



MOABB

Benchmark your BCI algorithms on a rich collection of Motor Imagery, ERP, SSVEP and c-VEP datasets

Workshop 2: Designing Brain-Computer Interfaces, from theory to real-life scenarios

Pierre Guetschel

Data-Driven NeuroTechnology Lab, Radboud University, Donders Institute, Nijmegen, Netherlands
Currently affiliated with META, Paris, France

Mother Of All BCI Benchmarks (MOABB)

- Python library
- to compare/benchmark
- classification algorithms
- used for Brain-Computer Interfaces (BCI).

Design Philosophy

Reproducible research in BCI built on a rich Python ecosystem to design FAIR benchmarks with the help of a community.

Design Philosophy

Reproducible research in BCI built on a rich Python ecosystem to design FAIR benchmarks with the help of a community.

Why open source matters



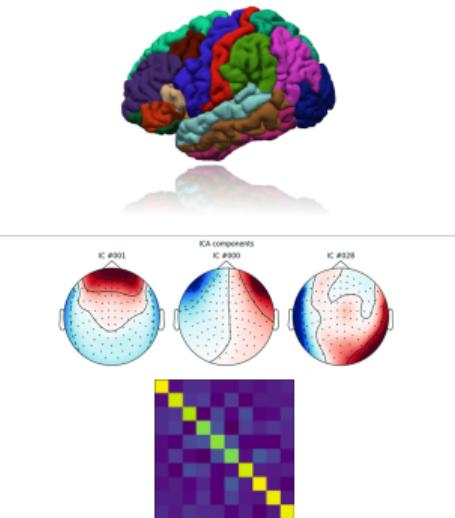
Reproducibility issues

Freesurfer Popular software for extracting features from MRI
→ Software variation lead to different conclusions

ICA Popular matrix factorization problem
→ Different results with different machines

eigs/eigsh Popular solver for eigenvalues decomposition
→ Solvers can lead to different outcome

Neurophysiological analysis is complex, require advanced processing
⇒ **Need for collective efforts to build open science**



Why Do We Need MOABB?

Reproducible research in BCI has a long way to go...

- Unavailable code
- Exotic data format/data structure/language/toolboxes
- Preprocessed data (including errors)

No comprehensive benchmark of BCI algorithms

Huge waste of time for everyone

⇒ MOABB aims to be the standard benchmark for any new paper

- Comprehensive benchmark of popular BCI algorithms
- Extensive list of freely available EEG datasets
- Ranking algorithms with fair evaluations

Built on a rich ecosystem

Design Philosophy

Reproducible research in BCI built on a rich Python ecosystem to design FAIR benchmarks with the help of a community.

<https://github.com/mne-tools/mne-python>

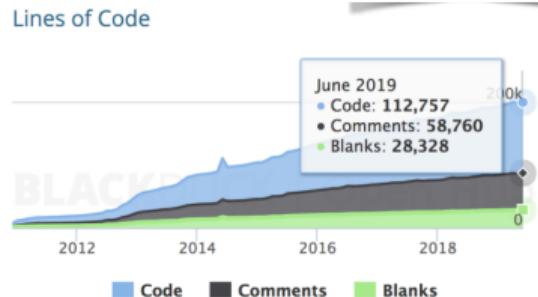
History

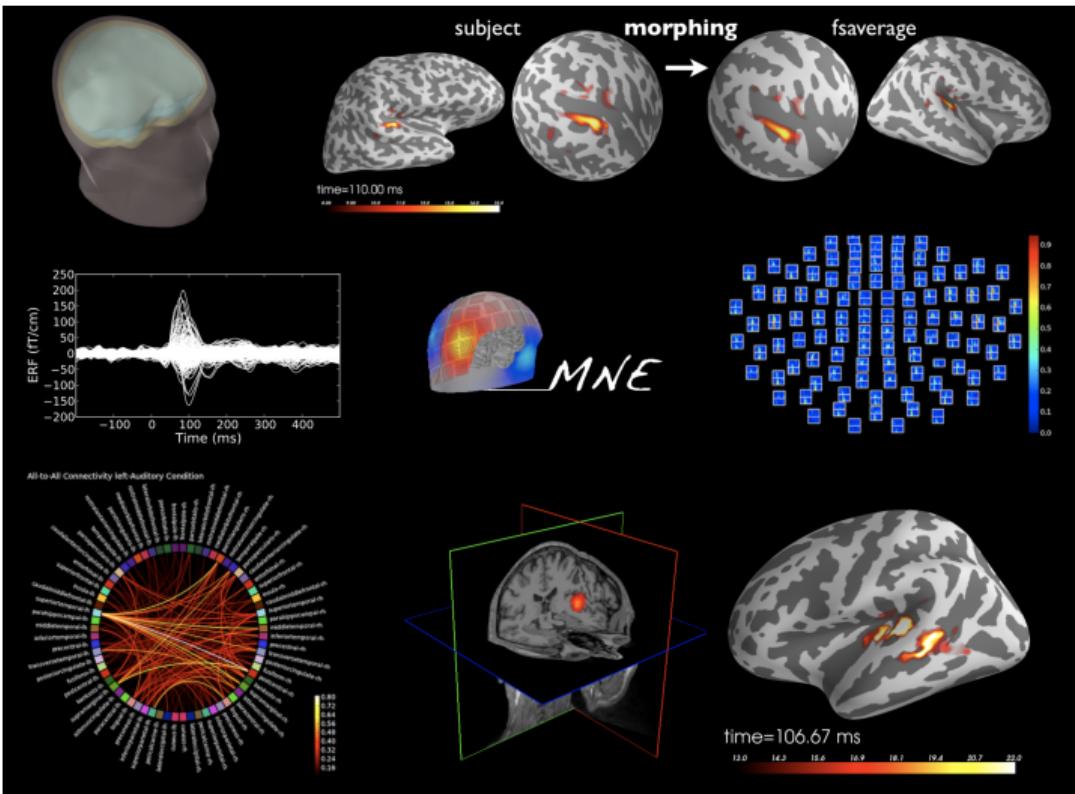
- based on C code developed for 18 years by Matti Hämäläinen
- Python started in 2010 at MGH, Boston

