Doc

1 CONTENTS

2 Setup 2

3 Evaluation mode 2

4 Showing dataset (valve sequence) 3

4.1 show_data_chart.py 3

4.2 show_data_cmap.py 3

5 Deep Learning model 3

5.1 train_deep.py 3

5.2 load_deep.py 3

5.3 train_deep_partial_samples.py 4

5.4 train_deep_incremental.py 4

6 Machine Learning model 4

6.1 train_dtree.py 4

6.2 train_dtree_multioutput.py 4

7 Cross-validation and charts 4

7.1 cross_check_<mode>.py 4

7.2 eval_results_overview.py 5

7.3 eval_results_cross_check.py 5

8 Experiment description 5

8.1 Exp 31 5

8.2 Exp 34 5

8.3 Exp 37 5

8.4 Exp 38 5

8.5 Exp 39 6

9 Resources 6

2 SETUP

Download the model files using *download.bat*. This will create the archive on your machine to data/models/archive.zip.

Install 7z. Make sure to install to “C:\Program Files\7-Zip\7z.exe”, otherwise update the path in *unpack.bat* and *archive.bat* to match your install location.

Run *unpack.bat* to extract the saved models from the archive.

Run *archive.bat* to add the models to the archive.

Update config.yml to use one of the two main sets of experiments:

root_model_container: "data/models/unpacked/one_hot_encoding"

root_model_container: "data/models/unpacked/no_encoding"

3 EVALUATION MODE

Each deep learning model is selected as the best model from 5 attempts. The best and average models are selected for evaluation. Models can be trained or loaded from a file (generated from a previous training) and the accuracy, training time and size on disk can be shown as graphical representations.

The dataset can be selected as csv files from *data* folder. Each file represents an experiment. Models are trained for each experiment and then compared. Cross-validation is using every combination of model and dataset for validation, while the models are trained on the matching datasets only, showing an evaluation of performance in terms of overfitting.

Raw evaluation data is saved as csv files in data/output. Charts are generated from these files, showing the results as plots, bar charts and colormaps, according to the evaluation type (raw data, comparison, cross-validation). Charts are saved in *figs* folder.

Trained models are saved in data/models. Current training experiment is saved in data/models/crt. Models can be loaded from data/models/crt by default, or another folder can be specified in code.

4 SHOWING DATASET (VALVE SEQUENCE)

4.1 show_data_chart.py

Show data as timeseries

Use raw csv files with measurements from valve sequence experiment

4.2 show_data_cmap.py

Show data as colormap representation (binary valve states)

Use raw csv files with measurements from valve sequence experiment

5 DEEP LEARNING MODEL

5.1 train_deep.py

The model is trained and evaluated (accuracy train/test, training time, size on disk) and the results are saved into a csv file.

The script will fetch experiment data from csv files

The outputs (valve state) is binarized

Model training is done and the result is saved to a file in data/models/crt

*The model can be Dense or RNN, specified by use_rnn = False/True

5.2 load_deep.py

The model is loaded from a file and evaluated (accuracy train/test, training time, size on disk) and the results are saved into a csv file.

The model is loaded from file that is specified in code: filenames, model_filenames.

The data can be loaded from a file specified in code: root_data_folder, root_crt_model_folder.

*The model can be Dense or RNN, specified by use_rnn = False/True

5.3 train_deep_partial_samples.py

The target file is bookmarked with keypoints e.g. opening each valve and the model is trained incrementally. Each step is saved.

A combined model can be extracted using incremental_training.py

5.4 train_deep_incremental.py

Train a single model incrementally for each experiment dataset

6 MACHINE LEARNING MODEL

6.1 train_dtree.py

Run decision tree classifier and save models with .skl extension in data/models/crt

Warning: deprecated

6.2 train_dtree_multioutput.py

Run decision tree/random forest classifier and save models with .skl extension in data/models/crt

The model is set from modules/classifiers.py

Select between decision tree and random forest with: use_randomforest

7 CROSS-VALIDATION AND CHARTS

7.1 cross_check_<mode>.py

Evaluates each model in data/models/crt on each dataset and creates a cross-validation report. Find out if a model trained on a dataset is good for predicting data from other datasets as well.

*For cross_check_deep.py, the model can be Dense or RNN, specified by use_rnn = False/True

Output to /data/output/cross_check_<mode>.csv

7.2 eval_results_overview.py

Get top models for each experiment from data/models/crt and show model accuracy results as bar chart. Show model performance, training time and size on disk. Compare models on matching datasets.

7.3 eval_results_cross_check.py

Showing data from /data/output/cross_check_<model>.csv as colormap grid.

Should be run after cross_check_<model>.py

*The model is specified in code: deep, deep_rnn, dtree_1, dtree_2_multioutput

*The dataset is specified in code: train, test

8 EXPERIMENT DESCRIPTION

8.1 Exp 31

pump 80%

manual adjust close valves

8.2 Exp 34

pump 50%

no manual adjust

8.3 Exp 37

pump 80%,

return pattern

no manual adjust

valve comp mode

8.4 Exp 38

pump 60%

return pattern

no manual adjust

valve comp mode

8.5 Exp 39

pump 80%

gray code pattern

no manual adjust

valve comp mode

9 RESOURCES

https://www.geeksforgeeks.org/decision-tree-implementation-python/