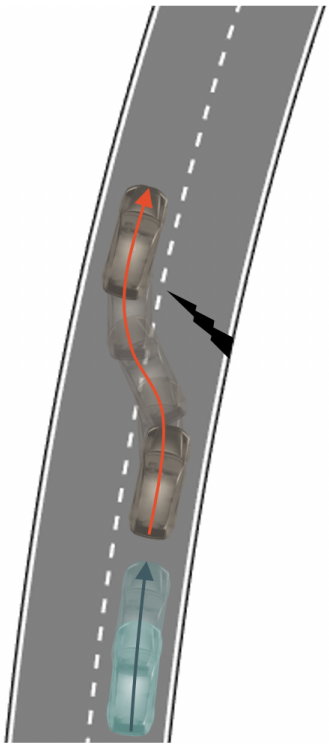

Detection of Evasive Maneuvers for Surrounding Vehicles

A machine learning approach to classification of evasive events

Master's thesis in Engineering Mathematics and Computational Science &
Systems, control and mechatronics

SEBASTIAN OLESZKO
ALEXANDER WÖLFINGER

DEPARTMENT OF MATHEMATICAL SCIENCES

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2021
www.chalmers.se

Master's thesis 2021

Detection of Evasive Maneuvers for Surrounding Vehicles

A machine learning approach to classification of evasive events

SEBASTIAN OLESZKO
ALEXANDER WÖLFINGER

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mathematical Sciences
Division of Applied Mathematics and Statistics
Chalmers University of Technology
Gothenburg, Sweden 2021

Detection of Evasive Maneuvers for Surrounding Vehicles
A machine learning approach to classification of evasive events
SEBASTIAN OLESZKO
ALEXANDER WÖLFINGER

Supervisor: Adam Lilja, Department of Advanced Safety, Aptiv
Supervisor: Konstantinos Konstantinou,
Department of Mathematical Sciences, Chalmers
Examiner: Aila Särkkä, Department of Mathematical Sciences, Chalmers

Master's Thesis 2021
Department of Mathematical Sciences
Division of Applied Mathematics and Statistics
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: A possible scenario where a target vehicle preforms an evasive maneuver to avoid a potential danger, in this case a damaged road section. The following vehicle's sensor will have an obstructed view of the possible danger and ADAS can not react in time.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2021

Detection of Evasive Maneuvers for Surrounding Vehicles
A machine learning approach to classification of evasive events
SEBASTIAN OLESZKO
ALEXANDER WÖLFINGER
Department of Mathematical Sciences
Chalmers University of Technology

Abstract

Advanced Driver-Assistance Systems is a great aid to avoid traffic accidents, but today it relies on sensor sight. An evasive maneuver from a target vehicle is an example of a scenario where these systems have limitations. This thesis investigates the possibility to classify an evasive maneuver using a machine learning approach with a limited test-track dataset combined with a real-world dataset. Time series classification and data augmentation are used for different machine learning models for comparison.

The result shows that the models are capable of predicting evasive maneuvers, however, it is clear by these results that further work to advance these models should be made. Data augmentation for time series data showed promising results in increasing the variety of input data and therefore increasing the limited dataset.

A conclusion is made that a search algorithm that finds evasive maneuvers in larger datasets can be developed using the methods that are implemented in this thesis. Thus reducing data limitations that were an issue in this thesis. Future work could expand upon the models to include more complex features such as traffic interaction.

Keywords: ADAS, Evasive maneuver, Time series classification, Data augmentation

Acknowledgments

Firstly we would like to express our gratitude to our supervisor at Aptiv, Adam Lilja for the endless help and input with useful comments, remarks and engagement through the learning process of this master thesis. Furthermore we would like to thank Joachim Rukin for introducing us to the problem statement as well for the support during the process. Also, we would like to thank Aptiv for giving us the trust and support to pursue this project and providing us the tools to do so.

Moreover, we also want to express our gratitude to our supervisor and examiner at Chalmers, Konstantinos Konstantinou and Aila Särkkä that took on this work and helping us on the way, this thesis would not have been possible without your work and dedication.

Sebastian Oleszko & Alexander Wölfinger, Gothenburg, May 2021

Contents

1

Introduction

Autonomous driving and Advanced Driver-Assistance Systems (ADAS) have come a long way in recent years, systems such as Automated Emergency Steering (AES) and Automated Emergency Braking (AEB) can help drivers to avoid accidents in traffic. These systems can be a great asset but also have limitations, such as sight. Most commonly both AES and AEB use radar or vision systems and will react in the last second to an obstacle that is in the vehicles heading direction.

An issue with this sight limitation can be that the radar and vision system on the host vehicle is obstructed by another vehicle, as shown in Figure 1.1, where there is a stationary object in front of the target vehicle. When this target vehicle naturally avoids the obstacle a peel and reveal scenario appears, the host vehicle in this scenario can then have moved into the dangerous area shown in Figure 1.1 before the systems can detect and respond to the stationary object. There can be multiple different scenarios where a target vehicle performs an evasive maneuver to avoid danger including a stationary/slow-moving object, damaged road surface or wildlife.

Previous studies have been made regarding human-driver behavior and lane changing scenarios [1], where several suggest a Hidden Markov Model (HMM) approach to determine dangerous situations with surrounding vehicles. Many studies have been carried out with respect to the host vehicle's human-driver behavior. To be able to predict and detect a scenario presented in Figure 1.1 it is desirable to present an approach with respect to a target vehicle and the specific behavior in an evasive maneuver to be able to contribute features for existing automated emergency systems.

1.1 Purpose

This work aims to investigate whether it is possible to model a target vehicle human-driver evasive maneuver and classify such behavior. We aim to compare different machine learning approaches, probabilistic and neural network based, and compare these models to evaluate to what level these can be used for classifying evasive maneuvers on target vehicles. Furthermore, an investigation into how fast the models can detect an evasive maneuver will be performed, where this can be used as one metric to evaluate the performance of the models.

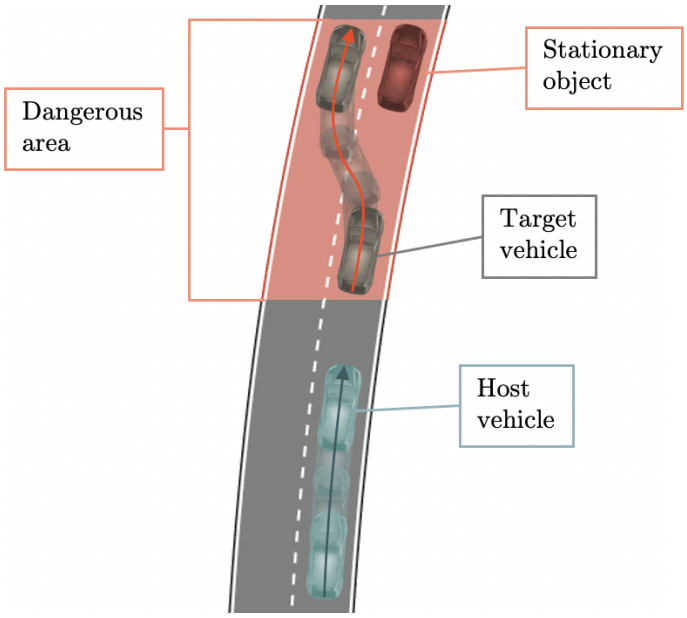

Figure 1.1: A possible scenario where a target vehicle performs an evasive maneuver to avoid a stationary object on the road, the following vehicle's (host vehicle) safety system has an obstructed view on the stationary object and can not detect and respond before moving into the marked dangerous area where it won't have enough time to perform emergency braking or emergency steering.

In addition, a pre-study is to be carried out where a thorough exploration of the data and its feature content, looking into noise level and comparing this between host vehicle, target vehicle and real-world driving scenarios. Input features will also be explored with feature selection methods and feature engineering and additional data augmentation methods.

Previous studies on human-driver behavior are mostly investigating test-track data and/or simulated data [1]. In this study, we aim to use test-track data incorporated with real-world data[1] to investigate how these models would perform in online implementations on ADAS and use real-world data as one metric in evaluating the models.

1.2 Limitations

An evasive maneuver is an uncommon event in real-world driving scenarios, and a dangerous scenario to simulate on a test-track for data logging. This creates a clear limitation on the dataset size that can be used to train the models in this study, which needs to be taken into account when choosing models for investigation. Simulation software can be used to extend the dataset, this is requiring a model for the evasive maneuver. Therefore software simulation will not be performed during this study.

[1]Data logs collected on varied driving scenarios on highways, country roads and city traffic.

Within the area of machine learning, there are numerous model approaches that can be investigated and a few will be argued for and investigated here, since as already stated, data limitation is a factor that needs to be taken into account. Complex models and deep neural networks that need large datasets will not be investigated.

1.3 Related work

The survey "A Survey of Autonomous Driving: Common Practices and Emerging Technologies" [1] outlines different studies regarding autonomous driving including human-driver behavior. The specific case of an evasive maneuver has not been thoroughly studied, however, studies regarding surrounding human-driver intention scenarios which can be argued to be similar have been carried out. A scenario-adaptive approach to predicting surrounding vehicle's driving behaviors in dynamically changing traffic scenarios was proposed in [2]. This study is using HMM's to approach this problem and to enhance previous models. Another study [3] also uses HMM to predict surrounding vehicle's driving behaviors, in this case for long-term scenarios. A study in dangerous cut-in maneuvers was carried out in [4] using an unsupervised HMM with the Baum-Welch algorithm. It uses a dataset with dangerous and normal lane-change scenarios to determine the probability of the potential danger caused by the cut-in case. In [5] a novel approach based on Support Vector Machine and Bayesian filtering is proposed for lane-change intention prediction, where the results show that driver intention to change lanes was on average predicted 1.3 seconds in advance. An approach with Bayesian network classifier[6] was used to predict maneuvers of individual drivers on highways.

1.4 Structure of the report

The report begins with a background chapter, containing a theoretical formulation of the evasion classification problem and the theoretical background for the models that were studied. This is followed by a chapter on the data that was used in training and evaluating the models discussed in the background. Specifically, methods of data collection, feature selection and engineering, and data augmentation were introduced. The subsequent chapter introduces the implementation methods and how the models were evaluated and compared. The results of the different models, augmentation techniques and variations on the dataset are presented along with interpretations of the results and feasibility of the models in real-world scenarios. Lastly, the results and insights from this report are discussed along with conclusions and the authors' ideas for further research on the topic.

2

Background

This chapter introduces the methods that were used for evasion classification and the theoretical background.

2.1 Evasion classification

Determination of the evasive state was modeled as a classification task of the observed vehicle's intent. Principally, the evasive state can be considered as a latent unobservable state which is to be inferred from observable data. Due to limitations in the amount of data, it has been assumed that the input space is not thoroughly covered, thus, unsupervised learning approaches have been dismissed in favor of a supervised learning approach where labels are added by the subjective understanding of the authors of this thesis. Furthermore, since the event and data are time-dependent, a time series classification was modeled. The definition of the problem, which is the foundation of the modeling throughout Chapter 2, is stated as follows.

Let $\mathcal{X} \subset \mathbb{R}^p$ be the input space, where p is the number of features and $\mathcal{Y} = \{0, 1\}$ be the possible labels of the latent evasive state, where 0 corresponds to normal driving and 1 corresponds to an evasive maneuver. The task of a time series classifier is to find a function $f : \mathcal{X}^T \mapsto [0, 1]^T$ which maps the inputs of a time series of length T to the probability distribution of the labels in that time series. Hence, an optimization problem which is defined by the loss function $\mathcal{L}$ is formulated. In its simplest form, a time series classification problem with 0-1 loss $\mathcal{L}(\hat{y}, y) = I(\hat{y} \neq y)$, where I is the indicator function, $y_i \in \mathcal{Y},\ i = 0, ..., t$ are the labels and $\hat{y}_i \in \mathcal{Y},\ i = 0, ..., t$ are the predictions at time t, has an optimal solution

$$\hat{y}_t = \arg\max_{j \in \mathcal{Y}} P(j|x_0, ..., x_t), \tag{2.1}$$

where $x_0, ..., x_t \in \mathcal{X}$ are the t first observations of the time series. The aim of the models presented in this report is to find an optimal solution to this problem given one or a number of loss functions. Throughout this report, observations from the time series will be denoted $x_t \in \mathcal{X}$ and the corresponding evasion labels $y_t \in \mathcal{Y}$ where $t = 0, ..., T$.

2.2 Support Vector Machine and Bayesian Filter

The supervised learning technique Support Vector Machine (SVM) is a discriminative classification method that is used as a baseline for comparison. The method aims to find the optimal hyperplane to separate the data into the labeled classes and has previously been successfully applied in the field of driver intent prediction [7]. Below follows a brief problem definition and mathematical formulation of the method.

Given observations x_t and binary labels y_t the goal is to find the weights w and biases β such that the optimization problem

$$\min_{w,\zeta_{0:T}} \quad \frac{1}{2}w^T w + C\sum_{t=0}^{T}\zeta_t \tag{2.2}$$

$$\text{s.t.} \quad y_t(w^\top \phi(x_t) + \beta) \geq 1 - \zeta_t \tag{2.3}$$

$$\zeta_t \geq 1,\ t = 1, ..., T \tag{2.4}$$

is minimized. Additionally, slack variables ζ_t are introduced to allow for errors in the prediction where a completely separating hyperplane cannot be found. The parameter $C \in \mathbb{R}^+$ thus becomes the inverse regularization factor for the optimization problem. To allow for non-linear relationships, a mapping $\phi : \mathcal{X} \to \mathcal{V}$ of the input vectors to another space $\mathcal{V}$ is applied. The mapping is implicit and is implemented through a kernel $K : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}$. The only restriction on the mapping is that $\mathcal{V}$ needs to be an inner product space. The kernels that were used were, the linear kernel

$$K_{\text{Linear}}(x, x') = \langle \phi(x), \phi(x') \rangle = x^\top x' \tag{2.5}$$

and the radial-basis function (RBF) kernel

$$K_{\text{RBF}}(x, x') = \langle \phi(x), \phi(x') \rangle = \exp\left(-\gamma ||x - x'||_2^2\right) \tag{2.6}$$

where γ is a hyperparameter for the kernel and $||\cdot||_2$ is the l^2-norm.

Support Vector Machines assumes that observations are independent thus not making the method ideal to model time-dependent data. One way of introducing time dependency is through a Bayesian filter that takes into account the previous predictions of the latent state y_t. The implementation of the Bayesian filter to the SVM classifier was based on the work of [7].

The Bayesian filter takes a sequence of predictions $\hat{y}$ from the SVM classifier and regards them as samples of the random variable $Y \in \mathcal{Y}$ where

$$P(Y = 0|\theta) = \theta \tag{2.7}$$

is the probability of the target not performing an evasive maneuver. The problem can thus be formulated as a problem of estimating the expected value of the parameter θ. With the aim of finding the posterior $P(\theta|\hat{y})$, we assume that the output of

the SVM follows a binomial distribution and choose a beta distributed prior on the parameter. Hence, since

$$\theta \sim \text{Beta}(a, b) \tag{2.8}$$

$$y \sim \text{Binomial}(T, \theta), \tag{2.9}$$

conjugacy gives the posterior with regards to these distributions

$$P(\theta|\hat{y}) = \frac{P(\hat{y}|\theta)P(\theta|a,b)}{P(\hat{y})} = \frac{\Gamma(n+a+m+b)}{\Gamma(n+a)\Gamma(m+b)}\theta^{n+a-1}(1-\theta)^{m+b-1}, \tag{2.10}$$

where $n = \sum_{t=0}^{T} I(\hat{y}_t = 0)$, $m = \sum_{t=0}^{T} I(\hat{y}_t = 1)$, Γ is the Gamma function and I is the is the indicator function. We can then compute the expectation

$$E(\theta|\hat{y}) = \int_0^1 \theta P(\theta|\hat{y}) d\theta = \frac{n+a}{n+a+m+b}. \tag{2.11}$$

In order to improve the accuracy of the filter, [7] suggests the use of a discount function, which gives less weight to classifications further back in time. This is extra important in imbalanced classification tasks, such as this one, since $n \gg m$ for a large time window. Consequently, the most probable class would be $\hat{y}_t = 0$ unless the time window length T is sufficiently small. The discount function is defined as

$$d_t = c^{T-t}, \quad d_0 = c^T \tag{2.12}$$

for $t = 1, ..., T$ where $0 < c < 1$ is a constant factor of exponential decrease in weight. With this addition, Equation (2.11) can be re-written as

$$E(\theta|\hat{y}) = \frac{\sum_{t=1}^{T} d_t I(\hat{y}_t = 0) + d_0 a}{\sum_{t=1}^{T} d_t I(\hat{y}_t = 0) + d_0 a + \sum_{t=1}^{T} d_t I(\hat{y}_t = 1) + d_0 b} \tag{2.13}$$

where each observation is individually weighted with the discount factor. The final step in the algorithm is then to find a threshold τ for the expectation in Equation (2.13), which balances the precision and recall of the classification. At each time-step, the expectation is computed and compared against this threshold, resulting in the final classification.

2.3 Recurrent Neural Network

A Recurrent Neural Network (RNN), first introduced by Elman [8], is a non-linear supervised learning method that has been successfully applied in situations where data is sequential [9] such as time series classification. The method can be modeled as a directed graph along the temporal axis of the data, where the memory of previous nodes is preserved through hidden units. These properties make the model suitable for evasive maneuver prediction since it both captures non-linear properties of the data and its temporal dependency.

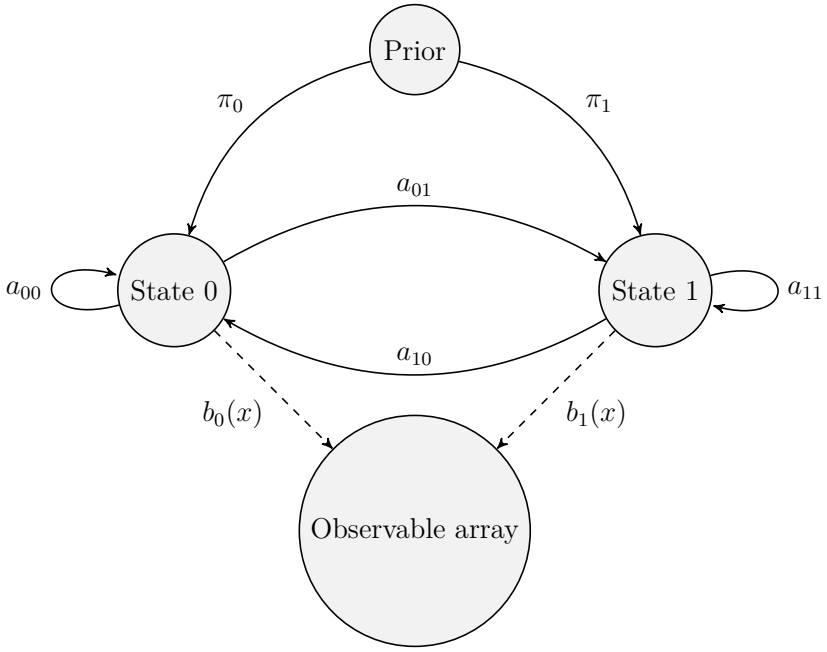

Figure 2.1: Representation of a Hidden Markov Model (HMM), $\lambda = (\pi, \mathbf{A}, \mathbf{B})$. This model has two states, $j = 0, 1$, where 0 corresponds to normal driving and 1 to an evasive maneuver. $\pi_j \in \pi$ is the initial state probability distribution, $a_{ij} \in \mathbf{A}$ is the state transition probability distribution and $b_j(x) \in \mathbf{B}$ is the observation symbol probability distribution.

An extension of the Elman Recurrent Neural Network is introduced in [10] as a Long Short-Term Memory (LSTM) Recurrent Neural Network. The authors of this paper introduced a variant of the neural network which is more capable of capturing both long-term relationships as well as shorter ones. The LSTM architecture was chosen for evasion classification due to the high sampling rate of the observations, thus requiring the ability to capture longer-term patterns in the data.

An apparent limitation with the method comes from the difficulty of training the weights of the neural network. The objective function of the optimization problem can often be non-convex and stochastic, meaning that finding an optimal solution could be difficult. Furthermore, since classification of evasive maneuvers is a problem with a relatively large limitation on dataset size, convergence of the optimization algorithm is difficult to achieve.

2.4 Hidden Markov Model

Hidden Markov Model, further referred to as HMM is a probabilistic model that assumes that the model is a Markov process with some observable variables x_t that are dependent on hidden states y_t. A HMM can be derived by $\lambda = (\pi, \mathbf{A}, \mathbf{B})$, where π is the initial state probability distribution, $\mathbf{A}$ is the state transition probability distribution and $\mathbf{B}$ is the observation symbol probability distribution. [11]

The implementation of an HMM can be done using both supervised and unsupervised learning, since the training data used in this project has been labeled, the approach described here is supervised. In Figure 2.1 an illustration of HMM ($\lambda = (\pi, \mathbf{A}, \mathbf{B})$) is shown. The initial state probability distribution $\pi_j \in \pi$, ($j \in \{0, 1\}$), describes the initial distribution of the hidden state

$$\pi_j = P(y_0 = j) \quad j \in \{0, 1\}, \tag{2.14}$$

where y_0 is the state for the first time step ($t = 0$). The state transition probability $a_{ij} \in \mathbf{A}$ ($i, j \in \{0, 1\}$) is the probability of a transition between hidden states

$$a_{ij} = P(y_t = j | y_{t-1} = i) \quad i, j \in \{0, 1\},\ 1 \leq t \leq T. \tag{2.15}$$

The observation symbol probability $b_j(x_t) \in \mathbf{B}$, where x_t is the observation sequence of the different features for time $t \in T$

$$b_j(x_t) = P(x_t | y_t = j) \quad j \in \{0, 1\},\ 0 \leq t \leq T. \tag{2.16}$$

The observation symbol probability is implemented as a probability density function. Since the data used in this study does not necessary have a normal Gaussian distribution but a sum of multiple distributions, the implementation that is used is a Gaussian kernel distribution [12].

$$\hat{b}'_j(x) = \frac{1}{nh} \sum_{l=1}^{n} K\Big(\frac{x - x_l}{h}\Big), \tag{2.17}$$

where n is the total number of data points and h is the bandwidth (further discussed below). $\hat{b}'_j(x)$ denotes the non normalized observation symbol probability, and $\hat{b}_j(x) = \frac{\hat{b}'_j(x)}{\sum_{j=0}^{1} \hat{b}'_j(x)}$. K is the kernel, which is a standard normal distribution with $\mu = 0,\ \sigma^2 = 1,\ f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$. Merged with Equation (2.17) results in

$$\hat{b}'_j(x) = \frac{1}{nh} \sum_{l=1}^{n} \Big(\frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x - x_l}{h})^2} \Big). \tag{2.18}$$

The bandwidth h should be selected as small as the data allows with respect to trade-off with bias and variance. Several different algorithms can be used to estimate an optimal trade-off for the bandwidth, one common approach is the Mean Integrated Squared Error

$$MISE(h) = E\Big[\int (\hat{b}_j(x) - b_j(x))^2 dx \Big], \tag{2.19}$$

which is the error between the Gaussian kernel distribution and the real distribution. Naturally the real distribution is unknown so h needs to be selected. And can be done by Silverman's rule of thumb [13]

$$h = \Big(\frac{n(p + 2)}{4} \Big)^{\frac{-1}{p+4}}, \tag{2.20}$$

where, n is the total number of data points and p is the number of features.

2.5 Random Convolutional Kernel Transform

Building on the success of convolutional neural networks for time series classification, RandOm Convolutional KErnel Transform (ROCKET) is a method that uses random convolutional kernels and simple linear classifiers to achieve state-of-the-art results [14]. In essence, the ROCKET in combination with logistic regression acts as a single layer convolutional neural network followed by a softmax layer. However, as suggested by [14], a ridge regression classifier with parameters chosen through cross validation was used instead of logistic regression.

The choice of this method to model evasive maneuvers comes from the observation that it may be difficult for a convolutional neural network to learn "good" kernels on small datasets [14]. Thus, using a neural network approach to find complex patterns may result in bad generalization. By using random kernels with weights that are not trained on the dataset, there is a larger chance that some of these kernels extract the relevant features from the data without learning as many parameters.

An inherent limitation of the ROCKET method in the context of this thesis is that ROCKET is a many-to-one time series classification method where a series of inputs results in a single classification. In comparison, the real-time problem of detecting evasive maneuvers is a many-to-many problem where data should be processed at every time step and return a value that represents the probability that an evasive maneuver is happening. A solution to this problem is to use a sliding window approach where the previous T observations constitute the input time series, the disadvantage being that there will be a significant time delay ($T - 1$ time steps) before a classification is produced. Consequently, the sliding window size T is a hyperparameter for real-time processing but the method was primarily used in the event finding algorithm.

3

Data

The problem is defined as a supervised learning problem where data is collected by the host vehicle to predict a latent binary state "Evasion". Thus, a dataset of relevant descriptors is required in addition to labels that represent the evasive state at the time of data collection. The datasets are presented in Table 3.1, all of which were provided by Aptiv. A combination of test track data collection and real-world driving was used both for training and validating the models.

3.1 Data collection

Collection of data logs have been performed on three occasions, stated in Table 3.1, where target evasion 1 was performed on AstaZero test-track in Borås, Sweden. Target evasion 2 and Target evasion test were performed on a test-track in Krakow, Poland. All data collection used the same set-up of radars, camera and software/-tracker version. For the real-world driving dataset, a collection of data logs were collected from varied driving scenarios in Gothenburg, Sweden. For this dataset, the sensor set-up is the same as with the test-track logs but not necessarily the same version of the software/tracker.

The sensor set-up can be seen in Figure 3.1, the set-up contains four corner short range radars, one forward mid range radar and one forward camera. The raw-data output from these sensors was then processed and filtered through the tracker

Table 3.1: Datasets that were used for training and testing the models. Additionally, the number of evasive scenarios in each dataset and the number of observations are presented. Target evasion 1, 2 and test are a dataset collected on test tracks with labeled evasive maneuvers. Real-world driving is a dataset collected on varied driving scenarios on highways, country roads and city traffic.

Dataset	Evasive scenarios	Total observations
Target evasion 1 (track)	23	9367
Target evasion 2 (track)	30	14167
Target evasion test (track)	36	22834
Real-world driving	0	87364

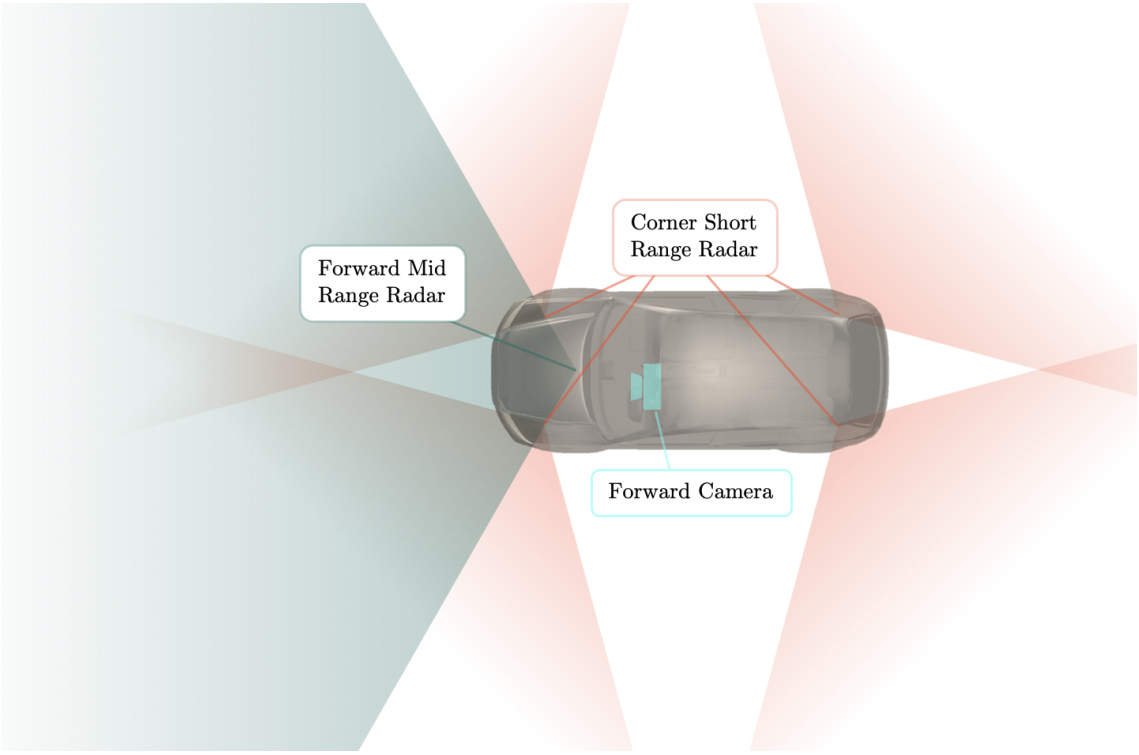

Figure 3.1: Sensor set-up on data collecting vehicle consisting of four corner short range radars, one forward mid range radar and one forward camera.

software that outputs different features for host and surrounding target vehicles.

3.2 Features

Features were chosen using feature selection methods (Section 3.2.1) and prior knowledge, that was confirmed during exploratory data analysis of the dataset. The coordinate system is originated on the center of the front bumper on the host vehicle, shown in Figure 3.2, with side-ways movement in the lateral domain and forward/backward movement in the longitudinal domain. It was found that features in the lateral domain showed a distinctive pattern during an evasive maneuver and that lateral acceleration of the target vehicle proved significant. This finding was expected since an evasive maneuver could be defined as an unusually strong and rapid reaction of the steering wheel, thus making features that are related to this action strong predictors. In Table 3.2 the features extracted from the tracker software can be seen, this set of features is selected from prior knowledge and later reduced with feature selection algorithms.

3.2.1 Feature selection

The logs used in this project have a large number of features, therefore a feature selection method is applied. The stepwise forward/backward method is a simple greedy algorithm (it will find local min/max values) that uses a forward algorithm

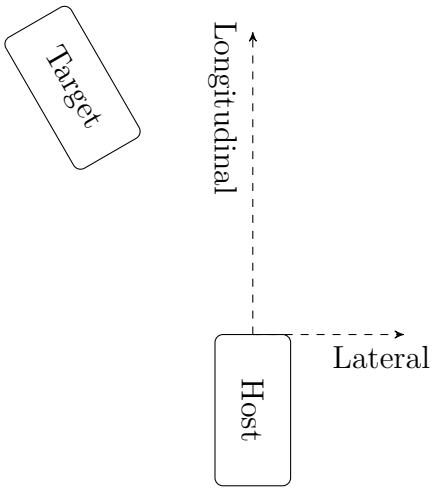

Figure 3.2: Coordinate system for the feature data, coordinates follows the host vehicle with origin on the center of the front bumper. Target vehicle follows the same coordinate system, with respect to the host vehicle. Side-way features move in the lateral domain and forward/backward features move in the longitudinal domain.

Table 3.2: Features that were extracted from the datasets. The features are divided into three categories; "Host" encompasses data from the host vehicle, "Target" features from each car in the observed area and "Meta" data about the sensors'/algorithms' confidence about each observation.

	Name	Description	Unit	Domain
Host				
	x_0	Lateral acceleration	[m/s^2]	$\mathbb{R}$
	x_1	Longitudinal acceleration	[m/s^2]	$\mathbb{R}$
	x_2	Speed	[m/s]	$\mathbb{R}$
	x_3	Side slip	[rad]	$[-\pi, \pi]$
	x_4	Steering angle	[rad]	$[-\pi, \pi]$
Target				
	x_5	Lateral acceleration	[m/s^2]	$\mathbb{R}$
	x_6	Longitudinal acceleration	[m/s^2]	$\mathbb{R}$
	x_7	Lateral velocity	[m/s]	$\mathbb{R}$
	x_8	Longitudinal velocity	[m/s]	$\mathbb{R}$
	x_9	Lateral position	[m]	$\mathbb{R}$
	x_{10}	Longitudinal position	[m]	$\mathbb{R}$
	x_{11}	Heading	[rad]	$[-\pi, \pi]$
	x_{12}	Width	[m]	$\mathbb{R}^+$
	x_{13}	Length	[m]	$\mathbb{R}^+$
	x_{14}	Pointing	[rad]	$[-\pi, \pi]$
Meta				
	x_{15}	Fusion confidence	[−]	$[0, 1]$

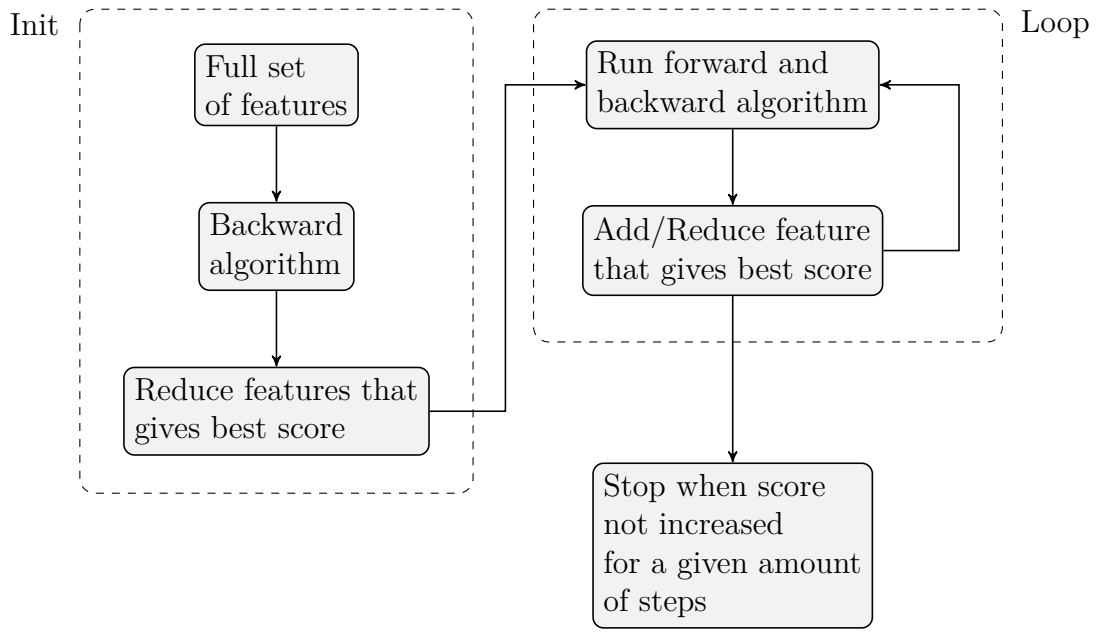

Figure 3.3: Implementation of stepwise forward/backward algorithm. The algorithm starts with a set of all available features and performs the backward algorithm to reduce the set of features, by training a model with one less feature and performing this for all features in the set. At the end of the initiation part, the algorithm removes the features that output the highest error. Moving into the loop part, where both forward and backward algorithm is preformed and compared, which results in adding or removing the features that produced the best score, this is then repeated until a threshold is reached.

combined with a backward algorithm. The forward algorithm starts with training a non-specific model with one feature at a time for all features and adds the trained model with the least error to the new feature set that is being built. This new feature set is then extended by training the model used with the previously selected feature in addition to one extra from the set of features not yet being used, adding the model containing the features with the least error each time until a specified threshold has been reached, in this case, the threshold will be reached when the F_1 score (see Section 4.2) has not been improved for 5 steps. The backward algorithm starts with a full set of features and removes the feature in each step that has the highest error, and eventually stops when the set threshold is reached. When using stepwise forward/backward, one step performs both backward and forward and chooses the step that results with the lowest error, as can be seen in Figure 3.3.

3.2.2 Feature Engineering

During the exploration of the data, an issue with the translation of the coordinate system for target features was detected. The motion/position features of the target vehicle (position, velocity and acceleration) are expressed in a coordinate system relative to the host vehicle (see Figure 3.2). Thus, a change of coordinate system is needed to observe the target as an independent entity. An example of a situation

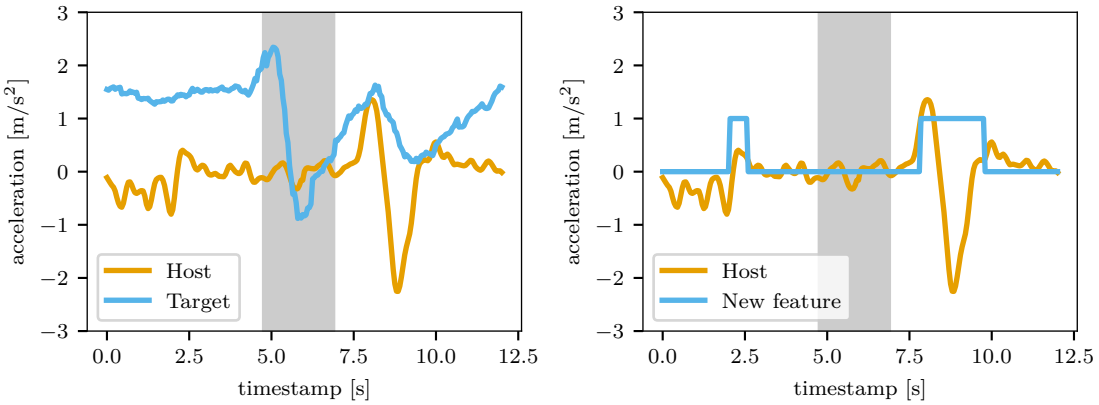

Figure 3.4: Example of the host acceleration influencing observed acceleration of the target vehicle. The left figure shows both the host and target lateral acceleration around an evasive maneuver (the shaded area). In this plot, the influence of the host maneuver can be observed in the target acceleration. The right plot shows the engineered feature (host lateral movement) together with the host lateral acceleration.

where this becomes a problem is to observe a constant speed target vehicle while the host vehicle performs a turn. The raw sensor data will observe the target as performing an equivalent turn, except mirrored. Such a scenario can be seen in Figure 3.4 (left) where, both the target's and host vehicle's lateral acceleration is plotted. The target vehicle performs an evasive maneuver in the gray area which is indicated by the large jump in lateral acceleration. However, since the host vehicle follows, it performs a similar evasive maneuver slightly thereafter which has an observable effect on the target acceleration as well.

Due to difference in how the motion features are computed, a direct translation of the coordinate system was not possible. The solution was to engineer a feature that would be used in combination with the target acceleration data. Using an ad-hoc solution that was constructed by inspecting the host vehicles lateral acceleration and attempting to detect large values or changes, the feature was constructed as

$$x_t^{new} = \begin{cases} 1, & \text{if } |x_t^0| > 1 \quad \text{or} \quad \sum_{i=0}^{9} |x_{t-i}^0 - x_{t-i+1}^0| > 1 \\ 0, & \text{otherwise} \end{cases} \qquad t \in \mathbb{N},\ 9 \leq t \leq T. \quad (3.1)$$

From visual inspection of the time series around a host evasive maneuver, shown in the right plot in Figure 3.4, the feature works as intended.

3.3 Augmentation

Crucial to the success of most machine learning methods is the use of large amounts of quality data. This is especially true for deep learning methods which tend to

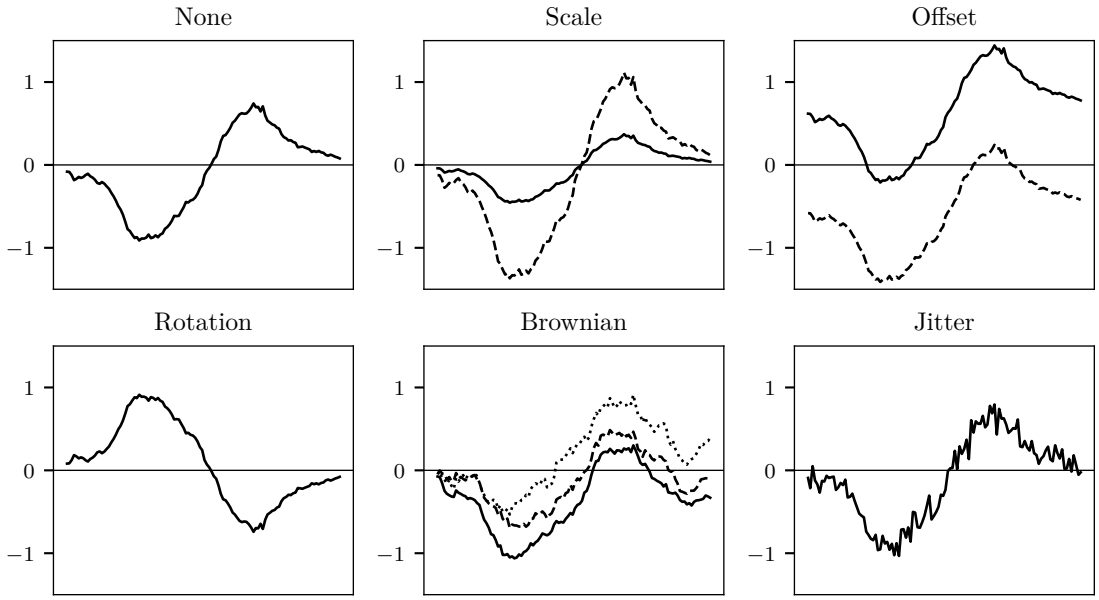

Figure 3.5: Examples of the five data augmentation techniques: *scale*, *offset*, *rotation*, *brownian noise*, *jitter*. The curves in the plots are time series data of the lateral acceleration for a target vehicle during an evasive maneuver that has been augmented with the corresponding methods.

overfit or not generalize well to unseen data without having very large datasets during training. Furthermore, observations of evasive maneuvers in real-world driving seem to be remarkably sparse, requiring the use of artificial methods such as collecting data of evasive scenarios on a test track or through simulation. A solution to lacking data quantity is to utilize data augmentation to create new synthetic data. Data augmentation has been shown to be an effective method of increasing the explored input space without requiring new labels or data collection [15]. Thus, data augmentation was performed in an attempt to increase the size and quality of the dataset.

Basic approaches for time series data augmentation falls into one of three groups: time domain, frequency domain or time-frequency domain [15]. In this thesis, only time domain transformations were applied due to their simplicity and natural interpretability, which allowed for reasoning around which features could be augmented without risking generalizability. Five transforms were applied: *rotation*, *scaling*, *brownian noise addition*, *offset*, *jitter* which can be seen in Figure 3.5. The rotation transform was applied by flipping the sign of the original time series

$$x'_t = -x_t, \tag{3.2}$$

however, this is only possible if there is symmetry in the feature such as for lateral features. Jittering creates synthetic sensor noise and is implemented by adding random Gaussian noise

$$x'_t = x_t + 0.1\epsilon_t \quad \epsilon_t \sim \mathcal{N}(0, 1), \tag{3.3}$$

which was scaled down by the factor 0.1 to match the scale of real observed sensor noise. The jitter noise is independent of previous time-steps, however, a version of correlated random-walk noise (Brownian noise) was implemented by adding a random-walk to the original time-series

$$x'_t = x_t + \sum_{i=0}^{t} 0.03\epsilon_i \quad \epsilon_i \sim \mathcal{N}(0,1), \tag{3.4}$$

where again a scale factor (0.03) was used to maintain realistic feasibility of the resulting time series. The motivation behind Brownian noise comes from observing that the size and position of vehicles in the tracker tend to "sway" over time. Thus the idea is that this motion resembles a random walk for short time windows, and due to the relatively short time length of the evasive maneuvers time series, the noise could imitate this behavior.

The scaling was achieved by randomly multiplying the entire feature vector by a constant factor of either 0.5 or 1.5,

$$x' = \begin{cases} 0.5x, & z = 0 \\ 1.5x, & z = 1 \end{cases} \tag{3.5}$$

where $z \sim$ Bernoulli(0.5). The choice of scaling factor was chosen to be a large enough to introduce variety to the input space but still remain physically feasible during an evasive scenario. Finally, the offset augmentation was implemented using addition of a random number from a uniform distribution

$$x' = x + z, \quad z \sim \mathcal{U}(-1,1). \tag{3.6}$$

An issue that arises for all transformations is that many features in the dataset are highly correlated, hence making it important to maintain the relationship between variables. Thus, a special subset of features was used in combination with the data augmentation, and the transformations were only applied to the lateral and longitudinal acceleration of a target vehicle. Furthermore, to preserve the interpretation of the data, the rotation and Brownian noise was only applied to the lateral acceleration.

4

Methods

In this chapter, we present the methods that were used to implement and evaluate the models.

4.1 Model implementation

The models were trained on one of two datasets. The first dataset *Evasive* contains the training data of evasive scenarios, the second dataset *Evasive+RW* includes ten randomly chosen logs from the real-world data and the evasive set. The second dataset was included to increase the number of observations and hopefully the variation of the input space. Hence, this dataset was used in modeling the probabilistic Hidden Markov Model.

The feature selection method from Section 3.2.1 was used to generate two sets of features. The resulting feature sets are shown in Table 4.1, where the HMM model was used to generate *Set 1* and *Set 2* with slight variations in the training data. Since *Set 2* had slightly fewer features, it was tested on the other models and was used if there was little improvement to use the increased number of features in *Set 1*. The final set of features is *Set 3* which only uses Target acceleration features and the engineered feature. This feature set was selected to make use of data augmentation in order to increase variety in the input data and was only used with the LSTM model.

Table 4.1: Three sets of features that were used. The first two feature sets were selected using the feature selection algorithm from Section 3.2.1. The last one was restricted to allow for the use of augmentation methods. Feature descriptions can be found in Table 3.2.

	Selected features			
Set name	Host	Target	Meta	Engineered*
Set 1	$x_0, x_1, x_2, x_3, x_4, x_5$	$x_5, x_6, x_7, x_8, x_9, x_{13}, x_{14}$	-	-
Set 2	x_0, x_1, x_3, x_4, x_5	$x_5, x_6, x_7, x_8, x_{11}$	-	-
Set 3	-	x_5, x_6	-	x_{new}

*See Section 3.2.2

A summary of the training data that was used, selected features, data augmentation and hyperparameters can be found in Table 4.2.

Table 4.2: Model details of the selected: training dataset, feature set, data augmentation and hyperparameters.

Model	Dataset	Features	Augmentation	Parameters	
SVM/SVM-BF	Evasion+RW	Set 1	No	Kernel	RBF
				C	150
				γ	1.5
				a	1
				b	10
				c	0.99
				T	10
				τ	0.5
HMM	Evasion+RW	Set 1	No	h_0	0.9
				h_1	0.9
LSTM	Evasion	Set 3	Yes	n_{layer}	2
				n_{hidden}	128
				n_{epoch}	70
				lr	0.001
				d	0.2
ROCKET	Evasion	Set 2	No	α	1.5
				T_w	100
				s	10

As a post-processing step for all of the models, any prediction that was made on speeds lower than 6 m/s was set to the non-evasive state (0). The reason behind this step was that it was noticed that stationary objects had larger uncertainty and fluctuations, which often resulted in false detections. Thus, since evasions can be assumed to both be more dangerous at higher speeds and more likely, it was considered a reasonable limitation of the models to only be able to predict evasive maneuvers at speeds greater than 6 m/s.

4.1.1 Support Vector Machine - Bayesian Filter

The SVM-BF method was implemented using the machine learning framework `scikit-learn` [16], which was extended with the Bayesian filter. Since only the SVM has learnable parameters, training was performed separately without time dependence of observations, and the Bayesian filter step was only added in evaluation. The model has a total of eight tuning parameters, three for the SVM and five for the Bayesian filter. The SVM parameters are: the kernel K, the kernel parameter γ and the inverse regularization strength C; the Bayesian filter parameters are: the beta prior parameters a and b, time window length T, discount factor c and threshold τ.

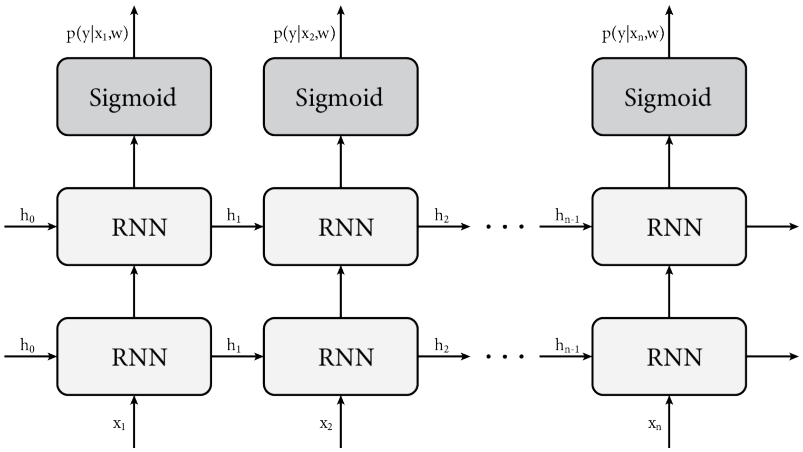

Figure 4.1: Model architecture for the recurrent neural network. The input vector x_t was fed to two layers of (LSTM) RNN cells and the output layer consisted of a single fully connected linear layer, followed by a Sigmoid activation function.

4.1.2 Long Short-Term Memory - Recurrent Neural Network

The machine learning framework PyTorch [17] was used for implementing the LSTM-RNN model. A two-layer LSTM architecture was used with 128 hidden states. The output layer consisted of a fully connected layer to a Sigmoid activation function, which then resulted in a number that is interpreted as the probability that an evasive maneuver is taking place. A schematic view of the neural network architecture can be seen in Figure 4.1. The model was trained using an Adam optimizer [18] for 70 epochs, with a learning rate of 0.001 and Binary Cross-Entropy loss. A dropout of 0.2 was added during training between the two layers.

4.1.3 Hidden Markov Model

The HMM was implemented as explained in Section 2.4, using the training dataset in combination with ten randomly chosen logs from the real-world dataset. The tuning parameters for the HMM contain the bandwidth h for the kernels (Gaussian kernel density estimate). Both kernels, normal and evasive, can be tuned individually. The bandwidth is as discussed in Section 2.4 estimated with Silverman's rule of thumb, it is then tuned by a scale factor of this first estimate.

4.1.4 Random Convolutional Kernel Transform

ROCKET was implemented using the machine learning frameworks `sktime` [19] and `scikit-learn` [16]. Specifically, `sktime` was used to compute the kernels and transform the data while `scikit-learn` was used to implement the Ridge Classifier. Two pre-processing steps were performed, first, the input data was standardized based on the training data, second, the time-series data was split into equal-sized time windows $T_w = 100$. This step was made to ensure an equal size of the input for the ROCKET transform.

4.2 Model evaluation

The models were evaluated in two steps, first using 5-fold cross validation on the training dataset and then, after being trained on the full training dataset, using the two test sets, one with evasive maneuvers and one with real-world data

The choice of classification metrics is a non-trivial task. For a model to be valuable as an evasion detection algorithm in real-time use, it needs to perform with low time delays, few false detections and a sufficiently high detection rate. Specifically, this means that there is a balance between

$$\text{recall} = \frac{tp}{tp + fn} \tag{4.1}$$

and

$$\text{precision} = \frac{tp}{tp + fp} \tag{4.2}$$

of the classifier. Where tp is true positive, the number of predicted evasion maneuvers that was labeled as an evasion maneuver. And in the same manner, fp is false positive and fn is false negative. The intuition behind these two metrics is that; precision indicates the ability not to falsely predict a negative sample as positive, and recall is the ability to find all of the positive samples. A combination of these two metrics is defined as

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}. \tag{4.3}$$

The F_1-score gives equal importance to both precision and recall and is thus a suitable measure of the performance of each classification model. This metric was used primarily in hyperparameter tuning, while the comparison of models was performed using an overview of all metrics combined. Lastly, a measure of the time delay, from the labeled start of the evasive maneuver to the first predicted positive label, was constructed. The algorithm measures the average distance from the predicted start to the true start of an evasive maneuver.

Real-world data was evaluated with the assumption that no evasive maneuvers exist in the dataset, thus any positive label is assumed to be an incorrect prediction. Thus, the evaluation metrics that were chosen to study this dataset was the False Positive Rate

$$FPR = \frac{fp}{fp + tn}, \tag{4.4}$$

where tn is true negative, and the number of consecutive detections. The definition of a consecutive detection is that, a set of consecutive positive predicted labels is seen as one detection, and that a single negative label breaks the detection.

5

Results

In this section, we compare the performance of the five evasive maneuver classifiers for real-time detection. Then we evaluate the method of time-series augmentation to improve training of the Recurrent Neural Network. Lastly, we present the results of adding non-evasive driving examples to the draining dataset.

5.1 Evasion classification

The five models are evaluated using cross validation in Table 5.1, both on a previously unobserved test dataset of evasive maneuvers and a dataset of real-world data in Table 5.2. The different datasets are outlined in Section 3.1.

For the cross validation results on the training data in Table 5.1, both the mean and standard deviation of the five folds are presented, and the best score for each metric is displayed in bold. These results are intended to estimate the generalizability and performance of the models at the same time. Therefore, if the datasets were varied enough and the models are properly tuned, these scores should be achievable in the test dataset evaluation.

Table 5.1: Classification results for 5-fold cross validation on the training set of evasive scenarios. The mean and standard deviation of the evaluation metrics: Precision, Recall, F_1-score and detection delay (Lag) are presented.

	Cross Validation Mean				Cross Validation Std.			
Model	Precision	Recall	F_1	Lag	Precision	Recall	F_1	Lag
SVM	0.713	0.643	0.673	0.7	0.153	0.067	0.104	0.2
SVM-BF	0.724	0.594	0.649	0.7	0.148	0.066	0.095	0.1
HMM	0.590	**0.654**	0.616	0.7	0.075	0.093	0.064	0.2
LSTM	0.802	0.611	**0.686**	**0.6**	0.110	0.070	0.033	0.2
ROCKET	**0.978**	0.418	0.556	4.2	0.031	0.229	0.210	2.4

The final validation of the models was done with the previously unobserved test

dataset and the real-world dataset, which are presented in Table 5.2. It is observed that all models performed significantly worse on the test data. Specifically, the precision was lower and hence also affecting the F_1-score. The results on the real-world data are mixed but mostly showed that the models (except for ROCKET) are not good enough to implement in real-time in their current state. Since the dataset contains 87364 observations with an approximate sample rate of 50 ms, the total observed time is roughly 72 min. In order for any of the models to be usable, the number of false detections needs to be a lot lower, especially since no model is able to score lower than 2.4 false detections per minute. Moreover, the total observation time is not the actual driving time, it is instead tied to each observed target vehicle. Thus, even more evasions are predicted per minute of actual driving time.

Table 5.2: Classification results on the test and real-world data for each model. The test data are presented with metrics: Precision, Recall, F_1-score and detection delay (Lag), and the real-world data with metrics: False Positive Rate (FPR) and the number of detected evasive scenarios (Detections).

	Test Data				Real-World	
Model	Precision	Recall	F_1	Lag	FPR	Detections
SVM	0.357	0.591	0.445	0.7	0.078	586
SVM-BF	0.365	0.527	0.431	0.8	0.067	383
HMM	0.485	0.543	**0.512**	0.7	0.051	176
LSTM	0.353	**0.696**	0.468	**0.5**	0.069	406
ROCKET	**0.700**	0.094	0.166	2.5	**0**	**0**

From the results in Tables 5.1 and 5.2, models can be divided into two clear groups from these results, one containing the first four models *SVM*, *SVM-BF*, *HMM*, *LSTM*, and one with *ROCKET*. The first group of four models all have relatively fast detections and can therefore be used as real-time classifiers, while ROCKET requires a window of "future" data to make a prediction. Thus, the ROCKET model is much more suitable as a ground-truth/search algorithm and was therefore tuned in a way that reduced the number of false detections, which slightly lowered the F_1-score, in order to obtain a very high precision of 0.978 in the cross validation results. Furthermore, the model managed to run on the entire real-world dataset without making a false prediction which indicates that the model achieved the intended results. However, the trade-off is that the recall on the test data went down significantly, and even the precision was lower than expected on this data. In explaining these results, one can consider the possibility that evasive maneuvers have significantly different patterns from case to case, and that even if results on the training data was cross validated, the data came from just two data collection occasions. Although, on inspection of all of the predictions that were made on the test data in Appendix A, it is observed that the metrics do not show the entire picture of the results since the ROCKET model correctly classifies 5 out of 36 scenarios,

with just one false detection slightly prior to one of the evasions.

The classification behavior of the remaining four models is also analyzed using the plot of two samples in the test dataset which can be seen in Figure 5.1. In the two plots, the models' predictions are shown as bars and the shaded areas indicate the time intervals where an evasion occurred. The effect of the Bayesian filter is more apparent by visualizing the results. Both the regular SVM and the SVM-BF are aligned in most predictions. However, very short detections, such as the one slightly before the 50-second mark in the top plot, are removed by the filter. Additionally, even though it is hard to distinguish from the plots, the Bayesian filter delays classification very slightly in comparison to the regular SVM. Thus, the filter essentially slows down state transitions and uses that technique to reduce false detections at the cost of prediction delays.

From the two samples that are shown in the figure, the HMM is the most accurate model in terms of making a prediction close to the real evasive event and few temporally further away from it. This observation is also backed up by the metrics.

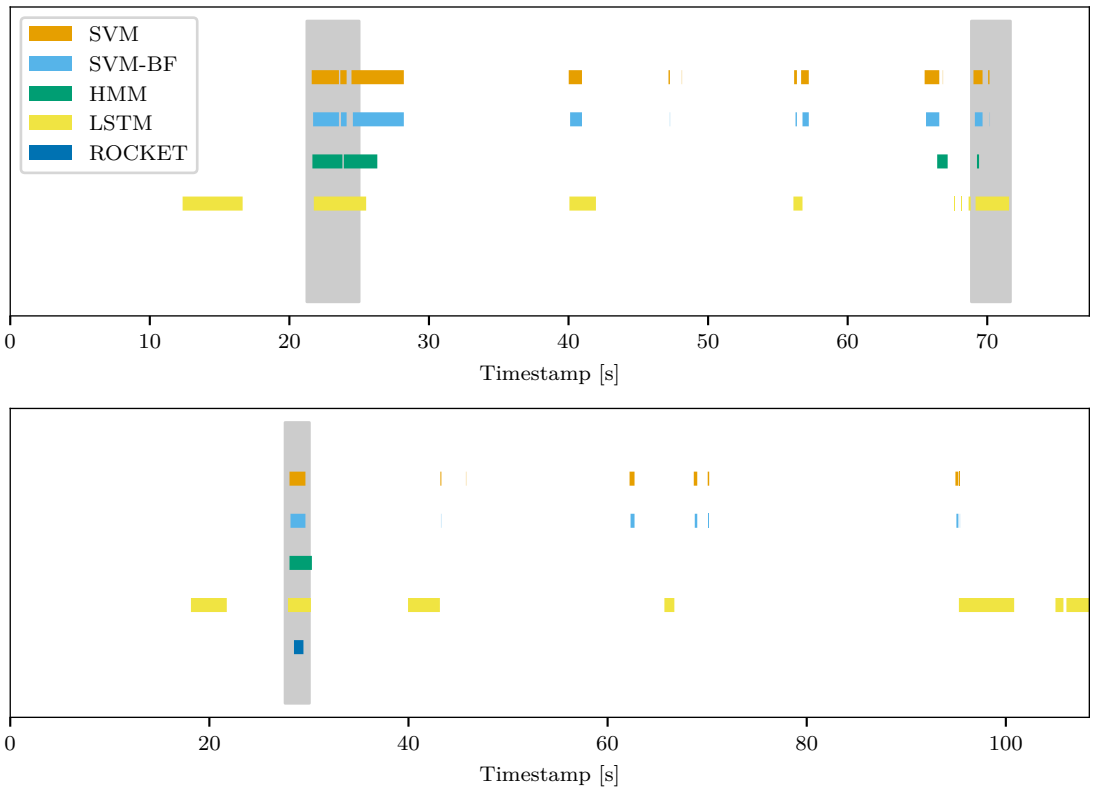

Figure 5.1: Two samples from the validation dataset. Evasive maneuvers are performed in the time intervals that are indicated with the shaded area. Each of the five models' predictions are presented as bars in the plots.

5.2 Improving LSTM training with augmentation

It was hypothesized that the recurrent neural network would have issues with overfitting for a small dataset. This was the explanation for including data augmentation in the training process of this model. The results of including duplicate data with an augmentation method are presented in Table 5.3. Three of the augmentation methods showed interesting results; Scale and Rotation resulted in the highest precision and F_1-score respectively, and the Brownian noise addition had a comparable F_1-score to the unaugmented dataset (None), but with a substantially higher standard deviation. The higher standard deviation was caused by one of the folds having lower metrics than the rest, resulting in both a lower mean and higher standard deviation.

Table 5.3: Inclusion of data augmentation in the training of the LSTM Recurrent Neural Network. The mean and standard deviation of the evaluation metrics are presented for an LSTM model that has included each of the augmentation methods.

	Cross Validation Mean				Cross Validation Std.			
Method	Precision	Recall	F_1	Lag	Precision	Recall	F_1	Lag
None	0.681	0.501	0.565	0.9	0.137	0.133	0.117	0.2
Rotation	**0.784**	0.508	0.590	**0.6**	0.136	0.160	0.092	0.1
Brownian	0.659	0.449	0.523	0.8	0.197	0.214	0.219	0.2
Jitter	0.648	0.558	0.594	0.8	0.136	0.069	0.079	0.1
Scale	0.726	**0.567**	**0.626**	0.7	0.074	0.154	0.106	0.3
Offset	0.626	0.534	0.551	0.7	0.130	0.199	0.143	0.1

The three augmentation methods, Scale, Rotation and Brownian, were then added to the training of the model, and the training curves were then analyzed in Figure 5.2. In this experiment, the model was trained one of the folds in the training data in 100 separate training sessions. The line is the mean of the learning curves from these sessions and the shaded area around the curve indicates the standard deviation. The "Unprocessed" curve in the figure shows the issues with instability of the learning curves when attempting to train the LSTM model. Training without augmentation is not stable enough to guarantee generalization since it struggles to learn the same patterns in the data on each training session. The blue "Augmented" curve shows the same experiment but where the model was trained on augmented data and clearly shows an improvement, both in average results and in reduced standard deviation of the final training epoch. Furthermore, it should be noted that augmented data contains more samples per epoch since the time series was duplicated for each augmentation. However, similar results were observed by increasing epochs by a factor of four (to the same number of training samples as the augmented data).

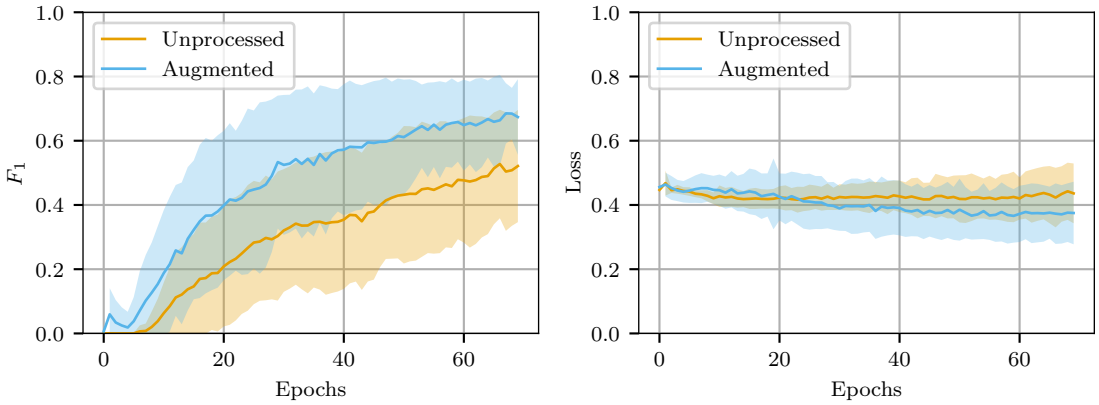

Figure 5.2: A comparison between the RNN learning curves for F_1-score (left) and binary cross-entropy loss (right). One fold of the cross validation were trained 100 times, and the curves show the average validation score for each metric. The shaded area around the curves indicates the estimated standard deviation of the results.

6

Discussion

In this chapter, we discuss the results of this report and highlight a few important topics that are essential in understanding the studied problem.

6.1 Data limitations

At the foundation of this thesis is the problem of using machine learning methods with a limited and small dataset. Data limitation can be considered a big problem in "edge-case" scenarios, where events are rare but have a large impact on the performance of a system. In our case, the system is an autonomous vehicle and the performance primarily encompasses the idea of passenger safety.

Our approach in working with this problem has been to make the most out of the data that is available, primarily through data augmentation but also some feature engineering. Although some success was seen from utilizing these methods, it is evident that it does not solve the data limitation issue. In order to implement a machine learning model that generalizes to unseen data and performs well, a lot more training examples are needed. Thus, more labeled datasets are needed for edge-case scenarios such as evasive maneuvers. This conclusion led to the implementation of the ROCKET model in this thesis since it has the potential to be used as in search algorithm for similar events without resulting in an absurd amount of false positives that need to be manually sorted away by a human. Furthermore, alternate approaches should be considered as the first choice for problems with data limitations. Non-machine learning methods could in many cases be sufficient, especially where domain knowledge may generate more information to the model than a machine learning method. Such methods are preferable since the limited data that exists can be used to validate the model, and that theoretical arguments and proofs could corroborate these results. Alternatively, as will be discussed in Section 6.2, unsupervised or semi-supervised methods could be possible options. However, in the case of modeling a latent and unobservable variable, such as the intent to perform an evasive maneuver, a machine learning approach makes sense which is why this approach was chosen as the scope of this thesis.

6.2 Supervised vs. unsupervised approach

As a continuation of the discussion above in Section 6.1, there are some issues that have been discovered in using supervised learning techniques in this thesis. Due to the unobservable nature of the target variable y, the labeling process was in many ways arbitrary and subjective to the authors' opinions of what an evasive maneuver is. Furthermore, the human that was driving the target vehicle did not necessarily decide to start the maneuver at a specific time, instead, it is quite possible that a combination of risk and perceived danger changed over the course of the maneuver. The complexity of the event is also a reason to why a single binary variable may not be sufficient to capture this complexity. Specifically, we noted that predictions that started slightly before the labeled evasion and continued to slightly after it, could yield metric scores that are worse than a detection that just captured a small part of the evasion and also made false predictions that were not at all connected to the evasion. A toy example of this problem is shown in Figure 6.1, where both predictions have equal F_1-scores but the results are very different in terms of usability of the model. Thus, it raises the question of whether a supervised learning approach is ideal for the problem.

An alternate formulation to the problem could allow for unsupervised, or semi-supervised learning models to be used instead. Areas of unsupervised learning techniques that could apply for this thesis are, for example, anomaly detection [20] or an unsupervised variation on the HMM. Such methods have the benefit of not suffering from the problems above, but also that they could more easily extend to other edge-cases of events that rarely happen. The most promising evidence that a supervised learning approach could be the solution to this problem comes from the high precision of the ROCKET model in our results. Thus, there is a pattern to be found in evasive maneuvers that do match the labels of our datasets. Under the assumption that a supervised learning method is sufficient, it could be interesting to study the results using a different metric/loss function. A possibility would be to use "soft" labels instead of "hard" ones, where the labels would be interpreted as a probability distribution instead of exact binary labels. Then, predictions that are made before and after an evasive maneuver is less penalized. Through this, the problem can be studied using the Kullback-Leibler divergence of the predicted probability and the labeled probability density function. A limitation with this approach is however that only methods that output interpretable probabilities are suitable for use.

6.3 Validation metrics

The classification metrics where presented in Section 4.2 and used to present the results in Table 5.2, namely Precision, Recall, F_1, FPR and Lag. These validation metrics can be argued to be a good measurement when evaluation many different machine-learning models, however as mentioned earlier there is a clear problem when visually inspecting the plotted results, as we can see in the toy example Figure 6.1. This study has still used these metrics as a presentation tool for the results since it can give an indication of the performance of the different models. We can see

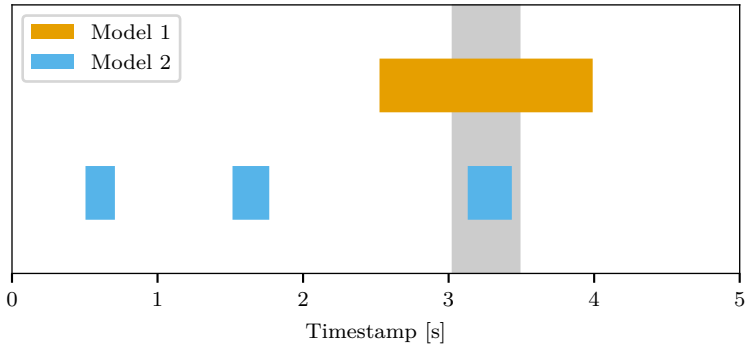

Figure 6.1: A toy example of the issues with using hard labels and classification metrics. The shaded area is the true label of the evasive maneuver, while the two bars represent predictions from two arbitrary models. Both Model 1 and Model 2 have a F_1-score of 0.5, however Model 2 makes a significantly worse prediction than Model 1.

in Table 5.2 that SVM, SVM-BF, HMM and LSTM all have similar results in all these metrics but when inspecting Figure 6.1 there are some notable differences in where the different models make their predictions. The remaining model, ROCKET however differs, by having high precision and low recall and a considerably longer Lag value.

Furthermore, looking into the results presented from the real-world dataset. Where the metric FPR is used, since in this large dataset there are no evasive maneuvers present, and therefore desirable to get as low value as possible here. As seen in Table 5.2, ROCKET has no false positive predictions on this dataset, but in turn, has a large Lag value. As mentioned in Section 5.1, ROCKET then loses its value as a real-time classifier but can be used as a ground-truth/search algorithm. This could show to be very useful as future work, where many limitations in this study are related to data limitation, for evasive maneuvers. As there exists a large number of real-world data, the use of a search algorithm to increase the dataset for evasive maneuvers could show to be a good asset in future work.

6.4 Model complexity

As mentioned earlier the limited data has been a factor, which can also be seen when looking closer at the results. This study has presented four different models to approach this, SVM-BF, HMM, LSTM and ROCKET. SVM uses a hyperplane to separate the data into classes, since the SVM assumes time independence the BF has been implemented to make the model time-dependent. This model is simplistic and as mentioned before has been used as a base model to compare against. And it shows that a simplistic model performs well compared to more complex models when limited data is used, looking at Table 5.2 we can see that the SVM-BF has a mean F_1 score of 0.431 for the test validation with a detection lag of 0.8. Both HMM and LSTM performs better on these points but still within a small margin.

As we can see on the detections presented in Figure 5.1 the SVM-BF model detects the labeled evasion maneuvers in addition to a few false detections, similar to what we can see with LTSM.

Moreover, the probabilistic model approach HMM that uses a Gaussian kernel distribution for the observation symbol probability does not necessarily demand a large dataset, to some extend, which made this model a theoretical good candidate. We can see in Table 5.2 that the HMM has a mean F_1 score of 0.512 for the test validation with a detection lag of 0.7. Which is an improvement from the results seen from SVM-BF, both in F_1 score and lag. It also has fewer false detections on the real-world dataset compared to both SVM-BF and LSTM with a false positive rate of 0.051. As seen in Figure 5.1 there is a clear difference when observing HMM and comparing with the predictions made with SVM-BF and LSTM, where there are noticeable more predictions far away from the labeled evasion maneuver.

7

Conclusion

The purpose of this thesis, as stated in Section 1.1, was primarily to investigate if it is possible to classify an evasive maneuver. As seen from the test results presented in Table 5.2, where it shows that the models are capable of predicting evasive maneuvers on test-track, however, it is clear by these results that further work to advance these models should be made. The aim to use Lag as a metric to investigate how fast the models could classify an evasive maneuver has shown satisfying results where the conclusion that ROCKET can be used as a ground-truth/search algorithm on existing datasets to help gather larger datasets for training purposes of the promising models HMM and LSTM. The purpose also stated that the thesis aim to thoroughly explore the feature content and use feature selection methods combined with data augmentation. Feature selection using an algorithmic approach was found to produce varied results from a small change in the model or dataset. Thus, it was concluded to not be an ideal approach, using domain knowledge and avoiding the inclusion of highly correlated features seemed to produce similar results. Meanwhile, data augmentation for time series data showed a lot of promise in increasing the variety of input data. The technique allowed for the use of a much smaller and more interpretable set of features while maintaining comparable results to a larger set of features. Lastly, this thesis aimed to incorporate test-track data and real-world data, and the metric used to evaluate how the different models performed on real-world data was False Positive Rate (FPR), where the results are presented in Table 5.2. The real-world dataset included a large number of observations (87364) with no evasive maneuvers. This metric was created for the purpose of investigating the usability for ADAS, where a high FPR would result in a high number of triggered responses from ADAS where it would not be necessary. The investigation showed that a well performing model to be implemented in ADAS should show a high precision, low Lag in combination with a low FPR on the real-world dataset.

In order to improve the results of this thesis, we suggest that future work approach the problem from a wider viewpoint, considering unsupervised methods and including more descriptive features that can capture the complexity of human behavior in driving situations. An example of such features are vehicle-to-vehicle/traffic interactions to put observed motion behavior into more context, an approach that has been explored by [21]. Additionally, they applied Dynamic Bayesian Networks for maneuver prediction, which would be an interesting approach to take for this problem as well, since the HMM led to interesting results. Traffic interactions could

help to identify scenarios where the target vehicle is fast-moving and a slow vehicle is seen to be in its path. In such a situation, it would make sense that the target vehicle performs a fast take-over maneuver, and thus reducing the probability that it is an evasive maneuver. It should however be noted that this approach requires more data of evasive maneuvers in a real traffic setting, whereas the data in this thesis only came from a test track without additional vehicles on it.

An additional point of future work is to study risk in combination with detecting evasive maneuvers. In particular, it is possible that the high rate of false positives is acceptable if the detection algorithm only flags events that happen during high-risk scenarios. To draw a parallel to human behavior, we can often detect situations where we believe another driver makes a strange maneuver, but choose to ignore them due to it happening "far away" or not being an immediate risk to us. Therefore, risk assessment is a logical next step in this work and could increase the usability of the evasion classifiers that were proposed in this thesis.

Lastly, additional work could go into developing augmentation methods that are constrained to physical feasibility. A problem that was found with existing data augmentation methods for time series data is that they often destroy the interpretability of the features. Hence, it is possible that the patterns that are hoped to be found in the data are also removed by introducing augmentation methods. By making smart augmentation algorithms that are constrained to physics, this issue could be avoided while getting the benefits of including augmentation.

Bibliography

[1] E. Yurtsever et al. “A Survey of Autonomous Driving: Common Practices and Emerging Technologies”. In: *IEEE Access* 8 (2020), pp. 58443–58469. DOI: 10.1109/ACCESS.2020.2983149.

[2] X. Geng et al. “A Scenario-Adaptive Driving Behavior Prediction Approach to Urban Autonomous Driving”. In: *Applied Sciences* 7.4 (Apr. 2017), p. 426. DOI: 10.3390/app7040426.

[3] V. Gadepally, A. Krishnamurthy, and Ü. Özgüner. “A Framework for Estimating Long Term Driver Behavior”. In: *Journal of Advanced Transportation* (2017), pp. 1–11. DOI: 10.1155/2017/3080859.

[4] P. Liu et al. “Classification of Highway Lane Change Behavior to Detect Dangerous Cut-in Maneuvers”. In: (2016).

[5] P. Kumar et al. “Learning-based approach for online lane change intention prediction”. In: (2013), pp. 797–802. DOI: 10.1109/IVS.2013.6629564.

[6] M. Bahram et al. “A Combined Model- and Learning-Based Framework for Interaction-Aware Maneuver Prediction”. In: *IEEE Transactions on Intelligent Transportation Systems* 17.6 (2016), pp. 1538–1550. DOI: 10.1109/TITS.2015.2506642.

[7] G. S. Aoude et al. “Driver Behavior Classification at Intersections and Validation on Large Naturalistic Data Set”. In: *IEEE Transactions on Intelligent Transportation Systems* 13.2 (2012), pp. 724–736. DOI: 10.1109/TITS.2011.2179537.

[8] J. L. Elman. “Finding Structure in Time”. In: *Cognitive Science* 14.2 (1990), pp. 179–211. DOI: https://doi.org/10.1207/s15516709cog1402_1.

[9] A. Sherstinsky. “Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network”. In: *CoRR* abs/1808.03314 (2018). arXiv: 1808.03314. URL: http://arxiv.org/abs/1808.03314.

[10] S. Hochreiter and J. Schmidhuber. “Long Short-term Memory”. In: *Neural computation* 9 (Dec. 1997), pp. 1735–80. DOI: 10.1162/neco.1997.9.8.1735.

[11] D. Jurafsky. “Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition”. In: (2009), pp. 239–242.

[12] B. A. Turlach. “Bandwidth Selection in Kernel Density Estimation: A Review”. In: (2013).

[13] B. W. Silverman. “Density estimation for statistics and data analysis”. In: (1986).

[14] A. Dempster, F. Petitjean, and G. I Webb. “ROCKET: Exceptionally fast and accurate time classification using random convolutional kernels”. In: *Data Mining and Knowledge Discovery* (2020). DOI: https://doi.org/10.1007/s10618-020-00701-z.

[15] Q. Wen et al. “Time Series Data Augmentation for Deep Learning: A Survey”. In: *arXiv e-prints*, arXiv:2002.12478 (Feb. 2020). arXiv: 2002.12478 [cs.LG].

[16] “Scikit-learn website for regression models”. In: (Dec. 2020). URL: https://scikit-learn.org/stable/supervised_learning.html#supervised-learning.

[17] A. Paszke et al. “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: (2019). Ed. by H. Wallach et al., pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[18] D. P. Kingma and J. Ba. “Adam: A Method for Stochastic Optimization”. In: (2017). arXiv: 1412.6980 [cs.LG].

[19] M. Löning et al. “sktime: A Unified Interface for Machine Learning with Time Series”. In: (2019).

[20] V. Chandola, A. Banerjee, and V. Kumar. “Anomaly Detection: A Survey”. In: *ACM Comput. Surv.* 41.3 (July 2009). ISSN: 0360-0300. DOI: 10.1145/1541880.1541882.

[21] J. Li et al. “A Dynamic Bayesian Network for Vehicle Maneuver Prediction in Highway Driving Scenarios: Framework and Verification”. In: *Electronics* 8.1 (2019). ISSN: 2079-9292. DOI: 10.3390/electronics8010040. URL: https://www.mdpi.com/2079-9292/8/1/40.

A

Validation results

In this chapter, all of the plots for the validation results are presented.

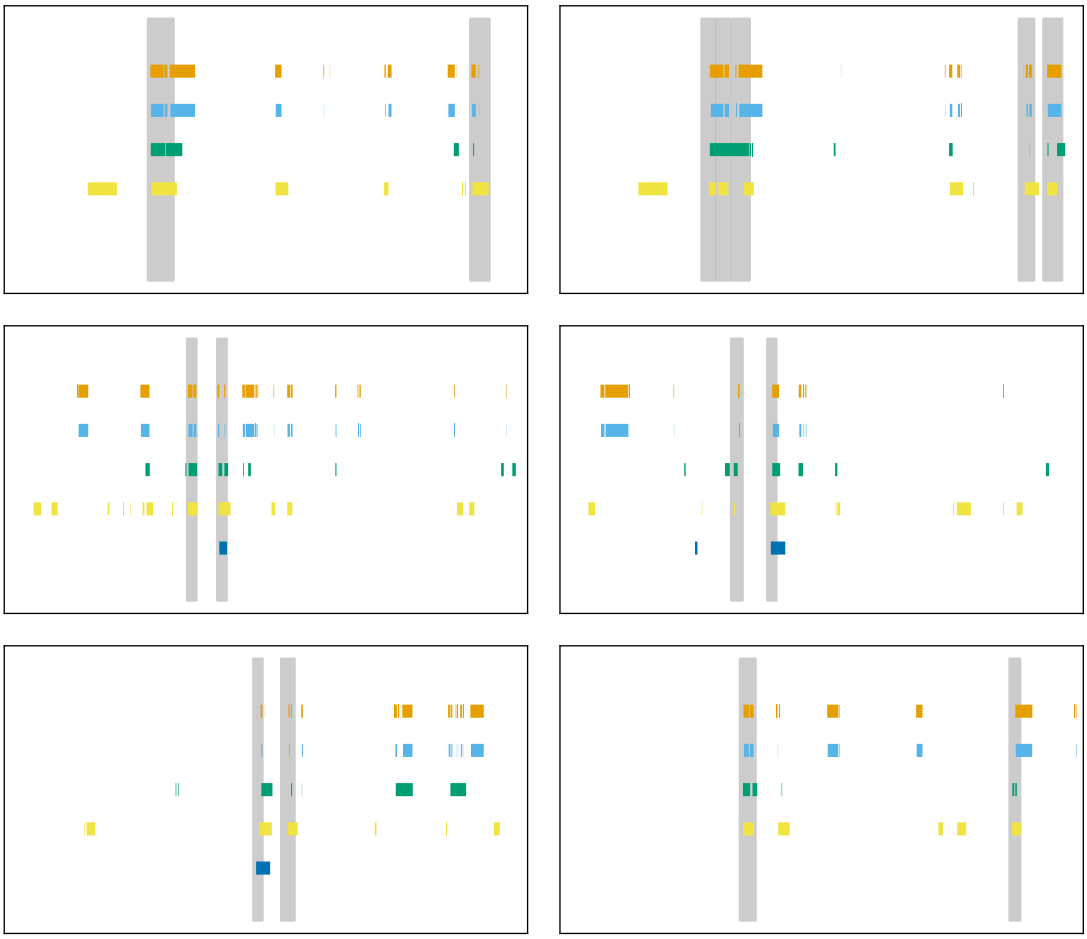

Figure A.1: The first six samples from the validation dataset. Evasive maneuvers are performed in the time intervals that are indicated with the shaded area. Each of the five models' predictions are presented as bars in the same order as in Figure 5.1. The order of the models are (from top to bottom) SVM (*orange*), SVM-BF (*light blue*), HMM (*green*), LSTM (*yellow*), ROCKET (*dark blue*).

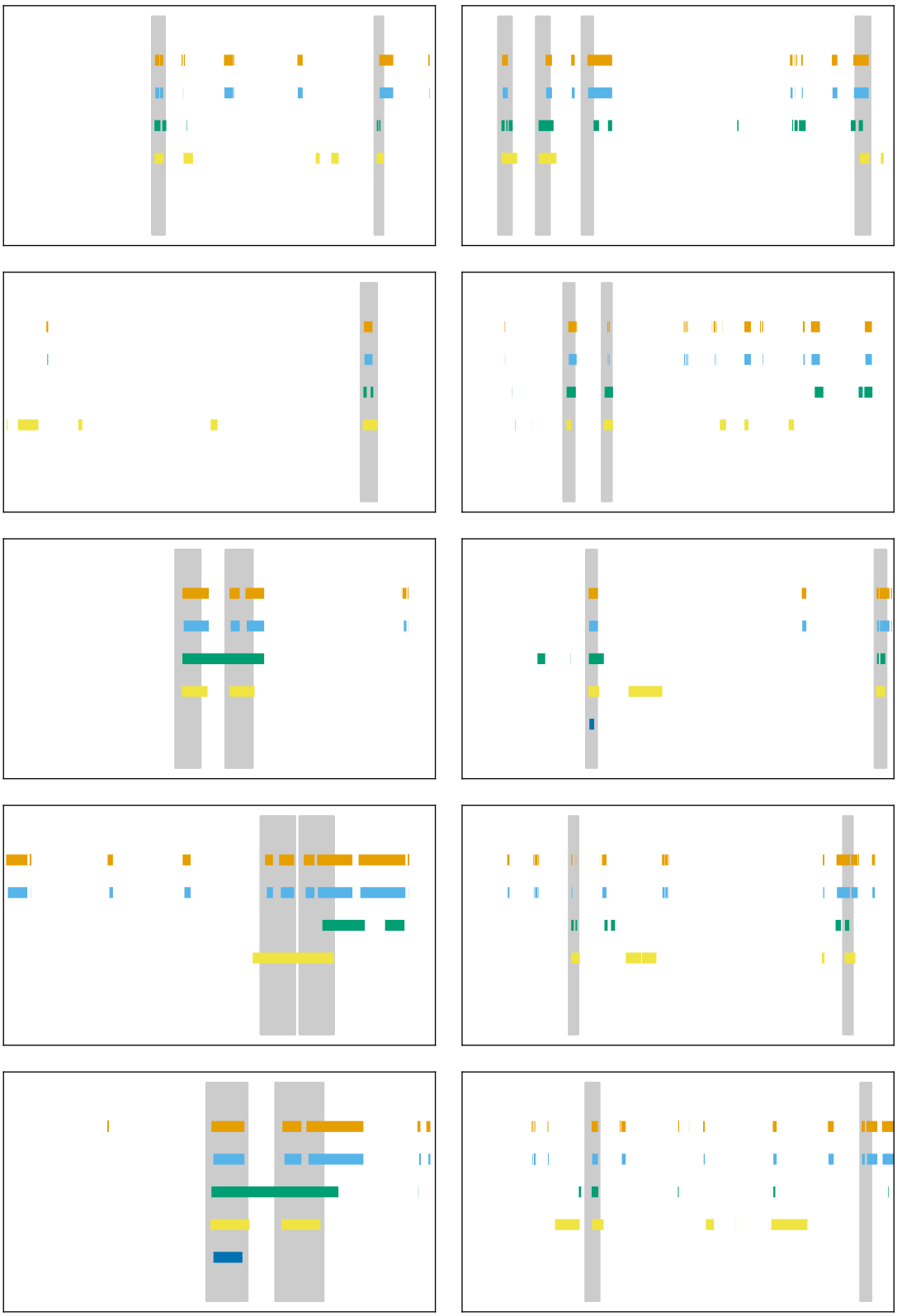

Figure A.2: The remaining samples from the validation dataset (see Figure A.1).

DEPARTMENT OF MATHEMATICAL SCIENCES
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY