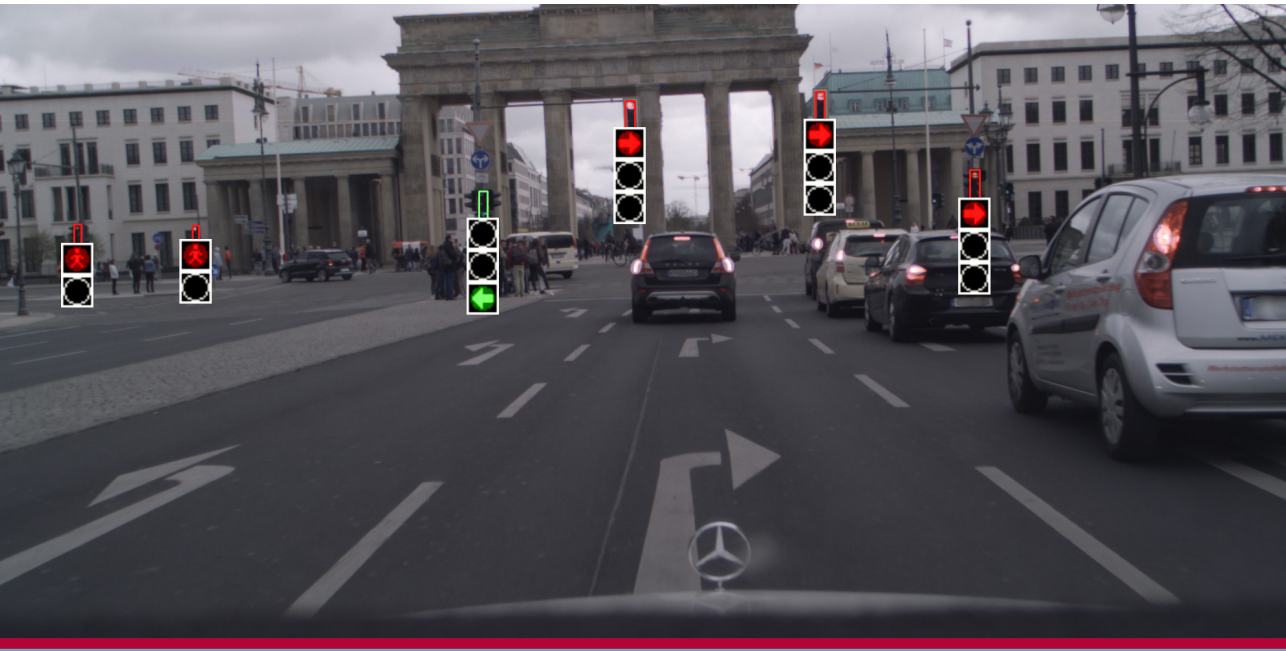
universität ulm



One- and Two-Stage Traffic Light Detection and Attribute Classification Using Convolutional Neural Networks

Julian Müller

Band 39

Schriftenreihe des Instituts für Mess-, Regel- und Mikrotechnik

Julian Müller

One- and Two-Stage Traffic Light Detection and Attribute Classification Using Convolutional Neural Networks

Schriftenreihe des
Instituts für Mess-, Regel- und Mikrotechnik
Universität Ulm

Herausgeber:
Prof. Dr.-Ing. Klaus Dietmayer

Band 39

Julian Müller

One- and Two-Stage Traffic Light Detection and Attribute Classification Using Convolutional Neural Networks

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.dnb.de abrufbar.

Dissertation, Universität Ulm,
Fakultät für Ingenieurwissenschaften, Informatik und Psychologie, 2021

Impressum

Universität Ulm
Institut für Mess-, Regel- und Mikrotechnik
Prof. Dr.-Ing. Klaus Dietmayer
Albert-Einstein-Allee 41
89081 Ulm
http://www.uni-ulm.de/mrm

Eine Übersicht über alle Bände der Schriftenreihe finden Sie unter
http://www.uni-ulm.de/mrmschriften.

Diese Veröffentlichung ist im Internet auf dem Ulmer Volltextserver (http://vts.uni-ulm.de) verfügbar und dort unter der Lizenz “B - Standard (ohne Print-On-Demand)” publiziert.
Details zur Lizenz sind unter http://www.uni-ulm.de/index.php?id=50392 zu finden.

Institut für Mess-, Regel- und Mikrotechnik der Universität Ulm 2021
Print on Demand

ISBN 978-3-941543-60-7 e-ISBN 978-3-941543-61-4



ulm university universität
uulm

One- and Two-Stage Traffic Light Detection and Attribute Classification Using Convolutional Neural Networks

DISSERTATION

zur Erlangung des akademischen Grades eines

DOKTOR-INGENIEURS
(Dr.-Ing.)

der Fakultät für Ingenieurwissenschaften, Informatik
und Psychologie der Universität Ulm

von

Julian Müller
aus Ehingen (Donau)

Gutachter:	Prof. Dr.-Ing. Klaus Dietmayer
	Prof. Dr.-Ing. Rainer Ott
Amtierender Dekan:	Prof. Dr.-Ing. Maurits Ortmanns

Ulm, 23.07.2021

Acknowledgements

At first I want to thank Prof. Dr.-Ing. Klaus Dietmayer for his trust and support during the last years ensuring optimal conditions and laying the foundation for this thesis. I also want to thank all colleagues at the Institute of Measurement, Control and Microtechnology at Ulm University who have supported me. Special thanks go to Claudia Fricke and Dr.-Ing. Michael Buchholz supporting me to solve all organisational issues.

I would like to thank Prof. Dr.-Ing. Rainer Ott for his extensive support during my entire PhD time. Without his countless contributions, comments and suggestions this thesis would not have been this successful.

Furthermore, I want to express my thanks to Dr.-Ing. Ulrich Kreßel for supporting me since my first day at Mercedes-Benz. This includes providing tools, hardware and infrastructure, but also the great working atmosphere in Ulm and plenty of technical discussions and private conversations. Moreover, I want to thank all his employees, PhDs and students to contribute to the great working atmosphere.

My thanks go to Dr.-Ing. Frank Lindner for numerous technical discussions. His distinct knowledge in almost all aspects supported me to take the best decisions and helped to improve my system and thesis.

I want to thank my colleague and friend Dr.-Ing. Andreas Fregin for his great preliminary work and plenty of fruitful discussions. I am really happy and proud about our joint achievements, particularly the publication of our traffic light dataset, but also the conference journeys as major highlights of recent years.

Special thanks go to my parents Ingrid and Herbert and sister Franzi for supporting me during my entire life, study and PhD time. I am really thankful for our great relationship.

Finally, I want to thank my love Alina for her understanding with regard to numerous working hours during the late evening and weekend. Without your unbroken support this thesis would not have been possible.

Abstract

This thesis addresses the topic of traffic light detection and attribute classification using Convolutional Neural Networks (CNNs). One of the most popular one-stage detection approaches, namely Single Shot Detection/Detector (SSD), is adapted in order to tackle the small object detection problem traffic light recognition systems are facing.

A particular focus lies on the adaptation of reference boxes, so called prior boxes, with respect to which bounding boxes are relatively predicted. A novel strategy to determine prior box dimensions and locations is presented within this thesis. Initially, a weighted k-IoU clustering approach is presented to determine prior box dimensions. This approach is tailored to the data imbalance and the desired target metric intersection over union expressing detection quality. Adapting prior boxes to the expected object dimensions helps to improve the entire learning process.
In addition, a novel strategy is introduced in order to overcome the fixed, grid-based prior box locations being state-of-the-art. It proposes to adaptively distribute prior boxes with respect to its dimensions. The proposed adaptive prior box locations are mathematically derived and guarantee an optimal matching during training, from which particularly small objects benefit.

Beyond traffic light detection, state and pictogram classification is addressed within this thesis. Two different methodologies are developed and compared. The first is a straight-forward extension of the one-stage methodology used for detection by utilizing additional convolutional layers predicting state and pictogram confidences, respectively. A loss function adaptation and attribute-exclusive data handling is developed and presented.
Because implicit classification within deep convolutional neural networks faces several challenges and disadvantages, a second methodology is developed and compared. This methodology can be considered as a two-stage classification, in which the initial detections from SSD are classified by two subsequent state and pictogram classifiers. The network is designed as an efficient and fast, light-weight convolutional network and operates on the regions of interest obtained from the previous detection step. Instead of high receptive fields, this guarantees a focus on local features, which turns out to be crucial for pictogram classification. An extensive investigations section shows the effectiveness of the developed strategies to overcome data imbalance.

Because traffic light detection and classification methods are used in safety-critical systems, they are supposed to express uncertainty. The network output confidences turn out to be poorly calibrated. Hence, methods to calibrate the detection as well as attribute confidences are presented. Depending on the actual state of the traffic lights, calibration quality varies, which is tackled by using an attribute-wise calibration aligning the detection ability with respect to its true state.

Throughout the entire thesis, the focus lies on extensive and application-orientated investigations and evaluations. For this purpose, the DriveU Traffic Light Dataset (DTLD) is made publicly available for researchers in the field of traffic light recognition. It is the to-date largest traffic light database with the highest amount of annotated attributes. The presented results support the effectiveness of the introduced novelties in prior box adaptations and attribute classification methodologies and show, that the remaining error cases preponderantly are actual human annotation errors.
Within the evaluation in this thesis, dataset quality turns out to be one bottleneck when evaluating operating points at low false positives rates. The narrowing gap between human and artificial performance suggests to support human object annotation using the object detectors. A relabeling workflow is presented, which generates potential annotation errors by an automated training and evaluation pipeline. Further experiments investigate the impact of annotation errors on the performance and proof the significance of dataset quality for safety critical applications, such as driver assistance or autonomous systems requiring very low false positive rates. As part of this thesis, the enhanced dataset annotations are made publicly available on the dataset servers.

The final results are very promising. The system is able to detect 98% of all relevant traffic lights larger 10 pixels width at only one false positive each 100 images. At this operating point, state classification reaches more than 99 percent for the active states red, green and red-yellow and more than 95 percent recall for the pictograms circle, arrow_left, arrow_straight, arrow_right, pedestrian and tram.

Kurzfassung

Die vorliegende Arbeit adressiert das Thema der Lichtsignaldetektion und Attributklassifikation unter Nutzung von faltenden, neuronalen Netzwerken. Eines der populärsten Verfahren, der sogenannte Single Shot Detector (SSD), wird im Rahmen dieser Thesis adaptiert, um die Problematik der Erkennung sehr kleiner Objekte zu bewältigen. Dies ist eine der größten Herausforderungen von Lichtsignalerkennungssystemen.

Verstärkter Fokus liegt hierbei auf der Adaption von Referenzboxen, sogenannter Prior Boxes, zu welchen die Lichtsignale relativ prädiziert werden. Eine neuartige Strategie zur Dimensionierung und Positionierung der Prior Boxes wird vorgestellt. Zunächst wird ein gewichteter k-IoU Clustering-Ansatz zur Dimensionierung der Prior Box Größen präsentiert. Jener ist speziell zugeschnitten auf das vorliegende Datenungleichgewicht sowie die Zielmetrik Intersection over Union, welche die Detektionsgüte ausdrückt. Die Anpassung der Prior Boxes an die zu erwartenden Eingangsdaten unterstützt den Lernprozess.
Zusätzlich wird eine neuartige Strategie zur Bestimmung der Prior Box Positionierung präsentiert, welche die gängige, raster-basierte Positionierung ersetzt. Stattdessen entwickelt diese Arbeit eine mathematisch motivierte, adaptive Positionierung in Abhängigkeit der Prior Box Größe. Dies garantiert ein optimales Matching während des Trainingsprozesses, von welchem insbesondere die Erkennungsleistung auf sehr kleinen Objekten profitiert.

Über die Detektion hinaus wird die Klassifikation des Lichtsignalzustandes sowie des Lichtsignalpiktogramms in dieser Arbeit adressiert. Zwei unterschiedliche Methoden werden entwickelt und verglichen. Das erste Verfahren stellt eine logische Erweiterung der für die Detektion genutzten One-Stage Methodik durch Verwendung zweier zusätzlicher Faltungsschichten vor, welche Zustand- und Piktogrammkonfidenzen implizit prädizieren. Eine Adaption der Verlustfunktion, sowie eine attributabhängige Datenvorverarbeitung wird entwickelt und evaluiert.
Da die implizite Klassifikation im Rahmen von tiefen, neuronalen Netzen einigen Herausforderungen und Nachteilen gegenübersteht, wird eine weitere Methode vorgestellt. Jene kann als zweistufige Klassifikation angesehen werden, in welcher die initialen Detektionen des SSD Verfahrens mit Hilfe von zwei nachfolgenden Klassifikatoren, jeweils in Zustand und Piktogramm klassifiziert werden. Das Netzwerk ist als ef-

fizientes und rechenarmes Faltungsnetzwerk entworfen, welches lediglich auf der Region of Interest operiert. Dieses Vorgehen garantiert einen Fokus auf lokale, pixelbasierte Merkmale, welche für die Piktogrammklassifikation von wichtiger Bedeutung sind. Umfangreiche Untersuchungen bestätigen die Effektivität der Strategien, um die Erkennungsleistung auf unterrepräsentierten Klassen deutlich zu erhöhen.

Da Lichtsignalerkennungs- und Klassifikationsmodule ihre Anwendung typischerweise in sicherheitskritischen Systemen finden, ist das Ausdrücken von Unsicherheit ein bedeutender Faktor. Da sich die Netzwerkkonfidenzen als dekalibriert herausgestellt haben, beschreibt diese Arbeit Methoden zur Kalibrierung der Detektions- und Attributkonfidenzen. Da Untersuchungen eine zustandsabhängige Kalibrierungsgüte offenbart haben, beschreibt diese Arbeit eine attributweise Kalibrierung, welche die Detektionsgüte in Abhängigkeit des Zustandes angleicht.

Im Rahmen dieser Arbeit liegt ein verstärkter Fokus auf ausgiebigen und anwendungsorientierten Untersuchungen und Evaluierungen. Hierfür wurde der DriveU Traffic Light Dataset (DTLD) verwendet, welcher für Forschungszwecke im Feld der Lichtsignalerkennung veröffentlicht wurde. Dieser Datensatz kann als bis dato größter Datensatz für Lichtsignalerkennung betrachtet werden und enthält die höchste Anzahl an annotierten Lichtsignalen und Attributen. Die auf dem Datensatz erhobenen Evaluierungen unterstreichen die Effektivität der vorgestellten Neuheiten im Bezug auf die Prior Box Dimensionierung sowie Attributklassifikationsmethodik. Die Evaluierung hat die Datensatzqualität als Flaschenhals der Evaluierung von Arbeitspunkten mit sehr geringen Falschalarmraten herauskristallisiert. Die sich schließende Lücke zwischen menschlicher und künstlicher Erkennungsperformance indiziert die Nutzung von Objektdetektoren zur Unterstützung der menschlichen Objektannotation. In dieser Arbeit wird ein Korrekturworkflow vorgestellt, welcher potentielle Annotationsfehler mit Hilfe einer teil-automatisierten Trainings- und Evaluierungspipeline aufdeckt. Weiterführende Experimente diskutieren den Einfluss der Datensatzqualität auf die Evaluierbarkeit von sicherheitskritischen Systemen, welche typischerweise sehr geringe Falschalarmraten erfordern. Im Rahmen dieser Arbeit werden die korrigierten Datensatzannotationen veröffentlicht.

Die erzielten Ergebnisse, im Detail eine Erkennungsrate von 98% auf relevanten Lichtsignalen größer 10 Pixeln Objektbreite, bei im Mittel einem Falschalarm je 100 Bilder, übertreffen bisherige Erkennungsleistungen unter Nutzung konventioneller Verfahren um mehr als eine Größenordnung. Die Zustandsklassifikation erreicht auf den aktiven Zuständen rot, rot-gelb und grün eine Klassifikationsrate von mehr als 99 Prozent. Zusätzlich erreicht die Piktogrammklassifikation eine Erkennungsrate von mehr als 95% auf den relevanten Klassen.

Contents

1 Introduction **1**
- 1.1 Characterization of Traffic Lights in Germany . . . 4
- 1.2 Challenges . . . 9

2 Related Work **13**
- 2.1 Research on Traffic Light Detection . . . 14
  - 2.1.1 Conventional Traffic Light Detection . . . 14
  - 2.1.2 CNN-based Traffic Light Detection . . . 15
- 2.2 Research on Traffic Light State Recognition . . . 19
- 2.3 Research on Traffic Light Pictogram Recognition . . . 20
- 2.4 Discussion . . . 21

3 Outline and Main Contributions **25**
- 3.1 Traffic Light Detection using Single Shot Detection (SSD) . . . 26
- 3.2 Extending SSD for State and Pictogram Recognition . . . 28
- 3.3 Confidence Calibration . . . 29
- 3.4 Dataset Relabeling - Significance for Performance and Evaluation Reliability . . . 30
- 3.5 Publications . . . 31

4 Dataset and Evaluation Metrics **33**
- 4.1 Overview of Traffic Light Datasets . . . 33
- 4.2 DriveU Traffic Light Dataset (DTLD) . . . 38
  - 4.2.1 Camera Images . . . 38
  - 4.2.2 Labels . . . 38
  - 4.2.3 Dataset Statistics . . . 40
- 4.3 Evaluation Metrics . . . 43
  - 4.3.1 Detection Metrics . . . 43
  - 4.3.2 Multiclass Metrics . . . 47

5 Traffic Light Detection using Single Shot Detection (SSD) **49**
- 5.1 Requirements and Goals . . . 50
- 5.2 Single Shot Detection . . . 55
  - 5.2.1 Basic Principle . . . 56

5.2.2 Basis Network . . . 59
5.2.3 Design of Prior Boxes - Optimizing Prior Box Dimensions and Locations . . . 63
5.2.4 Bounding Box Regression and Loss Functions . . . 71
5.2.5 Non-Maximum Suppression . . . 75
5.3 Experimental Investigations and Evaluation . . . 76
5.3.1 Statistical Variance of Unique Experiments . . . 78
5.3.2 Impact of Image Conversion . . . 79
5.3.3 Impact of Ignore Regions . . . 81
5.3.4 Impact of Data Augmentation . . . 82
5.3.5 Impact of Network Depth and Prior Box Locations . . . 85
5.3.6 Impact of Training Data Amount . . . 90
5.3.7 Hyperparameter Optimization . . . 91
5.3.8 Detailed Results . . . 92
5.4 Discussion . . . 98

6 Extending Single Shot Detection for State and Pictogram Recognition 101
6.1 Requirements and Goals . . . 103
6.2 One-Stage Detection and Attribute Classification . . . 105
6.3 Two-Stage Attribute Classification . . . 107
6.3.1 Network Design . . . 107
6.3.2 Data Collection . . . 109
6.4 Experimental Investigations and Evaluation (One-Stage) . . . 112
6.4.1 Baseline Experiments . . . 113
6.4.2 Impact of Ignore Regions . . . 116
6.4.3 Data Augmentation . . . 117
6.4.4 Impact of Network Depth and Prior Box Stride . . . 119
6.4.5 Detailed Results . . . 121
6.5 Experimental Investigations and Evaluation (Two-Stage) . . . 124
6.5.1 Baseline Experiments . . . 127
6.5.2 Data Input Investigations . . . 128
6.5.3 Network Modifications . . . 130
6.6 Best of Two Worlds . . . 132
6.7 Discussion . . . 135

7 Confidence Calibration 137
7.1 Related Work . . . 137
7.2 Definitions, Measures, Metrics and Pre-Investigations . . . 138
7.3 Confidence Calibration Methodologies . . . 143
7.4 Attribute Calibration Extensions . . . 144
7.5 Evaluation . . . 145
7.5.1 Detection Confidence Calibration . . . 145

7.5.2 Attribute Confidence Calibration 147

8 Dataset Relabeling - Significance for Performance and Evaluation Reliability 149
8.1 Relabeling Workflow 150
8.2 Results 153
8.2.1 Significance of Dataset Quality for Evaluation Reliability . . 153
8.2.2 Significance of Dataset Quality for the Network Performance 154
8.3 DriveU Traffic Light Dataset v2 156

9 Discussion and Perspectives 157

A Decompanding Curve 161

B Supplementary Material on Prior Boxes 163
B.1 Mathematical Relationship between Intersection Over Union and Prior Box to Ground Truth Misalignment 163
B.2 Prior Box Dimensions vs. Runtime 165
B.3 Prior Box Locations 165

C Further Investigations 167
C.1 False Positive Categorization Using Scene Segmentation 167
C.2 Attribute Classification Performance w.r.t. Position and Size 169

Acronyms 171

List of Symbols 173

Bibliography 175

Eigene Veröffentlichungen 185

Chapter 1

Introduction

Despite an increasing number of vehicle registrations, the number of road fatalities clearly decreased during the last thirty years (compare statistics of the "Statistisches Bundesamt" [StBA20]). Besides a progress in safety technology, a strong rise in vehicles equipped with driver assistance systems helped to decrease the total number of accidents. Alongside with a reduction of CO_2 in terms of an establishment of electric vehicles, autonomous vehicles are the most hyped and discussed topics in the automotive industry, in which a radical rethinking can be observed.

While twenty years ago driver assistance systems were only available in upper or luxury class vehicles, nowadays also small car segment vehicles offer a wide range of assistance systems. This includes collision avoidance systems (breaking due to obstacles, vehicles), blind spot assistant, traffic sign recognition or pedestrian protection systems and others.
Up to 2020, no car manufacturer offered a mainly vision-based traffic light assistant for series cars. This changed with a new release of the Tesla Autopilot [Tesla20]. The traffic light and stop sign control released in 2020 supports the driver by decelerating in case of red lights. However, Tesla explicitly mentions, that the traffic light assistant is in Beta mode and the driver has to manually confirm a correct system decision at each intersection by using the accelerator pedal or autopilot stalk.

Several reasons exist making a development of a traffic light assistant more challenging than other assistance systems. In comparison to other road objects, which have to be detected, traffic lights are rather small and additionally have to be detected at high distances. Moreover, traffic lights and many non-traffic light regions share similar features, which increases the probability of false alarms. Current methods tackling traffic light recognition are typically camera-based, because most alternative sensors are not able to recognize color information required to determine the traffic light state. From the perspective of a long-term solution for autonomous vehicles, all traffic lights could be equipped with Vehicle-to-Infrastructure (V2I)

sensors transmitting state information to the vehicle. A similar system is available in Audi's upper-class vehicles [Audi19]. Since 2017, they offer a V2I-based Green Light Optimized Speed Advisory in the US, which was also introduced in Germany in 2019. However, the system requires connected traffic lights transmitting traffic light status to the servers of Audi. Consequently, the system is only available at selected intersections. Even for the case that all traffic lights are connected, a second, vision-based path, is hardly avoidable. This helps to recognize not registered traffic lights (construction sites) or handle potential attacks on the systems.

Besides the detection of traffic lights, meaning a localization in the image and in world coordinates, additional traffic light attributes have to be classified in order to solve the overall task. Figure 1.1 illustrates the most important attributes of traffic lights. Not all attributes necessarily have to be recognized, however some are mandatory. *The most important attribute is the dynamic state.* Traffic lights have to be continuously detected in order to monitor potential changes in phase. This is a substantial difference to related tasks, such as traffic signs, which are usually static. *Besides a vision-based color recognition, a further key problem is the determination of the relevant traffic light.* Relevant describes the traffic light or multiple traffic lights controlling the planned route of the vehicle. In the event of a traffic light assistance system, this information may not be known. In the case of autonomous vehicles, the planned route is generally known - in this case this task simplifies to finding the state for a specific lane. Because estimating relevancy from a vision-based system is risky and highly error-prone, a more reasonable proceed is to find the relevant traffic lights by using several sources of information instead of an end-to-end learning. Autonomous vehicles have access to high definition maps. The detected traffic lights could be matched with map information, whereas the state is received by the vision system.

A purely vision-based determination of relevant traffic lights is more challenging. A coherent proceed is to determine a relationship between the detected traffic lights and all lanes/routes a vehicle can take. Pictograms are used to facilitate this assignment, indicating the belonging of each traffic light by arrow, circular, pedestrian or other shapes installed on the active lamps. In consequence, being able to predict pictogram information from a vision system seems to be one important key to tackle the relevancy question. Due to the importance of pictogram information, this thesis presents a system, which is able to predict the pictogram class besides the detection and state classification task. In comparison to state recognition, pictograms are static - consequently a once correctly determined pictogram is sufficient. The main challenge is the requirement on resolution - which is clearly higher than for the detection and state recognition task. In particular, arrow-shaped and pedestrian/cyclist pictograms show high spatial frequencies complicating a far range recognition at low object resolutions.

From the current level of knowledge, all further traffic light attributes presented in

Figure 1.1 are not mandatory essential to design a traffic light assistant. The number of bulbs does not offer a direct additional information - it may indicate flashing lights, which is often the case for one aspect traffic lights. Occlusion prediction may support to estimate the uncertainty of the prediction, i.e. if a traffic light is occluded, a wrong state or pictogram classification is more likely. With regard to the relevancy question, orientation can be a valuable attribute. More precisely, the rotation of traffic lights often transfers information, whether a traffic light is important for a vehicle or for other lanes. However, no publication to date proofed, that predicting orientation can offer a certain value.

This thesis addresses the three most essential tasks for traffic light recognition: traffic light detection, traffic light state recognition and traffic light pictogram recognition. The approach is based on a convolutional neural network object detection methodology, which showed its potential in related object detection tasks. Extensive adaptations are presented, which increase the detection capability on the one hand and enable a classification of additional attributes on the other hand. The presented

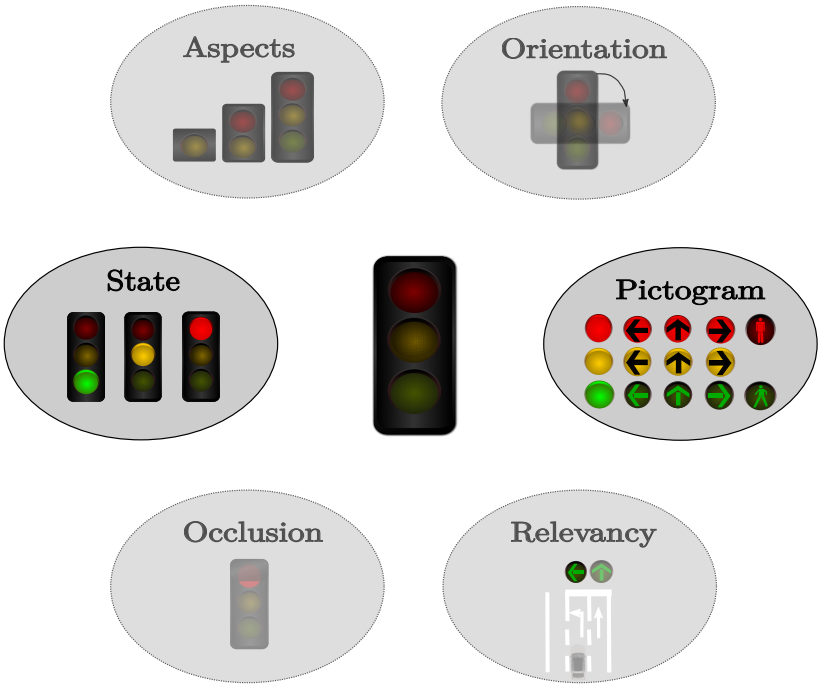


Figure 1.1: Overview of traffic light attributes, which can be predicted from vision approaches. Within this thesis, the focus lies on detection as well as state and pictogram recognition.

method is trained and evaluated purely on recordings on German streets and thus aims to detect traffic lights in Germany. Therefore, the largest published traffic light dataset is used to produce reliable and robust results.

In Germany, the construction form of traffic lights is prescribed by law [RiLSA15]. Section 1.1 gives detailed insights into the construction forms of traffic lights. All existing traffic light states and most existing pictogram variations are presented, which lay the foundation to design a recognition system. Section 1.2 presents typical challenges in traffic light recognition aggravating to design reliable and accurate systems. It shows, why traffic light recognition is still a state of the art problem. A detailed overview on the main contributions and topics within this thesis is given in Chapter 3.

1.1 Characterization of Traffic Lights in Germany

Common construction forms of traffic lights in Germany and important properties are prescribed by law, see [RiLSA15] for details. The construction form has to be considered when designing systems for traffic light recognition. There are differences in other countries/continents, which are not discussed in this thesis. Common traffic light types, as well as their construction sizes are shown in Figure 1.2. For traffic light detection, the following construction and installation properties are important:

- **Orientation**: In Germany, traffic lights have to be vertically installed. This differs from other countries, for instance in the United States (US) or Asia, where traffic lights are often horizontally installed.

- **Number of aspects**: Frequently three, rarely two or one, very rarely more than three. Two or one aspect are used at railway/pedestrian crossings. Four aspects are common for traffic lights regulating rail traffic. In Germany, one or two aspect traffic lights are frequently flashing.

- **Back panels**: Frequently used, mostly in black and rarely in white, more rarely without panel. However, the equipment is linked to local authority and there are cities, which install many traffic lights without back panels. This can lead to difficulties when using machine learning approaches, because edges of the panel are important features.

- **Size**: Depending on the speed limit, two different traffic light sizes are prescribed. A light diameter of 200 millimetres is commonly used at road velocities up to 50 km/h, above a diameter of 300 millimetres is installed. However,

300 millimetres can also be utilized at potentially critical intersections. The diameter mainly influences model-based approaches.

Traffic Light State: Besides the detection of traffic lights, determining their state is another crucial task. Although a communication between traffic light and modern vehicles is planned, a comprehensive equipment of traffic lights with communication technology will consume a lot of time. Moreover, for the purpose of safety a redundant path has to exist. Traffic lights at construction sites may not be equipped with communication technology or not updated in maps, which also strengthens the need for a vision-based system.
Figure 1.3 illustrates the typical traffic light state sequence in Germany and most parts of Europe.

- **Red**: indicates to stop. Duration varies depending on the intersection or traffic volume.
- **Red-yellow**: indicates a turn to green - prepare to proceed. Duration is linked to the speed limit.
- **Green**: indicates proceeding. Duration varies depending on the intersection or traffic volume.
- **Yellow**: indicates a turn to red - prepare to stop. Duration is linked to the speed limit.
- **Off**: keep with the priority rules. Turning off traffic lights is only allowed after a yellow phase of at least 5 seconds duration.

While red, green and off cycles are not restricted to a fixed duration, the minimum red-yellow phase is 1 second and up to a maximum duration of 2 seconds. The yellow phase is linked to the legal speed limit - at least 3 seconds at 50 km/h, 4 seconds at 60 km/h and 5 seconds at 70 km/h. In order to not overtax the driver, the *off* state is only allowed after a yellow cycle of at least 5 seconds duration. In the evening hours in small cities with less traffic, traffic lights are frequently turned off.

Traffic Light Pictogram: Intersections are frequently equipped with a higher number of traffic lights. In order to simplify understanding of an intersection - *which traffic light controls which lane or direction?* - pictograms are used. Frequently, traffic lights are laterally aligned to the lane they control. However, in many situations no clear assignment between traffic light(s) and lane(s) is possible. In order to resolve ambiguities, a classification of the traffic light pictogram is necessary for driver assistance systems or autonomous systems.

Traffic lights referring to straight traffic are mostly equipped with a colored circular pictogram or a colored straight arrow. Figure 1.4 shows various appearances of traffic lights utilized to control straight traffic in Germany. There are two noticeable properties: The arrow color and the number of components an arrow consists of. In case of green, the arrow is colored with dark background, whereas for the other states a black arrow on top of an active, circular bulb is used. This can affect the pictogram classification, as for the second variant the circular bulb feature exists in both pictogram classes, arrow_straight and circle. The second noticeable property is, that green arrows consist of two components, whereas all other arrows only consist of one.

Traffic lights controlling left-turn lanes are shown in Figure 1.5. The observations mentioned for straight arrows are also present for left arrows. Moreover, there exist left arrows rotated by 45°. This complicates the differentiation to straight arrows. Right arrow pictograms are illustrated in Figure 1.6. A special case is shown in the most-right example. Despite a red traffic light, a protected right turn is allowed. In the United States this is permitted with exceptions without additional signs. In Germany, an additional sign with a green right arrow (no bulb) symbolizes a permission for turning right. It can also be symbolized by an additional traffic light with one, two or three aspects.

The widest range of appearance can be observed for pictograms utilized for pedestrian and cyclist traffic lights. In Germany, pedestrian lights typically consist of two aspects symbolizing a standing (red) and walking (green) pedestrian. Some cities also use pedestrian lights with three aspects, where an additional red light can help people with color blindness or deficiency. From the perspective of pictogram classification using machine learning approaches, pedestrian and cyclist signs are the most challenging due to the high spatial frequency. This is particularly challenging at low object resolutions leading to a blurring of pictogram into a simple colored blob. Unfortunately, the recognition of pedestrian and cyclist pictograms is moreover complicated due to a lack of standardization. A popular example is the so-called *Ampelmännchen*. Prior to German reunification, a male figure was used in East Germany, whereas a generic figure was used in West Germany. The Ampelmännchen is still used in Berlin and can nowadays also be observed in western German cities such as Heidelberg or Saarbrücken. Another variant uses both, pedestrian and cyclist sign, in one pictogram. The most common pedestrian and cyclist pictograms are shown in Figure 1.7.

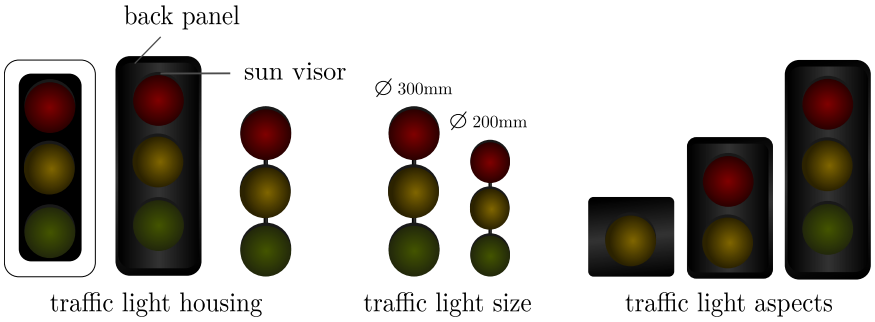


Figure 1.2: Common traffic light types in Germany consist of one, two or three aspects. A higher number of aspects can be observed for traffic lights controlling train and bus traffic. Black/white back panels and sun visors help to improve the visibility. All traffic light examples are visualized with no active lights.

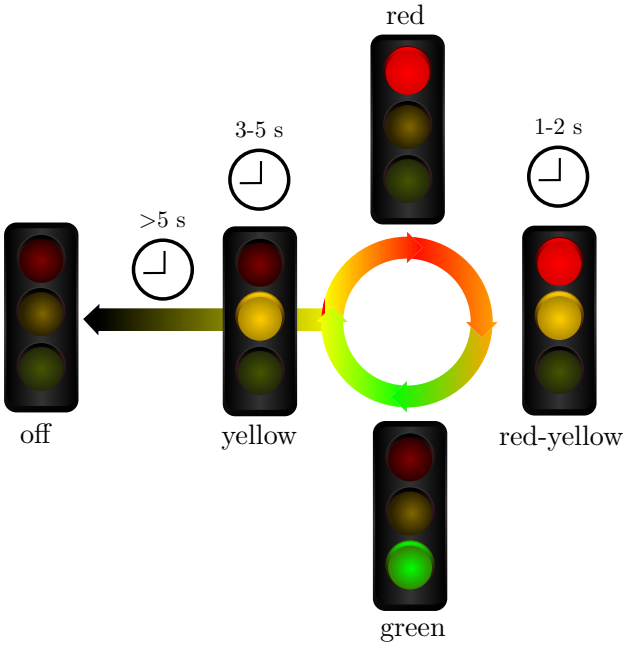


Figure 1.3: Common traffic light sequence on German and Europe streets with the states red, red-yellow, yellow and green. A yellow state is mandatory (at least 5 seconds) before deactivating traffic lights.



Figure 1.4: Common traffic lights controlling straight traffic. The pictogram is either circular or straight-arrow-shaped. A noticeable difference between the state green and the others can be seen.

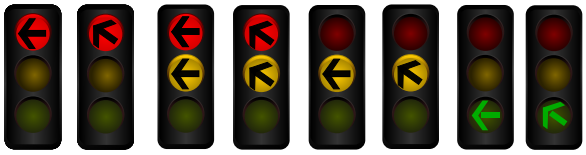


Figure 1.5: Common traffic lights with left arrow pictogram. For the states red and yellow, a colored circle with black arrow is used. For the state green a colored arrow is used.



Figure 1.6: Common traffic lights with right arrow pictogram. A noticeable difference between the state green and the others can be observed. An additional sign with a green arrow allows a protected right turn under caution.

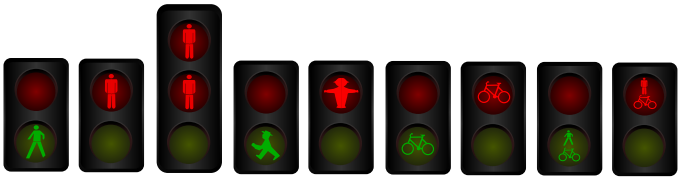


Figure 1.7: Variety of the most common pedestrian and cyclist pictograms. Due to historic reasons two pedestrian signs exist.

1.2 Challenges

A closer look on todays driver assistance systems shows, that only one manufacturer offers a vision-based traffic light assistant in beta mode, but there already exist several systems for pedestrian, cyclist or vehicle detection on the market. Already on driver assistance level, a traffic light assistant could avoid violations of red traffic lights at intersections or allow the driver to rest during red cycles. When it comes to detection, which describes a localization of traffic lights in camera images, there exist known problems, which make reliable detection difficult. Moreover, these problems also impact state and pictogram classification, which are necessary to solve the overall task. In the following, challenges for camera-based traffic light recognition are listed.

Object resolution: Due to requirements on real-time capability, sensors with large image resolutions are critical in automotive applications. Traffic lights are objects of small size, which moreover have to be detected at high distances to ensure comfortable braking. The detection of objects of small resolution is still a state of the art problem, which causes a higher number of misdetections. Moreover, the resolution shows an even stronger impact on pictogram classification due to the high spatial frequency. In easy words, pictograms are very rich in details, which blur for small resolutions. Pedestrian traffic lights at only several pixels resolution often look similar to traffic lights with a circular pictogram. An example scene showing far traffic lights at low resolution is shown in Figure 1.8a.

Light conditions: The two main features characterizing traffic lights are the active lights and the traffic light housing. A further potential feature can be poles, on which the housing is typically installed. While a dark environment often highlights the active lights, it complicates the detection of the housing. At another extreme, strong sunlight, in particular a low sun, leads to adverse effects on the recognition of the active lights. A cloudy environment on the day is beneficial for traffic light detection and state recognition. Two examples see Figure 1.8b.

Sources of False Positives: While humans rarely classify a non-traffic light as traffic light, this frequently happens when using traditional traffic light detection methods. Especially the urban environment contains many sources of false positives, such as rear lights of vehicles, window facades of buildings or traffic signs. In general, false positives often share the most common features with true traffic lights: active lights/blobs, black regions of rectangular shape and objects close to poles. Without considering context information, many image regions containing dark regions with bright blobs appear traffic light-similar and challenge machine learning or image processing approaches, examples see Figure 1.8c.

Appearance: Although the types of traffic lights are regulated by law, there is still a large variation in the appearance of traffic lights in camera images caused by various properties and effects:

- **Orientation**: In general, traffic lights are either horizontally (US, Asia) or vertically (US, Europe) installed. Moreover, traffic lights are rotated around the pole for a better visibility in close range. In consequence, state and pictogram of rotated traffic lights are poorly visible in far range. Depending on the position of the vehicle, the appearance of the traffic light in the camera image varies.

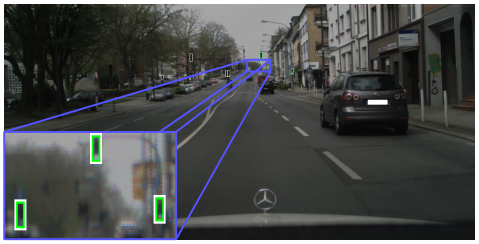
- **Number of light components**: Most common traffic lights consist of three aspects. However, traffic lights with one (often flashing), two or more than three aspects exist.

- **Visors and back panels**: This helps to recognize the traffic light and state in a sunny environment. However, especially the back panels show varying size, shape (round/square corners) or color (white/black).

- **Bulb vs Light-Emitting Diode (LED)**: Nowadays, bulbs are often replaced by LEDs, which are more energy-efficient. Mixed bulb types impact the bulb brightness changes, which can can also influence the recognizability. Bulb intensity is one crucial factor for a successful detection and state classification.

Flashing traffic lights: Especially traffic lights with one or two aspects are often flashing. Yellow flashing symbolizes *caution*, e.g. at pedestrian crossings. From the perspective of state recognition, flashing bulbs are challenging as the camera can exposure exactly during toggling between on and off, which produces ambiguous states.

Lack of standardization: The previous section showed insights into the characterization of traffic lights in Germany. Although traffic lights basically look similar, there exist differences in the type of traffic light back panel, sun visor or the type of pedestrian light. Especially for the latter many exceptions exist. Different versions of the *Ampelmännchen*, the *Elvis Presley* traffic light in Friedberg, the *Martin Luther* traffic light in Worms or animated *Ampelmännchen* are easy to interpret for humans. However, rarely captured special pictograms challenge machine learning-based approaches, which require a high amount of training samples. Examples see Figure 1.8d.



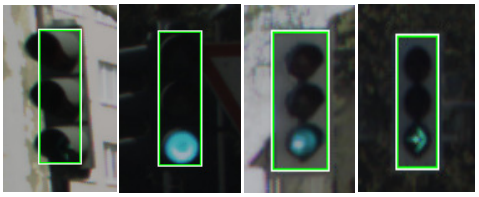
(a) Object resolution



(b) Light conditions



(c) Common false positives



(d) Appearance variations

Figure 1.8: Object resolution, light conditions and appearance variations are common challenges for traffic light detection systems. Rear lights of vehicles, traffic or street signs are frequent sources of false positives. In general, regions containing blobs, dark background or vertical structures (e.g. poles) are potential false positives.

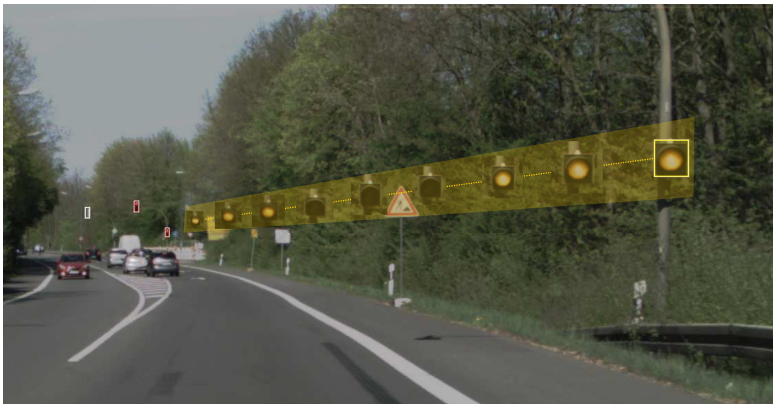


Figure 1.9: Flashing traffic lights are challenging for image-based traffic light detection. Because images capture snapshots, ambiguous states can occur in case the camera exposures while the state toggles between on and off.

Chapter 2

Related Work

In the early 2000s only a few works on traffic light recognition were published, whereas over the last years the number of publications strongly increased. On the one hand, this can be explained by the rising number of datasets allowing more researchers to work on this topic. On the other hand, more and more manufacturers and universities work on the topic of traffic light recognition as part of automated driving systems. Most vision-based recognition systems rely on the architecture shown in Figure 2.1. After image acquisition and converting the image into the desired color space, object hypotheses are generated. Object hypotheses generation is performed on a single image and generates object candidates aiming to cover all existing objects and generating as less false candidates as possible. In order to reject false candidates and attach a confidence value to each hypothesis, a classification step is performed. This can either be a binary classification (detection only) or a multiclass classification (detection and state recognition). Depending on the system manifestations, other system components may be added, such as additional verifiers, map integration or others. The following introduces traffic light research categorized into the publications addressing traffic light detection as well as state and pictogram recognition, respectively.

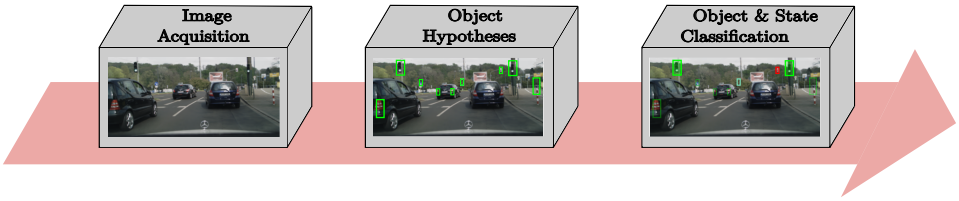


Figure 2.1: Common traffic light detection pipeline used in most systems: image acquisition and transformation, hypotheses generation and classification.

2.1 Research on Traffic Light Detection

Most early detection systems rely on three fundamental traffic light features: *color, intensity gradient and shape*. These state of the art methods are presented in terms of conventional traffic light detection methods in Section 2.1.1. Modern methods using CNNs instead of hand-crafted features are introduced in Section 2.1.2.

2.1.1 Conventional Traffic Light Detection

Color-based methods: This group represents the majority of detection approaches. Color-based approaches aim to extract the traffic light lamp colors, i.e. red, yellow and green from the background. Many works use the common RGB color space [DCS12; FD18; LKK04; OO10], others report benefits of other color spaces such as HSL/HSV [CHL+11; GJX+10; LADT11; MS15; NDZ10; WJYW11] or CIELab [HMUI15]. The segmentation approaches differ in using simple thresholding operations [CHL+11; DCS12; GJX+10; LADT11; MS15; OO10] or using machine learning approaches, such as Support Vector Machine (SVM), k-Nearest Neighbors (kNN) and others [FD18; LKK04; NDZ10]. Figure 2.2 illustrates a segmentation result from [FD18; Fre21] for the colors red, yellow and green. While the traffic light colors are successfully segmented, also background pixels from traffic signs, house roofs and rear lights of other vehicles are extracted.

Intensity gradient-based methods: This group represents approaches typically being applied to a grayscale input image searching for bright regions circumscribed by darker regions. The most popular approach is the spotlight detector from de Charette et al. [DN09a], which is used by various others in slightly modified versions [FMD17a; NDZ10; TPBN14]. Spotlight detection is based on a grayscale input, to which the white top-hat operator is applied. This leads to a binary image highlighting bright regions smaller than a defined structuring element. An edge detection algorithm returns edges, in particular traffic light lamp edges. Using connected component analysis, blobs can be extracted. Most approaches use other heuristic filters, e.g. size, aspect ratio or convexity filter to reduce the number of blobs per image. Figure 2.4 illustrates the results of the most important spotlight detection interim steps applied to a DTLD scene [FMKD18].

Shape-based methods: The shape of the traffic light lamp is the third property, which is used by various approaches for detection. However, there only exist approaches to describe the circular shape using hough circle transform. These methods [LKK04; OO10] mostly use a minimum and maximum circle radius to filter inappropriate results. [CHL+11] use an ellipsoidal formulation to overcome distortion

and occlusion effects. Figure 2.3 illustrates a result of hough circle transform applied to a scene of DTLD. Besides the traffic light lamps, trees and traffic signs are common sources of false positives.

Others: Other approaches are based on map information [BMP15; FU11; LADT11], which is used to project objects into the camera images leading to regions of interest. The use of stereo information is presented in [FMD17b] showing a significant reduction of false positives and detection accuracy by using 3D filters, homogeneity filters and size enhancement of previous detection steps. HOG features are used in various works [BMP15] as well as haar-wavelets in an AdaBoost classifier.

Classification Methods: Most detection methods based on conventional hypotheses generators use a subsequent classification module to reduce the number of false positives. Histograms of Oriented Gradients (HOG) [BMP15; SMA15] are frequently utilized features and often combined with a SVM. Instead, other works use Local Binary Patterns (LBP) features [SMA15]. Besides the ability to describe traffic lights, HOG features also are suitable to classify the traffic light state for three aspects traffic lights. Other works also use a multilayer perceptron for classification. Haar wavelets are often used as features [SMA15], e.g. combined with an AdaBoost approach in [DN09b] mainly focusing on features describing edges as a dominant characteristic of traffic lights.

2.1.2 CNN-based Traffic Light Detection

The *DeepTLR* [WWZ16] is the first published CNN designed for traffic light detection. It uses a segmentation approach to distinguish between traffic light and background pixels. A bounding box regression is applied to receive the final bounding box detections. Bach et al. [BRD17] also use a segmentation approach in a multi-camera system as well as a Faster R-CNN approach [BSD18] for traffic light detection as well as state and pictogram classification. They show an extensive evaluation considering different requirements on object resolution concluding issues in detecting very small objects. Behrendt et al. [BNB17] use a You Only Look Once (YOLO) [RDGF16a] approach to detect and classify traffic lights as well as an additional network for tracking. Kim et. al [KHJ18] present a work on the impact of color spaces for CNN-based traffic light recognition. They report, that the conventional RGB and normalized color space performs best, with a significant problem in detecting the state yellow. In another work, [KYHJ19] present a binary segmentation approach combined with an eight-neighbour connected components approach for hypotheses generation. Instead of using a two-stage approach, the authors utilize a fully-connected network for multiclass classification. They report a benefit for smaller objects due to the use of a pixel-based approach.



Figure 2.2: Traffic light candidates obtained from color segmentation.



Figure 2.3: Traffic light candidates obtained from hough circle detection.

