TUM School of Computation, Information and Technology
Technical University of Munich

Interdisciplinary Project

Accident Detection on the A9 Test Stretch Using Roadside Sensors

Unfallerkennung auf der A9 Teststrecke mittels sich am Straßenrand befindenden Sensoren

Supervisor Prof. Dr.-Ing. habil. Alois C. Knoll

Advisor Walter Zimmer, M.Sc.

Author Daniel Lehmberg, B.Sc.

Date March 11, 2024 in Munich

Disclaimer

I confirm that this interdisciplinary project is my own work and I have documented all sources and material used.

Munich, March 11, 2024

(Daniel Lehmberg, B.Sc.)

Abstract

This work focuses on developing an automatic accident detection for the A9 test stretch, which is a test stretch for autonomous driving near Munich. The time between the occurrence of an accident and the arrival of medical assistance significantly impacts whether the passengers of a vehicle survive an accident. As an automatic accident detection would reduce this time, it has the potential to help save lives. However, most of the existing accident detection methods have never been tested on real traffic data of a test stretch. Therefore, this thesis investigates whether it is possible to reliably detect accidents on the A9 test stretch in real-time using roadside sensors. To achieve this, two accident detection methods have been implemented: a rule-based and a learning-based accident detection. Both have been integrated into the A9 test stretch and optimized. The conducted experiments show that the learning-based accident detection achieves a precision of 0.8 and a recall of 1.0 on the A9 test data and thus a high accident detection accuracy. Furthermore, the runtime of the learning-based accident detection for one second of a recording does not exceed 127 ms which makes it fast enough to detect accidents in real-time. Based on these results, it can be concluded that it is possible to reliably detect accidents on the A9 test stretch in real-time.

Zusammenfassung

Diese Arbeit beschäftigt sich mit der Entwicklung einer automatischen Unfallerkennung für die A9 Teststrecke, die eine eine Teststrecke für autonomes Fahren in der Nähe von München ist. Die Zeit zwischen dem Auftreten eines Unfalls und dem Eintreffen medizinischer Hilfe hat einen erheblichen Einfluss darauf, ob die Insassen eines Fahrzeugs einen Unfall überleben oder nicht. Da eine automatische Unfallerkennung diese Zeit verringern würde, hat sie das Potenzial dazu beizutragen, Leben zu retten. Die meisten der bisher existierenden Unfallerkennungsmethoden wurden jedoch noch nie mit echten Verkehrsdaten einer Teststrecke getestet. Deshalb wird in dieser Arbeit untersucht, ob es möglich ist, Unfälle auf der A9 Teststrecke mittels sich am Straßenrand befindenden Sensoren zuverlässig und in Echtzeit zu erkennen. Dazu wurden zwei Unfallerkennungsmethoden entwickelt: eine regelbasierte und eine auf Deep Learning basierende Unfallerkennung. Beide Unfallerkennungen wurden in die Softwarearchitektur der A9-Teststrecke integriert und für diese optimiert. Die durchgeführten Experimente zeigen, dass die lernbasierte Unfallerkennung eine Präzision von 0,8 und einen Recall von 1,0 erreicht und damit eine hohe Unfallerkennungsgenauigkeit. Außerdem beträgt die Laufzeit der regelbasierten Unfallerkennung maximal 127 ms, was diese schnell genug macht, um Unfälle in Echzeit zu erkennen. Basierend auf diesen Ergebnissen kann die Schlussfolgerung gezogen werden, dass es möglich ist, Unfälle auf der A9 Teststrecke mittels sich am Straßenrand befindenden Sensoren zuverlässig und in Echtzeit zu erkennen.

Contents

Chapter 1

Introduction

1.1 Motivation

Each year over a million people die from the consequences of road traffic accidents. In 2021, there were "an estimated 1.19 million road traffic deaths" [13, page 16] according to the World Health Organisation. This makes road traffic injuries one of the most common causes of death. In fact, it was the twelfth leading cause of death in 2019 when all ages are considered. In the age group from five to 29 years, it was even the main cause of death in 2019 [13, page 16]. Also in Germany, there are still many people dying from the consequences of an accident. In 2021, for example, there were 2562 road traffic deaths [2].

However, the number of road traffic deaths could be reduced if post-crash care would be provided more quickly. This is because post-crash care is extremely time-sensitive. Even a delay of just a few minutes "can make the difference between life and death" [13, page 50] after an accident. Consequently, timely and proper post-crash care would increase the likelihood of survival following a crash. Furthermore, also the risk for lifelong disability or other long-term impacts of road traffic injuries could be reduced by faster post-crash care [13, page 50]. Making post-crash care faster would therefore have significant benefits. It could be achieved by using automatic accident detection, which in turn could automatically notify the emergency services immediately after an accident happens.

1.2 Problem Statement

However, such a system is not used in practice yet. In some countries like the ones in the EU there already is a similar, but less powerful system called *eCall*. This system automatically connects the occupants of a vehicle to the closest emergency-response network in case of a serious accident and transmits important information like the current location. Nevertheless, the system is limited to serious accidents and does not exist in many countries. Furthermore, there are still many cars without it as it has only been mandatory in the EU since April 2018 [22].

Therefore, there is still the need for an accident detection system that works for all cars, in all countries, and for all kinds of accidents. In recent years many accident detection methods have been published. However, most of them have never been tested on real traffic data of a test stretch.

1.3 Objectives

This thesis investigates whether it is possible to reliably detect accidents on the A9 test stretch using roadside sensors. Furthermore, it is investigated whether it is possible to detect accidents on the A9 test stretch in real time using roadside sensors. For this purpose, automatic accident detection has to be implemented for this test stretch. The accident detection should be able to reliably detect accidents, focusing on the freeway section of the A9 test stretch. Furthermore, it should be possible to use it to search for accidents in the recorded traffic data of the A9 test stretch. This would make it possible to collect accident data for the future creation of an accident dataset. After implementing such an accident detection, extensive experiments must be conducted to investigate the accident detection accuracy of the implemented accident detection as well as its runtime. Based on the results of these experiments, the research questions of this thesis regarding the reliability and real-time capability of accident detection for the A9 test stretch can be answered.

1.4 Contribution

The main contribution of this thesis is summarized as follows:

- I improve the scenario detection of the A9 test stretch by making it possible to apply it to a large amount of data and by adding an image extraction feature and a logging feature for detected accidents and standing vehicles.
- Together with the work achieved by Marc Pavel we provide a complete pipeline for investigating the 100 TB of recordings from the freeway section of the A9 test stretch. Our experiments show that this pipeline works for detecting new accidents in the stored recordings.
- I realize a learning-based accident detection for the A9 test stretch that could be used for detecting accidents in the recorded data. For optimizing the learning-based accident detection, among other things, a custom accident dataset is created and an extensive empirical analysis is carried out.
- My created event log of special events in selected recordings of the A9 test stretch can be used for identifying appropriate training data for various detection tasks.
- Finally, I evaluate the learning-based accident detection and compare it to the rule-based accident detection using recordings from the A9 test stretch. For this comparison, recordings from the created event log are used. The results show that it is possible to reliably detect accidents on the A9 test stretch using roadside sensors. In fact, the learning-based accident detection even achieves a precision of 0.8 and a recall of 1.0 on the test data. The rule-based accident detection does not achieve such high accuracy values, but in return has a much lower runtime, as it only needs 127 ms to process one second of a recording. This shows that it is also possible to detect accidents on the A9 test stretch in real time using roadside sensors.

Chapter 2

Background

2.1 A9 Test Stretch

The *A9 test stretch* is a test stretch for research on autonomous driving that is part of the Chair of Robotics, Artificial Intelligence and Real-time Systems of the Technical University of Munich (TUM). It has seven measuring points distributed over a total length of 3.5 km along the B471 federal highway and the A9 freeway near Munich [5]. With that, a total of 75 sensors are located at these measuring points, including several RGB cameras [4]. These cameras cover three different types of traffic situations: a freeway, a highway, and a crossing. Figure 2.1 shows an example image for each of these types.

Figure 2.1: Example images taken by the RGB cameras of the A9 test stretch.

The collected traffic data is used to create real-time digital twins of all traffic participants and to automatically detect them. The information about the detected objects is stored as so-called *rosbags*, together with other data like the images taken by the RGB cameras. These images have a resolution of 1920x1200 pixels and show streets from a slightly elevated front view, also called a steep elevated view.

There are four cameras located on the freeway section of the test stretch. Two of them are recording the respective section of the A9 freeway in the northern direction. They are both located at the same measurement point and differ in their focal length. The other two cameras are recording the same section of the A9 freeway but in the southern direction. They also differ in their focal length and are both located at a second measurement point [5].

2.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) play an important role in many accident detection methods. CNNs are a special form of *Deep Neural Networks* (DNNs) and are typically used for image recognition tasks [7, page 445]. They are usually composed of a few convolutional layers, followed by a pooling layer, then again a few convolutional layers, followed by a pooling layer, and so on [7, page 460]. In the end, a feedforward neural network, composed of one or more fully connected layers, is added [7, page 460f.]. This architecture is illustrated in Figure 2.2.

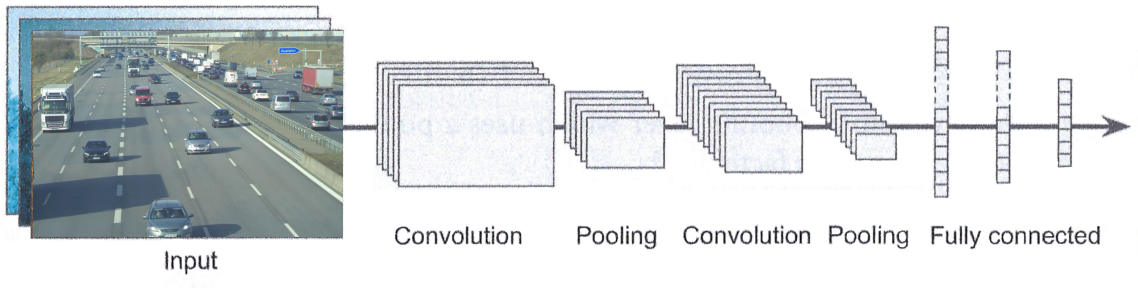

Figure 2.2: Typical architecture of a CNN (based on [7, page 461]).

The main building blocks of a CNN are *convolutional layers*, *pooling layers*, and *fully connected layers*. Convolutional layers learn specific features of an image by so-called *kernels*. Unlike in normal *Artificial Neural Networks* (ANNs), the neurons of a convolutional layer are only connected to a small region of the previous layer [11]. Pooling layers "aggregate the inputs [of its neurons] using an aggregation function such as the max or mean" [7, page 457]. This way, the number of parameters and, therefore, also the computational load and memory usage are reduced [7, page 457]. The fully connected layers then work like in normal ANNs and provide the prediction of the CNN [7, page 461].

One of the most famous CNNs used for object detection tasks is *You Only Look Once* (YOLO). YOLO is an "extremely fast and accurate object detection architecture" [7, page 489] originally proposed by Joseph Redmon et al. in 2015 [16]. It is a *Convolutional Neural Network* (CNN) with several improvements like *skip connections* and using images of different scales for training [7, page 489f.]. In the last years, several YOLO implementations have been published like YOLOv8 or YOLOv9. There are also pre-trained models available of the different YOLO versions which can be used as a baseline for training own models.

2.3 Accident Detection

Accident detection is the process of automatically detecting accidents in given traffic sensor data. This sensor data is usually images or the frames of a video. However, also other kinds of traffic sensor data are possible like audio recordings or GPS data. The accident detection can be performed in the vehicle itself, based on the vehicle's sensor data, or in some kind of traffic infrastructure based on roadside sensor data.

In recent years, many different accident detection methods have been published. In general, they can be categorized into two different types: rule-based accident detection methods

and learning-based accident detection methods. This differentiation is shown in Figure 2.3.

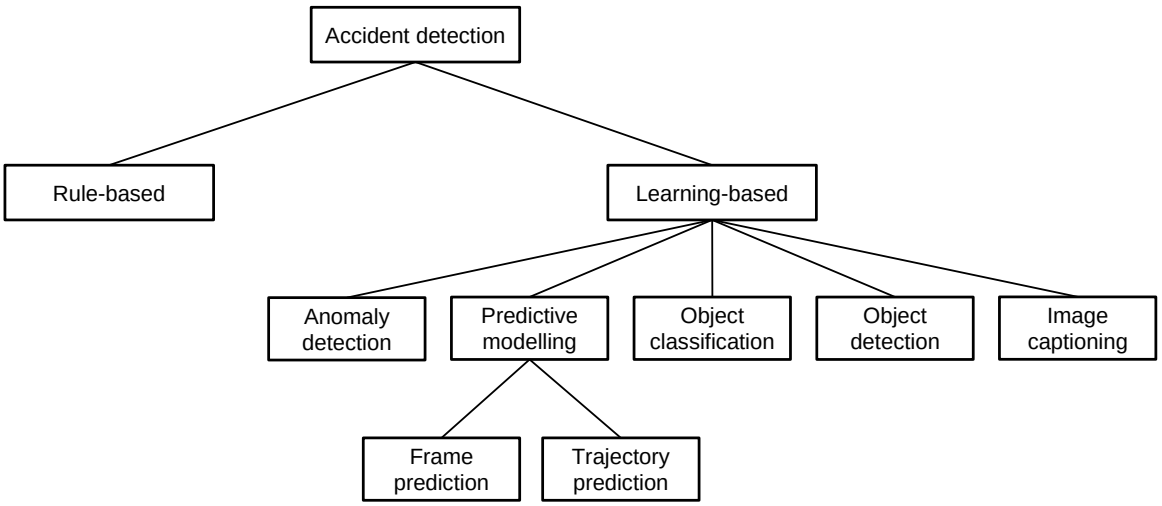

Figure 2.3: Overview of the taxonomy of accident detection methods.

Rule-based accident detection methods use hand-crafted predefined rules to detect accidents in the input data. When creating these rules, it is therefore possible to build on knowledge about accidents and convert it into rules. However, rule-based accident detection methods have the drawback that they can never cover all possible accident scenarios. As the rules used to detect accidents are hand-crafted and there are theoretically an infinite number of possible accident scenarios, you can never have a rule for every possible accident. Therefore, it is not possible to detect all accidents by a rule-based accident detection [7, page 2f.].

This problem is solved by learning-based accident detection methods. The reason for this is that they do not need handcrafted, predefined rules for detecting accidents. Instead, they use labeled accident data to learn the rules for detecting accidents in the input data itself.

Learning-based accident detection methods can be differentiated into five different types: anomaly detection, predictive modeling, object classification, object detection, and image captioning. If anomaly detection is used to detect accidents, it is tried to recognize inputs that look "abnormal or novel to the model according to previously seen normal samples during training" [18, page 1]. There are several features in which accident detection can search for anomalies. Commonly used features are the trajectory of vehicles, the acceleration of vehicles [17], or the class labels of detected objects in a video frame [25].

Another type of frequently used learning-based method for detecting accidents is based on predictive modeling. The idea of predictive modeling is "to make an accurate prediction" [10, page 1] based on some input data. This idea can be used for accident detection by predicting how a traffic situation will develop. If the predicted traffic situation deviates from the actual traffic situation by a certain amount it is assumed that an accident has happened. Because otherwise, the actual traffic situation would be similar to the predicted one. The most common information predicted for detecting accidents is the frames of a video and the trajectories of the vehicles.

Also, object classification is a commonly used method for learning-based accident detection. Usually, a binary classification is then applied to each vehicle present in the input data. This classifier then predicts for each vehicle whether it is damaged or not. If a damaged car

is found an accident is detected [15].

A similar learning-based accident detection approach is to use object detection. It is realized by applying an object detection model that has been trained to detect accidents in input images. By doing so, it is not only detected whether there are accidents in the input data but also the location of the accidents in the respective input images [21].

Recently, a new accident detection approach based on image captioning has started to be explored. The image captioning process is done using visual language models like GPT-4V [12]. Such approaches generate descriptive captions for input images that answer the question of whether or not there is an accident in it. This way accidents can be detected in input images. Additionally, these kinds of approaches can also provide a description of the detected accident as well as possible reasons why it happened [24].

Chapter 3

Related Work

3.1 Accident Detection Methods

In recent years, many different accident detection methods have been published. As described in Chapter 2.3, they can be categorized into rule-based accident detection methods and learning-based accident detection methods. The latter can be based on anomaly detection, predictive modeling, object classification, object detection, and image captioning. Sometimes several of these approaches are combined into a single accident detection method.

The *Self-Supervised Consistency* learning framework (SSC-TAD) is a predictive model using frame prediction and object location prediction to detect accidents. The idea behind SSC-TAD is that there is a spatial-temporal consistency of features over multiple frames of a video. In particular, the temporal consistency of the location of road participants plays an important role in detecting accidents. Based on the previous frames SSC-TAD predicts the current frame as well as the bounding boxes of the objects in the current frame. The predicted results are then compared with the actual results, namely the current frame and the bounding boxes of the objects in the current frame. If the predictions are not consistent with the actual results it is concluded that an accident must have happened. Therefore, an accident is then detected by SSC-TAD [6].

A completely different approach for detecting accidents is used by the accident detection method proposed by Sabry et al. in 2021. It combines rule-based crash estimation and learning-based object classification. First, YOLOv3 is used to detect all vehicles in every tenth frame of an input video. Then, the MOSSE tracker is used to track the detected vehicles through the frames of a video. Using the results of the MOSSE tracker, a crash estimation is performed by calculating the location of the tracked objects ten frames ahead. Afterward, the distance of the calculated future center of each vehicle pair is calculated. If it falls below a certain threshold the vehicle pair is marked as a potential accident candidate. The final step is to apply a *state vector machine* (SVM) on an optical flow vector to perform a binary classification of whether there is an accident in the input data or not. This approach has the huge drawback that it only works for accidents in which two or more vehicles crash with each other [17].

This problem is solved by the accident detection method proposed by Pillai et al. in 2021. It is achieved by just checking whether a vehicle is damaged and not even considering other vehicles for the accident detection task. In general, the accident detection process is split into three stages: vehicle detection, vehicle tracking, and accident classification. First, *Mini-YOLO*, a custom object detection model based on the pre-trained MobileNet-v2 model, performs ob-

ject detection on the input images. Then, *Simple Online Real-time Tracking* (SORT) is applied to track the detected vehicles. Each of the tracked vehicles has a damage status variable, indicating whether the vehicle is damaged or not. In addition to the tracking algorithm, the detection results are also used to extract an image of each detected vehicle in the input data. This image is extracted using the predicted bounding box and therefore only contains the respective vehicle. The extracted images of the vehicles are then all individually fed into an SVM that performs a binary classification of whether the vehicle in the input image is damaged or not. If the classification step concludes that a vehicle is damaged, it checks the damage status variable stored for each vehicle. If the vehicle has not been damaged before, an accident is detected. Otherwise, the vehicle has already been damaged, therefore no accident is detected. Using this approach makes accident detection rather computationally inexpensive and therefore suitable for real-time usage [15].

Another accident detection method fast enough for real-time usage is the one proposed by Zhou et al. in 2022. In the first step, a multilayer neural network encodes the temporal features of a video. Then, the frames of the input video are clustered to detect potential accident frames. On these potential accident frames *Faster-RCNN* is applied to detect the vehicles in the frame. The spatial relationship of these detected vehicles is then encoded with the CNN features received from the last convolutional layer of the Faster-RCNN model. This results in coding matrices that are classified using an SVM. The classification results then correspond to the accident detection results for a given input frame [25].

3.2 Accident Datasets

For learning-based accident detection methods, it is important to have enough high-quality training data. Therefore, this chapter gives an overview of some publicly available accident datasets. The most important information about these datasets is summarized in Table 3.1.

All datasets consist of images or videos as most learning-based accident detection methods use some kind of visual information as input. Apart from the type of data and the year of publication, the datasets also differ in where the data was taken and therefore in the view perspective. Images and videos taken by a dashcam correspond to the vehicle view. If they were taken by roadside cameras on the ground they show the roadside view and if they were taken by elevated roadside cameras they correspond to the steep elevated view. Some of the datasets contain a mix of all three mentioned recording locations. Additionally, the datasets also differ in the content of the data. In some datasets, only the accidents themselves are visible in the images or videos, in other datasets, the whole street can be seen, including the accident. All presented datasets are allowed to be used for research purposes except for the *yoloaccident* dataset which does not provide any information about its license.

Image Datasets

Many accident datasets are image datasets. That means that they only contain images of accidents and the respective labels. One rather big image dataset is the *Accident-Images-Analysis dataset*. It consists of 2,398 accident images taken by roadside cameras on the ground. All the images only contain the accident itself. This means that no further context of the whole scene is provided. A big drawback of the Accident-Images-Analysis-Dataset is that its images

Name	Data	Year	#Accidents	Perspective	Image content	Labels
AIAD [8]	images	2018	2,398	R	accident	A
YA [9]	images	2022	50	R	accident	A, BB
CCTVF [3]	images	2020	462	SE	street	A
ADM [20]	images	2023	1,741	mixed	mixed	A, BB
AC [14]	images	2020	13,338	mixed	mixed	*none*
DA [23]	images	2023	691	V + R	mixed	A, BB
CCD [1]	videos	2020	1,500	V	street	A, E
Argus [17]	videos	2020	153	SE	accident	A
CADP [23]	videos	2018	1,416	mixed	street	*none*

Table 3.1: Overview of some publicly available accident datasets. The datasets are compared regarding the type of data, the year of publication, the number of accidents, and the content of the image or frame. Furthermore, they have different view perspectives, namely roadside view (R), steep elevated view (SE), and vehicle view (V). Apart from that their images and videos differ in the labeled information. While most datasets only have labels whether there is an accident in an image or video (A), some of them have labels also providing information about a bounding box (BB) or external factors (E) like the time or the weather conditions during the recording. The accident datasets considered in this comparison are the *Accident-Images-Analysis-Dataset* (AIAD), the *yoloaccident* dataset (YA), the *Accident Detection From CCTV Footage* dataset (CCTVF), the *Accident detection model* dataset (ADM), the *accidents* dataset (AC), the *Deep Accident* dataset (DA), the *Car Crash Dataset* (CCD), the *Argus* dataset and the *Car Accident Detection and Prediction* dataset (CADP).

only have a resolution of 28x28 pixels which is too small for most use cases. Furthermore, only an accident label is given and no bounding box for the exact location of the accident in the images [8].

A pretty similar but much smaller dataset is the already mentioned yoloaccident dataset. Its images are also taken by roadside cameras on the ground and also only contain the accident itself. Unlike the Accident-Images-Analysis-Dataset the labels of the yoloaccident dataset do not only provide information on whether there is an accident in the input data but also the bounding box for the position of the accident. However, the yoloaccident dataset is with its 50 images a fairly small dataset [9].

The *Accident Detection From CCTV Footage* dataset is a much bigger dataset that is also much more similar to the use case of the A9 test stretch. It consists of 462 accident images which were taken by roadside cameras from a steep elevated view. In addition to that, the images contain the whole street and not just the accident itself. However, no labels with bounding boxes for the accidents are provided [3].

As already mentioned, not every accident dataset only contains images from one perspective or with one type of image content. Sometimes the different types are mixed, usually to have a bigger dataset in the end. If that is the case it varies from accident to accident whether the image of an accident is taken by a roadside camera on the ground, an elevated roadside camera, or a dashcam. Furthermore, they then also differ in whether only the accident or the whole street is visible in the image. A dataset that belongs to this category is the *Accident detection model* dataset. With 3,250 accidents it is relatively large for an accident dataset. The labeled information includes whether there is an accident in the image as well as the bounding box of an accident [20].

Another image dataset that contains mixed perspectives and mixed image content is the

accidents dataset. It is much bigger than all the other datasets as it contains 13,338 accidents. However, the images in the dataset differ in their resolution and aspect ratio. 9,303 of the images for example only have a resolution of 28x28 pixels and are therefore, as already mentioned, too small for most use cases. Furthermore, the dataset also includes some images of planes. There are no labels provided by this dataset. However, this is not a problem as the dataset only contains videos with accidents [14].

To get a bigger and more diverse dataset synthetic images can be used. One dataset following this approach is the *Deep Accident* dataset. To achieve a diverse set of accidents, 691 accident scenarios were generated based on crash reports published by the *National Highway Traffic Safety Administration* (NHTSA). For each of these accidents, the dataset does not only contain images taken by a single camera but by multiple ones. Specifically, there is data from four vehicles and one infrastructure component available for each generated accident. This means that the dataset contains images taken by dashcams as well as images taken by an elevated roadside camera for each of the accidents. However, the domain gap problem could arise when synthetic accident images are used for training [23].

Video Datasets

Apart from image datasets, there are also video datasets covering accidents. A rather big one is the *Car Crash Dataset* (CCD) which contains 1,500 videos of an accident. All the videos are taken by the dashcam of a vehicle and always contain the whole street and not just the accident itself. It also provides quite a lot of labeled information like whether there is an accident present in a frame of a video, whether a video was taken during the day or the night, and the weather conditions. However, no bounding box of the accident is provided [1].

