1. Install SUMO
2. Make sure the SUMO\_HOME system variable is set and points to the SUMO installation path
3. Download TAPAS Cologne and git clone LuST datasets
4. Execute „netconvert -s lust.net.xml --plain-output-prefix“ and „netconvert -s cologne2.net.xml --plain-output-prefix“ in the scenario folders
5. Open clustering\_code/dataset-import.py and correct the xml file paths at the bottom of the code
6. Run python dataset-import.py, two csv files should be created dataset-cgn-tl.csv and dataset-lust-tl.csv
7. Fire up Jupyter Notebook and open clustering\_code/clustering.ipynb
8. Run all cells, two additional csv files should be created dataset-cgn-tl-clusters.csv and dataset-lust-tl.clusters
9. Run either python pycharm-project\pybrain\_learner\dqn\_main.py –help to see available parameters or look them up in the code. The file will create a json file containing all rewards of the simulation and dump the neural net’s weights in an h5 file.
10. For repeated runs I created simple shell scripts in the shell\_scripts folder
11. Run script.sh 2>&1 | tee log.txt to get the simulation output in a text file. The execution is endless and runs simulations with 500, 1000, 1500, 2000… iterations
12. Parse the output using the pycharm-project\pybrain\_learner\traci\_result\_parser.py file. It will convert the log output to a pandas dataframe compatible csv file.
13. Run the rl-analysis.ipynb notebook to create charts and reward function evaluation

|  |  |  |
| --- | --- | --- |
| Filename | Path | Explanation |
| Dqn\_main | pycharm-project\pybrain\_learner | Main code, run this to start a learning process |
| Run\_sumo\_nolearner | pycharm-project\pybrain\_learner | Executes a plain sumo simulation run that uses the scenarios own traffic light control |
| Run\_sumo\_random | pycharm-project\pybrain\_learner | Executes the scenario with random traffic lights |
| Test\_gpu | pycharm-project\pybrain\_learner | To see if theano successfully uses the gpu |
| Traci\_result\_parser | pycharm-project\pybrain\_learner | Use this to parse simulation log output |
| Dqn\_brain,agent,memory, fulldqn\_brain, reward\_functions | pycharm-project\pybrain\_learner | Part of the dqn implementation, see their comments |
| Dataset-import | Clustering\_code | Used for converting sumos xml files to a dataset |
| Clustering.ipynb | Clustering\_code | Jupyter Notebook containing the clustering analysis |
| Renders | Clustering\_code | Some Code i took from another udacity project ☺ |
| Colormap | Clustering\_code | A colormap |
| \* | Shell\_scripts | Simple linux shell scripts to run simulations with 500 simulation step increments (see step 11.) |
| rl-analysis.ipynb | reinforcement-result-analysis/ | Jupyter notebook for evaluating the simulation results |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |