Universität Ulm | 89069 Ulm | Germany
Faculty of Engineering, Computer Science and Psychology
Institute of Neural Information Processing

Deep Learning for Person Detection in Multi-Spectral Videos

Master's Thesis at University of Ulm

Author:
Daniel König
daniel.koenig@uni-ulm.de

Examiner:
Prof. Dr. Heiko Neumann
Dr. Michael Teutsch

Supervisor:
Dr. Michael Teutsch, Michael Adam
Christian Jarvers, Georg Layher

2017

Version June 2, 2017

Contents

Abstract

Person detection is a popular and still very active field of research in computer vision [ZBO+16] [BOHS14] [DWSP12]. There are many camera-based safety and security applications such as search and rescue [Myr13], surveillance [Shu14], driver assistance systems, or autonomous driving [Enz11]. Although person detection is intensively investigated, the state-of-the-art approaches do not achieve the performance of humans [ZBO+16]. Many of the existing approaches only consider Visual optical (VIS) RGB images. Infrared (IR) images are a promising source for further improvements [HPK+15] [WFHB16] [LZWM16]. Therefore, this thesis proposes an approach using multi-spectral input images based on the Faster R-CNN framework [RHGS16]. Different to existing approaches only the Region Proposal Network (RPN) of Faster R-CNN is utilized [ZLLH16]. The usage of two different training strategies [WFHB16] for training the RPN on VIS and IR images separately are evaluated. One approach starts using a pre-trained model for initialization, while the other training procedure additionally pre-finetunes the RPN with an auxiliary dataset. After training the RPN models separately for VIS and IR data, five different fusion approaches are analyzed that use the complementary information of the VIS and IR RPNs. The fusion approaches differ in the layers where fusion is applied. The Fusion RPN provides a performance gain of around 20 % compared to the RPNs operating on only one of the two image spectra. An additional performance gain is achieved by applying a Boosted Decision Forest (BDF) on the deep features extracted from different convolutional layers of the RPN [ZLLH16]. This approach significantly reduces the number of False Positives (FPs) and thus boosts the detector performance by around 14 % compared to the Fusion RPN. Furthermore, the conclusions of Zhang et al. [ZLLH16] are confirmed that an RPN alone can outperform the Faster R-CNN approach for the task of person detection. On the KAIST Multispectral Pedestrian Detection Benchmark [HPK+15] state-of-the-art results are achieved with a log-average Miss Rate (MR) of 29.83 %. Thus, compared to the recent benchmark results [LZWM16] a relative improvement by around 18 % is obtained.

Acknowledgements

I would first like to thank my supervisors Dr. Michael Teutsch and Prof. Dr. Heiko Neumann for the possibility of writing this Master's Thesis and the valuable guidance. Special thanks go to Dr. Michael Teutsch for the intense supervision and support. Furthermore, I would like to thank Hensoldt for the opportunity of writing this work and the support during the last six month. I would also like to thank Christian Jarvers and Georg Layher from the University of Ulm, and Michael Adam as well as the colleagues at Hensoldt for their helpful discussions. Additionally, I want to thank my family for providing me with unfailing support and continuous encouragement throughout my years of study.

1 Introduction

The field of computer vision provides many tasks of automizing visual perception such as surveillance [Shu14], autonomous driving [Enz11]. Therefore, oject detection has received great attention during recent years. Person detection is a specialization of the more general object detection. While object detection deals with detecting more than one object class, person detection only distinguishes between person and background. Person detection is a very popular topic of research, especially in the fields of autonomous driving, driver assistance systems, surveillance, and security. Despite extensive research on person detection, there is still a large gap compared to the performance of humans as Zhang et al. [ZBO+16] evaluate with establishing a human baseline on the popular Caltech Pedestrian Detection Benchmark dataset [DWSP12]. In this thesis, the term *person detection* is used instead of pedestrian detection, because the challenges are considered as general detection problem not strictly related to the field of autonomous driving and driver assistance systems. Obviously, all pedestrians are persons, but not all persons are pedestrians.

Challenges of person detection are deformation and occlusion handling [OW13] [OW12] [DWSP12], detecting persons at multiple different scales [ROBT16], illumination changes [VHV+16], and the runtime requirements the algorithms should meet to be used in real-time. Deformation handling describes the changing appearance of persons depending on the person's pose and the point of view of the camera, as well as how to model the changing appearance. Considering the example of autonomous driving: while a car passes a person, the camera acquires the person in many different poses, yielding to different appearances of the person in the image. Occlusion means that the person is hidden by other objects. Persons can be occluded by other persons, cars, or parts of the environment such as trees, fences or walls. The person's size in the image depends on the person's distance to the camera. Hwang et al. [HPK+15] provide a coarse categorization. They classify the persons closer than 11 meters as near or large-scale persons. These persons correspond to a height of around 115 pixels or more in the image. The medium-scale persons are in a distance between 11 and 28 meters from the camera, and appear with a height between 45 and 115 pixels in the image. Far (small-scale) persons have a distance of 28 meters or more, and a height smaller than 45 pixels in the image.

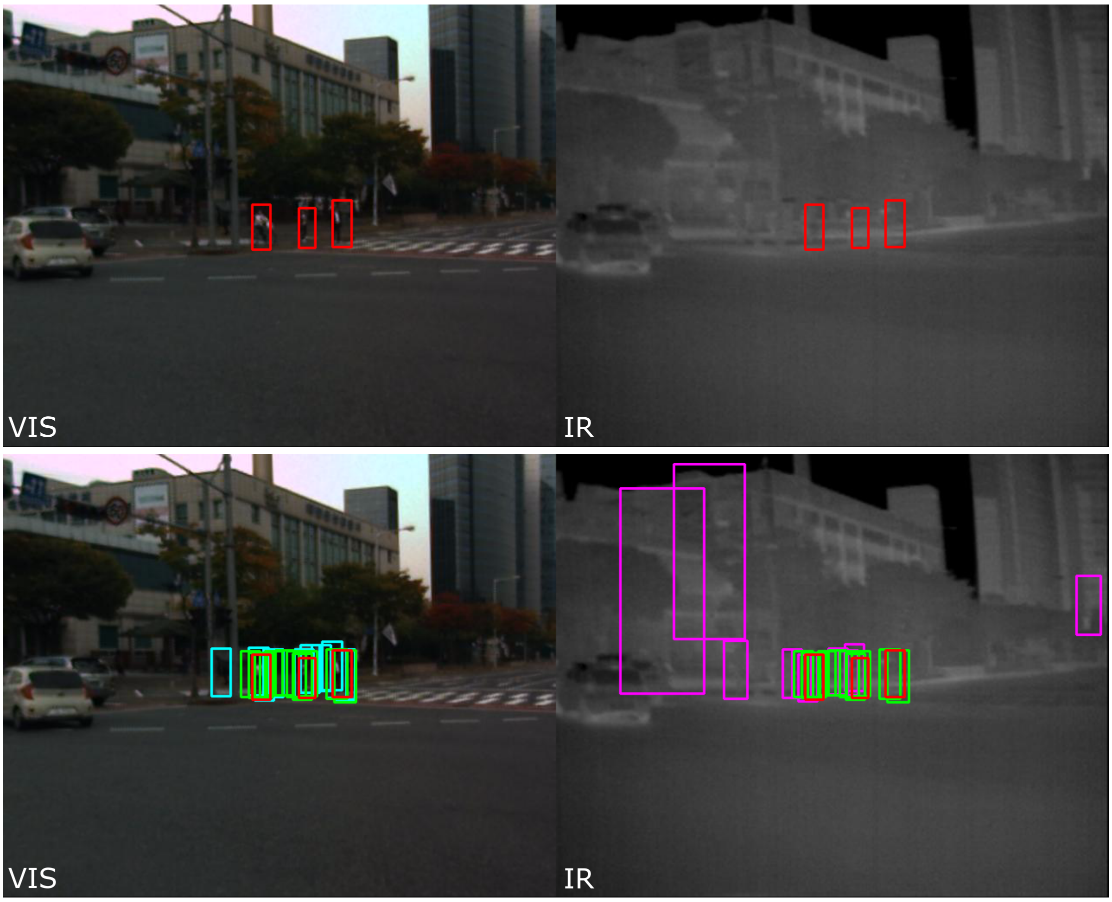


Figure 1.1: (First row) Illustration of the Ground Truth (GT) bounding boxes (red) on a VIS and IR image pair. (Second row) Plotted detection result bounding boxes for one detector trained on VIS images only (cyan), one on IR images only (magenta), and the other detector trained on both, VIS and IR images (green). The multi-spectral image belongs to the KAIST dataset [HPK+15].

An additional big challenge of person detection is to retain the algorithm's robustness for different environments including daytime and nightime scenes, different weather conditions such as rain or fog, and illumination changes [VHV+16]. Therefore, Hwang et al. introduce the KAIST Multispectral Pedestrian Detection Benchmark dataset [HPK+15] consisting of Visual optical (VIS) images and thermal Infrared (IR) images. The additional use of the IR images increases the robustness and detection performance. This assumption is proved in this work. Figure 1.1 shows a pair of VIS and IR images of the KAIST dataset. In the image pair on the top, the annotated Ground Truth (GT) bounding boxes are visualized by red boxes. The image pair at the bottom shows GT boxes together with the detection boxes of three different detectors. On the VIS image, the detections of a detector trained and working only on VIS images are depicted with cyan boxes. In the IR image, the detections of a detector only designed for IR images are illustrated by magenta boxes. The green boxes in both images show the detection results of a detector using both images (VIS and IR) for training and evaluation. The detections of the VIS detector provide higher quality compared to the IR detector, which responds many boxes predicting objects in the background. By considering the detector using both image types, the improved localization of the detection boxes around the GT boxes can be recognized. Furthermore, the false detections of the IR detector are suppressed.

One goal of this work is to show the synergy effect of using VIS and IR images together instead of separately, as motivated with Figure 1.1. The complementary information of VIS and IR images, which can be combined to enhance the detection results is analyzed in this thesis. Wagner et al. [WFHB16] do similar evaluations in their work. They use deep learning and implement two different fusion approaches to fuse the information of VIS and IR images. Their architectures are explained in Section 6.3. Deep learning is a class of machine learning algorithms that are based on neural networks. Convolutional Neural Networks (CNNs) are very popular popular neural networks used for deep learning and have a large spectrum of applications. Table 1.1 shows the results of Wagner et al., evaluated using a KAIST testing subset. The CaffeNet-RGB detector describes a detector only trained and evaluated on VIS images, analogously the CaffeNet-T detector is only trained on IR images. Both detection results are comparable, the IR detector outperforming the VIS detector. With fusing the two sub-networks and using VIS and IR data, Wagner et al. achieve a significant performance boost showing the synergy effect of using both image types, instead of only one of them.

In the work of Liu et al. [LZWM16], they present a fusion approach for fusing VIS and IR information based on the popular detection framework Faster R-CNN [RHGS16] (Section 6.2). Similar to Wagner et al., they train and evaluate the Faster R-CNN sub-networks separately on VIS and IR data. The results of their approach are listed in Table 1.2. The Faster R-CNN-C detector is based on only using VIS images, whereas the Faster R-CNN-T detector is based on only using IR images. With fusing the two sub-networks halfway, they are able to decrease the log-average Miss Rate (MR) significantly. Furthermore, Liu et al. compare the detections of

Benchmark Results [WFHB16]	**MR (%)**
CaffeNet-RGB	56.52
CaffeNet-T	54.67
Late Fusion CNN	**43.80**

Table 1.1: Excerpt of the results of Wagner et al. [WFHB16] for comparing the results for using multi-spectral images instead of only VIS or only IR images evaluated on the KAIST dataset.

the two detectors (Faster R-CNN-C and Faster R-CNN-T). They recognize that the results of both detectors contain detections detected by both, but as well the results contain detections, which are in each case not detected by the other detector. Thus, their results additionally motivate the evaluation of using multi-spectral images for person detection.

Benchmark Results [LZWM16]	**MR (%)**
Faster R-CNN-C	50.36
Faster R-CNN-T	47.35
Halfway Fusion Faster R-CNN	**36.22**

Table 1.2: Excerpt of the results of Liu et al. [LZWM16] for comparing the results for using multi-spectral images instead of only VIS or only IR images evaluated on the KAIST dataset.

1.1 Multi-spectral Images

In this thesis, multi-spectral images are denoted as image pairs consisting of Visual optical (VIS) images containing three channels (RGB), and thermal Infrared (IR) images. The IR images are gray-scale images with values in the range between 0 and 255 representing the thermal radiation acquired by the IR camera. VIS images are highly sensitive to external illumination and contain fine texture pattern details [ZWN07] [LRH07]. This yields to a high diversity and variance in person appearance.

In comparison, IR images have different image characteristics than VIS images. The measured intensity of persons depends on clothing to some extend. Furthermore, the texture details are lost as the body temperature is usually relatively constant [ZWN07] [PLCS14]. Teutsch et al. [TMHB14] state that person detection in IR is commonly better than in VIS. There are multiple categorizations of IR radiation. According to Byrnes [Byr08], the spectrum of wavelengths can be categorized into five categories. Near Infrared (NIR) spans between wavelengths from 0.75 to 1.4 µm, Short-Wavelength Infrared (SWIR) from 1.4 to 3 µm, Mid-Wavelength Infrared (MWIR) from 3 to 8 µm, Long-Wavelength Infrared (LWIR) from 8 to 15 µm, and Far Infrared (FIR) from 15 to 1000 µm. NIR and SWIR depend on active lighting similar to images acquired in the VIS spectrum. MWIR is also called Intermediate Infrared (IIR). NIR and SWIR are sometimes called reflected infrared, whereas MWIR and LWIR are referred to as thermal infrared. For comparison, the ISO division scheme (ISO 20473:2007) subdivides the wavelengths in three categories only: Near Infrared (NIR), Mid Infrared (MIR) and Far Infrared (FIR). Especially the LWIR spectral range from 8 to 12 µm is interesting for person detection, as the human body radiates with a LWIR wavelength of 9.3 µm [HPK+15] [GFS+16] [SIMP07].

1.2 Differentiation of Important Terms

This section introduces important computer vision terms for the remainder of this thesis. The three terms classification, detection, and regression are discriminated. In this thesis, approaches for person detection are analyzed. Thus, it is important to review the meaning of detection compared to classification and regression. The task of image classification means that the image content is analyzed and that the image is labeled w.r.t. its content. For example, if there is an image showing a dog, the resulting label should be *dog*. The image is categorized into a pre-defined number of discrete classes. Recognition is often used synonymously to classification. To give a second example for classification: We have an unknown fruit that is yellow, 14 cm long, has a diameter of 2.5 cm and a density of X. What fruit is it? For this kind of problem, classification is used to classify the object as a banana as opposed to an apple or orange. In the field of machine learning, classification is used to predict the class of an object (discrete values), whereas regression is used to predict continues values. An example for regression is: We have a house with W rooms, X bathrooms and Y square meter size. Based on other houses in the area that have recently been sold, how much can we sell the house for? This problem is a regression problem, since the output is a continuous value. Another example for regression is linear regression that models the relationship between two scalar variables. For each value of the explanatory variable, the linear regression model predicts the dependent variable w.r.t. the determined regression model. Regression models a function with continuous output.

For classification it is not important where the object exactly is located in the image, as long as the contained object can be classified correctly. Instead, for solving the detection task, localization and classification have to be considered. In an image that contains multiple objects, the task of detection is to know where each object is situated and its class. For example, there is an image acquired by a driving car. The driver assistance system has the task of finding the location of each person the image contains. The difference between classification and detection is that the detection additionally has to find the position of each object (localization) instead of only categorizing objects (classification). The number of objects contained in an image and therefore the number of detection outputs is variable, whereas classification and regression output only one resulting value. For predicting object positions in images there are CNNs that are trained for solving a regression problem [FGMR10] [GDDM14] and can be used for localizing objects. The relationship of the three terms is that object detection needs localization and classification, and localization can be achieved and improved by using regression.

1.3 Outline

The remainder of this thesis is structured as follows. Recent approaches and benchmarks are reviewed in Chapter 2. For choosing a deep learning framework, Chapter 3 provides an overview of existing frameworks and defines criteria for choosing a framework. An overview of popular person detection datasets is provided in Chapter 4. The annotations and their characteristics of three datasets of choice are analyzed. In Chapter 5, the filtered channel features based detectors are reviewed, and implementation details are provided. Chapter 6 describes the VGG-16 network and the Faster R-CNN framework that are the fundamentals of this thesis. After reviewing some inspirational work, fusion approaches and the additional usage of a boosted forest classifier are explained. In Chapter 7, the approaches of this work are evaluated and compared to recent baseline results. The conclusion in Chapter 8 provides a summary of the thesis and proposes future work.

2 Related Work

In the last few years, diverse efforts have been made in order to improve the performance of person detection. These efforts involve computational performance as well as detection performance (localization and classification). This chapter provides an overview of different approaches for person detection in VIS and/or IR images. In the first section three coarse concepts are introduced that can be used to subdivide different person detection approaches. The remaining sections explain different approaches and match them to one of the three base concepts.

2.1 Generic Person Detector

In Chapter 1, the difference between classification and detection are explained. In Figure 2.1 the subdivision scheme for categorizing person detection concepts is illustrated. The input image is processed from left to right and the output consists of bounding boxes and their labels. The key processing elements are localization and classification (inspired by [ZWN07]). The localization part is responsible for finding and proposing Regions of Interest (RoIs), whereas the classification part matches these RoIs with appropriate labels, that represent their content. While the classification part is performed for each RoI separately, the localization part is performed once on the entire input image.

The first row in Figure 2.1 represents the typical approach for object detection in general as well as person detection. First, candidate regions are generated, also called RoIs. The proposal generation part in its simplest form is represented by the sliding window paradigm, where a window of fixed sizes and scales is shifted with a fixed stride over the input image, and each position of the window corresponds to one proposal. The goal of the proposed RoIs of an input image is, to use them for processing highly discriminative features (feature extraction). Based on the generated feature pool for each RoI, its content is classified. In many approaches there are some post-processing steps like non-maximum-suppression (NMS), regression

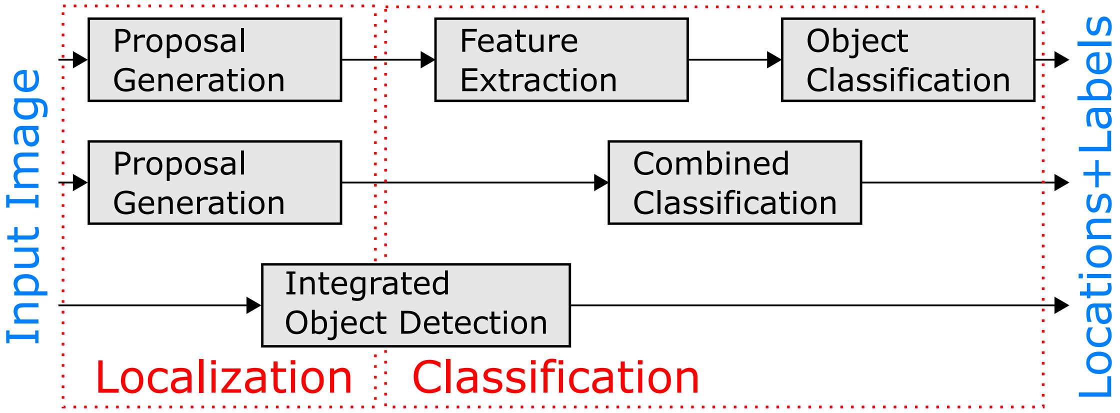


Figure 2.1: Abstracted overview of the different architectures for person detection and a possibility for categorizing the concepts into three categories. Each row represents one concept for person detection. Person detection requires localization and classification.

(improvement of localization) or a part of the candidates is rejected depending on the candidate score which gives a confidence measure estimate for the candidate belonging to a certain class. By using a score threshold RoIs can be discarded. In the case of person detection there are only two label classes (person/background).

The second row in Figure 2.1 is typical for deep learning approaches. The first stage is similar to the first row. In order to meet real-time requirements, the sliding window paradigm cannot be utilized to classify each sliding window with a complex classifier. The third row in Figure 2.1 describes deep learning approaches that use regression for simultaneously localizing and classifying objects. Therefore, these approaches are considered as integrated object detection methods. In the following section, some popular proposal generation approaches are presented. The last type for categorizing detection approaches is based on deep learning, too. There are some Convolutional Neural Network (CNN) architectures which can be trained in an end-to-end fashion and use regression for localization to propose candidates across the whole input image.

2.2 Proposal Generation

After introducing the subdivision scheme for categorizing object/person detectors, this section introduces the different approaches for generating proposals. Some of them are used for object detection in general rather than for person detection only. The goal of proposal generation is the reduction of candidate regions based on objectness measures which are computationally efficient. The naïve method is the exhaustive search approach, which is often called sliding window paradigm [PP00][VJ01b] [FGMR10] [DT05]. Sliding windows are used with different scales and aspect ratios and the windows are shifted across the input image with a fixed number of pixels between two locations (stride). In the boundary regions padding can be applied. Most of the recent object and person detectors are described and evaluated by Hosang et al. [HBDS16]. They state that the reason for using a proposal generator is to reach low miss rate with considerably fewer windows than generated with an exhaustive search approach. The candidate reduction generates a significant speedup and enables the use of more sophisticated classifiers. Since a large number of windows have to be considered for proposal generation, these algorithms have to be computationally efficient. Hosang et al. [HBDS16] categorize the proposal algorithms into grouping, window scoring and some alternative proposal methods.

The most popular region proposal approach belongs to the grouping methods and is called Selective Search [UVGS13]. Selective Search uses a variety of color spaces with different invariance properties, different similarity measures for grouping segmented regions, and varies the starting regions in order to generate class-independent object proposals. Selective Search has been widely used as proposal generator for many state-of-the-art object detectors, such as Regions with CNN features (R-CNN) by Girshick et al. [GDDM14] and Fast R-CNN by Girshick [Gir16], which can be applied for person detection as well. Another grouping proposal method by Endres and Hoiem [EH10] utilizes hierarchical segmentation from occlusion boundaries. They solve graph cuts with different seeds and parameters to generate segments. Based on a wide range of cues, the proposals are ranked.

The window scoring proposal methods Edge Boxes by Zitnick and Dollár [ZD14] and Objectness by Alexe et al. [ADF12] are among the most prominent approaches. Objectness rates candidate windows according to multiple cues whether they contain an object or not. Cues are e.g. color contrast, edge density, superpixels straddling, location and size of the candidate window. The Edge Boxes algorithm is similar to Objectness and starts with a sliding window pattern. Zitnick and Dollár state that the number of contours wholly enclosed by a bounding box indicates the likelihood of that box containing an object. Based on an edge map neighboring edge pixels of similar orientation are clustered together to form long continuous contours. The score of a box is computed by considering the edges within the candidate window and those straddling the window's boundary.

The Multibox method by Erhan et al. [ESTA14] is based on deep CNNs. They tailor the AlexNet of Krizhevsky et al. [KSH12] towards solving the localization problem. Here, object detection is formulated as a regression problem to the coordinates of several given bounding boxes, where the proposals are class agnostic. The output of Multibox is a fixed number of bounding boxes, with their individual scores expressing the network's confidence that this box contains an object. The successor of Multibox is Multi-Scale Convolutional Multibox MSC-Multibox by Szegedy et al. [SRE+14]. In comparison to the predecessor, MSC-Multibox improves the network architecture for bounding box generation by including multi-scale convolutional bounding box predictors (prediction tree). Two similar approaches, the Region Proposal Network (RPN) of Faster R-CNN by Ren et al. [RHGS16], and You Only Look Once (YOLO) by Redmon et al. [RDGF15] (see Section 2.4) can be used as proposal generators as well. Ghodrati et al. [GDP+15] introduce the coarse-to-fine inverse cascade for their proposal generator. They exploit the high-level feature maps well adapted for recognizing objects, as well as the feature maps of lower convolutional layers with simpler features, having a much finer spatial representation of the image.

Filtered channel features such as Aggregate Channel Features (ACF) [DABP14] or Checkerboards [ZBS15] in combination with a Boosted Decision Forest (BDF) [DTPB09] trained with AdaBoost [FHTO00] are used as object [DABP14] and person detectors [ZBS15]. The implementation of these detectors in Piotr's Computer Vision Matlab Toolbox [Dol] uses the soft cascade method of Bourdev and Brandt [BB05] [ZV08]. The soft cascade threshold can be modified for increasing the number of False Positives (FPs), but simultaneously decreasing the number of False Negatives (FNs). In other words, this modification lowers the miss rate of the given detector. Additional FPs can be handled by the subsequent classification stage. Li et al. [LLS+16] and Wagner et al. [WFHB16] e.g. use the ACF as proposal generator.

2.3 Detection using Hand-Crafted Features

Considering Figure 2.1, in this section, mostly object and person detectors of the first row are reviewed. Hand-crafted features are created by using manually designed filters such as Sobel filter, or at least methods such as Histogram of Oriented Gradients (HOG) [DT05], which are designed by humans and work according to a fixed sequence of processing steps for generating a feature pool. As these methods are computationally efficient, most of them use the sliding window approach as proposal generator. In the follow-up the extraction of discriminative features from the proposed regions are explained and which algorithms are used for classification. As stated by Zhang et al. [ZBS15], by knowing the classifier choice, the classification quality cannot be deduced automatically; rather the features used as input are of higher importance. In the remainder of this section, the mentioned methods are based on Support Vector Machines (SVM) [VV98] [OFG+97] and/or boosted classifiers [DTPB09] [BOHS14] trained using different boosting techniques (Discrete AdaBoost, Real AdaBoost and LogitBoost) [FS96] [FS99] [FHTO00] [Bre01]. The weak classifiers of the boosted classifiers are usually decision trees [Qui86] and therefore they are called BDF. In this work the name BDF is chosen, because it consists of multiple decision trees which are trained by a boosting technique to work as one strong classifier. Other names are boosted forests or boosted decision trees. In Figure 2.1, SVM and BDF represent the object classification element in the first row.

The first popular papers in the field of general visual object detection were published by Viola and Jones [VJ01a] [VJ01b]. Their methods applied to person detection were used for many years as benchmark results. By introducing integral images they were able to achieve state-of-the-art results and real-time processing. Viola and Jones use Haar-like features which can be easily calculated by using the integral images. Based on these features, a classifier is trained using AdaBoost [FS95]. AdaBoost selects a small number of discriminative features out of a big feature pool and uses classifier cascades to reject background hypotheses with simple classifiers in early stages and train more complex classifiers in higher stages. A few years later, Dalal and Triggs introduced human detection by using Histograms of Oriented Gradients (HOG) [DT05]. They generate a feature pool consisting of HOG features and use a SVM for classification.

While the algorithms mentioned before are used for processing VIS images, Suard et. al [SRBB06] adapt the approach of Dalal and Triggs for IR images. Zhang et al. [ZWN07] evaluate the usage of Edgelet features [WN05] together with cascaded classifiers trained using AdaBoost [VJ01a] or a cascade of SVM classifiers [CV95], to HOG features [DT05] trained with a cascade of SVM classifiers only. Their results are evaluated on VIS and IR images. Davis and Keck [DK07] use a two-stage approach for person detection in IR images. The first stage is a fast screening procedure using a generalized template to find potential person locations. The second stage evaluates the potential person locations by creating four feature images (Sobel gradient operators for 0°, 45°, 90° and 135° orientations) and applying AdaBoost for training. Munder and Gavrila [MG06] use Haar Wavelets [PP00], Local Receptive Fields (LRF) [JDM00] and PCA coefficients [JDM00] in combination with a SVM classifier. Gerónimo et al. [GSLP07] use Haar Wavelets as well, but utilize AdaBoost instead of a SVM for classification. Miezianko and Pokrajac [MP08] use modified HOG features and a linear SVM for person detection in low resolution IR images. Keeping Figure 2.1 in mind, the mentioned approaches all belong to the first row, whereas generating the Edgelets, Haar or HOG features correspond to the feature extraction stage. For evaluating VIS images, Wojek et al. [WWS09] present a good evaluation overview of different feature extraction methods (HOG, Haar and combination) in combination with different classifiers (SVM, AdaBoost and MPLBoost [BDTB08]) on their own pedestrian dataset.