In a nutshell

- 361 contributors, 280k LOC
- mature codebase, large dev team
- ~ 33 years of efforts (COCOMO)

⇒ BSD licensed (commercial use ok)
⇒ Mac / Linux / Windows





Scikit-learn – accessible machine learning

<http://scikit-learn.org>

■ Machine learning for all

- ⇒ No specific application domain
- ⇒ No requirements in machine learning

■ High-quality Pythonic software library

- ⇒ Interfaces designed for users

■ Community-driven development

- ⇒ BSD licensed, very diverse contributors

Easy:

```
1 from sklearn.svm import SVC  
2 classifier = SVC()  
3 classifier.fit(X_train, Y_train)  
4 Y_test = classifier.predict(X_test)
```

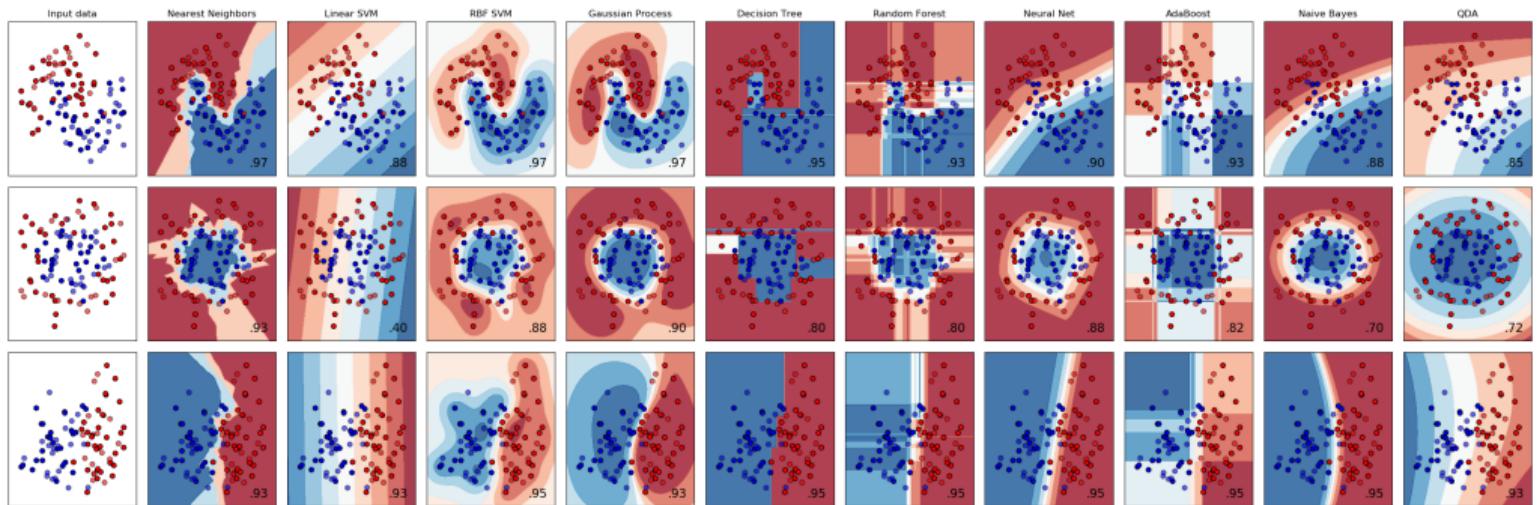
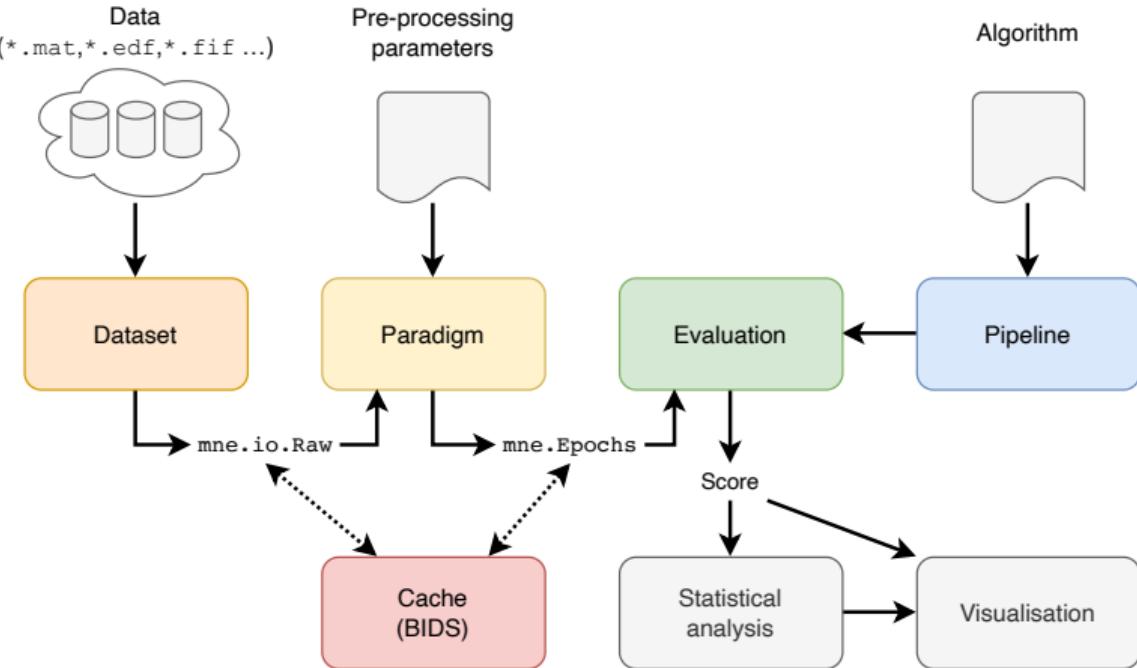


