
NN for CHF

Dumont Jules

Oct 02, 2023

CONTENTS:

1	CHF_model_api	1
1.1	CHF_model_api package	1
2	Indices and tables	5
	Python Module Index	7
	Index	9

CHF_MODEL_API

1.1 CHF_model_api package

1.1.1 Submodules

1.1.2 CHF_model_api.config module

1.1.3 CHF_model_api.model module

```
class CHF_model_api.model.MyModel(hparams: dict | None = None, model_name: str | None = None,  
                                   auto_save: bool = False, process_number: int | None = None)
```

Bases: object

class that either create a CHF prediction model from scratch using hparams dictionnary or load a saved one from model_name from a .h5 file and his hp from json

createNewModel() → None

create new model based on hparams

displayPerf()

init_callbacks() → list

action taken at evry epoch for monitoring

loadData() → None

check if corresponding valid/train set already computed, if no compute and add the new data in DATA

loadMyModel(model_name: str) → None

Create a model based on hparams given in parameterss or use a saved one based on his name and load it from saved_models directory set also all the attributes

lrScheduler(epoch: int, lr: float) → float

program a decrease of the learning rate

classmethod makeDataCopies(jobs, seed, input_number)

makeRealPredictions(features_data: list) → list

make CHF predicitons

plotResult() → None

save(overwrite=False) → None

save model in .h5 format, to make predictions save also the hparams of the model in json files

saveResults() → None

compute and save metrics in hparams

train(logs=True, callbacks=True, train_epochs=None) → dict

train the model over train_epochs epochs and save the results in the attribute hparams

vizualize() → None

save a png file containing vizual rpz of the network

CHF_model_api.model.batch_nrmse(y_true, y_pred) → float

1.1.4 CHF_model_api.myTensorboard module

class CHF_model_api.myTensorboard.MyTensorboard(saved_models_name: List[str])

Bases: object

add_hparams()

add_model(model) → None

add_saved_model(model_name: str) → None

copy_log(name) → None

copy logs in ./logs/modelname/train et /validation to the same directories in ./hparams_tuning/sessionname

copy_logs() → None

get the logs of all the models to compare

init_models() → List[MyModel]

get the saved models by creating object my_models based on the name of the saved models

load_tb() → None

log_Hparams() → None

log_Hparams_range() → None

make_Hparams_range() → None

set_up_directories() → None

create validation and training directory and also a directory for each different model (with diff hp)

1.1.5 CHF_model_api.optimizer module

class CHF_model_api.optimizer.MyOptimizer(generic_hparams: dict, opti_architecture: bool = True, opti_dropout: bool = False, opti_learning_rate: bool = False, opti_lr_decrease: bool = False, opti_optimizer: bool = False, opti_activation_method: bool = False, type: int = 1, jobs: int = 2, trials: int = 5, db_name: str | None = None, metric: str = 'mpe', verbose: int = 0)

Bases: object

This class create an optimizer for hyperparameter of deep neural network to predict CHF using mainly optuna library

access_data_index_safe() → list
make_archi_type(*trial*, *type*, *opti_hparams*) → list
make_around_archi(*trial*, *opti_hparams*)
make_decreased_archi(*trial*) → list
make_increased_archi(*trial*) → list
make_random_archi(*trial*) → list
objective(*trial*)
opti_hparams_safe(*trial*) → dict
 Return safely the guess optimized of , the hparameters and the epochs number
optimize_my_models()
train_and_report(*model*, *trial*) → None

1.1.6 CHF_model_api.tools module

CHF_model_api.tools.RemoveDirectoryContent(*directory_path*) → None
 just a function to clean when we want to reset
CHF_model_api.tools.eraseHparams(*model_name: str*) → None
 Erase stored hyperparameters
CHF_model_api.tools.getHparamsSavedModel(*model_name: str*) → dict
 return a dict with all the hyperparameters saved in the directory hparams in saved_models
CHF_model_api.tools.myMsle(*y_true*, *y_pred*) → float
 Compute mean squared logarithmic error
CHF_model_api.tools.nrmse(*y_true*, *y_pred*) → float
 Compute normalised root mean squared error
CHF_model_api.tools.plotResults(*predictions: list*, *y_val: list*, *save_fig: bool = False*) → None
 Plot the the graph of the predicted value in functino of the measured values
CHF_model_api.tools.remove_backups(*name*) → None
CHF_model_api.tools.reset_directories() → None
CHF_model_api.tools.saveHparams(*model_name: str*, *hparams: dict*) → None
 save the dict containing results and hyperparameters in hparams idrectory
CHF_model_api.tools.stdMP(*y_val*, *predictions*) → float
 compute the standart deviation of the ration Measured / predict from l
CHF_model_api.tools.utilsNnConfig(*model*)
CHF_model_api.tools.visualizeNn(*model*, *name*, *description=False*, *figsize=(10, 8)*)
 Plot the structure of a keras neural network.

