Командный практический тур

Во время финала Олимпиады КД НТИ по профилю «Нейротехнологии и когнитивные науки» вам предстоит построить и провести экспериментальное исследование по выявлению нейрофизиологических маркеров эмоционального состояния и типов эмоциональных реакций человека. Эмоции могут проявляться в мимике, голосе, одежде, поведении людей, а могут отражаться в процессах, которые недоступны прямому наблюдению других, а иногда и осознанию самого человека. Изучение биосигналов очень важно для нейротехнологий в целом, в частности, анализ биосигналов может применяться для понимания состояния людей, которые не могут его выразить.

Для этого вам будет необходимо решить 13 задач, при этом необходимо будет распределить между участниками команды следующие роли: Программист и инженер-физиолог (в таблице представлено рекомендуемое распределение по задачам). Один человек может взять на себя несколько ролей. В качестве испытуемого вы будете работать с Аватаром (с человеком, который находится на площадке с оборудованием).

С правилами по работе с аватаром вы можете ознакомиться в регламенте.

Максимальное количество баллов — 100 баллов. В таблице ниже представлен график открытия задач и количество баллов, которые можно будет получить за задачи.

Таблица III.2.1: Баллы за задачи, график и роли

№	Кол-во баллов	Время открытия	Срок сдачи	Кол-во попыток	Рекомендованные роли	Работа с аватарами
1	4	9:10 по МСК 23.02.2020	13:40 по МСК 23.02.2020	3	Программист	-
2	4	9:10 по МСК 23.02.2020	13:40 по МСК 23.02.2020	3	Программист	-
3	5	9:10 по МСК 23.02.2020	13:40 по МСК 23.02.2020	3	Программист	да
4	5	9:10 по МСК 23.02.2020	13:40 по МСК 23.02.2020	2	Инженер-Физиолог	да
5	5	9:10 по МСК 23.02.2020	13:40 по МСК 23.02.2020	1	Инженер-Физиолог	-
6	4	8:20 по МСК 24.02.2020	13:40 по МСК 23.02.2020	3	Программист	-
7	5	8:20 по МСК 24.02.2020	13:40 по МСК 23.02.2020	2	Программист, инженер-физиолог	да
8	6	8:20 по МСК 24.02.2020	13:40 по МСК 23.02.2020	1	Инженер-физиолог, программист	-
9	8	8:20 по МСК 24.02.2020	13:40 по МСК 23.02.2020	3	Программист	-
10	5	8:20 по МСК 25.02.2020	13:40 по МСК 25.02.2020	3	Программист	-
11	10	8:20 по МСК 25.02.2020	13:40 по МСК 25.02.2020	1	Программист, инженер-физиолог	да
12	19	8:20 по МСК 25.02.2020	13:40 по МСК 25.02.2020	3	Программист	-
13	20	8:20 по МСК 26.02.2020	11:00 по МСК 26.02.2020	3	Программист	-

Задачи можно выполнять параллельно и сдавать в любом порядке. Внимательно ознакомьтесь с критериями оценивания задач (указаны после каждой задачи), количеством попыток и сроками для сдачи заданий.

23 февраля

Задача III.2.1.1. (4 балла)

Поочередно вывести на экран компьютера в среде PsychoPy типовые стимулы: строчка текста, картинка (графический файл), простая фигура (круг или квадрат). Время показа: 3 секунды, пауза между стимулами (чистый серый экран) — 2 сек.

Количество строчек текста, картинок и простых фигур: по 10 шт. Всего — 30 стимулов. Порядок стимулов произвольный. Стимулы в последовательности предъявлений могут повторяться.

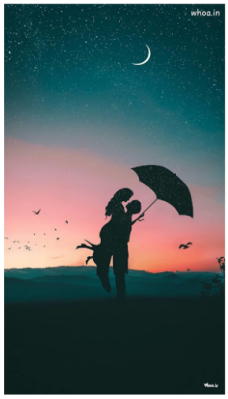
Формат ответа: скрипт в виде файла с расширением .py, реализующий заданную процедуру; графические файлы, которые будут выводиться на экран компьютера. Решение проверяется в PsychoPy v2020.2.10.

Критерии оценивания: 1 балл за вывод текста, 1 балл за вывод картинки (графического файла), 1 балл за вывод простой фигуры, 1 балл за корректное время.

Решение

```
from psychopy import visual, core  # import some libraries from PsychoPy
from time import time
from random import random, randint, shuffle

class WindowHandler:
    window = None
    visuals = {}

    def __init__(self, size, name, units=None):
        self.window = visual.Window(size, monitor=name, units=units)

    def addVisual(self, visualType, name, **args):
        visual = visualType(self.window)
        for key, val in args.items():
            exec(f'visual.{key} = {val}')
        self.visuals[name] = visual

    def deleteAllVisuals(self):
        self.visuals = {}

    def deleteVisual(self, name):
        self.visuals.pop(name)

    def update(self):
        for visual in self.visuals.values():
            visual.draw()
        self.window.update()

    def sleep(self, secs):
        begTime = time()
        while time() - begTime < secs:
            self.update()

    def waitState(self, stateToEval):
        while not eval(stateToEval):
            self.update()

    def close(self):
        self.window.close()


mywin = WindowHandler([800,600], name="testMonitor")

objects = []

for i in range(10):
```

```
47      objects.append([visual.ImageStim, {'image': f'\'{i}.jpg\''}])
48
49  for _ in range(10):
50      objects.append([visual.TextStim,
51                      {'text': '\'' + ''.join([chr(randint(65, 122)) for _ in
                        ↪  range(randint(1, 10))]) + '\'',
52                       'size':[1, 1]}])
53
54  for _ in range(10):
55      objects.append([visual.Rect, {'width': random()*1.8 + 0.1, 'height': random()*1.8
        ↪  + 0.1}])
56
57  shuffle(objects)
58
59  # print(objects)
60
61  for object in objects:
62      mywin.addVisual(object[0], 'last', **object[1])
63      mywin.sleep(3)
64      mywin.deleteVisual('last')
65      mywin.sleep(2)
66
67  mywin.close()
```

0 

1 

2 

3 

4 

5 

6 

7 

8 

9 

Задача III.2.1.2. (4 балла)

Сделать скрипт с использованием PsychoPy, который будет выводить последовательно десять изображений и содержать шкалу для оценки испытуемым этих изображений по любому удобному для вас признаку (оценки от 1 до 10 с шагом 1). Стимулы в последовательности предъявлений не повторяются.

Результатом работы скрипта должен быть словарь Python, где в качестве ключа будут имена файлов продемонстрированных изображений, а в качестве значений — оценки испытуемого. Словарь сохраняется в текстовый файл по завершении работы скрипта.

Формат ответа: скрипт в виде файла с расширением .py, реализующий заданную процедуру; графические файлы, которые будут выводиться на экран компьютера. Решение проверяется в PsychoPy v2020.2.10.

Критерии оценивания: 1 балл за вывод на экран изображений, 1 балл за вывод шкалы, 2 балла за сформированный словарь Python.

