Machine Learning in Advanced Driver-Assistance Systems

Contributions to Pedestrian Detection and Adversarial Modeling

von der Fakultät für Elektrotechnik, Informationstechnik und Medientechnik der
Bergischen Universität Wuppertal
genehmigte

Dissertation

zur Erlangung des akademischen Grades
eines Doktors der Ingenieurwissenschaften

von
M.Sc. Farzin Ghorban Rajabizadeh
aus
Wuppertal

Wuppertal 2019

Tag der mündlichen Prüfung:	18. Januar 2019
Hauptreferent:	Prof. Dr.-Ing. Anton Kummert
Korreferent:	Prof. Dr.-Ing. Reinhard Möller

Chapter 1

Introduction

Nowadays computers are ubiquitous. They contribute to making our lives more convenient and secure. They have the potential ability to save human lives, which is well demonstrated in their deployment in modern vehicles. In the context of advanced driver-assistance systems (ADAS), vehicles are equipped with multiple sensors including lidar, radar, and camera all of which record the vehicle's environment in addition to intelligent algorithms for analyzing and understanding the recorded data. For understanding the vehicle's environment, ADAS unite multiple modules such as forward collision detection [135], obstacle detection [182], lane guidance [179], traffic sign recognition [114], and pedestrian detection [185]. Statistics show that over 90 percent of road accidents occur due to human errors[1]. A vehicle's ADAS can, in advance, alert the driver of hazardous conditions or actively intervene in such situations to reduce the human error and potentially reduce road accidents.

This study contributes to the research in modern ADAS on different aspects. The two main contributions comprise both pedestrian detection, that is recognizing and localizing pedestrians in images, and synthetic traffic sign generation. In chapters 2 and 3, we outline relevant research on object detection then discuss methodologies and data that are used to rank our approaches and compare them to the state of the art.

Methods deployed in ADAS must be accurate and computationally efficient in order to run fast. Ideally, they are required to execute in real time on embedded platforms. In chapter 4 and [66, 62], we introduce a novel approach for pedestrian detection that is specially designed for low-consumption hardware. Concretely, we identify the proposal evaluation phase as the computational bottleneck of two-stage cascades that involve a

[1] https://www.dekra-roadsafety.com/media/dekra-verkehrssicherheitsreport-2016-de.pdf

convolutional neural network (CNN) as the second component. As for the first component, we employ a cascaded boosted forest (CBF) detector. In order to economize on the computational cost of the arrangement, we share the feature pyramid that the CBF detector constructs and forward only features that belong to the promising locations in the image to the CNN classifier. In this manner, the expensive feature computation is done once and features are reused by the CNN. For evaluating the features, we design a small-sized CNN that can rapidly process the small proposal dimensions and has a sufficient depth to achieve an accurate classification quality. The CNN is trained from scratch. In various evaluations its optimal operational point, training routine, and location in the pipeline are determined. We demonstrate that our approach can achieve a high performance while running in real time with 30 frames per second without being parallelized and without the use of a GPU. Furthermore, we introduce multiple versions of our approach. The results concluded that our three-stage cascade ranks as the fourth best-performing method reported on one of the challenging pedestrian datasets that are available online.

The other challenge we face with ADAS would be the issue of training efficient detection methods which requires human effort. This would be an extensive manual annotation for preparing training data. In chapter 5 and [65], we introduce a novel approach and insights to make CBF detectors a more data efficient. We decompose a detector into its fundamental parts in order to obtain a better understanding of how the different components contribute to the detection quality. A crucial insight from our evaluations is that the underlying AdaBoost algorithm in CBF frameworks not only copes with highly imbalanced numbers of positive (pedestrians) and negative (backgrounds) training samples but it also benefits from a relatively high number of negative samples. This insight is relevant for many multi-scale object detection tasks since the number of available positive samples in datasets usually is a fraction of the number of the negative samples. In order to exploit the asymmetry in the datasets, however, it is essential to optimize the training routine, especially the sample selection and gathering process. We propose an approach for gathering a sufficient number of high-quality samples without the need for any data augmentation technique. We demonstrate that our approach effectively prevents overfitting and, therefore, allows increasing the model capacity without incurring the risk of performance reduction or poor generalization. Our approach is orthogonal to known researches and can, therefore, be employed in existing CBF detection methods without decelerating the detector. We demonstrate comparisons to the state of the art where we rank as second-best among CBF detectors on two challenging pedestrian datasets. This is achieved while using a relatively small number of simple aggregated channel features, which allows our detector to run multiple times faster than competitors.

