**Hướng dẫn**

# Cài thư viện

Các thư viện cần thiết được mô tả trong file **requirements.txt**. Để cài đặt sử dụng câu lệnh:

***pip install -r requirements.txt***

# Thiết lập tham số

# Mô hình dự đoán

Các thông số của mô hình dự đoán được thiết lập tại class **ModelSetting()**. Các thông số này bao gồm:

* **name**: tên mô hình
* **path**: đường dẫn của file mô hình dự đoán
* **channels**: danh sách các kênh được sử dụng để dự đoán
* **sfreq**: tần số lấy mẫu của dữ liệu đưa vào mô hình
* **t\_min**: thời điển bắt đầu của dữ liệu so với hành động. Đơn vị là giây.
* **t\_max**: thời điểm bắt dầu của dữ liệu so với hành động. Đơn vị là giây.
* **l\_freq**: lọc thông thấp
* **h\_freq**: lọc thông cao
* **shape**: cặp thông số về kích thước dữ liệu đầu vào của mô hình và số dữ liệu sử dụng trên mỗi kênh. Ví dụ ((1,56), 28) thì (1,56) là kích thước đầu vào của mô hình, 28 là số điểm dữ liệu lấy trên 1 kênh

Khi muốn thay đổi mô hình dự đoán thì cần phải cập nhật file mô hình và các thông số tương ứng của mô hình mới

# Luồng dữ liệu vào

Luồng dữ liệu vào là các [Lab Streaming Layer (LSL)](https://github.com/sccn/labstreaminglayer) được thiết lập trong file api\_inlet.py. Các thông số của luồng này bao gồm:

* **host**: địa chỉ API được khởi chạy
* **ET\_stream\_name**: tên của luồng dữ liệu gửi từ phần mềm blife

***channels.csv*** là file chứa thứ tự các kênh của tín hiệu EEG thu được từ thiết bị emotiv. Nếu sử dụng các thiết bị hoặc cấu hình khác thì có thể sử dụng đoạn code dưới đây để tạo file channels.csv tương ứng

*import pandas as pd*

*from pylsl import StreamInfo*

*streams = resolve\_stream('name', 'EmotivDataStream-EEG')*

*inlet = StreamInlet(streams[0])*

*ch\_labels = []*

*ch = inlet.info().desc().child("channels").child("channel")*

*for k in range(inlet.info().channel\_count()):*

*ch\_labels.append(ch.child\_value("label"))*

*ch = ch.next\_sibling()*

*df = pd.DataFrame(ch\_labels)*

*df.to\_csv('channels.csv', header=False)*

Có 2 địa chỉ của API là:

* ***'/update-eeg'*** dùng để nhận tín hiệu eeg và cập nhật vào bộ nhớ buffer chờ xử lý.

Tham số được truyền vào là file JSON có 2 thuộc tính 'eeg\_data' và 'timestamp'. Thuộc tính 'eeg\_data' nhận giá trị là một mảng hai chiều m\*n chứa giá trị của m sample, mỗi sample có dữ liệu của n kênh. Thuộc tính 'timestamp' nhận giá trị là chuỗi m số thực lưu thời điểm các sample tương ứng được ghi lại. Các giá trị này được lấy trực tiếp ở luồng dữ liệu truyền ra từ emotiv.

* ***'/update-et'*** dùng để nhận tín hiệu gõ bàn phím (ET) và cập nhật vào bộ nhớ. Sau đó quá trình dự đoán sẽ được thực hiện.

Tham số được truyền vào là file JSON có 2 thuộc tính 'is\_clicked' và 'timestamp'. Thuộc tính 'timestamp' là thời điểm một nút được lựa chọn trong phần mềm ET. Thuộc tính 'is\_clicked' luôn là True để thông báo là hành động đã được thực hiện.

# Luồng dữ liệu ra

Luồng dữ liệu ra cũng là 1 LSL. Các thông số của luồng dữ liệu ra sẽ được thiết lập tại class **Outlet()**. Các thông số được thiết lập bao gồm:

* **name**: tên của luồng dữ liệu ra
* **n\_channels**: số lượng kênh của dữ liệu ra
* **data\_type**: loại dữ liệu ra (nên để là Maker)
* **sfreq**: tần số đưa tín hiệu ra
* **source\_id**: tên nguồn đưa tín hiệu ra
* **outlet**: luồng tín hiệu ra (không sửa)
* **channel\_name**: tên các kênh của dữ liệu ra

Mặc định dữ liệu ra sẽ có dạng **[[điểm dữ liệu], thời gian tương ứng]**. Nếu không cung cấp chuỗi thời gian thì sẽ tự động lấy thời gian của hệ thống. Để gửi thời gian mong muốn thì thêm vào nó vào tham số **timesample** khi gọi hàm **send\_sample()**

# Các tham số khác

Thông số chung sẽ được thiết lập tại class **Setting()** trong file **config.py**. Các thông số này bao gồm:

* **RAW\_SAMPLING\_RATE**: tần số lấy mẫu của tín hiệu điện não
* **BUFFER\_TIME**: độ dài đoạn tín hiệu điện não được lưu trữ liên tục. Đơn vị là giây.
* **use\_ica**: có sử dụng ica hay không.
* **PREDICT\_COUNT**: số lần dự đoán sau khi bấm phím

***Lưu ý***: thời gian thực hiện ica khoảng 0.8s/1s dữ liệu

# Khởi chạy chương trình

Khởi chạy API bằng câu lệnh:

***uvicon main:app --host x.x.x.x -- port y***

với x.x.x.x là địa chỉ host và y là port khởi chạy API. Địa chỉ mặc định là 127.0.0.1:8000

Khởi chạy nhận luồng dữ liệu và đưa vào API:

***python api-inlet.py***