

The logo consists of a horizontal rectangle divided into three equal-width vertical sections. The left section is magenta, the middle section is yellow, and the right section is teal. The word "Eyedrone" is written in white, sans-serif font across the middle of the rectangle, with the 'E' starting in the magenta section and the 'e' ending in the teal section.

Eyedrone

TEAM



Albert



Antonio



Eva

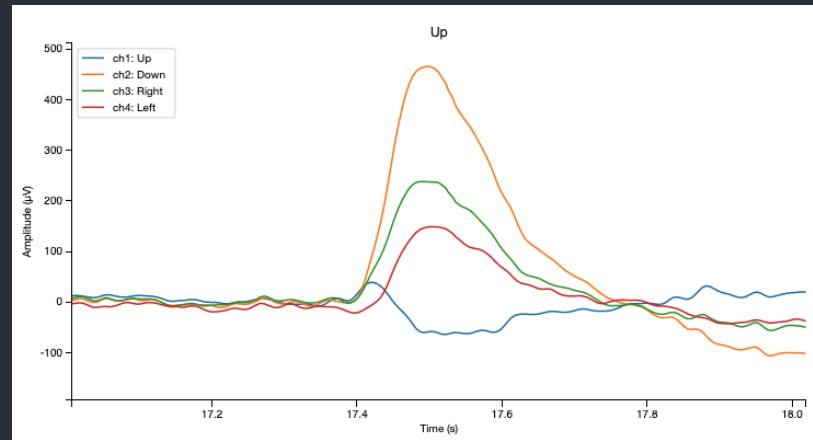
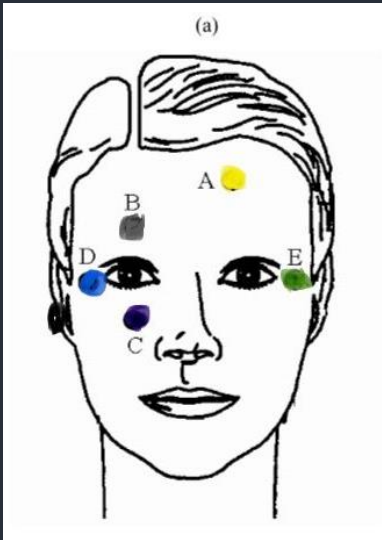


