**BM Imaging Report 3**

2014250136 류제성

rs-fMRI를 이용한 functional connectivity 를 분석하는 과정이었다. functional connectivity를 분석하는 과정은 structure connectivity와는 달리 성격의 유사성으로 communication 정도를 정량화하는 과정으로 볼 수 있다. 그 과정으로서 뇌를 90개의 영역으로 나누어 생각하였다.

**A. Images of functional Connectivity**

|  |  |  |
| --- | --- | --- |
|  | Normal(01) | Abnormal(02) |
| 1 | C:\Users\류제성\AppData\Local\Microsoft\Windows\INetCache\Content.Word\0101.png |  |
| 2 |  |  |
| 3 | C:\Users\류제성\AppData\Local\Microsoft\Windows\INetCache\Content.Word\0103.png |  |
| 4 |  |  |
| 5 |  |  |

(범위를 -1부터 1까지로 잡으니, 상관관계가 0인 지점이 중간이 되어, 보고서의 목적대로 그래프가 잘 표현이 되지 않는다고 판단하여, 상관관계가 0~1인 지점을 범위로 하여 그래프 작성하였다.)

**B. Calculated small worldness**

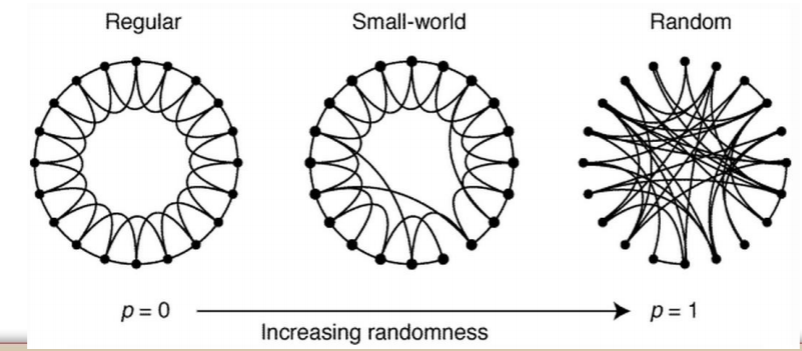
|  |  |  |
| --- | --- | --- |
|  | Normal(01) | Abnormal(02) |
| 1 | 1.048357 | 1.042981 |
| 2 | 1.040275 | 1.033851 |
| 3 | 1.048386 | 1.044676 |
| 4 | 1.050602 | 1.054408 |
| 5 | 1.045472 | 1.051155 |
| average | 1.046618 | 1.045414 |

**C. Qualitative comparison between normal/abnormal subjects**

clustering coefficient(이하 CC) : Number of edges among neighbors i/ Maximum possible number of edges among neighbors

node 주변에 있는 node들이 얼마나 잘 연결되어 있느냐를 나타내는 수이다.

charateristic path length(이하 CPL): path length(edge 갯수)의 평균값



randomness가 증가할수록 CPL은 짧아지고 CC는 감소한다. Regular network는 CPL 이 길고 CC가 크다는 의미이다. 두 가지 단점을 보완한 것이 바로 Small-world이며, CPL은 줄이고 CC는 크게 만들어주는 network이다. 간단히 말해 small-worldness는 CC와 CPL의 비율을 수치로 나타낸 값이다.

위 그래프는 정상군과 비정상군의 각각 5명의 사람들의 CC,CPL, randomized CC, randomized CPL을 평균내어 비교해본 그래프이다. 한눈에 봐도 눈에 띄는 차이가 없다.

또한 connectivity matrix에서도 눈에 띄는 차이를 그래프로도 발견하기 어려웠다.

알츠하이머 유무와 connectivity 및 small-worldness에는 연관성이 없다고 결론지었다.

