`biacpype` User Manual

Adcock Lab
Center for Cognitive Neuroscience, Duke University

September 11, 2018

Contents

1 Overview

biacpype serves as a pipeline for converting raw fMRI data from Brain Imaging & Analysis Center (BIAC) at Duke University to the new standard Brain Imaging Data Structure (BIDS) format. The main parts of biacpype contains:

- `util`: BIAC dataset validation
- `biac2bids`: conversion from BIAC to BIDS
- `helper`: helper scripts

In a nutshell, `biacpype` serves as an adapter to the `bxh2bids` code developed by John Graner from LaBar lab. This is the core of the conversion; all kudos to John!

The structure of this repository is:

```
|-biacpype // serves as the library
|    |
|    |_biac2bids
|    |_ ...
|
|-scripts  // scripts for running a variety of pipelines
|    |
|    |_convert_to_bids.py
|    |_ ...
|-tutorial // a tutorial notebook
|
|-docs     // documentation
    |_ ...
```

To install dependencies of this repo, run

```
pip install -r requirements.txt
```

The following section will introduce main modules of `biacpype`, and the corresponding scripts for using the pipelines.

If you have any questions, please email Preston at linxing.jiang@duke.edu or prestonj@cs.washington.edu.

You can also go to Gitter Chatroom: https://gitter.im/MotivatedMemoryLab/biacpype to ask for help! You will need to log in with your GitHub account.

2 util

This is the module which serves as the utility module. It contains validation functionality to let you validate your dataset from BIAC to check if it is ready to be converted.

2.1 Dataset Structure

The pipeline makes the following assumptions on the format of your raw data from BIAC:

```
|-Data
|     |
|     |-Func
|     |   |
|     |   |-<[date_]subject>
|     |   |        |
|     |   |        |-<biac5_subject_task_run>.bxh
|     |   |        |-<biac5_subject_task_run>.nii.gz
|     |   |        |-...
|     |   |        |-series_order_note.tsv
|     |   |-...
|     |
|     |-Anat
|         |-<[date_]subject>
|         |        |
|         |        |-<biac5_subject_task_run>.bxh
|         |        |-<biac5_subject_task_run>.nii.gz
|         |        |-...
|         |        |-series_order_note.tsv
|         |-...
|
|-biac_id_mapping.tsv
```

Explanations:

- `Data` folder has to contain `Func` and `Anat`, and they must have the exact same folders
- Subfolders in `Func` and `Anat` are in format `[date_]subject` where `[date_]` is optional. E.g. `19354` and `20140101_19354` are both acceptable.
- Each file in `Func` and `Anat` must in format `biac5_subject_task_run`. Usually, `task` is a single digit number, `run` is two-digit. E.g. `biac5_19354_4_01.bxh`
- Each subfolder in `Func` and `Anat` **must contain** a `series_order_note.tsv` to tell the pipeline what each `task` number stands for. E.g. 4 stands for "TRAIN". Requirements for this file are later explained.
- In the same folder containing `Data`, there **must contain** a `biac_id_mapping.tsv` which tells the pipeline the mapping from BIAC_ID (e.g. 19354) to the session name (e.g. Session-1) and the Real_ID used by your lab (e.g. 101). Requirements for this file are later explained.

Requirements on `series_order_note.tsv` are as follows:

4	LOCALIZER
7	TRAIN1
...	...

Note:

- The values must be **tab** separated.
- The first column serves as the primary key (they must be unique).
- The first column must be task code, and the second column must be the task name. There can only be two columns.
- If you have fmap data task code, be sure to name the translation **"fmap"** exactly. In this way, the data will be put in `fmap` folder after BIDS conversion, not `anat` folder.

Requirements on `biac_id_mapping.tsv` are as follows:

BIAC_ID	[Session]	Real_ID
19354	SRM	101
19338	SPM	102
19368	SPM	101
...	...	...

Note:

- The values must be **tab** separated.
- The first column serves as the primary key (they must be unique)
- The headers must follow the rules (watch letter cases)!
- If your experiment does not have multiple sessions, you can ignore the `Session` column

2.2 Scripts

The script associated with this module is `scripts/validate_biac_study_folder.py`. You need one command line input, `STUDY_PATH`.

Run

```
python scripts/validate_biac_study_folder.py -h
```

to see help messages.

Run

```
python scripts/validate_biac_study_folder.py STUDY_PATH
```

to validate your dataset.

If there are errors, they will be printed out to console like this:

```
### Following erros happend: ###
...
```

All the error logs are saved in `biacpype/logs/validation.log`. If there are no errors, you should see the following printed out:

```
Your study path passed validation!  You are now ready for conversion
```

3 biac2bids

`biac2bids` module is the pipeline for converting raw data from BIAC in forms of `bxh` and `nifti` to BIDS format. It wraps the `bxh2bids` code developed by John with validation, automatic Json file generation, and naming clean-up. The workflow is as follows:

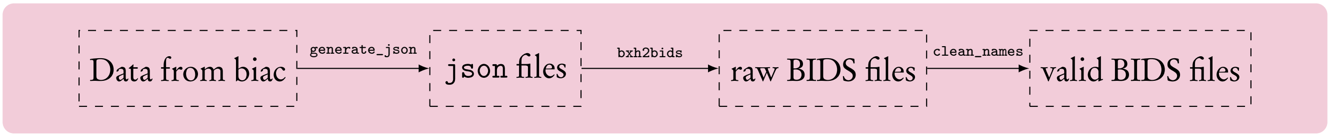


Figure 1: `biac2bids` workflow

3.1 Scripts

The script associated with this module is `scripts/convert_to_bids.py`. There are four parameters the user has to give to command line. They are:

- `STUDY_PATH`: the path to your study file (which contains `Data` and `bids_id_mapping.csv`)
- `JSON_OUTPUT_PATH`: the path where the user wants the json files to be saved
- `BIDS_PATH`: the path where the user wants the new BIDS format data to be saved
- `LOG_PATH`: the path where the user wants the logs to be saved

Run

```
python scripts/convert_to_bids.py -h
```

to see help messages.