Figure: Comparison of several Scikit-learn classifiers on synthetic datasets.

Design Philosophy

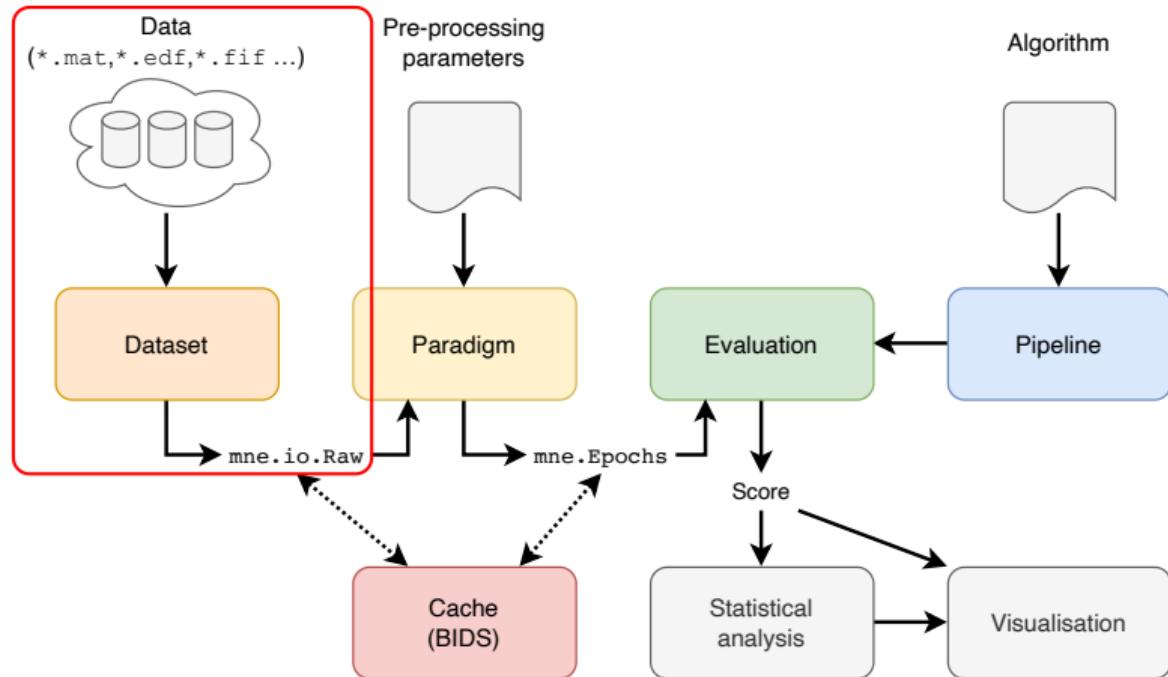
Reproducible research in BCI built on a rich Python ecosystem **to** design FAIR benchmarks with the help of a community.

MOABB Architecture



⇒All the components of MOABB

MOABB Architecture: Datasets



Dataset

- Stored locally, converted in MNE format
- Pick only subjects you need

MOABB Architecture: Datasets

Dataset list



- Wrapper around any data format (.mat, .edf, .fif, etc.);
- 16 Motor Imagery, 16 ERP, 7 SSVEP and 1 c-VEP datasets available;
- Only public datasets;
- Only un-processed datasets.

MOABB Architecture: Datasets

Dataset list



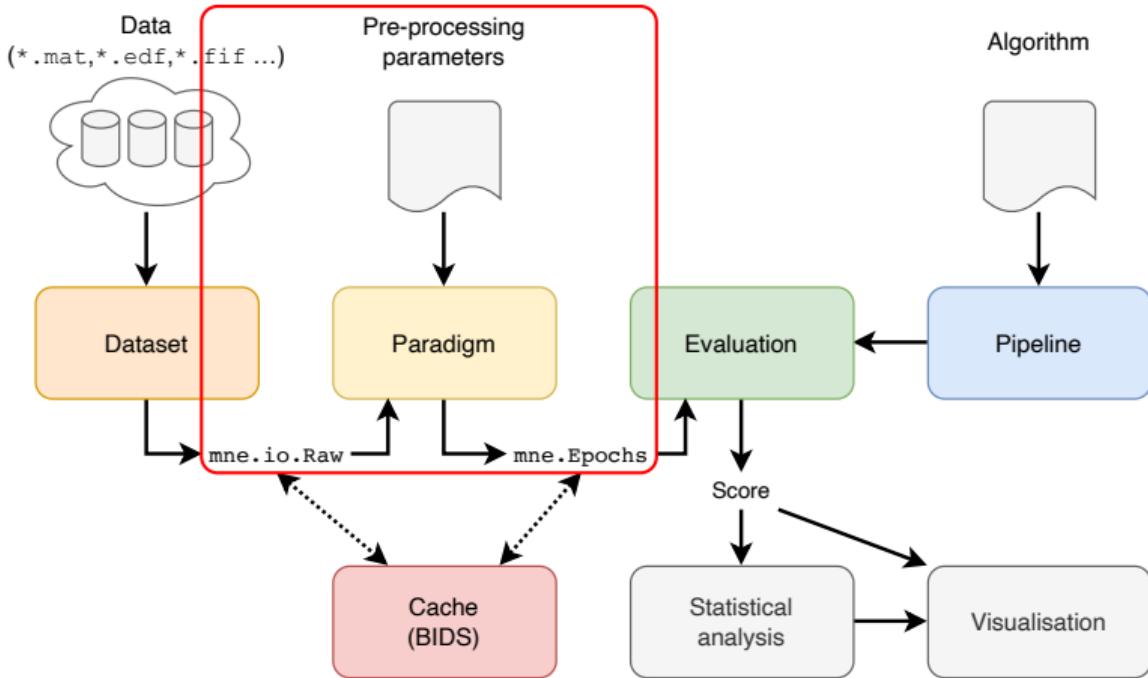
- Wrapper around any data format (.mat, .edf, .fif, etc.);
- 16 Motor Imagery, 16 ERP, 7 SSVEP and 1 c-VEP datasets available;
- Only public datasets;
- Only un-processed datasets.

```
1 from moabb.datasets import BNCI2014_001
2 dataset = BNCI2014_001()
3 data = dataset.get_data(subjects=[1, 2])
4
5 data # dict[int, dict[str, dict[str, mne.io.Raw]]]
```

Returns nested dictionary containing:

- subjects,
 - sessions of the subjects,
 - and runs within the sessions.

MOABB Architecture: Paradigm



Paradigm

- Motor Imagery, P300, SSVEP, c-VEP
- Preprocessing

MOABB Architecture: Paradigm

```
1 from moabb.paradigms import MotorImagery
2 paradigm = MotorImagery(fmin=1, channels=['C3', 'C4'])
3 X, labels, metadata = paradigm.get_data(dataset)
4
5 X          # numpy.ndarray
6 labels     # list[str]
7 metadata   # pandas.DataFrame
```

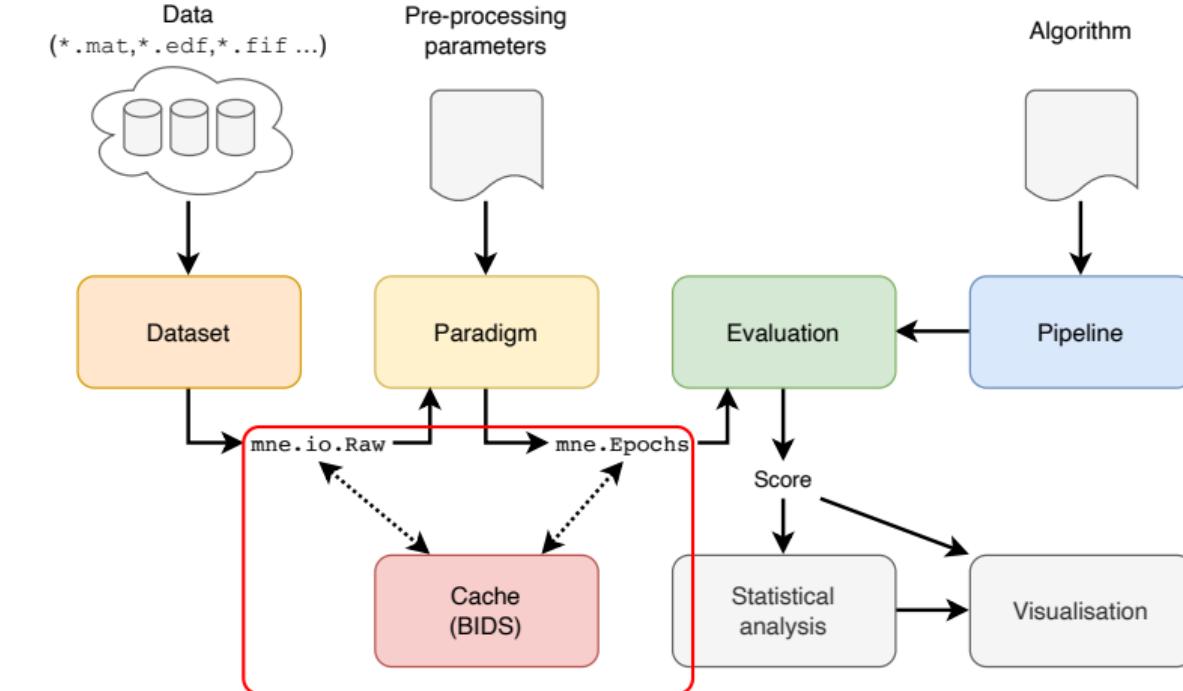
MOABB Architecture: Paradigm

```
1 from moabb.paradigms import MotorImagery
2 paradigm = MotorImagery(fmin=1, channels=['C3', 'C4'])
3 X, labels, metadata = paradigm.get_data(dataset)
4
5 X          # numpy.ndarray
6 labels     # list[str]
7 metadata   # pandas.DataFrame
```

Alternatively, you can get MNE epochs:

```
1 epochs, _, _ = paradigm.get_data(dataset, return_epochs=True)
2
3 epochs      # mne.Epochs
```

MOABB Architecture: Cache ★new★



Cache

- Disk cache: **raw** (.edf) or **pre-processed** (_epo.fif or .npy)
- In **BIDS** format for easy export.

