# Domain Background

The project domain will be the analysis and optimization of traffic light control using machine learning. Efficient ways of organizing traffic light control in an urban environment are important for numerous reasons like lowering fuel consumption, grid lock prevention and overall faster transportation times. Additional challenges include prioritizing traffic dynamically for emergency or police cars or rerouting traffic in case of accidents. Nowadays traffic lights are controlled by static algorithms using time slots. A machine learning approach could potentially optimize traffic. Because traffic lights today are autonomous, unmanaged systems a simulation environment to test a machine learning strategy is necessary.

# Problem Statement

The goal is to develop a reinforcement learning strategy to control traffic lights directly in a simulated environment using SUMO[[1]](#footnote-2). The following tasks need to be accomplished:

1. Preprocess and analyze the simulation scenarios for the cities of Luxembourg[[2]](#footnote-3) and Cologne[[3]](#footnote-4)
2. Use one or more clustering algorithms to find meaningful clusters for the traffic light controlled intersections
3. Choose and implement a reinforcement learner for each of the clusters which controlls the traffic lights directly using SUMO’s Python interface TraCI[[4]](#footnote-5)
4. Compare the first implementation to random behavior and the scenarios default traffic light algorithms in terms of the aggregates simulation statistics SUMO provides
5. Improve the reward function and re-run the simulations

The reinforcement learner will hopefully improve the overall traffic statistics in terms of total waiting times and fuel consumption.

**Evaluation Metrics**

The following metrics will be used to evaluate the solution model:

* Duration: average trip duration
* WaitingTime: average time spent standing (involuntarily)
* TimeLoss: average time lost due to driving slower than desired
* DepartDelay: average time vehicle departures were delayed due to lack of road space
* Emergency Stops: The number of emergency stops card had to make

They are provided by the simulation and are directly related to the overall efficiency of the traffic system.

# Clustering Analysis

## Introduction

The full clustering analysis can be seen in the Jupyter Notebook clustering\_code/clustering.ipynb file. I decided to run the clustering analyses on three datasets:

1. The TAPAS Cologne (cgn) dataset
2. The LuST (lust) Datasets
3. A combination of the two

The idea behind this approach was to see if the different scenarios would have similar clustering results and to get a deeper understanding of SUMO scenarios. Additionally it was interesting to see if a combined dataset would benefit from including more data.

## Importing the Data

Before clustering could be implemented, the dataset needed to be converted into a pandas dataframe. The following values are collected:

* junction\_id: the id of the intersection
* junction\_type: is always traffic\_light[[5]](#footnote-6)
* x,y,z: the physical coordinates of the junction in the simulations
* isRoundabout: Whether the intersection is part of a roundabout
* trafficlight\_count: an array containing the number of traffic lights the intersection controlls, the trafficlight’s ID and a list of all the connections to the intersection
* avg\_lane\_speed,avg\_lane\_length& standard deviations: Because every intersection has a different amount of lanes, the mean was used to aggregate their speed limits and lengths. To mitigate the information loss the standard deviation was also computed.
* edge\_types: the different types of edges[[6]](#footnote-7) connected to this intersection
* edge\_priorities: The average of the edge’s priorities. Because these values are very close together, the standard deviation was not included
* number of lanes: the total amount of lanes in the junction

The script that generates csv files for scenario files can be found in the “clustering\_code/dataset-import.py” file[[7]](#footnote-8).

## Preprocessing the Datasets

A few columns had to be dropped or reformated. For example, initially I included the x,y and z coordinates in the clustering process. The idea was that maybe intersections that are nearby each other in a geographical sense, could be similar. Due to later analyses I decided to exclude them.

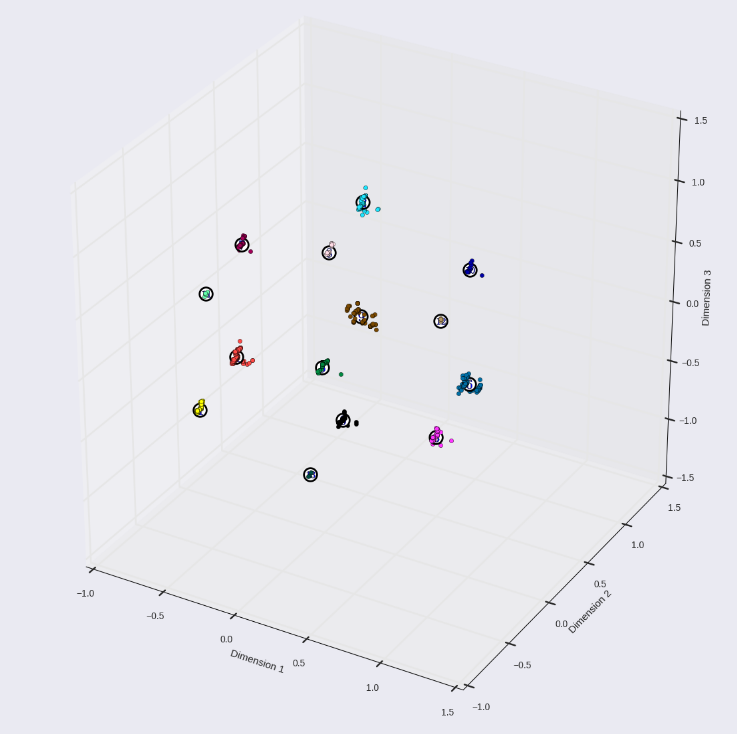
The isRoundabout was converted from {true|false} to {0|1}. One more challenging tasks was to convert the different edge types. I used sklearn’s MultiLabelBinarizer(MLB) to accomplish this task. Also it was interesting to run the clustering analyses with and without the MLB to see how much of an impact the types would have[[8]](#footnote-9). Including the MLB-generated values resulted in different results for the datasets. For the cgn dataset including the types resulted in more or the same amount of clusters depending on the algorithm. For the lust dataset it resulted in more or the same clusters with one exception where it reduced the amount of clusters. Finally, I decided to include the MLB because the amount of traffic lights controlled where better distributed among the clusters. Without the MLB sometimes one cluster would include above 90% of all intersections with n-traffic lights and the other m-clusters would include only one intersection.

