SEEG PROC
操作手册

冯帆

August 14, 2019

1 简介

SEEG PROC 项目能够解决 SEEG 脑电数据中 mark 的识别问题，以便于将被试在响应期间的脑电数据从原始脑电数据中分离出来。项目内使用 Python 3.7 和 MATLAB 2016b 作为主要的编程语言。整个项目托管在 GitHub 账户`https://github.com/Bearsuny/seeg_proc.git` 上。使用前需要访问上述网址，点击右侧绿色 Clone or download 按钮，并解压缩到任意目录下，解压缩后的文件目录如图 1所示。

2 前期准备

2.1 安装工具依赖

SEEG PROC 是在 Ubuntu 16.04 操作系统下开发的。该项目既可以运行在 Ubuntu 16.04 及以上的系统内，也可以运行在 Windows 10 操作系统中。在使用之前，需要在相应的操作系统中安装 Python 3.7 及以上版本以及 MATLAB 2016b。

此外，还需要安装一些 Python 工具包。请打开命令行窗口（Terminal/CMD），并切换到 SEEG PROC 的根目录。如图 2所示，输入 pip install -r ./requirement.txt 后回车，该命令会自动从网络上下载项目所需要的工具包 (NumPy, Pandas, tqdm, sklearn, matplotlib)。

2.2 被试数据整理

（1）在 SEEG PROC 的根目录下，新建一个名字叫 data 的文件夹。

（2）在 data 文件夹中，新建一个名字以 0 开头、后缀被试序号的文件夹（比如：序号为 15 的被试，其文件夹的名字叫 015）。

（3）该被试的 SEEG 原始文件（*.edf）以及 Eprime 的行为数据文件（*.csv）放到该被试的文件夹中，整理好的 data 文件夹的结构如图 3所示。

2.3 配置文件的修改

打开 preprocessing 文件夹下面的 config.py 文件，需要修改的地方有以下几处，对于每个被试，修改好的 config.py 文件如图 4所示。

