Emil Øien Lunde, Thomas Wolff

Travel Time Prediction, a comparison study on a common data set

Master's thesis, spring 2015

Artificial Intelligence Group
Department of Computer and Information Science
Faculty of Information Technology, Mathematics and Electrical Engineering

Abstract

- This is were the abstract will be...

Preface

This work was conducted as part of the authors' Master's degree at NTNU. It is an extension of the work done in Øien Lunde and Wolff [2014]. Sections 1.1, 2.1 and 2.2 in this paper are repeated from that report.

Emil Øien Lunde, Thomas Wolff
Trondheim, March 25, 2015

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Section 1.1 introduces the concept of Intelligent Transportation Systems and conceptual approaches to travel time prediction. Section 1.2 describes the goal with this study, and states the research questions this work sets out to answer. Section 1.3 describes the research method employed during this study. Section 1.4 summarizes this work's contributions. Finally, section 1.5 describes the structure of the rest of the paper.

1.1 Background and Motivation

Section 1.1.1 gives an overview of Intelligent Transportation Systems (ITS). Next, Section 1.1.2 introduces conceptual approaches to estimating or predicting traffic variables.

1.1.1 Intelligent Transportation Systems

ITS is a term describing systems and services in the transportation sector using information and communication technology [Norwegian Public Roads Administration, 2007]. ITS are often highly complex, consisting of multiple levels of hardware and software each responsible for performing different tasks. To collect road data, multiple sources such as cameras, loop detectors, and GPS devices are used. Since road networks span thousands of square kilometers, the distributed systems responsible for collecting and transmitting this data need to be capable of dealing with events such as faulty detectors and lost connections between transmission devices. Additionally, data is collected continuously, producing large amounts of raw data. The systems responsible for processing this data need to be very efficient to keep up with the incoming data streams whilst being able to extract useful information from big data sets. Finally, the extracted information needs to be distributed to the end users, such as traffic control centers or website users, requiring good quality of service in terms of availability.

The Norwegian Public Roads Administration's (NPRA) system for providing drivers with travel time information is an example of an ITS. The NPRA collects actual travel times from automatic vehicle identification systems on selected roads. This data needs to be processed in order to remove incorrect travel times, for example caused by vehicles stopping in between two detectors. Then, based on the observed data, the system is able to calculate expected travel times. This procedure is described in more detail in Section 2.1.5. The travel time information is finally made available to drivers through a website[1].

Aside from the challenges involving a highly complex distributed system, what makes ITS interesting is their environmental and economical benefits, in addition to their potential in increasing traffic safety. Commision of the European Communities [2001] stated that ITS have the potential to reduce travel times by up to 20%, whilst network capacity can be increased by $5-10\%$. If this is achieved, CO_2 emissions and transportation costs can be significantly reduced. The impact of ITS on traffic safety has been estimated to reduce rear-end collisions by $10-15\%$ because of advanced information and control strategies. Additionally, automatic incident detection systems, making managing emergency situations easier, have increased survival rates when accidents occur. These are all promising properties, but developing systems delivering reliable information such that these results are achieved is not trivial.

In terms of increasing traffic efficiency, providing drivers with expected travel times is a common approach. Travel time is a variable most commuters can relate to. Based on the travel time, it is easy to deduct what the current traffic situation is. This makes it a suitable variable to use in traffic information systems. A survey, reported in Chung et al. [2004], suggests that 78% of drivers are willing to change their departure time or take a different route if they are informed about delays. This indicates that providing commuters with traffic information is a useful tool for efficiently distributing traffic in the road network. Another area where travel time information is useful is trip planning, which requires predicting what travel times will be in the future. Transportation companies rely on this information to optimize their routes, potentially reducing costs and increasing efficiency.

1.1.2 Estimating or Predicting Traffic Variables

The traffic domain is highly complex as it is affected by a great variety of factors. These include aspects like human behaviour and interaction, how signalized in-

[1] www.reisetider.no

tersections are programmed, and weather conditions. Many of these factors are non-deterministic. Consider the example of cars driving on a road section with a signalized intersection. The fact that the amount of time spent waiting for a green light is different for individual vehicles may lead to different travel times, even though the traffic density is constant. Also, other factors like human interactions, and accidents are difficult to predict. The relationships between these factors and how they affect traffic conditions are highly complex, making it difficult to incorporate all of them into one single model. Analytical models, like macroscopic traffic flow theories, are therefore limited in their capability to make accurate predictions.

Another approach to predicting traffic state is by using data driven approaches. The benefit one can draw from these methods is that they do not make assumptions about the underlying data, and do not attempt to describe how the different variables in the domain interact. In contrast, data driven methods are blind to the semantics of the data, and only attempt to discover patterns in it by looking at many training examples. Machine learning is an example of a data driven method which estimate the true function that maps inputs to outputs in the underlying domain. In the case of traffic, this can be to find a model that captures the relationship between previous and future travel times.

Researchers are constantly attempting to find ways of improving prediction accuracy. Russell and Norvig [2010] state that one possible approach is to use ensemble learning. In short, ensemble learning combines predictions from several models to make a single prediction. This technique is described in more detail in Section 2.1.2.

In order to further increase prediction accuracy, one can imagine including more data sources like accidents, incidents, weather data, GPS data from taxis, and bluetooth devices in cars. This may contribute with valuable information that makes data driven methods incorporate new relationships in their models.

Traditionally, machine learning techniques used to predict travel times have been based on using historical data. Over time, as changes in the underlying data set occur, their models become outdated and their prediction accuracies deteriorate. Online learning methods are able to incorporate the most recent data in their models. This makes them very adaptable, a desirable property for systems deployed in the real-world. Online learning is described in more detail in Section 2.1.3.[2]

Øien Lunde and Wolff [2014] conducted a structured literature review to investigate in what degree ensemble learning and online learning have been employed in the traffic domain. The authors reported that various techniques for both

[2]Thus far, the text in this section has been repeated from Øien Lunde and Wolff [2014] with some smaller modifications. The text that follows in the rest of this section is to be considered new and not a part of Øien Lunde and Wolff [2014].

ensemble learning and online learning have been employed in the traffic domain. The results demonstrate that ensemble learning provides higher prediction accuracy compared to its baseline methods. Additionally, online methods have proven to be better at adapting to changes in the underlying data set. However, the structured literature review did not reveal any experiments that compares the different techniques on a common data set. This makes is difficult to say which methods performs best and in which situations.

In order to gain knowledge as to how the different techniques compare to each other, this work sets out to conduct experiments that measure the methods' performance on a common data set.

1.2 Goals and Research Questions

Goal To find the best ensemble and online learning technique(s) for predicting travel times for a given road section

Research Question 1 Given a set of baseline methods, which ensemble learning technique improves the baselines' predictions the most?

Research Question 2 Given a set of baseline methods, which ensemble learning technique yields the best prediction accuracy?

Research Question 3 Which online learning technique yields the best prediction accuracy?

Research Question 4 How do online learners compare to offline learners?

1.3 Research Method

Researchers proposing new solutions often test their methods on a few carefully selected data sets that accentuate the methods' positive abilities. This makes it difficult to assess which methods to use in real-world travel time prediction systems. These systems need to be able to provide as accurate predictions as possible whilst simultaneously dealing with varying traffic conditions, noisy/missing data, and limited computational capability. Therefore a comparison study is conducted to evaluate different travel time prediction methods based on multiple attributes. By testing the different methods on real-world data with varying properties and taking all the important factors such as prediction accuracy, robustness, and computational complexity into account, a fair comparison of the methods can be made.

In Machine Learning in general, experiments are based on observing changes to the output of a model when changing the input or some other parameter. Usually the output is a prediction of a variable. Error metrics, indicating how often the prediction is wrong and by how much, are commonly used to measure the performance of the model. Experiments are an obvious choice when evaluating machine learning models since computers make it easy to repeat experiments and analyze the results afterwards.

Being able to repeat experiments is essential since conclusions drawn from the experiments need to be based on a significant amount of data. Metrics found through statistical analysis of the experimental results provide a good basis for comparison. Important properties such as prediction accuracy and computational requirements (CPU and memory usage) can be easily measured. The metrics can be used to evaluate the methods objectively and to identify interesting properties of each method. Another advantage of statistical analysis is that confidence intervals can be used to calculate the significance of the results.

validitet: hvor vidt man kan ekstrapolere resultatene til andre domener

bruk cohen som basis for metode

1.4 Contributions

- State our contributions here

1.5 Thesis Structure

Chapter 2 gives an introduction to important concepts used throughout this paper and motivates this research. Chapter 3 describes the different methods that are evaluated. Chapter 4 describes the experiments and the results. An evaluation and discussion of the results are presented in Chapter 5.

