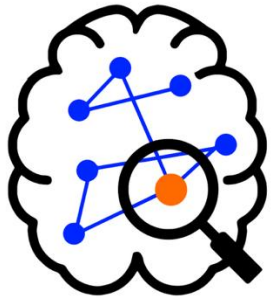


BCI Meeting 2025 - WS19 - June 5

HappyFeat: an interactive and efficient BCI framework for clinical applications

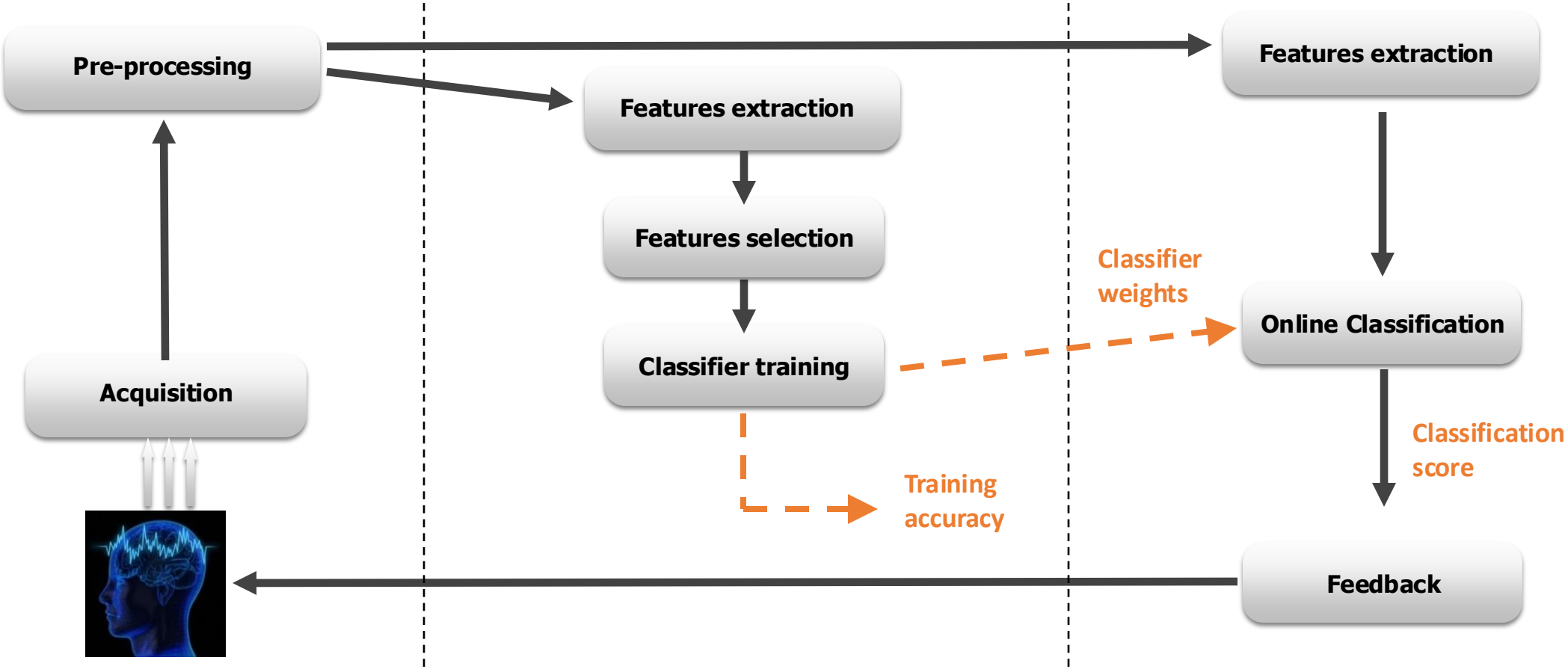
Arthur Desbois

Inria Paris, NERV team, ICM (Paris Brain Institute)

Table of contents

Table of contents

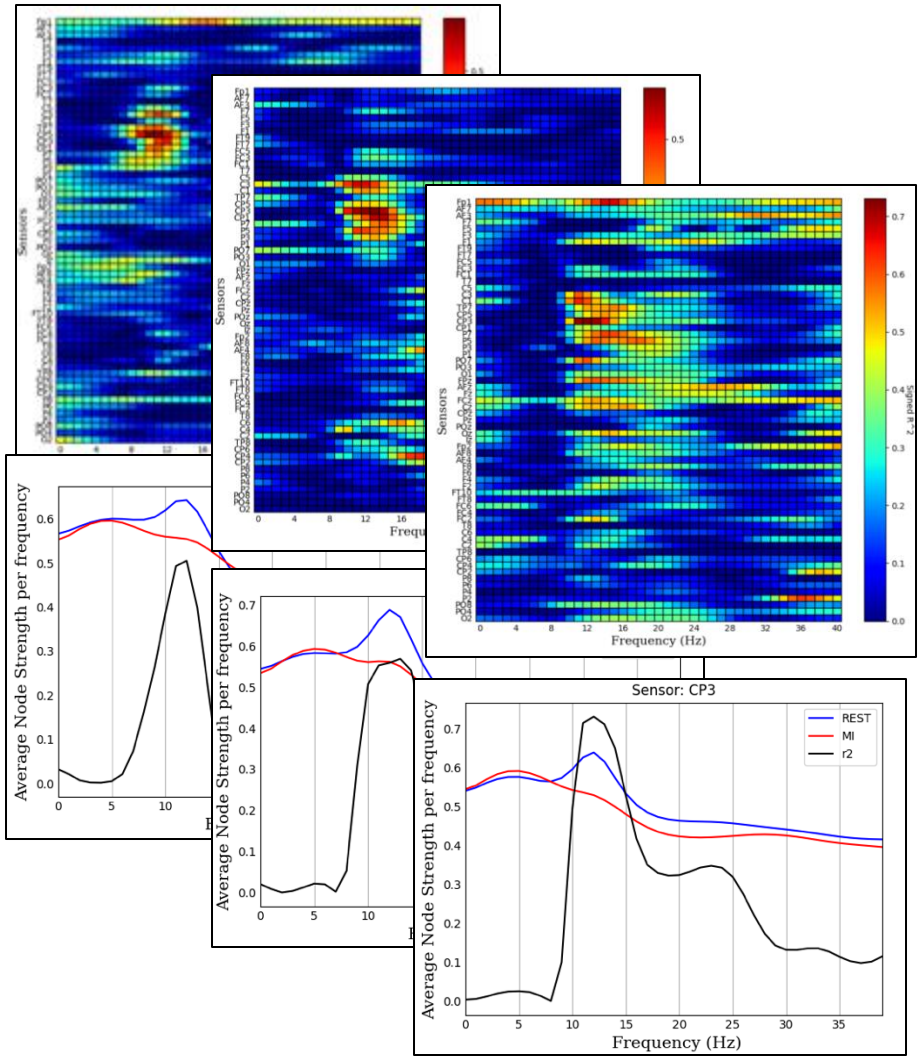
The BCI loop


Pre-processing
Features extraction
Features selection
Classifier training
Acquisition
Classifier weights
Training accuracy
Features extraction
Online Classification
Classification score
Feedback
Phase 1 – EEG Acquisition
Phase 2 – Calibration (offline feature selection + Training)
Phase 3 – Closed-loop online BCI

Some context…

- « Features of Interest » (FOI)
  - Selecting adequate FOIs is a **crucial step for BCI performance.**
  - After EEG signals acquisition, an **analysis phase** is needed to select best FOIs.
    - ➔ Scientific softwares (i.e., MATLAB)
    - ➔ Manual step, expertise needed
  - If this analysis phase is too long, a lot can change in the meantime:
    - EEG sensors impedances
    - Subject's brain behaviors
    - Subject's attention & motivation

➔ Signal characteristics might be very different between Acq/Training phase and Online phase…



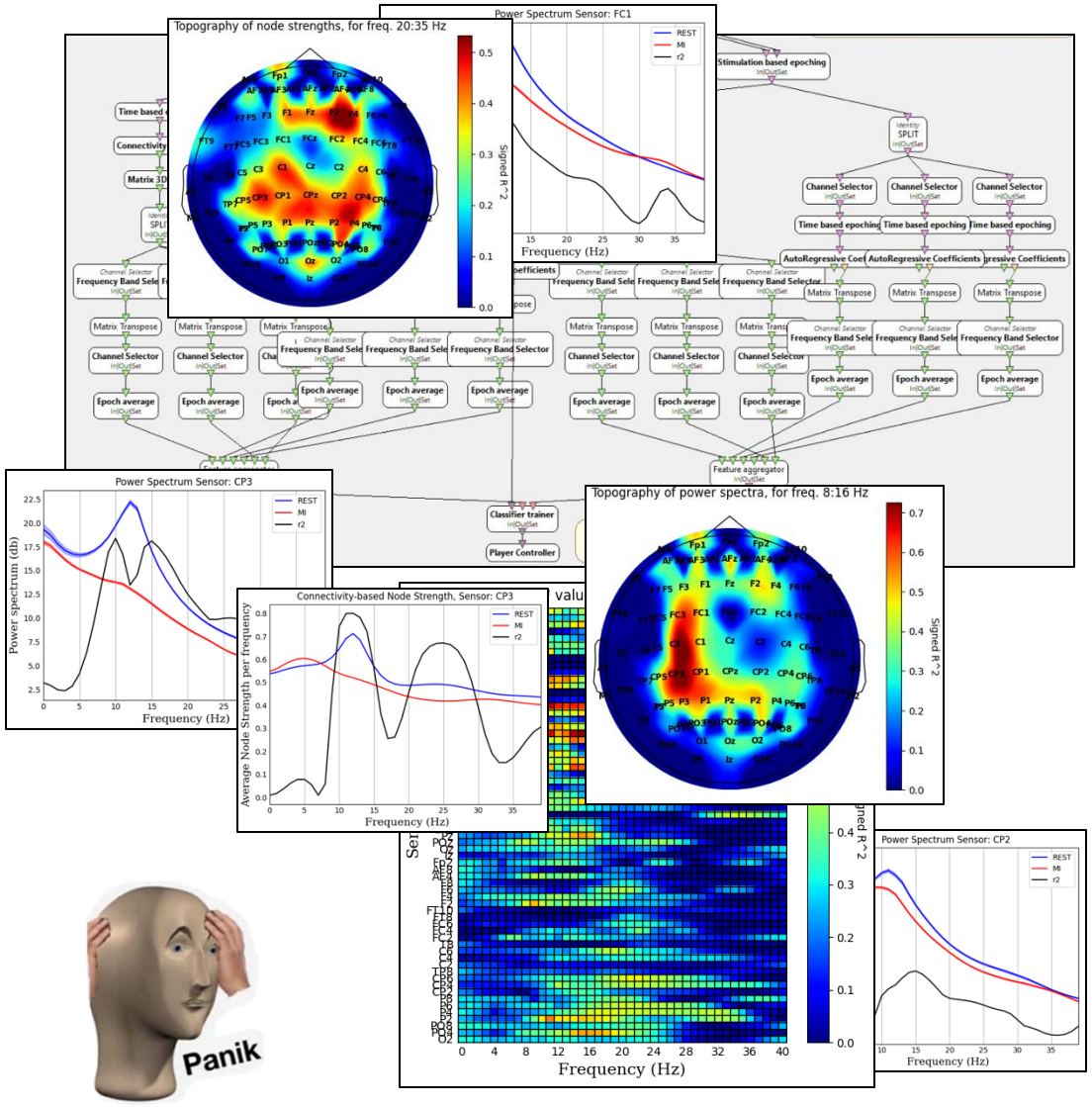
Some context…

The “analysis phase” involves many manipulations. An example:

- Setting up “**feature extraction**” pipelines (e.g. OpenViBE)
- Finding FOIs through **visualization** (e.g. with Matlab, Python, etc.)
- Setting up & running **training** pipelines (e.g. OpenViBE)

- … and maybe **going again** through those steps **multiple times** until “correct” features have been found, or to account for inter-run variability

➔ **Tedious, error-prone, hard to achieve in a limited time**



Some context…

- **Experimental / clinical context**
  - BCI protocols (and more specifically Motor Imagery (MI)) :
    - Long duration
    - Highly demanding in focus / concentration
    - Subjects may be affected by cognitive disorders

  ⊕ Need to select adequate features in order to obtain satisfactory performances

  - With the software tools available in 2020:
    - ➔ **difficult in experimental setting, unrealistic in clinical setting.**
    - ➔ **Not an option to use « new » features (eg. graph-based) in real conditions**

  ➔ Decision :
  rationalize, simplify & automate the feature extraction/selection steps as much as possible.

HappyFeat

Python-based framework for facilitating MI pipelines

Main focus:

Simplify & accelerate

feature extraction,

feature selection,

classifier training

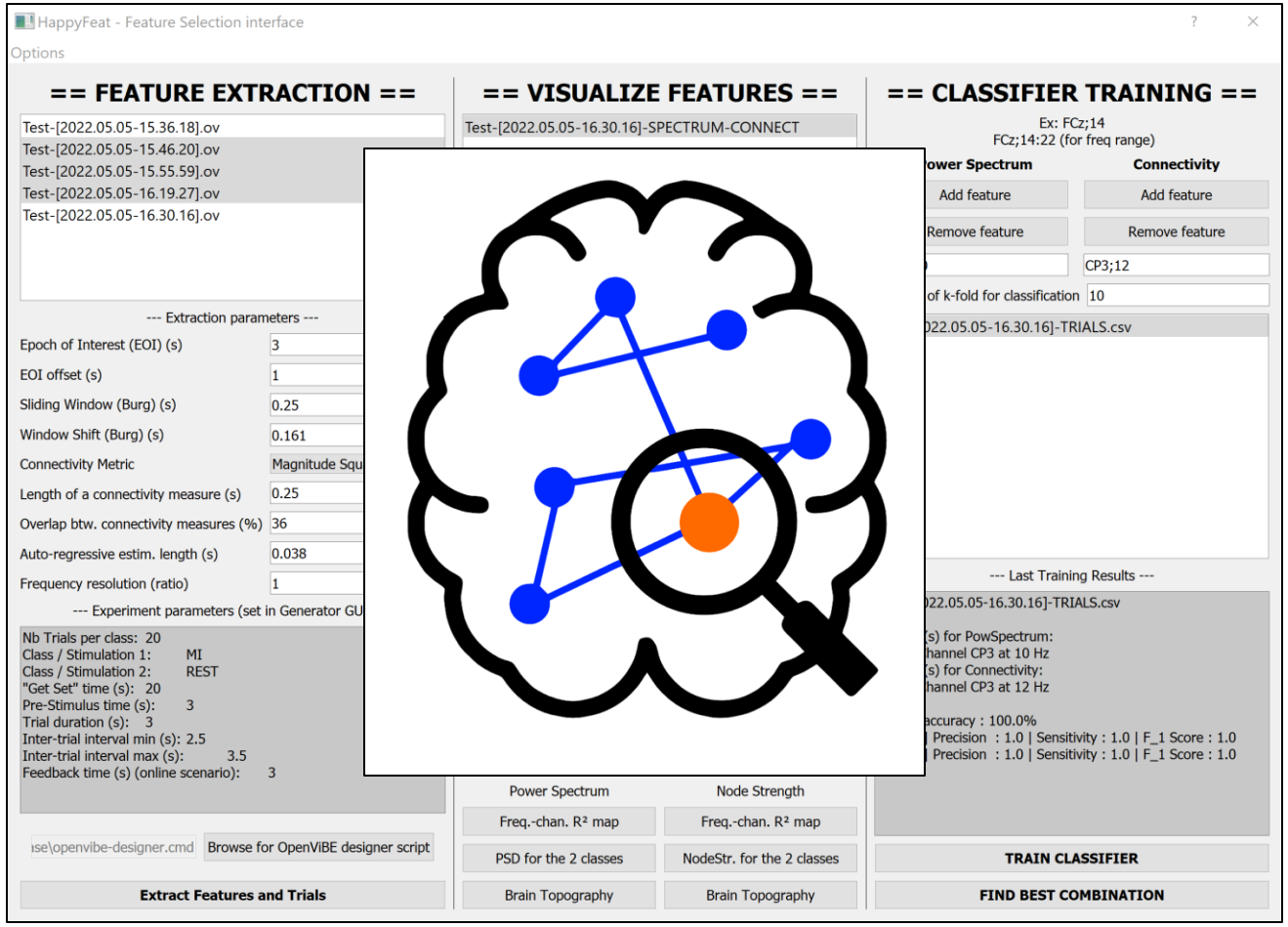


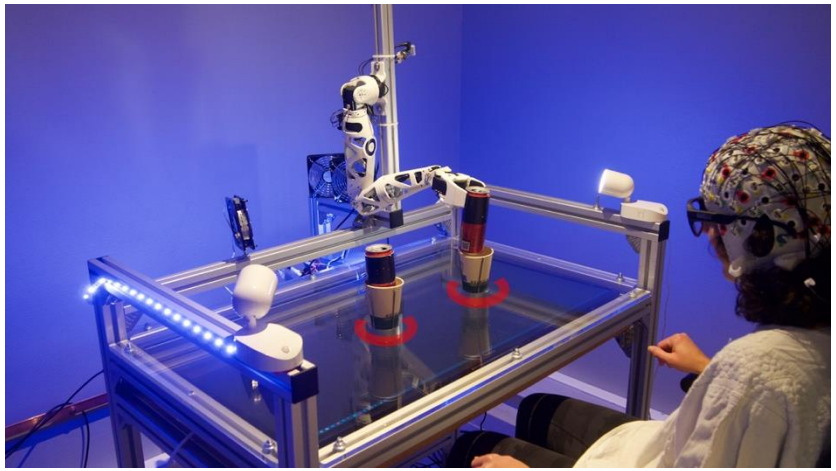
Table of contents

HappyFeat – Main Concepts

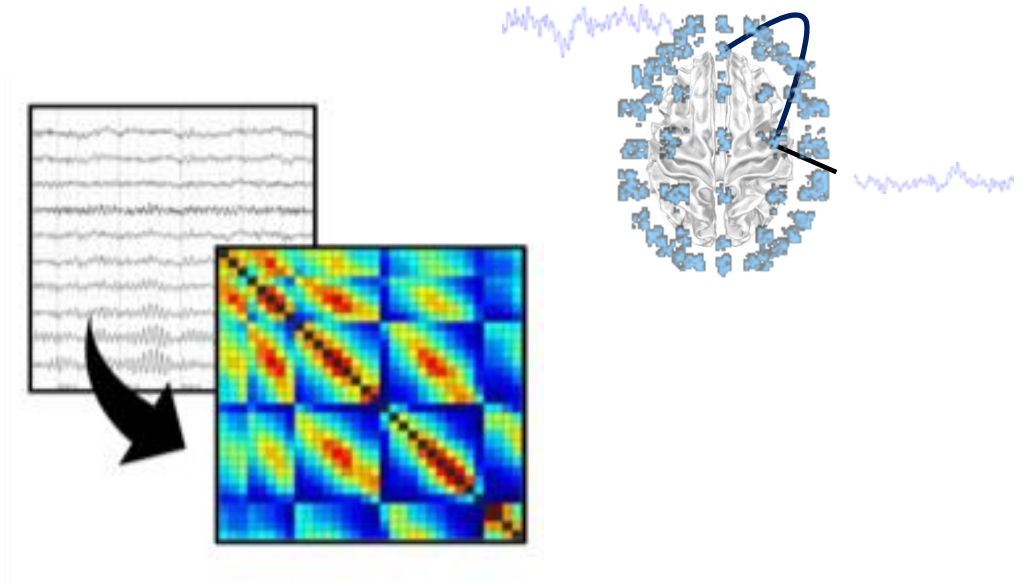
- Main objectives
  - « Online » :

    Simplify the setup and usage of BCI pipelines in experimental conditions
  - « Offline » (… to online) :

    Facilitate the integration & comparaison of new innovative metrics



BRACCIO protocol – Venot et al., ICM/NERV



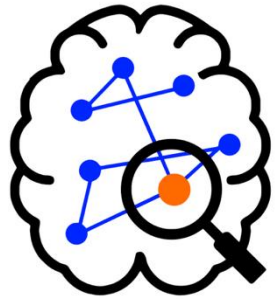
HappyFeat – Main Concepts

- **Key feats & mechanisms:**
  - **Clean, risk-free, reproducible environment**

    ➔ avoid unnecessary & error-prone manipulations.
  - **Trial-and-error oriented workflow**

    ➔ all steps can be repeated quickly & as many times as needed
  - **Unified "dashboard" GUI + Workspace management system**
  - **Direct link with BCI software**: OpenViBE (C++) *(and TimeFlux (Python) since v0.3.0)*

    ➔ no scenario edition/manipulation necessary: everything is **automated**!



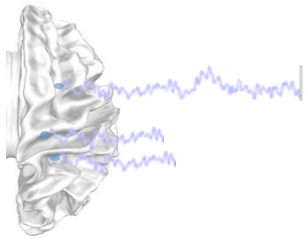
- **Two main use cases:**
  - **Make BCI pipelines smoother/easier to use** and allow **reproducibility** of exps.
  - **Prospective works** & **comparison of alternative features of interest** (connectivity, networks…)

HappyFeat – Main Concepts

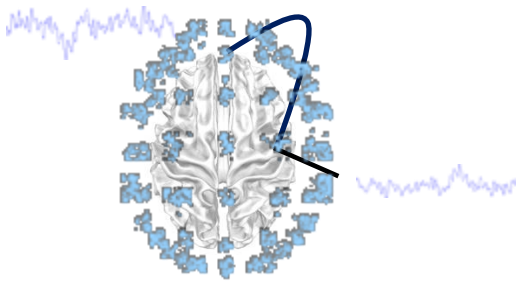
Unified, dashboard-like GUI

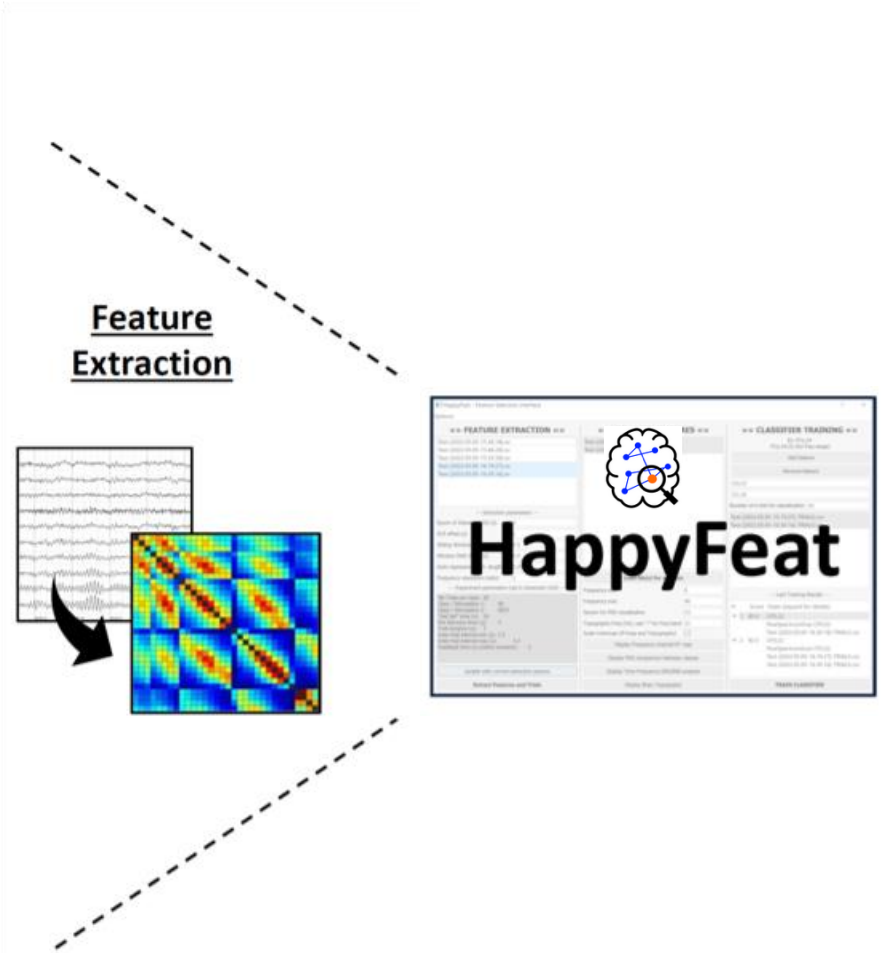
HappyFeat – Main Concepts

- Standard **local measures**

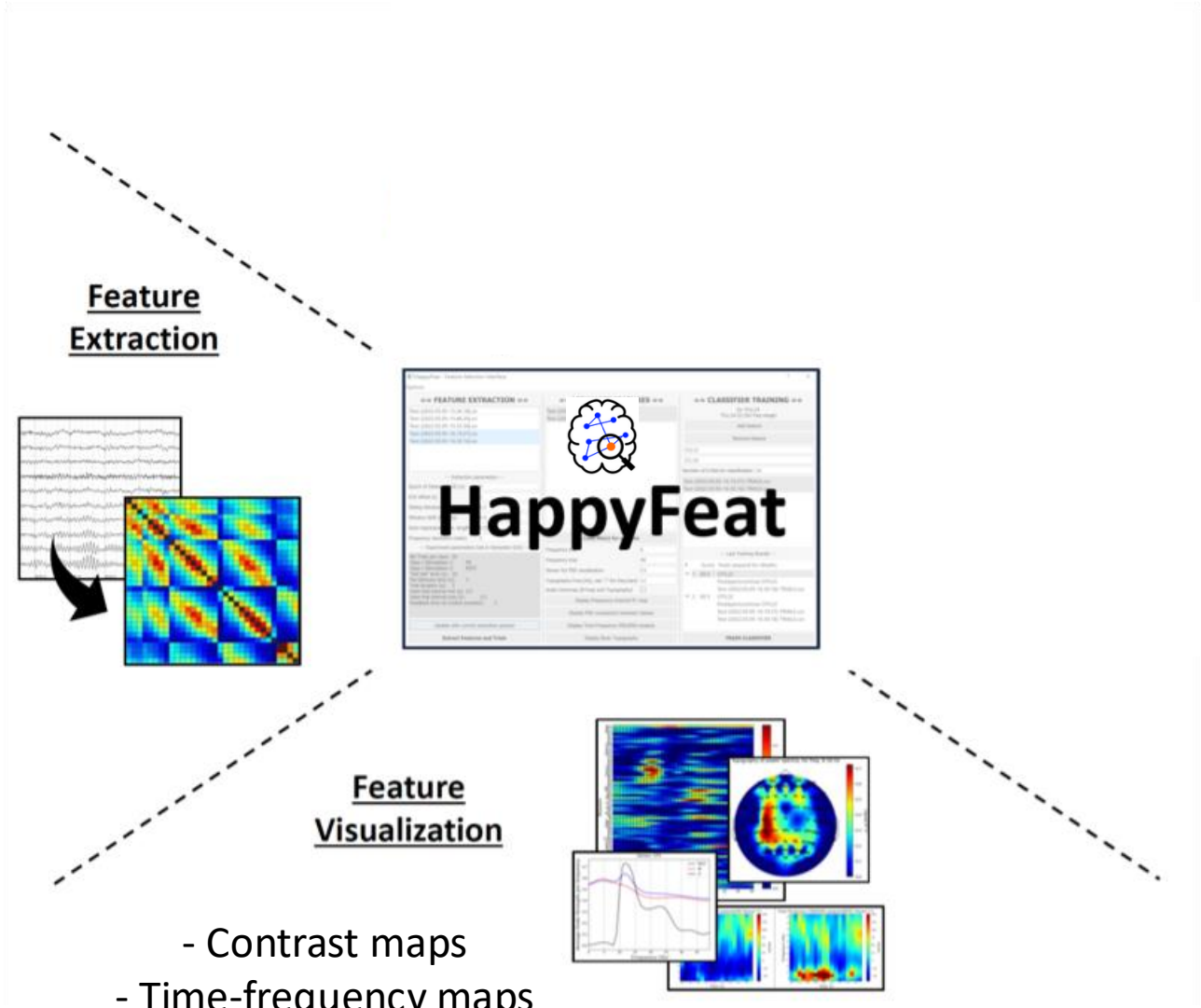


- **Network-based** approaches



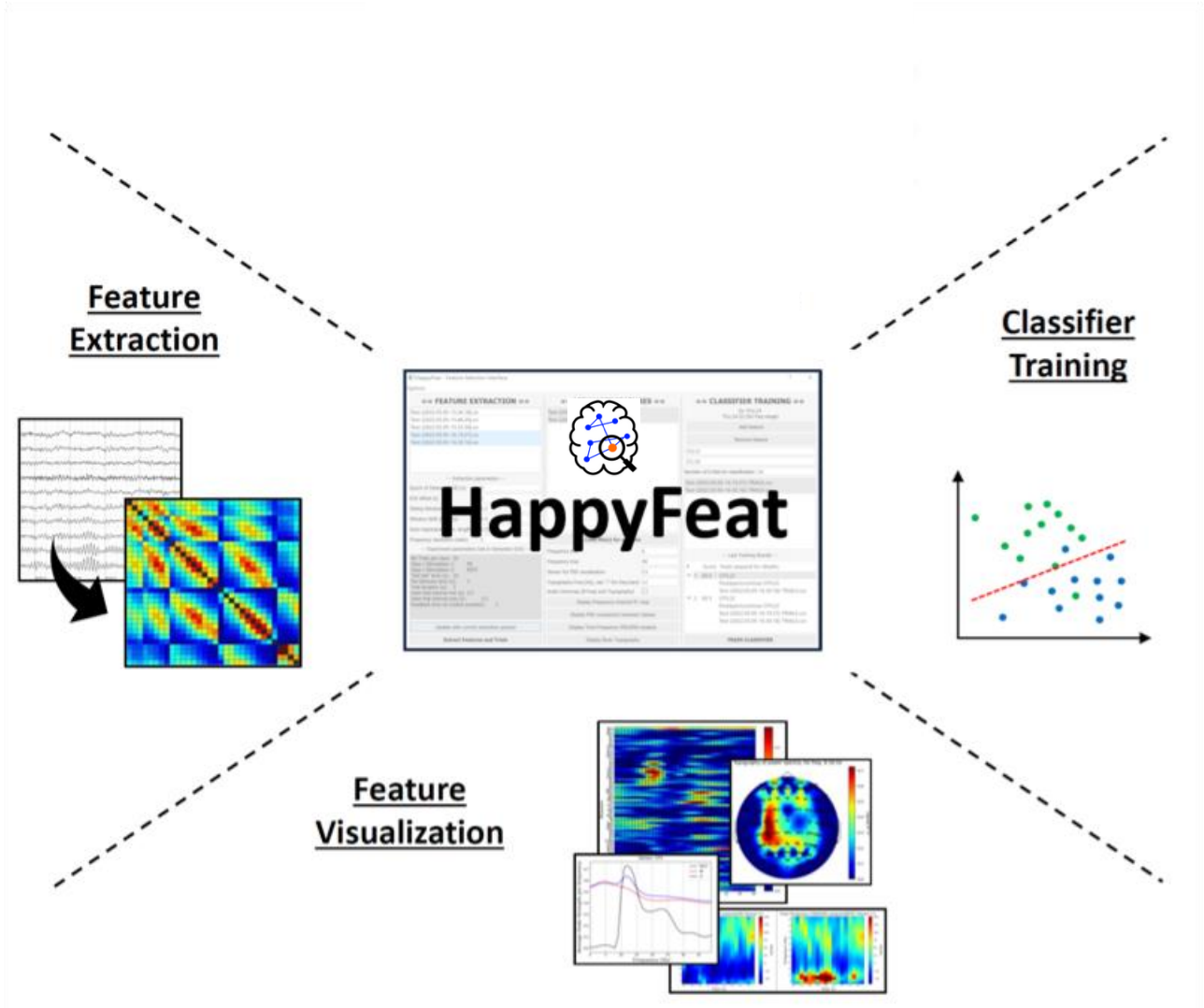


HappyFeat – Main Concepts



- Contrast maps
- Time-frequency maps
- Topographies

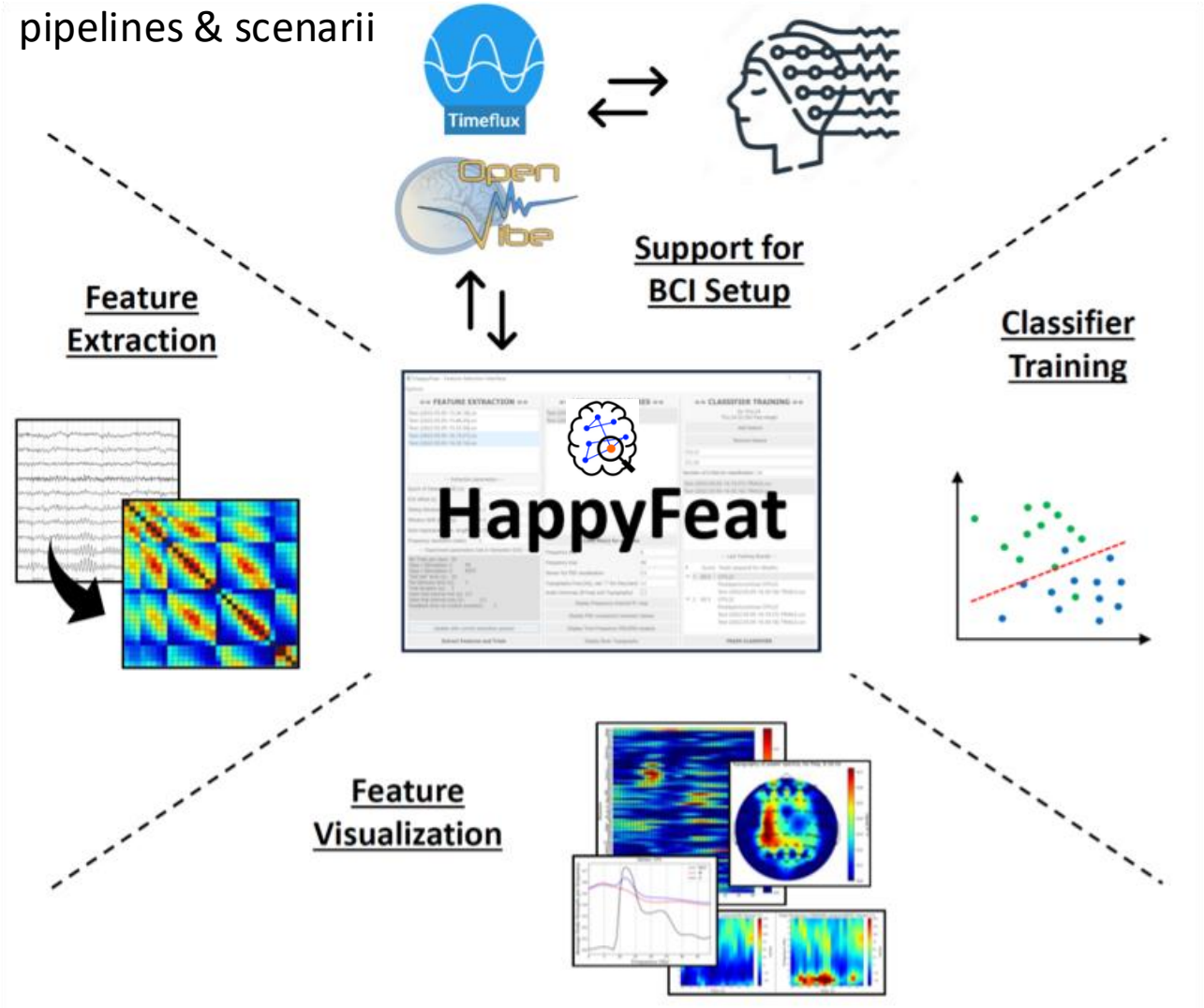
HappyFeat – Main Concepts



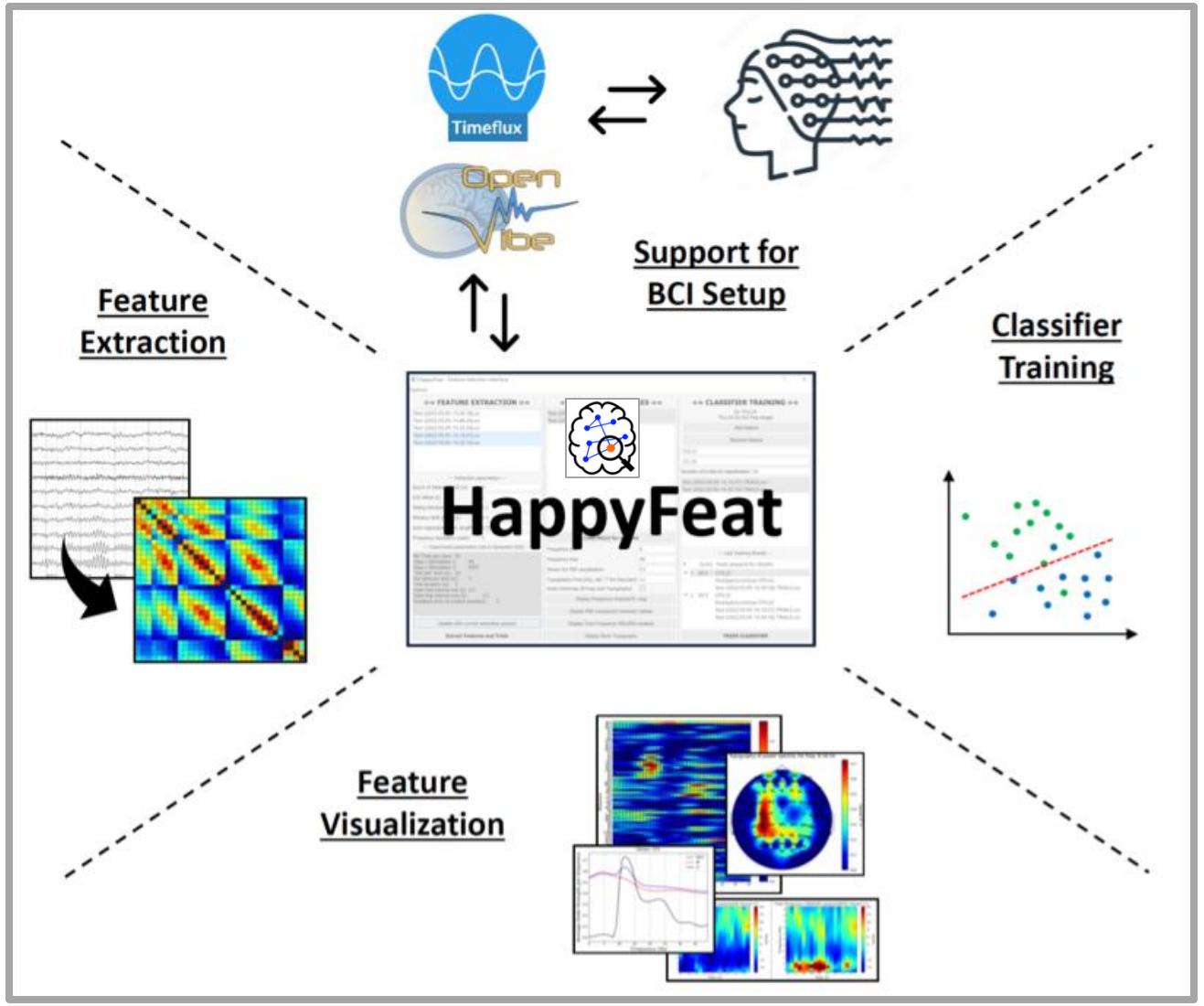
Tool for the decision

HappyFeat – Main Concepts

Automatic manipulation of pipelines & scenarii



HappyFeat – Main Concepts

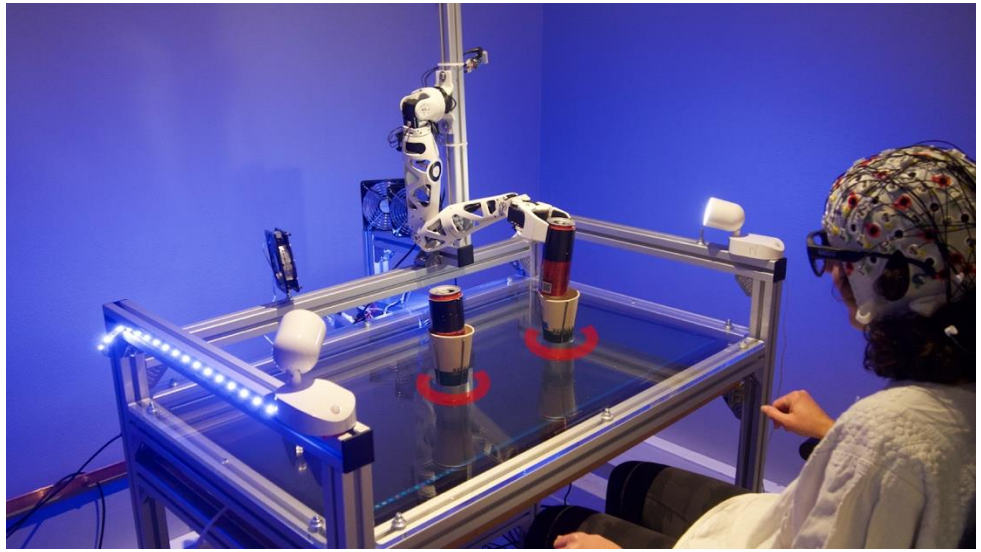


Workspace to ensure **reproducibility** and **replicability** of all the manipulations

HappyFeat – Experimental + clinical settings

- **2022-2024 : BRACCIO protocol (15 healthy subjects)**
  - Hybrid BCI system: **64 EEG channels + eye-tracking glasses** + video table **+ robotic arm**
  - **Themes:** Binding, sense of agency, mixing standard and connectivity-based features

→ **Venot et al. -** *Intentional binding for noninvasive BCI control*
(Journal of Neural Engineering, July 2024 - 10.1088/1741-2552/ad628c)

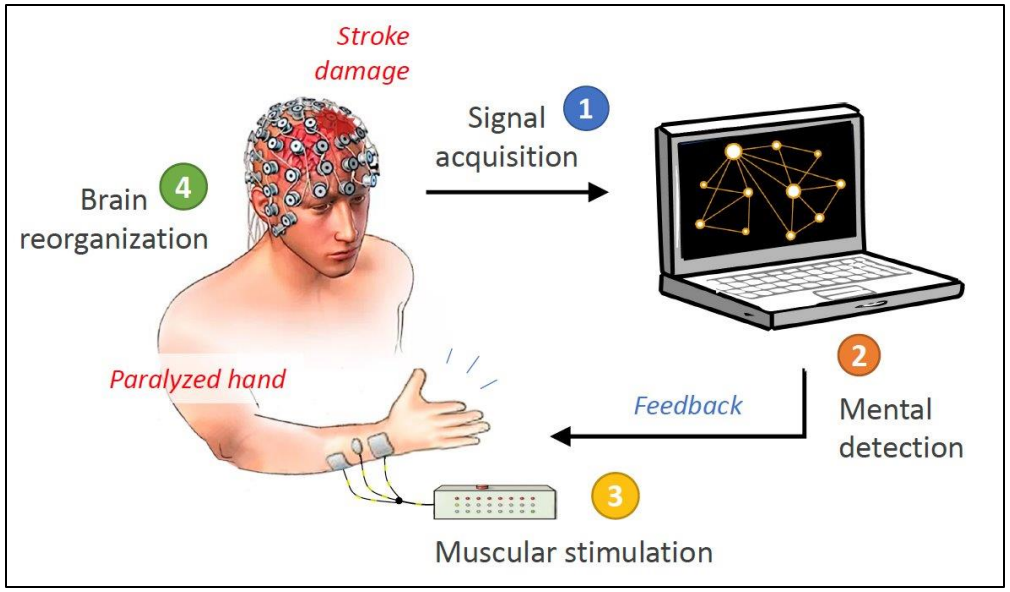


BRACCIO protocol – Venot et al., ICM/NERV

- **HappyFeat :**
  - **Setup** of (OpenViBE) scenarios (feat. extraction, visualization, training…)
  - **Helping the (manual) selection** of optimal classification features during the exp. (PSD)
  - **Acceleration & simplification of the protocol**

HappyFeat – Experimental + clinical settings

