-ndmeasure.readthedocs.io.

1.1 Features

- TODO

1.2 Credits

This package was created with Cookiecutter and the dask-image/dask-image-cookiecutter project template.

CHAPTER 2

Installation

2.1 Stable release

To install dask-ndmeasure, run this command in your terminal:

```
$ pip install dask-ndmeasure
```

This is the preferred method to install dask-ndmeasure, as it will always install the most recent stable release.

If you don't have pip installed, this Python installation guide can guide you through the process.

2.2 From sources

The sources for dask-ndmeasure can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/dask-image/dask-ndmeasure
```

Or download the tarball:

```
$ curl  -OL https://github.com/dask-image/dask-ndmeasure/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```

CHAPTER 3

Usage

To use dask-ndmeasure in a project:

```
import dask_ndmeasure
```

CHAPTER 4

API

4.1 dask_ndmeasure package

dask_ndmeasure.**center_of_mass**(*input*, *labels=None*, *index=None*)

Find the center of mass over an image at specified subregions.

Parameters

- **input** (*ndarray*) – N-D image data
- **labels** (*ndarray, optional*) – Image features noted by integers. If None (default), all values.
- **index** (*int or sequence of ints, optional*) – Labels to include in output. If None (default), all values where non-zero `labels` are used.

  The `index` argument only works when `labels` is specified.

Returns **center_of_mass** – Coordinates of centers-of-mass of `input` over the `index` selected regions from `labels`.

Return type ndarray

dask_ndmeasure.**extrema**(*input*, *labels=None*, *index=None*)

Find the min and max with positions over an image at specified subregions.

Parameters

- **input** (*ndarray*) – N-D image data
- **labels** (*ndarray, optional*) – Image features noted by integers. If None (default), all values.
- **index** (*int or sequence of ints, optional*) – Labels to include in output. If None (default), all values where non-zero `labels` are used.

  The `index` argument only works when `labels` is specified.

Returns **minimums, maximums, min_positions, max_positions** – Values and coordinates of minimums and maximums in each feature.

Return type tuple of ndarrays

dask_ndmeasure.**histogram**(*input*, *min*, *max*, *bins*, *labels=None*, *index=None*)

Find the histogram over an image at specified subregions.

Histogram calculates the frequency of values in an array within bins determined by `min`, `max`, and `bins`. The `labels` and `index` keywords can limit the scope of the histogram to specified sub-regions within the array.

Parameters

- **input** (*ndarray*) – N-D image data
- **min** (*int*) – Minimum value of range of histogram bins.
- **max** (*int*) – Maximum value of range of histogram bins.
- **bins** (*int*) – Number of bins.
- **labels** (*ndarray, optional*) – Image features noted by integers. If None (default), all values.
- **index** (*int or sequence of ints, optional*) – Labels to include in output. If None (default), all values where non-zero `labels` are used.

  The `index` argument only works when `labels` is specified.

Returns **histogram** – Histogram of `input` over the `index` selected regions from `labels`.

Return type ndarray

dask_ndmeasure.**label**(*input*, *structure=None*)

Label features in an array.

Parameters

- **input** (*ndarray*) – An array-like object to be labeled. Any non-zero values in `input` are counted as features and zero values are considered the background.
- **structure** (*ndarray, optional*) – A structuring element that defines feature connections. `structure` must be symmetric. If no structuring element is provided, one is automatically generated with a squared connectivity equal to one. That is, for a 2-D `input` array, the default structuring element is:

```
[[0,1,0],
 [1,1,1],
 [0,1,0]]
```

Returns

- **label** (*ndarray or int*) – An integer ndarray where each unique feature in `input` has a unique label in the returned array.
- **num_features** (*int*) – How many objects were found.

dask_ndmeasure.**labeled_comprehension**(*input*, *labels*, *index*, *func*, *out_dtype*, *default*, *pass_positions=False*)

Compute a function over an image at specified subregions.

Roughly equivalent to [func(input[labels == i]) for i in index].

Sequentially applies an arbitrary function (that works on array_like input) to subsets of an n-D image array specified by `labels` and `index`. The option exists to provide the function with positional parameters as the second argument.

Parameters

- **input** (*ndarray*) – N-D image data
- **labels** (*ndarray, optional*) – Image features noted by integers. If None (default), all values.
- **index** (*int or sequence of ints, optional*) – Labels to include in output. If None (default), all values where non-zero `labels` are used.

  The `index` argument only works when `labels` is specified.
- **func** (*callable*) – Python function to apply to `labels` from `input`.
- **out_dtype** (*dtype*) – Dtype to use for `result`.
- **default** (*int, float or None*) – Default return value when a element of `index` does not exist in `labels`.
- **pass_positions** (*bool, optional*) – If True, pass linear indices to `func` as a second argument. Default is False.

Returns result – Result of applying `func` on `input` over the `index` selected regions from `labels`.

Return type ndarray

dask_ndmeasure.**maximum**(*input*, *labels=None*, *index=None*)

Find the maxima over an image at specified subregions.

Parameters

- **input** (*ndarray*) – N-D image data
- **labels** (*ndarray, optional*) – Image features noted by integers. If None (default), all values.
- **index** (*int or sequence of ints, optional*) – Labels to include in output. If None (default), all values where non-zero `labels` are used.

  The `index` argument only works when `labels` is specified.

Returns maxima – Maxima of `input` over the `index` selected regions from `labels`.

Return type ndarray

dask_ndmeasure.**maximum_position**(*input*, *labels=None*, *index=None*)

Find the positions of maxima over an image at specified subregions.

For each region specified by `labels`, the position of the maximum value of `input` within the region is returned.