The *Argus* dataset is another video dataset. However, it is much smaller than the CCD dataset. It contains videos of 153 accidents, taken by elevated roadside cameras. The video frames only contain the accident itself and not much of the surroundings of the accident. The only labeled information provided is if there is an accident in a video or not. So, there is no information on which frames of the videos contain an accident [17].

This problem also applies to the *Car Accident Detection and Prediction* dataset (CADP). In fact, it does not even have any labels. However, this is, as already mentioned, not a problem as the CADP dataset only contains videos of accidents. The recording location of the videos in this dataset is mixed. Consequently, it depends on the accident whether the video was recorded by a dashcam, by a roadside camera on the ground, or by an elevated roadside camera. All the 1,416 accident videos show the whole street on which an accident happened and not just the accident itself [19].

Chapter 4

Methodology

As mentioned in Chapter 2.3, accident detection methods can in general follow a rule-based approach or a learning-based approach. To answer the research question of this thesis, both of these approaches have been realized. This means that rule-based accident detection and learning-based accident detection have been implemented for the A9 test stretch.

4.1 Event Log

For developing the rule-based and the learning-based accident detection method it is important to have some rosbags covering different kinds of special events like accidents, standing shoulder events, or traffic jams. On the one hand, this is necessary to get some insights into how different metrics differentiate between the different events. This information is needed for the development of the rule-based accident detection. On the other hand, it is necessary to manually classify some rosbags whether there is an accident in them or not for the development of learning-based accident detection. For example, a custom dataset for model training can be created based on the results of the event log. Finally, the information of an event log can be used as test data for comparing the rule-based and the learning-based accident detections.

The event log was created for a bunch of recorded rosbags from the A9 test stretch and can be seen in Table 4.1. For creating this event log the images from all the respective rosbags were extracted. In total, they cover almost four hours. The images were then manually inspected to classify the rosbags regarding interesting events like accidents, standing shoulder events, or traffic jams. It should be noted that for the most part rosbags were used for which it was already known that some kind of interesting event could have happened.

Particularly important for the development of accident detection are the already-known accidents in the recorded data. In total, four accidents have been known before the development of accident detection. The first one happened on the 8th of April 2021 at 12:11:53. This accident happened as a yellow car lost control, drove into the crash barrier, and then into the back of a white van. Figure 4.1 shows two images from this accident. In the first one, the moment can be seen in which the car crashes into the crash barrier. The second one shows the collision of the car with the white van.

Date	Start Time	End Time	Events
10/13/2020	18:52:23.25	18:53:23.02	Traffic jam
02/11/2021	11:32:00.93	11:33:00.93	Emergency vehicle, vehicle transporter
03/26/2021	13:41:41.00	13:42:13.20	Emergency vehicle, vehicle transporter
04/08/2021	11:30:00.93	11:33:00.93	**Accident**
05/15/2021	15:40:01.73	15:41:01.73	*No events*
05/15/2021	15:52:01.74	15:53:01:73	Standing shoulder
05/15/2021	16:44:01.74	16:45:01.73	Standing shoulder
05/15/2021	16:54:01.73	16:55:01.73	Vehicle transporter, standing shoulder
05/15/2021	16:55:01.74	16:56:01.73	Emergency vehicle, standing shoulder
05/15/2021	17:04:01.74	17:05:01.73	Standing shoulder
07/29/2021	19:44:24.15	19:45:24.07	Standing shoulder
07/30/2021	09:25:24.07	09:26:24.07	Standing shoulder
10/21/2021	10:21:44.36	10:22:44.35	**Accident**
03/07/2022	12:25:08.80	12:55:08.72	*No events*
03/07/2022	12:55:08.72	13:10:08.68	Tow truck
03/28/2022	17:19:15.35	17:34:15.34	**Accident**
03/28/2022	17:49:15.34	18:04:15.34	Emergency vehicle
05/11/2022	16:14:43.38	16:29:43.37	Emergency vehicle, vehicle transporter, tow truck
05/11/2022	16:29:43.37	16:44:43.38	Vehicle transporter
05/11/2022	16:44:43.38	16:59:43.38	Emergency vehicle, vehicle transporter
05/11/2022	16:59:43.38	17:14:43.37	Vehicle transporter
05/11/2022	17:29:43.38	17:44:43.37	Emergency vehicle, vehicle transporter
05/22/2022	17:14:43.38	17:29:43.37	**Accident**, standing shoulder
05/22/2022	17:44:43.38	17:59:43.37	Emergency vehicle, standing shoulder

Table 4.1: Event log of selected recorded rosbags from the A9 test stretch.

Figure 4.1: Accident on the A9 test stretch from the 8th of April 2021 (front left in the image).

The second known accident happened on the 21st of October 2021 at 07:28:53. The reason for this accident is that it was quite windy. As a result, the trailer of a blue car tipped over. This made the car lose control and crash into a delineator next to the hard shoulder of the A9 freeway. Figure 4.2 shows this whole scene in four images.

On the 28th of March 2022 at 17:19:18 another accident happened on the freeway section of the A9 test stretch. It was caused by a vehicle running into the end of a traffic jam. As a result, it collided with two other vehicles and made a 360-degree spin. The process of this accident is illustrated in Figure 4.3 using four images.

Figure 4.2: Accident on the A9 test stretch from the 21st of October 2022 (back left in the image).

Figure 4.3: Accident on the A9 test stretch from the 28th of March 2022 (front right in the image).

The last accident known before the development of accident detection occurred on the 22nd of May 2022 at 17:19:30. This time a vehicle burnt down on the hard shoulder. Figure 4.4 shows how the vehicle first started to smoke and later started burning.

Figure 4.4: Accident on the A9 test stretch from the 22nd of May 2022 (back right in the image).

An overview of all the mentioned accidents is given in Table 4.2.

Date	Time	Description
04/08/2021	11:31:52.54	Car crashes into crash barrier and standing van
10/21/2021	10:22:13.17	Car tips over as its trailer was blown over
03/28/2022	17:19:17.54	Car collides with 2 vehicles and makes 360-degree spin
05/22/2022	17:19:29.97	Car burns on hard shoulder

Table 4.2: Overview of accidents on freeway section of A9 test stretch that were known before the development of accident detection.

4.2 Rule-Based Accident Detection

4.2.1 Concept

The rule-based accident detection is performed on rosbags as this is the format in which the recorded data from the A9 test stretch is stored. These rosbags provide, among other things, the position of each vehicle for each timestamp. The vehicles recorded in a rosbag have a unique ID that stays the same while driving through a section of the test stretch. The position data of the vehicles is used to calculate some basic information about each vehicle like its velocity, in which line it drives or its distance to the vehicle in front and behind it. This information is then used by the already existing scenario detection of the A9 test stretch to detect traffic scenarios like a vehicle standing on the hard shoulder. Afterward, some of the calculated vehicle information and detected scenarios are used by rule-based accident detection to decide whether or not there is an accident in the data for a particular timestamp. This decision is made based on some hand-crafted, predefined rules that try to describe the scenarios in which accidents happen. For example, if the distance between two vehicles falls below a certain threshold that depends on the vehicle's velocity the rule-based accident detection detects an accident. A detailed explanation of how the rules of the rule-based accident detection work and how they are implemented can be found in Marc Pavel's documentation.

4.2.2 Improvement of Scenario Detection

As explained in Chapter 4.2.1 rule-based accident detection uses information like the distance between two vehicles or their velocity for detecting accidents. This kind of information is provided by the scenario detection or can be calculated using the information available in the scenario detection.

The scenario detection iterates through the data stored in a rosbag and tries to identify multiple kinds of scenarios like traffic jams, whether a vehicle drives faster than allowed, or whether a vehicle is standing on the hard shoulder. As an accident is also just a special kind of traffic scenario accident detection was integrated into the scenario detection.

Because scenario detection serves as a baseline for accident detection, the values provided by it must be accurate. Otherwise, the rule-based accident detection will not work reliably.

However, this was not the case. Therefore, many aspects have been improved in the scenario detection.

Apart from the data calculation of the scenario detection, its general functionality has been improved as well. A big improvement that makes it possible to apply scenario detection to huge amounts of rosbags is that it now works in a generic way regarding the ROS topic name. Consequently, it is no longer necessary to specify the ROS topic name when the scenario detection is performed. Instead, it now automatically checks for each rosbag what the name of the needed ROS topic is. If none of the required ROS topics is available in the respective rosbag, it will be skipped so that the execution of the scenario detection on an entire directory of rosbags is not interrupted.

When applying scenario detection to large amounts of recorded rosbags, it is important for later analysis that the results are stored in a useful way. For this reason, scenario detection has been extended to generate a useful directory structure for input data stored in a directory structure corresponding to the hierarchy `year/month/day/hour/`. This directory structure has been chosen as it is used by the recorded data from the A9 test stretch. For each rosbag, in which an accident is detected, a directory is created in which the scenario detection results for the current rosbag are stored. The scenario detection considers all cameras available in a rosbag and extracts and stores three images for each of them. More information about the extracted images is given in Chapter 4.2.3. Furthermore, a JSON file containing the statistics of the detected scenarios is created regardless of whether an accident has been detected. However, no images are extracted and no directory for a rosbag is created if no accident is detected.

To ensure that the results of the accident detection can be easily analyzed and used for further tasks, the most important information about each detected accident is stored in a CSV file. Apart from the year, month, day, and hour of the respective rosbag, also its name and the exact timestamp are saved. Like the extracted images and generated statistics, also the CSV file is saved in the specified output directory.

Apart from the scenario detection improvements mentioned above, many other changes have been made. For example the `standing_shoulder` values, the `velocity` values, and the `cut in` and `cut out` values work now. Moreover, multiple calculations like the distance calculation have been made much faster. More information about these and other additional improvements can be found in Marc Pavel's documentation.

4.2.3 Implementation of Image Extraction

To obtain the images of detected accidents an image extraction feature was implemented. This feature can not only be used to extract images of detected accidents but also to extract images of detected breakdowns or detected vehicles standing on the hard shoulder. It can be activated using the `-img` argument, followed by the kind of event, for which the images should be extracted. The possible options are `accident`, `breakdown`, `accident_breakdown` and `standing_shoulder`. Here, `accident_breakdown` means that the images of all accidents and all breakdowns are extracted. When activated, the image extraction feature automatically extracts three images for each detected event. This is done for every camera that is available in the current rosbag. For extracting the images the rule-based accident detection provides the timestamp of the input rosbag for which an event has been detected. This times-

tamp is then used by the *image extraction node* that extracts the three images by decoding the image information stored in the ROS messages of the rosbag. Figure 4.5 illustrates the image extraction process.

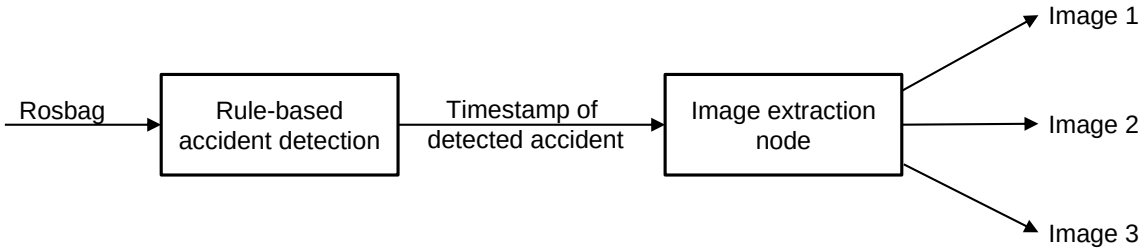

Figure 4.5: Image extraction process of the rule-based accident detection.

The image extraction extracts the image, that is the closest after the timestamp in which an event has been detected and that can be extracted separately. In general, it is not possible to only extract the image of a timestamp, in which an event has been detected, as the image encoding works in a way that makes it only possible to extract every tenth frame separately. Therefore, the next individually extractable frame is always used for image extraction. It always used the next individually extractable frame and not the closest one, as an image of a few milliseconds after an event happened is more useful than an image from a few milliseconds before an event happened. For example, a few milliseconds before an accident is detected, it probably has not even happened yet. But a few milliseconds after an accident is detected it should still be visible in the image. In addition, two additional images are extracted for each detected event. One from 4.8 seconds before the timestamp that is used to extract the already mentioned image and one from the timestamp 4.8 seconds after it. 4.8 seconds have been chosen for the interval because, with the given 25 frames per second (FPS), this corresponds to a frame number that can be divided by ten. This ensures that the frame can individually be extracted. As the image extraction iterates over the ROS messages in the correct chronological order, it first extracts the image 4.8 seconds before the detected event, then the one from the event itself, and finally the one 4.8 seconds after the detected event. Figure 4.6 illustrates how the correct timestamps for extracting individual images are found. All three extracted images are stored in the directory for the camera by which the images have been taken. This directory is, as mentioned in Chapter 4.2.2, created as a subdirectory for each input rosbag for which an event has been detected.

The reason why only three images are extracted for each detected event is the need for a low runtime. When a whole sequence of 9.6 seconds, which spans from 4.8 seconds before an event to 4.8 seconds after an event, is extracted this takes much longer than just extracting three frames. This computational overhead is not worth the additional extracted images as the three extracted images should be sufficient to have an image of the detected event. Additionally, also the iteration over the ROS messages in the `image_extractor_node` has been changed in a way that improves the computational speed. Instead of iterating over all ROS messages of a rosbag, the iteration is now stopped after the third image is extracted.

To ensure that the image extraction works automatically for every rosbag, regardless of the cameras and rosbag topics available in a rosbag, it was implemented in a generic way. Similar to scenario detection, image extraction automatically figures out which of the required ROS topics are available in a rosbag. Furthermore, also the ID of the available cameras is derived automatically. This way the image extraction works in a generic way and can be

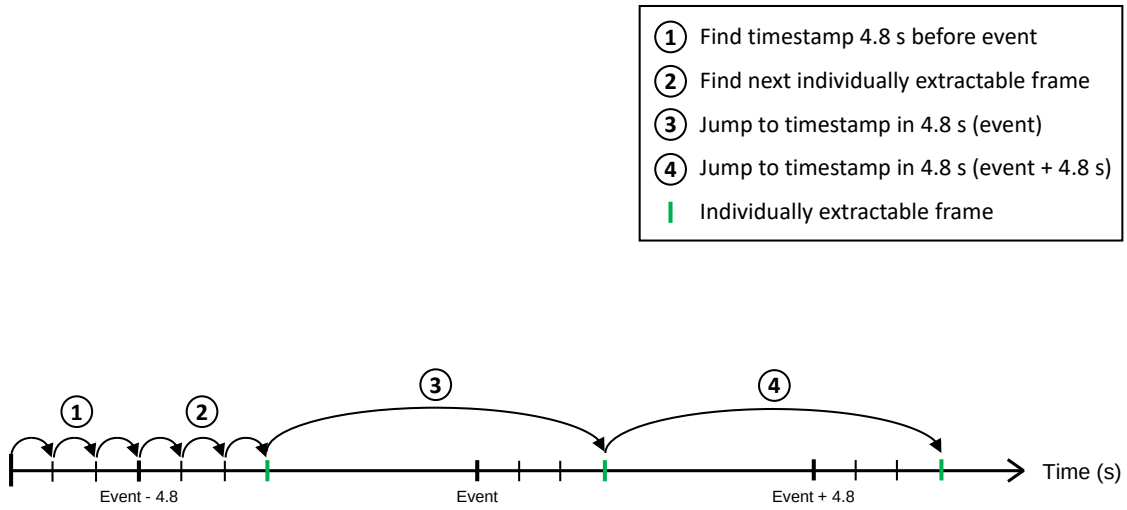

Figure 4.6: Process of finding the correct timestamp to extract individual images.

applied to a huge amount of recorded rosbags.

4.3 Learning-based Based Accident Detection

In addition to the rule-based accident detection also a learning-based accident detection was implemented.

4.3.1 Concept

The core of the learning-based accident detection is the pre-trained *accident detection model* from Shubhankar Shandilya [21]. This model was created by using transfer learning. For this, the *YOLOv8s Detect model* pre-trained on the COCO dataset was fine-tuned on an accident dataset created by Shandilya.

This dataset consists of 947 training images, 154 validation images, and 157 test images which makes 1258 images in total. On each of them, a preprocessing has been applied which resizes the images to 640x640 pixels. The images in the data set differ considerably in terms of the type of recording and the content of the image. While many of the images were taken from a slightly elevated view, there are also some images taken by vehicle dashcams or by roadside cameras on the ground. Furthermore, some of the images contain the accident with its surroundings and some just the accident vehicles without the surroundings. Figure 4.7 shows four example images from this dataset.

Many of the images contain an accident, whose label always also contains the respective bounding box. However, the bounding boxes are quite imprecise for many images. They often do not include the whole accident vehicle, too much of the background, or even parts of vehicles that are not involved in the accident at all. A few example images with imprecise bounding boxes are given in Figure 4.8. However, the final version of the learning-based accident detection does not use this pre-trained model anymore as it was replaced by a custom-

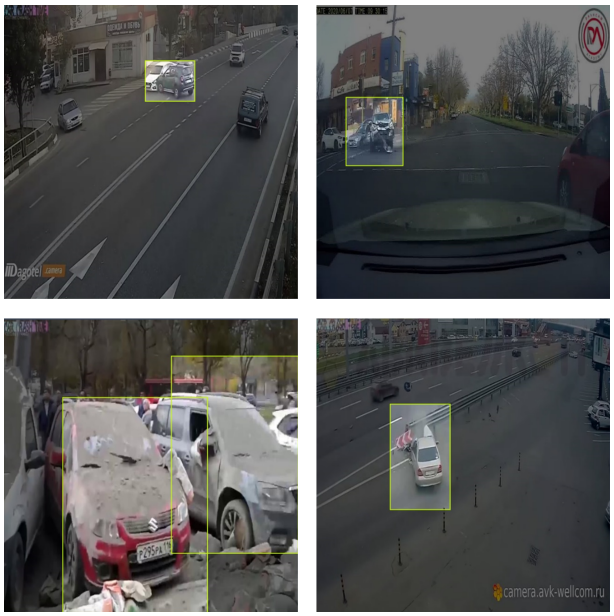

Figure 4.7: Example images from the dataset used to fine-tune the pretrained *YOLOv8s Detect model*.

trained model. More information about that can be found in Chapter 4.3.4.

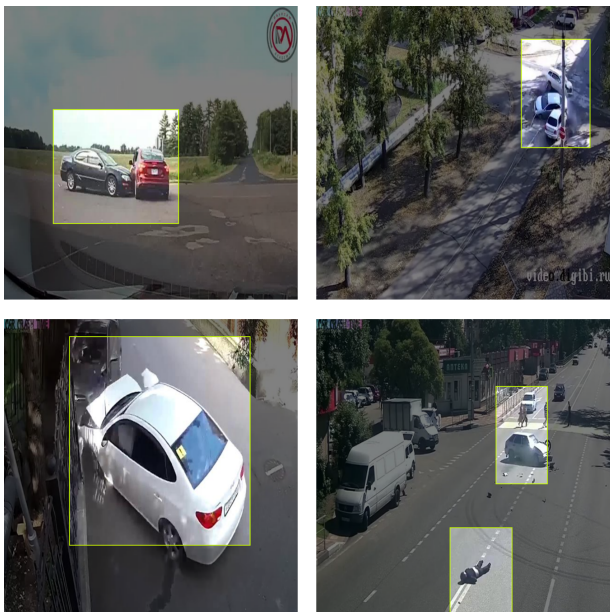

Figure 4.8: Example images from the dataset used to fine-tune the pre-trained *YOLOv8s Detect model* with imprecise bounding boxes.

On a high level, the learning-based accident detection model works as follows: First, an image, for which the accident detection should be applied, is fed into the accident detection model. Then the forward pass of the model is performed which applies the accident detection to the input image. Afterwards, the detection result is returned. It contains a range of information, including the label, the confidence for this label, and the coordinates of the bounding box. The whole process is illustrated in Figure 4.9.

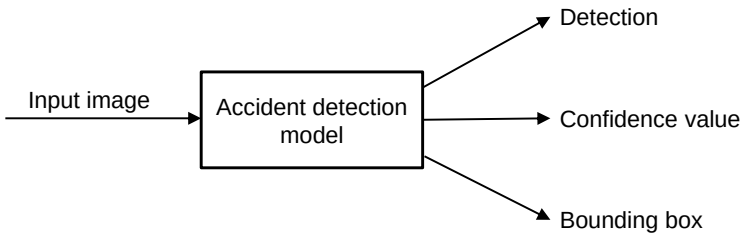

Figure 4.9: Accident detection process using an accident detection model.

4.3.2 Integration

To be able to use the pre-trained accident detection model on the sensor data of the A9 test stretch, it was integrated into the scenario detection. The reason for this, as already mentioned, is that accidents are one of the many scenarios that should be detected by scenario detection. By setting the `-acc` argument of the scenario detection to *ml* the learning-based accident detection is used for the scenario detection instead of the rule-based one.

As the core of the learning-based accident detection is a pre-trained YOLOv8 model, images are required as input. However, the recordings from the A9 test stretch are, as already mentioned, stored as rosbags. Therefore, the images stored in the rosbags must be extracted to feed them into the accident detection model. This is realized by using the image extractor node that is also used by the image extraction feature of the rule-based accident detection. The image extraction step is initiated by scenario detection. For a given set of rosbags, the images from all cameras available in the respective rosbags are extracted. Extracting and saving all the images at once is done because it is the computationally least expensive way to obtain all the images from a recording. The reason for this is that single images can not efficiently be extracted on their own as explained in chapter 4.2.3. After the image extraction, the learning-based accident detection iterates over all extracted images and feeds every single one of them into the accident detection model. This model then performs the actual accident detection. Afterward, its output is processed by the *accident detection handling module*. This module has several purposes. On the one hand, it performs several checks to increase the probability of making correct detections. How this works is described in detail in Chapter 4.3.3. On the other hand, it handles the output of the accident detection model. Three different types of output are created when the learning-based accident detection is applied. First, the amount of accidents in the input data is counted and saved in a JSON file that stores statistics of detected scenarios. Secondly, images of the detected accidents are created that include bounding boxes, score values for detections, and the respective labels. Finally, a CSV file is created that stores information about each detected accident, including its timestamp. As a final step all extracted images of a rosbag are removed again after they were fed into the accident detection model. The whole process of the learning-based accident detection process is illustrated in Figure 4.10.

As the accident detection model is applied to every single input image it provides an output whether there is an accident or not for each of the images. This means that if an accident happens, the accident detection will detect an accident in a whole sequence of consecutive frames. Nevertheless, this is only treated as one accident and therefore only increases the accident counter by one. Moreover, it also only creates one new entry in the CSV file.

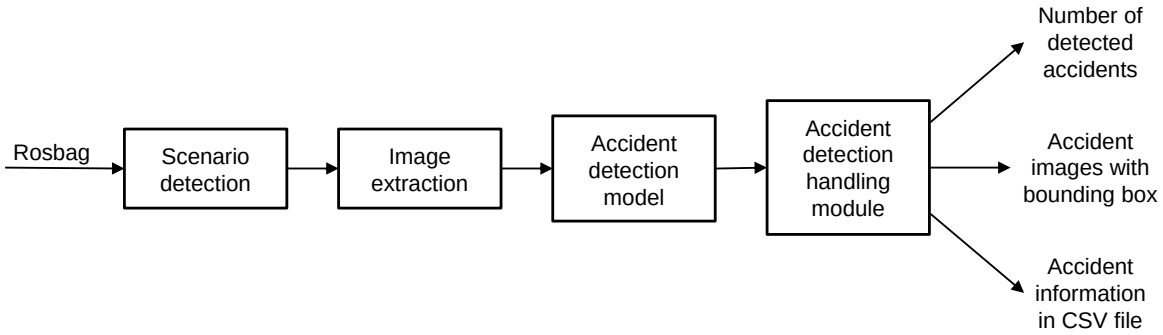

Figure 4.10: Learning-based accident detection process.

4.3.3 Improvement

To improve the performance of learning-based accident detection, several measures were taken. First of all, the pre-trained accident detection model was replaced by a self-trained model. To obtain this model a custom dataset was created with images that are closer to the use case of the A9 test stretch. Furthermore, this dataset is almost three times larger than the one used to train the pre-trained model. This custom dataset is described in detail in Chapter 4.4. Using this dataset, the YOLOv8x model, which was pre-trained on the COCO dataset, was fine-tuned. More information about the newly trained model can be found in Chapter 5.3.

Additionally, the confidence value of a detection result, which must be achieved as a minimum to be detected as an accident, has been adjusted. It was increased from 0.25 to 0.8 as the experiments described in Chapter 5.2 suggested this to be the most promising value for the A9 test stretch.

To further reduce the number of incorrectly detected accidents, an additional check has been added. The accident detection model now has to detect an accident in at least three consecutive timestamps to treat it as an actual accident. This can be done as the input data corresponds to the frames of a video in chronological order. So if an accident occurs, it can be seen in a whole sequence of images and not only in a single image. Furthermore, many of the images used for training the accident detection model were taken after the accident happened and not in the exact moment in which the accident happened. Consequently, the accident detection model can not distinguish between the moment an accident happened and the time after the accident in which the accident vehicles are still close together. Therefore, an accident should always be detected in more than just three images if it is an actual accident. If the condition with three consecutive frames is not met, the detected accident will be ignored. This idea is illustrated in Table 4.3 for a sample scenario. A zero value in one of the result rows in the table means that the respective component thinks there is no accident at this timestamp. Accordingly, a value of one means that the respective component thinks that there is an accident at this timestamp. For the sample values in the table, the accident detection model detects an accident for timestamp two. However, no accident is detected for the timestamps one and three. Therefore, the detection result for timestamp two was probably wrong and should consequently not be considered as an accident. The described check for three consecutive timestamps with a detected accident ensures exactly that. Therefore, the accident detection handling module result for timestamp two is zero and not one.

As already mentioned, accidents are detected for a whole sequence of timestamps. How-

Timestamp	1	2	3	4	5	6	7
Model result	0	1	0	0	1	1	1
Handling module result	0	0	0	0	**1**	1	1

Table 4.3: Conversion of model results to handling module results for a sample scenario. It can be seen that the model result for timestamp two is one, whereas the handling module result for this timestamp is zero. The reason for this is that the detection for timestamp two is not part of a sequence of three consecutive timestamps with a detected accident and is therefore filtered out.

ever, the actual moment at which an accident occurs corresponds to just a single timestamp. Therefore, the relevant timestamp for the accident detection is the one in which the accident has been detected for the first time. In the example given in Table 4.3 this is timestamp five. This timestamp is referred to as *accident timestamp* from now on. For all detected accidents, the accident timestamp is written to a CSV file, together with the name of the file in which the accident was detected and the ID of the camera that took the respective image.

As there are usually several cameras providing images at the same time, the information of all the available cameras is fused. However, if an accident is detected in the images of more than one camera, it should be treated as a single accident and not as multiple ones. This idea is realized by the accident detection handling module. It fuses the detected accidents of all cameras for each timestamp, also taking into account that not every camera records at the same frequency. For figuring out the accident timestamp, the fused results are used by taking the accident timestamp for which the accident has been detected first. This procedure is illustrated in Table 4.4, again using a sample scenario.

Timestamp	1	2	3	4	5	6	7
Result camera 1	0	0	0	0	1	1	1
Result camera 2	0	0	1	1	1	1	1
Result camera 3	0	0	0	0	0	0	0
Result camera 4	0	0	0	0	0	0	0
Combined detection result	0	0	**1**	1	2	2	2

Table 4.4: Fusion of accident detection results of multiple cameras for a sample scenario. It can be seen that the accident timestamp for this scenario is three, as this is the first timestamp at which the accident is detected in the images of any of the four cameras.

For the sample scenario used in Table 4.4, the accident timestamp is three as this is the first timestamp at which the accident is detected in the images of any of the four cameras. Although the accident is detected in the images of two of the cameras, the total number of detected accidents is only increased by one as camera one and camera two detected the same accident.

4.3.4 Model Training

To find the best-performing learning-based accident detection model for the A9 test stretch, five different models were trained. Furthermore, also the pre-trained accident detection

model, mentioned in Chapter 4.3.1 was considered. With that, six learning-based accident detection models exist, which differ in the used dataset, the resolution of the input images, the batch size used for training, and the number of epochs for which they were trained. Table 4.5 gives an overview of these six models.

Name	Description	Image size	Batch size	#Epochs
m1	Pre-trained model	640	16	22
m2	Fine-tuned model	1920	6	63
m3	Fine-tuned model	1600	8	80
m4	Fine-tuned model	1280	16	120
m5	Fine-tuned model	960	8	106
m6	Fine-tuned model	640	32	67

Table 4.5: Overview of the six learning-based accident detection models.

The pre-trained accident detection model was created by Shubhankar Shandilya and was published in his accident detection model GitHub repository [20]. This model was created, as already mentioned in Chapter 4.3.1, by fine-tuning the YOLOv8s detection model using an image size of 640 pixels and a batch size of 16. It is called m1 from now on.

Models m2 to m6 were all created by fine-tuning the YOLOv8x detection model, pre-trained on the COCO dataset, with a custom self-labeled dataset. This dataset is introduced in Chapter 4.4. Each model was trained on one or multiple NVIDIA GeForce RTX 3090 GPUs with 24 GB of VRAM. How many GPUs have been used for training varies from model to model depending on the availability of the GPUs during the training process. The image size used for the different models is 1920 pixels (m2), 1600 pixels (m3), 1280 pixels (m4), 960 pixels (m5) and 640 pixels (m6). The training process of each of the models was started for 300 epochs. However, none of them trained for the full 300 epochs as *early stopping* was used which always made the training process terminate beforehand. The exact number of epochs, for which each of the six models was trained, can be found in Table 4.5. For each training process, the largest possible batch size for the number of available GPUs and the resolution of the training images have been used. The exact batch size for each of the models is shown in Table 4.4.

4.4 Custom Dataset

To improve the performance of the learning-based accident detection, a custom dataset was created. As a baseline, it uses version two of the dataset, published by the author of the pre-trained accident detection model mentioned in Chapter 4.3.1. This dataset consists of 2,517 training images, 371 validation images, and 262 test images which makes 3,250 images in total. These images all have a resolution of 640x640 pixels. However, not all of the images in this dataset contain accidents. There are accidents present in 1,321 of the training images, 244 of the validation images, and 176 of the test images. Figure 4.11 shows some example accident images from this dataset. As can be seen, the dataset consists of images taken from different perspectives. Many of the images are taken by elevated roadside cameras and therefore show the accidents from a steep elevated view. Additionally, there are also images taken by roadside cameras on the ground or by dashcams which results in images showing

the accidents from a roadside view or a vehicle view. While many of the images show the whole street, including the accident, this is not the case for all images. Some of them only show the accident itself without providing any further context. So, not all the images of this dataset match the type of images taken by the A9 test stretch. However, most of them do. Furthermore, no larger accident dataset has been found that is so similar to the A9 test stretch and also provides labeled bounding boxes for the accidents.

Figure 4.11: Example images from version two of the dataset, published by the author of the pre-trained accident detection model mentioned in Chapter 4.3.1.

To further increase the size of the dataset and to make it more similar to the images taken on the A9 test stretch, for which the accident detection is developed, accident images from the A9 test stretch have been added. They all have a resolution of 1920x1200 pixels. Unlike the downloaded dataset, all the self-labeled images contain an accident. In total, 475 accident images were extracted from the four accidents in the event log presented in Chapter 4.1. Table 4.6 shows how many images have been used from each of the accidents.

Accident date	Train	Validation	Test	Total
04/08/2021	53	7	7	67
10/21/2021	125	16	16	157
03/28/2022	121	15	15	151
05/22/2022	80	10	10	100

Table 4.6: Number of self-labeled images from the four known accidents.

After extracting these images, they have been labeled with a bounding box for the accident. Some examples from these self-labeled images can be seen in Figure 4.12.

The self-labeled accident images were manually split using an 80-10-10 split which resulted in 379 training images, 48 validation images, and 48 test images. The split in training, validation, and test set has for all four accidents been made in a way that the images of each

Figure 4.12: Example images from the self-labeled accident images.

of the splits are consecutive frames of one or two short sequences of the respective accident. As the custom dataset combines the images from the mentioned published accident dataset and the self-labeled images, it contains 3725 images in total. 2896 of them are part of the training set, 419 of the validation set, and 410 of the test set. Table 4.7 gives an overview of the described composition of the custom dataset.

Dataset	Train	Validation	Test	Total
Downloaded	2,517 (1,321)	362 (176)	371 (244)	3,250 (1,741)
Self-labeled	379 (379)	48 (48)	48 (48)	475 (475)
Total	2,896 (1,799)	410 (224)	419 (292)	3,725 (2,216)

Table 4.7: Number of images of the custom dataset. The number in brackets is the number of accident images.

Chapter 5

Experiments and Results

Extensive experiments were carried out on various datasets to do an empirical analysis of the learning-based accident detection and to evaluate the performance and the runtime of the rule-based and the learning-based accident detection. Based on the results of these experiments, the two accident detections are compared and the two research questions are answered.

5.1 Evaluation Metrics

For evaluating and comparing the rule-based and the learning-based accident detection several metrics have been used. The basic components used by the metrics measuring the accuracy of the accident detection are:

- the number of *True Positives* (TP) which describes the number of correctly classified accidents [7, page 91].
- the number of *False Positives* (FP) which describes the number of wrongly classified accidents [7, page 91].
- the number of *False Negatives* (FN) which describes the number of accidents wrongly classified not to be an accident [7, page 91]. FN therefore corresponds to the number of accidents that have not been detected.

To answer the research question, of whether it is possible to reliably detect accidents on the A9 test stretch using roadside sensors, it is necessary to measure the accuracy of the accident detection. This can be done by using the metrics *precision* and *recall*. The precision measures how many of the accidents detected actually are accidents [7, page 91].

$$precision = \frac{TP}{TP + FP} \tag{5.1}$$

It is often used in combination with *recall*, which is also called *sensitivity* or *true positive rate*. It describes the ratio of accidents that are correctly detected by accident detection [7, page 91]. Consequently, it measures how many of the actual accidents have been detected by the accident detection.

$$recall = \frac{TP}{TP + FN} \tag{5.2}$$

An even more accurate way to measure the accuracy of accident detection is by comparing the detected accidents and their bounding boxes with the labels of the corresponding images. The metric used for this is the *mean Average Precision* (mAP). It is calculated by using the *Average Precision* (AP) of the accident class. For figuring out the AP, the maximum precisions with at least 0% recall, 10% recall, 20% recall, and so on until 100% recall are calculated. The AP is then the mean of all the calculated maximum precisions. To ensure that the size and location of the bounding boxes are predicted correctly, the *Intersection over Union* (IoU) is considered for the computation of the mAP. As Figure 5.1 shows, the IoU is the intersection of the predicted and real bounding boxes divided by their union. The mAP is noted as mAP@0.5 if a predicted accident should be considered as correct as soon as the IoU is greater than 0.5. If different IoU thresholds should be taken into account, the mean achieved with all the considered IoU values is computed. For example, for the IoU values 0.5, 0.55, 0.60, ..., 0.95, the mean of the mAP for each mentioned IoU value is taken. This is then noted as mAP@[.50:.95] [7].

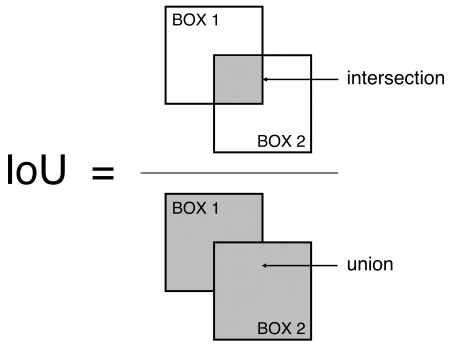

Figure 5.1: Visualization of the formula for the IoU (*BOX 1* represents the ground truth bounding box and *BOX 2* the predicted one).

5.2 Empirical Analysis of Learning-Based Accident Detection

To obtain the best-performing learning-based accident detection model for the A9 test stretch, an empirical analysis was carried out.

Model Selection

First of all, the six different models presented in Chapter 4.3.4 have been compared regarding the accident detection accuracy they achieve. Figure 5.2 and 5.3 provide a visual comparison of the accident detection results for two example images obtained by the six accident detection models. For both of these images, a confidence threshold of 0.25 has been used as this is the default value of yolov8.

For Figure 5.2 it can be seen that m2, m4, and m5 correctly detect the accident. Model m3 also detects the accident but as two accidents instead of just one. The models m1 and m6 do not detect the accident. Consequently, m2, m4, and m5 are the best-performing models on the given accident image. It is noticeable that the predicted bounding boxes are pretty

m1 (640 pixels) **m2** (1920 pixels) **m3** (1600 pixels)

m4 (1280 pixels) **m5** (960 pixels) **m6** (640 pixels)

Figure 5.2: Visual comparison of the accident detection results (purple) for a test image with a labeled accident (turquoise) obtained by the six accident detection models. It can be seen that m2, m4, and m5 correctly detect the accident. Model m3 also detects the accident but as two accidents instead of just one. The models m1 and m6 do not detect the accident.

similar to the ground truth bounding boxes. This is probably because the images in the training data and the test data do not deviate much as they are frames from the same accidents.

Figure 5.3 shows the results of the six accident detection models on an example image that does not contain an accident. It is noticeable that most of the models correctly do not detect an accident in it. However, m1 and m3 detect one and therefore have a false positive detection for the example image. Consequently, the best-performing models for this image are m2, m4, m5 and m6.

m1 (640 pixels) **m2** (1920 pixels) **m3** (1600 pixels)

m4 (1280 pixels) **m5** (960 pixels) **m6** (640 pixels)

Figure 5.3: Visual comparison of the accident detection results (purple) for a test image without an accident obtained by the six accident detection models. It can be seen that most of the models correctly do not detect an accident in it. However, m1 and m3 detect one and therefore have a false positive detection for the example image.

Based on the two example images, it seems like m2, m4, and m5 are the accident detection models that enable the highest accident detection accuracy. To be able to draw a general conclusion, the accident detection accuracy achieved by the six accident detection models was measured. This was done by applying all the models to the test set of the custom dataset and measuring what mAP they achieved on it. By doing this experiment it is evaluated how well the accident detection models work, especially in comparison to the pre-trained model m1. Table 5.1 presents the computed mAP@0.5 and mAP@[.50:.95] for the 419 test images from the custom dataset. The results are visualized in Figure 5.4. For measuring the mAP the image size used for training a model has also been used for the inference step of the corresponding model.

Model	mAP@0.5	mAP@[.50:.95]
m1	0.385	0.173
m2	0.699	0.410
m3	0.818	0.479
m4	0.906	**0.548**
m5	**0.917**	0.537
m6	0.875	0.532

Table 5.1: Comparison of the mAP@0.5 and the mAP@[.50:.95] achieved by applying the six learning-based accident detection models to the custom test set mentioned in Chapter 4.4. For each of the models, the input images have been rescaled to the image size the model has been trained with. The experiment shows that m5 achieves the highest mAP@0.5 and the second highest mAP@[.50:.95]. As the exact location of the bounding box is not as important as whether the accident is detected at all, the mAP@0.5 value is more important in the use case of accident detection. Therefore, m5 is the model with the highest accident detection accuracy.

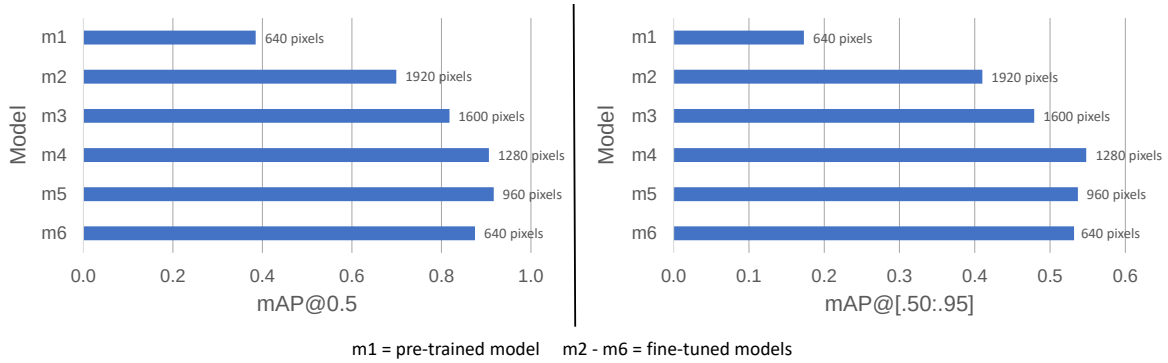

Figure 5.4: Visualization of the mAP@0.5 and the mAP@[.50:.95] achieved by applying the six learning-based accident detection models to the custom test set. It is noticeable that m5 achieves the highest mAP@0.5 and the second highest mAP@[.50:.95]. As the mAP@0.5 is more important for the use case of an accident detection than the mAP@[.50:.95], m5 is the model with the highest accident detection accuracy.

As it can be seen in Figure 5.4, m5 achieves the highest mAP@0.5 and the second highest mAP@[.50:.95]. Because the exact location of the bounding box is not as important as whether the accident is detected at all, the mAP@0.5 is more important than the mAP@[.50:.95] for the use case of accident detection. Therefore, m5 is the best-performing model for the custom test set regarding accident detection accuracy. Model m4 is the second best model as it achieves the second highest mAP@0.5 and the highest mAP@[.50:.95]. A slightly worse accident detection accuracy is achieved by m6. Model m2 already performs significantly worse than the other models. The worst-performing model is the pre-trained

model m1. The mAP@0.5 achieved by it corresponds to just 42% of the mAP@0.5 achieved by m5, and the mAP@[.50:.95] achieved by it corresponds to just 32% of the mAP@0.5 achieved by m5. This means that creating the custom dataset and fine-tuning a pre-trained model with it was beneficial in terms of accident detection accuracy.

In summary, the qualitative results, as well as the quantitative results achieved on the custom test set, suggest model m5 to be the best-performing accident detection model for the A9 test stretch as it achieves the highest accident detection accuracy on the test set. By looking at the results presented in Table 5.1 it could be assumed that m5 is just the best-performing model because it was trained using an image size of 960 pixels. As a result, the image size used to train m5 is closer to 640 pixels, which the majority of the training images have as width and height, than the image size used to train the models m2 to m4. However, an additional experiment has shown that this assumption is not true. For this experiment, the accident detection accuracy of m2 and m5 was compared only on the A9 images from the custom test set which all have a resolution of 1920x1200 pixels. As m2 was trained using an image size of 1920 pixels, it was chosen for this comparison. For comparing the accident detection accuracy, the mAP@0.5 and the mAP@[.50:.95] have again been used. The results of this experiment show that m5 performs better than m2 even on images with a high resolution. Both models achieve a mAP@50 of 0.995. However, m5 achieves a mAP@[.50:.95] of 0.881 whereas m2 only achieves a mAP@[.50:.95] of 0.863. This shows that training with a smaller image size and using a smaller image size for the inference step does not decrease the accident detection performance on the A9 test stretch but actually even improves it a bit. Therefore, the smaller image size should not lead to m5 performing worse on images from the A9 test stretch.

As m5 was identified as the best-performing accident detection model for the A9 test stretch, it was chosen to be used for learning-based accident detection. Therefore, it is also the only model used for further experiments. Using m5 with an image size of 960 pixels also has the advantage that the inference time is lower than if m2 to m4 were used with the image size the respective model was trained on.

Confidence Threshold

As mentioned in Chapter 4.3.3, the accident detection accuracy can further be improved by adjusting the confidence threshold used for detecting accidents. Therefore, the achieved accident detection accuracy using different confidence thresholds has been analyzed for the accident detection model. For this experiment, several rosbags have been selected from the event log covering normal traffic, a traffic jam, a vehicle transporter, a standing shoulder event, and four different accidents. The exact rosbags used for this are listed in Table 5.2. From the accident rosbags, only the time period covered by the custom training and validation set was used. This is done to avoid already using the remaining parts of the accident rosbags for adjusting the parameters of the learning-based accident detection. This way the parts of the accident rosbags are still unknown to the learning-based accident detection and are therefore suited for comparing the learning-based and the rule-based accident detection.

Table 5.3 presents the results of the described experiment with different confidence thresholds. It can be seen that with a confidence value of 0.8 and 0.9 a precision of 1.0 is achieved. Therefore, all these models only detected actual accidents as accidents on the described test data. Furthermore, the model achieves a recall of 1.0 regardless of whether the confidence

Date	Start Time	Event
10/13/2020	18:52:23.25	Traffic jam
05/15/2021	16:44:01.74	Standing shoulder
05/15/2021	16:54:01.73	Vehicle transporter, standing shoulder
05/15/2021	16:55:01.74	Vehicle transporter, standing shoulder
05/15/2021	17:04:01.74	Standing shoulder
07/29/2021	19:44:24.15	Standing shoulder
04/08/2021	11:31:00.93	Accident
10/21/2021	10:21:44.36	Accident
03/28/2022	17:19:15.35	Accident
05/22/2022	17:14:43.38	Accident

Table 5.2: Information about test data used for the experiment regarding the best confidence threshold for the learning-based accident detection. For the accident rosbags only the time periods covered by the custom training and validation set were used.

value is 0.5, 0.6, 0.7, 0.8, or 0.9. The reason for this is that the model did not miss any accidents in the test data and therefore has zero false negative detections, independently of the used confidence threshold. Depending on the confidence value a different number of accidents is detected. For the tested confidence values the number of detected accidents varies between five and sixteen, whereas the optimal value would be four, as there are a total of four accidents in the test data. The reason for the too high number of detected accidents is that the model sometimes detects an accident, then not detects the accident for a few timestamps, and then detects it again. Consequently, it counts the accidents as two accidents instead of one. These are not false positive detections as the detected accident still is a real accident.

Confidence threshold	Precision	Recall	#accidents
0.5	0.19	**1.0**	**5**
0.6	0.36	**1.0**	**5**
0.7	0.5	**1.0**	6
0.8	**1.0**	**1.0**	8
0.9	**1.0**	**1.0**	16

Table 5.3: Comparison of the precision, recall, and number of detected accidents achieved by the learning-based accident detection on some test data from the A9 test stretch, depending on the used confidence threshold. While a value of 1.0 is the best achievable value for precision and recall, a value of four would be the best result for the number of detected accidents. It can be seen that with a confidence value of 0.8, the highest precision and recall and the third lowest number of detected accidents are achieved. As high precision and recall are more important than having a low number of detected accidents for the use case of accident detection, a confidence threshold of 0.8 leads to the best-performing learning-based accident detection.

Investigating the experiment results has revealed two reasons why the number of detected accidents is too high. First of all, the accidents are sometimes obscured in images by another vehicle, most of the time by trucks. Therefore, it is not visible for a few frames and can consequently not be detected in them by learning-based accident detection. However, the respective accidents are detected again when not being obscured anymore. As a result, the total number of detected accidents is increased again, although it is still the same accident being detected. The other reason found is that the accident with the burning vehicle is not detected constantly and therefore leads to multiple detections.

Overall, using a confidence value of 0.8 seems to be the most promising configuration as this is the confidence value with which the highest precision and the highest recall are achieved. That with this configuration only the third lowest number of accidents is achieved is not so problematic because precision and recall are far more important metrics for the use case of accident detection than the number of detected accidents. After all, detecting actual accidents twice and therefore having a too high number of detected accidents is better than having some false positive detections as it would be the case with a confidence value of 0.7 or lower.

As a confidence value of 0.8 was identified as the confidence threshold that leads to the best-performing accident detection model for the A9 test stretch, it is chosen to be used for learning-based accident detection. Therefore, this confidence value is used for all further experiments.

5.3 Accuracy

To answer the research question of whether it is possible to reliably detect accidents on the A9 test stretch using roadside sensors, the reliability of the two implemented accident detections was investigated. This was done by measuring the accuracy of the accident detection for both implemented accident detection approaches. The metrics used for measuring the accident detection accuracy are precision and recall. They are used instead of the mAP as the rule-based accident detection does not provide a bounding box for the location of a detected accident. Therefore, the mAP achieved by rule-based accident detection can not be calculated. However, precision and recall are also suited for describing the detection accuracy, especially in the use case of accident detection where only the timestamp of an accident and not the exact locating of an accident in the image is relevant. If the implemented accident detections achieve a precision greater than 0.7 and a recall greater than 0.85, they are considered reliable. The recall value to be achieved is higher than the precision value as it is more important to not miss actual accidents than to not detect too many incorrect accidents.

For measuring precision and recall for the two accident detections, several rosbags have been selected as test data. They were chosen from the event log and cover a wide range of special events as well as the four known accidents. As all the rosbags in the event log have been manually inspected and classified regarding special events such as accidents, it is known for each of them whether there is an accident in it or not. Consequently, they can be used as test data. The exact rosbags used for this experiment are listed in Table 5.4. For the accident rosbags only the time period that is not included in the custom training or validation set has been used.

Measuring the accident detection accuracy was done on the event level. Therefore, the number of accidents detected by the respective accident detection was measured and not the number of timestamps for which an accident has been detected. This is done because the goal of accident detection is to provide all detected accidents, not the number of timestamps for which an accident was detected.

Table 5.5 presents the results of applying the rule-based accident detection and the learning-based accident detection to the described test data. It can be seen that rule-based accident

Date	Start time	Event
02/11/2021	11:32:00.93	Vehicle transporter
04/08/2021	11:31:00.93	Accident
04/08/2021	11:31:00.93	Accident
05/15/2021	15:40:01.73	*No event*
05/15/2021	15:52:01.74	Standing shoulder
07/30/2021	09:25:24.07	Traffic jam
10/21/2021	10:21:44.36	Accident
03/28/2022	17:19:15.35	Accident
05/22/2022	17:14:43.38	Accident

Table 5.4: Information about the test data used for measuring the accuracy of the rule-based and the learning-based accident detection. For the accident rosbags only the time periods not used for the custom training or validation set have been considered.

detection achieves a higher precision whereas learning-based accident detection achieves a higher recall. The precision achieved by the rule-based accident detection is so high because not a single false positive detection is made on the test set. However, rule-based accident detection has the major disadvantage that it only detects two of the four accidents. The learning-based accident detection, on the other hand, detects every single accident in the test data. As a result, it achieves a recall of 1.0. The reason for the lower precision is that the learning-based accident detection also makes some false positive detections. Overall, the learning-based accident detection achieves better results regarding precision and recall and therefore achieves a higher accident detection accuracy on the A9 test data. Consequently, learning-based accident detection is more suited for detecting accidents on the A9 test stretch.

Accident detection approach	Precision	Recall
Rule-based	**1.0**	0.5
Learning-based	0.8	**1.0**

Table 5.5: Comparison of precision and recall achieved by rule-based and learning-based accident detection on some test data from the A9 test stretch. It can be seen that rule-based accident detection achieves a higher precision whereas learning-based accident detection achieves a higher recall. However, the learning-based accident detection overall achieves better results regarding precision and recall and therefore achieves a higher accident detection accuracy on the A9 test data.

All in all, it can be seen that rule-based accident detection does not fulfill the mentioned criteria for reliability, whereas learning-based accident detection fulfills them. Consequently, it can be concluded that the accident detection reliable detects accidents in the test data recorded by the roadside sensors of the A9 test stretch. Therefore, the research question of whether it is possible to reliably detect accidents on the A9 test stretch using roadside sensors can be answered with 'yes'. In fact, 100% of the accidents in the test data have been detected by the implemented accident detection. Furthermore, there were almost no false positive detections made in the experiment.

However, it is not possible to say with certainty whether the implemented accident detection also works so well on new accidents. The reason for this is that the test set contains only four accidents, which means that a general statement can only be made to a limited extent. However, there are no other daytime accidents known in the data of the A9 test

stretch. Therefore, it was not possible to carry out a more comprehensive test regarding the accident detection accuracy on the A9 test stretch. Consequently, it is just not possible with the existing data to verify the generalization capability of the accident detection model for the A9 test stretch. Nevertheless, the overall generalization capability of the learning-based accident detection has been shown by the experiment described in Chapter 5.2. In this experiment, the final accident detection model achieved a very high mAP@0.5 on the custom test set which not only includes a few accident images from the A9 test stretch but also many additional accident images from the downloaded dataset. The latter also includes many individual images that are not part of a sequence which is also partially included in the training data. As the high mAP@0.5 indicates that the learning-based accident detection also works well for the latter part of the test set, it is shown that the model can generalize to new, unseen data. An additional experiment on a crash compilation on YouTube confirmed that new, unseen accidents can be detected if the accident images at least partially match the setting of the training data.

Concerning the different cameras on the freeway section of the A9 test stretch, no trend can be seen in the test data as to whether one of the cameras detects accidents better. Each of the four accidents is detected by only one of the cameras, namely by the camera that takes the images in which the accident can best be seen. Therefore, it seems like the camera does not influence the accident detection accuracy.

When investigating the false positive detections of the learning-based accident detection with a lower confidence threshold than now used, it is noticeable that all of them correspond to the same situation, namely a vehicle that is not fully visible and drives on the right side of the freeway in the northern direction. Furthermore, the respective vehicle is always pretty much at the same location. The reason for this could be that some accident images in the custom training and validation set only contain parts of a vehicle as the remaining part of the vehicle is either hidden by another vehicle or not visible in the image. Figure 5.5 shows an example image for such a case. However, there are not many images that correspond to this situation in the custom dataset and not a single one where there is an accident vehicle at the respective location that is not fully visible in the image. In the end, this problem is not relevant because almost all of these false positive detections only occur with a confidence threshold lower than 0.8. Consequently, the final learning-based accident detection is barely affected by it.

Figure 5.5: Potentially problematic sample image from the custom training set (left) that could maybe cause the problem of detecting only partly visible vehicles at a specific location on the freeway section of the A9 test stretch as an accident (right).

To further investigate the accident detection accuracy of the rule-based accident detection, it was applied to a bunch of recordings from the A9 test stretch. Therefore, a complete

pipeline for investigating the recordings from the A9 test stretch was implemented. It automatically downloads the recordings bit by bit, performs a pre-selection of potential accident rosbags, applies the rule-based accident detection on it, and creates multiple outputs like extracted accident images and a CSV file that includes information about all detected accidents. The experiment using this pipeline already found one new accident. This shows that the implemented pipeline works and can therefore be used for looking for accidents in the recordings from the freeway section of the A9 test stretch. More information about this pipeline and the results of the experiment using it can be found in Marc Pavel's documentation.

5.4 Runtime Analysis

Apart from the accident detection accuracy also the runtime of the two accident detections has been analysed. This was done to answer the research question of whether it is possible to detect accidents on the A9 test stretch in real time using roadside sensors. To achieve this in real time the runtime of the accident detection needs to be low enough to process the input data in real time, which has 25 frames per second. This means that the runtime of the accident detection for one second of a recording must be lower than one second and the one for processing one minute of a recording must be lower than one minute.

The runtime of the accident detections was investigated by measuring the time taken to perform the accident detection on some test data. The data used for this are two one-minute rosbags from the 8th of April 2021 (from 11:30:01 to 11:32:01) and two one-minute rosbags from the 11th of May 2022 (from 16:14:43 to 16:15:43 and from 16:29:43 to 16:30:43). The former contains the data from two cameras, the latter the data from four cameras. It was not necessary to also differentiate between test data with and without an accident, as this does not have a significant impact on the runtime for either approach. The runtime analysis was performed on an Intel Core i9-9900KF with 3.60 GHz and an NVIDIA GeForce RTX 2080 SUPER GPUs with 8 GB of VRAM. While the rule-based accident detection uses only the CPU, the learning-based approach uses the GPU for the inference step and therefore uses both the CPU and the GPU.

Table 5.6 presents the results of the runtime analysis achieved by the rule-based and learning-based accident detection on the mentioned test rosbags. The runtime was measured in seconds and is given for an entire one-minute rosbag and for a single second of a one-minute rosbag. It can be seen that the rule-based accident detection is much faster than the learning-based accident detection. If two cameras are available in a rosbag, rule-based accident detection is 47 times as fast as learning-based accident detection. For four available cameras, it is even 63 times as fast as the learning-based approach. Furthermore, rule-based accident detection is independent of the number of cameras that provide images for the investigated time period. However, its runtime depends on the amount of vehicles present in the sensor data. In contrast, the runtime of the learning-based accident detection is proportional to the number of cameras provided for the investigated time period and independent of the number of vehicles visible in the images. The utilization of the NVIDIA GeForce RTX 2080 SUPER GPU in this experiment was almost 50% with a memory usage of 1.9 GB.

As the runtime of the rule-based accident detection depends on the number of vehicles present in the sensor data, the measured runtime analysis results for the rule-based accident

	Runtime with 2 cameras (s)		Runtime with 4 cameras (s)	
Accident detection approach	Per second	Per minute	Per second	Per minute
Rule-based	**0.086**	**5.173**	**0.127**	**7.614**
Learning-based	4.072	244.298	7.995	479.689

Table 5.6: Comparison of the runtime achieved by rule-based and learning-based accident detection on accident rosbags. It can be seen that the rule-based accident detection is much faster than the learning-based accident detection.

detection are not fully representative. However, they still show the order of magnitude and that rule-based accident detection is much faster than learning-based accident detection.

Overall, it can be seen that rule-based accident detection fulfills the mentioned criteria for real-time capability, whereas learning-based accident detection does not fulfill it. Consequently, it can be concluded that accidents can be detected in real-time in the test data recorded by the roadside sensors of the A9 test stretch. Therefore, the research question of whether it is possible to reliably detect accidents on the A9 test stretch using roadside sensors can be answered with 'yes' as well. In fact, rule-based accident detection is more than seven times faster than what is required for real-time usage.

A significant part of the runtime of the learning-based accident detection is required for extracting the images. More specifically, it takes 0.853 seconds for one second from a recording of two cameras and 1.658 seconds for one second from a recording of four cameras. This means that almost 21% of the runtime of the learning-based accident detection is spent extracting the images, meaning preparing the input data.

5.5 Comparison

To conclude, the results presented in Chapter 5.3 and 5.4 are summarized in Table 5.7. Based on that, the rule-based and the learning-based accident detection are compared in a clear and concise manner.

	Accuarcy		Runtime per second (s)	
Accident detection approach	Precision	Recall	2 cameras	4 cameras
Rule-based	**1.0**	0.5	**0.086**	**0.127**
Learning-based	0.8	**1.0**	4.072	7.995

Table 5.7: Comparison of the accuracy and runtime of the rule-based and learning-based accident detection on some test data from the A9 test stretch. The accident detection accuracy is compared based on the achieved precision and recall. For the runtime comparison, the runtime for one second of a recording is given, depending on whether there are images of two cameras or four cameras available in a rosbag. It can be seen that learning-based accident detection achieves a higher accident detection accuracy while rule-based accident detection achieves a much lower runtime.

As it can be seen in Table 5.7 the learning-based accident detection achieves a similar precision and a much higher recall than the rule-based accident detection. Therefore, learning-

based accident detection achieves a higher accident detection accuracy. In terms of runtime, the rule-based accident detection performs much better than the learning-based accident detection, as it is many times faster. This means that both implemented accident detections have their advantages and disadvantages. While the rule-based accident detection will detect fewer actual accidents than the learning-based accident detection, it is much faster which makes it more suitable for applying it to huge amounts of data like the recordings from the A9 test stretch. The learning-based accident detection, on the other hand, is more suited if it is important to detect as many accidents as possible. However, none of the approaches makes it possible to achieve both, reliable and real-time detection at the same time. Therefore, there is a trade-off between accident detection accuracy and runtime of an accident detection. As a result, it has to be decided on a case-by-case basis which of the two approaches is better suited for a given use case.

Again, the presented accuracy results used for the comparison are only generally valid to a limited extent as there were only four accident recordings from the A9 test stretch available for computing these metrics. However, the general trend likely is the same as the results of the two approaches differ drastically and the setup for new data from the A9 test stretch does not change compared to the setup used for recording the custom test set.

Chapter 6

Future Work

There are several tasks that can be worked on to further improve the performance of the accident detection implemented for the freeway section of the A9 test stretch.

First, the learning-based accident detection could be extended by also detecting pedestrians. As the accident detection was developed for the freeway section of the A9 test stretch where usually no pedestrians should be, a detected pedestrian would be a strong indication of a potential accident. This information could then be fused with the result of the accident detection model and also be considered for the final decision of whether an accident is detected for a certain timestamp or not.

Furthermore, fusing the results of multiple accident detection methods would also increase the accuracy of the accident detection. One promising approach would be to fuse the results of the presented learning-based accident detection with the results of a vision-language model. As Zhou et al. showed in the paper "Vision Language Models in Autonomous Driving and Intelligent Transportation Systems" [24], vision-language models such as GPT-4V can detect accidents in an input image. Therefore, doing such a high-level fusion should further improve the accident detection accuracy.

Moreover, the problem described in Chapter 5.3 that only partly visible vehicles are sometimes detected as an accident could be investigated in more detail. Solving this problem would probably further reduce the number of false positive detections and therefore increase the accident detection accuracy.

Another way to improve the accident detection for the A9 test stretch is to improve the dataset used for fine-tuning. First of all, it would be beneficial to replace the downloaded dataset, which is part of the custom dataset, with another accident dataset with images with a higher resolution. Then, it would not be necessary to upscale and downscale all the training data. Furthermore, the input images during inference would then not have to be scaled down which would mean that the objects to be detected are larger and can therefore most likely be detected better. Secondly, re-labeling the recorded accidents from the A9 test stretch, this time with more context around the vehicles could also help to improve the detection accuracy.

By improving the generalization capability of accident detection, it can be made more reliable for new, unseen data. This can be achieved by using stronger regularization, data augmentation, and a bigger dataset. The latter can be realized by using rule-based accident detection to find new accidents in the recordings of the A9 test stretch. It is estimated that at least five to ten more accidents, each with at least 100 labeled frames, should be added to the training set to have a certain diversity in the accident frames from the A9 test stretch.

Furthermore, it would be good if a few accidents on the A9 test stretch were not included in the training set. They could then be used for making a more general valid test of the reliability of the accident detection on the A9 test stretch.

Finally, the number of detected accidents, used for the generated scenario statistic, could be improved by only increasing the total accident counter if a certain amount of time has passed between two detected accidents. When the time period between two detected accidents is smaller than the specified threshold, it is very likely that an accident was just not detected for a few milliseconds and that the new detection still belongs to the same accident. Another way to improve the total number of detected accidents as well as the accident detection itself, would be to track the frames, for example with the *Poly-MOT* tracker.

Regardless of whether the accident detection is further improved or not, there are several options on how to make use of the implemented accident detection for the A9 test stretch.

First of all, the provided pipeline for investigating the recordings from the A9 test stretch can be applied to the remaining recordings that have not been covered yet by the experiment described in Chapter 5.3. Furthermore, it can also be tested on the intersection of the A9 test stretch. As the complete pipeline for investigating the recordings already exists and has been shown to work, it would just be necessary to manually inspect the extracted images of the detected accidents. This could lead to the discovery of further accidents that happened on the freeway section of the A9 test stretch. These detected accidents could then be used to create and publish an accident dataset from the A9 test stretch. As there is still a lack of high-quality accident datasets, this would be a valuable contribution to the scientific community and the development of accident detection methods. After all, not having enough high-quality and publicly available accident data remains a significant obstacle to the development of a reliable accident detection system and is therefore an important topic to be addressed. The created accident dataset could then, as already mentioned, also be used for fine-tuning the classification layer of the fine-tuned accident detection model. This should further improve the accuracy of the accident detection and increase the generalization capability of the model.

One way to further automate the investigation of the recordings from the A9 test stretch would be to combine rule-based accident detection and learning-based accident detection. Instead of manually inspecting the extracted images from the potential accidents detected by the rule-based approach, they could be fed into the learning-based accident detection model. The learning-based approach would then take over the task of classifying the accidents detected by the rule-based accident detection, into true positive and false positive detections. However, this could lead to not detecting some accidents that have initially been detected by rule-based accident detection, for example, if the extracted images do not show the accident clearly. This problem could be reduced by applying the whole learning-based accident detection to the accident rosbag instead of just applying the accident detection model to the extracted images. Because then all images from an accident are used for learning-based accident detection instead of just a handful of images. This increases the probability that an accident is clearly shown in the images and therefore detected by learning-based accident detection. Furthermore, the check whether an accident has been detected in three consecutive timestamps would then be used which would reduce the number of false positive detections and therefore lead to better accident detection results.

Another way to make use of the implemented accident detections is to add one of them to the live system. As the learning-based accident detection achieves a much higher accident

detection accuracy on the A9 test data it is more suited for this task. However, due to its high runtime, it would be necessary to filter out potential accident data before being applied. This could for example be done by using scenario detection to detect standing events, traffic jams, and breakdowns beforehand. Then, the learning-based accident detection could just be applied to these potential accident rosbags. If an accident is detected on the live system, an accident warning could be added to the website of the A9 test stretch and a potential future app. Furthermore, if it works reliably enough, it could even be added feature to automatically report a detected accident to the local fire department and police station, including further information such as images of the accident.

Finally, the implemented accident detections could also be used to get further insights into how and why accidents happen. This can be achieved by taking a closer look at the accident recordings found by the accident detection. By investigating which scenarios happened right before the accident or which other aspects such as time and weather conditions have in common, it can be tried to find some general patterns that make the accuracy of an accident more likely. The insights of this analysis could then be used to calculate the live probability of an accident on the freeway section of the A9 test stretch. This information could be displayed on the website of the A9 test stretch as well as in a potential future app.

In the future, technology advancements can be expected to significantly enhance accident detection systems. In particular, it is anticipated that they can reliably detect accidents in real-time, using data from roadside sensors as well as vehicle sensors. In addition, accident detection systems will most likely automatically and immediately notify emergency services and send additional information such as the location of the accidents and images of it. Furthermore, accident reports could be automatically distributed to the closest available emergency vehicles without the need for a human to coordinate this process. This would further reduce the time for post-crash care. It can also be assumed that the focus will shift from accident detection to accident prediction in the next few years, as preventing accidents is even better than automatically detecting them and will therefore be the ultimate goal. However, there are still many steps to be taken and a lot of work and research to be done to get there, particularly in terms of reliably detecting and predicting accidents in poor visibility conditions such as fog or complex traffic situations such as in India.

Chapter 7

Conclusion

The time between the occurrence of an accident and the arrival of medical assistance significantly impacts whether the passengers of a vehicle survive an accident. This time could be reduced by having reliable, automatic accident detection. Consequently, automatic accident detection has the potential to help save lives. However, most of the existing accident detection methods have never been tested on real traffic data of a test stretch. Therefore, this thesis has investigated whether it is possible to reliably detect accidents on the A9 test stretch using roadside sensors. To achieve this, two accident detections have been developed: a rule-based accident detection and a learning-based accident detection. Both have been integrated into the scenario detection of the A9 test stretch and have been optimized. For the rule-based accident detection, the scenario detection has been improved and an image extraction feature has been added. For learning-based accident detection, several measures have been taken to improve its accident detection accuracy. A big part of this optimization was achieved by training multiple models with a self-created custom accident dataset, also including labeled accident images from the A9 test stretch. By doing an extensive empirical analysis the best performing configuration for the learning-based accident detection has been identified.

The conducted experiments have shown that it is possible to reliably detect accidents in test data from the A9 test stretch. In fact, the learning-based accident detection even achieves a precision of 0.8 and recall of 1.0 on the test data. This means that the learning-based accident detection not only detects each accident in the test data but also does not have many incorrect detections. However, this value can only be considered generally valid to a limited extent as the test data only contains four accidents. The rule-based accident detection in principle also works but does not achieve such good accuracy values. Apart from the accident detection accuracy also the runtime of the two implemented accident detections has been investigated. While rule-based accident detection is fast enough for real-time usage, learning-based accident detection is too slow. Consequently, it is also possible to detect accidents on the A9 test stretch in real time using roadside sensors. However, none of the implemented approaches makes it possible to achieve both, reliable and real-time detection at the same time.

By developing an accident detection for the A9 test stretch, the foundation was laid to create an accident dataset. The accident images for such a dataset could be obtained by using a combination of the two implemented accident detections. First, the implemented pipeline for finding accidents in the recordings of the A9 test stretch can be used to find potential accident rosbags. Secondly, learning-based accident detection can be applied to these potential accident recordings to find the actual accidents in them.

List of Figures

List of Tables

Bibliography

[1] Bao, W., Yu, Q., and Kong, Y. "Uncertainty-based Traffic Accident Anticipation with Spatio-Temporal Relational Learning". In: *ACM Multimedia Conference*. May 2020.

[2] Bundesamt, S. *Zahl der Verkehrstoten sinkt im Jahr 2023 voraussichtlich leicht auf 2 750*. URL: https://www.destatis.de/DE/Presse/Pressemitteilungen/2023/12/PD23_471_46241.html.

[3] C, C. K. *Accident Detection From CCTV Footage*. 2020. DOI: 10.34740/KAGGLE/DSV/1379553. URL: https://www.kaggle.com/dsv/1379553.

[4] Creß, C., Bing, Z., and Knoll, A. "Intelligent Transportation Systems Using External Infrastructure: A Literature Survey". In: *arXiv* (2021).

[5] Creß, C., Zimmer, W., Strand, L., Fortkord, M., Dai, S., Lakshminarasimhan, V., and Knoll, A. "A9-Dataset: Multi-Sensor Infrastructure-Based Dataset for Mobility Research". In: *33rd IEEE Intelligent Vehicles Symposium (IV)*. 2022.

[6] Fang, J., Qiao, J., Bai, J., Yu, H., and Xue, J. "Traffic Accident Detection via Self-Supervised Consistency Learning in Driving Scenarios". In: *IEEE Transactions on Intelligent Transportation Systems* 23.7 (2022), pp. 9601–9614. DOI: 10.1109/TITS.2022.3157254.

[7] Géron, A. *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. 2nd ed. O'Reilly Media, Incorporated, 2019. ISBN: 9781492032649.

[8] Ghatee, M. *Accident-Images-Analysis-Dataset*. visited on 18/01/2024. 2018. URL: https://github.com/mghatee/Accident-Images-Analysis-Dataset.

[9] Khandelwal, S. *yoloaccident*. visited on 18/01/2024. 2018. URL: https://drive.google.com/drive/u/0/folders/1_4p1mW0BzDX0ZoHdaMIU5c7YOniOKzO2?direction=a.

[10] Kuhn, M., Johnson, K., et al. *Applied predictive modeling*. Vol. 26. Springer, 2013.

[11] O'Shea, K. and Nash, R. *An Introduction to Convolutional Neural Networks*. visited on 30/01/2024. 2015. URL: https://arxiv.org/pdf/1511.08458.pdf.

[12] OpenAI. "GPT-4V(ision) System Card". In: visited on 09/03/2024. 2023. URL: https://api.semanticscholar.org/CorpusID:263218031.

[13] Organization, W. H. *Global status report on road safety 2023*. World Health Organization, 2023, ix, 81 p.

[14] Pawar, M. *accidents.zip*. visited on 18/01/2024. 2020. URL: https://drive.google.com/file/d/1o0D7vnGUZHS72is6n1jV1ge2BDfObzVi/view.

[15] Pillai, M. S., Chaudhary, G., Khari, M., and Crespo, R. G. "Real-time image enhancement for an automatic automobile accident detection through CCTV using deep learning". In: *Soft Computing* 25.18 (2021), pp. 11929–11940. ISSN: 1433-7479. DOI: 10.1007/s00500-021-05576-w.

[16] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. *You Only Look Once: Unified, Real-Time Object Detection*. 2016. arXiv: 1506.02640 [cs.CV].

[17] Sabry, K. and Emad, M. "Road Traffic Accidents Detection Based On Crash Estimation". In: *2021 17th International Computer Engineering Conference (ICENCO)*. 2021, pp. 63–68. DOI: 10.1109/ICENCO49852.2021.9698968.

[18] Salehi, M., Sadjadi, N., Baselizadeh, S., Rohban, M. H., and Rabiee, H. R. *Multiresolution Knowledge Distillation for Anomaly Detection*. 2020. arXiv: 2011.11108 [cs.CV].

[19] Shah, A., Lamare, J. B., Anh, T. N., and Hauptmann, A. "CADP: A Novel Dataset for CCTV Traffic Camera based Accident Analysis". In: *arXiv preprint arXiv:1809.05782* (2018). First three authors share the first authorship.

[20] Shandilya, S. *Accident detection model Dataset*. https://universe.roboflow.com/accident-detection-model/accident-detection-model. Open Source Dataset. Apr. 2023. URL: https://universe.roboflow.com/accident-detection-model/accident-detection-model.

[21] Shandilya, S. *Accident-Detection-Model*. https://github.com/shubhankar-shandilya-india/Accident-Detection-Model. 2023.

[22] Union, E. *eCall 112-based emergency assistance from your vehicle*. visited on 05/01/2024. URL: https://europa.eu/youreurope/citizens/travel/security-and-emergencies/emergency-assistance-vehicles-ecall/index_en.htm#:~:text=Your%20eCall%20system%20works%20in,or%20where%20it%20is%20registered..

[23] Wang, T., Kim, S., Ji, W., Xie, E., Ge, C., Chen, J., Li, Z., and Ping, L. "DeepAccident: A Motion and Accident Prediction Benchmark for V2X Autonomous Driving". In: *arXiv preprint arXiv:2304.01168* (2023).

[24] Zhou, X., Liu, M., Zagar, B. L., Yurtsever, E., and Knoll, A. C. *Vision Language Models in Autonomous Driving and Intelligent Transportation Systems*. 2023. arXiv: 2310.14414 [cs.CV].

[25] Zhou, Z., Dong, X., Li, Z., Yu, K., Ding, C., and Yang, Y. "Spatio-Temporal Feature Encoding for Traffic Accident Detection in VANET Environment". In: *IEEE Transactions on Intelligent Transportation Systems* 23.10 (2022), pp. 19772–19781. DOI: 10.1109/TITS.2022.3147826.