MOABB Architecture: Cache ★new★

Time for one subject of Zhou2016

3.8 seconds

Without cache

1.4 seconds

Raw

```
1 data = dataset.get_data(  
2     cache_config={'save_raw': True, 'use': True})
```

Epochs

0.8 seconds

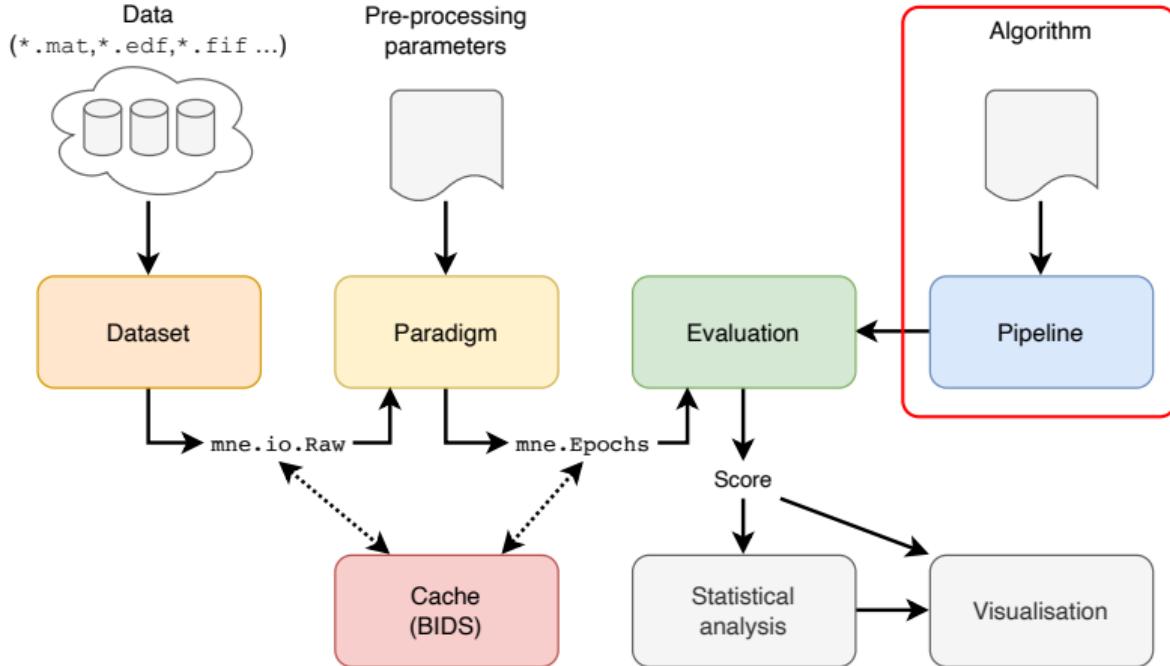
```
1 epochs, _, _ = paradigm.get_data(  
2     dataset,  
3     return_epochs=True,  
4     cache_config={'save_epochs': True, 'use': True})
```

Numpy array

0.6 seconds

```
1 X, _, _ = paradigm.get_data(  
2     dataset,  
3     cache_config={'save_array': True, 'use': True})
```

MOABB Architecture: Pipelines



Pipelines

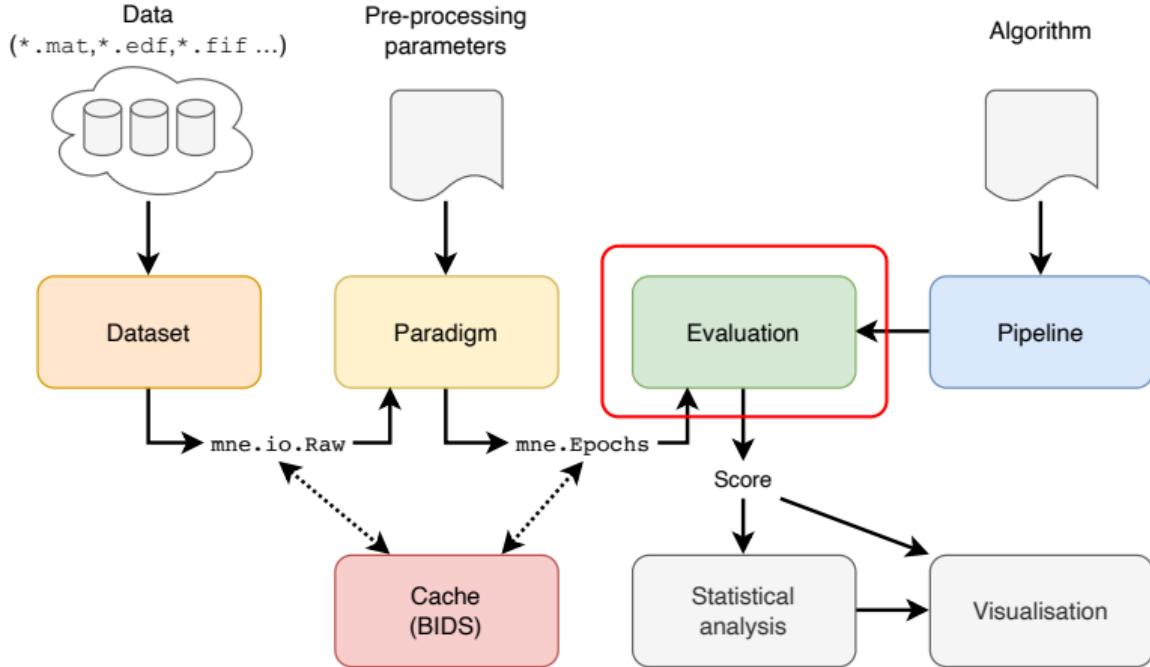
- All steps required for obtaining a prediction
- Scikit-learn style

