Machine learning for BCI

Mikhailova Basil (1820239058)
Ivanova Anastasiia (1820239050)

Contents

01

Motivation

02

Objectives of the project

03

Resources, tools, subject overview

04

Data collection trial description

05

Performance and code

06

Conclusion

01 Motivation

Imagine if your teacher could know whether you actually paid attention in your last Zoom class. Or, imagine if you could prepare your next presentation using only your thoughts.

These scenarios might soon become a reality thanks to the development of Brain-Computer Interfaces (BCIs).

To put it in the simplest terms, think of a BCI as a bridge between your brain and an external device.

02 Objectives of the project

Why we chose ML for BCI

ML for BCI is a very modern and hot field of research and study.
It has potential of improving lives, because it has real-life practical application, for example in helping paralyzed or injured people.

The goal

To create an LDA model and a CNN capable of recognizing a person’s thoughts about moving.
Compare two models performance and implementation.

03 Resources and Tools

Google Collab

Training a neural network requires high computing power of the computer. In order not to store a large amount of data on your device and to work effectively with the network, the platform Google Collab was chosen as the development environment

MNE-Python

MNE-Python is an open-source Python module for processing, analysis, and visualization of EEG and MEG

PyTorch

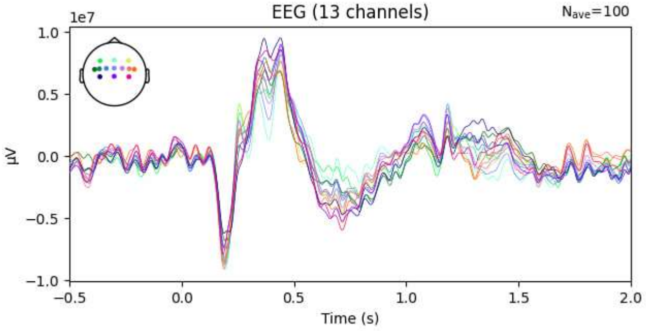
PyTorch is an optimized Deep Learning tensor library based on Python and Torch and is mainly used for applications using GPUs and CPUs. PyTorch was chosen since it uses dynamic computation graphs and is completely Pythonic

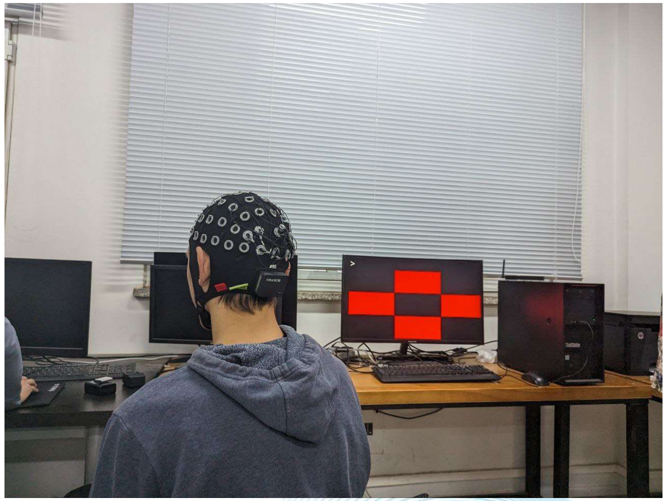
Scikit-Learn

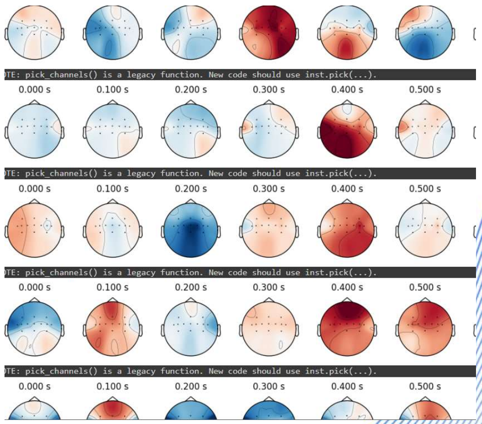
Scikit-Learn, also known as sklearn is a python library to implement machine learning models and statistical modelling.

