NISSAN RESEARCH CENTER
SILICON VALLEY

Algorithm Design for Traffic Signal Timings Predictions of Vehicle-Actuated Controlled Intersections using Support Vector Regression

An Application at Nissan Research Center Silicon Valley

by

R. Scheepjens

In partial fulfillment of the requirements for the degree of

Master of Science

in Transport, Infrastructure and Logistics

at the Delft University of Technology

Supervisor:	Prof. dr. ir. Alexander Verbraeck	TPM, Systems Engineering
Thesis committee:	Dr. ir. Maria Salomons	CITG, Transport & Planning
	Dr. Rudy Negenborn	3ME, Maritime and Transport Technology
	Dr. ir. Ali Mortazavi	Nissan Research Center Silicon Valley

People, remember, if you don't know which door to open,
always account for variable change.

- Micky Rosa on Monty Hall problem

Acknowledgements

This thesis is the final deliverable during my time as a student of the Master of Science in Transport, Infrastructure & Logistics (MSc. TIL) at the Delft University of Technology. I would like to express my gratitude towards the people who have contributed to the result of this graduation thesis.

Fist of all, I would like to thank my graduation committee for all the time and effort they have put into supporting me to successfully finish my graduation project. I would like to thank Ali Mortazavi for his endless support during the different stages of this project. Secondly, I would like to thank Maria Salomons for asking the right questions during our meetings and for providing valuable feedback on my report. Also I would like to thank Rudy Negenborn for his contribution and critical notes on my research. Last but not least, I would like to thank Alexander Verbraeck for his guidance and expertise, as well as opening the doors for me to Nissan and giving me the freedom to develop my own ideas.

Furthermore, I would like to thank Maarten Sierhuis who granted me the opportunity to stay at the Nissan Research Center Silicon Valley for 6 months. Also I would like to thank all the members of the NRC-SV, who have adopted me as one of their own from the very start. I am grateful for the opportunities by Nissan during my internship. Words cannot describe the experience during the 6 months in Silicon Valley. In the midst of so much energy and intelligence, it could be accelerating, intimidating but always challenging. This aspect is not reported in my thesis and the reader would perhaps have difficulty to believe it when it was, but it is inerasably stored in my memory like the practical engineering experience that is indeed reported.

Special thanks go out to my family and NT, who always believe in me. The courage and significance that you give me are unparalleled.

R. Scheepjens
May 2016, Rotterdam

Management Summary

In this project the prediction of traffic signal timings of vehicle-actuated controllers for self-driving vehicle technology is investigated. The vehicle-actuated controller is able to adjust the duration of the green and the red signal based on incoming vehicles. The control installation detects the incoming vehicles by using detector loops. The controller is able to change its output signal with short lead times to react on fluctuations in the current traffic state. The controller does not know its future input signal (vehicles to be detected) and therefore is not able to predict its own future state. The future state is described as the moment of switching between the green and red phase and vice versa. Information about the future switching timings of the red and the green signal supports the decision-making layer of the autonomous vehicle. As a result, the information about switching timings prevents the vehicle of a stop-and-go driving pattern and reduces computation power drastically by avoiding stopping and starting several systems in the vehicle. The research question in this project is formulated as following:

How to predict the traffic signal timings of a vehicle-actuated controlled intersection as accurately as possible for in-vehicle usage, utilizing data retrieved from the traffic light controller?

A literature study results into the formulation of conditions and requirements for the prediction algorithm. The following major requirements are taken into account while designing the prediction algorithm: the prediction algorithm should be capable of predicting the switching timings of both the red and green signal. Predicting the switching timings refers to predicting the duration of the green and the red signal. The target variable in the prediction model is the absolute second in the signal phase on which the signal changes and therefore predicts the total time left in the phase of the green and the red signal for every second in the current phase. The permissible prediction error in terms of absolute deviance for both the predicted green and red switching timings is 4 seconds (predefined by Nissan). The prediction algorithm is primarily designed for the prediction of vehicle-actuated controllers that serve a single modality. The prediction algorithm makes use of data retrieved from the same controller for which it tries to predict the switching timings. Data transmission is needed between the controller and the vehicle by using an intermediate hub, such as a cloud structure. The data transmission that is needed for the algorithm follows a single directed path from the controller to the vehicle. No communication back to the controller and also no communication between other vehicles is necessary for predicting the switching timings. The prediction algorithm could be deployed and computed in either the vehicle or the cloud structure that is used for data transmission.

Furthermore, the literature study includes an overview of current prediction approaches. A benchmark of several mathematical approaches, which were used in previous studies, show that the Support Vector Regression (SVR) approach has the biggest potential in predicting the switching timings of the vehicle-actuated controller. The SVR is a statistical learning approach that needs historical data for training patterns in order to predict a given target variable. As part of the literature study a mathematical and computational understanding of the SVR approach is given.

The data and modeling part of this project describes the retrieval and preprocessing of data, model fitting and model selection of the prediction models. The prediction models are trained and tested within a case study. The case study contains an intersection that is located in the Netherlands in the province of North-Holland. The intersection serves a single modality and has a regular cross

geometry including 10 signal groups. Data retrieved from the controller for prediction purposes is called V-log data. The use of V-log data plays a key role in both the development process of the prediction algorithm and the eventual use of the algorithm. V-log data needs to be preprocessed before it can be used for training and predicting the SVR algorithm.

The preprocessing of raw V-log data into training and test data follows a fixed sequential order of steps. Preprocessing of the raw V-log include the actions: manual feature extraction, dataset construction, quality assessment of the data, feature selection, scaling and subsetting into train- and test datasets. The final output of the data preprocessing is useable training and test data for which the amount of corrupted data is reduced significantly. Special attention should be paid to the quality assessment of V-log data. Empirical results show that poor loop detector health is a huge threat for an accurate prediction model. Corruption in the V-log data can be detected by using standard statistical metrics that reveal unexpected control patterns such as jittering loop detectors.

The modelling part consist of an extensive model selection procedure that covers the assessment of trained SVR models in which different settings and data usage of the SVR are tested. The design of the prediction algorithm has led to **two prediction models for each signal group individually**: one prediction model for the switching timings of the green signal and a second model for the red signal. The major difference between the two prediction models is the selection of data from an initial dataset for training and predictions. We demonstrate that the green prediction model only uses data selected from the signal group for which it also predicts the switching timings of the green signal. The red prediction model uses data selected from all signal groups to predict the switching timings of the red signal. For both the green and red prediction model we showed that data retrieved from the first and third detector loops are successfully used for predicting the switching timings. Besides the loop detector data also data describing the internal states of the signal cycle is highly relevant for predicting the switching timings.

The intersection from the case study consists of 10 signal groups for which each 2 models (red and green prediction model) were trained, accounting for a total of 20 models needed to predict the entire intersection.

Model results from the case study show that the developed prediction algorithm is capable of accurately predicting the switching timings of a (fully) vehicle-actuated controlled intersection serving a single modality. The best performing green prediction model reports a mean absolute error of 0.85 seconds and a hit rate of 99%. The mean absolute error indicates the difference between the true switching time and the predicted switching time, while the hit rate gives the percentage of predicted switching timings within a 4 seconds permissible error. The best performing red prediction model shows a mean absolute error of 2.42 seconds and a hit rate of 81%. Improvements in the prediction performance can be gained by numerical optimization of the trained models. Due to time constraints in this project the optimization is not performed and is recommended for further research.

Besides the reported validation metrics, the model results show that the SVR is able to predict expected control behavior. The SVR models predict for every second in the cycle the number of seconds left until the moment of switching. The expected control behavior is that the number of predicted seconds until the moment of switching follows a countdown behavior. Model results show that the prediction algorithm is able to predict such countdown behavior.

Recommendations for Nissan primarily concern the optimization of the current prediction algorithm and the implementation of the prediction algorithm as operational technology. It is recommended to test the prediction algorithm as operational technology in a real time setting such as the simulation environment that is present at the Nissan Research Center Silicon Valley.

Recommendations for further use of V-log data for prediction purposes are primarily related to the logging of the V-log and the use of statistical procedures to asses the quality of V-log. Recommended changes in the logging of V-log include the use of a different data format/extension such as .CSV instead of the initial V-log extension and a lower frequency of logging. Modeling results show that a logging of 1 second time interval of V-log is needed compared to the initial 1/10th of a second time interval. By using a 1 second time interval the amount of processed data is reduced by a factor 10.

Recommendations for further research are related to the research approach and methodology, as well as the development of the prediction algorithm in this project. The results of this project show that statistical learning techniques such as the SVR approach and the use of V-log produce added value for both self-driving vehicle technology and traffic light control. An example of further research is the detection of faulty detector loops or detecting whether parameter settings of a controller need to be updated by analyzing V-log data. Secondly, it is recommended to upgrade the current prediction algorithm by including additional sources of information to create a more robust prediction algorithm. An example is the use of queue estimators when loop detector data is missing or the queue of vehicles exceeds the detection range of the detector loops. Additional, the arriving pattern of vehicles play an important role in the behavior of the controller. Catching the dynamics of the arrival pattern in a variable would be an upgrade for the developed algorithm and could be considered as further research. Finally, the interfacing of a prediction application that utilizes the prediction algorithm is recommended for further research. Examples of topics are data transmission and data latency of V-log, the installation of the prediction algorithm in the traffic light controller or the vehicle and the frequency of retraining the prediction model respecting the chosen prediction application.

Table of content

List of Figures

1 Introduction

Nissan Motor Co., Ltd., Japan's second-largest automotive company, is headquartered in Yokohama Japan, and is part of the Renault-Nissan Alliance. Operating with more than 244,500 employees globally, Nissan sold almost 5.2 million vehicles and generated revenue of 10.5 trillion yen (USD 105 billion) in fiscal 2013. Nissan delivers a comprehensive range of more than 60 models under the Nissan, Infiniti and Datsun brands (NISSAN, 2015a). In 2013 the Renault-Nissan alliance opened a new research facility in Sunnyvale, the heart of Silicon Valley, to become its hub for research on autonomous vehicles, connected vehicles and human Interface to enhance the experience of autonomous and connected vehicles. In addition, planning and advanced development of connected vehicle services and the design of user interface systems are assigned as a complementary field to the above disciplines (NISSAN, 2013). That same year Nissan published to have plans to have commercially viable autonomous driving vehicles on the road by 2020 (NISSAN, 2014a).

1.1 Advances of self-driving vehicle technology

Recent developments in self-driving vehicle technology are moving in a rapid pace. Similair to Nissan, other automotive manufacturers such as Mercedes, Ford, BMW and Audi have opened research facilities that are dedicated to self-driving vehicle technology (KPMG, 2014). Besides the automotive manufacturers also tech companies such as Google, Baidu and Apple started research programs to develop self-driving vehicle technology.

The advancements in self-driving vehicle are classified according a 5-level structure that indicate the level of automation of a system. The levels of automation follow the international standard of J3016 that describes the taxonomy and definitions for vehicle automated driving systems (SAE, 2015). The lowest level, level 0, indicates a level of autonomy in which the driver is continously in control of the speed and direction of the vehicle. The level of autonomy increases during each level, until full automation of the vehicle is achieved. The final level (fifth level) contains systems that perform the lateral and longitudinal driving tasks in all situations during the entire journey. In level 5 no driver is required. Current systems that are researched and brought into production are located in the first and second level of automation. Level 1 assumes that the driver continuously performs the longitudinal or lateral dynamic driving task. Examples of such systems are park assistance, lane keep assistance and adaptive cruise control. Level two assumes that the vehicle is able to perform manouvres automatically, but the driver must monitor the dynamic driving task and the driving environment at all times. An example of such a system is traffic jam assistance.

KPMG released halfway 2015 an industry report in which the main automotive manufactueres were interviewed and asked what level of automation will cause the biggest stepping stone to fully autonomous driving. All interviewed automotive manufacturers indicated that systems in level 3 will be a crucial point during the transition phase from partly autnomous driving to fully autnomous driving due to the required complexity of the systems (KPMG, 2015). Level 3 systems perform longitudinal and lateral driving tasks. The vehicle recognises its performance limits and requests the driver to resume the dynamic driving task with sufficient time margin. The difference between level 2 and level 3 is the frequency of monitoring the systems by the driver. In level 3 the driver does not need to monitor the dynamic driving task nor the driving environment at all times, but the driver must always be in a position to resume control. level 2 asks the driver to always monitor the driving task and driving environment at all times. An example of a level 3 system is the *intersection pilot* that enables the vehicles to negotiate city cross-roads without driver intervention. Carlos Ghosn, CEO of Nissan, announced that intersection-autonomy will be introduced to Nissan's models at the end of this decade (NISSAN, 2014b). As part of the intersection pilot the vehicle communicates with traffic light systems to retrieve information about the current state of the traffic lights and (future) switching timings (NISSAN 2015b). Focus of current Nissan's self-driving vehicle technology is mainly targeted towards the stand-alone autonomous vehicle technology in which the vehicle drives itself without communicating with other vehicles or infrastructure systems. To transition to level 3 systems, the Nissan vehicles need to adapt connectivity services such as connectivity with traffic lights. Maarten Sierhuis, Research Director of the Nissan Research Center Silicon Valley, stated in a presentation in early 2015 that connected systems such as vehicle-to-infrastructure systems like the intersection pilot are the highest level in complexity, but are of utmost importance for getting the autonomous vehicle to the market.

1.2 A joint project with the province of North-Holland

In early 2014 Nissan Research Center Silicon Valley and the Province of North-Holland partnered up for a joint project called 'Traffic Signal Timings Prediction'. This project is part of the connected vehicles research team of Nissan. The connected vehicles research team is currently doing research on intelligent systems that uses the advances in wireless communications and in particular vehicular communications with infrastructure side systems or ITS (Intelligent Transport Systems). This type of communication is also called vehicle-to-infrastructure or V2I. North-Holland contributes to the joint project by supplying rich data coming from traffic light systems of North-Holland as well as giving access to the settings and information about the used control logic. Nissan contributes to the joint project by developing a prediction algorithm for the prediction of traffic signal timings.

1.3 Motivation for developing a prediction algorithm

A prediction algorithm for predicting the traffic signal timings serves several purposes. Firstly, the output of the developed prediction algorithm supports the decision-making layer of the autonomous vehicle (NISSAN, 2015b). The prediction algorithm could be part of in-vehicle control agents that have access to sensors and actuators in the vehicle. A model prediction control (MPC) framework could be applied to control agents in the vehicle, for which the prediction algorithm supplies information about signal timings. As a result, the control agent can take into account all available information and therefore is able to anticipate on undesirable behavior of the vehicle in the future at an early stage (Negenborn, 2010). The information about switching timings could prevent the vehicle of a stop-and-go driving pattern and reduce computation power drastically by avoiding stopping and starting several systems in the vehicle (NISSAN, 2015b).

Secondly, the prediction algorithm could be used to optimize the control structure of the vehicle-actuated controller. The optimization could lead to a more efficient controller. Similar to in-vehicle control agents, a model predictive control framework could be applied to the vehicle-actuated controller for which the prediction algorithm supplies the controller about expected durations of the traffic signal (Asadi et al., 2011). As result, the controller is able to find optimal control measures for the next control cycle (green and red signals) that causes the controller to achieve optimal behavior of the traffic flows. The design of a model predictive controller is not the topic of this project, however the developed prediction algorithm could serve as prediction component within a MPC structure. The focal point in this project is to research the most effective prediction approach that supplies in-vehicle systems with information about traffic signal timings.

1.4 Problem background

In this section the problem and challenges behind the prediction of traffic signal timings is further elaborated upon. The section results in a problem statement which defines the topic of this research. To understand the research challenges, background information is given about traffic light control and traffic light signal timings data.

1.4.1 Brief background information: Traffic light control

The basic mechanism of a traffic light controller can be described by *approaches*, *phases*, *cycles, conflicts* and *signal groups*. An *approach* is a lane or multiple lanes that lead to an intersection. A *signal group* is a set of approaches that are controlled by the same traffic light and therefore share the same traffic light signal (Highway Capacity Manual, 2010). A signal group is often referred as movement group, but in this project the term signal group is being used. The signal groups have fixed numeric labels at an intersection. Labels of signal groups will play a major part in retrieval of the correct data for prediction purposes. More information will be given in Chapter 6 about labeling of the signal groups.

The controller distinguishes several *phases or blocks* that determine (vehicular or pedestrian) signal groups running the green signal at the same time (Wilson & De Groot, 2006). Blocks are separated from other blocks by avoiding conflicts between signal groups. In Figure 1 a conflict, denoted with its *conflict area*, is shown. A rule of thumb is that the two signal groups sharing a conflict area should not get a green signal from the controller during the same phase.

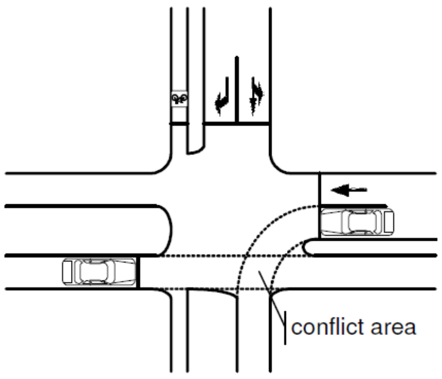


Figure 1 Conflict area (van Katwijk, 2008)

Although creating conflict areas should be avoided, some movements are allowed to proceed during a phase even though they cause conflicts. Pedestrians are commonly allowed to proceed across intersections even though right-turn movements are occurring. These movements are called permitted, while protected movements are those without any conflicts (Highway Capacity Manual, 2010). Protected conflicts are all conflicts that are controlled. The signal groups that cause together the largest conflict group is usually called *decisive conflict group.* The decisive conflict group determines the *maximum cycle time* of the traffic light. *The cycle time* is the time during which all signal groups had the right to turn green. In other words, all phases had the possibility to get a green signal. A second type of cycle exists; a signal group has its own cycle time that is called the *signal* *cycle* and represents the time from the beginning of a certain phase to its re-occurrence after all other means of traffic have been serviced in their respective phases (Protschky et al., 2014a).

1.4.2 Control Schemes

Predicting the signal timings is the central subject in this study. The signal timing elements within each signal cycle for a signal group include the *green*, *yellow* and *red* phases. In order to guarantee a safe operation at an intersection the traffic light controller must always respect certain requirements regarding the minimum length and green-yellow-red sequence (Highway Capacity Manual, 2010). No immediate switch from green to red is possible; a green signal is always followed by a yellow signal. How the controller assigns phase lengths and order of the phases of different signal groups is determined by the control scheme. The area of traffic light control is dominated by three different control schemes. Control schemes are also denoted as 'controllers'. The three main control schemes are:

1) **Pre-timed control:** signal timings follow a fixed schedule from a set of predetermined plans, which were developed off-line on the basis of historical traffic data. The controller usually contains a few schedules based on the time of the day and day of the week. A pre-timed or also called fixed-time controller operates without detectors installed at the intersection. The duration and order of all green phases remain fixed and are not adapted to current traffic demand (Highway Capacity Manual, 2010).
2) **Vehicle-actuated control:** the controller operates according to fluctuations in the traffic state by using detector loops. This detection is performed by using real-time control based on detection loop signals. Signals are triggered by the presence of vehicles. There is a variety in types of vehicle-actuated controllers. They all adjust the length of the current phase in response to fluctuations in the presence of vehicles (Wilson & De Groot, 2006).
3) **Adaptive control:** the adaptive controller monitors the state of the entire intersection. Based on current state and/or traffic upstream the intersection the controller decides to either continue the current green phase or to switch to a different phase. In contrast the vehicle-actuated controller decides to either extend the active green phase or to switch to the next phase based on whether vehicles are still present on the approaches of the active green phase. A vehicle-actuated controller can be considered to suffer from tunnel vision as it does not consider traffic on the other approaches when determining the duration of a green signal (van Katwijk, 2008). Traffic-adaptive control differs from vehicle-actuated control because it can evaluate a set of feasible control decisions and choose a decision that is optimal with respect to its current objectives.

Literature on the topic of traffic signal timings shows a lot of misusage and confusion about the definition of adaptive and vehicle-actuated control. In many cases 'adaptive' control is being used in cases where the control logic of a vehicle-actuated controller is explained.

1.4.3 Traffic light signal timings data: V-log & SPaT

The main data sources for predicting the traffic signal state are considered to be V-log (*verkeerskundige log*) and SPaT data (*Signal Phase and Timing data*). SPaT is an internationally standardized message format for traffic light signals. The SPaT data format is standardized, but the content in the SPaT message is not standardized. V-log is an unstandardized data format (Vialis, 2012), but recognized by Dutch road authorities as most efficient way to log traffic signal data. The province of North-Holland stored V-log data for the past 3 years of every intersection in the region of North-Holland.

The current state of the traffic signals is trivial when the road authority makes its signal phase and timing data available. The messages that contain the current traffic signal states will be described according the SPaT protocol. However, predicting the future state of the traffic lights is not always an easy task even under the condition that SPaT messages are available for the vehicle (Bodenheimer, 2014). For this project we will look at raw data retrieved from the controller. V-log data is raw data coming from the controller capturing all the information that is presented in the SPaT data.

V-log is logged in a Discrete Event Simulation analogy and is compressed to secure transmission efficiencies without losing performance and complexity of the data (Vialis, 2012). V-log is transmitted at a fixed five-minute interval to a central database. Additionally, a logging is transmitted at a frequency of 1/10 of a second. This results in a logging that is approaching a real-time state updating. V-log data contains signals regarding: detector loops, internal signals, external signals and speed detection (Vialis, 2012). As a rule of thumb, the SPaT message only contains output data of the external signal parameters (green, yellow and red). Internal parameters describe transitions between different control schemes within the controller. More information about internal parameters can be found in section 3.2.3. The detector loop data is logged binary and internal/external parameters are logged with predefined coding. All data is logged in a temporal database. Raw V-log data needs to be preprocessed before it can be useful for prediction purposes. In Chapter 7 more information is given about the applied preprocessing techniques in this project. Below a graphical representation is given of the raw V-log data. To understand the data better, one should think in terms of *input* and *output* data for the controller. Input and output data is shown for one signal group in Figure 2. The input data is marked as blue bars at row 021 until 0212. The raw data is logged in binary coding; 0 indicates no vehicle detected and 1 indicates a detection of at least one vehicle. Output data of the controller are the traffic signals at rows 02.

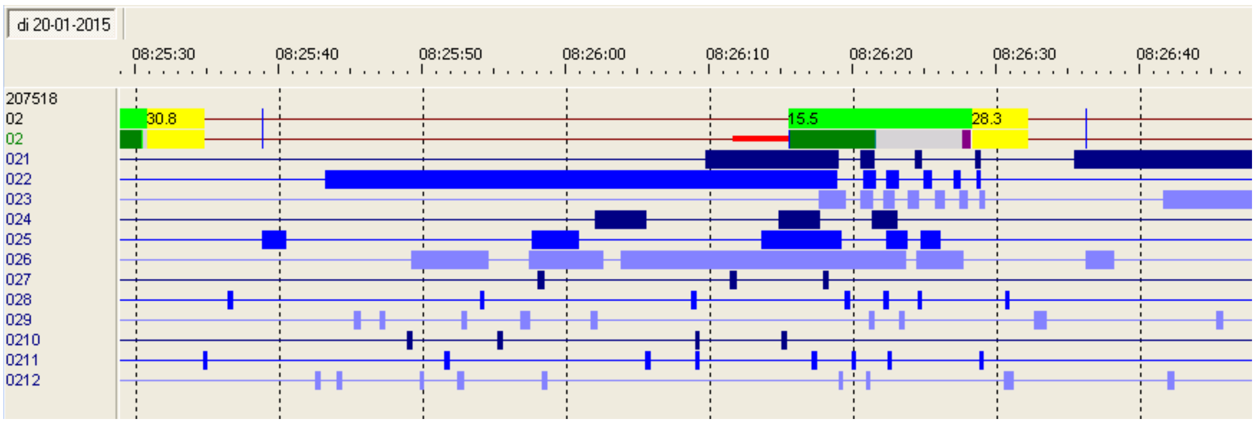


Figure 2 Graphical representation of V-log data of one signal group

1.5 Problem statement

Previous section describes three major control schemes: pre-timed, vehicle-actuated and adaptive control. Predicting the traffic signal timings for *pre-timed controllers* is a trivial execution since it assumes a pre-defined static control scheme. This means that the traffic control scheme follows a relatively simple scheme in an endless loop. The prediction of *vehicle-actuated* and *adaptive controllers* is considered to be far more complex due to their flexible assignment of green phases and cycle lengths. Nowadays almost 85% of the traffic light controllers in the Netherlands are vehicle-actuated controlled (van Katwijk, 2008). Exact numbers of other countries are unknown. Bodenheimer et al. report that of 95% all the traffic light controllers in Hamburg Germany are vehicle-actuated controlled (Bodenheimer et al., 2014).

1.5.1 Flexibility of the vehicle-actuated controller as main prediction obstacle

The vehicle-actuated controller is considered to be a flexible control scheme. This flexibility causes the controller to be difficult to predict. The flexibility is partly caused by the short lead times of 1 second at which the signal can change (Bodenheimer et al., 2014). The controller does change its signal fast because it operates according to fluctuations in the traffic state by using detector loops. Detector loops send out signals that are triggered by the presence of vehicles. Detection is performed by using real-time control based on detection loop signals (Wilson & De Groot, 2006). The controller does not know its future input (traffic) and therefore is not able to predict its own future state (Weisheit & Hoyer, 2014). Even if the controller would have information about the number of approaching vehicles, pedestrians, busses or cyclists, it still would have difficulties with predicting its future state. The reason is twofold:

1) The assignment or extension of a green signal for a signal group is not always directly linked to traffic dedicated for that particular signal group. The control feature of the vehicle-actuated controller is called *inducing green* (meerealiseren) and *continue parallel green* (meeverlengen) (Wilson & De Groot, 2006).
2) Speeding behavior of a vehicle determines the extension of the green phase and therefore influences the cycle length. Situations occur that vehicles are detected by the detector loops, but do not pass a threshold of required *gap times*. Therefore, not only the presence of a vehicle is important, but also the speeding behavior of that particular vehicle is important.

As a result of the flexibility, the controller shows variability in the following output: 1) the sequence of signal groups getting assigned a green phase by the controller, 2) the cycle length of a signal group (signal cycle) and 3) the duration of the green and red signal during signal cycle (Wilson & De Groot, 2006). In conclusion, the flexibility of the vehicle-actuated controller makes is a challenging task to predict the traffic signal timings.

1.6 Research objective and questions

In this section the research objective, research questions and sub-questions are presented. The research objective is to develop an algorithm for traffic signal timings prediction of vehicle-actuated controlled intersections. The research question has been formulated as following:

How to predict the traffic signal timings of a vehicle-actuated controlled intersection as accurately as possible for in-vehicle usage, utilizing data retrieved from the traffic light controller?

During the process of designing the algorithm several decisions need to be made. The following sub-questions are formulated to support decision-making during the development of the prediction algorithm:

1) What functional and non-functional requirements need to be taken into account for the development of a prediction algorithm that is able to predict the switching timings of a vehicle-actuated controller?
    - What is the prediction target of the prediction algorithm?
    - What is the level of integration and interaction of the desired prediction algorithm?
    - What is the level of permissible error on the prediction target?
2) What information regarding the control logic of the vehicle-actuated controller is necessary to take into account for retrieving V-log data that is used for predicting the switching timings?
    - How does the vehicle-actuated controller controls its output signals based on detector loop data?
3) What type of prediction approach complies to the functional and non-functional requirements for a prediction algorithm to predict the switching timings of a vehicle-actuated controller?
    - What prediction approaches are mentioned in the literature in the field of traffic signal timings prediction?
    - What approach is currently used by Nissan for the prediction of traffic signal timings and why would a different approach be preferred?
4) What preprocessing steps need to be performed to transform V-log data to data that can directly be injected into the prediction algorithm?
    - What data in the V-log need to be retrieved that catches the dynamics of the vehicle-actuated controller?
    - How can quality assessment be performed on V-log to determine data corruption?
5) Under what circumstances does the prediction model perform satisfying and under what circumstances does the prediction model fail?
6) How applicable is the prediction algorithm on different scenario's such as different intersections, multiple modalities and times of the day?

1.7 Research approach and methodology

In this section the research approach and methodology is introduced that is applied to answer the research question and sub-questions. The research predominantly requires information about: 1) functional and non-functional requirements of the prediction model, 2) the control logic of the vehicle-actuated controller, 3) current algorithms/predictions models for prediction of traffic signal timings, 4) V-log data and its usage for predictions. Information concerning the four stated topics are gathered by a literature study and interviews with researchers from Nissan and experts from the province of North-Holland. To construct the prediction algorithm, the following key activities are performed: 1) an intersection must be chosen for which V-log data is retrieved, 2) retrieval and preprocessing of the raw V-log data, 3) computation of the algorithm and 4) validation of the constructed prediction model(s).

1.7.1 A case study: vehicle-actuated controlled intersection in North Holland

In this project a case study is performed in which the developed prediction algorithm is tested and validated. The case study consists of a single vehicle-actuated controlled intersection that is located in the region of North-Holland. From this intersection data is retrieved that will be used for both the development and testing of the prediction algorithm. Prior to the case study an explanation will be given about the control logic of the vehicle-actuated controller which is deployed in North-Holland. By knowing and defining the control logic valuable information can be used for the design of the prediction model. Information about the control settings of the controller is gathered by interviews with experts from North-Holland and gathered from supplied documentation by those experts. As part of the case study an intersection is selected in North-Holland.

1.7.2 Algorithm selection and design strategy

Algorithm selection and design is the most comprehensive part of this project. Firstly, an extensive research study is conducted on current prediction models in the field of traffic light signal timings prediction. It shows that some models already do exist but operate on simplified scenarios and follow different mathematical concepts. Examples are the Kalman Filter (Protschky et al., 2014b), Markov chains (Bodenheimer et al., 2014), Support Vector Machine (Weisheit & Hoyer, 2014) and Probability theory (Mahler & Vahidi, 2012). Each mathematical concept deals with the prediction challenge in its own way and was selected due to the availability of data and application purposes. The goal is to give an overview of current used algorithms after which the most promising mathematical approach is selected. Where needed, background information about the chosen mathematical approach is given. The documentation about the chosen mathematical framework is not an attempt to give a comprehensive understanding of the mathematics behind the approach. The documentation will cover sufficient information to understand preceding chapters regarding the development of the algorithm. The algorithm will be selected according the prediction requirements. To scope the project a design strategy is formulated. As part of this the design strategy the approach is respected to start simple and add complexity. In this project the focus is on intersections that serve a single modality. The algorithm is trained and validated for the morning peak only. Intersections that serve modalities such as bicycles, pedestrians and public transport are not part of the case study. Once the model is developed it could be tested and used for other times of the day and additional modalities.

1.7.3 Computation

In this study numerous activities are performed that need the use of a programming language. For this project the statistical software R is used as programming language for performing data analysis, preprocessing of the raw data and the development of the algorithm. R is an open source programming language for statistical computing and graphics (RStudio, 2016). The source code of R is written in C and Fortran. The capabilities of R are extended by the use of user-created packages (Matloff, 2011). R-studio will be used as a graphical interface on top of the R core software. The Google R Style guide is used in this project to develop standardize code for readability and reproducibility (Ledolter, 2013). Although the developed scripts respect the Google R Style guide, the final script cannot be considered as operational software. The development of a faster version of the algorithm as a fully operating software plugin is considered to be a topic of future research.

1.7.4 Data retrieval and preprocessing of V-log data

As mentioned in the problem background, North-Holland is providing this joint project with all existing V-log data of traffic signal systems in North-Holland of the past 3 years. Several executions need to be performed in order to run a prediction model properly on the V-log data. Firstly, the right data needs to be retrieved from the controller. Insights about the control logic will support the decision-making process of selecting the correct data from the V-log data. Secondly, the data needs to be preprocessed and stored in a useable data format. This task should not be underestimated since the raw V-log data is coded in a binary and categorical fashion and is logged in a temporal database. Most prediction algorithms need numerical data and run into trouble when binary data is used (Mitra & Acharya, 2003). Preprocessing of the V-log data includes the transformation of binary V-log to numerical data and create variance in the data. Other preprocessing techniques such as *scaling* will need to be executed before the data can be injected into a prediction model (Hastie et al., 2009). Thirdly, the data needs to be inspected for data corruption and missing values. The inspection of the data is performed by univariate and bivariate statistical analysis. A univariate analysis gives descriptive statistics of one single variable. Univariate analysis include statistical summaries of the metrics: *mean, mode, variance, min value, max value*, and the *standard deviation*. Bivariate statistics involve the analysis of two variables to determine the empirical relationship between them. In order to see if the variables are related to one another, it is common to measure the dependence in terms of correlation.

1.7.5 Validation of the prediction model

The generalization performance of a prediction model relates to its prediction capability on independent test data. Assessment of this performance is extremely important in practice, since it gives us a measure of the quality of the chosen model (Mitra & Acharya, 2003). *Validation* is understood to be *''the process of determining the degree to which a model is an accurate representation of the real world from the perspective of the intended uses of the model'* (Los Alamos, 2004). The validation procedure relies on the chosen mathematical framework that is used as prediction approach. A method for estimating prediction error is cross validation. This method directly estimates the expected extra-sample error, which is the average generalization error when the developed prediction model is applied to an independent test sample (Hastie et al., 2009). To execute an efficient cross-validation procedure, which utilizes the amount of available data as much as possible, K-fold cross validation is used. In K-fold cross validation the dataset is split into a number of K-folds. The model is trained on K-1 fold data, where after the model predicts the data of the remaining 1 fold of data. This process is repeated K times (every time with a different fold as prediction data), after which the parameter estimates of each fold are averaged to determine the performance of the classifier (Hastie et al., 2009). It is likely to get reliable validation statistics that approaches the expected prediction error on a test set when a large number of observations (samples) are used.

1.8 Research relevance

1.8.1 Relevance for the automotive industry

Several automotive manufacturers publicly stated to bring semi-and fully autonomous vehicles to the road within 10 years (KPMG, 2014). Focus of the self-driving vehicle technology is currently focused on the stand-alone autonomous vehicle technology in which the vehicle is executing driving tasks without communicating with other vehicles or infrastructure systems (KPMG, 2015). To transition current level of automation from partial autonomous driving towards fully autonomous driving, the in-vehicle systems need connectivity with infrastructure systems such as traffic lights. This project gives understanding what value traffic light data has for the vehicle and what obstacles need to be taken into account while developing self-driving vehicle technology that deals with intersections. Regardless of the level of intelligence of the vehicle, intersections remain a highly complex area for the self-driving vehicle. Any system that gives predictive power to the vehicle regarding the future of such an area contributes to the safety and acceptance of self-driving vehicle technology. This project and forthcoming insights support developments of the self-driving vehicle technology.

1.8.2 Relevance for road authorities

To our knowledge no project has been performed in which V-log data was used for predictive purposes for self-driving vehicle technology. This research shows the relevance of V-log data for autonomous vehicles and how the structure and content could be changed in order to be compatible with self-driving vehicle technology. Additional, this project could serve as a use case in which V-log data is used in combination with statistical learning approaches. Furthermore, road authorities could gain insights form this project in how self-driving vehicle technology embeds traffic light control logic for predictive purposes and what information is needed to do so. As turns out sufficient loop detector health is important to support signal timings prediction models. Currently the SPaT protocol is being designed and agreed on by governmental institutes. The format of the SPaT message is already being agreed on, but the content is not fixed yet. Results and insights gained from the research as described in this thesis could open up new directions of the content of the SPaT message. It could prevent the SPaT message from containing non-informative data for the vehicle and inclusion of data that is highly relevance for prediction purposes.

1.8.3 Relevance for the TIL domain

As mentioned before, this project stresses the importance of high quality data retrieved from the traffic light controller for prediction purposes. In-car systems that include connectivity of the vehicle with infrastructure systems benefit from knowledge of the TIL (Transport, Infrastructure and Logistics) domain. Developing new TIL infrastructure systems on the other side also benefits from knowledge about self-driving vehicle technology that interfaces the vehicle with the infrastructure. This project adds value to the TIL domain in giving insights how traffic data produces added value to self-driving vehicle technology and what intermediate steps are required at the infrastructure side to produce that added value. More important, this project serves as an example that the design of future TIL projects should take interfacing with self-driving vehicle technology into account. At last, this project shows that not only technical research on the prediction of switching timings is necessary to achieve a successful implementation of a prediction algorithm, but also alignment of the road authorities, regulatory institutes (responsible for protocols) and the automotive industry is needed to achieve a successful implementation.

1.8.4 Societal relevance

In this project the province of North-Holland and Nissan Research Center Silicon Valley are engaging in order to benefit from knowledge of the use of V-log for self-driving vehicle technology. As a result, a better understanding of used technologies for both North-Holland and Nissan is gained. Understanding technologies on both sides of the infrastructure and the vehicle opens up doors to coordination and alignment of used technologies by the government and the automotive industry. The coordination between infrastructure systems and the newly developed self-driving vehicles technology could prevent unsafe situations around intersections. Eventually road users such as drivers, pedestrians and cyclists benefit from safe intersections.