MOABB Architecture: Pipelines

A Scikit-learn pipeline example:

```
1 from moabb.pipelines.features import LogVariance
2 from sklearn.svm import SVC
3 from sklearn.pipeline import make_pipeline
4 classifier = make_pipeline(
5     LogVariance(), # step 1
6     SVC(),          # step 2
7 )
8
9 classifier # sklearn Pipeline
```

MOABB Architecture: Evaluations



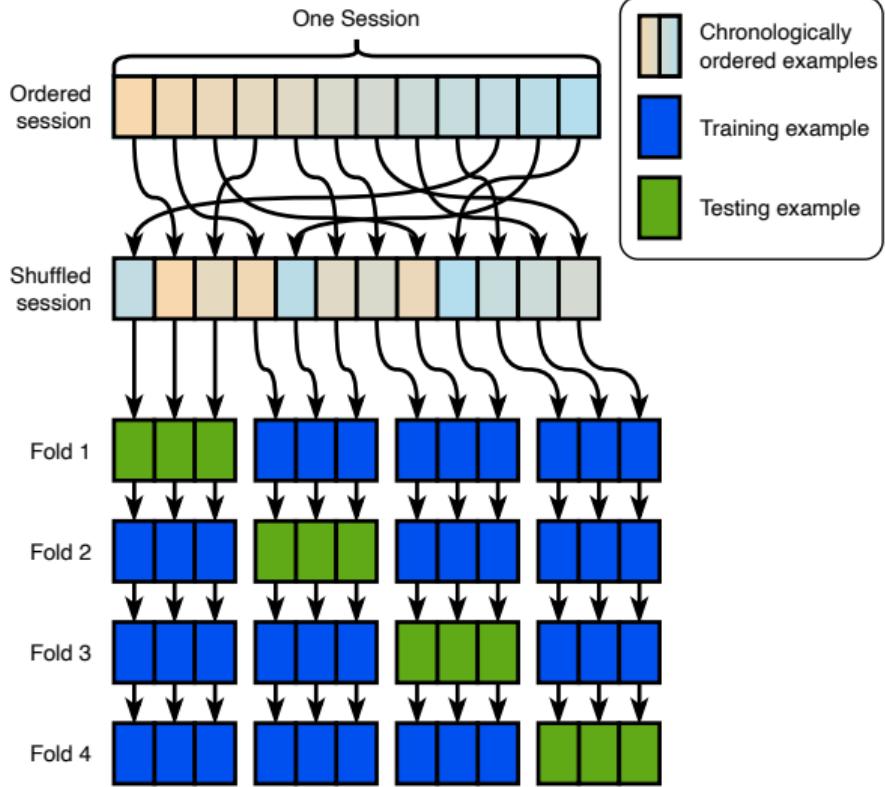
Evaluations

- Defines a scoring method (AUC, accuracy, ...)
- within or across session, across-subject, ...

MOABB Architecture: Evaluations – Within-Session



- Within-session,
- Shuffled,
- 5-folds cross-validation.

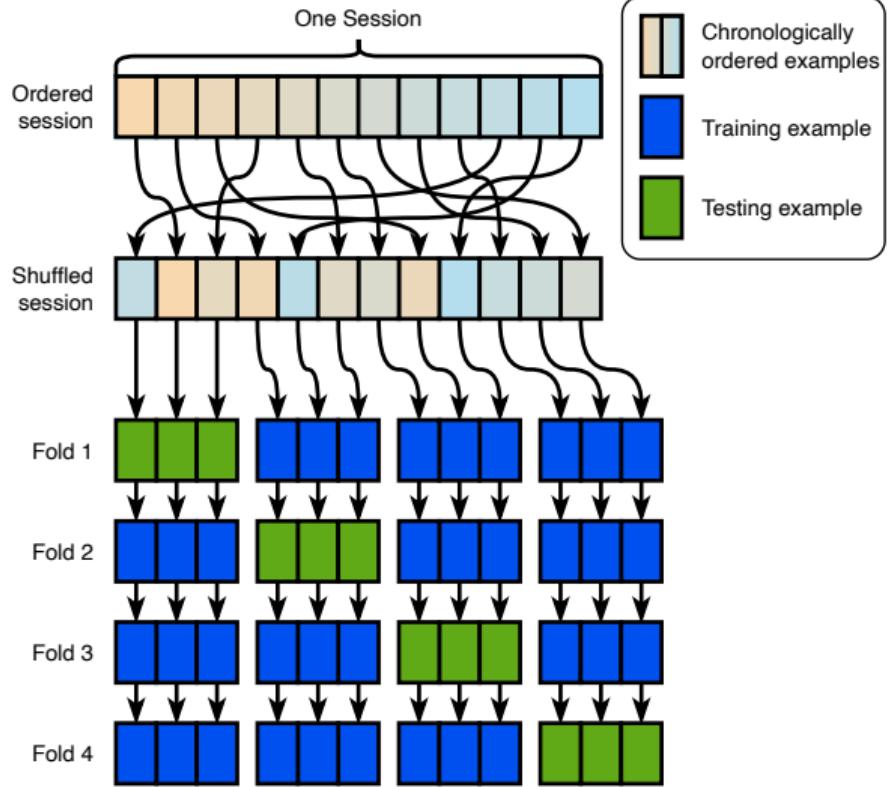


MOABB Architecture: Evaluations – Within-Session

- Within-session,
- Shuffled,
- 5-folds cross-validation.

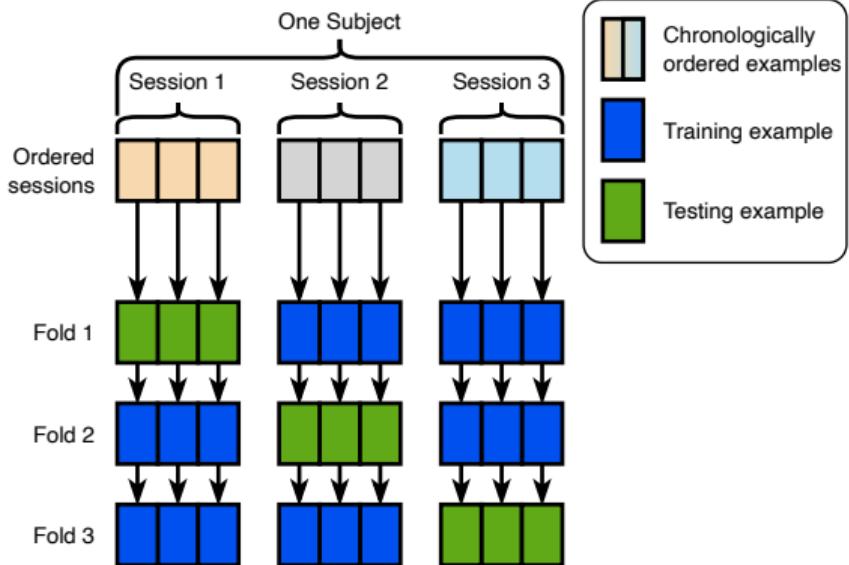
Arriving soon:

- Within-session,
- Pseudo-online evaluation,
- (respecting chronological aspect of data).



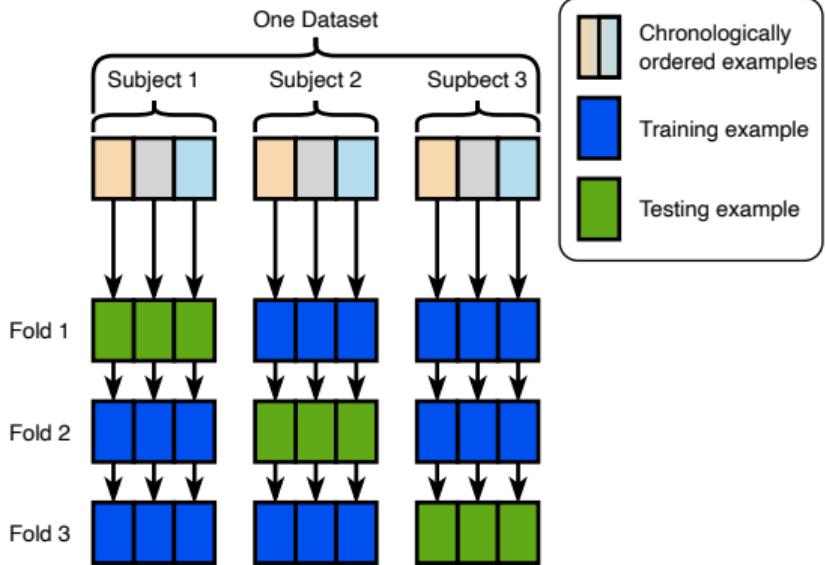
MOABB Architecture: Evaluations – Cross-Session

- Within-subject,
- Leave-one-session-out cross-validation.



MOABB Architecture: Evaluations – Cross-Subject

- Within-dataset,
- Leave-one-subject-out cross-validation.



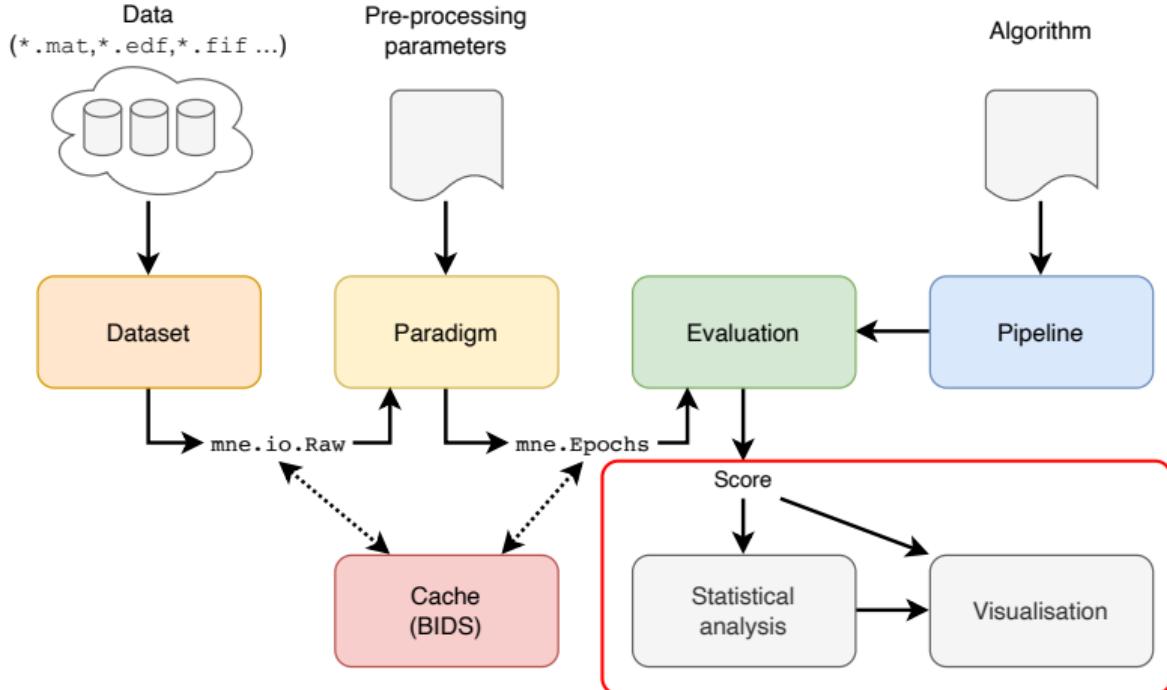
MOABB Architecture: Evaluations

```
1 from moabb.evaluations import WithinSessionEvaluation
2 evaluation = WithinSessionEvaluation(
3     paradigm=paradigm, datasets=[dataset])
4 results = evaluation.process({'AM+SVM': classifier})
5
6 results # pandas.DataFrame
```

