cellseg

Release 0.1.0

Nelson Gonzabato

Sep 23, 2021

CONTENTS:

CHAPTER

ONE

CELLSEG: MULTICLASS CELL SEGMENTATION

CHAPTER

TWO

DEVELOPMENT STAGE

- [x] Read Tiff Images
- [x] Read Non Tiff Images
- [x] Write Data Transformers and Loaders
- [ ] Write functional model plus scripts
- [ ] Modify model weights/layers
- [ ] Read stacked tiff images/videos

CHAPTER

THREE

INTRODUCTION

`cellseg` is a PyTorch (`torch`) based deep learning package aimed at multiclass cell segmentation.

CHAPTER

FOUR

INSTALLATION

```
pip install cellseg
```

Or if you want to build from source

```
git clone git@github.com:Nelson-Gon/cellseg.git
cd cellseg
python setup.py install
```

CHAPTER

FIVE

USAGE

5.1 Script mode

View images

```
python -m cellseg -d data/train -t "image" -n 4 -s 512
```

To get help

```
python -m cellseg --help
#usage: __main__.py [-h] -d IMAGE_DIRECTORY -s IMAGE_SIZE -t TARGET -n NUMBER
#
#optional arguments:
#  -h, --help            show this help message and exit
#  -d IMAGE_DIRECTORY, --image-directory IMAGE_DIRECTORY
#                        Path to image directory containing images and
#                        masks/labels
#  -s IMAGE_SIZE, --image-size IMAGE_SIZE
#                        Size of images
#  -t TARGET, --target TARGET
#                        Target images to show
#  -n NUMBER, --number NUMBER
#                        Number of images to show
```

5.2 Programming mode

Importing relevant modules

```
from cellseg.data import DataProcessor
from cellseg.model import CellNet
from cellseg.utils import DataProcessor, show_images
```

Creating a a model object

```
my_model = CellNet()
```

Load training data

```
train_data = DataProcessor(image_dir="data/train/images", label_dir="data/train/images
↪", image_suffix="tif")
```

View loaded images or masks

```
show_images(train_data, number = 8, target="image")
```

Training

CHAPTER

SIX

CELLSEG

6.1 data module

class cellseg.data.**DataProcessor**(*image_dir*, *label_dir*, *target_size=(512, 512)*, *image_suffix='tif'*)

Bases: Generic[torch.utils.data.dataset.T_co]

transform(*image*, *mask*)

Parameters

- **image** – Image to be transformed.
- **mask** – Label/Mask to be transformed.

Returns Transformed images that have been randomly cropped, flipped, and resized to the target size.

6.2 model module

class cellseg.model.**CellNet**(*input_shape=32*, *channels=1*)

Bases: torch.nn.modules.module.Module

forward(*x*)

Defines the computation performed at every call.

Should be overridden by all subclasses.

Note: Although the recipe for forward pass needs to be defined within this function, one should call the Module instance afterwards instead of this since the former takes care of running the registered hooks while the latter silently ignores them.

---

training: bool

6.3 utils module

cellseg.utils.**show_images**(*dataset_object*, *number=None*, *fig_size=(20, 20)*, *target='image'*)

Parameters

- **dataset_object** – An object of class DataProcessor.
- **number** – Number of images to plot
- **target** – Type of images to show. One of "image" or "mask", defaults to image
- **fig_size** – Figure size, defaults to (20, 20)

Returns A plot showing images or labels

CHAPTER

SEVEN

CELLSEG: MULTICLASS CELL SEGMENTATION

Version 0.1.0

- Initialized script mode.
- Optimized imports
- Updated docs to reflect proper DataProcessor usage.
- Initial tests.
- `DataProcessor` now errors if image and mask/label lengths differ.
- Dropped thresholding methods. Please use pyautocv or any other image processing package of your choice.
- Refactored `show_images` to handle follow current `DataProcessor` logic. Fixed a bug that caused rows and column switch.
- `DataProcessor` now returns a dictionary containing an image, its label, and index.
- Reading stacked tiff images is no longer supported for now.
- `dir_type` was dropped in `DataProcessor`. Only provide a directory to `image_dir` containing images and masks.
- Added data from cytounet
- Versioning is now automated.
- Updated docs to show that this is a work in progress.
- Updated docs to use explicit imports.

version 0.0.0

- Preserve name on PyPI
- Fixed issues with `show_images` showing blank images for masks (labels).
- Fixed issues with `uint16` not working with `PIL`.
- `DataProcessor` can now transform images to a given target size.
- Renamed `DataLoader` class to `DataProcessor` to avoid conflicts with `torch.utils.data.DataLoader`
- Added initial simple CNN model with a single layer
- Added `show_images` in `utils.py` to allow quick visualization of a given number of images from a given stack of images.
- Implemented data loaders.
- Conceptualized project

CHAPTER

EIGHT

CONTRIBUTING TO CELLSEG

This document provides guidelines for contributions to `cellseg`.

Kinds of contribution

- Typo fixes
- Documentation enhancements
- Pull requests

Fixing typos and enhancing documentation

To fix typos and/or grammatical errors, please edit the corresponding `.py` or `.md` file that generates the documentation.

Please also update the docs using `sphinx`

Pull Requests

- Please raise an issue for discussion and reproducibility checks at issues
- Once the bug/enhancement is approved, please create a Git branch for the pull request.
- Make changes and ensure that builds are passing the necessary checks on Travis.
- Update `changelog.md` to reflect the changes made.
- Do the following:

```
bash scripts/mkdocs.sh #projectnamehere
```

- Releasing

```
bash scripts/release.sh
```

The above does the following:

- Makes `dist` with `python setup.py sdist` at the very minimum. Ensure everything necessary is included in `Manifest.in`.
- Uploads `dist` to test.pypi.org with `twine upload --repository-url https://test.pypi.org/legacy/ dist/*`
- If everything looks good, asks you to upload to pypi.org with `twine upload dist/*`

Please note that the 'cellseg' project is released with a Contributor Code of Conduct. By contributing to this project, you agree to abide by its terms.

See also for a guide on Sphinx documentation.

CHAPTER

NINE

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

C

INDEX