Решение

```
from psychopy import visual, core  # import some libraries from PsychoPy
from time import time
from random import random, randint, shuffle

class WindowHandler:
    window = None
    visuals = {}

    def __init__(self, size, name, units=None):
        self.window = visual.Window(size, monitor=name, units=units)

    def addVisual(self, visualType, name, **args):
        visual = visualType(self.window)
```

```
14          for key, val in args.items():
15              exec(f'visual.{key} = {val}')
16          self.visuals[name] = visual
17
18      def deleteAllVisuals(self):
19          self.visuals = {}
20
21      def deleteVisual(self, name):
22          self.visuals.pop(name)
23
24      def update(self):
25          for visual in self.visuals.values():
26              visual.draw()
27          self.window.update()
28
29      def sleep(self, secs):
30          begTime = time()
31          while time() - begTime < secs:
32              self.update()
33
34      def waitState(self, stateToEval):
35          while not eval(stateToEval):
36              self.update()
37
38      def close(self):
39          self.window.close()
40
41
42  mywin = WindowHandler([800,600], name="testMonitor")
43
44  objects = []
45
46  for i in range(10):
47      objects.append([visual.ImageStim, {'image': f'\'{i}.jpg\''}])
48
49  for _ in range(10):
50      objects.append([visual.TextStim,
51                      {'text': '\'' + ''.join([chr(randint(65, 122)) for _ in
                         ↪  range(randint(1, 10))]) + '\'',
52                       'size':[1, 1]}])
53
54  for _ in range(10):
55      objects.append([visual.Rect, {'width': random()*1.8 + 0.1, 'height': random()*1.8
        ↪  + 0.1}])
56
57  shuffle(objects)
58
59  # print(objects)
60
61  for object in objects:
62      mywin.addVisual(object[0], 'last', **object[1])
63      mywin.sleep(3)
64      mywin.deleteVisual('last')
65      mywin.sleep(2)
66
67  mywin.close()
```

Задача III.2.1.3. (5 баллов)

Изучить интерфейс программы-визуализатора. Изучение осуществляется без использования платформы Arduino (которая доступна только аватарам), но это нужно для формирования правильных указаний аватарам. Далее, совместно с аватарами осуществить запуск установки для сбора сигналов (без надевания электродной системы на испытуемого). Сделать скриншоты окна программы-визуализатора и обозначить на этих скриншотах:

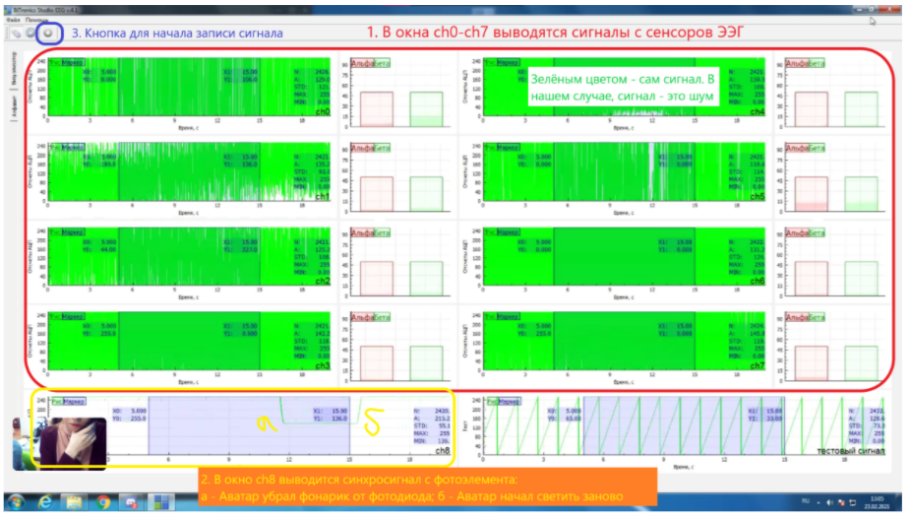
- поля, где должны выводиться сигналы ЭЭГ;
- поле, где выводится синхросигнал от фотодиода;
- запись регистрируемых сигналов средствами программы-визуализатора в файл (длительность записи — не менее 10 сек и не более 20 сек).

Формат ответа: скриншоты с обозначением полей программы-визуализатора с выводом тех или иных сигналов; запись сигнала в файле (средствами программы-визуализатора).

Критерии оценивания: корректные скриншоты полей для визуализации сигнала ЭЭГ (с примером сигнала в них) — 2 балла; корректный скриншот поля для визуализации синхросигнала (с примером сигнала в них) — 2 балла; корректная запись сигналов — 1 балл.

Решение

https://drive.google.com/file/d/1TOSfv4FqKBNgD7qzP7ao1OicVgdAqmIR/view?usp=sharing



Задача III.2.1.4. (5 баллов)

Выполнить функциональные тесты и продемонстрировать в ЭЭГ:

- миографические артефакты;
- глазодвигательные (моргательные) артефакты.

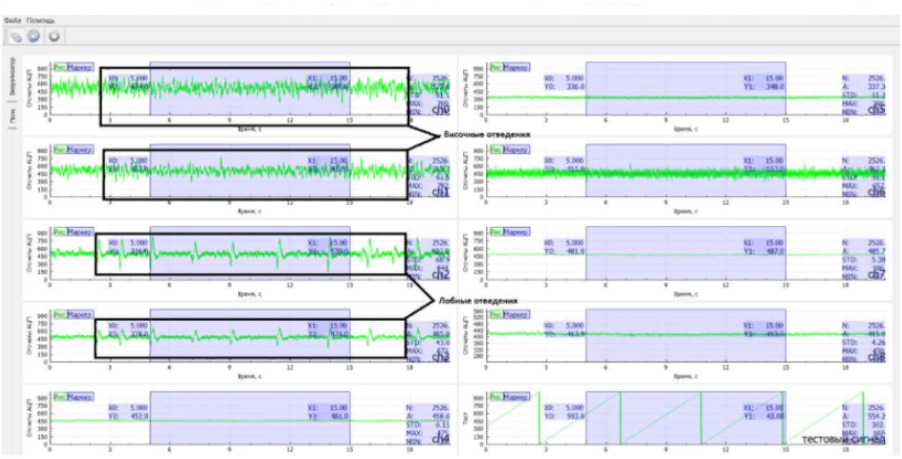
Выполнить функциональный тест на альфа-ритм, при появлении альфа-ритма в ЭЭГ сделать скриншот соответствующего участка.

Сделать скриншоты с характерными артефактами. Артефакты подписать. Если артефакты проявляются в разных каналах по-разному, пояснить причину этих различий. Дать текстовое описание функционального теста на альфа-ритм.