1.9 Thesis overview

The thesis is divided into three parts that are set up in the following way:

- **Part I:** contains the literature study described in chapters 2 to 5. Chapter 2 describes the functional and non-functional requirements that set guidelines for the development of the prediction model. The requirements are used in Chapter 5 to select a suitable prediction approach. Chapter 3 contains valuable information about the vehicle-actuated controller that is needed to understand what information is logged in the V-log data. Chapter 4 summarizes the current algorithms and mathematical frameworks that are used for traffic signal timings prediction. Chapter 4 finishes with the most optimal type of mathematical approach suited for this research. Chapter 5 gives a deeper understanding of the chosen mathematical framework and the computation of the mathematical approach. The goal of Chapter 5 is not to give a fully comprehensive overview of the mathematics behind the algorithm. The Chapter will provide the reader with sufficient information to understand preceding chapters regarding the development of the algorithm.
- **Part II:** contains the data and modeling part described in chapters 6 to 8. Chapter 6 describes which intersection is used as a case study and how this intersection was selected. From the selected intersection data is retrieved that serves as input for Chapter 7. Chapter 7 describes the preprocessing of V-log data into usable data for prediction. Chapter 8 explains the development of the prediction model and all decisions that were made during the model selection.
- **Part III:** contains the results and final remarks described in chapters 9 to 11. Chapter 9 gives the validation results and deeper insights concerning the prediction behavior on given test data. Chapter 10 discusses all the relevant decisions during the research and discusses the implementation and future perspective of the prediction algorithm. Chapter 11 contains the conclusion of the research and reaches back to the initial goal and research questions of this project. Furthermore, Chapter 11 gives recommendations for Nissan and the further use of V-log data and the developed prediction algorithm. Additional recommendation for further research are given.

Figure 3 gives an overview of the key activities in this thesis.

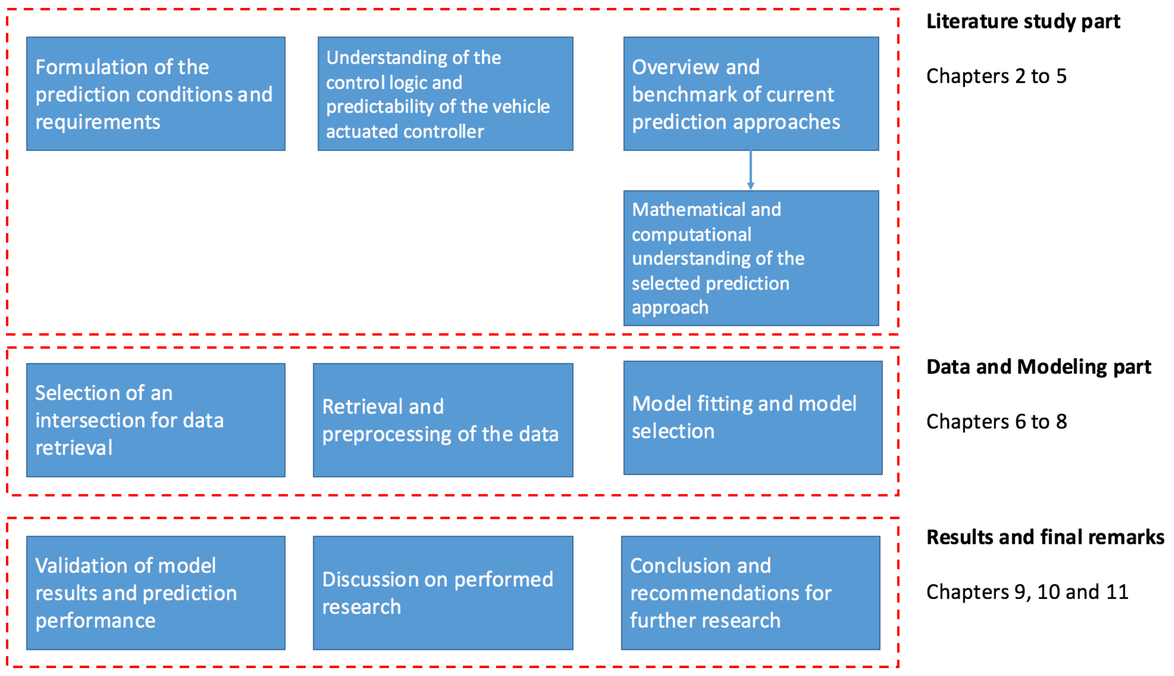

Formulation of the prediction conditions and requirements
Understanding of the control logic and predictability of the vehicle actuated controller
Overview and benchmark of current prediction approaches
Mathematical and computational understanding of the selected prediction approach
Literature study part
Chapters 2 to 5
Selection of an intersection for data retrieval
Retrieval and preprocessing of the data
Model fitting and model selection
Data and Modeling part
Chapters 6 to 8
Validation of model results and prediction performance
Discussion on performed research
Conclusion and recommendations for further research
Results and final remarks
Chapters 9, 10 and 11

Figure 3 Key activities described in this thesis

PART I

Literature Study

2 Conditions and requirements for the prediction model

The research objective is to develop an algorithm for traffic signal timings prediction of vehicle-actuated controlled intersections. To develop an effective algorithm, it should be clear what requirements should be met. This Chapter introduces conditions and requirements that scope and bring focus to the development of the prediction model.

2.1 Previous prediction applications

The prediction of traffic light controllers in previous research is usually related to Green Light Optimal Speed Advisory (GLOSA) applications. A GLOSA system contains a prediction component for pre-timed or semi vehicle-actuated control schemes (Seredynski et al., 2013b). The prediction component usually consists of an algorithm. The level of complexity of the control schemes in previous research is considerably lower compared to the control scheme used in this project. Previous studies explicitly stated the application for which the prediction component is needed. An example of such a study is the Traffic Light Assistant application by Audi (Zweck & Schuch, 2013). The focus of these studies is angled more towards the integration of the prediction application instead of the theory behind the prediction algorithm. In this project the focal point is placed on a proof of concept of an algorithm to predict the controller and therefore has the ambition to formulate a prediction algorithm instead of a full application.

An example of a conducted GLOSA research is the Audi Travolution project. The project describes an on-board system that receives real-time traffic signal data from individual traffic signals via dynamic short-range communication (DSRC) and provides speed recommendations to the driver (Brain et al,. 2009). The system also provides time-to-green and time-to-red information to the driver. However, the system does not make predictions for the traffic signals and therefore can only provide recommendations that account for traffic lights within the short DSRC range. In addition, Green Driver provides a mobile telephone application that predicts the states of traffic signals based on real-time traffic signal information from a traffic center (Apple et al., 2011). The application calculates the speed required to pass through a traffic light before it turns red, and provides suggested routes to avoid red lights. However, Green Driver makes predictions for only pre-timed control schemes and is not able to produce predictions for vehicle-actuated controlled intersections.

The prediction application that shows similarities with to the prediction task of this project has the name SignalGuru (Koukoumidis et al., 2014). SignalGuru uses crowdsourced data to detect and predict the traffic signal schedule. Initially SignalGuru uses windshield-mounted mobile telephones to acquire real-time images of traffic signals and determine the color of the traffic signal. The system analyzes data acquired from other vehicles to recommend when drivers should reduce their speed to avoid idling at a red light. In Chapter 4 more information and examples are given regarding previous prediction algorithms for the prediction of traffic signal timings.

2.2 Prediction target variable

Research shows different prediction targets when the future traffic signal state is predicted. On abstract level two prediction targets are being used: either making a *probability statements* about every second in a cycle (Mahler & Vahidi, 2012) or making statements of *the switching time* in terms of classification or regression (Weisheit & Hoyer, 2014). In previous research probability statements were always focused on the probability of having green time at a certain second in a cycle.

The red signal is treated as an inverse probability as counterpart of the estimated probability function of the green signal. Having probability estimations is from a statistical point of view pleasant, since it shows a direct measurement of accuracy, but lead to compulsory decisions. The obstacle is how the threshold of a probability of green is set to indicate a true green. Barthauer and Friedrich state in their research a probability threshold of 0.7 to indicate green time (Barthauer & Friedrich, 2014). Setting this threshold seems arbitrary and several researches making probability statement about traffic signal states do not clearly indicate their threshold. The second and more useful target variable for the prediction model is estimating the *transition time or switching time* of the signal. The output of the prediction should describe *the absolute second of the cycle* at which the phase signal of the traffic light changes. By doing so, estimations of phase lengths and cycle lengths can be made.

2.3 Functional requirements for the prediction model

The functional requirements describe what the prediction model should be capable of doing. The functional requirements are determined partly by requirements that were stated in comparable studies and partly by design statements of Nissan. The prediction model has the following functional requirements:

1) **Prediction target:** the prediction model should be capable of predicting the switching time of both the green and red signal (see section 2.2).
2) **Prediction target:** the prediction model should be capable of predicting the end of the current phase. The moment of the end of the current phase and entering the next phase is called the switching moment. The switching time is the exact time in seconds at which the switching moment takes place. The switching time is related to the prediction target (see section 2.2).
3) **Prediction horizon:** the prediction model is primarily focused on predicting switching timings during the morning peak. However, the prediction model should respect generalization and scalability of the prediction model towards the afternoon, evening peak and night.
4) **Prediction horizon:** from the perspective of a GLOSA systems, the prediction model only predicts switching timings of the first upcoming intersection. Literature about GLOSA systems often mentions lower and upper bounds of the prediction horizon. Although developing a GLOSA system is not the subject of this research we did include information about the lower and upper bound of the prediction horizon. See section 2.5 for more information.
5) **Data usage:** the prediction model should utilize V-log data that is retrieved directly from the controller for which the model tries to predict its switching timings.
6) **System integration**: the assumption is made that the prediction model is located and computed in a cloud structure or in the vehicle. Nevertheless, it is always assumed that the cloud structure is retrieving data from the controller. The cloud structure sends either raw

V-log data to the vehicle or the predicted switching timings (only in case when the prediction model is deployed in that same cloud structure). In other words: the vehicle-to-infrastructure connection does include direct communication between the controller and the vehicle.

7) **Scalability:** the prediction model should be capable of predicting the switching timings for each individual signal group of the intersection. See section 3.1 for more information about signal groups.
8) **Scalability:** the prediction model is primarily designed to predict switching timings of the vehicle-actuated controller. It is likely to assume that the prediction model can also be used for predicting switching timings of semi-vehicle-actuated controllers. However, semi-vehicle-actuated controllers are not the focus of this research and are outside the scope of this project.
9) **Scalability:** the prediction model should be scalable to more than one intersection serving a single modality that may have various types of intersection layouts/geometries. The prediction model should be independent of intersection layout and controller design that is within the boundaries of a vehicle-actuated controller.
10) **Scalability:** the prediction model is primarily focused on intersections serving a single modality. However, the prediction model should respect generalization and scalability towards intersections serving multiple modalities such as busses, vehicles, pedestrians and bicycles.

To scope the prediction goal, the following functional requirements indicate what the prediction model should not be capable of:

11) **Data usage:** the prediction model for predicting the switching timings will not use any data coming from the vehicle. The prediction model will only use data from the controller.
12) **Prediction horizon**: the switching time of both the green and red signal only applies on the current phase onto the next phase. The prediction model will not be designed to predict multiple switching moments ahead in time. Previous research indicate that the stochastic nature of traffic makes it to impossible to accurately predict multiple switching moments ahead in time for vehicle-actuated controllers.
13) **System integration:** the prediction model does not take into account multiple intersections such that the prediction output could be directly used for routing purposes. The prediction model could be used for routing purposes if multiple prediction models would predict switching timings for intersections in a given area. This would be the topic of a different research.
14) **System integration:** the prediction could be used for a GLOSA (Green Light Optimal Speed Advisory) system. The GLOSA system itself is not the goal of this research and is outside the scope of this project. In the context of a GLOSA system the functional requirements described in this section would match the predictions needs of a GLOSA system. See section 4.1 for more information about GLOSA.
15) **System integration:** the prediction model does not utilize a vehicle-to-vehicle component. Nor does the prediction model take into account the driving behavior of surrounding vehicles that in reality might influence the prediction model at the moment of predicting the switching time. No data is accessible for this project of such driving behavior.

16) **System integration:** the prediction model does not contain a component in which the vehicle transmits information back to the controller. The data stream is a one-way direction from the controller to the vehicle in which the data streams through a cloud structure.

2.4 Non-functional requirements for the prediction model

The non-functional requirements are determined in consultation with Nissan. Previous performed studies hardly mention non-functional requirements. The following non-functional requirements describe how the prediction model should behave:

1) **Prediction accuracy:** the accuracy of the prediction model should be as high as possible. A level of permissible error of 4 seconds around the predicted switching is allowed. The error of 4 seconds was given by researchers of the Nissan Research Center Silicon Valley. The 4 seconds permissible error describes a switching timing that is incorrectly predicted for approximately two passing vehicles. No previous studies indicate how a permissible error was constructed and only mentioned the final accuracies after prediction. In addition, no empirical justified reasoning is present why a 4 seconds boundary is preferable compared to a 3 or 5 seconds error boundary. An increase of the error boundary will lead to a higher hit rate of correctly predicted switching timings. A decrease of the error boundary will lead to a lower hit rate of correctly predicted switching timings.
2) **Computation time:** the computation time should be relatively low. The prediction algorithm will most likely be deployed in the vehicle. The vehicle has limited amount of available computation power to predict the switching timings. Ideally we would like to quantify this non-functional requirement. However, such statements are closely related to the application for which the prediction is used. In case of a GLOSA system, the computation time of the prediction should be in ranges of seconds. General taken we accept prediction models that show computation time in ranges of seconds and minutes. Models that need hours and even days to compute should be avoided.
3) **Flexibility of data usage:** the prediction algorithm should be capable of using V-log data and therefore should be able to deal with the non-linearity characteristics of the V-log data.

2.5 Prediction horizon in case of a GLOSA system

The design of a GLOSA system is not the subject of this research. Because it is likely that the technology described in this document is suited for a GLOSA system we describe what prediction horizon is needed for a GLOSA system. Looking at the current research on the evaluation of traffic light signal predictors, a variety of prediction horizons are shown. Some of these systems take strong assumption of the dynamics of the traffic controller into account, leading to ambitious prediction horizons of several minutes ahead. However once the control logic becomes more complex and uncertainty is added to the data, the prediction horizons decrease significantly. The prediction horizon will be denoted by an upper bound and lower bound.

2.5.1 Lower bound of the prediction horizon in case of a GLOSA system

Figure 4 is a visualization of the prediction horizon needed for a GLOSA system. *The lower bound* of the prediction time horizon is given by the time for sending a message to a vehicle, its reaction time and a possible brake to standstill. Barthauer and Friedrich state that 6 seconds would be sufficient enough for passengers cars (Barthauer & Friedrich, 2014). When taking an average approaching speed of 50 km/h into account, <u>*6 seconds*</u> leads to approximately 83 meters. The lower bound should be seen as the last 6 seconds before the vehicle reaches the intersection. During these 6 seconds a real time state updating by direct communication between the controller and vehicle would be preferable instead of a prediction.

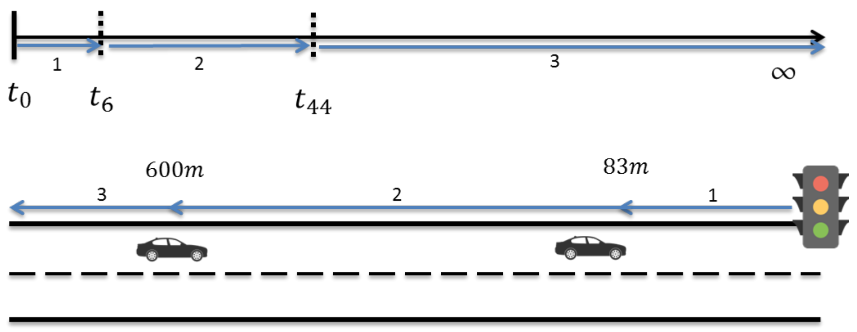


Figure 4 Visualization of prediction horizon

2.5.2 Upper bound of the prediction horizon in case of a GLOSA system

The upper bound of the time horizon can be described in terms of cycle lengths or distance of the vehicle towards the intersection. The problem is that the vehicle-actuated controller unlike other controllers does not have a fixed phase and cycle length. Research shows that phase and cycle lengths for the vehicle/actuated controller can vary quite drastically (Krijger, 2013). Therefore, making generic statements beforehand about the prediction horizon using cycle length is difficult, since this could be in the range of tens of seconds or minutes. Especially in the validation process the prediction will probably show artificially underperformance or overperformance since two cycles are hard to benchmark due to their variety in length. Another factor is how useful the prediction becomes when the prediction horizon is to large and therefore is amplifying noise in the prediction. Research shows a huge variety on the decision of the upper bound of the prediction horizon. Tielert et al. describe that for speed advisory type systems the benefit of fuel reduction becomes negligible for information distances greater than 600 meters (Tielert, et al., 2010). Tielert et al. did supply the system with predictions of the traffic signal state up to 87 seconds. The average speed of vehicles was considered to be 50km/h, resulting in a prediction horizon of 1200 meters. Barthauer and Frierich argue that the distance between two subsequent stop lines rarely exceeds 500 meters (Barthauer & Friedrich, 2014). In their opinion predicting beyond 26 seconds (500 meters at average speed of 50 km/h) is not useful for a GLOSA system. It was also recommended not to exceed the 600 meter limit due to negligible fuel economy benefits. The upper bound for a GLOSA system is therefore set at 600 meters, taking an average speed of 50km/h into account leading to an <u>*upper bound of 44 seconds*</u>.

2.6 Summary Chapter 2

In this Chapter the following sub-question is answered: *What functional and non-functional requirements need to be taken into account for the development of a prediction algorithm that is able to predict the switching timings of a vehicle-actuated controller?* To answer the sub-question, the following three sub-sub-questions are answered in this chapter:

- *What is the prediction target of the prediction algorithm?*

The prediction algorithm should be capable of predicting the switching timings of both the red and green signal. The target variable in the prediction model is the absolute second in the signal phase on which the signal changes.

- *What is the level of integration and interaction of the desired prediction algorithm?*

The prediction algorithm is primarily designed for the prediction of vehicle-actuated controllers and could be used as part of a GLOSA system. The algorithm is designed to be scalable towards intersections that serve a single modality. The prediction algorithm makes use of data retrieved from the same controller that it tries to predict. Data transmission is needed between the controller and the vehicle by using an intermediate hub such as a cloud structure. The data transmission, which is needed for the algorithm, follows a single directed path from the controller to the vehicle. No communication back to the controller and also no communication between other vehicles is necessary. The prediction algorithm could be deployed and computed in either the vehicle or the cloud structure that is used for the data transmission.

- *What is the level of permissible error on the prediction target?*

The permissible prediction error for both the predicted switching timings of the red and green signal is 4 seconds. The permissible error is predefined by Nissan. Prior studies regarding the prediction of traffic switching timings did not report permissible error boundaries on the prediction.

3 The vehicle-actuated controller

In this Chapter the vehicle-actuated controller is further elaborated on. In Chapter 1 three different control schemes are mentioned of which the vehicle-actuated controller is one of them. The vehicle-actuated controller is the most common deployed traffic light controller in the Netherlands and is optimized to the needs of an efficient traffic throughput (van Katwijk, 2008). As described in Chapter 1, the optimization and flexibility of the controller scheme makes it difficult to predict its switching timings. Unlike the pre-timed controller, the vehicle-actuated controller produces different lengths of green phases, cycle lengths and also can deviate from the predefined sequences to a certain degree in which the signal groups can have a green signal (Wilson & De Groot, 2006). Additional the vehicle-actuated controller can adapt its control scheme to incidental situations such as the arrival of busses or trams. To describe the vehicle-actuated controller a case study is being used. The case study is limited to the area of North Holland in the Netherlands and an intersection that is controlled by a vehicle-actuated control scheme. The goal of this Chapter is to give an explanation about the control logic of the vehicle-actuated controller. Additionally, insights are given into what components of the controller are useful for prediction purposes.

3.1 Geometry and labeling of signal groups

An intersection consists of multiple signal groups. Usually each signal group has its own label, that is often a number. The unique labelling of signal groups is convenient while retrieving data of an intersection and improves the use of the prediction algorithm. The V-log data logs the data of signal groups according its a label number. Figure 5 shows how the signal groups of a standard intersection are labeled.

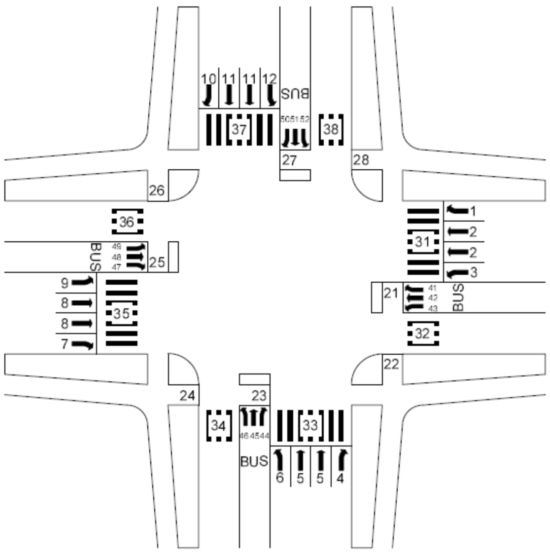


Figure 5 Labeling of signal groups (van Katwijk, 2008)

Signal groups are sorted in modalities in which each modality has its own label (Wilson & De Groot, 2006). In the table below the allocated labels for each modality is given.

Modality	**Standard labels**
Vehicles	1- 12
Bicycles	21- 28
Pedestrians	31 - 18
Public transport	41 - 52

Table 1 signal group labels

Technically the labeling in Figure 5 is not the same as the labeling of the signal groups. For example, approaches 11 do not have the same signal group number '11', but have the number 11.1 and 11.2. Since the signal groups share the same signal state as the labels of the approaches/lanes this label is considered as one single label for the two signal groups. The labeling of the signal groups reveals a variety in possible geometries of an intersection. As the complexity of an intersection's geometry increases so does the control structure (i.e., the composition and the sequence of the green phases).

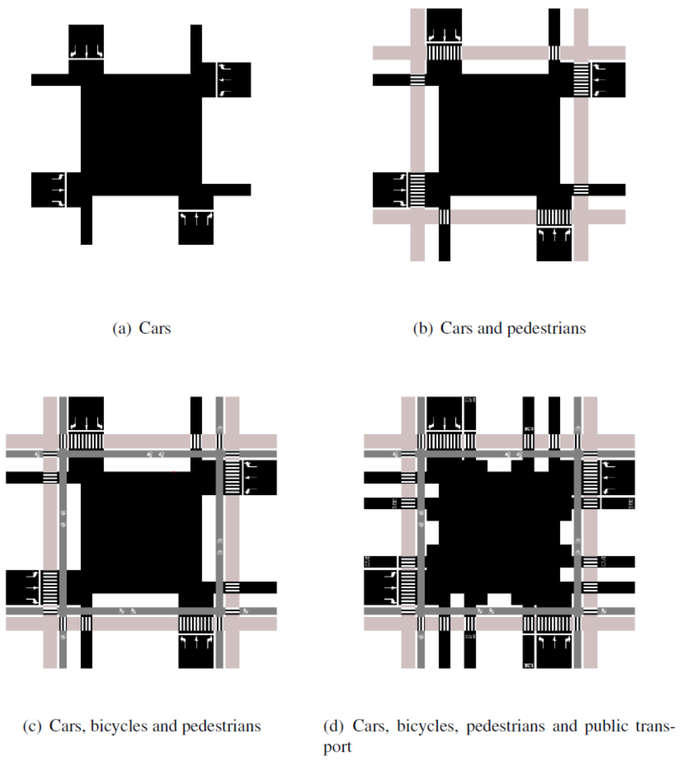

(a) Cars
(b) Cars and pedestrians
(c) Cars, bicycles and pedestrians
(d) Cars, bicycles, pedestrians and public transport

Figure 6 Common intersection geometries (Muller & de Leeuw, 2006).

In many European countries such as the Netherlands, in contrast to the U.S.A., controlled intersections often have separate infrastructure for bicycle and public transport movements (van Katwijk, 2008). The main reason for this is to improve traffic safety by separating the weaker from the stronger road users in time and space, and to realize priority treatment of certain categories of road users. As a result, the geometry of an intersection can be rather complex as shown in Figure 6. The structure of the vehicle-actuated controller and its control cycle is commonly based on the decisive conflict group. The decisive conflict group is a group of movements that have the biggest conflict in terms of time of all possible combination of signal groups (Wilson & De Groot, 2006). The decisive conflict group grows in size when the intersection facilitates more approaches and modalities. The object of the controller is to sequence the decisive conflict group as efficient as possible, but despite being sequenced efficiently the decisive conflict demands the longest cycle time. The selection of optimal sequence of realization of green phase is determined my minimization of internal lost times by avoiding long clearance times.

3.2 Control logic of vehicle actuated controller in North-Holland

In North Holland the control approach that is being used show huge similarities with the defined control approach by Rijkswaterstaat (the executive agency of the Dutch Ministry of Infrastructure and the Environment). The method is called the RWSC-approach (Wilson & De Groot, 2006). The control approach can be understood by dividing the method into two levels: 1) block procedure and 2) signal group completion. Figure 7 gives a simplified overview of the two levels within the vehicle-actuated controller. Loop detector data serves as input for both levels of the controller. Within the block procedure the sequence of the realized signal groups is determined. The block procedure is considered to be at a higher level in the control logic. In the signal group completion level is determined how much realization time each phase gets assigned. The signal group completion is considered to be at a lower level in the control logic. A block is defined by this method as a collection of signal groups that can be green simultaneously.

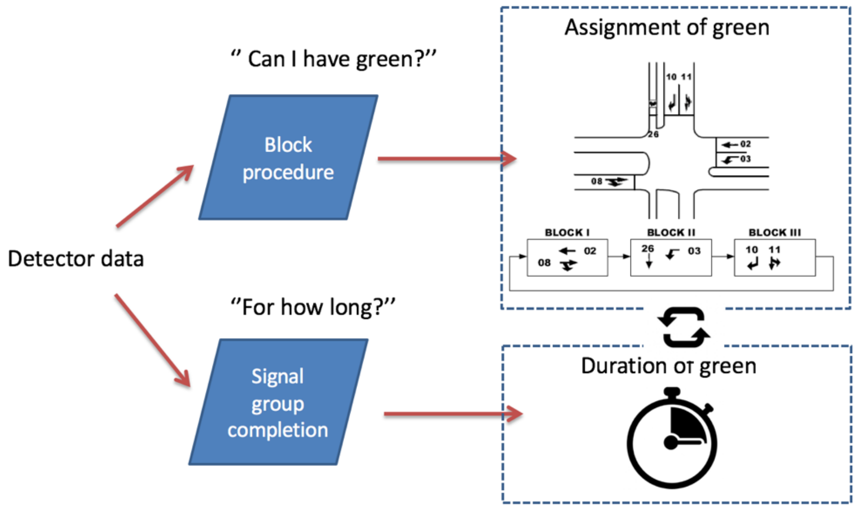


Figure 7 Simplified overview of the levels of control within the vehicle-actuated controller

3.2.1 Block procedure

An important feature of the block procedure is the assignment of signal groups to blocks. A block is defined as a collection of signal groups that can be green simultaneously (Wilson & De Groot, 2006). The sequence in which signal groups get green is specified in the so-called block structure. To determine which signal groups belong to which block the decisive conflict group is being used. Each signal group of the decisive conflict group represents one block. More signal groups that are not in the decisive conflict group are added to the blocks. Figure 8 shows a block structure. The selection of signal groups belonging to a block is based on mutual conflicts. The sequence of the blocks, consisting of multiple signal groups each sharing the same mutual conflicts is determined by minimizing a loss function (van Katwijk, 2008). The loss function minimizes the switching time between the decisive signal groups defined in each block. We describe three realization methods: the primary realization, the primary ahead realization and the alternative realization. To cover every detail of the realization types we would need much more documentation and is unnecessary for this study. The following description of the realization methods should give a clear and concise understanding of the philosophy behind the realization type.

The standard realization procedure of the block, **the primary realization**, is as follows: if a block is in an active state, the signal groups of the block are allowed to become green. It is necessary to remember that in a primary realization a signal group that is allowed to become green has the formal right on having green. If all signal groups of a block in a primary realization became green or if the decision has been taken to skip the green phase for the signal group the next block becomes active (van Katwijk, 2008). The signal groups of an active block get "the turn to become green" as soon as all conflicting signal groups of the preceding block are in the state yellow, red, or parallel green. From this moment on, the signal group completion takes over the further realization of the signal group. As soon as each signal groups of the block had its moment to become green, the next block becomes active. Input for this procedure is information from detectors and from the status of the signal groups.

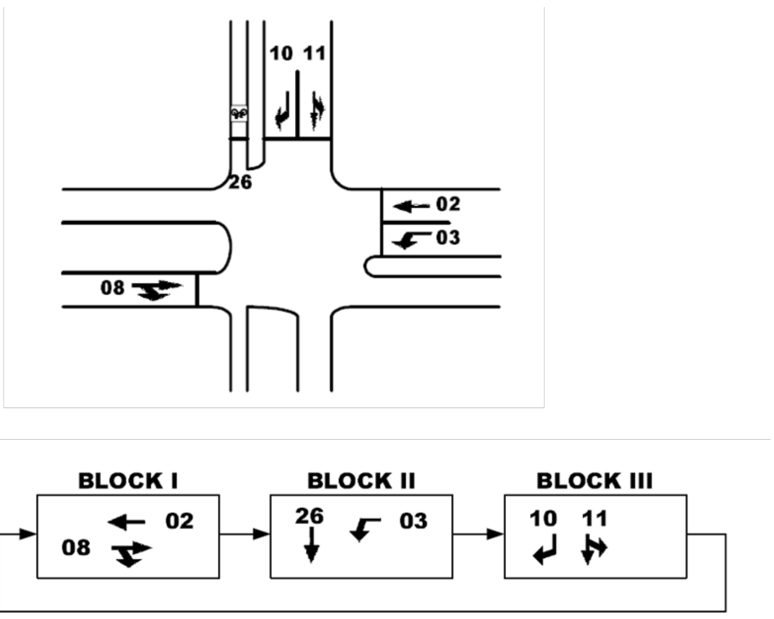


Figure 8 RWS controller configuration: intersection geometry and block structure (van Katwijk, 2008)

The controller can also use the realization strategy ***primary ahead realization*.** Within this realization a signal group of only the *next* block in the sequence can have the ability to turn green. For example: all signal groups of the intersection are red and block I is active (see Figure 8). Assume that there is no request for green for signal group 8 and that there is demand for signal groups 2 and 3. Signal group 2 and 8 are both allowed to turn green. As there is no demand for signal group 8, this signal group waives the right to turn green. Then Block II becomes active, and signal group 8 loses its right to become green. Because all conflicting signal groups of signal group 3 are red, this signal group can become green simultaneously with signal group 2 (even though they belong to different blocks). This shows that the block procedure offers the possibility for signal groups of different blocks to become green simultaneously.

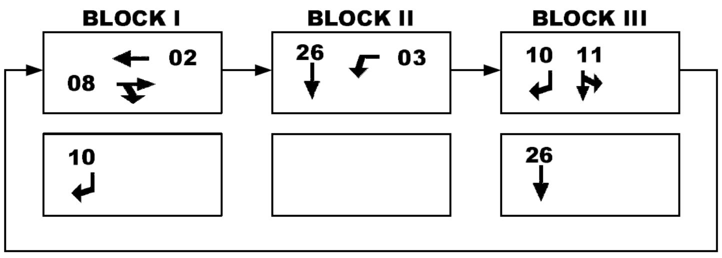


Figure 9 Alternative realization (van Katwijk, 2008)

The realization method within the vehicle-actuated controller that is being used in North Holland is called the alternative realization method. The name is referred to an additional ***alternative realization*** that the control can execute (van Katwijk, 2008). The main idea of this realization method is to fill up time gaps in which other signal groups can turn green that do not cause conflicts of signal groups operating in a **primary realization** (the primary right to have a green signal). In the case of Figure 9 signal group 10 can have the right to turn to the green signal while Block 1 has assigned to turn green. Initially signal group 10 is located in Block 3. The idea behind the alternative realization method is to fill up unnecessary greentime during the realization of a block that has no conflict with signal groups from other blocks.

3.2.2 Hierarchy in realization methods

The controller assigns green signals to blocks and signal groups in a hierarchical way. The hierarchy for a single modality intersection is summarized in chronological realization order: 1) primary realization, 2) primary ahead and 3) alternative realization. A signal group can be realized one cycle as primary realization and during another cycle as alternative realization depending on the control decisions. Briefly, a signal group that has the realization *primary realization* has overruling power compared to a signal group that has an *alternative realization*. The *priority realization* that is assigned to incoming buses at an intersection overrules every other type of realization. The priority realization will stop any other realization, where after the controller will proceed in the primary realization with the first next upcoming block at which the scheme was overruled by the priority realization. The method of realization for a green phase is logged into the V-log data. This could be valuable information for predicting the switching time of a green and red phase. When a signal group is being assigned a green phase under an alternative realization, the duration of the green phase is likely to be small.

3.2.3 Signal group completion

On a lower level of the controller the signal group completion is being executed. At this level the duration of the red and green time is assigned to signal groups (Wilson & De Groot, 2006). To determine the duration of the green phase the controller uses detector loop data. Detector loop data will be further explained in the next section. Figure 10 shows the fixed sequences of phases during a cycle of a particular signal group. The controller can skip phases by setting the lead time of a phase at almost 0 seconds. Theoretically it does not skip a phase but reduces the lead time to such a small interval that it is not noticeable.

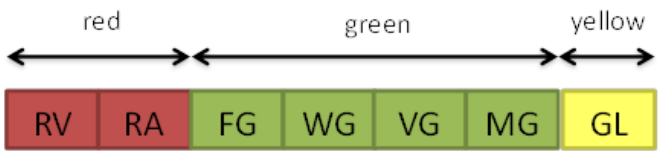


Figure 10 Sequence of signal group completion

In total there are 7 sub-phases. V-log data contains data concerning each of the 7 phases from Figure 10. We expect that data regarding the 7 sub-phases that is logged within the V-log data will give predictive power and/or explanatory power for the prediction model. This assumption seems relevant, since the 7 sub-phases explain internal state changes within the signal group completion level which determines the duration of the red and green phase. Below a brief description is given for each sub-phase.

RV – *Red before request* (rood voor aanvraag). In this phase the controller is switched from a yellow phase to a red phase. The condition for the controller to have the *state red before request* is if controller at the level of block procedure did not assign the signal group to have the right to turn green. The control state will be red before request at the moment there is no traffic demand for a particular signal group.

RA – Red after request (rood na aanvraag). When the controller detects a signal from the detector loops*,* due to one or more incoming vehicles, the block procedure assigns the right to turn green to the requested signal group. There is a time delay in the assignment of the RA state from the controller to the signal group because it needs to calculate if the signal group indeed has right to turn green. Once the signal group reach the RA state it has a high probability to turn green. The right to turn green can be overruled by an incoming buss on another signal group. This signal group will get the assignment of a priority realization, meaning that it overrules any RA state. The duration of the RA state is determined by clearance times of conflicting signal groups running green. The duration of RA is not fixed, but shows some variability because a signal group can have multiple conflicts.

FG – *Fixed green* (vast groen). The FG state is fixed at usually a 4 seconds interval. This 4 seconds is the minimal green time that the signal group completion gives before extension of green time is deduced from detector loop data. This means that a green phase length prediction can never be lower than 4 seconds.

WG – *Waiting green* (wachtgroen). Waiting green starts when fixed green phase ends. Waiting green is theoretically only used after the extension of the green phase has expired and no conflicting signal groups have traffic demand. The controller decides that it is not necessary to change the green signal since there is no conflict present at the intersection. During the morning peak the waiting green phase rarely occurs since there is usually traffic demand on other signal groups. Once an intersection is synchronized with an intersection, the waiting green phase can be used to artificially extend the green phase such that incoming vehicles hit a green light.

VG – *extending green* (verlenggroen). At the same time that waiting green starts, the extending green phase also starts. The extending green phase is the most crucial part of the green phase that causes the variability in green phases based on detector loop information. The extending green phase has two functions: 1) the current queue of vehicles gets time to enter the intersection, 2) incoming vehicles get the chance to hit the green light and enter the intersection. The extending of the green phase is determined by so called 'gap times' that are measured by the detector loops. In the next section the detector loop configuration is explained. It is important to mention that 'gap times' are reached when a vehicles reaches certain time thresholds between detector loops. The thresholds of the 'gap times' are fixed parameters within the controller.

MG – *prolonging green* (meeverlenggroen). The prolonging green phase usually gets realized for a signal group when other signal groups in the same block still have traffic demand that is served by the green light. In such a case it would be useless to change the green signal to a red signal because conflicting signal groups still need to wait for the other signal groups in that same block to be finished realizing the green signal.

GL – *yellow* (geel). The duration of the yellow phase has commonly a fixed interval. The intersection that is selected for the case study have a fixed yellow time of 5 seconds.

3.3 Use of detector loops

The vehicle-actuated controller is able to react in the traffic demand because it has detector loops located in the the ground at several locations at the intersection. The detector loop configuration typically consists of three or four detector loops located in the ground for each lane (Wilson & De Groot, 2006). Figure 11 illustrates the location of loop detectors for the southern four lanes. The loop detectors give a high binary signal once a vehicle passes the loop detector. Otherwise the loop detector gives a low binary signal. The data that is produced by the detector loop data indicate events: the passing of a vehicle. A vehicle that approaches the intersection would first be detected by the third detector loop data, followed by the second and last by the first detector loop.

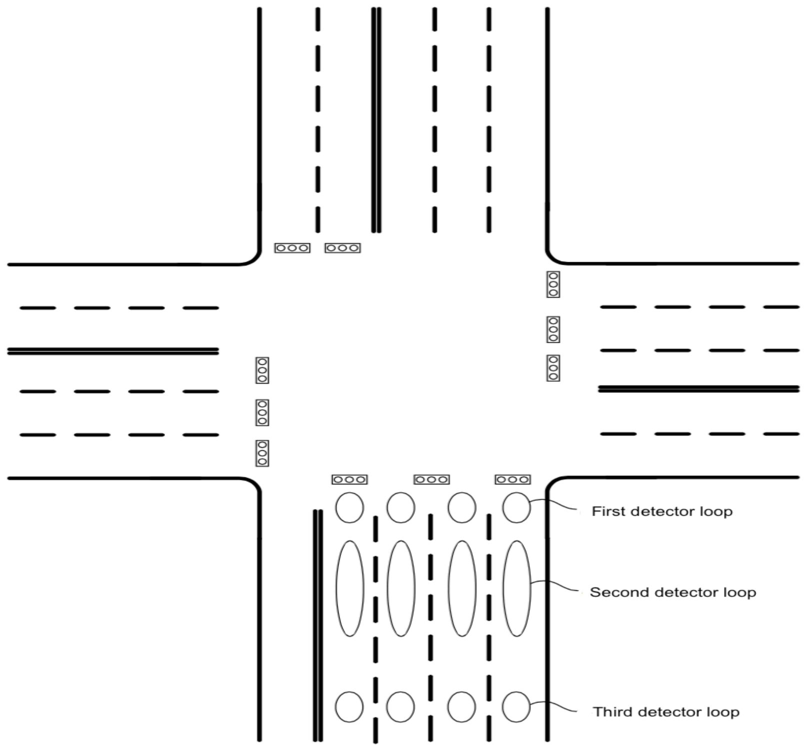


Figure 11 Location of detector loops

On average, the first detector loop is usually located 5 meters before the stop line. The second loop detector, that consists of a larger detector, is located at 20 meters before the stop line. For turning movements, the second detector loop is located at 10 meters before the stop line. The third detector loop is located at 60 meters before the stop line. In some cases, a fourth detector loop is present at 80 meters. Since the fourth detector loop is not always present we do not take this loop detector into consideration for the development of the algorithm.

3.3.1 Gap times of detector loops

The detector loops work with a mechanism called gap times. The gap time is the elapsed time between two high binary signals due to a passing vehicle (Wilson & De Groot, 2006). Once a vehicle passes a detector loop within the interval of the gap time, the green signal is extended. Each detector loop (first, second and third) has its own predefined gap time. These gap times are fixed parameters for each intersection and could be adjusted due to the located of the intersection or the expected traffic demand to ensure a more efficient throughput. The first and second detector loop have the primary goal to make sure that the green duration is sufficient to serve the queue of vehicles that was present during the previous red phase. The third detector loop is used to extend the maximum green duration to a longer green duration due to incoming vehicles. The gap time for the third detector loop is approximately 3 seconds for intersections in North-Holland. For the second loop detector this is approximately 0.5 seconds and the first detector loop also has a gap time of approximately 3 seconds for intersections in North-Holland. Theoretically the third detector loop should give the most predictive power, since it is located at a larger distant in space and time to the stop line compared to the first and second loop detector. But the third detector loop is tricky and not always directly linked to an extension of the green signal. The reason is that the third detector could give a high binary signal once an incoming vehicle is detected, but at the same time the signal exceeded the gap time. In such a case the vehicle is indeed detected but does not influence the extension of the green duration. We also could state that not only the presence of a vehicle at a particular lane will trigger a possible extension of the green signal, but also the speed of the vehicle is important. The speed of the vehicle relates to exceeding the gap times on the detector loops. If the vehicle has a rather low speed during the extending green phase it is likely that the controller will switch off the green signal due to exceeding gap times.

3.3.2 Request signal

A relevant data stream within the controller that might be useful for prediction purposes is the request signal. The request signal gives a high binary value when a vehicle is detected by the detector loops during a red phase (Vialis, 2012). This particular signal triggers the controller to arrange the signal group that request a green signal to get a green phase. Within the V-log data the request signal is logged with binary coding, such that a high binary signal indicates that a vehicle is detected and that the signal groups asks for a green phase. The request signal occurs during the red phase: red before request. After the request is send to the controller, the red phase changes from red before request to red after request. Once the signal group reaches the red after request state it has a high probability to turn green. Bottom-line is that the binary signal regarding the request signal has some predictive power, since it occurs previously before a green light. In theory this should give predictive power for the red duration prediction model.

3.4 Summary Chapter 3

In this Chapter the following sub-question is answered: *what information regarding the control logic of the vehicle-actuated controller is necessary to take into account for retrieving V-log data that is used for predicting the switching timings?* To answer this sub-question the following sub-sub-question is answered in this Chapter: *How does the vehicle-actuated controller controls its output based on detector loop data?*

Theoretically the vehicle-actuated controller controls its output signal for sets of signal groups at the same time. However, the controller has a high level of flexibility. Due to the flexibility the controller controls each signal group individually. The labelling of the signal groups is important for retrieving the correct V-log data from the controller.

The vehicle-actuated controller consists of two main control levels that are called the block procedure level and signal group completion level. Both levels describe how the controller regulates its output signals based on input signals. Loop detector data serves as input for both levels of the controller and therefore is important data for the prediction model. Within the block procedure level, the sequence of the realized signal groups is determined. The block procedure level is considered to be at a higher level in the control logic. Data that describes the block procedure level is considered to be important for the prediction model. In the signal group completion level is determined how much realization time each phase gets assigned. The signal group completion is considered to be at a lower level in the control logic. The signal group completion level describes 7 internal states that are logged in V-log. Data regarding the 7 internal states is considered to be valuable for the prediction model.

The detector loops are located at several locations in the ground. The controller decides to extend the green phase by measuring gap times between detected vehicles. The relation between detected vehicles and reached gap times is not logged in V-log. The detector loops submit a request signal when an incoming vehicle is detected. The possibility to extend until a certain amount of time is also determined by the block procedure control level within the controller.

4 Selection of a prediction approach

The goal of this Chapter is to give insights in the used algorithms in the field of traffic light signal prediction. An overview of the different algorithms and details of their application, usage of data and performance are given. Not every algorithm tries to predict the same phenomenon. For example, the Support Vector Regression predicts the *switching time* of a traffic signal state whereas the Kalman Filter makes a *probability statement* for every second within a cycle. This Chapter results into the selection of a prediction approach that matches the requirements from Chapter 2. The selection is supported by a benchmark of several prediction approaches that are described in this Chapter.

4.1 GLOSA (Green Light Optimal Speed Advisory)

In general, the prediction problem could be used for Green Light Optimal Speed Advisory (GLOSA) system. GLOSA is seen as the dominant system that uses a prediction component that predicts traffic light states. The GLOSA system date back to 1983 and was introduced by Volkswagen (Eckhoff et al., 2013). Because of the hardware limitation at that time the GLOSA system was cancelled for further development. In 2009 Audi continued a similar GLOSA system as a concept under the Travolution project, currently little is published about this project (Bodenheimer et al., 2014). The goal of GLOSA is to use accurate current information and future predictions about traffic signal states in order to guide drivers with optimal speed advice towards an intersection. The GLOSA application minimizes acceleration, braking and improves fuel economy (Seredynski et al.,, 2013b). In a simulated environment the GLOSA system showed a performance of 7% reduction in average fuel consumption and 89% in average stop times (Katsaror et al., 2011). A GLOSA system usually consists of two predictors: distance-travelling predictor and a traffic signal state predictor. The basic principle is to calculate a speed recommendation based on the distance to the traffic light and the time to the next signal change. For this the GLOSA system predicts the future traffic signal state. GLOSA implementations show that traffic signal timings messages are wirelessly transmitted from roadside systems to the vehicle using IEEE802.11p (dedicated short-range communication) or other network technology such as UMTS. Till so far only GLOSA systems are implemented that take into account pre-timed controllers, however research does show attempts being made of GLOSA systems predicting vehicle-actuated controllers. In the section below relevant algorithms are presented that are found during a literature study.

4.2 Statistical approaches versus non-statistical approaches

Prediction approaches can be categorized into three groups:

1) *Prediction methods without the use of statistical procedures*, that approach the prediction with a complete replication and simulation of an intersection. This approach depends on complete knowledge of the mechanism and programming of the particular traffic light and real time-time availability of switching time data with a latency of not more than a few seconds (Protschky et al.,, 2014b). Predicting a reasonable horizon of state changes requires prediction on both traffic demand and its behavior in terms of velocity. When doing so uncertainty increases drastically since there will always be an offset of simulated driving behavior and real driving behavior. Predicting the driving behavior of one vehicle directly affects the driving behavior of other vehicles. As was shown in Chapter 3, the extension of green time is influenced by the velocity of the vehicle approaching an intersection. Replication of an intersection in a simulation environment lead to a computational intense task when one wants to simulate every intersection.
2) *Statistical procedures*. In contrast to simulation approaches, statistical models are more scalable and could operate faster once a fitted and calibrated model is updated with real time data (Koukoumidis et al.,, 2012). The downside is that statistical models need data to be trained and calibrated. Problems arise when too much noisy data is injected in the model. If not executed properly, the model could be fitted too much to the noise instead of the real dynamics of the controller. Predicting dynamical models such as the vehicle-actuated controller with statistical approaches usually requires complicated models and preprocessing of the data.
3) *Hybrid approaches*. The term hybrid is being used to label models that use control logic incorporated in a statistical framework (Protschky et al., 2014b). Usually the control logic would be used as input for replication and simulation of the controller. Examples are detector loop information or public transport arrival schemes.

All approaches struggle with the same issue. Once more data is made available, the less generic the model becomes. The loss in generality is due to the diversity of traffic light management of road authorities and local authorities (Protschky et al., 2014b). Because of the requirements to design a scalable, generic and fast prediction as possible a statistical procedure will be selected. In the following section an overview is given of possible statistical procedures for predicting the traffic signal timings.

4.3 Overview of prediction approaches in the field of traffic signal timings prediction

In this part of the study six types of predictions approaches/methods are being described. The first five approaches: Support Vector Machine, Support Vector Regression, Kalman Filter, Probability Theory and Markov-Chains are found during a literature study on the prediction of traffic signal timings. The sixth method is the previous approach of Nissan that makes use of an Artificial Neural Network.

4.3.1 Support Vector Machines

Weisheit and Hoyer developed a Support Vector Machine (SVM) predictor, that predicts the switching timings of traffic signals for vehicle actuated controllers (Weisheit & Hoyer, 2014). The

SVM model was primarily designed for a GLOSA system. The SVM is statistical learning approach that is used for classification purposes (Hastie et al., 2009). It feels counterintuitive to use a classification model to predict the switching timings. The desired output of a switching timings prediction model is the exact second of the cycle for which the traffic signal switches from green to red. The switching timing should be considered as a numerical output instead of a class. This problem was solved by treating each second in the cycle as a class. The yellow phase is considered to be part of the green phase and no information in the article was given how the end time of red and start time of green for the next cycles were being calculated. The SVM classifier learns from historical signal timings and induction loop data. The induction loop data is being used by the classifier to derive a state of traffic volume. This state of traffic volume is mapped by the classifier to the behavior of signal timings (switching moment of a state). No detailed information was given how an intermediate classification was used to map detector loop data to traffic volumes. The underlying data model that is decisive for the classification consists of detector data of all detectors of an intersection as input and the resulting switching timings are treated as target variables. Since the detector loop data is non-linear a kernel function is used to transform the data into a higher dimensional feature space to make it possible to perform the linear separation in the induction loop data.

Weisheit and Hoyer used variables for training and predicting the SVM that indicate traffic volume and signal timings from the past three cycles and an additional variable describing detector loop data of its current cycle. Information of the current cycle is perceived as real time updating of the prediction. The model reaches an accuracy of 97% of predicted end of green timings of the current phase. The accuracy was calculated by using a contingency table of predicted frequencies and observed frequencies. The SVM is designed to be easily extendable for extra input data and easy adaptable to other intersections with multiple modalities. The SVM as shown in the published paper of Weisheit and Hoyer is limited to one modality, ignoring pedestrians, public transport and cyclists. A drawback of this research is that they used fixed cycle lengths. An important detail since in our case we do not have fixed cycle lengths that makes the SVM as a classifier difficult to use.

4.3.2 Support Vector Regression

Koukoumidis et al. presented in 2012 the prototype SignalGuru that uses windshield-mounted smartphones and their camera's to collaboratively detect and predict the schedule of the traffic signals (Koukoumidis et al., 2012). The application can predict traffic signal switching timings for vehicle-actuated traffic controllers. Koukoumidis et al. refer to predicting the switching timing as the future transition of the light and the length of the current or next phases. The predictions are injected to a GLOSA application informing the driver about a desired speed. The prediction is based on a Support Vector Regression (SVR) model. The SVR is a special class of the SVM. The SVR uses the same principles as the SVM, but the output is a real number instead of a class. The SVR is more complex than the SVM, but the main idea is always the same: to minimize error, individualizing the hyperplane which maximizes the margin, keeping in mind that part of the error is tolerated (Smola & Schölkopf, 2003). The SVR uses only data from the traffic signal timings of the last five cycles. Koukoumidis et al. found that using a longer history of data does not significantly improve the traffic signal prediction accuracy. The input data are: 1) the past five lengths of the specific phase that is being predicted, 2) the lengths of the past phases in the same cycle, 3) total length of the past 5 cycles, 4) loop detection saturation information for the past 5 cycles. Koukoumidis et al. are unclear about what loop detection saturation information is.

The SVR was trained on a week of history data, which was sufficient enough to obtain results. The computation time for predicting (not training) the SVR output was measured on 21 ms (Koukoumidis et al., 2012). Furthermore, the research shows that the SVR does not need to get continuously retrained. Koukoumidis et al. state that once every four to eight months the model should be retrained. The accuracy is defined in term of the *mean absolute error*. The accuracy is therefore defined as an averaged absolute deviation in seconds from the true switching time. For a vehicle-actuated controlled intersection in Singapore an error was reached if 2.45 seconds in predicting the length of the current phase with the SVR. No information was given about the complexity of the tested intersection likewise the amount of different modalities. Statements about its adaptability to other intersections and modalities are not being made. The SVR can easily be extended for more data sources.

4.3.3 Kalman Filter

Protschky et al. show that a Kalman Filter technique is being used for predicting the traffic signal state for vehicle-actuated controllers (Protschky, 2014b). This research was executed together with BMW. Unlike the SVM and SVR the Kalman Filter makes a probability statement for every second within a cycle for a phase. By doing so an overview of probability distributions of green signals are given for every consecutive phases in a total cycle of the intersection. Protschky et al. tried to satisfy the trade-off between data availability and applicability by establishing a prediction model relying on minimal information. A restrictive assumption that was made in this research is that the cycle time is fixed. This indicates that the adaptive controller that they investigated is not as flexible as the vehicle-actuated controller. The input data for the Kalman Filter are frequency distributions of green signals for every second in the cycle, that show variation over different amounts of considered cycles. Law of large numbers dictates that by averaging over a large number of data (in this case cycles) true parameters can be approached. The input data is only limited to historical switching data. In the research the use a *number of considered cycles (NCC)* of 20 cycles. Each cycle has a length of 90 seconds. The main principle of the Kalman Filter is a state updating mechanism by correcting its predicted probability states. In the figure below a visualization of the applied Kalman Filter on signal timings is given. The red curves describe the actual frequency distributions of green signaling, the blue ones describe the predicted probability distributions.

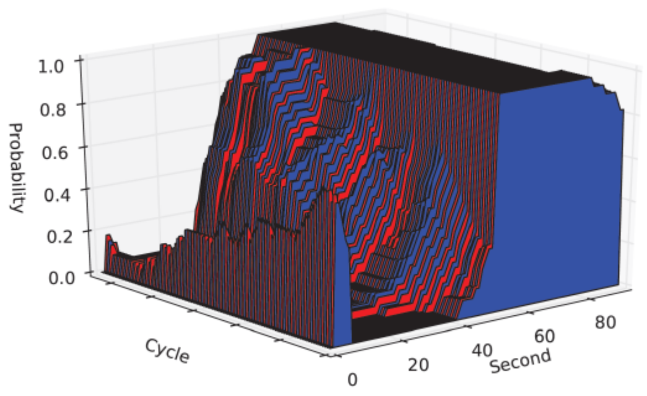


Figure 12 Illustration of the Kalman Filter process (Protschky, Wiesner, & Feit, 2014)

Protschky et al. use an evaluation metric consisting of two parameters: *a posteriori accuracy* indicating the number of correct predictions in percentage of total predictions and *availability* in percent of number of potentially possible predictions. The parameters correlate negatively, so a trade-off should be made. Increasing the availability leads to a decrease in the a posteriori accuracy. The BMW-requirement for using the Kalman Filter for GLOSA purposes is a 95% (a posteriori) accuracy-requirement. An availability of 71% with an accuracy of 95% was achieved. An advantage of the Kalman Filter is that it easily can deal with data dropouts. In case data is not available the Kalman Filter switches to a prediction mode. The Kalman filter is computational intense compared to the SVM and SVR method. The model however saw public transportation as seldom events so were being treated as outliers. An additional model should be made to take public transportation also into account. No information was given about the degree of complexity of the intersections such as modalities and traffic conditions that were taken into account by the Kalman Filter.

4.3.4 Probability Theory

Several research studies are done on using probability theory for traffic light signal prediction. TomTom performed a research to optimize routing based on probabilistic statements of traffic signal timings (Krijger, 2013). The prediction system based on probability theory of Mahler and Vahidi uses Signal Phase and Timing (SPaT) information to make probabilistic statements of upcoming phases of a cycle (Mahler & Vahidi, 2012).

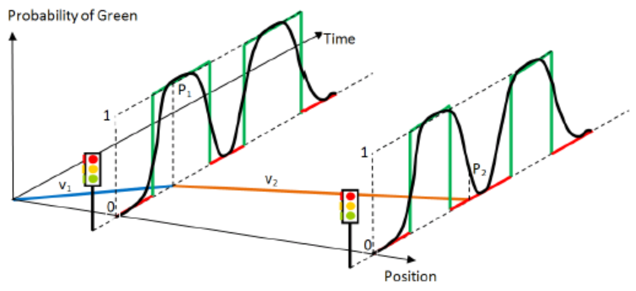


Figure 13 Schematic of velocity planning based on probability of green for two consecutive lights (Mahler & Vahidi, 2012)

Mahler and Vahidi designed the application for adaptive controlled intersections, but no explicit statements are made whether a variable cycle length is taken into account. From the denotation of the probability model can be derived that the cycle length is indeed fixed, meaning that the degree of adaptiveness for which this model was designed is low. Assumption for their prediction model are: 1) the current phase (color) and 2) the average red and green lengths for a signal are known. They use this information to predict the probability of green over the planning horizon. No performance measurements of the actual traffic signal state predictions were given.

Krijger describes probabilistic models that use red and green time distributions for intersections over different segments of the day (Krijger, 2013). He defines a model for pre-timed controllers and a model for vehicle-actuated controllers. No multiple modalities were taken into account and no comments on expected performance of the model including multiple modalities are given. His results show that during rush hours the variations in green and red timings is low for straight on traffic signal groups. Therefore, it seems to be valuable for making predictive probability statements. However, for intersections that handle low traffic volumes, distribution functions are not informative and show low probabilities. Additional a lot of signal groups controlling turning movement show weak distribution patterns. The figure below shows such a weak pattern for two time segments of the day for turning traffic.

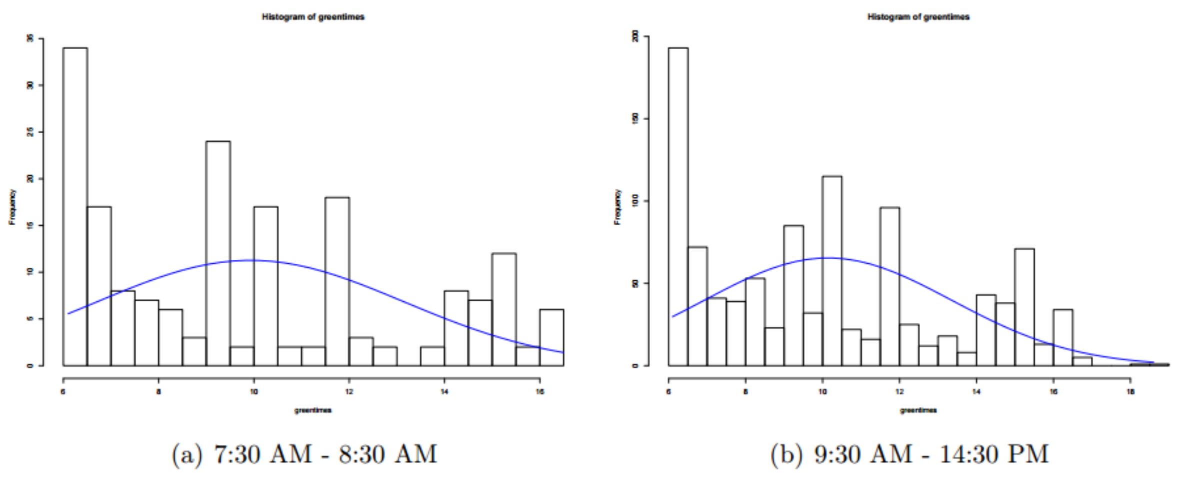


Figure 14 Green timings signals for a turning traffic at an intersection in the city of Helmond in the Netherlands (Krijger, 2013)

The conclusion from the research executed by TomTom is that only signal groups at a vehicle-actuated controlled intersection that show normal distribution functions are useful to use for probabilistic models. Due to the high variance of the phase and cycle length, strong probability statements of traffic signal timings cannot be made. This is shown by the behavior of the probability function as output of his model. The function convergences fast towards a probability of 0.50 of signal timings, leading to non-informative predictions.

4.3.5 Markov-chain

Barthauer and Friedrich show the usage of a Markov-chain based approach to predict traffic signal timings of vehicle-actuated controllers in a microscopic traffic simulation software package (Barthauer & Friedrich, 2014). The Markov-Chain was tested on a vehicle-actuated controller with fixed cycle length. The intersection that was simulated does include different modalities such as pedestrians, bicycles and trams. The Markov-chain is a mathematical approach that represents the dynamics of the traffic light system as a controller graph. Such a graph consists of nodes that represent a state of signal combinations and edges for possible transitions between them. This controller graph needs to be provided by the operator or can be statistically derived from historical data. One state represents an entire state of the signals for all lanes. The controller graph is transformed to a transition graph to reliably predict the order and the time offset of the transition (Bodenheimer et al., 2014). To forecast a traffic light signal the probability of the transition has to be predicted. Therefore, the correct order of the transitions and their lengths have to be determined.

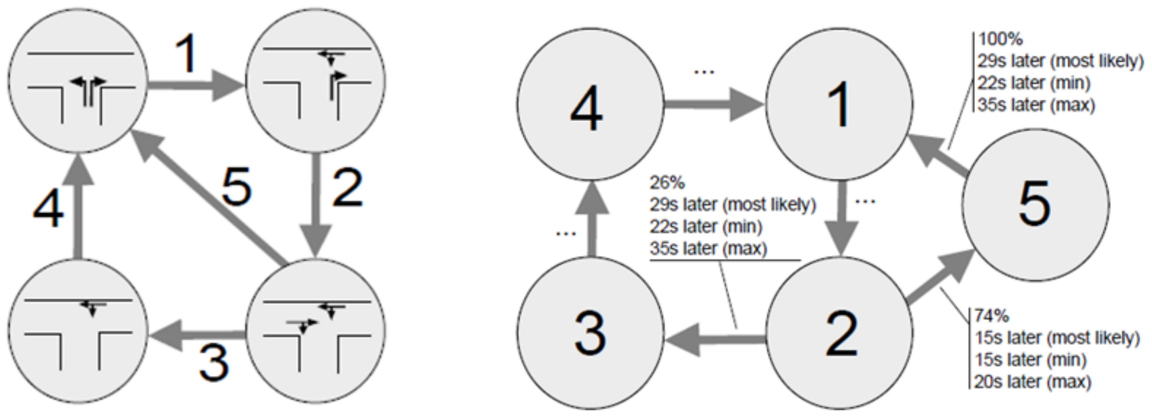


Figure 15 Controller graph on the left and transition graph on the right (Barthauer & Friedrich, 2014)

By using historical data these probabilities, referred as possible transitions between states, are being calculated. The signal states have to comply with the Markov property. This means that their occurrence depends only on a directly preceding signal state. All signals must be indicated in a correct order and need to take into account sufficient blocking times for the intersection to be cleared. A sample of 75 min for a comparable day and time window is indicated to be sufficient to make a prediction model (Bodenheimer et al., 2014). A problem arises with the Markov model when detector loop data is incorporated for vehicle-actuated controllers. The detector loop data amplifies the amount of edges towards different states (nodes) enormously to cover all possible combinations of detector occupations. The amount of nodes and edges in the graph explode when the intersection contains a lot of lanes, multiple modalities and control settings for different segments of the day. Eventually this causes computational inefficiencies leading to latencies on data transmission. Additional in complex situations the Markov-chain needs relatively a large amount of data to retrieve it transition graph and transition probabilities.

4.3.6 Previous prediction approach of Nissan: Artificial Neural Network

Nissan experimented with Artificial Neural Networks (ANN) to provide the vehicle with predictions of future traffic signal timings. The ANN was fitted to a vehicle-actuated controlled intersection in North-Holland. The intersection did not include other modalities besides passenger's cars or trucks. The fitted ANN model by Nissan showed promising prediction accuracies, but it should be taken into account that the ANN model was fitted to an oversimplified traffic scenario holding strong assumptions about the dynamics at an intersection. For example, the model was constructed and validated on only one signal group and with a small amount of data in terms of size and number of features. The principle of neural networks is to extract linear combinations of the input as derived features, and then model the output as a nonlinear function of these features (Hastie et al., 2009). Tasks that neural networks can perform include pattern classification, clustering or categorization, function approximation and predictions. ANN's is a statistical learning model that tries to copy the behavior of biological nervous systems by providing a mathematical model of numerous neurons connected in a network. No rules or programmed information about the dynamics of the system need to be specified beforehand (Mitra & Acharya, 2003). ANN has been applied successfully in the prediction of traffic information such as speed, flow and travel time (Murphey et al., 2013).

The ANN fitted by Nissan was trained to predict the *end time of the green phase*. By predicting the end of the green phase the switching time is calculated. It is assumed that the current phase is a green signal, such that is does not need to predict the start of the green signal. The ANN is therefore able to predict the duration of the green time and only the switching moment from *green to red*. The most time consuming part of the ANN is the feature/variables selection. Once too many features are selected curse of dimensionality can become a problem. The curse of dimensionality ties into the fact that non-parametric approaches such as the ANN often performs poorly when the amount of features is large (James et al., 2014). One might think that as the number of features used to fit a model increases, the quality of the fitted model will increase as well. In general, adding additional features that are truly associated with the response will improve the fitted model. Theoretically this could lead to a reduction in test set error. However, adding noise features that are not truly associated with the response will lead to deterioration in the fitted model. As a result, an increase in the test set error. This is because noise features increase the dimensionality of the problem that increases the risk of over fitting.

The ANN developed by Nissan was able to predict the green time duration by using detector loop data extracted from the V-log data. The total accuracy of an error of 5 seconds of deviation of the true green duration gave an accuracy of 97%. In a high traffic volume the performance was an overall accuracy of 98% with the same 5 seconds error. It was not tested how well the ANN performs when taking other modalities such as busses, pedestrians and bicycles into account.

4.4 Benchmark and selection of prediction framework

Appendix A contains a benchmark table of the previous six approaches. The six prediction approaches are compared according the following functional requirements from Chapter 2:

1. **Prediction target:** Switching time versus probability statement
2. **Prediction horizon:** Ability to predict the prediction horizon
3. **Data Usage:** Dealing with non-linearity of the controller
4. **Scalability**: Max degree of control flexibility
5. **Data Usage**: Utilization of V-log data
6. **Scalability**: Scalability to other intersections and modalities

The Support Vector Regression and Support Vector Machine are merged into one approach since the mathematical methodologies show a lot of comparisons. The table in appendix A shows that the methods that include probability theory and the Kalman Filter are not suited for the prediction of vehicle-actuated controllers. The prediction application needs an absolute second within the cycle time that the traffic light signal changes. Both the Kalman Filter and Probability theory cannot directly provide this information. The Markov-Chain approach also makes probability statements, however the probability in the Markov-Chain approach indicates a probability of transition to another state instead of a switching time. The transition graph of the Markov-chain approach is able to predict the order and the time offset of the transitions and therefore is able to predict the switching time. The biggest pitfall of the Markov-chain and therefore not useful for the prediction application is: "*as edges in the transition graph have to cover all possible combinations of detector occupations, this may cause a problem in computational efficiency possibly introducing latencies. Furthermore, this can become a difficult requirement for the database storage and the amount of sample needed to run a proper diagnosis*'' (Bodenheimer et al., 2014).

4.4.1 Support Vector Machine versus Support Vector Regression

The Support Vector Regression (SVR) is a version of the Support Vector theory that fits the nature of the prediction application better compared to the Support Vector Machine (SVM). The SVM uses as classification approach while the SVR uses a regression approach. By using a classification trick to define every second in the cycle as a class, the switching time can be determined at the exact second by the SVM. However the study that was conducted by predicting the switching time by using SVM, used strong assumptions about a fixed signal cycle lengths within the vehicle-actuated controller. This means that the expectation of the switching time of phases always was within a range of a limited amount of seconds. Each second is treated as a class, leading to relatively few classes for which the SVM made a distinction. As described in Chapter 3, the vehicle-actuated controller shows variability in both phase lengths and signal cycle lengths. Because of this dual variability the SVM will end up with a big set of classes (seconds in a cycle). The problem that the SVM might have at that moment is that it cannot make a clear distinction between two classes. In contrast the SVR uses the mathematically approach of the SVM but is able to give a numerical output instead of a class prediction.

4.4.2 Support Vector Regression versus Artificial Neural Network

A significant advantage of SVR over ANN is that the solution to an SVR is global and unique, while ANNs can suffer from multiple local minima. Some say that the SVR is easier to use compared to the ANN (Hsu et al., 2003). Others argue that the non-linear kernel trick within the SVR results into a lack of transparency when results are interpreted (van Belle & Lisboa, 2013). At the same time interpretation of results is one of the main drawbacks of the ANN due to its mathematical complexity (James et al., 2014). Unlike ANNs, the computational complexity of SVRs does not directly depend on the dimensionality of the input space. The reason that SVRs often outperform ANNs in practice is that they deal with the biggest problem with ANNs: SVRs are less prone to over fitting (Hastie et al., 2009). The development of ANNs follow a heuristic path; extensive experimentation preceding theory. The development of SVRs involves theory first then implementation and experiments. As is shown in chapter 3, theory about the dynamics of the controller can provide predictive power. For example, the mechanism of requesting the green signal and assignment of the phases by the block structure is logged in the V-log. If no knowledge about the dynamics of the system was present, ANN would make more sense to be used. An important part of the process of training an ANN is the feature selection. Unlike the ANN the SVM approach does not attempt to control model complexity by keeping the number of features small. Theoretically this makes the ANN less useable for extending the model with new data sources. The SVR approach is at this point more suited as a core algorithm in the prediction framework. The reason is that the SVR is mathematically a slightly lower risk to this project and is more likely to utilize knowledge that is gained from knowing the control logic.

4.5 Summary of Chapter 4

In this Chapter the following sub-question is answered: *What type of prediction approach complies to the functional and non-functional requirements for a prediction algorithm to predict the switching timings of a vehicle-actuated controller?*

A benchmark of several mathematical approaches, that were used in previous studies, show that the Support Vector Regression (SVR) approach has the biggest potential in predicting the switching timings of the vehicle-actuated controller. The benchmark includes criteria that are related to the functional requirements from Chapter 2. The biggest advantage of the SVR approach is its fast computation time, scalability towards other intersections and the use of non-linear input data. The SVR technique matches the condition to produce a numerical output that equals the switching time in absolute seconds. The SVR enables the model developer to choice a kernel that is unique to the structure of V-log data. By doing so the SVR deals with the non-linear characteristics of the data.

Furthermore, the Support Vector Regression approach isn't influenced directly by the dimensionality of the input space and is therefore less prone to overfitting. We easily can propose a relative large dimension of input space without harming instantly the prediction performance. As a results, the SVR approach can be extended with additional sources of data that might be needed to predict the switching timings.

To answer the sub-question, the following two sub-sub-questions are answered:

- *What prediction approaches are mentioned in the literature in the field of traffic signal timings prediction?*

A literature review resulted in five prediction approaches that were used in previous research to predict traffic signal timings. The five prediction approaches are: Support Vector Machine, Support Vector Regression, Kalman Filter, Probability theory and Markov-chains. The prediction approaches used in previous studies predicted the switching timings under different conditions such as different prediction targets, degree of controller flexibility and data usage.

- *What approach is currently used by Nissan for the prediction of traffic signal timings and why would a different approach be preferred?*

The current prediction approach by Nissan includes the use of Artificial Neural Networks. Empirical model results of Nissan show that the ANN approach is capable of predicting the switching timings of a vehicle-actuated controller. However, the ANN is less useable for extending the model with new data sources and features. V-log contains a relative high amount of features, but the ANN wants to keep the amount of used features as small as possible due to overfitting. The SVR approach is mathematically a slightly lower risk to this project and is more likely to utilize knowledge that is gained from knowing the control logic.

5 The Support Vector Regression approach

In this Chapter is further elaborated on the Support Vector Regression. A mathematical description will be provided and also the computational side of the Support Vector Regression approach is explained. This Chapter concerning the SVR approach supplies the reader with sufficient information to understand preceding modeling Chapters. For a more profound understanding of the SVR is referred to the publications *A Tutorial on Support Vector Regression* by Smola and Schölkopf and *Support Vector Machines in R* by Karatzoglou and Meyer. For a more general understanding of the Support Vector theory approach is referred to *The Elements of Statistical Learning* by Hastie, Tibshirani and Friedman.

5.1 Historic background

The Support Vector algorithm is a statistical learning theory, which has been developed over the last three decades by Vapnik and Chervonenkis (Smola & Schölkopf, 2003). In a nutshell, the Support Vector theory enables the model to generalize trained or learned behavior to unseen data. The Support Vector Machine, the most well-known variant of the Support Vector theory, was developed at AT&T Bell Laboratories by Vapnik and his co-workers. Its initial use was focused on OCR (optical character recognition) and object recognition tasks. Today the Support Vector Machine is considered to be one of the standard tools within the statistical learning paradigm (Hastie et al., 2009). The Support Vector Machine is from nature a classification model that predicts data points to a predefined class. The SVM is not suited for the prediction task in this project because a numerical output is needed. In contrast to the SVM, the SVR does provide a numerical output. In the following sections is explained how Support Vector theory can be applied for the prediction problem as described in this study.

5.2 Mathematical formulation

The basic idea of the Support Vector Regression (SVR) is derived from the classification method Support Vector Machine (SVM). Although the mathematics behind the SVR is different and considered to be more complicated compared to the SVM, the basic idea of the SVR is similar to the SVM. The goal of the SVR is to produce a model (based on the training data) which predicts the target values of the test data given a selected set of features from the test data (Smola & Schölkopf, 2003). The selected features need to catch as much prediction power as possible. In its simplified form the SV theory applies a simple linear method to the data but in a high-dimensional feature space. By performing a linear method in a high-dimensional feature space, the linear model is non-linearly related to the input space (Karatzoglou & Meyer, 2006). Although the theory considers a high-dimensional space, it does not involve any computations in that high-dimensional space.