D. Code

import matplotlib.pyplot as plt  
import nibabel as nib  
import bct  
from scipy.stats import pearsonr  
import numpy as np  
for A in range(1,3):  
 for B in range(1,6):  
 aaldata= nib.load('../Scripts/Data/0%d\_0%d\_aal.nii.gz'%(A,B))  
 fmridata=nib.load('../Scripts/Data/0%d\_0%d\_fmri.nii.gz'%(A,B))  
  
 aal\_value = open('../scripts/Data/aal\_values.txt')#aal value txt 불러오기  
 aal\_list=[]  
 for line in aal\_value :  
 a=int(line.strip())  
 aal\_list.append(a)  
  
 ROIlist=[]  
 for i in range(90):  
 ROIlist.append([])  
  
 time\_avg\_list=[]  
 random\_list=[]  
 random\_list2=[]  
  
  
 aal=aaldata.get\_data()  
 fmri=fmridata.get\_data()  
  
  
  
  
 for i in range(128):  
 for j in range(128):  
 for k in range(35):  
 for J in range(90):  
 AAL=aal[i][j][k]  
  
 if AAL==aal\_list[J]:  
 ROIlist[J].append((i,j,k)) #각 aal value 해당하는 index 값의 모임.  
  
 avg=0  
 for j in range(90):  
 avg\_list = []  
 for k in range(97):  
 sum = 0  
 for i in range(len(ROIlist[j])):  
 x=ROIlist[j][i][0]  
 y=ROIlist[j][i][1]  
 z=ROIlist[j][i][2]  
 sum+=fmri[x][y][z][k]  
 avg = sum / len(ROIlist[j])  
 avg\_list.append(avg) #1\*97 fmri time series  
 time\_avg\_list.append(avg\_list) #90\*97 aal value별 time series  
  
 correlation\_matrix = np.zeros([90,90],dtype=np.float32)  
  
 for i in range(90):  
 for j in range(90):  
 correlation\_matrix[i][j]=pearsonr(time\_avg\_list[i],time\_avg\_list[j])[0] #90\*90 connectivity matrix  
 correlation\_matrix[i][j]=abs(correlation\_matrix[i][j])  
   
 plt.imshow(abs(correlation\_matrix))  
 plt.colorbar()  
 plt.clim(0,1)  
 plt.show() #양의 상관관계그래프 띄우기  
  
 binary\_matrix=correlation\_matrix  
 for i in range(90):  
 for j in range(90):  
 if binary\_matrix[i][j]!=0:  
 if abs(binary\_matrix[i][j])<0.1: #threshold 0.1  
 binary\_matrix[i][j]=0  
 else:  
 binary\_matrix[i][j] = 1 #binarized  
  
  
 coefficient\_vector=bct.clustering\_coef\_bu(binary\_matrix)  
 sum1=0  
 for i in range(len(coefficient\_vector)):  
 sum1+=coefficient\_vector[i]  
 coefficient=sum1/len(coefficient\_vector) #coefficient vector 평균내어 Cf 계산  
 print(coefficient)  
 charlength=bct.charpath(binary\_matrix) #Lf , lambda\_ 제외한 나머지값들 주석처리했음  
 print(charlength)  
  
 for i in range(20):  
 ran\_matrix=bct.randmio\_und(binary\_matrix,10)[0]  
 random\_coefficient\_vector=bct.clustering\_coef\_bu(ran\_matrix)  
 sum2=0  
 for i in range(len(random\_coefficient\_vector)):  
 sum2 += random\_coefficient\_vector[i]  
 ran\_coefficient = sum2 / len(random\_coefficient\_vector)  
 random\_list.append(ran\_coefficient) #random한 20개 coefficent list  
 ran\_length = bct.charpath(ran\_matrix)  
 random\_list2.append(ran\_length) #random한 20개 pathlength list  
  
 sum3=0  
  
 for i in range(len(random\_list)):  
 sum3+=random\_list[i]  
 ran\_coefficient\_avg=sum3/len(random\_list) #Cr  
 print(ran\_coefficient\_avg)  
 sum4=0  
 for i in range(len(random\_list2)):  
 sum4+=random\_list2[i]  
 ran\_length\_avg=sum4/len(random\_list2) #Lr  
 print(ran\_length\_avg)  
  
 small\_worldness=(coefficient/ran\_coefficient\_avg)/(charlength/ran\_length\_avg) #Sf= (Cf/Cr)/(Lf/Lr)  
 print(small\_worldness)