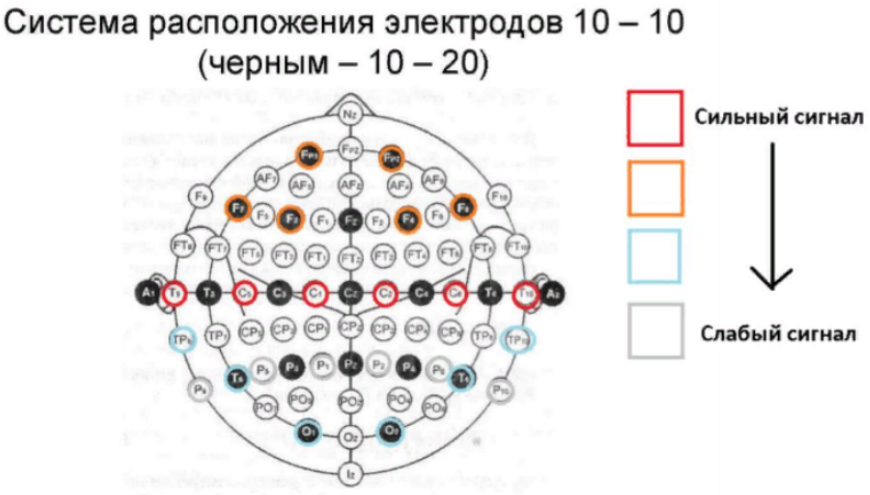
Формат ответа: скриншоты с выделенными и подписанными артефактами. Текстовое пояснение в формате .doc (либо его аналог разной степени выраженности артефактов в каналах ЭЭГ. Текстовое описание функционального теста на альфа-ритм.

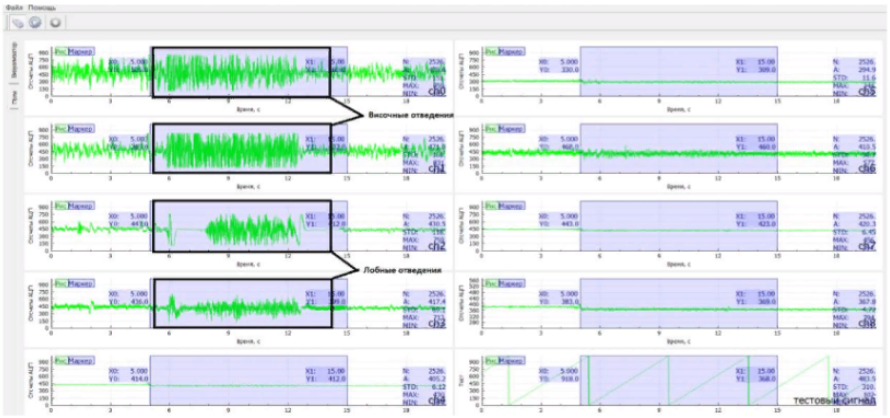
Решение

Движение глаз во время записи ЭЭГ вызвало артефакт из-за электрического поля, создаваемого лобными и круговыми мышцами при их напряжении (в данном случае при процессе моргания), которое регистрировали датчики.Такой тип артефакта называется глазодвигательным. Таким образом, канал, в котором будет зарегистрирован такой артефакт, прямо зависит от расположения датчиков этого канала на голове пациента. На скриншоте отведения (FP1- F3-F7) и (FP2-F4-F8) обозначены лобными отведениями,и т.к они располагаются на лобной части черепа, вблизи круговых и лобных мышц, амплитуда артефакта высока и является ярко выраженной. Артефакты меньшей амплитуды регистрировались также на височных отведениях (C1-C5-T9) и (C2-C6-T10), это является следствием большей удаленности от круговых и лобных мышц, чем лобные отведения. В прямоугольниках находятся сигналы глазодвигательного артефакта, по мере удаления расположения датчиков от лобной части головы, амплитуда артефакта значительно снижается. На втором изображении представлена используемая конфигурация отведений.

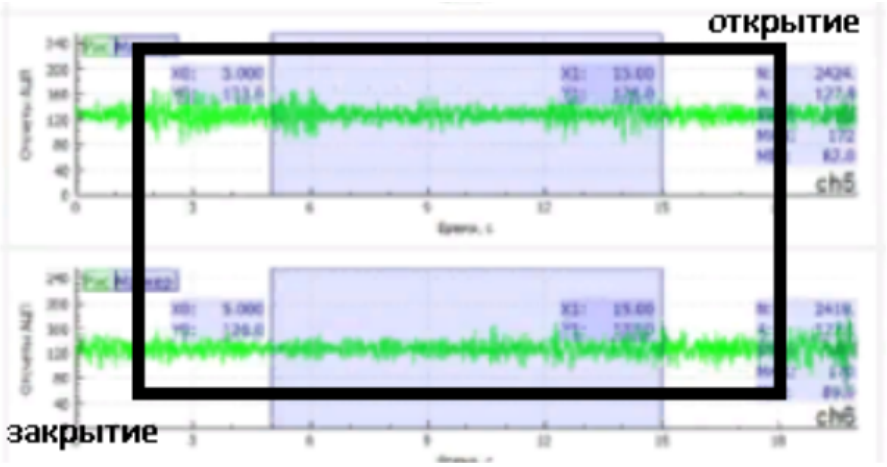


Данный ЭМГ артефакт появляется при мышечном напряжении жевательных мышц, а в свою очередь при напряжении мышцы к ней поступает электрический сигнал, потенциал которого регистрируют отведения. Такой тип артефакта называется миогенным. Таким образом, канал, в котором будет зарегистрирован такой артефакт, прямо зависит от расположения датчиков этого канала на голове пациента. Поскольку мышцы не целиком покрывают череп, некоторые датчики будут находиться на большем расстоянии от них, что даст более низкую амплитуду артефакта, по сравнению с теми датчиками, что находятся рядом с мышцами. На скриншоте отведения (C1-C5-T7) и (C2-C6-T8) обозначены височными отведениями,и т.к они располагаются на височной части черепа, вблизи жевательных мышц, амплитуда артефакта высока и является ярко выраженной. Артефакты меньшей амплитуды регистрировались также на лобных (FP1- F3-F7) и (FP2-F4-F8) и затылочных отведениях (O1-T5-TP9),(O2-T6-TP10),(P1-P5-P9),(P2-P6-P9), это является следствием большей удаленности от жевательных мышц, чем височные отведения. В выделенных прямоугольниках находится ЭМГ артефакт челюстных мышц, по мере удаления датчиков от источника сигнала (челюстных мышц) можно заметить снижение амплитуды сигнала.





Альфа-ритм (8-12Гц) выявляется, когда исследуемый закрывает глаза или находится в расслабленном состоянии. Замечено, что альфа-ритм затухает, если открыть глаза, а также в состоянии умственного напряжения. Лучше всего выражен в затылочных отведениях(ch4,5,6,7). При закрытии глаз ритм возникает, при открытии изчезает



Критерии оценивания: По 1 баллу за демонстрацию каждого артефакта. По 1 баллу за верное пояснение различной степени выраженности. 1 балл за описание функционального теста на альфа-ритм.

Задача III.2.1.5. (5 баллов)

Предложите методы (способы или приемы) направленной модификации эмоционального состояния человека. Для каждого метода опишите по два примера (можно больше) его использования для модификации эмоционального состояния в различных направлениях. Например, один пример — модификация в более радостное состояние, другой — в более грустное или в состояние страха. Можете предложить более двух примеров. Максимальное количество методов — 10.

По возможности рекомендуется подкрепить изложение ссылками на источники (ссылки на статьи в интернете, на публикации в журналах, на главы в книгах и пр.). Отсутствие таких ссылок не будет приводить к штрафным баллам, однако при возникновении спорных вопросов может быть полезным.

Формат ответа: текстовое описание методов и примеров их реализации для различных вариантов модификации эмоционального состояния.

Критерии оценивания: По 0,5 балла за каждый метод. Методы без примеров их разнонаправленной реализации не засчитываются.

Решение

Показ видеоролика	1. Показ видеоролика, где дедушка остается один в канун Рождества 2. Показ видео со смехом ребенка	https://psyjournals.ru/exp/2018/n2/Pankratova_Lusin.shtml
Прослушивание аудиозаписей	1. Прослушивание грустной мелодии 2. Прослушивание резвой мелодии	
Показ картинок	1. Показ картинки с котиком 2. Показ картинки у могилы	
Моделирование реальной ситуации — ролевая игра с участием помощника экспериментатора	1. Помощник экспериментатора заходит в комнату и вступает в конфликт с испытуемым 2. Начать рассказывать о недавнем путешествии и попросить рассказа о своем	
Метода автобиографических воспоминаний. Испытуемые должны были вспомнить и описать события, произошедшие с ними в течение последних нескольких лет, которые вызвали в них наиболее сильные положительные или отрицательные эмоции.	1. Попросить вспомнить самый добрый момент из его жизни 2. Попросить вспомнить самый жуткий момент из жизни	
Индукция эмоций посредством подарка	1. Подарить подарок в плохой упаковке, но очень хороший 2. Подарить подарок в яркой упаковке, но плохой	
Индукции с помощью изменения выражения лица	1. Показать счастливое лицо и попросить повторить 2. Показать грустное лицо и попросить повторить	
Индукции посредством социального взаимодействия	1. Рассказать анекдот 2. Рассказать грустную историю	
Индукции посредством еды	1. Дать попробовать что-то горькое 2. Дать попробовать что-то сладкое	
Индукции посредством запаха	1. Дать понюхать дуриан 2. Дать понюхать духи	

24 февраля

Задача III.2.2.1. (4 балла)

Доработайте скрипт из задачи 2 (демонстрация картинок и ответы испытуемого) таким образом, чтобы создать область на стимульном экране, которая поможет Вам осуществить разметку ЭЭГ-сигнала при помощи фотодатчика (фотодиода). Метки в потоке данных должны указывать на эпизоды демонстрации стимульного изображения для обеспечения возможности дальнейшей работы с данными эпизодами.

Формат ответа: скрипт в виде файла с расширением .py, реализующий заданную процедуру; набор графических файлов. Решение проверяется в PsychoPy v2020.2.10.

Критерии оценивания: 4 балла за доработку скрипта областью для разметки потока данных при помощи фотодиода.

Решение

```
from psychopy import visual
from time import time

win = visual.Window([1000, 800], monitor="testMonitor", units="height")

 # , size=[0.6, 0.6]
rect = visual.Rect(win, width=0.1, height=0.1, pos=[0.5, -0.4])
paths = [f"{i}.jpg" for i in range(10)]
i = 0
result = {}

begshow = time()
rect.fillColor = 'Black'
while time() - begshow < 2:
    rect.draw()
    win.flip()


for image_path in paths:
    ratingScale = visual.RatingScale(win, low=1, high=10, acceptPreText='Оцените',
    ↪  stretch=2, pos=[0, -0.6])
    image = visual.ImageStim(win, pos=[0, 0.1], image=image_path)
    image.size /= image.size[1]/0.5

    begshow = time()
    while ratingScale.noResponse:
        if time() - begshow < 0.2:
            rect.fillColor = 'White'
        else:
            rect.fillColor = 'Black'
        ratingScale.draw()
        image.draw()
        rect.draw()
        win.flip()
    rating = ratingScale.getRating()
    result[image_path] = rating

```

```
with open('result.txt', 'w') as file:
    file.write(str(result))
```

Задача III.2.2.2. (5 баллов)

Выполните запись ЭЭГ на испытуемом одновременно с работой доработанного в задании № 6 скрипта. Убедитесь в корректности разметки потока ЭЭГ-данных. В качестве ответа представьте скрипт, который будет анализировать канал синхрометок в записанных данных ЭЭГ и в качестве результата своей работы: (1) возвращать количество эпизодов демонстрации стимульных изображений и (2) визуализировать данные ЭЭГ и канал синхрометок.

Формат ответа: скрипт в виде файла с расширением .py, реализующий заданную процедуру; файл с данными ЭЭГ, записанными в ходе решения задачи и которые обрабатываются скриптом.

Критерии оценивания: 3 балла за корректную обработку, 2 балла за визуализацию. Решение проверяется в среде Anaconda 5.2. Для среды Anaconda 5.2 возможна установка дополнительных пакетов расширений при условии использования стандартного инсталлятора для данных пакетов (conda install -c name_of_package).

Решение

```
#!/usr/bin/env python
# coding: utf-8

# In[75]:


import numpy as np

import matplotlib
import matplotlib.pyplot as plt

matplotlib.style.use('seaborn-deep')
# get_ipython().run_line_magic('matplotlib', 'inline')


# In[76]:


eeg_mx = [[], [], [], [], [], [], [], [], [], []]
# print(eeg_mx)
file = open('24_2_2021_12_49_20.txt', 'r')
for i in file:
    a = list(map(int, i.split()))
#     print(len(a))
    for i in range(len(a)):
        eeg_mx[i].append(a[i])

file.close()


# In[77]:


```

```
count = 0
tr = False
for i in eeg_mx[8]:
#     print(i)
    if i > 150 and tr == False:
        count += 1
        tr = True
    elif i <= 150:
        tr = False
print(count)


# In[78]:


for i in range(10):
    plt.figure(figsize=(100, 10))
    k = i
    plt.plot(np.arange(len(eeg_mx[k])), eeg_mx[k])
    plt.show()
```

Задача III.2.2.3. (6 баллов)

Ознакомьтесь со статьей по ссылке (`https://drive.google.com/file/d/1MlNh8FI 1mzUuVOQbA638itE5TUcTa0az/vie w?usp=sharing`). Ответьте на следующие вопросы:

1. Активация в каких областях мозга (зонах, ядрах или отделах) и с какой стороны (справа или слева) связана с позитивными эмоциями?
2. Выраженное проявление позитивных или негативных эмоций отражается в спектральных характеристиках ЭЭГ. В каком диапазоне частот проявляются эти изменения? Что происходит со спектральной мощностью в данном диапазоне в лобных долях левого и правого полушария при негативных эмоциях?
3. Какой уровень среднего количества ошибок достигнут для классификаторов на основе LDA и CSP?

Формат ответа: текстовое описание параметров ЭЭГ-сигнала на основе анализа литературы, в формате .doc или аналогичных свободных форматах. В текстовом описании указать, к какому вопросу относится тот или иной ответ.

Критерии оценивания: по 2 балла за каждый правильный ответ.

