BenchNIRS: benchmarking framework for machine learning with fNIRS

University of Nottingham
UK | CHINA | MALAYSIA

FNIRS.UK'23
SEPTEMBER 14-15, COLCHESTER, UNIVERSITY OF ESSEX, CAMPUS

Johann Benerradi, Jeremie Clos, Aleksandra Landowska, Michel F. Valstar, Max L. Wilson

```
pip install benchnirs
```

1. BACKGROUND

Lack of community standards for applying machine learning to fNIRS data
⟶ published works often claim high generalisation capabilities, but sometimes with poor practices or missing details in the paper (*Kapoor & Narayanan 2022*: ***"Leakage and the Reproducibility Crisis in ML-based Science"***).

Lack of open-source benchmarks
⟶ hard to objectively evaluate the performance of models when it comes to choosing them for brain-computer interfaces.

Dataset	Classes
Herff et al. 2014 *n*-back	1-, 2-, 3-back
Shin et al. 2018 *n*-back	0-, 2-, 3-back
Shin et al. 2018 word generation	baseline, word generation
Shin et al. 2016 mental arithmetic	baseline, mental arithmetic
Bak et al. 2019 motor execution	right-hand, left-hand, foot

Table 1.

2. FRAMEWORK

Open-source Python benchmarking framework ⟶ best practice machine learning methodology to evaluate models classifying fNIRS data on 5 open access datasets (*Table 1.*).

Machine learning methodology with **nested cross-validation** ⟶ optimisation of models and evaluation without bias ⟶ expected performance of a brain-computer interface.

Application programing interface (API):
⟶ **load any of the 5 open access fNIRS datasets**
⟶ **perform preprocessing and signal processing on the data** (filtering, baseline correction, motion artefact correction, channel rejection, region of interest averaging)
⟶ **optimise and evaluate machine learning models** (including deep learning) with robust methodology, taking advantage of GPU capabilities if available on the user's machine

3. RESULTS & DISCUSSION

Benchmarking on the 5 datasets of 6 baseline models: linear discriminant analysis (**LDA**), support-vector classifier (**SVC**), k-nearest neighbours (**kNN**), artificial neural network (**ANN**), convolutional neural network (**CNN**), and long short-term memory (**LSTM**).

Investigation of the influence of different factors on the classification performance: number of training examples, duration of each epoch, sliding window vs. simple epochs, within subject vs. unseen subject classification.

Results for unseen subject classification (*Figure 1.*): performance typically lower than the scores often reported in literature ⟶ **predicting unseen data remains a difficult task**. Other results can be seen in the journal paper.

Baseline models ⟶ reference point to **demonstrate advances in a comparable way**.

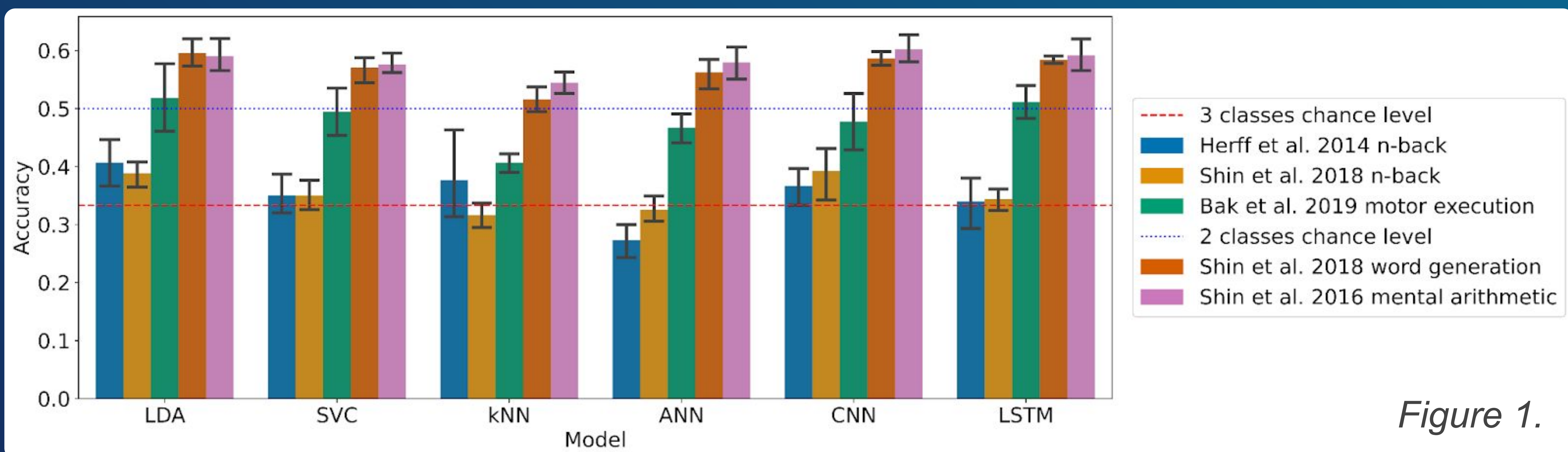

Figure 1.

Simple practical use case: after defining a model structure in PyTorch, one can use *BenchNIRS* to load a dataset, process the data and run nested cross-validation with hyperparameter fine tuning, all in only a few lines of code.

```
Python 3.x.x
import benchnirs as bn
epochs = bn.load_dataset("bak_2019_me", path)
data = bn.process_epochs(epochs)
results = bn.deep_learn(my_model, *data)
print(results)
```

4. OPEN TO CONTRIBUTIONS

Set of **recommendations for methodology decisions and writing papers** when doing machine learning with fNIRS data: checklist found in *BenchNIRS*.

Repository open to community contributions:
⟶ improving recommendation checklist
⟶ adding support for new open access datasets
⟶ improving implementation of machine learning methodology and production of results and figures

Acknowledgements
Supported by the Engineering and Physical Sciences Research Council (EP/T022493/1, EP/V00784X/1)

Brain and Physiological Data Group