An impulse for new approaches was given with the introduction of the Caltech dataset for pedestrian detection by Dollár et al. [DWSP09] [DWSP12]. The Caltech dataset is explained in depth in Chapter 4. Based on this new dataset, Dollár et al. evaluate existing methods based on HOG, Haar or gradient features, using AdaBoost and SVM for classification. By introducing channel features, especially Integral Channel Features (ICF), Dollár et al. [DTPB09] succeeded in creating a new benchmark. The channel features detectors are described in detail in Chapter 5. ICF uses a sliding window over multiple scales of the input feature channels. The feature channels are computed by applying different linear and non-linear transformations (normalized gradient magnitude, histogram of oriented gradients and LUV color channels) on the input image. LUV denotes the CIE (L*u*v*) color space (CIELUV). Thus, the input image (3 channels) is used to get more discriminative feature channels (e.g. 10 channels). Based on these feature channels, higher-order features are randomly generated by applying weighted sums of the feature channels. The generated feature pool is utilized for classification by training a BDF. For improving the computational performance Dollár et al. [DBP10] examined the approximation of multi-scale gradient histograms as well as multi-scale features. Instead of computing the channel features for multiple scales separately, they compute channel features for several scales and approximate the channel features for the remaining scales. Another approach for improving the computational performance of detectors based on channel features is introduced by crosstalk cascades of Dollár et al. [DAK12]. They exploit the correlations by tightly coupling detector evaluation of nearby windows. Using soft cascades [ZV08] [BB05], Benenson et al. [BMTV12] are able to speed up the time for processing an image. The soft cascade aborts the evaluation of non-promising detections if the score of a given stage drops below a learned threshold. Benenson et al. [BMTV13] consider general questions like the size of the feature pool generated for classification, which training method to use (Discrete AdaBoost, Real AdaBoost or LogitBoost), or whether to use data augmentation or not. A good overview and evaluation of different data augmentation techniques for channel features detectors is provided by Ohn-Bar and Trivedi [OBT16].

Based on the introduction of the ICF detector other detectors of the channel features detector family are reviewed. Dollár et al. [DABP14] analyze how to construct fast feature pyramids by approximation (similar to [DBP10]). Additionally, they create a new benchmark detector with the very popular Aggregate Channel Feature (ACF) detector. ICF and ACF differ in that the ACF creates the channel features like ICF, but instead of computing the integral images and randomly summing patches to create high-level features, the ACF uses the generated feature channels as single pixel lookups which represent the feature pool. On the feature pool a cascaded BDF is trained using AdaBoost. Nam et al. [NDH14] improve the detector performance of ACF by replacing the effective but expensive oblique (multiple feature) splits in decision trees by orthogonal (single feature) splits over locally decorrelated data. Instead of using the feature channels directly for training, decorrelating filters are applied per channel generating the so-called Locally Decorrelated Channel Features (LDCF). Considering the channel features in a more generic way, is introduced by Zhang et al. [ZBS15]. Based on the channel features of the ACF detector they applied different filters to improve the discriminative power of the feature pool (similar to LDCF). The following popular so-called filtered channel features based detectors are presented: ACF, re-definition of InformedHaar filters as InformedFilters, SquaresChntrs, Checkerboards, RandomFilters, already mentioned LDCF, and PcaForeground filters. These filters are applied on the channel features, and the resulting channel features are classified using AdaBoost. A new filter to be used like the other channel features filters is proposed by Cao et al. [CPL15]. For designing their filters they exploit some simple inherent attributes of persons (i.e. appearance constancy and shape symmetry). They call the new features side-inner difference features (SIDF) and symmetrical similarity features (SSF). Similar to the simple HOG and Haar detectors, good performing approaches like the ACF detector are also used for person detection in IR images [BVN14][HPK$^+$15]. Based on the introduced filtered channel features, some improvements are proposed. Costea et al. [DN16] generate multi-resolution channels and semantic channels to improve the feature pool, resulting in an enhanced detector performance. Representing a multi-resolution approach as well, Rajaram et al. [ROBT16] propose to train multiple ACF models with different model sizes. While testing, all models are run on the corresponding scales of the feature pyramid and the derived bounding boxes of the different models are accumulated.

There are two popular evaluations based on the Caltech dataset. Benenson et al. [BOHS14] provide a good overview of different person detectors. Additionally, they state that person detection can be enhanced by improving the featues, adding additional context information (e.g. optical flow and/or disparity map), and by a more diverse training dataset (i.e. few diverse persons are better than many similar ones). The second evaluation is done by Zhang et al. [ZBO+16]. They compare the filtered channel features based detectors (ACF, Checkerboards, LDCF, etc.) as well as current deep learning architectures like AlexNet [KSH12] and the VGG-16 [Gir16]. Furthermore Zhang et al. provide a human baseline to evaluate the detector performance compared to humans. The second contribution is the analysis of reasons for detector failures such as small scale, occlusion or bad annotations. Some of the results are considered in Chapter 4.

The first publications about using VIS and IR images in a complementary fashion are given by Toressan et al. [Tor04], Ó Conaire et al. [ÓCO+05], Fang et al. [FYN+03], St-Laurent et al. [SIMP07], and Choi et al. [CP10]. They fuse the VIS and IR information/features on different levels. Hwang et al. [HPK+15] introduce the very popular KAIST Multispectral Pedestrian Detection Benchmark dataset. They also adapt the ACF detector for generating channel features based on multi-spectral image data, which is explained later in Chapter 5. The KAIST dataset is described in more detail in Chapter 4. Based on the ACF detector, Afrakhteh and Miryong [AM17] analyze how a confidence measure can be defined, which can be used for selecting either using the VIS ACF detector results or detections of the IR ACF detector. They state that by choosing only the appropriate detector's detections they are able to reduce the Number of False Positives Per Image (FPPI).

2.4 Detection using Machine-Learned Features

In contrast to the previous section, this one presents methods that are able to learn how to generate a feature pool depending on the training data, and not using predefined features. The learning procedure not only learns to select and combine the appropriate features, but also learns how to design the filters for generating discriminative features. The feature extraction techniques in the remainder are based on trained neural networks, while the channel features of the ACF detector are generated using pre-defined filters, independent of the dataset. CNNs are able to learn appropriate filters and therefore do not rely on manually designed features. Compared to Figure 2.1, there are approaches of all three types. While representatives of the first row are presented in Subsection 2.4.1, representatives of the second and third row are described in Subsection 2.4.2 and Subsection 2.4.3, respectively.

2.4.1 Classification using SVM or BDF Classifiers

This subsection introduces Convolutional Neural Networks (CNN) for feature extraction. Yang et al. [YYLL16] choose the sliding window paradigm as proposal generator for person detection, to ensure comparability for evaluating different feature extraction methods. They compare several popular CNN models for feature extraction, which are AlexNet [KSH12], Visual Geometry Group (VGG) nets [SZ15] and GoogLeNet [SLJ+15]. The generated feature maps are used to train by a BDF. They also evaluate the differences between using feature maps of different hierarchical levels such as conv3_3 or conv4_3 of VGG-16. They show that the mid-level feature maps perform best when used together with BDF and combining the machine-learned features with (hand-crafted) channel features, additionally improves the classification performance.

Chen et al. [CWK+14] use a combination of ACF detector and CNNs. The ACF detector is used for proposal generation due to its robustness towards changing image quality such as image noise or altered image acquisition, compared to the Selective Search algorithm. The candidate regions are warped to a fixed size, which is required for using the proposal as CNN input. For the warped window, AlexNet [KSH12] is utilized to perform feature extraction. The resulting feature vector serves as input for the SVM that classifies the candidate window.

Hu et al. [HWS+16] do their experiments based on the results of Yang et al. They state that compared to other applications in computer vision, CNNs are less effective on person detection. As possible reason they mention the non-optimal network design for person detection. Based on the VGG-16 net, they extensively evaluate the feature maps extracted from different levels of the network by training a BDF like Yang et al. They finetuned the VGG-16 model on the Caltech dataset and discovered improvements. The best results are achieved for the convolutional layers between conv3_3 and conv5_1. With the feature maps of conv3_3, conv4_3 and conv5_1 Hu et al. train separate BDF models and fuse the results by score averaging. By additionally incorporating pixel labels they achieve a log-average Miss Rate (MR) of 8.93 % on the Caltech dataset.

Zhang et al. [ZLLH16] propose a similar approach. They utilize the RPN of Faster R-CNN [RHGS16], which does proposal generation by solving a regression problem. The RPN is described in Subsection 2.4.3, in which regression methods are reviewed. The regions of the feature maps corresponding to the individual proposals are extracted and used for training a BDF. Zhang et al. also combine feature maps of different levels to improve detection performance instead of combining the scores of different BDF models. Their final result on the Caltech dataset is 9.6 % log-average miss rate. A very similar approach is presented by Zhang et al. [ZCST17]. Additionally they consider the RPN in combination with the BDF for infrared (IR) data.

2.4.2 Classification using Neural Networks

Referring to Figure 2.1, this subsection introduces methods, which can be categorized into the second row. Based on generated proposals, these algorithms perform feature extraction and classification combined within one architecture. All approaches are based on neural networks with fully connected and/or convolutional layers. Ouyang and Wang [OW13] present a unified deep model for jointly learning feature extraction, a part deformation model, an occlusion model, and classification. With deformation and occlusion handling, they simultaneously tackle two main challenges in person detection. Based on Felzenszwalb et al. [FGMR10], Luo et al. [LTWT14] use the idea of Deformable Part-based models (DPM) and train a Switchable Deep Network (SDN) which is able to learn mixtures of different body parts and complete body appearances for person classification. This architecture addresses the occlusion challenge of person detection.

Similar to Chen et al. [CWK+14], Wang et al. [WYL+15] use ACF for proposal generation and instead of only using a CNN for feature extraction, they propose a CNN for feature extraction and classification. Their approach is utilized for face detection, but it can be easily adapted for person detection. For training, Wang et al. use hard-negative mining and iteratively collect false positive samples from the background images using the previously trained model. These samples are appended to the training data. Verma et al. [VHV+16] adopt the idea of using the ACF detector as proposal generator as well. Depending on the confidence scores, the proposals are passed directly to the output if the score provides a clear confidence, or to a Mixture of Expert (MoE) CNNs otherwise. The MoE consists of several different CNNs each with a different architecture addressing different person shapes and appearances.

Tian et al. [TLWT15] address the problem of CNNs confusing positive with hard negative samples. They give examples where a tree trunk or wire pole are similar to persons considered from certain viewpoints. To tackle this problem they add person attributes and scene attributes to the learning process using scene segmentation datasets. Pedestrian attributes can be e.g. *carrying backpack* and scene attributes are e.g. road or tree.

Hosang et al. [HOBS15] evaluate CNNs for the task of person detection in general. They extensively analyze small and big CNNs, their architectural choices, parameters, and the influence of different training data, including pre-training. Architectural choices are for example the number and size of convolutional filters or the number and type of layers. Popular CNN architectures such as AlexNet [KSH12], CifarNet [Kri09] and R-CNN [GDDM14] are considered.

Based on the approaches of Felzenszwalb et al. [FGMR10] and Luo et al. [LTWT14], Tian et al. [TLWT16] address occlusion handling for person detection by training multiple different CNNs, each responsible for detecting different parts of a person. These so-called part detectors generate a part pool, containing various semantic body parts. The output of the part detectors is directly used without combining the different scores by a SVM, BDF or an additional CNN. The method is called DeepParts and is evaluated for three different network architectures used for the part detectors: Clarifai [ZF14], AlexNet [KSH12], and GoogLeNet [SLJ+15].

One challenging task of pedestrian detection is the variance of different scales. Lu et al. [LZL] propose a Scale-Discriminative Classifier (SDC) that contains numerous classifiers to cope with different scales. Proposals are generated and a CNN is applied fo feature extraction. But instead of just using the features after the highest convolutional layer, they construct a high resolution feature map of fixed size that combines high-level semantics feature maps (up-sampling) and low-level image features (down-sampling). For each proposal, the appropriate classifier is selected and its input is generated by RoI pooling on the high resolution feature map. This implies that for each candidate window only one classifier is applied. Bunel et al. [BDX16] particularly address the topic of detecting persons at a far distance. They analyze the appearance of small persons and explicitly design their CNN for far-scale persons by adapting the filter sizes to achieve appropriate receptive fields. They re-size medium and near-scale persons by down-scaling to increase the amount of training data. They also apply hard negative mining for training their CNN.

Lin and Chen [LC15] address the problem of hard negative samples, too. They state that CNNs are easy to confuse. Therefore, they propose parallel CNNs: one

CNN is trained with all available data (positive and negative samples); the other CNN is trained for separating the hard negative samples from the positive samples. Candidate windows are sampled by an ACF detector. Both CNNs are based on GoogLeNet [SLJ+15]. To increase the precision of the detected region, bounding box regression is performed, as suggested in R-CNN [GDDM14].

Up to now, only approaches for person detection on VIS images are considered. For applying multi-spectral person detection two methods are presented. Choi et al. [CKPS16] show that machine-learned features have more discriminative power compared to hand-crafted features and that the combination of VIS and IR images provide additional discriminability and are therefore complementary. For proposal generation the Edge Box method is applied. Separately for VIS and IR inputs, two individual Fully Convolutional Networks (FCNs) [LSD15] are trained. Each of them provides a confidence map as result of the FCN. For feature maps of higher levels the discriminative power increases, but its resolution decreases at the same time. Therefore Choi et al. [CKPS16] extract and utilize the feature maps of intermediate layers additionally to the output feature map (confidence map). For each proposal a feature vector is extracted, using Spatial Pyramid Pooling (SPP) [HZRS15b]. This feature vector serves as input for Support Vector Regression (SVR) [CL11] that is trained to classify the candidate regions. For all positive classified proposals an accumulated map is created. The accumulated map together with the confidence maps are combined for finally localizing the person or rejecting the proposal. A second approach for multi-spectral person detection is given by Wagner et al. [WFHB16]. For proposal generation they use the multi-spectral ACF detector [HPK+15], often referred to as ACF-T-THOG, according to the additional IR channel. They introduce two fusion architectures: the Early Fusion architecture fuses the VIS and IR images (4-channel input) before putting them into CaffeNet [JSD+14] for feature extraction and classification. The Late Fusion architecture trains two CaffeNets separately, one for VIS input images (3-channel) and the other for IR images (1-channel). They first use ImageNet [RDS+15] as large auxiliary dataset for developing general low-level filters. Based on this pre-training step, they utilize the Caltech dataset [DWSP12] for an additional pre-finetuning and use the KAIST dataset [HPK+15] for the final finetuning.