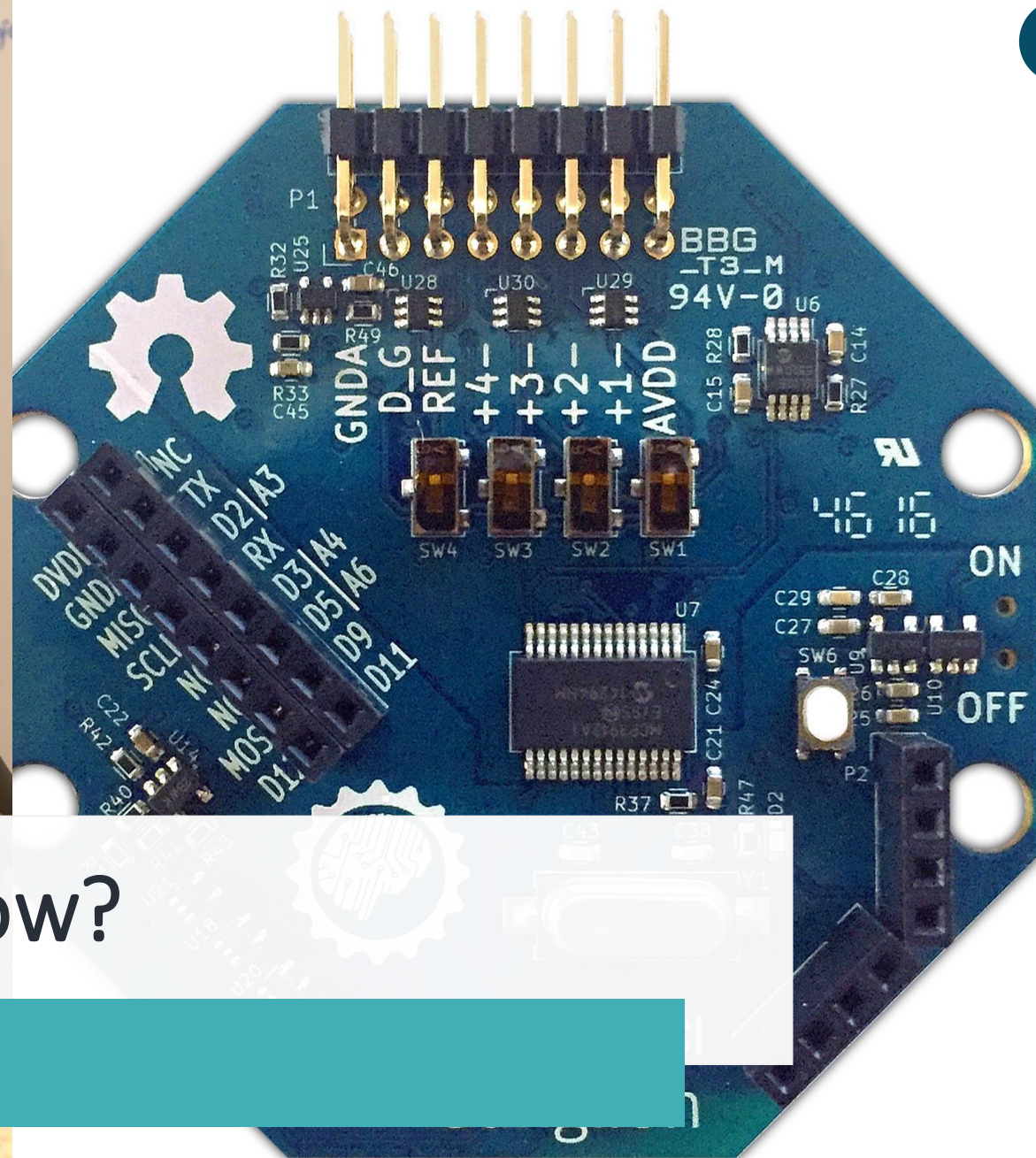
Asma



Ali

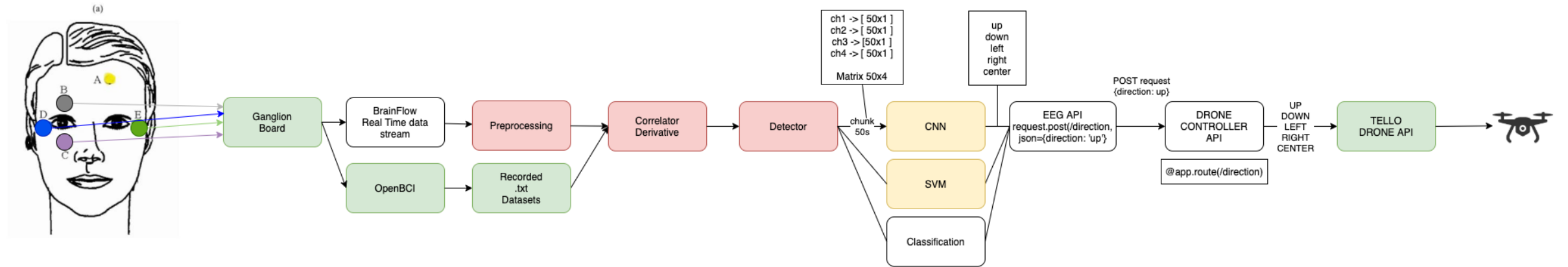
Move a **drone** using **your eyes**





How?

The end-to-end system



1. Data acquisition, preprocessing and detection

- Get raw data from sensors
- Process in real time
- When a change is detected, send a chunk of data as input to the neural network

2. Classification using AI

- Get movement of the eye as output

3. Drone controller

- Send movement to Drone
- Drone executes the movement

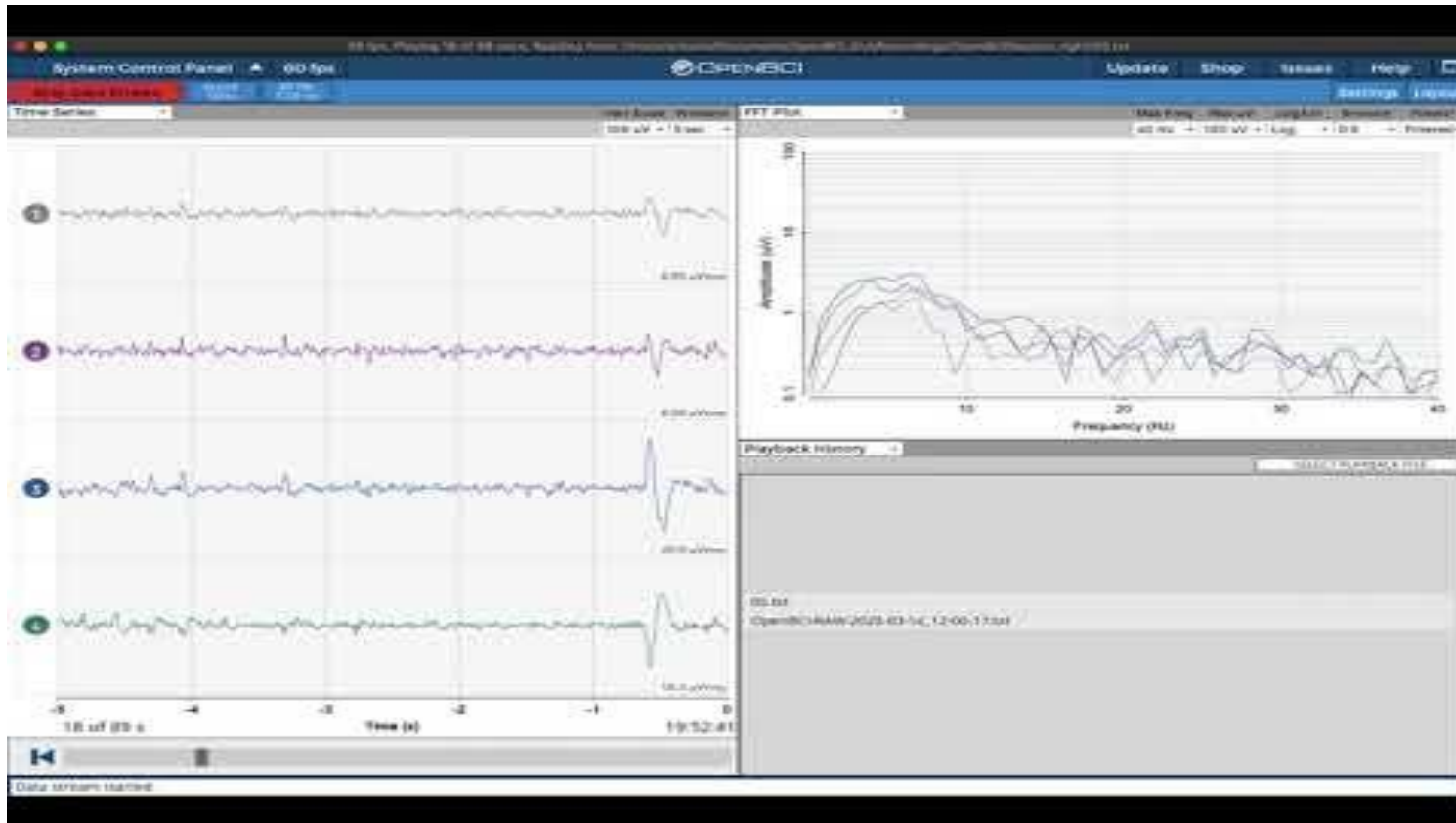
Microservices approach:

parts are interchangeable. Example: instead of using eye movements, we could use brain "thoughts", just by changing the first part