Parameters

- **input** (*ndarray*) – N-D image data
- **labels** (*ndarray, optional*) – Image features noted by integers. If None (default), all values.
- **index** (*int or sequence of ints, optional*) – Labels to include in output. If None (default), all values where non-zero `labels` are used.

  The `index` argument only works when `labels` is specified.

Returns maxima_positions – Maxima positions of `input` over the `index` selected regions from `labels`.

Return type ndarray

dask_ndmeasure.**mean**(*input*, *labels=None*, *index=None*)
Calculate the mean of the values of an array at labels.

Parameters

- **input** (*ndarray*) – N-D image data
- **labels** (*ndarray, optional*) – Image features noted by integers. If None (default), all values.
- **index** (*int or sequence of ints, optional*) – Labels to include in output. If None (default), all values where non-zero `labels` are used.

  The `index` argument only works when `labels` is specified.

Returns **means** – Mean of `input` over the `index` selected regions from `labels`.

Return type ndarray

dask_ndmeasure.**median**(*input*, *labels=None*, *index=None*)
Find the median over an image at specified subregions.

Parameters

- **input** (*ndarray*) – N-D image data
- **labels** (*ndarray, optional*) – Image features noted by integers. If None (default), all values.
- **index** (*int or sequence of ints, optional*) – Labels to include in output. If None (default), all values where non-zero `labels` are used.

  The `index` argument only works when `labels` is specified.

Returns **medians** – Median of `input` over the `index` selected regions from `labels`.

Return type ndarray

dask_ndmeasure.**minimum**(*input*, *labels=None*, *index=None*)
Find the minima over an image at specified subregions.

Parameters

- **input** (*ndarray*) – N-D image data
- **labels** (*ndarray, optional*) – Image features noted by integers. If None (default), all values.
- **index** (*int or sequence of ints, optional*) – Labels to include in output. If None (default), all values where non-zero `labels` are used.

  The `index` argument only works when `labels` is specified.

Returns **minima** – Minima of `input` over the `index` selected regions from `labels`.

Return type ndarray

dask_ndmeasure.**minimum_position**(*input*, *labels=None*, *index=None*)
Find the positions of minima over an image at specified subregions.

Parameters

- **input** (*ndarray*) – N-D image data
- **labels** (*ndarray, optional*) – Image features noted by integers. If None (default), all values.

- **index** (*int or sequence of ints, optional*) – Labels to include in output. If None (default), all values where non-zero `labels` are used.

  The `index` argument only works when `labels` is specified.

Returns **minima_positions** – Maxima positions of `input` over the `index` selected regions from `labels`.

Return type ndarray

`dask_ndmeasure.`**`standard_deviation`**(*input*, *labels=None*, *index=None*)

Find the standard deviation over an image at specified subregions.

Parameters

- **input** (*ndarray*) – N-D image data
- **labels** (*ndarray, optional*) – Image features noted by integers. If None (default), all values.
- **index** (*int or sequence of ints, optional*) – Labels to include in output. If None (default), all values where non-zero `labels` are used.

  The `index` argument only works when `labels` is specified.

Returns **standard_deviation** – Standard deviation of `input` over the `index` selected regions from `labels`.

Return type ndarray

`dask_ndmeasure.`**`sum`**(*input*, *labels=None*, *index=None*)

Calculate the sum of the values of the array.

Parameters

- **input** (*ndarray*) – N-D image data
- **labels** (*ndarray, optional*) – Image features noted by integers. If None (default), all values.
- **index** (*int or sequence of ints, optional*) – Labels to include in output. If None (default), all values where non-zero `labels` are used.

  The `index` argument only works when `labels` is specified.

Returns **sum** – Sum of `input` over the `index` selected regions from `labels`.

Return type ndarray

`dask_ndmeasure.`**`variance`**(*input*, *labels=None*, *index=None*)

Find the variance over an image at specified subregions.

Parameters

- **input** (*ndarray*) – N-D image data
- **labels** (*ndarray, optional*) – Image features noted by integers. If None (default), all values.
- **index** (*int or sequence of ints, optional*) – Labels to include in output. If None (default), all values where non-zero `labels` are used.

  The `index` argument only works when `labels` is specified.

Returns **variance** – Variance of `input` over the `index` selected regions from `labels`.

Return type ndarray

CHAPTER 5

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

5.1 Types of Contributions

5.1.1 Report Bugs

Report bugs at https://github.com/dask-image/dask-ndmeasure/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

5.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

5.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

5.1.4 Write Documentation

dask-ndmeasure could always use more documentation, whether as part of the official dask-ndmeasure docs, in docstrings, or even on the web in blog posts, articles, and such.

5.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/dask-image/dask-ndmeasure/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

5.2 Get Started!

Ready to contribute? Here's how to set up *dask-ndmeasure* for local development.

1. Fork the *dask-ndmeasure* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/dask-ndmeasure.git
```

3. Install your local copy into an environment. Assuming you have conda installed, this is how you set up your fork for local development (on Windows drop *source*). Replace *"<some version>"* with the Python version used for testing.:

```
$ conda create -n dask-ndmeasureenv python="<some version>"
$ source activate dask-ndmeasureenv
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions:

```
$ flake8 dask_ndmeasure tests
$ python setup.py test or py.test
```

   To get flake8, just conda install it into your environment.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

5.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.7, 3.4, 3.5, and 3.6. Check https://travis-ci.org/dask-image/dask-ndmeasure/pull_requests and make sure that the tests pass for all supported Python versions.

5.4 Tips

To run a subset of tests:

```
$ py.test tests/test_core.py
```

CHAPTER 6

Credits

6.1 Development Lead

- John Kirkham, Howard Hughes Medical Institute <kirkhamj@janelia.hhmi.org>

6.2 Contributors

None yet. Why not be the first?

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

Python Module Index

d

Index