Robustness Analysis on Machine Learning Based Prediction Models Used for Threat Assessment

Gustav Hultberg Hermansson, Simon Wu

June 2021

CHALMERS

Department of Electrical Engineering

CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden, 2021

Abstract

Traffic fatalities remains a great problem for society today, being the leading cause of deaths for children and young adults. Following the fast development in technology, *Advanced Driving Assistance Systems* (ADAS) in vehicles take an ever increasing role in the struggle to reduce traffic related deaths and injuries. A key component of ADAS is to accurately and robustly predict potential future threats within its limits and capabilities. In this thesis, an *Operational Design Domain* (ODD) was determined to first define the capabilities and limitations of the model to be developed. Then, a *Machine Learning* (ML) based *Threat Assessment* (TA) model was developed for detecting unintended lane departures during driving. The model was constructed using an *Artificial Neural Network* (ANN) trained on real-world time series driving data, and a proposed *Decision Making* (DM) logic. The threat assessment performance was evaluated and benchmarked against a kinematic model. The model's performance in the ODD was evaluated by investigating the underlying causes for the model's failures. A simulation tool was developed and used together with on-vehicle cameras for this purpose. The benchmarking results show that the ML based TA model has increased performance compared to the kinematic model. Moreover, the results of the ODD evaluation indicate that the main underlying cause for the model's failures lies in its inability to correctly interpret driving behaviours.

Acknowledgments

First and foremost, we would like to thank our supervisors John Dahl and Dr. Gabriel Rodrigues de Campos for the great support we have received through the whole thesis. Their guidance have been incredibly helpful for us to achieve the final results presented in this thesis, as well as our own development both technically and personally. We would also like to thank our academic supervisor Professor Jonas Fredriksson for taking on the responsibility and making this thesis possible. Finally, we would like to thank all the employees at Zenseact for having us, and for creating such a welcoming and helpful atmosphere. We are very grateful for having the opportunity to perform our master thesis at Zenseact.

Contents

List of Figures

List of Tables

Nomenclature

ADAS	Advanced Driver Assistance Systems
AEB	Automatic Emergency Break Assist
AI	Artificial Intelligence
ANN	Artificial Neural Network
BLIS	Blind Spot Information System
CNN	Convolutional Neural Network
CVM	Constant Velocity Model
DM	Decision Making
FN	False Negative
FP	False Postive
FPR	False Postive Rate
LDW	Lane Departure Warning
LKA	Lane Keeping Aid
LSTM	Long Short Term Memory
ML	Machine Learning
MLP	Multilayer Perceptron
MSE	Mean Square Error
ODD	Operational Design Domain
RNN	Recurrent Neural Network
SBTM	Single Behaviour Threat Metric
TA	Threat Assessment / Threat Assessor
TLC	Time to Lane Crossing
TN	True Negative
TP	True Positive
TPR	True Positive Rate

1 Introduction

1.1 Advanced Driver Assistance Systems

Every year, around 1.35 million people die from accidents related to road fatalities [1], being the leading cause of death among children and young adults. According to [2], 99% of the high way crashes recorded between 1996-1997 in the United States were directly caused by driver behavioral errors, among which driver inattention was a major contributing factor. It is therefore reasonable to believe that the number of road fatalities would be reduced considerably if drivers receives support from in-car *Advanced Driver Assistance Systems* (ADAS).

ADAS are in-vehicle electronic systems designed to enhance vehicle safety in an intelligent and autonomous way. By the usage of camera, radar and ultrasonic sensors, a tool-chain of computational software as well as a safe human-vehicle interface, ADAS can assess driving errors and potential threats from the surroundings and react accordingly in critical situations. Depending on the application area, ADAS can either interact with the driver through warnings or directly intervene with an automated maneuver. A simple example of widely implemented in-car ADAS today are *Emergency Auto Brake Assist* (EAB) system, used for avoiding collisions with objects in front of the ego-vehicle. A meta-analysis based on traffic accident statistics from six countries states that vehicles equipped with City-EAB were able to effectively prevent 38% of front-to-rear collisions [3]. Other functionalities of ADAS can for example be: warn and assist drivers to keep the vehicle in lane, help to maintain a safe distance to the surroundings as well as blind spot detection. Thus, ADAS is expected to have an increasing role during the safety rating procedure of New Car Assessment Programs among multiple nations [4], in the desire to build safer vehicles and to reduce worldwide traffic related fatalities.

Following the fast development of technology, new sensor systems and realtime computational software have accelerated the evolution of ADAS. ADAS is comprised of four building blocks, being *Sensing*, *Threat Assessment* (TA) & *Decision Making* (DM) and *Actuation* [5]. A framework developed for guidance in the development of ADAS [6] provides the following steps: *1. Identification of the concept ADAS*, *2. Identify attributes that define the Operational Design Domain* (ODD), *3. Identify object and event detection and response capabilities*, *4. Identify and assess failure modes and failure mitigation strategies*. As can be concluded from the steps above, any potential limitations in the system should be carefully identified before a new developed ADAS can be put in use.

1.2 Threat Assessment

The concrete definition of "threats" varies depending on the area of application for ADAS. For a *Lane Keeping Aid* (LKA) or *Lane Departure Warning* (LDW) software, a present or potential future lane departure is treated as the threat; while for a *Blind Spot Information System* (BLIS) or EAB, surrounding objects of the ego vehicle are considered as the threats. Regardless of what threat the system handles, the technical core of TA lands on how to accurately and robustly predict driver-mistakes and identify present or future threats, by the usage of mathematical prediction models.

The implementation of real-time TA approaches faces major challenges arising from the high dimensionality of available signals received from in-vehicle sensors. How to properly extract essential information and establish an applicable model, that can predict the target threat using the selected inputs is therefore a cornerstone of any proposed approach. Different approaches tackle this problem from different angles, and TA approaches can on a higher level be divided into methods based on physics and methods based on *Machine Learning* (ML).

1.2.1 Physics Based Models

Physical models are established based upon physical rules and insights. A simple physical model is the *Constant Velocity Model* (CVM). The CVM can further be categorized in a sub-category under physical models called *Single-Behavior Threat Metrics Models* (SBTM), which are models that are subject to simplifications in the problem formulation. The TA problem formulation can then be reformulated as being driven by threat metrics based on the single future behaviours of all participants. These simplifications arise from the difficulty, or impossibility in knowing exactly what the intention of the driver is, and the fact that measurements are nearly never perfect [5]. The CVM can be used to predict lane departure threats assuming the longitudinal velocity and yaw of the vehicle stay constant during the whole prediction horizon. Such models are useful when the road is relatively straight with low variability in the velocity of the vehicle. Their weakness however lies in the ability to predict the distance to the lane marker on roads with high variability in curvature.

1.2.2 Machine Learning Based Models

With the recent bloom of *Artificial Intelligence* (AI) leading to advances in stronger in-car hardware, ML-based TA models have increased in popularity among many areas of ADAS. ML-based TA models purely rely on learning from data, thereby establishing a non-parametric model assuming that connections exist between the selected inputs and target outputs. ML models have the potential of better capturing hidden connections between input signals and defined target outputs, which could otherwise be difficult to formally describe. A common context would be to train an ML-based model to take in historical vehicle information for prediction of a future vehicle state. Popular ML-based models for the above mentioned problem formulation are for example *Artificial Neural Network* (ANN)-based models [7], *Recurrent Neural Network* (RNN)-based models [8] and multiple step prediction models based on linear-regression [9] and [10].

When trained on sufficiently large and carefully pre-processed data, ML-based models are expected to rely on the learning from the training data and adapt to new data when being put in use in reality [5]. The basic assumption is that the new data comes from the same distribution as the training data. It is obvious here that the design of a proper training data set for a specific task is crucial for a ML-based TA approach to function accurately and robustly. Using a LKA as an example, the balance between non-lane departure driving data and lane departure data should be carefully considered, since the selection ratio would highly affect the outcome performance metrics, for example recall and F1-score. One way to easily overcome this problem would be to use the *true positive rate* (TPR) and *false positive rate* (FPR), since TPR only depends on departure data and FPR vice versa.

1.3 Lane Keeping Aid

According to [11], 53% of all traffic accidents in the United States between 2016-2018 were directly caused by lane departures. Thus, LKA software is an indispensable area of application of ADAS, designed to detect and prevent unintended lane departure. Using the ADAS approach developed by Volvo [12] as an example, the logic of an LKA actuation can be briefly summarized as follows:

- The system continuously estimates the present and future driving path using sensor information from camera and radar.
- Assesses threats using prediction model, and makes decisions on whether to interfere based on the critically of the situation as well as the system limitations.
- Intervenes with a suitable steering actuation to prevent the ego-vehicle from unintentionally leaving its original lane. If the driver decides to counteract the actuation, the steering actuation will be scaled down based on the drivers activity.

Following the steps in the above described LKA logic, it is easy to identify the three building blocks of ADAS mentioned in Section *1.1*, being *Sensing*, *TA* and *Actuation*.

1.4 Operational Design Domain

The goal of the ODD for an ADAS is to define the scope of environments and situations for which the ADAS is considered valid and safe to operate in.

According to the Society of Automotive Engineers (SAE), there are six levels of vehicle automation. They range from no automation (level 0), where the driver is in full control of the vehicle to full automation (level 5), where no human driver is needed, and the vehicle is able to carry out all driving tasks. For each of these levels, there is an ODD in which the automated system or feature thereof is designed to work. SAE J3016 defines an ODD as "*operating conditions under which a given driving automation system or feature thereof is specifically designed to function, including, but not limited to, environmental, geographical, and time-of-day restrictions, and/or the requisite presence or absence of certain traffic or roadway characteristics*"[13]. The ODD is limited for every level of vehicle automation except 5, which should have the same mobility as a human driver. Thus, an ODD puts limitations on the vehicle state and the driving environment, where the driving environment is the most comprehensive part. Vehicle states include among others vehicle speed and steering angle. Driving environment includes but is not limited to types of road on which the vehicle is driving, such as the quality of the lane markers, road curvature, lane width, weather and specific road elements, for example tunnels, forks and intersections.

2 Project Description

The goal of this thesis was to develop a TA model to detect unintended lane departures, based on an ANN trained on real world data and a proposed DM logic. The model performance was evaluated and benchmarked against a classical CVM-based TA model. In addition, the model's performance was evaluated by assessing the model's ability in handling various driving scenarios

in the ODD using a real world data set in simulations. The proposed ANN-based TA model is for the rest of this thesis referred to as the "ANN TA".

2.1 Scope

The ANN was trained on sampled time series data obtained from in-car sensors, to predict the lateral distance of the ego-vehicle to the left and right lane marker, respectively, H seconds ahead. The ANN outcome was then used as a basis for the DM logic to assess unintended lane departures. Different sparse sampling patterns, also called *offset patterns* Γ, and *prediction horizons* H were examined to investigate the potential for reducing the model's complexity while maintaining its predictive performance.

The performance of the ANN TA model was validated by benchmarking against the CVM-based TA model. The benchmarking was conducted by using real world data to simulate the behaviour of the different systems during drivings, and then compare their corresponding TA outcome with respect to TPR, FPR and accuracy.

The model robustness in the defined ODD were investigated by 1), identifying driving segments for which the ANN TA failed to handle, and 2), analyzing the cause behind ANN TA performance error by visually analyzing the corresponding video stream, collected from the in-car top-camera. The failure scenarios were manually classified and divided into several categories, used to determine the likely reason for system failure.

2.2 Limitations

Due to the project being carried out in simulations using a collected real-world data set, the project scope didn't cover the implementation of sensing and actuation, mentioned as two of the building blocks of ADAS in Section *1.1*.

2.3 Related Work

In this thesis, the literature study was directed towards ML-based TA techniques used for lane departure detection. Five approaches related to the project scope were investigated, where the focus was placed on the approaches' data set construction, methodological details, model evaluation metrics as well as potential limitations, if could be identified, within the approaches.

In [14], an online personalized driver model which predicts forthcoming vehicle trajectories was proposed, where the overall aim was to produce a LKA system with low false warning rate. The authors combined a Gaussian mixture model and hidden Markov model to estimate the vehicle's lateral displacement. The model was trained and tested on realistic driving data collected from the Safety Pilot Model Deployment program [15], and obtained a false-positive rate of 3% in the LKA application. Low false warning rate is obviously a crucial metric when measuring the performance of any ADAS systems. However, from a practical point of view, other metrics are equally important to obtain a complete picture of the systems accuracy and robustness.

A *Support Vector Machine* (SVM) model using a *Radial* basis function kernel was proposed in [16] for predicting lane departures. The model was trained and tested on time series data generated

from VIRTTEX by testers who were sleep-deprived to represent drowsy drivers. The final approach obtained over 99% sensitivity and specificity over a 0.2-second prediction horizon, with under 1% *False Positive* (FP) and *False Negative* (FN) classification outcome. However, it can be argued that a prediction horizon of 0.2 seconds is far too short for being implemented to vehicles in reality, and that the performance evaluation therefore is too optimistic.

As have been briefly discussed above, the performance of a proposed TA system should be evaluated comprehensively in order to identify the system's limitations. Also, the prediction aspect of a TA system should be balanced between good accuracy and applicability.

In [17] the authors explore the effectiveness of a sliding window binary classification *Multi-Layer-Perceptron* (MLP) with logistic activation functions, to predict unintentional lane departures. The authors tested three different gradient descent methods, namely scaled conjugate descent, conjugate descent and gradient descent. In this work, the scaled conjugate descent gave best results. The training data was generated in a simulator to represent drowsy driving on straight and dry roads during night time. Different vehicle variables (input signals) were tested to predict lane departures 0.2 and 0.5 s ahead of the actual lane departure. Having 80 data points (amounting to 4 s) prior to the lane departure event, a sliding window of 8 data points gave the best prediction result over the 0.5 s prediction horizon. Eight different sets of variable combinations were tested in the study, where the combination of lateral lane position and lateral velocity yielded the highest *True Positive* (TP) and lowest *False Positive* FP. The overall recall and precision for the 0.2 s prediction horizon were 99.74% and 99.66% respectively. However, the result dropped for the 0.5 s prediction horizon, where the recall and precision decreased to 99.23% and 85.49%. It was evident here that increasing the prediction horizon negatively affected the quality of the predictions.

The paper [9] used a *Multiple Linear Regression* (MLR) model to implement a computationally efficient LKA system. A direct *h-step prediction model* was proposed, taking in multiple time series of driving input signals, and outputs a prediction of the future distance to the lane markers with high performance. This was then compared with the detected lane markers from the *Front Looking Camera* (FLC) to decide whether a lane departure is to happen in the future. Here, it was assumed that the prediction problem is *Vector Auto-Regressive*, meaning that any future states can be fully derived from previous states. The major contribution of the approach was the incorporation of *Sparse Historic Samplings* (SHS) to pre-process the input time series in real-time. Logarithmic sparse sampling showed to successfully reduce the computational complexity without any loss in the prediction accuracy. The model performance was then evaluated with respect to two aspects: the *Mean Square Error* (MSE), representing the accuracy of the path prediction, and the TP and FP of the LKA triggerings, representing the accuracy of the lane departure prediction. The proposed model showed to outperform the traditionally used CVM when benchmarked using all the above mentioned criteria.

The paper [7] presents a TA and DM approach for a LKA system using a *Multiple Non-Linear Regression* (MLNR) method. A MLP network consisting of four neuron layers with sigmoid activation function was proposed, which outputs a binary result of whether a lane departure is going to happen. The network was trained with the *Adaptive Moment Estimation* (ADAM) optimizer based on binary cross-entropy loss function. Furthermore, the input signals were standardized and sparsely sampled, yielding both memory efficiency and limits the noise influence from the input

signals. The TP and FP were evaluated based on the classification outcome and mean triggering time precision, and the results were benchmarked against a CVM model. The MLP model showed to significantly outperform CVM in both TP and FP.

2.4 Contributions

The main contribution of this project is the evaluation of the ANN TA model's performance in the defined ODD, as well as a performance benchmarking against the CVM.

2.5 Outline

The main sections of this thesis are presented in the following order: *Theory*, *Method*, *Result*, *Discussion* and *Conclusions*. The theory section describes the relevant theory of ANN and the CVM used as benchmark. The method section presents the methodology for pre-processing and annotating data, training and evaluating the ANN TA, as well as how the system failures were inspected within the established ODD. The results from the ANN TA training, evaluation and system failure assessment are presented in the result section. Discussions about the selection of prediction horizon, the performance of the ANN TA and the CVM, as well as failure mode assessment are presented in the discussion section. Finally, conclusions about the results and propositions for future work are presented in the conclusions section.

3 Theory

Section *3* details the theory behind the ML-based prediction models used in this thesis. Section *3.1* covers the essential background for ANN, which was used as the basis for the ANN TA prediction model. Section *3.2* presents the theoretical background of the CVM used as a benchmarking system.

3.1 Artificial Neural Network

This section deepens in how an ANN functions and how it may be used to solve a regression problem with time series data. Section *3.1.1* explains the general theory behind the construction of an ANN model. Section *3.1.2* describes how an ANN can be trained to learn the relation between the input and target output data. Section *3.1.2* also describes mini-batching, regularization and drop out which are techniques that are used during training to reduce overfitting and generalize the network traning.

3.1.1 Artificial Neural Networks

The goal of an ANN, also known as a MLP model or deep feed forward neural network, is to approximate a target function $\mathbf{y} = \mathbf{f_{tar}}(\mathbf{x})$ by establishing a mapping $\hat{\mathbf{y}} = \mathbf{f}(\mathbf{x}; \mathbf{w}, \boldsymbol{\theta})$. Here, $\mathbf{f}$ refers to the network configuration and $\mathbf{w}$ and $\boldsymbol{\theta}$ to the network parameters.

Although the mapping $\mathbf{f}(\mathbf{x}; \mathbf{w}, \boldsymbol{\theta})$ can contain a large amount of parameters, ANNs are considered to be non-parametric. A non-parametric model is a model that makes no assumptions about the

underlying distribution for the data generation. In the case of ANNs, the input data distribution has no meaning to how the mapping is constructed. Instead, the mapping is learned by iteratively tuning the parameters using a large amount of data, which is also referred to as a *data-driven model*. In general, non-parametric models are more robust and flexible [18]. However, no conclusions can be drawn regarding the mechanisms of the underlying function approximated by the ANN [19].

Feed forward neural networks are considered networks because they consist of several functions that are nested together, which together forms $\mathbf{f}(\mathbf{x}; \mathbf{w}, \boldsymbol{\theta})$. Take for example a feed forward network composed of one input layer, one hidden layer and one output layer, where each layer consists of an arbitrary number of neurons. The connection between two adjacent layers can be represented by a function $y = f(\mathbf{x})$ where $\mathbf{x}$ is a vector of the neuron states from the previous layer. The two layers can be denoted $f^{(1)}$ and $f^{(2)}$. These functions are nested together to form the final function $f(\mathbf{x}) = f^{(2)}(f^{(1)}(\mathbf{x}))$. The number of nested functions yield the depth of the network, hence the name *deep* neural networks.

A classical feed forward neural network is a collection of one-way connected artificial neurons. The network is constructed by layers of neurons, where the one-way connection takes place between two adjacent layers through weights. This means that neuron states are weighted before passed into the next layer of neurons. The smallest functioning unit consists of neurons with states $V^{(i,l-1)}$ where $(i, l-1)$ is neuron number i in layer $l-1$, pointing into one arbitrary neuron with a state $V^{(j,l)}$ in the next layer l, as illustrated in Figure 1.

Continuing with the example of a network with one input layer $\mathbf{x}$, one hidden layer $\mathbf{V}$ and one output layer $\mathbf{O}$, the states of the neurons in each of layer can be computed as follows:

$$V_j = g(b_j) \quad \text{with} \quad b_j = \sum_k w_{jk} x_k - \theta_j, \tag{1}$$

and

$$O_i = g(B_j) \quad \text{with} \quad B_j = \sum_j W_{ij} V_j - \Theta_j, \tag{2}$$

where $\mathbf{w_{jk}}$ are the weights connecting input neuron k to neuron j in the hidden layer, $\mathbf{W_{ij}}$ are the weights connecting neuron j in the hidden layer to neuron i in the output layer, θ_j is the bias for neuron j in the hidden layer, Θ_i is the bias of neuron i in the output layer and g is the *activation function*. The activation function is a function that is applied to the neuron's intermediate state b_j or B_i. Every layer must not have the same activation function, but every neuron in the same layer has the same activation function. The input layer has no activation function, thus, the output from the input layer is simply $\mathbf{x}$. The example network is illustrated in Figure 1.

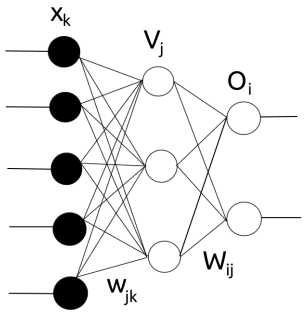

Figure 1: Illustration of a neural network with one hidden layer.

As mentioned before, the output **O** becomes the result of the nested function. The computation performed by the network becomes

$$O_i = g\left(\sum_j W_{ij} g\left(\sum_k w_{jk} x_k - \theta_j\right) - \Theta_i\right) \tag{3}$$

By using a non-linear differentiable (or piecewise differentiable) activation function, the network is able to learn complex non-linear relationships in the data as opposed to a linear activation function which may only learn linear relationships. Common activation functions include the *sigmoid function* $\sigma(x) = \frac{1}{1-e^{-x}}$, the *tanh function* $\tanh(x) = \frac{1}{1-e^{-2x}} - 1$, and the *Rectified Linear Unit* $ReLu(x) = max(0, x)$. See Figure 2.

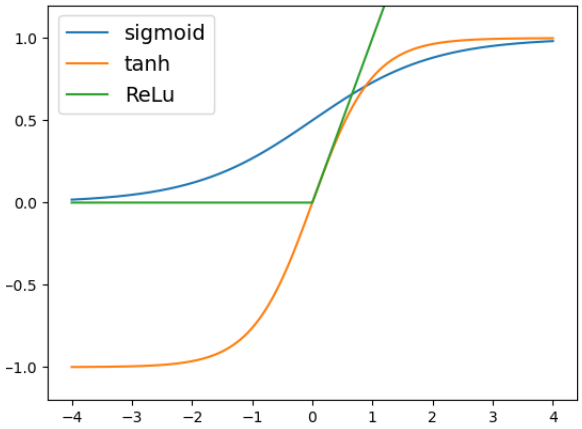

Figure 2: Three common activation functions: sigmoid, tanh and ReLu.

Connecting to the statement made in the beginning of this section, the goal of the network is to approximate some unknown function $\mathbf{f}(\mathbf{x})$. During training, the aim is to train the network so that the function of the output layer $\mathbf{f}(\mathbf{x})$ is equal to $\mathbf{f_{tar}}(\mathbf{x})$.

3.1.2 Training

When applying artificial neural networks, the strategy is to learn $f(\mathbf{x}; \mathbf{W}, \boldsymbol{\theta}) \approx f_{tar}(\mathbf{x})$ by gradually tuning the parameters $\mathbf{W}, \boldsymbol{\theta}$ utilizing *gradient descent*. This is achieved with backpropagation. Initially, one must define a loss function $\mathbf{L}(\mathbf{f}(\mathbf{x}; \mathbf{W}, \boldsymbol{\theta}), \mathbf{f}_{tar}(\mathbf{x})) = \mathbf{L}(\hat{\mathbf{y}}, \mathbf{y})$. Here, the squared error loss function is used for illustration purposes. Furthermore, every sample in the input data $\mathbf{x}$ must have an associated target $\mathbf{y}$ which the network aims to approximate.

Gradient descent is carried out as follows: first define a loss function L:

$$L = \frac{1}{2} \sum_n (y_i - O_i)^2 \tag{4}$$

where O_i is the output of the ANN and y_i is the target. Then, calculate the increment for which the weights between layer n and m should be updated:

$$\Delta W_{mn} = -\nu \frac{\partial L}{\partial W_{mn}} \quad \text{and} \quad \Delta w_{mn} = -\nu \frac{\partial L}{\partial w_{mn}} \tag{5}$$

where ν is the *learning rate*, and

$$\frac{\partial L}{\partial W_{mn}} = \sum_i (y_i - O_i) \frac{\partial O_i}{\partial W_{mn}}, \tag{6a}$$

$$\frac{\partial O_i}{\partial W_{mn}} = \frac{\partial}{\partial W_{mn}} g\left(\sum_j W_{ij} V_j - \Theta_i \right) = g'(B_i) d_{im} V_n. \tag{6b}$$

$$\tag{6c}$$

It is important to point out that the values V_j do not depend on W_{mn}. Finally, equation 5.1 can be written as

$$\Delta W_{mn} = -\nu \frac{\partial L}{\partial W_{mn}} = -\nu (y_m - O_m) g'(B_m) V_n. \tag{7}$$

The weight increments of the hidden layer are calculated in a similar way, but now the chain rule needs to be applied three times instead of two:

$$\frac{\partial L}{w_{mn}} = -\sum_i (y_i - O_i) \frac{\partial O_i}{\partial w_{mn}}, \tag{8a}$$

$$\frac{\partial O_i}{\partial w_{mn}} = \frac{\partial}{\partial w_{mn}} g\left(\sum_j W_{ij} V_j - \Theta_j \right) = g'(B_i) \sum_j W_{ij} \frac{\partial V_j}{\partial w_{mn}}, \tag{8b}$$

$$\frac{\partial V_j}{\partial w_{wm}} = \frac{\partial}{\partial w_{mn}} g\left(\sum_k w_{jk} x_k - \theta_j \right) = g'(b_j) d_{jm} x_n, \tag{8c}$$

where

$$d_{jm} = \begin{cases} 1, & \text{if} \quad j = m \\ 0, & \text{otherwise} \end{cases}, \tag{9}$$

finally yielding

$$\Delta w_{mn} = \nu \sum_i (y_i - O_i) W_{im} g'(B_i) g'(b_m) x_n. \tag{10}$$

To update the weights and perform gradient descent one performs the following:

$$W_{mn} \leftarrow W_{mn} + \Delta W_{mn}, \tag{11}$$

and

$$w_{mn} \leftarrow w_{mn} + \Delta w_{mn}. \tag{12}$$

The biases also need to be updated, and the update increment of the biases Θ and θ are calculated in a similar way, yielding a slightly different expression.

For a full batch training, this process is carried out once per sample in the data set. When the whole data set has been run through the training algorithm, one *epoch* has finished. Normally, several epochs are needed to attain an acceptable model. When dealing with large data sets, it is not feasible to use the entire data set at once during each epoch. This is due to memory limitations inherent to the hardware used during training. This problem can be solved by using *mini-batches*. Instead of using the whole data set at once and performing only one gradient descent step per epoch, one can use a smaller random subset of the data. This process allows for a smaller memory foot print as well for better generalization capabilities of the network [20]. However, mini batching generally decreases the rate at which the network converges [21].

A problem that arises when training a neural network on large data sets is *overfitting*. Overfitting emerges when the model learns the data "by hand". This means that when the model is presented with different data from the same distribution, it performs worse, even if the loss from the training data is low. To combat this phenomenon, a small subset of the data, the *validation set* can be used. During training, the model can be continuously evaluated on the validation set, which is not included in the training data. The model's loss on the training set will always decrease, but since the model is not presented with the validation set, its loss on the latter set will not continuously decrease. When the model starts overfitting on the training set, the validation loss will start to increase, and this is a sign that training should be stopped.

As mentioned above, overfitting is a big problem for ANN. Overfitting can be remedied with for example early stopping when the loss on the validation starts to increase after it has been decreasing. L_1, L_2 regression and *dropout* may also be used for this purpose [22].

L_1- and L_2 regression involves penalizing large weights in the network. This is done by adding a penalty term in the loss function:

$$Loss = L(f(x; (\mathbf{W}, \boldsymbol{\theta}, y) + \lambda \sum_i |w_i| \tag{13}$$

or

$$Loss = L(f(x; (\mathbf{W}, \boldsymbol{\theta}), y) + \lambda \sqrt{\sum_i w_i^2} \tag{14}$$

where equations 13 and 14 are L_1 and L_2 respectively, and λ is a positive non-zero constant. The implications of adding such a penalizing factor are that the network will favor smaller weights. This leads to more generalizing capabilities as well as avoidance of vanishing gradients.

Dropout can also be used to reduce overfitting. By setting the weights of arbitrary neurons in the network to 0 with some probability for each epoch during the training process, the network learns to use the data in a more generalized manner, because it has to use the remaining non-zero weights to make a prediction. Thus, the network must learn to compensate for the missing neurons, which has a robustifying effect on the network.

Another fundamental problem that arises from the central part of the training algorithm of ANN, being gradient descent, is *vanishing gradients*. As can be seen from Figure 2, for large absolute values of x for the sigmoid and tanh functions, the gradient approaches 0. If the gradient approaches 0, no adjustment will be done to the parameters, as can be seen in equation 10 and 11. This problem leads to slow or no learning. There are several remedies to this problem. A common one is to use the ReLu function, whose derivative is constant for all input values. However, the ReLu function has its own drawbacks. The fact that all negative input values yield an output value of 0 in the ReLu function can cause "dying neurons". This means that no gradients flow backward during backpropagation, hindering the learning process of the neural network [23].

3.2 Constant Velocity Model

In order to assess the performance of the proposed ANN TA, a base-line model was used as benchmark, namely the CVM-based TA. The CVM is a suitable benchmark due to its simple nature. It does not include dynamics that are inherent to a specific vehicle, and simply relies on Newton's laws of motion instead. In doing so, one removes the need to estimate the parameters needed for example in a *Dynamic Model*. In a dynamic model, parameters such as lateral forces need to be estimated. These parameters may change over time and depend on for example tire friction, vehicle mass, and front wheel angle, which makes it difficult to achieve a performance that is easily reproducible. Easy reproducibility is important when benchmarking.

As mentioned before, CVM produces metrics based on the single future behaviour of traffic participants. This metric can be defined either within the time domain or spatial domain, depending on how the ADAS is constructed to operate. A temporal CVM uses the *Time to Lane Crossing* (TLC) metric defined as

$$TLC_t = \frac{d_t^{\diamond}}{v_{lat,t}}. \tag{15}$$

Here, d_t is the lateral distance from the vehicle edge to the lane marker, and the diamond is a wild card symbol, which can be either the left or right side of the vehicle. TLC refers to the time for the vehicle to cross the lane marker from its current position.

In this thesis, the CVM was modified to be based on a metric defined within the spatial domain,

being the future distance to the lane marker $d^{\diamond}_{t+h}$:

$$d_{t+h} = d_t + v^{\diamond}_{lat,t} * H. \tag{16}$$

The future distance $d^{\diamond}_{t+h}$ refers to the distance from the vehicle edge to the left and right side lane markers h time steps ahead.

4 Method

This section details the chosen methodology used to achieve the project goals described in Section *2*. The section is divided into four parts. Section *4.1* describes the raw and pre-processed data sets used for training, validating and testing the ANN TA. Section *4.2* focuses on how TA is carried out by the ANN TA as well as how the comparison was made with the CVM model. Section *4.3* details how the performance evaluation of the ANN TA was conducted. Section *4.4* describes the definition of ODDs and how they are used in TA performance evaluation.

The complete work flow of Section *4* is illustrated in Figure 3, which can be divided into six main blocks: *Signal extractor*, *Event seeker*, *Splitter*, *TA*, *Training* and *Performance evaluation and ODD assessment*.

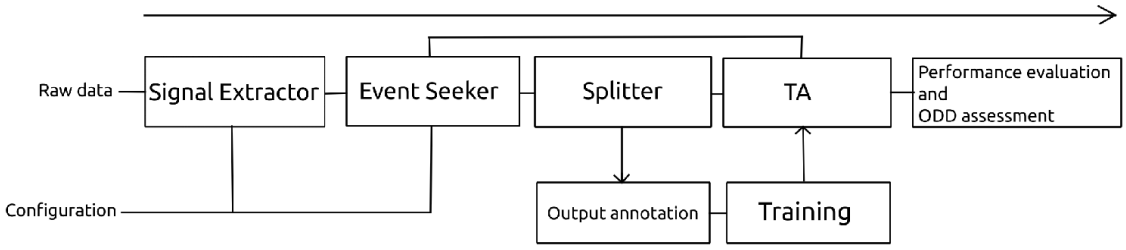

Figure 3: Schematic image of the project work-flow.

4.1 Data

Section *4.1.1* describes the signal configuration in the raw data set and the signal extraction techniques used to create the training, validation and testing data sets. Section *4.1.2* explains how the target output is annotated for these data sets. Finally, Section *4.3* briefly mentions the usage of the top camera video.

4.1.1 Signal Configuration

The real-world data set used for this thesis was collected by Zenuity and managed by Zenseact. The data set consists of sensor signals obtained by multiple vehicles from field test drivings in Sweden, Eastern Europe, Malaysia and the US. The signals were collected using a sampling frequency $f_s = 40Hz$, and includes among others ego-vehicle motion signals and road geometry signals. The purpose of the field test was to sample data from daily drivings having all in-car ADAS functionalities turned off. The field test data covers various scenarios in different weather and road conditions, and is still used for software development within Zenseact.

For this project, the sensor signals of interest are a subset of the ego-vehicle motions and road geometry signals, being the:

- Coefficients of the sensor estimated lane marker polynomial $p_t^\diamond = a_{0,t}^\diamond + a_{1,t}^\diamond x + a_{2,t}^\diamond x^2 + a_{3,t}^\diamond x^3$ to the left and right side of the ego-vehicle, relative to the center of the ego-vehicle rear.

- Range of view $r_t^{\diamond}$ relative to the center of the ego-vehicle rear, within which the polynomial $p_t^{\diamond}$ is considered valid (m).
- Vehicle longitudinal velocity v_t (m/s).
- Vehicle yaw rate w_t (rad/s).
- Vehicle front-wheel steering angle γ_t (rad).

The 13 above mentioned signals are illustrated in Figure 4, in the context of an arbitrary time instance during driving.

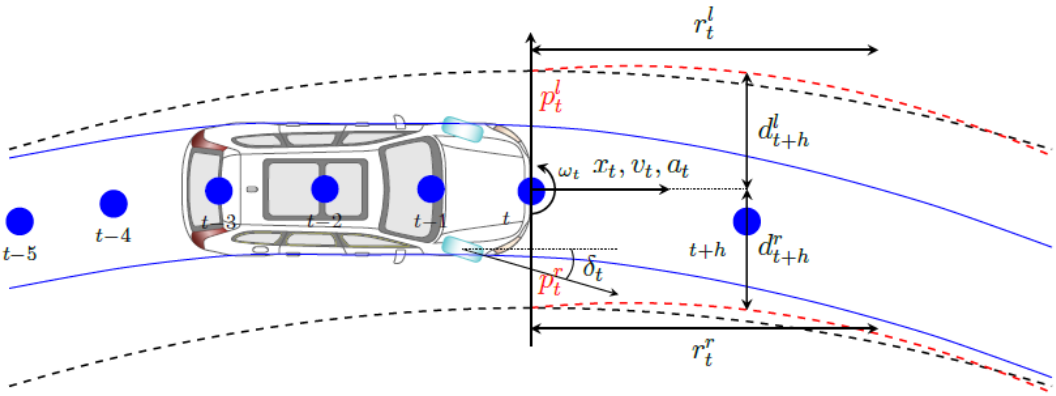

Figure 4: Visualization of the 13 signals of interest, in context of an arbitrary time instance t. The black dashed lines show the ground truth lane markers, the blue lines the vehicle side-edges, and the red dashed lines the sensor estimated lane markers. The blue dots refer to the vehicle state for a time instance t as well as the historical states five time steps back [24].

The raw unprocessed data consists of approximately 316000 one minute *sequences* of collision-free highway and country road drivings. From the raw unprocessed data, the *Signal extractor* block extracts the 13 signals of interest along with other signals that are required for the functionality of the other blocks.

The *Event Seeker* and *Splitter* blocks follow *Signal Extractor*. *Event Seeker* identifies sequences that satisfy specific conditions, while *Splitter* truncates the sequences into *segments* of lengths of $L = 495$ time steps. In the end, two sets of data sets were created: a normal driving set and a departure driving set, defined as $\mathbb{N}$ and $\mathbb{D}$ respectively. $\mathbb{D}$ is further randomly divided into a training set $|\mathbb{D}_{trn}| = 9632$, a validation set $|\mathbb{D}_{vld}| = 784$ and a test set $|\mathbb{D}_{tst}| = 784$, following the proportion $86:7:7$. $|\mathbb{N}| = 4000$ is only used for testing and can therefore be denoted as $\mathbb{N}_{tst}$.

$\mathbb{D}$ contains truncated segments of length $L = 495$ with one and only one time instance corresponding to a lane departure event, located at the very last time step of the segment. Time instances that satisfy the below formula are treated as a "lane departure event":

$$p_t^{left}(Veh_{len}) - \frac{Veh_{wid}}{2} > 0 \quad || \quad p_t^{right}(Veh_{len}) - \frac{Veh_{len}}{2} < 0, \tag{17}$$

here, Veh_{len} and Veh_{wid} are the actual vehicle length and width of the corresponding vehicle. Segments containing events are then further truncated into n time step long segments, where $n = 4 * H * f_s$ for $\mathbb{D}_{trn}$ and $\mathbb{D}_{vld}$, and $n = 400$ for $\mathbb{D}_{tst}$. The selection of segment lengths are inspired by [9]. The segment also needs to satisfy the following conditions:

- The road must have a radius of minimum 200 meters.
- The ego lane must have a lane width of maximum 4 meters.
- The vehicle must travel at a speed of 60 km/h (17m/s) or more.
- The vehicle does not perform a lane change during or up to 3 seconds after the event
- The segment only contains one event at the last time step
- The sensor estimated lane markers must be of good quality
- The difference between a signal at time t and $t + 1$ may not exceed a certain threshold

These conditions are evaluated using the 13 signals of interest as well as other signals extracted in the *Signal Extractor* block. For example, the *lane-change-indicator* signal was used to identify and eliminate lane departures that arises from intentional lane changes, and the *lane-width* signal was used to eliminate wide lanes. After applying *Signal Extractor*, *Event Seeker* and *Splitter*, $\mathbb{D}$ contains filtered time series signal segments which represents driving that leads to an unintended lane departure.

$\mathbb{N}$ contains driving segments without any lane departure events. Every time step within the segment satisfies the formula:

$$p_t^{left}(Veh_{len}) - \frac{Veh_{wid}}{2} < 0 \quad \&\& \quad p_t^{right}(Veh_{len}) - \frac{Veh_{wid}}{2} > 0 \tag{18}$$

As mentioned above, $\mathbb{N} = \mathbb{N}_{tst}$ is only used for testing. The segment length for $\mathbb{N}_{tst}$ is $n = 400$, and needs to satisfy the above mentioned conditions except the requirement of containing an event at the last time step. $\mathbb{N}$ contains filtered time series signal segments which represent driving that does not lead to any lane departures.

4.1.2 Target Output Annotation

After Section *4.1.1*, input data for the ANN TA is constructed. In order for the ANN to be trained, target outputs also need to be defined. As mentioned in the project scope in Section *2.1*, the ANN aims to predict the future distance to the lane marker $d^{\diamond}_{t+h}$, $h = H * f_s$ time steps ahead. $d^{\diamond}_{t+h}$ can be calculated using the lane marker polynomial $p^{\diamond}_t = a^{\diamond}_{0,t} + a^{\diamond}_{1,t}x + a^{\diamond}_{2,t}x^2 + a^{\diamond}_{3,t}x^3$. For a chosen h, the target output $y_{t,h}$ for a time instance t is simply

$$y_{t,h} = d^{\diamond}_{t+h} = p^{\diamond}_{t+h}(0) = a^{\diamond}_{0,t+h} \tag{19}$$

For every segment in $\mathbb{D}$, target outputs were created for 8 different H using Equation 19, being H = [0.2s, 0.5s, 0.75s, 1s, 1.25s, 1.5s, 1.75s, 2s], which corresponds to h=[8, 20, 30, 40, 50, 60, 70, 80]. Note that for the corresponding $h = H * f_s$, H *seconds ahead* and h *steps ahead* refers to the same future aspect but measured in different units. The span and granularity of H was inspired by [9].

The span of H comes from a balance between empirical knowledge of ANN prediction capabilities and the prediction's usability in reality. Intuitively, shorter H should provide better prediction performance, as showed by [17]. However, longer H is more valuable for any ADAS application. $H < 0.2s$ would for example result in a accurate but impractical prediction, since a warning $0.2s$ ahead would be almost impossible for the driver to react to, nor for an automated system to safely intervene. On the other hand, $H > 2s$ would result in a valuable but unstable prediction, since a prediction from ANN $2s$ ahead are expected to be very unstable in scenarios where the driving environment varies frequently.

4.1.3 Top Camera Video

Each 1-minute sequence from the raw data set has a corresponding 1-minute long video recorded by the in-vehicle top camera. These videos were collected to add a context of the traffic situation to the input signals in the failure mode assessment.

4.2 Threat Assessment

This section describes how threat assessment is carried out by the ANN TA and the CVM-based TA, which corresponds to the *TA* block illustrated in Figure 3. Section *4.2.1* introduces the concept of time series windowing, which is an essential part of how the input historical data is sampled for the ANN TA. Section *4.2.2* details the network construction and training for this project. Section *4.2.3* presents the constructed CVM used for benchmarking. Then section *4.2.4* explains the decision making logic used by a TA. At the end of this section, the whole work flow for constructing the ANN TA and the CVM-based TA should be clear.

4.2.1 Time Series Windowing

The ANN TA requires feeding in a time series of historical signal data. The essential technical point here is how to properly sample these time series from the entire driving history. This gives rise to the problem of balancing between computational complexity and model accuracy. An intuitive sampling technique would be to use a fixed number of consecutive previous samples using a uniform sampling rate coinciding with the sample time $T_s = 1/f_s = 0.025s$. However, this could potentially lead to over sampling problems, due to too small differences between consecutive time steps. Consequently, a sparse sampling technique using a specific sample offset pattern could be a better solution. An offset pattern is defined as:

$$\Gamma = \{\gamma_0, \gamma_1, ..., \gamma_n\} \tag{20}$$

where γ_n can be arbitrarily chosen while fulfilling the constraints $\gamma_i \in \mathbb{Z}$, $\gamma_i >= 0$ and remains unique in Γ. The sampled time series using Γ for a specific time step t would therefore become:

$$\{t - \gamma_0, t - \gamma_1, ..., t - \gamma_n\} \tag{21}$$

4.2.2 Artificial Neural Network Construction and Training

Several artificial neural network types are available for solving a time series regression problem, some of which are vanilla neural networks, *Long-Short-Term-Memory* (LSTMs) networks, *Recurrent Neural Networks* and *Convolutional Neural Networks* (CNNs). Every type of network has its

advantages and disadvantages. In this project, vanilla neural network models are used, denoted as ANNs since it is considered as the most classical ANNs. One advantage of ANNs is it's relatively low computational complexity compared to the other types. Low computational complexity infers faster training and validation. Moreover, low computational complexity is a major factor in real-time applications such as threat assessment in traffic situations.

Six different ANN architecture candidates were constructed as a basis for the ANN TA. As inspired by [17], these ANN all had five fully connected layers with three hidden layers sharing the amount of neurons, being 16, 40, 64, 128, 256 and 512 neurons. For each of the network architectures, training was conducted individually for the target outputs annotated according to the 8 different prediction horizons H.

Every network was supplied with the same input training data $\mathbb{D}_{trn}$. The offset pattern that was applied to the input signals for obtaining sparse sampled historical time series are presented in Table 1.

$\mathbf{\Gamma}$	**Offset pattern**	**Depth** (s)
Γ_1	[0,1,2]	0.05
Γ_2	[0,1,14]	0.35
Γ_3	[0,7,14]	0.35
Γ_4	[0,2,9,14]	0.35
Γ_5	[0,5,10,15,20,25,30,35,40]	1
Γ_6	[0,1,2,3,4,5,9,14,20,39]	0.975
Γ_7	[0,10,20]	0.5

Table 1: The seven offset patterns tested for the network configurations.

The selection of the offset patterns was inspired by previous work [9] where the authors investigate different types of offset patterns. The authors state that a logarithmic pattern with a depth of 0.5-1 seconds is enough for maintained performance. This corresponds to the recent 20-40 historical samples before the current time instance.

The combination of 6 different network architectures, 8 target output annotations and 7 offset patterns amounts to $6 * 7 * 8 = 336$ different network configurations that were to be trained. During training, an initial learning rate of 0.001 was used together with an ADAM optimization algorithm [25]. MSE was used as the loss function. Since the network has two outputs, left side prediction d^l_{t+h} and right side prediction d^r_{t+h}, the average of the loss of the two outputs was used to calculate the MSE. All layers except the output layer used the ReLu activation function.

4.2.3 Constant Velocity Model Construction

For benchmarking purposes, a CVM was developed to predict a future lateral distance to the left and right side of the lane marker (DLC) h time steps ahead. Following the assumption that the longitudinal velocity stays the same within the prediction horizon H, the predicted lateral distance $\hat{d}_{t+h}$ to the lane marker h time steps ahead can be expressed by:

$$\hat{d}^o_{t+h} = p^\diamond_t(0) + u^\diamond_t H, \tag{22}$$

where $h = H * f_s$, and $u_t^\diamond$ is the lateral velocity toward the lane markers on left and right side.

Notice that the $u_t^\diamond$ is not included in the extracted signals, but can instead be calculated using $p_t^\diamond$ and v_t as follows:

$$u_t^\diamond = v_t \sin \psi_t^\diamond, \tag{23}$$

and the instantaneous rate of change $\psi_t^\diamond$ of the distance to the lane marker can be calculated by:

$$\psi_t^\diamond = \left| \frac{dp_t^\diamond}{dt} \right|_{x_t=0} = a_{1,t}^\diamond. \tag{24}$$

Therefore, the predicted lateral distance $\hat{d}_{t+h}$ expressed using extracted signals becomes:

$$\hat{d}_{t+h}^\diamond = p_t^\diamond(0) + u_t^\diamond H = p_t^\diamond(0) + v_t H \sin a_{1,t}^\diamond. \tag{25}$$

4.2.4 Decision Making

For a time instance t and a specified H, the model's predicted lateral distance to the lane marker $d_{t+h}^\diamond$ can be utilized to make a decision Q on whether or not a warning or steering intervention should be carried out. Q can be defined as follows:

$$Q_t = \begin{cases} 1, & d_{t+h}^\diamond - \frac{w}{2} < \tau \\ 0, & otherwise \end{cases}, \tag{26}$$

where τ is a manually defined threshold for how the vehicle should be positioned in relation to the lane marker for an intervention to be triggered. The predicted $d_{t+h}^\diamond$ uses the center of the vehicle rear as a reference. This means that when $\tau = 0$, the vehicle rear wheel needs to be exactly on or outside of the lane marker at time instance t to trigger the system. $\tau < 0$ means that the rear wheel needs to be $|\tau|$ m outside of the lane marker for a trigger, and vice versa. The time instance that results in a trigger is defined as the event time instance for the rest of the thesis. Note that based on the DM logic defined above, both ANN and CVM can output multiple event time instances for one segment.

The event time instance can further be used for calculating the model's so called *triggering time* for a given segment. The triggering time is defined as the time difference between the very first event time instance to the end of the segment. This means that triggering time is always negative. As explained in Section *4.1.1*, segments in $\mathbb{D}$ are designed to have one and only one lane departure at the end of the segment. Therefore, the triggering time refers to the time between the event time instance and the segment end. Since the definition of the triggering time requires the segments to have a lane departure, it can not be calculated for segments in $\mathbb{N}$. For every model, a *mean triggering time* can then be calculated by averaging over all triggering times obtained from $\mathbb{D}$. From here, the ANN TA and the CVM are fully constructed.

4.2.5 Tuning of the Triggering Time

In this project, τ is initially set to 0. Then, Algorithm 1 is used to re-calibrate τ to yield a mean trigger time that is approximately equal to the H used for the target output annotation. The aim of this procedure is to obtain a fair baseline between each of the trained ANN TA models and the

CVM.

The mean triggering time of the TA model is tuned by iteratively modifying the threshold τ in Equation 26 for which the TA triggers a warning. The threshold is iteratively increased or decreased based on if the TA triggers a warning too late or too early. The algorithm ends when the mean triggering time is approximately equal to the prediction horizon. See Algorithm 1.

Algorithm 1 Tune the triggering time of the ANN

$\tau \leftarrow 0$
$t \leftarrow MeanTrigTime(\tau)$
$\boldsymbol{\tau} \leftarrow \tau, \mathbf{t} \leftarrow t$
if $h \leq t$ **then**
  while $h \leq max(\mathbf{t})$ **do**
    $\tau \leftarrow \tau - \Delta\tau$
    $t \leftarrow MeanTrigTime(\tau)$
    $\boldsymbol{\tau} \leftarrow \tau, \mathbf{t} \leftarrow t$
  end while
else
  while $h \geq min(t)$ **do**
    $\tau \leftarrow \tau + \Delta\tau$
    $t \leftarrow MeanTrigTime(\tau)$
    $\boldsymbol{\tau} \leftarrow \tau, \mathbf{t} \leftarrow t$
  end while
end if
Tuned trigger time: $\tau_{tuned} \leftarrow Interpolate(\mathbf{t}, \boldsymbol{\tau}, h)$

4.3 Performance Evaluation

In this section, two performance evaluation methods will be presented: statistical performance and TA performance. Statistical performance evaluation is only conducted for the ANN TAs. The statistical performance evaluation evaluates the network's ability to correctly predict the lateral distance to left and right lane markers. Threat assessment performance is evaluated for the ANN TAs and the CVM-based TA. The threat assessment performance describes a TA's ability to accurately predict a lane departure. To summarize, the statistical performance targets the accuracy of the predicted $d^{\diamond}_{t+h}$, while the TA performance targets the accuracy the predicted event time instance compared to the lane departure.

4.3.1 Statistical Performance

The statistical performance of the ANN's ability to predict the correct distance to the lane marker h steps ahead is evaluated using MSE. The MSE is the most common choice of loss function when solving regression problems, and is defined as follows:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y - \hat{y})^2, \tag{27}$$

here, $y = a^{\diamond}_{t+h}$ and $\hat{y} = d^{\diamond}_{t+h}$.

Since the network has two outputs, the MSE will be the average over the two. The network might perform better for one of the two outputs and still yield an acceptable MSE. It is also worth mentioning that since the MSE is a sample-wise average, the MSE might be low for normal driving situations, and thus yield a good results. However, it can be higher in situations that deviate from normal driving behavior, such as lane departures, which are less prevalent in the data.

4.3.2 Threat Assessment Performance Evaluation

A TA's ability to predict lane departures can be evaluated using accuracy, TPR and FPR. These metrics can be calculated from a confusion matrix, which uses the number of true positives, true negatives, false positives and false negatives. See the following equations.

$$\text{TPR} = \frac{\text{TP}}{TP + FN} \tag{28}$$

$$\text{FPR} = \frac{\text{FP}}{TP + FN} \tag{29}$$

$$\text{Accuracy} = \frac{\text{TP + TN}}{TP + TN + FP + FN} \tag{30}$$

To measure these metrics, $\mathbb{D}_{tst}$ and $\mathbb{N}_{tst}$ were used. For segments in $\mathbb{N}_{tst}$ the TA can either yield a FP or a TN; if the TA at any time step predicts a lane departure, it is classified as an FP; if it does not predict a lane departure, then the segment is classified as a TN.

Classifying the segments from $\mathbb{D}_{tst}$ is more complex. Here, the $4H$ seconds long segments are divided into 2 windows: a *normal driving window* spanning from segment start to $-2H$ before the segment end, and a *departure driving window* spanning from $-2H$ before segment end to segment end. This is illustrated in Figure 5. The departure driving window is also called the *acceptance window*, meaning the window within which a predicted departure can be classified as a true departure. If a lane departure is predicted within the normal driving window, it is classified as a FP. However, because the TA predicted a lane departure in the normal driving window, no evaluation will be made inside the lane departure window. Thus, the segment is also classified as a FN. The reason no evaluation is made inside the lane departure window is that if a lane departure is detected during normal driving in reality, the TA will be disabled due to an actuation being carried out. Thus, the input to the TA is no longer coming from the driver, but the vehicle performing an actuation. If no lane departure was detected during the normal driving window of the segment, it will be classified as a TN. Evaluation is then made on the lane departure window of the segment. If a lane departure is predicted on the correct side, it is classified as a TP. If a lane departure is detected on the wrong side, it will be classified as a FP, as the TA failed in warning on the correct side. Finally, if no lane departure is found, it is simply classified as a FN.

For the rest of the thesis, positives (TPs and FPs) will be referred to as "ANN TA triggerings". Segments classified as FP will be referred to as "FP segments". Similarly, segments classified as a TP will be referred to as "TP segments".

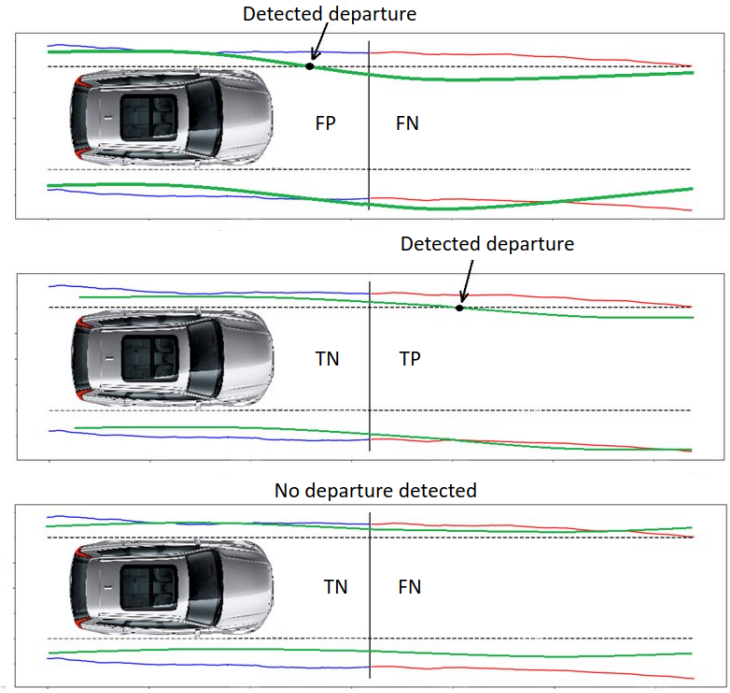

Figure 5: Visualization of the three possible TA outcomes in a segment in $\mathbb{D}_{tst}$. The outcome depends on the location of the event time instance. The blue part is the interval corresponding to the normal driving window, and the red part is the acceptance window. Both windows are $2H$ long. The blue and red lines are the sensor estimated lane markers, while the black dashed lines are the car side-edges. The green lines are the ANN TA predicted lane markers.

4.4 Operational Design Domain Evaluation and Failure Mode Assessment

For a classical physics-based model, an ODD needs to be carefully defined before the model construction. This is because the model is based on the physical assumptions that can be drawn from the ODD. However, for a TA model established in a data driven manner, the ODD for which the model is designed to operate in is defined by the data set used during the model training. For the ANN TA in this thesis, the ODD is the vehicle states and driving environments captured within the segments from $\mathbb{D}_{trn}$. These can be reflected by among others the 6 restriction conditions applied in the Splitter block when truncating segments, as described in Section *4.1.1*. Since the segments in the data sets were constructed from the same raw data, segments from $\mathbb{D}$ and $\mathbb{N}$ belong to the same ODD.

In order to evaluate an ADAS in a defined ODD, the system needs to be tested in various *scenarios* in the ODD. These scenarios are usually collected through real-word driving or simulated in a virtual environment. In this thesis, the performance of the ANN TA in the ODD was evaluated using *semi-virtual scenarios* derived from segments in $\mathbb{D}_{tst}$ and $\mathbb{N}$.

A semi-virtual scenario for a segment is defined as the composition of two parts: a Python simulation of the time series signals and the corresponding ANN TA outcome of a segment, together with the segment's top camera video. The time series signals provide the time-evolution information of among others the ground truth lane markers ($p_t^{\diamond}$) and the vehicle state. The python simulated ANN TA outcome are the same as the real outcome if the ANN TA would have been implemented in-vehicle. The top camera video visualizes the real-world traffic context for the segment. Together, the three parts constitutes a virtual scenario similar to the real-world driving scenario from which the segment was collected.

A simulation tool named *Birdview* was designed to serve in the evaluation process. For a given segment, Birdview visualizes the sensor estimated lane markers, the ANN TA predicted lane markers together with the threat assessment result using a live plot. The ANN TA predicted lane markers are visualized by interpolating over the predicted lane marker positions over different H, $d_{t+h}^{\diamond}$. By synchronizing the segment's top camera video with the Birdview live plot, a semi-virtual scenario as mentioned above could be visualized, in which the context of an ANN TA triggering could be identified. Figure 6 illustrates such a semi-virtual scenario. The semi-virtual scenario of a segment will be referred to as a *scenario* for the rest of this thesis.

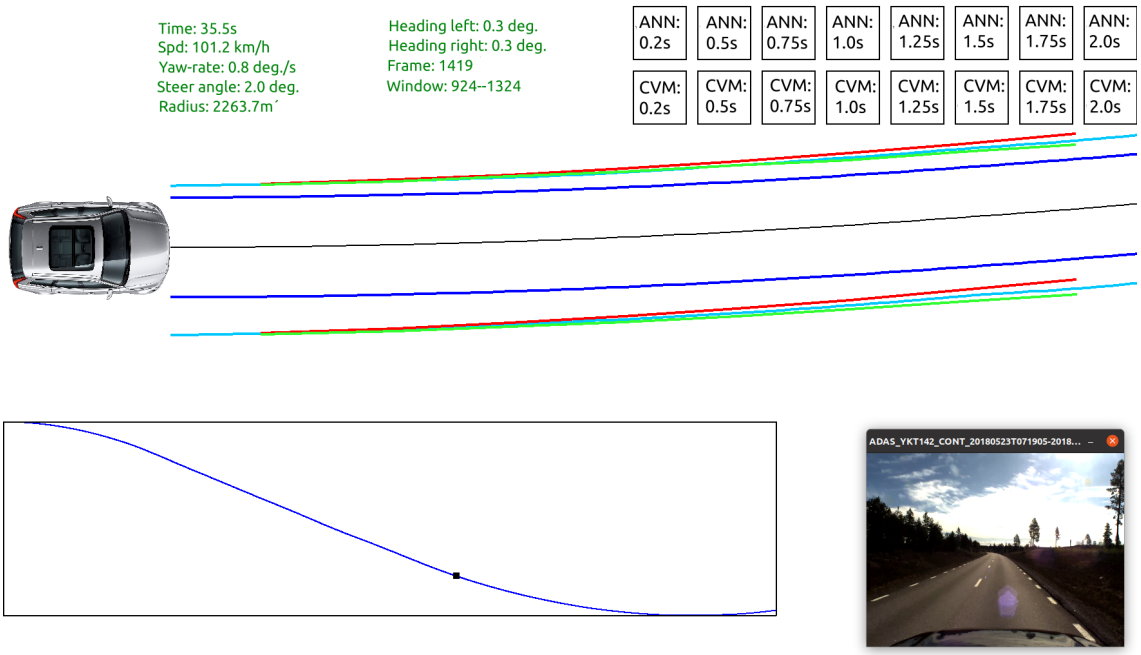

Figure 6: Visualization of a scenario using Birdview and the top camera video. The Birdview live plot includes the predicted lane marker position (green lines), the sensor estimated lane markers (turquoise lines) and vehicle side-edges (dark blue lines). In the upper right corner, TPs and FPs are visualized for both the ANN TA and CVM over H = [0.2,0.5,0.75,1,1.25,1.5,1.75,2] on the top row and bottom row respectively. Green indicates a TP, red indicates a FP, while white represents a negative. The bottom right corner shows the top camera feed. Bottom left corner shows the entire path for the whole one minute sequence, and the top left corner shows the current time (s) and frame, as well some ego-vehicle input signals for the current frame. The "window" refers to the frames for which the ANN TA is evaluated on.

ODD evaluation was carried out on all scenarios in which the ANN TA mistakenly triggered (FP), as well as a subset of scenarios for which ANN TA triggered correctly (TP). The reason to inspect scenarios which were classified as TP were to give a complimentary understanding of the FP cases. Having the time event instance of the ANN TA triggering and the complementary traffic context, the *failure modes* or *success modes* of these scenarios could be classified, presented in Table 2 and Table 3. The complete tables with descriptions for each failure and success modes are shown in Appendix A. A failure mode is defined as the contingency in a scenario that does not result in a true lane departure but which the ANN TA classified as a positive. A success mode is defined as the contingency in a scenario that results in a true lane departure, which the ANN TA classified as a positive. The failure and success modes were further grouped into categories, sub-categories and sub-sub-categories as can be seen in the tables. This was to identify categories with varying granularity for which the system fails or succeeds, as well as for analysing the distribution between the different groups. For clarity, active driving behaviour is defined as driving where the driver is actively placing the vehicle in a sub-optimal position in the lane to keep a safe distance from threats in traffic. Passive driving behavior is defined as driving where the driver places themselves in a sub-optimal position in the lane without any apparent threats. Here, sub-optimal position

refers to a vehicle placement in the lane that is regarded as being generally unsafe compared to being in the middle of the lane, for example being close to the lane marker. In the context of TP and FP scenarios, a sub-optimal position refers to a position that triggers the ANN TA. The position does not necessarily have to be a true departure from lane, but close enough to confuse the ANN TA logic. Multiple sub-sub-categories can share a scenario depending on the ANN TA trigger context. However, a scenario can not be classified into both the active sub-sub-category and passive sub-sub-category simultaneously, which is intuitive. For the rest of the thesis, scenarios that are classified as FP will be referred to as failure scenarios, and scenarios classified as TP as success scenarios.

101 out of 123 failure scenarios from $\mathbb{D}_{tst} + \mathbb{N}$ were used when identifying failure modes. The 22 failure scenarios without a top camera video were excluded. For the success scenarios, a random subset of TP segments from $\mathbb{D}_{tst}$ were used, being 101 out of 735 TP segments. Success modes were used to complement the observed active and passive driving behaviour sub-sub-category distribution for failure modes. Therefore, other sub-sub-categories are excluded from the success mode table.

Table 2: Table of failure mode and their classes, sub-classes and sub-sub classes.

Categories	**Sub-categories**	**Sub-sub-categories**	**Failure Mode**
Sensor	Luminosity		Low exposure
	Weather		Rain
Scenario	Lane tracking	Lane marker	Arrow in lane
			Unstable double tracking
			Type change
			Rumble strip
		Road condition	Crest, slope or bump
	Driving behaviour	Active driving behaviour	Close to lane marker
			Taking out curve
			Cutting curve
			Drifting
			Preparing for lane change
		Passive driving behaviour	Close to lane marker
			Taking out curve
			Cutting curve
			Drifting
Tuning	DM tuning		TP followed by FP

Table 3: Success modes and their categories, sub-categories and sub-sub-categories with short descriptions.

Categories	Sub-categories	Sub-sub-categories	Success mode
Scenario	Driving behaviour	Active	Planned maneuver
			Corrective maneuver
		Passive	Inattention

5 Result

This section presents the results from the performance evaluation of the ANN TA and the CVM-based TA. Section *5.1* presents results from the threat assessment- and statistical performance evaluation of the 336 network configurations. The aim of this section was to find the optimal network configuration, as well as investigate the effect of using different offset patterns Γ and prediction horizons H. Section *5.2* shows the threat assessment results for the CVM-based TA, which was used to benchmark against the ANN TA. In Section *5.3* the results from the ODD analysis of the ANN TA with the optimal configuration found in Section *5.1.1* is presented.

5.1 Artificial Neural Network-Based Threat Assessor Result

In Section *5.1.1* the result of the ANN TA threat assessment performance evaluation is presented. Here, the process of choosing the best performing network to use in the ODD assessment is described. Section *5.1.2* presents the result of the statistical performance evaluation carried out on each of the best performing network architectures.

5.1.1 Threat Assessment Performance

The 336 network configurations, mentioned in Section *4.2.2*, were trained on $\mathbb{D}_{trn}$ and tested on $\mathbb{D}_{tst}$ and $\mathbb{N}$. As mentioned in Section *4.2.2*, the network configuration has three free parameters: the network architecture, the offset pattern Γ and the prediction horizon H. Since the thesis aimed to obtain an ANN TA with the best possible performance, practical aspects regarding the selected ANN's parameters were not considered. Therefore, the ANN TA with the best performance obtained in theory might not be the optimal choice in practice, i.e., the selected prediction horizon might be too short to be effective. To meet the project scope described in Section *2.1* of investigating Γ's and H's effect on the TA's predictive performance, the optimal ANN architecture was first determined. Using the performance metric *accuracy*, the best configuration for each of the 6 constructed network architectures is presented in Table *4*. Here, for every fixed ANN architecture, Γ and H were treated as free parameters and the combination yielding the best accuracy was chosen. As can be observed from Table 4, for all ANN architectures, the best performance was obtained at the $0.5s$ prediction horizon. Furthermore, uniform and logarithmic offset patterns with depth ≈ 1 second showed to provide the best results. The ANN with $[128, 128, 128]$ neurons ended up having the best TA performance metrics by yielding both the highest TPR (0.938) and accuracy (0.969) as well as lowest FPR (0.0258).

Table 4: Comparison of the TA performance result over all network architectures for their respective best configurations

Γ	Architecture	Horizon	TP	TN	FP	FN	TPR	FPR	Accuracy
Γ_5	[512, 512, 512]	0.5	733	4653	131	51	0.935	0.0274	0.967
Γ_6	[256, 256, 256]	0.5	734	4658	126	50	0.936	0.0263	0.968
Γ_6	**[128, 128, 128]**	**0.5**	**735**	**4661**	**123**	**49**	**0.938**	**0.0258**	**0.969**
Γ_5	[64, 64, 64]	0.5	734	4658	126	50	0.936	0.0263	0.968
Γ_6	[40, 40, 40]	0.5	732	4655	129	52	0.934	0.0270	0.967
Γ_5	[16, 16, 16]	0.5	735	4659	125	49	0.938	0.0261	0.969

After concluding that the best architecture was $[128, 128, 128]$, the offset pattern's effect on the networks TA performance was investigated. Configurations for every Γ with architecture $[128, 128, 128]$ and $H = 0.5$ were evaluated and the results are presented in Table 5. Here, $H = 0.5$ was a natural choice according to the above mentioned conclusions from Table 4. $\Gamma_6 = [0, 1, 2, 3, 4, 5, 9, 14, 29, 39]$ had the best performance with the highest TPR (0.938)/accuracy (0.969) and the lowest FPR (0.257). It is worth mentioning that out of the 49 segments classified as FN's, only 2 were actually yielded from not correctly predicting a departure in the departure window of $\mathbb{D}_{tst}$. The rest of the FN's segments were due to a FP in the normal driving window of $\mathbb{D}_{tst}$.

Table 5: Comparison of the TA performance result over all offset patterns for the best network architecture and their respective best prediction horizon

Γ	Architecture	Horizon	TP	TN	FP	FN	TPR	FPR	Accuracy
Γ_1	[128, 128, 128]	0.5	721	4633	151	63	0.920	0.0316	0.962
Γ_2	[128, 128, 128]	0.5	727	4640	144	57	0.927	0.0301	0.964
Γ_3	[128, 128, 128]	0.5	727	4641	143	57	0.927	0.0299	0.964
Γ_7	[128, 128, 128]	0.5	728	4649	135	56	0.929	0.0282	0.966
Γ_4	[128, 128, 128]	0.5	730	4647	137	54	0.931	0.0286	0.966
Γ_5	[128, 128, 128]	0.5	730	4657	127	54	0.931	0.0265	0.967
Γ_6	**[128, 128, 128]**	**0.5**	**735**	**4661**	**123**	**49**	**0.938**	**0.0257**	**0.969**

Having fixed the optimal network architecture to $[128, 128, 128]$ and offset pattern to $\Gamma = [0, 1, 2, 3, 4, 5, 9, 14, 29, 39]$, Table 6 shows the TA performance results over different prediction horizons H. Coinciding with the observations from Table 4, $H = 0.5$ provides best TA performance with the highest TPR (0.938)/accuracy (0.969) and lowest FPR (0.0257).

Table 6: Comparison of the TA performance result over all prediction horizons for the best network architecture and their respective best offset pattern.

Γ	**Architecture**	**Horizon**	**TP**	**TN**	**FP**	**FN**	**TPR**	**FPR**	**Accuracy**
Γ_6	[128, 128, 128]	0.2	610	4554	240	174	0.778	0.0502	0.926
Γ_6	**[128, 128, 128]**	**0.5**	**735**	**4661**	**123**	**49**	**0.938**	**0.0257**	**0.969**
Γ_6	[128, 128, 128]	0.75	738	4636	148	46	0.941	0.0310	0.965
Γ_6	[128, 128, 128]	1.0	734	4564	220	50	0.936	0.0460	0.952
Γ_6	[128, 128, 128]	1.25	697	4406	378	87	0.889	0.0790	0.916
Γ_6	[128, 128, 128]	1.5	660	4205	579	124	0.842	0.121	0.874
Γ_6	[128, 128, 128]	1.75	612	3968	817	172	0.781	0.171	0.822
Γ_6	[128, 128, 128]	2	548	3701	1085	236	0.699	0.227	0.763

From here, the best NN-based TA has been determined. Its configuration is a three layer feed forward ANN with $[128, 128, 128]$ neurons, an offset pattern $\Gamma = [0, 1, 2, 3, 4, 5, 9, 14, 29, 39]$ and a prediction horizon $H = 0.5$.

5.1.2 Statistical Performance

The MSE and the *Root Mean Square Error* (rMSE) of the networks in Table 4 are presented in Table 7. It can be observed that the differences in root mean square error between the networks are small. The best configuration in this regard has a root mean square error of 0.0854, which can be visualized in the context of the vehicle speed restriction applied to the data set segments, presented in Section *4.1.1*. When driving at the lowest allowed speed (17 m/s), the ANN TA predicts the lateral distance $8.5 (= 17\text{m/s} \cdot 0.5\text{s})$ meters in front of the vehicle. At this distance, the average difference between the predicted lane marker position $\hat{a}^{\diamond}_{0,t+h}$ and the sensor estimated lane marker $a^{\diamond}_{0,t+h}$ is approximately 8.5 cm.

Table 7: Comparison of MSE and rMSE over all network architectures for their respective best configurations

Γ	**Architecture**	**Horizon**	**MSE**	**rMSE**
Γ_5	[512, 512, 512]	0.5	0.00808	0.0899
Γ_6	[256, 256, 256]	0.5	0.00778	0.0882
Γ_6	[128, 128, 128]	0.5	0.00762	0.0873
Γ_5	[64, 64, 64]	0.5	0.00751	0.0866
Γ_6	[40, 40, 40]	0.5	0.0073	0.0854
Γ_5	[16, 16, 16]	0.5	0.00727	0.0853

As can be seen from Table 7, network architectures with fewer neurons in each layer provides lower MSE and rMSE, which is as expected. This is because smaller networks converge faster, and larger networks need more data to converge. However, the difference in statistical performance between the configurations are deemed insignificant. Therefore the best configuration found in Section *5.1.1* with 128 neurons, offset pattern $\Gamma = [0, 1, 2, 3, 4, 5, 9, 14, 20, 39]$ and $H = 0.5$ is still treated as the optimal configuration.

5.2 Constant Velocity Model-Based Threat Assessor Result

For the CVM, the only free parameter is the prediction horizon. The TA performance is evaluated over different H, and the results are presented in Table 8. The performance shows a similar trend to the ANN-based TA, where the best performance is found for the $H = 0.5$ seconds prediction horizon. The CVM consistently performs worse for all prediction horizons except for 0.2s.

Table 8: Results of the CVM TA performance over different horizons

Horizon	TP	TN	FP	FN	TPR	FPR	Accuracy
0.2	617	4589	195	167	0.787	0.0408	0.935
0.5	**711**	**4598**	**186**	**73**	**0.907**	**0.0389**	**0.953**
0.75	682	4452	332	102	0.870	0.0694	0.922
1.0	642	4195	589	142	0.819	0.123	0.869
1.25	593	3864	921	191	0.756	0.192	0.800
1.5	518	3493	1293	266	0.661	0.270	0.720
1.75	452	3105	1683	332	0.577	0.351	0.638
2	394	2785	2007	390	0.503	0.419	0.570

5.2.1 Constant Velocity Model-Based versus the Artificial Neural Network-Based Threat Assessor Benchmark

The performance of the CVM and the performance of the ANN is visualized and compared in the following figure.

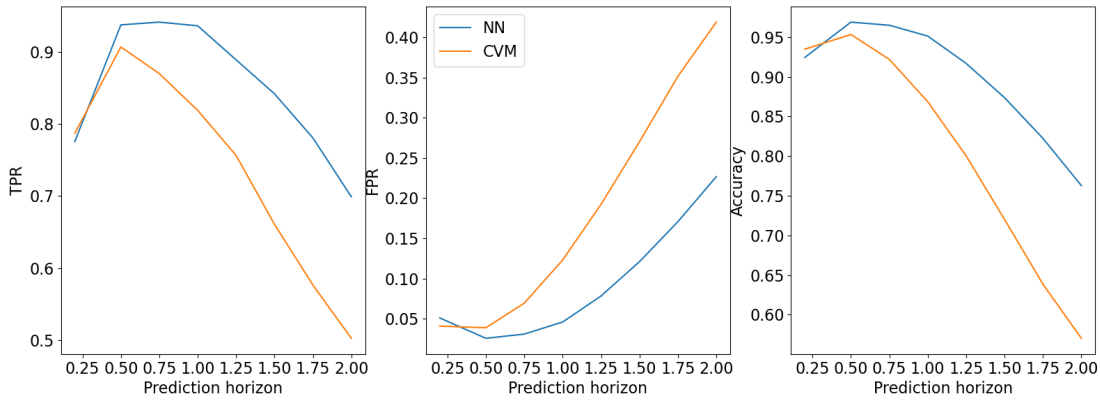

Figure 7: Comparison of the TA performance of the CVM and the ANN in terms of FPR, TPR and accuracy.

5.3 Operational Design Domain Evaluation and Failure Mode Assessment

The results of the ODD investigation using failure modes and success modes with categories of different granularities are presented in Figures 8, 9 and 10.

Figure 8 illustrates the proportional distribution of each of the failure modes and their belonging sub-sub-categories. It is evident from the figure that the two largest sub-sub-categories are active and passive driving behaviour, indicating that driving behaviour is the largest source of error for the system performance within the ODD. A relatively small number of failure modes were attributed to the sensing category and the lane tracking sub-category. These two categories refers to scenarios in which the sensor estimated lane marker polynomial signal were disturbed due to external factors that consequently affects the DM logic. Such external factors could for example be poor weather conditions or uneven terrain. A hypothesis could be drawn here: either the ANN TA was robust against poor lane marker polynomial signals, or the Event Seeker block efficiently omitted all segments with polynomials of poor quality. The hypothesis could have been tested by running the ANN TA on scenarios from non-filtered data segments, but that was not carried out for this thesis. Also, only 5.5 % of the failures were attributed to model tuning. These failures are reflected by the ANN TA triggering a few time step before the start of the acceptance time window. In this thesis, model tuning showed to be a minor problem. However, if wished to be handled, tuning problems can be addressed by further tuning τ until the ANN TA provides desired results. However, by tuning τ, the number of failure modes may increase in other categories.

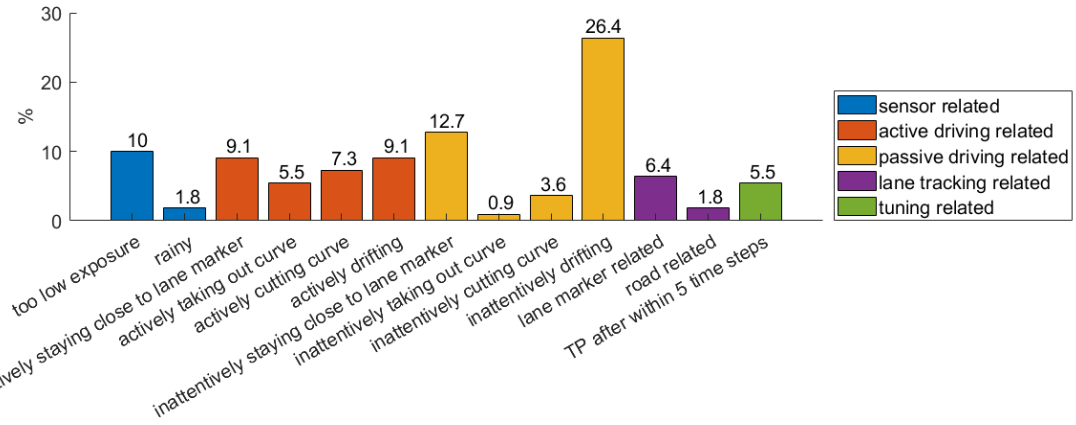

Figure 8: Histogram illustrating the failure modes distribution for FP scenarios.

Figure 9 further illustrates the proportional distribution of FP scenarios of the driving behaviour sub-sub-categories and their failure modes. The most frequent failure mode sub-sub-category were *drifting* and *staying close to the lane marker* for both active and passive driving behaviour sub-categories. *Staying close to the lane marker* refers to behaviour that continues for over 1 second, where 1 second was defined according to the depth of the chosen offset pattern. Intuitively and evidently, both *drifting* and *staying close to the lane marker* is driving behaviour that could result in a lane departure. Taking *staying close to the lane marker* as an example, as all lane departures are naturally preceded by a vehicle placement close to the lane marker, a vehicle positioned close but within the lane marker border would be difficult for the ANN TA to handle. In order to increase the network's confidence in predicting true lane departures, a more robust decision making algorithm could be implemented. For example, the ANN TA could be designed to trigger if and only if three

lane departures are detected within three consecutive time steps. However, this solution would theoretically provide more FNs.

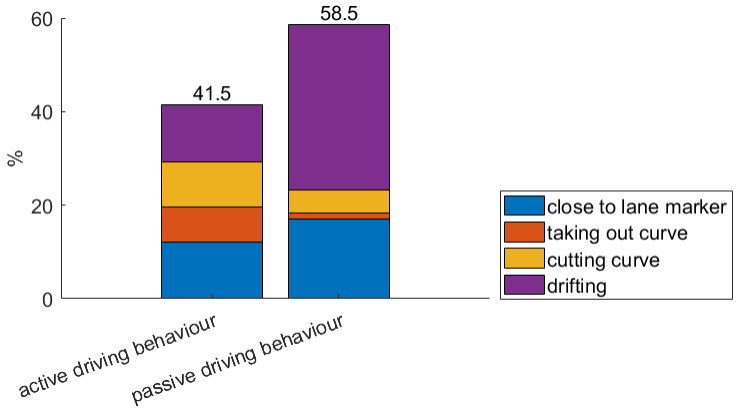

Figure 9: Histogram illustrating the distribution of active and passive driving behaviour sub-sub-categories and their failure modes for FP scenarios.

Figure 10 presents the proportional distribution of the driving behavior success modes for TP scenarios. Here, *planned maneuver* refers to maneuvers that are well planned due to a foreseeable threat, for example observing a truck in the opposite lane and therefore creating a safe margin between the ego-vehicle and the truck. *Corrective maneuver* refers to maneuvers where the driver is forced to act quickly, for example creating a safe margin between the ego-vehicle and a suddenly appearing vehicle. *Inattention* refers to driving behaviour that arises due to low or misplaced focus. By the given definitions, it was obvious that the active driving sub-sub-category contains all types of maneuvers, while the passive driving behaviour sub-sub-category were all caused by inattention. It can be observed that the TA performance is approximately equal regarding active and passive driving for TP scenarios. For active driving behaviour, planned maneuvers was the clear majority. A hypothesis could be made here: either the network had a stronger capability of handling planned maneuvers, or planned maneuvers were more prevalent in the constructed data set. To investigate this hypothesis more rigorously, a data set containing the same amount of planned and corrective maneuvers should be used.

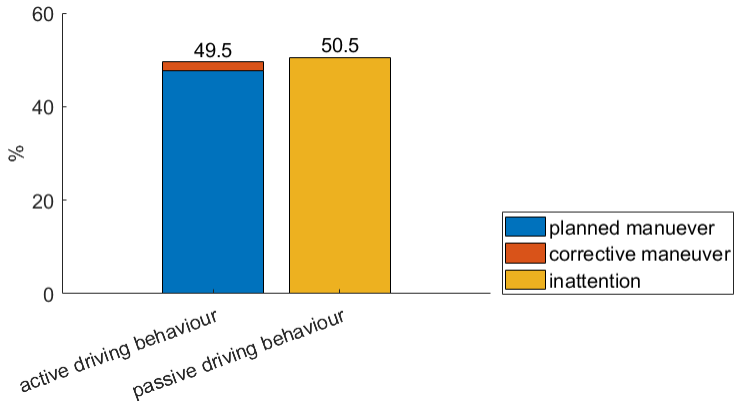

Figure 10: Proportional distribution of active and passive success modes for true positive scenarios. The stacked bar plot shows the percentage of active driving behaviour scenarios versus passive driving behaviour scenarios. Active driving is divided into planned maneuver and corrective maneuver, while all passive driving behaviours are treated as inattentive.

6 Discussion

6.1 Data set

The Event Seeker block described in Section *4* was designed to find unintended lane departures from the raw data set using the 13 selected input signals. However, as can be seen from Figure 10, a substantial amount of failure modes identified from TP scenarios were caused by active driving behavior. In the context of TP scenarios, an active driving behaviour means that the true lane departure occurred from an intentional driving action. An important question thereby becomes: does active driving behaviour which resulted in a TP necessarily mean that the lane departure was made intentionally? The distinction here can be vague, and can be better discussed in the context of a traffic situation. If a driver departs from the lane while giving room to a cyclist, should the lane departure be seen as an intentional lane departure? A counter argument would be that the intention here was to avoid the cyclist rather than departing from the lane, and that this intention then unintentionally led to a lane departure. As explained in Section *4.1.1*, having identified a lane departure event in the raw data sequence, the restriction applied to eliminate intentional lane departure was to use the lane-change-indicator signal. Therefore, in this thesis, unintentional lane departures are defined as all departures that does not arise from a planned lane change. As have been discussed here, this definition might not be sufficient, and additional aspects needs to be considered.

Furthermore, using time series of the 13 given signals as inputs, it is virtually impossible for the ANN TA to discern between lane departures with or without intention for active driving. To increase the network's ability to discern between these two types of scenarios, additional signals may be extracted and used by the Event Seeker block. Alternatively, an object detection algorithm may be used to identify objects in traffic that might be the cause to the lane departure. This technique could potentially reduce the amount of active lane departures in the training data set as

well as the amount of unnecessary and unwanted warnings from an LKA.

6.2 Artificial Neural Network-Based Threat Assessor

6.2.1 Selection of Offset Pattern

As can be observed from Table 5, the selected offset pattern Γ does not have a significant effect on the over all performance of the ANN TA. Even though shorter offset patterns could be argued to provide less computational complexity for the system when being applied in-vehicle, the differences are believed to be insignificant. Therefore, in this project, $\Gamma = [0, 1, 2, 3, 4, 5, 9, 14, 29, 39]$ was determined to be the optimal offset pattern.

6.2.2 Selection of Prediction Horizon

Choosing the optimal prediction horizon H was not as straightforward as choosing architecture and offset pattern. A balance between theoretical prediction accuracy and practicality in reality needs to be considered. As stated above, ANN configurations with $H = 0.5$ showed to outperform other configurations. However, 0.5 seconds may be too short a LKA steering actuation to take place. For an LKA to return the vehicle to the middle of the lane within 0.5 seconds, a relatively high torque to the steering wheel needs to be applied. This may result in an unpleasant driving experience and thus not practical when applied in reality. Moreover, in the context of a warning system, 0.5 seconds is too short for the driver to react to. However, by inspecting the scenarios where the best network (the one trained on 0.5s) fails, and assessing the underlying failure modes, one can conclude which failure modes are the most difficult for the networks to handle. Moreover, the source of error for a network trained on 2 seconds and one trained on 0.5 seconds is the same. A segment classified as a false positive for a network trained on 0.5 s is almost always preceded by a false positive attained by the networks with longer prediction horizons. In other words, if the best network fails, the other networks will with high certainty also fail given the same scenario. This could be verified by comparing the identified FP/FN scenarios on networks sharing the same configuration except H. Segments classified as FP/FN for ANN with $H = 0.5s$ was almost always included in the list of FP/FN found for $H = 2s$.

6.2.3 Constant Velocity Model Based and the Artificial Neural Network Based Threat Assessor Performance Comparison

For the prediction horizons that are meaningful (1s to 2s) the CVM performs substantially worse than the ANN. There is a peculiarity however in the performance of the CVM over different prediction horizons. Namely the difference between 0.2 seconds and 0.5 seconds prediction horizons. The intuitive notion that a shorter prediction horizon should yield better accuracy is not valid looking at the results. The accuracy of the CVM for 0.2s prediction horizon is 0.935, and for 0.5s prediction horizon the accuracy is 0.953, which can be seen in Table *8*. One possible explanation for this phenomenon is that for the 0.2s prediction horizon, the acceptance window is much smaller than for the 0.5s prediction horizon. Their respective windows are 16 and 80 time steps long. This means that the room for error is also much smaller the 0.2s prediction horizon. This phenomenon is visualized in Figure 11.

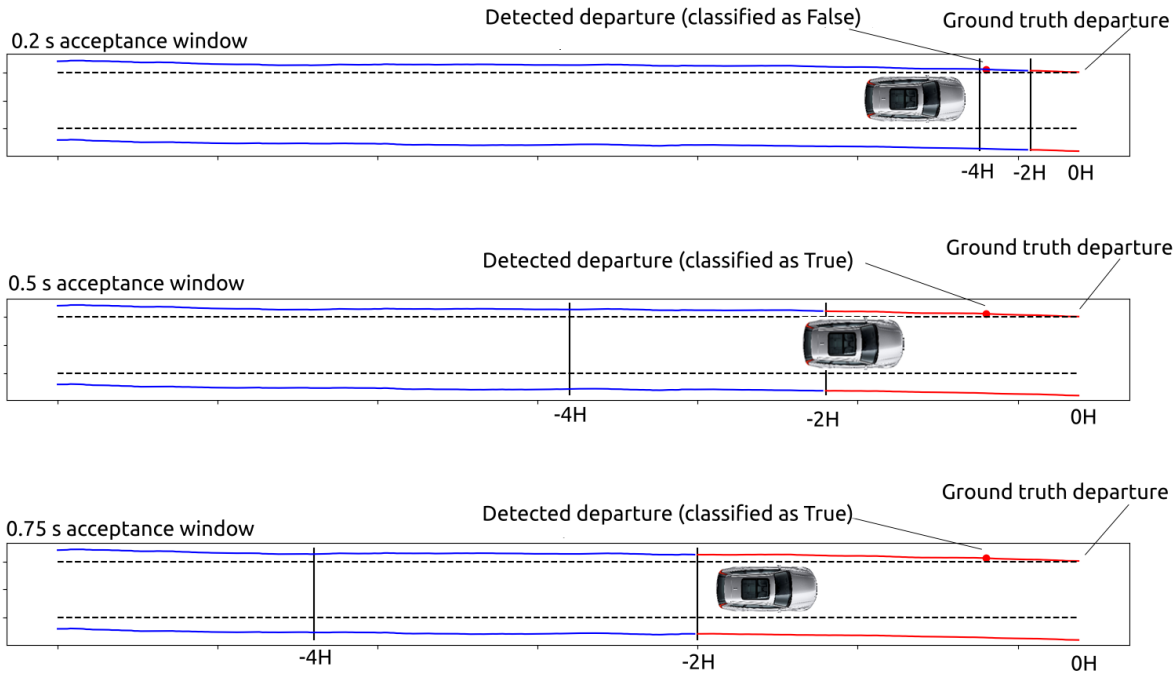

Figure 11: Visualization of the effect of increasing the prediction horizon. A departure detected at the red circle in the image may be classified as a true or false positive depending on the normal and departure driving window defined by the used prediction horizon.

The drop in accuracy for the CVM from 0.5s prediction to 2s prediction horizon is expected however. The assumption that the longitudinal and lateral velocity is constant during the whole prediction horizon becomes less and less valid the longer the prediction horizon.

The ANN shows the same trends, the accuracy is low for 0.2s prediction horizon, and increases for 0.5s, and then drops monotonically for longer prediction horizons. The reason for the low accuracy for 0.2s is believed to be the same as for the CVM, that the acceptance window is much smaller. Furthermore, two interesting observations can be made. Firstly, the accuracy drops at a higher rate for the CVM than for the ANN. Secondly, the accuracy of the CVM for the 0.2s prediction horizon is better than the ANN of the same prediction horizon, hinting that the neural network might be unnecessarily complex for such short prediction horizons. The better performance of the CVM for a 0.2s prediction horizon suggests that the assumption that the longitudinal and lateral velocity stays constant over 0.2 seconds is acceptable.

6.3 Operational Design Domain Evaluation and Failure Mode Assessment

This section will discuss the ODD of the best ANN TA. Section *6.3.1* will motivate the selection of the chosen failure and success modes and their categories, sub-categories and sub-sub categories.