2.4.3 Regression based Detection

This subsection presents methods for the third row in Figure 2.1. As explained in Chapter 1, the classification output is described by discrete labels (person/no person), whereas the regression output has a continuous range (e.g., score and/or bounding box coordinates). Instead of proposing RoIs and classifying them, a regression outputs the coordinates directly together with a confidence score.

A simple approach by Szegedy et al. [STE13] is the formulation of object detection as a regression problem to object bounding box masks. They use AlexNet [KSH12] and re-design it for having a regression layer instead of a softmax classifier as last layer. After re-sizing the input image, the resulting binary mask represents one or several objects: 1 means the appropriate pixel lies within the bounding box of an object of a given class, and 0 otherwise. This methods is strongly related to semantic segmentation.

Sermanet et al. [SEZ+13] propose CNNs to use for classification, localization, and detection. They show how a multi-scale sliding window method can be efficiently implemented using a CNN based on AlexNet [KSH12]. The classification layers are replaced by a regression network and are trained to predict object bounding boxes. Since the complete input image is processed once, only the regression layers need to be re-computed for each location and scale. Sermanet et al. call their approach Overfeat.

According to Girshick et al. [GDDM14], Overfeat can roughly be seen as a special case of their proposed R-CNN. R-CNN stands for Regions with CNN features, since the features of region proposals are computed using CNNs. Their object detection system consists of three modules: (1) category-independent region proposal generation, (2) CNN for feature extraction of fixed length, and (3) a set of class-specific linear SVMs. Girshick et al. use Selective Search to find region candidates. For each RoI, a fixed length feature vector is extracted, using a CNN (AlexNet). Since the CNN requires a fixed size input, each proposed region is warped to a fixed size, regardless of size and aspect ratio of the proposal. To reduce the localization error, they train a linear regression model [FGMR10] to predict a new detection window, improving the initial Selective Search region proposals.

The successor of R-CNN is proposed by Girshick [Gir16] and is called Fast R-CNN because of its achieved computational performance gains. A bottleneck of the R-CNN is that it has to perform the forward pass for each object proposal without sharing computations. For proposal generation they analyze different methods, but Selective Search performs best. In order to accelerate the approach, the Fast R-CNN network takes as input an entire image and a set of object proposals. First, the entire input image is processed with several convolutional and max pooling layers to produce a stack of feature maps (conv feature maps). For each object proposal a RoI pooling layer extracts a fixed length feature vector from the feature map. Each feature vector is fed into a sequence of fully connected layers that finally branch into two sibling output layers: one produces the softmax probability estimates over the different object classes (classification) and another layer outputs four real-valued numbers representing the coordinates, for each of the object classes (regression). Processing the time consuming convolutional layers just once for each image, the already mentioned processing bottleneck is avoided. For this approach, the RoI pooling layer is the key element, which is the special case of SPP [HZRS15b]. By mapping the proposed regions of the input image to the conv feature maps, they avoid the need for re-computing the convolutional layers for each proposal. The RoI pooling layer is a good alternative to image warping and pools the proposed regions of the conv feature maps to a fixed length input vector for classification and bounding box regression.

Based on Fast R-CNN, Li et al. [LLS+16] adapted the general object detection task to person detection. Their approach is called Scale-Aware Fast R-CNN (SAF R-CNN). By considering the resulting feature maps (conv feature maps), they realize that persons with different spatial scales exhibit very different features. This fact of undesired large intra-category variance in features, makes them propose a scale-aware network architecture. Therefore they use a trunk of convolutional layers, similar to the Fast R-CNN. After the trunk, the conv feature maps are split into two branches. One branch is trained for detecting small person instances and the other branch is responsible for large-size instances. Both branches consist of several convolutional layers to produce feature maps for specific scales. On these scale-specific feature maps, RoI pooling is applied to generate fixed length feature vectors, used for classification and regression. The results of both sub-networks are combined by scale-aware weighting. Another method is proposed by Shrivastava et al. [Shr16]: they introduce online hard example mining for boosting the Fast R-CNN training.

A similar approach using Fast R-CNN is proposed by Najibi et al. [NRD16] with their muti-scale Grid of fixed bounding boxes with CNNs (G-CNN) method. They work without proposal algorithms. Instead they start with a multi-scale grid of fixed bounding boxes and train a regressor to iteratively move and scale elements of the grid towards objects. The object detection problem is re-formulated as regression problem, meaning to find the path from a fixed grid to boxes tightly surrounding the objects. With this method they are able to reduce the number of boxes which have to be computed compared to Fast R-CNN.

An advancement of Fast R-CNN is the very popular Faster R-CNN approach for object detection by Ren et al. [RHGS16]. Since this work is based on Fast R-CNN, the approach is reviewed in short and details are explained in Chapter 6. One key element of their work is the introduction of the RPN. They observe that the convolutional feature maps used by Fast R-CNN can also be used for generating region proposals. On top of these convolutional feature maps they construct the RPN by adding a few additional convolutional layers that simultaneously regress region bounds and determine objectness scores at each location on a regular grid. The RPN is a FCN and can be trained in an end-to-end fashion. End-to-end learning refers to omitting any hand-crafted intermediary algorithms and directly learning the desired solution of a given problem from the sampled training data. The main difference compared to Fast R-CNN is that Selective Search as region proposal method is replaced by the RPN. Instead of using pyramids of images or filters, Ren et al. introduce anchor boxes that serve as references at multiple scales and aspect ratios. Based on the conv feature maps that are the same for RPN and Fast R-CNN classification network, every point of the feature map represents one reference point (*anchor*). The computational performance gain is achieved by sharing the convolutional layers used for region proposal with the Fast R-CNN, leading to near cost-free feature maps for the classification network. Zhang et al. [ZLLH16] state with their observations that for person detection, the RPN as stand-alone achieves comparable results to the Faster R-CNN (RPN + Fast R-CNN). Therefore Zhang et al. propose to omit the Faster R-CNN classification network to improve the detector performance. While introducing a new pedestrian dataset called CityPersons, Zhang et al. [ZBS17] analyze the Faster R-CNN architecture and state it fails to handle small scale objects (persons). Thus, they propose five modifications: (1) changing the anchor scales, (2) input up-scaling, (3) finer feature stride, (4) ignore region handling, and (5) changing the solver used for training. Those modifications yield to a performance boost.

Redmon et al. [RDGF15] propose a similar approach to Ren et al. called You Only Look Once (YOLO). According to Figure 2.1 this architecture corresponds to the third row. YOLO divides the input image into a squared grid. If the center of an object is within a grid cell, that grid cell is responsible for detecting that object. Each grid cell predicts a certain number of bounding boxes and confidence scores for those boxes. These confidence scores represent the model's confidence that the predicted box contains an object. Like Faster R-CNN, YOLO is a FCN. Since there is no separation into region proposal generation and region classification, the result is achieved by forward passing the image only once, making this object detector very fast compared to other algorithms such as Faster R-CNN, which have to consider each region proposal in order to process one image. The main drawback is that each grid cell can only contain one object due to the architectural design.

Bappy and Roy-Chowdhury [BRC16] present a method of region proposal generation by considering a CNN that activates semantically meaningful regions in order to localize objects. These activation regions are used as input for a CNN to extract deep features. These features are utilized to train a set of class-specific binary classifiers to predict the object labels. In order to reduce the object localization error, the regression method of Felzenszwalb et al. [FGMR10] is adopted for better training results.

Cai et al. [CFFV16] propose an approach called Multi-Scale CNN (MS-CNN) inspired by the RPN of Ren et al. [RHGS16]. The MS-CNN consists of a proposal sub-network and a detection sub-network. As they want to improve the ability of the proposal sub-network to be more scale-invariant, they not only use the output feature maps of the last convolutional layers, but the feature maps of different intermediate convolutional layers as well. For each branch of the convolutional trunk, there are detection layers for bounding box regression and classifying class labels. All proposals of the different branches make up the resulting set of proposals. For training the branches in an end-to-end fashion a multi-task loss is introduced. Similar to the Faster R-CNN, each proposal of the proposal sub-network is processed by the object detection sub-network. Although the proposal network could work as a detector itself, it is not strong enough, since its sliding windows do not cover objects well. To increase detection accuracy, the detection network is added. Similar to Faster R-CNN, RoI pooling is utilized to extract fixed length feature vectors for each proposal, which are classified by a small classification network. Compared to Faster R-CNN, the feature maps are up-sampled before RoI pooling by using a deconvolutional layer. They state that this is necessary because higher convolutional layers respond very weakly to small objects.

An approach for using Faster R-CNN for multi-spectral person detection is proposed by Liu et al. [LZWM16]. They adopt the Faster R-CNN framework for the KAIST dataset [HPK+15]. Unlike the algorithms presented before, this one is based on the KAIST dataset, which provides multi-spectral videos, each frame containing one pair of VIS and IR images. Therefore, they introduce four different fusion architectures, which fuse the feature maps at certain intermediate convolutional layers. The Early Fusion fuses the feature maps after pool1, the Halfway Fusion after pool4 and the Late Fusion after pool5. Pool$X$ denotes the pooling layer of the Xth convolutional layer. In Chapter 6, the different fusion options are described in detail. For fusing two types of feature maps, the two stacks of feature maps are concatenated in channel dimension and for reducing the channel dimension, network-in-network (NIN) [LCY13] is applied. A fourth option for fusion is the Score Fusion, where the VIS and IR subnets are handled separately and the resulting scores are merged by equally weighting. The Halfway Fusion outperforms the other approaches.

3 Deep Learning Frameworks

This chapter introduces different frameworks used for deep learning. First, an overview of available frameworks is provided. A list of criteria, which need to be satisfied by a suitable framework is presented. Based on these criteria the most suitable framework for this work is chosen.

In Table 3.1 eleven deep learning frameworks are listed. All frameworks have different features that have to be taken into account. The first criterion is that the framework of choice has to be open-source. Only frameworks satisfying this constraint are considered in this table. A second important criterion is the core language of the framework. The core language enables assumptions about the computational performance of the appropriate framework. Additionally, it has to be considered if there is parallel computing support for frameworks, especially for those written in languages such as Python and Matlab, to reduce training runtime. Even if the core language itself is slow, this can boost the performance to an acceptable level. Computational performance is extremely important, especially for deep learning. Even on parallel computing architectures and C++-based frameworks, training of neural network models can last days, weeks, or months. The next column gives an overview of available binding languages, also called wrapper. Wrapper enable us to call the original framework functions via interfaces out of other languages. This can provide many advantages for debugging and enables visualizing the trained models. For example, a Matlab wrapper can be used to forward an image through the network and simultaneously visualize the filter weights and activations. In order to understand a given network architecture this is essential.

The remaining columns give some additional information: *CPU* and *GPU* provide information if the framework supports the usage of Central Processing Units (CPUs) and/or Graphics Processing Units (GPUs) for training and evaluating the CNNs models. Efficient model training requires support for Graphics Processing Unit (GPU). The column named *Network Layers* shows if all layers that are necessary for creating common CNN architectures are available. *Visualization Layers* indicates if all layers of a common CNN architecture can be visualized according to the techniques of Zeiler and Fergus [ZF14]. The last column (*Pre-trained Models*) indicates the possibility whether pre-trained models are provided for the appropriate framework or not. Pre-trained models are used for finetuning, to avoid training from scratch. In this way the low-level convolutional filters can be re-used and the weights adapted by training the filters of higher convolutional layers. Important is that the models of common architectures are provided, such as AlexNet [KSH12], GoogLeNet [SLJ+15], VGG-16, and VGG-19 [SZ15].

A green tick (✓) means the functionality is available. Orange ticks (○) indicate that the functionality can be transferred from other implementations or git branches, but is not implemented in the original implementation or master branch. A red cross (✗) signals that this functionality is not available and has to be implemented by yourself.

There are some minor decision criteria. Most of the frameworks enable their users to define the network architecture in a declarative way and a few imperatively. Declarative definitions mean that the network is defined by specifying blocks, which have a certain functionality controlled by parameters and connecting them. For example, a convolutional layer gets an input, applies filters on the input, and outputs the resulting feature maps. Furthermore, it is able to perform backpropagation. The behavior of the layer can be controlled and adapted by setting parameters such as kernel size, stride and padding. For users, the convolutional layer is a black box that can be parameterized. On the opposite, if the convolutional layer is defined imperatively, each instruction needed to perform the convolution operation has to be written manually. Therefore the declarative strategy is considered as the most suitable, since one can define the convolutional network in an abstract manner without caring about implementational details. All models except for Torch7 support declarative model definitions. Therefore, the Torch7 framework of Collobert et al. [CKF11] is rejected. MXNet by Chen et al. [CLL+15] is the only framework, which provides both possibilities of network architecture definition.