What is EEG?
An electroencephalogram

An electroencephalogram (EEG) is a test that measures electrical activity in the brain using small, metal discs (electrodes) attached to the scalp. Brain cells communicate via electrical impulses and are active all the time, even during asleep. This activity shows up as wavy lines on an EEG recording.



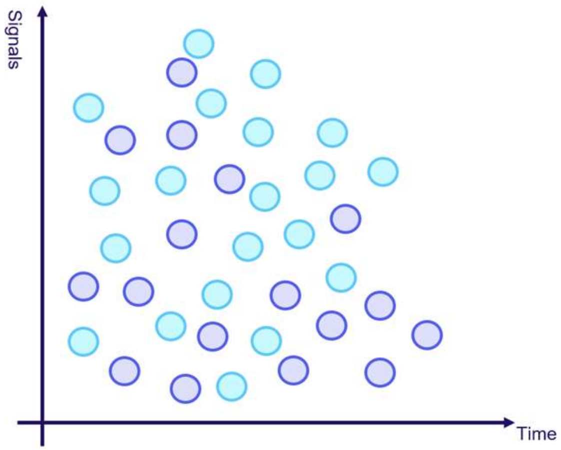



```
TE: pick_channels() is a legacy function. New code should use inst.pick(...).
TE: pick_channels() is a legacy function. New code should use inst.pick(...).
TE: pick_channels() is a legacy function. New code should use inst.pick(...).
TE: pick_channels() is a legacy function. New code should use inst.pick(...).
```

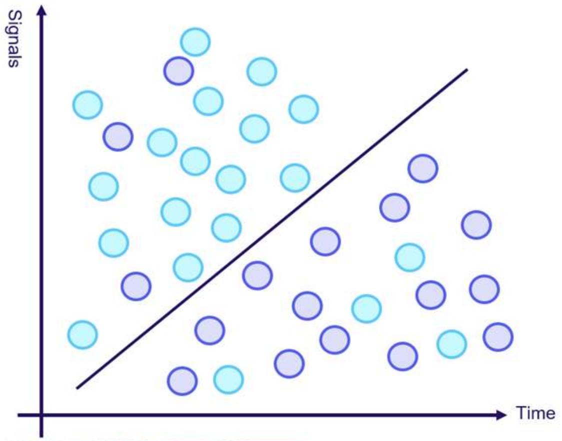
What is an LDA classifier?
linear discriminant analysis classifier

LDA is a linear binary classifier that projects an $\boldsymbol{m}$ dimensional input vector $\boldsymbol{x}$ onto a hyperplane that divides the input space into two halfspaces. LDA is a popular technique for classifying BCI data.

Before LDA

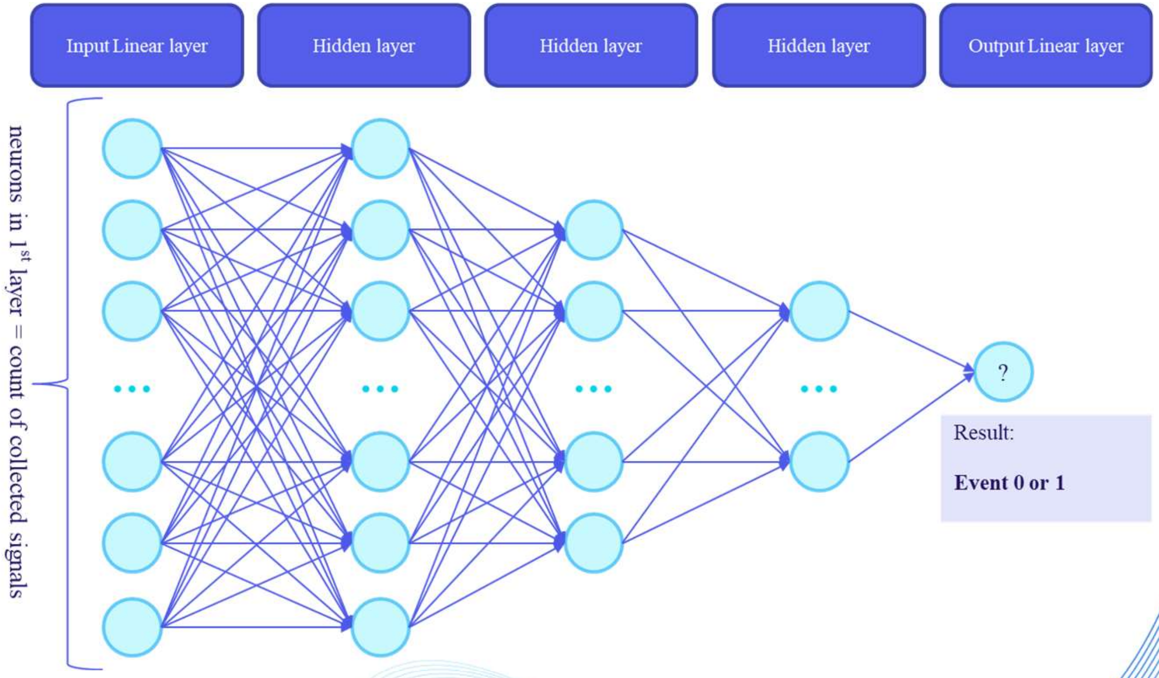


After LDA



What is CNN?
A Convolutional Neural Network

CNN is a type of deep learning algorithm that is particularly well-suited for image recognition and processing tasks. It is made up of multiple layers, including convolutional layers, pooling layers, and fully connected layers. The convolutional layers are the key component of a CNN, where filters are applied to the input to extract features.



Result:

Event 0 or 1

04 Data collection trial description

Trial structure

Trial structure within the synchronous training paradigm. The task for the user was to perform sustained right hand versus both feet movement imagery starting from the cue (second 3) to the end of the cross period (second 8). A trial started with 3 s of reference period, followed by a brisk audible cue and a visual cue (arrow right for right hand, arrow down for both feet) from second 3 to 4.25. The activity period, where the users received feedback, lasted from second 4 to 8. There was a random 2 to 3 s pause between the trials.

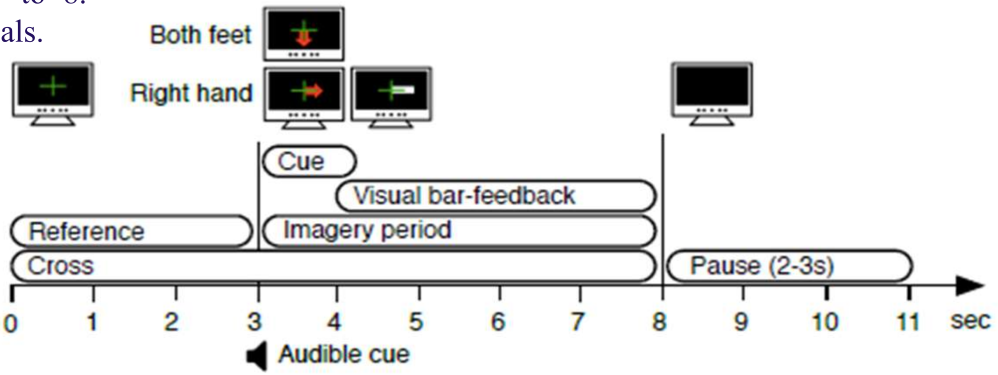
Data format

Three .mat files are given:

EEGdata.mat: holds the EEG signal in µV in a matrix (datapoints × channels) of double values.

label.mat: holds the true labels for every trial according to the visual cues that were displayed in the paradigm during the recording

trial.mat: indicates the position in data points where every trial starts.



05

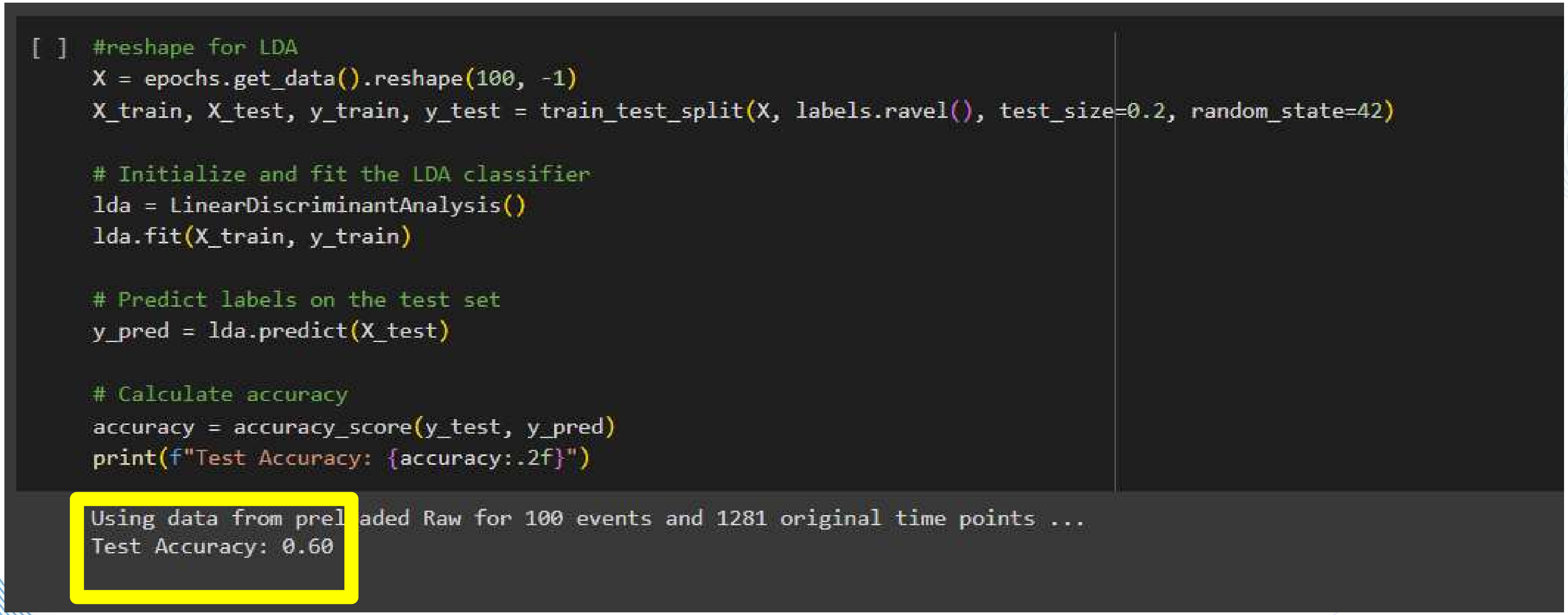
Performance and code

see at: https://colab.research.google.com/drive/1471DANT4NY7AguuMJ70L9IIrNIBz6n0p?usp=sharing

