Junifer and Julearn: From Neuroimaging to Machine Learning models without expert-level coding skills

Federico Raimondo[1,2], Synchon Mandal[1,2], Sami Hamdan[1,2], Shammi More[1,2], Leonard Sasse[1,2,3], Vera Komeyer[1,2,4], Amir Omidvarnia[1,2], Simon B. Eickhoff[1,2], Kaustubh R. Patil[1,2]

1 Institute of Systems Neuroscience, Heinrich Heine University Düsseldorf, Düsseldorf, Germany; **2** Institute of Neuroscience and Medicine (INM-7: Brain and Behaviour), Research Centre Jülich, Jülich, Germany; **3** Max Planck School of Cognition, Stephanstrasse 1a, Leipzig, Germany; **4** Department of Biology, Faculty of Mathematics and Natural Sciences, Heinrich-Heine-University Düsseldorf, Düsseldorf, Germany

f.raimondo@fz-juelich.de

Introduction

- Current increasing popularity in Neuroimaging + Machine Learning [1] due to big data and computing power.
- But…
  - High impact on results from decisions
    - in data processing [2]
    - In ML modelling [3]
  - Misconceptions in ML lead to distorted or even invalid findings [4]
  - Early-career researchers require
    - Domain-specific knowledge (e.g. behavioral sciences, biology)
    - ML-specific knowledge (e.g. cross-validation, learning algorithms)
    - Highly developed technical skills (e.g. High-Performance Computing)

Goals

Develop tools:

- Minimal coding
  - If possible, no coding at all
- Easy to learn, use and share
  - But still powerful, including complex methods!
- Suitable for users from diverse backgrounds
  - even with non-STEM formal training
- Minimize user-related errors, E.g.:
  - Wrong template spaces in neuroimaging
  - Data-leakage in ML [4]
- Extensible libraries (e.g. allow to develop novel methods)

Neuroimaging Feature Extraction

- Pipeline Configuration Tool

- No-code tool (YAML file)
- Easy to run for entire datasets
  - HPC or local (GNU-Parallels)
- Built on state-of-the-art tools: AFNI, FSL, ANTS, Nilearn
- Examples:

https://juaml.github.io/junifer

```yaml
workdir: /tmp

datagrabber:
    kind: DataladAOMICID1000
    native_t1w: true

preprocess:
  - kind: fMRIPrepConfoundRemover
    detrend: true
    standardize: true
    strategy:
        motion: full
        wm_csf: full
        global_signal: full
    low_pass: 0.08
    high_pass: 0.01
    masks:
      - inherit
      - compute_epi_mask
      - threshold: 0
  - kind: SpaceWarper
    reference: T1w
    on: BOLD
    using: ants

markers:
  - name: FC-Schaefer100x17
    kind: FunctionalConnectivityParcels
    cor_method: correlation
    parcellation: Schaefer100x17
    masks:
      - inherit

storage:
  kind: HDF5FeatureStorage
  uri: /data/group/riseml/fraimondo/2024_HIP/features/ds003097_FC_native/ds003097_FC_native.hdf5

queue:
  jobname: aomic_fc_native
  kind: HTCondor
  env:
    kind: conda
    name: neurodc
  mem: 8G
  disk: 5G
  verbose: info
```

Dataset to use (can also be specified in terms of file-naming patterns)

One or more data transformation steps

Remove Confounds (e.g. motion parameters)

Warp-back to native space using ANTS

One or more markers

Compute functional connectivity

Storage specification

Run in HPC:

- 1 job per subject using a HTCondor queue
- Use a conda environment
- Ask for 8G of RAM and 5G of disk per job

```yaml
workdir: /tmp

datagrabber:
  kind: DataladAOMICID1000
  types:
    - FreeSurfer

markers:
  - name: brainprint
    kind: BrainPrint

storage:
  kind: HDF5FeatureStorage
  uri: ./storage/brainprint/aomicid1000_brainprint.hdf5
```

ShapeDNA [5] (BrainPrint [6]) in 14 lines!

