NN for CHF

Dumont Jules

Sep 30, 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.myModel module

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

Bases: `object`

`DATA = {}`

To complete

`displayPerf`()

classmethod **`makeDataCopies`**(*jobs*, *seed*, *input_number*)

`makeRealPredictions`(*features_data: list*) → list

`plotResult`() → None

`save`(*overwrite=False*) → None

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

`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.myModel.`**`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

optimize_my_models()

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.**extractFromPdf**(*path*) → DataFrame

create a csv file based on Groeneveld 2006 LUT pdf take 2 min to run

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.**loadData**(*data_seed: int = 1*, *input_number: int = 5*) → dict

take the data from a csv containing data SI units or create it from the Groeneveld 2006 LUT pdf return a dict containing the keys: 'validation_targets', 'validation_features','training_features' 'training_targets', 'mean' 'std'(mean and std of the training_features before normalization we need to keep when predicting) and is meant to be add to DATA[seed] = {'validation':…}

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*, *y_val*, *save_fig=False*)

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