NN for CHF

Dumont Jules

Oct 02, 2023

CONTENTS:

CHAPTER

ONE

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 logaritmic error

`CHF_model_api.tools.`**`nrmse`**`(`*y_true*, *y_pred*`)` → float

Compute normalised root mean suqared 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 1

`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

CHAPTER TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

INDEX