Section *6.3.2* will go into depth about how identified failure and success modes can provide insight into how the ANN TA performance may be improved.

An important factor here is that all segments used for building scenarios on which the ODD was evaluated come from the raw data processed using the Event Seeker block. Therefore, the conditions applied when truncating sequences as described in Section *4.1.1* become the ANN TA's ODD. An example of a constraint applied to the Event Seeker block that becomes a part of the ODD is that the vehicle velocity needs to be greater than $60km/h$ for optimal ANN TA performance.

6.3.1 Active and Passive Driving Behaviour

The failure modes that were chosen to be investigated for the ANN TA, presented in Table 2, are modified failure modes used by Zenseact. The idea behind the construction of Table 2 was to categorize failure modes into categories with proper granularity to capture the observed failure scenarios on different levels as accurately as possible. *Luminosity*, *lane marker* and *road conditions* are classical sub-categories for ADAS failure modes used in the industry. In comparison, the introduction of *active driving behaviour* and *passive driving behaviour* as failure mode sub-sub categories may not be intuitive. A motivation for failure mode assessment when defining an ADAS ODD is to obtain insights about the limitations and capabilities of the system, and how it may be improved. Thus, the failure modes need to be be descriptive. Active and passive driving behaviours are descriptive failure mode sub-categories in the sense that they can target system short-comings which can't be improved by purely enhancing the system's hardware.

Active and passive driving behaviours are identified by inferring the driver's intention behind a TA triggering action. Here, a TA triggering action refers to the time evolution of data signals that lead up to the event time instance. The time period of a triggering action is manually determined for each given scenario, and varies from scenario to scenario. By placing the triggering action in a visual traffic context, the intention behind the TA triggering action can be determined.
For FP scenarios, active driving behaviour refers to intentional TA triggering actions that are not true lane departures. Examples of such actions are among others: staying close to the lane marker to create a safe margin between the ego-vehicle and passing by vehicles, vehicle placement due to oncoming vehicles in the opposite lane, and cutting a curve for a shorter trajectory during safe circumstances. In contrast, passive driving behaviour refers to all TA triggering actions that arise from inattentiveness. These are usually indicated by unstable driving trajectories or actions carried out without any discernible reason.

Having investigated FP scenarios in which the ANN TA erroneously dealt with the driver's intention, it was also of interest to validate TP scenarios with respect to driver's intention. By inspecting active and passive driving behaviour for the FP and TP scenarios together, a general idea of the ANN TA performance regarding driving behaviour can be obtained.

6.3.2 Operational Design Domain Evaluation Results

From the results shown in Figure 2 and Figure 9, it is clear that most failure scenarios are related to driving behaviours, of which passive driving behaviour has the largest contribution. It can thereby be stated that rather than hardware issues such as lane marker tracking, the largest source of error for the ANN TA arises from interpreting the driver's intention. However, as mentioned in

Section *4.2.4*, segments from $\mathbb{D}$ and $\mathbb{N}$ are segments that does not contain any anomalies in the signals. Anomalies are often related to hardware issues, such as worse sensor performance due to night driving or poor weather conditions. Therefore, the data set used for this thesis is not optimal for identifying sensor related failure modes.

By combining results from Table 9 and Table 10 a small difference in the ANN TA's performance for active and passive driving behaviours can be observed. The difference is too small to draw any enlightening conclusions from however. Another observation here is the unexpectedly high ratio of active drivings behaviours found within the TP scenarios. As described in the project scope in Section *2*, the thesis aimed to target the detection of unintended lane departures. This was considered in Section *4.1.1* when specifying the conditions which the data set segments must satisfy, being no lane change for 3 seconds after the detected lane departure event. It is evident that this restriction does not exclude all intentional lane departures made by the drivers.

7 Conclusions and Future Perspectives

In this thesis, an ANN-based threat assessment model (denoted as ANN TA) was developed to predict unintended lane departures. The proposed ANN TA was summarized according to every step in the guidance for an ADAS system development mentioned in Section *1*.

Step 1: *"Identification of the concept ADAS"*. The ANN TA consisted of a three hidden layer artificial neural network with 128 neurons in each layer and a decision making algorithm. Using an offset pattern $[0, 1, 2, 3, 4, 5, 9, 14, 20, 39]$, the ANN takes in sparsely sampled historical time series data from 13 selected signals and outputs the predicted lateral distance to the left and right side lane markers Hs ahead. The decision making algorithm then bases its decision on the ANN output and a defined threshold, corresponding to how much the vehicle side-edge is allowed to overlap the sensor estimated lane markers, to determine whether an unintended lane departure is going to take place.

Step 2: *"Identify attributes that define the operational design domain(ODD)."* The ODD was determined as the vehicle states and driving environments captured in the training data set, as well as the 6 restrictions applied when filtering the data set.

Step 3: *"Identify object and event detection and response capabilities."* The scope of this thesis only covers event detection part. The ANN TA performance was evaluated by dividing the test segments into two windows, a normal driving window and an acceptance time window. The prediction outcome of each segment was classified in terms of true positive, false positive, true negative and false negative, depending on the time instance of the predicted lane departure. The ANN TA performance showed best results at a 0.5s prediction horizon, achieving a TPR=0.937, a FPR=0.0257 and a $Accuracy = 0.969$. For $H \geqslant 0.5s$, the ANN TA outperformed the Constant Velocity Model with respect to FPR, TPR and accuracy.

Step 4: *"Identify and assess failure modes and failure mitigation strategies."* The ANN TA was evaluated using scenarios in the Operational Design Domain, in which the model was designed to function. These scenarios were visualized using a Birdview live plot tool which illustrates the ANN TA outcome, as well as the top camera videos. Failure modes and success modes were found,

together with their respective categories, to identify underlying cause for the model failure. The major cause of failure was determined to be due to from the system's misinterpretation of driving behaviours, among which passive driving behaviour appeared to be the most difficult for the model to interpret. Other failure modes such as lane tracking- or road geometry estimation errors were found but were not as prevalent as the driving behavior failure modes. By combining the simulation with the camera feed of the corresponding traffic situation, it was possible to identify the driver behaviour failure modes. This would be significantly more difficult without the camera feed. Therefore, it's paramount for the failure mode investigation to use other means beyond only simulations, to yield other valuable information about the traffic context.

7.1 Future Work

In this work, only vanilla neural networks were trained and evaluated. Other types of networks which in theory handles time series input data equally well or better exist, for example RNNs and LSTMs. Using these networks as a basis would provide insight in the difference in performance between different type of networks for Lane Keeping Aid systems. Moreover, validation of the defined ODD was limited to inspecting scenarios from pre-collected data. Testing the ANN TA in real-world driving would offer a more valid evaluation of the system's ODD. As the majority of failure modes were identified as driving behaviour related, future work could consider using input signals that reflects the driver intentions. Other tools for discerning driver intention include object detection algorithms and face tracking algorithms which could be used together with a *driver monitoring camera*.

References

[1] World Health Organization. Road traffic injuries, last modified on 7 feb. 2020. `https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries`, 2020.

[2] D. L. Hendricks, M. Freedman, P. L. Zador, and J. C. Fell. The relative frequency of unsafe driving acts in serious traffic crashes. *Contract*, DTNH22-94-C-05020, January 2001.

[3] B. Fildes, M. Keall, N. Bos, A. Lie, Y. Page, C. Pastor, L. Pennisi, P. Thomas, and C. Tingvall. Effectiveness of low speed autonomous emergency braking in real-world rear-end crashes. *Accident Analysis and Prevention. 81, pp.24-29.*, 2015.

[4] M. van Ratingen, A. Williams, A. Lie, A. Seeck, P. Castaing, R. Kolke, R. Kolke, G. Adriaenssens, and A. Miller. The european new car assessment programme: A historical review. *Chinese Journal of Traumatology*, 19, 01 2016.

[5] J. Dahl, G. Rodrigues de Campos, C. Olsson, and J. Fredriksson. Collision avoidance: A literature review on threat-assessment techniques. *IEEE Transactions on Intelligent Vehicles, vol. 4, no. 1, pp. 101–113*, 2019.

[6] L. Staplin, T. Mastromatto, K. H. Lococo, W.Kenneth Gish, and J. O. Brooks. The effects of medical conditions on driving performance (report no. dot hs 812 623. *Washington, DC: National Highway Traffic Safety Administration.*, September 2018.

[7] J. Dahl, R. Jonsson, A. Kollmats, G. Rodrigues de Campos, and J. Fredriksson. Automotive safety: a neural network approach for lane departure detection using real world driving data. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, 2019.

[8] K. Min, D. Kim, J. Park, and K. Huh. Rnn-based path prediction of obstacle vehicles with deep ensemble. *IEEE Trans. on Vehicular Technology, vol. 68, no. 10, pp. 10 252–10 256*, 2019.

[9] J. Dahl, G. Rodrigues de Campos, and J. Fredriksson. A path prediction model based on multiple time series analysis tools used to detect unintended lane departures. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–7, 2020.

[10] J. Dahl, G. Rodrigues de Campos, and J. Fredriksson. Performance and efficiency analysis of a linear learning-based prediction model used for unintended lane-departure detection. *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[11] U.S. Department of Transportation. Roadway departure safety. `https://safety.fhwa.dot.gov/roadway_dept/`, August 2020.

[12] Volvo. Lane keeping aid (lka) - function. `https://www.volvocars.com/en-th/support/manuals/s60/2013w46/driver-support/lane-keeping-aid/lane-keeping-aid-lka---function`, July 2018.

[13] H. Farah, S. Bhusari, P. van Gent, F. A. M. Babu, P. Morsink, R. Happee, and B. van Arem. An empirical analysis to assess the operational design domain of lane keeping system equipped vehicles combining objective and subjective risk measures. *IEEE Transactions on Intelligent Transportation Systems*, 22(5):2589–2598, 2021.

[14] W. Wang, D. Zhao, W. Han, and J. Xi. A learning-based approach for lane departure warning systems with a personalized driver model. *IEEE Transactions on Vehicular Technology*, 67(10):9145–9157, 2018.

[15] D. Bezzina and J. Sayer. Safety pilot model deployment: Test conductorteam report hs-812 171. *National Highway Traffic Safety Administration*, 2015.

[16] A. A. Albousefi, H. Ying, D. Filev, F. Syed, K. O. Prakah-Asante, F. Tseng, and H. Yang. A support vector machine approach to unintentional vehicle lane departure prediction. In *2014 IEEE Intelligent Vehicles Symposium Proceedings*, pages 299–303, 2014.

[17] J. M. Ambarak, H. Ying, F. Syed, and D. Filev. A neural network for predicting unintentional lane departures. In *2017 IEEE International Conference on Industrial Technology (ICIT)*, pages 492–497, 2017.

[18] P. Cízek and S. Sadikoglu. Robust nonparametric regression: A review. *Wiley Interdisciplinary Reviews: Computational Statistics*, 12, 2020.

[19] X. Shaoming. Comparative models in customer base analysis: Parametric model and observation-driven model. *Journal of Business Economics and Management*, 21:1731–1751, 10 2020.

[20] D. Masters and C. Luschi. Revisiting small batch training for deep neural networks, 2018.

[21] L. Mu, Z. Tong, C. Yuqiang, and A. J. Smola. Efficient mini-batch training for stochastic optimization. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, page 661–670, New York, NY, USA, 2014. Association for Computing Machinery.

[22] X. Ying. An overview of overfitting and its solutions. *Journal of Physics: Conference Series*, 1168:022022, feb 2019.

[23] A. F. Agarap. Deep learning using rectified linear units (relu), 2019.

[24] J. Larsson and M. Sjöstedt. A lane departure detection system basewd on uncertainty aware machine learning, 2020.

[25] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2014. cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

A Failure Modes Table with Description

Failure mode	**Description**
Low exposure	Tunnel or night
Rain	Low vision range
Arrow in lane	Arrow tracked instead of lane marker
Unstable double tracking	Unstable tracking due to two adjacent lane markers
Type change	Tracking disturbance due to lane marker type change
Rumble strip	Intentionally uneven lane marker for vibration
Crest, slope or bump	Bad vision range and tracking due to uneven terrain
Close to lane marker	Placement close to the lane marker for over 1s to create a safe margin between ego-vehicle and threat
Taking out curve	Driving on the outer lane marker of a curve to create a safe margin between ego-vehicle and threat
Cutting curve	Driving on the inner lane marker of a curve to create a safe margin between ego-vehicle and threat
Drifting	Driving towards the lane marker to create margin
Preparing for lane change	Waiting for safe lane change close to the lane marker
Close to lane marker	Placement due to inattention
Taking out curve	
Cutting curve	
Drifting	
TP followed by FP	The system predicts a departure too early
Success mode	
Planned maneuver	Maneuver carried out with time margin to avoid threats
Corrective maneuver	Rapid maneuver due to sudden appearing threats
Inattention	Driving behaviour due to inattention