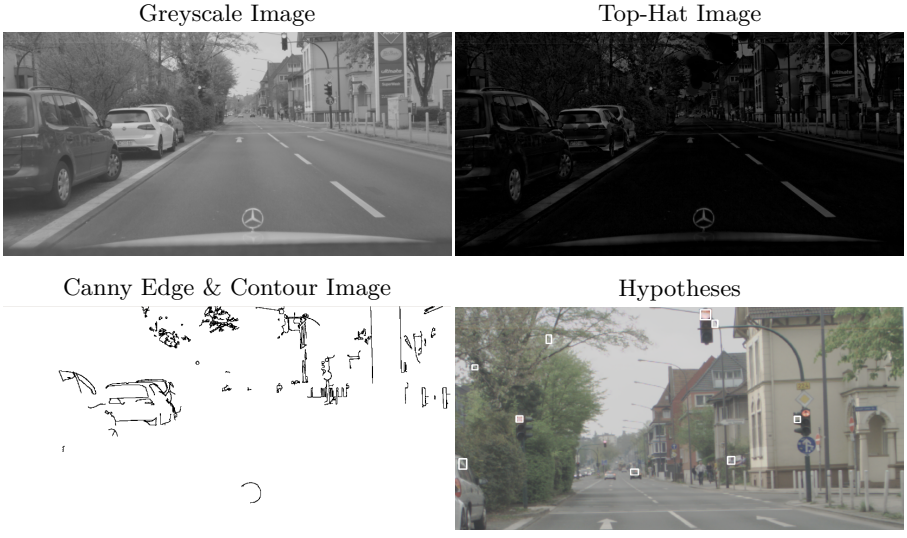
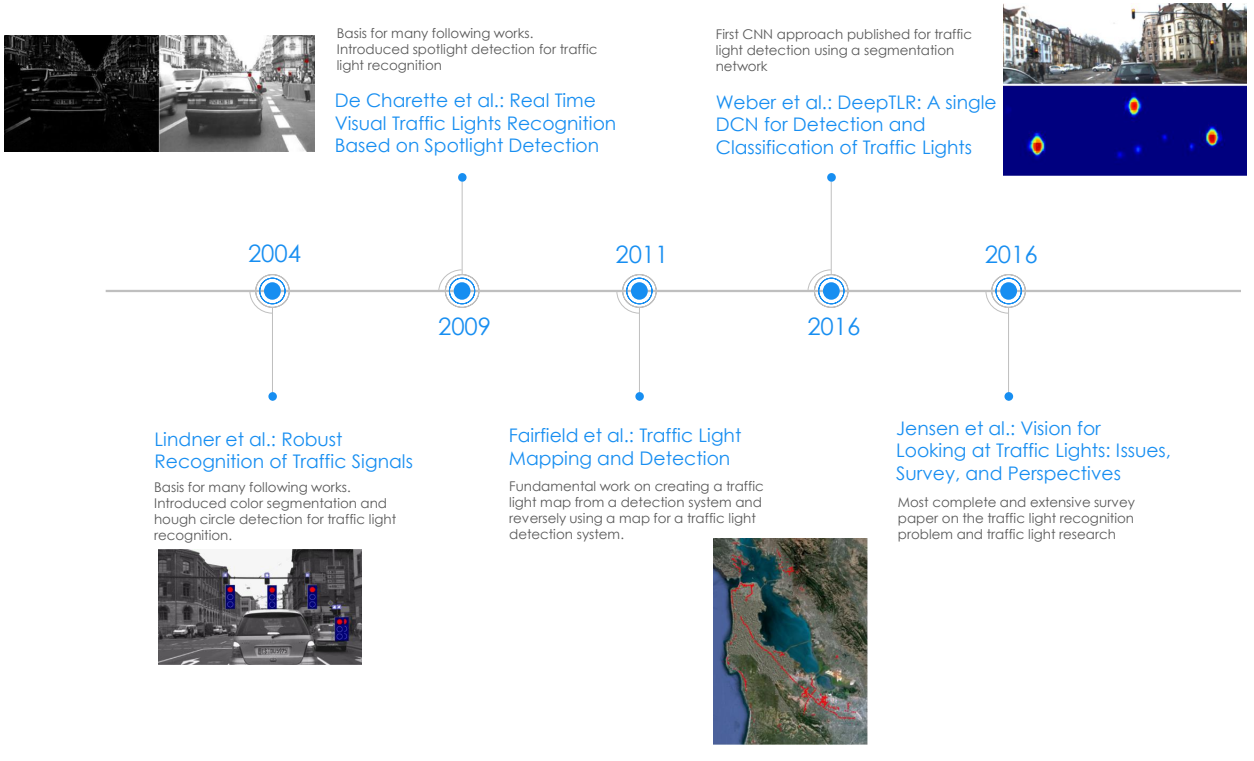


Figure 2.4: Traffic light candidates obtained from spotlight detection.

Weber et al. [WHZ18] enhance their DeepTLR network by also predicting the state and arrow information in a hierarchical manner. They are able to detect objects down to a width of 4 pixels at 12 frames per second, whereas a clear decrease in recall can be reported for small objects. [PGC+19a] present an approach combining prior information in the form of a created map with a CNN-based detection system based on the YOLO-v3 detector [RF18]. Their prior map is generated by using a Light Detection and Ranging (LiDAR) point cloud projected into the camera image. LiDAR points inside a 2D bounding box detection are then projected into global coordinates. Because of bounding box misalignments, sparse LiDAR points for small, distant objects and small imprecisions of localization, they mention the need for manual correction. They conclude the requirement for a higher performance in order to be able to use the system on road, especially to solve the problem of finding the relevant traffic light. Kim et al. [KCH+18] present a traffic light recognition approach for self-driving vehicles tested in the San Francisco Bay area. They use a SSD approach with a subsequent spatio-temporal refinement module reducing the number of false positives. The results after spatio-temporal refinement are passed to the vehicle controller.

The first publication tackling the problem of relevancy described in [FD16] is [LW18]. They present a CNN taking detections and the camera image as an input and predicting relevancy property for each detection. They use an output vector mapping, in which the output vector length equals the image width. The position and width of the detection automatically determines the position in the output vector, which is selected for relevancy decision. They extend this work in [LLW19] to predict the traffic light to lane assignment for all relevant traffic lights. For this purpose, additional information such as road arrows, direction signs and road lines are used. All experiments are compared with the human baseline. The performance turns out to be comparable for easier scenarios, however in difficult scenarios humans outperform the network.

Parts of Chapter 5 are published in [MD18] describing a SSD approach for traffic light detection. The main modifications are adaptive prior boxes and a change in base network leading to a performance gain in detecting small objects. The principle of adaptive prior box locations is also used by Lee et al. [LK19] The authors use an encoder-decoder network based on a ResNet architecture trained with a focal loss approach [LGG+18]. In their experiments, their methods are able to outperform a Faster R-CNN [RHGS17] Feature Pyramid Network [LDG+17] approach on the Laboratory for Intelligent and Save Automobiles (LISA) dataset [MTM12] and Bosch Small Traffic Lights Dataset (BSTLD) [BNB17]. [BCKW19] use a CNN for traffic light detection as well as countdown timer detection, which is frequently installed close to traffic lights in China.



Adapted and reprinted with permission from [LKK04] © 2004 IEEE, [DN09a] © 2009 IEEE, [FU11] © 2011 IEEE, [WWZ16] © 2016 IEEE.

Figure 2.5: Timeline of popular traffic light publications [DN09a; FU11; JPM+16; LKK04; WWZ16] in terms of citations, impact and novelty.

2.2 Research on Traffic Light State Recognition

Most early methods utilize color-based approaches which implicitly include a state recognition. Those methods are not further discussed in this section. Non-color based conventional detection methods or CNN-based methods do not perform an implicit state determination. The following state of the art in state recognition refers to hand-crafted features for state classification and CNN-based state classification, respectively.

Hand-crafted Features: Besides color-based solutions, there is little research on alternative features. This can be explained by color being the only feature expressing the correct state in all cases. While active light position is also one prominent feature, it does not represent the correct state in all cases, for instance in the case of one aspect traffic lights. However, some works focusing on three aspects exclusively use model-based approaches. Chiang et al. [CHL+11] use LBP to create feature vectors for each detection RoI. The feature vectors are classified by a SVM into four classes: red, yellow, green and non-traffic light.

CNN-based State Recognition: [PGC+19b] predict two classes, red and green, from an Yolo-v3 architecture without any further network or loss modifications. They mix yellow samples into the red class and report problems predicting red-yellow traffic lights due to the lack of sufficient samples. [MD18] extend a Single Shot Detection approach to also predict the traffic light state. This is done by extending the loss function and adding an additional convolutional layer separately predicting five state confidences. A rather unconventional proceed is presented in the work of Saini et al. [SNK+17]. Based on conventional traffic light detection (MSER blob detection and HOG-SVM), the state recognition is performed using a CNN. They use a trimmed-down LeNet network with 2 convolutional and 2 pooling layers and fully-connected layers to receive the output confidences. They also train only two outputs: green and red. [KYHJ19] also use a LeNet for classification, however they already used a CNN approach for the detection step. The state and inlay classification is performed only on the detection regions - and their classification network is larger than the one previously used. Their system predicts three classes red, green and yellow, but with a poor performance on yellow traffic lights.
Fern et al. [FGSS18] present a comparison of traffic light datasets for state recognition. They present a quantitative comparison and evaluate their methods on all datasets. A comparison between a ResNet and MobileNet architecture as well as a random forest approach is presented. The first two methods outperform the random forest method, however the recall on the state yellow is significantly lower than for red and green. For CNN training, a weighted cross-entropy loss is used to tackle the data imbalance.

2.3 Research on Traffic Light Pictogram Recognition

There has been little research on the topic of traffic light pictogram recognition. One main reason is the lack of traffic light datasets containing pictogram labeling information. Prior to the publication of LISA dataset in 2015, no dataset contained pictogram labels. The Bosch Small Traffic Lights Dataset of 2017 also contains arrow information. However, both datasets only include some types of inlays - but do not cover all possible types appearing on streets. The first dataset containing several arrow inlays, as well as pedestrian, bus and cyclist inlays is the DriveU Traffic Light Dataset published in 2018. Due to the short period of time, less publications were able to address this topic and suggest proper solutions. Similar to the previous section, the following state of the art is separated into systems using hand-crafted features and CNN-based solutions.

Hand-crafted Features: [MS15] use HOG features in combination with three different classifiers, namely kNN, Linear Discriminant Analysis (LDA) and SVM. A previous step extracts the active light bulb, which helps to remove redundant information. They report, that the classification results get clearly more stable for higher object resolutions. Moreover, pictograms on red lights seem to be more challenging than on yellow and green lamps. Shi et al. [SZZ16] also use HOG descriptors calculated on R, G and B image channels to classify into circular, arrow straight, arrow left and arrow right. They present promising results on the inlay recognition. The utilized dataset is rather small and contains less than 100 examples for some classes. Cai et al. [CGL12] present a classical traffic light recognition system, which detects bounding boxes around the active lights in a first step. The arrow recognition is then based on Gabor wavelet transform by applying 2D gabor filters to receive a feature vector. This feature vector is classified by using a nearest neighbour approach. Unfortunately, the evaluation is not extensive enough to allow a deep interpretation of the system abilities. Said et al. [SHA17] use simple features such as the bulb area, bulb compactness, bulb ratio and others for classification of inlays. There are no evaluations proofing the operability of the approach.

CNN-based Pictogram Recognition: Weber et al. [WHZ18] present a hierarchical approach, which classifies the object in level 1, the state in level 2 and the arrow direction in level 3. Unfortunately, the evaluation does not provide extensive evaluations for each task, which makes an interpretation of the pictogram results difficult. [CH16] describe an entire system consisting of traffic light detection, state and arrow classification and a tracking module. The traffic light detections are fed into a Principal Component Analysis (PCA) classifier consisting of learned filter banks combined to a histogram and classified by a SVM. All recognition results

are combined into one result metric giving less insights into the arrow classification results. Bach et al. [BSD18] extend a Faster R-CNN approach to also predict the traffic light inlays. They use an image-wise balancing scheme to tackle the unbalanced inlay labels. The evaluation shows promising results, whereas the recognition results decreases for straight and especially right arrows - according to the number of samples in the dataset.

Others: John et al. [JZM16] use map information containing arrow directions to determine the traffic light pictogram information during runtime. The detected traffic lights are matched to the closest traffic light provided by the map. They mention another vision-based method for arrow detection using LAB color-based thresholding, which is not clarified in detail. Fairfield et al. [FU11] also determine the pictogram information from map.

2.4 Discussion

A closer look on the state of the art showed, that most approaches use color or intensity-based approaches. In particular, color-based approaches are benefitial, because the detection step implicitly determines the traffic light state. Most recently, those approaches are displaced by the use of CNNs. The main characteristics of the mentioned conventional methods and CNN-based methods are interpreted in Figure 2.6. It illustrates a kiviat chart for the presented common state of the art methodologies and compares their ranking with respect to the properties recall, false positive rate, small object detection performance as well as runtime and resource requirements. Values towards the outside direction symbolize good properties, whereas values towards inside symbolize poor properties.

A striking property of all conventional methods is a poor false positive rate. This is caused by the type of features, color, bright spots and circles, which are numerous in each single image. Traffic light colors appear at vehicles, rooftops, traffic signs, bright spots appear in trees, vehicles, buildings. Even circular and elliptical shapes appear more frequent than generally assumed, indeed making all three features appropriate to detect traffic lights but many background objects as well. The great advantage of the mentioned approaches is simplicity, less demand on amount of labeled data and their lower requirement on resources.

The observable replacement of conventional methods by CNN-based approaches is due to several reasons. On the one hand, many open-source datasets were published [BNB17; FMKD18; Lara13; MTM12] containing image data and annotated traffic lights. These enable the use of learning-based methods in the first place also

for researchers of institutions, which can not fall back on own datasets. Labeling datasets of sufficient quantity and quality requires high expenditure of time and money. On the other hand, progress in parallelization hardware (especially Graphics Processing Units (GPUs)) allows CNNs to be deployed in real-time. Besides performance, this is one of the most important criterion of a traffic light detection system. The main strength of CNN-based detection is the detection performance in terms of both, reaching high detection rates and low false detection rates. This is mainly due to the ability to use context information. A still open problem is the detection of very small objects, i.e. objects with low resolution caused by the commonly high depth of base networks using a high number pooling layers. While this increases the ability to describe large and complex objects, it often comes along with a loss in local information required for small object detection. While small object detection is not as important for many object detection tasks, it is essential for traffic light recognition as traffic lights have to be detected in far range to ensure comfortable braking.

There is a high number of research on traffic light state recognition when considering

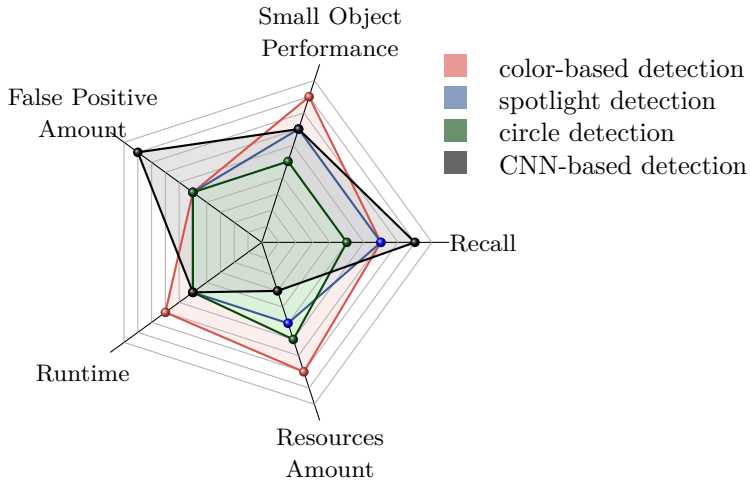


Figure 2.6: Kiviat chart comparison of state of the art methods considering five main properties: recall, false positive rate, small object performance, runtime and resource amount. Values towards the outside direction symbolize good properties, whereas values towards inside symbolize poor properties. For instance, CNN-based methods reach low false positive values (good) at high recall values (good) with the bottleneck of a high amount of resources (poor).

color-based solutions. Alternative methods are rare - especially due to the fact, that the color-based solution is the only valid solution. Other features - like the position of the active lamp (for instance by using HOG features) - may work in many cases, but not in all. One example are traffic lights of one aspect (one light bulb). The state can only be determined by color information. Utilizing light position should be treated with care. Since the use of CNNs for traffic light detection, state recognition is frequently performed in addition to the detection task. Most solutions work well, with a consistent problem in detecting yellow states due to the lack of sufficient samples. In terms of state recognition one noticeable advantage of CNNs is the ability to describe the state off. This state is not addressed by a single publication - because it is hardly solvable by any conventional approach, due to the missing light feature. However, a recognition of this state is necessary for autonomous vehicles - and avoiding misclassifications with other states is still a state of the art problem.

The presented publications on the topic of traffic light pictogram showed a lack of proper solutions. This is, on the one hand, caused by the lack of datasets offering a high number of labeled pictograms. Most datasets only label a subset of all possible pictogram types. A further challenge is then data imbalance, arrows are rare and circular traffic light bulbs are more frequent. Highly unbalanced data is a challenge for modern object detectors. The presented works often lack of presenting pictogram-exclusive evaluations. Instead, they include the classification performance into a single scalar metrics, which makes an estimation of the performance difficult.

Chapter 3

Outline and Main Contributions

The goal of this work is to boost the recognition performance on the task of traffic light detection and attribute classification by applying and adapting a well-established object detection methodology, namely single shot detection. Hereby, the focus is not on developing an entirely new approach for object detection using convolutional neural networks. Instead, an existing methodology is adapted to face the challenges of the traffic light recognition use case. Because traffic light detection can also be referred to as a small object detection problem, the most important components limiting the performance of small object detection are particularly examined.

Besides the detection of traffic lights, a prediction of traffic light attributes is necessary to design entire traffic light recognition systems. From the perspective of application, the traffic light state (or phase) is one crucial attribute. It determines whether a vehicle has to break, should prepare to break or is allowed to pass the intersection. Therefore, the approach used for detection is extended in two different ways to predict the traffic light state.

A further important attribute is the traffic light pictogram indicating to which road user (vehicles, pedestrians, cyclists, tram, bus, ...) or direction (straight, turn-left, turn-right) it refers. Analogous to state prediction, different ways to extend the detection approach are presented and evaluated.

Special attention is paid to the evaluation of all three tasks. Therefore, the largest traffic light database is used, which is published in the course of this thesis for public use. Evaluations are done in an extensive and application-oriented manner showing the recognition abilities, but also showing limitations and giving an outlook on further improvement possibilities.

Because both, detection and attribute confidences, turn out to be not well-calibrated, Chapter 7 focuses on confidence calibration. This helps to align the confidence values

to the true correctness likelihood. This is an important criterion from the perspective of safety and for using the object detections in a multi object tracking framework.

A further important topic addressed in this thesis is the impact of relabeling cycles on the recognition results. Because modern object detectors are able to detect objects at very small false positive rates, the requirements on the quality of datasets is rising. This work shows, that many operating points cannot be evaluated due to the quality of datasets. It requires several relabeling cycles to rise the dataset quality in order to obtain meaningful results.

3.1 Traffic Light Detection using Single Shot Detection (SSD)

A view on the state of the art in traffic light detection presented in Chapter 2 showed, that the current trend moves from conventional feature extraction to the use of convolutional neural networks. This follows the trend in other object detection problems, where CNNs clearly outperform earlier methods. Due to the performance advantages, this work makes use of CNNs for traffic light detection. During the last five years, one methodology has mainly established for object detection, the so called Single Shot Detector (SSD). SSD is one of the fastest CNN-based object detection approaches, which makes it predestined for traffic light detection. Because these approaches are mainly developed on datasets containing objects with high resolutions, several adaptations are necessary.

The main contributions and novelties in this chapter can be summarized as:

Tackling the small object detection problem: One crucial component of all modern CNN-based detectors is the design of reference boxes, also referred to as *anchor boxes* (Faster R-CNN) or *prior boxes* (SSD). These reference boxes are predefined in a fixed grid and all objects are predicted relative to these reference boxes. Adapting the sizes (width and height) of the prior boxes is done by developing a weighted k-IoU approach, a modification of the well-known k-means clustering. Moreover, the approach is modified to support a class weight for underrepresented traffic lights classes with one, two and four aspects. This avoids overfitting towards traffic lights with three aspects and enables the possibility to reliably detect traffic lights with one, two or four aspects. To the best knowledge, all other works focused only on a detection of traffic lights with three aspects.

Adaptive Prior Box Locations: All methods use a similar approach for de-

termining the prior box locations. Depending on the feature map size and the size of the input image, the feature cell size is calculated (mapping of one pixel in the feature map to a region in the input image). All approaches place the prior boxes in the centre of each feature cell. This proceed has one main limitation: Deep basis networks with small feature maps consequently will have a larger prior box stride (distance between prior boxes), which is adverse for small object detection. On the other hand, very early layers will be beneficial due to small strides, but the features are more powerful in deep layers. *In order to avoid the disadvantage of large strides, this work proposes to use adaptive prior box locations, which makes the prior box stride (step size) independent of the feature map size.* This thesis shows a mathematical way to calculate the prior box locations with respect to the prior box size in order to ensure a detection of very small objects. The evaluation proofs the powerful impact of this proceed - especially for very small traffic lights.

In-Depth Optimization: During recent years, several publications also used one-stage or two-stage detectors for traffic light detection. However, most works lack of thorough investigations and adaptations in order to guarantee an optimal network configuration. There is a large variety of optimization possibilities using CNNs. Naturally, not all possibilities can be investigated. However, by using a reasonable proceed many parameters can be optimized. This thesis investigates the most promising parameters, including:

- **Data preprocessing**: This includes the preprocessing of the images on the one hand and the labeled data on the other hand. Different configurations of positive/negative sample selections are presented. Moreover, a parametrization of ignore regions is implemented, which are neglected during training step and may help the network to focus on the *relevant* cases.

- **Data augmentation**: Data augmentation techniques are investigated, which turned out to be successful in related publications.

- **Training data amount**: It is shown, how the amount of training data influences the detection result. Results of this investigation show whether utilizing more data helps to further improve the performance.

- **Network depth**: It is investigated in which feature layer the prediction is optimal. This is related to finding the optimal receptive field size.

- **Further hyperparameters**: Parameters controlling the optimization of the network are optimized in a last step. This guarantees an optimal training convergence without overfitting.

In-Depth Evaluation: The evaluation methodology and metrics are tailored to

the traffic light recognition task. This covers the requirements on the distance in which objects have to be detected, but also the types of traffic lights, which are required to be detected. It is evaluated, how well the approach is performing on different test subsets. Moreover, the absolute number of training and test images is higher than in all other published works to date. This is due to the use of DTLD, the largest traffic light dataset in this field. This helps to obtain more meaningful results - also when only evaluating on subsets of the entire dataset. The extensive label attributes help to further distinguish the types of not detected traffic lights. The evaluation also presents the types of falsely predicted traffic lights (i.e. false positives). This investigation shows the amount of true errors and labeling errors. An automatic false positive categorization using a scene segmentation approach is presented, which determines frequent sources of false positives.

3.2 Extending SSD for State and Pictogram Recognition

In addition to traffic light detection, two methods for additionally predicting the traffic light status and pictogram are developed and presented. Within Chapter 6, the following contributions and novelties are presented. To the best knowledge, no other publication compares the ability of one and two-stage methodology for traffic light attribute classification.

One-Stage Attribute Prediction: An extension of the well-established SSD approach is presented, which enables to predict status and pictogram. The prediction of both attributes is physically independent of the detection and confidence prediction guaranteeing less negative impact on the detection itself. Therefore, the SSD framework and loss function is adapted and ignore regions are implemented in order to independently select suitable training samples for each task. Metrics are developed, which are able to select the optimal training snapshots with respect to the balanced performance of all three tasks.

Two-Stage Attribute Prediction: As a second methodology, Chapter 6 presents a two-stage approach for state and pictogram prediction. Both attributes are predicted by a separate, subsequent CNN which operates on the Region of Interests (RoIs) determined by the previous SSD detection step. Since high receptive fields are not necessary for local features such as color and shape, the network is designed as a light-weight network guaranteeing both, low runtime and high accuracy.

Best of Two Worlds: Based on numerous optimizations and evaluations, it

turns out to be the most beneficial to combine both methods. To the best knowledge, no other work used this methodology to combine the advantages of high receptive fields (implicit classification) and small receptive fields (explicit classification) depending on attribute and class. The optimal methodology performs a target road user classification (differentiation between vehicle, bicycle, tram and pedestrian) in the implicit SSD framework and a subsequent inlay classification for vehicle pictogram shapes purely relying on local features inside the bounding box.

In-Depth Optimizations: Similar to Chapter 5, extensive optimizations are developed and performed in order to boost the recognition performance. For both methods, data augmentation, label filtering and preprocessing are investigated. For the implicit classification, selecting the optimal prediction feature maps is one crucial task, which is also related to the adaptive prior box proceed developed in the previous chapter. The results proof, that those profound optimizations have a significant impact on the recognition result. None of the state of the art publications offer a similar amount of optimizations and investigations.

3.3 Confidence Calibration

Investigations on the calibration quality of the SSD output showed, that both, detection and attribute confidences, are poorly calibrated. The desired property is that confidences express the true correctness likelihood. This is required, because traffic light detection finds application in safety-critical systems for driver assistance and automated driving.

Object Confidence Calibration: This thesis presents two methods for confidence calibration. The detection confidences are calibrated using a histogram binning calibration and a particularly tailored state-wise histogram binning calibration. The latter calibrates the detection confidences with respect to the predicted state. This helps to close the gap between the detection performance being dependent on the actual state. Moreover, a sample-wise binning approach is utilized instead of fixed confidence intervals, because the output confidences of SSD are not uniformly distributed.

Attribute Confidence Calibration: Attribute calibration is exemplified by state confidence calibration. Therefore, the binary histogram binning calibraton method is extended to a multiclass problem. For almost all classes, the calibration error is reduced. Moreover, the balanced classification result is improved by calibration. Recall on underrepresented classes increases partially by more than 10 percent. Especially off lights are improved, which showed a significant underestimation regarding

the true correctness likelihood.

3.4 Dataset Relabeling - Significance for Performance and Evaluation Reliability

The mentioned in-depth evaluations of detection as well as attribute classification showed impressive performance values. Compared to recent conventional detection methods, CNNs are able to reach very low false positive rates. *When observing the error cases of the presented detection approach determined by evaluation, many system errors turn out to be labeling errors.* Especially for operating points with decreasing false positive rates, the percentage amount of labeling errors increases. This proofs, that CNNs are able to catch up with human performance or even exceed it in some cases.

The utilized DriveU Traffic Light Dataset is labeled by human labelers and extensive quality control is not performed. When observing the labeling errors determined by the potential CNN error cases, it shows, that many labeling errors have similar patterns. Humans are often influenced by where humans expect traffic lights to appear. In some frames, traffic lights appear in unusual image regions, such as through bushes, trees or mirrored in windows. In those cases, CNNs are more unbiased and can support the human labeling process. Because the evaluation of traffic light detection systems require reliable evaluation numbers, Chapter 8 presents an in-loop relabeling cycle, in which relabeling objects are determined by the evaluation of the system. These relabeling objects are proposals, which are verified by human labelers and corrected if necessary.

After relabeling, two key questions are answered. *Which impact does rising dataset quality have on the system performance?* In order to investigate this, two networks are trained (training set with and without relabeling) and evaluated on a similar test set. The second key question is: *Which impact does dataset relabeling have on the evaluation results?* This question shall give insights into the reliability of results, which can be expected. It shall stimulate researchers to rethink evaluation results, which are strongly dependent on the test set quality, especially with regard to modern object detectors reaching human-similar performance. Special attention and effort can not only be paid to improve algorithms, but also to the data itself.

3.5 Publications

In the course of this thesis, several conference contributions were published, which are listed in the following. Some publications are part of the present thesis - some others are not part of this thesis, but topics of research during recent years.

[FMD17b] *A. Fregin, J. Müller and K. Dietmayer. Three Ways of Using Stereo Vision for Traffic Light Recognition. IEEE Intelligent Vehicles Symposium, 2017.*

[FMKD18] *A. Fregin, J. Müller, U. Kreßel and K. Dietmayer. The DriveU Traffic Light Dataset: Introduction and Comparison with Existing Datasets. IEEE International Conference on Robotics and Automation (ICRA), 2018.*

[FMD17a] *A. Fregin, J. Müller and K. Dietmayer. Feature Detectors for Traffic Light Recognition. International Conference on Intelligent Transportation Systems, 2017.*

[MD18] *J. Müller and K. Dietmayer. Detecting Traffic Lights by Single Shot Detection. International Conference on Intelligent Transportation Systems, 2018.*

[MFD17] *J. Müller, A. Fregin and K. Dietmayer: Multi-Camera System for Traffic Light Detection: About Camera Setup and Mapping of Detections. International Conference on Intelligent Transportation Systems, 2017.*

[MFD18] *J. Müller, A. Fregin and K. Dietmayer. Disparity Sliding Window: Object Proposals from Disparity Images. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2018.*

In the course of the ICRA publication [FMKD18], the DriveU Traffic Light Dataset (DTLD) was published for public, non-commercial use. The data itself, including camera images, disparity images as well as labeled data can be downloaded after registration. More information are available at `http://traffic-light-data.com/`.

Chapter 4

Dataset and Evaluation Metrics

Evaluation is one of the most crucial topics when designing modern machine learning approaches. On the one hand, evaluation is needed to rate the abilities of a system or its components. On the other hand, it also shows the weaknesses laying the foundation for continuous improvements. Evaluation can be done in a quantitative (using mathematical measures) and qualitative manner, i.e. visually inspecting results. This chapter describes tools and metrics used for a qualitative evaluation within this thesis.

An overview of published traffic light datasets is presented in Section 4.1. The selected dataset for this thesis, the DriveU Traffic Light Dataset, (DTLD) is introduced in Section 4.2 in detail. All important metrics used in subsequent chapters are introduced in Section 4.3.

4.1 Overview of Traffic Light Datasets

Research on traffic light recognition started in the mid 1990s using grayscale and color images. Up to 2010, all publications evaluated their systems using non-public datasets. During years four main traffic light datasets were published. Other datasets contain traffic lights besides other object classes or other publications mention private datasets, which were not published.

LaRA [Lara13]: The Traffic Lights Recognition (TLR) public benchmark of La Route Automatisee (LARA) in 2010 is the first public traffic light dataset. It contains 11 179 images with a resolution of 640×480 pixels in 8 bit depth recorded in Paris, France.

LISA [MTM12]: In 2015, the Laboratory for Intelligent & Save Automobiles (LISA) started the *Intelligent Vehicles and Application (VIVA)* challenge, where the

Table 4.1: Traffic light dataset comparison. DTLD contains the highest number of labels, the highest image resolution and additional depth data. From visual inspection DTLD also showed the best properties. However, LISA is the only dataset containing bulb labels and night time annotations.

		LARA [Lara13]	LISA [MTM12]	BSTLD [BNB17]	DTLD [FMKD18]
QUANTITATIVE	**Image resolution [pixels]**	640x480	1280x960	1280x720	2048x1024
	Image depth [bits]	8	8	8 & 12	8 & 16
	Frame rate [1/s]	25	16	16	15
	# Images	11,179	43,007	13,427	40,979
	# Objects	9,168	51,826	24,242	232,039
	# Cities	1	1	17	11
	Depth data	✗	✗	✗	✓
	Calibration	✗	✓	✗	✓
QUALITATIVE	**Human annotated**	✓	?	✓	✓
	Label completeness	- -	-	+	+
	Label consistency	+	- -	++	++
	Image quality	-/+	+	+	++
	Weather/Daylight variance	- -	++	+/-	+
LABEL ATTRIBUTES	**Bounding box**	✓	✓	✓	✓
	Bulb Bounding box	✗	✓	✗	✗
	State	✓	✓	✓	✓
	Aspects	✗	✗	✗	✓
	Occlusion	✗	✓	✓	✓
	Orientation	✗	✗	✗	✓
	Direction	✗	✗	✗	✓
	Relevancy	✗	✗	✗	✓
	Pictogram	✗	✗/ ✓	✓	✓

Reprinted and adapted with permission from [FMKD18] © 2018 IEEE.

LISA traffic light dataset was published. LISA contains 43 007 images with 51 826 traffic light labels containing additional bulb labels. The images are recorded in 8 bit depth with a resolution of 1280×960 pixels at day and night time. Unfortunately, the labels seem not to be human annotated or are at least partly generated by algorithms. The size of one label object strongly varies in consecutive frames with a visible correlation to the size of the light bulb. This indicates bounding boxes generated by algorithms. Besides the traffic light bounding box, bulb box, state and occlusion are also annotated. Left arrows are marked as well, however without other pictograms such as right and straight arrow or pedestrian.

BSTLD [BNB17]: The Bosch Small Traffic Lights Dataset was published in 2017 and contains annotated traffic lights in street scenes in the San Francisco area. The camera images are published in 12 and 8 bit depth with a resolution of 1280x720 pixels. An overall number of approximately 25 000 traffic lights is annotated in 13 500 images. Worth mentioning is the difference in the extent of labeling when comparing training and test data. While the training data contains class labels including state and pictogram information, the test set only contains state information.

DTLD [FMKD18]: The DriveU Traffic Light Dataset was published in 2018 and exceeds the three mentioned in terms of image resolution, image depth, labeled images and objects. 41 000 images are recorded in 11 German cities containing about 230 000 annotated traffic lights. Furthermore disparity data and calibration is published.

Others: There exists a higher number of non-published datasets, which are mentioned in various publications, from which the most promising in terms of quantity are [BMP15; SMA15; WHZ18]. Moreover, traffic light annotations are part of various other datasets and benchmarks such as Common Objects in COntext (COCO) [LMB+14a], which only contains bounding box labels. Traffic lights are also labeled in segmentation datasets such as Cityscapes [COR+16], however segmentation annotations are not optimal for accurate object detection. Consequently, segmentation data is out of scope for this thesis.

Comparison: A qualitative and quantitative comparison of the four introduced traffic light datasets is shown in Table 4.1 by comparing three main categories. The first statistics are quantitative measures. In this category, *DTLD clearly exceeds the others due to the highest amount of labels, image resolution and depth and the availability of depth from disparity and calibration data.* The second category groups qualitative measures such as label completeness/consistency or image quality. In this category, BSTLD and DTLD stand out. However, some measures may be subjective impressions biased towards the requirements for this thesis. One main weakness of LISA are algorithm-created annotations causing larger label inaccuracies varying from frame to frame. In general, non-human annotated datasets have limited use to train accurate object detectors. BSTLD labels seem to be very accurate but struggle with a bad white balance, which adaptively varies over time. DTLD shows a good label and image quality and weather and daylight variance, but a lack of recordings at night.

The third category, label attributes, compares all attributes which are labeled besides the bounding box. *DTLD contains the highest number of labels, it is the only dataset offering the attributes direction, the number of aspects and all pictogram variants.* Because this offers to design methods predicting these attributes and moreover offers possibilities for evaluation (e.g. excluding/ignoring specific attribute types such as pedestrian traffic lights) DTLD is used for this thesis.

Figure 4.1 compares the label completeness of one example scene for all four datasets. While LARA only labels a subset of all traffic lights (even not all frontal ones), LISA shows a better completeness. However, for incomprehensible reasons some traffic lights are also not annotated. BSTLD and DTLD label all visible frontal traffic lights. In some of the images DTLD also contains labels for all non-frontal traffic

lights. Figure 4.2 compares the label consistency based on bounding box crops. LARA, BSTLD and DTLD show accurate bounding box labels aligned to the traffic light housing. Only LISA shows inaccurately aligned labels, which strengthen the presumption of non-human annotated labels. In many cases, the bounding boxes are too large and strongly vary from frame to frame. This is very critical for training and evaluation.



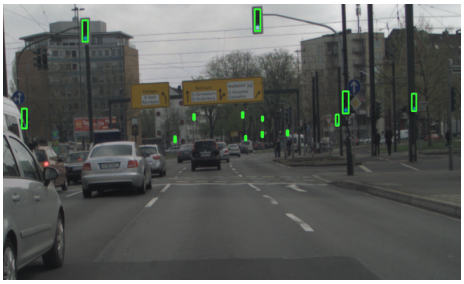
Example of LARA dataset



Example of LISA dataset



Example of BSTLD



Example of DTLD

Reprinted with permission from [FMKD18] © 2018 IEEE.

Figure 4.1: Comparison of the annotation completeness in all four traffic light datasets from [FMKD18]. LARA and LISA dataset do not include fully-annotated frames. They mainly annotate traffic lights controlling the planned vehicle trajectory and down to a minimum object resolution. Frequently, only a small number of existing objects are annotated. BSTLD includes fully annotated frames up to a minimum resolution. Noticeable is the green tint pointing to an incorrect white balancing. DTLD includes fully annotated frames down to the human recognition limit.



Reprinted with permission from [FMKD18] © 2018 IEEE.

Figure 4.2: Annotation comparison between all four datasets from [FMKD18]. The annotation color indicates the traffic light state. In LARA, the traffic lights are consistently annotated. In some cases, objects with a small resolution (see example 5) have inaccurate bounding boxes. The double light in example 4 is caused by motion blur. In LISA dataset, objects are not consistently annotated. In some cases, the traffic lights are concise around the housing, in other cases the traffic lights are annotated with a lot of background. BSTLD provides accurate and consistent annotations. DTLD attaches value to the consistency and quality of annotations. If visible to the human eye, the annotations are annotated concise around the housing.

4.2 DriveU Traffic Light Dataset (DTLD)

The DriveU Traffic Light Dataset (DTLD) published in 2018 [FMKD18] arised in terms of a research cooperation between Daimler AG and Ulm University called *driveU*. This section gives further insights into the dataset including camera images, disparity images and the annotation data. Section 4.2.1 describes the camera and disparity images. The label attributes are presented in Section 4.2.2. Important statistics concerning further chapters as well as an overview on training and test set are shown in Section 4.2.3.

4.2.1 Camera Images

DTLD is recorded in eleven German cities using a stereo camera setup. From this setup, the left raw camera image as well as the disparity image in lower resolution is published. The left raw image has a resolution of 2048x1024 pixels in 12 bit depth. The disparity images have a size of 1024x440 pixels and are received from a modified Semi Global Matching (SGM) algorithm [GF07; Hir08]. The camera image is downsampled before SGM calculation in order to guarantee real-time capability. Along with the intrinsic and extrinsic calibration, the disparity image can be used to calculate the depth for each traffic light label. This allows to present results with respect to depth, which is an important system measure. An example scene is shown in Figure 4.3. The wide-angle camera images are not officially published.

Recordings in eleven cities helped to obtain appearance variations of available traffic lights types. Variations in appearance due to different sun visors or back panels (with/without, black/white, round/squared) are often dependent on the city. Weather conditions are slightly changing during recordings, from cloudy to sunny and with varying sun azimuth. Variations in season are not given, because all recordings were performed within two weeks.

4.2.2 Labels

DTLD provides labels for the left stereo camera image. The labels include bounding box coordinates, attribute dictionary, a track identity and further vehicle data referred to one image frame.

Bounding Box: DTLD provides the bounding box annotation in the common $BB = \{x, y, w, h\}$ notation, whereas x and y are top left corner coordinates and



Reprinted with permission from [FMKD18] © 2018 IEEE.

Figure 4.3: Example scene of DTLD from [FMKD18]: Left and right stereo camera image (upper), wide-angle camera and disparity image.

width w and height h, respectively. In comparison to other datasets, DTLD has strict rules concerning the bounding box annotations. For instance, in DTLD all traffic lights are only annotated around the traffic light housing. Back panels and sun visors are not included as these are not normed and could lead to inconsistent labels. Furthermore, occluded labels are estimated in their original expansion (without occlusion) in order to avoid high aspect ratio variations.

Attributes: Each DTLD label is annotated using a set of attributes. Each bounding box is assigned to seven attributes:

- **Aspects**: Describes the number of light units of the traffic light. There exist four classes/tags: {*one aspect, two aspects, three aspects, four aspects*}.
- **Direction**: Describes the direction of a traffic light with respect to the ego-vehicle. There exist four classes/tags: {*front, left, right, back*}.
- **Occlusion**: Describes occluded or truncated (image borders) traffic lights. There exist 4 classes/tags: {*occluded, not occluded*}.
- **Orientation**: Describes whether traffic lights are horizontally or vertically installed. There exist two tags: {*horizontal, vertical*}.

- **Pictogram**: Describes the traffic light inlay or symbol. There exist eight classes/tags: {*circular, arrow straight, arrow left, arrow straight left, arrow right, pedestrian, cyclist, bus/tram*}.

- **Relevancy**: A traffic light is relevant if it is valid for the planned route of the vehicle. Traffic lights of the following intersections are not tagged as relevant There exist two tags: {*relevant, not relevant*}.

- **State**: Describes the traffic light state/phase. There exist five tags: {*off, red, red-yellow, yellow, green*}.

Figure 4.4 illustrates all annotated attributes and the respective attribute classes.

Track identity: Each unique traffic light instance has an assigned track identity during one sequence. One sequence consists of all images recorded at one unique intersection until passing it. For DTLD, one out of 15 images is annotated (approximately one labeled image per second) resulting in a two-digit number of labels for most instances. The track identity is of particular interest for the evaluation, especially when evaluating tracking performance. For instance, assuming a system with a detection rate of 99 percent, the missed objects are more fatal in the case that one instance is never detected. On the other hand, not detecting one real world instance in 1 percent of all frames is not critical, because this can be easily compensated by tracking.

Vehicle Data: DTLD provides further vehicle data such as GPS, velocity or yaw rate, which are not used in this thesis yet mentioned for the purpose of completeness.

4.2.3 Dataset Statistics

DTLD provides a dataset split into a training and test set. The split is done in a manner to ensure, that each traffic light instance only appears in one of both sets. Furthermore, both splits contain images of all eleven cities in order to guarantee a high variation. The training set amounts 2/3, the testing data 1/3 of the entire dataset. Table 4.2 and 4.3 show the statistics for both, training and testing data, as well as the number of objects per state and pictogram class. Six DTLD versions are shown, the original one as well as four subsets filtered by a minimum width and different ignore regions. These subsets will be further introduced in Section 5.3 and used for training and evaluation. In general, the states *yellow*, *red_yellow* and *off* are highly underrepresented, which is one major challenge for detection and state classification. With regard to pictogram recognition, circle is overrepresented, while DTLD shows a lack of the pictograms arrow_right, bicycle and arrow_straight.

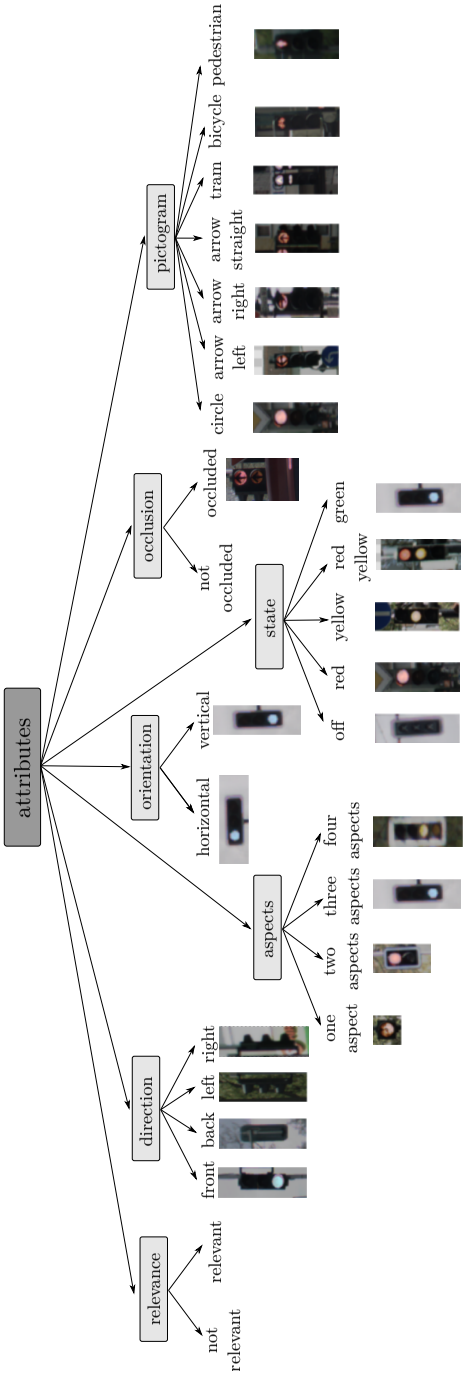


Figure 4.4: Annotated attribute classes of the DriveU Traffic Light Dataset. Image crops show examples for most classes extracted from DTLD.

Table 4.2: DTLD and subset statistics with respect to the traffic light state classes. DS1-DS4 are explained and used in Chapter 5, whereas DS_STATE is introduced in Chapter 6.

	TRAINING							TESTING						
	# images	# objects	# red TLs	# green TLs	# yellow TLs	# redyellow TLs	# off TLs	# images	# objects	# red TLs	# green TLs	# yellow TLs	# redyellow TLs	# off TLs
DTLD (all)	28.5k	159.9k	48.9k	67.5k	5.4k	2.3k	35.8k	12.5k	72.1k	20.8k	29.6k	2.0k	0.9k	18.7k
DTLD (DS1)	23.9k	54.7k	17.6k	31.3k	2.4k	0.9k	2.5k	10.2k	23.5k	7.5k	13.5k	1.0k	0.4k	1.0k
DTLD (DS2)	20.9k	44.4k	14.0k	25.5k	2.0k	0.8k	2.0k	8.7k	18.5k	5.9k	10.7k	0.8k	0.3k	0.8k
DTLD (DS3)	22.8k	41.6k	11.2k	27.8k	1.4k	0.8k	0.6k	9.8k	17.7k	4.8k	11.7k	0.6k	0.4k	0.3k
DTLD (DS4)	19.7k	34.4k	9.1k	23.0k	1.2k	0.6k	0.5k	8.3k	14.4k	3.9k	9.5k	0.5k	0.3k	0.3k
DTLD (DS_STATE)	23.9k	54.7k	17.6k	31.3k	2.4k	0.9k	2.5k	10.2k	23.5k	7.5k	13.5k	1.0k	0.4k	1.0k

Table 4.3: DTLD and subset statistics with respect to the traffic light pictogram classes. DS1-DS4 are explained and used in Chapter 5, whereas DS_PICTO is introduced in Chapter 6.

	TRAINING										TESTING									
	# images	# objects	# circle	# arrow-left	# arrow-straight	# arrow-right	# tram	# pedestrian	# bicycle	# unknown	# images	# objects	# circle	# arrow-left	# arrow-straight	# arrow-right	# tram	# pedestrian	# bicycle	# unknown
DTLD (all)	28.5k	159.9k	81.9k	10.6k	5.6k	1.3k	8.0k	27.5k	3.8k	21.0k	12.5k	72.1k	34.8k	4.7k	2.2k	0.6k	4.1k	12.9k	1.8k	10.9k
DTLD (DS1)	23.9k	54.7k	42.5k	7.1k	3.9k	1.0k	-	-	-	-	10.2k	23.2k	18.1k	3.1k	1.5k	0.4k	-	-	-	-
DTLD (DS2)	20.9k	44.4k	34.5k	5.8k	3.2k	0.9k	-	-	-	-	8.8k	18.5k	14.3k	2.6k	1.2k	0.4k	-	-	-	-
DTLD (DS3)	22.8k	41.6k	36.1k	1.7k	3.3k	0.5k	-	-	-	-	9.8k	17.7k	15.5k	0.6k	1.4k	0.1k	-	-	-	-
DTLD (DS4)	19.7k	34.4k	29.8k	1.5k	2.7k	0.4k	-	-	-	-	8.3k	14.4k	12.5k	0.6k	1.2k	0.1k	-	-	-	-
DTLD (DS_PICTO)	21.4k	54.3k	34.5	5.8k	3.2k	0.9k	3.7k	4.9k	1.3k	-	9.1k	23.0k	14.3k	2.8k	1.3k	0.4k	1.9k	1.9k	0.6k	-

Size Distribution: The size distribution of all and only relevant DTLD labels is compared in Figure 4.5. It shows the ground truth width distribution, which follows a $1/x$ distribution down to the labeling resolution limit of approximately 5 pixels. Moreover, it clearly shows, that *traffic light detection is a small object detection problem.* Assuming a constant camera frame rate and constant velocity while approaching an intersection, this leads to a much higher number of small objects. A detection of small traffic lights is required in order to guarantee a safe and comfortable braking in case of red lights. More details on distance requirements are presented in Section 5.1.

Aspect Ratio Distribution: Figure 4.6 illustrates the aspect ratio distribution of traffic lights with respect to the annotated number of aspects. Soft borders between the distribution for each number of aspects are clearly visible. Therefore, two main reasons exist. On the one hand, very small objects can cause extreme aspect ratios because of quantization and labeling inaccuracies. On the other hand, non-frontal traffic lights (e.g. tag *left*, *right*) cause higher aspect ratios, which lead

to a higher variance towards large aspect ratios. By only considering relevant traffic lights (b), the curves show a significantly smaller variance, because most of small and left or right direction traffic lights are ignored. However, traffic light detection needs to be able to detect a high aspect ratio variance, which is challenging for many object detection methodologies.

Traffic Light Label Locations: Figure 4.7 illustrates a 2D distribution of all DTLD labels with respect to the label centre position in front of an example scene of DTLD. Red regions indicate a high number of labels in one bin, blue regions weakly occupied regions. Because of the normed installation position of traffic lights and flat roads, traffic lights only occur in the upper half of the image. Moreover, traffic lights likely occur in the image centre. This is an indirect effect of the size/depth distribution (Figure 4.5) - objects of rising distance move towards the vanishing point of the image.

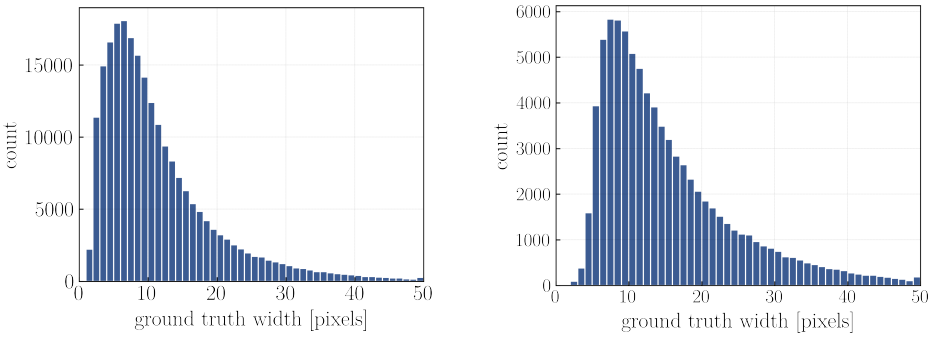
4.3 Evaluation Metrics

This section introduces metrics used for evaluation in the following chapters. Section 4.3.1 presents metrics used for evaluation of the detection system in Chapter 5. In consequence, these metrics are designed for evaluating binary classification problems. The metrics introduced in Section 4.3.2 are used to evaluate multiclass systems, i.e. useful to evaluate the presented state and pictogram recognition results in Chapter 6.