Machine Learning

- Easily create and evaluate models
  - Tabular data, using pandas
- Estimate model performance using cross-validation
  - Prevent data-leakage
- Easy hyperparameter tuning (using nested CV)
- Built on top of scikit-learn [7]
- Example:

https://juaml.github.io/julearn

```python
from junifer.storage import HDF5FeatureStorage
from julearn.api import run_cross_validation
from julearn.pipeline import PipelineCreator
import pandas as pd
import seaborn as sns
from pathlib import Path

t_path = Path(__file__).parent

storage = HDF5FeatureStorage(t_path / "data/aomicid1000_brainprint.hdf5")

df_eigen = storage.read_df("FreeSurfer_brainprint_eigenvalues")
df_areas = storage.read_df("FreeSurfer_brainprint_areas")
df_volumes = storage.read_df("FreeSurfer_brainprint_volumes")
df_volumes = df_volumes.reset_index().set_index("subject")
df_volumes["gm_vol"] = df_volumes["lh-pial-2d"] + df_volumes["rh-pial-2d"]

df_demographics = pd.read_csv(t_path / "data/participants.tsv", sep="\t")
df_demographics.rename(columns={"participant_id": "subject"}, inplace=True)
df_demographics = df_demographics.set_index("subject")

columns = [
    "4th-Ventricle", "Brain-Stem", "Left-Cerebellum-White-Matter",
    "Left-Cerebellum-Cortex", "Left-Thalamus-Proper", "Left-Caudate",
    "Left-Putamen", "Left-Pallidum", "Left-Hippocampus", "Left-Amygdala",
    "Left-Accumbens-area", "Left-VentralDC", "Right-Cerebellum-White-Matter",
    "Right-Cerebellum-Cortex", "Right-Thalamus-Proper", "Right-Caudate",
    "Right-Putamen", "Right-Pallidum", "Right-Hippocampus", "Right-Amygdala",
    "Right-Accumbens-area", "Right-VentralDC", "lh-white-2d", "rh-white-2d",
    "lh-pial-2d","rh-pial-2d",
]

df_data = df_eigen[columns].unstack(-1).reset_index().set_index("subject")
df_data.columns = df_data.columns.map(
    lambda x: x if isinstance(x, str) else f"{x[0]}_{x[1]}"
)
df_data = df_data.join(df_demographics)
df_data = df_data.join(df_volumes["gm_vol"])


X = [x for x in df_data.columns if any(x.startswith(y) for y in columns)]
X_types = {"continuous": [".*"]}

creator = PipelineCreator(problem_type="classification")
creator.add("zscore")
creator.add(
    "svm",
    C=(0.001, 100, "log-uniform"),
)

search_params = {
    "kind": "optuna",
    "cv": 5,
}
scores, model, inspector = run_cross_validation(
    X=X,
    y="sex",
    data=df_data,
    X_types=X_types,
    search_params=search_params,
    model=creator,
    return_train_score=True,
    return_inspector=True,
    cv=5,
)

predictions = inspector.folds.predict()
to_merge = df_data[["sex", "gm_vol"]].iloc[predictions.index]
predictions.index = to_merge.index
to_plot = pd.concat([predictions, to_merge], axis=1)

to_plot["correct"] = to_plot["repeat0_p0"] == to_plot["target"]
sns.boxplot(data=to_plot, x="sex", hue="correct", y="gm_vol")
```

Read data from Junifer (or other tabular format)

Rename columns and combine DataFrames

Define features and feature types

Easy model-specification (e.g. SVM):

- Tune the optimal C from a log-uniform distribution [0.001, 100]

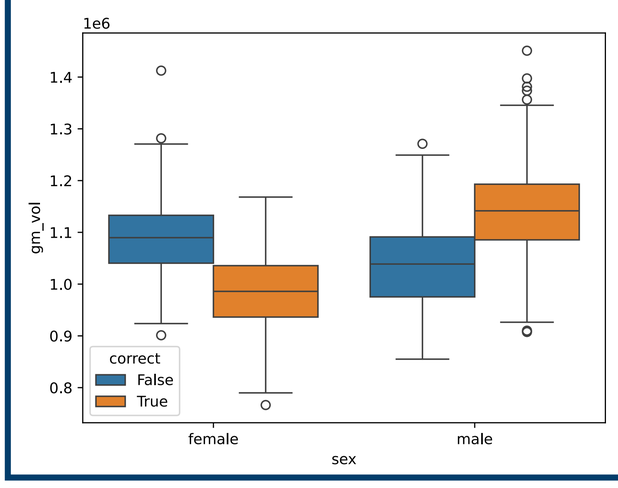
Use Optuna for hyperparameter tuning

Evaluate the model using Cross-Validation

Inspect the results:

- E.g. do we have a bias?