Chapter 2

Background Theory and Motivation

This chapter covers relevant background theory and presents the literature review. In Section 2.1, key terminology used throughout this paper is described. Section 2.2 describes how the literature review was conducted. The findings of the literature review and the motivation for this research is presented in Section 2.3.

2.1 Background Theory

In this section, relevant concepts for predicting travel times are introduced. Machine Learning is described in Section 2.1.1. Next, Sections 2.1.2 and 2.1.3 describe two important machine learning concepts referred to in this study, namely ensemble learning and online learning, respectively. Section 2.1.4 covers traffic data collection and describes some important variables used in traffic modeling. The solution for travel time estimation in use at the NPRA as of 2015 is covered in Section 2.1.5.

2.1.1 Machine Learning

Machine learning is a term used in Computer Science covering methods for learning computers different tasks by investigating data. Mitchell [1997] defines machine learning as:

> A computer program is said to **learn** from experience E with respect to some class of tasks T and performance measure P, if its

> performance at tasks in T, as measured by P, improves with experience E.

Given many data points of the form $(\mathbf{x}, y)$, where $\mathbf{x}$ is a input vector of features and y is the corresponding output value, the goal of machine learning is to estimate the true function $F(\mathbf{x})$ that maps input vectors $\mathbf{x}$ to output values y. This estimated function $M(\mathbf{x})$ is called a model of the true function. Given a new input vector $\mathbf{x}_{new}$, whose output value is not known, this model can be used to predict the output $y_{\mathbf{x}_{new}}$.

2.1.2 Ensemble Learning

Basic machine learning methods, like artificial neural network, k-nearest neighbor and support vector machine, use one single model to make predictions. Single model learners have proven to make accurate predictions, but even higher accuracy can still be achieved. The idea behind ensemble learning is to combine several hypotheses in to one prediction. Consider an ensemble of five hypotheses, where majority voting is used. For the ensemble's hypothesis to be incorrect, at least three of those five hypotheses have to be incorrect [Russell and Norvig, 2010]. This is the motivation behind the approach. An ensemble of learners collectively making a prediction is more likely to be correct, especially if the learners in the ensemble have different bias with regards to the training data. By having a low correlation between the errors, it is less likely that all of the learners are mistaken at the same time. The ensemble is able to mask the weakness of each individual learner, thus increasing prediction accuracy.

2.1.3 Online Learning

Online learning is a machine learning approach where the underlying model is trained incrementally on the most recent data. As soon as new data becomes available, the error of the model is computed by comparing its output with the actual values. This way the model can be adjusted so that the prediction error is reduced. By constantly adjusting the model it is able to adapt to changes in the data set that happen over time. In the case of travel time prediction, the model is updated as soon as new travel time information becomes available. This allows the model to adapt to changes in traffic conditions such as new lanes being added to a road section causing a decrease in congestion, or tolls on one road leading to a change in traffic volume.

2.1.4 Traffic Domain

This section introduces the most common variables used to express the state of traffic and the approaches used to measure these variables. The section is based on Haugen and Aakre [2014].

Travel Time

Travel time is defined as the time it takes to drive from point a to point b. It can be expressed as

$$t = t_b - t_a$$

where t is travel time, t_a is the point in time where the vehicle passed point a, and t_b is the point in time where the vehicle passed point b.

Traffic Flow

Traffic flow is defined as the number of cars passing through a certain point on the road per unit time. It can be expressed as

$$q = \frac{n}{T}$$

where q is traffic flow, n is the number of cars, and T is time. Traffic flow is commonly given in vehicles per hour.

Density

Density is defined as the number of cars per unit length. It can be expressed as

$$k = \frac{n}{l}$$

where k is density, n is the number of cars and l is length. Density is commonly given in vehicles per km.

Collecting Traffic Data

There are several approaches to collecting traffic data. This section covers the most commonly used approaches.

Inductive loop detectors are loops of electrical wire integrated in the road. Electrical current runs through the wire, creating a magnetic field. When a car passes through the loop, a controller can register this because the car changes the magnetic field. Inductive loop detectors can measure several variables like traffic flow and car length. Since inductive loop detectors are installed in pairs, a few meters apart from each other, speed can also be measured.

Piezoelectric wires are another approach to collecting traffic data. They are also integrated in the road, but run across the road. These wires emit an electrical signal when compressed. This way a controller can register when cars run over it. Piezoelectric wires can be used to measure variables such as traffic flow, speed, the number of axles on a car and the weight per axle.

Radars are also used to collect traffic data. They can measure traffic flow, speed, and classify cars by their lengths.

Although their main task is to validate the payment of cars driving on toll roads, AutoPass tags can be used to collect travel time data as well. AutoPass tags are installed in many cars, and sits in the top of the windshield. These tags enable identification of individual vehicles, such that travel times between two measurement points can be registered.

2.1.5 Current Solution at the NPRA

The Norwegian Public Roads Administration (NPRA) are responsible for collecting traffic data on roads in Norway. They use detectors in the road, like loop detectors and piezoelectric cables, to detect passing vehicles. Additionally, many cars in Norway are equipped with electronic devices (AutoPass) used for automatic toll payments. Measurements from AutoPass devices are what the NPRA use today to estimate the current travel times on road sections in Norway. The current solution provides commuters with up-to-date travel time information based on travel times collected from the last five minutes. These measurements are used as input to the algorithm described in Wahl and Haugen [2005]. In this algorithm, the individual travel times are placed in groups of 1 minute intervals, e.g. the travel times 25:00 and 25:59 are placed in the 25 minute group. The estimated travel time is calculated based on the interval with the most registered travel times and its neighboring intervals. A weighted average of the travel times of these intervals is the final estimated travel time.

2.2 Structured Literature Review Protocol

This section gives an overview of the structured literature review [Kofod-Petersen, 2014] process. A more extensive and detailed description of the process can be found in Appendix A.

To find relevant literature, a search string that covers the main aspects of this research is developed. This search string is used to retrieve articles from two search engines: IEEEXplore[1] and Engineering Village[2]. As most of the retrieved literature is not relevant to this study, a screening process to filter out

[1] `http://ieeexplore.ieee.org/search/advsearch.jsp?expression-builder`
[2] `http://www.engineeringvillage.com/search/expert.url?CID=expertsearch`

the irrelevant literature is employed. This is done in a top-down approach, first filtering the articles by title, second by abstract, and finally by full text.

The first two filtering steps are used to ensure that the articles included further in the screening process are in the right domain, and that the work exhibits solutions that are relevant to this study. To decide which articles are relevant, inclusion criteria regarding the contents of the titles and abstracts are developed. An example of an inclusion criterion is: "The title indicates that the article predicts, estimates or models road traffic variables, e.g. traffic flow, speed, congestion or travel time". This criterion ensures that articles regarding estimation or prediction of traffic variables are included, as they are of high relevance to this study. Another example of an inclusion criterion is: "The article describes a solution that can easily be extended or adapted to fit our research". By employing this inclusion criterion, articles that present solutions that can be adapted to our problem are included. After evaluating each article based on every criteria for title and abstract, the most irrelevant literature is filtered out.

The last step in the screening process is to assess the quality of the work presented in the different articles. During the quality assessment, questions such as "Is the method/algorithm thoroughly explained?" and "Does the test evidence support the findings presented?" have to be answered to identify the good research. The articles are given a score in the range from 0 to 1 for each quality assessment question. To ensure a certain quality, only articles with a total score above a set threshold is included in the state of the art review.

2.3 Motivation

The structured literature review presented in Øien Lunde and Wolff [2014] reveals several ensemble learning techniques employed in the traffic domain, ranging from simple linear combinations to more complicated non-linear schemes. It also describes several online learning techniques used for making predictions in the traffic domain. This section presents the key findings in the structured literature review, and motivates the methods tested further in this research.

2.3.1 Inclusion criteria

Due to time limitations, only a small selection of the techniques described in Øien Lunde and Wolff [2014] can be included in the experiments in this study. Therefore a set of inclusion criteria are developed to narrow down the number of methods being tested. These criteria are:

Inclusion Criterion 1 The paper employs an ensemble learning technique or an online learning technique and

Inclusion Criterion 2 The technique described in the paper is available through an open source implementation or is sufficiently explained in order to be implemented

and

Inclusion Criterion 3 The paper focuses on improving prediction accuracy

Since this research sets out to compare ensemble learning techniques as well as online learning techniques, it is essential that the proposed approach in a paper falls into one of these two categories. This is reflected by inclusion criterion 1. Furthermore, as this research has time restrictions, it is an absolute necessity that the methods being tested does not require much implementation, as the main focus in this work is on comparing methods, not spending time implementing them. Inclusion criterion 2 ensures that only papers presenting methods requiring a minimum of implementation is included. As several papers presented in Øien Lunde and Wolff [2014] focused on other aspects of travel time prediction than prediction accuracy, like computational cost, inclusion criterion 3 is present to ensure that the main focus of the proposed method is increasing prediction accuracy.

Of all the papers resulting from the structured literature review in Øien Lunde and Wolff [2014] only those employing ensemble learning or online learning techniques are candidates for the experiments in this study due to inclusion criteria 1. Three of these candidate techniques did not satisfy all the criteria. Zhu and Shen [2012] describes a technique that makes use of the notion of traffic regimes. As the authors do not give an accurate definition of different traffic regimes the paper does not satisfy inclusion criteria 2, and is therefore not included. van Hinsbergen et al. [2009] is not included as the paper focuses on reducing computational cost, and thus does not satisfy inclusion criteria 3. Although Lu [2012] satisfies all the inclusion criteria, it is not included in the experiments. Since the proposed method's results are compared to a primitive baseline, it is not justifiable to spend time implementing because from their comparison it is difficult to say whether the method is good or not.

2.3.2 Methods

Sun [2009] investigates whether or not using a multitask learner will improve upon using a simple task learner when predicting traffic flow. Additionally, the authors look at the effect of using an ensemble of multitask learners compared to just a single multitask learner. In contrast to single task learners, which predicts one variable at a time, multitask learners make predictions for several variables simultaneously. The motivation for using multitask learners is that they may incorporate more information during training because the different learning tasks share some common properties. The method they use for combining predictions from their ensemble of multitask learners is called bagging which is described in more detail in section 3.1. The results from the experiments conducted in Sun [2009] illustrate that bagging may improve prediction accuracy compared to using just one model to make predictions.

All though not present in any of the papers reported in the structured literature review in Øien Lunde and Wolff [2014], boosting is an ensemble learning technique that is included here due to completeness of the most common ensemble learning techniques. Boosting is explained in more detail in section 3.2.

Bagging combines its baseline models simply by calculating an average. This means that the baselines are combined without regards to their performance. Another approach is to assign a weight to each model in the ensemble, representing its contribution to the ensemble's output. The idea behind using a weighted average is that the weights can be calculated based on each models' prediction error, thus allowing the best performing models to become more prominent in the ensemble. This can help improve the overall performance of the ensemble. In Li et al. [2014] three different weighting schemes are explored, namely least square, ridge regression and lasso. Their results indicate that the weighting schemes increase prediction accuracy compared to the best performing baseline model, but it is difficult to determine which weighting scheme is best. Additionally, they conclude that using a limited number of baseline models generally improves prediction accuracy. The strength of lasso compared to the other weighting schemes is that it can automatically perform model selection (by assigning zero weight to some models), thus outputting a weighted average of only the best performing models. This feature, i.e. being able to automatically select the appropriate baseline models, makes lasso an interesting approach. The weighting schemes tested are not compared to a simple (non-weighted) average or to other ensemble approaches. Thus, although demonstrating the effectiveness of combining predictions with a weighted average, the research does not indicate whether this approach is preferable to any other ensemble approach.

[Stathopoulos et al., 2008] argues that the relationships between the baseline models in an ensemble are not necessarily linear, which is why a Fuzzy Rule Based System (FRBS) capable of representing non-linear relationships between baseline models is proposed in Stathopoulos et al. [2008]. FRBSs are explained in more detail in Chapter 3. In general a FRBS works by explicitly defining a set of rules that convert inputs to an output. Since these rules are explicitly defined, one is able to incorporate expert knowledge and represent relationships between models that are impossible to represent by linearly combining them. The contribution of this paper is somewhat similar to Li et al. [2014]. The experimental results demonstrate that the ensemble outperforms the best performing baseline in terms of prediction accuracy, but none of their experiments compare the proposed method to another ensemble approach. Because this approach exhibits some interesting features, more specifically its ability to represent non-linear relationships and to incorporate expert knowledge, it is included in the comparison study.

So far, the methods described focuses on how to combine predictions from an ensemble of models into one, hopefully more accurate, prediction. The other branch of methods that the structured literature review in Øien Lunde and Wolff [2014] reveals, are methods capable of updating its underlying model each time new observations arrive. These methods are often called online methods or incremental methods because they learn from one observation at a time in addition to having a learning phase in parallel with the prediction phase. In contrast, offline methods first learn, then predicts. Van Lint [2008] presents an online method for predicting travel time using the extended Kalman filter (EKF) for training the weights of an artificial neural network. The author makes the assumption that the weights in the artificial neural network does a random walk along some high-dimensional path, and that the artificial neural network does a non-linear observation of those weights when making a prediction. The EKF works in an iterative fashion, updating its state vector and variance estimate each time step. This makes it suitable for doing online learning. A more detailed description of the method can be found in section 3.5. The results reported in Van Lint [2008] demonstrates that the online EKF algorithm does not perform as good as an offline trained model. However, it has the benefit of being online, which makes it more attractive for use in a real time system. Offline methods are limited to the data it is exposed to during training. This makes it hard for offline methods to predict outcomes for situations it has never seen before. Online methods, on the other hand, updates its model continuously as new observations arrive. This way it can adapt its model to the changes in the underlying data that happen over time. This leads to a more adaptive model that does not get outdated if there is a change in the underlying data.

In Haworth et al. [2014] a novel online learning approach for travel time prediction called Local Online Kernel Ridge Regression (LOKRR) is presented. The approach is based on creating multiple kernels, each corresponding to a specific time interval of the day, using ridge regression to predict travel times based on historical data. One of the most significant drawbacks with using a single kernel to represent the entire data set is that the amount of information it can incorporate is limited due to computational complexity. This makes it difficult to capture for example seasonal variations in the traffic data. LOKRR aims to deal with this limitation. Since each kernel in LOKRR only needs to incorporate data from a certain time of day, more historical data can be used in each kernel. This makes it easier to detect cyclic patterns, such as rush hours, in the data. Additionally, the parameters of each kernel can be tuned individually. This enables each kernel to adapt to its underlying data distribution more effectively, thus increasing prediction accuracy. Another important feature in LOKRR is that it uses a sliding window approach to update the kernels with the most recent data, enabling it to adapt to changes in the data. The advantages of LOKRR are demonstrated in the experiments presented in Haworth et al. [2014]. It outperforms a support vector machine and a neural network for certain prediction horizons in addition to providing better prediction accuracy during non-recurrent congestion scenarios.

In summary what motivates this research is the lack of comparison between the state of the art prediction approaches. The experiments described in the literature are conducted to accentuate the positive features of the proposed methods and to demonstrate their superiority over established methods. It is difficult to determine whether or not the methods will exhibit the same behaviour on data with different properties and when there are limitations on space and computational complexity. This research aims to test the selected approaches on similar data sets to determine what approaches are most suitable to use in a real-world travel time prediction system.

Chapter 3

Architecture/Model

To give a theoretical backdrop for the rest of this study, this chapter explains the methods used in this comparison study in more detail. Sections 3.1, 3.2, 3.3 and 3.4 explain the ensemble techniques Bagging, Boosting, Lasso, and Fuzzy Rule Based System, respectively. Section 3.5 explains the Censored Extended Kalman Filter, and finally section 3.6 explains Local Online Kernel Ridge Regression.

3.1 Bagging

Bagging is an ensemble learning technique introduced in Breiman [1996]. All though the technique is actually more concerned with how the models in the ensemble are trained than how the models are combined, it is considered an ensemble learning algorithm. In bagging, during the training phase, the different models are trained on a subset of the original training data. The subsets are sampled uniformly with replacement from the training data. When making a prediction from the ensemble, the mean of the predictions from the individual members of the ensemble is used for regression problems, and majority voting is used for classification. Breiman [1996] explains that bagging may work well for ensembles of machine learning methods that are unstable. However, one has little to gain from using bagging when the ensemble consists of machine learning methods that are stable. A machine learning method is referred to as stable if a change in the training set makes little to no change in the learned model. However, a machine learning method is referred to as unstable if a change in the training set makes large changes in the learned model.