The SVR model has two classes; the ε - Support Vector Regression (ε-SVR) and υ - Support Vector Regression (υ-SVR) . The difference between the two SVR classes is how regularization on the features is performed. υ-SVR controls the number of support vectors and training errors by one single parameter υ. ε-SVR uses two parameters to control the number of support vectors and training errors (Chang & Lin, 2002). Although some argue that the parameters in the ε-SVR are too arbitrary and difficult to explain most of the successful SVR models are ε-SVR models. In this research the ε - Support Vector Regression class is used as SVR model. In ε-SV regression our goal is to find a function $f(x)$ that has at most ε deviation from the actually obtained targets y_i. In this research y_i represents the length of the green phase (green time prediction model) or length of the red phase (red time prediction model). In other words, the SVR does not care about errors as long as they are less than ε, but it will not accept any deviation larger than this and estimates a regression function that is at the same time is flat as possible. (Karatzoglou & Meyer, 2006). To explain the SVR, it is preferable to describe the case of a simple linear function:

$$f(x) = \langle w, x \rangle + b \quad \text{with } w \in \chi, b \in \mathbb{R} \tag{1}$$

Where $\langle .,\rangle$ denotes the dot product in χ. Where χ denotes the space of the input patterns that is described by the vector w and the data features x. w is often referred to as the margin (Hastie et al., 2009). The variable b corresponds to the intercept. The property of maintaining flatness is achieved by minimizing towards a small w. To do so the norm of w is minimized i.e. $\|w\|^2 = \langle w, w \rangle$. This gives the optimization problem:

$$\begin{aligned} &minimize \quad \tfrac{1}{2}\|w\|^2 \\ &subject\ to \quad \begin{cases} y_i - \langle w, x_i \rangle - b & \leq \varepsilon \\ \langle w, x_i \rangle + b - y_i & \leq \varepsilon \end{cases} \end{aligned} \tag{2}$$

Where y_i denotes the target variable and x_i denotes a set of predictors. In this project the target variable is the switching time that is described as how many seconds are left before the current signal switches from red to green or from green to red. The set of predictors are for example features retrieved from loop detector data. The function in (2) assumes that all pairs of (x_i, y_i) can be approximated with ε precision. In other words the convex optimization problem is *feasible*, however this is usually not the case because data points are not always scattered symmetrical or evenly distributed in the feature space. In such a case it is hard to meet the requirement to minimize $\frac{1}{2}\|w\|^2$ with ε precision (Hastie et al., 2009). To achieve feasibility the SVR adopts the so-called soft margin principle in which slack variables ξ_i, ξ_i^* are introduced (Smola & Schölkopf, 2003). The slack variables cope with infeasible constraints of the optimization problem as described in (2). By doing so the minimization function has the form:

$$\begin{aligned} &minimize \quad \tfrac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}(\xi_i + \xi_i^*) \\ &subject\ to \begin{cases} y_i - \langle w, x_i \rangle - b & \leq \varepsilon + \xi_i \\ \langle w, x_i \rangle + b - y_i & \leq \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* & \geq 0 \end{cases} \end{aligned} \tag{3}$$

The constant $C > 0$ is a regularization parameter and describes the trade-off between the flatness of f and the amount of deviations larger than ε is tolerated. The optimal value for C can be estimated by cross-validation. Generally taken, a small C allows many data points to be close to the margin w (Hastie et al., 2009). This process is the so called ε- insensitive loss function $|\xi|_\varepsilon$ (Smola & Schölkopf, 2003) described by:

$$\left|\xi\right|_{\varepsilon} = \begin{cases} 0 & if \left|\xi\right| \leq \varepsilon \\ \left|\xi\right| - \varepsilon & otherwise \end{cases} \tag{4}$$

Figure 16 illustrates the mechanism of the insensitive loss function. The figure shows that only the points outside the grey region contribute to the cost. The insensitive loss function ignores errors of size less than ε (Hastie et al., 2009). Figure 16 illustrates that the deviations described by the slack variable are penalized in a linear fashion. For more information about the insensitive loss function and loss functions in general see Chapter 12 of the book *The Elements of Statistical Learning* (Hastie et al., 2009).

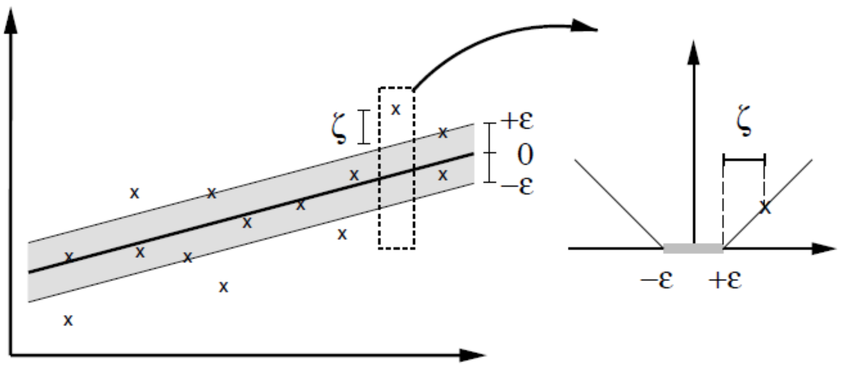


Figure 16 The soft margin loss for a SVR (Smola & Schölkopf, 2003)

5.2.1 Dual programming

In most cases the optimization problem (3) can be solved more easily optimized in its dual formulation (Smola & Schölkopf, 2003). The key idea is to construct a Lagrange function from the optimization problem (3) by introducing a dual set of variables. For details see (Van der Bei, 1994). The Lagrangian formulation of the optimization function (3) is:

$$L := \quad \tfrac{1}{2}\|w\|^2 + C\textstyle\sum_{i=1}^{l}\left(\xi_i + \xi_i^*\right) - \sum_{i=1}^{l}\left(\eta_i\xi_i + \eta_i^*\xi_i^*\right) \tag{5}$$

$$-\sum_{i=1}^{l}\alpha_i(\varepsilon + \xi_i - y_i + \langle w, x_i\rangle + b)$$

$$-\sum_{i=1}^{l}\alpha_i^*(\varepsilon + \xi_i^* + y_i - \langle w, x_i\rangle - b)$$

In (5) L denotes the Lagrangian and $\eta_i, \eta_i^*, \alpha_i, \alpha_i^*$ are the Lagrange multipliers. The Lagrange multipliers from (5) have to satisfy non negativity constraints, if not data points will vanish. For details see the book *Practical Methods of Optimization* (Fletcher, 2013). Data points that vanish do no longer influence the minimization of the margin. As a result, the SVR approach relies only on a subset of data points that are outside the tube space. Figure 16 illustrates that data points that influence the margin are either at the boundary of the tube or just outside the tube due to the slack variables. The data points at the boundary of the tube or just outside the tube and influenced by the slack variables are called the Support Vectors.

Substituting the partial derivatives of the Lagrangian with respect to the variables (w, b, ξ_i, ξ_i^*) yields the following dual optimization problem (Smola & Schölkopf, 2003):

$$maximize \begin{cases} -\frac{1}{2}\sum_{i,j=1}^{l}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\langle x_i, x_j\rangle \\ -\varepsilon \sum_{i=1}^{l}(\alpha_i - \alpha_i^*) + \sum_{i=1}^{l} y_i(\alpha_i - \alpha_i^*) \end{cases} \qquad (6)$$

$$subject\ to \sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0\ and\ \alpha_i, \alpha_i^* \in [0, C]$$

In (6) the notation $\langle x_i, x_j\rangle$ represents the kernel transformation of the input space. See section 5.3 for more information about the kernel transformation. The final solution function of the SVR can be rewritten as (Smola & Schölkopf, 2003):

$$f(x) = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)\langle x_i, x_j\rangle + b \qquad (7)$$

The function in (7) is the so-called *Support Vector expansion* (Smola & Schölkopf, 2003) and can be completely described as a linear combination of the trainings patterns in x_i.

5.2.2 Support Vectors

In the previous section is explained that the Lagrange multipliers from have to satisfy non negativity constraints. If the multipliers fail to do so the data points will vanish and will not contribute to the regression line. The data points that do no vanish are called Support Vectors. The Support Vectors are data points that lie at the boundary of the tube space. By introducing slack the SV can deviate from that boundary such that the SV is just outside the tube. The conclusion is that the complexity is independent of the dimensionality of the input space of the dataset and only depends on the number of SV's. By constructing the Support Vectors the Karush-Kuhn-Tucker (KKT) conditions are used (Smola & Schölkopf, 2003). Once a feasible solution is met the product between the dual variables and constraints will vanish and become zero. This is represented by:

$$\alpha_i\left(\varepsilon + \xi_i - y_i + \langle w, x_i\rangle + b\right) = 0\ \ and\ \alpha_i^*\left(\varepsilon + \xi_i^* + y_i - \langle w, x_i\rangle - b\right) = 0 \qquad (8)$$

$$\left(C - \alpha_i\right)\xi_i = 0\ and\ \left(C - \alpha_i^*\right)\xi_i^* = 0 \qquad (9)$$

Smola and Schölkopf draw the following conclusions from (8) and (9): firstly only samples (x_i, y_i) with corresponding $\alpha_i^{(*)} = C$ are located outside the ε-insenstive tube. Secondly the Lagrange multipliers always satisfy $\alpha_i * \alpha_i^* = 0$, meaning that there can never be a set of α_i, α_i^* that are both nonzero. From (8) it follows that **only** the Lagrange multipliers are allowed to become nonzero for $|f(x_i) - y_i| \geq \varepsilon$ (Smola & Schölkopf, 2003). How dominant the Lagrange multipliers are (α_i, α_i^*) in (8) is controlled by C (regularization parameter). For data points within the tube no regularization is needed and α_i, α_i^* will vanish. Eventually data points within the tube will not add any value to function (7) since the make sure the data points are multiplied by zero. The examples/data points that come with no vanishing coefficients are called Support Vectors.

5.3 Kernels

The Support Vector theory is well-known because of its 'kernel trick'. The kernel trick makes it possible for the SVR to deal with data that includes non-linear patterns. In Figure 17 a simplified illustration of the transformation of the data from input space to feature space due to a kernel is shown.

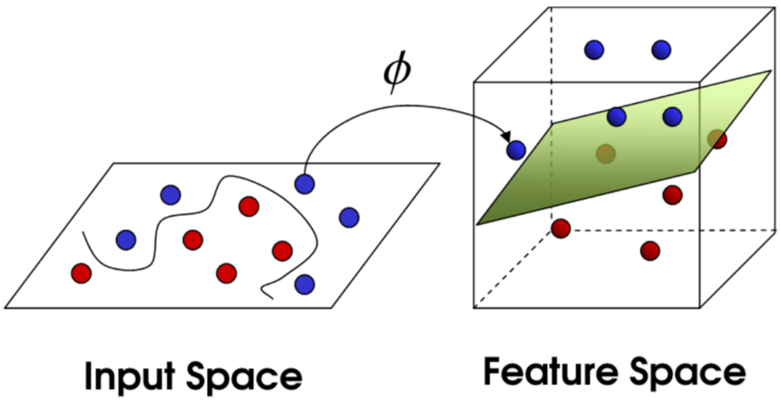


Figure 17 Mechanism of a kernel

The SVR is be able to deal with non-linear data by mapping the data x_i by a map $\Phi : \chi \to F$ into a feature space F after which the standard linear SV algorithm is applied in that feature space. However, the computation time of the SVR becomes easily infeasible when the amount of features increases (Smola & Schölkopf, 2003). To obtain a computation feasible transformation the 'kernel trick' is applied. The 'kernel trick' is that only the dot product in the feature space are calculated instead of the explicit mapping of Φ. It is sufficient to know $k(x, x') := \langle \Phi(\mathrm{x}), \Phi(\mathrm{x}') \rangle$ (Smola & Schölkopf, 2003). The dot products are injected into the SV optimization problem as described in the function of (6). In Figure 18 is illustrated how the kernel transforms the input space from a non-linear pattern to a linear pattern.

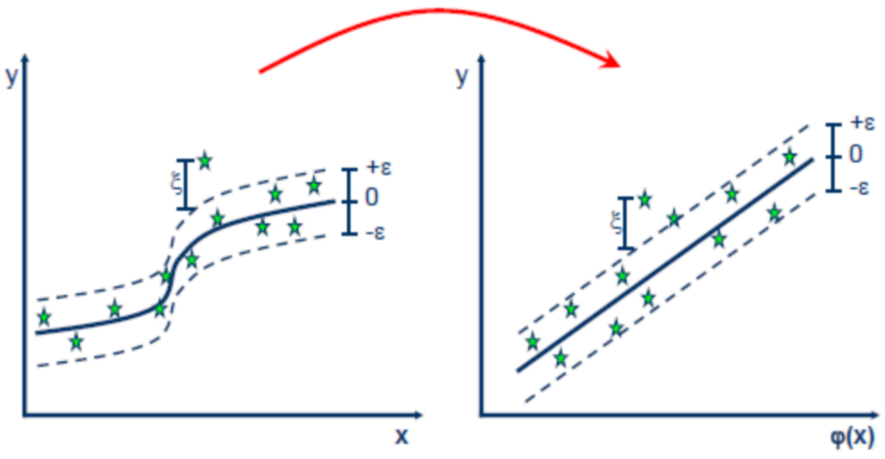


Figure 18 Kernel trick in combination with the SVR

The kernel $k(x_i, x_j)$ in the SVR optimization function is shown in the function of (12) below:

$$maximize \begin{cases} -\frac{1}{2}\sum_{i,j=1}^{l}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)k(x_i, x_j) \\ -\varepsilon \sum_{i=1}^{l}(\alpha_i - \alpha_i^*) + \sum_{i=1}^{l} y_i(\alpha_i - \alpha_i^*) \end{cases} \quad (12)$$

$$subject\ to \sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0\ and\ \alpha_i, \alpha_i^* \in [0, C]$$

As addition the expansion of (7) can be written as:

$$w = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)\, k(x_i, x_j) + b \quad (13)$$

The difference with the linear case is that w is not calculated explicitly and that finding a flat w is determined in the feature space and not in the input space. This makes the decision of a suited kernel of high importance since it directly determines the flatness of the SVR. A kernel can be constructed manually, but usually a kernel is selected from a predefined set of kernels. The reason is that the SVR practitioner needs to be skilled to understand what a possible relation could be between the input space and the computational gain in a high dimensional space feature space. Since it is unlikely to make a visual representation of this feature space defining a custom made kernel is complex. Below eight different kernels are given that are well known in combination with the SVR approach (Karatzoglou & Meyer, 2006):

1) The Linear kernel: the simplest of all kernel functions.

$$k(x, x') = \langle x, x' \rangle \quad (14)$$

2) The Gaussian Radial Basis Function (RBF) kernel.

$$k(x, x') = \exp\,(-\sigma\|x - x'\|^2) \quad (15)$$

3) The polynominal kernel.

$$k(x, x') = (scale\ \cdot\ \langle x, x' \rangle + offset)^{degree} \quad (16)$$

4) The Hyperbolic tangent kernel

$$k(x, x') = \tanh\,(scale\ \cdot\ \langle x, x' \rangle + offset) \quad (17)$$

5) The Bessel function kernel υ

$$k(x, x') = \frac{Bessel_{(\tau+1)}^{n}(\sigma\|x - x'\|)}{(\|x - x'\|)^{-n(\tau+1)}} \quad (18)$$

6) The Laplace Radial Basis Function.

$$k(x, x') = \exp\,(-\sigma\|x - x'\|) \quad (19)$$

7) The ANOVA radial basis kernel, this kernel is often used in a regression problem

$$k(x, x') = (\sum_{k=1}^{n} \exp\,(-\sigma(x^k - x'^k)^2))^d \quad (20)$$

8) Linear splines kernel, similar to the ANOVA kernel this kernel is used for regression problem.

$$k(x, x') = 1 + xx' \min(xx')^2 - \frac{(\min(x, x')^3)}{3} \tag{21}$$

In Chapter 8 a kernel will be selected by performing model selection. In the next section is explained what software is necessary to compute the SVR algorithm including the above mentioned kernels.

5.4 Computation

Karatzoglou et al. reviewed several software packages for the statistical programming language R that could be used for the computation of a SVR algorithm. They benchmarked 4 main R packages for using Support Vector (SV) functions:

- The **e1071 package** is based on the award winning libsvm package that is coded in C++ which makes the package relatively fast compared to other SV related packages in R (Karatzoglou & Meyer, 2006). This package has a limited amount of predefined kernels that can be used but offers a build in optimization framework performing a grid search over specified parameter ranges.
- The **kernlab package** is designed to have more flexibility with kernel functionalities. It offers a broader range of kernels and enables the user to switch between kernels on an existing algorithm and even create a custom kernel. Kernlab is mainly programmed in R, but uses the optimizers found in the python package libsvm which provides an efficient optimization in C++ (Karatzoglou & Meyer, 2006). An additional advantage is that the kernlab package gives access to almost any output statistics of the SVR. As a drawback it uses the non-conventional S4 class mechanism to store data.
- The **klaR package** is a simplified support vector package that is available for non-commercial use. This package is limited in kernel selection and output statistics. The klaR package was initially built for classification problems instead of regression problems.
- The **svmpath package** approaches the optimization of the regularization problem according a path method. The package contains the functionality to optimize SV problem for all values of the regularization parameter with a small fraction of the computational cost of fitting a single model. The model however contains a small limited number of kernels.

Karatzoglou et al. show that the packages e1071 and kernlab outperform the packages klaR and svmpath both in performance and computation time. The computation time of the packages klaR and svmpatch is almost double the computation time of e1071 and kernlab. Since the optimizer of e1071 and kernlab are both from the same source they show almost the same computation time and performance. For this study two criteria are important: 1) flexibility in kernel selection and 2) low computation time. Both criteria are related to the non-functional requirements from Chapter 2. The flexibility of a chosen kernel is connected to the non-linear characteristics of the V-log data (non-functional requirement). Not every kernel will perform similar and needs to be validated by model selection. The kernlab package offers the widest range of possible kernels. The criterion of a low computation by the selected SVR software is related to the non-functional requirement of maintaining a low computation time in Chapter 2. The conclusion is that the **kernlab package** favors both factors and therefore is used for the computation of the SVR model.

5.5 Summary Chapter 5

In this Chapter a deeper understanding is given about the SVR approach. First the mathematical formulation of the SVR is described. The output of the SVR is a real number and represents the switching timing of the current signal (red or green). The SVR consists of an optimization function that can be solved in its dual formulation. The SVR is able to deal with the non-linear characteristics of V-log by using a kernel transformation. The kernel expands the input space to a feature space in which a linear combination of the features is calculated. The 'kernel trick' involves the calculation of only the dot products in the feature space instead of the explicit mapping. The explicit mapping of the initial input space becomes infeasible once the amount of features grows. The kernel trick speeds up the computation time of the SVR. Therefore, the SVR approach and its kernel trick meets the non-functional requirement of maintaining a low computation time (see Chapter 2). The kernel should be selected according the characteristics of the V-log data. The selection of a kernel is performed by model selection. Model selection will be described in Chapter 8. In the final SVR model the best performing kernel is selected for predicting the traffic switching timings. Finally, in this Chapter the decision on the used SVR software is explained. The **software package kernlab** is considered to be the best fit for this project due to its flexibility of kernel selection and its relative high computational speed.

PART **II**

Data and Modeling

6 Selection of a vehicle-actuated intersection

As part of the case study an intersection in the region of North-Holland is selected. From this intersection V-log data is retrieved that is used in preceding chapters for the development of the prediction algorithm. In this Chapter an intersection is selected that matches the following functional requirements from Chapter 2:

1) **Scalability:** the prediction model is primarily designed to predict switching timings of the vehicle-actuated controller. It is likely to assume that the prediction model can also be used for predicting switching timings of semi-vehicle-actuated controllers. However, semi-vehicle-actuated controllers are not the focus of this research and are outside the scope of this project.
2) **Scalability:** the prediction model is primarily focused on intersections serving a single modality.

6.1 Selection criteria

The selection criteria for a particular intersection for this research are as following:

- The controller of the intersection has to follow the vehicle-actuated control logic as described in Chapter 3.
- The controller controls one modality (vehicles, no busses).
- Simple geometry; a typical cross geometry is preferred (see section 1.7).
- The intersection serves a traffic volume that shows variability during the day. Meaning that at least a morning peak should be visible according queue lengths and amount of reached maximum green times during the morning peak.
- Availability of V-log data for at least one month. In this study the experimental design is based on a minimum of at least one month.
- No major hardware defects concerning detector loops, since this will artificially influence the relationship between traffic and control output.

An extra criterion should have been selected (see Chapter 9 for more information). The extra criterion is related to the degree of isolation of the selected intersection. The extra criterion should have prevented the selection of an intersection that has a coordination with nearby intersections. The structure of the vehicle actuated controller is not important for the intersection selection. It is only of importance that the control scheme of the controller is vehicle actuated controller.

6.2 Location of the intersection

In consultation with the province of North-Holland the intersection marked in Figure 19 has been selected. This intersection does not violate any of the given selection criteria in the previous section. The intersection is called Laan van Darmstadt and is located between Alkmaar and Heerhugowaard. The intersection is located above Alkmaar and Amsterdam. It is expected that signal groups serving vehicles driving south on the intersection are considered to have more traffic volume in the morning peak due to traffic commute. The expected morning commute of the intersection satisfies the criteria for having variability in traffic volume during the day. Figure 19 shows the location of the intersection on Google maps.

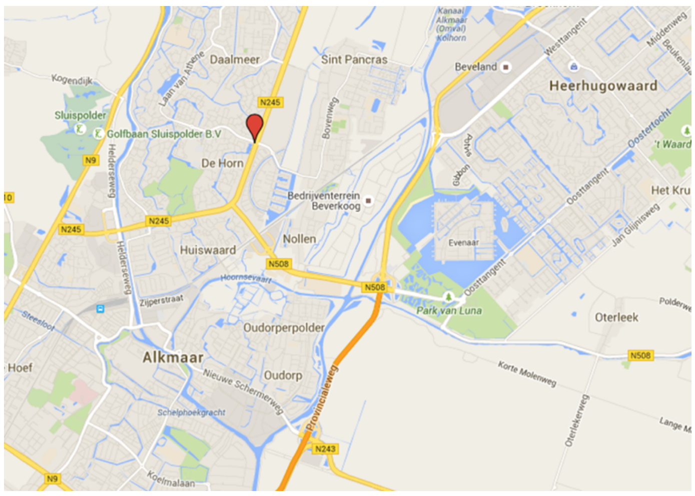
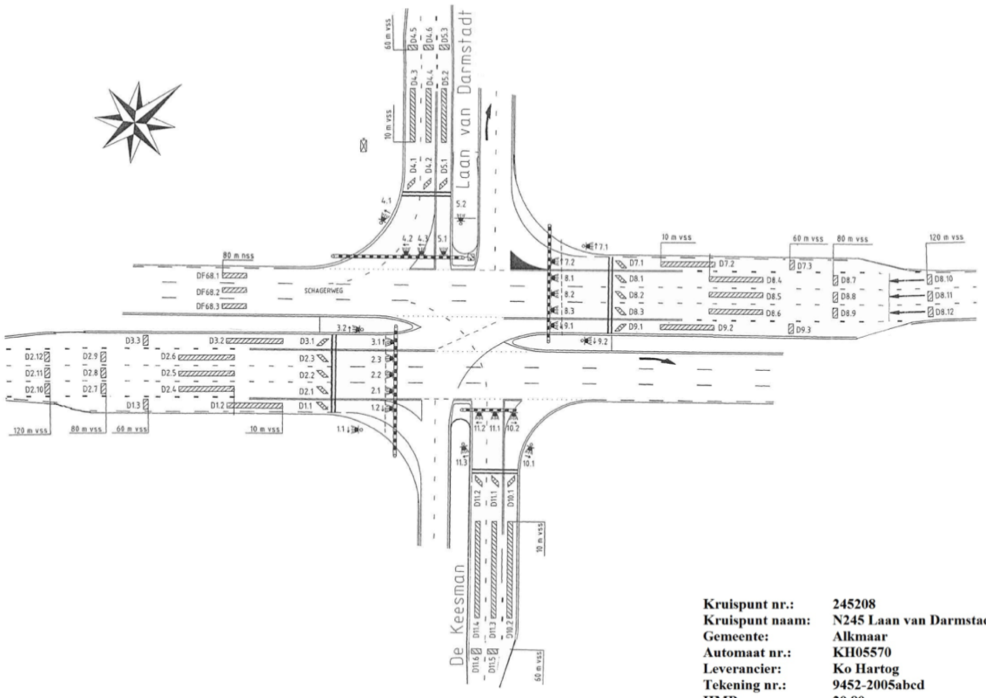


Figure 19 Location of the selected intersection

6.3 Inspection of the geometry of the intersection

Figure 20 shows the map of the intersection from which the geometry of the intersection is visible. Figure 20 shows that signal groups 1 and 3 might share double count issues with signal group 2 and signal groups 7 and 9 might have the same problem with signal group 8. The third detector loop of 2 is located more upstream and drivers may decide to switch lane just after they passed the third detector loop from lane 2 towards 1 or 3. At this point this particular vehicle is counted again by the third detector loop of 1 and 3. Experts from North Holland stated that this is an unlikely event but could happen. This would be considered as adding noise to the prediction model.



Kruispunt nr.:	245208
Kruispunt naam:	N245 Laan van Darmstadt
Gemeente:	Alkmaar
Automaat nr.:	KH05570
Leverancier:	Ko Hartog
Tekening nr.:	9452-2005abcd
HMP:	20,80

Figure 20 Map of the intersection Laan van Darmstadt

6.4 Labeling of the detector loops

In this research the prediction algorithm will target particular signal groups. These signal groups have a particular number as a label. Figure 21 shows the labeling of the signal groups for the intersection at Laan van Darmstadt. Figure 21 indicates how the detector loops are numbered according their dedicated approach. This information is important for the feature selection that will be described in Chapter 7.

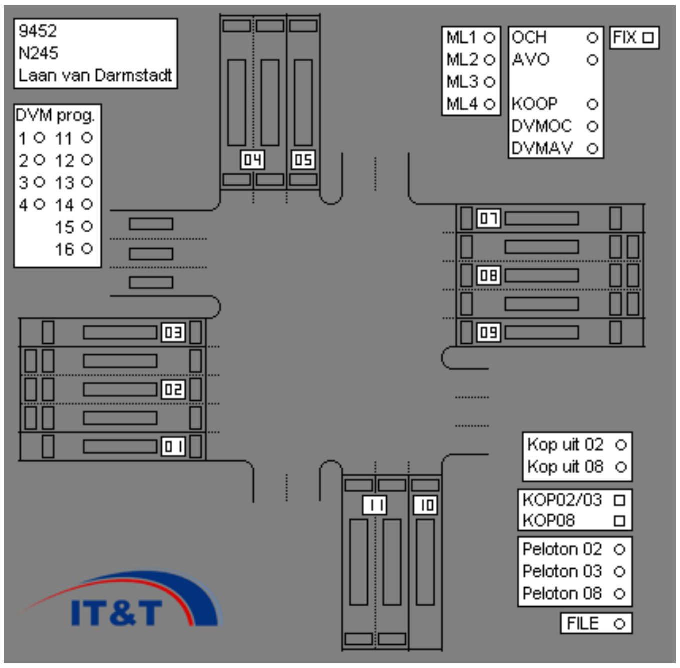


Figure 21 Labelling of the detector loops

6.5 Summary of Chapter 6

In this Chapter the intersection *Laan van Darmstadt* is selected for which V-log data is retrieved from the morning peak. The intersection is located between Alkmaar and Heerhugowaard in the province of North-Holland in the Netherlands.

7 Preprocessing of the data and feature selection

In this Chapter the preprocessing of the V-log data is described. The raw V-log data needs to be preprocessed in order to be used for a statistical model such as the Support Vector Regression (SVR) algorithm. As will be described in this Chapter, the process of transforming V-log data is intense and time consuming but is of utmost importance to achieve satisfying prediction results. Special attention is paid to troubleshooting of corrupted V-log data.

7.1 Data pipeline

The data pipeline is a virtual trajectory that the raw V-log data follows before it can be injected into the SVR model. The figure below gives an overview of the data pipeline that is custom made for the prediction of traffic signal timings and is therefore unique to this project. The data pipeline is considered to be part of the prediction algorithm. The SVR will not be able to predict the signal timings without following the steps described in the data pipeline.

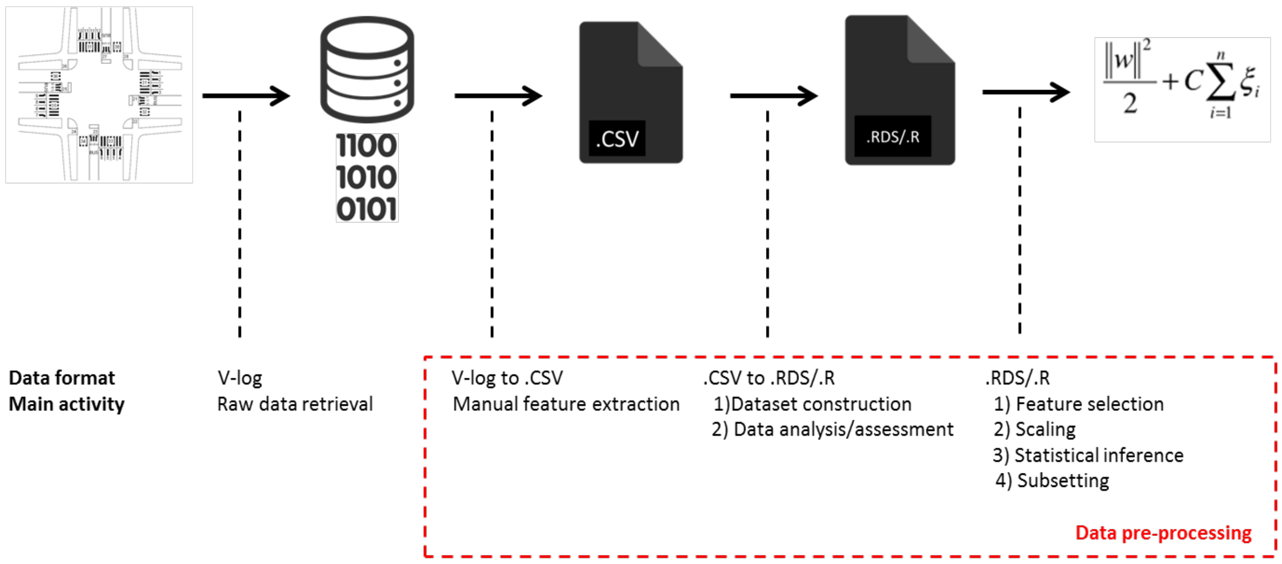


Figure 22 Data pipeline for preprocessing raw V-log to training data

Within the data pipeline the data changes three times from data format. The first data format: V-log contains mainly binary data and is transformed to a readable data format for the statistical software program R. The binary data describes the controller according a DEV's (discrete event simulation) analogy and needs to be transformed into meaningful features that show variance in their feature vector. Statistical learning methods are likely to train and predict poor on binary data and need variance in the data to discriminate learning patterns (Hastie et al., 2009). Examples of created features are countdowns, timers and cumulative vehicle counts. The process of formulating features is called manual feature extraction and is explained in the next section.

The second change in data format is from .CSV to .RDS/.R data format. The reason for this is twofold:

1) .RDS is a serialization data format that lets the programmer translate data structures or object states into a format that can be stored (for example in a file or memory buffer) and reconstructed later in the same computer environment (Wikipedia, 2015). In this case the environment is the global environment in the statistical software program R. When the resulting series of bits is reread according to the serialization format, it can be used to create a semantically identical clone of the original object. Serialization a convenient property of the .RDS data format, since this leads to a reduction in preprocessing time.

2) .RDS files have the property to load faster compared to .CSV documents (Matloff, 2011). The total dataset consists of at least 300 .CSV files adding up to over 50 gigabyte of data. Using .RDS data format instead of .CSV files will lead to a significant lower time to preprocess the data.

The end product at the right side of the data pipeline is data that is ready to enter the SVR model. At this point the data has the property that it does not create logical errors in the SVR model and reduces computation time as much as possible. The activities within the red demarcation in Figure 22 represent preprocessing of the data. Preprocessing in data mining/ data science prevents the principle of 'garbage in, garbage out'. Garbage in, garbage out means that the prediction model will unquestioningly process unintended input data ('garbage in') and produce undesired output ('garbage out').

7.2 Preprocessing

Preprocessing of the raw V-log predominantly covers the following actions in sequential order:

- manual feature extraction (section 7.3),
- dataset construction (section 7.4) ,
- quality assessment of the data (section 7.5),
- feature selection (section 7.6),
- scaling (section 7.7),
- subsetting into train- and test datasets (section 7.8).

Each preprocessing action is necessary to make the knowledge discovery and prediction power in the training phase of the model as big as possible. The final output of the data preprocessing is useable training and test data. In the sections below is described how the preprocessing is executed. Throughout the preprocessing procedure a lot of mistakes, confusing and undesired results can occur. This project had setbacks that mainly have their origin in preprocessing the data. Possible pitfalls and problems are highlighted in the sections below.

7.3 Manual feature extraction

Feature extraction starts from an initial set of measured data and builds derived features from this initial dataset (James et al., 2014). The reason for this is that the total initial dataset might contain redundant and non-informative data. In this project the raw data and initial data is V-log data. To make an efficient and effective model we would like to extract data that is intended to be informative and facilitates the learning and generalization steps within the algorithm. Feature extraction is related to dimensionality reduction because we try to reduce the dimensionality of the initial dataset to a smaller dataset without losing important information. The feature extraction process should not be confused with the feature selection process. In this study feature selection is related to model selection. In order to select the best performing features, the features need to be designed. The formulation and designing of features is called feature extraction. Feature extraction is traditionally done by using techniques such as *principal component analysis, factor analysis* or *singular value decomposition.* The reasons why these techniques were not used are:

- Classical feature extraction techniques are not directly useable on data following a discrete event analogy,
- If we were able to use classical feature extraction techniques, we would neglect the knowledge that we gained from knowing how the vehicle-actuated controller works. Since we know how the controllers work, we are likely to understand what features should be extracted in order to construct an informative dataset.
- If we would be able to use classical feature extraction techniques, we possibly would end up with features that have no form of interpretation. In that case troubleshooting relies only on pure statistical procedures.

The feature extraction is called 'manual' since it is coordinated by manually looking at what information the controller logs into the V-log and how this could be transformed into features. It is important to define features that capture the discrete event data as much as possible without losing information. We would like to maintain information in the data regarding how the controller reacts on the initial V-log data. It would not be beneficial to exclude detector loop data from the extraction process since this is important input data for the controller. **<u>It should be clear that the manual feature extraction procedure for V-log data is not a straightforward process that can be copied from a textbook.</u>** The feature extraction procedure depends highly on the type of data. The process of extracting features from V-log is not described in any prior research. Once the method of extraction is determined, it can be applied to V-log data from other intersections and does not necessarily depend on the parameter settings of the vehicle-actuated controller. The manual feature extraction relies on the knowledge that is gained from Chapter 3 and knowing what type of data works the best for a SVR algorithm, but this is not trivial. To illustrate this problem, we have to recall how V-log data is logged by North Holland:

> The V-log data shows event-based activities of the controller that are represented by binary data in the V-log. The V-log contains messages regarding detector loops, internal signal parameters, external signal parameters and speed detection. The V-log needs to be translated to a data layout in which every row indicates a time instant and every column represents data that serves as input or output of the controller for that particular time instant. The aggregation level of a time instant is $1/10^{th}$ of a second. This is because the V-log logs its data at this frequency. As a result, every logged time instant looks almost identical to preceding time instants given the data in the columns. When we look at a much bigger period of time, for example an hour of V-log data, we see that the overall differences between time instants are low. We would see more differences between time instants when more features are added to the feature space. But this 'difference' is numerically limited since it can only reveal its difference by either showing a zero or a one. At this point statistical learning algorithms such as the SVR will run into trouble. Statistical learning requires computing distances between data points or similarities between data points. Therefore it is beneficial to have a dataset that shows numerical variance in the data. If we

would use the raw V-log data directly we would end up with ones and zeros showing relatively low variation in similarities and distances between data points. For example: the initial V-log at a given time instant shows either a zero or a one on detected vehicles by a detector loop. By cumulating events, in this case incoming vehicles, more information for each time instance is created. The cumulative count shows a higher variance for the detector loop feature compared to the binary data in the V-log.

The manual feature extraction results into 112 features that describe the behavior of one signal group. Appendix B contains the full list of manual extracted features given for one signal group. In the table below a summarized version of the full list of extracted features is given.

Information type	Feature count	General description of features	Example of a feature
Indexing features	3*	Non-informative features, but necessary for filtering	Date: Date of the logged time instance
Loop detector data	37	Information about incoming vehicles. This includes features about current moment, vehicles during the current phase and previous phases up to 5 red and green phases back in time.	AtTdet3_MaxVol: Cumulative vehicle count at the third detector loop at current time
Controller output states	19	Information about the red and green phases. This category of features includes timers, countdowns and absolute lengths of red and green phases.	R_Length1: Absolute length in time of one red phase before current red phase
Internal signal states	55	Features that reveal information about the behavior of the two control levels within the vehicle-actuated controller: group completion and signal group completion.	SCycle_Time: Timer of the start to end of cycle of that particular signal group

Table 2 Overview categories manual feature extraction

*the indexing features are only applied once to a dataset, these are not unique features to a signal group.

Besides the *indexing features*, that are non-informative for the SVR algorithm, three main feature categories can be distinguished. In Table 2 these categories are briefly described. Below each feature category is further elaborated:

Loop detector data: loop detector data is extracted from the first and the third detector loop. The third detector loop is considered to be of high importance. The reason for this is that the controller extends it green for a big part based on detected vehicles on the third detector loop. Extension of the green signal is not only determined by the presence of a vehicle at the third detector loop, but also its measured gap time plays an active role in the extension of the green signal. Gap times indicate to a large extend the arrival pattern of vehicles. Gap times are currently not logged in V-log data. Besides the third detector, also the first detector loop data is of important. The first detector loop is a better indicator compared to the third detector loop for historical green and red phases. The first detector loop indicates the amount of vehicles that pass the actual green light. Historical loop detector data is extracted from the previous 5 cycles for a particular signal group. The assumption is that 5 consecutive cycles before the current cycle might contain valuable information about the upcoming green duration. In the research by Koukoumidis et al. it is shown that more than the 5 previous cycles do not significantly improve the prediction power (Koukoumidis et al., 2012).

Controller output signal states. This category of features describes information about the signal output of the controller. The control output for a signal group is either the assigned red or green duration in seconds. Data regarding the yellow signal is also retrieved, but not used for prediction purposes. Likewise, the detector loop features, the controller output is extracted from the current phase and the 5 previous cycles of that particular signal group. The current phase can be a red signal or a green signal. The features include timers, countdowns and absolute lengths of the phases. The timer gives the expired time at the current time instant for the running phase (green or red). The countdown is the inverse of the timer. The countdown features can only be extracted from historical data but will serve as prediction target within the SVR. Additional to the timers and countdown of the current signal output, the absolute duration or lengths of the previous 5 green and red phases are extracted. The absolute length of the previous 5 green and red phases is not a timer but a constant value.

Internal signal states. The internal signal state features are designed to describe as closely as possible the behavior of the two control levels within the vehicle-actuated controller. The two control levels are the *group completion* and *signal group completion*. Briefly stated, the group completion level assigns a green light to a signal group and the signal group completion level determines the duration of the green light. An example of a feature is the timer of the signal cycle. This timer is the elapsed time of the start until the end of the cycle of that particular signal group. Another example is what realization method is being used when the green light is running: primary or alternative realization. For each realization method a timer, countdown and absolute length is extracted that is applicable for the given time instant.

7.4 Construction of the dataset

In the previous section is explained that for each signal group 112 different features are extracted. The selected intersection Laan van Darmstadt (see Chapter 6) contains 10 signal groups. The dataset that results from the feature extraction contains 1120 features. An efficient construction of the datasets is required to avoid infeasible computation time and chaotic dataset structures. As will turn out, the size of the dataset reaches into millions of data points. Data of one day coming from one signal group is logged in one single .CSV file. The matrix below gives a representation of the layout of such a single CSV file:

$$\begin{bmatrix} T_1\_Feature_1 & \cdots & T_1\_Feature_{112} \\ \vdots & \ddots & \vdots \\ T_{72000}\_Feature_1 & \cdots & T_{72000}\_Feature_{112} \end{bmatrix}$$

The time step between each row is 1/10th of a second. The original data contains a 24 hour span. Within this study we only investigate the morning peak. The morning peak indicates the hours between 7:00 AM and 9:00 AM. In total the dataset of a morning peak consists of 72000 rows. Each row indicating a time instant of 1/10th of a second. For each day all the independent signal group .CSV-files of one intersection were binded columnwise. After all the signal groups are column-binded together of that particular day, all available days are row-binded. The matrix below gives an abstract representation of the final raw dataset after which all preprocessing techniques are executed.

$$\begin{bmatrix} [\mathbf{Day1}\_Feature_{1\ldots112}\_\mathbf{SignalGroup1}] & \cdots & [Day1\_Feature_{1\ldots112}\_\mathbf{SignalGroup11}] \\ \vdots & \ddots & \vdots \\ [\mathbf{Day27}\_Feature_{1\ldots112}\_SignalGroup1] & \cdots & [Day27\_Feature_{1\ldots112}\_SignalGroup11] \end{bmatrix}$$

The reason for the chosen data layout is to safe computation time while preprocessing the data. The statistical software program R suffers from high computation time when for loops or look up functions are used. Since the size of the data is big, an efficient approach is desired. It is preferable to use vectorized functions as much as possible (Matloff, 2011). By maintaining the described dataset layout, filter functions can be vectorized. Vectorizing functions significantly speeds up the computation time. The next step in the preprocessing part is to assess the quality of the dataset and filter out segments of the dataset that guarantee a desired data quality.

Specifications of the constructed dataset:

- The dimension of the initial dataset is 216030 by 1122 accounting for over 240 million data points.
- The data is retrieved from intersection with id number: 245208, Laan van Darmstadt
- The data covers the month April 2014 (30 days). No data failures were reported in the V-log.
- The filtered dataset covers the morning peak from 7:00 AM until 9:00 AM

7.5 Quality assessment of the retrieved data

In this section is described how quality of the data V-log data can be checked and what implications it can have when the data is not clean. Securing a high quality of data is especially important for data that is used for training the SVR models. It turns out that assessing the quality of V-log is not an easy task and, if not performed properly, can lead to serious consequences for the prediction accuracy. Assessing data directly means that baseline data exists from which (statistical) properties can be quantified as 'good' or 'satisfying'. The definition and assessment of having 'good' V-log data becomes challenging when such baseline data is not available and can only be derived from that same available dataset that we would like to assess. This is the exact circumstance in which the quality assessment is performed in this study. Additional standard statistical procedures such as outlier detection fail when they are used directly on the dataset. The reason why this happens will be explained in this section.

7.5.1 Encountered issues: detector loops causing corrupted data

Throughout this study a major setback was encountered due to corruption in the dataset. This resulted into trained models that were giving a dissatisfying output. Predominantly the output showed a high prediction error. But alarming was that the trained models showed unexpected behavior according the theory of how the controller works. Being more precise, in some regions of the dataset low correlations were found between counted vehicles by the first detector loop and the amount of assigned green time. Accidentally we found out that the third detector loop was jittering and therefore artificially was extending the green time, while the cumulative vehicle count on the first detector loop was relatively low. This pattern also was detected by looking at deviant patterns between the correlation of both the cumulative count of the two detector loops and the correlation between either the cumulative vehicle count of the first or third detector loop and the given green time. Both the correlations should be relatively high in order not to raise suspicion. With help of the province of North-Holland it was discovered that one of the third detector loops showed hardware problems by giving frequently a high signal without detecting a vehicle (jittering loop detector). As a result, the cycle time of the entire intersection was artificially extended by this single detector loop affecting output signals of other signal groups. The threshold setting of 'sufficient' correlation is arbitrary. For the assessment of the final version of the dataset we used the correlation between the first detector loop and the given green time. The minimum correlation threshold was set to be 0.80. No hard facts support the 0.80 threshold. Important to mention is that hardware or software failures do not necessarily always lead to missing data. In the case of missing values, the hardware failure is easy to detect. In the case that a hardware or software failure leads to inconsistency of the data a quality assessment framework is needed.

7.5.2 Four step procedure to assess V-log quality

Identifying low correlations in the previous example, that address a weak relationship between traffic and given green time, was step one in determining a low data quality. The hardest part is defining metrics and clear boundaries on how to classify poor quality in the data. In other words, how should poorness of relationships be quantified between features and the prediction target? A second important question is how representative and consistent is the dataset to make statements about generalization of the prediction model? Since the dataset has a huge dimensionality it is not feasible to check the data entry by entry and line by line, so we have to rely on statistical metrics that summarize the data to a certain degree. To our knowledge, these questions were not answered by any previous research study in the field of traffic signal timing prediction. However, the encountered setbacks due to data corruption showed that assessment of the dataset is of utmost importance. To assure a desired quality standard of the data the following 4 step approach was used:

- Definition of key features as quality indicators.
- Segmentation of the dataset to avoid non-informative variability.
- Definition of 'standard' patterns in the data that can be considered to be normal behavior.
- Perform outlier detection on segmented dataset respecting patterns from 3.

Below each step is further elaborated on.

Performance metrics on key features were defined that served as indicators of the quality of the data. The knowledge that is gained from Chapter 3 (vehicle-actuated controller) serves as input for defining key features. The used statistical metrics on the particular features are:

Statistical metric	Performed on features	Indexing feature names
Correlation	Cumulative count of vehicles on the first detector loop and the duration of the green light and between the cumulative count for the first and third detector loop.	GDet1_MaxVol, G_Length, GDet3_MaxVol
Arithmetic Mean	Cumulative vehicle count on the first detector loop	GDet1_MaxVol
Maximum	Cumulative vehicle count on the first detector loop	GDet1_MaxVol
Standard Deviation	Cumulative vehicle count on the first detector loop	GDet1_MaxVol
Arithmetic Mean	The realized green time	G_CD
Maximum	The realized green time	G_CD
Standard Deviation	The realized green time	G_CD
Maximum Number	Number of green phases	-
Arithmetic Mean	time from start of a random block until the next start of that same block for a dominant signal group	Block2Block_Length

Segmentation of the dataset is performed to detect unusual patterns in the dataset. The problem is that the initial extracted variance of the data is pooled over the whole dataset. This means for example that weekdays and weekend days are pooled together in the same assessment metrics such as correlation, arithmetic mean and standard deviation. But by theory we know that weekdays and weekend days have different control settings. So by definition the controller output that is shown in the extracted data shows different statistical properties. The pulled variance over the whole dataset without segments will mask deviant behavior since it will be averaged over different segments. The chosen segments will be used as a basis for the next step in the assessment procedure: defining standard patterns. The following decisions regarding segmentation were made:

1) A distinction is made between morning, afternoon, evening and night. In this study only the morning peak (7AM – 9AM) is investigated. Other time periods of the days will have different standard patterns. This probably is traced back to different traffic demands and therefore different controller input/output. Similar behavior can possibly be detected on holidays versus normal weekdays.
2) An aggregation level of one day (one morning peak) is taken as data segment. If the segmentation is to loose and the segments are too big, such as complete weeks, unusual behavior might not be detected. If we would detect strange behavior, we should delete or correct data for the whole week. This is not efficient since we would like to exclude only the days that caused deviant behavior.
3) The controller settings are different for weekdays versus weekend days, so we are likely to find different standard patterns between weekdays versus weekend days.

Defining standard patterns. To specify standard patterns, we investigate the key features within their segmentation. We pull statistics for each of those key features for individual days of the week, describing only the morning peak with special notification for weekdays versus weekend days. We end up with a table as shown in appendix C. At this stage we need threshold that state what is considered to be 'normal' or 'standard' control behavior and what is considered to be 'deviant' control behavior. This is challenging because every intersection and it accompanying traffic demand is different. As a result it is difficult to set general hard thresholds to classify normal and abnormal behavior. For example what is considered to be a sufficient correlation between traffic volume and realized green time? Or what is a normal/standard number of realized green phases in a morning peak? It seems that hard threshold can become arbitrary and open for discussion. To keep it simple and straightforward a four threshold were used:

1) A correlation of at least 0.80 on weekdays is needed between traffic demand and the amount of given green in seconds. The correlation is averaged over all cycles during the morning for weekdays.
2) Negative min and max values of traffic volume indicate corrupted data. The min and max value are pooled from a single cycle and is not averaged over the whole morning peak.
3) The mean traffic volume should be at least 8 vehicles. The mean traffic is averaged over all cycles during the morning peak for weekdays.
4) Average block to block time should be at least 90 seconds. This average is pooled over every cycle during the morning peak for weekdays.

We only used 4 metrics while we defined 9 metrics. The remainder 5 metrics were used in case the above 4 metrics showed strange behavior to get more explanatory insights into the data corruption.

Deleting outliers. At this point we defined every element that is needed to assess whether a complete day should be deleted from the dataset. As the table shows in appendix C two days are excluded from the dataset: Monday 7 April 2014 and Thursday 17 April 2014 (day before Easter holiday). These two days show correlation below 0.80 between traffic volume and realized green time. The relatively low correlation does not necessarily indicate corruption in the data, but could indicate that the control behavior does not show normal or standard behavior. The weekend days were excluded by default since different control settings are used and also deviant traffic volume is expected. The difference between weekdays and weekend days is clearly visible while investigating the results on the performance metrics in appendix C.

Point of Discussion:

To test the quality of the V-log data, data coming from the dominant signal group was analyzed. For the selected intersection; the Laan van Darmstadt the dominant signal group is signal group 8 (see Chapter 3). Signal group 8 is expected to have the highest traffic volume during the morning peak and therefore will influence the controller input/output the most. Deviating behavior in this signal group probably leads to deviating behavior for the total intersection. The most preferred situation would be to have metrics that could assess the total intersection for each segment. This however is quite a time consuming task and should be included for further research.

7.6 Feature selection

Feature selection is an important requirement for a regression-type model when the amount of features is large. The Support Vector Regression model does not perform feature selection automatically, because it uses quadratic regularization (Smola & Schölkopf, 2003). The Support Vector theory has rather a remarkable effect on feature selection; it is sometimes not efficient to reduce the number of features. Research shows that accuracy on a test set (not training data) starts to degrade when the amount of features is reduced. Until today there is no hard explanation for this behavior (Hastie et al., 2009). Different approaches can be used to select the optimal set of features. An example of such an approach is a greedy search algorithm that adds features to the model according their explanatory variance. The risk of such a greedy algorithm is that the optimal model is likely to be missed. The strategy in this study is to predefine several sets of features with a different rationale and test these sets of features on their performance. This method can be considered as a *Best-Subset Selection* method (Hastie et al., 2009). Initially such a method will try to fit all possible combinations of features. This leads to high computation times and should be avoided. In this case the feature selection is part of the model selection. Model selection is described in the next Chapter in more detail. A more statistical complex feature selection procedure could be used, but we would like to use our knowledge about the traffic light controller as much as possible. Also a more statistical feature selection procedure does not give any guarantee for a better performance. We are likely to risk a lot of computation time by using complex feature selection procedures. Six different rationales were used for the design of featuresets. Eventually the six rationales lead to six featuresets for each signal group. During the model selection in the next Chapter the best performing featureset is chosen. The design of a featureset has a significant amount of impact on the prediction accuracy and is therefore at the center of the model selection procedure. The next section describe each featureset and rationale in more detail.

7.6.1 Six different featuresets

Each featureset is labelled with a number starting from 1 up to 6. The major difference between the first three featuresets and the last three featuresets is the number of signal groups that are involved from which features are selected for each set. Figure 23 illustrates the major difference between the first three featuresets and the last three featuresets. Featuresets 1, 2 and 3 are narrow-minded since these featuresets only contain features/data that are extracted from the same signal group for which we would like to predict the switching timings. In contrast, featureset 3, 4, and 6 have a more holistic featureset approach. These featuresets include features/data that are extracted from multiple signal groups and therefore should be able to catch more information about the dynamics of the controller. After the manual feature extraction we ended up with 112 features for each signal group. Appendix A shows the complete list of extracted features for one signal group. At an abstract level the 112 features for one signal group can be categorized into four categories: indexing features, loop detector features, controller output states features and internal signal states features. The indexing features are non-informative, but are added to every featureset for filtering purposes.

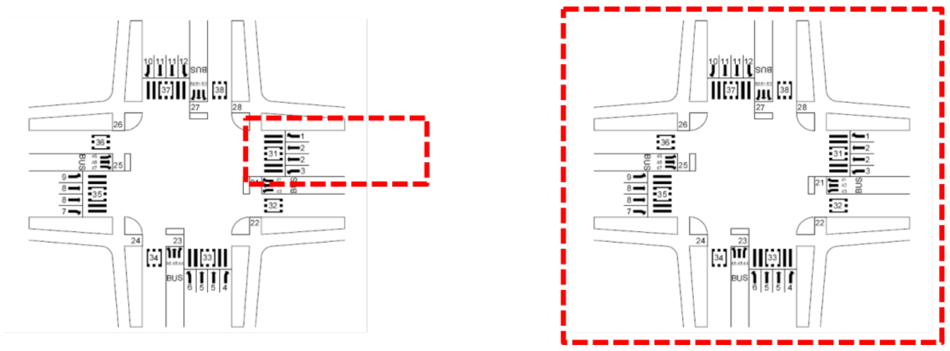


Figure 23 Difference between a narrow minded featureset and a more holistic featureset

Featureset 1 contains the basic controller output features such as green and red durations of the previous 5 cycles. Additionally, loop detector data is included in featureset 1. **Featureset 2** is the same set of features as featureset 1 but adds internal signal state features. From Chapter 3 we derive that the internal signal states are determined at the second control level: the signal completion level. Featureset 3 takes all features from featureset 2, but performs a selection on this subset of features by looking at the strength of the relationship between the target variable and each feature. The strength of the relationship is measured with the Pearson's correlation metric. The reasoning behind **featureset 3** is to create a parsimonious model that includes the most relevant features. Ideally we would also check for confounding factors, but this is rather a complex mechanism in data such as retrieved from a controller. By definition features are confounding since they should depend on each other. We set a threshold of 9 features that featureset 3 can select from the features in featureset 2. If we would allow a higher number of features the model loses its level of parsimony and approaches featureset 2 in content of features. Additionally, we often saw empirically that correlations for featureset 3 started to deteriorate after the ninth feature. **Featureset 4** follows the same rationale as featureset 1, but includes the basic controller output features and loop detector features from all signal groups into one set. For the investigated intersection this means that we are taking 10 signal groups into account in featureset 4 and only 1 in feature 1. **Featureset 5** follows the same rationale as featureset 2, but similar to featureset 4 takes all signal groups into account. Featureset 5 contains basic controller output features, loop detector features and internal signal state features from all signal groups. **Featureset 6** shows similarities with featureset 3. Featureset 6 takes features from features 5 as initial set and performs a feature selection based on Pearson's correlation. Unlike featureset 3 we set a limit of 20 selected features. If we would select more than 20 features the featureset set loses it level of parsimony. Similar to featureset 3 we saw that correlations began to deteriorate after the twentieth feature.

7.7 Scaling of the data

Scaling the data before applying the SVR is important and should be done carefully. Scaling in statistical learning is important to avoid features in greater numeric ranges to dominate features in smaller numeric ranges. Especially in regression problems, features in greater numeric ranges will artificially add weights to their features when scaling is not applied for these features (Hsu et al., 2010). Another advantage of scaling is to avoid numerical complexity. The kernel trick as explained in Chapter 5 uses the dot products (inner products) of feature vectors. For some kernels such as the linear and polynomial kernel, in which a relative large feature space is created, scaling reduces the computation time drastically. It is preferable to scale the data to a zero-one numerical scale. Standard feature scaling is usually executed by unity-based normalization. This is described as $X' = \frac{X - X_{min}}{X_{max} - X_{min}}$. We found out by using different scaling approaches on the V-log data that unity-based normalization produces problems. Large green and red, that account for a small percentage of the dataset, pull the large part of the dataset close to zero after scaling is applied. This makes the data non-informative. To account for this problem the standard score of the feature vector is used for scaling: $X' = \frac{X - \mu}{\sigma}$. The standard score uses the mean and standard deviation of a population. This sets a restriction on how scaling is applied on the total dataset versus a small subset. In this study is decided to scale the total dataset based on the mean and standard deviation of the feature vector of the total dataset. In this way the feature vector of the total dataset is referred to as the population. After scaling is applied the dataset is subsetted to smaller datasets. The reason for this order of scaling and subsetting is to maintain the same averaged behavior as reference point within each subsetted dataset (train and test data). If scaling is applied on every subset of data, extreme behavior can possibly be smoothed away by the scaling procedure.

7.8 Subsetting data into train data and test data

During the formulation of the prediction model in the next Chapter we have two separate goals:

1) **Model selection**: estimating the performance of different models in order to choose the best one.
2) **Model assessment/validation**: having chosen a final model, estimating its prediction error (generalization error) on new data.

If we are in a data-rich situation, the best approach for both problems is to randomly divide the dataset into three parts: a training set, a validation set and a test set. The training set is used to fit the models, the validation set is used to estimate prediction error for model selection and the test set is used for assessment of the generalization error of the final chosen model. Ideally, the test set should be kept aside and be used only at the end of the analysis. For this study we do have access to a lot of V-log data. But it turned out we need a time consuming quality assessment procedure on the V-log data. We end up with only one month of clean V-log data. We do not have such a data-rich situation and we need to use the k-fold cross validation method for a big part of the model selection. The test set consisted of the 'most regular day' from the dataset that we have. We are interested in being capable of predicting the normal behavior of the controller. Once we are capable of producing satisfying accuracies for the 'most regular day' we can state that we are able to predict the control behavior. This 'most regular day' is kept in a 'vault' and is excluded from any training set. By doing so, this test data is completely unseen for the trained models. The test set consisted of all weekdays from April 2014 minus the 'most regular day'.

7.9 Summary of Chapter 7

In this Chapter the following sub-question is answered: *What preprocessing steps need to be performed to transform V-log data to data that can directly be injected into the prediction algorithm?*

Preprocessing of the raw V-log predominantly covers the following actions: manual feature extraction, dataset construction, quality assessment of the data, feature selection, scaling and subsetting into train- and test datasets. The final output of the data preprocessing is useable training and test data for which the amount of corrupted data is reduced significantly. The manual feature extraction results into 112 features for each signal group extracted from the raw V-log data. The features are categorized in three categories: loop detector features, controller output state features and internal signal state features. The key philosophy during the feature extraction is to create features that show numerical variance that is needed to discriminate learning patterns in the data. Raw V-log data is logged in a binary fashion that is not suited for training the SVR.

The SVR algorithm relies heavily on clean training data. Multiple experiments show that poor loop detector health cause corruption in the retrieved dataset and is therefore a huge threat for effectively predicting the switching timings. To secure a clean and representative dataset we developed a quality assessment procedure. The assessment was performed by using standard statistical metrics on stratified segments of the preprocessed V-log data. By following the proposed quality assessment procedure deviant behavior such as extremely extended green phases or jittering loop detectors can be detected prior to training the SVR model.

Feature selection was performed in the context of model selection. We demonstrate that the green prediction model only uses features selected from the signal group for which it also predicts the switching timing of the green signal. The red prediction model uses features selected from all signal groups to predict the switching timing of the red signal. During the development of the prediction algorithm we saw a significant impact of different scaling techniques on the data. We propose scaling as shown in standard score values or Z-values. We want to emphasize that stratified scaling might lead to severe problems in finding discriminative learning patterns in the transformed V-log data by the SVR. As a solution, the population parameters from the total dataset should be used to transform the training data to standard score values.

8 The prediction model

In this Chapter the development of both the green and red prediction models are described. The SVR method does not have a fixed approach for designing an effective prediction model (Hsu et al., 2010). The SVR method has several facets that can be adjusted and optimized during the fitting of the model. First the facets for model selection are determined after which the experimental design and modeling strategy is explained. The major part of this Chapter is devoted to model selection of the best performing SVR model for predicting the switching timings of the red and green signals.

8.1 Facets for model selection

The SVR method has several facets that can be optimized, adjusted or estimated. One of the major pitfalls of the SVR model is that preliminary optimization of fitted models can lead to short-sighted decision making during the model selection. The following facets of the SVR need attention:

a) **Type of kernel**; the type of kernel directly depends on the input feature space (Hsu et al., 2010). The kernel strongly influences the estimated margin and support vectors and is unique to the V-log data. In section 5.3 Kernels, several standard kernels were stated. Six kernels are incorporated in the *Kernlab* R package that is used in this study. All six kernels will be tested during the model selection.

b) **Set of features**; the rationale behind feature selection in this study is described in the previous Chapter (section 7.6). Briefly, we would like to test predefined sets of features and select the most optimal feature set.

c) **Hyperparameters**; based on the type of kernel, zero or multiple hyperparameters need to be tuned. For example the most basic kernel the linear kernel: $k(x, x') = \langle x, x' \rangle$ has zero hyperparameters. The RBF kernel $k(x, x') = \exp(-\sigma\|x - x'\|^2)$, that is labeled as 'a first reasonable choice', has one hyperparameter (Karatzoglou & Meyer, 2006). Similar to the kernel, the hyperparameters depend directly on the input space. In contrast to the kernel type, the tuning of the hyperparameters are not fixed to the type of data. This means that optimization is needed for every single fitted prediction model. Whereas the kernel type is more fixed to the type of data.

d) **Regularization parameter**; specific regularization parameters of the SVR are the epsilon ε and cost C parameters (Smola & Schölkopf, 2003). The epsilon and cost parameter are referred to as regularization parameters. Unfortunately, there are no fixed rules that can be applied to determine the most suitable combination of parameter settings.

Four different facets are described that can be adjusted. Preliminary optimization or biased model selection decisions can be made because of the absence of an experimental design or modelling strategy. For example:

The type of kernel and its hyperparameters directly depends on the input space. If we select a different featureset, automatically the input space changes. As an effect the optimal kernel and hyperparameters are totally different. Additionally, hyperparameter and regularization parameter optimization could lead to different degrees in accuracy gains for different kernels. The linear kernel does not have room for optimization of hyperparameters, because the linear kernel has zero hyperparameters. While other kernels do have one or more *hyperparameters. Preliminary optimization might lead to misguided model selection choices due to over-optimistic model results (Fox, 2009).*

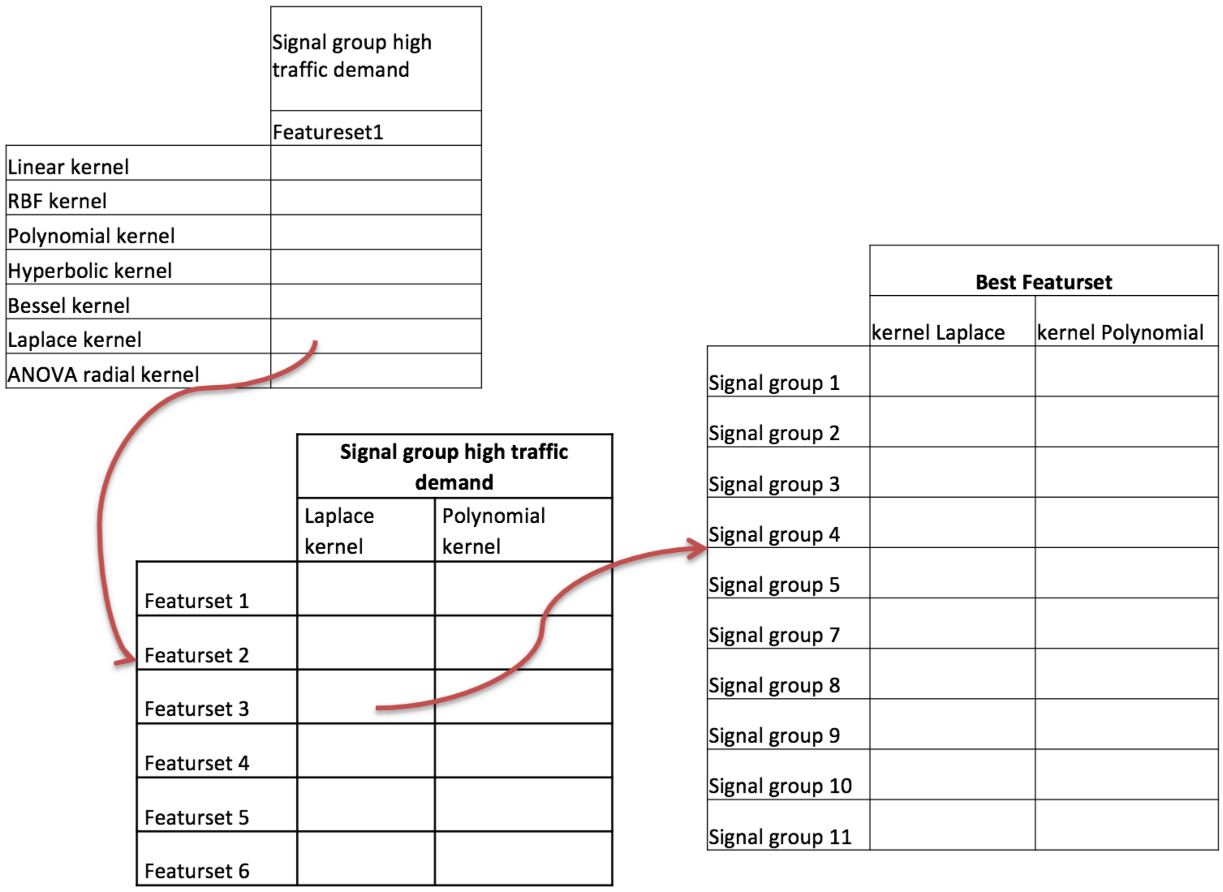
Questions that arise are: in what order do we try to fix facets of the model selection? When do we consider to optimize the hyperparameters and regularization parameters? Both questions relate to the amount of available computation time and amount of available and reliable data for this project. The next section explains a strategy to efficiently find the best performing SVR models.

8.2 Experimental design and modelling strategy

In this Chapter two the development of two prediction models is described. The first prediction model is capable of predicting the switching timings of the green signal. The second prediction model is capable of predicting the switching timings of the red signal. To efficiently develop both prediction models the following modelling strategy is used:

1. Construct the green prediction model and apply a reduced experimental design on the red prediction model. By doing so, we take the winning elements of the model selection from the green prediction model and apply those decisions directly on the red prediction model.
2. Focus the model selection procedure on the signal group with most traffic demand.
3. Use featureset 1 as initial input space. Featureset 1 contains only data regarding detector loop data and controller output signals. This data should contain sufficient explanatory variance for the prediction of a vehicle actuated controller.
4. Find the best performing kernel.
5. Find the best performing feature set.
6. Speed up the computation time over the full factorial design by determining the optimal amount of training data.
7. Deploy the algorithm on all signal groups.

Figure 24 illustrates the experimental design for the model selection of the green prediction model. It follows a fractional factorial design for the selection of a kernel type and a full factorial design for the selection of a feature set. A full factorial design for both kernel type and the feature sets is desired, but leads to extensive higher computation time. The fractional factorial design only uses a fraction of the full factorial design by limiting the amount of runs. The fractional factorial design for the kernel type includes only featureset 1. See the previous Chapter for more information about the six different featuresets. Featureset 1 consists of the biggest share of relevant V-log data and therefor will determine the input space for a large part.



	Signal group high traffic demand
	Featureset1
Linear kernel	
RBF kernel	
Polynomial kernel	
Hyperbolic kernel	
Bessel kernel	
Laplace kernel	
ANOVA radial kernel	

	Signal group high traffic demand	
	Laplace kernel	Polynomial kernel
Featurset 1		
Featurset 2		
Featurset 3		
Featurset 4		
Featurset 5		
Featurset 6		

	Best Featurset	
	kernel Laplace	kernel Polynomial
Signal group 1		
Signal group 2		
Signal group 3		
Signal group 4		
Signal group 5		
Signal group 7		
Signal group 8		
Signal group 9		
Signal group 10		
Signal group 11		

Figure 24 Experimental design for model selection

8.3 Model selection and preliminary results

This section describes the model selection and its forthcoming preliminary output results. The model selection is considered to be the major part of the development of the prediction model.

8.3.1 Output metrics

In the following sections and in the next Chapter output tables are presented. The tables contain output values for several metrics that are used for model selection and model validation. The validation process is described in more detail in section 8.5. The validation metrics that are mentioned in the output tables are: *Root Mean Square Error* [RMSE], *Mean Square Error* [MSE], *10-fold cross validation error* [Cross_error] and the *training error* [Train_error]. Next to the validation metrics 4 other metrics are often mentioned in the output tables. These metrics are: *the training time of the models* [Systimes], *number of selected support vectors* [SVs], *the minimum green countdown value* [Min_pred] and the accuracy in terms of *hit rate* [Hitrate]. The metric Systimes is arbitrary, since the measured Systimes will be different when other computers are used. However, the Systimes is relevant for comparison between models. The relative difference in Systimes between the models gives information about which model should be preferred over another. The number of Support Vectors gives explanatory insights when Systimes are high. Additionally, the number of support vectors could give information about how discriminative a kernel is. The Min_pred metrics indicates the minimum value that is being predicted by the prediction model. It should be obvious that a negative number of predicted green values is not tolerated. Negative prediction outputs are wrong by default, since negative green values would indicate that the switching time of the current signal is located in the past. The Hitrate is based on a permissible error in terms of absolute deviance of 4 seconds around the predicted switching timing. The 4 seconds is based on the duration of being 2 vehicles off from the true switching timing. Once the predicted switching moment is within the 4 seconds permissible error range, the prediction is labelled as a hit. If it is not within the 4 seconds permissible error, it is not treated as a hit. If the permissible error would be smaller than 4 seconds, the predicted output has a smaller chance of being labeled as a 'hit'. In contrast, if the permissible error would be larger than 4 seconds, the predicted output has a higher chance of being labeled as a 'hit'. The level of permissible error was set by researchers of Nissan. The metrics Systimes, Min_pred, RMSE, MAE, Cross_error and Train_error are measured in seconds. The Hitrate is measured in percentage and the number of support vectors is measured in the amount of estimated support vectors.

8.3.2 Best performing kernel

The high dimensional feature space in which the initial feature space is transformed by a selected kernel is difficult to visualize. Therefore, selecting the optimal kernel can only be determined by testing the kernel on real train and test data (Hsu et al., 2010). The R-software package *kernlab* contains six standard kernels. The six standard kernels that are mentioned in Table 3 are: the linear kernel (lineardot), the Gaussian radial basis function kernel (rbfdot), the polynomial kernel (polydot), the hyperbolic tangent kernel (tahndot), the Laplace radial Basis function (laplacedot) and the ANOVA radial basis kernel (anovadot). See section 5.3 for the mathematical description of the kernels. The six kernels are benchmarked to select the best performing kernel(s) for further model development. Three days of data were selected from the training set for benchmarking the kernels. Two days were used for training and one day was used for prediction. The data that was used for prediction was not used for training. As described in section 8.2 the kernels are tested by using the featureset 1 as input space. Featurset 1 contains only loop detector data and data of the output states of the controller. The most important metrics for the kernel selection are the Systimes, Min_pred and Mean Absolute Error (MAE).

	Systimes	SVs	Min_pred	Hitrate	RMSE	MAE	Cross_error	Train_error
rbfdot	16.04	4955	19.08	0.19	13.71	11.43	7.50	41.87
anovadot	162.60	2341	-49.62	0.29	61.76	15.21	1573.74	0.12
tahndot	20.11	5384	-12994.74	0.19	4924.43	3989.59	5563.85	203116.95
polydot	**42.07**	**4504**	**-15.13**	**0.33**	**10.56**	**8.12**	**9.39**	**0.46**
lineardot	45.99	5277	-15.89	0.31	10.67	8.20	9.45	0.46
laplacedot	**7.91**	**1844**	**4.18**	**0.27**	**10.90**	**8.79**	**1.49**	**0.01**

Table 3 Model Selection: comparison of Kernels

Table 3 shows that the kernel anovadot shows relatively high computation time and should be rejected. High computation times should be avoided, since this could make it difficult to complete the study. The tahndot kernel shows a low prediction accuracy and should not be taken into consideration for further model development. The rbfdot does not score well on the MAE and MIN_pred metrics and therefore is rejected for further model development. Three kernels are left: polydot, lineardot and laplacedot. These three kernels scored the best on the MAE metric. The laplacedot shows the biggest prediction potential, since the minimum prediction value is above zero. Negative prediction values would indicate an unreliable prediction model. The computation time of the Laplace kernel is the lowest of all kernels (simultaneously selects the smallest amount of support vectors). Initially the model selection was continued with the three kernels: polydot, lineardot and laplacedot. But preliminary results showed that the polydot and lineardot have almost similar prediction results. The similarity in output results is not strange, because mathematically they are almost identical. Due to the fact that the polydot kernel is slightly faster and has a better performance compared to the lineardot kernel, the model selection includes only the Laplace kernel (laplacedot) and the polynomial kernel (polydot). Conclusion: the best performing kernels are the Laplace and polynomial kernel. These kernels are fixed for the V-log data and are recommended to be used in studies involving Support Vector theory and V-log data.

8.3.3 Best performing featureset

Table 4 gives the output of the fractional factorial design between the six feature sets and the two best performing kernels. The six feature sets are described in section 7.6. At this point more focused is placed on the accuracy of the prediction models. The overall objective of model selection is to select the best performing model therefore the metrics Hitrate and MAE become central for the model selection procedure. As will be described in section 8.5, the Hitrate and MAE metrics are the most important validation metrics in this project.

	Kernel	Systimes	SVs	Min_pred	Hitrate	RMSE	MAE	Cross_error	Train_error
Featureset 1	polydot	41.72	4504	-15.13	0.33	10.56	8.12	9.39	0.46
Featureset 1	laplacedot	7.59	1844	4.18	0.27	10.90	8.79	1.52	0.01
Featureset 2	polydot	75.60	4461	-11.84	0.36	11.04	8.30	8.82	0.40
Featureset 2	laplacedot	47.47	3275	7.04	0.25	11.76	9.29	2.25	0.01
Featureset 3	**polydot**	**18.79**	**4577**	**-10.73**	**0.40**	**10.16**	**7.55**	**10.00**	**0.52**
Featureset 3	**laplacedot**	**13.75**	**4332**	**0.35**	**0.40**	**10.58**	**7.79**	**9.52**	**0.34**
Featureset 4	polydot	531.6	150	-50.00	0.17	18.65	14.72	0.93	0.00
Featureset 4	laplacedot	244.8	1972	20.48	0.19	14.07	11.55	4.77	0.01
Featureset 5	polydot	669.00	303	-58.38	0.10	20.89	16.85	0.67	0.05
Featureset 5	laplacedot	148.20	2671	19.33	0.19	14.16	11.51	12.36	130.34
Featureset 6	polydot	18.79	4600	-9.28	0.35	10.09	7.68	9.47	0.47
Featureset 6	laplacedot	13.75	4295	0.51	0.32	10.72	8.31	6.56	0.07

Table 4 Model selection: comparison of featursets

The featureset with the highest hitrates and lowest mean absolute errors are the correlation based featuresets. These featuresets have the numbers 3 and 6. For the green prediction model the model selection proceeds with featureset 3, since this featurset scores the best on the metrics: Hitrate, MAE and Systimes. Featureset 3 selects its features based on correlation between historical switching timings of the green signal and data retrieved ***only*** from the same signal group that we also want to predict. The difference between featureset 3 and 6 is that featureset 6 retrieves features from the entire intersection and therefor incorporates data from all signal groups into one prediction model. Featureset 3 ignores data coming from other signal groups. Conclusion: featureset 3 is the best performing feature set for the green prediction model.

8.3.4 Optimal amount of training data

As turns out, there is not 'one' green prediction model and 'one' red prediction model for an intersection. Every signal group has its own green and red prediction model. This can be seen in the full factorial design in Figure 24. The intersection as part of the case study has 10 signal groups. In total 20 prediction models need to be trained and validated for the intersection. Until now the prediction models were trained on 2 days of training data and predicted 1 day of training data. **The definition of 'trainingday' might be confusing, to clarify: one trainingday means data coming from one morning peak.** Statistical learning models such as the SVR algorithm are likely to learn and predict better once the model is trained on a larger dataset. Training the prediction model on a large dataset comes at a high cost such that a larger dataset increases the computation time. To speed up computation time it is relevant to know how much training data is sufficient to train the models. In the figure below the computation time (in seconds) as a function of increased training data for both kernels is shown.

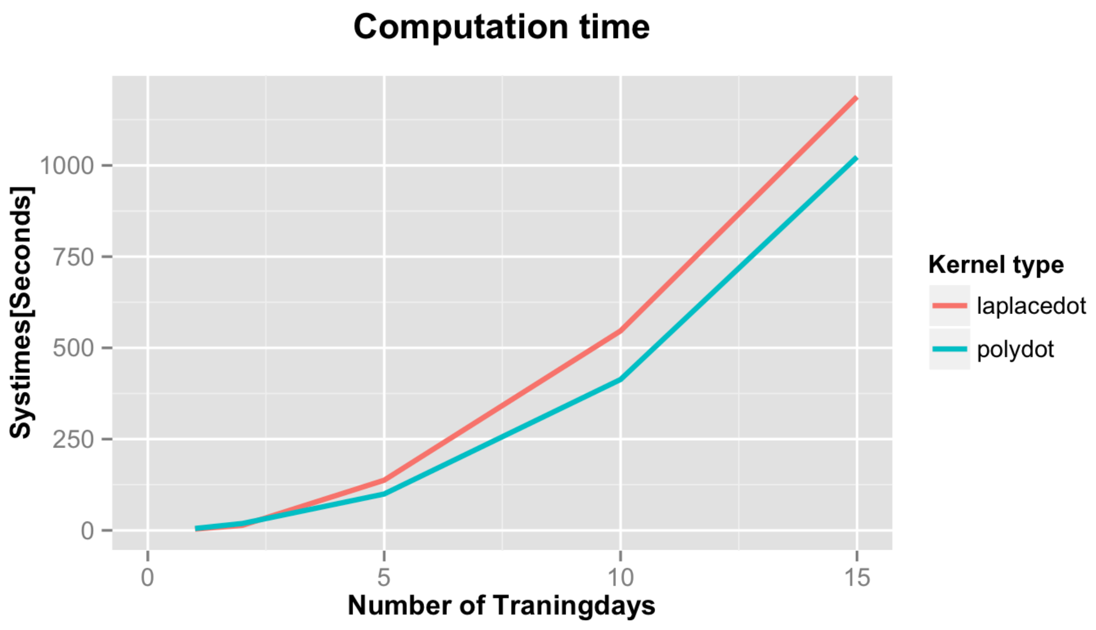


Figure 25 Computation time as a function of the amount of training data

Figure 25 reveals why the amount of training data should be chosen carefully. The computation time increases exponentially with the amount of used training data. To see what level of accuracies can be reached with the amount of used training data, the test error should be taken into account. In Figure 26 preliminary prediction results of the green prediction model (test error) as the function of the amount of used training data is shown. The upper limit of training days is given by the amount of available training days (weekdays only). Ideally the models are tested with a maximum of 50 to 100 training days. Unfortunately, this amount of data is not available in this project. Also the computation time would increase drastically (see Figure 25) if a relatively huge amount of training data is used. To give an impression of the data size for the experiments used in Figure 25 and Figure 26; the full dataset consists of 15 morning peaks and accounts for 1.8 million data points. Each data point covers data in 10 dimensions adding up to a total of 18 million data entries. The 10 dimensions of feature set 1 are advantageous compared to featureset 6. Featureset 6 consists of 25 dimensions and has 2.5 times the data entries of featureset 3. Both the learning curves for the two kernels in

Figure 26 show decreasing behavior and show an approximation of their asymptotic error. Both the curves tend to stabilize from a minimum of 5 training days on.

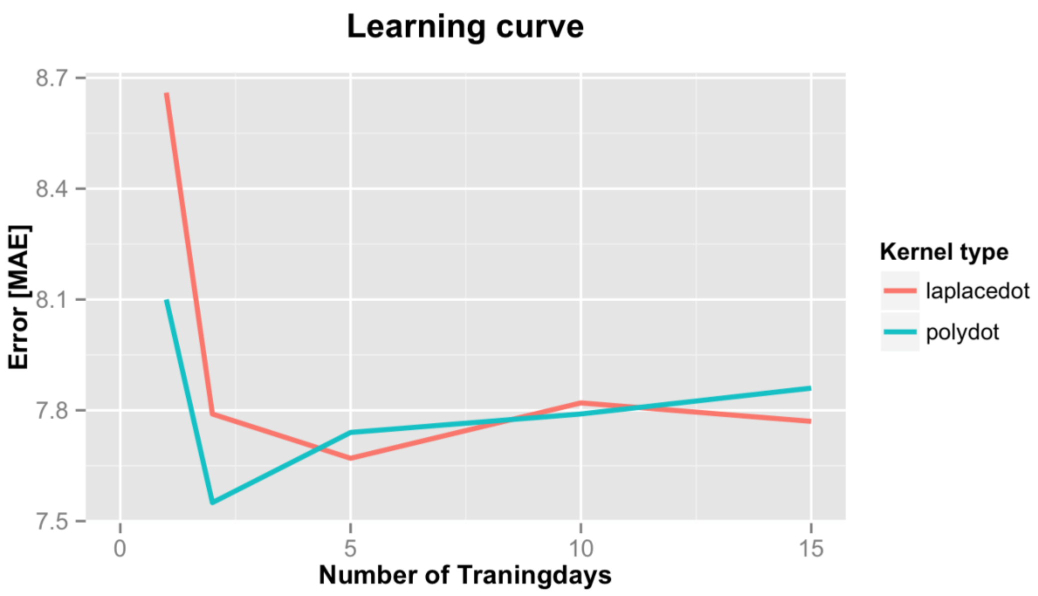


Figure 26 Test error/learning curve as the function of the amount of number of training days

Conclusion: 5 trainingdays is an optimal number of trainingdays. Both the approximation of the asymptotic test error as the computation time seems to be optimal at this level of trainingdays.

8.4 Overfitting and the curse of dimensionality

Overfitting occurs when the model complexity is increased and the model is trying to fit the training data to closely. During overfitting, the model is not trying to learn the underlying mechanism of the data, but is fitting the model to the noise in the data (Hastie et al., 2009). As a result, the generalization performance of the prediction model goes down. Assessment of this performance is extremely important, since it guides the choice of selecting the best model. But the SVR technique does not set its effective parameters based on the dimension of the input space like for example the ordinary least squares regression.

The reason that SVR tend to be relatively resistant to over-fitting, even in cases where the number of features is greater than the number of observations, is that it uses regularization and kernel selection. They key to avoiding over-fitting lies in a careful choice of the kernel (see section 8.3.2 for more information) and tuning of the kernel and regularization hyperparameters (see section 8.7 for more information). Every type of data has it unique properties and therefore the effect of the regularization parameters and kernel is different in every study involving different type of data. As shown in section 8.3.2 the difference in prediction error of the kernels is relatively big, causing the V-log data to dependent on the kernel selection. The reason why the SVR the relatively resistant to over-fitting is that the SVR approach could select a number of data points from a number of features that is less than the number of features that are presented to the SVR. In a way the number of support vector are effective parameters, but this is theoretically not completely true. A high number of support vectors does not immediately cause the model to overfit. Overfitting in a SVR model could only be seen in higher dimensions, causing the margin around the SVR tube to be wiggly and unstable (Hastie et al., 2009). Unfortunately, it is impossible to visualize the higher dimensions and can only assess overfitting based on test and training results. In practice, the risk of overfitting a SVR depends on the choice of the kernel, regularization parameters and the kernel hyperparameters (Cawley & Talbot, 2010). This means that SVRs don't solve the problem of overfitting but shift the problem from model fitting to model selection in which the choice of the kernel and regularization cost parameter C play a dominant role. Figure 27 gives insights in overfitting of the SVR. The explanation of possible overfitting is given after the figure.

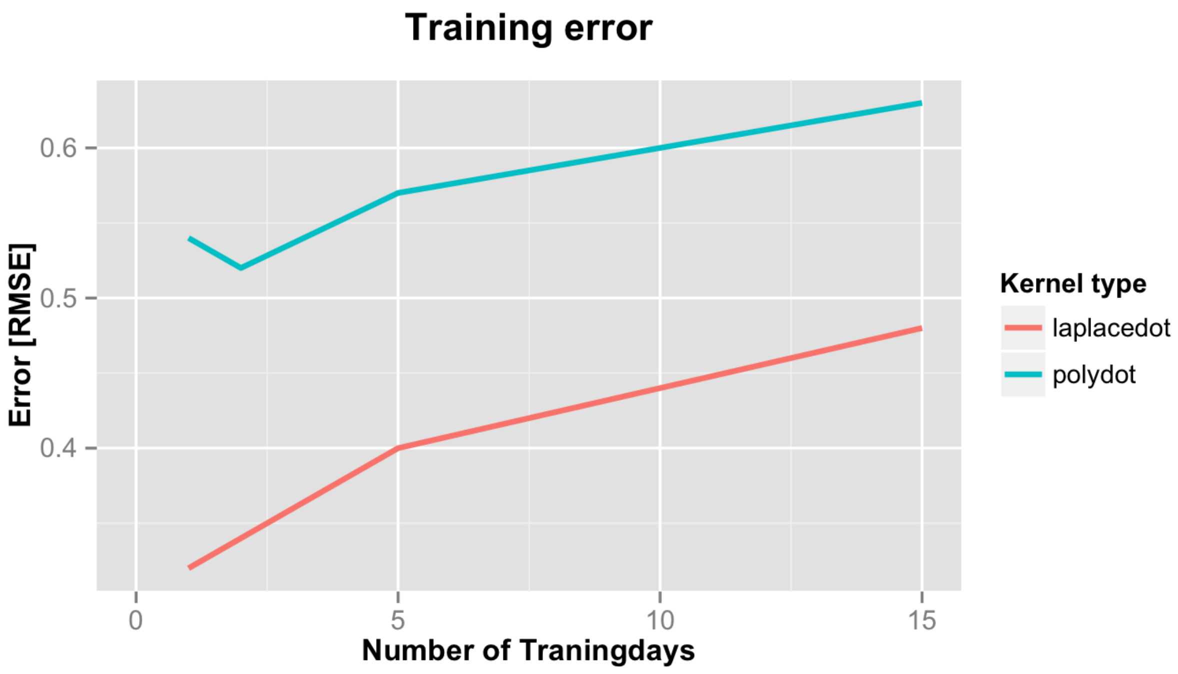


Figure 27 Training error as a function of the amount of training data

The Training error in Figure 27 correspond to the same experiments from Figure 26. The training error in Figure 27 increase for both the kernels, while the test error in Figure 26 is decreasing. Overfitting would mean that the training error is going down, while the test error is increasing. It is counterintuitive that the training error is increasing while test error is decreasing (Hastie et al., 2009). To understand this phenomena, the relation of the amount of training data and the model complexity in the SVR should be explained. Model complexity in a SVR can be expressed as the number of selected support vectors. If the amount of training data is increased also the chance of inclusion of more support vectors is increased. The increase of training error in Figure 27 shows a slight increase of training error that can be explained by the inclusion of more support vectors while presenting more training data to the SVR. Another explanation for the slight increase in training error is how the training error is constructed. The training error is the average loss over the training sample. When millions of extra data points are added to the models, it is likely to end up with a higher chance of wrongly predicted data points. Figure 27 and Figure 26 do not show overfitting since the test error stabilizes and does not show a rapid increase while the model complexity increases.

8.5 Important statements concerning validation procedure

In the previous section is explained what output metrics are used for assessing the model performance. To assess the generalization error, the true test data or test set is kept in a 'vault'. Meaning that trained models did not see the test data during its training process. Another relevant element is to determine what metrics should be used as assessment criteria. In previous tables 3 different type of metrics were mentioned that could be used as assessment criteria: Hitrate, Mean Absolute Error (MAE) and Root Mean Square Error (RMSE). The training error and cross validation error were given as Root Mean Square Error. While the test error is given in Hitrate, MAE and RMSE. During the model selection process an odd behavior of the cross validation procedure (10-fold cross validation) was discovered. Normally the cross validation error is always smaller than the test error, but always bigger than the training error (James et al., 2014). The cross validation is often a reliable approach of the generalization error or true test error (Hastie et al., 2009). However, preliminary model results indicate that the cross validation error metric is not a stable model selection metric and therefore should be used with high caution. In Table 5 examples of odd behavior of the cross validation error are highlighted. Often the cross validation error is overly optimistic about the performance of the trained models. For example, feaureset 4 has a RMSE cross validation error of 0.93 seconds, while the MAE test error is 14.72 seconds. The same behavior can be seen for featureset 5 with a cross validation error of 0.67 seconds and a MAE test error is 16.85 seconds. A second suspicious behavior is that the cross validation error is not always below the test errors: MAE or RMSE.

	Kernel	Systimes	SVs	Min_pred	Hitrate	RMSE	MAE	Cross_error	Train_error
Featureset 1	polydot	41.72	4504	-15.13	0.33	10.56	8.12	9.39	0.46
Featureset 1	laplacedot	7.59	1844	4.18	0.27	**10.90**	**8.79**	**1.52**	0.01
Featureset 2	polydot	75.60	4461	-11.84	0.36	11.04	8.30	8.82	0.40
Featureset 2	laplacedot	47.47	3275	7.04	0.25	**11.76**	**9.29**	**2.25**	0.01
Featureset 3	polydot	18.79	4577	-10.73	0.40	10.16	7.55	10.00	0.52
Featureset 3	laplacedot	13.75	4332	0.35	0.40	10.58	7.79	9.52	0.34
Featureset 4	polydot	531.6	150	-50.00	0.17	**18.65**	**14.72**	**0.93**	0.00
Featureset 4	laplacedot	244.8	1972	20.48	0.19	14.07	11.55	4.77	0.01
Featureset 5	polydot	669.00	303	-58.38	0.10	**20.89**	**16.85**	**0.67**	0.05
Featureset 5	laplacedot	148.20	2671	19.33	0.19	14.16	11.51	12.36	130.34
Featureset 6	polydot	18.79	4600	-9.28	0.35	10.09	7.68	9.47	0.47
Featureset 6	laplacedot	13.75	4295	0.51	0.32	10.72	8.31	6.56	0.07

Table 5 Instability of the cross validation error during feature selection

A reason why the cross validation error is unstable and therefore unreliable is rooted in how the sampling within the k-fold cross validation procedure is executed. We figured out that standard k-fold cross validation methods are not usable for V-log data. The reason is twofold: firstly, the data points are measured at a high frequency. Although the high frequency logging is advantageous for the learning process, it is not beneficial for the standard k-fold cross validation. The sampled *test fold* within the k-fold cross validation will contain data points that look similar to data points that were left in the remaining *training folds*. As a result, the k-fold cross validation error mimics the behavior of the training error. This behavior can be seen in Table 5 where in the highlighted cases the cross validation error moves towards the training error instead of the MAE or RMSE. Secondly, V-log has properties of time series in which a data point is heavily correlated with previous data points. By definition sampling away data points from a time series as a k-fold procedure relates to an interpolation procedure. Interpolating a time series is relatively less difficult compared to predicting completely unseen data for the time series (extrapolating) (Müller et al., 1997). Instead of using standard k-fold cross validation a hold out cross validation procedure was used. A complete training day was left aside and this particular day was used as pseudo-test data. In terms of k-fold cross validation we stratified folds in complete morning peaks instead of the random sampling of data points. True test data, that is used for model validation, is not part of the training data and is kept in a ‘vault’. True test data is used in the next chapter for the generalization error of the trained models. During the model validation we primarily looked at the mean absolute error (MAE) metric instead of the root means square error (RMSE) metric. The RMSE artificially makes small prediction error smaller and large prediction errors bigger. Since we linked the amount of predictions seconds we are off to the amount of vehicles we are off, the prediction models need to be assessed on the absolute deviance. As a result, the MAE metric is our main validation metric.

8.6 Model selection for the red prediction model

The model selection in the previous Chapter is described from the point of view from the green prediction model. For a large part the preprocessed data and model selection decisions can be directly copied to the red prediction model. During the preprocessing of the data, also data for the red signal timings was preprocessed similar to the data for the green signal timings. Examples of red signal timings features are: red duration of the current phase, past five phases and countdown timers of the red signal. The only facet that is different in the model selection procedure for the red prediction is the featureset selection. The red prediction model was tested on both featureset 3 and featureset 6. Previous experiments indicate that these two featuresets showed the biggest potential for the green prediction model. To reduce computation time, the SVR models are only fitted to these two featuresets.

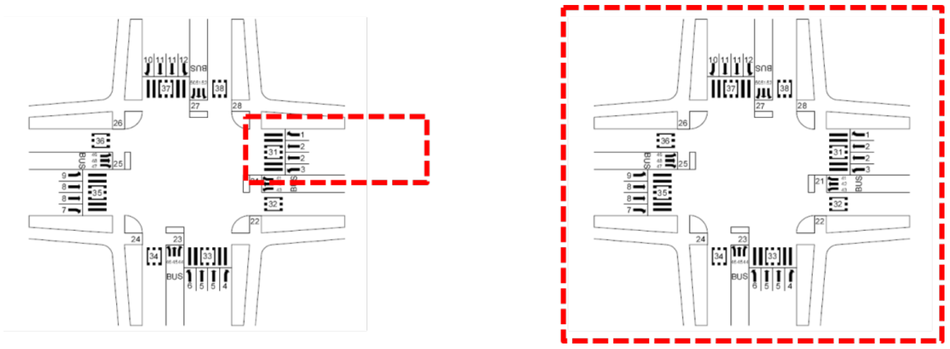


Figure 28 Difference between featureset 3 and featureset 6

Figure 28 illustrates the difference between featureset 3 and featureset 6. Both featuresets are based on selecting features according the highest correlation between features and the prediction target. The prediction target in the red prediction model is the red countdown timer. Featureset 3 contains only data from the signal group (e.g. signal group 02) that the prediction model tries to predict. Featureset 6 contains data from all signal group while it is trying to predict one particular signal group. In appendix D the output of the model selection on both the featuresets are given. The training data is retrieved from signal group 8, which is the same signal group that is used for the model selection of the green prediction model. Additionally, the amount of training data is varied to see if the red prediction model has different asymptotic error behavior. The red prediction model does not show deviant behavior on its learning curve compared to the green prediction model and therefore also 5 trainingdays of data is preferred as optimal training set.

8.7 Optimization of hyperparameters and regularization parameters

The computation of the SVR contains several parameters that can be optimized. Firstly, the SVR technique contains regularization parameters. Secondly, a kernel can have hyperparameters that can be tuned during the training process (Hsu et al., 2003). Often the regularization parameters are referred to as hyperparameters. We save the definition of hyperparameters for parameters concerning the kernel. Whether the kernel has hyperparameters depends on the chosen kernel. More hyperparameters give room for more optimization, but also cause higher computation time that is needed for optimization. During a fast explorative optimization of the parameters was found that optimization of the parameters is indeed beneficial. To cope with time constraint in this project further development of the models is proceeded with the default settings of the hyperparameters.

The regularization parameters specific to the SVR are the epsilon ε and cost C parameters. Unfortunately, there are no fixed rules that can be applied to determine the most suitable combination of parameter settings. Chang and Lin argue that the maximum value of epsilon ε should not exceed the following rule: $(max_{i=1,\dots,l}y_i - min_{i=1,\dots,l}y_i)/2$ (Chang & Lin, 2002). According to the current knowledge no effective lower bound of epsilon is theoretically defined. Intuitively it should be closely related to the prediction target variable. In this study the target variable is the green or red duration. Similar to the hyperparameters of the kernel, the default setting of the regularization parameters within kernlab R-package give proper results for initial model selection. To use a more pragmatic approach for optimizing the parameters a stratified cross-validation in combination with Grid-search can be used (Hsu et al., 2003). The grid-search computes a number of pairs for the regularization parameters and hyperparameters. The combination of parameters with the best cross-validation accuracy is chosen as best performing model. The Grid-search follows *a full-factorial design* or *fully crossed design*, meaning that every combination of parameters is computed. Due to limited amount of time for this study extensive optimization of the parameters are recommended for further research. We did execute a couple of random optimization checks with a limited grid-search including randomly picked values. The preliminary optimization checks were promising and even under limited amount of training data the test prediction results showed an optimization of almost 20% of increase in accuracy (MAE error).

8.8 Summary of Chapter 8

This Chapter predominantly describes the model selection procedure. The model selection procedure is supported by a custom made modelling strategy and an experimental design for predicting switching timings. The modelling strategy results into a more efficient experimental design to find the optimal settings for the SVR prediction models. Decisions during the model selection are supported by the knowledge gained from investigating the control logic, the preprocessing of the V-log data and the mathematics behind the SVR. 'The algorithm' is therefore more than only the code of the SVR and starts from the point where raw V-log data is transformed to a useable training and test set. The model selection procedure that is described in this Chapter was not described in any prior study. Informed decisions based on empirical findings will make this study easy to reproduce and useful for reproducing similar prediction models. Important findings in this Chapter regarding the green and red prediction models are:

General points both the green and red prediction model:

- Best performing kernels: Laplace and Polynomial kernels.

- Optimal amount of morning peaks/ trainingdays for which the green prediction model must be trained: 5 days (weekdays only). These days can be randomly selected.
- Preliminary optimization of the hyperparamters and regularization parameters should not be performed. This will lead to a complex study design leading to high computation time. Default settings in the kernlab R-package are sufficient for model selection and validation of the models.
- Standard k-fold cross validation gives an overly optimistic generalization error. A stratified hold out cross validation procedure is preferred.
- Featureset selection based on correlation gives the best set of features.

Specific to the green Prediction model:

1) The preferred featureset selection approach for the green prediction model selects only features that contain data from the signal group that it tries to predict. This is described as featureset 3 (see section 7.6.1 for more information).

Specific to the red Prediction model:

2) The preferred featureset selection approach for the red prediction model selects features that contain data from all signal group while it is trying to predict one particular signal group. This is described as featureset 6 (see section 7.6.1 for more information).

PART **III**

Results and Final Remarks

9 Model results

In this Chapter the final results of the prediction capabilities of both the green and red prediction models are given. The prediction results contain information about the generalization error, but also a deeper understanding about the prediction behavior is given. The algorithm described in the previous Chapter is deployed on all signal groups of the intersection from the case study. See Chapter 6 for more information about the chosen intersection N245 Laan van Darmstadt.

9.1 Results green prediction model on full intersection

Table 6 gives is a summarized version of the output table in Appendix E. In total the intersection has 10 signal groups. The labelling goes up to 11, but signal group 6 is not existing. For each of the 10 signal groups a green prediction model was trained. The metrics: Systimes, RMSE, MAE and Cross_error are measured in seconds. The metric Hitrate is measured in percentage. See section 8.3.1 for more information regarding the output metrics. In Table 6 is shown which kernel is preferred by the model selection after the training process. The computation time, denoted as Systimes, include the training time of the model and the prediction of the new data. Some models are relatively fast, while other models are relatively slow. In Table 10 in Appendix E the amount of Support Vectors for each model is given. It seems that there is a relation between slower models and the higher amount of selected Support Vectors. This behavior is seen under the circumstances that each model gets presented the same amount of data. As explanation: models with higher computation time are less discriminative on the data. This does not necessarily mean that the prediction models do perform worse than more discriminative models. For example, signal group 3 has a higher Hitrate and a lower MAE compared to signal group 9. Signal group 3 needs approximately 21 minutes to train and predict and signal group 9 approximately 1.5 minute to train and predict. When a model selects a high number of support vectors there is probably room for optimization. In theory we could search for more explanatory features that make the model more discriminative and therefore selects a lower amount of Support Vectors. This eventually could also speed up the computation time.

Signal group	Kernel	Systimes	Hitrate	RMSE	MAE	Cross_error
1	polynomial	656.57	0.14	26.25	19.76	29.25
2	polynomial	151.70	0.25	15.95	11.46	14.64
3	**polynomial**	**1262.16**	**0.84**	**2.94**	**2.14**	**3.19**
4	**polynomial**	**68.80**	**0.78**	**3.31**	**2.68**	**4.09**
5	**Laplace**	**253.96**	**0.99**	**1.25**	**0.85**	**1.30**
7	Laplace	401.40	0.29	15.82	11.20	12.74
8	Laplace	156.06	0.40	10.44	7.67	9.63
9	**Laplace**	**97.19**	**0.60**	**7.34**	**5.02**	**3.85**
10	**Laplace**	**97.46**	**0.54**	**7.54**	**5.12**	**6.87**
11	**polynomial**	**1841.55**	**0.90**	**3.03**	**2.20**	**3.21**

Table 6 Results Green prediction model on full intersection

Six models in Table 6 (models 3, 4, 5, 9, 10 and 11) are highlighted in bold with a green color. The green color indicates that the model performs satisfying, although the mean absolute error (MAE) on the test set of these models is above the 4 seconds error boundary. The reason why models with a MAE around 5 seconds are labeled as satisfying is twofold: 1) upcoming section 9.3 shows that these models catch the underlying mechanism of a countdown and predict expected control behavior (see section 9.3), 2) preliminary optimization shows that relatively significant accuracies gains of around 20% can be achieved by optimization of the hyperparemeters and regularization parameters (see section 8.7). Furthermore, there seems to be a hidden factor why signal groups are either predicting relatively good or predicting not good at all. Figure 29 illustrates the prediction capabilities of the Green prediction model for the intersection *N245 Laan van Darmstadt* in a graphic.

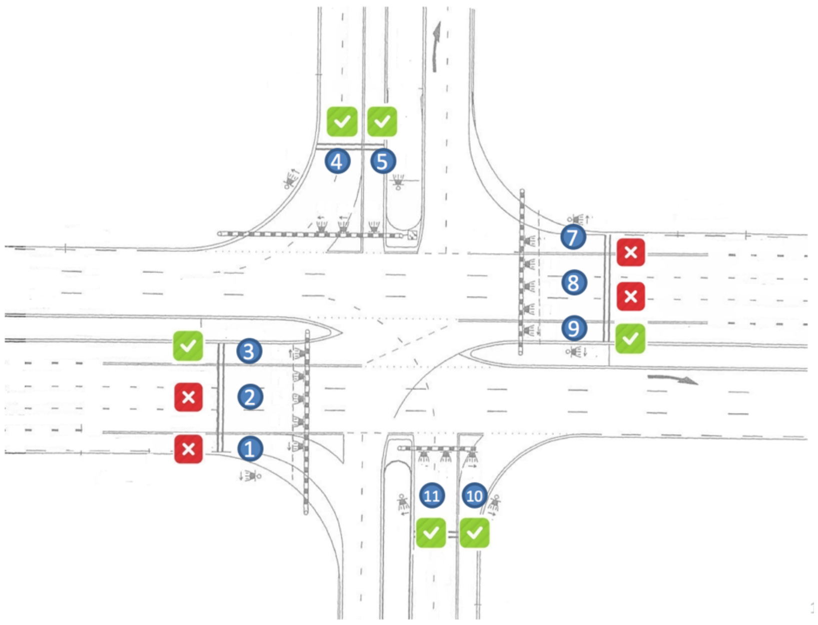


Figure 29 Graphical representation of results Green prediction model on full intersection

9.2 Results red prediction model on full intersection

Table 7 gives a summarized version of the model output of Table 11 in Appendix E. The same interpretation as the green prediction model output from the previous section applies to Table 7. Model output from Table 7 shows that the red prediction model has a preference for the Laplace kernel while the green prediction model shows an equal mix of kernel preference. Although the red prediction model does perform satisfying on five signal groups the other five signal groups are extremely off. Furthermore, the red prediction models have on average a high computation time compared to the green prediction models. Signal group 1 reaches almost 87 minutes (5219.29 seconds) for training and prediction. The fastest red prediction model takes over 10 minutes (606.46 seconds) for training and prediction.

Signal group	Kernel	Systimes	Hitrate	RMSE	MAE	Cross_error
1	Laplace	5219.29	0.81	4.45	2.42	2.38
2	Laplace	649.38	0.70	9.17	4.79	3.40
3	polynomial	2355.61	0.20	20.41	13.86	25.21
4	Laplace	4645.37	0.56	8.65	5.65	4.79
5	Laplace	1803.74	0.37	21.86	11.99	14.03
7	Laplace	606.46	0.73	4.75	3.27	2.44
8	Laplace	880.24	0.49	7.05	5.32	3.08
9	Laplace	483.71	0.02	182.98	133.26	20.97
10	Laplace	1253.76	0.35	16.35	11.38	9.68
11	Laplace	1807.74	0.45	11.53	7.90	5.14

Table 7 Results Red prediction model on full intersection

Figure 30 illustrates the prediction capabilities of the Red prediction model for the intersection *N245 Laan van Darmstadt* in a graphic.

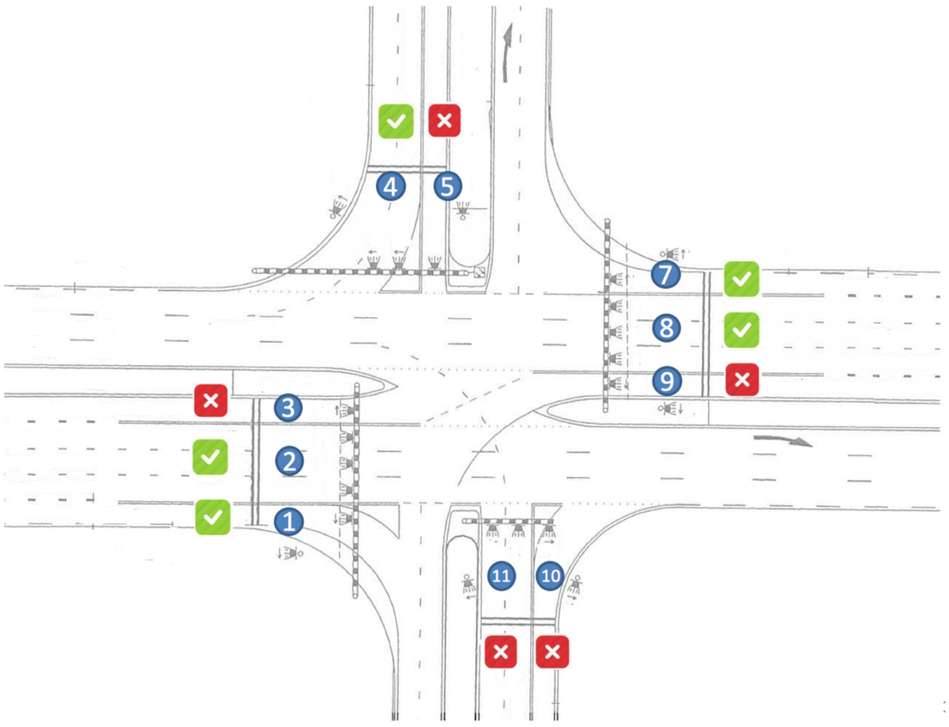


Figure 30 Graphical representation of results Red prediction model on full intersection

9.3 Behavior of the prediction models

The Mean Absolute Error (MAE) is used as the main validation criterion in the output tables of the green and red prediction models. Besides the average prediction error in absolute deviance (MAE) also the behavior of the prediction is relevant and informative for assessing the performance of the prediction algorithm. The MAE metric does report the generalization error, but it does not reveal into greater detail the behavior of the prediction model. To get more insight in the behavior of the controller a couple of complete phases during the morning peak are sampled for further investigation. Figure 31 shows such a sampled green phase of signal group 8. The Figure 31 contains the true countdown line versus the predicted countdown line. The y-axis is the countdown of the green signal and represents the number of seconds left in the phase at every given second during that phase. The prediction model predicts for every second in the phase how many seconds there are left until switching to another signal. The x-axis represents the timer of the green signal and will increase after every second in the phase. The prediction behavior in Figure 31 is seen in the majority of the sampled green phases: in early stages when the green signal is high the predicted countdown is close to the true countdown. However, when the timer of the green signal gets closer to its maximum (in Figure 31 the maximum is 35 seconds) the accuracy of the predicted countdown is deteriorating and the error is amplifying. The predicted countdown line in Figure 31 does not follow a decreasing straight line or a monotone decreasing function, but it does go down. In conclusion, the trained model understands that is should predict its switching timings as a countdown and therefore matches expected control behavior.

The error in the last few seconds of the countdown is probably due to high variance in the data. The variance is likely to be created by the controller due to several characteristics of the controller: 1) fixed states such as the fixed green time already were realized, 2) loop detector thresholds might be reached by incoming vehicles or might be missed and therefore misses green extension and 3) the controller changes its state due to priority rules on other signal groups.

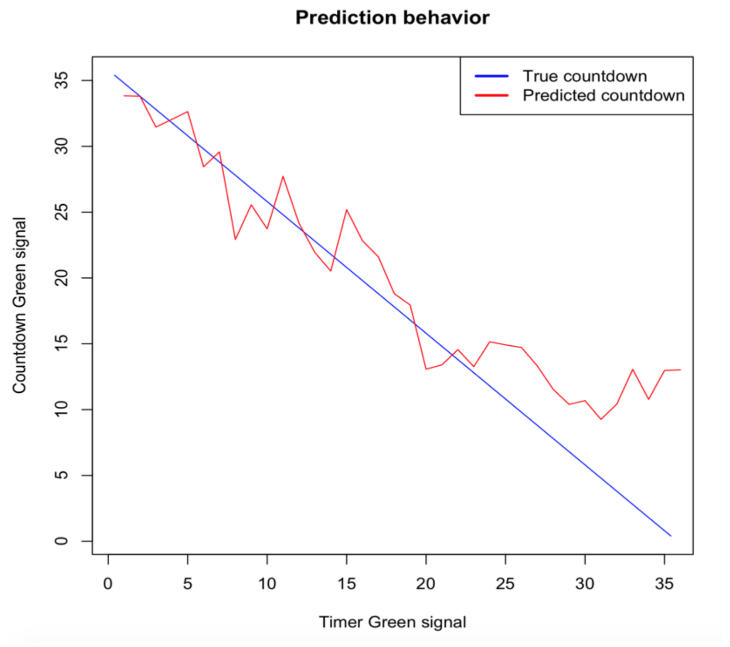


Figure 31 Prediction behavior

Figure 31 reveals that prediction models trying to predict a vehicle actuated controller might be poorly assessed by averaged metrics such as the Mean Absolute Error (MAE) or Root Mean Square Error (RMSE). The majority of the phase in Figure 31 was predicted correct, but the last seconds of the phase amplified the overall error.

9.4 Troubleshooting: a visible pattern

The green and red prediction model output (Table 6 & 7) shows that either the prediction model predicts adequate or it predicts the signal timing extremely off. The red prediction model has trouble predicting the turning movements (see Figure 30), while the green prediction model has trouble predicting the through movements (see Figure 29). When the output of both the green and red prediction model are layered on top of each other a pattern is shown. Figure 32 reveals the pattern between failing predictions of the green and red models.

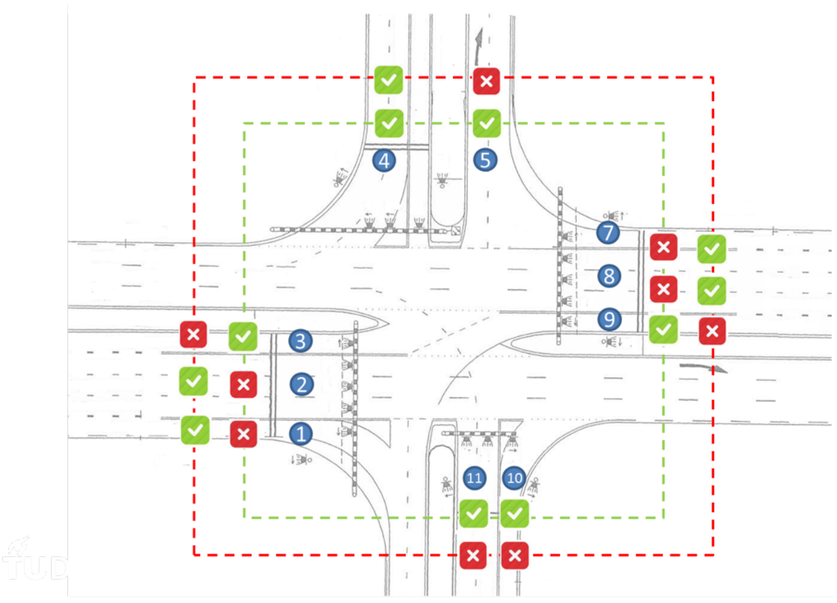


Figure 32 Graphical representation of the detected pattern in model output

The control scheme of the controller indicates that signal group 1, 2, 7 and 8 are likely to have a green signal during the same phase. This explains why the green prediction model is not able to predict any of those four signal groups correct if three other signal groups of the subset 1, 2, 7 and 8 are unpredictable. The pattern from Figure 32 could also indicate that the majority of conflicting signal groups with signal groups 1, 2, 7 and 8 are unpredictable. By doing standard statistical inference checks such as plotting histograms of several features, unexpected behavior in the green durations were found. In the V-log data of the test dataset, green phases longer than the maximum green time plus green time extension were found. The maximum green time plus time from the extension of the green signal should in theory give at maximum 40 till 50 seconds of green for signal group 8. Figure 33 shows that the the controller logs green phases with a maximum duration of above 65 and even 80 seconds.

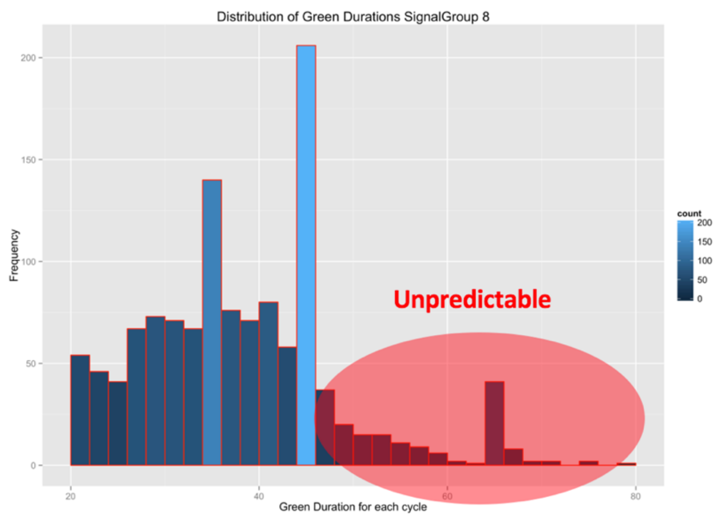


Figure 33 Illustration of unexpected green duration phases

Two question arise: why are these extremely extended green phases a problem for the green prediction model? And why do these extremely extended green phases occur? The first question is partly answered by analyzing Figure 34. Figure 34 shows how the trained prediction model behaves when it tries to predict the extremely extended phases of green time. The graph shows a green phase of 65 seconds. The fitted model simply does not recognize the true countdown from the first second onwards. This behavior is different as illustrated in Figure 31 where the predicted countdown predicts reasonable close to the true countdown in the early part of the green phase. In other words, the prediction model does not catch the control logic of those extremely extended green phases. A statistical explanation for not recognizing the control logic could be that the underlying mechanism to predict the extremely extended green phases is not present in the data. In conclusion, the data on which the prediction model is trained does not include the essential information to predict the extremely extended green phases.

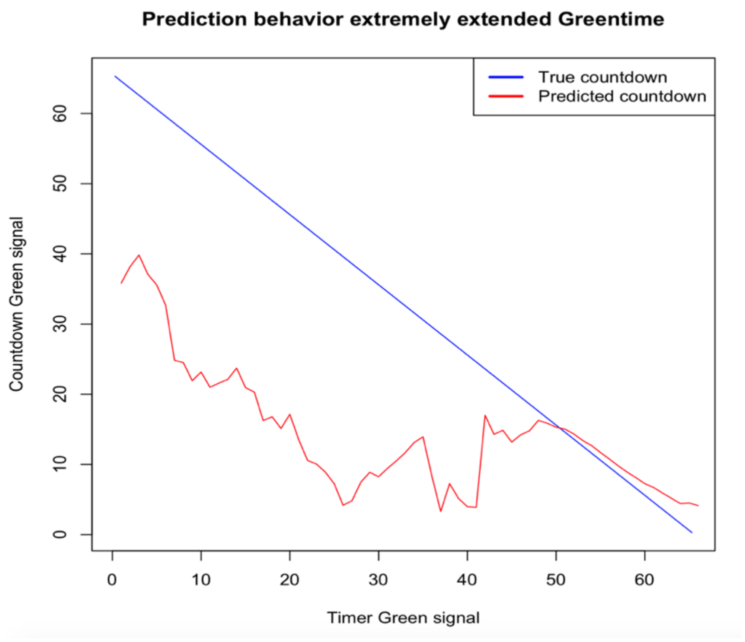


Figure 34 Prediction behavior of extremely extended Green time

It turns out that the chosen intersection N245 Laan van Darmstadt has a synchronized connection with an intersection upstream from the viewpoint of signal group 8. During the morning peak there is a chance that the intersection upstream of the intersection N245 gives a control signal to artificially extend the green phase of intersection N245 so that approaching vehicles hit the green light. Information about the synchronization was not supplied by North-Holland nor data about this synchronization was included into the data. Unfortunately, the quality assessment framework at that time only included the arithmetic mean as a quality assessment metric. The arithmetic mean smoothens out outliers in the green duration features.

9.5 Feature preference of the green and red prediction model

Both the green and red prediction model select a set of features from the initial set of manually extracted features (over 1000 features). The difference between the feature sets of the green and red prediction model is described in Chapter 7 in more detail. Briefly, the difference between the two prediction models: the green prediction models select features, that only contain data from the signal group that the prediction model tries to predict. The red prediction models select features, that contain data from all signal groups while it is trying to predict one particular signal group. The green prediction model has a narrow minded feature set selection, while the red prediction model performs a more holistic feature set selection.

In appendix F two tables are mentioned that summarize the frequency of features that the green and red prediction models select while training on the data from the full intersection N245 Laan van Darmstadt. In total 10 green prediction models and 10 red prediction models were trained for the intersection. The relative frequency of occurring features in the selected featuresets for the 10 green and 10 red prediction models contains interesting information. For example: the red prediction model selects predominantly features that contain data regarding the internal parameters states of the controller. The green prediction model predominantly selects features that contain data about the detector loops. Furthermore, both models hardly select features that contain historical information about previous red and green phases. This is in contrast with the findings in the study performed by Weisheit and Hoyer. Weisheit and Hoyer used the classification class of the Support Vector Theory (SVM) on a semi vehicle-actuated controller with fixed phase lengths (Weisheit & Hoyer, 2014). Weisheit and Hoyer report that data from the previous three phases result into improvements of the prediction error of the Support Vector Machine classification model. In another study in which the SVR was used on basic controller output data, the data consisted of historical data up to the previous five phases (Koukoumidis et al., 2012). The difference in findings regarding the historical data could be explained by the use of other data from the controller besides detector loop data. In this project other features besides detector loop data are used for predicting the switching timings.

Another finding, which might be of interest of suppliers of the V-log data, is that the red prediction model tends to pay more attention to third detector loop of a signal group. In the green prediction model we see that data coming from the first and third detector loop is almost equal. We would expect that the third detector loop would have major preference over the first detector loop since the third detector loop has a slight advantage of detecting a vehicle earlier in time.

9.6 Summary of Chapter 9

In this Chapter the model results of the green and red prediction models are given for the intersection Laan van Darmstadt. The best performing green prediction model reports a mean absolute error of 0.85 seconds and a hitrate of 99%. The mean absolute error indicates the difference between the true switching timing and the predicted switching timing, while the hitrate gives the percentage of predicted switching timings within a 4 seconds permissible error in terms of absolute deviance from the true switching timing. The best performing red prediction model shows a mean absolute error of 2.42 seconds and a hitrate of 81%. Improvements in the prediction performance can be gained by numerical optimization of the trained models. On average the red prediction models needed more computation time compared to the green prediction models. The fastest red prediction model needed 606.46 seconds for training and prediction. The fastest green prediction model needed 68.80 seconds for training and prediction. Troubleshooting indicates that the prediction performance of several signal groups can be improved by adding features that describe the synchronization link with an intersection upstream the investigated intersection.

Besides the reported validation metrics, the model results show that the SVR is capable of predicting expected control behavior. The SVR models predict for every second in the cycle the amount of seconds left until the moment of switching. The expected control behavior is that the amount of predicted seconds until the moment of switching follow a countdown behavior. Model results show that the SVR is able to predict such countdown behavior and therefore matches expected control logic.

10 Discussion

In this Chapter several topics are being discussed that give perspective on the usability and reliability of the developed prediction algorithm. In sequential order the following topics are discussed: first, the selected mathematical prediction approach and possible other preferred approaches are discussed. Secondly, the differences with previous research that uses the Support Vector Theory is described. Thirdly, the usability of V-log data for prediction purposes is discussed. Next, the assumptions that are made in the current prediction algorithm and the case study are discussed. Last, the implementation of the prediction algorithm and developments around self-driving vehicle technology that could influence the use of the prediction algorithm are discussed.

10.1 The selected prediction approach

In Chapter 4 six prediction approaches are described that were used in prior research to predict traffic signal timings. A benchmark of the six described prediction approaches resulted into the selection of the Support Vector Regression (SVR) as prediction approach. Many of the prior researched prediction approaches are not suited to predict the switching timings of the vehicle-actuated controller that shows variability in both the phase length and cycle length. Chapter 8 describes that the developed SVR models do not require knowledge about the structure of the controller and therefore the design of the models does not depend on the structure of the controller. However, the structure of the controller does make it harder for the SVR approach to be able to predict the switching timings accurately (see Chapter 9 for more information). If the structure of the controller is angled towards a pre-timed control scheme, similar to other previous researched method, the SVR is able to predict the switching timings more accurately.

The Support Vector Machine (SVM) algorithm has almost the same mathematical approach as the SVR. In Chapter 4 is explained that the SVM approach is not preferred for the prediction of signal timings of a vehicle-actuated controller due to its classification nature. In theory, the SVM could predict the switching timings by treating every second in the signal cycle as a class (Weisheit & Hoyer, 2014). The vehicle-actuated controller that is used in the case study shows variability in the signal cycle. The amount of possible seconds of switching becomes large, leading to a large set of classes. By treating every second in the cycle as a class, the SVM algorithm becomes a multi-class classifier that uses posterior probabilities to select the most probable class of switching (James et al., 2014). When the signal cycle shows weak distribution for switching timings for every second, the SVM will tend to assign higher posterior probabilities to switching timings (second in a cycle) that is most frequent in the training data. By default, the most frequent second of switching in the training set will have an advantage of being selected as switching moment. This problem could be fixed by treating segments of 4 or 5 seconds as a class instead of every second as a class. However, weak posterior probabilities could still occur in classes that are not occupied with a lot of data. The SVM approach in combination with broader class segments might work satisfying in cases where the switching timings are uniformly distributed in the training set.

A second method, that was not investigated in this project due to the classification nature of the method and also was not mentioned in previous research, is the use of decision trees. Again every second or an interval of seconds could be treated as a class. The problem of predicting the most frequent class could be prevented by using techniques such as boosting or random forest. Both techniques are using variance reduction methods by producing many uncorrelated trees (random forest) or weak learners (boosting). Currently boosting in combination with decision stumps (single decision trees) is outperforming almost any classification model (Hastie et al., 2009). For more information about random forest and boosting see the book *The Elements of Statistical Learning* by Hastie et al. The biggest disadvantage of using classification models over regression type models are the low posterior probabilities for classifying the switching timings in low density regions. Data sparsity due to low density regions could be prevented by expanding the interval of the class, but the expansion will affect the prediction accuracy by adding artificial uncertainty due to a 4 or 5 seconds interval.

10.2 Differences with previous research

The prediction of traffic light controllers in previous research is usually related to Green Light Optimal Speed Advisory (GLOSA) systems. A GLOSA system contains a prediction component for pre-timed or semi vehicle-actuated control schemes. The level of complexity of these control schemes is considerably lower compared to the control scheme used in North-Holland. Previous studies explicitly stated the application for which the prediction component is needed. The focus of these studies is angled more towards the integration of the prediction application instead of the theory behind the prediction algorithm. In this project the focal point is placed more on a proof of concept of an algorithm to predict the controller and therefore has the ambition to formulate a theory instead of an application. Two prediction approaches showed similarities with the prediction approach in this project.

The prediction application that shows similarities with to the prediction task of this project has the name SignalGuru (Koukoumidis et al., 2012). Koukoumidis et al. investigated the prediction of vehicle-actuated controllers by using Support Vector Regression. The Support Vector Regression makes use of basic historical controller output signals and lane saturation information. No background information is given about what data represents lane saturation. The intersection that was used for a pilot is located in Singapore and follows a high level of complexity. Koukoumidis et al. do not use data that describe internal states of the controller which is the biggest difference with this project. The developed SVR models in this project utilize the data coming from the controller to a greater extend by including data related to internal states of the controller.

Weisheit and Hoyer propose a prediction model that utilizes the Support Vector Machine approach for predicting the switching timings of a semi vehicle-actuated controlled intersection with fixed cycle lengths. Because of the fixed cycle lengths, the prediction task is treated as a classification problem. The prediction of a fully vehicle-actuated controlled intersection cannot easily be formulated as a classification task. Weisheit and Hoyer state that historical data up to three previous cycles is beneficial. The model results from Chapter 9 show that historical data from the previous three cycles is not important for the SVR approach and could be ignored. Weisheit and Hoyer do not use use data that describe internal states of the controller.

To our knowledge no comparable study gives information about preprocessing of the raw data retrieved from the traffic light controller. Also no study includes information about what analytics are used to assess the quality of the used data.

10.3 The use of V-log data

In this project V-log data is extensively used for predicting the output state of the traffic light controller. Several insights were gained throughout the process concerning the usability of V-log data for prediction purposes. The first concern regarding the usability of V-log data for a prediction application is related to the extraction of the data and its data format. The actual extraction of the raw data from the controller was beyond the technical capabilities of this project. As part of the joint project, North-Holland developed software to retrieve the V-log data and transform the data to .CSV extension. As a drawback, the developed software was a black box and I relied heavily on the correctness of the code. The process of retrieving V-log data and transforming the data into .CSV extension was completely new for North-Holland. The novelty of the system caused several datasets to be incorrect. Luckily professionals in the field of traffic light control and V-log from North-Holland were involved in the project and problems with the data could be discussed rapidly. The data format of raw V-log is not usable for statistical learning approaches and therefore by default raw V-log data cannot be directly used for prediction purposes. After the transformation to a .CSV extension, the V-log data becomes useable for prediction methods such as the SVR.

The second concern regarding the usability of V-log data for a prediction application is related to the quality and availability of V-log data. In theory large amounts of V-log data was made available for this project, but in practice only a relative small amount of data was useable for training and testing. The data plumbing caused the delivered datasets to be reduced significantly because the data contained either corruption or many days of data were missing. As an example I was primarily interested in using complete weeks as training data to correct for factors that occur only on a particular weekday. If the models would only be trained on Mondays, it would probably show deviant prediction behavior for predicting data from Fridays. It turned out that the existence of complete weeks of data was not trivial due to the poor health of the loop detectors, causing complete days to miss loop detector data. Besides the use of V-log data for training purposes, the prediction algorithm needs V-log data to predict the future signal timings. In case of missing loop detector data, which occurs often in reality, the prediction algorithm is not capable of predicting the switching timings. In conclusion, the current quality of V-log is too volatile for supporting the SVR prediction algorithm that predicts the switching timings in a real time application.

A third point of attention for further use of V-log data for self-driving vehicle technology is the inclusion of the correct data. It would be preferable if V-log data contains the prediction information itself and the vehicle does not need to predict the switching timings. In the situation that the vehicle is producing the prediction, the V-log data should at least contain the list of data features with numerical data that is mentioned in Appendix F.

10.4 Developed algorithm and case study

In the previous section is explained how the current quality of V-log causes issues for training the prediction models and predicting the switching timings. Low quality V-log is caused both by missing data and corruption in the data. The prediction accuracy of the developed algorithm is influenced directly by injecting low quality V-log into the model for both training and predicting the switching timings. During the case study was noticed that corrupted V-log or missing data in the V-log occurs frequently. As a result, the SVR prediction algorithm is vulnerable and lacks robustness for using it in a real time application. In the case study corruption in the data was detected by a quality assessment framework. The quality assessment framework was constructed with limited amount of time and therefore is not comprehensive enough to detect a wide variety of corruption in the data.

The quality assessment uses basic statistical metrics that might be to rigid to detect patterns. Patterns are often smoothed out when the data is stratified to too large segments. The used interval for segmentation is one complete day. One could debate on whether such a time interval is too broad or too narrow.

A second point of discussion regarding the quality assessment are the used thresholds that indicate if corruption was present in the dataset. An example is the minimum amount of correlation between the cumulative vehicle count and the green duration. We arbitrary set the minimum correlation at 0.80 to show expected control behavior. The threshold could also have ben set at 0.75, because no strong argumentation is present to set the threshold at 0.80. I think much can be gained by a comprehensive mechanism that can detect unexpected behavior in large amounts of V-log data.

The case study included a selection of an intersection that meets the design strategy for the developed algorithm. The strategy was to start with a simple geometry and a single modality and if possible add complexity to the model by testing the algorithm on other intersections. This set up was rather ambitious, because the development of the prediction model for a simple geometry and single modality needed all the available time. In collaboration with North-Holland we selected the intersection N245 Laan van Darmstadt as a case study. We naively decided to choose an intersection of which we taught to be a 'standalone' intersection and is not influenced by other surrounding intersections. It turned out that this intersection did have a synchronization connection with an intersection located upstream of the investigated intersection. Due to the synchronization link, the retrieved V-log data contained extremely extended green durations. These green durations occurred incidentally and therefore showed weak training patterns in the data. However, once the trained model was deployed on a test set, the test error amplified for four particular signal groups. Eventually we proposed an additional quality assessment metric for the assessment framework to detect such deviant patterns for future research. By accident we found out that the undetected synchronization link between the two vehicles gave us a comparable situation of signal cycles. Signal cycles that can be predicted accurately and signal cycles that cannot be predicted accurately. Chapter 9 gives an analysis of the prediction behavior of both signal cycles.

The validation of the model output is given by several error metrics in this project. During the project many lessons were learned about validating prediction models based on V-log data. Model results show that the standard k-fold cross validation is a poor assessment metric for approximating the true test error. During the validation of the green and red prediction model on the full intersection we noticed that averaged error metrics do not give enough information about the behavior of the prediction. We are primarily interested in achieving the lowest prediction error, but the model also should produce prediction output that matches expected control behavior. So besides standard validation metrics that summarize the model output, also the expected behavior of the controller output was compared to the prediction output. Previous research related to the prediction of switching timings are mainly focused on reporting averaged error metrics such as the MAE and RMSE without describing the set up of the validation procedure.

10.5 Implementation and future perspectives

The transition of the current level of autnomous driving towards fully autnomous driving is not clear and not well defined yet. It is dificult to do so, because the transition is influenced by aspects such as involvement of regularisation, standardization of protocols, evolving or lack of compatable technology at the infrastructure side and produced added value by self-driving vehicle technology versus willingness to pay by the consumer (KPMG, 2015). The developed prediction algorithm in this project serves as intermediate technology that bridges the gap between current level of automation and fully autnomous driving. However, some assumptions were made during the design process that could effect the implementation of the algorithm. For example, the assumption was made that there is no communication between vehicles as well as communication back from the vehicle to the traffic light controller. Once communication between vehicles and a two-sided communication between the vehicle and the traffic light controller is realized (full connectivity), the execution of traffic light control will probably be totally different as stated in this project. For example, once the vehicles are submitting their location real time towards the controller at a high frequency starting from a considerable large distance from the intersection, the controller is able to estimate future switching timings by filling in upcoming blocks of signal groups that will get a green signal. In such a case the future traffic demand is no longer a black box for the traffic light controller and the controller is could be able to deterministically predict its future switching timings. Possibly the deterministic prediction will be more accurate compared to the SVR algorithm and therefore makes the use of the SVR algrotihm insignificant for predicting the switching timings. The SVR algorithm still could be used if the loop detector features are replaced by features that give information about the number of expected vehicles during a given time span. An advantage for SVR in the situation of full connecitvity could be that the SVR is able to predict the switching timings for a larger prediction horizon compared to a deterministic approach. In case of fully autonomous vehicles the traffic light controller will probably have a totally different control logic in which individually vehicles or groups of vehicles will be coordinated to enter the intersection.

Secondly, it is assumed that the algorithm has access to rich and clean V-log data. In the previous section is described that the V-log data in its current structure and quality is not usable for supporting robust technology. The question arises which stakeholder will take responsibility for installing a prediction algorithm, guarantees access to V-log and performs quality checks on the data and the prediction performances? Interfacing and implementation of the predicting application is considered to be future research (see Chapter 11). Below a brief overview is given what the pros (+) and cons (-) are for each stakeholder that could take responsibility for the implementation of the prediction algorithm:

Automotive manufacturer: Nissan

+ Ability to influence the speed and quality of the technology advances to a greater extend once Nissan is implementing the prediction algorithm itself. However, Nissan will be relying on the access to data from road authorities.

+ Ability to align in-car technology with infrastructure systems in an early stage.

- V-log data is a data standard only used in the Netherlands and prediction circumstances will be different in other countries for which the prediction algorithm needs to be compatible. At this moment it is unclear what the differences are in technology advances of V-log from other countries.

- Low quality of V-log makes it challenging to bring the prediction algorithm into production. Nissan has limited influence on securing high quality V-log.

Road authority: North-Holland

+ Direct access to the software and hardware of the controller and large amounts of V-log for prediction optimization purposes.
+ Ability to secure a higher level of V-log quality and a higher level of prediction accuracy by adjusting the software and hardware of the controller. Ability to revise contracts with supplier of traffic light controllers to account for high quality V-log data.
+ The implementation of the prediction algorithm is more efficient when it is located at the controller side instead of in the vehicle. The prediction algorithm makes a prediction that can be used to inform multiple vehicles once it is installed at the controller side.
+ Uniformity of prediction information from the controller towards all approaching vehicles could lead to a higher safety around intersections by avoiding conflicting prediction situations.
- Speed of technology advances need to be aligned by the speed of technology advances of the automotive industry. Road authorities do not have commercial interests for speeding up technology advances.

KPMG mentioned in a report from 2015 that the annual economic benefit of connected and autonomous vehicles will grow to £51 billion (approximately 77 billion US dollar) for the UK alone by 2030 in which service providers will enter the market (KPMG, 2015). Connectivity of the vehicle with other vehicles, devices and infrastructure systems is seen as one of the major drivers behind the business concept 'vehicles as a service'. The implementation of a prediction algorithm as mentioned in this project could be implemented by a third party that commercializes both V-log data and the prediction information. Liability of the connectivity services and the provided information will be the biggest barrier for these service providers. A framework for determining liability on the transition of control from the vehicle to the driver of semi- automated technology would provide clarity including the application of current civil and criminal law (KPMG, 2015). In conclusion, the implementation and interfacing of the prediction application is probably the most efficient when it is executed by the road authority. The implementation and interfacing of the prediction algorithm is considered to be further research.

11 Conclusions and recommendations

In 2014 the province of North-Holland and Nissan Research Center Silicon Valley started a joint project called Traffic Signal Timings Predictions. Central to the joint project is the development of a prediction model that is able to predict the switching timings of vehicle-actuated controlled intersections. This project is performed as part of the joint project between Nissan and North Holland. The research question is formulated as following:

How to predict the traffic signal timings of a vehicle actuated controlled intersection as accurately as possible for in-vehicle usage, utilizing data retrieved from the traffic light controller?

In this Chapter the main conclusions and recommendations resulting from this project are discussed. First, several conclusions are drawn regarding the sub-questions as formulated for this project. Secondly, recommendations are given for Nissan on the way forward after this project. Additional, recommendations are given for anyone who is interested to work with V-log data. Finally, recommendations for further research are presented.

11.1 Overall conclusion

This project has shown that the developed Support Vector Regression prediction algorithm is capable of accurately predicting the switching timings of a (fully) vehicle-actuated controlled intersection serving a single modality. The best performing green prediction model reports a mean absolute error of 0.85 seconds and a hitrate of 99%. The mean absolute error indicates the difference between the true switching timing and the predicted switching timing, while the hitrate gives the percentage of predicted switching timings within a 4 seconds permissible error in terms of absolute deviance from the true switching timing. The best performing red prediction model shows a mean absolute error of 2.42 seconds and a hitrate of 81%. Improvements in the prediction performance can be gained by numerical optimization of the trained models. Due to time constraints in this project the optimization is not performed and is therefore recommended for further research.

Besides the reported validation metrics, the model results show that the SVR is capable of predicting expected control behavior. The SVR models predict for every second in the cycle the amount of seconds left until the moment of switching. The expected control behavior is that the amount of predicted seconds until the moment of switching follow a countdown behavior. Model results show that the SVR is able to predict such countdown behavior and therefore matches the expected control logic. The algorithmic approach of the Support Vector Regression (SVR) is at the heart of the prediction algorithm and shows potential to be easily deployed on other intersections that are independent of the intersection lay-out and different times of the day.

Although the SVR prediction algorithm is capable of accurately predicting the switching timings, the prediction algorithm lacks robustness for being used as operational technology. The lack of robustness is caused by insufficient quality of V-log data.

11.2 Conclusions regarding the sub-questions

The research question has been further specified using sub-questions. The sub-questions represent the structure of the project and therefore to a large extend correspond to the chapters in this report. The sub-questions guided the process of gaining knowledge and insights that were needed throughout the development of the prediction algorithm.

- *What functional and non-functional requirements need to be taken into account for the development of a prediction algorithm that is able to predict the switching timings of a vehicle-actuated controller?*

The following requirement were formulated: the prediction algorithm should predict the switching timings of both the red and green signal. The design of the prediction algorithm has led to two prediction models: one prediction model for the switching timings of the green signal and a second model for the prediction of the switching timings of the red signal. The target variable in both models is the absolute second in the signal phase on which the signal changes. The permissible prediction error for both the predicted green and red switching timings in terms of absolute deviance is 4 seconds (predefined by Nissan). The prediction algorithm is primarily designed for the prediction of vehicle-actuated controllers and could be used as part of a GLOSA system. The algorithm is designed to be scalable towards intersections that serve a single modality. The prediction algorithm makes use of data retrieved from the same controller for which it predicts the switching timings. Data transmission is needed between the controller and the vehicle by using an intermediate hub such as a cloud structure. The data transmission, which is needed for the algorithm, follows a single directed path from the controller to the vehicle. No communication back to the controller and no communication between other vehicles is necessary to predict the switching timings. The prediction algorithm could be deployed and computed in either the vehicle or the cloud structure that is used for the data transmission.

- *What information regarding the control logic of the vehicle-actuated controller is necessary to take into account for retrieving V-log data that is used for predicting the switching timings?*

Theoretically the vehicle-actuated controller controls its output signal for sets of signal groups at the same time. However, the controller has a high level of flexibility and due to the flexibility the controller practically controls each signal group individually. This has led to the development of an algorithm that is capable of predicting each signal group individually. The labelling of the signal groups is important for retrieving the correct V-log data from the controller for both training and prediction purposes.

The vehicle-actuated controller consists of two main control levels, which describe how the controller regulates its output signals based on the input signals. Both control levels use detector loop data as input data to control the traffic flow at the intersection. For both the green and the red prediction model, the model results show that data retrieved from the first and third detector loops are successfully used for predicting the switching timings. Data from both detector loops are injected as individual features into the prediction algorithm. Dominant detector loop data for the prediction of switching timings of the red signal is the so-called request signal. The request signal is logged by the controller when a vehicle is detected by the detector loop during a red phase.

Besides the detector loop data, the model results indicate that data describing the 7 internal states of the signal cycle, which are logged by the controller, are highly relevant for prediction purposes. The duration and switching moment of a signal phase is predominantly described by the signal group completion level.

The order of signal groups that get assigned a green phase is predominantly determined in the block procedure level of the controller. Both the red and green prediction model rely heavily on data that describe the block procedure level. Example of such data features are: signal cycle length and the duration of a block cycle. Results of the developed models showed that data features containing historical output signals, such as the previous red and green duration of the 5 consecutive cycles, do not have a large influence for predicting future switching timings.

- *What type of prediction framework complies to the functional and non-functional requirements for a prediction algorithm to predict the switching timings of a vehicle-actuated controller?*

A benchmark of several mathematical approaches that were used in previous studies show that the Support Vector Regression (SVR) approach has the biggest potential in predicting the switching timings of the vehicle-actuated controller. The biggest advantage of the SVR approach is its fast computation time, scalability towards other intersections and the use of non-linear input data. The SVR technique matches the requirement to produce a numerical output that describes the switching timing in absolute seconds. An important feature of the SVR approach is the use of a so-called kernel trick. The SVR enables the model developer to choice a kernel that is unique to the structure of V-log data. By doing so the SVR deals with the non-linear characteristics of the data. Both the Laplace and polynomial kernels turned out to be numerically a good fit for uncovering discriminative patterns within the V-log data in higher dimensions.

Furthermore, the Support Vector Regression prediction performance is not influenced directly by the dimensionality of the input space and therefore less prone to overfitting. As a results, the SVR approach can be easily extended with additional sources of data that might be needed to predict the switching timings.

- *What preprocessing steps need to be performed to transform V-log data to data that can directly be injected into the prediction algorithm?*

The raw V-log data cannot directly be injected into a SVR model. The preprocessing of raw V-log data to useable training and test data follows a fixed sequential order of steps. Preprocessing of the raw V-log include the actions: manual feature extraction, dataset construction, quality assessment of the data, feature selection, scaling and subsetting into train- and test datasets. The final output of the data preprocessing is useable training and test data for which the amount of corrupted data is reduced significantly. The manual feature extraction results into 112 features for each signal group extracted from the raw V-log data. The features are categorized in three categories: loop detector features, controller output state features and internal signal state features. The key philosophy during the feature extraction is to create features that show numerical variance that is needed to discriminate learning patterns in the data. Raw V-log data is logged in a binary fashion that is not suited for training the SVR.

The SVR algorithm relies heavily on clean training data. Multiple experiments show that poor loop detector health cause corruption in the retrieved dataset and is therefore a huge threat for accurately predicting the switching timings. To secure a clean and representative dataset a quality assessment procedure is developed in this project. The assessment was performed by using standard statistical metrics on stratified segments of the preprocessed V-log data. By following the proposed quality assessment procedure unexpected behavior in the data such as extremely extended green phases or jittering loop detectors can be detected prior to training the SVR model.

Feature selection was performed in the context of model selection. Model results demonstrate that the green prediction model only uses features selected from the signal group for which it also predicts the switching timing of the green signal. The red prediction model uses features selected from all signal groups to predict the switching timing of the red signal. During the development of the prediction algorithm we saw the impact of different scaling techniques on the data. We propose scaling as shown in standard score values or Z-values.

- *Under what circumstances does the prediction model performs satisfying and under what circumstances does the prediction model fail?*

The prediction performance of the SVR algorithm is directly related to the quality and size of the data on which the SVR is trained. Quality of the data describes the presence of missing values and artificial noise due to software or hardware failures of the controller. Model results show that noisy and corrupted data is usually caused by jittering detector loops. Jittering detector loops do not cause missing values by default and can therefore only be detected by statistical inference on the V-log data. In case of missing values, the SVR runs directly into trouble. Poor health of the loop detectors can cause low prediction accuracies. The most ideal situation is the availability of complete and clean loop detector data produced by detector loops that do not show hardware or software failures. Besides the quality and size of the data also representativeness of the data is necessary to obtain an accurate prediction of the signal timings. The SVR is only able to generalize learned patterns from training data to new input data if the training data is representative for expected control behavior in the new input data. No hard rules can be stated about how often certain patterns need to occur in the training data before the SVR is able to generalize the patterns to new data. Model results indicate that at least data from 5 weekdays is necessary to train the SVR model for one signal group to predict data coming from a random other weekday. Less available data will lead to a decrease in prediction accuracy. More available data does not harm the prediction performance, but is not necessarily required. Besides the amount and quality of the data, also the content of the data is relevant. Model results show that not only loop detector data is used for training the models, but also data describing the internal states of the controller is used for training by the prediction models.

Furthermore, empirical results show that the prediction algorithm performs poorly for several signal groups who are influenced by a synchronization of an upstream intersection. We assume that poorly predicted switching timings due to synchronization can be easily solved by including data concerning the synchronization into the prediction model. The incomplete data due to synchronization shows the importance of a comprehensive dataset that includes all possible expected behavior of the controller. In most ideal situation road authorities supply the modeler for each intersection the expected behavior by giving information about the parameter settings, synchronization connection etc.

- *How applicable is the prediction algorithm on different scenario's such as different intersections, multiple modalities and times of the day?*

An advantage of the developed algorithm is its easy scalability to other vehicle-actuated controlled intersections that serve a single modality independent of its intersection layout. The scalability is caused by two elements: firstly, the algorithm is designed to produce a prediction model for each signal group individually instead of the intersection as a whole system. As a result, the algorithm makes no distinction between intersections that consist of a different amount of signal groups. Secondly, the prediction algorithm uses features that can be extracted from V-log data from intersections that serve a single modality. No feature labels that are unique to a particular intersection are used for training and prediction. Before the prediction model is able to predict a new intersection the prediction model should be trained on data coming from that intersection.

A second advantage of the prediction algorithm is its transferability to different modalities. The SVR allows the input space of the training data to be expanded without excessively harming the prediction performance. It is possible that due to the inclusion of data from other sources a different kernel should be selected. Generally taken, the Laplace and polynomial kernel can be used for any further prediction model that includes V-log data and the SVR technique.

Furthermore, it is expected that the prediction algorithm is able to train and predict switching timings on other times of the day. The training process for other times of the day (afternoon, evening and night) is similar compared to the morning.

11.3 Recommendations for Nissan

Several recommendations for Nissan can be made based on the outcome of this project. The recommendations concern a number of additional steps that are recommended if Nissan would like to implement the prediction algorithm as operational technology and if Nissan would like to optimize current prediction algorithm.

The first recommendation relates to optimization of the current prediction algorithm. The parameters of prediction algorithms are not yet extensively optimized due to time constraints in this project. The SVR approach leaves room for optimization of several regularization parameters and hyperparameters. Preliminary model results showed significant decreases in prediction error after simple optimization of the regularization parameters. Optimization could be performed by using grid search on a combination of parameter settings for both the regularization and hyperparameters. The process of optimization is described in Chapter 8.

The second recommendation is to rewrite the base code to another programming language. Transferring the base code to another programming language is both useful for the optimization process and also useful for further integration of the prediction algorithm as operational technology. Current scripts are coded in the R programming language that effectively served the purpose of testing the POC (proof of concept). Numerous tricks were used to speed up computation time in the R-programming language. However, once optimization schemes are tested such as a full grid-search, it is recommended to lower the computation time by using faster programming languages. Recommended programming languages are: C++ or Python. For integration purposes rewriting the code to the recommended programming languages will make it easier to integrate with other autonomous vehicle software.

Thirdly, it is recommended to retrieve a list of deviant parameter settings and synchronization connections of intersections before training the prediction model on intersections in further projects. Knowing expected control behavior before validation of the prediction models significantly increases the success of the model selection. Unexpected behavior of the controller might lead to the inclusion of specific features that could describe the unexpected controller behavior. An example is the synchronization connection between two intersections in the case study that could have been predicted with data regarding the synchronization. It is not preferable for Nissan to do the data plumbing of the V-log data. During this project numerous amount of times expert knowledge from control engineers of the Province of North-Holland was needed to preprocess the data or troubleshoot mistakes in the data.

The fourth recommendation relates to testing the algorithm for different intersection geometries, other times of the day and intersections serving multiple modalities. In this project is demonstrated that the developed prediction algorithm is able to predict the switching timings during the morning peak of intersections serving a single modality. The SVR approach allows to add complexity to the prediction algorithm without directly harming the prediction performance. It is expected that different geometries and different times of the day will not lead to significant changes in the model structure and its prediction performance. Adding multiple modalities to the prediction model will likely lead to the extraction of additional features from the V-log whereas a new featureset selection procedure needs to be designed.

A fifth recommendation is related to bringing the developed algorithm to the current simulation environment that is located in the test lab of Nissan Research Center Silicon Valley. By doing so, the developed algorithm could be tested as operational technology. At this moment the simulation environment only contains maps that include US area's. The current developed algorithm is tested for Dutch vehicle-actuated controllers. It is recommended to investigate the performance of the developed algorithm on data coming from US controllers. We assume that the Laplace and polynomial kernel will still reveal discriminative patterns in high dimensions for V-log data from US controllers. The basic mechanism of loop detector input, internal states and output signals will be comparable with Dutch vehicle-actuated controllers.

11.4 Recommendations with regards to the use of V-log

This project shows that V-log data has potential to give explanatory and predictive power about the situation on and around intersections. Self-driving vehicle technology can benefit from knowledge forthcoming of V-log data and the automotive industry is likely to utilize V-log data to a greater extend in the near future. The developed algorithm for predicting the switching timings is a proof of concept of the usage of V-log. However, V-log data does have drawbacks that add complexity to any developed model that uses V-log. The following recommendations are given to anybody who is interested in using V-log for explanatory or predictive technology.

First, the V-log data comes in a data extension that is not useful for statistical models. To our knowledge the V-log data format is logged in a data extension that cannot be injected in any statistical software. During this project V-log was transformed to .CSV-extension before it was used for further preprocessing.

Secondly, the raw V-log data is logged in a binary fashion following a Discrete Event Simulation analogy (DEVS). Since the data consists primary of binary values, the initial features for prediction purposes show little numerical variance. Any statistical learning model will detect training patterns more effective when features are measured on a numerical output scale containing variance. In this project we manually created these features and tried to include any possible predictive behavior of the controller in those features. For further research, we strongly recommend to pay close attention to what type of features should be designed and whether they truly catch the expected behavior of the controller that is initially logged in a binary fashion.