1.1.7 Module contents

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`

PYTHON MODULE INDEX

C

`CHF_model_api`, 4
`CHF_model_api.config`, 1
`CHF_model_api.model`, 1
`CHF_model_api.myTensorboard`, 2
`CHF_model_api.optimizer`, 2
`CHF_model_api.tools`, 3

INDEX

A

`access_data_index_safe()`
(*CHF_model_api.optimizer.MyOptimizer*
method), 2

`add_hparams()` (*CHF_model_api.myTensorboard.MyTensorboard*
method), 2

`add_model()` (*CHF_model_api.myTensorboard.MyTensorboard*
method), 2

`add_saved_model()` (*CHF_model_api.myTensorboard.MyTensorboard*
method), 2

B

`batch_nrmse()` (in module *CHF_model_api.model*), 2

C

CHF_model_api
module, 4

CHF_model_api.config
module, 1

CHF_model_api.model
module, 1

CHF_model_api.myTensorboard
module, 2

CHF_model_api.optimizer
module, 2

CHF_model_api.tools
module, 3

`copy_log()` (*CHF_model_api.myTensorboard.MyTensorboard*
method), 2

`copy_logs()` (*CHF_model_api.myTensorboard.MyTensorboard*
method), 2

`createNewModel()` (*CHF_model_api.model.MyModel*
method), 1

D

`displayPerf()` (*CHF_model_api.model.MyModel*
method), 1

E

`eraseHparams()` (in module *CHF_model_api.tools*), 3

G

`getHparamsSavedModel()` (in module
CHF_model_api.tools), 3

I

`init_callbacks()` (*CHF_model_api.model.MyModel*
method), 1

`init_models()` (*CHF_model_api.myTensorboard.MyTensorboard*
method), 2

L

`load_tb()` (*CHF_model_api.myTensorboard.MyTensorboard*
method), 2

`loadData()` (*CHF_model_api.model.MyModel* method),
1

`loadMyModel()` (*CHF_model_api.model.MyModel*
method), 1

`log_Hparams()` (*CHF_model_api.myTensorboard.MyTensorboard*
method), 2

`log_Hparams_range()`
(*CHF_model_api.myTensorboard.MyTensorboard*
method), 2

`lrScheduler()` (*CHF_model_api.model.MyModel*
method), 1

M

`make_archi_type()` (*CHF_model_api.optimizer.MyOptimizer*
method), 3

`make_around_archi()`
(*CHF_model_api.optimizer.MyOptimizer*
method), 3

`make_decreased_archi()`
(*CHF_model_api.optimizer.MyOptimizer*
method), 3

`make_Hparams_range()`
(*CHF_model_api.myTensorboard.MyTensorboard*
method), 2

`make_increased_archi()`
(*CHF_model_api.optimizer.MyOptimizer*
method), 3

make_random_archi()
 (*CHF_model_api.optimizer.MyOptimizer*
 method), 3

makeDataCopies() (*CHF_model_api.model.MyModel*
 class method), 1

makeRealPredictions()
 (*CHF_model_api.model.MyModel method*), 1

module
 CHF_model_api, 4
 CHF_model_api.config, 1
 CHF_model_api.model, 1
 CHF_model_api.myTensorboard, 2
 CHF_model_api.optimizer, 2
 CHF_model_api.tools, 3

MyModel (*class in CHF_model_api.model*), 1

myMsle() (*in module CHF_model_api.tools*), 3

MyOptimizer (*class in CHF_model_api.optimizer*), 2

MyTensorboard (*class in*
 CHF_model_api.myTensorboard), 2

N

nrmse() (*in module CHF_model_api.tools*), 3

O

objective() (*CHF_model_api.optimizer.MyOptimizer*
 method), 3

opti_hparams_safe()
 (*CHF_model_api.optimizer.MyOptimizer*
 method), 3

optimize_my_models()
 (*CHF_model_api.optimizer.MyOptimizer*
 method), 3

P

plotResult() (*CHF_model_api.model.MyModel*
 method), 1

plotResults() (*in module CHF_model_api.tools*), 3

R

remove_backups() (*in module CHF_model_api.tools*),
 3

RemoveDirectoryContent() (*in module*
 CHF_model_api.tools), 3

reset_directories() (*in module*
 CHF_model_api.tools), 3

S

save() (*CHF_model_api.model.MyModel method*), 1

saveHparams() (*in module CHF_model_api.tools*), 3

saveResults() (*CHF_model_api.model.MyModel*
 method), 1

set_up_directories()
 (*CHF_model_api.myTensorboard.MyTensorboard*
 method), 2

stdMP() (*in module CHF_model_api.tools*), 3

T

train() (*CHF_model_api.model.MyModel method*), 2

train_and_report() (*CHF_model_api.optimizer.MyOptimizer*
 method), 3

U

utilsNnConfig() (*in module CHF_model_api.tools*), 3

V

visualizeNn() (*in module CHF_model_api.tools*), 3

vizualize() (*CHF_model_api.model.MyModel*
 method), 2