Acquiring labeled training data is costly and time-consuming, particularly in the case of traffic sign recognition, since countries do not use unified traffic signs plus different traffic signs do not occur equally often. Due to these difficulties, it requires many hours of acquisition and preparation to obtain a large number of well-balanced and labeled training samples. In chapter 6 and [63, 64], we investigate the use of synthetic data and the involvement of advanced learning approaches with the aspiration to reduce the human efforts behind the data preparation and to make the training of recognition models more data efficient. For these purposes, we employ the approach of generative adversarial networks (GANs). Our study comprises two contributions. Primarily, we algorithmically and architecturally adapt the adversarial modeling framework to the image data provided in ADAS, the so-called red-clear-clear-clear images. We demonstrate that our framework can process multiple channels that have different resolutions and textures, and generate real-looking red-clear-clear-clear traffic sign samples. Our framework allows adaptation of known approaches that we use to enable the generator to create specific samples and even to change incisive attributes of the samples. We also demonstrate that a variation of our framework can transfer visual properties. Secondarily, we study and discuss relevant researches that successfully employ synthetic data for training traffic sign recognition models. Based on the studies and detailed analyses and evaluations of our framework, we discuss future research directions and conclude that GANs can contribute in multiple ways to the training of traffic sign classifiers.

Chapter 7 concludes this work with a summary, discussion, and perspectives for future research.

Chapter 2

Machine Learning for Object Detection

2.1 Introduction

Object detection is one of the most important disciplines in image understanding. Detection methods are required for localizing an object of interest within an image. With advancements in computer vision, numerous detection frameworks have been developed. This chapter provides a brief overview of recent methods for object detection with a special focus on pedestrian detection under real-world conditions [23, 11].

Systematic overview. A majority of multiscale detection methods discussed in this study can be decomposed into fundamental subprocesses as shown in figure 2.1. Some methods may further include pre- or post-processing steps, employ the subprocesses in a different order, or omit some subprocesses. In the following, we briefly describe the major tasks of each subprocess and review them in greater detail in the coming sections:

- Proposal generation: Classification methods predict the class membership of a given patch, which usually has a predefined dimension. Proposal generation methods define the search space for a classifier. In other words, proposal generation methods present patches from different locations and scales in an image to the classifier and reformulate the localization task as, for example, an iterative classification task.

- Feature creation: Classification methods use feature creation functions to map the image/patch into a feature space where the most relevant characteristics of the image/patch are emphasized. These characteristics have various levels of complexity and can include shapes, colors, edges, abstract information, etc.

- Classification: Both processes mentioned above manage the input stream to the classifier. In the case of pedestrian detection, the classifier functions in a binary manner and discriminates the input patches between pedestrians (positives) and backgrounds (negatives). The locations of the classified pedestrians are marked, for example, using bounding boxes that surround the pedestrians.

- Bounding box clustering: The number of detections (bounding boxes) is usually weakly correlated with the number of objects in an image. To understand the contents of an image, a method is often employed to cluster neighboring detections and remove redundant boxes.

- Bounding box regression: Similar to classification methods, a regression method receives an input patch but predicts its correct bounding box location.