Conclusion



- Building and evaluating ML models from neuroimaging is not an easy task.
  - Even reproducing them is hard! [8]
- Interpreting results to obtain insights can be tricky [9]
  - E.g. sex prediction is biased by the total intracranial volume
- Junifer and Julearn enable domain experts without highly-developed programming and technical skills to analyze brain images and build complex ML pipelines
- Neuroimaging and ML experts can easily extend the libraries with custom methods.
- We can't showcase all the features here (e.g.)
  - Neuroimaging-specific models:
    - Connectome-based Predictive Modelling [10]
  - Interactive ML model comparison plots
    - With corrected t-tests for ML [11]
  - Easily share features and code
- Core developers from the Applied Machine Group, at the Institute of Neuroscience and Medicine – 7: Brain and Behavior, at Jülich Research Center: We can understand ML and brain-imaging research questions.

TRY IT YOURSELF!

References [1] J. Wu, J. Li, S. B. Eickhoff, D. Scheinost, and S. Genon, 'The challenges and prospects of brain-based prediction of behaviour', Nat Hum Behav, vol. 7, no. 8, Art. no. 8, Aug. 2023, doi: 10.1038/s41562-023-01670-1. **[2]** G. Antonopoulos, S. More, F. Raimondo, S. B. Eickhoff, F. Hoffstaedter, and K. R. Patil, 'A systematic comparison of VBM pipelines and their application to age prediction', NeuroImage, vol. 279, p. 120292, Oct. 2023, doi: 10.1016/j.neuroimage.2023.120292. **[3]** S. More et al., 'Brain-age prediction: A systematic comparison of machine learning workflows', Neuroimage, vol. 270, p. 119947, Apr. 2023, doi: 10.1016/j.neuroimage.2023.119947. **[4]** L. Sasse et al., 'On Leakage in Machine Learning Pipelines'. arXiv, Nov. 07, 2023. doi: 10.48550/arXiv.2311.04179. **[5]** M. Reuter et al., 'Laplace-Beltrami spectra as 'Shape-DNA' of surfaces and solids.' Computer-Aided Design. 2006. doi: 10.1016/j.cad.2005.10.011. **[6]** C. Wachinger et al., 'BrainPrint: a discriminative characterization of brain morphology'. Neuroimage. 2015;109:232-48 doi: 10.1016/j.neuroimage.2015.01.032. **[7]** F. Pedregosa et al., 'Scikit-learn: Machine Learning in Python', Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2012, doi: 10.1007/s13398-014-0173-7.2. **[8]** K. J. Gorgolewski and R. A. Poldrack, 'A Practical Guide for Improving Transparency and Reproducibility in Neuroimaging Research', PLOS Biology, vol. 14, no. 7, p. e1002506, Jul. 2016, doi: 10.1371/journal.pbio.1002506. **[9]** S. M. Lundberg and S.-I. Lee, 'A Unified Approach to Interpreting Model Predictions', in Advances in Neural Information Processing Systems, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., Curran Associates, Inc., 2017. **[10]** X. Shen et al., 'Using connectome-based predictive modeling to predict individual behavior from brain connectivity', Nat Protoc, vol. 12, no. 3, pp. 506–518, Mar. 2017, doi: 10.1038/nprot.2016.178. **[11]** C. Nadeau and Y. Bengio, 'Inference for the Generalization Error', Machine Learning, vol. 52, no. 3, pp. 239–281, Sep. 2003, doi: 10.1023/A:1024068626366.

Acknowledgments: These projects were supported by the Helmholtz Imaging Platform (project NimRLS, BrainShape)