Manual for Artery Segmentation

Step-wise guide to setup the environment and using the resources.

Environment setup

Use `conda` to create a virtual environment.

```
pip install monai
pip install torchio
pip install notebook
```

`TorchIO` is required only if you are using `resizing` and `pixel spacing` transforms. In the beginning I used `resizing` and `spacing` transforms. However, recently I excluded them.

Folder Structure

Create a folder for dataset. Inside the folder create five more folder and sub-folders as follows as follows.

```
data_new
|-data_original
  |-train
    |-volumes
    |-masks
|-data_processed
  |-train
    |-volumes
    |-masks
|-data_splitted
  |-train
    |-volumes
    |-masks
  |-val
    |-volumes
    |-masks
|-data_16_groups
  |-train
    |-images
    |-labels
  |-val
    |-iamges
    |-labels
|-data_test
  |-data_original
  |-data_patients
  |-data_groups
```

Preprocessing

Steps

- Checking labels
- Resizing and fixing pixel spacing
- Splitting data
- Creating sub-volumes
- Remove empty

Scripts

Following scripts will be used.

```
label_mismatch_fix.ipynb
label_extra_class_fix.ipynb
label_swap.ipynb
preprocessing_train.ipynb
create_slice_groups.ipynb
remove_empty_volumes.ipynb
preprocessing_test.ipynb
```

Procedure

While using any script, write the input and output paths according to your folder structure.

- Create the dataset folders as shown in the tree above.
- Copy all the training volumes and masks inside `data_original/train/` folder.
- Copy all the test volumes inside `data_test/data_original/` folder.
- After organizing the data, the first step is to check and fix the labels.
    - Manually open each mask in **3D Slicer** and verify the labels.
        - Note the names of masks for which the `labels` are opposite. We will use these names later.
    - Use the `label_mismatch_fix.ipynb` or `label_extra_class_fix.ipynb` to fix the **extra class issue**.
        - Input dataset directory should be `/data_new/data_original`
        - Give an output directory to store the resulting masks
        - Copy and paste these masks inside `/data_original/train/masks` folder.
    - Then use `label_swap.ipynb` to fix the issue of **opposite labels**.
        - Give input directory path i.e. `/data_original/train/masks`
        - Save resulting masks at desired location by providing `out_path`

        - Copy and paste these masks inside `/data_original/train/masks` folder.
- (Optional) Next step is to `resize` and fix `spacing` of training data. Use `preprocessing_train` for that purpose. The resulting data will go inside `data_prcoessed` folder.
- Use `data_split.ipynb` to split the training data into `train` and `val` data. The resulting data should go inside `data_splitted` folder.
    - Provide input path as `/data_original/train`
        - If you see `data_16_orignal` instead of `data_original`, then change it to `data_original`
    - Provide output folder path as `data_splitted` to save the results
- Use `create_slice_groups.ipynb` to create sub-volumes of the patients. Do it for both validation and training data. Also do it for both `masks` and `volumes`. Resulting data should go in `data_16_groups` folder. Put training data inside train folder and validation inside val folder.
    - Training and validation data
        - Do it for training volumes and masks
        - Then for validation volumes and masks
        - Provide `in_path` and `out_path`
            - For example

```
# For validation volumes
in_path = '/media/trojan/evo/3D-CT-Artery-Segmentation/data/data_new/data_splitted/val/volumes'
out_path = '/media/trojan/evo/3D-CT-Artery-Segmentation/data/data_new/data_16_groups/val/images'

# For validation masks
in_path = '/media/trojan/evo/3D-CT-Artery-Segmentation/data/data_new/data_splitted/val/masks'
out_path = '/media/trojan/evo/3D-CT-Artery-Segmentation/data/data_new/data_16_groups/val/labels'
```

        - In paths, replace `data_augmented` with `data_splitted`. We are not using augmentation data anymore.
- Nest use `remove_empty_volumes.ipynb` to remove the images and labels with no `foreground` class.
    - Do for `train` and then `val` data
    - Paths should be `/data_new/data_16_groups/train` and `/data_new/data_16_groups/val`

Training

Use `train.ipynb` to train the model. Save the resulting model at a desired location.

Testing

For testing there are three notebooks.

- `test.ipynb` to test a single model on both `validation` and `test` data and then save results.
- `test_ensemble.ipynb` to test three models ensemble on both `valdiation` and `test` data.
- `test_ensemble_fast` to test three models ensemble on `test` data and save results.

💡 Pay attention to the code for some hard-coded stuff like `paths`, `patient numbers` etc.