3.2 Boosting

Boosting is one of the most common ensemble learning techniques [Russell and Norvig, 2010]. Like Bagging, Boosting re-samples the training data to construct an ensemble of models. However, the sampling technique in Boosting is different from the one used in Bagging. In Boosting, each training example is given a weight. This weight corresponds to the importance of the example, and is initialized to 1 for all examples, meaning that each example is equally important. The training examples for the first model in the ensemble are drawn from this uniform distribution. During training, this model will correctly classify some examples, while other examples will be misclassified. Before constructing the next model in the ensemble, the weights for each example in the training data are updated, increasing the weights of the misclassified examples and reducing the weights of the correctly classified examples. This way, the misclassified examples are given higher importance, and have a higher probability of being sampled by the next model in the ensemble. This iterative process continues until K models are constructed, where K is a parameter to the boosting algorithm. When making predictions, a weighted majority vote from the models in the ensemble is used, where the weight of each model is proportional to the number of correctly classified examples it has predicted during training.

3.3 Lasso Ensemble

Both Bagging and Boosting include all models in the ensemble when making the final collective prediction. The idea behind Lasso Ensemble is to only include the best performing models in the ensemble when making the final prediction.

Imagine having sensors along a road, registering traffic data at N locations for T time steps. This data forms a matrix $\mathbf{F} \in \mathbb{R}^{N \times T}$ of recent traffic observations. Consider an ensemble of K models, whose task is to make predictions about traffic variables at these N locations for each time step. Each model k makes its prediction $g_n^t(k)$ for location n at time t. This forms a matrix of predictions $\mathbf{G}_k \in \mathbb{R}^{N \times T}$. The model's predictions are put together in a vector $\mathbf{G} = [\mathbf{G}_1, \mathbf{G}_2, \ldots, \mathbf{G}_K]$. The ensemble's collective prediction h_n^t for location n at time t is a weighted average of the member's predictions:

$$h_n^t = \sum_{k=1}^{K} w_k g_n^t(k)$$

where w_k is the weight corresponding to model k. The ensemble's predictions for all N locations for all T time steps forms a matrix $\mathbf{H}$, and is given by $\mathbf{H} = \mathbf{GW}$, where $\mathbf{W} = [w_1, w_2, \ldots, w_K]$. Different $\mathbf{W}$ will result in different predictions, and in turn different prediction accuracies. How do we find the optimal set of weights $\hat{\mathbf{W}}$?

One of the proposed solutions to this question in Li et al. [2014] is called Lasso Ensemble. In this approach the optimal set of weights is given by solving the following equation:

$$\hat{\mathbf{W}} = \underset{\mathbf{W}}{\arg\min} \|\mathbf{F} - \mathbf{GW}\|_2^2 + \lambda\|\mathbf{W}\|_1$$

where $\|\mathbf{F} - \mathbf{GW}\|_2^2$ is the squared l_2 norm of the difference between the true traffic variables and the predictions made by the ensemble, $\|\mathbf{W}\|_1$ is the l_1 norm of the weights, and λ is a regularization term controlling the importance of the l_1 norm of the weights. Li et al. [2014] explain that having this l_1 regularization of the weights leads to a sparse solution, corresponding to many weights being zero. This means that only some of the models is included when making the final prediction.

3.4 Fuzzy Rule Based System

Fuzzy Rule Based Systems (FRBS) are machine learning approaches where a set of rules define their behaviour. These rules can either be manually created, thus enabling the ability to incorporate expert knowledge, or automatically created based on a data set. FRBSs consist of a fuzzyfication stage, an inference system, and a defuzzyfication stage.

The fuzzyfication stage assigns input values to fuzzy sets. Fuzzy sets are sets whose elements have degrees of membership. What degree of membership an input is given to a fuzzy set is defined by a membership function. Basically what a membership function does is that it returns a probability that some value is a member of a fuzzy set. Imagine a system predicting people's weight based on their height. Two possible fuzzy sets describing a persons height could be *short* and *tall*. The height 185 cm can for example be assigned the degrees 0.2 and 0.8 to *short* and *tall*, respectively.

The inference system is used to generate outputs based on the fuzzyfied input. It consists of a number of if-then rules, referred to as the rule base, that define what to do when some condition is satisfied. The rule base basically constitutes a complete mapping from all possible inputs to outputs. In the example FRBS above, a rule could be "If a person is *tall* then that person is *heavy*". In contrast to typical black-box approaches such as Artificial Neural Networks, the behaviour of FRBSs can be easily interpreted because their behaviour is explicitly described in the rule base.

The responsibility of the defuzzification stage is to convert the values representing membership degrees returned by the inference system into one single output. For example converting membership degrees of *heavy* and *light* into kilograms.

Stathopoulos et al. [2008] use a FRBS to combine forecasts from two different models predicting traffic flow. The traffic flows are mapped to the fuzzy sets *low*, *medium* and *high*. The FRBS consists of two different rule bases, each preferring one of the models over the other as seen in figure 3.1. When predicting traffic flow at time $t + 1$, the rule base giving priority to the best performing model at time t is selected. The output from the rule base is then converted back to a traffic flow value using the center of gravity algorithm.

IF predicted flow of model **A** is LOW ***AND*** predicted flow of model **B** is LOW, ***THEN*** flow at time t+1 is LOW
IF predicted flow of model **A** is LOW ***AND*** predicted flow of model **B** is ***NOT*** LOW, ***THEN*** flow at time t+1 is LOW
IF predicted flow of model **A** is MEDIUM ***AND*** predicted flow of model **B** is MEDIUM, ***THEN*** flow at time t+1 is MEDIUM
IF predicted flow of model **A** is MEDIUM ***AND*** predicted flow of model **B** is ***NOT*** MEDIUM, ***THEN*** flow at time time t+1 is MEDIUM
IF predicted flow of model **A** is HIGH ***AND*** predicted flow of model **B** is HIGH, ***THEN*** flow at time t+1 is HIGH
IF predicted flow of model **A** is HIGH ***AND*** predicted flow of model **B** is ***NOT*** HIGH, ***THEN*** flow at time t+1 is HIGH

Figure 3.1: Rule base used in Stathopoulos et al. [2008].

3.5 Censored Extended Kalman Filter

To explain the Censored Extended Kalman Filter, this section starts by briefly explaining the Kalman Filter and its extension, the Extended Kalman Filter. Next, this section explains how the Extended Kalman Filter can be used for training the weights of an artificial neural network, and how estimates of travel times make the Censored Extended Kalman Filter applicable for an online learning setting.

3.5.1 Kalman Filter

The Kalman Filter is an approach for making estimations about some state vector $\mathbf{x_t}$ at time t from noisy observations $[\mathbf{z_0}, \mathbf{z_1}, \ldots, \mathbf{z_{t-1}}]$ from time $t = 0$ to time t. It is assumed that there is a linear relationship between the state vector $\mathbf{x_t}$ at time t and the state vector $\mathbf{x_{t+1}}$ at time $t+1$. Additionally, it is assumed that the process evolves with some uncertainty and that this uncertainty can be modelled as a term following a zero mean Gaussian distribution. These assumptions yield the basis for the transition model, describing the relationship between successive states in the system, which can be described by the following equation:

$$\mathbf{x_{t+1}} = \mathbf{F}\mathbf{x_t} + \mathbf{B}\mathbf{u_t} + \epsilon_{process}$$

where, $\mathbf{x_{t+1}}$ is the state vector at time $t + 1$, $\mathbf{x_t}$ is the state vector at time t, $\mathbf{F}$ is a matrix that represents the relationship between the variables in the state vector at time t and $t+1$, $\mathbf{B}$ is a matrix that represents the effect that the input $\mathbf{u_t}$ to the system at time t has on the state vector. $\epsilon_{process}$ is the Gaussian error term that represents the uncertainty with which the system evolves. $\epsilon_{process}$ has covariance matrix $\mathbf{Q}$.

It is further assumed in the Kalman Filter that the state vector $\mathbf{x_t}$ can not be directly observed. One can only gain insight to the state vector through observations of related variables in observation vector $\mathbf{z_t}$. It is assumed that there is a linear relationship between the unobservable state vector $\mathbf{x_t}$ and the observation vector $\mathbf{z_t}$. It is assumed that the observations are noisy, which means that there is some uncertainty related to the observations. This uncertainty is assumed to follow a zero mean Gaussian distribution. Putting this together, the observation model, describing the relationship between observations and the state vector, can be expressed by the following equation:

$$\mathbf{z_t} = \mathbf{H}\mathbf{x_t} + \epsilon_{observation}$$

where $\mathbf{z_t}$ is the observation at time t, $\mathbf{x_t}$ is the state vector at time t, $\mathbf{H}$ is a matrix that represents the relationship between the observations and the state vector. $\epsilon_{observation}$ represents the uncertainty related to the observation, following a zero mean Gaussian distribution with covariance matrix $\mathbf{R}$.