A third recommendation for further use of V-log for prediction purposes is the use of detector loop data and gap times. For the development of the prediction algorithm data streams of the first and third detector loops were used for each signal group. The relative frequency of selected data of those two detector loops show that the detector loops are almost as equally important. A confounding factor can be designed that explains the behavior of the two detector loops and gap times of the detector loops into one factor. In current prediction models the cumulative vehicle count is used as a predictor, but a counted vehicle does not necessary lead to an extension of the green signal.

A fourth recommendation is the level of time aggregation for which the V-log data is logged. The raw V-log data is logged at an interval of 1/10th of a second, approaching a real-time updating scheme. During the development of the prediction model we did not encounter a significant drop in the prediction accuracy when a dataset was constructed representing data at an interval of 1 second. Using an interval larger then 1 seconds gave a decrease in prediction performance. For predicting the switching timings there is no added value to log the data at 1/10th of a second instead of 1 second time interval.

The fifth recommendation is related to the validation of the prediction algorithm. The model validation shows that standard k-fold cross validation gives an overly optimistic estimation of the true test error. For any further research involving V-log data and a statistical learning technique it is recommended to use a stratified cross validation. The stratified cross validation contains folds of complete time segments or complete signal cycles for a signal group. More information about problems that may arise by using standard k-fold cross validation is given in Chapter 8.

The sixth and final recommendation is the use of a quality assessment framework. The described preprocessing procedure in this report stresses the importance of clean and complete data that does not contain corrupted patterns. Securing a V-log dataset that complies to a high quality standard is rather difficult. We noticed that problems in the V-log data are primarily caused by the poor health of loop detectors. Both hardware and software failures cause the V-log data to be incomplete and corrupted. The designed algorithm asks for data coming from two loop detectors from each signal group. The intersection from the case study contains 10 signal groups. The amount of loop detectors from which data is retrieved, adds up to 20. We strongly recommend to check thoroughly of the selected intersection, from which V-log data is retrieved, does not show hardware and software failures for each of the signal groups. After a hardware and software check is performed a quality assessment, which is based on statistical metrics, should be used to double-check for corruption.

11.5 Further Research

A number of recommendations for further research results from this project. These recommendations particularly concern the overall research approach and methodology, as well as the development of the prediction algorithm in this project.

This project shows that statistical learning approaches such as the Support Vector Regression are useful for analyzing and predicting traffic light controllers. Besides statistical learning techniques also standard statistical techniques are useful in combination with V-log data. In this report we propose a quality assessment procedure that includes standard statistical metrics to assess the quality of the used data coming from the traffic light controller. The quality assessment is not fully comprehensive and is recommended as subject for further research.

A second recommendation for further research is to design a procedure/mechanism that is capable of curing inefficiencies in loop detector data. Often data is missing or the data is corrupted by jittering behavior of the detector loops. To design robust prediction models, it would be recommended to develop a system that could prevent or smoothens out the influence of corrupted loop detector data that shows such inefficiencies.

A third recommendation regarding the development of the prediction algorithm is to research the influence of arrival patterns of vehicles on the prediction performance. The developed algorithm takes cumulative vehicles into account as primarily explanation for the extension of the green signal. By theory we do know that arrival patterns and driving behavior effects the extension of the green signal. Partly this is measured by the gap times for each detector loop. The key question is how information regarding the gap times can be retrieved from the V-log data and designed into a feature.

A fourth recommendation for further research that possibly could improve the prediction performance is the use of queue length estimators. Queue length estimators are predictive models that are capable of predicting the amount of vehicles that will be waiting for each lane. The predicted queue could indicate to what extend the duration of the blocks will be utilized for particular signal groups. Such information could be used as an additional source of information for the developed prediction algorithm.

Finally, the interfacing and implementation of a prediction application that utilizes the prediction algorithm is recommended for further research. Examples of topics are data transmission and data latency of V-log, the installation of the prediction algorithm in the traffic light controller or in the vehicle and the frequency of retraining the prediction model respecting the chosen prediction application.

12 Bibliography

Apple, J., Chang, P., Clauson, A., Dixon, H., Fakhoury, H., Ginsberg, M. L., & Smith, B. (2011). Green Driver: AI in a microcosm. In *twenty-fifth AAAI conference on artificial intelligence*, San Fransisco, CA. Abstract retrieved from http://www.aaai.org/ocs/index.php/AAAI/AAAI11/paper/view/3648/4094

Asadi, B., & Vahidi, A. (2011). Predictive Cruise Control: Utilizing upcoming traffic signal information for improving fuel economy and reducing trip time. *Control Systems Technology, IEEE Transactions on*, *19*(3), 707-714.

Barthauer, M., & Friedrich, B. (2014). Evaluation of a signal state prediction algorithm for car to infrastructure applications. *Transportation Research Procedia*, *3*, 982-991.

Braun, R., Busch, F., Kemper, C., Hildebrandt, R., Weichenmeier, F., Menig, C. & Preßlein-Lehle, R. (2009). TRAVOLUTION–Netzweite optimierung der lichtsignalsteuerung und LSA-fahrzeug-kommunikation. *Straßenverkehrstechnik*, *53*, 365-374.

Bodenheimer, R., Brauer, A., Eckhoff, D., & German, R. (2014). Enabling GLOSA for adaptive traffic lights. In *Vehicular Networking Conference (VNC), 2014,* 167-174.

CAWCR. (2015). *Centre for Australian Weather and Climate Research, Forecast Verification*. Retrieved May 2th, 2015, from CAWCR: http://www.cawcr.gov.au/projects/verification/#What_makes_a_forecast_good

Cawley, G. C., & Talbot, N. L. (2010). On over-fitting in model selection and subsequent selection bias in performance evaluation. *The Journal of Machine Learning Research*, *11*, 2079-2107.

Chang, C. C., & Lin, C. B. (2002). Training v-support vector regression: theory and algorithms. *Neural Computation*, *14*(8), 1959-1977.

Eckhoff, D., Halmos, B., & German, R. (2013, December). Potentials and limitations of green light optimal speed advisory systems. In *Vehicular Networking Conference (VNC), 2013,* 103-110.

Fletcher, R. (2013). *Practical methods of optimization*. Chichester: John Wiley & Sons.

Fox, J. (2009). *Applied regression analysis and generalized linear models*. USA: Sage Publications.

Hastie, T., Friedman, J., & Tibshirani, R. (2009). *The elements of statistical learning* (Vol. 1). Berlin: Springer.

Highway Capacity Manual (2010). National Research Council, *Transportation Research Board*, Washington, D.C.

Hsu, C. W., Chang, C. C., & Lin, C. J. (2003). A practical guide to support vector classification. Retrieved June 6th, 2015, from National Taiwan University: http://www.csie.ntu.edu.tw/~cjlin

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2014). *An introduction to statistical learning* (Vol. 112). New York: Springer.

Jeffers, J. (1988). *Practitioner's handbook on the modelling of dynamic change in ecosystems*. Chichester: John Wiley & Sons.

Karatzoglou, A., Hornik, K., & Meyer, D. (2006). Support vector machines in R. *Journal of statistical software*, *15*(9), 1-28.

Katsaros, K., Kernchen, R., Dianati, M., & Rieck, D. (2011). Performance study of a Green Light Optimized Speed Advisory (GLOSA) application using an integrated cooperative ITS simulation platform. In *Wireless Communications and Mobile Computing Conference (IWCMC), 2011,* 918-923.

Koukoumidis, E., Martonosi, M., & Peh, L. S. (2012). Leveraging smartphone cameras for collaborative road advisories. *Mobile Computing, IEEE Transactions on*, *11*(5), 707-723.

KPMG (2014). *Self-driving cars.* Retrieved on August 12th, 2015, From KPMG:https://www.kpmg.com/US/en/IssuesAndInsights/ArticlesPublications/Documents/self-driving-cars-are-we-ready.pdf

KPMG (2015). *Connected and autonomous vehicles.* Retrieved on April 11th, 2016, From KPMG: https://www.kpmg.com/BR/en/Estudos_Analises/artigosepublicacoes/Documents/Industrias/Connected-Autonomous-Vehicles-Study.pdf

Krijger, P. (2013). *Master's thesis traffic light prediction for TomTom devices*. Retrieved on April 23th, 2015, from Kivi:https://www.kivi.nl/uploads/media/56434864d6330/PaulKrijgerMScThesis.pdf

Ledolter, J. (2013). *Data mining and business analytics with R*. Chichester: John Wiley & Sons.

Thacker, B., Doebling, S., Hemez, F., Anderson, M., Pepin, J., & Rodriguez, E. (2004). *Concepts of model verification and validation* (No. LA-14167). Los Alamos National Lab., Los Alamos, NM, US.

Mahler, G., & Vahidi, A. (2012). Reducing idling at red lights based on probabilistic prediction of traffic signal timings. In *American Control Conference (ACC), 2012*, 6557-6562.

Martonosi, M., Koukoumidis, E., & Peh, L. (2012). Leveraging smartphone cameras for collaborative road advisories. *Mobile Computing, IEEE Transactions on*, *11*(5), 707-723.

Matloff, N. (2011). *The art of R programming*. San Fransisco: No Starch Press.

Mitra, S., & Acharya, T. (2003). *Data mining: multimedia, soft computing, and bioinformatics*. Chichester: John Wiley & Sons.

Muller, T., & de Leeuw, M. (2006). New method to design traffic control programs. *Transportation Research Record: Journal of the Transportation Research Board*, 1978, 68-75.

Müller, K., Smola, A., Rätsch, G., Schölkopf, B., Kohlmorgen, J., & Vapnik, V. (1997). Predicting time series with support vector machines. In *Artificial Neural Networks—ICANN,* 999-1004.

Murphey, Y., Park, J., Kristinsson, J., McGee, R., Kuang, M., & Phillips, T. (2013). Intelligent speed profile prediction on urban traffic networks with machine learning. In *Neural Networks (IJCNN), 2013,* 1-7.

Negenborn, R. R. (2010). *Intelligent infrastructures* (Vol. 42). H. Hellendoorn, & Z. Lukszo (Eds.). Dordrecht: Springer.

NISSAN (2013). *Renault-Nissan alliance opens bigger Silicon Valley research center to enhance advanced research and development.* Retrieved March 16th, 2015, from Nissan news: http://nissannews.com/en-US/nissan/usa/releases/renault-nissan-alliance-opens-bigger-silicon-valley-research-center-to-enhance-advanced-research-and-development

NISSAN (2014a). *The autonomous vehicle*. Retrieved on March 12th, 2015, from Nissan news: http://www.nissanusa.com/blog/autonomous-drive-car

NISSAN (2014b). *Carlos Ghosn outlines launch timetable for Autonomous Drive technologies*. Retrieved April 15th, 2016, From Nissan news: http://nissannews.com/en-US/nissan/usa/releases/carlos-ghosn-outlines-launch-timetable-for-autonomous-drive-technologies

NISSAN. (2015a). *Nissan announces pricing for 2016*. Retrieved on March 16th, 2015, from Nissan news: http://nissannews.com/en-US/nissan/usa/releases/nissan-announces-pricing-for-2016-gt-r

NISSAN. (2015b). *Internal publication traffic signal timings prediction.* Sunnyvale, Silicon Valley, CA, US.

Protschky, V., Feit, S., & Linnhoff-Popien, C. (2014a). Extensive traffic light prediction under real-world conditions. In *Vehicular Technology Conference (VTC Fall), 2014,* 1-5.

Protschky, V., Wiesner, K., & Feit, S. (2014b). Adaptive traffic light prediction via Kalman Filtering. In *Intelligent vehicles Symposium Proceedings, 2014,* 151-157.

RStudio. (2016). *RStudio products*. Retrieved February 11th, 2016, from Rstudio: https://www.rstudio.com/products/rstudio/

SAE. (2015). *Automated driving, international standard J3016.* Retrieved April 27th, 2016, from SAE International: http://www.sae.org/misc/pdfs/automated_driving.pdf

Seredynski, M., Dorronsoro, B., & Khadraoui, D. (2013a). Comparison of green light optimal speed advisory approaches. In *Intelligent Transportation Systems-(ITSC), 2013,* 2187-2192.

Seredynski, M., Mazurczyk, W., & Khadraoui, D. (2013b). Multi-segment green light optimal speed advisory. In *Parallel and distributed processing symposium workshops & PhD forum (IPDPSW), 2013,* 459-465.

Smola, A. J., & Schölkopf, B. (2003). A tutorial on support vector regression. *Statistics and computing*, *14*(3), 199-222.

Tielert, T., Killat, M., Hartenstein, H., Luz, R., Hausberger, S., & Benz, T. (2010). The impact of traffic-light-to-vehicle communication on fuel consumption and emissions. In *Internet of Things (IOT), 2010*, 1-8.

Van Belle, V., & Lisboa, P. (2013). White box radial basis function classifiers with component selection for clinical prediction models. *Artificial intelligence in medicine*, *60*(1), 53-64.

Vanderbei, R. J. (1994). LOQO: An interior point code for quadratic programming. *Optimization methods and software*, *11*(1-4), 451-484.

Van Katwijk, R. T. (2008). *Multi-agent look-ahead traffic-adaptive control*. Delft: Delft University of Technology.

Vialis (2012). *V-Log protocol en definities*. Unpublished document, Vialis Traffic BV, Houten, The Netherlands.

Weisheit, T., & Hoyer, R. (2014). Prediction of switching times of traffic actuated signal controls using support vector machines. In *Advanced Microsystems for Automotive Applications 2014* (pp. 121-129). Springer

Wikipedia (2015). *Serialization*, Retrieved August 10, 2015, from Wikimedia Foundation: https://en.wikipedia.org/wiki/Serialization

Wilson, A., & De Groot, H. (2006). *Handboek verkeerslichtenregelingen* (Vol. 2006). Ede: CROW.

Zweck, M., & Schuch, M. (2013). Traffic light assistant: Applying cooperative ITS in European cities and vehicles. In *Connected Vehicles and Expo (ICCVE), 2013,* 509-513.

13 APPENDIX A: Overview of benchmarked prediction approaches

Requirements	Probability theory	Support Vector Machine/Regression	Kalman Filter	Markov-Chain	Neural Network
Switching time/ probability statement	Probability statement (Mahler & Vahidi, 2012)	Switching time (Weisheit & Hoyer, 2014)	Probability statement (Protschky et al., 2014b)	Switching time (Barthauer & Friedrich, 2014)	Switching time
Ability to predict the prediction horizon	Yes	Yes	Yes	Yes	Yes
Dealing with non-linearity of the controller	No	Yes	Yes	Yes	Yes
Max degree of control flexibility	Pre-timed (Krijger, 2013)	Vehicle-actuated (Koukoumidies et al., 2012)	Vehicle-actauted, but fixed cycle (Protschky et al., 2014b)	Vehicle-actauted, but fixed cycle (Barthauer & Friedrich, 2014)	Vehicle-actuated, confirmed by by empirical model results of prior research by Nissan
Utilization of V-log data	No, only uses conditional probabilities on given green phases (Mahler & Vahidi, 2012)	Yes (Weisheit & Hoyer, 2014)	No, uses only green frequency distributions (Protschky et al., 2014b)	No, is not able to utilize loop detector data (Bodenheimer, Brauer, Eckhoff, & German, 2014).	Yes, confirmed by empirical model results of prior research by Nissan
Scalability to other intersections and modalities	Not easy: problems with low traffic volumes. Needs a lot of conditional probabilities (Krijger, 2013)	Easy: new data can added to the model and less prone to overfitting compared to the ANN (Hsu et al., 2003)	Easy: but has problems with intersections including multiple modalities (Protschky et al., 2014b)	Not easy, needs different transition matrix for every new intersection (Barthauer & Friedrich, 2014).	Unknown: might be useable on other similar intersections but suffers fast from curse of dimensionality (Hastie et al., 2009)

Biggest advantage	Low computation time, mathematically understandable.	Prone to overfitting. A relative large amount of data can be presented to the model without harming the generalization error (Hastie et al., 2009)	Stable predictions under semi vehicle-actuated controller scenario (Protschky et al., 2014b)	Once transition graph and matrix are determined, model predicts fast and accurate (Barthauer & Friedrich, 2014)	No rules of the dynamics of the controller need to be specified (James et al., 2014)
Biggest disasdvantage	Sparseness of dataset lead to noninformative probability distributions (Krijger, 2013)	Parameter settings of the Kernel gets complex (Smola & Schölkopf, 2003)	Computational intense, does not show immediate changes in output (Protschky et al., 2014a)	Chance of insufficient data to calculate probabilities of transfer rate (Jeffers, 1988).	Difficult to interpret results due to mathematically complexity. Possible difficulties with local optimum (James et al., 2014)

14 APPENDIX B: Overview of manually extracted features

index	featurename	description
1	date	Date of the logged V-log data in the dataset
2	index	Row index
3	X10th_sec	1/10th of a second index, every row indicates 1/10 of a second
4	R_Time	Timer of current red signal
5	R_CD	Countdown of current red signal
6	R_Length	Absolute length in time of current red phase
7	R_Length1	Absolute length in time of one red phase before current red phase
8	R_Length2	Absolute length in time of the second red phase before current red phase
9	R_Length3	Absolute length in time of the third red phase before current red phase
10	R_Length4	Absolute length in time of the fourth red phase before current red phase
11	R_Length5	Absolute length in time of the fifth red phase before current red phase
12	G_Time	Timer of current green signal
13	G_CD	Countdown of current green signal
14	G_Length	Absolute length in time of current green phase
15	G_Length1	Absolute length in time of one green phase before current green phase
16	G_Length2	Absolute length in time of the second green phase before current green phase
17	G_Length3	Absolute length in time of the third green phase before current green phase
18	G_Length4	Absolute length in time of the fourth green phase before current green phase
19	G_Length5	Absolute length in time of the fifth green phase before current green phase
20	Y_Time	Timer of current yellow signal
21	Y_CD	Countdown of current yellow signal
22	Y_Length	Absolute length in time of current yellow phase
23	Block2Block_Time	Timer of current block and that same block in the next cycle
24	Block2Block_CD	Countdown of current block and that same block in the next cycle
25	Block2Block_Length	Absolute length in time of current block and that same block in the next cycle
26	Block_Time	Timer of start to end of current block
27	Block_CD	Countdown of start to end of current block
28	Block_Length	Absolute length in time of start to end of current block
29	InterCycleTime_cnt	The summation of green phases of all signal groups of a total cycle of the intersection. This variable gives the elapsed time within the summation of passed green phases
30	InterCycleTime_cnt1	InterCycleTime_cnt, but one cycle before current cycle

31	InterCycleTime_cnt2	InterCycleTime_cnt, but second cycle before current cycle
32	InterCycleTime_cnt3	InterCycleTime_cnt, but third cycle before current cycle
33	InterCycleTime_cnt4	InterCycleTime_cnt, but fourth cycle before current cycle
34	InterCycleTime_cnt5	InterCycleTime_cnt, but fifth cycle before current cycle
35	InterCycleNumb_cnt	Number of green phases that are summed in InterCycleTime_cnt
36	InterCycleNumb_cnt1	Number of green phases that are summed in InterCycleTime_cnt1
37	InterCycleNumb_cnt2	Number of green phases that are summed in InterCycleTime_cnt2
38	InterCycleNumb_cnt3	Number of green phases that are summed in InterCycleTime_cnt3
39	InterCycleNumb_cnt4	Number of green phases that are summed in InterCycleTime_cnt4
40	InterCycleNumb_cnt5	Number of green phases that are summed in InterCycleTime_cnt5
41	SCycle_Time	Timer of the start to end of cycle of that particular signal group
42	SCycle_CD	Countdown of the start to end of cycle of that particular signal group
43	SCycle_Length	Absolute length in time of the start to end of cycle of that particular signal group
44	AR_Value	Binary indication of alternative realization
45	AR_Time	Timer if AR_Value == 1
46	AR_CD	Countdown if AR_Value == 1
47	AR_Length	Absolute length in time of alternative realization from start to end if AR_Value == 1
48	Aan_Active	Binary indication of 'aanvraag' request parameter
49	Aan_Time	Timer if Aan_Active == 1
50	Aan_CD	Countdown if Aan_Active == 1
51	Aan_Length	Absolute length in time of request parameter from start to end if AR_Value == 1
52	CG_Value	Categorical value of CG value (realization method)
53	RA_Active	Binary indication of 'red before green' parameter
54	RA_Time	Timer if RA_Active == 1
55	RA_CD	Countdown if RA_Active == 1
56	RA_Length	Absolute length in time of start to end of RA_active parameter if FG_Active == 1
57	FG_Active	Binary indication when fixed green is active
58	FG_Time	Timer if FG_Active == 1
59	FG_CD	Countdown if FG_Active == 1
60	FG_Length	Absolute length in time of start to end of FG_active parameter is FG_active ==1
61	WG_Active	Binary indication when waiting green is active
62	WG_Time	Timer if WG_Active == 1
63	WG_CD	Countdown if WG_Active == 1
64	WG_Length	Absolute length in time of start to end of WG_active parameter is WG_active ==1
65	VG_Active	Binary indication when extending green is active
66	VG_Time	Timer if VG_Active == 1

67	VG_CD	Countdown if VG_Active == 1
68	VG_Length	Absolute length in time of start to end of VG_active parameter is VG_active ==1
69	MG_Active	Binary indication when prolonging green is active
70	MG_Time	Timer if MG_Active == 1
71	MG_CD	Countdown if MG_Active == 1
72	MG_Length	Absolute length in time of start to end of MG_active parameter is MG_active ==1
73	RV_Active	Binary indication of red before request parameter
74	RV_Time	Timer if RV_Active == 1
75	RV_CD	Countdown if RV_Active == 1
76	RV_Length	Absolute length in time of start to end of RV_active parameter is RV_active ==1
77	AtTdet1_MaxVol	Cumulative vehicle count at the first detector loop at current time
78	RDet1_MaxVol	Cumulative vehicle count at the first detector loop at current time when output signal is red
79	RDet1_MaxVol1	RDet1_MaxVol of one cycle before current cycle
80	RDet1_MaxVol2	RDet1_MaxVol of second cycle before current cycle
81	RDet1_MaxVol3	RDet1_MaxVol of third cycle before current cycle
82	RDet1_MaxVol4	RDet1_MaxVol of fourth cycle before current cycle
83	RDet1_MaxVol5	RDet1_MaxVol of fifth cycle before current cycle
84	GDet1_MaxVol	Cumulative vehicle count at the first detector loop at current time when output signal is green
85	GDet1_MaxVol1	GDet1_MaxVol of one cycle before current cycle
86	GDet1_MaxVol2	GDet1_MaxVol of second cycle before current cycle
87	GDet1_MaxVol3	GDet1_MaxVol of third cycle before current cycle
88	GDet1_MaxVol4	GDet1_MaxVol of fourth cycle before current cycle
89	GDet1_MaxVol5	GDet1_MaxVol of fifth cycle before current cycle
90	YDet1_MaxVol	Cumulative vehicle count at the first detector loop at current time when output signal is yellow
91	YDet1_MaxVol1	YDet1_MaxVol of one cycle before current cycle
92	YDet1_MaxVol2	YDet1_MaxVol of second cycle before current cycle
93	YDet1_MaxVol3	YDet1_MaxVol of third cycle before current cycle
94	YDet1_MaxVol4	YDet1_MaxVol of fourth cycle before current cycle
95	YDet1_MaxVol5	YDet1_MaxVol of fifth cycle before current cycle
96	AtTdet3_MaxVol	Cumulative vehicle count at the third detector loop at current time
97	RDet3_MaxVol	Cumulative vehicle count at the third detector loop at current time when output signal is red
98	RDet3_MaxVol1	RDet3_MaxVol of one cycle before current cycle
99	RDet3_MaxVol2	RDet3_MaxVol of second cycle before current cycle
100	RDet3_MaxVol3	RDet3_MaxVol of third cycle before current cycle
101	RDet3_MaxVol4	RDet3_MaxVol of fourth cycle before current cycle

102	RDet3_MaxVol5	RDet3_MaxVol of fifth cycle before current cycle
103	GDet3_MaxVol	Cumulative vehicle count at the third detector loop at current time when output signal is green
104	GDet3_MaxVol1	GDet3_MaxVol of one cycle before current cycle
105	GDet3_MaxVol2	GDet3_MaxVol of second cycle before current cycle
106	GDet3_MaxVol3	GDet3_MaxVol of third cycle before current cycle
107	GDet3_MaxVol4	GDet3_MaxVol of fourth cycle before current cycle
108	GDet3_MaxVol5	GDet3_MaxVol of fifth cycle before current cycle
109	YDet3_MaxVol	Cumulative vehicle count at the third detector loop at current time when output signal is yellow
110	YDet3_MaxVol1	YDet3_MaxVol of one cycle before current cycle
111	YDet3_MaxVol2	YDet3_MaxVol of second cycle before current cycle
112	YDet3_MaxVol3	YDet3_MaxVol of third cycle before current cycle
113	YDet3_MaxVol4	YDet3_MaxVol of fourth cycle before current cycle
114	YDet3_MaxVol5	YDet3_MaxVol of fifth cycle before current cycle

15 APPENDIX C: Output of the quality assessment

Day	Date	Correlation	Mean.Traffic	Max.Traffic	Sd.Traffic	Mean.Green	Max.Green	Sd.Green	Green.Phases	Block2Block
tuesday	1-apr	0.83	10.81	35	8.05	209.78	672	140.62	71	1017.01
wednesday	2-apr	0.85	9.14	37	7.25	192.47	643	134.71	77	936.57
thursday	3-apr	0.9	10.74	42	8.36	207.34	792	143.18	71	1008.39
friday	4-apr	0.84	9.03	29	6.78	192.16	644	127.54	75	957.97
saturday	5-apr	0.26	1.46	11	1.62	136.89	907	143.4	137	492.25
sunday	6-apr	0.6	1.15	9	1.58	318.29	2673	455.12	150	357.83
monday	7-apr	0.78	10.26	34	7.59	205.34	712	133.59	71	1013.73
tuesday	8-apr	0.82	10.81	32	7.93	210.87	742	135.45	68	1054.57
wednesday	9-apr	0.83	8.84	27	6.64	190.32	663	132.24	75	960.33
thursday	10-apr	0.84	10.6	39	8.05	204.53	643	134.38	70	1023.17
friday	11-apr	0.82	8.53	29	6.81	183.62	662	125.89	78	925.87
saturday	12-apr	0.34	1.62	10	1.68	126.13	665	113.74	142	480.37
sunday	13-apr	0.6	1.14	8	1.36	237.47	1946	310.53	155	401.21
monday	14-apr	0.83	10.11	32	7.42	209.49	653	137.86	71	1008.34
tuesday	15-apr	0.85	11.07	36	8.53	216.39	743	148.32	70	1030.17

wednesday	16-apr	0.9	9.47	40	7.38	189.62	643	127.15	76	945.83
thursday	17-apr	0.74	10.44	31	7.73	212.56	643	137.08	69	1035.01
friday	18-apr	0.89	6.34	25	5.57	156.62	653	117.8	89	809.09
saturday	19-apr	0.3	1.43	9	1.56	109.31	684	102.24	146	462.9
sunday	20-apr	0.68	1.33	6	1.33	429.18	2128	452.25	121	430.66
monday	21-apr	0.5	1.21	7	1.31	315.15	2087	396.65	147	390.56
tuesday	22-apr	0.89	10.44	30	7.78	203.9	694	136.1	72	1000.54
wednesday	23-apr	0.86	8.78	32	6.97	179.17	643	122.49	78	919.76
thursday	24-apr	0.86	10.44	31	7.66	203	719	137.24	72	1009.21
friday	25-apr	0.88	8.71	28	6.95	177.17	663	121.03	79	918.43
saturday	26-apr	0.55	1.05	5	0.99	142.95	1155	169.8	157	392.19
sunday	27-apr	0.52	1	6	1.1	240.77	1433	243.85	141	432.21
monday	28-apr	0.82	7.78	28	6.19	194.28	644	141.13	84	860.67
tuesday	29-apr	0.79	8.41	27	6.54	189.8	643	129.42	81	885.7
wednesday	30-apr	0.83	6.95	23	5.65	171.96	652	129.79	90	795.19

16 APPENDIX D: Output red prediction models

Model selection output Red prediction model for featureset 3:

Kernel	Traindays	Systimes	SVs	Min_pred	Hitrate	RMSE	MAE	Cross_error
polydot	1	22.03	3272	-357.30	0.34	10.96	7.33	9.07
laplacedot	1	7.24	2182	0.55	0.26	12.75	9.88	2.63
polydot	2	91.20	7181	-17.01	0.36	9.29	7.25	9.29
laplacedot	2	87.00	4786	0.97	0.24	10.53	8.49	2.52
polydot	5	685.20	17679	-20.85	0.36	9.53	7.42	9.11
laplacedot	5	668.40	12303	0.97	0.33	10.27	8.10	2.36
polydot	10	4068.00	35386	-18.04	0.34	9.51	7.41	9.05
laplacedot	10	3744.00	24675	0.43	0.35	9.74	7.55	2.35
polydot	15	16272.00	52786	-19.47	0.34	9.54	7.46	9.05
laplacedot	15	5832.00	38222	0.64	0.38	9.65	7.29	2.36

Table 8 Model selection output for Red prediction model - Featureset 3 - Signal group 8

Model selection output Red prediction model for featureset 6:

Kernel	Traindays	Systimes	SVs	Min_pred	Hitrate	RMSE	MAE	Cross_error
polydot	1	28.82	3202	-7.17	0.42	8.18	6.26	7.35
laplacedot	1	9.75	2511	1.26	0.43	9.55	7.00	3.01
polydot	2	124.20	6547	-2.03	0.43	7.78	5.98	7.46
laplacedot	2	124.80	4951	0.63	0.42	7.90	6.05	2.97
polydot	5	1279.80	16413	-1.70	0.44	7.56	5.81	7.48
laplacedot	5	876.00	12762	0.02	0.49	7.05	5.32	3.08
polydot	10	10296.00	33084	-2.14	0.44	7.49	5.77	7.51
laplacedot	10	4932.00	25888	-0.06	0.53	6.65	4.93	3.10
polydot	15	32796.00	49216	-2.07	0.45	7.47	5.72	7.49
laplacedot	15	8244.00	38380	-0.45	0.54	6.45	4.78	3.16

Table 9 Model selection output for Red prediction model - Featureset 6 - Signal group 8

17 APPENDIX E: Output tables green and red prediction models on full intersection

The table below contains the model output of the Green prediction model on the full intersection *N245 Laan van Darmstadt.*

Kernel	Signal group	Systimes	SVs	Min_pred	Hitrate	RMSE	MAE	Cross_error	Train_error
polydot	1	656.57	22896	-2.73	0.14	26.25	19.76	29.25	0.82
laplacedot	1	934.94	20122	-0.18	0.16	28.37	21.34	19.92	0.28
polydot	2	151.70	12997	-0.15	0.25	15.95	11.46	14.64	0.94
laplacedot	2	274.53	11231	0.40	0.22	16.61	12.26	10.13	0.27
polydot	3	1262.16	4088	-0.71	0.84	2.94	2.14	3.19	0.40
laplacedot	3	811.30	3997	0.66	0.81	3.11	2.35	2.97	0.19
polydot	4	68.80	8205	-1.71	0.78	3.31	2.68	4.09	0.59
laplacedot	4	118.19	7666	0.45	0.77	3.40	2.73	3.51	0.21
polydot	5	510.79	2386	-2.01	0.99	1.29	0.97	1.39	0.21
laplacedot	5	253.96	2073	0.33	0.99	1.25	0.85	1.30	0.10
polydot	7	246.80	15816	-10.34	0.26	14.70	11.19	15.21	0.81
laplacedot	7	401.40	14465	-0.72	0.29	15.82	11.20	12.74	0.40
polydot	8	114.02	11348	-10.53	0.37	10.22	7.74	10.00	0.57
laplacedot	8	156.06	10968	0.52	0.40	10.44	7.67	9.63	0.40
polydot	9	195.32	1412	0.51	0.58	7.77	5.26	6.61	0.93
laplacedot	9	97.19	1177	1.84	0.60	7.34	5.02	3.85	0.14
polydot	10	81.91	9515	-2.63	0.50	7.97	5.62	7.73	0.85
laplacedot	10	97.46	8809	0.97	0.54	7.54	5.12	6.87	0.53
polydot	11	1841.55	4816	-3.10	0.90	3.03	2.20	3.21	0.28
laplacedot	11	2329.17	4623	0.69	0.82	3.31	2.49	2.77	0.07

Table 10 Green prediction model output on full intersection

The table below contains the model output of the Red prediction model on the full intersection *N245 Laan van Darmstadt.*

Kernel	**Signal group**	**Systimes**	**SVs**	**Min_pred**	**Hitrate**	**RMSE**	**MAE**	**Cross_error**	**Train_error**
polydot	1	7107.49	6314	-3.63	0.75	4.59	3.00	5.39	0.52
laplacedot	1	5219.29	3628	0.53	0.81	4.45	2.42	2.38	0.03
polydot	2	1137.00	13151	-8.66	0.65	11.12	5.84	10.27	0.47
laplacedot	2	649.38	9775	-1.24	0.70	9.17	4.79	3.40	0.02
polydot	3	2355.61	24033	-24.04	0.20	20.41	13.86	25.21	0.59
laplacedot	3	2215.16	21322	-2.08	0.23	22.66	15.44	13.36	0.09
polydot	4	1668.01	20941	-15.54	0.42	9.89	7.09	10.50	0.47
laplacedot	4	4645.37	15427	-0.52	0.56	8.65	5.65	4.79	0.03
polydot	5	5435.36	20379	-4.29	0.34	29.05	14.42	35.18	0.71
laplacedot	5	1803.74	16462	-5.03	0.37	21.86	11.99	14.03	0.06
polydot	7	667.63	11416	-3.34	0.73	5.68	3.53	6.42	0.18
laplacedot	7	606.46	8969	0.27	0.73	4.75	3.27	2.44	0.01
polydot	8	1347.60	14981	-2.57	0.54	7.08	4.99	7.02	0.13
laplacedot	8	880.24	11966	-0.08	0.57	5.65	4.29	2.85	0.01
polydot	9	2626.25	28318	-58.27	0.02	185.24	127.44	195.67	0.85
laplacedot	9	483.71	5834	19.62	0.02	182.98	133.26	20.97	0.01
polydot	10	933.54	20137	-6.15	0.29	15.93	11.79	15.29	0.62
laplacedot	10	1253.76	17481	-2.92	0.35	16.35	11.38	9.68	0.11
polydot	11	2539.03	23828	-18.61	0.26	12.71	9.88	13.43	0.26
laplacedot	11	1807.74	17130	-1.49	0.45	11.53	7.90	5.14	0.01

Table 11 Red prediction model output on full intersection

18 APPENDIX F: Preference of features for the green and red prediction models

Table 12 gives the frequencies of selected features by the Red prediction model fitted on a full intersection (intersection N245 Laan van Darmstadt).

Feature	Frequency count	Relative Frequency
Block2Block_Time	30	0.152
R_Time	29	0.147
Block_Time	28	0.142
Aan_Time	25	0.127
SCycle_Time	24	0.122
AtTdet3_MaxVol	11	0.056
G_Time	8	0.041
RA_Time	7	0.036
AtTdet1_MaxVol	5	0.025
InterCycleTime_cnt1	4	0.020
InterCycleTime_cnt2	3	0.015
RDet1_MaxVol	3	0.015
InterCycleTime_cnt5	2	0.010
MG_Time	2	0.010
R_Length4	2	0.010
R_Length5	2	0.010
FG_Time	1	0.005
GDet1_MaxVol4	1	0.005
GDet1_MaxVol5	1	0.005
GDet3_MaxVol	1	0.005
InterCycleTime_cnt3	1	0.005
R_Length1	1	0.005
R_Length2	1	0.005
R_Length3	1	0.005
RDet3_MaxVol1	1	0.005
RDet3_MaxVol3	1	0.005
RDet3_MaxVol5	1	0.005
VG_Time	1	0.005

Table 12 Overview of selected features by the Red prediction model on a full intersection

18.1 Frequency of green features

Table 13 gives the frequencies of selected features by the Green prediction model fitted on a full intersection (intersection N245 Laan van Darmstadt).

Feature	Frequency count	Relative Frequency
AtTdet1_MaxVol	9	0.098
G_Time	9	0.098
SCycle_Time	9	0.098
AtTdet3_MaxVol	8	0.087
Block_Time	8	0.087
Block2Block_Time	8	0.087
FG_Time	7	0.076
MG_Time	7	0.076
RDet3_MaxVol	5	0.054
InterCycleTime_cnt1	3	0.033
R_Length1	3	0.033
VG_Time	3	0.033
G_Length1	1	0.011
G_Length5	1	0.011
GDet1_MaxVol2	1	0.011
GDet3_MaxVol2	1	0.011
InterCycleTime_cnt2	1	0.011
InterCycleTime_cnt3	1	0.011
InterCycleTime_cnt4	1	0.011
R_Length2	1	0.011
R_Length4	1	0.011
R_Time	1	0.011
RDet1_MaxVol1	1	0.011

Table 13 Overview of selected features by the Green prediction model on a full intersection