（1）subject_no：修改为被试文件夹的名字。

（2）RAW_SEEG：修改成被试的 SEEG 原始文件（*.edf）的名字，只需要改文件名字（e.g. LAT.edf），不需要更改路径。

（3）RAW_EPRIME：与 RAW_SEEG 类似，只是将被试的 Eprime 行为数据的文件名与之替换（e.g. eprime.csv）。

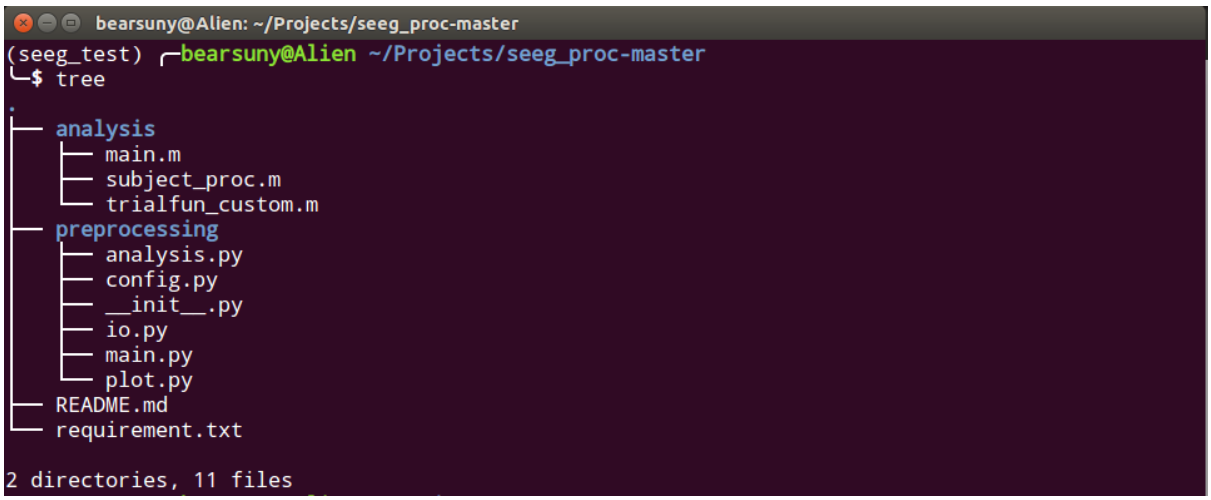

Figure 1: SEEG PROC 文件目录

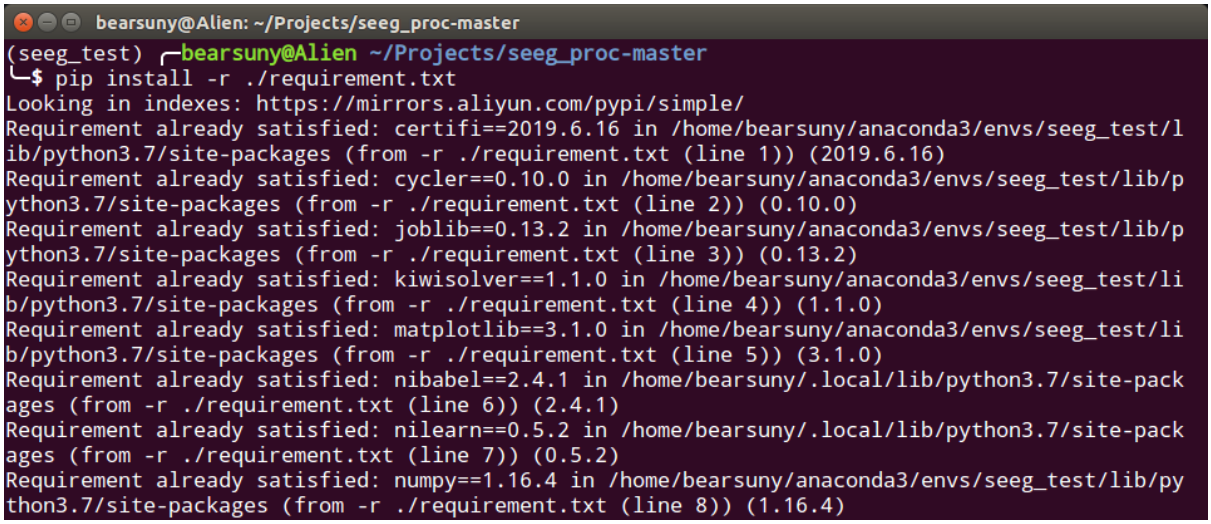

Figure 2: pip 命令安装 python 依赖包（省略后续输出）

3 Python 预处理

3.1 识别 Mark

在 SEEG PROC 的根目录下(seeg_proc-master)，输入 python -m preprocessing.main 命令后回车，即可开始 SEEG 数据中 mark 数据的识别。识别过程会比较长（大概 10 分钟），识别完成后，在 data 文件夹中的相应被试的文件夹下出现一系列的输出文件，该被试的文件夹结构如图 5所示。

（1）中间文件：POL DC12_POL DC11_POL DC10_POL DC09.npy，eprime.npy，eprime_reform.npy，mark.csv。这些文件是程序在识别 mark 过程中产生的中间文件，不需要了解他们的作用。

