Baden-Wuerttemberg Cooperative State University Mannheim

Project Report Integration Seminar

Traffic Sign Detection and Recognition

Course of Study Business Informatics

Field of Study Data Science

Author:	Max Bernauer, Philipp Dingfelder, Valentin Moritz Müller
Matriculation Number:	5763624, 8687786, 4616344
Company:	SAP SE, Schaeffler Technologies AG & Co. KG
Course:	WWI20DSB
Director of Studies	Prof. Dr. Bernhard Drabant
Academic Supervisor:	Prof. Dr. Bernhard Drabant
Completion Period:	14.11.2022 – 10.02.2023

Statutory Declaration

We herewith declare that we have composed the present thesis with the title "*Traffic Sign Detection and Recognition*" ourselves and without use of any other than the cited sources and aids. Sentences or parts of sentences quoted literally are marked as such; other references with regard to the statement and scope are indicated by full details of the publications concerned.

The thesis in the same or similar form has not been submitted to any examination body and has not been published.

Place, Date

Max Bernauer, Philipp Dingfelder, Valentin Moritz Müller

Contents

List of Figures

List of Tables

List of Acronyms

ADAS	advanced driver assistance systems
ADS	automatic driving systems
AI	artificial intelligence
CNN	convolutional neural network
CV	computer vision
GTSDB	German Traffic Sign Detection Benchmark
GTSRB	German Traffic Sign Recognition Benchmark
IC	image classification
mAP	mean Average Precision
ML	machine learning
NN	(artificial) neural network
OD	object detection
ResNet	residual neural network
TSR	Traffic Sign Recognition
VGG	Visual Geometry Group
YOLO	you only look once

1 Introduction

This project paper was created as part of the Integrationsseminar lecture at the Cooperative State University Baden-Wuerttemberg Mannheim and describes an two step approach to implement *Traffic Sign Recognition* (TSR) as a part of autonomous driving.

1.1 Motivation

Traffic signs are a fundamental part of our daily lives as they control the traffic through which we move every day. They provide various key insights like information about the condition of the road ahead and what to expect in terms of other road users like cyclists or children. (Chen et al., 2021, pp. 2233–2235) Although traffic signs provide a lot of important information for drivers, there are still many tragic accidents worldwide. According to the Waymo safety report, in 2016 1.35 million lives were lost trough traffic crashes worldwide. (Aslansefat et al., 2021, pp. 66–68) A better understanding of the reasons behind those crashes can be derived in Figure 1.1 created by the Federal statistical office of Germany (Statistisches Bundesamt).

The chart shows that six of the top nine reasons for driver-related accidents that cause personal injury are connected with with a disregard of the traffic signs. Many of the accidents simply happen because drivers overlook the traffic signs or ignore them. To counteract this, *advanced driver assistance systems* (ADAS) as well as *automatic driving systems* (ADS) are being developed. Those systems try to resolve this problems by leveraging the capabilities of *artificial intelligence* (AI), especially those of *computer vision* (CV) (Chen et al., 2021, pp. 2233–2235). Systems like ADAS and ADS are part of autonomous driving. To enable vehicles to drive autonomously, they need to communicate with other parts of the road traffic like other vehicles and roadside infrastructure (traffic sings, traffic light systems) (Aslansefat et al., 2021, pp. 66–68). To make this communication available, various components are needed. Those components include hardware (long distance radar systems, advanced sensors) and software (image recognition, real-time traffic data processing). To achieve progress in these areas and to meet the future demand for those systems automotive companies like Audi, BMW and Tesla as well as technology companies like Google are investing heavily. A prediction that shows the potential market for autonomous

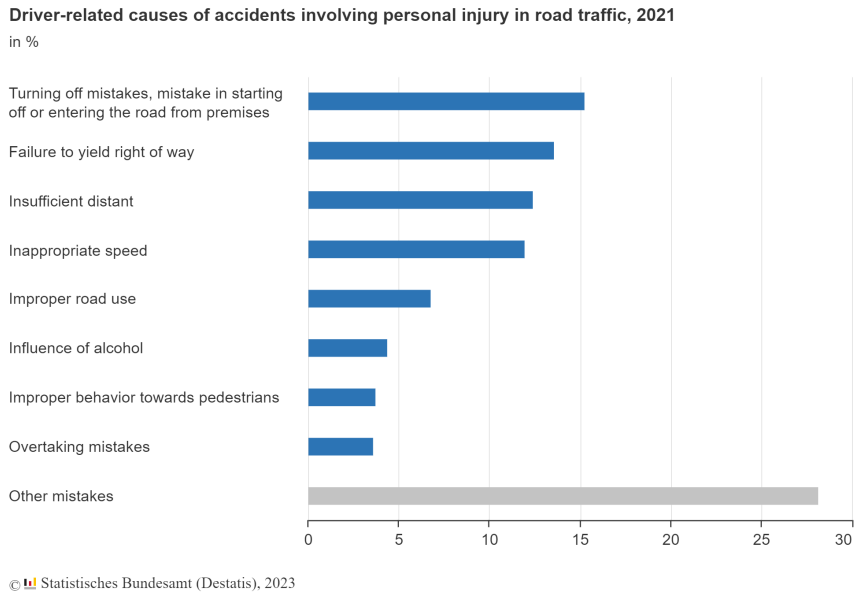

Figure 1.1: Driver-related causes of accidents involving personal injury in road traffic (Statistisches Bundesamt, 2022)

vehicles is that in 2025 20% of all cars sold in the US are self-driving (Kanagaraj et al., 2021, pp. 1011–1019). The points listed above underline the importance and relevance of autonomous driving systems in the future. As we are data science students, we focused on the software and modeling part of autonomous driving and tried to create an approach for detecting and classifying traffic signs.

1.2 Research Objectives

Regarding the focus on the software part of autonomous driving, the goal of this project was to implement a model that supports drivers by providing information about the traffic signs along the road. Thus to implemented a model that performs TSR to display the traffic signs crossed while driving in the instrument panel of the car and and remind the driver to obey the traffic rules. To reach this goal, a two step approach for performing TSR will be implemented. The approach is based on the combination of a model for detecting the traffic signs and a model that classifies the detected traffic signs. Detection and classification by the models is done by leverage the capabilities of CV. Therefore, in the following, the theoretical foundation for understanding the approach is formed by

giving an first overview about CV in general and how it is performed. Afterwards particular attention is paid to the sub-areas image classification and object detection as they are the main concepts behind the approach. To explain the procedure while implementing and combining the models the data basis will be discussed first. To give an overview about the data images of the recognition and the detection set are shown and the composition of the data sets is explained. Following on, the preprocessing steps are pointed out to show how the data basis was expanded trough data augmentation before training the models. Thereafter an overview about the two sides of the combined model is given before the combination is shown. The overview of the two sides includes the architecture of the models as well as an evaluation of their performance and the comparison with other models/approaches. To sum the learnings and the challenges of this project up, this report is finished by a conclusion. The conclusion firstly summarizes the procedure and the findings while implementing the two-step approach before the results of the project are reflected critically. During the critical reflection the challenges of the projects and limits are shown as well as an outlook to show what could be implemented additionally.

2 Theoretical Background

This chapter aims to deliver an introduction to computer as well as machine vision in general, and later to give an overview on the models and algorithms that are used to solve the problems posed by the research objectives this report is based on.

2.1 Introduction of Computer Vision

The field of CV is an important part of AI that falls in artificial vision, the discipline that deals with the extraction of information from images. It is distinct from the related field of machine vision. Both focus on artificial vision, but While CV focuses on processing and analyzing the images themselves, and thus mainly factors related to mathematics and computer science, machine vision concentrates on the practical implementation of systems capable of artificial vision and, because of that, requires the application of knowledge stemming from physics and engineering into account. In a real-world application, knowledge from both fields would need to be applied in conjunction to solve the task of traffic sign detection and recognition (Batchelor, 2012, pp. 10–18; Gad et al., 2018, pp. xvii–xix). Because of the limited time frame and thus scope of this project, this report and the following theoretical background focus solely on the CV aspects of the task of traffic sign recognition and detection.

From a CV perspective, the problem of traffic sign recognition and detection can be divided in two sub-problems: An *object detection* (OD) problem to identify whether a traffic sign is included in an image, and if so, where, and an *image classification* (IC) problem to identify the correct traffic sign. While many OD methods are able to conduct a classification, both tasks are separated in this project, because far more data is available in the German Traffic Sign Recognition Benchmark (GTSRB) data set, which is suitable for IC only, than in the German Traffic Sign Detection Benchmark (GTSDB) data set. Even though chronologically the OD needs to be done first, the IC is dealt with first in this report, because it is generally a less complex task and OD techniques usually build on the ones from IC (Wang et al., 2022, pp. 1–2; Gad et al., 2018, p. 188; Sermanet & LeCun, 2011, p. 2809; Wei et al., 2018, pp. 5884–5885).

A type of machine learning (ML) model that is frequently utilized in CV for a variety of tasks is a *convolutional neural network* (CNN). CNNs come in many different variations and are a type of *(artificial) neural network* (NN). NNs are a type of ML algorithm inspired by the human brain and applicable to a variety of problems. They consist of multiple layers with nodes or neurons. In their basic form they are fully connected, meaning each node in every layer is connected with every node in the next one. This, however, results in a large number of parameters and is not the case for CNNs. This type of NN utilizes eponymous convolutions, who are able to transform an input-tensor to a feature map, and filters as well as pooling. Each neuron in a convolutional layer only processes inputs in its receptive field, the size of which is determined by the filter. Pooling combines the output of a receptive field to its average value in *average pooling* or maximum value in *max pooling*. This is useful in dimensionality reduction. Fully connected layers are usually only used in special cases, for example for a classification of the images after they have been processed by convolutional layers. These properties lower the number of parameters and thus raise efficiency of training and predictions. Furthermore, tensors being a natural fit to represent image data makes them suitable for CV tasks (Gad et al., 2018, pp. 45, 183; Sermanet & LeCun, 2011, pp. 2809–2810; Wei et al., 2018, p. 5885; Simonyan & Zisserman, 2014, pp. 2–3). In the next two sections, different CNN models suitable for image classification and object detection and later used in the implementations in chapter 3 are described.

2.2 Image Classification Models

For the IC problem, two types of models are described and selected. The first was introduced by Oxford University's *Visual Geometry Group* (VGG) in 2014 and thus is usually called the VGG architecture. This configuration of CNN focuses on increased depth (at least compared to other models that were utilized at the time). A VGG model uses multiple convolutional blocks, each using one to four convolutional layers with a relatively small receptive field size of 3 x 3, followed by a max pooling layer. Five such blocks are followed by three large fully connected layers to carry out the classification. Figure 2.1 illustrates the architecture (Simonyan & Zisserman, 2014, pp. 1–3).

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224×224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figure 2.1: Overview of the different configurations of VGG models proposed in the original paper introducing the architecture
(Simonyan & Zisserman, 2014, p. 3)

Simonyan and Zisserman proposed and described different types of VGG networks, which are depicted in Figure 2.1. They are named after the number of weight layers, for example the model with 16 weight layers is commonly referred to as VGG-16 and range from 11 to 19 weight layers. This architecture shows good performance on image classification tasks and can be applied to a wide array of problems (Simonyan & Zisserman, 2014, pp. 3–14; He et al., 2016, pp. 772–773).

The other approach described in this section is the use of a *residual neural network* (ResNet). Very deep NNs are susceptible to the problem of vanishing or exploding gradi-

ents, where gradients become very small or very large, leading to the weights of the NN barely or rapidly changing and thus hampering training. Another problem is degradation, which causes the accuracy of deep classification models to saturate and then to quickly decline. ResNets are CNNs that solve this by the introduction of *residual learning*, which enables information flow across more than one layer by utilizing so-called *shortcut or skip connections* connected not to the next, but a later layer and performing identity mapping instead of a nonlinear activation (He et al., 2016, pp. 770–773).

This allows for the construction of much deeper NNs compared to networks without residual learning. Different ResNet variations exist and are named after the number of layers they consist of, for example ResNet50 or ResNet101. Such networks are able to deliver good performance, especially on complex problems and at the time of their introduction were able to outperform state-of-the-art methods on *ImageNet*, a database of images in 1000 different classes (He et al., 2016, pp. 774–777).

2.3 Object Detection Models

While different approaches to OD exist, many of them are repurposed classifiers, for example by evaluating various sections of the image to detect which sections matches the target class best. A more advanced approach is called *you only look once* (YOLO). As the name suggests, YOLO solves the object detection by a single look on the entire image, and thus is extremely fast and well-suited to real time problems like traffic sign detection. Furthermore, since the whole image is taken into account, contextual information can be learned and utilized for predictions (Redmon et al., 2016, pp. 779–780). Because of these advantages, this section focuses on YOLO models.

YOLO divides the input image into a n x n grid, with each cell formed by the grid being responsible for the detection of objects whose center is located inside it. For each object, a bounding box is predicted in form of its x and y coordinates, width and height as well as confidence. Multiple bounding boxes per grid cell are possible. In later implementations, an additional parameter indicating the class is returned. Combined with the bounding boxes, for each grid cell, a conditional class probability is predicted. Both predictions are then unified, as shown in Figure 2.2. Like all other models discussed, YOLO models are based on CNNs, but a difference to the IC models lies in the fact that YOLO outputs a tensor containing the predictions (Redmon et al., 2016, pp. 780–781; Wang et al., 2022).

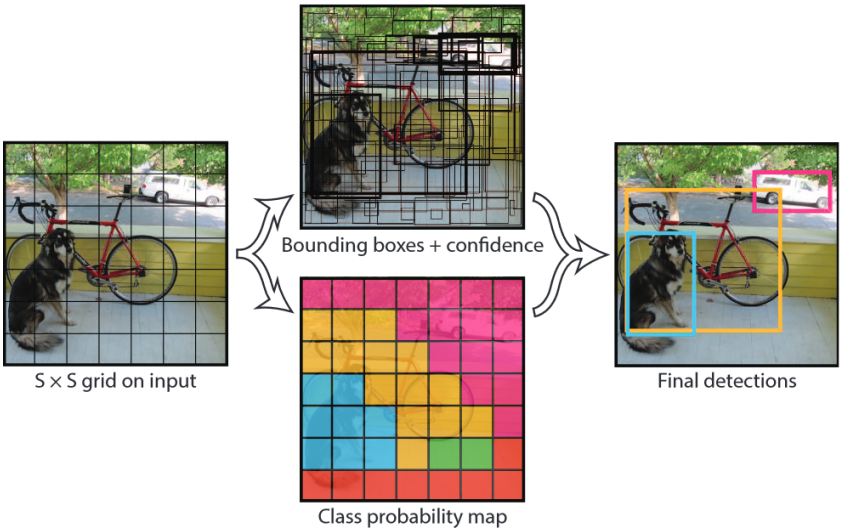

Figure 2.2: Explanation of the unification of bounding boxes and class probabilities in YOLO models
(Simonyan & Zisserman, 2014, p. 3)

Initially, YOLO, while faster, suffered from accuracy issues that placed it behind slower systems from a pure quality of predictions perspective. Due to its many desireable properties, YOLO has since been further developed, with new versions reaching as far as YOLOv7 in 2022. Each version introduced improvements, and many of the newer versions, including YOLOv7, are able to achive state-of-the-art results (Wang et al., 2022, pp. 1–2, 9; Redmon et al., 2016, p. 780).

3 Implementation

This chapter aims to show the process of implementing a two-step approach for TSR. It starts with an overview of the data sets for classification and detection and their preprocessing in section 3.1. Afterwards the details of the model implementations are shown as well as how they were combined to successfully extract and present the information from the traffic signs.

3.1 Data Preparation

For training the models two different data sets were used, as the tasks which need to be performed are slightly different. For the classification part the GTSRB which was provided at the International Joint Conference on Neural Networks in 2011 was used. The task addressed by the GTSRB data set is a single-image, multi-class classification problem as there is one input image and there can be various traffic signs in the image. In the data set there are more than 40 classes, reflecting all traffic signs that can be found on german streets. In total there are more than 50.000 images in the data set. The distribution of the images in the classes is displayed in Figure 3.1.

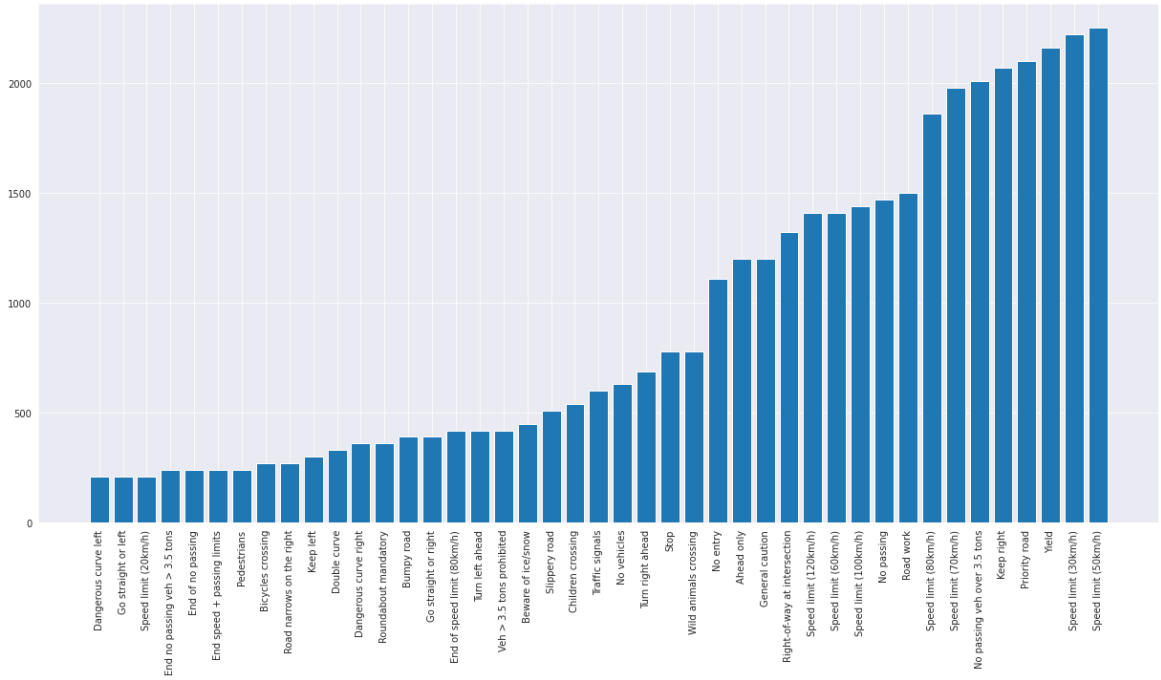

Figure 3.1: Class Distribution GTSRB
Source: (Own illustration)

The chart shows that the classes very imbalanced and there are, for example, a lot of pictures of speed limits and comparatively few concerning bicycles or pedestrians. While training this imbalance could be considered by the model trough setting class weights, however this imbalance did not show any negative effect when training and evaluating the performance of the model. The images of the GTSRB data set consists of 39,209 images for training and 12,630 images for testing, which are both resized into 30 x 30 pixel RGB channel images. The focus of the images for classification is on the traffic sign itself as it fills the biggest part of the picture. On the other hand on the images for traffic sign detection which are from GTSDB the traffic signs fill a smaller part of the picture which can be seen in Figure 3.2.

(a) GTSRB Images (b) GTSDB Image

Figure 3.2: Images of the data sets
(Own Illustration)

Another difference between the two data sets, besides the different tasks they supply, is that the GTSDB data set is much smaller as it contains only 506 images in total, 406 for training and 50 each for validation and testing. The tasks behind the GTSDB data set is a single-image detection problem as the model that handles it takes one image and provides bounding boxes for the detected traffic signs as output. In order to achieve good results despite this small number of images, transfer learning utilizing the seventh version of YOLO was used. Its implementation will be explained more detailed in section 3.3 later in this chapter.

To increase the data basis for the classification part several augmentation steps were performed. Through the augmentation steps the images from the GTSRB data set were modified in various ways to get more perspectives to boost the number of training examples. For performing the augmentation and getting the new perspectives the random_rotation, random_zoom, random_crop and random_translation layers of TensorFlow were used. The flip layer was not used, as it would reverse the meaning of traffic signs/labels such as the get right sign. The images would not fit the labels anymore which in turn would confuse the model during training. However, the other augmentation layers could be used without hesitation. For the training data of the detection part augmentation was not used as modifying the images would cause a mismatch between the coordinates of the traffic signs and the predefined bounding boxes. The position of the traffic sign simply changes

trough the augmentation and the predefined bounding boxes would therefore have had to be adjusted with additional effort for each image, and thus exceeds the scope of this project.

In the following, the implementation of the models for classifying and detecting the traffic signs will be discussed more detailed.

3.2 Implementation of Classification Models

For the IC task of traffic sign recognition, two CNN models are trained. The first one is based on a simplified VGG-16 architecture, but matches neither VGG-11 nor VGG-13. Because of that, it will be referred to as the *VGG-like model* in this paper and the evaluations. The other model uses the ResNet50 architecture. Both models are implemented with the library *TensorFlow* and utilize the same pre-processing and methodology of evaluation, and both contain the same data augmentation layers discussed in section 3.1, so their implementation only differs in regard to the model architecture.

The implemented VGG-like model uses only three convolutional blocks and much simpler fully connected layers in the end. Because the images in the GTSRB are small and thus scaled 30 x 30, and the original VGG architecture is intended for 224 x 224 images and the 1000 classes of ImageNet instead of the 43 classes in our data (Simonyan & Zisserman, 2014, p. 2), the use of a much simpler model is required to avoid overfitting caused by unreasonable model complexity. A simpler model also allows for faster training and faster inference in the final application.

The model is trained using an Adam optimizer and sparse categorical cross-entropy loss, which is suitable for multi-class classification. During training, the performance on the validation data set is monitored and triggers early stopping if no improvement to the loss occurs after 10 epochs. This prevents overfitting. The progress is displayed in Figure 3.3 and shows a very consistent performance across training and validation data.

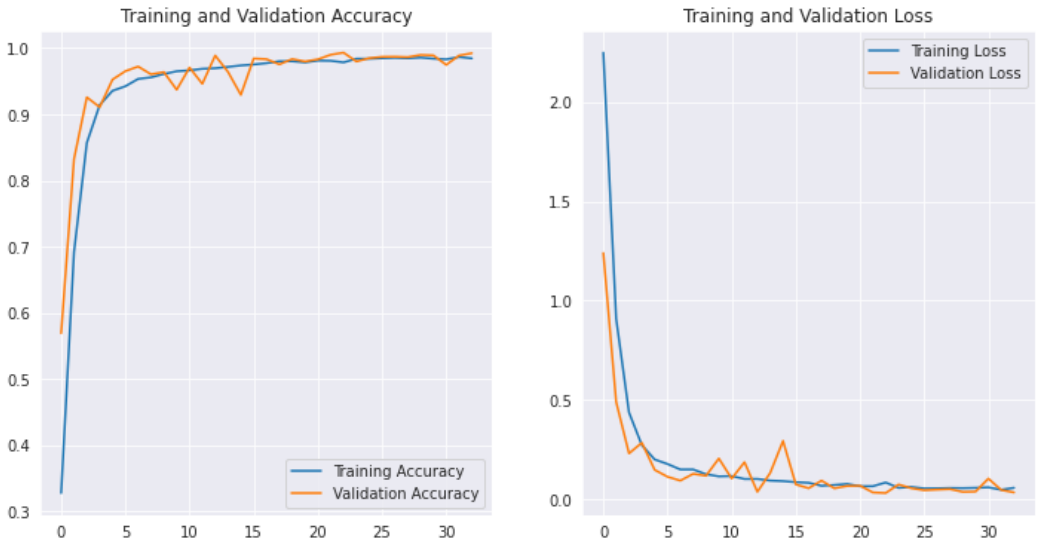

Figure 3.3: History of loss and accuracy for the VGG-like model (Own Illustration)

Later, an evaluation is conducted. The accuracy is determined on all sets, as shown in Table 3.1, predictions are made for all images in the test set, and additionally, precision, recall and F1-score are determined and illustrated in Table 3.2. From the predictions, confusion matrices are calculated and, due to the relatively high number of classes, filtered for labels which show issues. One such matrix is depicted in Figure 3.4, showing only classes with incorrect classifications of more than 1% of examples as one specific other traffic sign. Interestingly, most confusion arises between similarly looking or related traffic signs, for example the ones referring to vehicles above 3.5 tons or directions.

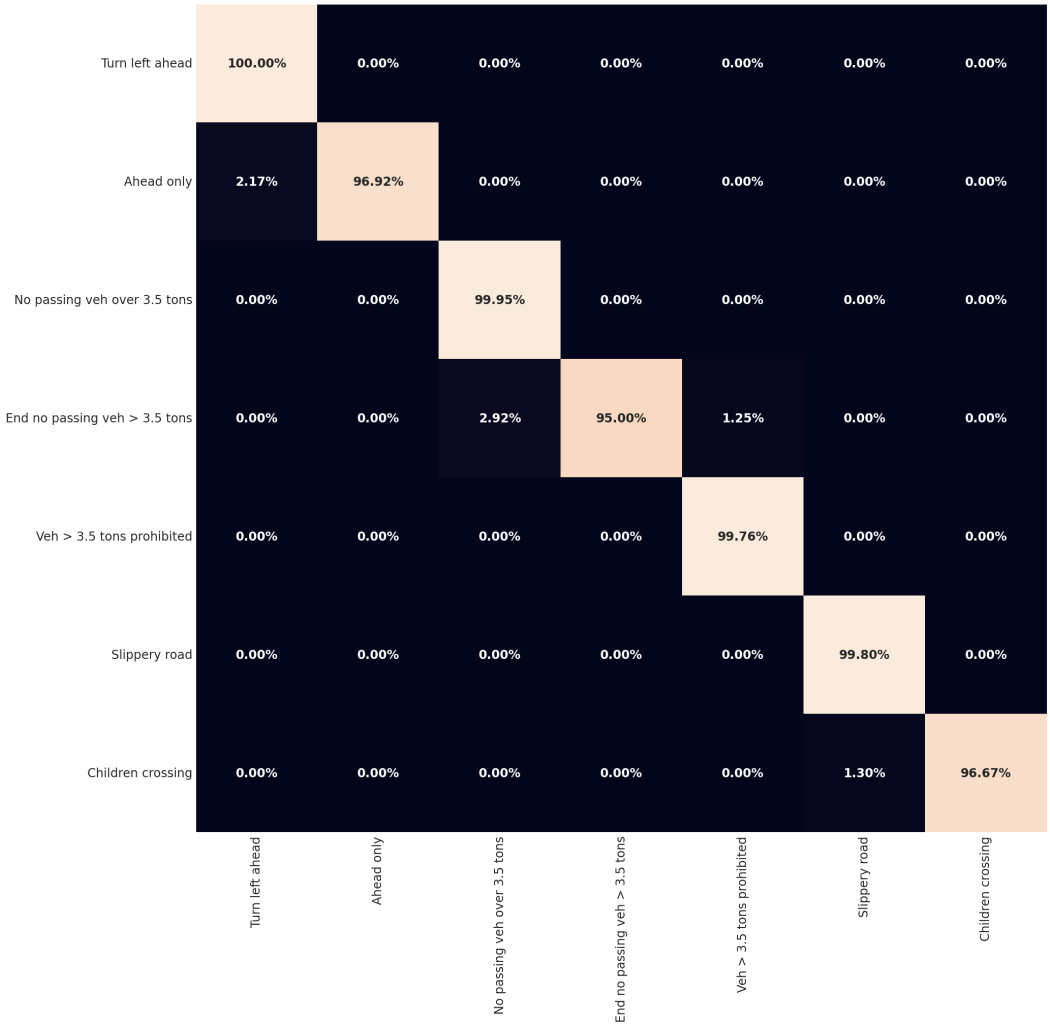

Figure 3.4: Confusion matrix of the VGG-like model on testing data (Own Illustration)

The ResNet is not implemented with an adjusted architecture. Because of that, the predefined model provided by TensorFlow is utilized. It is loaded without the weights pre-trained on the ImageNet database, since training from scratch is desired for nor comparability, and also feasible due to the sufficient size of the GTSRB data set. Furthermore, the default top-layer used to predict the 1000 classes contained in ImageNet is adjusted to reflect the 43 classes in GTSRB. The predefined model is then expanded with the data augmentation layers. Even though the further methodology largely matches the one used for the VGG-like model, one notable difference lies in training. Because the performance on the validation set is much less stable in the ResNet50 model, the patience for early stopping is increased to 20 epochs. The progress is shown in the following Figure 3.5.

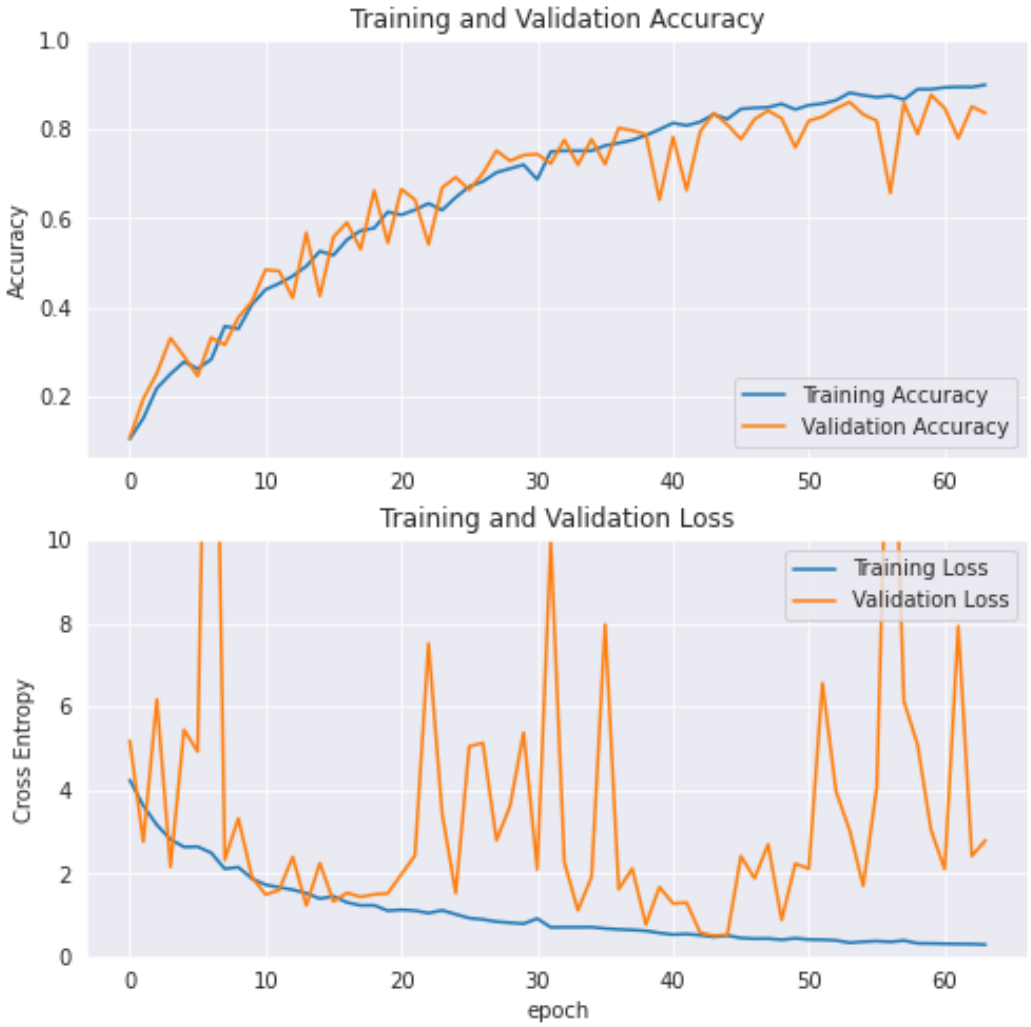

Figure 3.5: History of loss and accuracy for the ResNet50 model (Own Illustration)

After their training, both models are evaluated in respect to their accuracy on all three sets and additionally precision, recall and F1-score on the training data set. The results are depicted in Table 3.1 and Table 3.2.

Data Set	VGG-like model	ResNet50
Training	98.44%	89.99%
Validation	99.22%	83.62%
Testing	99.37%	84.49%

Table 3.1: Comparison of the implemented models in respect to their accuracy on all three sets

Metric	Average	VGG-like model	ResNet50
Accuracy	-	0.99	0.84
Precision	Macro	0.99	0.85
Precision	Weighted	0.99	0.86
Recall	Macro	0.99	0.84
Recall	Weighted	0.99	0.84
F1-Score	Macro	0.99	0.83
F1-Score	Weighted	0.99	0.85

Table 3.2: Comparison of the implemented models in respect to different error metrics on the testing set

As both tables show quite clearly, the VGG-like model outperforms the ResNet50 noticeably, and shows a very good performance across all metrics. Another thing to note is that the ResNet50 shows a much stronger tendency to overfit. This is not only evident from the strong difference between the accuracy on the ResNet50 models' performance on training and testing data displayed in Table 3.1, but can also be seen in Figure 3.5, especially concerning the loss. In contrast, the VGG-like model shows a consistent performance on all subsets of the GTSRB data set.

3.3 Implementation YOLOv7

As shown in section 2.3 YOLOv7 is the state-of-the-art image detection model for real time object detection. It achieves the best results in both targets, namely speed and accuracy. Due to this reason YOLOv7 is implemented for the traffic sign detection part. An overview over the implementation as well as the results will be part of the following chapter. The traffic sign detection data set has only a very limited amount of data as mentioned in section 3.1. Therefore, transfer learning is used to compensate this. The source model was an implementation of the published YOLOv7 model pre-trained on the MS COCO data set. By default, the MS COCO dataset contains only a few specific traffic signs. Therefore the traffic sign detection images were used to fine-tune this pre-trained model on our traffic signs to achieve a higher accuracy as well as a more specific model. In the original data set the traffic signs are ordered in 43 different classes, but with only very limited examples per class. It doesn't seem valuable to implement the classification task directly in the YOLOv7 model, because it is expected to result in poor accuracy due to the low number of samples.

Based on this it seems logical to change the class structure to binary with the two options "traffic sign" and "background". To also have the classification part included the classification model and the detection model will be merged in a prediction function, but this will be part of the following chapter.

The YOLOv7 model is fine-tuned over 50 epochs. The results of the training can be seen in Figure 3.6. In the case of object detection, three different loss functions are considered. The box loss indicates how well the respective bounding box is predicted. On the other hand the objectness loss considers the ability of the model to recognize whether an object is present. The classification loss would be relevant if more different objects are to be classified using the YOLO model. Since the GTSRB data set is simplified, this is not relevant and thus empty. During the training a continuous improvement can be seen. The mean Average Precision (mAP) can be increased to just over 80%.

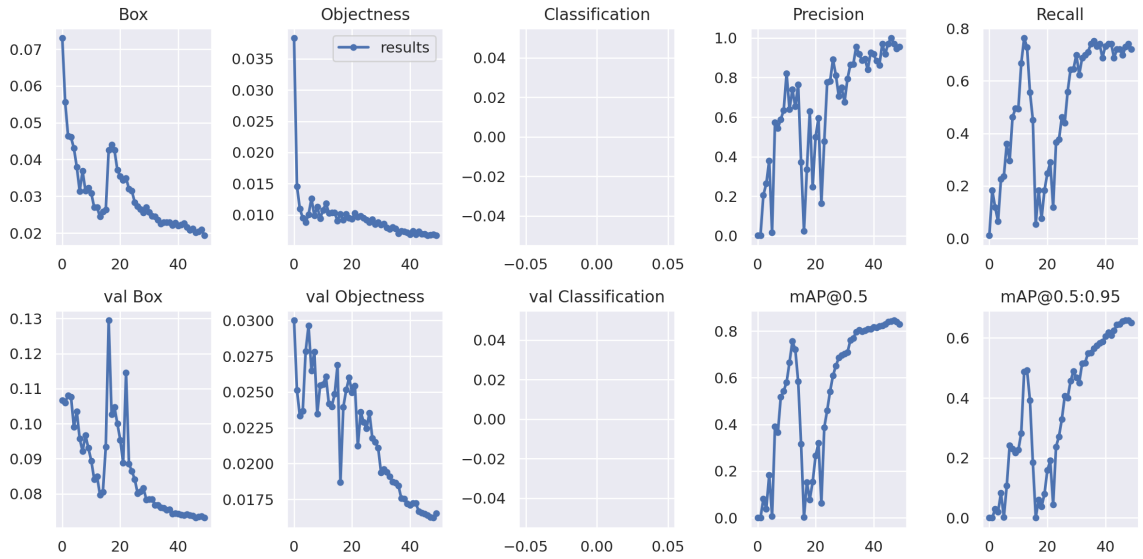

Figure 3.6: History of loss and mAP for the Yolov7 model
(Own illustration)

During evaluation the YOLOv7 model can detect a traffic sign in 78% of the cases. The mAP of 82.9% outperforms the MS COCO data set and source model that has an mAP of 51.4% clearly. This shows the efficiency of fine-tuning, although the smaller number of classes also simplify the task. The Precision Recall Curve also illustrates the good results, since, for example, up to a recall of approx. 70%, all traffic signs can be predicted appropriately.

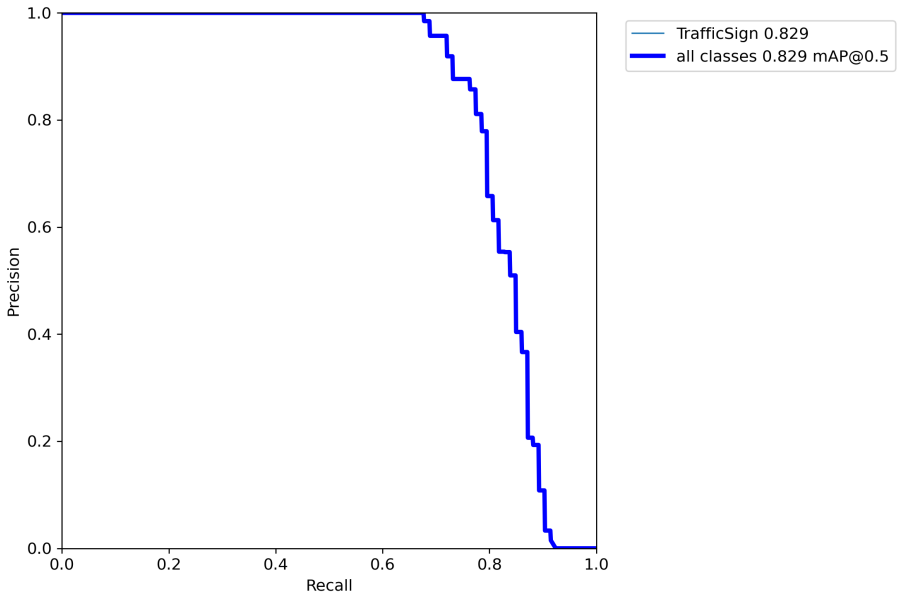

Figure 3.7: Precision-Recall curve of Yolov7 object detection (Own illustration)

With the YOLOv7 model it was possible to fine-tune the state-of-the-art object detection model for traffic sign detection. Although the number of training and testing data is quite less transfer learning can be used to achieve promising results.

3.4 Connection of both Models

After two models for the classification- and detection task could be implemented and evaluated in section 3.2 and section 3.3, the combination of both models follows in this chapter. Afterwards the model combination will be evaluated.

Due to the nature of the task the traffic sign has to be detected first. Afterwards the detected area in the picture can be classified. Therefore it is quite clear, that in the combination first the object detection model is used to predict the bounding box. Then, this area is cropped out of the original image and the VGG-like model classifies the image section.

The model combination was also evaluated on the GTSDB data set. Here, only the test data set of the YOLOv7 model is used, since the training of the object detection has already taken place on the training and validation data sets. Otherwise these data known

to the model could lead to falsified results. The disadvantage, however, is the small size of the test data set of 50 examples.

The accuracy of the model combination on the test data is only 40 percent. The results of the previous models would suggest a higher accuracy. This may be either because the errors are exponentiated across the two models or because the small sample size does not produce meaningful results.

For all traffic signs that were present more than 3 times in the data set, Figure 3.8 shows the Confusion matrix. This reveals that the frequent traffic signs are mostly predicted correctly. However, the model also tends to confuse similar traffic signs easily, for example "Speed limit (80km/h)" and "End of speed limit (80km/h)".

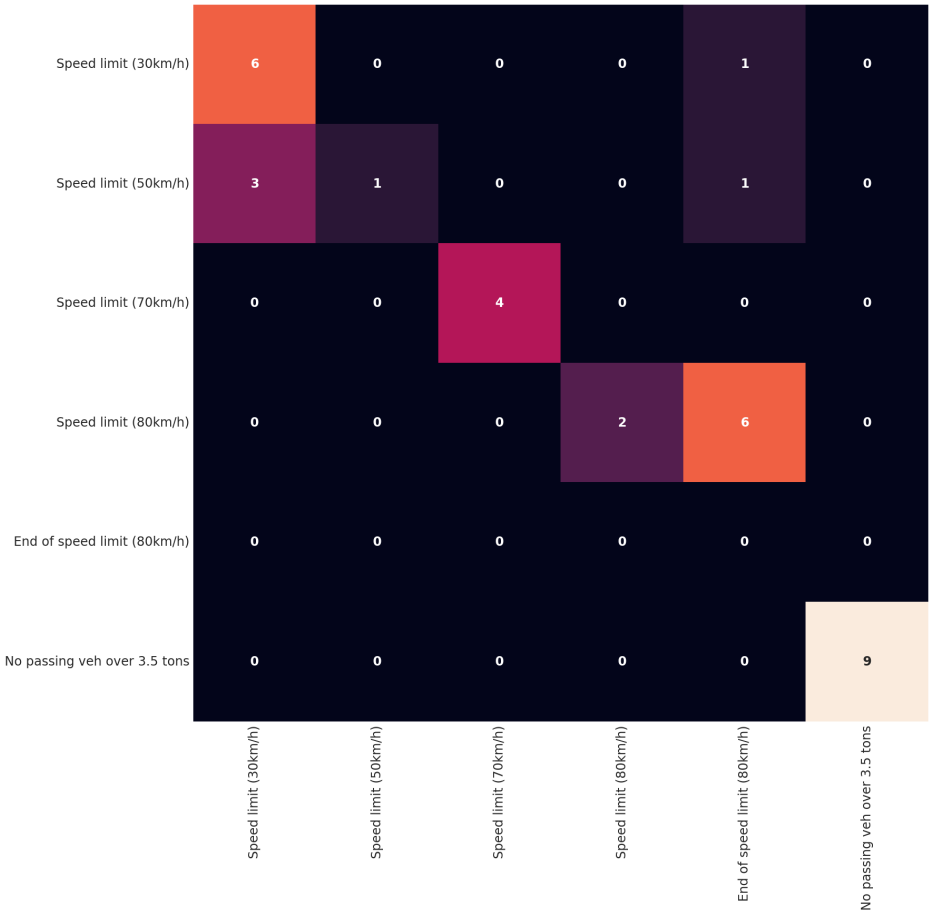

Figure 3.8: Confusion matrix for the combination of both models (Own illustration)

Another possibility for the evaluation is to use one recordings while driving. This has the

advantage that already two seconds of a video at 30 FPS provide a higher amount of images for evaluation than the test data set used above. For this reason and because this is closer to reality a video was also used to evaluate the model. The result can be seen in the further appendix of the paper.

In a visual evaluation of the result it can be seen that the model was able to recognize all traffic signs. Even though the model had some fluctuations in the classification, it predicted the appropriate traffic sign with a large majority. However, a long distance or a bad angle to the detection and classification object pose difficulties.

The overall accuracy of the models appears to be expandable at first glance. Nevertheless, an application in road traffic seems to be possible, because with a live video input each traffic sign appears on a multitude of images. Hereby, with a programming logic that still has to be implemented, the errors can be reduced significantly. The attached video already shows the effectiveness of the two combined models.

The focus of the model combinations was primarily on accuracy and appropriate evaluation. A more precise evaluation of the time duration of the models and the use of the models in a real time application is still pending.

4 Conclusion

4.1 Summary

The goal of this project, implementing a model that supports drivers by providing information about the traffic signs along the road, was achieved through an two step approach. To implement the approach the task was divided into two models, one for performing the classification and one for detecting the traffic signs.

For the classification part a simplified VGG-like model was built and outperformed a more complex ResNet50 model. The outperformance is based on the complexity of the task. The input images are not that complex which allows the simpler model to fit better on the data. The ResNet50 is simply way to complex for the input data and overfits as it does not generalize strongly and is to close on the input data. Another point that benefits the VGG-like model is the training speed and how long it takes to make a prediction. The VGG-like model is much faster than the ResNet50 which is really important when it comes to using the model in the real world. On the streets the prediction speed is crucial as every missed stop or pedestrians crossing sign can have drastic consequences like injured people or children.

For the detection part a pre-trained YOLOv7 model was used as the detection data set was really small and the pre-trained model absorbed this problem a bit and ensured a good performance.

To combine both models, the OD is done first, then the detected box is cropped from the image and a classification performed on the cutout.

4.2 Critical Reflection and Outlook

All in all the results of the project were satisfying, especially on their own, and the goal of providing the information from traffic signs for the driver was achieved. Nevertheless, there are some points that have to be noted here. Firstly it has to be mentioned that the data set for the detection part was really small which led to the fact that in any case a

pre-trained model had to be used as training a basic model on the data would have lead to a very bad performance. Another point which is directly connected to the detection data is the missing possibility to use data augmentation to boost the number of training samples. This lack is caused by the modification of the input images that is performed during the augmentation. Through modifying the pictures the coordinates of the traffic signs also change which is why you would have had to adjust the labels as well. This option of boosting the number of samples is too complex for the scope of this project, but would definitely be worth a try in a future project. But nevertheless it has to be mentioned that the combination of two powerful models (VGG-like and YOLOv7) lead to a good performance and more important, achieved this performance with a very low prediction time which is crucial in such a time-critical task. Compared to similar projects done by other researchers the VGG-like model performs also well as it reached a testing accuracy of 99.36%. With this accuracy it outperformed the results of Sermanet and LeCun, as well as Li et al., who both conducted the same task of traffic sign recognition on the GTSRB data set. For the case of Sermanet and LeCun, this might be caused by their paper being published in a 2011 when research in the area of CV was far less advanced than today. The VGG architecture for example was only formally described in 2014 by Simonyan and Zisserman (Sermanet & LeCun, 2011, p. 2811; Li et al., 2019, p. 6; Simonyan & Zisserman, 2014).

The OD however lead to some initial problems. One implementation of YOLOv7 lead to some difficulties, because even though it delivered a well enough performance during training, the predictions were cryptic and not processable, which required the switch to another implementation.

Another point that has to be mentioned is that the model does not take extreme weather conditions such as heavy rain or snow falls into account. This is important as extreme weather conditions make taking pictures of the direct environment of the car and thus detecting the traffic signs extremely difficult and in some cases even impossible. But for this project this is negligible as it is focused on CV and not on machine vision. The difference between the two is simply that for CV how the pictures are taken is irrelevant as it only processes the images itself. Nevertheless, in the future the combined model could be linked directly to a machine vision system like a camera on the hood of a car. This would make the model usable in daily life on the roads and not only in a highly restricted test environment like in this project. Additionally the model could be combined with other parts of autonomous driving systems like long distance radar systems or advanced sensor.

This would give the driver more insights into the environment through which he moves and in a final step would enable the car to drive autonomously. Another point that could be improved is the speed of the model. At this point the model is already very fast but through using more powerful hardware its speed could be boosted even more.

Bibliography

Aslansefat, K., Kabir, S., Abdullatif, A., Vasudevan, V., & Papadopoulos, Y. (2021). Toward improving confidence in autonomous vehicle software: A study on traffic sign recognition systems. *Computer*, *54*(8), 66–76. https://doi.org/10.1109/MC.2021.3075054

Batchelor, B. G. (2012). *Machine vision handbook*. Springer.

Chen, J., Jia, K., Chen, W., Lv, Z., & Zhang, R. (2021). A real-time and high-precision method for small traffic-signs recognition. *Neural Computing and Applications*, *34*(3), 2233–2245. https://doi.org/10.1007/s00521-021-06526-1

Gad, A. F., Gad, A. F., & John, S. (2018). *Practical computer vision applications using deep learning with cnns*. Springer.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Kanagaraj, N., Hicks, D., Goyal, A., Tiwari, S., & Singh, G. (2021). Deep learning using computer vision in self driving cars for lane and traffic sign detection. *International Journal of System Assurance Engineering and Management*, *12*(6), 1011–1025. https://doi.org/10.1007/s13198-021-01127-6

Li, W., Li, D., & Zeng, S. (2019). Traffic sign recognition with a small convolutional neural network. *IOP conference series: Materials science and engineering*, *688*(4).

Real-time traffic sign recognition in three stages [New Boundaries of Robotics]. (2014). *Robotics and Autonomous Systems*, *62*(1), 16–24. https://doi.org/https://doi.org/10.1016/j.robot.2012.07.019

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 779–788.

Sermanet, P., & LeCun, Y. (2011). Traffic sign recognition with multi-scale convolutional networks. *The 2011 international joint conference on neural networks*, 2809–2813.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Statistisches Bundesamt. (2022). Driver-related causes of accidents involving personal injury in road traffic. https://www.destatis.de/EN/Themes/Society-Environment/Traffic-Accidents/_Graphic/_Interactive/traffic-accidents-driver-related-causes.html

Sun, Y., Ge, P., & Liu, D. (2019). Traffic sign detection and recognition based on convolutional neural network. *2019 Chinese Automation Congress (CAC)*, 2851–2854. https://doi.org/10.1109/CAC48633.2019.8997240

Traffic sign recognition and analysis for intelligent vehicles. (2003). *Image and Vision Computing*, *21*(3), 247–258. https://doi.org/https://doi.org/10.1016/S0262-8856(02)00156-7

Wang, C.-Y., Bochkovskiy, A., & Liao, H.-Y. M. (2022). Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv preprint arXiv:2207.02696*.

Wei, L., Runge, L., & Xiaolei, L. (2018). Traffic sign detection and recognition via transfer learning. *2018 Chinese Control And Decision Conference (CCDC)*, 5884–5887.