The remainder of this chapter is organized as follows. In section 2.2, we outline some of the most important machine learning techniques used for

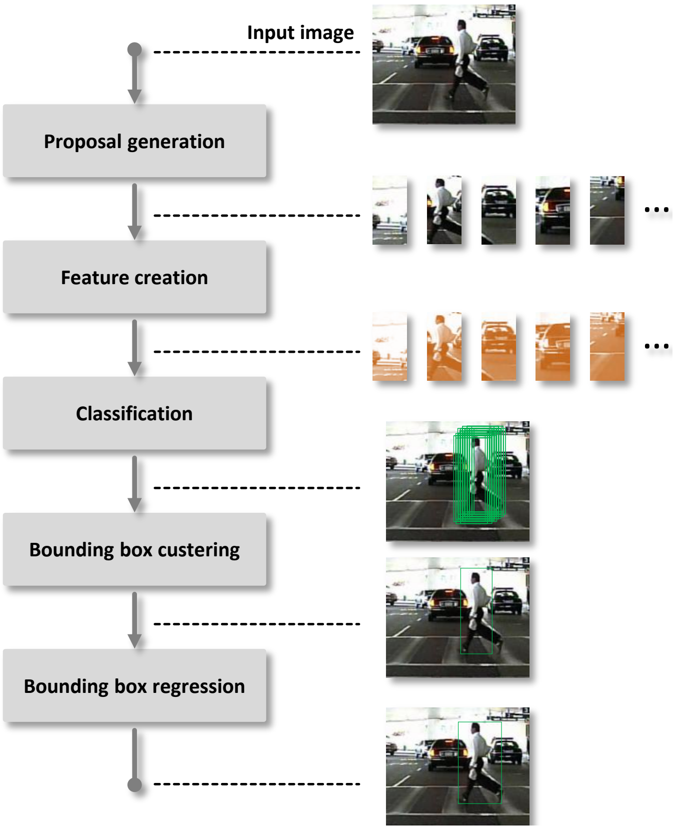

Figure 2.1: Decomposition of a pedestrian detection method in its fundamental subprocesses.

object detection and regression in images. In section 2.3, we review relevant state-of-the-art mechanisms for proposal generation. Section 2.4 describes feature representations with a special focus on those used in our study and finally, in section 2.5, post-processing algorithms used for bounding box clustering are discussed.

2.2 Machine learning

For solving some real-world problems, it is required to find a complex function that maps an input x into some desired output y. Machine learning approaches train a model that approximates such a function, as closely as possible, without being explicitly programmed or guided by rules but only implicitly through a set of samples [147]. This set is referred to as *training set* and is composed of N corresponding pairs $\mathcal{X} = \{x^{(1)}, \ldots, x^{(N)}\}$ and $\mathcal{Y} = \{y^{(1)}, \ldots, y^{(N)}\}$. Here, $x^{(i)}$ may represent an image patch, i.e., $x^{(i)} \in \mathbb{R}^{H^p \times W^p \times C^p}$, where H^p, W^p, and C^p refer to the height, width, and depth of the patch, respectively. The task is termed *classification* if $y^{(i)} \in \mathbb{N}^n$ with $n \geq 1$ (for $n > 1$, $y^{(i)}$ is usually one-hot encoded) and if $y^{(i)} \in \mathbb{R}^n$, the task is termed *regression*. The capability of the trained model to *generalize*, i.e., the ability to perform accurately on a set of new, unseen samples/tasks is an important property of these approaches and is sought to be maximized. The set of the new, unseen samples is referred to as the *test set*.

Usually, one distinguishes between three types of learning approaches [25]:

- Supervised learning: The training set comprises both $\mathcal{X}$ and the corresponding desired outputs $\mathcal{Y}$. During training, $\mathcal{X}$ and $\mathcal{Y}$ are presented to the model and the model parameters are adapted according to the distance between the produced and the desired outputs.
- Unsupervised learning: $\mathcal{X}$ is available but $\mathcal{Y}$ is not. $\mathcal{X}$ can be used, for example, to discover groups of similar samples within the training data, this is known as *clustering*.
- Reinforcement learning: The exact output of the function to be learned is unknown, and training relies on parameter adjustments based on two concepts: reward and penalty. In other words, if the model does not perform well enough, it is penalized and its parameters are adapted accordingly. Otherwise, it is rewarded, i.e., reinforcement occurs. The difference between reinforcement learning and supervised learning is that in reinforcement learning, optimal outputs must be discovered by a process of trial and error. Reinforcement learning