Installation of MemBrain

1. **Install via pip**
   Open a terminal and run: `pip install membrain`. This command installs the MemBrain-v2 package, including the segmentation tool `membrain segment`.

2. **Create a Conda Environment**
   Generate a conda environment to avoid dependency conflicts.

```
conda create -n membrain python=3.9
conda activate membrain
pip install membrain
```

Using MemBrain to Segment

1. **Activate the MemBrain Environment**
   Before running any MemBrain commands, ensure your environment is active:

```
conda activate membrain
```

2. **Organize your Files**
   Place your aligned, binned *.mrc tomograms from etomo in a directory named `membrain_inputs/`.

   Then, create an empty output directory:

```
mkdir -p ./membrain_seg_outputs/labelname
```

3. **Running MemBrain on a Tomogram**
   Navigate to your project folder: cd /path/to/your/project. Run the segmentation by using the following template:

```
membrain segment \
      --tomogram-path
./membrain_inputs/membrain_input_filename.mrc \
      --ckpt-path ~/membrain-
seg/models/MemBrain_seg_v10_alpha.ckpt \
      --out-folder ./membrain_seg_outputs/labelname \
      --store-probabilities
```

   Ensure that you replace the following placeholders:

   - `membrain_input_filename.mrc` with your file name

- `membrain_seg_outputs/labelname` with your actual output directory

After completion, the output folder will contain the following files:

- `* _MemBrain_seg_v10_alpha.ckpt_segmented.mrc`
- `*_scores.mrc`

The scores file will be used in the semantic segmentation step.

Example of a Full Run

Assume I have a file titled `Position5B_0602_deconv.mrc` in the directory `~/relan/Cryo-EMData/membrain_inputs/`. Assume that I have created the directory `membrain_outputs` within the Cryo-EM directory. Also assume that my membrain-seg installation was in the home directory.

The full commands would be:

```
cd /relan/Cryo-EMData/membrain_inputs

conda activate membrain

membrain segment \
  --tomogram-path ~/relan/Cryo-
EMData/membrain_inputs/Position5B_0602_deconv.mrc \
  --ckpt-path ~/membrain-seg/models/MemBrain_seg_v10_alpha.ckpt \
  --out-folder ~/relan/Cryo-EMData/membrain_outputs/position5b_0602 \
  --store-probabilities
```

You can also write the command in one line without the backslashes.

```
membrain segment --tomogram-path ~/relan/Cryo-
EMData/membrain_inputs/Position5B_0602_deconv.mrc --ckpt-path
~/membrain-seg/models/MemBrain_seg_v10_alpha.ckpt --out-folder
~/relan/Cryo-EMData/membrain_outputs/position5b_0602 --store-
probabilities
```

MemBrain takes roughly 3-5 minutes to segment a tomogram depending on CPU power.

Running a Batch Script on Multiple Tomograms

1. **Organize your Files with a Consistent Naming Pattern**
   We can call MemBrain once per file from inside a shell script instead of typing the command manually each time.

   Create a directory for your input files (e.g. membrain_inputs). Add all tomograms you want to segment in this folder. Ensure that they all follow some consistent naming pattern.

2. **Modify and Copy Script**
   Ensure that you modify your script to read the specific naming pattern you decide on. You can also edit the three variables (input directory, output base, and ckpt path) if necessary.

   The script should be copied into your input directory (the same directory as your tomograms you want to segment). Make sure to make the script executable by running:

   ```
   chmod +x run_membrain_batch.sh
   ```

3. **Navigate to the Directory and Run the Script**
   The script will first activate the membrain conda environment. Then, it will define *one input directory* and *one output base directory* where the results will be written. The shell script will loop over every file that matches the naming pattern you specify. For this example, we will follow the naming pattern:

   Position*_0602_deconv.mrc.

   The script will then create a matching sub-folder inside the output base folder and call membrain segment on the tomograms.

   Run the script with the command:

   ```
   ./run_membrain_batch.sh
   ```