Run

```
python scripts/convert_to_bids.py STUDY_PATH JSON_PATH BIDS_PATH LOG_PATH
```

to convert your dataset.

3.2 Common Errors During Conversion

1. `Log file already exists!? They should be time-stamped down to the minute!`

   This is because you run the conversion too frequently!

   Quick solution: delete all your logs and run it again.

2. `Error:root:scan description not found in template file!`

   This is because the scan description (printed out next line) is not saved in template file.

   Quick solution: add the description to the scan description file following its format.

   For example, if you see the following errors:

   ```
   Error:root:scan description not found in template file!
   Error:root:Description:  fMRI 481x5
   Error:root:Template File:  psd_types.json
   ```

   This means your data description is `fMRI 481x5` and the program does not what kind of data it is. You need to add it to `psd_types.json` as follows (assuming this functional data):

   ```
   {
       ...,
       "fMRI 481x5": {
           "type":"func",
           "label":"bold"
       }
   }
   ```

3. `Output file already exists: ...`

   This is because some bids files already exist at the BIDS path! (often because the conversion stopped in the middle)

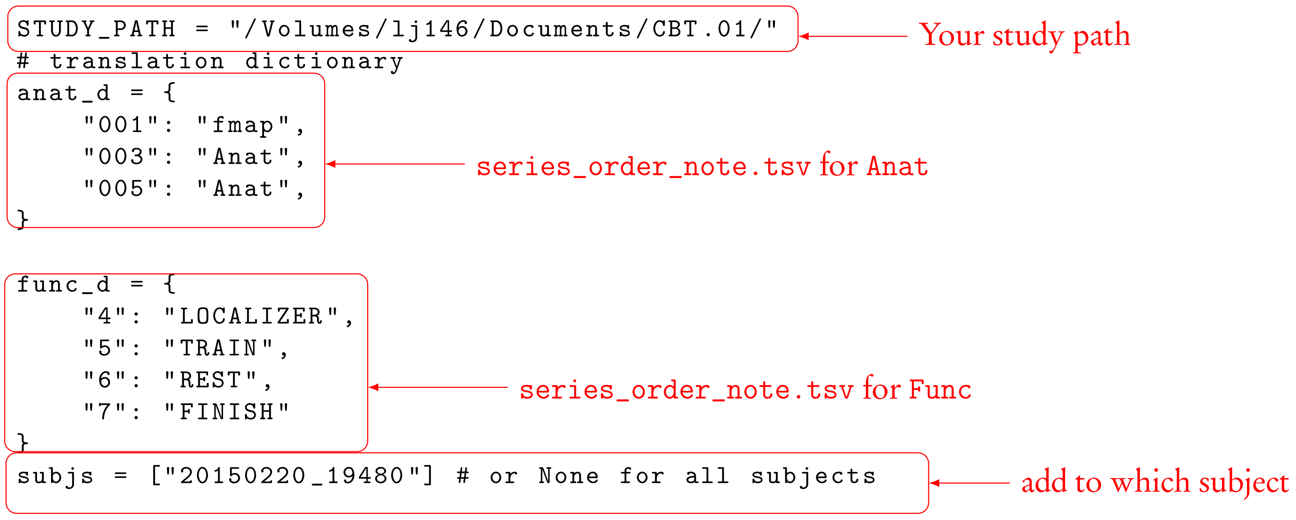
   Quick solution: delete the existing bids files.

4 helper

The `helper` module aims to provide some helper scripts to make the conversion process simply. If you have features which you think would be useful and the module does not cover, please let us know through GitHub Feature Requests, or better, build them in your forked branch and submit Pull Requests!

4.1 `create_series_order_notes.py`

If you are converting an old dataset, you may not have `series_order_note.tsv` ready. This script can help you with that. You will need edit the script and tweak a few things:


Your study path
series_order_note.tsv for Anat
series_order_note.tsv for Func
add to which subject

```
STUDY_PATH = "/Volumes/lj146/Documents/CBT.01/"
# translation dictionary
anat_d = {
    "001": "fmap",
    "003": "Anat",
    "005": "Anat",
}

func_d = {
    "4": "LOCALIZER",
    "5": "TRAIN",
    "6": "REST",
    "7": "FINISH"
}
subjs = ["20150220_19480"] # or None for all subjects

### creation begins ###
set_paths(STUDY_PATH=STUDY_PATH)
populate_file("Anat", anat_d, subjs=subjs)
populate_file("Func", func_d, subjs=subjs)
```

The four red rectangles are the areas you need to change to create the files. They should be straightforward to understand. Note that to add the files to **all** subjects, set

```
subjs = None
```

After you change the fields, run

```
python scripts/create_series_order_notes.py
```

to create the files.

Sometimes, not all of your subjects have the exactly same task code mapping. You will need to manually change these after the tsv files are populated to all folders.

5 Contributing and Asking for Help!

We hope you can contribute and help us develop this pipeline better through:

- Create GitHub Issues and Pull Requests to let us be aware of the uncaught bugs
- Chat on Gitter and ask for help
- Let us know what feature you wish the pipeline had