06

Conclusion

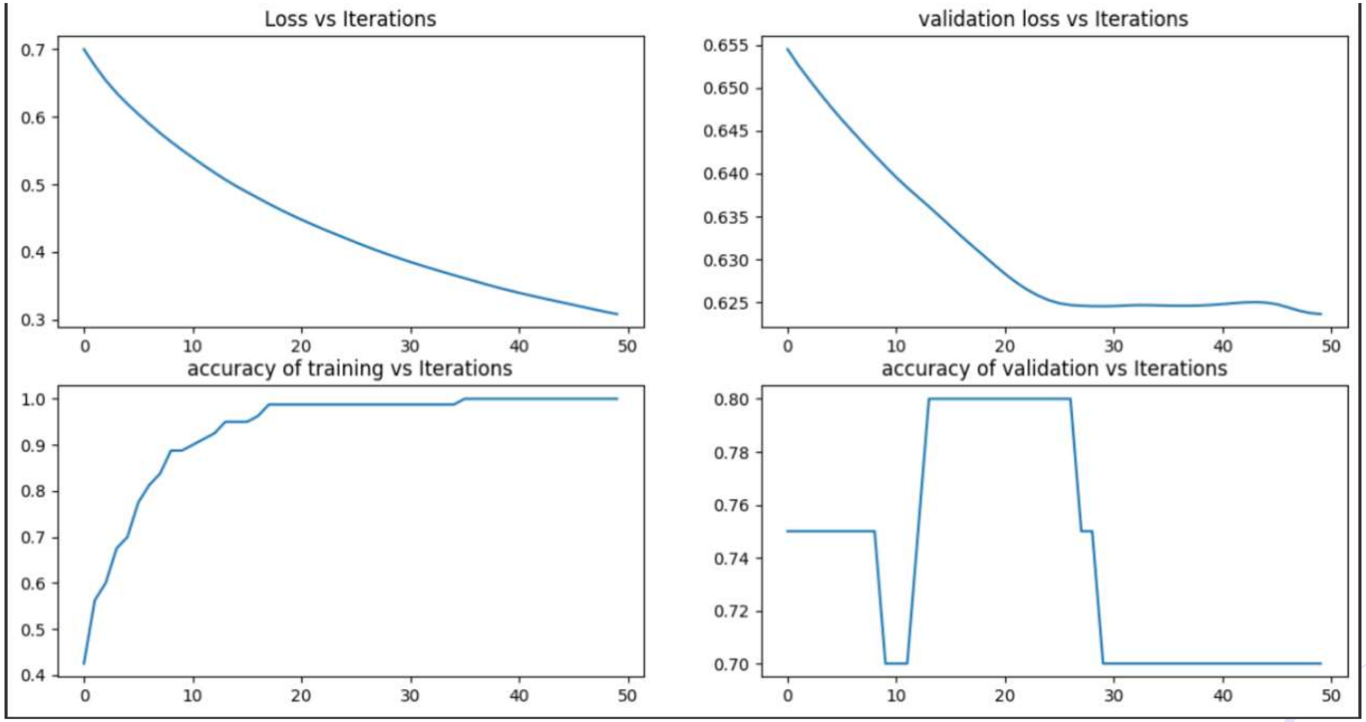
LDA results



```
[ ]  #reshape for LDA
     X = epochs.get_data().reshape(100, -1)
     X_train, X_test, y_train, y_test = train_test_split(X, labels.ravel(), test_size=0.2, random_state=42)

     # Initialize and fit the LDA classifier
     lda = LinearDiscriminantAnalysis()
     lda.fit(X_train, y_train)

     # Predict labels on the test set
     y_pred = lda.predict(X_test)

     # Calculate accuracy
     accuracy = accuracy_score(y_test, y_pred)
     print(f"Test Accuracy: {accuracy:.2f}")
```

```
Using data from prel aded Raw for 100 events and 1281 original time points ...
Test Accuracy: 0.60
```

CNN results


Loss vs Iterations
0.7
0.6
0.5
0.4
0.3
0
10
20
30
40
50
validation loss vs Iterations
0.655
0.650
0.645
0.640
0.635
0.630
0.625
0
10
20
30
40
50
accuracy of training vs Iterations
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0
10
20
30
40
50
accuracy of validation vs Iterations
0.80
0.78
0.76
0.74
0.72
0.70
0
10
20
30
40
50

Let’s compare the models

	CNN	LDA
Pros:	The CNN model has the potential to give results with higher accuracy score.	Easier to set up. Gives relatively good results
Cons:	The CNN model is harder to build and fine tune. Takes more time for the model to learn. Can overfit the dataset rather easily.	Result of accuracy of 60% is not much better that 50%.

Also to note: it would be preferable for building a real application to have dataset of more than 100 samples. Perhaps 1000 should do.

Thank you for your attention!

Q&A

CAT

CAT

DOG

CAT