- **2025-202X : BCINET protocol**
  - **Post-stroke rehabilitation using BCI and rTMS.** 64 EEG channels + muscular stimulation + TMS
  - **1st part:** 5 patients x 18 sessions
    Feasibility, fatigue evaluation, evolution of PSD and Functional Connectivity (as imag(Coh))
  - **2nd part:** 45 patients x 18 sessions: Evolution of BCI performance and of motor functions



Fabrizio de Vico Fallani, ERC Horizon 2020

- **HappyFeat :**
  - **Setup** of (OpenViBE) scenarios
    (feat. extraction, visualization, training…)
  - **Semi-automatic selection** of optimal classification features, **using PSD and im(Coh)**
  - **Actually making the protocol feasible!**

Table of contents

HappyFeat - Technical status & useful links

GitHub repo
https://github.com/Inria-NERV/happyFeat

Article:
https://doi.org/10.1016/j.simpa.2023.100610

Try the tutorial!
https://happyfeat.readthedocs.io/en/latest/user_guide/tutorial/

HappyFeat – Technical status & useful links

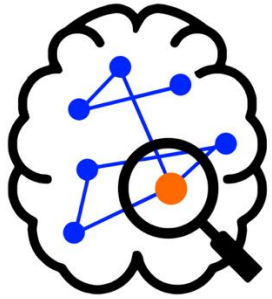
- **Version 0.3.0** (March 2025) – *patches soon, version 0.4.0 summer 2025*
- **Main dependencies :**
  - **Python 3.12.8**
  - **OpenViBE v3.6.0** or **Timeflux 0.17.2**
  - **MNE-python, PySide, Numpy, Plotly, Pandas** (full list on github…)

- **License**: Open Source BSD-3-Clause
- PyPi package: https://pypi.org/project/happyfeat/

- **Contributions** are possible (and welcome) via Github :
  - Standard, via fork + pull-requests
  - GitHub « issues » to track bugs, ideas, contributations

BCI Motor Imagery with HappyFeat in X-Men: First Class

Thanks for your attention! Any questions?