The Kalman Filter approach provides the following equations for making predictions about the system in question:

$$\hat{\mathbf{x}}_{t|t-1} = \mathbf{F}\hat{\mathbf{x}}_{t-1|t-1} + \mathbf{B}\mathbf{u_t}$$

$$\mathbf{P_{t|t-1}} = \mathbf{F}\mathbf{P_{t-1|t-1}}\mathbf{F}^T + \mathbf{Q}$$

where $\hat{\mathbf{x}}_{t|t-1}$ is the prediction of the state vector at time t given information up to time $t = t-1$, $\hat{\mathbf{x}}_{t-1|t-1}$ is the estimate of the state vector at time $t-1$ given information up to time $t = t-1$. $\mathbf{P_{t|t-1}}$ is the covariance matrix of $\hat{\mathbf{x}}_{t|t-1}$, representing the uncertainty related to the prediction made, $\mathbf{P_{t-1|t-1}}$ is the covariance matrix representing the uncertainty from the previous iteration.

After making an observation $\mathbf{z_t}$ at time t, the Kalman Filter approach provides the following equations for updating the estimation of the state vector and its uncertainty

$$\hat{\mathbf{x}}_{t|t} = \hat{\mathbf{x}}_{t|t-1} + \mathbf{K}(\mathbf{z_t} - \mathbf{H}\hat{\mathbf{x}}_{t|t-1})$$

$$\mathbf{P_{t|t}} = \mathbf{P_{t|t-1}} - \mathbf{K}\mathbf{H}\mathbf{P_{t|t-1}}$$

where $\hat{\mathbf{x}}_{t|t}$ is the updated estimate for the state vector, $\hat{\mathbf{x}}_{t|t-1}$ is the predicted state vector, $\mathbf{K}$ is the Kalman gain matrix which is given by $\mathbf{K} = \mathbf{P_{t|t-1}}\mathbf{H}^T(\mathbf{H}\mathbf{P_{t|t-1}}\mathbf{H}^T + \mathbf{R})^{-1}$.

The Kalman Filter works in an iterative fashion, starting with an initial state vector $\mathbf{x_0}$ and a corresponding covariance matrix $\mathbf{P}_0$ at time $t = 0$. Then it makes a prediction for the state vector $\hat{\mathbf{x}}_{1|0}$ and its uncertainty $\mathbf{P_{1|0}}$ for time $t = 1$. Next, it makes an observation $\mathbf{z}_1$ at time $t = 1$ providing the basis for updating the state vector $\hat{\mathbf{x}}_{1|1}$ and its uncertainty $\mathbf{P_{1|1}}$ for time $t = 1$. Now the process starts over again, making a prediction for the next time step, doing an observation, updating the state vector etc.

3.5.2 Extended Kalman Filter

The Kalman Filter approach described in the previous subsection assumes that the relationship between successive states, and the relationship between observations and state vector, is linear. However, this may not be sufficient to model all systems and processes. The Extended Kalman Filter generalizes the ideas from the Kalman Filter and assume that these relationships are non-linear. Allowing non-linear relationships in the model yields the following transition and observation model equations:

$$\mathbf{x_{t+1}} = f(\mathbf{x_t}, \mathbf{u_t}) + \epsilon_{process}$$

$$\mathbf{z_t} = g(\mathbf{x_t}) + \epsilon_{observation}$$

where f and g are arbitrary continuous, differentiable functions.

Since the transition and observation models are changed, the prediction and update equations have to be updated to reflect this non-linear relationship. The prediction equations for the Extended Kalman Filter can be expressed as:

$$\mathbf{x_{t|t-1}} = f(\mathbf{x_{t-1|t-1}}, \mathbf{u_t})$$

$$\mathbf{P_{t|t-1}} = \mathbf{J_f} \mathbf{P_{t-1|t-1}} \mathbf{J_f}^T + \mathbf{Q}$$

where $\mathbf{J_f}$ is the Jacobian of function f.

The update equations for the Extended Kalman Filter can be expressed as:

$$\hat{\mathbf{x}}_{t|t} = \hat{\mathbf{x}}_{t|t-1} + \mathbf{K}(\mathbf{z_t} - g(\hat{\mathbf{x}}_{t|t-1}))$$

$$\mathbf{P_{t|t}} = \mathbf{P_{t|t-1}} - \mathbf{K} \mathbf{J_g} \mathbf{P_{t|t-1}}$$

where $\mathbf{K} = \mathbf{P}\mathbf{J_g}^T(\mathbf{J_g}\mathbf{P}\mathbf{J_g}^T + \mathbf{R})^{-1}$ and $\mathbf{J_g}$ is the Jacobian of function g.

3.5.3 Artificial Neural Network Weight Training

The above explanations of the Kalman Filter and the Extended Kalman Filter are general and not written to be specific to any problem, process or system. This section explains how the Extended Kalman Filter can be applied for training the weights in an artificial neural network.

An artificial neural network consists of several layers of neurons. Each neuron outputs a signal which is constructed by applying a function, often a non-linear sigmoid shaped function, to a weighted sum of its input signals. A common structure of an artificial neural network is having an input layer, one or more hidden layers, and an output layer. The neurons are connected to each other with weighted edges, propagating signals from inputs to outputs. During training, the network is exposed to a set of input-output pairs. The network propagates the input signals through the network, and the output signal from the output neuron is compared to the correct output in the input-output pair. Then the weights are adjusted to make the difference between the network's output and the correct output as small as possible across all training examples.

The outputs of the network can be seen as a non-linear function of the input signals to the network and the weights in the network. The idea presented in Van Lint [2008] is that one can consider the weights in an artificial neural network during training as a state vector, where the artificial neural network does a non-linear observation of the weights through its outputs. Given this problem definition, one can apply the Extended Kalman Filter equations explained in subsection 3.5.2 to update the weights in the neural network in an incremental fashion.

3.5.4 Censored Extended Kalman Filter

A common setting for prediction tasks is that presented with an input $\mathbf{x}_t$ at time t, make a prediction $\hat{\mathbf{y}}_t$, observe the correct output $\mathbf{y}_t$ and update the model given the prediction error $\epsilon = \hat{\mathbf{y}}_t - \mathbf{y}_t$. This approach is applicable for many problems. However, it assumes that after the prediction $\hat{\mathbf{y}}_t$ is made, the correct output $\mathbf{y}_t$ is available. For many problems this is true, however, this is not the case for travel time prediction. Imagine predicting travel times for a road section stretching from point A to point B. When a vehicle arrives at point A at time t_A, one wants to make a prediction $\hat{y}_{AB}$ for how long it will take for that vehicle to arrive at point B, given the current traffic flow, traffic density and vehicle speeds. Making the prediction is straight forward. However, updating the model to correct for the prediction error $\epsilon = \hat{y}_{AB} - y_{AB}$ depends on knowing the actual travel time y_{AB} spent driving from point A to point B. This travel time is not available until time $t = t_A + y_{AB}$. This causes a problem: the model is not updated until the realized travel time is available, which can be an arbitrarily long time, making the model vulnerable to being outdated and not representative for the current traffic situation.

To reduce the delay of when the model is updated, Van Lint [2008] proposes an approach that makes use of a *censored* observation of the realized travel time. Imagine that at time t_A there is a point in time k, where $k < t_A$, for which a realized travel time y_k is available. Consider a point in time p, where $k < p < t_A$, for which no realized travel time is available. A censored observation of the travel time y_{AB} is $y^*_{AB} = t_A - p$. This censored observation of the actual travel time can be used as an estimate for the actual travel time, and the model can be updated at time t_A instead of at time $t = t_A + y_{AB}$.

3.6 Local Online Kernel Ridge Regression

In Haworth et al. [2014] a novel approach for travel time prediction called Local Online Kernel Ridge Regression (LOKRR) is introduced. The approach is based on using kernel ridge regression to generate a prediction for a data point as follows: $g(\mathbf{x}) = \mathbf{y}'(\mathbf{K} + \lambda\mathbf{I_n})^{-1}\mathbf{k}$, where $\mathbf{x}$ is a new observation, $\mathbf{K}$ is a kernelized version of input matrix $\mathbf{X}$ (a $n * p$ matrix with n observations of p variables), $\mathbf{y}$ is a vector of size n where element i corresponds to the true travel time for observation i in $\mathbf{X}$, and $\mathbf{k}$ is a vector of size n where element i is the value of the Gaussian radial basis function between the new observation $\mathbf{x}$ and row i in $\mathbf{X}$. $\mathbf{K}$ is generated by calculating the Gaussian radial basis function between every pair of observations in $\mathbf{X}$.

Since LOKRR is online it needs to calculate the inverse of the regularized kernel matrix, i.e. $(\mathbf{K} + \lambda\mathbf{I_n})^{-1}$, every time a new travel time is realized. Doing this from scratch is too expensive to do in an online setting. Therefore, a method that computes the inverse at time $t + 1$ based on the inverse at time t is used. This significantly reduces the computation time. The method is described in more detail in the paper.

What differentiates LOKRR from the other approaches described in this chapter, besides from using kernel ridge regression to make predictions, is that it organizes the data into different sets based on the time of day of the observations. It creates one kernel for each data set, meaning one kernel is responsible for making predictions for a certain time of day. This allows for tuning each kernel individually, making it possible to adapt to traffic patterns that are specific for a certain time of the day. It also makes the approach easy to parallelize.

Chapter 4

Experiments and Results

This chapter presents the experiments performed in this study along with their results. Section 4.1 describes the plan for which experiments is conducted, and why they are included. Section 4.2 explains how the experiments are performed. Section 4.3 presents the results following from the experiments that are conducted.

4.1 Experimental Plan

In order to conduct a comparison study, two data sets are created.

Data Set 1 Input: mean travel time for the past five minutes, traffic volume for the past five minutes. Output: actual travel time for an individual vehicle

Data Set 2 Input: mean travel time for the past five minutes, traffic volume for the past five minutes, mean vehicle speed for the past five minutes, number of trucks or buses for the past five minutes and mean distance between vehicles for the last five minutes. Output: actual travel time for an individual vehicle

The data is collected by the NPRA from highway E39 between Dusavik and Bogafjell in Rogaland, Norway. The travel time data is collected by registering IDs from AutoPass devices in cars and calculating the time spent between two measurement points. There are in total five measurement points along this road section. The vehicle speed and length is measured with loop detectors at six different locations.

Experiment 1 This experiment aims to answer Research Question 1 and 2. Four baseline models are trained on Data Set 1, namely Artificial Neural

Network, k-nearest neighbors, Kalman filter, and Support Vector Machine. The baselines are combined in three different ways, using Fuzzy Rule Based System with auto-generated rules, Lasso and a simple average of the baselines' predictions. In an attempt to answer Research Question 1, the performance metrics of Lasso, FRBS and the simple average are compared to the baseline models' performance metrics in order to determine which approach improves the baselines the most. Additionally, the performance metrics of each ensemble approach are compared to find the best performing approach in terms of prediction accuracy in an attempt to answer Research Question 2.

Experiment 2 Data Set 1 is also used when comparing the online learning approaches in an attempt to answer Research Question 3. Local Online Ridge Regression, Censored Extended Kalman Filter and TODO: INSERT ONLINE BASELINE METHOD are compared to each other in terms of prediction accuracy. Additionally, in order to investigate Research Question 4, the online approaches are compared to the offline baseline methods used in the ensemble approaches.

Experiment 3 In order to investigate Research Question 6, Experiment 1 is repeated using Data Set 2 instead of using Data Set 1. The results from this experiment are compared with the results from Experiment 1 to investigate whether or not including vehicle speed, number of trucks and buses, and distance between vehicles improves prediction accuracy.

4.2 Experimental Setup

- Describe the road section we are studying
- Explain the data set with training and testing data
- Explain the different models (which parameters, topology of ANNs)
- Explain the environment: hardware, software, programming language, OS

4.3 Experimental Results

- Present results
- Use tables and graphs with appropriate descriptions

Chapter 5

Evaluation and Conclusion

- Introduction to this chapter

5.1 Evaluation

- Objective observation of the results

5.2 Discussion

- What can the results imply?
- What conclusions can be drawn from this?
- What are the possible consequences of these observations?
- Discuss possible limitations to this work

5.3 Contributions

- State contributions

5.4 Future Work

- Present suggestions for future work
- Things we did not have time to do, but think are important or interesting to

investigate further

Bibliography

Bajwa, S. U. I., Chung, E., and Kuwahara, M. (2003). A travel time prediction method based on pattern matching technique. volume 21, pages 997 – 1010, Cairns, QLD, Australia.

Bouillet, E., Chen, B., Cooper, C., Dahlem, D., and Verscheure, O. (2013). Fusing traffic sensor data for real-time road conditions. Rome, Italy.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2):123–140.

Chen, D.-W. and Zhang, J.-P. (2005). Time series prediction based on ensemble anfis. In *Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on*, volume 6, pages 3552–3556 Vol. 6.

Chen, L. and Chen, C. (2007). Ensemble learning approach for freeway short-term traffic flow prediction. In *System of Systems Engineering, 2007. SoSE '07. IEEE International Conference on*, pages 1–6.

Chung, E., Warita, H., ul Islam Bajwa, S., and Kuwahara, M. (2004). Travel time prediction: issues and benefits. `http://its.iis.u-tokyo.ac.jp/pdf/2004-021.pdf`, accessed November 21, 2014.

Commision of the European Communities (2001). European transport policy for 2010: time to decide. Technical report. `http://ec.europa.eu/transport/themes/strategies/doc/2001_white_paper/lb_com_2001_0370_en.pdf`, accessed November 21, 2014.

Deb Nath, R. P., Lee, H.-J., Chowdhury, N. K., and Chang, J.-W. (2010). Modified k-means clustering for travel time prediction based on historical traffic data. volume 6276 LNAI, pages 511 – 521, Cardiff, United kingdom.

Dembczynski, K., Kotlowski, W., Gawel, P., Szarecki, A., and Jaszkiewicz, A. (2013). Matrix factorization for travel time estimation in large traffic networks. volume pt.II, pages 500 – 10, Berlin, Germany.

Dharia, A. and Adeli, H. (2003). Neural network model for rapid forecasting of freeway link travel time. *Engineering Applications of Artificial Intelligence*, 16(7-8):607 – 613.

Hao, L., Xiao-liang, Z., and Ke, Z. (2008). Travel time prediction for urban arterials based on rough set. *Journal of Highway and Transportation Research and Development*, 25(10):117 – 22.

Haugen, T. and Aakre, A. (2014). BA6058 Trafikkteknikk og trafikksikkerhet, Kompendium del Trafikkteknikk. Norwegian University of Science and Technology.

Haworth, J., Shawe-Taylor, J., Cheng, T., and Wang, J. (2014). Local online kernel ridge regression for forecasting of urban travel times. *Transportation Research Part C: Emerging Technologies*, 46:151 – 178.

Hofleitner, A., Herring, R., and Bayen, A. (2012). Arterial travel time forecast with streaming data: a hybrid approach of flow modeling and machine learning. *Transportation Research, Part B (Methodological)*, 46(9):1097 – 122.

Kofod-Petersen, A. (2014). How to do a structured literature review in computer science, version 0.2. `https://research.idi.ntnu.no/aimasters/files/SLR_HowTo.pdf`, accessed September 1, 2014.

Li, L., Chen, X., and Zhang, L. (2014). Multimodel ensemble for freeway traffic state estimations. *Intelligent Transportation Systems, IEEE Transactions on*, 15(3):1323–1336.

Li, Y., Fujimoto, R., and Hunter, M. (2009). Online travel time prediction based on boosting. In *Intelligent Transportation Systems, 2009. ITSC '09. 12th International IEEE Conference on*, pages 1–6.

Liu, H., van Lint, H., van Zuylen, H., and Zhang, K. (2006a). Two distinct ways of using kalman filters to predict urban arterial travel time. In *Intelligent Transportation Systems Conference, 2006. ITSC '06. IEEE*, pages 845–850.

Liu, H., Van Zuylen, H., Van Lint, H., and Salomons, M. (2006b). Predicting urban arterial travel time with state-space neural networks and kalman filters. Number 1968, pages 99 – 108.

Lu, C.-C. (2012). An adaptive system for predicting freeway travel times. *International Journal of Information Technology; Decision Making*, 11(4):727 – 47.

Ma, Y., Chowdhury, M., Sadek, A., and Jeihani, M. (2012). Integrated traffic and communication performance evaluation of an intelligent vehicle infrastructure integration (vii) system for online travel-time prediction. *Intelligent Transportation Systems, IEEE Transactions on*, 13(3):1369–1382.

Mak, W., Viti, F., Hoogendoorn, S., and Hegyi, A. (2009). Online travel time estimation in urban areas using the occupancy of long loop detectors. pages 383 – 390, Redondo Beach, CA, United states.

Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill, international edition.

Mu, T., Jiang, J., and Wang, Y. (2013). Heterogeneous delay embedding for travel time and energy cost prediction via regression analysis. *Intelligent Transportation Systems, IEEE Transactions on*, 14(1):214–224.

Nikovski, D., Nishiuma, N., Goto, Y., and Kumazawa, H. (2005). Univariate short-term prediction of road travel times. In *Intelligent Transportation Systems, 2005. Proceedings. 2005 IEEE*, pages 1074–1079.

Norwegian Public Roads Administration (2007). ITS-Strategi for Statens vegvesen. Technical Report 7. `http://www.vegvesen.no/_attachment/60628`, accessed November 20, 2014.

Ohra, Y., Koyama, T., and Shimada, S. (1997). Online-learning type of traveling time prediction model in expressway. In *Intelligent Transportation System, 1997. ITSC '97., IEEE Conference on*, pages 350–355.

Park, J., Murphey, Y., McGee, R., Kristinsson, J., Kuang, M., and Phillips, A. (2014). Intelligent trip modeling for the prediction of an origin-destination traveling speed profile. *IEEE Transactions on Intelligent Transportation Systems*, 15(3).

Rossi, A., Carvalho, A., and Soares, C. (2012//). Meta-learning for periodic algorithm selection in time-changing data. pages 7 – 12, Piscataway, NJ, USA.

Russell, S. and Norvig, P. (2010). *Arificial Intelligence, A Modern Approach*. Pearson, third edition.

Stathopoulos, A., Dimitriou, L., and Tsekeris, T. (2008). Fuzzy modeling approach for combined forecasting of urban traffic flow. *Computer-Aided Civil and Infrastructure Engineering*, 23(7):521 – 535.

Sun, S. (2009). Traffic flow forecasting based on multitask ensemble learning. pages 961 – 964, Shanghai, China.

Tam, M. L. and Lam, W. H. K. (2007). Real-time travel time estimation using automatic vehicle identification data in hong kong. volume 4413 LNAI, pages 352 – 361, Jeju Island, Korea, Republic of.

van Hinsbergen, C., van Lint, J., and van Zuylen, H. (2009). Bayesian committee of neural networks to predict travel times with confidence intervals. *Transportation Research Part C: Emerging Technologies*, 17(5):498 – 509.

Van Lint, J. (2008). Online learning solutions for freeway travel time prediction. volume 9, pages 38 – 47.

Vanajakshi, L. and Rilett, L. (2007). Support vector machine technique for the short term prediction of travel time. In *Intelligent Vehicles Symposium, 2007 IEEE*, pages 600–605.

Wahl, R. and Haugen, T. (2005). DynamIT - Dynamiske Informasjonstjenester for Transportsektoren, Systembeskrivelse og evaluering. Technical report, SINTEF Teknologi og samfunn, Transportsikkerhet og -informatikk.

Wang, Z. and Poslad, S. (2013). A personalised online travel time prediction model. In *Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on*, pages 3327–3332.

Wu, T., Xie, K., Xinpin, D., and Song, G. (2012). A online boosting approach for traffic flow forecasting under abnormal conditions. In *Fuzzy Systems and Knowledge Discovery (FSKD), 2012 9th International Conference on*, pages 2555–2559.

Yang, Z.-L., Li, Y., Song, Y.-W., Chen, X., and Zhang, J. (2012). Empirical study of robust combination of forecasts for short-term highway traffic flow forecast. In *Machine Learning and Cybernetics (ICMLC), 2012 International Conference on*, volume 4, pages 1372–1375.

Yao, X., Fischer, M., and Brown, G. (2001//). Neural network ensembles and their application to traffic flow prediction in telecommunications networks. volume vol.1, pages 693 – 8, Piscataway, NJ, USA.

Zheng, F. and van Zuylen, H. (2013). Trip travel time distribution prediction for urban signalized arterials. In *Intelligent Transportation Systems - (ITSC), 2013 16th International IEEE Conference on*, pages 1829–1834.

Zhu, J. (2011). An ensemble learning short-term traffic flow forecasting with transient traffic regimes. *Applied Mechanics and Materials*, 97-98:849 – 53.

Zhu, J. and Shen, L. (2012). A real-time layered neural network ensemble learning system for short-term traffic prediction. *International Journal of Advancements in Computing Technology*, 4(10):18 – 26.

Zhu, T., Kong, X., and Lv, W. (2009). Large-scale travel time prediction for urban arterial roads based on kalman filter. In *Computational Intelligence and Software Engineering, 2009. CiSE 2009. International Conference on*, pages 1–5.

Øien Lunde, E. and Wolff, T. (2014). Travel time prediction, a state of the art review.

Appendices

A Structured Literature Review Protocol

This section describes in detail how the structured literature review was conducted. Two search engines were used: IEEEXplore[1] and Engineering Village[2].

A.1 Specifying the Search Term

When doing a structured literature review, it is necessary to have a search string that represents the subjects of interest. Having this search string serves two purposes: it makes it possible to reproduce the search results given the archive(s) used. In addition it captures the different aspects of the research, and therefore yields a reduced number of results, hopefully with the relevant literature.

The search string is built up using three groups of terms. The different terms contained in one group are synonyms, have the same semantics or cover similar concepts. The terms in one group are joined using the logical OR operator, and each group is joined using the logical AND operator. In this way the search string is meant to yield the research that is the intersection of the different groups. The search string used is:

$$\begin{gathered}
(\text{“prediction”}\ OR\ \text{“forecasting”}\ OR\ \text{“estimation”}) \\
AND \\
(\text{“travel time”}\ OR\ \text{“transit time”}\ OR\ \text{“driving time”}\ OR\ \text{“traffic flow”}\ OR\ \text{“congestion”}) \\
AND \\
(\text{“ensemble learning”}\ OR\ \text{“machine learning”}\ OR\ \text{“artificial intelligence”})
\end{gathered}$$

By examining this search string one can identify three aspects that this research focuses on: prediction or estimation of traffic variables using ensemble learning or machine learning in general.

[1] `http://ieeexplore.ieee.org/search/advsearch.jsp?expression-builder`
[2] `http://www.engineeringvillage.com/search/expert.url?CID=expertsearch`

A.2 Search Results

The search was conducted on the 24th of September, 2014. Engineering Village returned 670 results and IEEE returned 477 results. Because many of the articles are unlikely to be suitable for this research, a filtering process is employed to narrow down the number of articles being read in its entirety.

The filtering process consists of three stages: filtering by title, filtering by abstract and finally filtering by full text. In this way irrelevant literature is filtered out as early in the process as possible. The next sections explains in more detail how the filtering stages are executed.

A.3 Filtering by Title

This stage filters the articles by looking at their title. In this way articles that obviously do not concern the current research are excluded, and do not go further in the filtering process. An article is included if its title indicate that it:

Inclusion Criterion 1 Is about Intelligent Transportation Systems

or

Inclusion Criterion 2 Is about ensemble learning

or

Inclusion Criterion 3 Predicts, estimates or models road traffic variables, e.g. traffic flow, speed, congestion or travel time.

Inclusion criterion 1 ensures that articles in the ITS domain are included. These articles are valuable to this research, as they can give insight to the domain in general and contributes to the pool of knowledge in the area. By using inclusion criterion 2 articles about ensemble learning are included. The reason that research done on ensemble learning in the ITS domain is interesting, is that it that an ad hoc search prior to this literature review showed that ensemble learning is a possible direction to go in to achieve more accurate predictions [Russell and Norvig, 2010]. Inclusion criterion 3 ensures that research done on predicting or estimating traffic variables no matter what method that is employed is included. After filtering the articles by these three criteria, 401 articles remained in total.

A.4 Filtering by Abstract

After filtering the articles by title, a more thorough approach has to be employed to identify the valuable articles. For an article to be included in the last filtering stage, its abstract has to indicate one or more of the following:

Inclusion Criterion 1 The main focus of the article is using AI method(s) to predict travel time, traffic flow, speed or congestion.

Inclusion Criterion 2 That the article gives insight to which data sources that are relevant in the ITS domain.

Inclusion Criterion 3 The article gives insight to how data aggregation or pre-processing impacts prediction or estimation of travel time, traffic flow, speed of congestion.

Inclusion Criterion 4 That the article solves a problem using ensemble learning.

Inclusion Criterion 5 The article describe a solution that can easily be extended or adapted to fit our research.

Since investigating what machine learning techniques that have been used to predict traffic variables is of interest, inclusion criterion 1 is employed to include articles that would be relevant in that regard.

Preliminary discussions with the NPRA revealed that it would be interesting to use several data sources in the prototype system to increase prediction accuracy. Inclusion criterion 2 ensures that articles that can give insight to what data sources that are relevant to use is included.

There is an ongoing discussion at the NPRA about how to aggregate the data, and how the aggregation method affects the estimation of traffic variables. Currently, the most common technique used for travel times at the NPRA is to use an average of the travel times the last five minutes. Inclusion criterion 3 ensures that any literature that touches upon this issue is included.

As stated in the previous section, ensemble learning is one of the main focuses in this research and inclusion criterion 4 ensures that research that have employed ensemble learning to solve a problem is included.

If there is work that is done in the ITS domain whose solution can be extended or adapted to predicting travel time, we can draw knowledge from their findings. Inclusion criterion 5 ensures that such research is included.

Employing these filtering criteria yielded 294 articles. Given that this work is conducted by two students for a specialization project, it is infeasible due to time limitations, to read 294 articles in the last stage of the process. In order to further reduce the number of articles, a ranking system is developed. The ranking system consists of ten criteria by which the articles are rated. Each criterion is given an integer weight, and the total score of each article is the sum of all the weights that article has received for each criterion. The criteria, with the corresponding weights shown in parentheses, are as follows:

Ranking Criterion 1 The article is poorly written (-5)

Ranking Criterion 2 The article has empirical results (+3)

Ranking Criterion 3 The work concerns predicting travel time (+2)

Ranking Criterion 4 The models used are based on travel times for road sections, not GPS data etc. (+2)

Ranking Criterion 5 The work concerns short term prediction (+1)

Ranking Criterion 6 The work concerns urban/signalized roads (+1)

Ranking Criterion 7 The work employs online learning (+1)

Ranking Criterion 8 The work predicts traffic variables for public transport (−1)

Ranking Criterion 9 The work employs ensemble learning (+4)

Ranking Criterion 10 The work uses multiple data sources (+1)

Because of the previous stages of the filtering process, all the articles given as input to this stage have a certain degree of relevance, and this stage is meant to find the most relevant amongst them. As indicated by the weight of ranking criterion 2, the fact that the articles base their findings on empirical results are important, as this increases the credibility and reproducibility of their work. Ranking criterion 9 shows that articles concerned with ensemble learning are also favorable because they give insight to the one of the main topics in our study. Articles that are poorly written are given a big penalty, as seen by ranking criterion 1, as this makes them hard to read and time is wasted. The rest of the criteria correspond to features that would be beneficial to this research, but not essential.

After ranking all 294 articles using the aforementioned ten criteria, the total scores were in the range [-5, 10]. A threshold of 6 was set, where articles with a score of 6 or higher are included. This yielded 54 articles, a reasonable number of articles to read for two people. The number of articles was further reduced due to duplicates being removed and that some full texts were not available. The final number of articles entering the full text filtering was 36, and can be found in Tables 2 and 3. It can be seen in from the tables that all articles with a total score of 6 or better have empirical results, and most of them predict travel time in addition to having other interesting features like using online learning or employing short term prediction. From Tables 2 and 3 we can also see that articles that do not predict travel time can be included, as long as they have enough other interesting features.

A.5 Filtering by Full Text

In this final filtering stage the goal is to assess the quality of the work done in the remaining articles. In order to do this, another ranking system is developed where each article is evaluated using the following seven quality criteria [Kofod-Petersen, 2014]:

Quality Criterion 1 Are system or algorithmic design decisions justified?

Quality Criterion 2 Is the method/algorithm thoroughly explained?

Quality Criterion 3 Is the experimental procedure thoroughly explained and reproducible?

Quality Criterion 4 Is it clearly stated in the study which other algorithms the study's algorithm(s) have been compared with?

Quality Criterion 5 Are the performance metrics used in the study explained and justified?

Quality Criterion 6 Are the test results thoroughly analysed?

Quality Criterion 7 Does the test evidence support the findings presented?

Six of the above quality criteria are taken directly from [Kofod-Petersen, 2014]. Question 2 is added because it is important that the methods described are reproducible. Each question is given a weighted answer: yes (1), in some degree (0.5) and no (0).

To ensure that the different questions were interpreted in the same way by both students reading the articles, a calibration round was done on 8 of the articles where both students read and evaluated all 8 of them. After agreeing on how the questions should be interpreted, the remaining 30 articles were divided equally and read separately.

After assessing all articles, their score were in the range [2.5, 7]. As this range shows, not all articles had the same level of quality. To ensure a certain level of quality for the articles that are going to be included as our final references, a threshold of 5 is set, where articles with score higher than or equal to this threshold is included as references in our state of the art review. The final articles are shown in Table 1.

Final Articles	
Ensemble	[Sun, 2009]
	[Li et al., 2014]
	[Zhu and Shen, 2012]
	[Stathopoulos et al., 2008]
	[van Hinsbergen et al., 2009]
Online	[Liu et al., 2006a]
	[Liu et al., 2006b]
	[Van Lint, 2008]
	[Haworth et al., 2014]
	[Lu, 2012]
	[Wu et al., 2012]
Hybrid Approach	[Hofleitner et al., 2012]
Vehicle Infrastructure Integration	[Ma et al., 2012]
Other	[Park et al., 2014]
	[Bouillet et al., 2013]
	[Dharia and Adeli, 2003]
	[Nikovski et al., 2005]
	[Tam and Lam, 2007]
	[Vanajakshi and Rilett, 2007]
	[Mu et al., 2013]

Table 1: Articles resulting from the structured literature review

Title	RC1	RC2	RC3	RC4	RC5	RC6	RC7	RC8	RC9	RC10	Total Score
Li et al. [2009]	0	3	2	0	0	0	1	0	4	0	10
Ma et al. [2012]	0	3	2	2	0	0	1	0	0	1	9
Rossi et al. [2012]	0	3	2	0	0	0	0	0	4	0	9
van Hinsbergen et al. [2009]	0	3	2	0	0	0	0	0	4	0	9
Zhu [2011]	0	3	0	0	1	0	0	0	4	0	8
Yang et al. [2012]	0	3	0	0	1	0	0	0	4	0	8
Zhu and Shen [2012]	0	3	0	0	1	0	0	0	4	0	8
Chen and Chen [2007]	0	3	0	0	1	0	0	0	4	0	8
Wu et al. [2012]	0	3	0	0	0	0	1	0	4	0	8
Tam and Lam [2007]	0	3	2	2	1	0	0	0	0	0	8
Sun [2009]	0	3	0	0	0	0	0	0	4	0	7
Mu et al. [2013]	0	3	2	0	1	0	0	0	0	1	7
Mak et al. [2009]	0	3	2	0	0	1	0	0	0	1	7
Li et al. [2014]	0	3	0	0	0	0	0	0	4	0	7
Liu et al. [2006a]	0	3	2	0	1	1	0	0	0	0	7
Zheng and van Zuylen [2013]	0	3	2	0	0	1	0	0	0	1	7
Chen and Zhang [2005]	0	3	0	0	0	0	0	0	4	0	7
Deb Nath et al. [2010]	0	3	2	2	0	0	0	0	0	0	7
Dembczynski et al. [2013]	0	3	2	2	0	0	0	0	0	0	7
Yao et al. [2001]	0	3	0	0	0	0	0	0	4	0	7

Table 2: Articles included in the quality screening process, part I

Title	RC1	RC2	RC3	RC4	RC5	RC6	RC7	RC8	RC9	RC10	Total Score
Bouillet et al. [2013]	0	3	2	0	0	0	1	0	0	1	7
Park et al. [2014]	0	3	0	2	1	0	0	0	0	1	7
Hofleitner et al. [2012]	0	3	2	0	0	1	0	0	0	0	6
Bajwa et al. [2003]	0	3	2	0	1	0	0	0	0	0	6
Dharia and Adeli [2003]	0	3	2	0	0	0	0	0	0	1	6
Zhu et al. [2009]	0	3	2	0	0	1	0	0	0	0	6
Wang and Poslad [2013]	0	3	2	1	0	0	0	0	0	0	6
Vanajakshi and Rilett [2007]	0	3	2	0	1	0	0	0	0	0	6
Ohra et al. [1997]	0	3	2	0	0	0	1	0	0	0	6
Van Lint [2008]	0	3	2	0	0	0	1	0	0	0	6
Hao et al. [2008]	0	3	2	0	0	1	0	0	0	0	6
Haworth et al. [2014]	0	3	2	0	0	1	0	0	0	0	6
Stathopoulos et al. [2008]	0	3	0	0	1	1	1	0	0	0	6
Lu [2012]	0	3	2	0	0	0	1	0	0	0	6
Liu et al. [2006b]	0	3	2	0	0	1	0	0	0	0	6
Nikovski et al. [2005]	0	3	2	0	1	0	0	0	0	0	6

Table 3: Articles included in the quality screening process, part II