Решение

1. С позитивными эмоциями связана активация левой фронтальной области лобной доли.
2. Эти изменения проявляются в диапазоне альфа-ритма (8-12Гц) . Чем более активирована зона, тем более низкие значения спектральной мощности. Соответственно, при негативных эмоциях в правом полушарии происходит понижение мощности, а в левом, наоборот, повышение.
3. LDA — 44.1 ± 5.2 (%)
   CSP — 46.9 ± 6.8(%)

Задача III.2.2.4. (8 баллов)

Обработайте ЭЭГ, полученную в задаче № 7. Для всех сочетаний стимулов и каналов ЭЭГ вычислите следующие метрики:

- максимальный размах в сигнале ЭЭГ;
- минимальный размах в сигнале ЭЭГ;
- максимальный пик-фактор сигнала ЭЭГ;
- минимальный пик-фактор сигнала ЭЭГ;
- максимальный размах спектра ЭЭГ;
- минимальный размах спектра ЭЭГ;
- максимальная асимметрия размаха спектра ЭЭГ между полушариями (для соответствующих друг другу пар каналов);
- максимальная асимметрия (разность) пик-фактора спектра ЭЭГ между полушариями (для соответствующих друг другу пар
- каналов).

В качестве ответа представьте скрипт, вычисляющий вышеуказанные метрики и выбирающие такие сочетания стимула и канала, для которых характерно максимальное значение соответствующей метрики.

Формат ответа: скрипт в виде файла с расширением .py, вычисляющий заданные характеристики; ЭЭГ-данные, полученные командой участников в задаче № 7, на которых вычисляются заданные характеристики.

Критерии оценивания: по 1 баллу за каждое верное сочетание стимула и канала, для которого характерно максимальное (минимальное в случае поиска минимального значения) значение той или иной метрики. Решение проверяется в среде Anaconda 5.2. Для среды Anaconda 5.2 возможна установка дополнительных пакетов расширений при условии использования стандартного инсталлятора для данных пакетов (conda install -c name_of_package).

25 февраля

Задача III.2.3.1. (5 баллов)

Сформируйте эксперимент по исследованию эмоциональной реакции на стимульные изображения. В качестве стимулов используйте изображения по ссылке (`https://drive.google.com/drive/folders/15fcRIBdJdFtQVxD74gLUFka-yfyNwx JT?usp=sharing`). Вы можете использовать все изображения или только часть на Ваше усмотрение. Оценка изображений по 10-бальной шкале, где

1 — очень неприятно (очень не нравится), 5 — нейтрально, 10 — очень приятно (очень нравится). Инструкцию для испытуемого (одного из аватаров) установите перед демонстрацией стимульных изображений. Переход к демонстрации изображений и прохождению исследования — по нажатию любой клавиши на клавиатуре.

Сохраняется запись ЭЭГ и ответы испытуемого.

Формат ответа: скрипт в виде файла с расширением .py, реализующий необходимую процедуру.

Критерии оценивания: 3 балла за вывод необходимых изображений и шкалы для ответов, 2 балла за сохранение ответов испытуемого.

Решение

Ответы испытуемого:

NAME	RATING
ns6.jpg	5
neg_5.jpg	5
n_2.jpg	
5 neg_1.jpg	5
n_5.jpg	5
ns7.jpg	5
neg_3.jpg	5
neg_9.jpg	5
n_1.jpg	5
ns19.jpg	5
pos_2.jpg	5
pos_10.jpg	5
pos_6.jpg	5
ns15.jpg	5
n_3.jpg	5
pos_5.jpg	5

Задача III.2.3.2. (10 баллов)

Проведите эксперимент на основе дизайна задачи № 10. При необходимости дополните текстовую инструкцию на экране устными пояснениями. Постарайтесь работать с аватарами таким образом, чтобы обеспечить наиболее естественные и хорошо выраженные реакции на стимульные изображения.

Если Ваш эксперимент будет достаточно продолжительным (более 5-10 минут), убедитесь, что Ваш испытуемый сможет нормально работать необходимое время. Также постарайтесь обеспечить стабильное электропитание установки, регистрирующей ЭЭГ. Для этого рекомендуем перед непосредственным началом эксперимента установить полностью заряженный источник питания (спросить у технического специалиста на площадке).

Формат ответа: файл с ЭЭГ-данными; текстовый файл, содержащий ответы испытуемого по каждому предъявленному изображению.

Критерии оценивания: 5 баллов за сохранение ЭЭГ в результате эксперимента, 5 баллов за сохранение ответов испытуемого.

Решение

Ответы испытуемого:

NAME	RATING	3.0
neg_9.jpg		
pos_4.jpg		10.0
n_3.jpg		9.0
pos_7.jpg		9.0
ns13.jpg		6.0
pos_9.JPG		9.0
neg_8.jpg		5.0
neg_5.jpg		3.0
pos_11.jpg		9.0
ns20.jpg		6.0
n_1.jpg		6.0
neg_12.jpg		4.0
pos_8.jpg		10.0
pos_12.jpg		10.0
pos_3.jpg		10.0
neg_7.jpg		4.0
n_5.jpg		6.0
ns3.jpg		6.0
pos_1.jpg		10.0
neg_3.jpg		3.0
n_2.jpg		9.0
ns8.jpg		5.0
ns16.jpg		8.0
ns2.jpg		7.0
ns11.jpg		7.0
ns5.jpg		6.0
ns4.jpg		5.0
neg_11.jpg		4.0
neg_1.jpg		3.0
ns19.jpg		3.0
ns6.jpg		5.0
pos_2.jpg		9.0
ns9.jpg		5.0
pos_10.jpg		10.0
ns1.jpg		5.0
ns12.jpg		5.0
ns17.jpg		7.0
neg_4.jpg		2.0
neg_2.jpg		3.0
neg_10.jpg		3.0
ns18.jpg		6.0
n_6.jpg		7.0
ns14.jpg		6.0
n_4.jpg		6.0
neg_6.jpg		2.0
pos_5.jpg		8.0
pos_6.jpg		9.0
ns10.jpg		5.0
ns7.jpg		5.0
ns15.jpg		7.0

Данные ЭЭГ:

https://drive.google.com/file/d/1DtqaCrcCezbnY2tpao4WXkOVdw5Fki4o/view?usp=sharing

Задача III.2.3.3. (19 баллов)

Постройте модель распознавания ответов от испытуемых «нравится/ не нравится» на основе собранных ЭЭГ-данных. Проверьте эффективность полученной модели в кросс-валидации: метрика — «balanced_accuracy», размер тестовой выборки — 20%, сплиттер — ShuffleSplit, random_state = 2, n_splits=100.

Формат ответа: критерий определения случаев «нравится» и «не нравится»; скрипт, строящий модель и выполняющий тест кросс-валидации.

Критерии оценивания: 1 балл за критерий случаев «нравится» и «не нравится»; балл за кросс-валидацию вычисляется по формуле $(R - 0.5) \cdot 2 \cdot (M - 1)$, где R — результат теста кросс-валидации в диапазоне от 0 до 1 с точностью до двух знаков после запятой, M — максимальный балл за данную задачу, при отрицательном значении балла за кросс-валидацию (при R меньше 0.5) команда получает 0 баллов за кросс-валидацию в данной задаче; баллы за критерий и за кросс-валидацию суммируются.

Решение проверяется в среде Anaconda 5.2. Для среды Anaconda 5.2 возможна установка дополнительных пакетов расширений при условии использования стандартного инсталлятора для данных пакетов (conda install -c name_of_package).

Решение

Для решения данной задачи используются следующие пакеты: scipy, matplotlib, numpy, sklearn, statsmodels Критерий: спектральная мощность альфа-ритма в лобных долях левого полушария. Диапазон частот проявления изменений в спектральных характеристиках ЭЭГ от 8 до 12 Гц (альфа-ритм). Спектральная мощность в лобных долях левого полушария при негативных эмоциях высокая или растет (низкая/падающая активность связана с высокой/растущей альфа спектральной мощностью). Спектральная мощность в лобных долях правого полушария при негативных эмоциях низкая или падает (высокая/растущая активность связана с низкой/падающей альфа спектральной мощностью). Спектральная мощность в лобных долях левого полушария при позитивных эмоциях низкая или падает (высокая/растущая активность связана с низкой/падающей альфа спектральной мощностью). Спектральная мощность в лобных долях правого полушария при позитивных эмоциях высокая или растет (низкая/падающая активность связана с высокой/растущей альфа спектральной мощностью)

Код для обработки данных:

```
import numpy as np
import matplotlib.pyplot as plt
import scipy.signal as sgn

f = open('25_2_2021_10_46_22.dat','r')
data_s = f.read()
f.close()
```

```
8
9  data = np.array([[int(i) for i in r.split() ]for r in data_s.split('\n') if r])
10 datach = data.T
11
12 filt1S = sgn.medfilt(datach[8],kernel_size= 64+1)
13 filt1Sbinary = filt1S > (filt1S.mean())
14 filt1Sbinary_f = sgn.medfilt(filt1Sbinary,kernel_size=64+1)
15 filt1Sbinary_f_d = np.diff(filt1Sbinary_f)
16
17 lasti = -1
18 dsis = []
19 for i in range(len(filt1Sbinary_f_d)):
20     if filt1Sbinary_f_d[i] > 0:
21         if lasti > 0:
22             dsis.append([datach[ch][lasti:i] for ch in range(8)])
23         lasti = i
24
25 DL = 21
26 DECIMATE_Q = 4
27 SLICE_SIZE = int(256//4*1.5)
28 def preproc(s):
29     sos = sgn.butter(8,[1,127],btype='bandpass',fs=256,output='sos')
30     s = sgn.sosfiltfilt(sos,s)
31     s = sgn.decimate(s,DECIMATE_Q)
32     s= s[:SLICE_SIZE]
33     s = np.array(s)
34     s = (s - s.min()) / (np.max(s) - np.min(s)) - 0.5
35     return s
36 data = [[preproc(channal) for channal in dt] for dt in dsis]
37
38
39 def max_razm(data):
40     ans = []
41     for chanal in data:
42         n = len(chanal)
43         ans.append(max(list(map(lambda x: abs(chanal[x]-chanal[x+1]),range(n-1)))))
44     return ans
45 def min_razm(data):
46     ans = []
47     for chanal in data:
48         n = len(chanal)
49         ans.append(min(list(map(lambda x: abs(chanal[x]-chanal[x+1]),range(n-1)))))
50     return ans
51 def max_razm_fft(data):
52     f = np.abs(np.fft.rfft(data))
53     return max_razm(f)
54 def min_razm_fft(data):
55     f = np.abs(np.fft.rfft(data))
56     return min_razm(f)
57 def max_assim_razm(data):
58     ans = []
59     for i in range(0,8,2):
60         r1 = np.array(list(map(lambda x: abs(data[i][x]-data[i][x+1]),
           ↪  range(data[i].shape[0]-1))))
61         r2 = np.array(list(map(lambda x: abs(data[i+1][x]-data[i+1][x+1]),
           ↪  range(data[i+1].shape[0]-1))))
62         ans.append(max(np.abs(r1-r2)))
63     return ans
64 def max_pik(data):
65     ans = []
```

```
    for chanal in data:
        m = max(np.abs(chanal))
        t = (sum(list(map(lambda x:x**2,chanal)))/chanal.shape[0])**0.5
        ans.append(m/t)
    return ans
def min_pik(data):
    ans = []
    for chanal in data:
        m = max(np.abs(chanal))
        t = (sum(list(map(lambda x:x**2,chanal)))/chanal.shape[0])**0.5
        ans.append(m/t)
    return ans
def max_assim_pic(data):
    ans = []
    for i in range(0,8,2):
        f1 = np.abs(np.fft.rfft(data[i]))
        f2 = np.abs(np.fft.rfft(data[i+1]))
        pik1 = (sum(list(map(lambda x:x**2,f1)))/f1.shape[0])**0.5
        pik2 = (sum(list(map(lambda x:x**2,f2)))/f2.shape[0])**0.5
        ans.append(np.abs(pik1-pik2))
    return ans
def MSR(data):
    ans = []
    for ch in data:
        ans += [np.sqrt(np.sum(np.array(ch)**2) / len(ch))]
    return ans
def var(data):
    ans = []
    for ch in data:
        v = np.std(ch)/np.mean(ch)
        ans.append(v)
    return ans
def only_EEG(data):
    ans= []
    for ch in data:
        ans += list(ch)
    return ans
def only_FFT(data):
    ans = []
    for ch in data:
        ans += list(np.abs(np.fft.rfft(ch))[1:])
    return ans
def wawes(data):
    ans = []
    delta = 1 * 64*2//SLICE_SIZE
    teta = 4* 64*2//SLICE_SIZE
    alpha = 8 * 64*2//SLICE_SIZE
    beta = 12 * 64*2//SLICE_SIZE
    gamma = 30 * 64*2//SLICE_SIZE
    for ch in data:
        f = np.abs(np.fft.rfft(ch))
        ans += [
            np.sum(f[delta:teta]),
            np.sum(f[teta:alpha]),
            np.sum(f[alpha:beta]),
            np.sum(f[beta:gamma]),
            np.sum(f[gamma:]),
        ]
    return ans