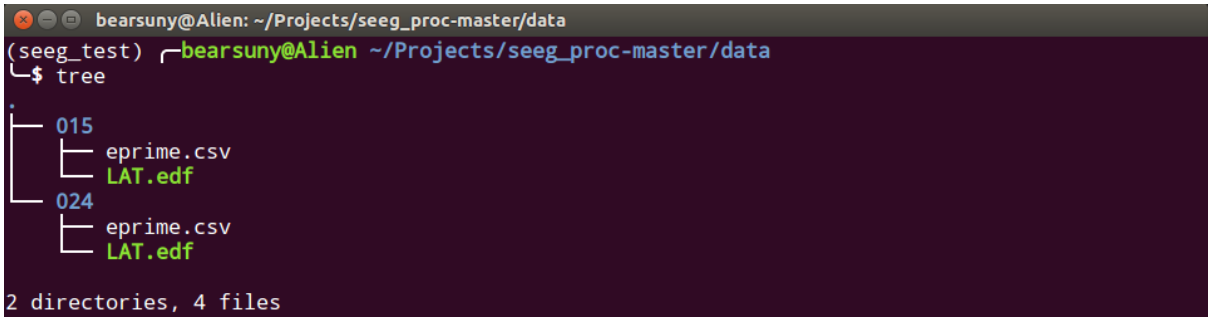

Figure 3: data 文件夹结构（多个被试）

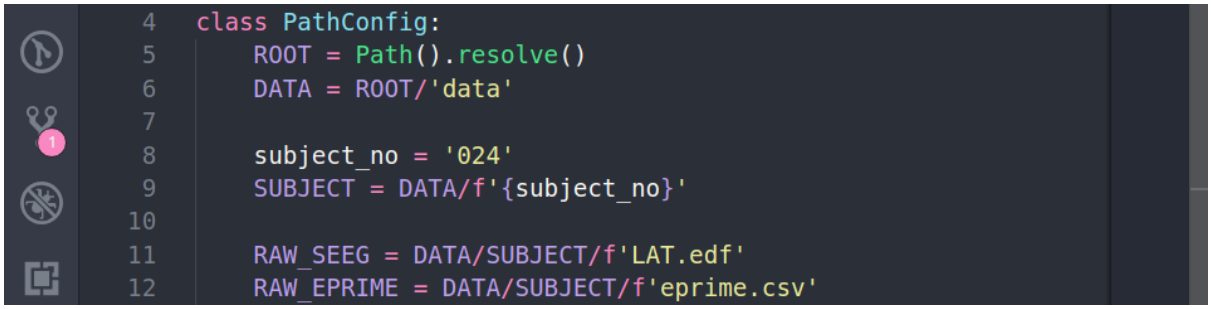

Figure 4: 每个被试的 config 文件

（2）结果文件：event.csv。该文件描述了原始 SEEG 信号中，每个 mark 的类型（type）、类型的值（value）、位置（sample）。该文件就是识别 mark 过程的结果文件。

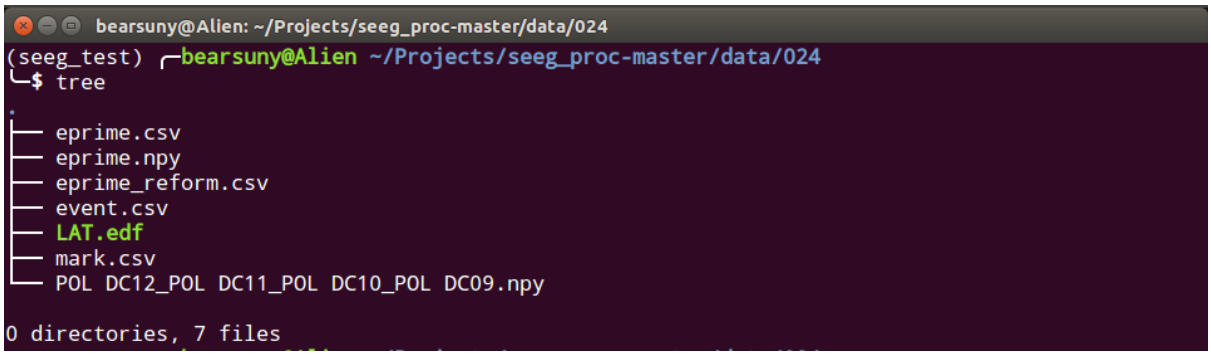

Figure 5: 单个被试在执行 main.py 后的生成文件

在识别 mark 的过程中，程序会产生一系列的输出，如图 6和图 7所示。从这些输出信息中可以获得一些信息：

（1）图 6的开头会打印 SEEG 原始文件（*.edf）的头信息，比如采集时间、采样频率等。

（2）接着，程序会打印 Eprime 行为数据文件中的 mark 数量以及各种类型 mark 的统计（e.g. Mark num in Eprime: 513）。

（3）然后，程序会打印从 SEEG 数据中根据 mark 通道识别出的各种类型的 mark 的统计，这样就可以和 Eprime 中的数据进行比对，来观察不同类型的 mark 在数量上是否一致。

（4）最后，程序会验证 Eprime 行为数据的 mark 的顺序是否与根据 mark 通道识别的 mark 的顺序一致。当出现 “marks are identical to events” 时，表明两者的顺序是一致的，也意味着识别工作的完成。