## Feature Scaling

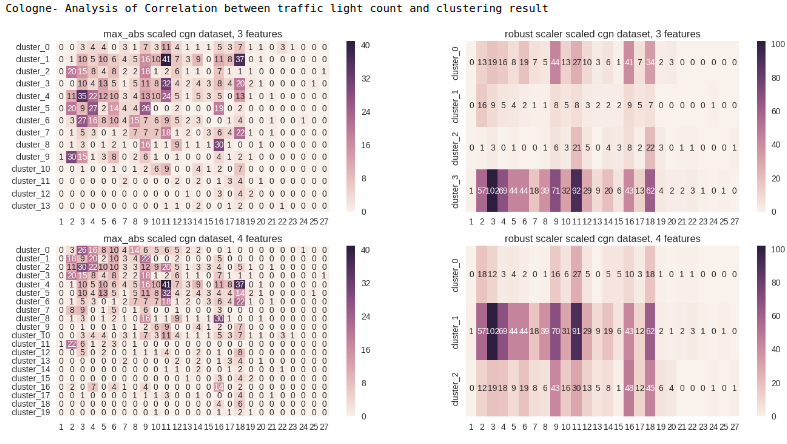
## Dimensionality Reduction using PCA

## Clustering using the Silhouette Score

Silhouette score[[9]](#footnote-10)

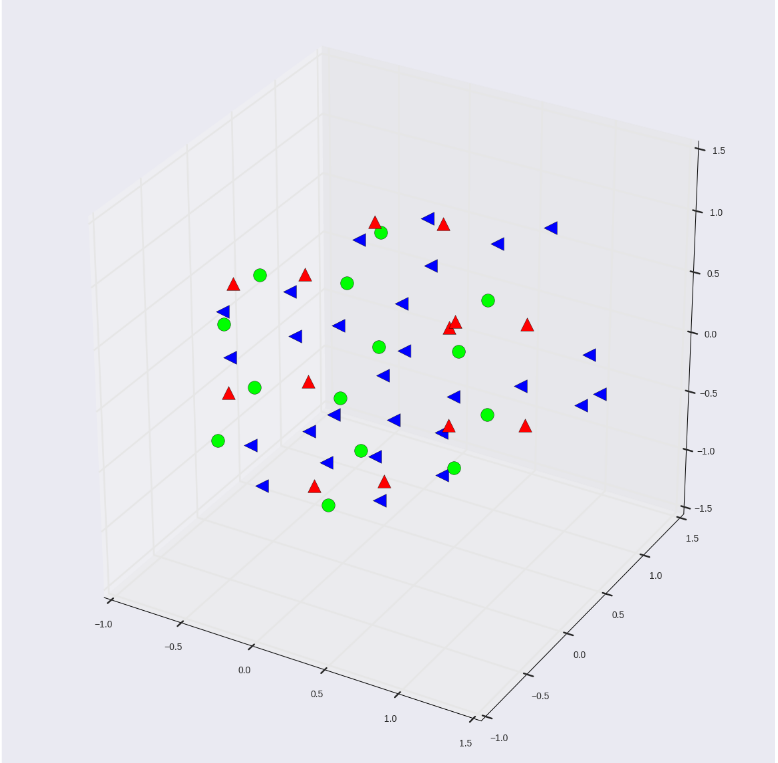
  
Illustration 1: cgn-dataset with max\_abs scaler and 3 Dimension PCA

## Analysis of Clustering Results



## Conclusion of Cluster Analysis

## Data Export

  
Illustration 2: green=cgn, red=lust, blue=combined, max\_abs scaled, 3 dimension PCA, center comparison

# Traffic Light Controll

# Algorithms and techniques

Pybrain, rl, dqn

benchmark

# Methodology

Preparing steps

implementations

refinement

# Reflection

# Improvement

1. www.sumo.dlr.de [↑](#footnote-ref-2)
2. https://github.com/lcodeca/LuSTScenario [↑](#footnote-ref-3)
3. http://sumo.dlr.de/wiki/Data/Scenarios/TAPASCologne [↑](#footnote-ref-4)
4. http://sumo.dlr.de/wiki/TraCI [↑](#footnote-ref-5)
5. Initially it was planned to incorporate different junction types in the clustering analysis and there were more than one type in the dataset [↑](#footnote-ref-6)
6. An example from TAPAS Cologne dataset: ['highway.primary', 'highway.primary', 'highway.residential'] [↑](#footnote-ref-7)
7. A few steps need to be executed before running the importer, they are documented in the comments [↑](#footnote-ref-8)
8. See MLB-NoMLB-comparison.docx for more details [↑](#footnote-ref-9)
9. http://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette\_score.html [↑](#footnote-ref-10)