Documentation - Expectation-Maximisation Algorithm in Docker by Esther Alberts

0.1 Installation requirements

An up-to-date installation of the Docker Daemon is required for the container to work. The detailed installation instructions and installer download links can be found at the Docker homepage[1]. The Docker image has been tested on Docker for Mac (Native version CE 17.03.1+), on Linux Ubuntu and Windows 10 (Native version 17.03).

0.2 Running the container

After installation, the following Docker command can be used to run the Expectation-Maximisation algorithm for brain tumor segmentation:[2]

```
docker run -v $(pwd):/data -it cberger/em_final python main.py
```

This command collects the container image from the public Docker Hub repository [3] (if there already is a local image, use "docker pull cberger/em_final" first) and mounts a volume called /data to the filesystem of the container. This volume provides the interface between the host and the container and is mounted in the current directory on the host. If the main.py file can't be executed, try adding --workdir=/usr/src/app to the Docker command to set the working directory to the folder containing main.py. If there are no files in the host directory, the script exits. If you don't want to run the code from your current directory, you can also replace the mount point -v $(**pwd**):/data with your own path, e.g. -v <full or relative path on host>:/data. However, all subfolders of /data have to be named according to the list below.
Note: For Windows, you have to adjust the command to always reflect a full Windows path, for example:

```
docker run -v "C:\Users\Christoph Berger\Desktop\data":/data -it
  cberger/em_final python main.py
```

On Windows systems, Docker will also likely require permission to mount the drive, which is not needed on Unix systems. There will be a simple pop-up requesting the permission.

0.3 Segmenting new volumes

I suggest testing the functionality of the algorithm with the example files provided to check if everything is working properly. You can download the whole code from Bitbucket[2] and extract the example files.
To segment new volumes, the following folders and files have to be placed inside the Input-Folder:

- Subfolder: preprocessed_data

  MR_Flair.703_mask.nii

  MR_Flair.703.nii

  MR_T1.704.nii

  MR_T1c.705.nii

  MR_T2.706.nii

- Subfolder: registered_atlas

  csf_reg_via_mask.nii

  grey_reg_via_mask.nii

  white_reg_via_mask.nii

After putting all the files in the mentioned folders and naming them accordingly, running the aforementioned command again should produce the desired output of segmented volumes. The output files are written to a newly created folder (Results) inside the initial data directory.
Important: Make sure that all your images, mask and atlas maps (GM, WM and CSF) are in the same reference space and have the same resolution and dimension as the example files. Output for image parameters from the example files:

```
image shape: (176, 216, 160)
image spacing: (1.0, 1.0, 1.0)
image data type: int16
```

0.4 Known bugs and hiccups

Update(01/07/2017): There is now an option to supply only some volume files, however, if no arguments are passed, the script looks for all files and breaks if it doesn't find them all. Additionally, the brain mask file always has to be supplied. The folder structure and file names still have to be the exact same way as in the example directory, regardless of which arguments are passed. Arguments:

```
usage: main.py [-h] [-a] [-t1] [-t1c] [-t2] [-f]
optional arguments:
-h, --help    show this help message and exit
-a, --all     To use all modalities: DEFAULT
-t1, --T1     Use this flag to look for the T1 volume: MR_T1.704.nii
-t1c, --T1c   Use this flag to look for the T1c volume: MR_T1c.705.nii
-t2, --T2     Use this flag to look for the T2 volume: MR_T2.706.nii
-f, --Flair   Use this flag to look for the Flair volume:
  MR_Flair.703.nii
```

Nevertheless, the folder structure and the file names still have to adhere to the strict layout as in the example files.

Attention: The implementation of argparse in main.py is rudimentary at best and may lead to unexpected errors. This is just preliminary to avoid the hassle of having to supply all files.

0.5 Questions, Authors, Licensing

If you have any questions, please contact me at c.berger [at] tum.de
The original algorithm was written by Esther Alberts, Ph.D. student at the Image-Based Biomedical Modeling Group (IBBM) at TU Munich. Her code and the corresponding licences can be found at her Bitbucket repository[2].

Christoph Berger, July 13, 2017

References

[1] "Docker installation instructions." https://docs.docker.com/engine/installation/. Accessed: 2017-06-27.

[2] E. Alberts, "Expectation-Maximisation algorithm for brain tumor segmentation." https://bitbucket.org/s0216660/brain_tumor_segmentation_em. Accessed: 2017-06-25.

[3] C. Berger, "Docker image on docker hub for em algorithm." https://hub.docker.com/r/cberger/em_final/. Accessed: 2017-06-17.