The KAIST dataset [HPK+15] is a rather small dataset for training a CNN. That is the reason why pre-trained models have to be used together with finetuning [WFHB16]. Therefore, all frameworks which do not provide the appropriate pre-trained models are discarded: cuda-convnet2 [Kri14], Decaf [DJV+14], Pylearn2 [GWF13], and Theano [BLP+12]. The OverFeat framework of Sermanet et al. [SEZ+13] is rejected due to missing GPU support and lack of available pre-trained models. The Darknet framework of Redmon [Red13] is not considered further due to sparse documentation and a limited number of available pre-trained models such as YOLO [RDGF15]. Since the baseline approach of Zhang et al. [ZLLH16], used in this work, is implemented in Matlab, a framework is preferred that is implemented in Matlab

Framework	License	Core Language	Binding Language	CPU	GPU	Network Layers	Visualization Layers	Pre-trained Models
Caffe [JSD+14]	BSD	C++	Python/Matlab	✓	✓	✓	○	✓
cuda-convnet2 [Kri14]	Apache License 2.0	C++	Python	✗	✓	✓	✗	✗
Darknet [Red13]	free use	C	-	✓	✓	✓	✗	○
Decaf [DJV+14]	BSD	Python	-	✓	✗	✗	✗	✗
MatConvNet [VL15]	BSD	Matlab	-	✓	✓	✓	○	✓
MXNet [CLL+15]	Apache License 2.0	C++	Python/R/Julia/Go	✓	✓	✓	✗	✓
OverFeat [SEZ+13]	unspecified	C++	Python	✓	✗	✗	✗	○
Pylearn2 [GWF13]	BSD	Python	-	✓	✓	✗	✗	✗
TensorFlow [AAB+16]	Apache License 2.0	C++	Python	✓	✓	✓	✗	✓
Theano [BLP+12]	BSD	Python	-	✓	✓	○	✗	✗
Torch7 [CKF11]	BSD	Lua	-	✓	✓	✓	✗	✗

Table 3.1: Comparison of current deep learning frameworks.

or provides an appropriate wrapper. This reason, and missing layers for network visualization according to Zeiler and Fergus [ZF14], lead to excluding the MXNet framework of Chen et al. [CLL+15] and the TensorFlow framework of Abadi et al. [AAB+16].

The two remaining frameworks are the Caffe framework of Jia et al. [JSD+14] and the MatConvNet of Vedaldi and Lenc [VL15]. Both frameworks can be used under the Berkeley Software Distribution (BSD) license, have CPU and GPU support, provide all layers that are required for the CNN approach, and the most common pre-trained models are available. Although MatConvNet is implemented in Matlab, its speed is comparable to Caffe when using the GPU support. The Caffe framework is chosen for this work, as it provides C++, Python and Matlab interfaces and offers more flexibility than the MatConvNet. Furthermore, Caffe has a large online community, which can be very helpful for getting the framework started and for further support. According to own experiences, questions in the Caffe user group are answered within a few days. Regarding the visualization layers, there is no unpooling layer in the original version, but it is possible to get this functionality from another source. This work is based on the source code of Zhang et al. [ZLLH16].

4 Datasets

This chapter gives an overview of existing datasets for person and/or pedestrian detection. First, the most important datasets and their characteristics are reviewed in Table 4.1. Next, one VIS image dataset (Caltech [DWSP12]), one IR dataset (CVC-09 [SRV+11]) and one multi-spectral dataset (KAIST [HPK+15]) are chosen according to their characteristics. The KAIST dataset is the main training and evaluation dataset, whereas the other two datasets are used for pre-finetuning [WFHB16]. The remaining sections present the detailed characteristics of the chosen datasets.

4.1 Dataset Overview

Table 4.1 lists all potential datasets. Several other datasets are not considered, as they provide only gray-scale images instead of RGB VIS images or the number of annotations is not sufficient. For training a CNN one wants to preferably use datasets acquired by moving platforms. In this way there is a constantly changing background and thus more variance when sampling negative samples. A static background can bias the set of negative samples. The KAIST dataset, which is the main evaluation dataset, has bounding boxes that are axis-aligned. Bounding boxes are rectangular boxes that are related to a certain region of an image and usually are labeled according to the content of that region. The bounding boxes are axis-aligned if their edges are parallel to the coordinate axes and thus they are not rotated. Datasets which are axis-aligned are preferred.

The following datasets are not considered due to their small number of annotation labels: OSU Thermal Pedestrian dataset (IR) of Davis and Keck [DK07], INRIA pedestrian dataset (VIS) of Dalal and Triggs [DT05]. The CVC-14 dataset (VIS+IR) of González et al. [GFS+16] is a multi-spectral dataset, but instead of RGB images only gray-scale images are provided. Hence, the dataset is discarded, as well as the Daimler dataset (VIS) of Enzweiler and Gavrila [EG09]. The LSIFIR dataset (IR)

Dataset Name	**Citation**	**Number of Annotations**	**VIS (RGB) Images**	**IR Images**	**Image Resolution** (in Pixels)	**Comments**
BU-TIV - Atrium	[WFTB14]	13,544	✗	✓	512 × 512	fixed camera
BU-TIV - Lab	[WFTB14]	87,485	✗	✓	512 × 512	fixed camera
BU-TIV - Marathon	[WFTB14]	265,069	✗	✓	1,024 × 512	fixed camera no *occluded* labels
Caltech	[DWSP12]	223,798	✓	✗	640 × 480	moving camera
CVC-02	[GSPL10]	13,181	✓	✗	640 × 480	moving camera
CVC-09	[SRV+11]	48,917	✗	✓	640 × 480	moving camera no *occluded* labels
ETH	[ELSa08]	13,247	✓	✗	640 × 480	moving camera
FLIR	[PLCS14]	6,743	✗	✓	324 × 256	hand-held camera
KAIST	[HPK+15]	81,469	✓	✓	640 × 512	moving camera
KITTI	[GLU12]	11,256	✓	✗	multiple	moving camera
LSIFIR	[OPN+13]	8,246	✗	✓	164 × 129	moving and fixed camera
TUD-Brussels	[WWS09]	1,421	✓	✗	640 × 480	moving camera

Table 4.1: Popular public datasets for person and pedestrian detection.

provided by Olmeda et al. [OPN+13] has a very low resolution compared to other IR datasets and is therefore excluded.

The listed datasets in Table 4.1 are considered and chosen according to several criteria. As there is no other multi-spectral dataset, the KAIST dataset of Hwang et al. [HPK+15] is chosen as main training and evaluation dataset. As VIS datasets there are following possibilities: Caltech dataset of Dollár et al. [DWSP12], the CVC-02 dataset of Gerónimo et al. [GSPL10], the ETH pedestrian dataset of Ess et al. [ELSa08], the KITTI dataset of Geiger et al. [GLU12], and the TUD-Brussels dataset of Wojek et al. [WWS09]. The Caltech dataset is chosen due to the large number of labels, which is essential for achieving good training results. A second argument for choosing the Caltech pedestrian dataset is its usage a as training and/or testing dataset in recent works [BOHS14] [ZBS15] [HWS+16] [ZLLH16] [ZBO+16] [ZBS17]. The following IR datasets are listed in Table 4.1: the BU-TIV dataset (MWIR) of Wu et al. [WFTB14], the CVC-09 dataset (LWIR) of Socarrás et al.

[SRV+11], the FLIR dataset (LWIR) of Portman et al. [PLCS14], and the LSIFIR dataset (LWIR) of Olmeda et al. [OPN+13]. As denoted in brackets, the remaining datasets comprise of MWIR or LWIR images and since the KAIST dataset consists of LWIR images, the BU-TIV datasets are rejected. The CVC-09 dataset is selected for pre-finetuning due to the large number of annotated bounding boxes compared to the other two IR datasets.

Pre-trained network models are used for training the CNNs of this work. As these pre-trained models are commonly trained on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) dataset of Russakovsky et al. [RDS+15], the characteristics of this dataset are considered in short. This work performs object detection, but pre-training on the large ImageNet dataset trains the CNN model for the task of object classification. The training on such large datasets helps forming suitable filter weights, which can be used as initialization for training a CNN, even if the pre-training is performed for another task. The training set of the ImageNet dataset consists of 1,281,167 images and each image has a label out of 1,000 possible classes. The validation set has 150,000 images labeled with one of the 1,000 object categories. Other object detection datasets are the PASCAL dataset of the VOC challenge by Everingham et al. [EVW+10] and the COCO dataset of Lin et al. [LMB+14]. Usually the CNN models pre-trained on the ImageNet dataset are utilized.

4.2 KAIST Multi-spectral Pedestrian Dataset

The KAIST Multispectral Pedestrian Benchmark dataset introduced by Hwang et al. [HPK+15] is used as main dataset for experiments and evaluations. In Table 4.2 different sub-datasets of the KAIST dataset, which are used in the remainder of this thesis, are introduced. The KAIST dataset has a fixed image size of 640×512 pixels.

To prevent the channel features detectors from overfitting, Hwang et al. re-sample the original dataset to reduce the amount of the training and testing data. The original re-sampling by Hwang et al. is given with skip 20. This means that out of 20 frames one image is added to the re-sampled smaller dataset. *KAIST-test-Reasonable* is a testing sub-dataset, which is re-sampled with skip 20 and contains persons with size equal to or larger than 50 pixels. In the original work of Hwang et al. they use persons of height equal to or larger than 55 pixels. To provide the same height threshold for Caltech, CVC-09 and KAIST dataset, the height threshold is

KAIST [HPK+15] (640 × 512)	**Number of Annotations**	**Number of Images** (Skip)	**Annotation Density** (Annotations per Image)
KAIST-test-Reasonable	1,639	2,252 (20)	0.73
KAIST-test-All	2,019	2,252 (20)	0.90
KAIST-train	1,394	2,500 (20)	0.56
KAIST10x-train	13,853	25,086 (2)	0.55

Table 4.2: Overview of different KAIST sub-datasets and their characteristics.

adapted to 50 pixels. All testing subsets comprise of annotations of persons, which are not occluded or partially occluded. Heavy occluded persons are excluded from the testing data and handled as ignore region. As the detectors are analyzed for a increased KAIST testing sub-dataset, containing small-scale persons, the *KAIST-test-All* comprises all annotated persons equal to or larger than 20 pixels. The training sub-datasets contain persons equal to and larger than 50 pixels, with only non-occluded persons, to avoid confusing the detector during training with occluded samples. Inspired by Nam et al. [NDH14] and applied by Hosang et al. [HOBS15] for the Caltech dataset, Liu et al. [LZWM16] extended the training subset by reducing the skip from 20 to 2. This measure increases the number of images linearly by factor 10. This bigger training sub-dataset is called *KAIST10x-train* according to the naming of Hosang et al.. The KAIST dataset contains images acquired at daytime and nighttime.

The distribution of the annotated bounding box heights is shown in the histogram in Figure 4.1. The red line separates the bounding boxes with height smaller than 50 pixels from those that are equal to or larger than 50 pixels. There is an accumulation of bounding boxes with a height around 60 pixels. The mean bounding box height for *KAIST-test-Reasonable* sub-dataset is $\overline{h} = 88.31$ pixels. According to the height histogram, the height threshold of the *KAIST-test-All* sub-dataset is set from 50 pixels to 20 pixels to make use of the complete dataset except for the heavy occluded labels.

In Figure 4.2, the bounding boxes of the KAIST testing subsets are categorized w.r.t. their occlusion level. The following categories are used: with the label *no occlusion* fully visible persons without any occlusions are tagged. Annotated persons, which are labeled with *partial occlusion* are at least 65 % visible. The label *heavy occlusion* denotes persons being visible by at least 20 %. Heavy occluded persons can be occluded by up to 80 % such that only the head and shoulder parts or feet are visible. Based on Figure 4.2, not only bounding box labels of 50 pixels height and

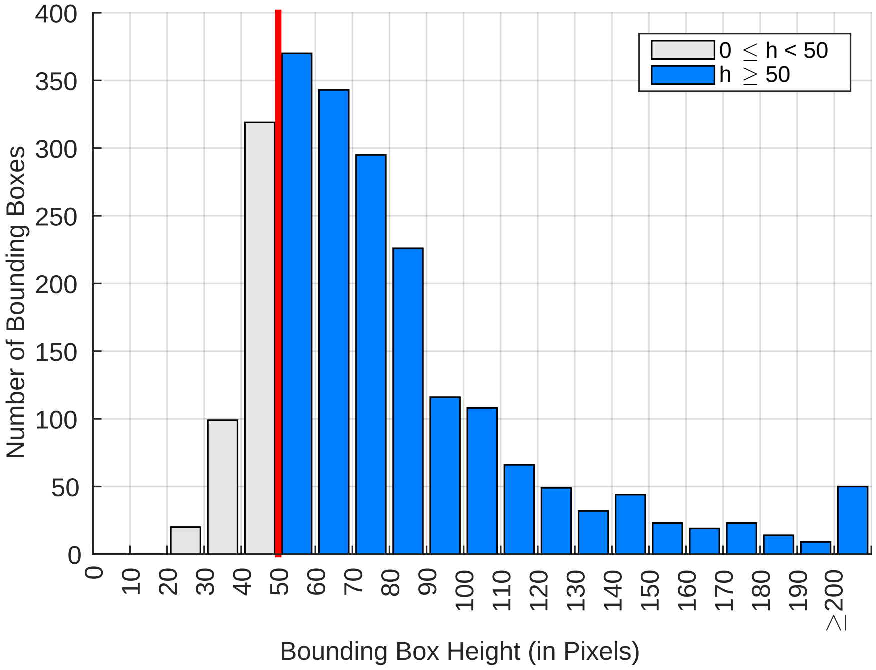


Figure 4.1: The histogram shows the distribution of annotated bounding box height for the *KAIST-test-Reasonable* sub-dataset.

higher should be considered. Thus, in this thesis small-scale persons are evaluated. In order to include small-scale persons, the *KAIST-test-All* testing sub-dataset is introduced.

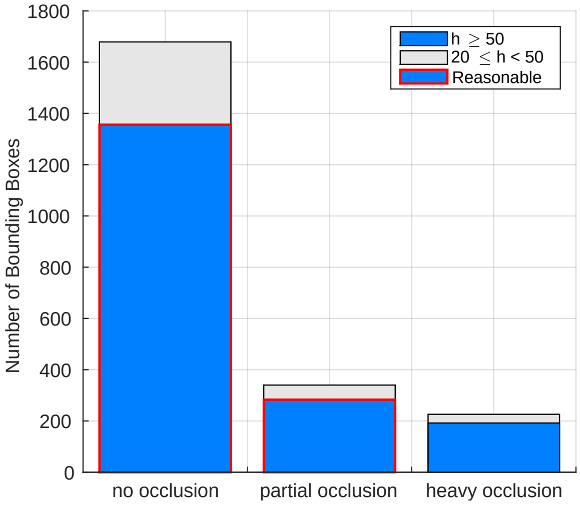


Figure 4.2: The occlusion histogram depicts the different fractions of the testing dataset. The blue bars represent bounding box heights equal to or larger than 50 pixels, whereas the grey bars stand for all bounding boxes smaller than 50 pixels. The red-rimmed parts of the bars make up the *KAIST-test-Reasonable* sub-dataset. All bounding boxes are categorized w.r.t. their occlusion level.

4.3 Caltech VIS Pedestrian Dataset

The Caltech Pedestrian Detection Benchmark dataset is a popular dataset for person detection in the field of autonomous driving and driver assistance systems. The dataset has been published by Dollár et al. [DWSP09] [DWSP12]. Similar to the previous section, the sub-datasets for training and evaluation are introduced in Table 4.3. The Caltech dataset has a fixed image size of 640 × 480 pixels. With the Caltech dataset, Dollár et al. also introduce the so-called *ignore regions*, which are similar to annotation boxes, but mark regions in the image which contain information that may confuse the detector training. An ignore region is for example a crowd of persons which cannot be labeled separately or a bicyclist that has similar appearance but should have the label 'bicyclist'. There are two reasons to use ignore regions. First, they are used to exclude critical regions for generating the training samples. Thus, it is guaranteed to avoid data that confuses the training procedure. The second reason affects the evaluation part. For matching between Ground Truth (GT) and detection boxes, detections that match to ignore GT boxes do not count as TPs and unmatched ignore GT boxes do not count as FNs.

Caltech [DWSP12] (640 × 480)	**Number of Annotations**	**Number of Images** (Skip)	**Annotation Density** (Annotations per Image)
Caltech-test-Reasonable	1,076	4,024 (30)	0.27
Caltech-test-All	2,590	4,024 (30)	0.64
Caltech-train	1,631	4,250 (30)	0.38
Caltech10x-train	16,376	42,782 (3)	0.38

Table 4.3: Overview of different Caltech sub-datasets and their characteristics.

Dollár et al. define the *Caltech-test-Reasonable* sub-dataset to contain annotated bounding boxes equal to or larger than 50 pixels. The labeled persons are not occluded or partially occluded. The testing set is extended by considering all bounding boxes with size equal to or larger than 20 pixels. The height threshold of 20 pixels is set according to the extended KAIST sub-dataset. Thus, it is ensured to have annoations of same height for training and evaluation on the KAIST and Caltech dataset. The extended Caltech testing sub-dataset is called *Caltech-test-All*. The standard training and evaluation sub-datasets consider one out of 30 images (*Caltech-test-Reasonable*, *Caltech-test-All* and *Caltech-train*). As for the KAIST sub-datasets, the *Caltech-train* as well as the *Caltech10x-train* subset contain bounding boxes equal to or larger than 50 pixels of persons that are non-occluded. As done for the KAIST-dataset and inspired by Zhang et al. [ZBS15] and Hosang et al. [HOBS15] the extended training sub-dataset (*Caltech10x-train*) is created by reducing the skip from 30 to 3.

In Figure 4.3 the distribution of bounding box heights for the testing subset of the Caltech dataset is visualized. The red line depicts the separation between the *Caltech-test-Reasonable* and the rest of the testing subset. There is a peak of bounding boxes with heights around 40 pixels. Compared to the KAIST dataset this means a shift by around 20 pixels. The Caltech dataset consists of smaller annotated bounding boxes compared to KAIST. This can be recognized as well when considering the drop of the mean bounding box height from $\overline{h} = 82.96$ pixels (*Caltech-test-Reasonable*) to $\overline{h} = 54.33$ pixels (*Caltech-test-All*). Furthermore, by considering the whole Caltech dataset and adapting the detectors to better localize and classify small-scale persons, the performance of person detection in general would be improved.

As in Section 4.2 the bounding boxes of the Caltech dataset are categorized w.r.t. their occlusion level. The following categories are considered: *no occlusion*, *partial occlusion* and *heavy occlusion*. The occlusion histogram shows that the number of

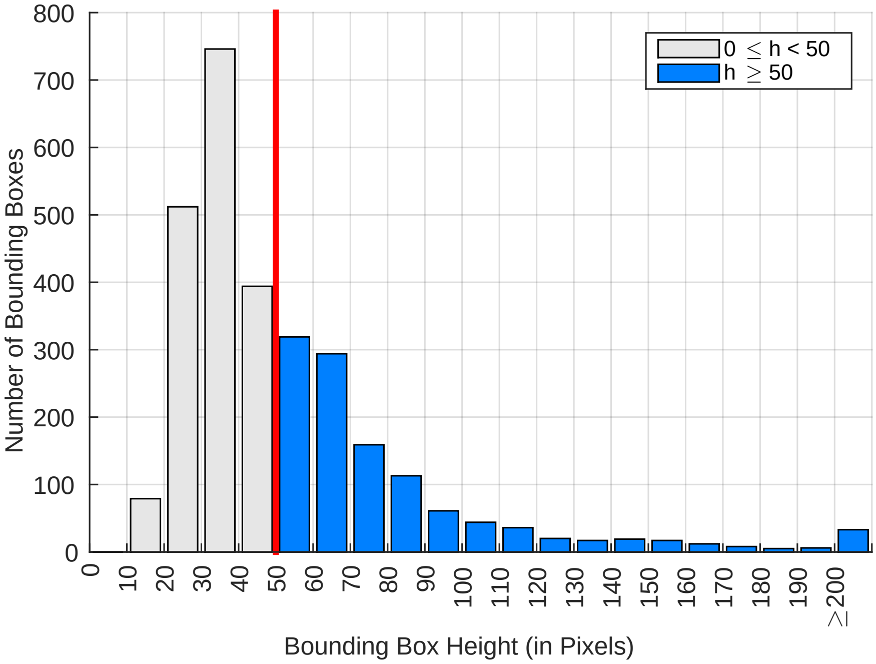


Figure 4.3: The histogram shows the distribution of annotated bounding box height for the *Caltech-test-Reasonable* sub-dataset.

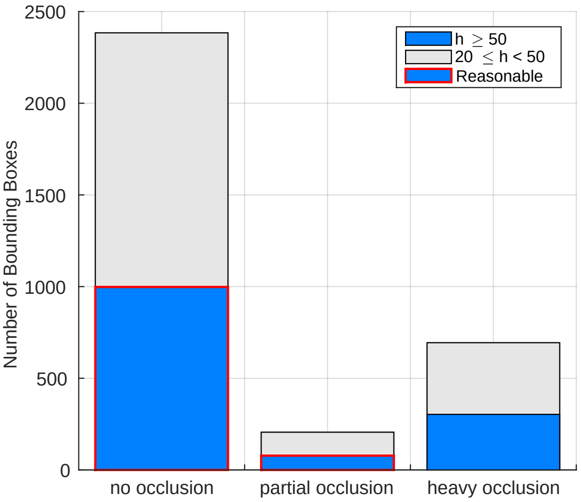


Figure 4.4: The occlusion histogram depicts the different fractions of the testing dataset. The blue bars represent bounding box heights equal to or larger than 50 pixels, whereas the grey bars stand for all bounding boxes smaller than 50 pixels, but larger than 20 pixels. The red-rimmed parts of the bars make up the *Caltech-test-Reasonable* sub-dataset. All bounding boxes are categorized w.r.t. their occlusion level.

annotations of the dataset can be doubled by additionally using bounding boxes of height smaller than 50 pixels. Recent works only consider the *Reasonable* testing subset for evaluation. Thus, there is much space left for improvements, although the MR of current baseline results is around 10 %. Using the bigger testing subset should encourage to improve the detectors for small-scale person detection and occlusion handling. A recently published dataset, which is compared to the Caltech dataset is published by Zhang et al. [ZBS17] and is called CityPersons.

4.4 CVC-09 IR Pedestrian Dataset

Similar to the other two datasets the CVC-09 FIR Sequence Pedestrian Dataset of Socarrás et al. [SRV+11] is introduced as well as the sub-datasets for training and evaluation in Table 4.4. This dataset is a Long-Wave Infrared (LWIR) dataset with daytime and nighttime scenes, similar to the KAIST dataset. The CVC-09 dataset has a fixed image size of 640 × 480 pixels.

CVC-09 [SRV+11] (640 × 480)	**Number of Annotations**	**Number of Images** (Skip)	**Annotation Density** (Annotations per Image)
CVC-test-Reasonable	1,018	432 (20)	2.36
CVC-test-All	1,052	432 (20)	2.44
CVC-train	1,482	420 (20)	3.53
CVC10x-train	15,058	8,418 (2)	1.79

Table 4.4: Overview of different CVC-09 sub-datasets and their characteristics.

Since there is no pre-defined re-sampling of the dataset, the re-sampling similar to the KAIST is adopted. To be comparable to the other two datasets, the goal is to have approximately the same number of annotation labels. Experimentally an appropriate skip is determined: one image out of 20 is sampled for the smaller subset of the dataset. Similar to the Caltech and KAIST dataset, the *CVC-test-Reasonable* sub-dataset is defined to contain annotated bounding boxes equal to or larger than 50 pixels and the *CVC-test-All* subset consists of bounding boxes equal to or larger than 20 pixels. By definition, both subsets have labels that contain non-occluded or partially occluded persons, although the CVC-09 dataset does not contain any occluded persons. Similar to the *CVC-test-Reasonable* sub-dataset, the small training sub-dataset *CVC-train* has skip 20. For generating a ten times bigger training subset that is called *CVC10x-train*, the skip is reduced from 20 to 2.

The approximately same number of labels ensures us to consistently evaluate the pre-finetuning approaches. In comparison to the other two datasets the annotation density is much higher. While the KAIST and the Caltech dataset contain less than one annotation label per image, the CVC-09 dataset contains more than two labels on average.

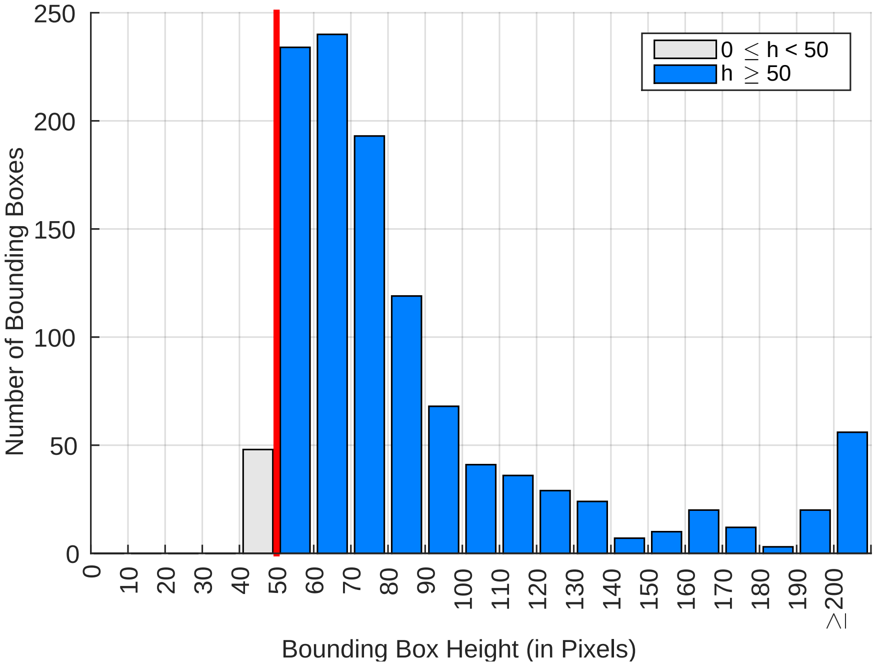


Figure 4.5: The histogram shows the distribution of annotated bounding box height for the *CVC-test-Reasonable* sub-dataset.

With an average bounding box height $\overline{h} = 91.34$ pixels for the *CVC-test-Reasonable* sub-dataset, the bounding boxes are larger than those of the other two datasets. Considering the histogram in Figure 4.5, it can be recognized that the distribution peak is shifted towards larger bounding box heights of around 70 pixels. Nearly all annotated bounding boxes belong to the *CVC-test-Reasonable* sub-dataset. This may bias the training results.

Figure 4.6 evidently shows that there are no occluded persons contained in the CVC-09 dataset. When analyzing this dataset visually, one recognizes that occluded persons are labeled as non-occluded persons or they are not labeled at all. There are neither occlusion labels nor ignore regions like for the Caltech dataset. This may harm the training procedure by generating occluded positive training samples or negative samples containing persons.

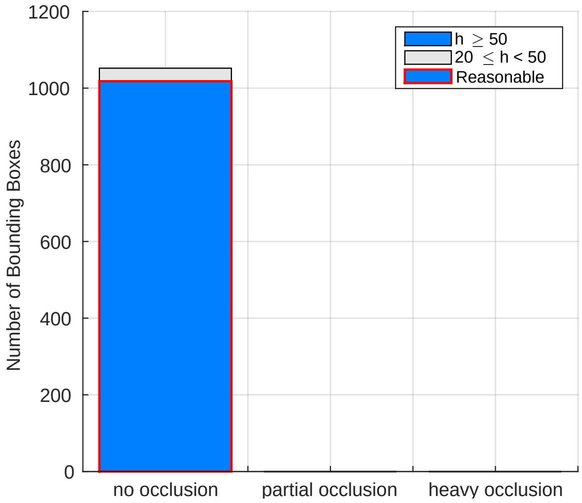


Figure 4.6: The occlusion histogram depicts the different fractions of the testing dataset. The blue bar represents bounding box heights equal to or larger than 50 pixels, whereas the grey bar stands for all bounding boxes smaller than 50 pixels. The red-rimmed part of the bar makes up the *CVC-test-Reasonable* sub-dataset. All bounding boxes are categorized w.r.t. their occlusion level.

4.5 Discussion

In the last section of this chapter, general challenges of the datasets are emphasized. Example images of the three datasets are used to illustrate these challenges. Furthermore, several works dealing with a deeper analysis of the datasets are presented.

Zhang et al. [ZBO+16] provide a good overview of person detection challenges. Therefore, they analyze the Caltech dataset. As reasons for False Positive (FP) detections they e.g. mention multiple detections of the same person (bad localization), vertical structures, or tree leaves (confusion). Possible sources of False Negatives (FNs) are small-scale persons, side views of persons, cyclists, or occlusions. Most likely sources are named first. Zhang et al. provide a human baseline for the Caltech dataset as well as improved annotations. They categorize error sources into three different error types: localization, background confusion, and annotation errors. They state that poor annotation alignment, missing annotations, or false annotations can crucially harm the training procedure. As introduced by Dollár et al. [DWSP09], Zhang et al. motivate the consistent usage of ignore regions. Hoiem et al. [HCD12] analyze error sources for object detection in general and reach similar conclusions. They see occlusions, large intra-class variation, various object poses, confusion with similar objects, localization errors, and textures in the background as common sources for missed or false detections. Ponce et al. [PBE^{+}06] state that the limited range of object variability within the datasets is the reason for object recognition errors. This object variablitiy comprises object appearance, different camera viewpoints, object orientation and position in the image, as well as small occlusions and background clutter. In their work, González et al. [GFS^{+}16] argue that the total number of pedestrians present in the sequences, the number of occlusions, the distribution of the pedestrian distance to the camera (bounding box size in pixels), the type of background (static or dynamic), the frame resolution, and the frame rate are factors that clearly have an influence on the results. For comparing datasets to each other, Torralba and Efros [TE11] perform cross-dataset generalization evaluation. They call the variance of different datasets the dataset bias, and propose several evaluation techniques. Deng et al. [DDS^{+}09] present a method for analyzing the dataset diversity by computing an average image. They try to analyze the constructed ImageNet dataset for the goal that objects in images should have variable appearances, positions, view points, poses as well as background clutter and occlusions. With computing the average image per class, in the best case (diverse images) the average image results in a blurrier average image or even a gray image. If the image provides little diversity, the average image is more structured and sharper.

Figure 4.7 shows four example images taken from the KAIST dataset. The red boxes represent the ground truth bounding boxes. The green arrow in Figure 4.7 (a) points at a person that is not labeled (missing annotation). The red bounding boxes of the persons on the stairs, are between 50 and 60 pixels in height. The yellow arrow marks a sitting person with a missing bounding box. This person could be labeled as an ignore region or as person, because on one side it is a person, but on the other side this person has an atypical pose and therefore could confuse the training process. Figure 4.7 (b) shows a difficult decision, whether to mark statues as persons or as ignore regions. Zhang et al. [ZBO^{+}16] define human shapes that are not persons such as posters or statues as ignore areas. But they have to be marked consistently.

Figure 4.7 (c) shows the usage of ignore regions due to groups of people or occluded persons (orange bounding boxes). The green arrows point at unlabeled persons which should be labeled as ignore regions if the height is to small, and as person otherwise. Figure 4.7 (d) conveys an impression of challenges in person detection during nighttime. The main difference between the KAIST and the

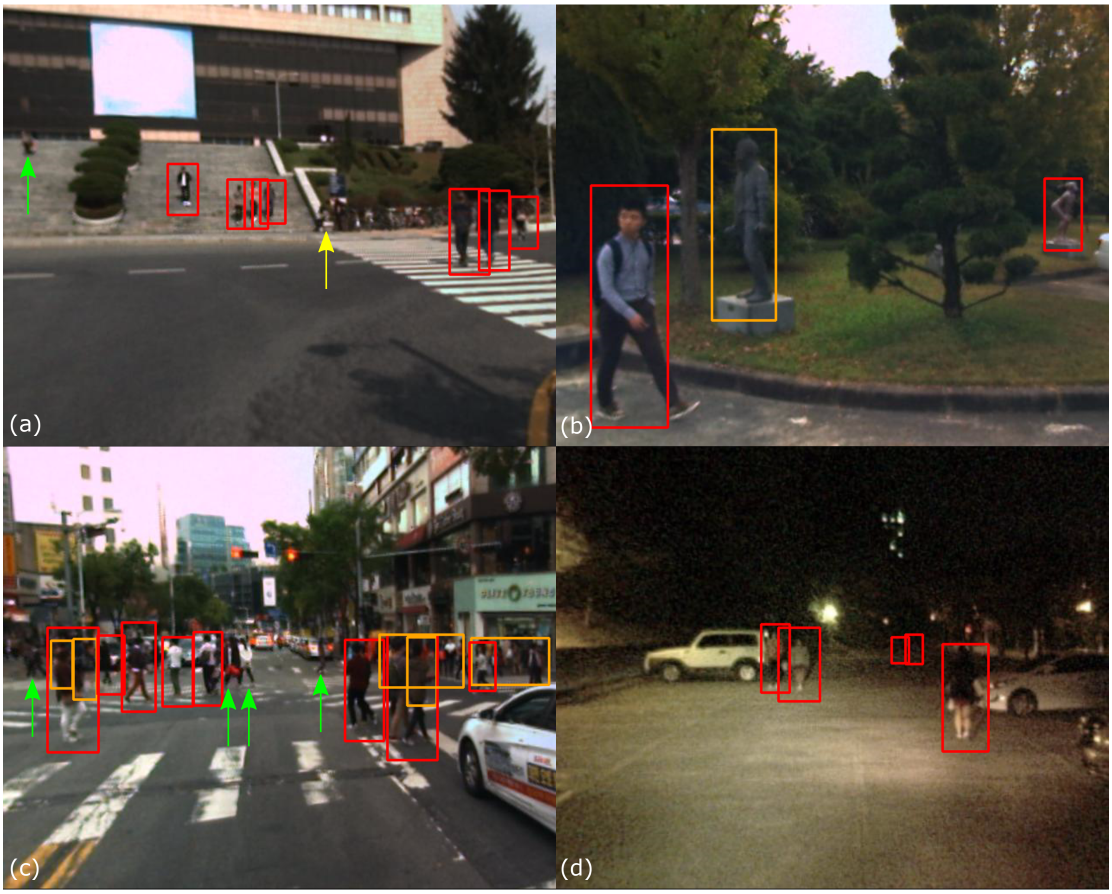


Figure 4.7: Example images of KAIST dataset. Red boxes show the GT data and orange boxes ignore regions. Green arrows point at persons without labels and yellow arrows at regions which should be marked as ignore regions.

Caltech dataset if only considering the VIS images is that the KAIST dataset contains scenes acquired at daytime and nighttime, whereas the Caltech dataset only contains images acquired at daytime. Additionally, missing, bad located, or false annotations affect the model during training and the results during evaluation. Thus, there is space for improving the annotations of the KAIST dataset.

For the Caltech dataset four example images are presented in Figure 4.8. Figure 4.8 (a) and (b) show small-scale persons. In Figure 4.8 (a), the orange bounding box has a height of 45 pixels and the person is occluded behind the pole of the traffic light, whereas the red box is 62 pixel high and therefore counts to the *KAIST-test-Reasonable* sub-dataset. Figure 4.8 (b) shows bounding boxes of height between 22 and 79 pixels. Figure 4.8 (c) shows appearance differences between small-scale

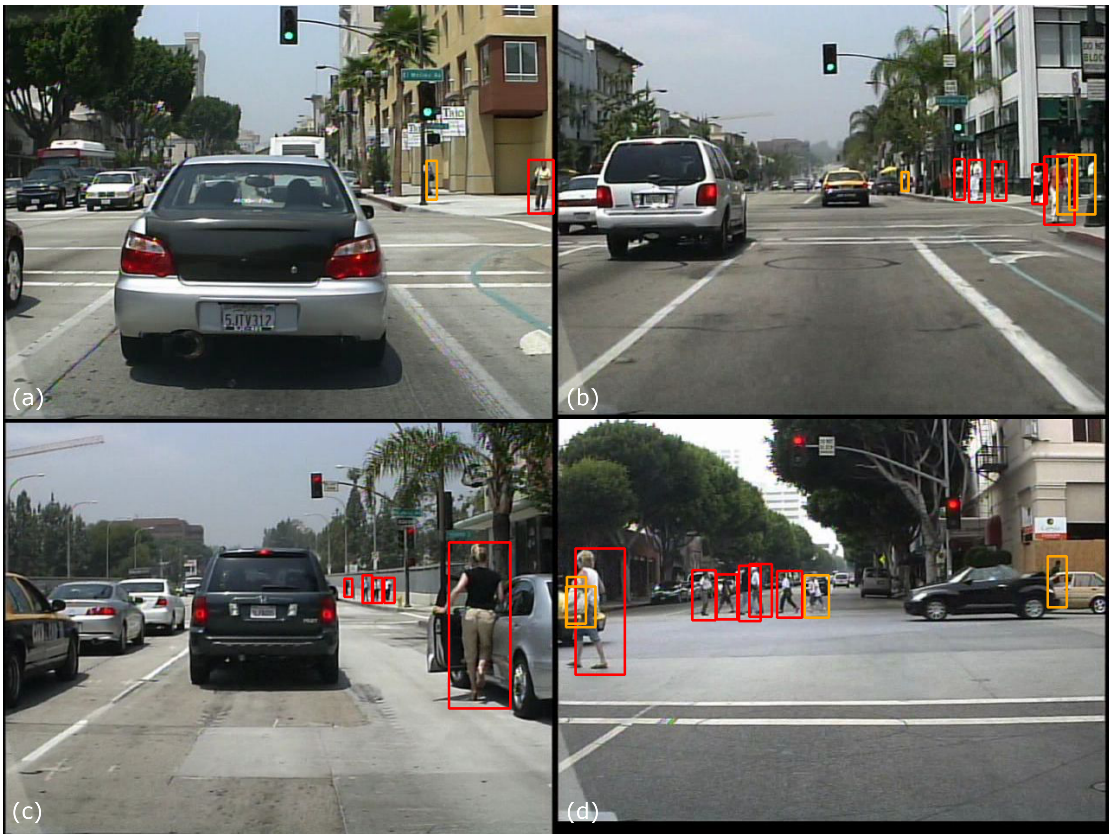


Figure 4.8: Example images of Caltech dataset. Red boxes show the GT data and orange boxes ignore regions.

and large-scale persons. More details of the large-scale persons can be recognized, whereas only the silhouettes of the small ones can be perceived. Figure 4.8 (d) is chosen for representing that even though the persons on the right and left side of the image are heavily occluded by a car, they are labeled as ignore regions to exclude this region from sample generation. It is stated that the Caltech dataset is a well labeled dataset for person detection, even more with the improved annotations of Zhang et al. [ZBO^{+}16].

Some representative examples of the CVC-09 dataset are shown in Figure 4.9. Figure 4.9 (a) and (b) demonstrate missing annotations. In Figure 4.9 (a), the two small persons (marked with green arrows) should be tagged with a person label as well as the two persons on right in Figure 4.9 (b). The motorcyclist marked with the yellow arrow should be annotated as ignore region or should get the label ’motorcyclist’, since the motorcyclist has a very similar appearance compared to persons, and therefore may affect the training process. In Figure 4.9 (b), the small

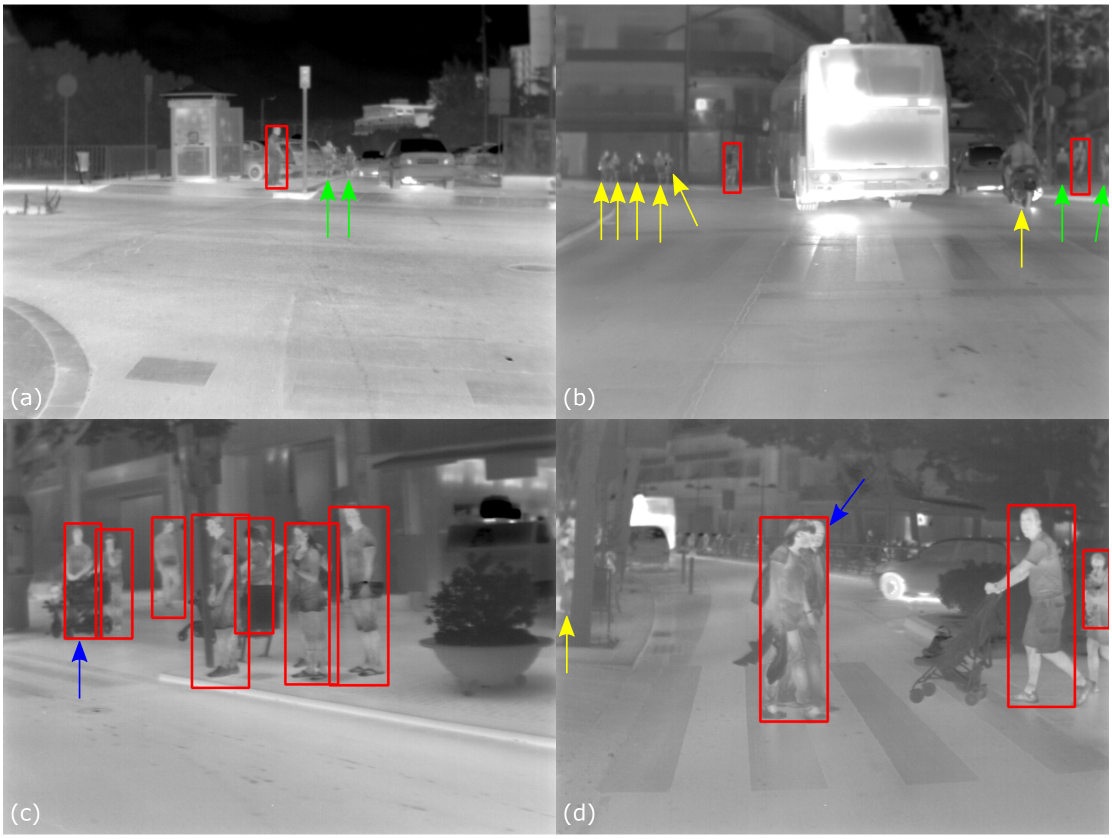


Figure 4.9: Example images of CVC-09 dataset. Red boxes show the GT data and orange boxes ignore regions. Green arrows point at persons without labels and yellow ones at regions which should be marked as ignore regions. The blue arrows indicate bounding boxes with occluded persons inside, which are labeled as non-occluded.

persons on the left, marked by the yellow arrows, have to be labeled as persons, too. Figure 4.9 (c) and (d) show occlusion situations (marked by blue arrows). In Figure 4.9 (c), the two most left persons are partially occluded by the buggy, and Figure 4.9 (d), there are two persons in the same label. The yellow arrow points at persons that are heavily occluded and therefore should be marked as ignore regions or heavily occluded. Compared to the other two datasets, the CVC-09 dataset is lacking of occlusion labels for each bounding box as well as the definition of ignore regions. Therefore, the models trained on this dataset may not perform as good as on Caltech, but different to the KAIST dataset, the quality of the IR images is better w.r.t. noise, and edge sharpness.

