BCI

Intelligent Systems

Oct 23, 2022

MAIN INFO:

CHAPTER

ONE

BCI LIB

1.1 Basic information

Implementation code for BCI models. The source code presented on the github.

All methods were implemented based on `pytorch` for simple parallelization by using `cuda`.

All information about this project can be found in the documentation.

1.2 Requirements and Installation

A simple instruction of installation using pip is provided near the source code.

More information about installation can be found in documentation installation page.

1.3 Example of use

A simple examples of module usage can be found in documentation example page.

CHAPTER

TWO

INSTALLATION

2.1 Requirements

- Python >= 3.6
- pip >= 22.0

2.2 Installing by using PyPi

2.2.1 Install

```
python3 -m pip install bci-ml
```

2.2.2 Uninstall

```
python3 -m pip uninstall bci-ml
```

2.3 Installing from GitHub source

2.3.1 Install

```
git clone https://github.com/intsystems/bci-ml.git
cd bci-ml
python3 -m pip install ./src/
```

2.3.2 Uninstall

```
python3 -m pip uninstall bci-ml
```

CHAPTER

THREE

EXAMPLE

3.1 Requirements

It is recommended make virtualenv and install all next packages in this virtualenv.

```
bci-ml==0.0.1
```

Include packages.

```
TODO
```

3.2 Preparing the dataset

TODO

CHAPTER

FOUR

BASE MODELS

The `bci.base_models` contains classes:

- `bci.base_models.BaseModel`
- `bci.base_models.IdentityModel`

class bci_ml.base_models.**BaseModel**

Base class for the BCI models.

fit(*input*)

Fit model for the given input data.

Parameters
input (*FloatTensor.*) – The tensor of the analyzed data.

forward(*input*)

Returns model prediction for the given input data.

Parameters
input (*FloatTensor.*) – The tensor of the analyzed data.

Returns
Model answers for the given input data

Return type
FloatTensor

class bci_ml.base_models.**IdentityModel**

A model which defines identity mapping. Mathematically define model $\mathbf{f}(\mathbf{x}) = \mathbf{x}$.

> **Warning:** It's just an example of BCI model, and cannot be used in real cases.

Example:

```
>>> _ = torch.random.manual_seed(42) # Set random seed for repeatability
>>>
>>> model = IdentityModel()
>>> X = torch.randn(2, 1) # Generate random tensor
>>> predict = model(X)
tensor([[0.3367],
    [0.1288]])
```

`forward`(*input*)

Returns model prediction for the given input data.

Parameters
`input` (*FloatTensor.*) – The tensor of the analyzed data.

Returns
Return similar tensor to input data.

Return type
FloatTensor

CHAPTER

FIVE

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

b

INDEX