4.3.1 Detection Metrics

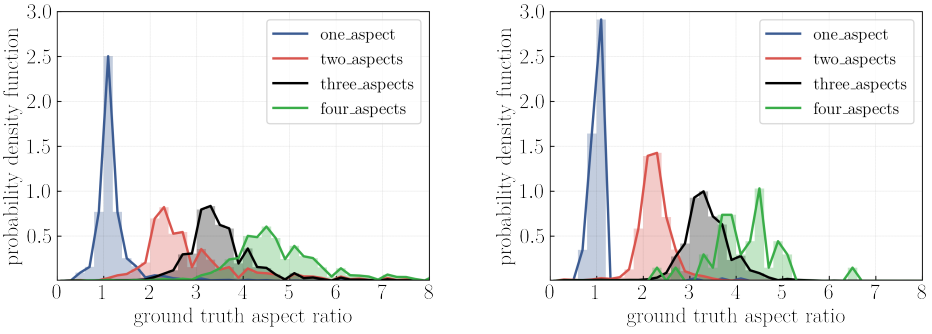
In terms of object detection, i.e. object localization, evaluation is typically performed image-wise. For each image $\mathbf{I}$, a set of ground truths $\underline{g}_\mathrm{I} = (g_0, g_1, ..., g_M)^\top$ (true traffic lights) and predictions $\underline{p}_\mathrm{I} = (p_0, p_1, ..., p_N)^\top$ (predicted traffic lights) exist. The ground truths and predictions of each image are compared. This proceed is done for all images N_I in the evaluation set.

Intersection Over Union: Originally, the *Intersection Over Union*, also known as *jaccard index*, is a similarity measure of two sets. In the context of object detection it is often used as an overlap measure between ground truth bounding boxes g and object candidates p to differentiate between true positives and false positives, see also [DWSP09].



(a) all (b) relevant

Figure 4.5: Size distribution of all DTLD ground truths.



(a) all (b) relevant

Figure 4.6: Aspect ratio distribution of all DTLD ground truths.

In this context, the Intersection Over Union (IoU) is defined as the intersection divided by the union of both bounding box rectangles, which can be formulated as

$$\text{IoU} = \frac{|p \wedge g|}{|p \vee g|} = \frac{|p \wedge g|}{|p| + |g| - |p \wedge g|}. \tag{4.1}$$

The IoU can have values in the range

$$0 \leq \text{IoU} \leq 1. \tag{4.2}$$

Values close to 1 express a high detection accuracy, whereas values close to 0 can be

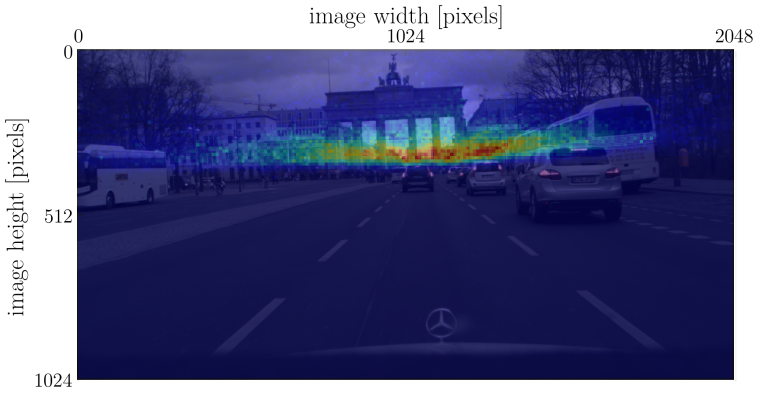


Figure 4.7: Traffic light label location of all DTLD ground truths. Red symbolizes highly, blue weakly occupied regions.

either caused by errors in position, size or both.

True Positives, False positives, Recall: A prediction is assumed as a *true positive* if it overlaps with at least one traffic light ground truth by a defined threshold $\mathcal{T}_{\mathrm{IoU}}$. If a prediction does not overlap with at least one ground truth, it is treated as a *false positive*.

$$p_i = \begin{cases} \mathrm{TP}, & \text{for} \quad \max(\mathrm{IoU}(\underline{g}_{\mathrm{I}}, p_i)) \geq \mathcal{T}_{\mathrm{IoU}}, \quad i \in \{0, ..., N\} \\ \mathrm{FP}, & \text{for} \quad \max(\mathrm{IoU}(\underline{g}_{\mathrm{I}}, p_i)) < \mathcal{T}_{\mathrm{IoU}}, \quad i \in \{0, ..., N\} \end{cases} \tag{4.3}$$

The most common choice is $\mathcal{T}_{\mathrm{IoU}} = 0.5$. This threshold can be increased for large objects and decreased for small objects. However, the requirements on localization accuracy may vary between different applications. Because this measure correlates well with visual impressions, it is a frequently used measure. If multiple predictions overlap with one ground truth, the one with the highest confidence is selected, while the others are counted as false positives. However, other evaluation strategies may not count multiple detections as false positives.

Recall and False Positives Per Image (FPPI): After categorization of each prediction into true and false positive and ground truths into true positive and false negative, single value metrics are used to express the entire detection ability. Those are then calculated over all images in the entire test set. *Recall* describes the percentage of correctly classified true objects, which can be formulated as

$$\text{recall} = \frac{\#\text{TP}}{\#\text{TP} + \#\text{FN}}. \tag{4.4}$$

The recall is also called *true positive rate* [Faw06]. The miss rate (MR) expresses the percentage of not detected, but existing objects

$$\text{MR} = \frac{\#\text{FN}}{\#\text{TP} + \#\text{FN}} = 1 - \text{recall}, \tag{4.5}$$

i.e. a miss rate of 0 percent is a perfect result, whereas high miss rate values express a poor detection ability [Pow11]. Besides the true positive rate, the false positive rate is an important measure allowing to compare the detection ability with the number of wrong predictions. Considering recall values without knowledge about the false positive rate is critical, because an infinite number of predictions automatically leads to 100 percent recall. In this work, a more intuitive false positive metric is used called *False Positives Per Image (FPPI)*. This metric is calculated as

$$FPPI = \frac{\#\text{FP}}{N_{\text{I}}}, \tag{4.6}$$

i.e. the sum of all false positives on the test set is divided by the overall number of images in the test set. Consequently, this metric expresses the average number of false positives per image. This metric is particularly important for autonomous systems as thereof an expected frequency of errors can be made. It is also used in many popular pedestrian detection publications [DWSP09]. False positive rates without knowledge of the number of evaluation images are not meaningful and often give a misleading impression.

ROC and Log-Average Miss Rate (LAMR): Typically, CNN object detectors predict a confidence value c for each bounding box prediction p_i. Different operating points of the system can be obtained by increasing or decreasing the confidence threshold. Picking high confidence thresholds leads to increasing miss rates, but decreasing FPPI values, low confidence thresholds decrease the miss rate, but increase the amount of false positives. Depending on the application, both extremes can be desired. A graphical illustration of all operating points of a detection system with respect to a varying threshold is called Receiver Operating Characterstic (ROC) [Faw06]. Figure 4.8 illustrates the miss rate ROC curve, whereas the FPPI value is plotted in logarithmic scale. The miss rate is visualized for different choices of FPPI and a varying threshold $\mathcal{T}_c$. In general, the curve contains as many operating points as existing, unique confidence values.
In order to compare different detection systems/approaches, a comparison of two curves is not the desired choice. A common metric to express ROC curve result by one numerical value is the Area Under Curve (AUC) [Faw06]. However, this method considers all operating points. From the perspective of application, there exist oper-

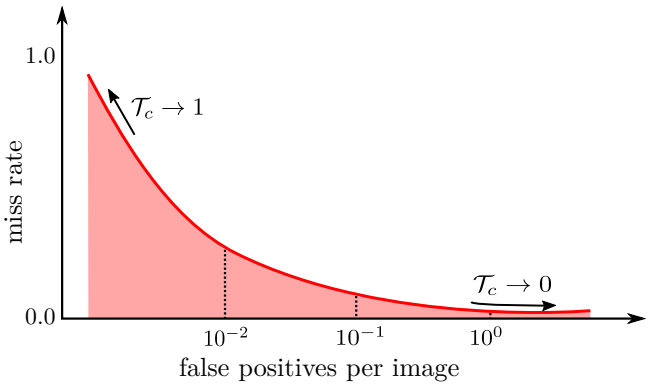


Figure 4.8: Miss rate curve with respect to false positives per image. Important values for log-average miss rate calculation are marked on the x-axis.

ating points of interest and other operating points (e.g. very high FPPI values) of no interest. An established metric is the *Log-Average Miss Rate* (LAMR) [DWSP12] used to compare different ROC curve results. It is defined as

$$\mathrm{LAMR} = \left(\prod_{i=1}^{N_{\mathrm{LAMR}}} \mathrm{MR}_i \right)^{N_{\mathrm{LAMR}}{}^{-1}} = \exp\left(\sum_{i=1}^{N_{\mathrm{LAMR}}} \ln(\mathrm{MR}_i) \right), \tag{4.7}$$

whereas the freedom in design is the number of scan points N_{LAMR} and the location of those. Mathematically, the LAMR can be interpreted as a geometrical mean. Figure 4.8 illustrates the operating points selected for LAMR calculation. The choice for this work is FPPI= $\{10^{-2}, 10^{-1}, 10^{0}\}$, i.e. $N_{\mathrm{LAMR}} = 3$. Investigations showed, that this choice is realistic on the one hand and useful from the perspective of application on the other hand.

4.3.2 Multiclass Metrics

Besides the detection of traffic lights, a classification of state and pictogram is also part of this thesis. Therefore, multiclass metrics have to be introduced in order to evaluate both tasks.

Confusion Matrix $\mathcal{C}$: A confusion matrix visualizes the true class ω (from ground truth) against the predicted class $\hat{\omega}$. Figure 4.9 illustrates a confusion matrix for a three classes problem. The values can either be absolute or relative (percentage).

The diagonal values expresses correct predictions, whereas the non-diagonal values express prediction errors, so called "confusions". For normalized confusion matrices the sum of values along the true class (row-wise sum) is equal one.

Micro-recall [SL09]: In order to express a confusion matrix result by scalar values, there exist common metrics. The terms *true positive, false positive, true negative, false negative* lose their validity for a multiclass problem instead of a binary classification. The main values of interest are the correct classifications, i.e. the main diagonal values. Wrong classifications, especially the information with which classes one class is frequently "confused", is easier to evaluate in a qualitative manner. In this thesis, two one-value metrics summarizing a confusion matrix result are used. The *micro-recall* describes the percentage number of correct predictions, independent of their class. This is reached by dividing the sum of diagonal values by the number of all confusion matrix values, formulated as

$$\mathcal{R}^{\text{micro}} = \frac{\sum_{i=1}^{N} \mathcal{C}(\omega_i, \hat{\omega}_i)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \mathcal{C}(\omega_i, \hat{\omega}_j)}. \tag{4.8}$$

Macro-recall [SL09]: The macro recall computes a recall value for each single class and averages class-wise recall values. In contrast to the micro recall, it helps to overweight underrepresented classes and is independent of the absolute number of samples per class. It is calculated as

$$\mathcal{R}^{\text{macro}} = \frac{1}{N} \sum_{i=1}^{N} \frac{\mathcal{C}(\omega_i, \hat{\omega}_i)}{\sum_{k=1}^{N} \mathcal{C}(\omega_i, \hat{\omega}_k)}. \tag{4.9}$$

true class ω_i \ predicted class $\hat{\omega}_i$	$\hat{\omega}_1$	$\hat{\omega}_2$	$\hat{\omega}_3$
ω_1	0.81 81	0.09 9	0.10 10
ω_2	0.08 10	0.91 120	0.01 2
ω_3	0.08 2	0.15 4	0.77 20

Figure 4.9: Common confusion matrix for a 3-class problem. Typically, the matrix contains absolute (red) or normalized values (black). Within this thesis, both values are plotted.

Chapter 5

Traffic Light Detection using Single Shot Detection (SSD)

Traffic light detection, i.e. the localization of traffic lights in the image, is typically the first and most crucial part. As traffic light detection is commonly the first module of traffic light recognition systems, not detected objects may lead to accidents, especially if instances are not detected during the entire intersection approach. A common representation of traffic light detections are bounding boxes surrounding the traffic light housing. Some publications (especially early approaches) also detect bounding boxes surrounding only the active lights.

As presented in Chapter 2, early methods for traffic light detection are all based on recognizing properties of the active light: color, brightness (spotlight detection) and shape (hough circle detection). Because these features also appear in many non-traffic light regions, subsequent verification and classification is necessary to reduce the high amount of false positives. From the perspective of application, these detection methods show several critical disadvantages. The requirement of these approaches on the visibility of the active light rule out a detection of traffic light with state off and rotated traffic lights (back, left, right) causing poor visibility of the active lights. Moreover, a detection of traffic lights with non-circular pictogram is complicated. This can be explained when considering several conventional methods, which use the initial blob detection size to estimate the traffic light housing bounding box. In case of arrows, the housing bounding box is often under- or overestimated. [FMD17b] tackle this problem by using stereo vision, which leads to a corrected bulb hypothesis size. A further problem are traffic lights with one, two or four aspects. Typically, traffic light hypotheses are estimated from the color and size of the initial bulb hypothesis. The created housing hypothesis is then generated assuming an aspect ratio of three. This procedure is not possible for a different number of aspects. To the best knowledge, no publication addressed this problem by using conventional (non CNN-based) methods.

Besides the presented limitations of conventional methods in terms of detection performance, there exist limitations concerning reachable false positive rates. Other works showed, that achieving high detection rates for color or spotlight detection can only be reached for operating points with a high double-digit or moderate three-digit number of false hypotheses [FMD17a]. The final false positive rates can be calculated by multiplying this number with the false positive rate of subsequent modules (verification/classification). Experience has shown, that common classification modules such as HOG-SVM reach detection rates of 95 percent at false positive rates of about 10^{-2}. In consequence, conventional traffic light recognition systems struggle in reaching small false positive rates together with high true positive rates.

During recent years, the leading methods of the most popular object detection benchmarks [EGW+10; LMB+14b] changed from methods based on hand-crafted features to the use convolutional neural networks. This development transferred to publications in the field of traffic light detection leading to several publications based on CNNs. CNN-based object detection methods can be further divided into approaches using external hypotheses generators and internal hypotheses generation. Because the former shows similar disadvantages to conventional methods, the presented method uses an approach outputting bounding boxes as well as object confidences in one step.

Section 5.1 gives further details about the requirements to traffic light detection from the perspective of a safe and comfortable system and sets goals on detection rate, false positive rate and detection extent. Section 5.2 then explains the basic method in detail and presents all modifications done to adapt on the problem of traffic light detection. Section 5.3 presents all experiments done to determine the optimal network configuration and presents an extensive evaluation and error analysis. A conclusion and discussion is given in Section 5.4.

5.1 Requirements and Goals

Because traffic light detection is the first module of a traffic light recognition system it has to fulfil several requirements, which are important for an entire recognition system. This includes the requirements on the distance, in which traffic lights have to be detected as well as the types of traffic lights (attribute classes), which are aimed to be detected. Further requirements are the desired performance measures, such as detection rate or false positive rate.

Detection Distance: The distance, in which traffic lights have to be detected, can be inferred from the braking speed s_B for comfortable braking and the velocity of

the vehicle. A simplified relationship between the braking distance of a vehicle and its braking acceleration a_{B} and velocity v_0 can be formulated as

$$s_{\mathrm{B}} = \frac{v_0^2}{2 \cdot a_{\mathrm{B}}} + (s_{\mathrm{buffer}}) \tag{5.1}$$

when assuming a constant deceleration [Gil92]. s_{buffer} denotes an additional buffer, e.g. the distance travelled resulting from the time between initial detection and track generation, i.e. until a stable track exists. Noticeable is the squared influence of the velocity. Doubling the velocity quadruples the braking distance, i.e. the distance in which traffic lights initially have to be detected. The formula allows a rough estimation, in which distance traffic lights have to be detected. Typically, traffic lights are installed in built-up areas, in which the speed limit amounts $50\frac{km}{h}$ or less. In non-built up areas, the speed limit has to be reduced at intersections resulting in a maximum velocity of $v_0 = 70\frac{km}{h}$. When assuming a comfortable braking of $a_B = 3\frac{m}{s^2}$, a traffic light system has to react at the latest in 63 meters distance. Table 5.1 shows the braking distance for different velocities. These values allow a good estimation in which distance traffic lights have to be detected to ensure a comfortable braking. From the perspective of safety, a higher braking acceleration can be assumed, which decreases the distance requirement. However, for this work the comfort scenario at $v_0 = 70\frac{km}{h}$ is assumed. The ability of detecting objects in high distances is mainly dependent on the image resolution and the focal length of the image sensor and lens. Mathematically, the traffic light width in pixels, denoted as w_{TL}^I, can be calculated as

$$w_{TL}^I = \frac{f_x \cdot w_{TL}^\omega}{Z^\omega}, \tag{5.2}$$

where f_x is the focal length in pixels (the utilized camera lens has a focal length of 2290.51 pixels), w_{TL}^ω is the traffic light width in real world and Z^ω is its distance in real world with respect to the image sensor.

Figure 5.1 illustrates the traffic light resolution with respect to its distance calculated for the DTLD stereo camera. As shown in the introduction (see Figure 1.2), two traffic light sizes ($w_{TL}^\omega = 0.27m$ at $v_0 \leq 50\frac{km}{h}$, $w_{TL}^\omega = 0.37m$ at $50\frac{km}{h} \leq v_0 \leq 70\frac{km}{h}$) are installed on German roads. Projecting these two models according to Equation (5.2) gives the traffic light size in pixels for varying distance. With help of this figure, requirements on the minimum detection resolution can be derived. For this purpose, only the blue curve as the worst-case scenario is considered. Two operating points are marked, which can be referred to as particularly important. The first is the already mentioned braking distance point at $v_0 = 70\frac{km}{h}$, which can be seen as the *must detection point*. In other words, a detection of all relevant traffic lights has to

be guaranteed at 10 pixels object width. Another operating point is marked in 100 meters distance. Typically, a traffic light recognition system only trusts detections which appeared several times in consecutive frames. Considering a buffer, a detection at resolutions of 8 pixels would be desirable. It is the *desirable detection point*.

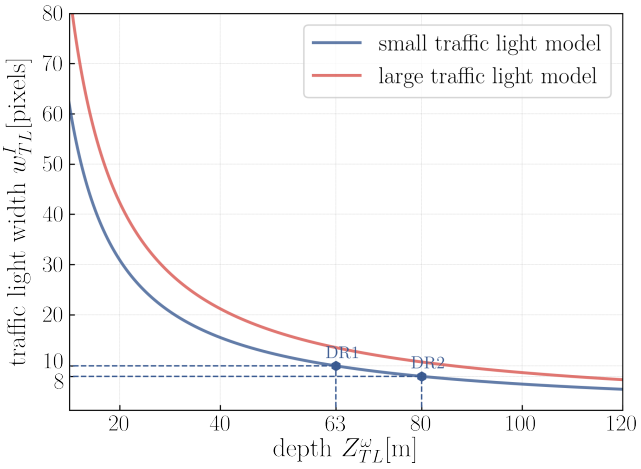


Figure 5.1: Traffic light size in pixels to distance relationship based on two commonly installed traffic light sizes installed on German roads. From the braking point derivations, 10 pixels is the required and 8 pixels is the desired resolution, as of which traffic lights have to be detected.

Table 5.1: Braking distance for different vehicle velocities and an assumed braking acceleration of $3\frac{m}{s^2}$. DR2 additionally considers a buffer braking distance.

Req.	$\boldsymbol{v_0}[\frac{\boldsymbol{km}}{\boldsymbol{h}}]$	$\boldsymbol{s_B}[\boldsymbol{m}]$	$\boldsymbol{w^\omega_{TL}}$**[pixels]**
-	30	12	80
-	50	32	20
DR1	70	63	10
DR2	70	63 + 17	8

Detection Classes: Besides the distance requirements, the types of traffic lights which are aimed to be detected have to be discussed as well. Fregin et al. [FD16]

discuss the term *relevancy* in their work. This influenced the labeling hierarchy of DTLD containing relevancy as one label attribute. Only detecting relevant traffic lights is a comprehensible proceed for autonomous driving, because the route of the vehicle can be assumed to be known. From the perspective of a driver assistance system, the planned route is not always known. The driver can suddenly switch lanes, which requires to also recognize not relevant traffic lights. In consequence, the term relevancy has to be extended for the driver assistance scenario. Besides the traffic lights responsible for the current route (which can be derived from the lane the vehicle drives), traffic lights responsible for all other driving maneuvers have to be considered as well.

With regard to DTLD, *specific evaluations and investigations can be performed using the extensive label attributes, for instance by excluding several classes from the classes required to be detected, so called ignore regions.* Table 5.2 illustrates the ignore regions for both scenarios. As mentioned, for autonomous driving a detection of relevant traffic lights may be sufficient, for the driver assistance scenario this extends to all frontal, vehicle traffic lights. Whether a detection of non-relevant traffic lights is required or not is still not entirely answered. Aiming to detect all traffic lights in a scene may help to an improved scene understanding, however it may also negatively impact the detection ability on relevant traffic lights. *This thesis presents various more evaluations on different subsets helping to improve assessability of the abilities of the developed system.*

Detection Accuracy: In this context, the term detection accuracy describes how accurate the localization of a labeled ground truth is. In machine learning, the most common metric is the IoU measuring a percentage overlap between ground truth and the predicted bounding box. Most approaches in literature use a threshold of $\mathcal{T}_{\mathrm{IoU}} = 0.5$ to express whether an object is detected or not. However, it has to be considered, that most of these approaches detect objects of large resolution. These objects often cover large parts of the entire image, while traffic lights cover clearly below one percent of the image. With decreasing object size, the average IoU theoretically reachable decreases. For this reason, *the presented approaches are evaluated with a lower threshold of $\mathcal{T}_{IoU} = 0.3$, which is the target threshold value.*

Detection Rate and False Positive Rate: Detection rate and false positive rate are closely interconnected. One task of object detection developers is to select an appropriate system operating point with respect to the entire system. In theory, operating points with high detection rates show adverse false positive rates and vice versa. Another difficulty is, that the impact of non-detections (false negatives) or false positives is hardly predictable. Because entire traffic light systems consist of various additional modules, such as High Definition (HD)-Map verification, tracking and others, some errors are more challenging than others. Sporadic, not detected traffic lights objects are not as critical as entirely undetected traffic lights. The

Table 5.2: Ignored classes for the scenario of driver assistance systems and autonomous vehicles. DTLD defines relevant traffic lights for the planned route of the vehicle. Because the route is not known in a driver assistance scenario, it is extended to all possible driving manoeuvres the driver can choose, i.e. traffic lights responsible for straight traffic or turning left or right.

Scenario	ignore_regions
Driver Assistance	back, left, right, tram/bus, pedestrian, cyclist
Autonomous Driving	not_relevant

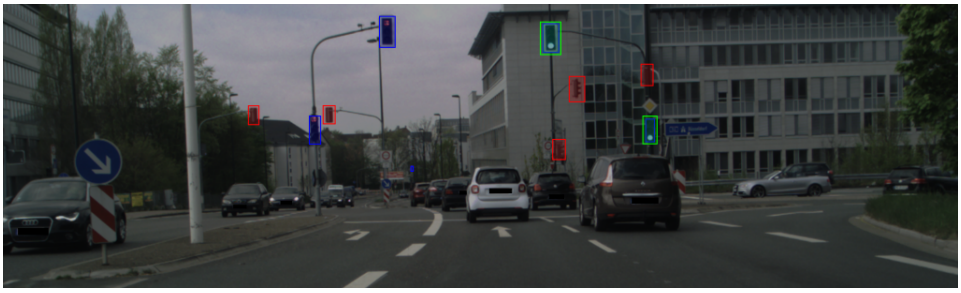


Figure 5.2: Categorization of traffic lights into relevant traffic lights (green), frontal vehicle traffic lights (blue) and non-frontal traffic lights (red). Except a potentially better scene understanding, detecting non-frontal traffic lights may not show any benefits.

former can usually be compensated by tracking modules. Consequently, generic statements regarding the meaning of miss rates are hard to make, because the impact of not detected objects is not clear without considering the entire system chain.

The same holds for false positives, which are mainly problematic when leading to a track creation. This is typically the case if the false positive appears in many consecutive frames and does show common traffic light properties (e.g. static real-world position). However, a clear statement on the impact of false positives can only be made when evaluating within the entire system. Instead, this thesis aims to present the performance bandwidth of the detection system, which offers various possible applications with different system characteristics.

5.2 Single Shot Detection

Initially, CNNs were mainly used in the field of image classification. Early methods, such as the *Region CNN (R-CNN)* of Girshick et al. [GDDM14] tried to also use powerful CNNs for object detection, i.e. a localization of objects in the image. Because these networks are not able to predict object coordinates, external candidate generators are used creating hundreds or thousands of object candidates per image. Each candidate is processed through a trained CNN and classified into one class. Because the feature vector for each region is independently calculated, this proceed is time-consuming, especially for a high number of candidates.

Fast R-CNN [Gir15] enhances this method by calculating the features on the entire input image, so that the feature vector is received by *cutting out the feature values for each region of interest projected into this feature map*. This helps to strongly reduce the computational effort and decrease runtime. The benefits of these two CNN methods compared to conventional methods are simply limited to richer features, which boost the classification performance. One weak point is the candidate

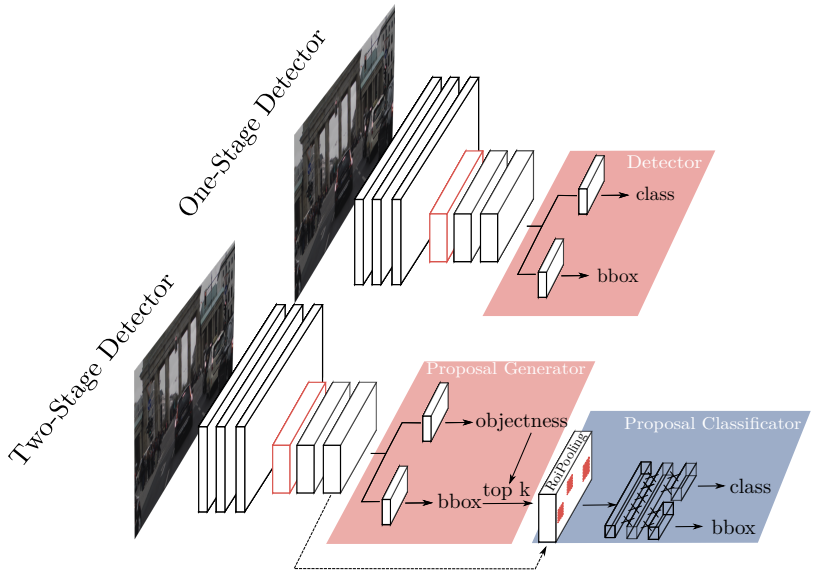


Figure 5.3: Comparison of one- and two-stage detectors. Both use an arbitrary basis network (e.g. VGG-16). While one-stage detection directly predicts the bounding box coordinates and object class, two-stage detectors use an additional classification network.

creation step, which is time-consuming and still based on hand-crafted feature design. Example methods for generic candidate generators are Selective Search [USGS13], Edgeboxes [LDG+17] or simply highly heuristic methods based on geometrical or color properties. In the case of traffic light detection, a color or blob detector as described in Chapter 2 could be used.

In order to avoid the disadvantages of candidate generation, Faster R-CNN [RHGS17] tackles this problem by also generating object candidates directly inside the CNN. An almost similar proceed is followed by the *Single Shot Detector (SSD)* [LAE+15]. Both methodologies share a similar principle, in which the output boxes are predicted relative to predefined boxes and a confidence vector is predicted for each of the boxes. Therefore, two convolutional layers are used, respectively. While the mentioned boxes are also the output boxes of SSD, Faster R-CNN refines the top k boxes (boxes with highest confidence values) in an additional, small network. The previous step is called region proposal network. Both methodologies are illustrated in Figure 5.3. Both approaches form the most state of the art CNN-based object detectors - namely one-stage and two-stage detection.

Both methodologies are based on an arbitrary feature extractor, a basis network, such as VGG-16 [SZ15] or the popular Inception architectures [SIVA17; SLJ+15; SVI+16]. One- and two-stage detectors built up the bounding box and class confidence on a selected feature map of the network. This feature layer is also used in the proposal classification network of two-stage detectors to perform a so called RoiPooling [RHGS17]. RoiPooling maps the output boxes of the proposal generator into the feature map, extracts features and classifies those features into bounding box coordinates and class confidences. Because of this additional step, two-stage detectors are commonly assumed to show a better performance than one-stage detectors. However, runtime significantly increases, which makes one-stage detectors predestined for runtime-critical applications. *Within this thesis, the decision is made in favour of the SSD approach, mainly because of runtime reasons.* The following section briefly deepens the principle of Single Shot Detection and presents all adaptations and investigations to adapt and optimize on the traffic light detection use case.

5.2.1 Basic Principle

Single Shot Detection is an object detection method based on a common convolutional network producing a fixed number of output bounding box coordinates and scores. The most important property of SSD and region proposal networks is, that the bounding boxes are not predicted in absolute coordinates, but relative to predefined reference boxes, in SSD called *prior boxes*. Faster R-CNN uses the

term *anchor boxes*. Both terms refer to the same principle.

The principle of offset prediction is visualized in Figure 5.4. As shown in Figure 5.3, two additional convolutional layers are set upon one single feature layer responsible for bounding box (bbox) and class confidence(s) prediction. The prediction layer is typically smaller than the input image by a certain factor. The deeper the network, the higher the dividing factor. Prior boxes are distributed over the input image according to a fixed grid, whereas the grid size is defined by the mapping factor between image and feature layer size. In other words, one pixel of the feature layer can be considered to be responsible for detecting objects in one corresponding cell in the input image. The feature cell size can be determined by ($\chi^w = \frac{\mathcal{I}^w}{\mathcal{F}^w}$, $\chi^h = \frac{\mathcal{I}^h}{\mathcal{F}^h}$), whereas $(\mathcal{I}^w, \mathcal{I}^h)$ are the input image dimensions and $(\mathcal{F}^w, \mathcal{F}^h)$ denotes the feature layer dimensions. For each cell, a fixed number of prior boxes (blue rectangles in Figure 5.4) is defined. The position of the prior boxes with respect to its corresponding feature cell is fixed. To the best knowledge, all popular one- and two-stage detectors only use one single prior box location per feature cell. This means, all prior boxes in one cell are located in the centre of the cell. Consequently, the prior box stride (step size) in x and y-dimension exactly corresponds to the feature cell dimensions.

During runtime, the task of the bounding box prediction layer is to predict offset vectors regressing the fixed prior boxes with respect to position and size. For each single prior box $\underline{p}_i$, four offset values $\underline{l}_i = (l^x, l^y, l^w, l^h)^\top$ are predicted. Defining the number of prior boxes per cell as K^ρ, it requires $4K^\rho$ channels in the bounding box prediction layer. In consequence, the number of channels of the bbox prediction layer is determined by K^ρ and each prior box has a fixed assignment to one quadruple of the prediction layer. One final output bounding box $\underline{b}_i = (b^x, b^y, b^w, b^h)^\top$ is obtained by

$$b_i^x = \mathcal{I}^w \cdot (p_i^x + l_i^x \cdot p_i^w) \qquad b_i^y = \mathcal{I}^h \cdot (p_i^y + l_i^y \cdot p_i^h), \tag{5.3}$$

whereas the predicted width and height are obtained by

$$b_i^w = \exp(l_i^w) \cdot p_i^w \mathcal{I}^w \qquad b_i^h = \exp(l_i^h) \cdot p_i^h \mathcal{I}^h. \tag{5.4}$$

p_i^x, p_i^x, p_i^w and p_i^h are the i-th prior box upper left coordinates as well as width and height, respectively [GDDM14; LAE+15]. All coordinates are multiplied by the image dimensions $\mathcal{I}^w$ and $\mathcal{I}^h$, because the network itself processes normalized coordinates in the range $[0, 1]$. In simple words, the task of the network is to predict an accurate offset vector $\underline{l}_i$ in order to regress the static prior box vector $\underline{p}_i$ to the actual object coordinates.

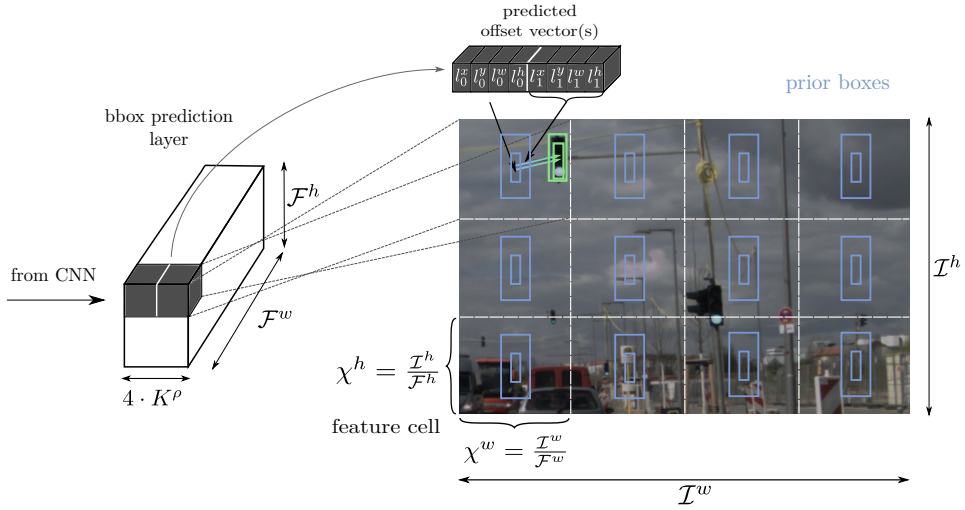


Figure 5.4: Principle of offset prediction with respect to predefined prior boxes (blue). The bounding box prediction layer spatial size $(\mathcal{F}^w, \mathcal{F}^h)$ equals the size of the selected feature map of the basis network. The association between the input image and the prediction layer established by feature cells (white). For each feature cell, a set of equal-sized prior boxes is defined and the network is trained to predict offsets for each single prior box. The third dimension of the bbox prediction layer is fixed and determined by the number of prior boxes per feature cell and the number of regression parameters (4 in this case, two for position, two for size).

An additional contribution of SSD is the use of multiple feature layers for prediction. Instead of using one single prediction layer (commonly the last layer) as utilized in [RDGF16b; SEZ+14], SSD utilizes multiple heads on top of various feature maps at different network stages. Small objects are more effectively predicted in early layers than large objects due to smaller receptive fields and prior box strides.

This section has shown, that prior boxes are one central element of modern object detectors. One design freedom is the number K^ρ and the design of each single box with respect to width, height or aspect ratio. Typically, the prior box sizes are adapted to the size distribution of the training set samples of the utilized dataset. In this work, Section 5.2.3 explains the choice of prior box locations and sizes in detail. In this thesis the prior box sizes are optimized according to a weighted k-IoU

clustering approach, a novel adaption of the well-established k-means algorithm. The target metric is replaced as well as a weighting is used in order to adapt on the present problem of traffic light detection. Furthermore, a novel free-style prior locations method is introduced in this thesis which overcomes the strict cell-based prior box locations. Instead of placing prior boxes in the feature cell centre, it presents size-based adaptive prior box locations. This allows larger strides for larger prior boxes and smaller strides for small prior boxes, but still guarantees fixed, symmetric locations in each cell. Section 5.2.2 gives insights into the utilized basis network, where the VGG-16 network used in the original SSD is replaced. Section 5.2.4 illustrates important theories involved into the bounding box regression in order to learn the network to correctly predict bounding box offsets and confidence values by using appropriate loss functions. Section 5.2.5 briefly shows, how multiple predictions on a single object are handled and merged to one single output box by the Non-Maximum Suppression (NMS).

5.2.2 Basis Network

As well as other popular publications, the original SSD approach uses the well-established VGG-16 network. This may be for the sake of comparison, because many early publications use the VGG network. However, some publications concluded, that VGG-16 does not show the best accuracy-runtime balance [CPC16], especially when comparing to more modern architectures. A comparison of popular basis networks is shown in Figure 5.5. The authors showed, that in particular the *Inception* and *ResNet* architectures [HZRS16] show a *better accuracy or higher speed, or at least a better compromise between these two properties. According to this work, the Inception-v3 base network is utilized for this thesis.* On the one hand, it lies within the set of real-time capable networks, which is crucial for automotive application. On the other hand, it still reaches a high accuracy in other object detection problems. Moreover, the architecture is particularly suitable for the traffic light use case, see in the following. All experiments are performed using the Inception-v3 network.

Inception-v3: The most important layers of the Inception-v3 network are shown in Figure 5.6. The first layers are alternating convolutional and pooling layers followed by three inception_a, four inception_b and two inception_c modules (red blocks). For simplification, only one module is drawn, respectively. As already mentioned, a so called SSD head (layers for bounding box and confidence prediction) can be set upon several feature maps. Finding optimal prediction layers is empirically performed. In general, most works use deeper network stages for predicting larger objects and earlier layers for small object prediction.

Figure 5.7 shows a more detailed view on one inception module. Inception modules

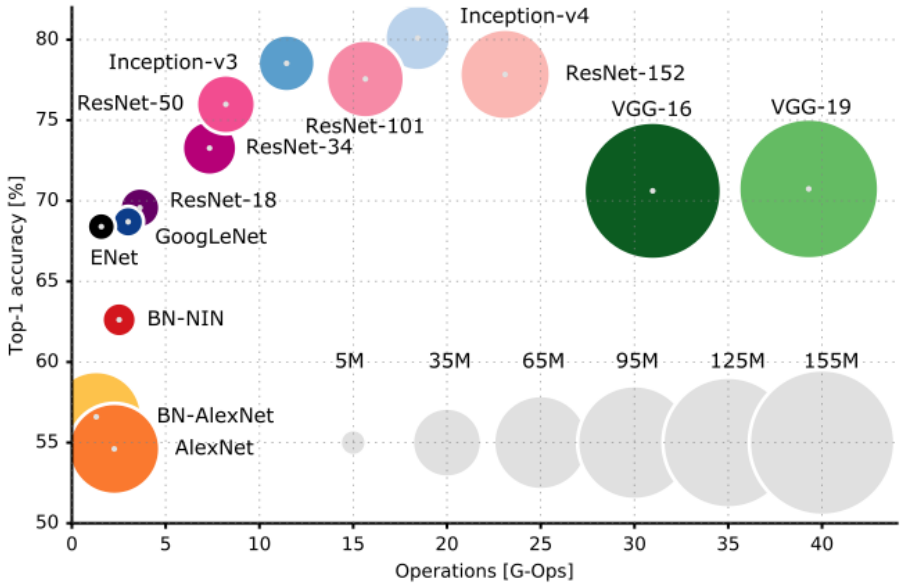


Figure 5.5: Comparison of popular CNN basis networks with respect to accuracy and computational effort from [CPC16]. Inception-v3 is one of the networks with the best accuracy-runtime effort.

consist of four different paths, which mainly differ in the number of convolutions and the size of filter kernels. They have two main advantages, speed and performance. *Dividing large filter kernels into smaller ones decreases runtime. Concatenating parallel paths leads to a concatenation of different receptive fields.*

Receptive field: The receptive field can be defined as the region in the input image, into which a certain neuron of the network is referring to. Deeper layers naturally have a higher receptive field as it is increased by pooling and convolutional operations. Because there is a close relationship between the size of a receptive field and the ability to detect objects, the feature maps used for detection have to be carefully chosen. In general, small receptive fields are able to describe local features, whereas larger receptive fields are able to describe global features and relationships. *With regards to traffic light detection a combination both, local and global features, may be optimal for classification.* Local features describing a traffic light are: colors, traffic light housing or edges or bright spots. More global features may be poles, intersections, road signs and others. *Due to the need for both feature types, the Inception architecture is assumed to be beneficial for the task of traffic light detection as it ensures a conservation of local information even in deep network layers.*

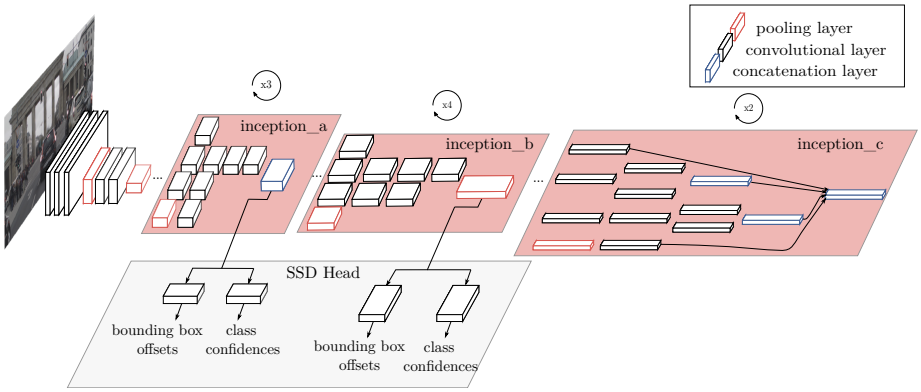


Figure 5.6: SSD with Inception-v3 basis network. Convolutional layers are shown in black, pooling layers in red and concatenation layers in blue. The network consists of three inception_a, four inception_b and two inception_c layers, from which only one is visualized, respectively. Convolutional layers on top of one or multiple feature layers are used for prediction.

This finding is also described in [ZKL+15], which show that features from each network stage can be used to classify objects at different levels of abstraction. They show, that depending on the layer stage edges, textures, objects and scenes are recognized - and all stages are valuable for object detection and classification. This differs from many other works, which only use the final network layer for classification. Consequently, one important part of designing object detectors is to investigate receptive fields of the utilized network. The receptive field in one arbitrary feature layer L can be calculated as

$$r_L = \sum_{l=1}^{L} \left((k_l - 1) \prod_{i=1}^{l-1} s_i \right) + 1, \tag{5.5}$$

where k_l is the feature kernels in layer l and s_i the kernel stride in the respective layer [ANS19]. This equation can be used for single-path networks with no parallel computation branches. Because inception modules utilized in Inception-v3 consist of parallel branches, small receptive fields are still present in deep layers. Table 5.3 shows the important parameters of each of Inception-v3's layers including layer size, prior box stride as well as the theoretical receptive field size calculated using the mentioned formula. Because all traffic lights in DTLD are located in the upper part of the image, only the upper image half is used as a network input. Consequently, the input size amounts 2048×512 pixels.

Table 5.3: Insights into inception-v3 network properties. The table shows the width and height of the most important network layers, as well as the prior box stride (assuming prior boxes in the feature cell centre) as well as the minimum and maximum receptive field.

layer	height [pixels]	width [pixels]	ratio	stride [pixels]	receptive field (min/max) [pixels]
input	512	2048	1	1	1x1
conv_1	255	1023	$\sim 1/2$	2	3x3
conv_2	253	1021	$\sim 1/2$	2	7x7
conv_3	253	1021	$\sim 1/2$	2	11x11
conv_4	124	508	$\sim 1/4$	4	23x23
inception_a1	62	254	$\sim 1/8$	8	31x31 / 63x63
inception_a2	62	254	$\sim 1/8$	8	31x31 / 95x95
inception_a3	62	254	$\sim 1/8$	8	31x31 / 127x127
inception_b1	31	127	$\sim 1/16$	16	47x47 / 351x351
inception_b2	31	127	$\sim 1/16$	16	47x47 / 543x543
inception_b3	31	127	$\sim 1/16$	16	47x47 / 735x735
inception_b4	31	127	$\sim 1/16$	16	47x47 / 927x927
inception_c1	15	63	$\sim 1/32$	32	79x79 / 1183x1183
inception_c2	15	63	$\sim 1/32$	32	79x79 / 1311x1311

Reprinted with permission from [MD18] © 2018 IEEE.

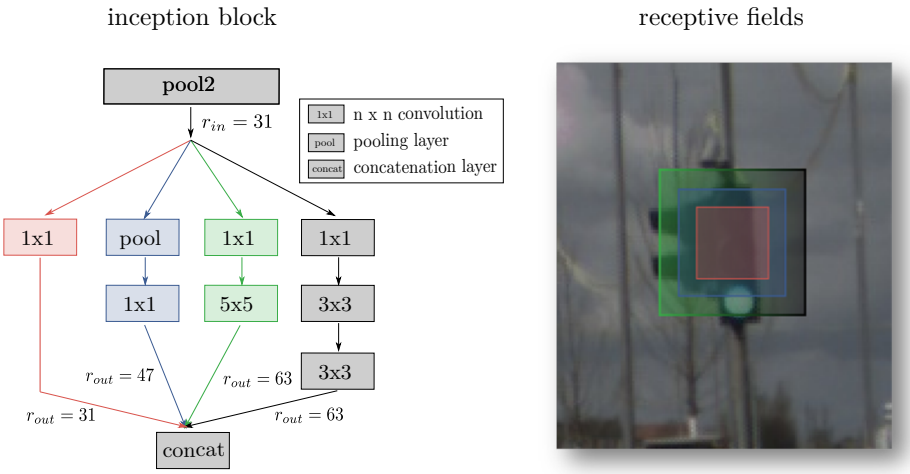


Figure 5.7: One inception module (here: inception_a1) consists of four parallel branches and concatenates multiple receptive fields r_{out}. This helps to combine local and context information, which may be beneficial for traffic light detection. The receptive field sizes for each branch are visualized in one example scene.

5.2.3 Design of Prior Boxes - Optimizing Prior Box Dimensions and Locations

During training in the Single Shot Detection framework, ground truths are matched with the predefined prior boxes according to the metric IoU [ESTA14]. Only ground truths are considered, which overlap with at least one prior box by more than $\mathcal{T}_{\text{IoU}} = 0.3$. Due to mathematical reasons, a usage of labels showing no overlap ($\mathcal{T}_{\text{IoU}} = 0.0$) is not possible. In consequence, one first and very crucial optimization developers have to consider, is to adapt the prior boxes to the type of objects which are aimed to be detected. Within this work, the first optimization is to adapt the prior box dimensions, i.e. width and height, with regard to the data distribution obtained from DTLD. In a second step, the already introduced prior box stride/locations is re-discussed and an alternative proceed is presented in order to improve the ability to detect small objects.

Adapting prior box dimensions

When adapting the prior box dimensions (width/height), two questions have to be answered. *How many prior boxes are needed? - and: How to choose the dimensions of the prior boxes?* The choice of the number of sizes is according to the slogan *as much as necessary, as little as possible*. The main motivation to keep the number of prior boxes low is runtime and additionally, that the optimization during training is simplified. An illustration of the impact of the amount of prior boxes on runtime is illustrated in Figure B.1. When designing prior boxes, the goal is to optimize the overlap between ground truths and prior boxes, while minimizing the absolute number of prior boxes. In easy words, the width and height of the prior boxes have to be chosen in order to match the training samples width and height and optimize the overlap values. An indirect method solving this is addressed by [RF18] using a k-means clustering algorithm in 2D space. Assuming all ground truths $[\underline{g}_0, \underline{g}_1, ..., \underline{g}_N]^\top$ of the training set, where $\underline{g}_n = [g^w, g^h]^\top$, the objective is to partition all samples into k sets $\mathbf{S}$

$$\arg\min_{\mathbf{S}} \sum_{c=1}^{k} \sum_{\underline{g}_i \in S_c} \|\underline{g}_i - \underline{\mu}_c\|^2 \tag{5.6}$$

and minimize the euclidean distance between cluster centre and all samples in the cluster. $\underline{\mu}_c$ is the centre of one cluster and k the number of overall clusters. The Lloyd-algorithm [Llo82] can be used to solve this problem, whereas the cluster centres are updated in each iteration until a defined rate of change is not exceeded anymore.

Although the choice of the number of clusters k is frequently considered as one main drawback of k-means clustering, it is beneficial for the present problem, as this number is desired to be controllable. Because the number of prior boxes directly influences the runtime, clear limitations can be set.

One main problem of conventional k-means algorithm is, that the utilized distance metric is the euclidean distance. For object detection, the metric of interest is the intersection over union, which is desired be maximized while minimizing the number of required prior boxes. Redmon et al. [RF18] proposed to modify k-means clustering distance function. A proper distance function for object detection, which maximizes the IoU, can be formulated as

$$\arg\min_{\mathbf{S}} \sum_{c=1}^{k} \sum_{\underline{g}_i \in S_c} 1 - \mathrm{IoU}(\underline{g}_i, \underline{\mu}_c), \tag{5.7}$$

where the IoU is calculated assuming two bounding boxes with equal centre according to Equation (4.1). This distance formula is tailored to object detection problem. Instead of the euclidean mean, the IoU is the target metric.

Figure 5.8 illustrates the matching results after applying k-IoU clustering on all training samples of DTLD larger 8 pixels width. Black dots symbolize the resulting clusters, blue dots symbolize unmatched ground truths, whereas red dots symbolize matched ground truths ($\mathrm{IoU}(\underline{g}_i, \underline{\mu}_c) \geq 0.3$). Even for low numbers of k, a high number of ground truths are sufficiently matched to the clusters. The most striking issue is, that independent of the number of clusters, ground truths with small aspect ratio are not matched. Samples with small aspect ratios represent one/two aspect traffic lights, which are strongly underrepresented in the training set.

In order to overcome the overfitting characteristic towards three aspects traffic lights, a novel class-weighted k-IoU clustering approach is proposed in this thesis. The weighted k-IoU clustering can be formulated as

$$\arg\min_{\mathbf{S}} \sum_{c=1}^{k} \sum_{\underline{g}_i \in S_c} \left(\eta_j^{\text{aspects}}\right)_i \cdot (1 - \mathrm{IoU}(\underline{g}_i, \underline{\mu}_c)), \tag{5.8}$$

where $\left(\eta_j^{\text{aspects}}\right)_i$ is the sample weight assigned to ground truth $\underline{g}_i$. The sample weight is determined by the frequency of the class $j \in [1, J]$ in the training set the

sample $\underline{g}_i$ is belonging to. The weights are calculated according

$$\eta_j^{\text{aspects}} = \frac{\max(\underline{N}^{\text{aspects}})}{N_j}, \tag{5.9}$$

i.e. by dividing the number of ground truths of the most frequent class by the number of elements of class j. The vector $\underline{N}^{\text{aspects}} = [N_1, N_2, ..., N_J]^\top$ contains the number of ground truths for each aspects class. $J = 4$ classes are used for one, two, three and four aspects traffic lights, respectively. Each ground truth is assigned to one cluster by its distance value. During cluster update, class weights are also considered, which can be formulated as

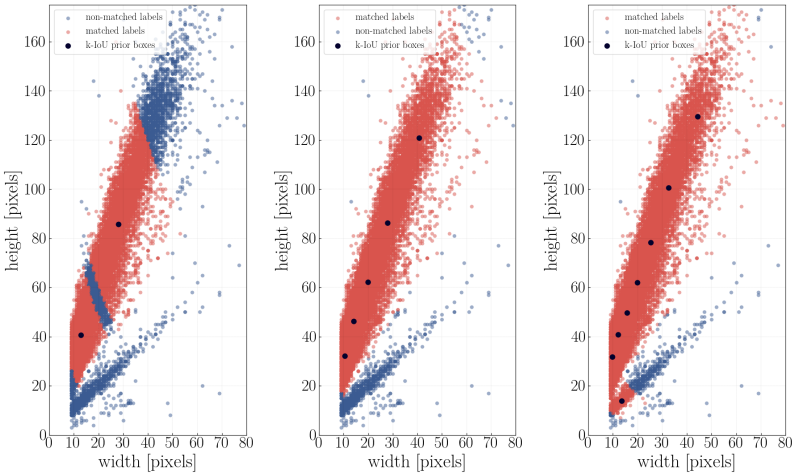
$$\underline{\mu}_c = \frac{1}{\sum\limits_{\underline{g}_i \in S_c} \left(\eta_j^{\text{aspects}}\right)_i} \sum_{g_i \in S_c} \left(\eta_j^{\text{aspects}}\right)_i \cdot \underline{g}_i. \tag{5.10}$$

The algorithm is repeated until the assignment between ground truths and cluster centres is equal to the last iteration step. The weights are determined by the number of samples per class in the DTLD training set.

Figure 5.9 illustrates the clusters for three choices of k from weighted k-IoU clustering. The figure clearly shows, that even for a low number of cluster centres k, a higher coverage for smaller aspect ratios is reached by using sample weights. This supports to improve detection on non-three aspects traffic lights. For all further experiments, $k = 8$ prior boxes is selected. Table 5.4 shows the numeric values of the black cluster centres (= prior boxes) obtained from weighted k-IoU clustering. In general, the majority of prior boxes is relatively small, which corresponds to the size distribution of the dataset (compare Figure 4.5). Moreover, high IoU values are more difficult to reach for small samples. In consequence, a higher number of small prior boxes helps to increase the matching rates for all small ground truths. Aspect ratios are in the range 1.00 and 3.83, which shows that all aspects are covered.

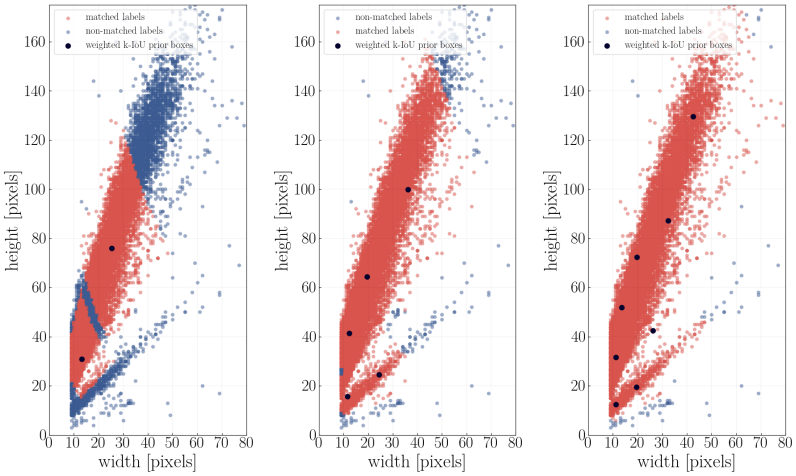
Table 5.4: Prior box sizes (width p^w, height p^h and aspect ratio p^h/p^w) from weighted k-IoU clustering. The to be determined prior boxes are synonymous with the resulting clusters centres $\underline{\mu}_1$ to $\underline{\mu}_8$. The prior box dimensions are fixed for all experiments in this thesis.

	$\underline{\mu}_1$	$\underline{\mu}_2$	$\underline{\mu}_3$	$\underline{\mu}_4$	$\underline{\mu}_5$	$\underline{\mu}_6$	$\underline{\mu}_7$	$\underline{\mu}_8$
$\boldsymbol{p^w}$	11.12	11.55	14.93	19.85	23.13	26.40	32.74	49.10
$\boldsymbol{p^h}$	13.52	35.26	57.17	19.86	76.91	42.08	111.88	103.87
$\boldsymbol{p^h/p^w}$	1.22	3.05	3.83	1.00	3.33	1.59	3.42	2.12



(a) $k = 2$ (b) $k = 5$ (c) $k = 8$

Figure 5.8: Prior boxes from k-IoU clustering.



(a) $k = 2$ (b) $k = 5$ (c) $k = 8$

Figure 5.9: Prior boxes from weighted k-IoU clustering. The matching rate is significantly increased, especially for aspect ratios equal one.

Adaptive Prior Box Locations

As already mentioned, most region proposal networks place the prior boxes in the feature cell centre location. For known image width $\mathcal{I}^w$ and height $\mathcal{I}^h$ as well as prediction feature layer width $\mathcal{F}^w$ and height $\mathcal{F}^h$, the prior box location centres in x- and y-direction are given as

$$p_i^x = \frac{i+0.5}{\mathcal{F}^w} \qquad p_j^y = \frac{j+0.5}{\mathcal{F}^h} \qquad i \in [0, \mathcal{F}^w),\ j \in [0, \mathcal{F}^h). \tag{5.11}$$

As mentioned, a feature cell describes the mapping of one pixel in a feature map to one region (cell) in the input image. The smaller the feature layer, the larger the feature cell. The property, that prior boxes are only placed in the cell centre is adverse for small object detection. It increases the feasible absolute distance between prior box and traffic light. This property is reinforced by deeper network layers (smaller feature layers → larger feature cells). Furthermore, small objects appearing in groups are challenging to be detected, because of an insufficient amount of prior boxes. During prediction, each prior box can only predict one real-world object.

The subsequent section introduces the adaptive prior box locations, an adaptation of the rigid prior box stride used in almost all one- and two-stage detectors. Instead of fixed grid-based prior box locations, an adaptive prior box stride is proposed with respect the prior box size. In theory, larger prior boxes require less locations (or: a larger stride can be used) than small prior boxes in order to guarantee high overlaps. However, rules to determine the absolute prior box stride with respect to the prior box size have to be developed. Figure 5.10 illustrates two equally-sized bounding boxes, which are misaligned by a percentage stride error $\mathcal{E}^s$ with respect to width p^w

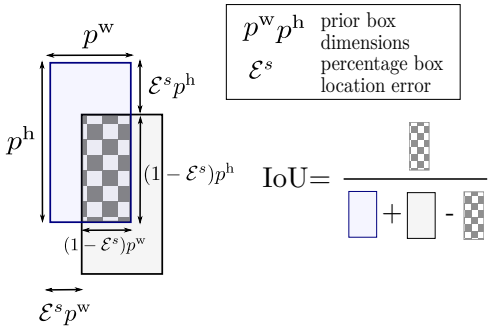


Figure 5.10: Percentage misalignment $\mathcal{E}^s$ between prior box (blue) and ground truth (black). For simplicity, both boxes are assumed as equally-sized.

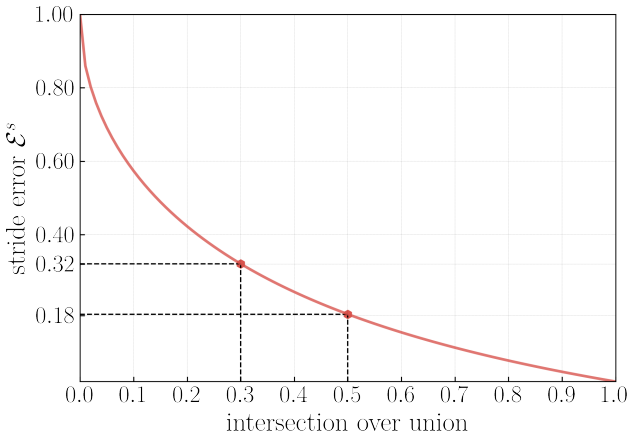


Figure 5.11: Stride error vs. intersection over union. A misalignment of 20% in x-and y-direction causes a drop of the intersection over union by 50 percent.

and height p^h. It helps to better understand the impact of position misalignments caused by high prior box strides on the expectable IoU values between prior boxes and ground truths. Reaching high IoU values is essential for a successful network training. The following derives the relationship between IoU and positioning error, which enables a derivation of the required prior box stride in order to optimize the detection potential. For simplification, equally sized boxes are assumed.

The relationship between IoU and box location/stride error $\mathcal{E}^s$ can be formulated as

$$\mathrm{IoU} = \frac{(1-\mathcal{E}^s)^2}{2-(1-\mathcal{E}^s)^2} \cdot \frac{\cancel{p^{\mathrm{w}} p^{\mathrm{h}}}}{\cancel{p^{\mathrm{w}} p^{\mathrm{h}}}}, \qquad 0 \leq \mathcal{E}^s \leq 1, \tag{5.12}$$

which is intuitively independent of the box size. This equation can be further simplified into

$$\mathcal{E}^s = 1 - \sqrt{\frac{2\mathrm{IoU}}{1+\mathrm{IoU}}} \tag{5.13}$$

resulting from solving a quadratic equation leading to a further yet invalid solution. Details see Appendix B.1.

Figure 5.11 illustrates Equation (5.13). Exemplary, the desired operating point of $\mathrm{IoU}_{\mathrm{min}} = 0.5$ is illustrated as well. In order to guarantee $\mathrm{IoU}_{\mathrm{min}} = 0.5$ as an

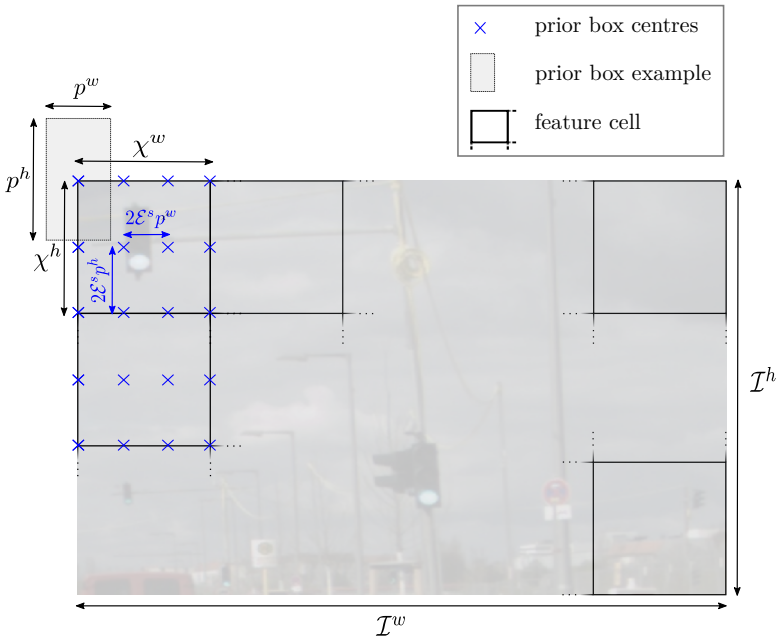


Figure 5.12: Prior box location geometries. The prior box locations are symmetrically designed with respect to the feature cell. The number of locations (step size) is linked to Equation (5.13) in order to guarantee a certain overlap between prior boxes and ground truths.

overlap between prior boxes and ground truths, a stride error of 18 percent has to be guaranteed. In consequence, the task when developing the prior box locations is to parametrize the locations in a manner, so that the misalignment between at least one prior box and all ground truths is smaller than 18 percent. Equation (5.13) can directly be used to calculate the percentage stride for each prior box. It amounts $2\mathcal{E}^s$. For a prior box with a width of $p^w = 10$ and height of $p^h = 30$ pixels, the stride amounts $2p^w\mathcal{E}^s = 3.6 \approx 4$ pixels in x-direction and $2p^h\mathcal{E}^s = 10.8 \approx 11$ pixels in y-direction. However, similar to the rigid baseline approach, this thesis aims to obtain symmetric locations for each cell. Consequently, the prior box locations only have to be determined relatively for one feature cell and not exclusively for the entire image. In order to obtain cell-symmetric locations (normalized to [0, 1]), the absolute, cell-normalized prior box stride is rounded to the closest multiples of $\frac{1}{n}$.

All prior box centres in x- and y-direction in one feature cell are defined as

$$\mathbf{\Lambda} = \begin{bmatrix} (\lambda_0^x, \lambda_0^y) & (\lambda_0^x, \lambda_1^y) & ... & (\lambda_0^x, \lambda_M^y) \\ (\lambda_1^x, \lambda_0^y) & (\lambda_1^x, \lambda_1^y) & ... & (\lambda_1^x, \lambda_M^y) \\ \vdots & \vdots & \vdots & \vdots \\ (\lambda_K^x, \lambda_0^y) & (\lambda_K^x, \lambda_1^y) & ... & (\lambda_K^x, \lambda_M^y) \end{bmatrix}, \tag{5.14}$$

where $0 \leq \lambda_k^x \leq 1$ and $0 \leq \lambda_k^y \leq 1$. Each of the prior box locations in the cell is given as

$$\lambda_i^x(p^w, i) = \begin{cases} i \cdot \overbrace{\text{RTF}\left\{\frac{2\mathcal{E}^s \cdot p^w}{\chi^w}\right\}}^{\lambda_1^x} & \lambda_1^x \leq \frac{1}{2} \qquad i \in (0, K] \quad (5.15a) \\ \frac{1}{2} & \lambda_1^x > \frac{1}{2} \quad (5.15b) \end{cases}$$

where $2\mathcal{E}^s$ is the allowed percentage step size (in order to guarantee a certain IoU value according to Equation (5.13)), p^w is the prior box width and χ^w is the feature cell width in pixels. Figure 5.12 illustrates all variables of Equation (5.15a). In words, the fraction calculates the prior box step size in x-direction normalized to the feature cell width. The RTF-operator rounds the cell-normalized step size to the closest fraction ($\frac{1}{n}$) in order to obtain symmetric locations in the feature cell. If the step size is larger than 50 percent of the feature cell size, only one location in the feature cell centre is used. For instance, the prior box stride of 4 pixels and a feature cell width of 10 pixels results in a normalized stride of $\text{RTF}\{\frac{4}{10}\} = \frac{1}{3}$, i.e. locations at $\{0, \frac{1}{3}, \frac{2}{3}\}$ in x-direction. This corresponds to the procedure the original SSD approach uses (Equation (5.11)). Inserting Equation (5.13) and $\chi^w = \mathcal{I}^w/\mathcal{F}^w$ into (5.15a) leads to

$$\lambda_i^x(p^w, i) = i \cdot \underbrace{\text{RTF}\left\{\frac{2p^w\mathcal{F}^w}{\mathcal{I}^w} \cdot \left(1 - \sqrt{\frac{2\text{IoU}}{1+\text{IoU}}}\right)\right\}}_{\lambda_1^x}. \tag{5.16}$$

Analogous, the normalized prior box locations in one feature cell in y-direction can be calculated by

$$\lambda_j^y(p^h, j) = j \cdot \underbrace{\mathrm{RTF}\left\{\frac{2p^h\mathcal{F}^h}{\mathcal{I}^h} \cdot \left(1 - \sqrt{\frac{2\mathrm{IoU}}{1+\mathrm{IoU}}}\right)\right\}}_{\lambda_1^y}, \qquad \lambda_1^y \leq \frac{1}{2} \tag{5.17}$$

where $j \in (0, M]$. The step size in y-direction is typically higher due to the aspect ratio of traffic lights. K and M are the number of locations in x- and y-direction, respectively. The number of locations can be calculated by $K = \frac{1}{\lambda_1^x}$ and $M = \frac{1}{\lambda_1^y}$.

Intuitively, the prior box locations $\mathbf{\Lambda}$ have to be individually determined for each single prior box $\underline{\mu}_1$ to $\underline{\mu}_8$ stated in Table 5.4. Moreover, depending on the size of the selected prediction feature layer ($\mathcal{F}^w \cdot \mathcal{F}^h$), all locations have to be recalculated.

Figure 5.13 exemplarily illustrates the determined prior box locations (blue) in one example cell using Equation (5.16) and Equation (5.17) when predicting all objects in layer inception_c. The prior box dimension of the illustrated locations is exemplified at the upper left location as a grey box. Intuitively, small prior boxes require a higher number of locations than large boxes. Moreover, the location stride can be increased in y-direction due to the aspect ratio of traffic lights. For a more detailed view on the prior box locations for other prediction layers, e.g. as investigated in Section 5.3.5, see Appendix B.3.

5.2.4 Bounding Box Regression and Loss Functions

During training, each ground truth sample is matched to the prior box with which it reaches the highest IoU value. Furthermore, each prior box is assigned to a ground truth if the overlap is higher than a defined threshold. This simplifies the training step as each prior box is matched to a much higher overall number of ground truths. The original matching threshold value of 0.5 is decreased to 0.3 due to the already mentioned small object issues. One drawback of this matching proceed is, that an assignment for ground truths not overlapping with any prior box can not be made. However, as derived in the previous section, an overlap for each ground truth can be guaranteed by carefully designed prior box sizes and adaptive locations. Using this proceed, the absolute number of unmatched ground truths can be neglected and amounts less than 1 percent.

During training, for each match between ground truth and prior box the relative offsets in x- and y-direction between ground truth g_i and prior box p_i are calculated

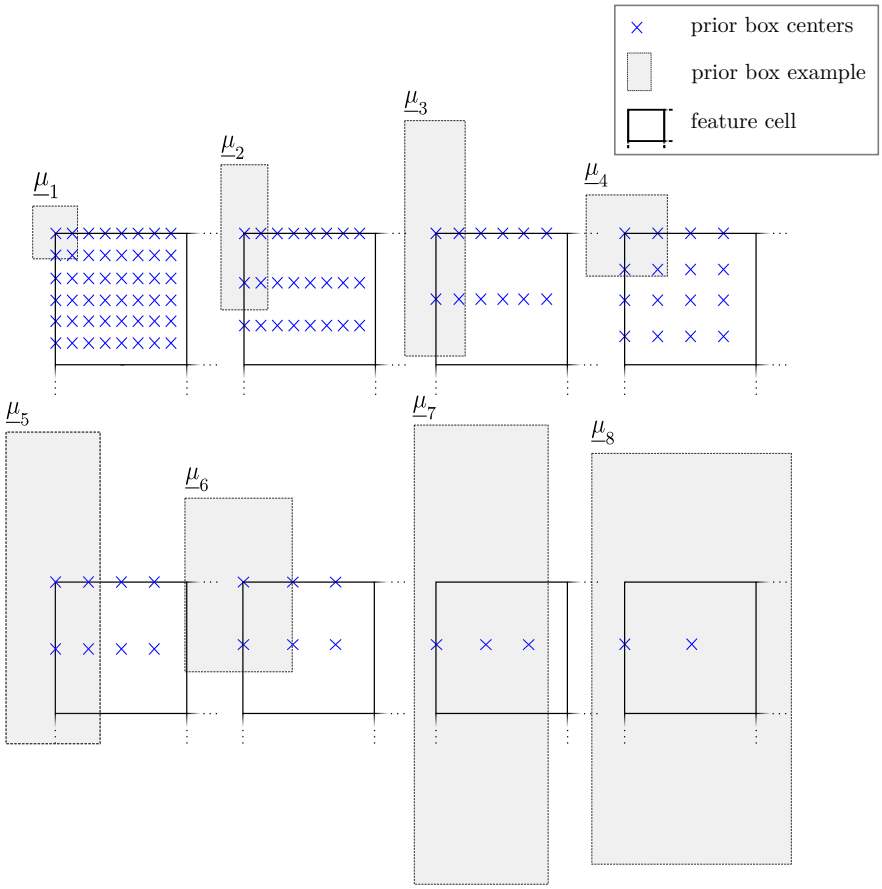


Figure 5.13: Adaptive prior box locations inside one feature cell of the inception_c layer for all eight prior boxes $\underline{\mu}_1$ to $\underline{\mu}_8$. The feature cell size amounts 32x32 pixels. The prior box stride mainly depends on the prior box dimensions. Larger boxes require less locations to guarantee a defined IoU overlap with all ground truths. Moreover, depending on the aspect ratio, the stride can differ in x- and y-direction. Prior boxes dimensions being larger than the feature cell size is very common, especially for deep networks and large objects.

as

$$\delta_i^x = \frac{g_i^x - p_i^x}{p_i^w} \qquad \delta_i^y = \frac{g_i^y - p_i^y}{p_i^h}. \tag{5.18}$$

The offsets are normalized by prior box width p_i^w and height p_i^h, respectively. The delta values express the normalized position misalignment between ground truth and prior box, which is desired to be predicted by the network (as already mentioned, the network predicts offsets with regard to the prior boxes, not absolute coordinates). Please note, that x and y are centre coordinates. Using centre instead of corner coordinates guarantees scale invariance. The width and height regression is done in a logarithmic manner, formulated as

$$\delta_i^w = \log\left(\frac{g_i^w}{p_i^w}\right) \qquad \delta_i^h = \log\left(\frac{g_i^h}{p_i^h}\right), \tag{5.19}$$

expressing the width and height difference between ground truth and prior box. During each training iteration, the localization loss $\mathcal{L}_{\text{loc}}$ is calculated over all matches m_{ik} ($\text{IoU}(\underline{g}, \underline{p}) > \mathcal{T}_{\text{IoU}}$) between actual offset δ and current offset prediction l (offset output vector of the network). A $\text{smooth}_{\text{L1}}$ loss [Gir15] is calculated for all four regression targets leading to

$$\mathcal{L}_{\text{loc}} = \sum_{i=0}^{N_p-1} \sum_{k=0}^{N_g-1} \sum_j m_{ik} \cdot \text{smooth}_{\text{L1}}(l_i^j - \delta_k^j) \qquad j \in \mathbf{C} = \{x, y, w, h\}, \tag{5.20}$$

where N_p is the number of prior boxes, N_g is the number of ground truths and $m_{ik} \in \{0, 1\}$ expresses, whether prior box i and ground truth k is matched. *The goal is to minimize the deviation between the offsets predicted by the network and "true" offset between prior box and ground truth.* Mathematically, the $\text{smooth}_{\text{L1}}$ loss is defined as

$$\text{smooth}_{\text{L1}}(\text{x}) = \begin{cases} \frac{1}{2}x^2 & |x| < 1 \\ |x| - \frac{1}{2} & |x| \geq 1. \end{cases} \tag{5.21}$$

In consequence, this loss less penalizes very small errors, while larger errors are linearly punished. Other works emphasize the benefits of L1-loss over the L2 loss due to more robustness, especially in case of outliers. Figure 5.14a illustrates the L1 loss in blue and for the purpose of comparison the L2 loss in red. For $|l_i^j - \delta_k^j| < 1$ the smooth L1 loss has a smaller gradient than a linear function. For larger errors $|l_i^j - \delta_k^j| \geq 1$ the loss gets linear. This avoids very high loss values in case of outliers, which are obtained when using the L2 loss.

Confidence Loss: Besides the location prediction, one logit vector $\underline{z}_i$ is predicted for each bounding box. The logit vector is of length N and can be formulated as

$$\underline{z}_i = [z^{\mathrm{C0}}, z^{\mathrm{C1}}, \ldots, z^{\mathrm{CN}}]^\top, \qquad \sum_i z^i \neq 1 \tag{5.22}$$

In this chapter only two logit values are predicted, i.e. the present problem is a binary detection problem with $N = 2$ and $\underline{z}_i = (z^{\mathrm{C0}}, z^{\mathrm{C1}})^\top$. In order to guarantee probabilities between 0 and 1, a softmax function is applied to each logit value

$$c^i = \frac{\exp(z^i)}{\sum_m \exp(z^m)}, \tag{5.23}$$

which results in an output confidence vector

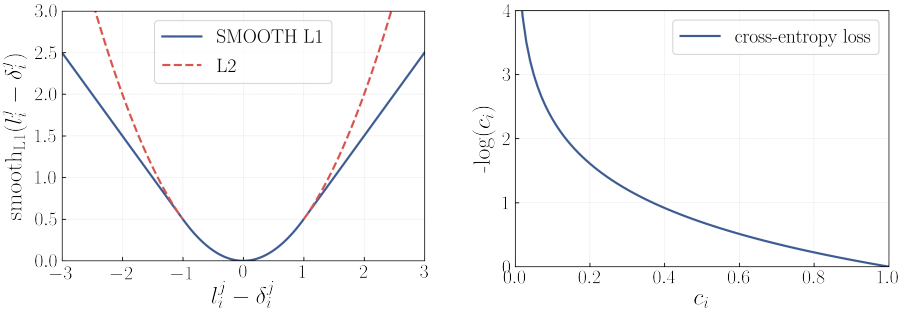
$$\underline{c}_i = [c^{\mathrm{C0}}, c^{\mathrm{C1}}, \ldots, c^{\mathrm{CN}}]^\top, \qquad \sum_i c^i = 1. \tag{5.24}$$

This transformation outputs a probability distribution, in which the sum of all class probabilities is equal to one. During training, a confidence loss is calculated in order to optimize the weights for logits prediction. For the binary classification problem, this can be formulated as

$$\mathcal{L}_{\mathrm{conf}} = \sum_{i=0}^{N_p-1} \sum_{k=0}^{N_g-1} m_{ik} \cdot \log(c_i^{\mathrm{C1}}) - \sum_{i=0}^{N_p-1} \sum_{k=0}^{N_g-1} (1 - m_{ik}) \cdot \log(c_i^{\mathrm{C0}}), \tag{5.25}$$

where c^{C0} denotes the background and c^{C1} the foreground (traffic light) confidence value. Figure 5.14b illustrates the cross-entropy loss in blue. It shows, that small errors (i.e. confidence values close to 1) are less penalized, whereas larger errors (confidence values close to 0) are strongly penalized according to the logarithmic function. By this, the network learns to predict a confidence close to 1 for traffic lights and 0 for non-traffic lights. The overall objective function is a weighted sum between the confidence and localization loss formulated as

$$\mathcal{L}_{total} = \frac{1}{N_{p,m}} (\mathcal{L}_{\mathrm{conf}} + \alpha \cdot \mathcal{L}_{\mathrm{loc}}), \tag{5.26}$$



(a) Smooth L1 loss.

(b) Cross-entropy loss.

Figure 5.14: Smooth L1 loss used for localization loss calculation and cross-entropy loss used for confidence loss calculation. Smooth L1 loss less penalizes small errors, while large errors are linearly penalized. It is more robust towards outliers. The cross-entropy loss strongly penalizes probabilities unequal to the desired value (1.0) in a logarithmic manner.

normalized by the number of matched prior boxes $N_{p,m}$ and using a weighting factor α [LAE+15]. During all experiments, the weighting factor is parametrized as 1, i.e. localization and confidence loss are equally weighted.

5.2.5 Non-Maximum Suppression

As mentioned, multiple prior boxes can be assigned to one ground truth during training. When applying the network, this can lead to multiple predictions on a single real world object, because multiple prior boxes are regressed to one single object instance. A so called Non-Maximum Suppression (NMS) step is used to group overlapping predictions and join them to one single prediction. The NMS used in this work groups all predictions, which overlap by a defined IoU threshold $\mathcal{T}_{\text{IoU,NMS}}$. From this group only one prediction with the highest confidence value is selected as the final output. However, parametrizing this threshold is critical. While many popular works performing object detection on datasets such as Pascal VOC propose to use $\mathcal{T}_{\text{IoU,NMS}} = 0.5$, this choice may be too high for object detection problems with very small objects. This thesis proposes to lower this threshold to $\mathcal{T}_{\text{IoU,NMS}} = 0.3$ in order to adapt to the traffic light use case. Recent research works introduced a learning based NMS [HBS17], which is not used in this work, because the NMS does not cause a significant amount of errors in this approach (see error analysis in

Section 5.3.8). During evaluation, multiple predictions on one single ground truth are treated as false positives.

5.3 Experimental Investigations and Evaluation

This section presents all investigations and evaluations performed in order to optimize the network with respect to the selected metrics. Because the approach itself has a too high amount of parameters to be entirely investigated and, moreover, many parameters are not expected to have a major impact, a suitable subset is selected. An additional limitation is the required time for both, training and evaluation, by taking one up to two days depending on the network configuration. *The focus lies on evaluating the core parts of the approach, which are assumed to be crucial for the network performance.* This includes the following investigations:

- **Image input**: DTLD contains 12 bit raw images converted into the required 8 bit network input. Different ways are presented. Moreover data augmentation leading to different image appearances is tested.

- **Ground truth input**: Different experiments investigating the impact of ground truth filtering are performed. Filtering includes size as well as class filters.

- **Prior box adaptations**: The proposed, novel methods to adapt prior box sizes and locations are evaluated. It investigates the impact of the adaptive locations with regard to the selected prediction layers.

- **Optimal feature layers**: The network has many layers, which can be used for prediction. The performance is evaluated based on different choices as well as a comparison of single and multi-layer prediction.

- **Data amount**: The impact of training set size is investigated.

- **Detailed evaluation**: This includes a high number of in-depth evaluations - showing evaluations on different subsets as well as false positive investigations. This helps to better understand error cases and find error patterns.

Implementation: The implementation of SSD and all introduced adaptations and novelties is done in the Caffe framework [JSD+14]. All prior box adaptations are implemented in C++, whereas the entire system for automatic training/evaluation is implemented in Python.

Image Cropping: Inspired by the results presented in Figure 4.7, all experiments are only performed on the upper half of the DTLD images. After cropping, the image size amounts 2048×512 pixels. There are two main reasons for this choice: runtime and GPU memory. Intuitively, halving the image size approximately halves the runtime. Moreover, the GPU memory of a NVIDIA Titan Xp is not sufficient to process the entire image using an Inception-v3 network. Even though cropping may impact the absolute results (less false positives), the core statements of all experiments are transferable to experiments on the full image. The image size in combination with the memory of the utilized GPU limit the batch size to one.

Key Metric: The key metric used for comparison of different experiments is the log-average miss rate. The values $\{10^{-2}, 10^{-1}, 10^{0}\}$ are used as sampling points, because within this range a high gradient is observable indicating a strong performance change. The LAMR value is calculated according to Equation (4.7) for four different *detection scenarios* corresponding to the findings in Section 5.1. They differ in the choice of the minimum width and ignore regions shown in Table 5.5. Predictions on ignored labels are not treated as false positives, but the ignored regions are also not required to be detected. DS1 can be considered as the desired detection scenario - whereas all traffic lights included in DS4 necessarily have to be detected.

Best Snapshot Selection: In order to select the best iteration point during training, a snapshot is stored every 10 000 iterations. All iterations are evaluated on the test set and the best model is automatically selected according to the key metric. All detailed evaluations are performed only on this snapshot. Typically, results are converging well and do not show overfitting characteristics, i.e. no decreasing test set performance for higher iteration numbers.

Further Training Details: Several parameters are kept constant during all experiments. This includes the image mean (60, 60, 60), learning rate (10^{-4}), batch size (1), weight decay ($5 \cdot 10^{-4}$), momentum (0.9) and gamma (0.1) as parameters of the stochastic gradient descent. During the training label matching step, all labels with an overlap larger than 0.3 with a prior box are used. The localization weight is equal to 1.0. For each positive sample, three negatives are used for training using a hard negative mining. Hard negative mining calculates the loss for all negatives and only uses the top-k ranked according the loss value (high loss values indicate hard examples). The training images are shuffled to avoid overfitting on one city because DTLD training file images are sorted city-wise.

Further Evaluation Details: Multiple detections on a single object are counted as false positives and assumed as an error of the NMS step. The prediction with highest confidence is treated as true positive.

Table 5.5: Evaluation constraints based on different choices of minimum width and ignored labels.

scenario	ignore regions	minimum width
DS1	back, left, right, tram, pedestrian, cyclist, occluded	8
DS2	back, left, right, tram, pedestrian, cyclist, occluded	10
DS3	not_relevant, occluded	8
DS4	not_relevant, occluded	10

5.3.1 Statistical Variance of Unique Experiments

Because the results of one unique experiment (equal configuration/parameters) shows a statistical variance, the most thorough proceed is to repeat each experiment multiple times and take the median/mean as a representative value for further comparisons. Because of the high training and evaluation time of one experiment and the lack of hardware this is not feasible in this thesis. Instead, this experiment shows the statistical range of one experiment by executing it ten times. The statistical variance of the result shows the range, in which potential metric improvements can be assumed to be statistically meaningful. This avoids the time-consuming proceed of repeating each experiment multiple times and taking the median.

Figure 5.15 shows the results after training one unique experiment ten times with exactly equal network parameters. Evaluation is done on each of the four introduced detection scenarios. The box plot illustrates the respective median value (central line) as well as the Interquartile Range (IQR) as the length of the coloured box. The distance between both outer lines amounts 3IQR. Black single points mark the result of each individual experiment. Median and IQR values are more robust than using mean and standard deviation.

The main conclusion of this experiment is, that *changes in log-average miss rate larger than 1 percent can be rated as an improvement/deterioration.* Differences smaller than this value may only be caused by statistical variance. These values are used for assessment of all following experiments.

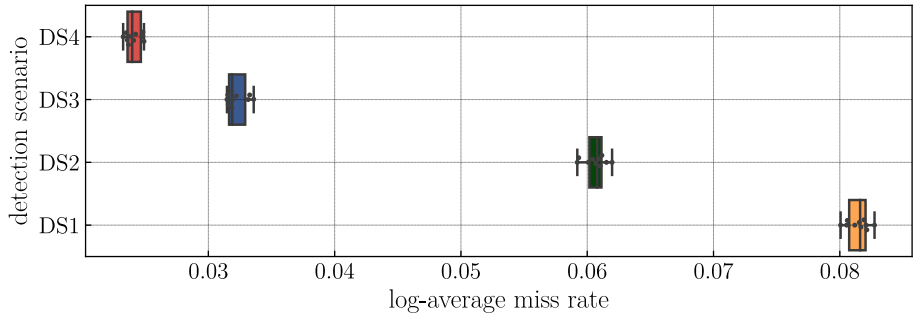


Figure 5.15: Statistical variance of a unique experiment during 10 runs. Differences smaller than 0.5 percent in LAMR may only be caused for statistical reasons. The results help to classify further comparison experiments.

5.3.2 Impact of Image Conversion

The images of DTLD are recorded in 12 bit depth with a camera using a bayer filter. CNN-based detection typically uses 8 bit representation and the pre-trained weights are typically trained on 8 bit images. For this reason, the DTLD raw images are initially converted before being used as training images.

8 Bit Compression: For this work, two different methods for compression are used. The most common way is a simple bit shift given as $\mathbf{I}_{8\text{bit}}(x, y) = \frac{1}{16} \cdot \mathbf{I}_{12\text{bit}}(x, y)$ in order to obtain the desired value range [0, 255]. A more sophisticated proceed for conversion uses the property, that the image sensor used for DTLD supports a High Dynamic Range (HDR) mode, which creates 16 bit images from two exposures. These 16 bit images are compressed to 12 bit according to the HDR compression curve in Figure A.1 before read out. Using this curve enables to the possibility to calculate a 16 bit image from the 12 bit before converting it into 8 bit, see sensor data sheet [ONS17].

Figure 5.16 illustrates the impact of both conversion methods. The upper images show the result from a simple down-shift. Many saturated pixels in the traffic light lamp are visible. This may especially affect the state recognition. The bottom image shows the result after applying the curve and shifting to 8 bit. The lamps are not saturated and clear colors are visible. Disadvantage is a slight decrease in dynamic range in darker regions, which is not expected to affect detection.

Demosaicing: Demosaicing or debayering is the process of color image reconstruction from an image recorded with a bayer color filter (1 channel). There exist different algorithms for demosaicing, which are compared in a work of Losson et al [LMY10]. The impact of a simple approach, bilinear interpolation, is compared with an edge-aware approach as utilized in [Its15].

Results: The impact of decompanding and the type of demosaicing is shown in Table 5.6. Using neither decompanding nor edge-aware debayering (standard debayering) produces the highest log-average miss rate. Edge-aware debayering produces a slight increase in performance. *Decompanding combined with edge-aware debayering shows the best performance with a significant decrease of approximately 1 percent log-average miss rate.* This best performing combination is used for all further experiments.



Figure 5.16: DTLD images with (right) and without (left) decompanding to 16 bit before bit shifting to 8 bit. Decompanding enhances the dynamic range in bright regions (compare sky and traffic light lamps) and helps to avoid saturated pixels inside the traffic light bulb.

Table 5.6: Impact of decompanding and edge-aware debayering. Decompanding and edge-aware debayering improve the detection performance in terms of the log-average miss rate.

decompanding	✗	✗	✓
edge-aware demosaicing	✗	✓	✓
LAMR(DS1)	0.086	0.081	**0.076**
LAMR(DS2)	0.065	0.061	**0.056**
LAMR(DS3)	0.036	0.032	**0.028**
LAMR(DS4)	0.027	0.024	**0.021**

5.3.3 Impact of Ignore Regions

The detection requirements presented in Table 5.5 show, below which object resolution and which sub-classes are not mandatory to be detected. A key question is, whether the usage of non-mandatory objects, i.e. objects smaller than 8 pixels and non-relevant classes, is beneficial with respect to the performance on mandatory objects. This question cannot be directly answered and has to be empirically investigated. There exist three different possibilities how to treat non-mandatory regions during the training process:

- **Positive regions**: Treat all non-mandatory regions as positives. The hope is, that the higher number of training examples supports the performance on mandatory objects.

- **Negative regions**: Treat all non-mandatory regions as negatives (background). However, this may lead to a confusion of the classifier as these regions show a similar appearance to positive examples.

- **Ignore regions**: Treat all non-mandatory regions as ignore regions, i.e. neither positive nor negative examples. Ignore regions have no impact on the network optimization. This version is not part of the original SSD implementation and is implemented for the investigations in this section.

Please note, that the goal is to optimize the performance on mandatory samples according to the scenarios DS1 to DS4. Experiments are executed to investigate the impact of the label selections.

- **Experiment 1**: Use all labeled samples for training, independent of size or class.

- **Experiment 2**: All non-mandatory regions are indirectly trained as background samples. Indirectly, because the SSD framework does not support an explicit selection of negative samples. Instead, it selects the hardest negatives based on their loss value (hard examples typically have a high loss). The non-mandatory samples are allowed to be selected as negative samples.

- **Experiment 3**: Ignore small samples and non-vehicle traffic lights. Differing from experiment 2, these regions cannot be added to the background class.

- **Experiment 4**: Ignore small samples and non-frontal traffic lights. Differing from experiment 3, all non-vehicle, frontal traffic lights (pedestrian, bus/tram) are also added to the traffic light class.

Table 5.7: Impact of label preprocessing on the detection result in terms of the log-average miss rate.

	Experiment No.			
	1	2	3	4
LAMR(DS1)	0.076	0.074	0.071	**0.070**
LAMR(DS2)	0.056	0.053	0.052	**0.050**
LAMR(DS3)	0.029	0.028	**0.022**	0.027
LAMR(DS4)	0.021	0.020	**0.016**	0.019

Although the evaluation aims to detect traffic lights as of 8 pixels, 6 pixels is used as a cut-off value during training. This avoids quantization effects of labels as well as labeling inaccuracies and weakens the border between positive and negative or ignored samples.

The results are shown in Table 5.7. As expected, the implementation of ignore regions shows a significant improvement compared to using all labels for training or adding non-mandatory classes to the background class. For the purpose of flexibility, the configuration in Experiment 3 (ignore regions) is used for all further experiments.

5.3.4 Impact of Data Augmentation

Inspired by the original approach [LAE+15] several data augmentation modes are investigated. The following modes are not used: mirroring, expanding and channel switching. Especially the mirroring and channel switching mode are not useful for traffic light recognition, because they distort color representation and active light position of traffic lights. All modes are applied with a probability of 50 percent, i.e. each mode is applied to every second image on average.

- **Image Normalization**: Each image is normalized by the dataset mean. The dataset mean amounts approximately [60, 60, 60] in RGB channel order.

- **Random Brightness**: Random brightness is added according to the rule

$$\mathbf{I}_{\mathrm{RB}}(x, y) = \mathbf{I}(x, y) + \beta \qquad -32 \leq \beta \leq +32 \tag{5.27}$$

  on R, G and B channel. The brightness bias β is randomly chosen between ± 32 to augment the images in a realistic manner.

- **Random Contrast**: Random contrast is added according to

$$\mathbf{I}_{\mathrm{RC}}(x, y) = \alpha \cdot \mathbf{I}(x, y) \qquad 0.5 \leq \alpha \leq 1.5 \tag{5.28}$$

  applied to all three channels, where α is randomly chosen between 0.5 and 1.5.

- **Random Hue**: For random hue operation, the RGB image is converted into Hue Saturation Value (HSV) space. Subsequently, Equation (5.27) is applied to the hue channel. An offset value of $-18 \leq \beta \leq +18$ is used.

- **Random Saturation**: For random saturation operation, the RGB image is converted into HSV space. Subsequently, Equation (5.28) is applied to the saturation channel. A multiplication value of $0.5 \leq \alpha \leq 1.5$ is used.

- **Random Color Balancing**: An unbalanced white balancing is one of the most common issues in relation to public datasets. In order to design the networks robust against white balancing errors it is modelled by using a random color balancing. According to Equation (5.28) each channel is multiplied by a random multiplication factor α_1 up to α_3. A multiplication factor between $\alpha_{\min} = 0.8$ and $\alpha_{\max} = 1.2$ is randomly chosen for this purpose.

Goal of data augmentation is to learn the network to handle different image "looks", which potentially appear because of rare weather/light conditions or due to white balancing issues. Figure 5.17 illustrates all investigated augmentation methods on a single scene. The modes are parametrized to produce slight changes in appearance in order to maintain a realistic image look.

Table 5.8 shows the results with and without data augmentation. Each data augmentation mode is tested independently and one experiment evaluates the impact of all modes applied in parallel. Unfortunately, the benefit reported in the paper [LAE+15] can not be confirmed here. Results are surprisingly similar, independent of the data augmentation mode itself. In contrast to the datasets used in [LAE+15], in which all images are recorded by various cameras with almost no similarity (focal length, brightness, resolution and others), DTLD has a constant setting leading to a more uniform image look. Apparently, a use of data augmentation is not needed if the dataset itself provides a variation due to recordings at different weather conditions, day times or seasons. However, all further experiments are performed with activated data augmentation because it is expected to show a superior performance when applying to other datasets or an other type of camera or lens.



Figure 5.17: Data augmentation modes: Original, random brightness, random contrast, random hue, random saturation and color balancing (top to bottom). The parametrization is chosen to obtain small modifications in order to maintain a realistic look.

Table 5.8: Impact of data augmentation methods on the detection performance in terms of the metric log-average miss rate. Data augmentation does not show the expected improvements. Other works applying SSD to other datasets reported significant benefits from data augmentation. The key difference between DTLD and these datasets is, that DTLD is recorded with a constant camera setting and generic datasets use images of various cameras, which requires much more robust detectors in order to handle diverse input streams.

no augmentation	✓						
random brightness		✓					✓
random contrast			✓				✓
random hue				✓			✓
random saturation					✓		✓
random balancing						✓	✓
LAMR (DS1)	0.070	0.069	0.069	0.071	0.070	0.071	0.073
LAMR (DS2)	0.050	0.050	0.050	0.052	0.051	0.052	0.054
LAMR (DS3)	0.027	0.026	0.025	0.027	0.027	0.027	0.027
LAMR (DS4)	0.019	0.019	0.019	0.020	0.019	0.020	0.020

5.3.5 Impact of Network Depth and Prior Box Locations

This section deals with the impact of the feature layers used for prediction as well as the impact of the proposed adaptive prior box locations. In general, finding the optimal prediction layer refers to finding the optimal receptive field size for the task of traffic light detection. Because this mainly has to be empirically determined, the SSD prediction head is set on top of different feature layers in various experiments. Other works often use the default layers used in the original paper, however the optimal feature layer is always dependent on the specific object class. The second property, the impact of the prior box stride, is investigated by a comparison between the original prior box locations and the proposed, adaptive prior box locations. The experiments are performed along with a variation of the prediction feature layer, because it has to be expected to have a major impact on the efficiency of adaptive locations. *In theory, the effect of adaptive prior locations increases for deeper network layers* (i.e. smaller feature maps and larger original prior box stride).

Optimal prediction layer: Figure 5.18 illustrates the log-average miss rate utilizing different feature layers, namely inception_a1 to inception_a3, inception_b1 to inception_b4 and inception_c1 to inception_c2 for prediction. Within this experiments, all sizes are predicted in one single feature layer. Using the original

prior box locations, the network layer inception_a3 shows the best performance independent of the detection scenario. An interesting property is the difference to the adaptive prior box locations, with which the optimal prediction layers move into the inception_b3 block. *This shows, that the performance drop in the original SSD for deeper network layers is mainly caused by the increased prior box stride. This can be partially compensated by the proposed adaptive prior box locations.*

Optimal prior box stride: Both properties, the choice of the prediction feature layer, as well as the prior box stride are investigated in parallel. Because different feature layers with varying size are used, an important conclusion will be, whether the adaptive step size has a greater impact for smaller feature layers, because in these layers the original absolute prior box stride is larger. Figure 5.18 illustrates the log-average miss rate for the original prior box stride (one location in the cell) in blue/red as well as the adaptive prior box locations (see Section 5.2.3) in light blue/ light red for DS1 and DS4. *Besides the significant gain in detection performance when using adaptive prior box locations, this trend strengthens for deeper network layers.* In each block, the feature map size is halved (compare Table 5.3) causing higher prior box strides leading to a decrease in performance. This can mostly be compensated by the proposed adaptive locations.

More details are shown in Figure 5.19, illustrating the impact of adaptive prior locations on the miss rate. The values are collected for FPPI=10^{-1} and based on inception_b as a prediction layer. The result confirms the presumption that the adaptive locations have a greater impact for smaller objects. In the range between 8 and 15 pixels, the miss rate can be reduced by a factor of two. For objects larger 15 pixels the performance is roughly equal. For these sizes, the original prior box stride is already sufficient and more prior box locations are not needed. This coincides with the findings and explanations in Section 5.2.3 and Table B.1. Using more prior box locations for larger prior boxes is likely not improving the detection ability, but increase the overall runtime.

Figure 5.20 illustrates a similar investigation for the experiment with inception_c as a feature layer. Instead of a stride of 16, the original prior box stride is 32 in this layer. This causes an increasing miss rate for even larger objects - which can be once more compensated by using the adaptive prior box locations. This result clarifies, that a decrease in performance for deeper network layers may not be caused by the features themselves, yet by the increased original prior box stride. *The proposed adaptive stride is able to avoid this side-effect making a reliable small object detection possible - independent of the feature map size.*

One vs. multiple feature layers: In contrast to other region proposal methods, SSD is able to use multiple feature layers for prediction. As mentioned, a typical proceed is to predict small objects in early and large objects in deep network stages.

In order to reduce the number of experiments, the previous results are used to limit the number of experiments. *For all experiments, the adaptive prior box locations turn out to be more powerful.* Unlike reported in the original paper, using more than one layer is not expected to have a significant impact. In the original paper, object detection is performed on PASCAL VOC containing various object classes (human, animals and other objects) showing a clearly higher variation in size than the present traffic light detection task.

Table 5.9 shows all evaluated network constellations. The choice of feature layers is based on the previous one-layer results. The performance is close to the best one-layer prediction cases and no gain in performance can be reported. The benefits of multiple layers reported in the original paper is mainly due to different object classes, which have different requirements on the receptive field. When only predicting one object class such as traffic lights this effect may not be present. Utilizing inception_a3 as the single prediction layer shows the best performance-runtime trade-off.

Runtime: Besides the log-average miss rate results, Table 5.9 also summarizes the runtime for each experiment. *In each configuration, the adaptive prior box network shows a slightly higher runtime.* This is mainly caused by the NMS step processing a higher number of bounding boxes and the final decoding of the increased number of predicted boxes. With regard to performance, the optimal feature layer map is inside the inception_b block. With regard to performance and runtime, selecting a feature map of the inception_a block seems to be the best choice.

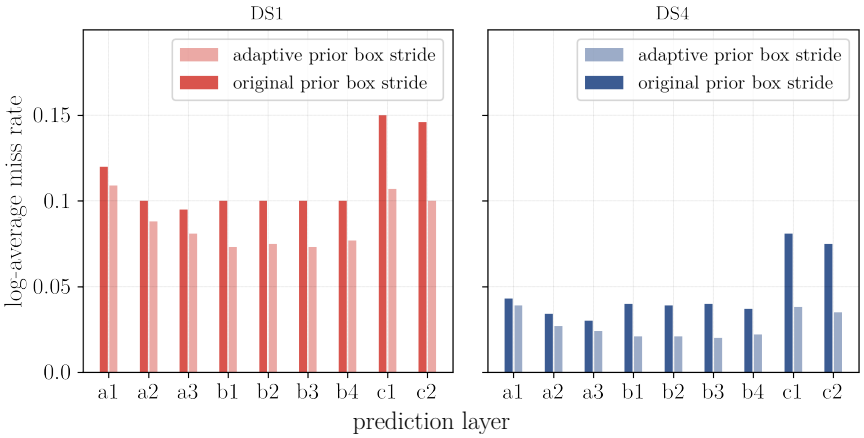


Figure 5.18: Impact of the prediction layer selection and the adaptive prior box stride on the log-average miss rate for DS1 (left) and DS4 (right). In general, the adaptive stride boosts the performance with rising impact for deeper network layers.

Table 5.9: Detailed results for single-layer and multi-layer prediction. The latter shows an improvement when using the original prior box locations, which vanishes when using the proposed, adaptive prior box locations.

	prediction feature layers									**log-average miss rate**				**runtime[ms]**
										original stride		adaptive stride		
	a1	a2	a3	b1	b2	b3	b4	c1	c2	DS1	DS4	DS1	DS4	
Single-Layer	✓									0.12	0.04	0.11	0.04	64 | 69
		✓								0.10	0.03	0.09	0.03	65 | 69
			✓							0.09	0.03	0.08	0.02	66 | 68
				✓						0.10	0.04	0.07	0.02	68 | 70
					✓					0.10	0.04	0.07	0.02	72 | 74
						✓				0.10	0.04	0.07	0.02	76 | 75
							✓			0.10	0.03	0.08	0.02	82 | 83
								✓		0.15	0.08	0.10	0.04	84 | 90
									✓	0.14	0.07	0.10	0.03	87 | 95
Multi-Layer			✓				✓	✓		0.08	0.03	0.08	0.02	89 | 89
	✓		✓							0.09	0.04	0.09	0.04	68 | 72
	✓	✓	✓							0.10	0.03	0.09	0.03	70 | 75
			✓				✓			0.09	0.03	0.08	0.02	85 | 86
			✓	✓			✓			0.08	0.03	0.07	0.02	88 | 91

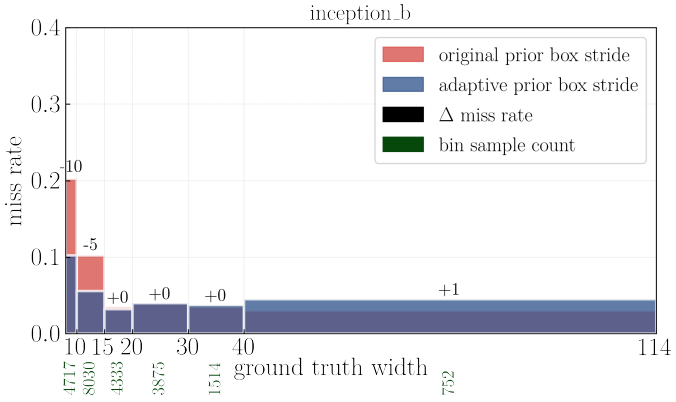


Figure 5.19: Size-dependent impact of adaptive prior locations in *inception_b*. As expected, a clear improvement for very small objects can be observed, which benefit from a higher number of prior box locations in each feature cell. Larger objects are hardly influenced. Values are collected at FPPI=10^{-1}.

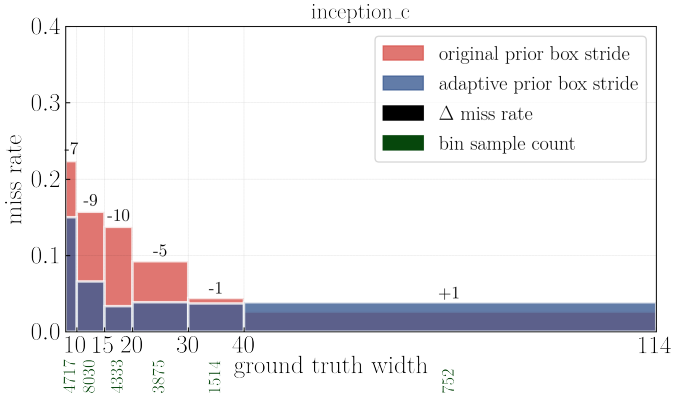


Figure 5.20: Size-dependent impact of adaptive prior locations in *inception_c*. As expected, the impact of the adaptive locations is even larger than in *inception_b* due to the smaller feature map causing a higher prior box stride. This can be compensated by a higher number of prior box locations inside one feature cell. Values are collected at FPPI=10^{-1}.

5.3.6 Impact of Training Data Amount

A common opinion on the impact of data quantity on detection ability of CNN-based methods is that more data improves performance. In order to get an impression on the impact of the amount of training data, four experiments are executed dividing the entire training set into 25, 50 and 75 and 100 percent portions. Four networks are trained using the mentioned training data portions and evaluated on the entire test set. It is guaranteed, that each subset contains images of all cities.
Figure 5.21 illustrates the log-average miss rate for DS1 to DS4 as a function of the training data portion. *A first finding is, that more data helps - however the curves indicate, that a convergence will occur at a certain extent.* This finding corresponds to the results of other works, such as on pedestrian recognition [BKFG19]. Instead of the typical proceed of just adding more data, a more promising approach is to investigate the current system on its weaknesses in order to add more specific data. The weaknesses show the edge cases challenging the system, which are often underrepresented in the dataset. In the present work, this can be done by using the extensive label attributes (orientation, occlusion, status, pictogram, ...) and determining the types of traffic lights which are poorly detected. Instead of recording more data, the mentioned investigations show *which data* may help to improve the system in a more efficient way. More detailed evaluations on sub-categories are presented in the following section.

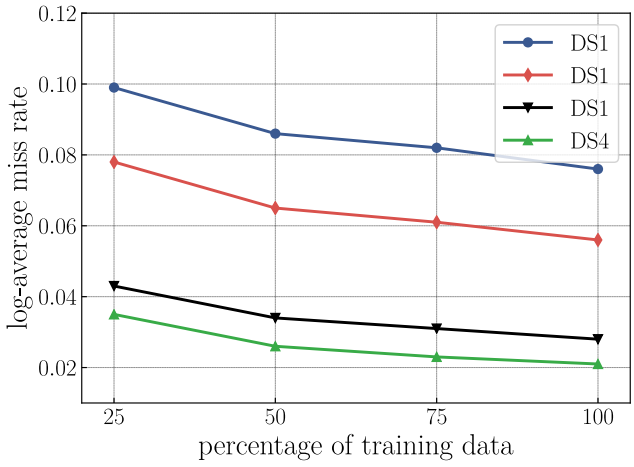


Figure 5.21: Impact of the amount of training data on the log-average miss rate. A higher amount of training data helps to improve the performance, however a certain convergence is recognizable.

5.3.7 Hyperparameter Optimization

One set of parameters not yet optimized are the parameters of the stochastic gradient descent used for minimizing the objective functions. These parameters are not part of SSD, but commonly used to optimize CNN-based approaches. Figure 5.22 illustrates the results with respect to the selected optimization parameters.

Learning rate: The learning rate controls how fast the network learns. The parametrization determines the step size during training process, with which an optimal solution is approximated. Very small values may lead to overfitting, very high step sizes ensure an avoidance of overfitting and a more global solution. Figure 5.22 supports this theoretical assumption by showing the optimal performance at approximately 10^{-3} among very large or small learning rates.

Momentum: Momentum is typically used to accelerate convergence by remembering the update of the previous iteration. Figure 5.22 proves, that the choice of the momentum value has a slight impact on the results.

Weight decay: Weight decay is an additional term in the weight update used to avoid overfitting. Figure 5.22 shows, that small choices are good parametrizations, whereas larger values worsen the result.

Gamma: The gamma parameter is used to decrease the learning rate during training. This supports convergence by starting with a moderate learning rate, which is decreased with rising iteration number. In detail, a gamma of 0.1 is applied at 80 000 and 120 000 iterations decreasing an initial learning rate of 10^{-3} to 10^{-5}.

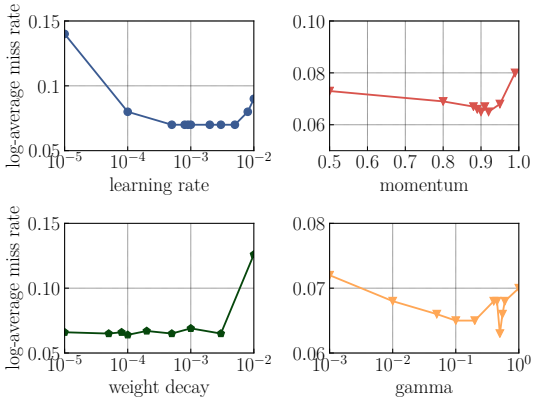


Figure 5.22: Results of hyperparameter optimization.

5.3.8 Detailed Results

Besides the already presented comparison of different experiment configurations, a closer look on the properties of the best network is presented in the following. A clear focus lies on evaluations, which show the strengths and weaknesses of the system and particularly help to understand how to improve the system in the future.

Full ROC characteristic: So far, the performance is evaluated by the LAMR metric only considering 3 points on the ROC curve. Figure 5.23 illustrates the entire ROC plot of the best training model for all four detection scenarios. All ROC plots use the detection output confidence as the discrimination threshold. A noticeable property, is that DS3 and DS4 almost reach 0 percent of miss rate for high FPPI values, whereas DS1 and DS2 are saturating at higher values. *This shows, that not the object resolution, but rather the type of traffic lights challenges the detector.* DS1 and DS2 include traffic lights responsible for other lanes (e.g. turning left/right), which are often rotated away from the vehicle and thus are more challenging to detect.
All four curves show a similar behaviour, between 10^{-1} and 10^{-3} the detection performance breaks down due to two main reasons. On the one hand, generating one false positive every 100 images can be rated as a very performant system with respect to the current state of the art. In the vehicle environment, many sources of false positives showing a similar appearance to traffic lights exist. On the other hand, databases with error rates smaller than 10^{-2}, i.e. less than one error each 100 images, hardly exist. Reaching very high annotation quality can only be ensured by extensive quality control, multiple labeling cycles and other techniques requiring a time- and cost-expensive proceed. In consequence, *the dataset quality limits the analyzability of operating points with very low false positive rates.* This presumption will be confirmed and addressed in Chapter 8.

Performance (aspects-wise): While conventional traffic light detection methods focus on traffic lights with three aspects, SSD simplifies to predict traffic lights of any aspect ratio. Figure 5.24 illustrates the miss rate curve for one aspect, two aspects, three aspects and four aspects, respectively. Despite carefully designed prior boxes described in Section 5.2.3, which are able to also match traffic lights with more/less than three aspects, the detection performance is significantly lower than for non-three aspects traffic lights. An additional issue is the absolute number of training examples for these cases - which is still low, even for a large-scale dataset such as DTLD. The miss rate curve for traffic lights with four aspects contains significantly less operating points because most traffic lights with four aspects are bus/tram traffic lights, which are ignored during evaluation (see Table 5.5). In general, it must be kept in consideration, that for traffic lights with one or two aspects high IoU values are harder to reach - which negatively affects the ROC curve.

Performance (state-wise): A further important property is the detection performance with respect to the traffic light state. Therefore, the detection performance is evaluated separately for each state. Five curves result, describing the miss rate for the states green, yellow, red-yellow, red and off. Figure 5.25 illustrates all five curves. A striking characteristic is, that the states green, red and red-yellow reach almost similar performance, whereas yellow and especially traffic lights with state off turn out to be more challenging to detect. The state red-yellow, which is rather underrepresented in the dataset, seems to benefit from its property of having two active lights - and is thus easier to detect than all other cases. The other extreme - the state off - has no active light and turns out to be the most challenging state to be detected. The state yellow ranks between off and the others, which may be caused by a low amount of yellow samples in the dataset, but also the lower intensity of the yellow light.

Performance (pictogram-wise): This evaluation shows the impact of the pictogram, i.e. circular and the direction arrows on the detection performance. Figure 5.26 shows the evaluation result. Despite the low amount of arrow pictograms, the detection performance for arrow left and straight is as good as for circular inlays. Only right arrows show a slightly higher miss rate, which aligns for higher FPPI values. This is an encouraging result - although arrow pictograms show a lower bulb intensity, the detection ability seems not be be significantly decreased.

Performance (position-wise): Figure 5.27 illustrates the miss rate values with respect to the ground truth position in x- and y-direction in the camera image. The values are raised at FPPI=10^{-1}. Instead of the upper left corner, the ground truth centre coordinates are used to ensure scale invariance.
The horizontal position (Figure 5.27a) seems to have an impact on the detection performance - with a symmetric rise of the miss rate towards the image borders. This may be due to two reasons. On the one hand, objects close to the image edges are very large and rare (compare Figure 4.7) - and the latter may influence the learning process. On the other hand, objects very close to the borders are likely truncated and thus more challenging to be detected.
The vertical position (see Figure 5.27b) seems to have a greater impact on the miss rate. Less surprising, the miss rate is higher at the upper image border. Because this likely causes cutting-off the active light, a subsequent state/pictogram recognition may be affected as well. For rising y-positions an approximately constant performance can be observed - which in turn strongly decreases for higher y-values. The amount of miss rate increase is surprising - to a lesser extent than an increase itself, because higher y-positions are correlated to a higher object distance. Visual investigation explained this property. Almost all objects within this bin are one aspect traffic lights, for which the performance is generally poor (see Figure 5.24).

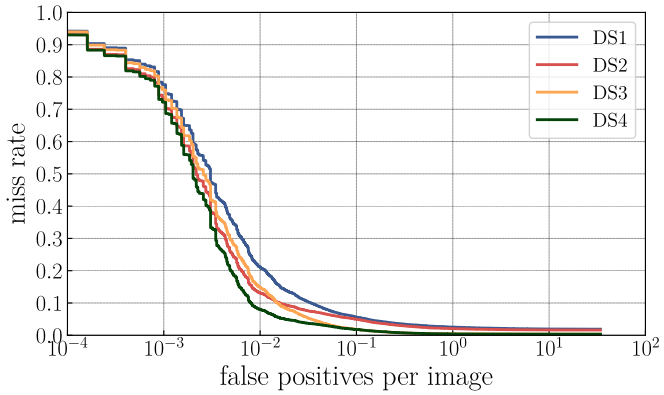


Figure 5.23: Miss rate curve of the best experiment. When focusing on relevant traffic lights, almost all instances can be detected for higher FPPI values. For very small FPPI values, small miss rates can not be reached due to two reasons: system limitations, but also the quality of the database.

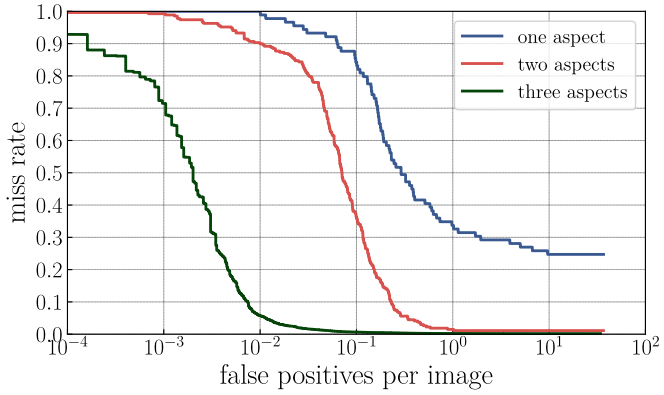


Figure 5.24: Miss rate (DS4) for different aspects, i.e. different number of light bulb components. Despite the use of prior boxes, which reach a high matching rate for non-three aspect traffic lights, the overall performance is significantly lower than for the standard case with three light bulbs. This may be caused by a lack of samples for these classes.

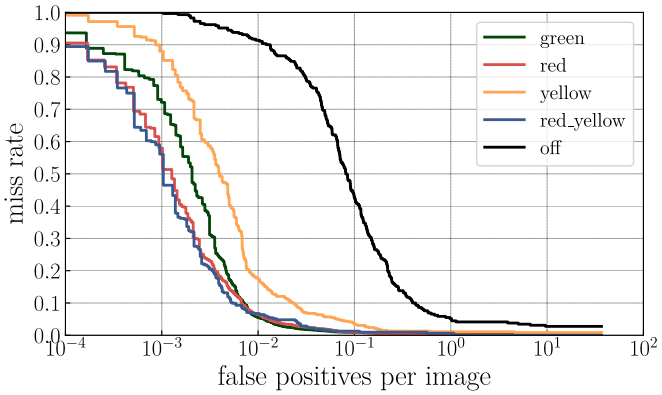


Figure 5.25: Miss rate (DS4) with respect to the traffic light state. Red, red-yellow and green perform best. Traffic lights switched off are the most challenging, most likely because the active light as the most important feature is not present.

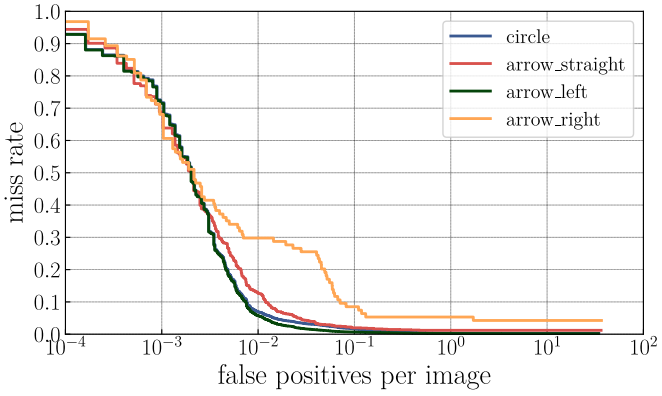
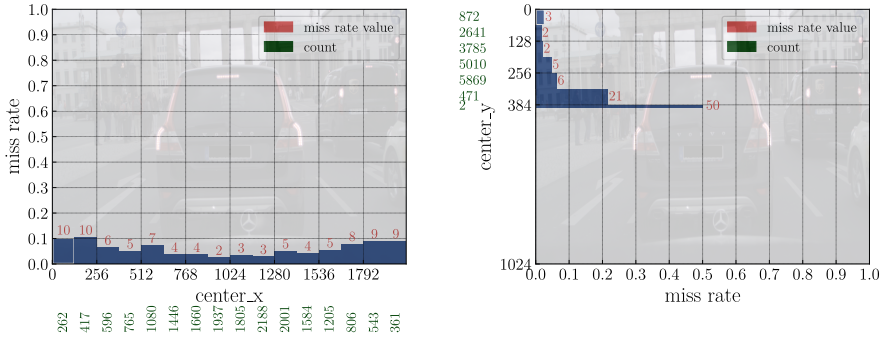


Figure 5.26: Miss rate (DS4) with respect to the traffic light pictogram. Despite the low amount of arrow pictograms, the detection performance for arrow left and straight is close to the performance on circular inlays. Only right arrows show a higher miss rate.



(a) Performance w.r.t x-position (b) Performance w.r.t y-position

Figure 5.27: Miss rate with respect to the ground truth position in the camera image in x- and y-direction. Image borders significantly decrease the detection ability. Both graphs are plotted for miss rate values at FPPI=10^{-1}.

False Positive Investigation: In order to quantitatively analyze the types of false positives, a categorization is performed. Therefore, the matching results with all ground truths is investigated. Categorization is performed into the following error clusters:

- **Multidetection**: This category describes multiple predictions on a single object, which all fulfil the condition IoU $\geq \mathcal{T}_{\text{IoU}}$. The detection with the highest confidence value is treated as a true positive, all others as false positives. This is typically caused by an imperfect NMS.

- **Touch non-matched**: All predictions fulfilling $0.0 < \text{IoU} \leq \mathcal{T}_{\text{IoU}}$, whereas no prediction fulfils IoU $\geq \mathcal{T}_{\text{IoU}}$.

- **Touch matched**: All predictions fulfilling $0.0 < \text{IoU} \leq \mathcal{T}_{\text{IoU}}$, whereas at least one prediction fulfils IoU $\geq \mathcal{T}_{\text{IoU}}$.

- **Touch ignores**: All predictions fulfilling $0.0 < \text{IoU} \leq \mathcal{T}_{\text{IoU}}$ on an ignore region.

- **Background**: All predictions fulfilling IoU $= 0.0$. This can either be "real" false positives or caused by not labelled traffic lights.

Figure 5.28 illustrates the percentage portion of false positives with respect to the

introduced categories for all operating points ($\mathcal{T}_{\text{IoU}} = 0.3$). Less surprising is the observation, that the percentage amount of background samples increases for high FPPI values. High FPPI values correspond to low confidences thresholds causing an increased number of unconfident predictions. Low FPPI values (i.e. high confidences thresholds) increase the amount of false positives on true objects, for instance caused by multidetections or non-sufficient IoU values. The percentage amount of multidetections is low, indicating an efficient non-maximum suppression. For higher confidence thresholds, the number of false positives caused by localization inaccuracies (not sufficient IoU due to misalignment) is increased.

Appendix C.1 presents a more detailed view on the sources of false positives, i.e. an automatic procedure to determine object classes, on which false positives occur by using a state of the art scene segmentation approach.

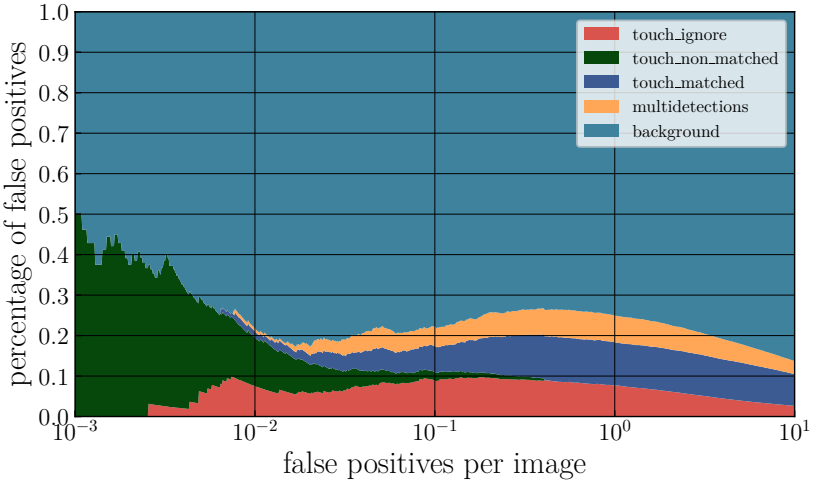


Figure 5.28: Identification of false positives types. The amount of "real" false positives (no overlap with one ground truth) increases for high FPPI values. False positives touching non-matched ground truths increase for low FPPI values (high confidence values), whereas false positives touching matched or ignored ground truths as well as multidetections mainly increase for high FPPI values.

5.4 Discussion

This chapter presented an extensive adaptation and optimization of the popular SSD approach for traffic light detection. *The basic network itself as well as the design of prior boxes and prediction feature layer selection turned out to be the key entry points in order to boost the detection performance.* The inception networks seem to have a high ability to combine local and global receptive fields and are able to hold local information even in deep network layers, which turned out be very beneficial for traffic light detection.

The main dilemma of one or two-stage detectors using reference boxes is a poor detection ability for small objects. Because small object detection is essential for traffic light detection, novel methods to parametrize prior box dimensions and locations have been presented. During evaluation, *the proposed methods proved its expected effectiveness by boosting the detection performance on small objects. The deeper the network itself, the more effective this method is - allowing a detection of small objects even in very deep networks.* Other researchers often tackle the small object problem by increasing the image input size, which dramatically increases the runtime, but does not address the systematic limitations. Instead, *the proposed method could support researchers in other object detection applications to reliably detect small objects without performing image upsampling.*

Because DTLD provides labeled attributes, a more extensive evaluation of the detection performance with respect to different classes is feasible. The presented evaluations show, which sub-classes are easier to detect and which classes are challenging to detect. *Traffic lights with less than three aspects, as well as traffic lights with the state off turned out to be the most challenging.* On the one hand, the mentioned classes are naturally harder to detect. On the other hand, the mentioned classes are highly underrepresented in the current dataset. *With regard to an enhancement of the presented system, it consequently requires not just more data, but a more balanced dataset. This requires a very sophisticated proceed of data recording with expert knowledge and a preferential recording of rare traffic light types.*

A heavily underestimated topic when evaluating neural network approaches is the test set quality. The results showed, that the detection system is able to reach high recall values even for FPPI=10^{-2}, i.e. one false positive every hundred images. In order to reliably evaluate operating points in this range, the test set quality has to exceed those error rates. For instance, in order to evaluate operating points smaller than FPPI=10^{-2}, clearly less than one label error per one hundred images is required. Experience has shown, that label qualities in similar dimensions can be assumed to be very ambitious and only be reached by redundant and extensive labeling methodologies. *To be realistic, this thesis does not assume DTLD to show*

the required labeling quality to evaluate FPPI=10^{-2}. This is supported by visual inspection of the evaluated false positives - which turn out to be true traffic lights overlooked during the labeling process. *Consequently, continuous progress in system performance requires parallel labeling optimization.* Chapter 8 discusses this topic in detail and presents a methodology to automatically use the presented detector for semi-automated dataset improvements.

Chapter 6

Extending Single Shot Detection for State and Pictogram Recognition

While early traffic light detection methods based on color segmentation or thresholding implicitly determine the traffic light state, modern object detection methods solely output the location of the object without state information. Because a pure traffic light detection is not sufficient, the presented Single Shot Detection approach has to be extended in order to predict state confidences.
Besides the traffic light state, the traffic light pictogram is another important attribute. Pictograms express the type of road user (e.g. pedestrian, vehicles, tram, bus, cyclist, ...) as well as directions (e.g. left, right, straight or combined directions) to which they refer.

Both attributes share one similar challenge with regard to system development, namely data imbalance, see Figure 6.1. It illustrates the state and pictogram proportions for each class, which is aimed to be predicted in this chapter. Green and red traffic light significantly exceed the number of yellow, red_yellow and off traffic lights, while circular and pedestrian traffic lights amount almost 80 percent of all pictograms. *This imbalance will be one major challenge addressed in this chapter.* Please note, that this figure solely shows classes aimed to be predicted. Other existing pictograms being very rare, such as arrow_straight_left, arrow_straight_right, u-turn, are neglected due to an insufficient amount of samples. Although DTLD is the largest research dataset available, there still exist classes which occur extremely seldom, so that the absolute number is not sufficient to develop a classification system.

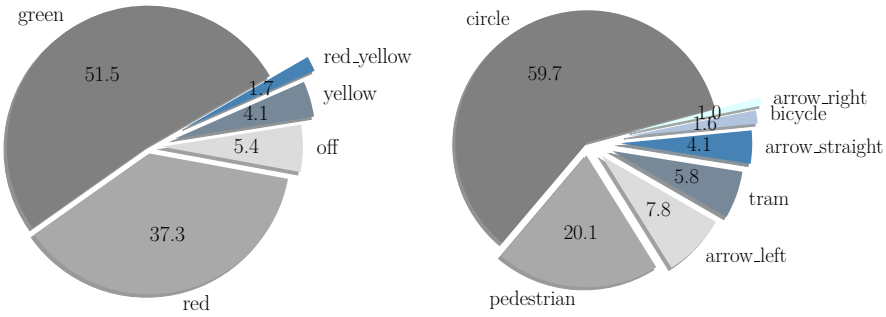


Figure 6.1: Data imbalance is one key challenge for state and pictogram classification. With regard to state classification, traffic lights with state red_yellow, yellow and off are highly underrepresented in comparison with red and green. In view of pictogram classification, arrow_right as well as straight and left arrows and tram/bicycle traffic lights are underrepresented.

Within this chapter, two core methods for state and pictogram classification are presented. The first is an extension of the SSD approach of the previous chapter. The SSD head is extended by additional layers predicting confidence values for each task. Therefore, the training process as well as the loss function is adapted. Similar to the previous chapter, various adaptations and investigations are performed, which optimize the performance step by step. The second method is based on a different principle. It performs a classification within a new, *light-weight network and operates on the RoI obtained from the preceding detection network.* Background of this methodology is the assumption, that both, state and pictogram classification, only require local, pixel-level features. *Very deep network commonly used for detection generate large receptive fields with context information, which may not be required for attribute classification.* Because both methods turn out be to beneficial for some classes, a combined approach is presented as well. This method combines the best of both worlds and helps to obtain an improved overall performance.

The chapter is built up as follows: Section 6.1 outlines the requirements and gives a brief introduction into both tasks. Sections 6.2 and 6.3 present both mentioned methods and give insights into design and implementation details. Inspired by the previous chapter, extensive investigations to improve the classification performance are part of Section 6.4.

6.1 Requirements and Goals

Analogous to the requirements and goals of the detection task described in Section 5.1, the traffic light state is aimed to be correctly recognized as of 8 pixels object width. The impact of resolution on the recognizability of the state is expected to be small - in contrast to the traffic light pictogram. A visual inspection of the labeled data shows, that *pictograms below 10 pixels are hardly recognizable by humans.* On the one hand, the theoretical recognizability is dependent on the pictogram and its spatial frequency itself. On the other hand, various factors such as viewpoint, weather, motion blur or simply by chance (LED flickering) can limited visibility. Figure 6.2 gives a first impression on the recognizability of pictograms with respect to the object width in pixels. *Within this thesis, a resolution of 12 pixels is assumed as the minimum width as of which pictograms are aimed to be detected.* From a human perspective, this seems to be a realistic and useful choice. However, rising resolution does not necessarily simplify classification as occlusions and distortions increase. Especially the sun visors occlude pictograms with rising lateral angle between vehicle and traffic light.

Ignore regions: Besides the resolution requirements, a definition of ignore regions for both tasks is useful from the perspective of application. Regarding the *traffic light state, the developed system is optimized on non-occluded vehicle traffic lights*, i.e. non-vehicle traffic lights are ignored during evaluation. *Pictogram classification is evaluated on non-occluded frontal traffic lights.* All back and side-facing traffic lights responsible for crossing traffic are ignored, because those do not provide relevant information for any possible direction. Moreover, even for side-facing traffic lights pictogram/state information is mostly not visible.

Performance: General statements are hard to make when considering the requirements on classification accuracy or acceptable misclassifications. However, because the traffic light state is a dynamic attribute, the required accuracy is assumed to be much higher than for pictograms. Pictograms are static and thus erroneous classifications are much easier to compensate by tracking/majority voting. Moreover, it is worth mentioning, that the criticality of classification errors even differs depending on the true class. For instance, the transition states yellow and red-yellow are less critical than a correct classification of red. *Red is the most critical state*, because a wrong classification most likely leads to critical situations. Green requires no action, while red-yellow/yellow only support an early breaking or acceleration. A correct classification of non-vehicle pictograms, such as pedestrian, tram or bicycle traffic lights, is required to determine traffic lights actually being relevant for the vehicle within the set of entirely detected traffic lights. Out of this reason, confusions between non-vehicle pictograms are assumed to have no practical relevance, as long as not being confused with vehicle lights and vice versa.

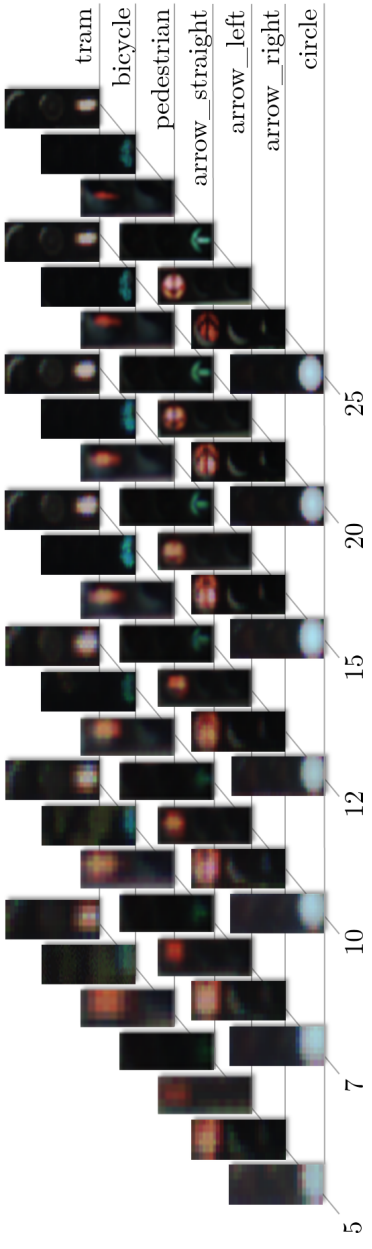


Figure 6.2: Examples of traffic light pictograms at different resolutions. Most samples are not differentiable below a width of 10 pixels. From a human perspective, a recognition as of 12 pixels should be possible. However, recognizability is not automatically simplified with rising resolution. For larger objects, occlusions by sun visors increase (see pedestrian at 20/25 pixels) and aggravate classification.

6.2 One-Stage Detection and Attribute Classification

The first introduced methodology for state and pictogram classification is a logical extension of the SSD head. One additional convolutional layer per attribute predicts the respective confidences, see Figure 6.3. One potential alternative solution could be an extension of the existing two class problem (background vs. traffic light) into a multiclass problem (red vs. green vs. yellow vs. red_yellow vs. off vs. background). However, this methodology turned out to decrease the detection performance, because in case of an unsure state, the confidences are shared between all five foreground classes. Utilizing additional convolutional layers helps to predict attributes physically independent of the detection path. In the following, this methodology is referred to as *implicit classification*, because classification is performed in a one-stage manner. In order to predict a suitable amount of confidences for each prior box, the prediction convolutional layers have to be correctly adjusted. Eight kernels per prior box ($K_{\text{picto}} = 8K_{\text{p}}$) are used to predict seven pictogram confidences and one background confidence, respectively. Six kernels per prior box ($K_{\text{state}} = 6K_{\text{p}}$) are trained to obtain five state confidences and one background confidence, respectively.

Loss function: The loss function used to train the detection network, Equation (5.26), is extended by adding a weighted state and pictogram loss, leading to

$$\mathcal{L}_{total} = \frac{1}{N_{p,m}}(\mathcal{L}_{\text{conf}} + \alpha \cdot \mathcal{L}_{\text{loc}} + \beta \cdot \mathcal{L}_{\text{state}} + \gamma \cdot \mathcal{L}_{\text{picto}}). \tag{6.1}$$

The weighting factors $\alpha = \beta = \gamma = 1$ are not optimized within this thesis and parametrized to guarantee an equal weight for each task. Assuming the attribute prediction layers to output logits $\underline{z}^{\text{attr}} = [z^{\text{C0}}, z^{\text{C1}}, ..., z^{\text{CN}}]^\top$, where $\sum z^i \neq 1$. Analogous to the confidence loss, the attribute loss is a cross-entropy loss with softmax, generating an attribute confidence vector $\underline{c}^{\text{attr}} = [c^{\text{C0}}, c^{\text{C1}}, ..., c^{\text{CN}}]^\top$, where

$$c^i = \frac{\exp(z^i)}{\sum_m \exp(z^m)}, \qquad \sum_i c^i = 1. \tag{6.2}$$

During the inference step, the state/pictogram of the n-th prediction is then determined by an argmax operation

$$y_n = \arg\max \underline{c}^{\text{attr}}, \tag{6.3}$$

i.e. by selecting the class yielding the maximum confidence value.

Ignore Regions: Because the existing SSD framework does not support ignore regions, an attribute-wise ignore regions handling is implemented. This enables to parametrize different ignore regions being filtered, as well as size filters, which are applied during training. Ignore regions handling is implemented for detection, state and pictogram classification, respectively. Non-vehicle and occluded traffic lights are excluded for state loss computation and non-frontal traffic lights are not utilized for pictogram optimization. In terms of size filtering, objects smaller 8 pixels are ignored for state optimization and objects smaller than 12 pixels are ignored for pictogram optimization. This helps to exclude potentially ambiguous labels due to resolution. Filtering is performed on the object width to guarantee aspects-invariance.

Limitations: As shown in Figure 6.3, *SSD has a strict architecture, in which all tasks are set up on one or multiple identical feature maps.* Performing state or pictogram classification in earlier layers than predicting detection confidences is not possible. This is required in order to correctly assign bounding box offsets, object confidences and state and pictogram confidences. Because detection strongly benefits from context information, which is likely not the case for predicting state, finding the optimal feature maps is a compromise between both optima. *According to theory, state classification performance only requires local features from early layers, while optimal detection is assumed to require deeper networks with larger receptive fields.*

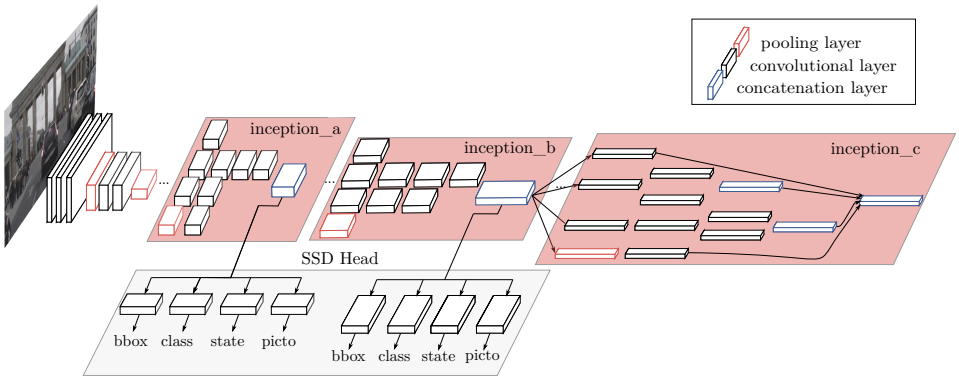


Figure 6.3: SSD width Inception_v3 basis network extended for state and pictogram prediction. Both attributes are physically independent of the bounding box and class confidence prediction. This example shows a prediction on top of two feature layers.

6.3 Two-Stage Attribute Classification

One major disadvantage of implicit classification is the restriction on utilizing identical feature layers for all tasks. This may be beneficial for the detection task, however it shows risks for the state and pictogram recognition task. With regard to state recognition, only information inside the bounding box is required. The risk of implicit classification is, that context information such as traffic light similar color from background pixels is undesirably considered for classification.

With regard to pictogram classification, the network may be able to correctly recognize the correct class by street topologies instead of the pictogram. Humans are mostly able to differentiate between vehicle and pedestrian traffic lights without recognizing the actual inlay at far distances. In order to guarantee no impact of context information on classification, the network input has to be restricted to the pure region of interest. This can be done by using a subsequent, external classifier, which input contains the pure detection box. However, the detection boxes are partly not perfectly aligned and small portions of background pixels can yet be present.

Figure 6.4 shows the methodology of the explicit, two-stage detector for state and pictogram classification. As mentioned, the detection boxes from SSD are cropped out of the entire image and resized to one common size. The optimal RoI size has to be empirically determined and may vary for state and pictogram, respectively. A common input size is required, because the network has a fixed number of input neurons. Resizing is performed by nearest-neighbour interpolation.

The classification network is a light-weight CNN outputting a defined number of confidences for each attribute. The winner class is determined by an argmax operation applied on the output confidence vector. For each attribute, a different network is designed, which may also differ in the number of input neurons (RoI size). More details on the network design, see Section 6.3.1.

6.3.1 Network Design

As mentioned, deep and large networks are not required for state and pictogram recognition. Instead, a small network consisting of several convolutional layers and one pooling layer is utilized. Two fully-connected layers map the feature layer outputs to an output vector containing one probability for each class. Figure 6.5 illustrates the layers of the network, as well as the feature map sizes in detail. In total, five outputs values are calculated (classes off, red, green, yellow, red_yellow) for state classification. Seven output classes are determined for pictogram classifica-

tion (classes circular, arrow_left, arrow_straight, arrow_right, pedestrian, cyclist, bus/tram).

RoI Resizing: There are different strategies in order to determine the optimal size for classification. Selecting a relatively high size may guarantee an optimal performance, but higher runtimes. Selecting small sizes may decrease the performance, but allows a fast computation. For state recognition, high resolutions are not expected to be beneficial. In theory, pictogram recognition requires higher resolutions, because inlays show a higher spatial frequency.
Finding the optimal aspect ratio is another issue. If only using one single classifier for each task, all samples have to be resized to one common aspect ratio. For this thesis, an aspect ratio of 2 is used - in order to be able to classify traffic lights equipped with one, two, three and four light bulbs. Higher aspect ratios lead to strong distortions for traffic lights with only one light bulb. However, using multiple classifiers for different aspect ratios is conceivable.

Background Class Issues: Besides the actual attribute classes, a background class confidence is commonly predicted. False positives of the previous detection approach are used as background training samples. Because most background samples are rear lights of vehicles or contain traffic-light similar colored blobs, the background samples resemble the foreground samples. This aggravates classification. Preliminary experiments showed, that predicting a background confidence downgrades performance on the relevant foreground classes. Consequently, background is not trained as a separate class. The main disadvantage of not predicting a background class is, that false positives are assigned to one state/pictogram class, but can not be rejected by the classification step.

Network Depth: State and pictogram rely on very local features which do not re-

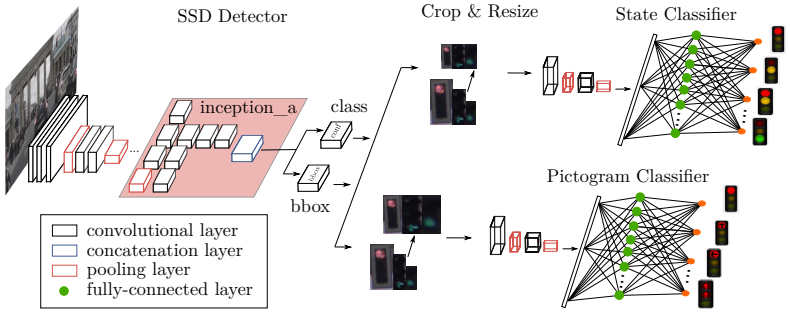


Figure 6.4: State and pictogram classification using additional small CNN networks (grey) operating on the SSD detection bounding boxes.

quire large receptive fields. For this thesis, a network consisting of five convolutional layers and one pooling layer is used, see Figure 6.5.

Filter Kernel Parametrization: While the weights of the feature kernels are learned by the network, the choice of the number of kernels and the kernel size is determined by hand. In the initial configuration, common and fast 3x3 kernels are used, a variation will be performed in the investigations.

Classification network: The final output of convolutions is mapped to output confidences by two fully-connected layers. The first layer can be considered as a linear down-mapping of the output neurons after convolution to the final number of output confidences. Similar to the detection approach, a softmax function with cross-entropy loss is used for training.

6.3.2 Data Collection

As mentioned, SSD prediction boxes are input of the RoI classifier, which may be misaligned with respect to the ground truth box. The training samples are generated from the SSD output in order to learn the classifier to handle potentially misaligned output boxes. A frame-wise matching of ground truths and predictions generates label values for each prediction. For the case of a match between prediction and ground truth ($\mathcal{T}_{\text{IoU}} \geq 0.3$), the prediction is used for training if it fulfils additional criteria, such as exceeding the minimum width and not being an ignore region.

Ignore Regions: Table 6.1 illustrates the mentioned criteria for state and pictogram classification, respectively. State recognition is trained and evaluated on frontal traffic lights responsible for vehicle traffic. The minimum width is aligned to the parametrization for the detection task in Chapter 5. For pictogram recognition, small samples are ignored for training and evaluation. Using samples of small resolutions

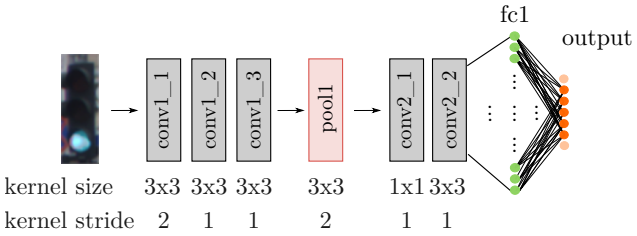


Figure 6.5: Light-weight classification network for local feature extraction. Kernel size and stride is given below each layer.

may decrease the performance on larger samples, because the classification space has to be widened. When visually inspecting the training samples (see Figure 6.2), a classification as of 12 pixels object width seems to be a suitable choice.

Visual Inspection: Figure 6.6 illustrates training samples extracted by the described approach. Noticeable is the generally well alignment of the traffic light predictions, expressing good localization properties of the former SSD approach. This is particularly important for a subsequent classification, because background pixels are suppressed. Moreover, the active light and pictogram is likely not cut-off, which is one major requirement for a successful classification.
When observing the state examples, a strong variation of the active bulb intensity can be recognized. Some light bulbs are hardly visible showing an appearance similar to off lights. Off light examples lack of strong features and look mostly like dark regions. Because black regions are shared across all state classes, the state off will be the most challenging to be recognized from a human perspective.
The pictogram examples shown in Figure 6.6 illustrate an even higher complexity. On the one hand, inlays have a higher spatial frequency, which is hardly recognizable at low resolutions. On the other hand, features vary in position and form, depending on the actual traffic light state. In case of circular, the state only changes the position of the circular feature. In case of arrows with state red and yellow, the arrow sign is black circumscribed by a colorized circle. For the state green, arrows are active and colorized, compare Figure 1.4 - 1.6. Bus, tram or subway traffic lights show a varying alignment depending on the state. However, most symbols show one common property: the white, active light. Cyclist lights have the largest spectrum of inlay variations. The most common is a bicycle inlay. There also exist white bicycle inlays, combined pedestrian and bicycle inlays or simply a circular inlay, which does not differentiate from a common circular vehicle light. Especially the latter is not differentiable from circular by only considering the region of interest. Context information is required, because the only distinctive feature to vehicle lights is the installation height and the traffic light real world size.

Table 6.1: Minimum width and ignore regions for state and pictogram classification.

Attribute	**Min. width** [pixels]	**Ignore regions**
State (train)	6	back, left, right, occluded, tram
State (test)	8	back, left, right, occluded, tram, pedestrian, bicycle
Pictogram (train)	10	back, left, right, off, occluded
Pictogram (test)	12	back, left, right, off, occluded

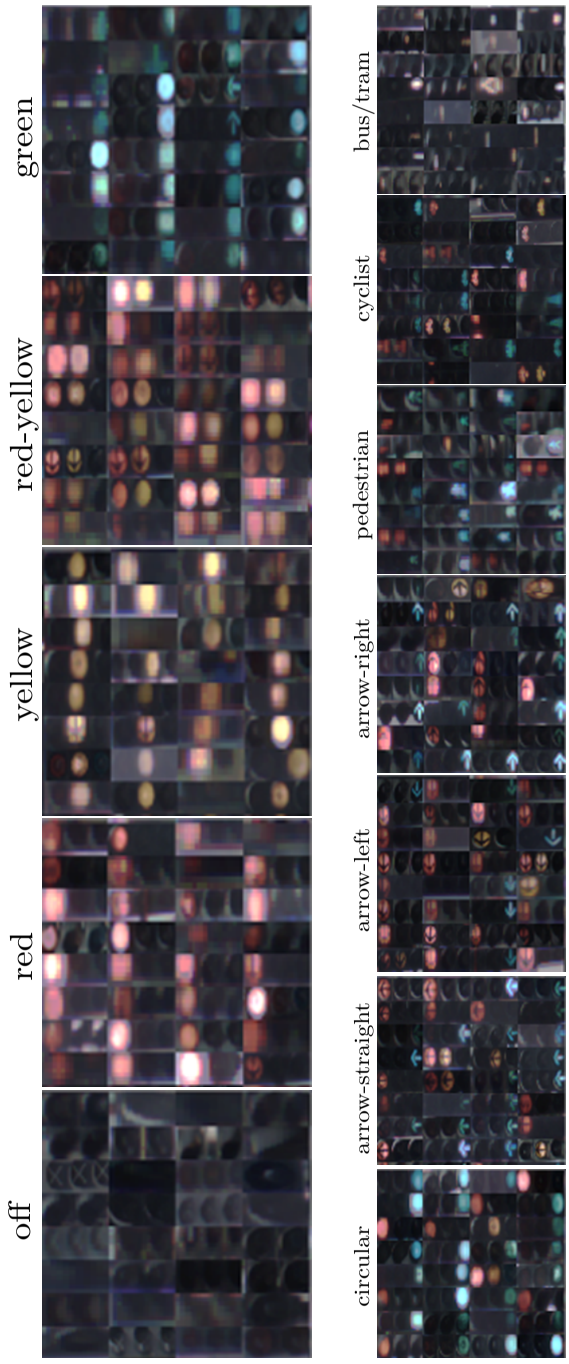


Figure 6.6: Training examples for state pictogram classes extracted from SSD predictions and ground truth information. Please note, that the images are brightened in order to ensure a better visibility.

6.4 Experimental Investigations and Evaluation (One-Stage)

This section presents extensive investigations in order to improve the implicit state and pictogram classification in the Single Shot Detection framework. Initially, multi-task optimizations are presented used to evaluate the introduced multi-task network.

Snapshot Selection: Similar to the detection task, the optimal training snapshot has to be extracted during the optimization of the network. Every 10 000 iterations one snapshot is stored and evaluated. Because the number of tasks increased to three, a rule has to be developed, which helps to extract the optimal snapshot considering all tasks. This can be done by combining three metrics expressing the performance of all three tasks by one single value and finding its maximum or minimum value (depending on the metric). The optimal network iteration $\mathcal{I}$ is selected according to

$$\begin{aligned} \underset{\mathcal{I}}{\operatorname{argmin}} \quad & \overbrace{\alpha\Big(\mathrm{LAMR}\Big)_{\mathcal{I}}}^{\text{detection metric}} + \overbrace{\beta\Big(1-\mathcal{R}_{\mathrm{state}}\Big)_{\mathcal{I}}}^{\text{state metric}} + \overbrace{\gamma\Big(1-\mathcal{R}_{\mathrm{pictogram}}\Big)_{\mathcal{I}}}^{\text{pictogram metric}} \\ \text{s.t.} \quad & 0 \leq \mathcal{I} \leq 130k, \end{aligned} \tag{6.4}$$

i.e. a weighted sum of the detection performance and state/pictogram classification performance. The state and pictogram metric itself is a weighted sum of the macro and micro recall calculated as

$$\mathcal{R} = \eta_{\mathrm{micro}} \cdot \underbrace{\frac{\sum_{i=1}^{N} \mathcal{C}(\omega_i, \hat{\omega}_i)}{\sum_{i=1}^{N}\sum_{j=1}^{N} \mathcal{C}(\omega_i, \hat{\omega}_j)}}_{\text{micro recall } \mathcal{R}^{\mathrm{micro}}} + \eta_{\mathrm{macro}} \cdot \underbrace{\frac{1}{N}\sum_{i=1}^{N} \frac{\mathcal{C}(\omega_i, \hat{\omega}_i)}{\sum_{k=1}^{N} \mathcal{C}(\omega_i, \hat{\omega}_k)}}_{\text{macro recall } \mathcal{R}^{\mathrm{macro}}}, \tag{6.5}$$

where η_{micro} and η_{macro} adjust between micro and macro recall. Please note, that both parameters are parametrized to guarantee $\eta_{\mathrm{micro}} + \eta_{\mathrm{macro}} = 1$. For the following experiments, both measures are equally weighted ($\eta_{\mathrm{micro}} = \eta_{\mathrm{macro}} = 0.5$). Micro and macro recall are calculated as introduced in Equation (4.8) and (4.9). This helps to keep the classification result balanced and avoid overfitting towards the overrepresented classes. LAMR is determined equal to the detection task.

6.4.1 Baseline Experiments

One first crucial investigation is the impact of additional state and pictogram classification on the detection results. Because attribute prediction is designed to be independent from detection, a large negative impact is not expected. However, there may be effects in optimization caused by sharing of the total loss values. The first three experiments compare the impact on the detection performance by adding one or both tasks to the network:

- Impact of state classification (two-task network)
- Impact of pictogram classification (two-task network)
- Impact of state and pictogram classification (three-task network)

Figure 6.7 shows the ROC curve expressing the detection performance for all three experiments compared to the pure detection network. Fortunately, *additional attribute prediction does not significantly influence the detection performance.* This supports the utilized approach, in which the attribute classification is independent of the detection task by using additional convolutional layers.

Figure 6.8 illustrates the state and pictogram classification performance for the single-task and multi-task setup. Single-task describes two different setups, in which one CNN is trained for detection and state recognition and another for detection and pictogram classification. The multi-task network performs all three tasks in one single network. This investigation is expected to show, whether the state recognition is influenced by pictogram classification or vice versa. The overall performance is similar, however *underrepresented classes show a much higher result variation*, see results for red_yellow or off. Rare classes may be neglected when not being depicted in the latest batch updates of the network. This leads to a higher performance variation depending on the current status of the training procedure (number of iterations). *The quality of the recognition is less unstable for classes with a high amount of training samples.* However, the result differences may not be caused by the multi-task setup, but by general fluctuations.

Further Potential for Optimization: With regard to the state confusion matrix, a clear correlation between the amount of training samples and the classification recall can be observed. Classes well-covered in DTLD (green, red) are correctly classified in almost all cases, whereas especially red_yellow and off traffic lights show room for improvement. The state *off* is the most challenging - because of a lack of data and moreover, because it is the visually most challenging class. The absence of an active light, the most dominant traffic light feature, leads to an aggravated

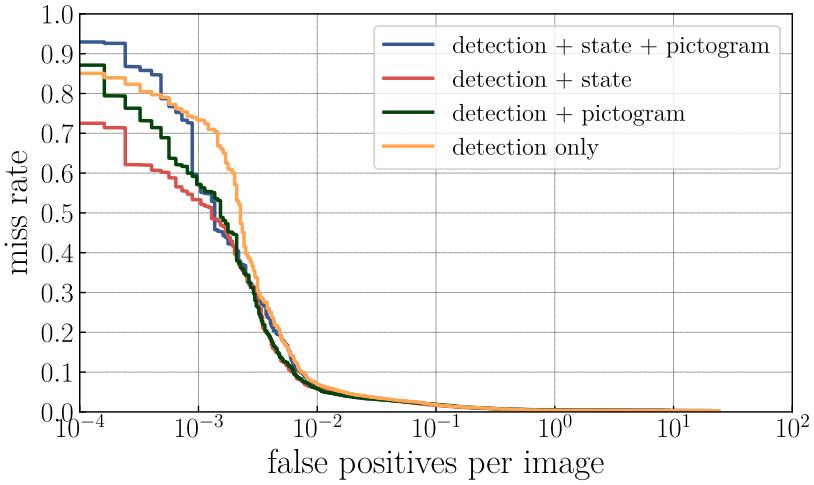


Figure 6.7: Extending Single Shot Detection for state and pictogram classification does not significantly influence the detection performance. Curves are virtually equal in the FPPI range $[10^{-2}, 10^{2}]$. This supports the selected method, in which state and pictogram prediction layers are physically independent of the detection path.

classification.

The findings of the impact of the sample amount on the performance can also be observed for pictogram recognition. The main challenge is the pictogram arrow-right which has the lowest amount of training samples. In general, distinguishing the arrow pictograms seems to be one major challenge. The recall on pedestrian and tram traffic lights is high and suggests, that *context information improves the performance*. Bicycle traffic lights are frequently interchanged with circle due to the already mentioned fact, that *many bicycle traffic lights have a circular instead of bicycle inlay*. In consequence, the mentioned misclassifications can rather be assessed as a limitation of the dataset. In this respect, an optimization of the performance on bicycle lights is not a focus of subsequent optimizations. The focus lies on improving the performance of arrow_right and the other direction signs by artificially increasing the number of training samples.

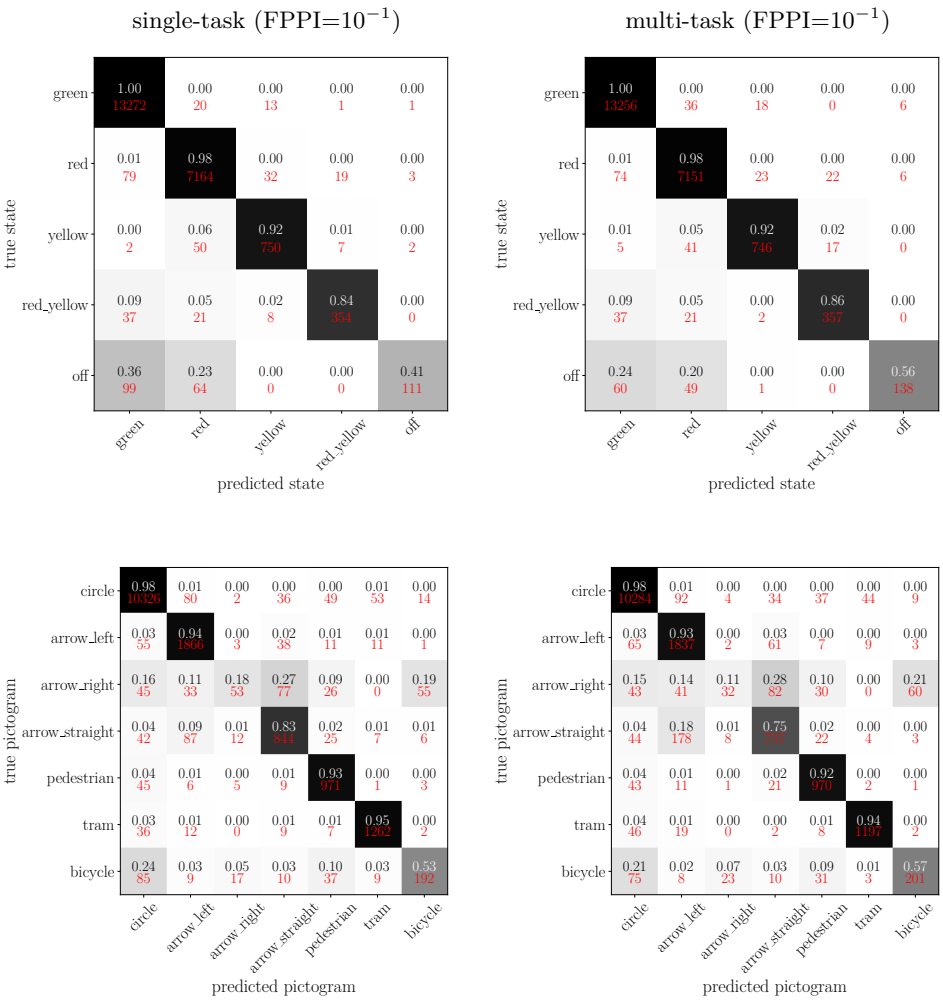


Figure 6.8: State and pictogram classification performance trained as single-task or multi-task network. Percentage values are shown in white/black, absolute numbers of samples are shown in red. The overall performance is similar. However, results for underrepresented classes shows a higher variation. The classes off, and arrow pictograms show room for improvements. A clear correlation between training sample amount and performance can be observed.

6.4.2 Impact of Ignore Regions

As mentioned, ignore regions for each attributes can be set during training. This section investigates the impact of different choices for the minimum width and ignored regions on the performance of both tasks.

Size filters: This filter is assumed to be particularly relevant for the pictogram classification task. The main question is, whether utilizing hardly recognizable training samples is useful. In DTLD, pictograms are correctly labeled - also in the case the pictogram is not recognizable in the current frame. Because pictograms are static, a derivation from the entire sequence is possible. This means, human labelers reversely annotate the sequences, ensuring correct label attributes even for far objects not showing a sufficient resolution. However, no annotation tag expresses, whether the pictogram is actually recognizable in the current frame. This does not offer the possibility to exclude indistinguishable samples.

Class filters: Only using vehicle lights significantly improves the performance for the status. Among frontal traffic lights, pedestrian/cyclist lights frequently have unusual light combinations, which downgrade the performance on vehicle lights.

Table 6.2 illustrates, that size filters only show a minor impact. More significant is the improvement by excluding non-vehicle traffic lights for state optimization. Pictogram classification is optimal when using labels larger equal 12 pixels.

Table 6.2: Impact of size filters and ignore regions on attribute classification.

	minimum width						**classes**		**results**	
	4px	6px	8px	10px	12px	14px	FL[1]	VL[2]	$\mathcal{R}^{\text{macro}}$	$\mathcal{R}^{\text{micro}}$
STATE	✓						✓		0.980	0.862
		✓					✓		0.980	0.853
			✓				✓		0.981	0.856
				✓			✓		0.981	0.849
					✓			✓	0.978	0.795
			✓					✓	**0.983**	**0.878**
PICTOGRAM		✓					✓		0.944	0.733
			✓				✓		0.944	0.732
				✓			✓		0.941	0.733
					✓		✓		**0.941**	**0.739**
						✓	✓		0.393	0.728

[1] only frontal traffic lights (exclude back, left, right)
[2] only vehicle traffic lights (exclude pedestrian, tram, bicycle)

6.4.3 Data Augmentation

As mentioned, one major challenge is class imbalance. In the present case, this particularly refers to the classes *red_yellow* and *off* for state prediction and the classes *arrow_right/straight* for pictogram prediction. This section presents data augmentation developed to tackle this issue. The presented modes particularly refer to an optimization of pictogram classification.

Random flip: *Random vertical flipping is a promising approach to improve pictogram classification by an increase of the number of left and right arrow pictograms.* One side effect of image flipping is the transformation of right-hand traffic into left-hand traffic, see Figure 6.9. It has to be empirically evaluated, whether this influences the detection performance. All bounding box coordinates are transformed as well. Besides the pictogram labels, flipping also affects the direction labels of the traffic light (left/right facing). Other effects are not expected. However, because the bicycle and pedestrian pictograms are non-symmetric, the effect on those classes has to be monitored. Table 6.3 shows, that random flipping lowers the macro recall, but increases the micro recall on pictograms. A closer look on the subclass performance explains this property. *Although flipping clearly improves the classification performance on right arrows, it lowers the recall on left arrows (less samples than without flipping).* Because the test set contains more left arrows, the overall performance is worse, whereas the average class-wise performance (independent of the number of objects per class) improves. Thus, this thesis proposes to use the subsequently introduced *flip duplication.*

Flip duplication: Similar to previous flipping, the images are vertically flipped. In contrast to the previous image augmentation mode, both images, flipped and non-flipped, are used for training. This helps to generate as much labels as possible for both arrow directions, arrow_left and arrow_right. Image shuffling guarantees, that both image files are very likely not subsequently called during the training process. By using this proceed, the number of training images is doubled.

Table 6.3 shows, that *flip duplication is able to improve both measures, macro and micro recall. It improves the performance on right arrows, but keeps the already high recall on left arrows.* The overall performance on pictograms is significantly increased, whereas state classification performance does not seem to be affected by flipping.

Random crop: Figure 4.7 shows, that DTLD, similar to almost all traffic light recognition datasets, contains a disproportionate amount of small objects. Especially large object bins are sparsely occupied. This can lead to problems during training, because the kernels used for attribute predictions are coupled to the prior box sizes.

More detailed, kernels responsible for predicting the attributes of large objects are optimized on a clearly lower number of examples. This can be problematic for rare object classes, which consequently suffer on sufficient samples at high object resolutions. Random cropping is an image augmentation mode, which crops a region of the entire input image and resizes it to the original input size. This helps to increase the number of large objects. Unfortunately, Table 6.3 shows that random cropping has a negative impact on the performance. For further experiments, flip duplication without random cropping is used.



Figure 6.9: Image cropping (top), image flipping (bottom) applied on the original image (centre). Flipping helps to compensate the lack of arrow_right pictograms, which naturally occurs when recording on German streets. Both, original and flipped images are used for training.

Table 6.3: Impact of various data augmentation methods on state and pictogram classification performance.

MODE	**all previous modes**	✓	✓	✓	✓
	random flip		✓		
	flip duplication			✓	✓
	random cropping				✓
RESULTS	LAMR (DS1)	0.062	0.061	**0.059**	0.062
	$\mathcal{R}^{\text{macro}}_{\text{state}}$	0.980	**0.982**	0.976	0.971
	$\mathcal{R}^{\text{micro}}_{\text{state}}$	0.847	0.845	**0.847**	0.839
	$\mathcal{R}^{\text{macro}}_{\text{pictogram}}$	0.937	0.928	**0.943**	0.933
	$\mathcal{R}^{\text{micro}}_{\text{pictogram}}$	0.689	0.727	**0.762**	0.701

6.4.4 Impact of Network Depth and Prior Box Stride

Similar to the evaluation of the detection task, an investigation of the prior box locations and the choice of the prediction layer(s) is necessary. Initial presumption is, that deep layers may not be beneficial, because state and pictogram classification do not require context information. An additional issue examined in this section is the impact of the adaptive prior boxes on the attribute classification performance. As already mentioned, for each additional prior box a higher number of network weights has to be optimized. This requires sufficient training samples. Especially for rare classes, this number can be too small leading to a worse performance - despite more prior boxes.

Table 6.4 illustrates the performance of the traffic light detection as well as state and pictogram prediction with respect to the prediction layer(s). The comparison between the performance values with original/adaptive prior boxes shows: *More prior boxes improve detection performance, but decrease attribute classification performance.* There are especially two networks, which perform best. Those are marked with bold letters. The first one is a single prediction layer network, showing the best attribute classification results. The second network is a multi-prediction layer network, which has a slightly lower attribute classification performance, but a higher detection performance (DS1/DS4). The decision is towards the latter, particularly due to the assumption, that pictogram classification can be improved by the introduced two-stage classification. The optimal system may be a combination of SSD and external classification. With regard to the former, the main priority lies on detection and state classification performance.

Table 6.4: Detailed results for single-layer and multi-layer prediction. Multi-layer prediction shows an improvement when using the original prior box locations, which vanishes when using the proposed, adaptive prior box locations. In total, 28 networks are trained and evaluated in order to obtain all results.

	prediction feature layers									**results**											
										original stride						**adaptive stride**					
	a1	a2	a3	b1	b2	b3	b4	c1	c2	DS1	DS4	$\mathcal{R}^{\text{macro}}_{\text{state}}$	$\mathcal{R}^{\text{micro}}_{\text{state}}$	$\mathcal{R}^{\text{macro}}_{\text{picto}}$	$\mathcal{R}^{\text{micro}}_{\text{picto}}$	DS1	DS4	$\mathcal{R}^{\text{macro}}_{\text{state}}$	$\mathcal{R}^{\text{micro}}_{\text{state}}$	$\mathcal{R}^{\text{macro}}_{\text{picto}}$	$\mathcal{R}^{\text{micro}}_{\text{picto}}$
Single-Layer	✓									0.09	0.03	0.966	0.773	0.936	0.781	0.073	0.022	0.968	0.777	0.928	0.734
		✓								0.076	0.023	0.972	0.817	0.938	0.770	0.061	0.017	0.973	0.823	0.943	0.821
			✓							0.071	0.019	0.975	0.862	0.940	0.795	0.057	0.015	0.976	0.847	0.943	0.762
				✓						**0.089**	**0.032**	**0.981**	**0.910**	**0.959**	**0.869**	**0.056**	**0.017**	**0.980**	**0.900**	**0.951**	**0.822**
					✓					0.087	0.031	0.981	0.903	0.948	0.819	0.055	0.015	0.977	0.875	0.950	0.821
						✓				0.085	0.031	0.978	0.893	0.945	0.810	0.059	0.017	0.976	0.876	0.942	0.790
							✓			0.89	0.033	0.980	0.907	0.945	0.779	0.058	0.016	0.975	0.874	0.938	0.760
								✓		0.129	0.069	0.975	0.859	0.927	0.723	0.071	0.018	0.964	0.760	0.915	0.693
									✓	0.128	0.069	0.968	0.796	0.913	0.654	0.071	0.020	0.964	0.751	0.909	0.680
Multi-Layer			✓				✓	✓		0.063	0.018	0.983	0.929	0.934	0.742	0.054	0.014	0.980	0.910	0.924	0.706
	✓		✓							0.076	0.021	0.972	0.800	0.936	0.784	0.071	0.020	0.969	0.771	0.941	0.775
	✓	✓	✓							0.076	0.021	0.973	0.821	0.944	0.799	0.066	0.018	0.971	0.799	0.933	0.711
			✓				✓			0.056	0.014	0.979	0.899	0.940	0.768	0.068	0.019	0.980	0.915	0.932	0.769
			✓	✓			✓			**0.065**	**0.019**	**0.982**	**0.920**	**0.945**	**0.799**	**0.059**	**0.015**	**0.980**	**0.910**	**0.934**	**0.751**

6.4.5 Detailed Results

This section presents detailed results of the best performing networks determined in the previous sections. Results are divided into quantitative, i.e. scalar values, as well as qualitative results in the form of visual inspection.

Quantitative Results

Figure 6.10a shows the state classification performance of the best performing network at FPPI=10^{-1}. Compared to the baseline experiments, a performance improvement for the class red_yellow is achieved. The main gain is the performance boost for *off* by almost 40 percent. Less surprising are several confusions between the classes red, yellow and red_yellow. More surprising is the high number of red_yellow traffic lights classified as green. However, this turns out to be caused by labeling errors, which is proved in the following subsection.

Figure 6.10b shows the pictogram classification performance of the best performing network at FPPI=10^{-1}. The results reveal a mixed picture: The performance on circle, arrow_left, pedestrian and tram is thoroughly satisfying, while especially arrow_right and bicycle traffic lights seem to be more challenging. By taking a closer look, this is not further surprising, because many bicycle traffic lights do not actually have a bicycle but circular inlay. Those cases are solely differentiable by the installation position - which is mostly below the vehicle lights. In consequence, a classification of bicycle lights using DTLD is actually limited by the database. A pure pictogram label does not seem to be sufficient - a differentiation between the target road user (which road user does the traffic light refer to?) and the actual inlay could be more effective and support development. Despite the utilized flip duplication augmentation mode guaranteeing an equal amount of training samples for arrow right and arrow left, the performance on right arrows is significantly decreased. In DTLD, the vehicle mostly drives straight, i.e. the vehicle is rarely driving on turning lanes. Traffic lights are mostly rotated to guarantee optimal visibility for vehicles driving on the respective lane. Frequently driving straight increases occlusions by sun visors for traffic lights with arrow pictograms. This makes arrow pictograms more challenging to classify in close range due to adverse viewing angles.

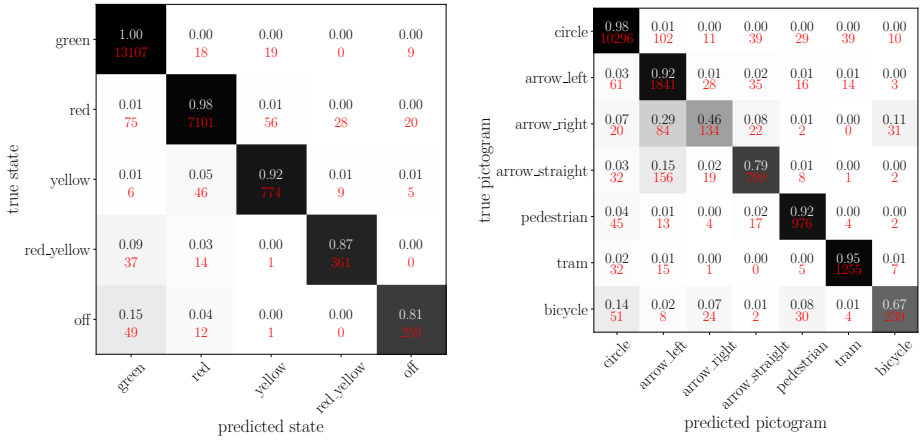
In general, many confusion between the different arrow pictograms can be observed. They share at least one common feature - arrow tip or arrow trunk. Moreover, arrows are often classified as circle. This error pattern is explained in the following section.

Pedestrian and tram traffic lights reach surprisingly high recall values. Although in particular pedestrian inlays with a high spatial frequency are not supposed to benefit from deep CNNs, the performance is high. This supports the presumption, that the network is able to identify pedestrian and tram traffic lights from context features such as pedestrian crossings and tram rails. From a human perspective, most pedestrian or tram/bus traffic lights are classifiable only by context information without considering the inlay shape.

Particularly interesting is the following comparison with the two-stage classifier, which cannot use context information and is restricted to the pure traffic light crop input. It clarifies, whether context information even supports pictogram classification and the optimal system may be a combination of both methodologies.

Qualitative Results - Closer Look on Errors

Figure 6.11 shows correctly and falsely classified samples in the form of a confusion matrix. This helps to get a better understanding of the error cases. Surprisingly, the



(a) state classification performance (b) pictogram classification performance

Figure 6.10: Attribute classification results of the best network determined by the investigations of Table 6.4. With regard to state recognition, the status off significantly improved. With regard to pictogram classification, a still open problem are arrow inlays. Values for both graphs are determined at FPPI=10^{-1}.

error plot for the attribute state shows, that the majority of apparent classification errors are actual labeling errors. Because the labeling tool supports copying labels from one frame into another, one initial labeling error is frequently present in the entire sequence.

Excluding labeling errors, green and red classification hardly shows any errors. In case of red, yellow and red yellow there are some reasonable error clusters, for instance yellow lights with two light bulbs and an upper active bulb. Because almost all red and three-bulb traffic lights show an active upper red light, *the classification network overweights the pure light position instead of the actual color.*

A second error cluster are traffic lights with state off. The performance on off lights seems to be a compromise between correctly classifying off and correctly classifying actual red/green/yellow lights showing a very weak brightness. In many situations, the borders between off and active can be very weak. In consequence, *off is the most challenging state*, which is moreover complicated by the small amount of training data.

Figure 6.12 shows correct and erroneous pictogram classification results. Please note, that for the main diagonal (correct classifications) only a portion of the entire sample set is visualized due to the lack of space. For non-diagonal samples (wrong classifications), all respective samples on the entire test set are shown. For circular traffic lights almost all evaluated misclassifications are caused by labeling errors. The majority of labeling errors have the state red, in which the arrows are often hardly visible. The evaluated recall on circle amounting 98% can be assumed to increase to almost 100% neglecting labeling errors.

Just as for circle, the majority of misclassifications on the arrow pictograms have the state red, even though the majority of DTLD samples are green. *Arrow classifications seem to be strongly affected by the state as it impacts the appearance.* The red, active circle often outshines the black arrow. In case of green, the active arrow on a black background shows a much better visibility.

With regard to pedestrian lights, only a small number of misclassifications can be observed. Pedestrian pictograms tend to show an appearance close to arrow straight (only if red) and circular when being captured at lower resolutions. Highly resolved and non-occluded pedestrian samples are rarely misclassified. Tram lights are rarely misclassified, also due to the unique feature, that tram lights are mostly white. A few misclassifications are caused by blurred samples, invisible light bulbs, labeling errors or confusions with arrow signs. Bicycle traffic lights show a lot of misclassifications, which however is put into perspective after visual inspection. Many traffic lights responsible for cyclist traffic actually have a circular pictogram. Those examples are (in the proper sense correctly) classified as circle. The second big

error clusters are classifications as pedestrian, which are mostly special pictograms containing a combined pedestrian and cyclist pictogram. *Both error clusters show room for improvement with regard to dataset labeling and suggest a separation between the actual pictogram and the target road user, i.e. the road user the traffic light is actually referring to.* For instance, a traffic light with circular symbol yet controlling bicycle traffic could be labeled as {target_road_user: bicycle, pictogram: circle} when introducing a target_road_user attribute.

All classes show several examples, in which the state pictogram can not be determined due to an adverse viewpoint. In the case the vehicle is close to the stop line, sun visors frequently occlude the active lights of traffic lights controlling turning lanes. This property worsens with rising lateral distance between traffic light and vehicle. In consequence, close traffic lights are not necessarily easier to be detected.

Further investigations are presented in Appendix C.2.

6.5 Experimental Investigations and Evaluation (Two-Stage)

Similar to the evaluation of Method I, the goal is to optimize the micro and macro recall at the operating point FPPI=10^{-1}. One main advantage during evaluation is the independence of the detection network by using separate classifiers, which simplifies the evaluation. Moreover, each attribute classification task can be considered to be independent, because one network is utilized per attribute. In consequence, the detection performance is not further monitored in this section.

Best snapshot selection: During training of both attribute classification networks, the best snapshot has to be extracted. It is determined by

$$\begin{aligned} \underset{\mathcal{I}}{\operatorname{argmax}} \quad & \left(\alpha \cdot \mathcal{R}^{\text{micro}}_{\text{attribute}} + \beta \cdot \mathcal{R}^{\text{macro}}_{\text{attribute}}\right)_{\mathcal{I}} \qquad \alpha + \beta = 1 \\ \text{s.t.} \quad & 0 \leq \mathcal{I} \leq 20k \end{aligned} \tag{6.6}$$

for both networks, where $\mathcal{I}$ is the number of training iterations. Consequently, micro and macro recall is equally weighted. Every 500 iterations, one snapshot is stored and evaluated. In contrast to the training of the SSD network, the number of iterations is significantly decreased. The batch size in the SSD training is limited to one due to the input image size requiring a high GPU memory. Because the RoI classifier operates on image crops, the batch size can be increased.

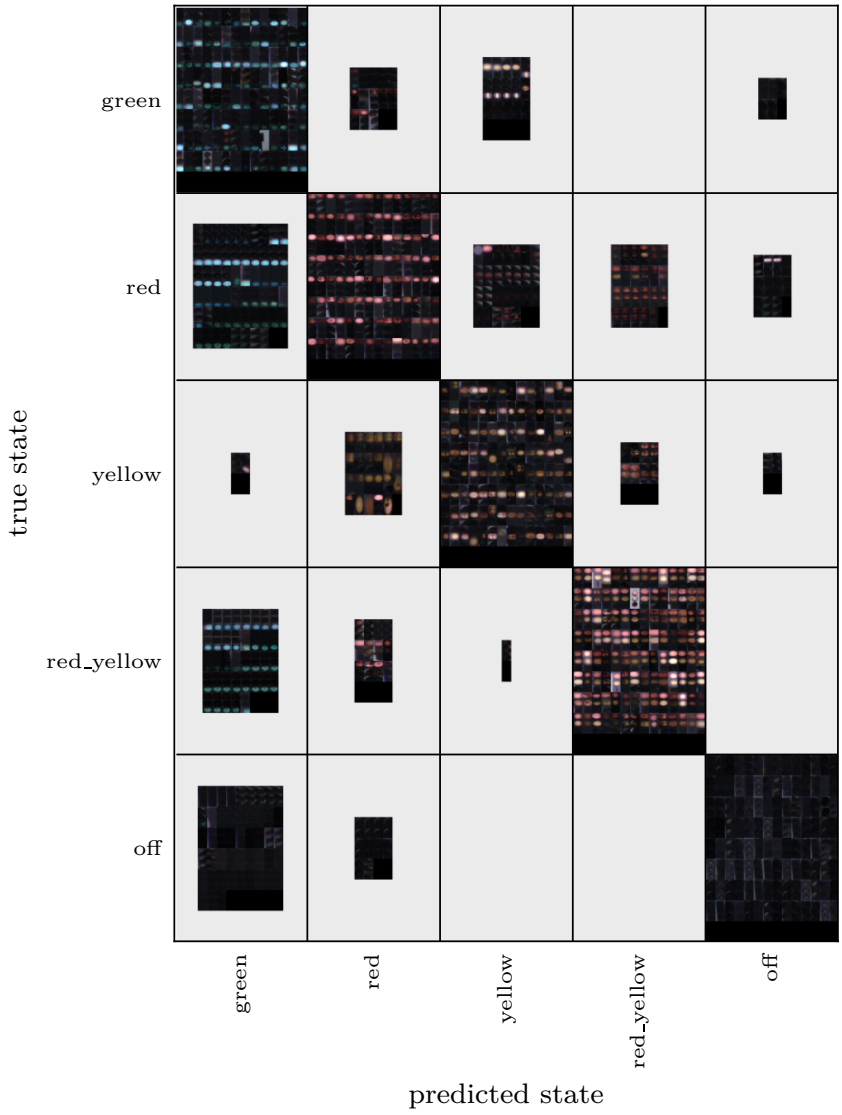


Figure 6.11: Correct and wrong state prediction examples in the form of a confusion matrix. There exist three main error clusters: labeling errors, light bulb position overfitting (see confusions red/yellow) and off-red/off-green confusions.

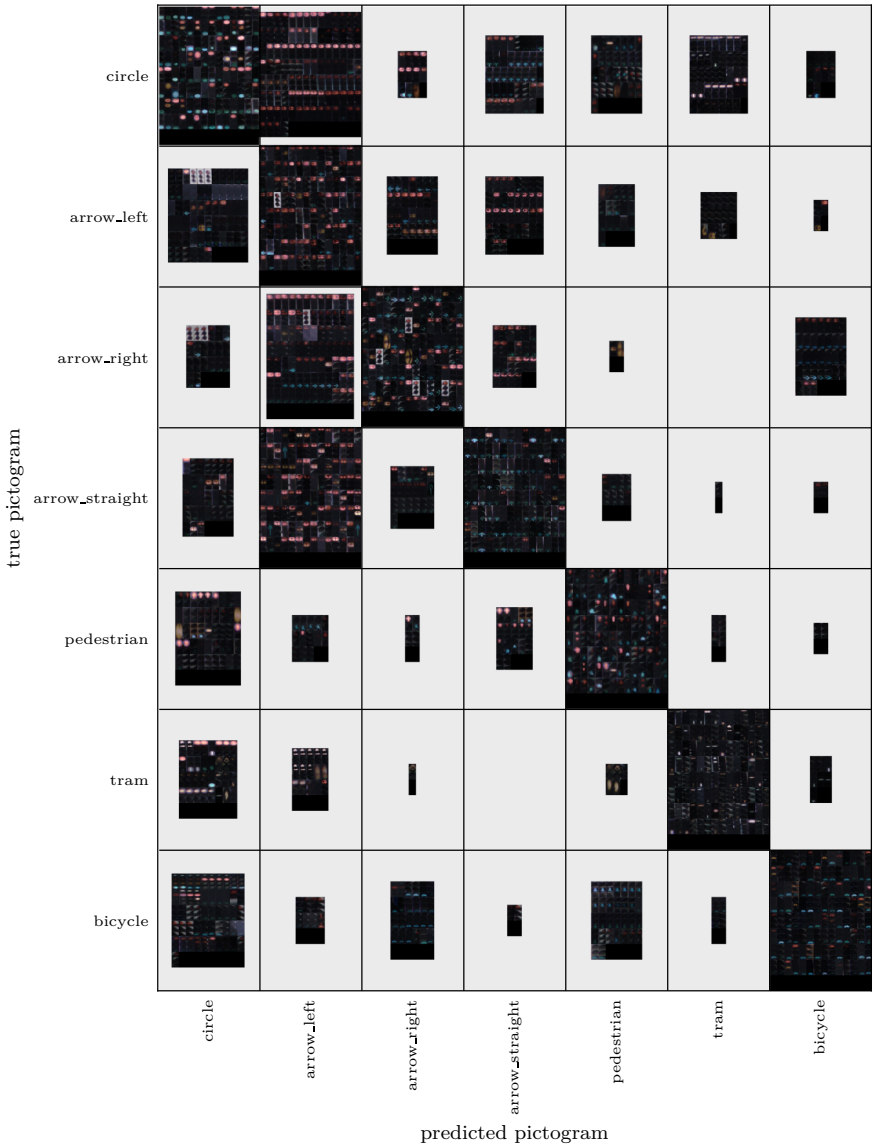


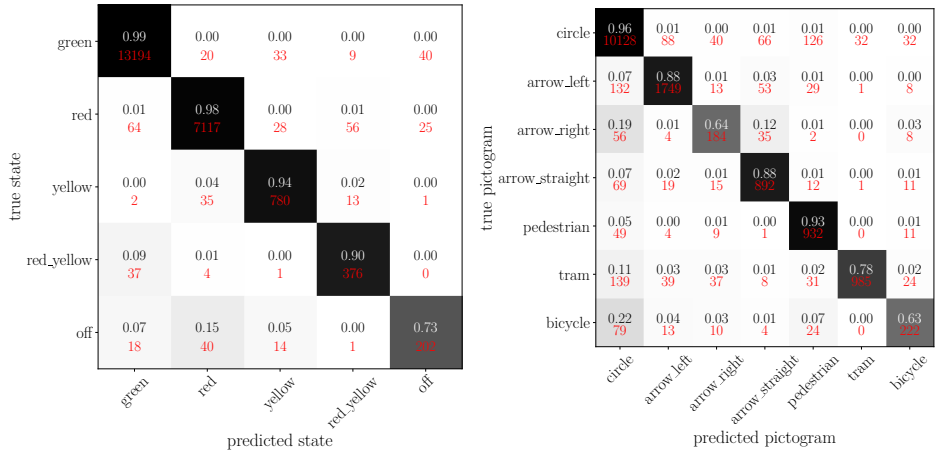
Figure 6.12: Correct and wrong pictogram prediction examples in the form of a confusion matrix. Two error clusters are visible: labeling errors (especially for circle) as well as arrow direction/circle confusions, which mainly occur for red arrow pictogram, which have an active color bulb.

6.5.1 Baseline Experiments

This section gives first insights into the initial, unoptimized results for both attributes. Both initial networks are trained with equal base parameters, in detail:

- Crop size 16×32: From theory, this should be a sufficient resolution for both tasks. However, following evaluations will investigate the optimal input size for both tasks.
- Training on SSD predictions. IoU between prediction and ground truth has to be larger 0.3. Adding bare (perfect) ground truth crops to the training set will be investigated in a later step.
- Training and optimization parameters are chosen according to the optimal hyperparameter configuration of the SSD network, see Section 5.3.7.

Figure 6.13 illustrates the baseline experiment results. With regard to state classification, the initial results are very promising and *already reach the overall performance of the best SSD network performing an implicit classification.* Yellow and red-yellow perform slightly better, whereas green and off are slightly inferior.



(a) Baseline state classification confusion matrix

(b) Baseline pictogram classification confusion matrix

Figure 6.13: Even in the baseline experiment the state classification result already reaches the performance of the best SSD network without performing any optimizations.

6.5.2 Data Input Investigations

Because Method II operates on regions of interests instead of entire scene images, the data input offers a higher degree of freedom for data augmentation. This section focuses on two parts - investigating the optimal RoI size as well as extending the training set in order to improve the overall performance.

Crop Size: The initial crop size is chosen as 16×32 pixels, which may not be optimal. In theory, pictogram classification requires a significantly higher resolution due to higher spatial frequencies. In general, the resolution should be kept as small as possible to avoid unnecessary classification runtime.
A first impression on the expectable appearance of the RoI sizes is illustrated in Figure 6.2. The results for both, state and pictogram classification, with respect to the input RoI size is shown in Table 6.5. Rising resolution seems to improve the attribute classification results with a peak at 16×32 pixels. Runtime also linearly increases with rising RoI size, however the classification time is not significant with respect to the target system runtime, which will be mainly determined by the large-scale SSD network. The optimal RoI size for pictogram classification is located at a significantly higher resolution of 24×48 corresponding to the theoretical assumptions. Moreover, the impact of resolution is much higher than for state classification. Pictogram classification significantly degrades for low input resolutions. All following investigations are performed on RoI sizes of 16×32 for state and 24×48 pixels for pictogram recognition, respectively.

Training Set Extension: The initial training set consists of SSD predictions marked as true positives by ground truth matching. One main advantage of the RoI classifier methodology is the simplified procedure of collecting augmented training data. This includes sample flipping or adding additional label data. The RoI classifier operates on the pure traffic light region, while in SSD data augmentation the impact on non traffic light regions has to be monitored. Three modifications are investigated, which seem to be promising:

- **Flipped predictions**: All matched predictions are flipped and added to the training set. Right arrows are transformed to left arrows and vice versa. In general, this approximately doubles the training set.

- **Labels**: All labels (without ignore regions or smaller samples) are added to the training set. This ensures a representation of false negatives (not covered in the matched predictions) in the training set.

- **Flipped labels**: Similar to the flipped predictions yet applied to labels. This increases the number of right arrows, which turn out to be a major challenge

for the system.

- **Flip selected labels/predictions**: This mode only flips selected classes. This helps to decrease the ratio between over- and underrepresented classes by only flipping the latter.

Table 6.6 compares the results for state and pictogram classification. State classification performance improves when using all modes, but flip only off traffic lights. This helps to tackle data imbalance. The balanced pictogram performance $\mathcal{R}^{\text{macro}}_{\text{pictogram}}$ is increased by 2.2 percent when using all modes, yet flip on the selected classes. A closer look on the class-wise results shows, that this is based on a significant improvement on right arrows by more than 10 percent. However, this investigation also shows a drop in performance for several classes (mainly non-symmetrical) when applying flip augmentation. When applying flip augmentation on a selected subset of classes (arrow-right, arrow-left, bicycle for pictogram, off for state), the best overall results are reached, see last column. This configuration is used for following investigations.

Table 6.5: Classification results for varying RoI input size. Please note, that state and pictogram classification results are independent and one network for each attribute is trained and evaluated, respectively.

RoI size	**10×20**	**12×24**	**14×28**	**16×32**	**18×36**	**20×40**	**22×44**	**24×48**	**26×52**	**28×56**	**30×60**
$\mathcal{R}^{\text{micro}}_{\text{state}}$	0.975	0.974	0.979	**0.980**	0.980	0.982	0.981	0.980	-	-	-
$\mathcal{R}^{\text{macro}}_{\text{state}}$	0.888	0.870	0.897	**0.908**	0.901	0.908	0.894	0.887	-	-	-
runtime [ms]	0.13	0.14	0.16	**0.17**	0.18	0.20	0.22	0.24	-	-	-
$\mathcal{R}^{\text{micro}}_{\text{pictogram}}$	0.870	0.902	0.911	0.918	0.925	0.937	0.939	**0.939**	0.939	0.940	0.943
$\mathcal{R}^{\text{macro}}_{\text{pictogram}}$	0.703	0.753	0.784	0.813	0.809	0.846	0.856	**0.859**	0.859	0.863	0.855
runtime [ms]	0.13	0.14	0.16	0.17	0.19	0.21	0.23	**0.25**	0.27	0.29	0.31

Table 6.6: Classification results for varying training set compositions. Flipping and adding pure labels in addition to matched (true positive) predictions particularly improves pictogram classification.

matched predictions	✓	✓	✓	✓	✓[1]
flipped predictions	✗	✓	✗	✓	✓[1]
labels	✗	✗	✓	✓	✓[1]
flipped labels	✗	✗	✗	✓	✓[1]
$\mathcal{R}^{\text{micro}}_{\text{state}}$	0.980	0.979	0.980	0.978	**0.983**
$\mathcal{R}^{\text{macro}}_{\text{state}}$	0.908	0.900	0.892	0.868	**0.916**
$\mathcal{R}^{\text{micro}}_{\text{pictogram}}$	0.937	0.933	0.939	0.934	**0.944**
$\mathcal{R}^{\text{macro}}_{\text{pictogram}}$	0.859	0.890	0.878	0.871	**0.881**

[1] only flip selected classes, state: *off*, pictogram: *arrow-right/left, bicycle*

6.5.3 Network Modifications

This section presents various network modifications in order to investigate the impact of feature kernel strides, the number of feature kernels and other important network parameters. With regard to network topology and parametrization, endless variations can be performed. This section focuses on the most promising ones, while keeping the basic topology of the selected network. The initial parameter choice used in previous experiments is inspired by other basis network according to a best guess approach.

Variation of filter kernel strides: Many basis networks utilize higher feature kernel strides in early feature layers. Background of this is a significant reduction of runtime, which is primarily determined by the early feature layers. An early reduction of the spatial size can either be reached by high feature kernel sizes and strides or pooling. As already discussed, pictogram classification requires local, pixel-level information and is thus incompatible with early pooling or high strides. Inception-v3 uses a kernel stride of 2 in the first feature layer. In theory, pictogram recognition should benefit from very small filter kernel strides, whereas state recognition may not require very small strides. Motivation to keep the kernel strides as large as possible is primarily a significant decrease of classification time. In order to illustrate the effect of the kernel stride, Figure 6.15 (left) shows the impact of stride variation in the first three convolutional layers. Because higher strides decrease the the spatial output size, padding is increased in order to guarantee fixed output dimensions independent of the configured strides. *As expected, small strides are very beneficial for pictogram classification.* Smaller strides help to generate more descriptive features with regard to the spatial frequency of pictograms. State recognition benefits from larger strides.

Variation of filter kernel amount: The initial number of filter kernels per convolutional layer is inspired by the Inception-v3 network. Because both, a very small and very high number of filter kernels, can negatively influence the performance, Figure 6.15 (centre) investigates a variation of the number of kernels in 25%, 50%, 150% and 200% of the initial amount of kernels in the first three layers. Motivation to keep the number of kernels as low as necessary is a reduction of runtime. The initial parametrization (100%) seems to be a suitable choice. For a higher amount of kernels, only slight improvements are recognizable.

Network Depth (Receptive Fields): As mentioned multiple times, from the perspective of theory both attributes do not require deep network layers or large receptive fields, respectively. In order to show the impact of the network depth, two further investigations are performed. The first truncates the network by removing the layers pool1, conv2_1 and conv2_2 (see Figure 6.14). The classification is performed on top of feature layer conv1_3. The second experiment extends the network by three additional layers pool2, conv3_1 and conv3_2 before classification. A truncation

shows a slight drop in performance, but already promising recall values. However, the extended network only shows improvement for pictogram classification. This indicates, that a higher number of network parameters may support classification problems with a higher number of classes, i.e. high number of output neurons.

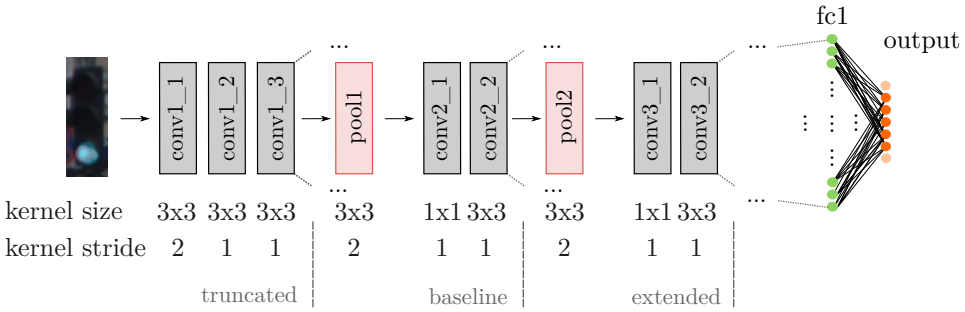


Figure 6.14: Three network forms investigated for attribute classification. Experiments on truncated, baseline and extended network give impressions on the impact of receptive fields/network size on the classification results.

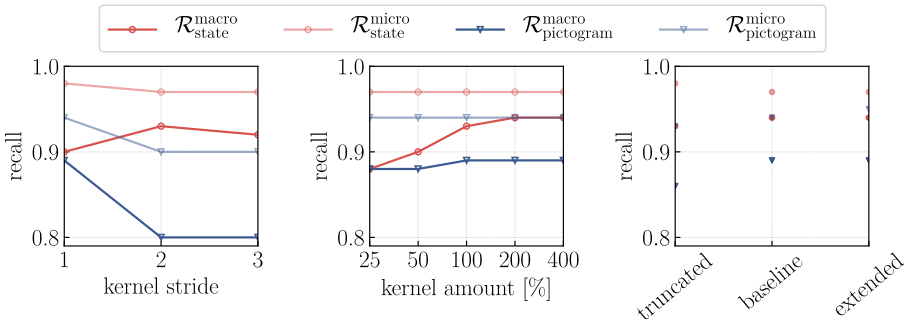


Figure 6.15: Impact of filter kernel stride and amount as well as network depth. Small strides are crucial for pictogram classification in order consider the high spatial frequency in the feature space. Increasing the number of filter kernels supports both attribute classifiers to a certain extent. Truncating the basis network slightly downgrades performance, while extending the network does not provide significant improvements. This supports the presumption, that small networks are sufficient.

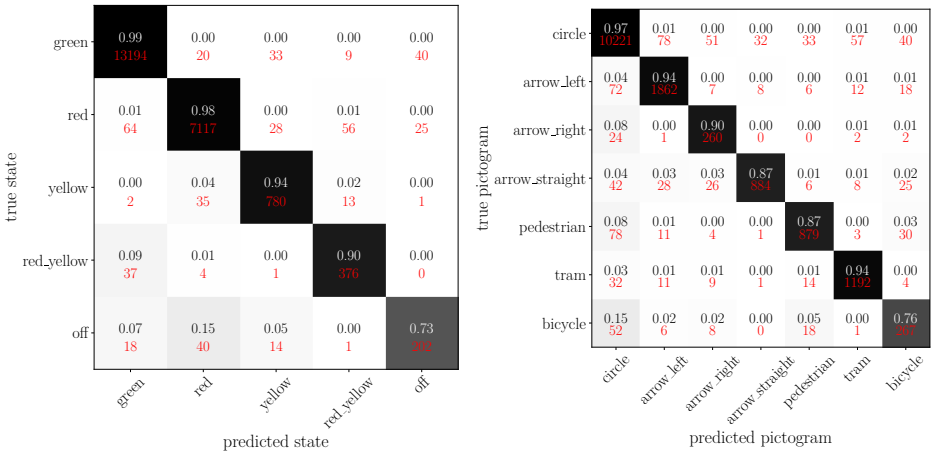


Figure 6.16: Best performing network results. Especially pictogram classification strongly benefits from the presented optimizations and underrepresented classes show strong improvements.

6.6 Best of Two Worlds

Previous results showed, that both methods yield advantages with respect to particular attribute classes. *While the implicit SSD solution reaches a higher recall on pedestrian and tram traffic lights, the explicit RoI classification outperforms SSD on arrow pictograms.* To bring both advantages together, Figure 6.17 presents a hierarchical classification scheme. In a first classification step, the so called target road user is determined by an implicit classification inside the SSD framework. This classification outputs one class of the set $\mathcal{S}_{\text{TRU}} = \{\text{tram, bicycle, pedestrian, vehicle}\}$. The output of the target road user classification is given by

$$y_{\text{PICTO}} = \begin{cases} x_{\text{INLAY}} & \text{if } y_{\text{TRU}} = \text{vehicle}, \\ y_{\text{TRU}} & \text{otherwise}, \end{cases} \qquad x_{\text{INLAY}} \in \mathcal{S}_{\text{inlay}}$$

where $\mathcal{S}_{\text{inlay}} = \{\text{arrow_right, arrow_left, arrow_straight, circle}\}$. In easy words, a subsequent inlay classification is only performed on classified vehicle lights. Figure 6.18 illustrates the results for the presented classification scheme, implicit target

road user classification as well as explicit vehicle inlay classification. It shows, that the advantages of both methods can be combined reaching the best overall performance. Bicycle traffic lights seem to be one major remaining challenge. However, this already turned out to be not only a limitation of classification, but of labeling as well.

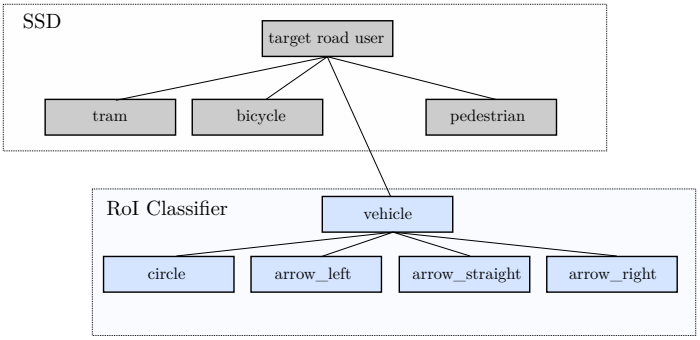
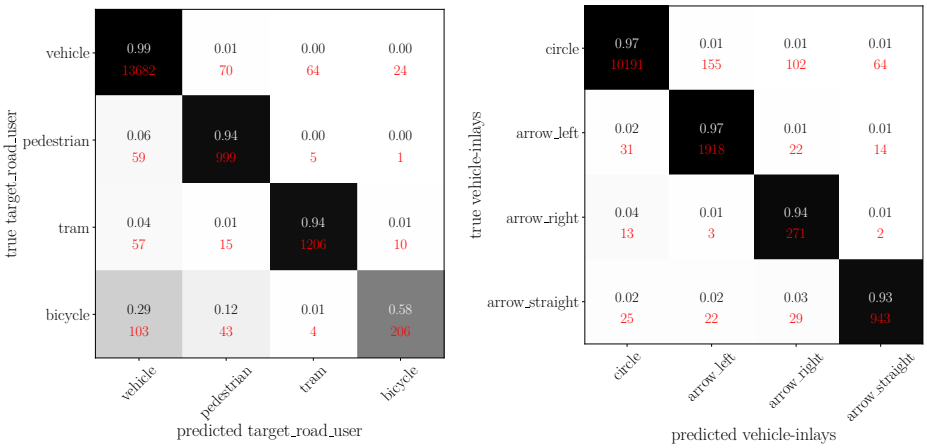


Figure 6.17: Hierarchical pictogram classification. The target road user is determined by an implicit classification in the SSD framework. Vehicle traffic lights are classified in a subsequent step into one inlay type circle, arrow_left, arrow_straight or arrow_right.



(a) Target road user classification result (b) Vehicle inlays classification result

Figure 6.18: A prediction of the target_road_user within SSD and inlay classification as a subsequent RoI classifier shows the best overall results.

Figure 6.19 illustrates detection and attribute classification results. Attribute classification result is visualized in the form of a schematic below the detection.



Figure 6.19: Qualitative detection and attribute classification examples. Detection boxes are colorized according to the predicted state and the schematics visualize the attribute predictions.

6.7 Discussion

This chapter has presented two methods to extend CNN-based object detectors for attribute classification with application to traffic lights. The first method is an extension of a common region proposal or classification head, by using additional convolutional layers predicting attribute probabilities. By extending the loss function, the network is able to predict status and pictogram. Because the strict SSD methodology requires all prediction tasks to be set upon one identical feature map, the choice of features may not be optimal. While the detection task benefits from deep networks, this is not necessarily the case for pictograms, which have a very high spatial frequency. Local, pixel-level features tend to vanish in deep layers required for pictogram classification.

A two-stage classification using independent classification networks as a second comparison method is utilized for two main reasons. It is entirely decoupled from the detection step and the network can thus be trimmed-down to avoid very large receptive fields. Another advantage is, that the network cannot use any feature outside the actual bounding box, because the input of the network is the pure region of interest. When performing an implicit classification in the SSD framework, the receptive fields significantly exceed the actual traffic light region, which heightens the probability, that background features are involved into the decision. This may be an important advantage when considering safety aspects.

Extensive investigations can be concluded as follows: Implicit classification strongly depends on the choice of feature layer(s) used for prediction. The optimal feature layers may not be the optimal feature layers for state or pictogram prediction, leading to a performance compromise. A further finding is, that the number of prior boxes seems to have a major impact on attribute classification. Investigations have shown, that this may be caused by the NMS step, in which, depending on the prior boxes, more boxes with varying state/pictogram have to be merged. The winner box selection is purely dependent on location and detection confidence and does not consider state or pictogram.

Investigations on the RoI classifier have shown, that *pictogram classification benefits from larger resolutions, while state recognition already has shown an optimal performance at lower resolutions.* The main benefits of this methodology are the flexibility in the choice of network architecture and data augmentation, which is much simpler than for the input of SSD. *This helped to increase the classification recall on right arrows by almost a factor of two by adding ground truth labels in addition to the SSD predictions and applying vertical flipping.* On the other side, SSD and RoI classifier cannot be jointly trained leading to a growing training procedure.

With regard to state recognition, both methods show a similar performance. A closer look on wrongly classified samples has shown, that the actual results are even better than the already impressive confusion matrix values. Nearly all green traffic lights are correctly classified, only very rare confusions between red, yellow and red-yellow occur, which seem to be mostly caused by either LED flickering or uncommon light positions. One of the most frequent error cases are yellow lights with an active uppermost light classified as red. *This indicates, that the network overweights the light position instead of its color. The biggest challenge are off lights, which visually lack of an active light.* However, the overall result is very promising and could be even improved by using track information from the perspective of an overall system.

Pictogram classification requires much more adaptations and has shown its full potential when combining strengths of both methods. *It turned out, that distinguishing between vehicle, pedestrian, tram and bicycle traffic lights is optimal, when predicting it in a first step in the SSD framework. This can be explained by context information, such as intersection topologies, which seem to support classification for those classes.* Many pedestrian traffic lights are recognizable without a visible inlay from far distances. However, vehicle inlays are usually not predictable from context information and need classification of local features inside the traffic light region. This is achieved by a subsequent RoI classifier for vehicle inlays. Combining both methods reaches a recall of approximately 95 percent for most pictograms. *The criticality of spontaneous pictogram misclassifications can be assumed to be lower than for the traffic light state. Pictograms are static and can thus be more easily compensated by tracking modules.*

Another finding of the chapter is the *need for an even more sophisticated labeling hierarchy.* This especially refers to a flag of the actual recognizability of pictogram/state and other attributes, which are currently inferred from the entire sequence and track information. Visual inspection has shown, that for many misclassifications the active light is not visible (mainly due to viewpoint) and thus should be excluded during evaluation. The current labeling attributes only enable filtering based on object size, but not on actual recognizability. Nevertheless, neglecting labeling errors and dataset limitations, the achieved results are very promising.

Chapter 7

Confidence Calibration

Single Shot Detection predicts one confidence vector for each single prior box. Similar to observed in other publications investigating CNN-based detectors [GPSW17], the predicted confidences do not express the true probability of the prediction as shown in Figure 7.1. Traffic light recognition is a safety-critical application increasing the requirements on calibrated confidences. It is also one crucial requirement of subsequent multi-object tracking modules, in which traffic light recognition results are commonly utilized. Consequently, the main goal of confidence calibration is, that confidence values do express the true correctness likelihood. In simple words, a set of 1000 predictions with a confidence of 50% should consist of 500 true positives and 500 false positives to be perfectly calibrated.

This chapter is separated into the following sections: A brief overview of confidence calibration methods is given in Section 7.1. Section 7.2 shows metrics and representations expressing the quality of confidence calibration. Section 7.3 presents methods to perform a confidence calibration on the present traffic light object confidences. Because state and pictogram confidences also show a poor calibration, Section 7.4 presents extensions to perform multi-class calibration. The calibration methods are evaluated in Section 7.5. Attribute calibration is exemplified by state calibration.

7.1 Related Work

In Guo et al. [GPSW17], an extensive overview on calibration of neural networks has been introduced. Various, existing methods are compared against each other. Histogram-binning [ZE01] calibrates confidences after dividing predictions into a defined number of bins, most commonly according to equal confidence intervals. One calibration value per bin is determined.

A generalization of histogram binning is isotonic regression [ZE02], which learns a piece-wise constant function, which - in contrast to histogram binning - optimizes the bin intervals as well.
The mentioned approaches are all non-parametric. Instead, Platt scaling [Pla+99] learns scalar parameters to transform the logits output of the network $\underline{z}_i$ (non-probabilistic, see Equation (5.22)). A simplified and well-established version of platt scaling is temperature scaling [GPSW17; NC05]. Temperature scaling learns one single parameter T, which is applied to the logit vector before softmax operation. The overview work of Guo et al. [GPSW17] compares all methods on a high number of datasets. They conclude, that performance is often independent of the complexity of the approach. Temperature scaling followed by histogram binning shows the best overall performance. Seo et al. [SSH19] present a calibration method within deep neural networks by stochastic inference. They introduce a variance-weighted confidence-integrated loss function, which enables to predict calibrated confidences in one shot. However, they conclude that overconfidence is alleviated, but the calibration error still amounts more than 12 and 5 percent, depending on the method. Since there is no comparison with traditional calibration methods, the quality of the method is hard to judge. When comparing to the results presented in [GPSW17], benefit over traditional methods is doubtful. In the following, histogram binning is used, investigated and adapted as the method of choice.

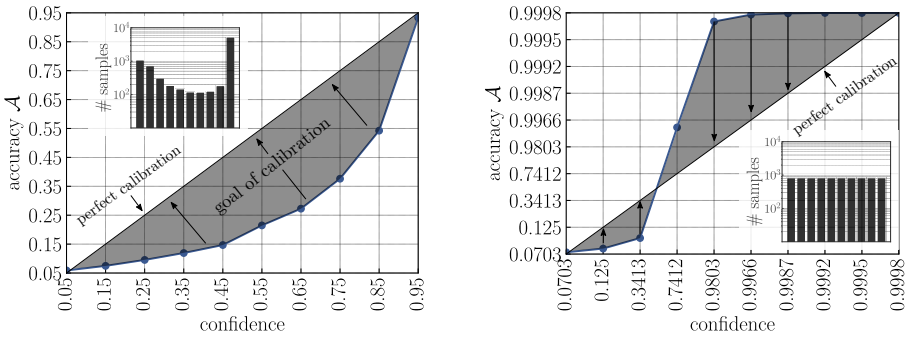
7.2 Definitions, Measures, Metrics and Pre-Investigations

Problem Formulation: Assuming a binary (detection) and multi-class classification (attribute classification) task with the ground truth $\omega \in \{\omega_0, ..., \omega_L\}$. The one- and two-stage detector/classification outputs $h(x) = (\hat{\omega}, \hat{c})$ for a given input x, where $\hat{\omega} \in \{\hat{\omega}_0, ..., \hat{\omega}_L\}$ describes the predicted class. Perfect calibration can be formulated as

$$\mathcal{P}(\hat{w} = w|\hat{c} = c) = c, \tag{7.1}$$

where $c_i \in [0, 1]$ is the predicted confidence of the network. From a set of 100 predictions, each with a confidence of 0.5, 50 predictions have to be true positives when assuming a perfectly calibrated system.

Figure 7.1a shows the distribution of SSD prediction output confidences (detection task) as black bars inside the figure. Please notice the logarithmic y-axis in the inner, sample-distribution plot. The distribution is highly unequal - the majority



(a) Fixed-width binning

(b) Fixed-sample binning

Figure 7.1: Reliability diagram comparison for the raw SSD output confidences (input of calibration), generated for bin centres obtained from fixed confidence width binning (a) and fixed sample count binning (b). While the majority of negative samples is located in the left-most bin, almost all positive samples are located in the right-most bin with absolute confidences larger 99 percent. Interestingly, SSD low confidences overestimate true accuracy, whereas high confidence values underestimate true accuracy (see right figure). Please notice the varying bin centre values of both figures (x-axis) resulting from both binning types.

of predictions show very low confidences close to zero (non traffic light regions) or very high confidences close to one (traffic light regions), respectively. In order to avoid overfitting towards very low confidences values, Figure 7.1 shows the reliability diagram for predictions filtered by FPPI $\geq 10^0$, i.e. the corresponding confidence value at FPPI=10^0 is determined and predictions with confidences smaller than this value are filtered. The raw, unfiltered SSD output contains approximately 200 boxes (200 boxes with highest confidence value) per image, which means that without filtering, more than 95% of all predictions used for calibration are located on non-traffic light regions with confidence values very close to zero. This is avoided by filtering almost all true negatives generating a better balance between positive and negative samples.

Reliability Diagrams: In order to visually represent the quality of calibration, reliability diagrams are commonly used. Figure 7.1 shows a reliability graph (blue dots), which visualizes the accuracy as a function of the confidence. Therefore, the confidences are grouped into K intervals. Each prediction of the network is assigned to one bin according to its confidence value. For each bin B_k, an accuracy $\mathcal{A}$ is

calculated according to

$$\mathcal{A}(B_k) = \frac{1}{|B_k|} \sum_{i \in B_k} f(i), \qquad f(i) := \begin{cases} 0 & \hat{\omega}_i \neq \omega_i \\ 1 & \hat{\omega}_i = \omega_i \end{cases}, \tag{7.2}$$

i.e. the accuracy expresses the normalized value of correct predictions. $\hat{\omega}_i$ and ω_i describe the predicted and true class, respectively and $|B_k|$ the number of samples falling into bin B_k. Bins can either be determined to guarantee a fixed confidence width or to equalize the number of samples for each bin. In easy words, accuracy expresses the amount of true positives among all bin samples.

Fixed-width binning vs. Fixed-sample binning: Binning into equally-sized bins is the most intuitive proceed when determining bin intervals. Fixed-width bins are determined as

$$B_k = \left[\frac{k-1}{K}, \frac{k}{K}\right) \text{ and } B_K = \left[\frac{K-1}{K}, 1\right], k \in \{1, ..., K-1\}, \tag{7.3}$$

where K is the number of bins. The main limitation of fixed-width binning is, that bin determination is independent of the actual confidence distribution, which is likely not uniformly distributed. As shown in Figure 7.1a (bar chart), the SSD output predictions (note log-scale on the right y-axis) are highly unequally distributed with an exponential increase towards confidences close to 0 and 1. Utilizing fixed-width binning, many bins contain only a small amount of samples, which consequently cannot guarantee to be well calibrated. Other bins are overrepresented and group a large amount of samples.
Instead, fixed-sample binning guarantees an approximately equal number of samples per bin (see Figure 7.1b). The (k/K)-th quantile of the confidence distribution is determined for $k \in \{1, ..., K-1\}$ leading to $K-1$ quantiles q_k [GPR20]. The intervals are then obtained by

$$B_1 = [0, q_1), B_K = (q_{K-1}, 1] \text{ and } B_k = [q_{k-1}, q_k]. \tag{7.4}$$

Within this thesis, the focus lies on utilizing fixed-sample binning, which is more suitable for the present SSD output confidence distribution.

Expected Calibration Error (ECE): The expected calibration error is a scalar measure to describe the quality of calibration. Similar to reliability diagrams, bin boundaries are determined by equalizing either bin interval width or the sample amount in each bin. ECE can be calculated by

$$ECE = \sum_{k=1}^{K} \frac{|B_k|}{M} \left| \mathcal{A}(B_k) - \frac{1}{|B_k|} \sum_{i \in B_k} \hat{c}_i \right|, \qquad M = \sum_{m=1}^{K} |B_m|, \tag{7.5}$$

i.e. a weighted subtraction of true accuracy $\mathcal{A}$ and actual average confidence values for each single bin. Weighting is performed depending on the number of samples for bin $|B_k|$, divided by the total number of samples M [NDZ+19].

Average Calibration Error (ACE): The average calibration error calculates the difference between bin accuracy (fraction of correct predictions) and average confidence, which can be formulated as

$$ACE = \sum_{k=1}^{K} \left| \mathcal{A}(B_k) - \frac{1}{|B_k|} \sum_{i \in B_k} \hat{c}_i \right|. \tag{7.6}$$

Notations are similar as introduced for ECE. In other words, *ACE equally weights the calibration quality in each bin independent of the number of samples per bin.* Please note, that for an equal number of samples in each bin ($|B_1| = |B_2| = ... = |B_K|$), the average calibration error equals the expected calibration error.

State-wise Reliability: As observed in Figure 5.25, a relationship between the detection performance and the true state exists. An aggravated detection can be observed for yellow and particularly off lights. This suggests to investigate the relationship between confidence reliability and the predicted state. For this purpose, the detection confidences are grouped state-wise and reliability is investigated.
Figure 7.2 shows the detection confidence distributions for matched predictions (IoU ≥ 0.3, consequently true positives) in red and unmatched predictions (IoU ≤ 0.3, consequently false positives) in blue, grouped by the predicted state. Black numbers show the numerical value for the percentage amount of matched predictions within each bin. This value corresponds to the accuracy $\mathcal{A}$ introduced in Equation (7.2). *One conspicuousness is the distribution of off lights, which shows a trend towards lower absolute confidence values. Intuitively, the detector appears to be less confident for off lights leading to a decreased performance on off lights compared to other states.* When performing a state-wise calibration, the possibility of improving the detection performance on several sub-classes is given. However, this may also affect the performance of the strong classes. While one general calibration model applied to all predictions does not affect the system characteristic (assuming mathematical monotony), a state-wise calibration changes the performance and the characteristic of the system. *The main motivation of state-wise calibration is to close the gap between off lights and active states investigated in Figure 5.25 without significantly downgrading the performance on the latter.* State-wise calibration is compared with general calibration in the following evaluation section.

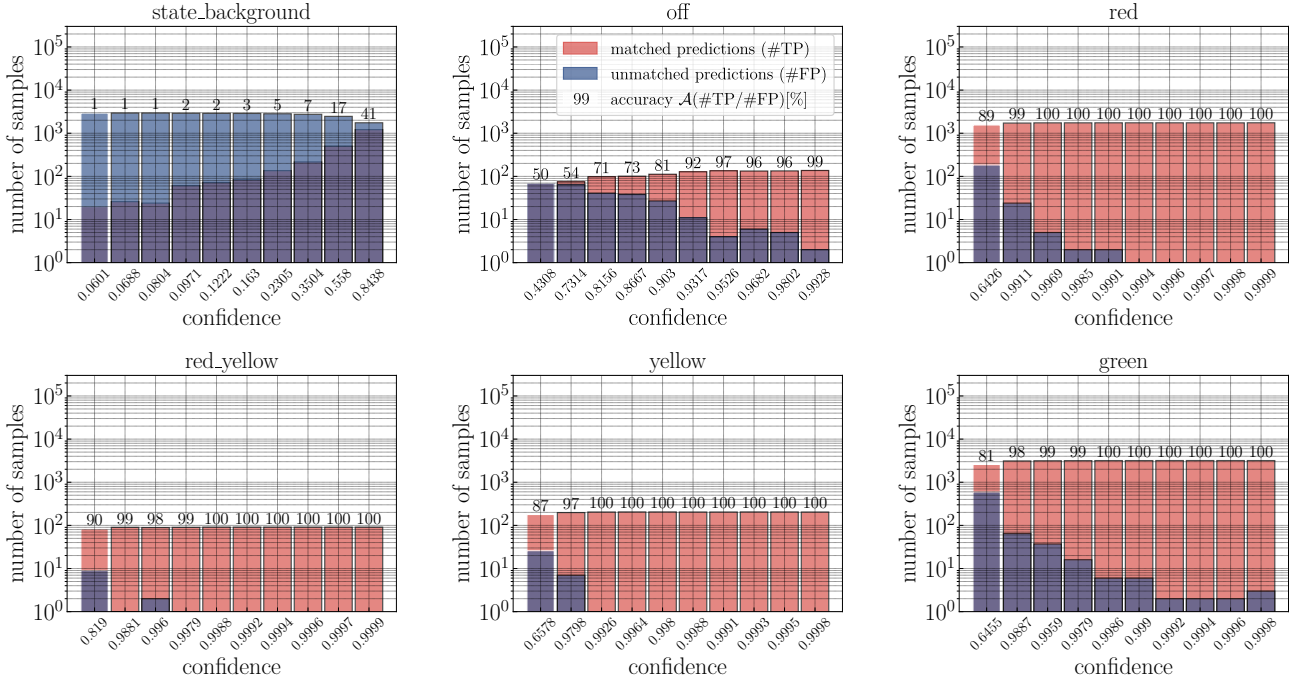


Figure 7.2: Confidence distribution for matched (IoU ≥ 0.3, consequently true positives) in red and unmatched predictions (IoU ≤ 0.3, consequently false positives) in blue with respect to the predicted state. Black numbers above each bin refer to the accuracy (reliability) within each bin. Bin intervals are determined to guarantee a fixed sample amount and are determined independently for each state. Please note, that the confidence distributions refer to the detection and not state classification confidences. Intuitively, off lights show a significantly lower accuracy. In consequence, detecting off lights and increasing false positives are interdependent.

7.3 Confidence Calibration Methodologies

Histogram-Binning Calibration [ZE01]: Similar to introduced in terms of reliability diagrams, histogram-binning calibration initially groups the predictions into K intervals, either defined by fixed confidence intervals or by intervals containing an equal number of predictions. For each bin K, a representative calibrated value $\mathcal{C}_k$ has to be determined. At inference, the predicted confidence $\hat{c}_i$ falls into one bin of B_k ($k \in \{1, ..., K\}$) and is calibrated to the value $\mathcal{C}_k$. The goal is to minimize the bin-wise squared loss between this calibration value and all predictions $\hat{c}_i$ falling into bin B_k, which can be formulated as

$$\min_{\mathcal{C}_1,...,\mathcal{C}_k} \sum_{k=1}^{K} \sum_{i \in B_K} (\mathcal{C}_k - \hat{c}_i)^2. \tag{7.7}$$

The solution of this equation results in $\mathcal{C}_k$, amounting the percentage number of positive (matched) samples from the set of predictions within bin k. Assuming 100 predictions falling into bin k, from which 70 are matched and therefore true positives, a test sample falling into this bin is calibrated to a probability of 0.7. Figure 7.3 exemplarily illustrates histogram-binning calibration using fixed-width and fixed-height binning, respectively. The more the sample (confidence) distribution

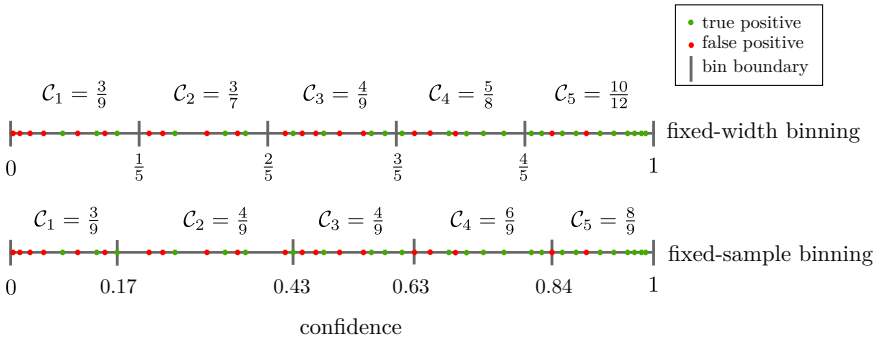


Figure 7.3: Histogram binning scheme ($K = 5$) according to fixed-width binning (top) and fixed-sample binning (bottom). The calibration value $\mathcal{C}_k$ for each bin is determined according to the number of true positives divided by the sum of true and false positives within the bin. The more the sample distribution diverges from a uniform distribution, the larger the difference between fixed-width and fixed-sample binning.

deviates from a uniform distribution, the larger the benefit of fixed-sample binning.

State-wise Histogram Binning Calibration: With regard to Figure 7.2, a state-wise detection confidence calibration is introduced. It performs confidence calibration with respect to the predicted state. For each class, one classification model is created according to Equation (7.7).

Filtering at fixed FPPI value: One disadvantage of histogram-based calibration methods is its quantized output. While the output probability of the network is only limited by float-precision, the output of histogram binning methods is quantized by the number of bins. Performing histogram binning calibration with $K = 10$ bins leads to at most 10 unique output confidence values. Typically, CNN-based detectors are operated at one confidence threshold which is selected according to one desired and constant system characteristic, such as false positives per image or miss rate. This is significantly aggravated, because many operating points are removed from calibration. In order to avoid this, it is feasible to perform confidence filtering before calibration. The confidence threshold at the desired operating point is determined before calibration and calibration is only performed on confidences of predictions larger than this threshold. Moreover, calibration is generated only on the predictions of interest. This thesis uses FPPI=10^0 as a threshold, i.e. the confidence threshold corresponding to this false positives per image value is determined and predictions with a smaller confidence than the threshold are not considered during calibration. A second advantage of this filter operation is, that bin boundaries are not overrepresented in low confidence regions. This is likely the case without filtering, because most predictions show confidences close to zero and bin determination according to the methodology described in Equation (7.4) consequently overfits bin boundaries towards less relevant confidence regions dominated by false positives.

7.4 Attribute Calibration Extensions

The previous section refers to calibration of binary problems, i.e. the calibration of the object detection confidence. Because the confidences predicted for state and pictogram classification are desired to be calibrated as well, histogram-binning has to be extended to be applied to multi-class problems.

The most common proceed to use histogram binning calibration is to create L calibration models, one for each class. For instance, $L = 6$ calibration models for state calibration are generated (background, off, red, red_yellow, yellow, green). Consequently, a one-versus-all calibration is performed. Input of calibration is the uncalibrated attribute confidence vector $\underline{c}^{\text{attr}} = [c^{C0}, c^{C1}, ..., c^{CL}]^\top$, where $\sum c^l = 1$.

By applying calibration, the calibrated output is $\underline{\hat{c}}^{\text{attr}} = [\hat{c}^{C0}, \hat{c}^{C1}, ..., \hat{c}^{CL}]^\top$. Because each confidence is independently calibrated, it yields $\sum \hat{c}^l \neq 1$.

Probabilities ($\sum \hat{c}^l = 1$) are obtained by dividing each element $\hat{c}^l$ by $\sum_{l=1}^{L} \hat{c}^l$ as a final step. Similar to Equation (6.3), the winner class for state/pictogram classification is determined by finding the maximum confidence value. Because each class is independently calibrated, attribute calibration can change the winner class and thus change the system characteristic [GPSW17; ZE02].

7.5 Evaluation

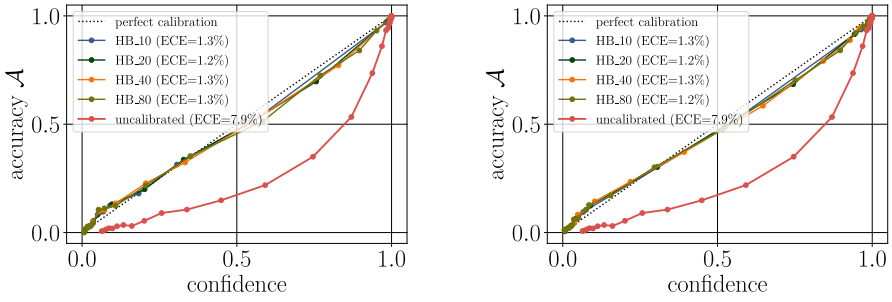
The following sections separately present evaluations regarding detection and attribute confidences calibration, respectively. Detection results are determined considering scenario DS4.

7.5.1 Detection Confidence Calibration

This section presents results of histogram binning (equidistant confidence intervals and sample-based binning) and its impact on the ROC curve. Important to notice is, that *all methods are evaluated on bins determined by sample-based binning.* This is aligned to the expected confidence input distribution shown in Figure 7.4a in red in order to improve coverage of frequently occurring confidence values.

Figure 7.4 compares the results for histogram binning with equal bin width (left) and equal sample amount (right). Legend names express the number of bins, for instance HB_10 refers to histogram binning using 10 bins. Fixed-sample binning shows a significant improvement compared to the uncalibrated confidences. The calibration quality is relatively independent of the number of bins. Curves are very similar, whereas a higher number of bins leads to a higher number of unique output confidences values, i.e. a higher quantization of the calibration output. This increases the number of selectable operating points, which may be desired for application.

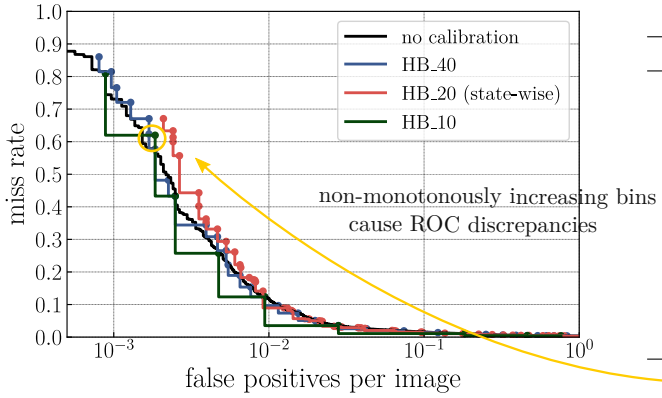
Figure 7.5 illustrates the ROC curves before calibration (black), after equal-sample amount histogram binning calibration using 10 (green) and 40 bins (blue) as well as state-wise calibration using 20 bins (red). As discussed, binning causes a discretization of the ROC curves according to the number of bins. Besides the discretization, the characteristic of the curves varies. While for $K = 10$ only one operating point is not located on the initial curve, this number increases for $K = 40$ and the state-wise



(a) Fixed-width binning

(b) Fixed-sample binning

Figure 7.4: Reliability diagram comparing uncalibrated and histogram-binning calibrated accuracy. Curves are obtained from fixed-width (a) and fixed-sample binning (b) calibration, respectively. Histogram binning improves ECE by almost 7% compared to uncalibrated. The choice of number of bins shows almost no effect, except a higher quantization of the output.



HB_10 calibration map

B_k	$\mathcal{C}_k$
[0.0000, 0.0843)	0.0074
[0.0843, 0.1657)	0.0227
[0.1657, 0.5168)	0.0662
[0.5168, 0.9656)	0.5249
[0.9656, 0.9949)	0.9641
[0.9949, 0.9982)	0.9927
[0.9982, 0.9991)	0.9978
[0.9991, 0.9994)	0.9995
[0.9994, 0.9996)	0.9993
[0.9996, 1.0000]	0.9996

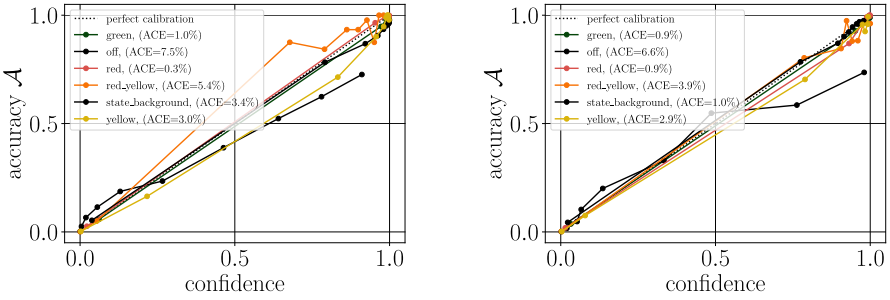
Figure 7.5: ROC curve before and after calibration for histogram binning using 10 and 40 bins as well as state-wise calibration using 20 bins. For high FPPI values, the operating points after calibration are on the original ROC curve. For smaller FPPI values, calibrated curves move away from the original curve, because calibration turns out to not be continuously monotonously increasing.

calibration. The calibration table helps to understand this side-effect, showing the calibration bin intervals and calibration values obtained from the calibration process. For high confidence values, the calibration is partially not monotonously increasing (see yellow block). This changes the system characteristic in the respective area leading to a discrepancy between initial and calibrated curve. Intuitively, non-monotonously increasing calibration is not a desired property. With decreasing FPPI values, the proportion of labeling errors (evaluated false positives which actually true positives) significantly increases. Moreover, the amount of false positives is vanishingly small. This leads to a much higher sensitivity of low FPPI values (corresponding to high B_k and $\mathcal{C}_k$ values) with regard to labeling and consequently to undesired calibration errors.

7.5.2 Attribute Confidence Calibration

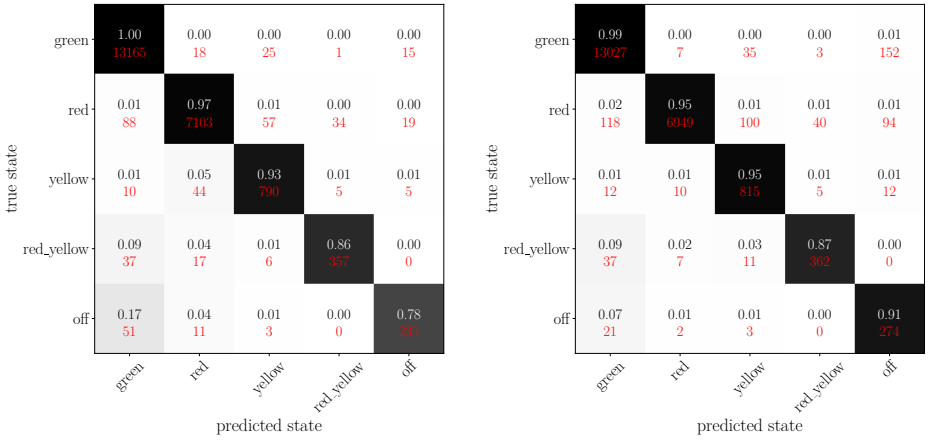
Within this subsection, attribute confidence calibration is evaluated exemplarily by means of state confidence calibration. Besides obtaining improved calibrated confidences for each state class, calibrating each confidence separately yields the advantage to potentially improve the system characteristic. The winner class determined by an argmax operation applied on the calibrated confidences, which may be influenced by the preceding calibration.

Figure 7.6 shows the reliability graph illustrating uncalibrated (a) and calibrated (b) reliability for each state class. All graphs are determined for $K = 20$ bins. Utilizing a higher number of bins is problematic, because this decreases the amount of samples per bin and causes statistical issues for underrepresented classes. Slight improvements are observable for all classes except red. However, especially red_yellow and off still show high calibration errors. In particular, off confidences are raised, see the most-right points for off (black) in both figures. Raising off confidences has two main effects, see Figure 7.7. One positive effect is the impact on classification recall for state off, which increases by 13%. The undesired side-effect are elevated misclassifications for green and red. The numbers show, that the micro recall $\mathcal{R}_{\text{state}}^{\text{micro}}$ is decreased while the class-balanced metric $\mathcal{R}_{\text{state}}^{\text{micro}}$ is significantly increased. A similar behaviour can also be observed for yellow and red_yellow, which allows to conclude, that state calibration particularly improves underrepresented classes. Which of both optima is the desired one depends on the application.



(a) before calibration (b) after calibration

Figure 7.6: Reliability diagram for uncalibrated (a) and calibrated (b) state confidences. In general, slight improvements with regard to ACE are observable. However, calibrating off and red_yellow is challenging, because of the lack of sufficient samples. This leads to calibration inaccuracies for both classes. In particular, off confidences are shifted towards higher values, compare the most-right points for off in both figures. This adjusts the state classification results, see Figure 7.7.



(a) before calibration (b) after calibration

Figure 7.7: Classification recall before/after state confidence calibration. Underrepresented classes improve (yellow: 93% → 95%, red_yellow: 86% → 87%, off: 78% → 91%), at the expense of a drop for red (97% → 95%) and green (100% → 99%) .

Chapter 8

Dataset Relabeling - Significance for Performance and Evaluation Reliability

Previous investigations on the type of false positives, as well as a closer look on erroneous classifications showed a demand for a higher dataset quality. This demand mainly refers to the quality of the testing set, required to produce meaningful and correct evaluation results. In the present case, the focus lies on two types of dataset errors: not labeled traffic light instances, which are yet recognized by the vision system, as well as erroneous attribute labels. If a prediction is located on a not labeled traffic light instance, evaluation categorizes this prediction as a false positive, in fact being a true positive. Non-labeled instances are problematic, especially considering operating points with low FPPI values. This means, that *evaluating operating points at low FPPI values presupposes a very high dataset quality.* When considering all alleged false positives determined by evaluation, many traffic light regions are recognizable, indicating the need for relabeling efforts.

Correcting wrong attribute labels is the second goal of this chapter. Because the introduced system predicts state and pictogram, relabeling also includes both types of attributes. Other attributes are not corrected within this thesis.

This chapter focuses on three main topics. Section 8.1 presents an automated relabeling pipeline, which creates relabeling proposals from evaluation of the presented detection and classification system. Section 8.2 presents a comparison of detection and attribute classification results using the initial and corrected, relabeled test set, respectively. Moreover, it presents a newly trained network using relabeled training data, which shows the relevance of the training set quality with regard to performance.

8.1 Relabeling Workflow

An overview of the relabeling workflow is shown in Figure 8.1. The label pool contains all labeled data as well as the raw data (image data and others). In the case of DTLD, this includes all recordings from 11 cities in Germany. A sequence-wise split generates training and test set, respectively. This guarantees a unique occurrence of samples of one real-world instance in either training or test set. Further rules, such as adding data from each city in both sets, are applied to guarantee an optimal appearance variation.

The training set is used for CNN training, which outputs one snapshot every 10 000 iterations. A best snapshot selector module determines the best snapshot and performs a bounding box matching. This step determines all overlaps between ground truth and predictions obtained from SSD. Afterwards, multiple basic evaluations are performed. The ROC curve evaluation provides, among others, the FPPI-confidence relationship required by other evaluations operating at fixed FPPI values.

The relabeling evaluator is subsequently called and determines false positives as well as wrong state and pictogram classifications. All three error types are collected for one defined FPPI value. For relabeling in this thesis FPPI=10^{-1} is used. The choice of this value is a trade-off between the percentage amount of true labeling errors and the amount of actual false positives. Usually, the amount of actual false positives exceeds labeling errors for rising FPPI values. FPPI=10^{-1} turns out to be an appropriate choice by keeping the amount of actual false positives low, but already cover a high amount of labeling errors. The relabeling evaluator modifies the original label files of the label pool by adding a relabeling tag to the affected label objects (in the case of a potential wrong attribute) or by adding a new instance (containing the coordinates of the prediction). An additional comment field indicates the labelers the type of error.

All sequences to be relabeled are distributed to various labelers. Please note, that the label format used for labeling differs from the dataset files generated by the label file generator. In the label pool, label files are divided sequence-wise. This simplifies a distribution to multiple labelers. In order to accelerate the relabeling process, the label tool contains a relabeling plugin grouping and constituting all relabeling objects of the label file.
Verifying each relabeling object is the first task of the labeler. Non-traffic lights (i.e. real false positive of the CNN) are initially removed. In the case of actual non-labeled traffic light instances, the predicted box coordinates have to be manually corrected. Similar to the common traffic light labeling proceed, all further attributes are complemented. An important additional step of relabeling is the verification of previous/subsequent images in affected sequences. Most likely, other frames of

the sequence are affected as well, leading to an additional labeling effort. After completing all described tasks, the relabeling object is marked as *resolved*.
In order to review the labeling step, two quality control stages are performed. All label files are quality-controlled by an additional labeler in a superficial manner. This helps to detect general labeling errors - such as a misunderstanding of the labeling rules. An automated quality control is subsequently executed, which checks the completeness of all label attributes, unrealistic aspect ratios or sizes and other properties. Striking findings are fed back to the respective labelers.

Finally, all affected dataset files are updated using the relabeled files after passing the sanity checks. Because the relabeling process does not change the number of labeled images, all new objects are simply added and all affected, existing objects are updated. For evaluation, the main goal of relabeling is to increase the accuracy of the results when comparing with the results on the initial test set. For training, the hope is, that a higher dataset accuracy boosts the performance of the network. In the following, the term initial training/test set refers to the original labels used up to this section. Relabeled training/test set refers to the labels after correction.

Statistics

Table 8.1 shows the amount of generated relabeling proposals with potentially erroneous state or pictogram, as well as overlooked objects (bounding boxes). A total number of 6306 bounding boxes, 1269 labels with potentially wrong state, as well as 1349 labels with potentially wrong pictogram attribute are collected for relabeling. The second row shows the actual number of corrected objects obtained after human correction. *The number of corrected bounding boxes is larger than the number of proposals by a factor of 10. Very likely, not annotated traffic lights in one frame are not annotated in other frames of the sequence as well.* Most sequences contain roughly 5-20 annotated images of one intersection. Labelers are instructed to examine all sequence frames with regard to the to be corrected object instance. If necessary, all additional frames are corrected as well. On this account, the number of corrected objects significantly exceeds the number of proposals.

Table 8.1: Number of relabeling proposals obtained by the introduced methodology and number of corrected objects.

	bboxes	**state**	**pictogram**
# relabel proposals	6306	1269	1349
# corrected objects	60 206	1616	2506

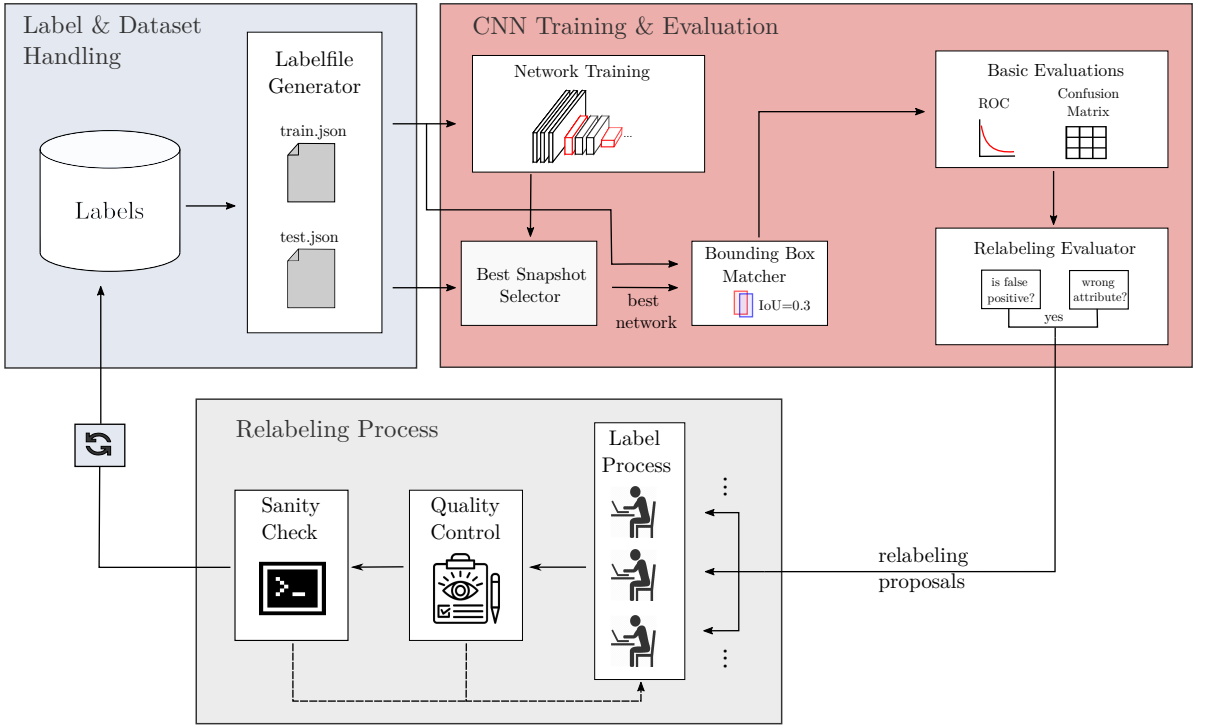


Figure 8.1: In a first step, the label files for training and evaluation are created from the label pool. After training the network, the best snapshot is determined. Basic evaluations are performed, which determine important values required for relabeling evaluation. The relabeling evaluation determines false positives and misclassified ground truths and creates relabeling files supported by the labeling tool. Multiple labelers verify the relabeling proposals. Manual and automatic quality control is performed to reduce further labeling errors.

8.2 Results

The following evaluations present a comparison between results determined on the initial training/evaluation set and the relabeled training/test set. Subsection 8.2.1 presents results determined for one experiment using the initial and relabeled test set. Please note, that results cannot be compared one-to-one, because the best snapshot evaluator determines the best snapshot based on the evaluation results on the respective test set in a first step. Detailed evaluations are subsequently performed solely on the best snapshot. The best snapshot selection can differ between evaluations on initial and relabeled set, respectively.

8.2.1 Significance of Dataset Quality for Evaluation Reliability

This section focuses on the pure impact of relabeling on the results. One network trained on initial training data is used and evaluated twice, using the initial and relabeled test data, respectively. The main aim of this investigation is to clarify, whether and to which extent dataset quality is required to obtain meaningful results.

Detection Results

Figure 8.2 shows a ROC curve comparison between evaluation on initial (blue) and relabeled (red) test data. The relabeled curve shows a significantly improved performance, indicating, that evaluation underestimates the performance for low FPPI values. For higher FPPI values, the performance is roughly equal, because "true false positives" overweight the pretended false positives from annotation errors. *In part, the log-average miss rate curve improves by more than one decade. The results clarify, that extensive data quality improvements are necessary for obtaining meaningful results.* Especially the transition from driver assistance systems to fully autonomous systems rises the requirements on decreasing false positive rates. Consequently, this by far increases the requirements on dataset quality.

Attribute Classification Results

Figure 8.3 shows the state and pictogram classification performance comparison on initial (left) and relabeled (centre) test data. With regard to Figure 6.11, already indicating large percentage labeling errors for red_yellow, the most significant improvement by 9% is obtained. Green remains at 99 %, whereas all other classes

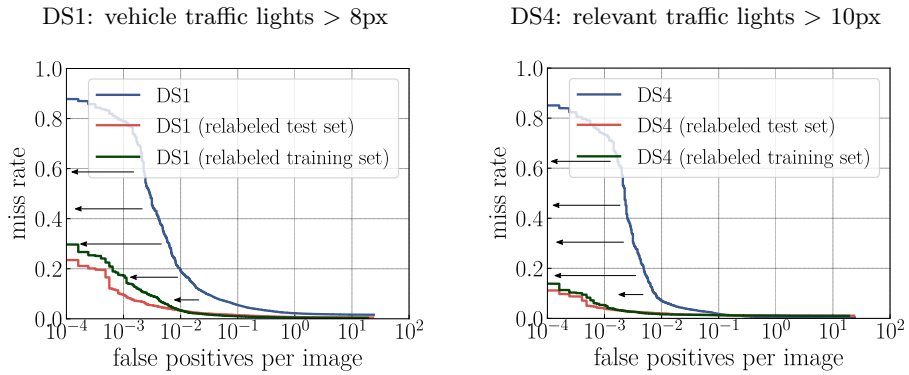


Figure 8.2: ROC curve comparison between evaluation on initial (blue) and relabeled test set (red). A third experiment retrains the identically parametrized network on the relabeled training set (green). Relabeling shows, that miss rates are significantly lower than initially determined for decreasing false positive rates. Importance of test set quality rises for evaluability at low FPPI values. Training set quality has no major impact on the performance.

except of yellow benefit. Red increases by 1.2% and off by 8%. A slight drop for yellow traffic lights may be explained by the described side-effect of a different snapshot (network weights) selection leading to a different system characteristic.

With regard to pictogram classification, a mixed picture is shown. Significant improvements are visible for circle (+1.7%) and arrow_straight (+2.2%), while performance arrow_right shows a decline in recall (-3.6%).

8.2.2 Significance of Dataset Quality for the Network Performance

As indicated in Chapter 3, the second question discussed within this chapter is: *Which impact rising dataset quality has on the system performance?* For this purpose, one network is trained on the relabeled training set and compared with the performance of the network trained on initial training labels.

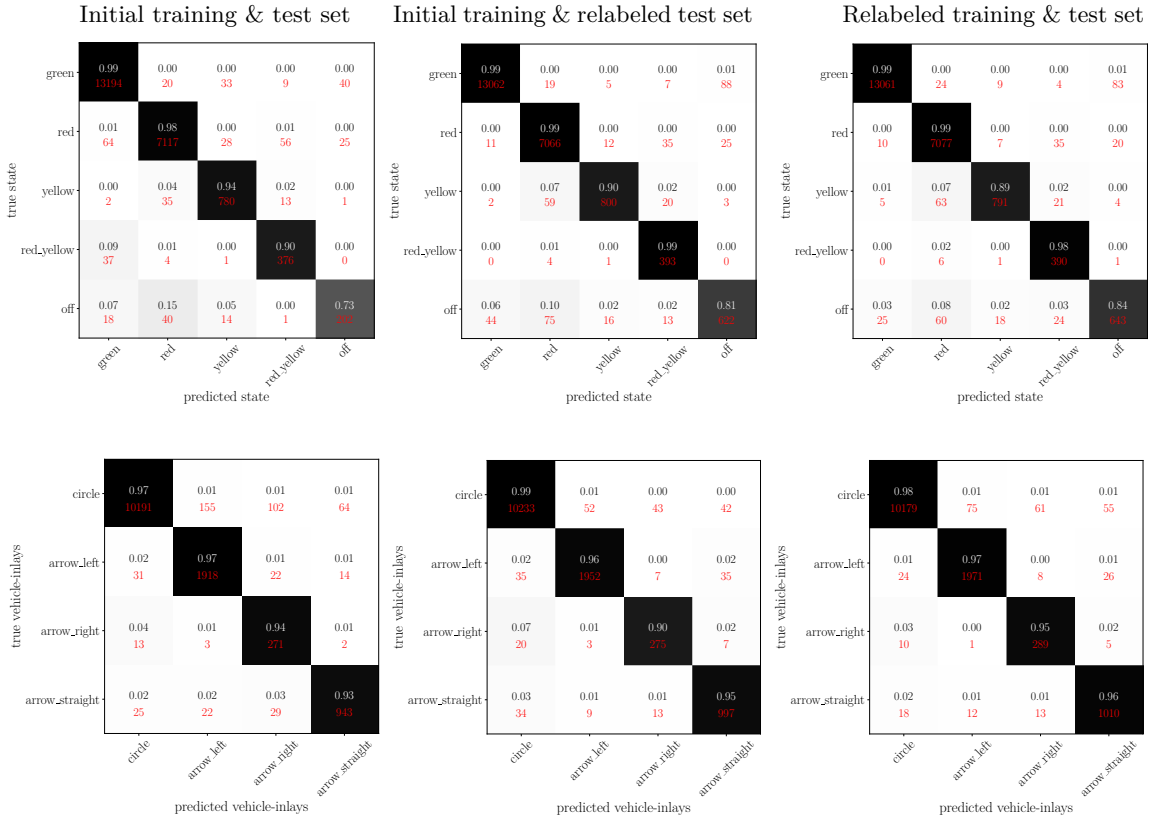


Figure 8.3: State and inlay classification performance evaluated on inital test set, relabeled test set and retrained and evaluated on relabeled training and test set.

Both networks are evaluated on the relabeled test set.

Detection Results

Figure 8.2 shows the performance of the network trained on the relabeled training set in green. Training set relabeling has practically no effect on the system performance. For FPPI values larger 10^{-2}, no impact is visible, whereas for FPPI smaller 10^{-2} the network trained on the initial training set even performs better. A small amount of labeling errors does not substantially downgrade the system performance. Slight performance drops may rather be explained by a higher amount of corner cases created from relabeling. The probability is high, that initially overlooked traffic lights are also hard edge-cases instead of obvious cases. Forcing a classifier to learn hard and rare cases often comes along with a decrease for the common cases.

Attribute Classification Results

State and vehicle inlay classification results after retraining with the relabeled training set are shown in the right column of Figure 8.3. State performance only shows slight recall changes, which seem to be caused by common variations when retraining networks. The vehicle inlays confusion matrix shows slight improvements, reaching more than 95 percent recall for all four classes.

8.3 DriveU Traffic Light Dataset v2

In the course of this thesis, the relabeled data generated within this chapter is made publicly available for other researchers. The new annotation files are accessible after registration (`http://traffic-light-data.com/`). Table 8.2 shows the number of training and test labels of the initial (v1) and relabeled (v2) dataset. Relabeling increases the number of labels by more than 25 percent.

Table 8.2: DriveU Traffic Light Dataset v2 increases by roughly 25%.

	# training labels	**# test labels**
DTLD v1	159 902	72 137
DTLD v2	200 263	91 982

Chapter 9

Discussion and Perspectives

This thesis has proposed a novel, CNN-based system for traffic light detection as well as state and pictogram classification. In this context, Chapter 5 presented multiple, novel adaptations to Single Shot Detection, as well as extensive investigations in order to optimize the detection performance. One key contribution is the design of prior boxes. This includes the introduced *weighted k-IoU clustering particularly tailored to the data imbalance problem.* By this proceed, an optimal adaptation of prior box dimensions to the expected object dimensions is guaranteed. With regard to the design of prior boxes, *the introduced adaptive prior box locations are a second contribution.* Adaptive locations overcome the fixed locations being state of the art, which turned out to be a bottleneck for small object detection. Evaluations emphasized the effectiveness of the proceed and substantiate the theoretical expectations, that *in particular smaller objects benefit from adaptive locations.* Moreover, all results are independent of the object class itself, making it applicable to other object detection problems as well.

Adaptations regarding the network architecture turned out to be a further effective contribution. Instead of the VGG-16 core network used in the original approach, an Inception-v3 architecture is utilized within this thesis. *The inception architecture combines both, small and large receptive fields, which is beneficial for traffic light detection.* Moreover, various layers of the network are investigated in order to determine the optimal prediction layers with regard to performance and runtime.

A particular focus of this thesis is on the evaluation of the presented system. On the one hand, the large-scale DriveU Traffic Light Dataset as the largest published database for traffic light recognition is utilized for all evaluations. DTLD was published in the course of this thesis and is already used by a large number of researchers in this field. On the other hand, extensive investigations and evaluations are performed, such as the impact of data amount, image and ground truth preprocessing, choice of prediction layers and prior boxes configuration and others. The extent of

evaluations exceeds state of the art research and is thus giving a more detailed and application-oriented view on the system characteristics.

Beyond traffic light detection, Chapter 6 addressed CNN-based state and pictogram classification. Hereby, two methods are developed. The first is a multi-task network in the form of an extension of Single Shot Detection by utilizing two additional layers predicting state and pictogram confidences, respectively. Therefore, the overall loss function is extended. This method can be referred to as an implicit classification. Evaluations on state classification showed, that the performance on active states (red, yellow, green) is superior to the non-active state off. In a few image frames, LED effects cause active lights with low intensity, which look similar to off lights. In this respect, the borders between off and active lights can be weak, which even challenges humans as well as machine learning. However, the large majority of test samples are correctly classified and misclassifications can be summarized to only appear in a sporadic manner.

With regard to pictogram classification, implicit SSD classification is more challenging to improve. It turned out, that despite many optimization, filtering and augmentation types, high recall rates are hard to reach in order to differentiate between the arrow directions. Performance on pedestrian/tram traffic lights is much higher, which indicates that these classes benefit from high receptive fields. From a human perspective, pedestrian and tram lights are commonly classifiable without already recognizing the pictogram itself, yet solely from context or road topologies. This is much harder for arrow signs. Implicit classification has one main drawback - which in parallel is the largest advantage for the detection task. High receptive fields typically generated within deep networks support detection, but lack of local, pixel-level information, which is mandatory for pictogram classification.

This issue is the central motivation for the second methodology developed within this thesis. A two-stage classification is introduced using a light-weight CNN, which operates on the region of interest obtained from the detection network. With regard to state classification performance, only slight improvements are reached, also caused by the already high baseline performance of the implicit classification. However, with regard to pictogram classification performance, a significant improvement is reached. In particular, a simplified data augmentation and the freedom in network design turned out to yield improvements. The lack of right arrow ground truths can be compensated by data augmentation. With regard to network design, utilizing small feature kernel strides in the early network layers turned out to be the most effective adaptation. However, the performance on pedestrian and tram lights drops compared to the implicit classification. This strengthens the already mentioned presumption, that these classes benefit from context information.
Consequently, this thesis proposed to use a hybrid approach for pictogram recognition. Within SSD, the so called target road user is predicted, which differentiates only

between the classes vehicle, pedestrian, tram and bicycle. In case of vehicle lights, a subsequent light-weight network differentiates between a circular and arrow-shaped pictograms in a second step. This combines the best of two worlds and optimizes the overall, class-wise recall for pictogram classification.

The topic of confidence calibration is addressed in Chapter 7 and shows, that detection as well as attribute confidences are poorly calibrated. Moreover, the output confidences are not uniformly distributed, which is tackled by determining bin boundaries by equalizing the number of samples per bin. By this, the expected calibration error is reduced by more than 6 percent. When calibrating attribute confidences a second side effect is observable. Calibration affects the classification results, because each class confidence is independently calibrated. The classification result turned out to be more balanced and underrepresented classes, such as off or red_yellow, are strengthened.

During evaluations of detection and attribute classification, the dataset quality turned out to be the main bottleneck. Background of this finding is not a poor dataset quality in general, yet a very powerful recognition system challenging human performance. When evaluating operating points at very low false positives per image values, non-labeled (overlooked) objects have to be minimized. For instance, evaluating operating points at FPPI=10^{-2} requires clearly less than one label error each 100 test images. When observing the false positives determined by the evaluation system as well as misclassified states/pictograms, a high number of label errors occur. This is discussed in Chapter 8, in which an automated relabeling system is presented, which uses the recognition system to automatically generate proposals being verified by human labelers and updated into the database. Evaluation showed, that relabeling affects results at low FPPI values, while for evaluating high FPPI values the requirement on dataset quality decreases. In consequence, dataset quality is one of the most important topics when developing modern object detectors with rising performance, because evaluability of object detectors with rising performance conditions a higher dataset quality. Contrary to evaluation, a small amount of annotation errors is neglectable when training convolutional neural networks. Experiments showed, that there is no significant impact of training set quality.

The final results are very promising. The system is able to detect 98% of all relevant traffic lights larger 10 pixels at only one false positive each 100 images. At this operating point, state classification reaches more than 99 percent for the active states red, green and red-yellow and more than 95 percent recall for the pictograms circle, arrow_left, arrow_straight, arrow_right, pedestrian and tram.

Besides the already mentioned dataset quality, further labeling efforts could help to improve the vision-based traffic light recognition. This includes tags expressing a non-decipherability of states/pictograms. The current implementation filters out

small objects from training based on size to avoid a utilization of unrecognizable pictograms. However, visual inspection showed, that also larger objects may not be recognizable due to many additional effects, such as viewpoint, occlusion or light conditions. Moreover, evaluation contains non-decipherable samples, which distort the results. Those samples should neither be used for training nor for evaluation. Further benefits can be expected from annotating the traffic light bulb in addition to the housing. Extending Single Shot Detection to predict light bulb coordinates can be efficiently used for the two-stage methodology. Instead of classifying the traffic light housing bounding box, the traffic light bulb box can be classified, which avoids classifying on background regions containing no relevant information.

Looking beyond traffic light detection and state as well as pictogram recognition, a traffic light assistant requires many additional components. This includes tracking, finding stop lines, detect and handle flashing lights, utilize map information and others. Combining all sources of information may be required to solve one of the most important and critical task of this field - namely determining the relevant traffic lights. Only by this, the traffic lights controlling the route of the ego vehicle can be determined and the vehicle can initiate an appropriate action by decelerating or passing the intersection.

Appendix A

Decompanding Curve

Figure A.1 illustrates the HDR compression curve mentioned in Section 5.3.2. The camera sensor is operated in HDR mode. Two exposures are performed and combined to a linearized 16 bit value for each pixel. This 16 bit image is compressed to 12 bits according to the figure. Using the 16-12 bit relationship, the 16 bit image can be reconstructed from the stored 12 bit image. Further Details see [ONS17].

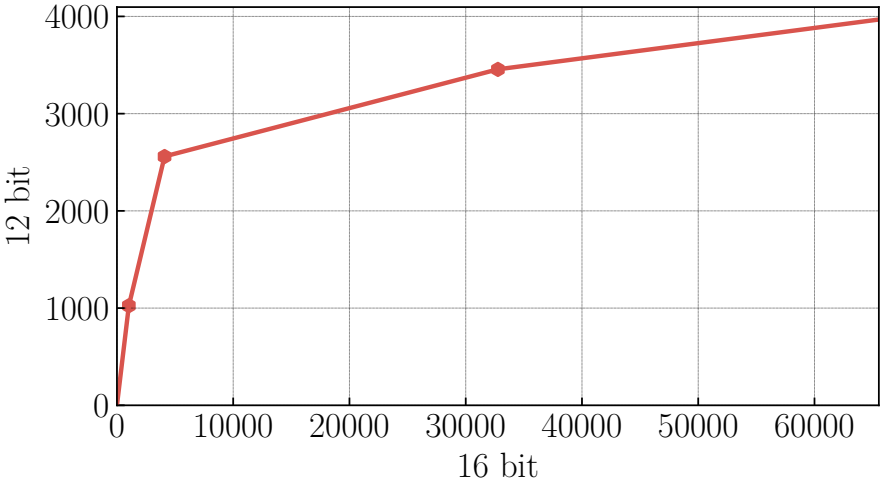


Figure A.1: Image sensor companding curve which describes the mapping from 16 to 12 bit and vice versa.

Appendix B

Supplementary Material on Prior Boxes

B.1 Mathematical Relationship between Intersection Over Union and Prior Box to Ground Truth Misalignment

The intersection over union between the prior box and ground truth can be formulated as

$$\text{IoU} = \frac{|p \wedge g|}{|p \vee g|} = \frac{|p \wedge g|}{|p| + |g| - |p \wedge g|} \tag{B.1}$$

Under the assumption of two rectangles with equal size (|p|=|g|), this simplifies to

$$\text{IoU} = \frac{|p \wedge g|}{2|p| - |p \wedge g|}. \tag{B.2}$$

Assuming a percentage error $\mathcal{E}^s$ (with respect to width/height, respectively) in x and y direction leading to

$$\text{IoU} = \frac{(1 - \mathcal{E}^s)^2}{2 - (1 - \mathcal{E}^s)^2}. \tag{B.3}$$

The main reason for this assumption is, that according to the IoU calculation higher step sizes can be assumed into the direction of the long side of the rectangle. The ratio of the step sizes should be equal to the ratio of the prior boxes ($\frac{p^{\text{h}}}{p^{\text{w}}} = \frac{\Delta p^{\text{x}}}{\Delta p^{\text{y}}}$).

Mathematically, the denominator may not be zero

$$2 - (1 - \mathcal{E}^s)^2 \neq 0 \qquad \Leftrightarrow \qquad \mathcal{E}^s \neq 1 + \sqrt{2} \tag{B.4}$$

which is given for the assumption

$$0 \leq \mathcal{E}^s \leq 1. \tag{B.5}$$

Transforming Equation (B.3) leads to

$$(1 - \mathcal{E}^s)^2 = \frac{2\text{IoU}}{1 + \text{IoU}} \tag{B.6}$$

which can be solved as

$$\mathcal{E}^s = 1 - \sqrt{\frac{2\text{IoU}}{1 + \text{IoU}}} \tag{B.7}$$

under the assumption of Equation (B.5).

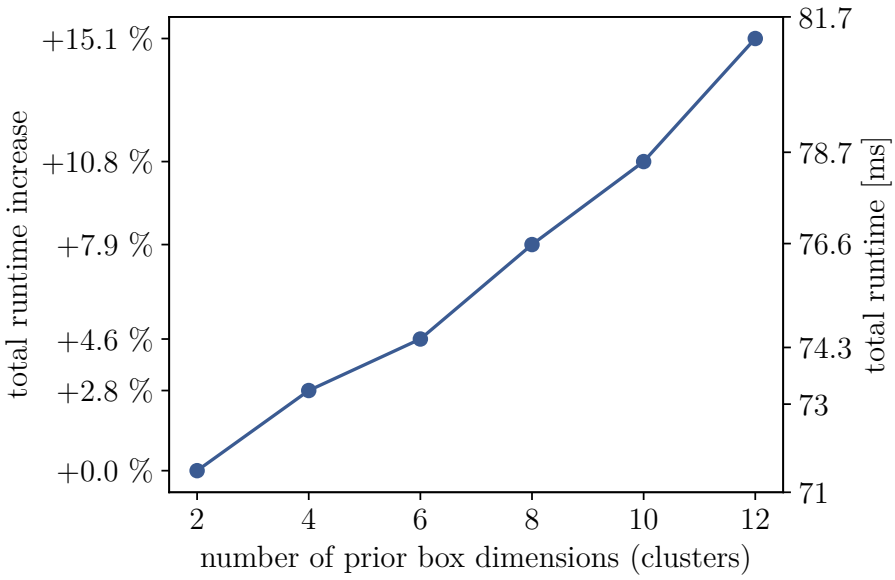


Figure B.1: Relationship between prior box dimensions and runtime. Runtimes are related to a single prediction layer (inception_b).

B.2 Prior Box Dimensions vs. Runtime

Figure B.1 shows the relationship between total network runtime and the number of prior boxes obtained from k-IoU clustering. A linear effect can be observed, which adds approximately 1 millisecond of runtime per box. Values are determined in a single prediction layer setup using a feature map of the inception_b block and adaptive locations.

B.3 Prior Box Locations

Table B.1 illustrates the adaptive prior box locations for each prior box dimension $\underline{\mu}_k$ with respect to different prediction feature layer choices. This table refers to the illustrations in Figure 5.13 and investigations presented in Section 5.3.5. λ^x and λ^y refer to the locations in x and y inside one cell, normalized to 1, respectively. For instance, $\lambda^x = [0, 1/2]^\top$ describes two prior box locations at the cell edge and center in x-direction. Intuitively, the amount of prior box locations increases with rising network depth (inception_a -> inception_b -> inception_c), because the cell size increases as well. Increasing cell sizes can be compensated by more adaptive locations leading to an approximately constant absolute prior box stride in pixels. Because most prior boxes have an aspect ratio close to three, λ^x is mostly larger than λ^y. For prior boxes with an aspect ratio close to one ($\underline{\mu}_1$ & $\underline{\mu}_4$), it yields $\lambda^x \approx \lambda^y$.

Table B.1: Prior box locations for all prior box sizes for predicting objects in inception_a, b or c, respectively. The relative values show the locations inside one cell (of normed size [1.0, 1.0]).

		$\underline{\boldsymbol{\mu}}_1$	$\underline{\boldsymbol{\mu}}_2$	$\underline{\boldsymbol{\mu}}_3$	$\underline{\boldsymbol{\mu}}_4$	$\underline{\boldsymbol{\mu}}_5$	$\underline{\boldsymbol{\mu}}_6$	$\underline{\boldsymbol{\mu}}_7$	$\underline{\boldsymbol{\mu}}_8$
	p^w	11.12	11.55	14.93	19.85	23.13	26.40	32.74	49.10
	p^h	13.52	35.26	57.17	19.86	76.91	42.08	111.88	103.87
	p^h/p^w	1.22	3.05	3.83	1.00	3.33	1.59	3.42	2.12
INCEPTION_A	λ^x	$\begin{bmatrix}0\\1/2\end{bmatrix}$	$\begin{bmatrix}0\\1/2\end{bmatrix}$	$\begin{bmatrix}0\\1/2\end{bmatrix}$	$[1/2]$	$[1/2]$	$[1/2]$	$[1/2]$	$[1/2]$
	λ^y	$\begin{bmatrix}0\\1/2\end{bmatrix}$	$[1/2]$	$[1/2]$	$[1/2]$	$[1/2]$	$[1/2]$	$[1/2]$	$[1/2]$
INCEPTION_B	λ^x	$\begin{bmatrix}0\\1/4\\2/4\\3/4\end{bmatrix}$	$\begin{bmatrix}0\\1/4\\2/4\\3/4\end{bmatrix}$	$\begin{bmatrix}0\\1/3\\2/3\end{bmatrix}$	$\begin{bmatrix}0\\1/2\end{bmatrix}$	$\begin{bmatrix}0\\1/2\end{bmatrix}$	$\begin{bmatrix}0\\1/2\end{bmatrix}$	$\begin{bmatrix}0\\1/2\end{bmatrix}$	$[1/2]$
	λ^y	$\begin{bmatrix}0\\1/4\\2/4\\3/4\end{bmatrix}$	$\begin{bmatrix}0\\1/2\end{bmatrix}$	$[1/2]$	$\begin{bmatrix}0\\1/2\end{bmatrix}$	$[1/2]$	$[1/2]$	$[1/2]$	$[1/2]$
INCEPTION_C	λ^x	$\begin{bmatrix}0\\1/8\\2/8\\3/8\\4/8\\5/8\\6/8\\7/8\end{bmatrix}$	$\begin{bmatrix}0\\1/8\\2/8\\3/8\\4/8\\5/8\\6/8\\7/8\end{bmatrix}$	$\begin{bmatrix}0\\1/6\\2/6\\3/6\\4/6\\5/6\end{bmatrix}$	$\begin{bmatrix}0\\1/4\\2/4\\3/4\end{bmatrix}$	$\begin{bmatrix}0\\1/4\\2/4\\3/4\end{bmatrix}$	$\begin{bmatrix}0\\1/3\\2/3\end{bmatrix}$	$\begin{bmatrix}0\\1/3\\2/3\end{bmatrix}$	$\begin{bmatrix}0\\1/2\end{bmatrix}$
	λ^y	$\begin{bmatrix}0\\1/6\\2/6\\3/6\\4/6\\5/6\end{bmatrix}$	$\begin{bmatrix}0\\1/3\\2/3\end{bmatrix}$	$\begin{bmatrix}0\\1/2\end{bmatrix}$	$\begin{bmatrix}0\\1/4\\2/4\\3/4\end{bmatrix}$	$\begin{bmatrix}0\\1/2\end{bmatrix}$	$\begin{bmatrix}0\\1/2\end{bmatrix}$	$[1/2]$	$[1/2]$

Appendix C

Further Investigations

C.1 False Positive Categorization Using Scene Segmentation

In order to perform an automatic categorization of false positives, scene segmentation can be used. Therefore, the object class, on which false positives are located, are collected. For this purpose, PSPNet [ZSQ+17], a powerful scene segmentation approach, is used. Figure C.1 shows the result after applying PSPNet on a DTLD scene. Two white rectangles inside the image show two false positives in the present scene. With use of the segmentation image, these false positives can be classified as a rear light of a car and vegetation (tree).

Applying this to all false positives, a pie chart can be created, see Figure C.2. The highest amount of false positives is caused by buildings, which includes window facades, bus stops, shop signs and other sources of false positives. The second error cluster are caused by vehicles including buses, cars and others. For this class, rear lights and mirror images of traffic lights in vehicle windows cause most false positives. The class traffic light groups all labeling errors in DTLD or also inaccuracies in detection causing the prediction to be misaligned. Other sources of false positives are regions with a high number of dark regions, but also bright blobs such as vegetation (especially treetops) or fences.

As a matter of fairness, it has to be mentioned, that the scene segmentation approach also shows inaccuracies causing errors in the false positive categorization. However, the presented results show a tendency corresponding to the visual impression, that most false positives are caused by buildings, vegetation, pole-like structures and other vehicles. With regard to traffic light features, all regions containing dark, rectangular regions with bright blobs are potential areas, in which false positives likely appear.



Figure C.1: Automatic false positive categorization using PSPNet. The scene segmentation image can be used to determine, on which object classes false positives typically appear. The image shows two false positives (white) on car and vegetation, respectively.

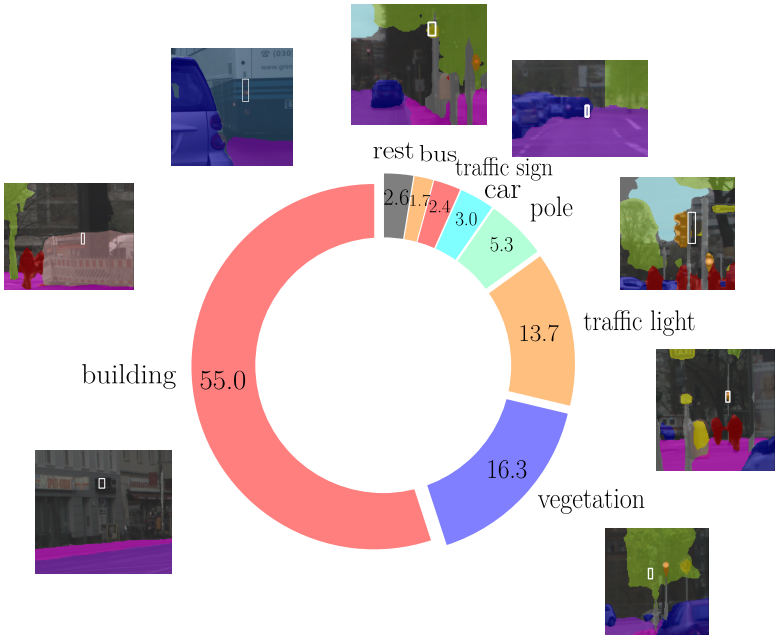


Figure C.2: Sources of false positives categorized according to the most prominent object classes. Most false positives are located on buildings, vegetations, poles and vehicles.

C.2 Attribute Classification Performance with Respect to Position and Size

Visual inspection of pictograms showed a clear relationship between object resolution and visibility/recognisability, see Figure 6.2. Depending on the object class and its respective spatial frequency, resolution has a different degree of impact. This statement is based on human impressions and therefore not necessarily transferable to machine learning. Consequently, the impact of object resolution on the system performance has to be empirically investigated.

Figure C.3 illustrates the classification recall with respect to all seven pictogram classes. A mixed picture is shown. While pictograms responsible for straight traffic (mostly circle and arrow_straight) are improving for rising object resolution, recall on pedestrian, bicycle and tram traffic lights decreases in close range. For arrow_left and arrow_right a mixed picture is shown. Performance is optimal in moderate range, whereas for small and very high resolutions performance drops. Drops for at large distances are caused by small resolutions not sufficient for arrow classification. This explanation is not applicable for tram or pedestrian because those classes can be more easily classified from pure context information. In close range, the lateral angle between the vehicle, which is mostly driving straight, and arrow signs as well as pedestrian/tram lights increases. This aggravates visibility, because the active light is often occluded by sun visors. The smaller the distance, the larger occlusion effect. In consequence, small resolutions at far distances as well as viewing angle issues in close range are challenging.

A slight impact of object position in x and y-direction on classification recall is observable in Figure C.4. The absolute recall for each bin is shown in red, whereas black numbers at the bin bottom show the absolute number of labels assigned to the respective bin. Recall values are slightly decreasing at the vertical image edges. This may be caused by the generally poor performance on particularly right arrows which likely occur in the right part of the image. An additional effect is background, which is likely more beneficial in the image center. Traffic lights occuring in the image center are in front of sky, which simplifies recognition.

With regard to the y-position (Figure C.4b), a slight drop can be observed for rising image row number. This generally correlates to the object depth and inversely correlates to object resolution. Because pictogram recognition is more challenging for low object resolutions, this finding is intuitive. Please note, that the performance gain shown in the last two bins is likely an statistical issue (12 and 1 sample in both bins, respectively).

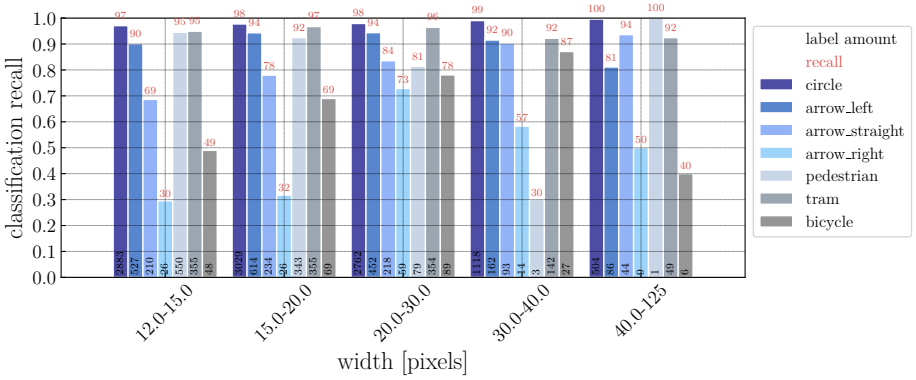
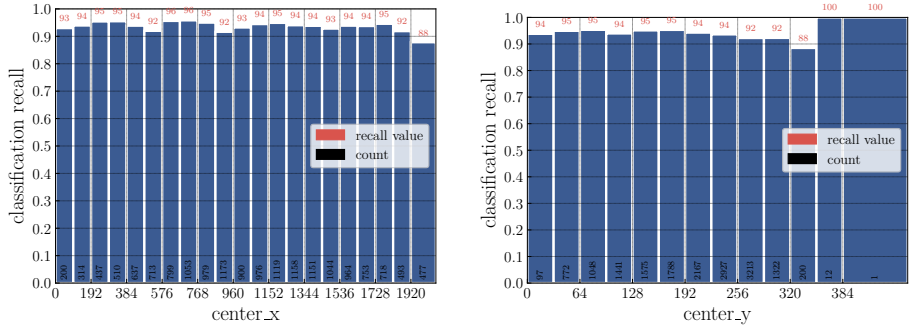


Figure C.3: Impact of object resolution on the classification recall. The best overall performance is reached at medium resolution/distances. Intuitively, very low resolutions are not benefitial. However, very large resolutions are also not optimal for some classes. Very high resolution often corresponds to adverse viewpoints in close range scenarios, in which the viewing angle increases. This leads to occlusion of pictograms, e.g. by the installed sun visors.



(a) Performance w.r.t x-position

(b) Performance w.r.t y-position

Figure C.4: Pictogram classification recall with respect to the object center position. Significant drops are observable at the right vertical image edge (many right arrows) and for increasing y-values (correlating to object distance).

Acronyms

ACE	Average Calibration Error	141
AUC	Area Under Curve	46
BSTLD	Bosch Small Traffic Lights Dataset	17
CNN	Convolutional Neural Network	ix
COCO	Common Objects in COntext	35
DTLD	DriveU Traffic Light Dataset	x
ECE	Expected Calibration Error	140
FPPI	False Positives Per Image	46
GPU	Graphics Processing Unit	22
HD	High Definition	53
HDR	High Dynamic Range	79
HOG	Histograms of Oriented Gradients	15
HSV	Hue Saturation Value	83
IoU	Intersection Over Union	43, 44
IQR	Interquartile Range	78
kNN	k-Nearest Neighbors	14
LAMR	Log-Average Miss Rate	46
LARA	La Route Automatisee	33
LBP	Local Binary Patterns	15
LDA	Linear Discriminant Analysis	20
LED	Light-Emitting Diode	10
LiDAR	Light Detection and Ranging	17

LISA Laboratory for Intelligent and Save Automobiles 17

NMS Non-Maximum Suppression 59

PCA Principal Component Analysis 20

ROC Receiver Operating Characterstic 46
RoI Region of Interest 28

SGM Semi Global Matching 38
SSD Single Shot Detection/Detector ix
SVM Support Vector Machine 14

US United States 4

V2I Vehicle-to-Infrastructure 1
VIVA Intelligent Vehicles and Application 33

YOLO You Only Look Once 15

List of Symbols

Mathematical Notation

x	scalar
$\underline{x}$	vector
$\mathbf{x}$	matrix

Evaluation

$\mathcal{T}_{\text{IoU}}$	intersection over union threshold
ω	true class
$\hat{\omega}$	predicted class
$\mathcal{C}(\omega, \hat{\omega})$	confusion matrix
$\mathcal{R}^{\text{micro}}$	micro recall
$\mathcal{R}^{\text{macro}}$	macro recall
η_{micro}	micro recall weight
η_{macro}	macro recall weight
$\mathcal{R}$	combined recall (micro & macro)

Single Shot Detection

χ^w, χ^h	feature cell dimensions
$\mathcal{F}^w, \mathcal{F}^h$	feature map dimensions
$\mathcal{I}^w, \mathcal{I}^h$	camera image dimensions
$\mathcal{I}$	iteration number
$\underline{z}$	logit vector (unnormalized network output)
$\delta^x, \delta^y, \delta^w, \delta^h$	misalignment in position and dimensions
$\mathcal{E}^s$	percentage misalignment between ground truth and prior box
b^x, b^y, b^w, b^h	prediction bounding box centre & dimensions
l^x, l^y, l^w, l^h	offset prediction centre & dimensions
p^x, p^y, p^w, p^h	prior box centre & dimensions
λ^x, λ^y	prior box location in x and y
$\mathbf{\Lambda}$	normalized prior box locations vector of one feature cell

r	receptive field

Evaluation Scenarios

DS1	all non-occluded, frontal traffic lights larger 8 pixels width
DS2	all non-occluded, frontal traffic lights larger 10 pixels width
DS3	all non-occluded, relevant traffic lights larger 8 pixels width
DS4	all non-occluded, relevant traffic lights larger 10 pixels width
DS_PICTO	all non-occluded, frontal traffic lights larger 12 pixels width
DS_STATE	all non-occluded, vehicle traffic lights larger 8 pixels width

Loss Functions

$\mathcal{L}_{\text{conf}}$	confidence loss
$\mathcal{L}_{\text{loc}}$	localization loss
α	localization loss/metric weight
$\mathcal{L}_{\text{picto}}$	pictogram loss
γ	pictogram loss/metric weight
$\mathcal{L}_{\text{state}}$	state loss
β	state loss/metric weight
$\mathcal{L}_{\text{total}}$	combined loss value

Confidence Calibration

$\mathcal{A}(B)$	bin accuracy
B	confidence bin
$\mathcal{C}$	calibration value
$\mathcal{P}$	perfect calibration
q	bin quantile

Bibliography

[ANS19] Araujo, André; Norris, Wade; and Sim, Jack: *Computing Receptive Fields of Convolutional Neural Networks*. 2019. URL: `https://distill.pub/2019/computing-receptive-fields` (visited on 03/15/2021).

[Audi19] *Audi vernetzt sich mit Ampeln in Deutschland*. 2019. URL: `https://www.audi-mediacenter.com/de/pressemitteilungen/audi-vernetzt-sich-mit-ampeln-in-deutschland-11649` (visited on 03/19/2021).

[BCKW19] Bao, Cuizhu; Chen, Chen; Kui, HaiLin; and Wang, Xiaoyang: *Safe Driving at Traffic Lights: An Image Recognition Based Approach*. In: *2019 20th IEEE International Conference on Mobile Data Management (MDM)*, pages 112–117, 2019.

[BKFG19] Braun, Markus; Krebs, Sebastian; Flohr, Fabian; and Gavrila, Dariu M.: *EuroCity Persons: A Novel Benchmark for Person Detection in Traffic Scenes*. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 41, no. 8, pages 1844–1861, 2019.

[BMP15] Barnes, Dan; Maddern, Will; and Posner, Ingmar: *Exploiting 3D semantic scene priors for online traffic light interpretation*. In: *2015 IEEE Intelligent Vehicles Symposium (IV)*, pages 573–578, 2015.

[BNB17] Behrendt, Karsten; Novak, Libor; and Botros, Rami: *A deep learning approach to traffic lights: Detection, tracking, and classification*. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1370–1377, 2017.

[BRD17] Bach, Martin; Reuter, Stephan; and Dietmayer, Klaus: *Multi-camera traffic light recognition using a classifying Labeled Multi-Bernoulli filter*. In: *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 1045–1051, 2017.

[BSD18] Bach, Martin; Stumper, Daniel; and Dietmayer, Klaus: *Deep Convolutional Traffic Light Recognition for Automated Driving*. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 851–858, 2018.

[CGL12] Cai, Zixing; Gu, Mingqin; and Li, Yi: *Real-time arrow traffic light recognition system for intelligent vehicle*. In: *Proceedings of the 2012 International Conference on Image Processing, Computer Vision, and Pattern Recognition, IPCV 2012*, volume 2, pages 848–854, 2012.

[CH16] Chen, Zhilu and Huang, Xinming: *Accurate and Reliable Detection of Traffic Lights Using Multiclass Learning and Multiobject Tracking*. In: *IEEE Intelligent Transportation Systems Magazine*, volume 8, no. 4, pages 28–42, 2016.

[CHL+11] Chiang, Cheng Chin; Ho, Ming Che; Liao, Hong Sheng; Pratama, Andi; and Syu, Wei Cheng: *Detecting and recognizing traffic lights by genetic approximate ellipse detection and spatial texture layouts*. In: *International Journal of Innovative Computing, Information and Control*, volume 7, no. 12, pages 6919–6934, 2011.

[COR+16] Cordts, Marius; Omran, Mohamed; Ramos, Sebastian; et al.: *The Cityscapes Dataset for Semantic Urban Scene Understanding*. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3213–3223, 2016.

[CPC16] Canziani, Alfredo; Paszke, Adam; and Culurciello, Eugenio: *An Analysis of Deep Neural Network Models for Practical Applications*. In: *CoRR*, pages 1–7, 2016.

[DCS12] Diaz-Cabrera, Moises; Cerri, Pietro; and Sanchez-Medina, Javier: *Suspended traffic lights detection and distance estimation using color features*. In: *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pages 1315–1320, IEEE, Anchorage, AK, 2012.

[DN09a] De Charette, Raoul and Nashashibi, Fawzi: *Real time visual traffic lights recognition based on spot light detection and adaptive traffic lights templates*. In: *IEEE Intelligent Vehicles Symposium, Proceedings*, pages 358–363, 2009.

[DN09b] De Charette, Raoul and Nashashibi, Fawzi: *Traffic light recognition using image processing compared to learning processes*. In: *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 333–338, 2009.

[DWSP09] Dollar, Piotr; Wojek, Christian; Schiele, Bernt; and Perona, Pietro: *Pedestrian detection: A benchmark*. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 304–311, 2009.

[DWSP12] Dollar, Piotr; Wojek, Christian; Schiele, Bernt; and Perona, Pietro: *Pedestrian Detection: An Evaluation of the State of the Art*. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 34, no. 4, pages 743–761, 2012.

[EGW+10] Everingham, Mark; Gool, Luc; Williams, Christopher K.; Winn, John; and Zisserman, Andrew: *The Pascal Visual Object Classes (VOC) Challenge*. In: *International Journal of Computer Vision*, volume 88, no. 2, pages 303–338, 2010.

[ESTA14] Erhan, Dumitru; Szegedy, Christian; Toshev, Alexander; and Anguelov, Dragomir: *Scalable Object Detection Using Deep Neural Networks*. In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2155–2162, 2014.

[Faw06] Fawcett, Tom: *An introduction to ROC analysis*. In: *Pattern Recognition Letters*, volume 27, no. 8. ROC Analysis in Pattern Recognition, pages 861–874, 2006.

[FD16] Fregin, Andreas and Dietmayer, Klaus: *A closer look on traffic light detection evaluation metrics*. In: *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 971–975, 2016.

[FD18] Fregin, Andreas and Dietmayer, Klaus: *Learning Traffic Light Colors*. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 709–715, 2018.

[FGSS18] Fernández, Carlos; Guindel, Carlos; Salscheider, Niels-Ole; and Stiller, Christoph: *A Deep Analysis of the Existing Datasets for Traffic Light State Recognition*. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 248–254, 2018.

[FMD17a] Fregin, Andreas; Müller, Julian; and Dietmayer, Klaus: *Feature detectors for traffic light recognition*. In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 339–346, 2017.

[FMD17b] Fregin, Andreas; Müller, Julian; and Dietmayer, Klaus: *Three ways of using stereo vision for traffic light recognition*. In: *IEEE Intelligent Vehicles Symposium, Proceedings*, no. June 2017, pages 430–436, 2017.

[FMKD18] Fregin, Andreas; Müller, Julian; Kressel, Ulrich; and Dietmayer, Klaus: *The DriveU Traffic Light Dataset: Introduction and Comparison with Existing Datasets*. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3376–3383, 2018.

[Fre21] Fregin, Andreas: *Systemischer Ansatz zur Detektion von Lichtsignalanlagen in Kamerabildern*. PhD thesis, Universität Ulm, 2021.

[FU11] Fairfield, N. and Urmson, C.: *Traffic light mapping and detection*. In: *2011 IEEE International Conference on Robotics and Automation*, pages 5421–5426, 2011.

[GDDM14] Girshick, Ross; Donahue, Jeff; Darrell, Trevor; and Malik, Jitendra: *Rich feature hierarchies for accurate object detection and semantic segmentation.* In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014.

[GF07] Gehrig, Stefan. K. and Franke, Uwe: *Improving Stereo Sub-Pixel Accuracy for Long Range Stereo.* In: *2007 IEEE 11th International Conference on Computer Vision*, pages 1–7, 2007.

[Gil92] Gillespie, Thomas D: *Fundamentals of vehicle dynamics.* Volume 400. Society of automotive engineers Warrendale, PA, 1992, pages 46–47.

[Gir15] Girshick, Ross: *Fast R-CNN.* In: *Proceedings of the IEEE International Conference on Computer Vision*, volume 2015 International Conference on Computer Vision, ICCV 2015, pages 1440–1448, 2015.

[GJX+10] Gong, Jianwei; Jiang, Yanhua; Xiong, Guangming; et al.: *The recognition and tracking of traffic lights based on color segmentation and CAMSHIFT for intelligent vehicles.* In: *2010 IEEE Intelligent Vehicles Symposium*, pages 431–435, 2010.

[GPR20] Gupta, Chirag; Podkopaev, Aleksandr; and Ramdas, Aaditya: *Distribution-free binary classification: prediction sets, confidence intervals and calibration.* In: *arXiv*, pages 1–33, 2020.

[GPSW17] Guo, Chuan; Pleiss, Geoff; Sun, Yu; and Weinberger, Kilian Q.: *On Calibration of Modern Neural Networks.* In: *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, 1321?1330, JMLR.org, Sydney, NSW, Australia, 2017.

[HBS17] Hosang, Jan; Benenson, Rodrigo; and Schiele, Bernt: *Learning Non-maximum Suppression.* In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6469–6477, 2017.

[HGDG17] He, Kaiming; Gkioxari, Georgia; Dollar, Piotr; and Girshick, Ross: *Mask R-CNN.* In: *Proceedings of the IEEE International Conference on Computer Vision*, volume 2017-Octob, pages 2980–2988, 2017.

[Hir08] Hirschmüller, Heiko: *Stereo Processing by Semiglobal Matching and Mutual Information.* In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 30, no. 2, pages 328–341, 2008.

[HMUI15] Haltakov, Vladimir; Mayr, Jakob; Unger, Christian; and Ilic, Slobodan: *Semantic Segmentation Based Traffic Light Detection at Day and at Night.* In: Gall, Juergen; Gehler, Peter; and Leibe, Bastian (editors): *Pattern Recognition*, pages 446–457, Springer International Publishing, Cham, 2015.

[HZRS16] He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; and Sun, Jian: *Deep residual learning for image recognition*. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2016-December, pages 770–778, 2016.

[Its15] Itseez: *Open Source Computer Vision Library*. 2015.

[JPM+16] Jensen, Morten Bornø; Philipsen, Mark Philip; Møgelmose, Andreas; Moeslund, Thomas Baltzer; and Trivedi, Mohan Manubhai: *Vision for Looking at Traffic Lights: Issues, Survey, and Perspectives*. In: *IEEE Transactions on Intelligent Transportation Systems*, pages 1–16, 2016.

[JSD+14] Jia, Yangqing; Shelhamer, Evan; Donahue, Jeff; et al.: *Caffe: Convolutional Architecture for Fast Feature Embedding*. In: *Proceedings of the 22Nd ACM International Conference on Multimedia*, pages 675–678. *MM '14*, ACM, Orlando, Florida, USA, 2014.

[JZM16] John, Vijay; Zheming, Lyu; and Mita, Seiichi: *Robust traffic light and arrow detection using optimal camera parameters and GPS-based priors*. In: *Proceedings of 2016 Asia-Pacific Conference on Intelligent Robot Systems, ACIRS 2016*, no. July, pages 204–208, 2016.

[KCH+18] Kim, Jinkyu; Cho, Hyunggi; Hwangbo, Myung; et al.: *Deep Traffic Light Detection for Self-driving Cars from a Large-scale Dataset*. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 280–285, 2018.

[KHJ18] Kim, Hyunkoo; H. Park, Ju; and Jung, Ho-Youl: *An Efficient Color Space for Deep-Learning Based Traffic Light Recognition*. In: *Journal of Advanced Transportation*, volume 2018, pages 1–12, 2018.

[KYHJ19] Kim, Hyunkoo; Yoo, Kook-Yeol; H Park, Ju; and Jung, Ho-Youl: *Traffic Light Recognition Based on Binary Semantic Segmentation Network*. In: *Sensors*, volume 19, page 1700, 2019.

[LADT11] Levinson, Jesse; Askeland, Jake; Dolson, Jennifer; and Thrun, Sebastian: *Traffic light mapping, localization, and state detection for autonomous vehicles*. In: *Proceedings - IEEE International Conference on Robotics and Automation*, pages 5784–5791, 2011.

[LAE+15] Liu, Wei; Anguelov, Dragomir; Erhan, Dumitru; Szegedy, Christian; and Reed, Scott E: *{SSD:} Single Shot MultiBox Detector*. In: *CoRR*, volume abs/1512.0, no. 1, 2015.

[Lara13] *La Route Automatisee: Traffic Lights Recognition (TLR) public benchmark*. 2013.

[LDG+17] Lin, Tsung-Yi; Dollár, Piotr; Girshick, Ross; et al.: *Feature Pyramid Networks for Object Detection*. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 936–944, 2017.

[LGG+18] Lin, Tsung-yi; Goyal, Priya; Girshick, Ross; He, Kaiming; and Dollar, Piotr: *Focal loss for dense object detection.* In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2018.

[LK19] Lee, Eunseop and Kim, Daijin: *Accurate traffic light detection using deep neural network with focal regression loss.* In: *Image and Vision Computing*, volume 87, pages 24–36, 2019.

[LKK04] Lindner, Frank; Kressel, Ulrich; and Kaelberer, Stephan: *Robust recognition of traffic signals.* In: *IEEE Intelligent Vehicles Symposium, Proceedings*, no. July 2004, pages 49–53, 2004.

[Llo82] Lloyd, S.: *Least squares quantization in PCM.* In: *IEEE Transactions on Information Theory*, volume 28, no. 2, pages 129–137, 1982.

[LLW19] Langenberg, Tristan; Luddecke, Timo; and Worgotter, Florentin: *Deep Metadata Fusion for Traffic Light to Lane Assignment.* In: *IEEE Robotics and Automation Letters*, volume PP, pages 1–1, 2019.

[LMB+14a] Lin, Tsung Yi; Maire, Michael; Belongie, Serge; et al.: *Microsoft COCO: Common objects in context.* In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 8693 LNCS, no. PART 5, pages 740–755, 2014.

[LMB+14b] Lin, Tsung-Yi; Maire, Michael; Belongie, Serge J.; et al.: *Microsoft COCO: Common Objects in Context.* In: *European Conference on Computer Vision (ECCV)*, 2014.

[LMY10] Losson, O.; Macaire, L.; and Yang, Y.: *Chapter 5 - Comparison of Color Demosaicing Methods.* In: Hawkes, Peter W. (editor): *Advances in Imaging and electron Physics*, volume 162. From the series *Advances in Imaging and Electron Physics*, pages 173–265, Elsevier, 2010.

[LW18] Langenberg, T. and Wörgötter, F.: *Automatic Traffic Light to Ego Vehicle Lane Association at Complex Intersections.* In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 1350–1357, 2018.

[MD18] Müller, Julian and Dietmayer, Klaus: *Detecting Traffic Lights by Single Shot Detection.* In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 266–273, 2018.

[MFD17] Müller, Julian; Fregin, Andreas; and Dietmayer, Klaus: *Multi-camera system for traffic light detection: About camera setup and mapping of detections.* In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 165–172, 2017.

[MFD18] Müller, Julian; Fregin, Andreas; and Dietmayer, Klaus: *Disparity Sliding Window: Object Proposals from Disparity Images*. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5777–5784, 2018.

[MS15] Michael, Matthias and Schlipsing, Marc: *Extending traffic light recognition: Efficient classification of phase and pictogram*. In: *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2015.

[MTM12] Mogelmose, Andreas; Trivedi, Mohan M.; and Moeslund, Thomas B.: *Vision-Based Traffic Sign Detection and Analysis for Intelligent Driver Assistance Systems: Perspectives and Survey*. In: *IEEE Transactions on Intelligent Transportation Systems*, volume 13, no. 4, pages 1484–1497, 2012.

[NC05] Niculescu-Mizil, Alexandru and Caruana, Rich: *Predicting good probabilities with supervised learning*. In: *ICML 2005 - Proceedings of the 22nd International Conference on Machine Learning*, no. 1999, pages 625–632, 2005.

[NDZ+19] Nixon, Jeremy V.; Dusenberry, Michael W.; Zhang, Linchuan; Jerfel, Ghassen; and Tran, Dustin: *Measuring calibration in deep learning*. In: *arXiv*, 2019.

[NDZ10] Nienhuser, Dennis; Drescher, Markus; and Zollner, J. Marius: *Visual state estimation of traffic lights using hidden Markov models*. In: *13th International IEEE Conference on Intelligent Transportation Systems*, pages 1705–1710, 2010.

[ONS17] *AR0331 1/3 Inch 3.1 Mp/Full HD Digital Image Sensor*. Rev. 14. ON Semiconductor, 2017. URL: `https://www.onsemi.com/pdf/datasheet/ar0331-d.pdf` (visited on 03/15/2021).

[OO10] Omachi, M. and Omachi, S.: *Detection of traffic light using structural information*. In: *Signal Processing (ICSP), 2010 IEEE 10th International Conference on*, volume 2, no. 2, pages 809–812, 2010.

[PGC+19a] Possatti, Lucas C.; Guidolini, Ranik; Cardoso, Vinicius B.; et al.: *Traffic Light Recognition Using Deep Learning and Prior Maps for Autonomous Cars*. In: *CoRR*, volume abs/1906.11886, 2019.

[PGC+19b] Possatti, Lucas C.; Guidolini, Ranik; Cardoso, Vinicius B.; et al.: *Traffic Light Recognition Using Deep Learning and Prior Maps for Autonomous Cars*. In: *Proceedings of the International Joint Conference on Neural Networks*, volume 2019-July, 2019.

[Pla+99] Platt, John et al.: *Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods*. In: *Advances in large margin classifiers*, volume 10, no. 3, pages 61–74, 1999.

[Pow11] Powers, D. M. W.: *Evaluation: From precision, recall and f-measure to roc., informedness, markedness & correlation.* In: *Journal of Machine Learning Technologies*, volume 2, no. 1, pages 37–63, 2011.

[RDGF16a] Redmon, Joseph; Divvala, Santosh; Girshick, Ross; and Farhadi, Ali: *You Only Look Once: Unified, Real-Time Object Detection.* In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016.

[RDGF16b] Redmon, Jospeh; Divvala, Santosh; Girshick, Ross; and Farhadi, Ali: *You Only Look Once: Unified, Real-Time Object Detection.* In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016.

[RF18] Redmon, Joseph and Farhadi, Ali: *YOLOv3: An Incremental Improvement.* In: *CoRR*, 2018.

[RHGS17] Ren, S.; He, K.; Girshick, R.; and Sun, J.: *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.* In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 39, no. 6, pages 1137–1149, 2017.

[RiLSA15] *Bundesministerium für Verkehr und digitale Infrastruktur: Richtlinien für Lichtsignalanlagen: RiLSA*, 2015.

[SEZ+14] Sermanet, Pierre; Eigen, David; Zhang, Xiang; et al.: *OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks.* In: Bengio, Yoshua and LeCun, Yann (editors): *2nd International Conference on Learning Representations, ICLR 2014*, 2014.

[SHA17] Said, Asaad F.; Hazrati, Mehrnaz Kh; and Akhbari, Farshad: *Real-time detection and classification of traffic light signals.* In: *Proceedings - Applied Imagery Pattern Recognition Workshop*, pages 1–5, 2017.

[SIVA17] Szegedy, Christian; Ioffe, Sergey; Vanhoucke, Vincent; and Alemi, Alexander A.: *Inception-v4, inception-ResNet and the impact of residual connections on learning.* In: *31st AAAI Conference on Artificial Intelligence, AAAI 2017*, pages 4278–4284, 2017.

[SL09] Sokolova, Marina and Lapalme, Guy: *A systematic analysis of performance measures for classification tasks.* In: *Information Processing and Management*, volume 45, no. 4, pages 427–437, 2009.

[SLJ+15] Szegedy, Christian; Liu, Wei; Jia, Yangqing; et al.: *Going deeper with convolutions.* In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 07-12-June, pages 1–9, 2015.

[SMA15] Salarian, Mehdi; Manavella, Andrea; and Ansari, Rashid: *A vision based system for traffic lights recognition.* In: *2015 SAI Intelligent Systems Conference (IntelliSys)*, pages 747–753, 2015.

[SNK+17] Saini, Sanjay; Nikhil, S.; Konda, Krishna Reddy; Bharadwaj, Harish S.; and Ganeshan, N.: *An efficient vision-based traffic light detection and state recognition for autonomous vehicles*. In: *IEEE Intelligent Vehicles Symposium, Proceedings*, no. June, pages 606–611, 2017.

[SSH19] Seo, Seonguk; Seo, Paul Hongsuck; and Han, Bohyung: *Learning for Single-Shot Confidence Calibration in Deep Neural Networks Through Stochastic Inferences*. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[StBA20] *Statistisches Bundesamt: Pressemitteilung Nr. 061 vom 27. Februar 2020*. 2020. URL: `https://www.destatis.de/DE/Presse/Pressemitteilungen/2020/02/PD20_061_46241.html` (visited on 10/18/2020).

[SVI+16] Szegedy, Christian; Vanhoucke, Vincent; Ioffe, Sergey; Shlens, Jon; and Wojna, Zbigniew: *Rethinking the Inception Architecture for Computer Vision*. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2016-December, pages 2818–2826, 2016.

[SZ15] Simonyan, Karen and Zisserman, Andrew: *Very Deep Convolutional Networks for Large-Scale Image Recognition*. In: *International Conference on Learning Representations*, 2015.

[SZZ16] Shi, Zhenwei; Zou, Zhengxia; and Zhang, Changshui: *Real-time traffic light detection with adaptive background suppression filter*. In: *IEEE Transactions on Intelligent Transportation Systems*, volume 17, no. 3, pages 690–700, 2016.

[Tesla20] *Tesla Motors: Autopilot and Full Self-Driving Capability*. 2020. URL: `https://www.tesla.com/support/autopilot` (visited on 03/15/2021).

[TPBN14] Trehard, Guillaume; Pollard, Evangeline; Bradai, Benazouz; and Nashashibi, Fawzi: *Tracking both pose and status of a traffic light via an Interacting Multiple Model filter*. In: *FUSION 2014 - 17th International Conference on Information Fusion*, no. July, 2014.

[USGS13] Uijlings, Jasper R.R.; van de Sande, Koen E.A.; Gevers, Theo; and Smeulders, Arnold W.M.: *Selective Search for Object Recognition*. In: *International Journal of Computer Vision*, 2013.

[WHZ18] Weber, Michael; Huber, Matthias; and Zöllner, J. Marius: *HDTLR: A CNN based Hierarchical Detector for Traffic Lights*. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 255–260, 2018.

[WJYW11] Wang, Chunxiang; Jin, Tao; Yang, Ming; and Wang, Bing: *Robust and Real-Time Traffic Lights Recognition in Complex Urban Environments*. In: *International Journal of Computational Intelligence Systems*, volume 4, no. 6, pages 1383–1390, 2011.

[WWZ16] Weber, Michal; Wolf, Peter; and Zöllner, Marius J.: *DeepTLR: A single deep convolutional network for detection and classification of traffic lights*. In: *2016 IEEE Intelligent Vehicles Symposium (IV)*, pages 342–348, 2016.

[ZD14] Zitnick, C. Lawrence and Dollár, Piotr: *Edge boxes: Locating object proposals from edges*. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 8693 LNCS, no. PART 5, pages 391–405, 2014.

[ZE01] Zadrozny, Bianca and Elkan, Charles: *Obtaining calibrated probability estimates from decision trees and naive Bayesian classifiers*. In: *International Conference on Machine Learning (ICML)*, pages 1–8, 2001.

[ZE02] Zadrozny, Bianca and Elkan, Charles: *Transforming classifier scores into accurate multiclass probability estimates*. In: *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 694–699, 2002.

[ZKL+15] Zhou, Bolei; Khosla, Aditya; Lapedriza, Agata; Oliva, Aude; and Torralba, Antonio: *Object Detectors Emerge in Deep Scene CNNs*. In: *International Conference on Learning Representations (ICLR)*, 2015.

[ZSQ+17] Zhao, Hengshuang; Shi, Jianping; Qi, Xiaojuan; Wang, Xiaogang; and Jia, Jiaya: *Pyramid Scene Parsing Network*. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6230–6239, 2017.

Eigene Veröffentlichungen

Teile dieser Dissertation wurden bereits in folgenden Fachbeiträgen veröffentlicht:

Fregin, Andreas; Müller, Julian; and Dietmayer, Klaus: *Three ways of using stereo vision for traffic light recognition.* In: *IEEE Intelligent Vehicles Symposium, Proceedings*, no. June 2017, pages 430–436, 2017 .

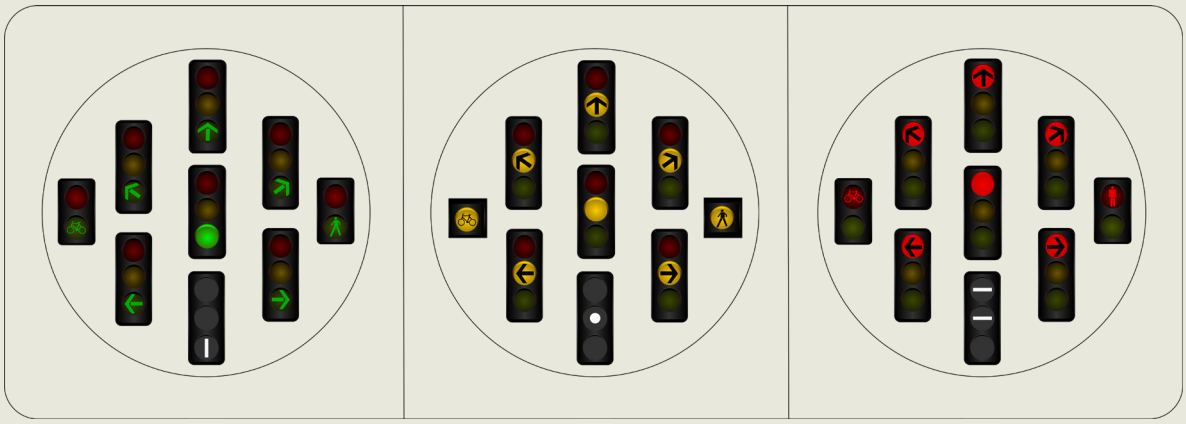
Fregin, Andreas; Müller, Julian; Kressel, Ulrich; and Dietmayer, Klaus: *The DriveU Traffic Light Dataset: Introduction and Comparison with Existing Datasets.* In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3376–3383, 2018.

Fregin, Andreas; Müller, Julian; and Dietmayer, Klaus: *Feature detectors for traffic light recognition.* In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 339–346, 2017.

Müller, Julian and Dietmayer, Klaus: *Detecting Traffic Lights by Single Shot Detection.* In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 266–273, 2018.

Müller, Julian; Fregin, Andreas; and Dietmayer, Klaus: *Multi-camera system for traffic light detection: About camera setup and mapping of detections.* In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 165–172, 2017.

Müller, Julian; Fregin, Andreas; and Dietmayer, Klaus: *Disparity Sliding Window: Object Proposals from Disparity Images.* In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5777–5784, 2018.



ISBN 978-3-941543-60-7

ISBN 978-3-941543-61-4 (E-Book)