```

```
126 from statsmodels.tsa.ar_model import AutoReg
127 from statsmodels.tsa.api import SARIMAX
128 
129 def getFeatures(X):
130     #первый признак, среднее абсолютная величина
131     a = np.abs(X).sum()/X.size
132     #второй признак, средная абсолютная производная (дифференциал) (x[i] - x[i-1])
133     b = np.abs(np.diff(X)).sum()/X.size
134     #третий признак, среднее колличество пересечений нуля, с порогом
135     #возможно я выбрал не оптимальное значение
136 
137     #                                                  порог
138     c = np.sum((X[:-1] * -X[1:])[abs(X[:-1] - X[1:]) > 0.1] > 0)/X.size
139 
140     #четвертый признак, среднее колличество, так сказать, пиков, с порогом на его
        ↪  высоту
141     #(np.diff(X[:-1]) * -np.diff(X[1:]) - это число положительно, если у пика есть
        ↪  центр,
142     # а его основания лежат строго по одну сторону
143     # >5 - это порог, его я также мог установить не оптимальным образом
144     d = np.sum((np.diff(X[:-1]) * -np.diff(X[1:]))>0.1)/X.size
145     # уже писал: используем модель авторегрессии, после ее обучения, используем ее
        ↪  веса, как признаки
146     # колличество весов = lags+1, плюс один, это константа, как у графика: y = kx + C
147     e = list(AutoReg(X, lags=10,old_names=False).fit().params)
148     #возращаем массив признаков
149     return np.array([a,b,c,d] + e)
150 
151 Xs = []
152 for d in data:
153     fich = []
154     fich+=list(max_razm(d))
155     fich+= list(max_razm_fft(d))
156     fich+=list(max_pik(d))
157     fich+=list(max_assim_pic(d))
158     fich+= list(max_assim_razm(d))
159     fich+=list(wawes(d))
160 #    fich+=list(MSR(d))
161 #    fich+=list(var(d))
162 #    for i in range(8):
163 #        fich+=list(getFeatures(d[i]))
164     Xs.append(fich)
165 Xs = np.array(Xs)
166 
167 f = open('y.txt','r')
168 ys = [r.split() for r in f.read().split('\n')[1:] if r]
169 f.close()
170 labelDecode = [-1,0,0,0,0,0,1,1,1,1,1]
171 ys = [labelDecode[int(float(d))] for (n,d) in ys]
172 ys = np.array(ys)
173 
174 
175 from sklearn.preprocessing import robust_scale
176 from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
177 
178 Xs1 = robust_scale(Xs)
179 
180 lda = LinearDiscriminantAnalysis(n_components=1)
181 ps = lda.fit_transform(Xs1,ys)
182 
```

```
ps1 = ps[:DL][ys[:DL] == 1]
ps2 = ps[DL:][ys[DL:] == 0]
ps = list(ps1) + list(ps2)
ys1 = [1] * 15 + [0] * 15

from sklearn.svm import SVC
from sklearn.model_selection import GridSearchCV
from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import
↪  train_test_split,ShuffleSplit,cross_validate,cross_val_score
from sklearn.feature_selection import VarianceThreshold
from sklearn.linear_model import LogisticRegression,Perceptron,RidgeClassifierCV


svc = SVC(C=1000,kernel='linear')
# svc = LinearDiscriminantAnalysis()
# svc = MLPClassifier(activation='tanh',learning_rate='adaptive',tol=0.1)



res =
↪  cross_val_score(RidgeClassifierCV(),ps,ys1,cv=ShuffleSplit(test_size=0.2,n_splits=100),
n_jobs=1,scoring='balanced_accuracy')
print((res.mean()-0.5)*2*18)
#print(res)
# svc.fit(Xs_train,Ys_train)
# svc.predict(xs_test),ys_test
```

26 февраля

Задача III.2.4.1. (20 баллов)

Доработайте модель распознавания ответов от испытуемых «нравится/ нейтрально/ не нравится» на основе собранных ЭЭГ-данных в задаче № 11. Проверьте эффективность полученной модели в кросс-валидации: метрика — «balanced_accuracy», размер тестовой выборки — 20%, сплиттер — ShuffleSplit, random_state = 2, n_splits=100.

Опишите процедуру выделения трех классов событий: какую информацию (переменные) для этого использовали, как на основе ее выделяли три типа событий. Это будет критерием для определения случаев «нравится», «нейтрально» и «не нравится».

Формат ответа: текстовый файл с описанием критерия определения случаев «нравится», «нейтрально» и «не нравится»; данные, на основе которых строится модель (были записаны в задаче № 11); скрипт, строящий модель и выполняющий тест кросс-валидации (скрипт загружает данные из той же папки, где находится он сам).

Критерии оценивания: 1 балл за критерий случаев «нравится», «нейтрально» и «не нравится»; балл за кросс-валидацию вычисляется по формуле $(R - 0.5) \cdot 2 \cdot (M - 1)$, где R — результат теста кросс-валидации в диапазоне от 0 до 1 с точностью до двух знаков после запятой, — максимальный балл за данную задачу, при отрицательном значении балла за кросс-валидацию (при R меньше 0.5) команда получает 0 баллов за кросс-валидацию в данной задаче; баллы за критерий и за кросс-валидацию суммируются.

Решение проверяется в среде Anaconda 5.2. Для среды Anaconda 5.2 возможна установка дополнительных пакетов расширений при условии использования стандартного инсталлятора для данных пакетов (conda install -c name_of_package).

Решение

Критерий: спектральная мощность альфа- и бета-ритмов в лобных долях правого полушария.

Диапазон частот проявления изменений в спектральных характеристиках ЭЭГ от 8 до 12 Гц (альфа-ритм). Спектральная мощность в лобных долях левого полушария при негативных эмоциях высокая или растет (низкая/падающая активность связана с высокой/растущей альфа спектральной мощностью). Спектральная мощность в лобных долях правого полушария при негативных эмоциях низкая или падает (высокая/растущая активность связана с низкой/падающей альфа спектральной мощностью). Спектральная мощность в лобных долях левого полушария при позитивных эмоциях низкая или падает (высокая/растущая активность связана с низкой/падающей альфа спектральной мощностью). Спектральная мощность в лобных долях правого полушария при позитивных эмоциях высокая или растет (низкая/падающая активность связана с высокой/растущей альфа спектральной мощностью).

Позитивные картинки были отмечены значительно более высокой валентностью, чем нейтральные и негативные. Значение валентности для нейтральных картинок было значительно выше, чем для негативных картинок. Возбуждение было значительно выше при стимуляции негативными и позитивными картинками, чем при стимуляции нейтрально окрашенными. Негативные картинки в среднем вызвали большее возбуждение, чем позитивные.

Валентность — аффективное качество, заключающееся в субъективной привлекательности или непривлекательности для человека предметов, событий или ситуаций. Термин также используется для описания чувственного тона ощущений, аффектов, поведения. «Нравится» — позитивные эмоции, «не нравится» — негативные эмоции, «нейтрально» — нейтральные эмоции, относительное спокойствие.

Код для обработки данных

```
import numpy as np
import matplotlib.pyplot as plt
import scipy.signal as sgn
from sklearn.model_selection import
↪  train_test_split,ShuffleSplit,cross_validate,cross_val_score
from sklearn.feature_selection import VarianceThreshold,RFE
from sklearn.preprocessing import robust_scale
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.linear_model import LogisticRegression,Perceptron,RidgeClassifierCV
from sklearn.ensemble import StackingClassifier
from statsmodels.tsa.ar_model import AutoReg
from statsmodels.tsa.api import SARIMAX

f = open('25_2_2021_10_46_22.dat','r')
data_s = f.read()
f.close()

```

```
data = np.array([[int(i) for i in r.split() ]for r in data_s.split('\n') if r])
datach = data.T

filt1S = sgn.medfilt(datach[8],kernel_size= 64+1)
filt1Sbinary = filt1S > (filt1S.mean())
filt1Sbinary_f = sgn.medfilt(filt1Sbinary,kernel_size=64+1)
filt1Sbinary_f_d = np.diff(filt1Sbinary_f)

lasti = -1
dsis = []
for i in range(len(filt1Sbinary_f_d)):
    if filt1Sbinary_f_d[i] > 0:
        if lasti > 0:
            dsis.append([datach[ch][lasti:i] for ch in range(8)])
        lasti = i

DECIMATE_Q = 4
SLICE_SIZE = int(256//4*1.5)
def preproc(s):
    sos = sgn.butter(8,[1,127],btype='bandpass',fs=256,output='sos')
    s = sgn.sosfiltfilt(sos,s)
    s = sgn.decimate(s,DECIMATE_Q)
    s= s[:SLICE_SIZE]
    s = np.array(s)
    s = (s - s.min()) / (np.max(s) - np.min(s)) - 0.5
    return s
data = [[preproc(channal) for channal in dt] for dt in dsis]

def max_razm(data):
    ans = []
    for chanal in data:
        n = len(chanal)
        ans.append(max(list(map(lambda x: abs(chanal[x]-chanal[x+1]),range(n-1)))))
    return ans
def min_razm(data):
    ans = []
    for chanal in data:
        n = len(chanal)
        ans.append(min(list(map(lambda x: abs(chanal[x]-chanal[x+1]),range(n-1)))))
    return ans
def max_razm_fft(data):
    f = np.abs(np.fft.rfft(data))
    return max_razm(f)
def min_razm_fft(data):
    f = np.abs(np.fft.rfft(data))
    return min_razm(f)
def max_assim_razm(data):
    ans = []
    for i in range(0,8,2):
        r1 = np.array(list(map(lambda x: abs(data[i][x]-data[i][x+1]),
        ↪  range(data[i].shape[0]-1))))
        r2 = np.array(list(map(lambda x: abs(data[i+1][x]-data[i+1][x+1]),
        ↪  range(data[i+1].shape[0]-1))))
        ans.append(max(np.abs(r1-r2)))
    return ans
def max_pik(data):
    ans = []
    for chanal in data:
        m = max(np.abs(chanal))
        t = (sum(list(map(lambda x:x**2,chanal)))/chanal.shape[0])**0.5
```

```
75          ans.append(m/t)
76      return ans
77  def min_pik(data):
78      ans = []
79      for chanal in data:
80          m = max(np.abs(chanal))
81          t = (sum(list(map(lambda x:x**2,chanal)))/chanal.shape[0])**0.5
82          ans.append(m/t)
83      return ans
84  def max_assim_pic(data):
85      ans = []
86      for i in range(0,8,2):
87          f1 = np.abs(np.fft.rfft(data[i]))
88          f2 = np.abs(np.fft.rfft(data[i+1]))
89          pik1 = (sum(list(map(lambda x:x**2,f1)))/f1.shape[0])**0.5
90          pik2 = (sum(list(map(lambda x:x**2,f2)))/f2.shape[0])**0.5
91          ans.append(np.abs(pik1-pik2))
92      return ans
93  def MSR(data):
94      ans = []
95      for ch in data:
96          ans += [np.sqrt(np.sum(np.array(ch)**2) / len(ch))]
97      return ans
98  def var(data):
99      ans = []
100     for ch in data:
101         v = np.std(ch)/np.mean(ch)
102         ans.append(v)
103     return ans
104 def only_EEG(data):
105     ans= []
106     for ch in data:
107         ans += list(ch)
108     return ans
109 def only_FFT(data):
110     ans = []
111     for ch in data:
112         ans += list(np.abs(np.fft.rfft(ch))[1:])
113     return ans
114 def wawes(data):
115     ans = []
116     delta = 1 * 64*2//SLICE_SIZE
117     teta = 4* 64*2//SLICE_SIZE
118     alpha = 8 * 64*2//SLICE_SIZE
119     beta = 12 * 64*2//SLICE_SIZE
120     gamma = 30 * 64*2//SLICE_SIZE
121     for ch in data:
122         f = np.abs(np.fft.rfft(ch))
123         ans += [
124             np.sum(f[delta:teta]),
125             np.sum(f[teta:alpha]),
126             np.sum(f[alpha:beta]),
127             np.sum(f[beta:gamma]),
128             np.sum(f[gamma:]),
129         ]
130     return ans+[ans[5*(i) + 1] / ans[5*(i) + 2] - ans[5*(i+1) + 1] / ans[5*(i+1) + 2]
    ↪  for i in range(0,7)]
131
132
133
```

```
134
135 def getFeatures(X):
136     #первый признак, среднее абсолютная величина
137     a = np.abs(X).sum()/X.size
138     #второй признак, средная абсолютная производная (дифференциал) (x[i] - x[i-1])
139     b = np.abs(np.diff(X)).sum()/X.size
140     #третий признак, среднее колличество пересечений нуля, с порогом
141     #возможно я выбрал не оптимальное значение
142
143     #                                                  порог
144     c = np.sum((X[:-1] * -X[1:])[abs(X[:-1] - X[1:]) > 0.1] > 0)/X.size
145
146     #четвертый признак, среднее колличество, так сказать, пиков, с порогом на его
        ↪  высоту
147     #(np.diff(X[:-1]) * -np.diff(X[1:]) - это число положительно, если у пика есть
        ↪  центр,
148     # а его основания лежат строго по одну сторону
149     # >5 - это порог, его я также мог установить не оптимальным образом
150     d = np.sum((np.diff(X[:-1]) * -np.diff(X[1:]))>0.1)/X.size
151     # уже писал: используем модель авторегрессии, после ее обучения, используем ее
        ↪  веса, как признаки
152     # колличество весов = lags+1, плюс один, это константа, как у графика: y = kx + C
153     e = list(AutoReg(X, lags=10,old_names=False).fit().params)
154     #возращаем массив признаков
155     return np.array([a,b,c,d] + e)
156
157
158 featuresMask = np.array([False, False, False, ..., False, False, False])
159
160 Xs = []
161 for d in data:
162     fich = []
163     fich+=list(max_razm(d))
164     fich+=list(min_razm(d))
165     fich+= list(max_razm_fft(d))
166     fich+= list(min_razm_fft(d))
167     fich+=list(max_pik(d))
168     fich+=list(min_pik(d))
169     fich+=list(max_assim_pic(d))
170     fich+= list(max_assim_razm(d))
171     fich+=list(wawes(d))
172     fich+=list(MSR(d))
173     fich+=list(only_EEG(d))
174     fich+=list(only_FFT(d))
175     fich+=list(var(d))
176     for i in range(8):
177         fich+=list(getFeatures(d[i]))
178     Xs.append(np.array(fich)[featuresMask])
179 Xs = np.array(Xs)
180
181
182 f = open('y.txt','r')
183 ys = [r.split() for r in f.read().split('\n')[1:] if r]
184 f.close()
185 # 1 - 4 - непонравилось
186 # 5-6 - нейтрально
187 # 7-10 - понравилось
188 labelDecode = [-1,0,0,0,0,2,2,1,1,1,1]
189 ys = [labelDecode[int(float(d))] for (n,d) in ys]
190 ys = np.array(ys)
```

```
191
192 Xs1 = robust_scale(Xs)
193
194 res =
    ↪  cross_val_score(RidgeClassifierCV(),Xs1,ys,cv=ShuffleSplit(test_size=0.2,n_splits=100),
    ↪  n_jobs=1,scoring='balanced_accuracy')
195 print(res.mean())
```