Results are easy to save as .csv

```
1 results.to_csv('my_results.csv')
```

MOABB Architecture: Results

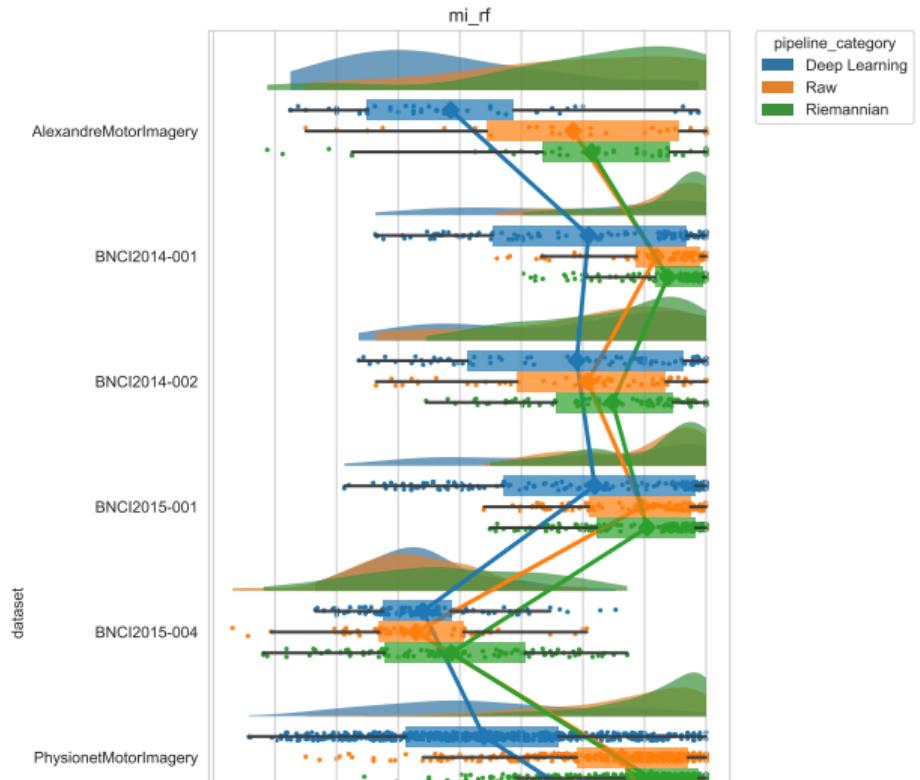
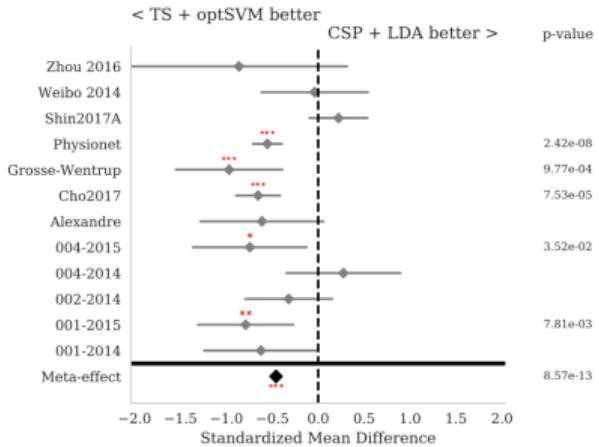


Results

- Statistics & visualization
- Results are stored in a DataFrame

Fair and Reproducible Benchmarks

1. Load multiple datasets
2. Apply pipelines
3. Run meta-analysis and plot



Design Philosophy

Reproducible research in BCI built on a rich Python ecosystem to design FAIR benchmarks **with the help of a community**.

An NTX Community Project

Maintainers:

Sylvain Chevallier Bruno Aristimunha



Igor Carrara



Pierre Guetschel



Sara Sedlar



Bruna Junqueira Lopes



Sébastien Velut



Founders:

Alexandre Barachant Vinay Jayaram



Contributors:



How To Get Started or Contribute

Getting-started tutorial



Check the github and the documentation

- <https://github.com/NeuroTechX/moabb>
- <https://neurotechx.github.io/moabb/>

Discuss during Office Hours or on Gitter

- <https://github.com/NeuroTechX/moabb/issues/191>
- https://gitter.im/moabb_dev/community

Possible contributions:

- Add new datasets,
- Add working examples and use cases,
- Help on character-level decoding of ERP and c-VEP,
- ...

Thank you !