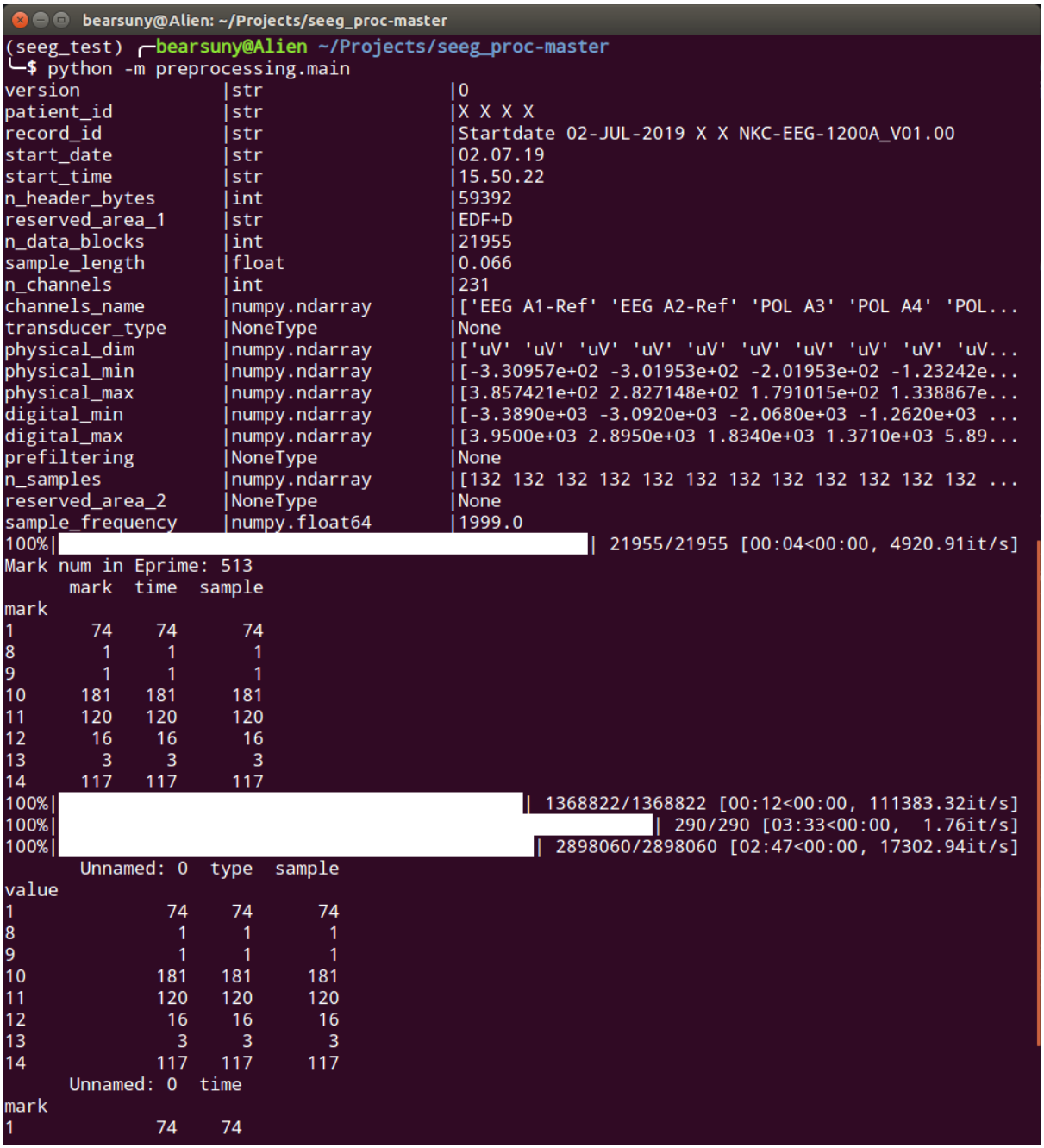

Figure 6: 识别 mark 的程序输出

Figure 7: 识别 mark 的程序输出（续）

3.2 小工具：观察信号

当程序输出例如“15th mark error.”时，则意味着两者不一致。在这里，我提供了一个可以观察 Eprime 行为数据的 mark 与 SEEG 信号数据的 mark 的贴合的程序。

（1）记录 SEEG 数据的采样频率（如图 6中的 sample_frequency=1999）。

（2）打开位于 data 文件夹/被试所在文件夹/event.csv 文件。如图 8所示，可以看到，block=8 的 sample 是 499791，将这个值记录下来。

（3）打开 preprocessing/plot.py 文件。将第 111 行的“np.load(mark_path)[:, 117442:]”更改为“np.load(mark_path)[:, 499791:]”，将第 123 行的 sample_frequency=[2000, 1000] 更改为 sample_frequency=[第一步记录的值, 1000]。

（4）回到 SEEG PROC 的根目录下，执行 python -m preprocessing.plot，即可观察 Eprime 行为数据与 SEEG 信号数据的贴合情况。

如图 9所示，该小工具有三个按钮，分别是向前、暂停和向后观察信号。如果要改变观察信号的起始位置、窗口的分辨率以及观察窗口的长度，则分别需要更改 plot.py 文件中的 start_sec、step_sec 以及 n_init_steps 这三个变量。

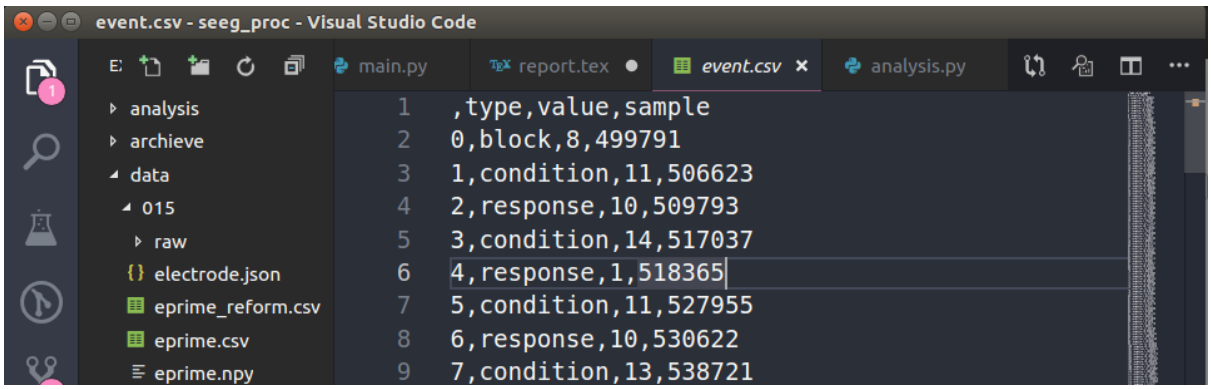

Figure 8: event.csv

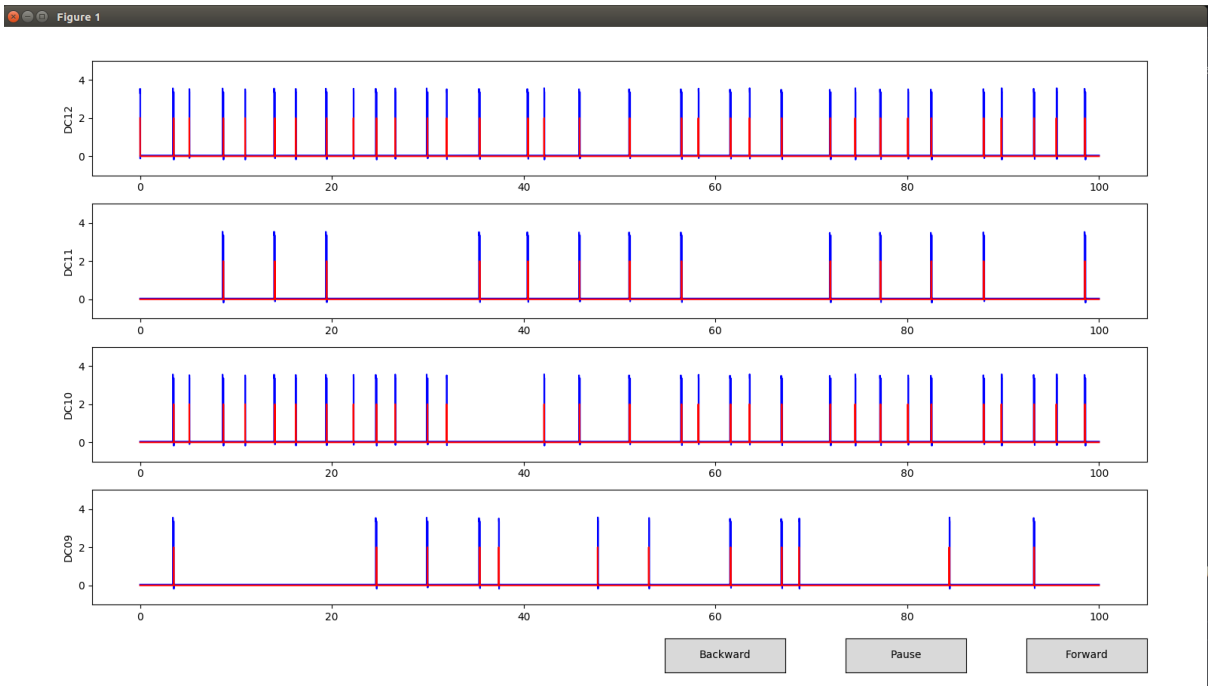

Figure 9: 观察两种 mark 的贴合情况

4 FieldTrip 分析

analysis 文件夹中的程序能够利用 FieldTrip 来分析 SEEG 数据，该程序支持多个被试一起分析。

4.1 安装工具依赖

（1）首先，将 tool.zip 文件解压到 SEEG PROC 的根目录下，并将 fieldtrip 导入到 MATLAB 2016b 中。

（2）将 tool 文件夹中的 experiment.json 以及 electrode.json 文件放到每个被试的数据所在的文件夹中。其中，electrode.json 文件需要根据不同的被试进行定义。该文件定义了不同的电极所属的脑区。如图 10所示，在实际操作中，蓝色的 A 应该被替换成脑区的名称（可以自定义），黄色的 POL 等则是电极的名称。因此，这一步需要首先将被试的 CT 图像中的 SEEG 电极进行标注（参考凯伦的步骤），然后根据标注的结果将电极分类到不同的脑区中。

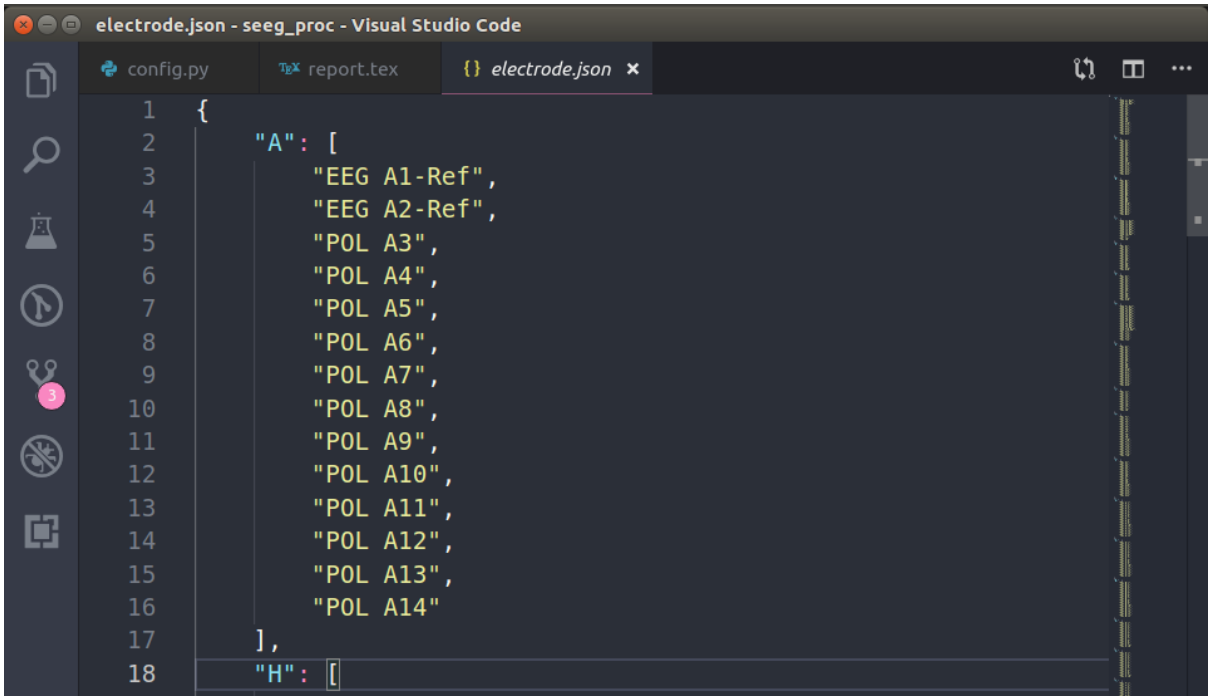

Figure 10: electrode.json

4.2 数据切割及分析

打开 analysis/main.m 文件，修改如下的值：

（1）第 9 行 g_cfg.subject_list：想要分析的被试数据所在的文件夹的名称。

（2）第 10 行 g_cfg.seeg_file_name：被试的 SEEG 数据文件（需要注意不同的被试的 SEEG 文件需要统一命名）。

（3）第 14 行 g_cfg.bad_channel：定义不需要分析的通道。

（4）第 30 行和第 31 行 cfg.trialdef.pretrig 以及 cfg.trialdef.posttrig：定义数据切割的时间段（以秒为单位）。

第 28 行和第 29 行是可选的，是用来确定要分析的是哪种 mark，具体的信息请参考 FieldTrip 的官方网站的教程。在设置完成之后，执行 main 文件即可。

我完成了不同通道的数据平均叠加的功能，对于每个被试具体的分析方法，请参照 analysis/subject_proc.m 文件。每个脑区平均叠加后的结果位于被试数据文件夹中的 avg 文件夹中。