BCI Meeting 2025 - WS19 - June 5

HappyFeat: an interactive and efficient BCI framework for clinical applications

Additional material

Arthur Desbois

Inria Paris, NERV team, ICM (Paris Brain Institute)

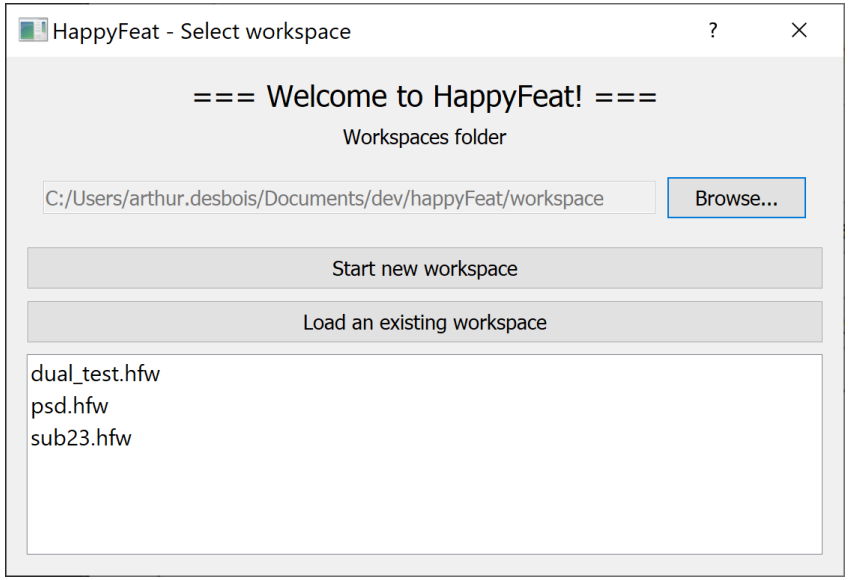
HappyFeat - Typical use case

1. **Workspace creation/selection**

- Create a new environment (file tree, configuration, data structures) for the experiment/analysis

  Ex: **sub23_session01**

- … or select an already existing one.



➔ All user settings & manipulations are saved and available in the workspace for future usage or sharing:

- Extraction parameters (spectral analysis, list of processed files…)
- Training attempts (runs & features used, training accuracy…)

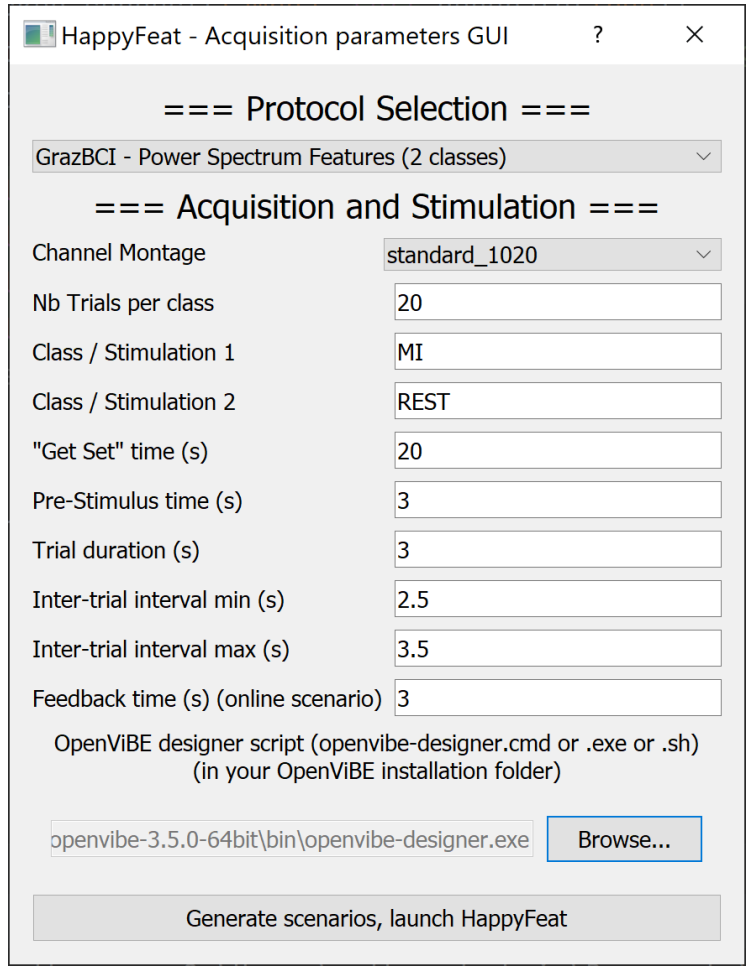
HappyFeat - Typical use case

1. Workspace creation/selection

- When creating a new workspace:
  - the feature type (PSD, Connectivity…) is chosen
  - experimental/acquisiton parameters are entered (electrode montage, stimulation names, experimental layout…)

➔ This step allows to automatically set up the OpenViBE acquisition scenario.

These experimental parameters are also important for the different processing steps of HappyFeat.



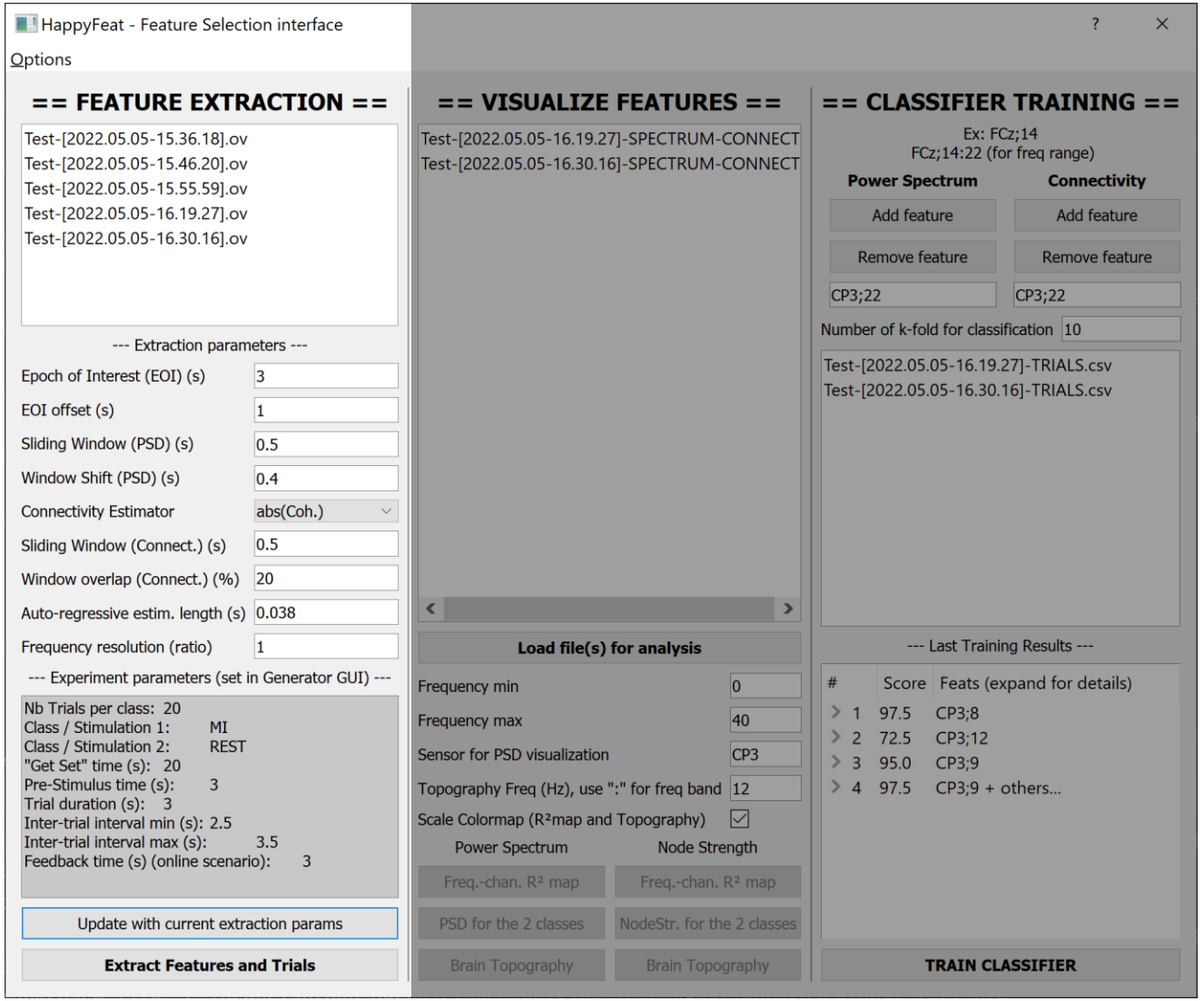
HappyFeat - Typical use case

1. Workspace creation/selection
2. **Feature Extraction**

- Files in the “signals” folder of the workspace are listed in the left panel
- Select one or multiple files…
- Enter the parameters for feature extraction…
- Press the button!

💡 Along the **workspace** mechanism, HappyFeat also uses **work sessions** to keep track of extraction parameters.

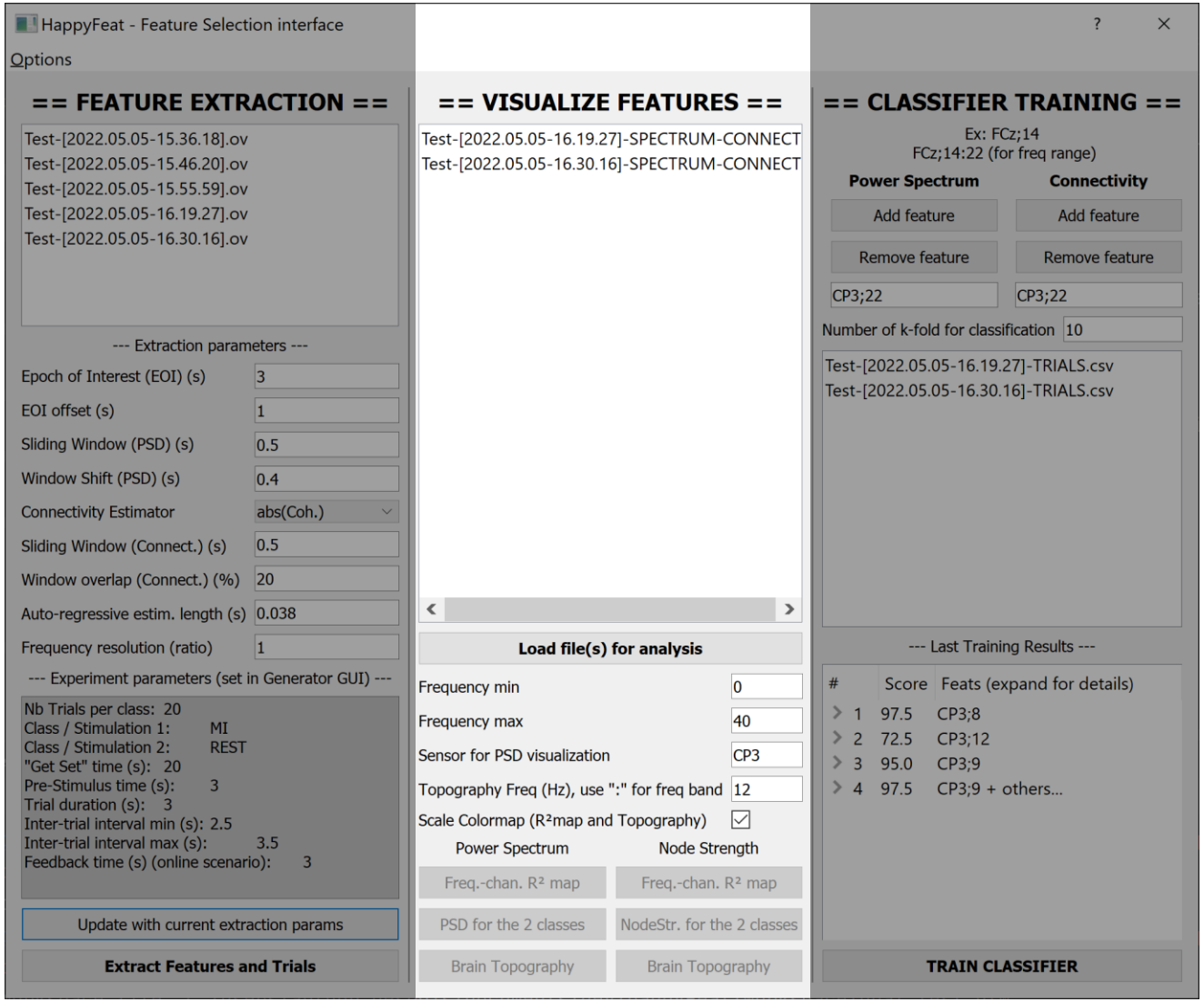
Therefore, all steps realized with a set of extraction parameters (which files have undergone extraction + training attempts) are stored and can be recovered.



HappyFeat - Typical use case

1. Workspace creation/selection
2. Feature Extraction
3. **Analysis & Feature Selection**

- "Extracted" files (runs) appear in the central panel.
- Select which runs to analyze…



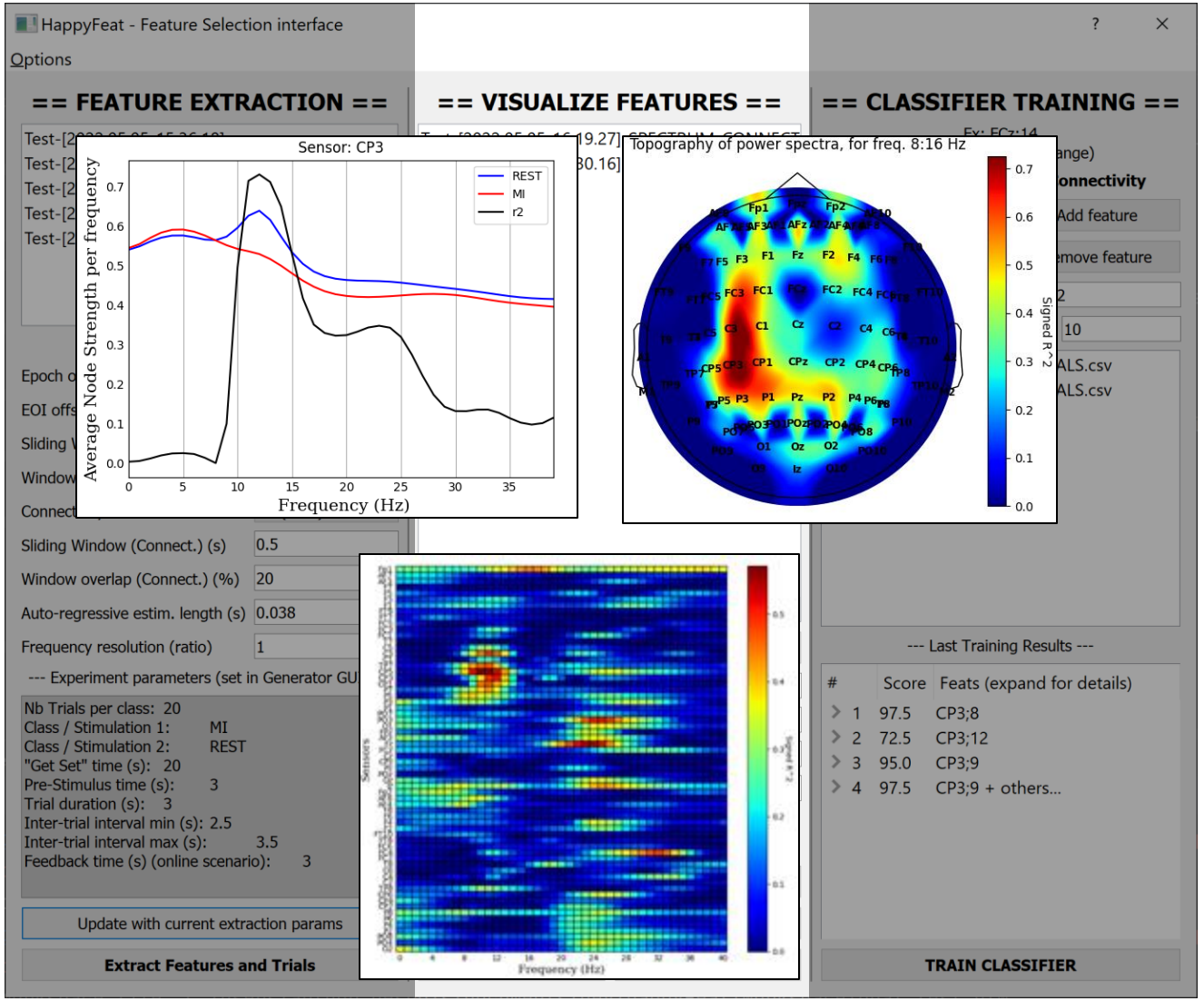
HappyFeat – Typical use case

1. Workspace creation/selection
2. Feature Extraction
3. **Analysis & Feature Selection**

- "Extracted" files (runs) appear in the central panel.
- Select which runs to analyze…

- And use the different **visualization tools** to select adequate features:
  - Channel/frequency R^2 map
  - Per-channel metric comparison
  - Per-frequency R^2 topography

  - More coming soon…



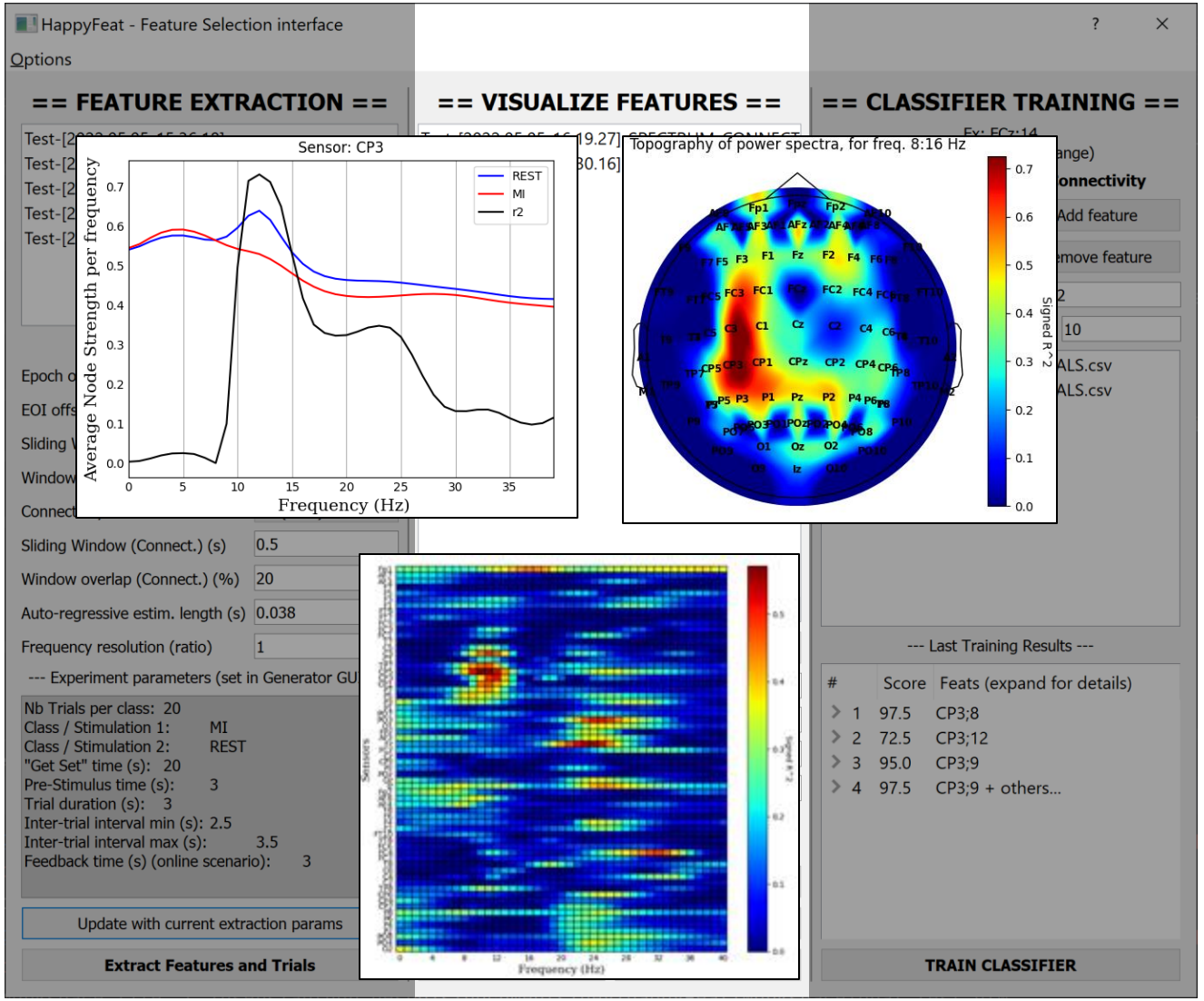
HappyFeat – Typical use case

1. Workspace creation/selection
2. Feature Extraction
3. **Analysis & Feature Selection**

- "Extracted" files (runs) appear in the central panel.
- Select which runs to analyze…

- And use the different **visualization tools** to select adequate features.

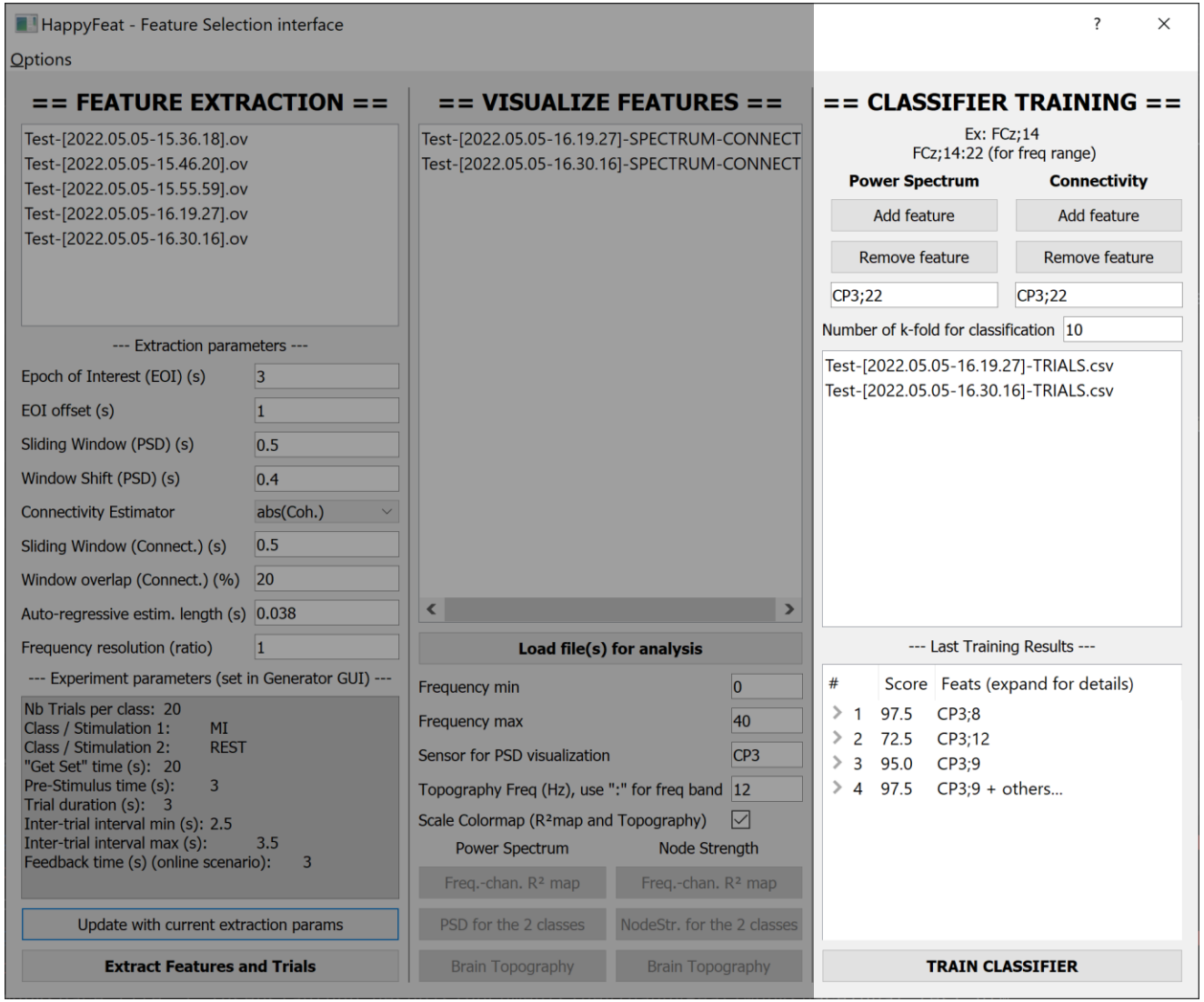
- … or use the "**AutoFeat**" mechanism to automatically select a set of (channel/frequency) pairs with the highest R2



HappyFeat - Typical use case

1. Workspace creation/selection
2. Feature Extraction
3. Analysis & Feature Selection
4. **Classifier training**

- "Extracted" files (runs) also appear in the right panel.
- Select one or multiple runs…
  … their trials will all be used for training.
- Select one or multiple feature(s) of interest (channel;frequency)
- Click the button!



HappyFeat - Typical use case

1. Workspace creation/selection
2. Feature Extraction
3. Analysis & Feature Selection
4. **Classifier training**

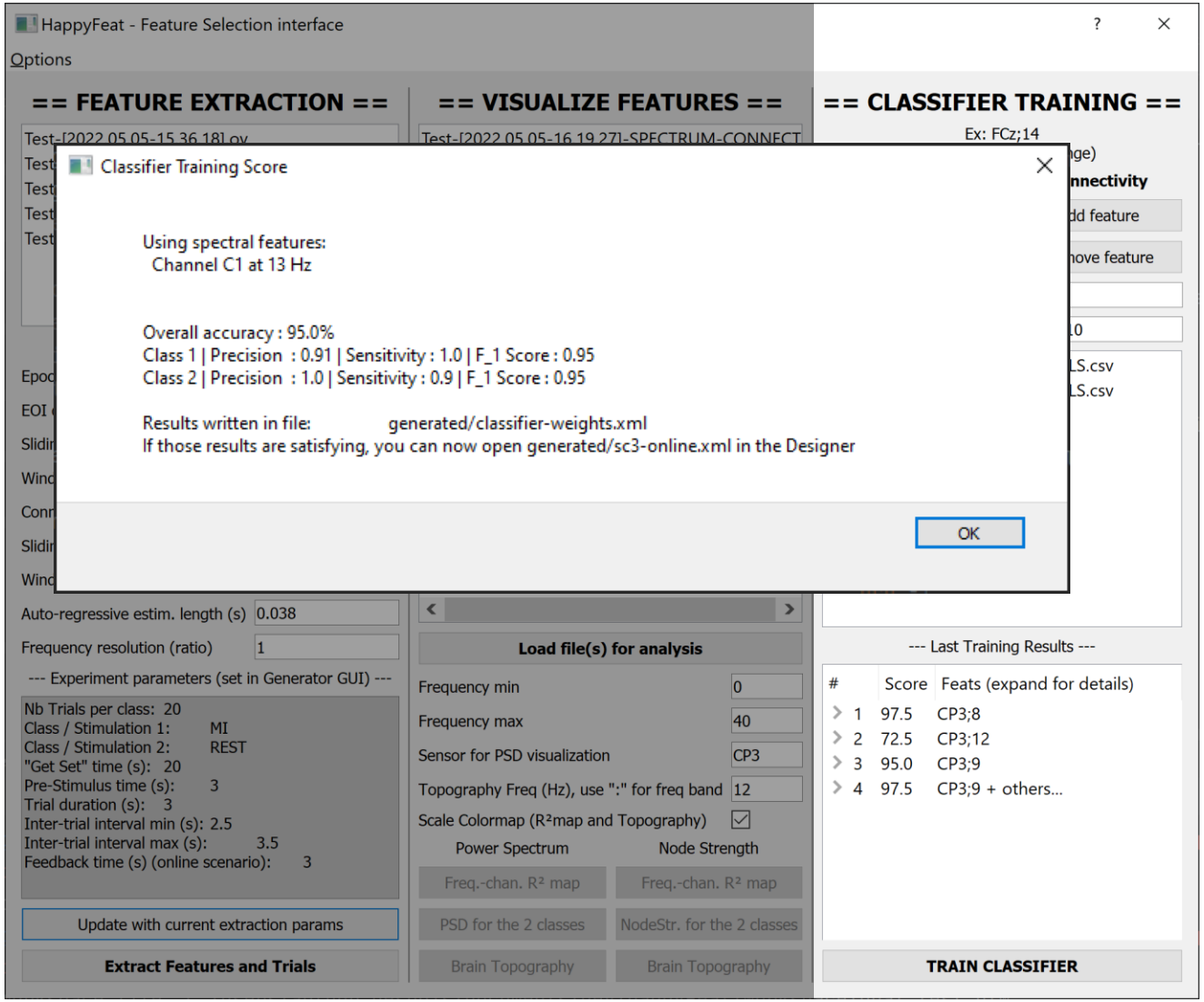
- "Extracted" files (runs) also appear in the right panel.
- Select one or multiple runs…
  … their trials will all be used for training.

- Select one or multiple feature(s) of interest (channel;frequency)

- Click the button!

➔ A training accuracy is provided

➔ All previous training results are available (displayed in the GUI, and classif. weights in the workspace)



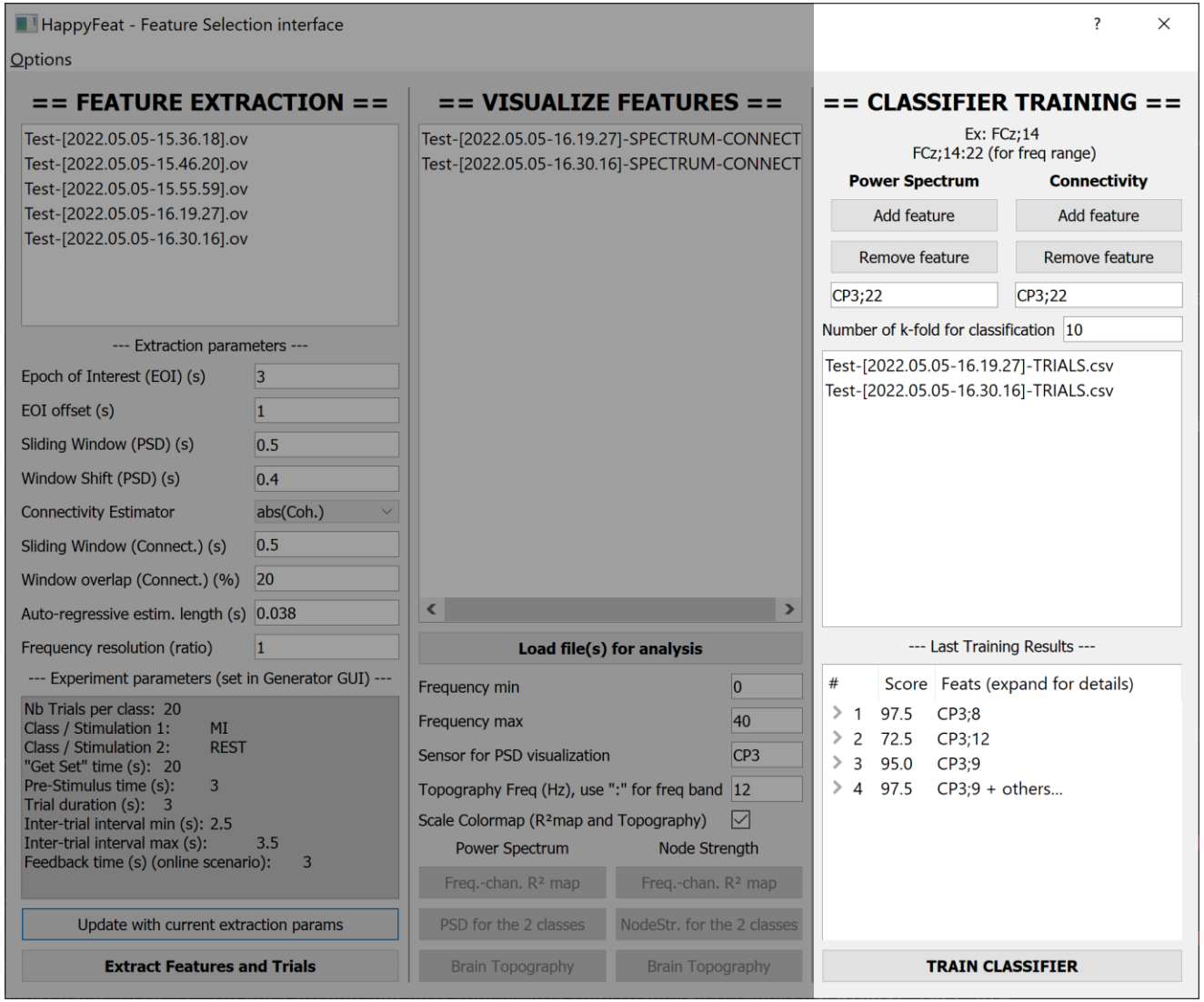
HappyFeat – Typical use case

1. Workspace creation/selection
2. Feature Extraction
3. Analysis & Feature Selection
4. Classifier training
5. **Online classification**

A satisfactory set of classification features has been found!

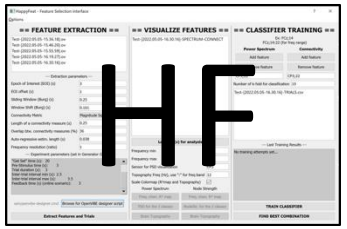
➔ Generate a **ready-to-use** OpenViBE (soon TimeFlux) scenario

- trained classifier
- selected classif.features



HappyFeat – How? (with openvibe)

1. **Workspace creation/selection**

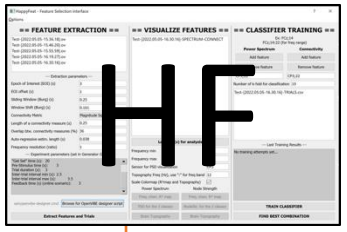


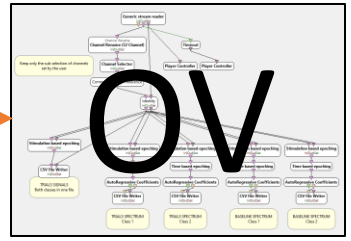
- Create a new environment (file tree, configuration, data structures) for the experiment/analysis
- … or select an already existing one.
- All user settings & manipulations are saved and available in the workspace for future usage or sharing:
  - Extraction parameters (spectral analysis, list of processed files…)
  - Training attempts (runs & features used, training accuracy…)

HappyFeat - How? (with openvibe)

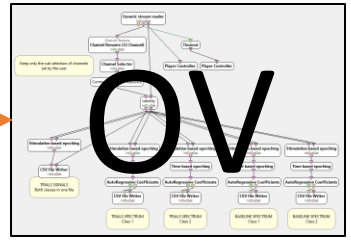
1. Workspace creation/selection
2. **MI Pipeline / “Feature type” selection**

- Selecting btw. multiple “template” scenarios depending on the use case (power spectrum, connectivity type…)
- Edit basic/common parameters (acquisition, extraction, training…)

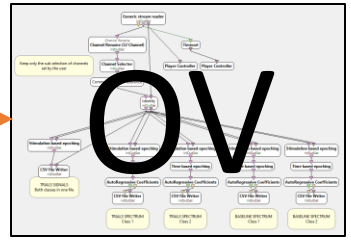




Extraction scenario



Training scenario

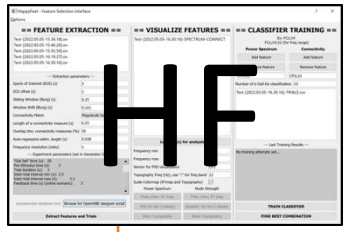


Online classif. scenario

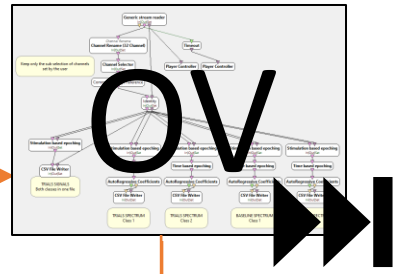
HappyFeat – How? (with openvibe)

1. Workspace creation/selection
2. MI Pipeline / “Feature type” selection
3. **Feature Extraction**

- **Select signal files**, and **extraction parameters** (lengths and overlap of windows, FFT size…)
- **Run** the generated extraction OpenViBE scenario (**in the background**) for all selected signal files:
  - ➔ **Extract metadata** (sampling freq, electrodes…)
  - ➔ Cut the signal to **regions of interest** (MI trials & baseline portions), **generate CSV file** with only these chunks (for the training step)
  - ➔ Apply a **signal processing** pipeline (PSD computation, connectivity measure…) to the signal chunks of interest, **generate CSV files for future analysis**
- **Runs in an autonomous thread:**
  You can do visualizations and training attempts for signals already processed in the meantime.



Extraction scenario



Signal 1 (.ov)

- Spectral Power (class 1) CSV
- Spectral Power (class 2) CSV
- Trials CSV
- Metadata CSV

Signal 2 (.ov)

- Spectral Power (class 1) CSV
- Spectral Power (class 2) CSV
- Trials CSV
- Metadata CSV

…

HappyFeat – How? (with openvibe)

1. Workspace creation/selection
2. MI Pipeline / “Feature type” selection
3. Feature Extraction
4. **Analysis, Feature Selection**

- Select one or multiple signals & load their spectral/connectivity data (CSV work files generated during “Feature Extraction”)

- Use different **Visualization Tools** to help find & select **features of interest** (FOIs) for training

  ➔ Frequency/channel R^2 map

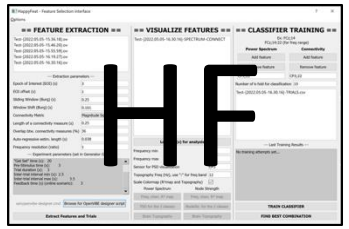
  ➔ PSD (or connect. metric) comparison btw. the 2 conditions (MI/REST) for a given electrode

  ➔ Time/frequency ERD/ERS analysis for each condition

  ➔ R^2 mapped as a brain topography for a given frequency (or range)

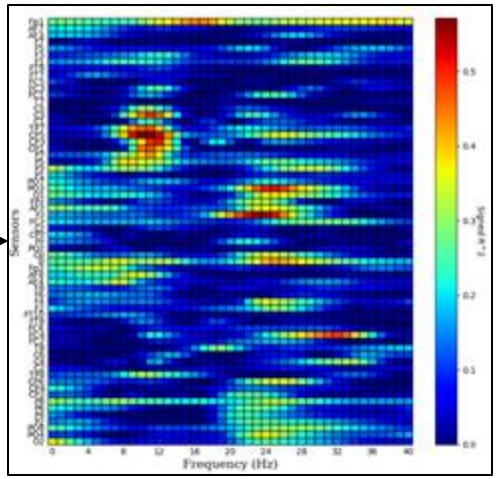
Combine as many visualization windows as you need

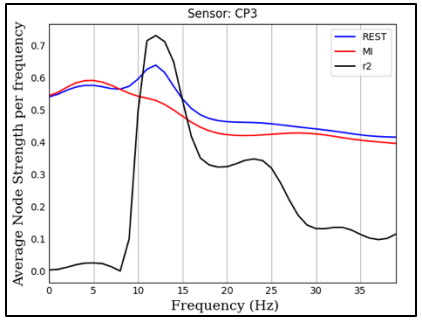
A “Dual metric” pipeline allows to show (for ex.) R^2 maps for both Power Spectrum and Connectivity in parallel

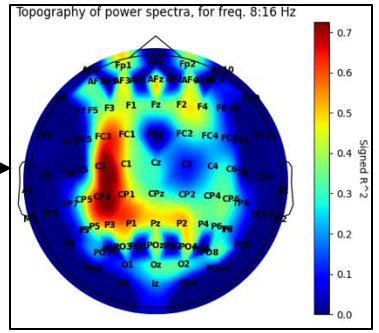


Signal 1
(.ov)

- Spectral Power (class 1) CSV
- Spectral Power (class 2) CSV
- Trials CSV
- Metadata CSV







HappyFeat – How? (with openvibe)

1. Workspace creation/selection
2. MI Pipeline / “Feature type” selection
3. Feature Extraction
4. Analysis, Feature Selection
5. **Classifier Training**

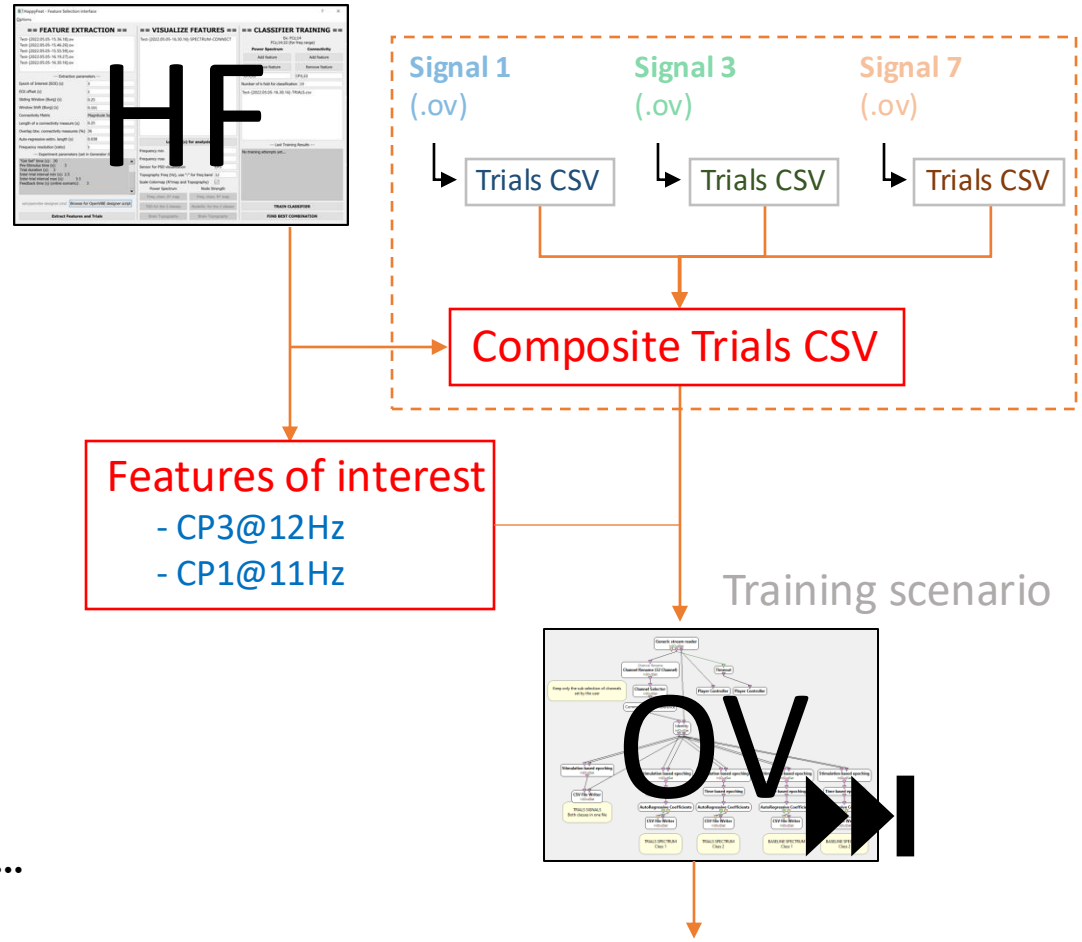
- Set one or more **Features of Interest** (Electrode/Freq.)

- **Select file(s)** with which you want to train your classifier
  If > 1 file : their trials **are automatically concatenated**

- **Run the Classifier Training scenario**
  (Auto. generated in step 1 + auto. edited with FOIs)
  ➔ Classification **ACCURACY** + **WEIGHTS**

Disappointing results? (“My accuracy is 50% ☹”)
Maybe try again with other features. **It only takes a few seconds…**

Satisfying results?! (“OMG 95%”)
Good news! The “Online Classification” scenario has already been automatically been updated with:
- Classifier training weights
- Features of interest used for training



Training accuracy “**score**”
+ Classifier **Weights**
+ **Online scenario updated**

Equipe et contributeurs

- **Arthur Desbois** (ingénieur Inria) - lead
  - Principal développeur et mainteneur de l'application (GUI et back end)
  - Supervision des travaux, gestion de projet, tests, packaging, documentation…

- **Tristan Venot** (PhD, ingénieur Inria)
  - Protocoles expérimentaux BCI (Braccio)
  - Aide au développement algo. et à l'amélioration de l'application (GUI et back end)

- **Wafa Skhiri** (stage M2 2024, étudiant PhD à partir de 2025)
  - Compatibilité HappyFeat + Timeflux
  - A partir de 2025 : aide au développement algo. (connectivité, graphes)

- **Camile Bousfiha** (étudiant PhD depuis 2023)
  - Protocole BCINET (50 patients, 18 sessions) utilisant HappyFeat

(+ Marie-Constance Corsi (PI) et Fabrizio De Vico Fallani (PI) – orientations stratégiques)