Ganglion Board + Sensors + Open BCI GUI

Recording datasets

- Usage of Ganglion Board (4-channel EEG) + golden cup sensors.
- Recordings made with OpenBCI GUI

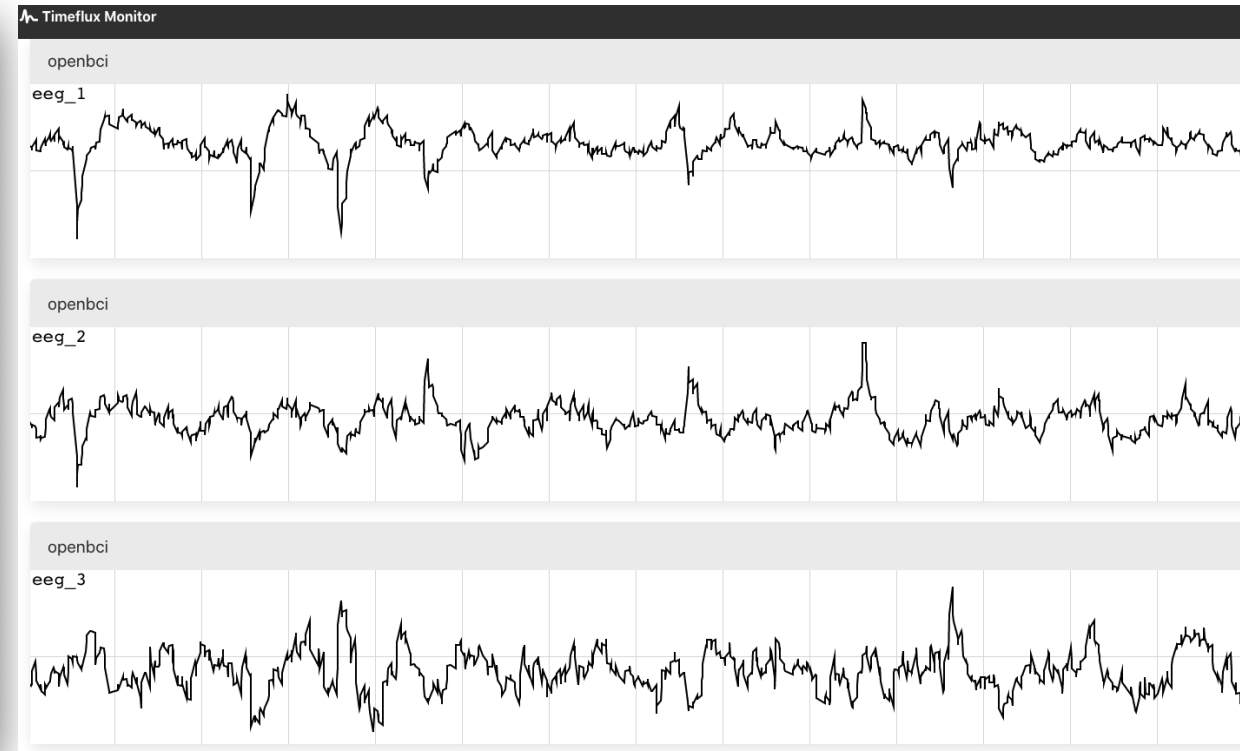


Recording example
right eye movement

Ganglion Board + Brainflow + Timeflux

Raw data acquisition directly from board for real-time detection

```
timeflux — Python • timeflux -d openbci.yaml — 130x32
...ockets/serverless-websockets-plugin/example — -bash
2020-05-23 17:25:33.698556900 peak {'value': -510.56849116267585, 'lag': 0.003026...
2020-05-23 17:25:33.926 DEBUG debug 24111 Out_left
label data
2020-05-23 17:25:33.814331055 valley {'value': -204.42215575029917, 'lag': 0.010525...
2020-05-23 17:25:33.927 DEBUG debug 24112 Out_right
label data
2020-05-23 17:25:33.814331055 valley {'value': -142.27007279188908, 'lag': 0.056656...
2020-05-23 17:25:34.531 DEBUG debug 24109 Out_up
label data
2020-05-23 17:25:34.441338062 peak {'value': 3617.432821471708, 'lag': 0.05271005...
2020-05-23 17:25:34.547 DEBUG debug 24110 Out_down
label data
2020-05-23 17:25:34.441338062 peak {'value': 3031.8414145477464, 'lag': 0.0526120...
2020-05-23 17:25:34.935 DEBUG debug 24112 Out_right
label data
2020-05-23 17:25:34.811512947 peak {'value': 105.04328003009631, 'lag': 0.624789,...
2020-05-23 17:25:34.935 DEBUG debug 24111 Out_left
label data
2020-05-23 17:25:34.811512947 peak {'value': 117.38832321927163, 'lag': 0.4789550...
2020-05-23 17:25:35.438 DEBUG debug 24109 Out_up
label data
2020-05-23 17:25:35.337376118 valley {'value': -1055.740906162932, 'lag': 0.1521520...
2020-05-23 17:25:35.453 DEBUG debug 24110 Out_down
label data
2020-05-23 17:25:35.337376118 valley {'value': -866.2011221329635, 'lag': 0.1525840...
2020-05-23 17:25:36.842 DEBUG debug 24109 Out_up
label data
2020-05-23 17:25:36.756176949 peak {'value': 178.91718929285219, 'lag': 0.4237520...
2020-05-23 17:25:36.867 DEBUG debug 24110 Out_down
label data
2020-05-23 17:25:36.756176949 peak {'value': 160.33652807469036, 'lag': 0.4270718...
```



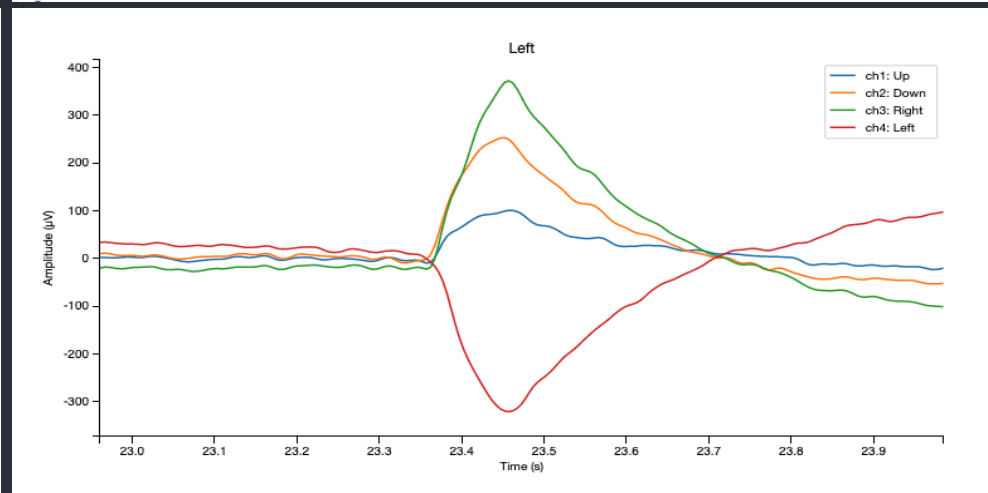
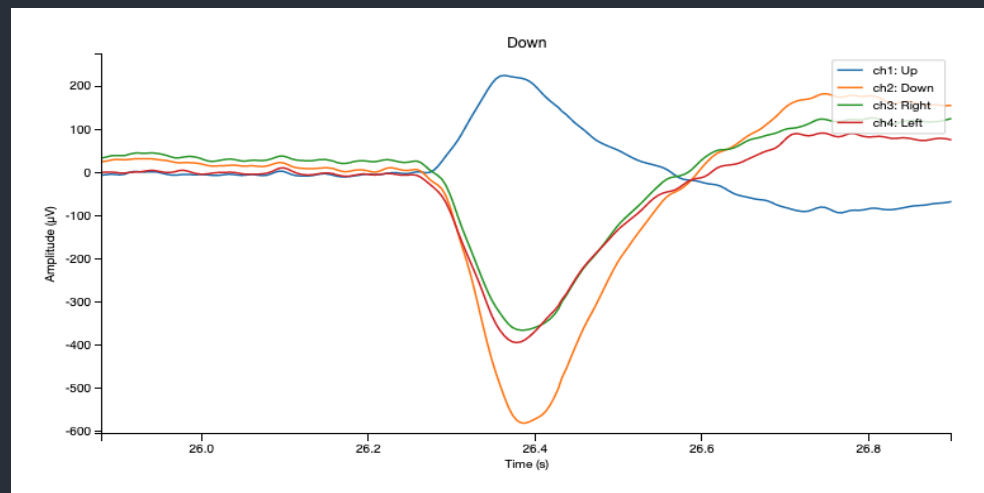
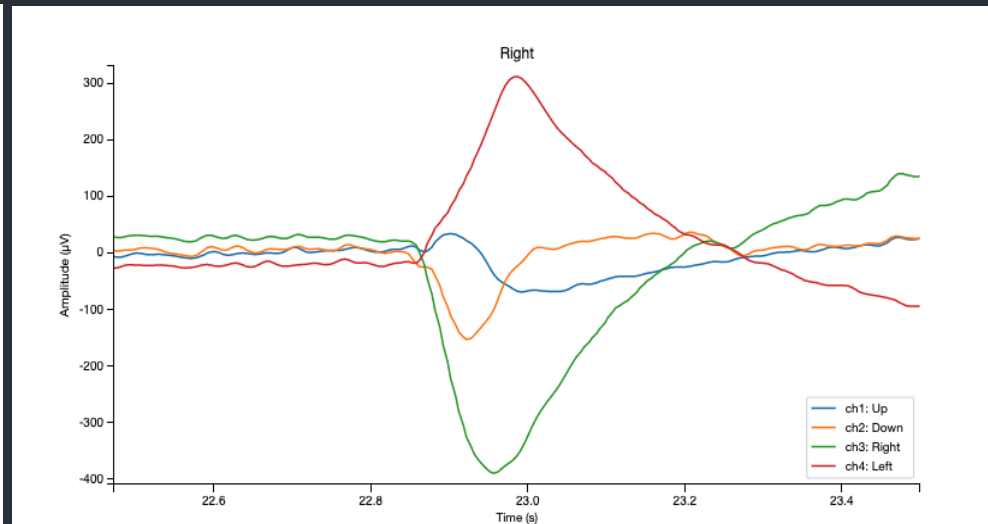
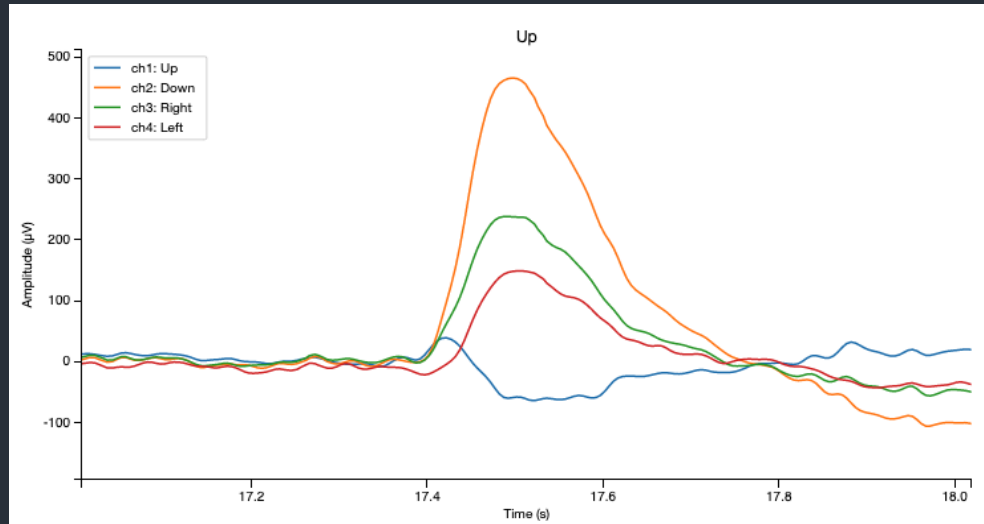
Unfortunately, there are some parts missing:

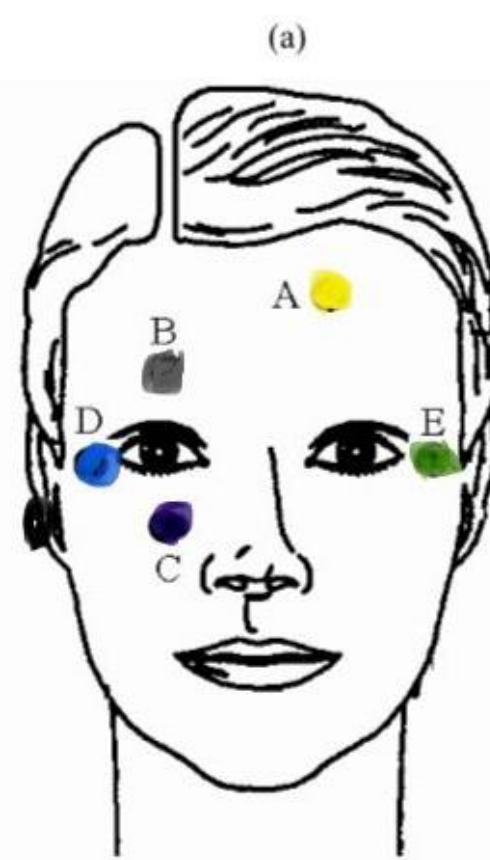
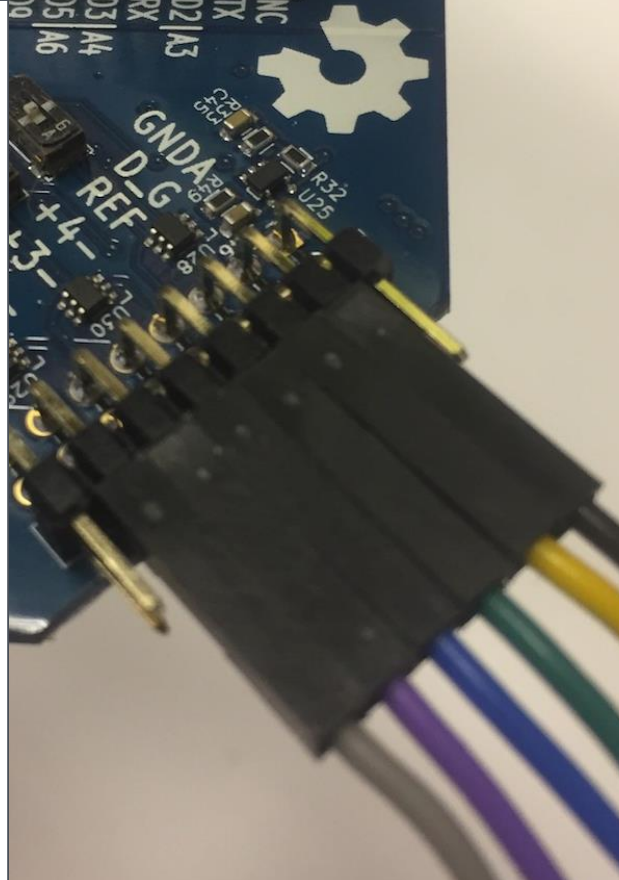
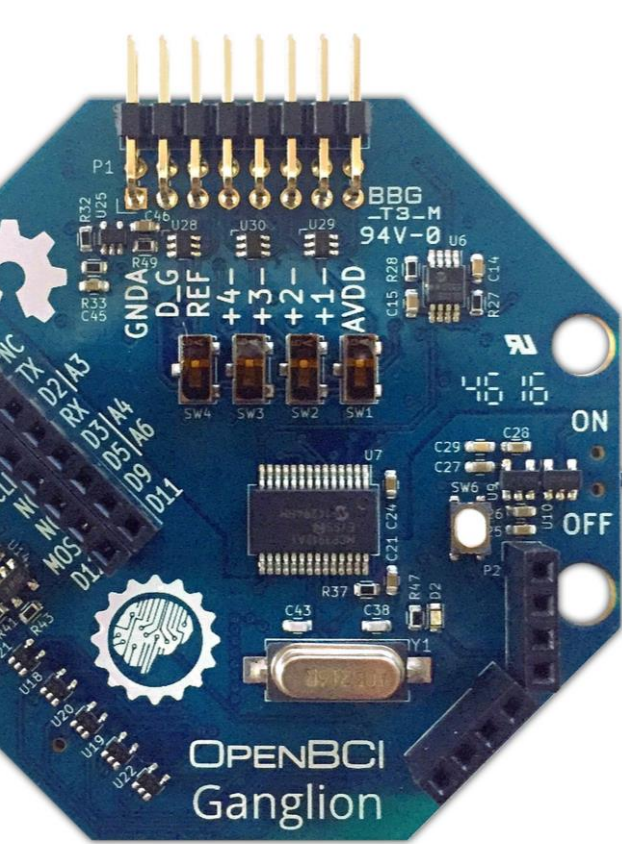
What is missing? Real Time processing.

- Get chunks of data in real time when an event happens (not only the event) and pass this as an input to the Neural Net. Making this work in real-time was not an easy work.
- Pass the output of the neural net to the Drone API in real time.
- Investigate REST vs Websockets to connect with the drone.



Dataset





Dataset creation

Using OpenBCI Ganglion Board (4-channels)

Tests done:

- 25x Center --> Up --> Center movement
- 25x Center --> Down --> Center movements
- 25x Center --> Left --> Center movements
- 50x Center --> Right --> Center movements
- 1x Long Blinking Test

Main problems...



We wanted to:

- Have multiple sources of data**
- From each have multiple recordings**
- Do data augmentation**

Lack of data

Learning

01

Data Augmentation

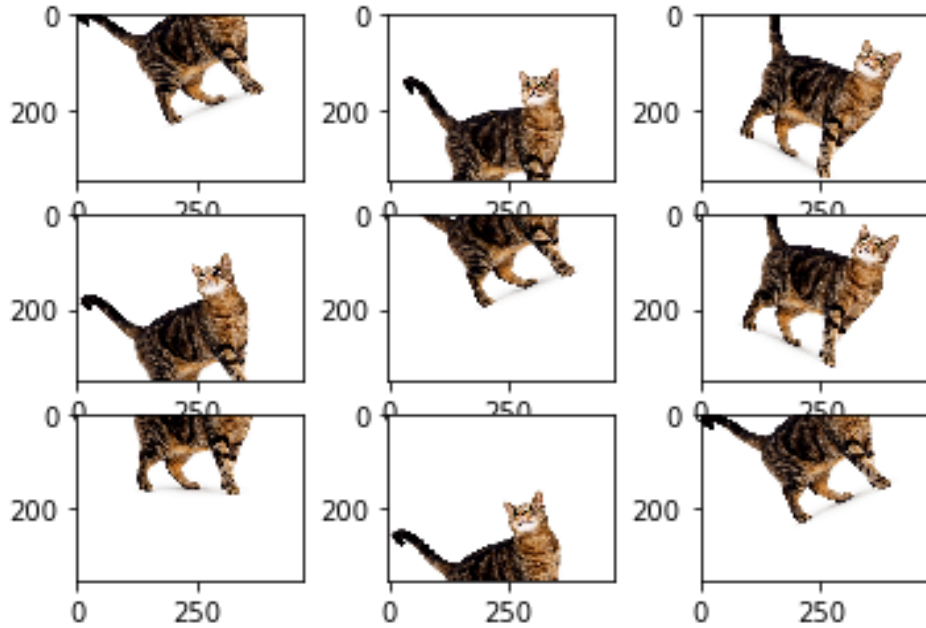
02

Machine Learning

03

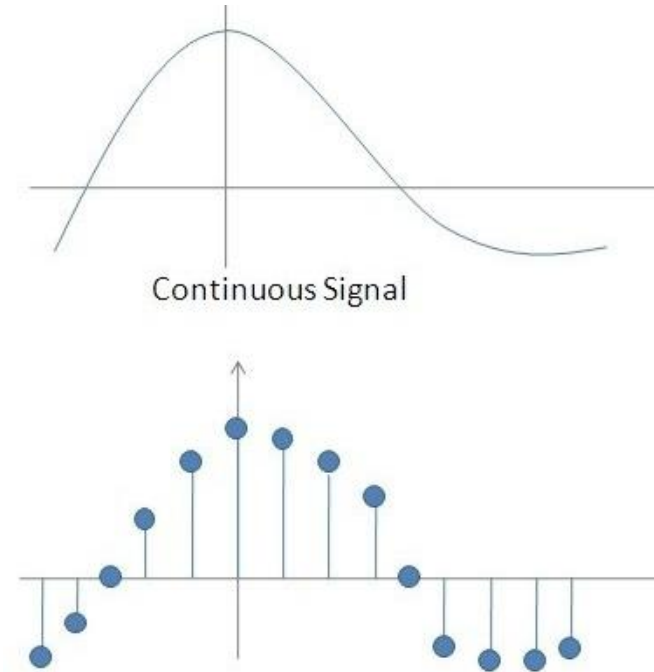
Deep Learning

Data Augmentation



Concept:

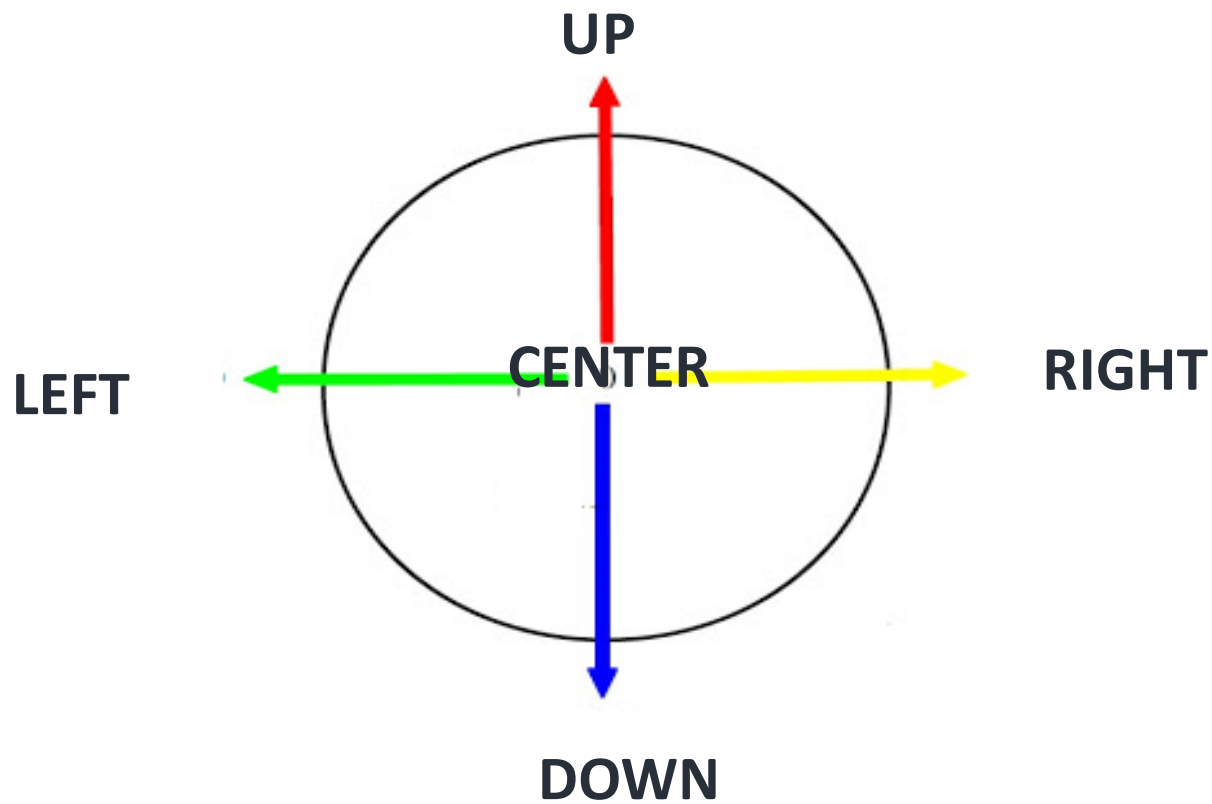
- Choose randomly P% of time points;
- Stretch them with [Min-Max]s



Helps:

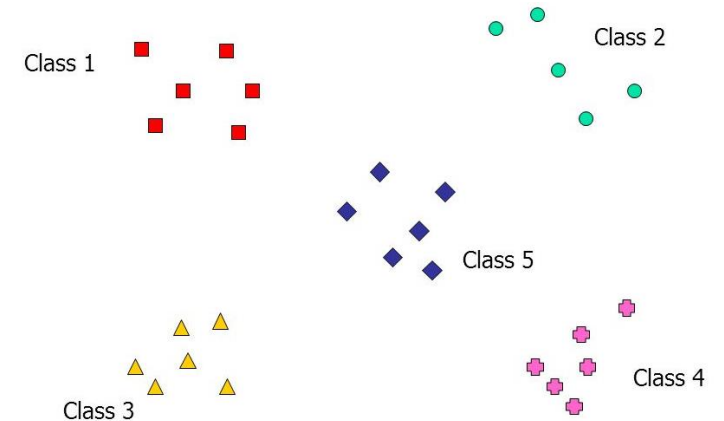
- Less biased model;
- Generated Noise;

Machine Learning: SVM



Feature Vector:

- Mean, Max , Min , Var;
- 5 classes



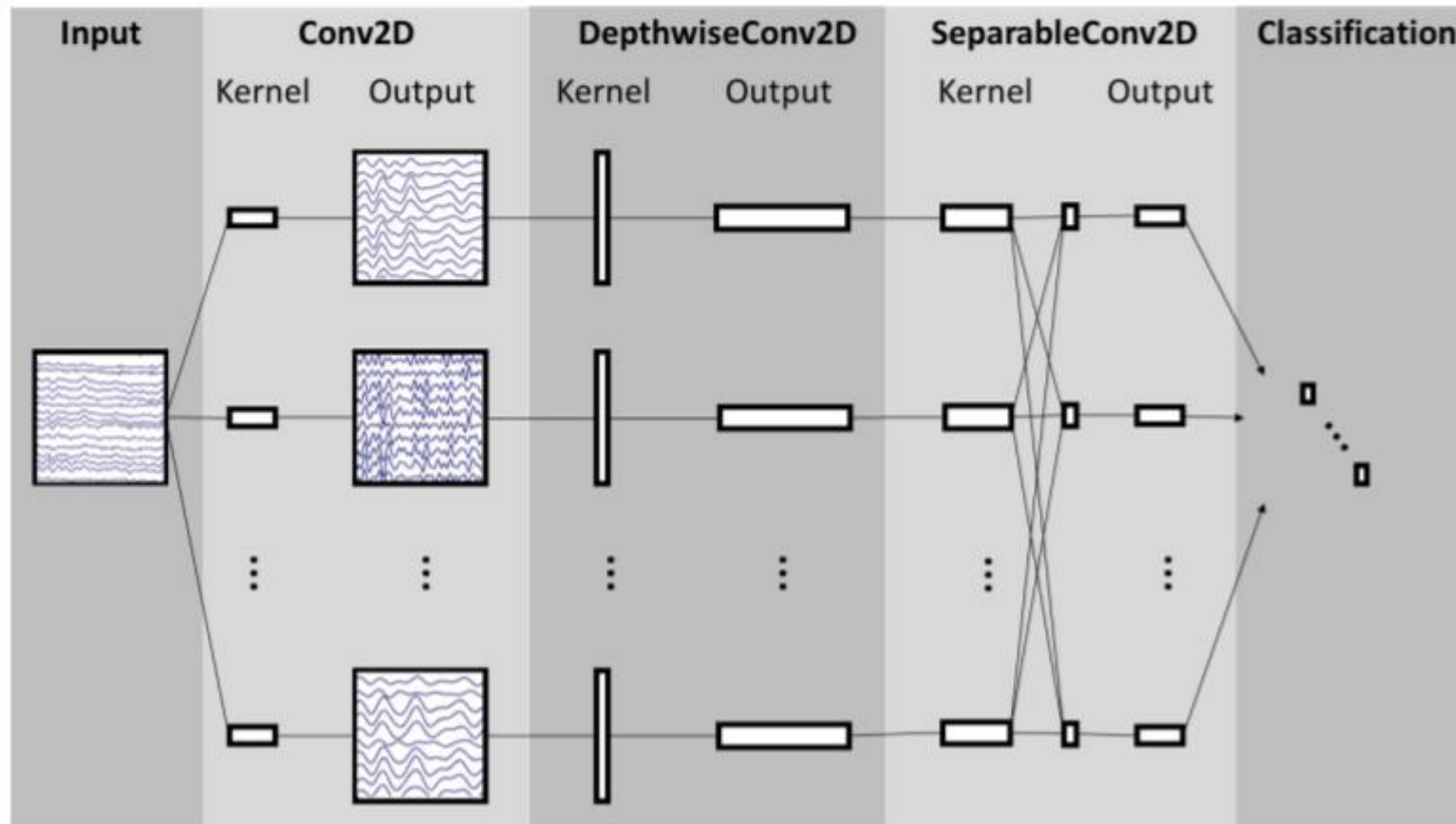
Result:

- **43% accuracy**

Deep Learning: EEGNet

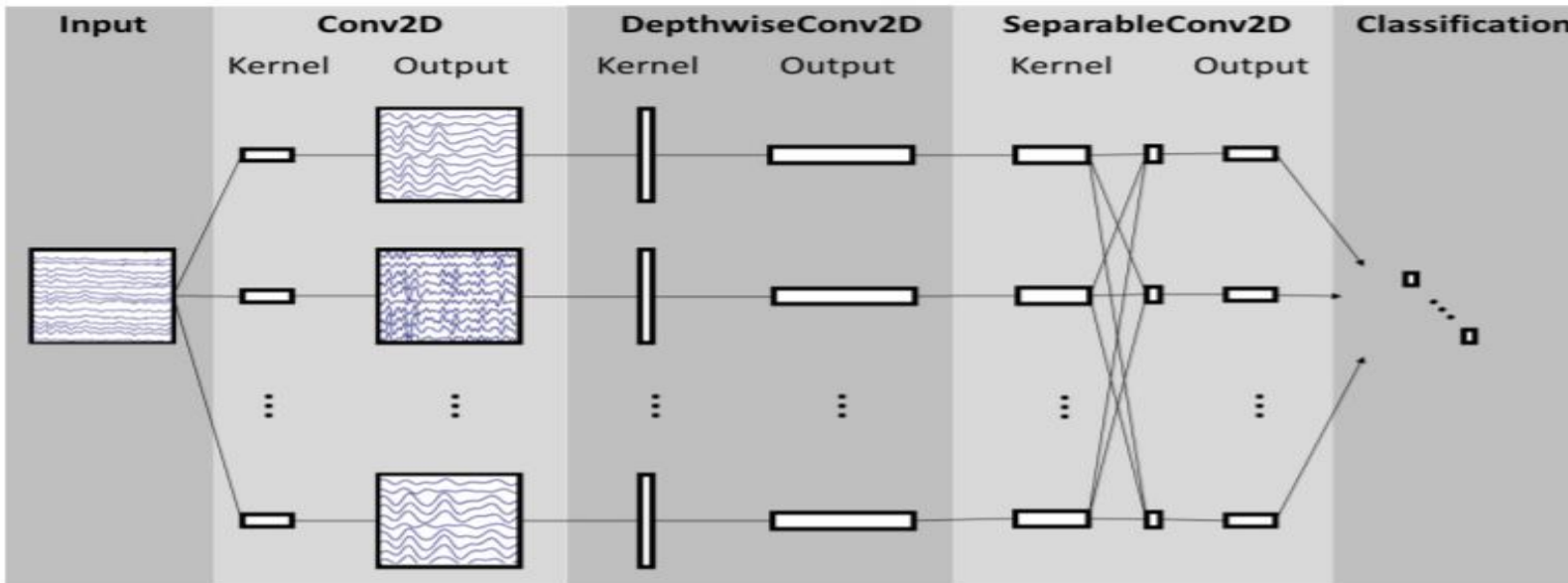
EEGNet: A Compact Convolutional Neural Network for EEG-based Brain-Computer Interfaces

Vernon J. Lawhern¹, Amelia J. Solon, Nicholas R. Waytowich, Stephen M. Gordon, Chou P. Hung, and Brent J. Lance



- *Fully supervised*
- *Few data*
- *Compact architecture*
- *Engineering*

Deep Learning: EEGNet



Training and Engineering:

- Number of channels 4
- Frequency 100Hz

Result:

- **50,1% accuracy**

THE DRONE

- Communications
- DroneApp connected with the Drone API
- Real time show
- Flask app for simple navigation controller
- Router port forwarding to be able to work remotely



Progress

Dataset creation: Collect EEG signal with 4 golden cup sensors

Done

01

02

Signal/data pre processing:
Data cleaning, noise

Done

03

04

API

Drone

05

Signal sending:
in real-time from a Board to PC and/or Cloud (Amazon Web Services)

Classification:

Data augmentation **Done**

SVM **Done**

Deep learning **On-going**