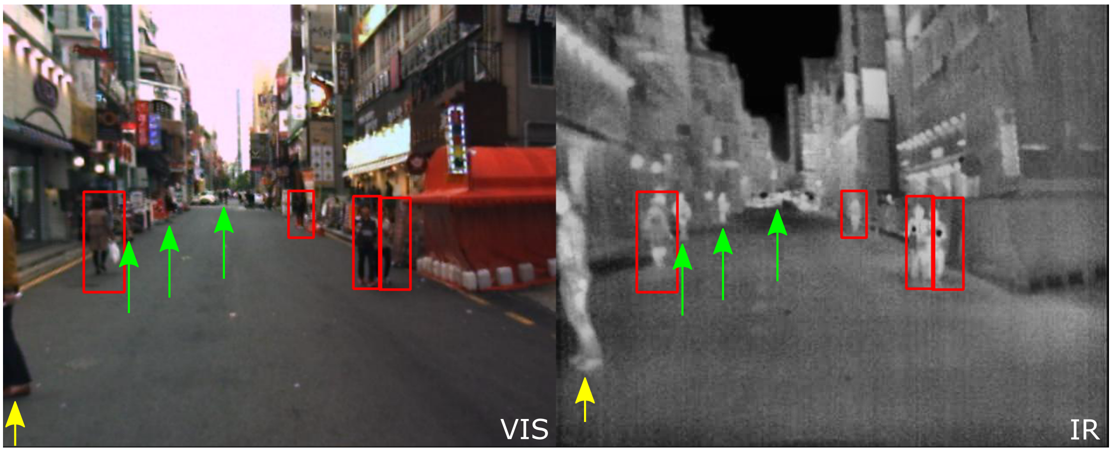


Figure 4.10: Example image from the KAIST dataset, which shows the registration error between VIS and IR images. This error increases towards the border areas of the image. Green arrows point at persons without labels and yellow ones at regions which should be marked as ignore regions.

The KAIST dataset set has three color channels (VIS) and one thermal channel (IR) per image. Figure 4.10 points out the challenge of image registration if images are acquired by two cameras simultaneously. An example is given in Figure 4.10 and marked with a yellow arrow. While in the VIS image the head and the second foot cannot be recognized, nearly the complete head and the second foot are visible in the IR image. Such imprecise registration can be recognized mostly towards the image borders. Registration errors are most critical for small-scale persons: while the bounding box fits exactly around the person in the VIS image, the bounding box in the IR image only contains half of the person. So, while the additional usage of the IR image generates complementary information to the VIS image, imprecise registration of the two images can destroy this positive effect during training.

5 Person Detection based on Filtered Channel Features

This chapter presents the two filtered channel features based detectors Aggregate Channel Features (ACF) and Checkerboards that are briefly introduced in Chapter 2. First, the principles of filtered channel features based detectors are recapitulated. Then, (1) the pipeline for pre-processing the input data for training the BDF are explained, (2) the BDF training parameters, and (3) how to use the detectors for proposal generation. The results of the filtered channel features based detectors are used as baseline algorithm for person detection.

5.1 Filtered Channel Features

The fundament of filtered channel features are the Integral Channel Features (ICF) by Dollár et al. [DTPB09]. Both methods (filtered channel features and ICF) generate a feature pool and a BDF is utilized for feature selection of the most discriminative features using AdaBoost [FS96] [FHTO00]. In literature, the terms BDF, boosted forest, or boosted decision trees are often used synonymously. BDFs are motivated by the Viola and Jones object detection framework [VJ01b]. The approach of organizing the different decision trees in a cascaded fashion is depicted in Figure 5.1.

A set of weak classifiers such as decision trees [Qui86] are used to evaluate the features extracted from an RoI. The first classifier (number 1) in Figure 5.1 is trained for the goal to eliminate a large number of negative examples with very little processing effort. All candidates classified as False (F) are not considered by the following weak classifiers and therefore are rejected as negative examples. The candidates classified as True (T) are passed to subsequent layers that eliminate additional negatives but require additional computation. The later the classifier is applied, the more complex features are evaluated for making a decision. The result

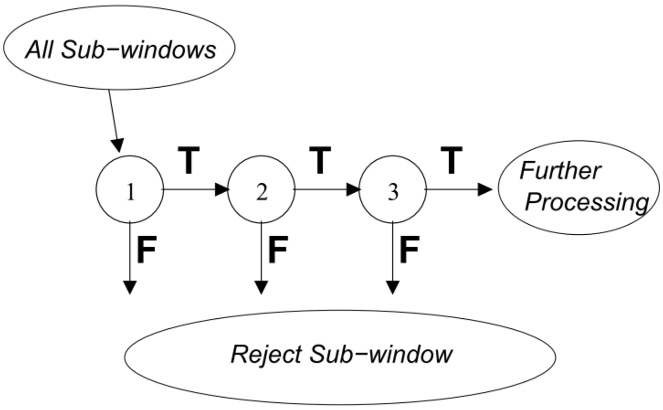


Figure 5.1: Schematic visualization of a classifier cascade [VJ01a].

of this cascaded structure is a strong boosted classifier consisting of a set of weak classifiers.

The BDF in this work utilizes the variant known as soft cascade [ZV08] [AFD13]. Stages in the cascade are constructed by training classifiers using AdaBoost and adjusting the cascade threshold to minimize FNs. Each stage in the cascade reduces the False Positive Rate (FPR), but also decreases the detection rate. With adjusting the threshold of the cascaded boosted classifier, the classifier is tuned to have a False Negative Rate (FNR) close to zero, but this increases the False Positive Rate (FPR) as well. This fact is exploited for using the filtered channel features based detectors as proposal generators.

5.1.1 Aggregated Channel Features (ACF)

Based on the ICF [DTPB09], Dollár et al. [DABP14] propose the ACF detector for object detection in general. ACF is the generalization of ICF. All detectors using channel features together with a BDF are summarized as filtered channel features based detectors.

In Figure 5.2 the ACF detector pipeline is illustrated by Rajaram et al. [ROBT16]. Given an input image I, several channels (feature channels) $C = \Omega(I)$ are computed. The following feature channels are generated: three channels of CIE (L*u*v*) color space (CIELUV) denoted by LUV (3 channels), normalized gradient magnitude M (1 channel) and gradient histogram O (6 channels), a simplified version of histogram of oriented gradients (HOG) [DT05]. Thus, there are 10 feature channels. Before the channels are computed, I is smoothed with a Gaussian filter. The

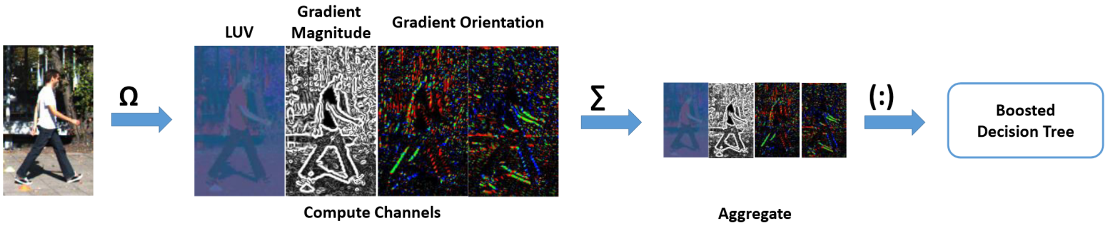


Figure 5.2: Pipeline of the ACF detector [DABP14]. LUV color channels, gradient magnitude, and histogram of gradient orientations are computed from the input image as $C = \Omega(I)$ [ROBT16].

processed 10 channels are sum pooled and passed through a Gaussian smoothing filter again to generate ACFs of lower resolution (denoted by $\sum$ in Figure 5.2). The created ACFs are used as single pixel lookups. To generate a feature vector used for BDF training, the channel features are vectorized (denoted by $(:)$). A multi-scale sliding window approach is applied on the channel features pyramid to detect objects of different scales. The feature pyramids are computed efficiently by approximating channel features of intermediate scales [DABP14], instead of scaling the input image and computing the channel features of each scale separately.

Dollár et al. [DABP14] use 2,048 depth-two decision trees and combine them by training with AdaBoost on $128 \cdot 64 \cdot 10/16 = 5,120$ candidate features (channel pixel lookups) in each 128 × 64 window (model size). The model size specifies the size of the sliding window used on different levels of the channel features pyramid. The split nodes in the decision trees are simple comparisons between feature values and learned thresholds. Usually, only a subset of the feature vector is used by the learned decision forest. AdaBoost performs both, the feature selection and learning the thresholds in the split nodes. For setting up a full image detection framework, Dollár et al. [DABP14] use the previously mentioned sliding window approach applied on multiple scales. NMS is applied to suppress multiple nearby detections. The NMS approach [DTPB09] suppresses the less confident of every pair of detections that overlap sufficiently w.r.t. the IoU overlap criteria [PBE+06].

5.1.2 Checkerboards Features

Starting from the ACF detector, Zhang et al. [ZBS15] generalize the basic architecture of ACF [DABP14] for different filtered channel features based approaches and therefore introduce the name of filtered channel features detectors. The input image is transformed into a set of feature maps as described in Subsection 5.1.1. The feature maps are sub-sampled by applying sum pooling over rectangular regions (aggregation). The resulting feature maps are vectorized and fed into a decision forest learned via AdaBoost. The key observation of Zhang et al. is that sum pooling of the feature maps can be re-written as convolution with a filter bank. Hence, the ACF detector consists of a single filter in its bank, corresponding to a uniform 4×4 pixels pooling region. Another filter bank is called Checkerboards. The Checkerboards filter bank is illustrated in Figure 5.3.

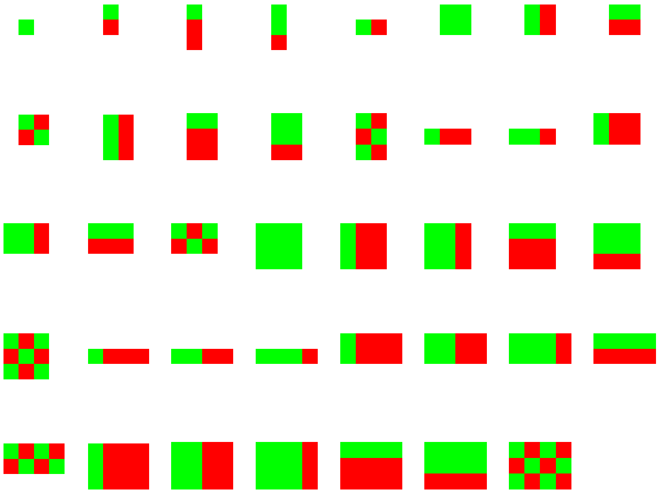


Figure 5.3: Illustration of the 39 4×3 filters belonging to the Checkerboards filter bank. {🟥 red, □ white, 🟩 green} indicate {-1, 0, +1}.

The Checkerboards detector is very similar to the ACF detector. The only difference is the applied filter bank used for generating the final feature pool that is used for the BDF. Zhang et al. [ZBS15] [ZBO+16] evaluate several channel feature detectors in their work. The Checkerboards detector outperforms the ACF detector by over 10 percentage points on the Caltech dataset.

5.1.3 Multi-spectral Channel Features

With introducing the KAIST dataset, Hwang et al. [HPK+15] also propose an adaptation of the ACF detector for multi-spectral images. As presented for the ACF detector, there are three different transformation types to generate the channel features: CIELUV color space (LUV), gradient magnitude (M), and histogram of gradient orientations (O). Hwang et al. test three versions of the modified ACF detector: (1) ACF+T, (2) ACF+T+TM+TO, and (3) ACF+T+THOG. Here, ACF denotes the aforementioned channel features (LUV+M+O) calculated for the RGB channels. T+TM+TO and T+THOG indicate the additional channel features extracted from the thermal channel. The T channel feature uses the thermal intensity directly. TM corresponds to the normalized gradient magnitude and TO to the histogram of gradient orientations of the thermal image. Therefore TM and TO use the same methods as already applied on color images for the standard color ACF. Instead, T+THOG uses the thermal image directly and the HOG feature [DT05] of the thermal image (THOG). Compared to TO with 6 histogram bins, THOG computes more gradient orientations and has additional normalization steps on the local histograms. Thus, the dimensionality of the feature vector passed to the BDF is higher for version (3) of the multi-spectral ACF. The ACF+T+THOG detector outperforms the other approaches and therefore the additional T+THOG channel features are used in the remaining work. Additionally, the Checkerboards method is adapted for multi-spectral images using the same channel features as ACF+T+THOG. The multi-spectral Checkerboards detector is denoted Checkerboards+T+THOG.

5.2 Implementation Details

This section provides implementation details and the parameters used for training and evaluating filtered channel features based detectors. For this thesis, the code provided by Hwang et al. [HPK+15] is used, which is based on Piotr's Matlab Toolbox [Dol]. For detection, the different channel features pyramid scales are computed from the input image. On these channel features pyramid scales, the sliding window paradigm is applied. Whereas the padded model size *modelDsPad* specifies the size of the sliding window on the channel features pyramid, the actual model size *modelDS* is a smaller window that is used to calculate the resulting detection coordinates. The coordinates of the detection results depend on the scales of the channel features pyramid and model size *modelDS*. The use of two model sizes allows to consider context information for classification [GDDM14] and simultaneously calculate the correct detection coordinates. The parameter *stride* specifies the spatial shift between neighboring detection windows. *cascThr* and *cascCal* parametrize the soft cascades and are described in detail later when using the filtered channel features based detectors as proposal generators. *nWeak* defines the number of weak classifiers (decision trees) per training stage and therefore implicitly determines the number of training stages (4 or 5 stages). *nPos* is the maximum number of positive windows to sample for each training stage and *nNeg* the maximum number of negative windows per stage. The parameter *nPerNeg* gives the maximum number of sampled negative windows per image. The negative samples are accumulated at each stage and the maximum number of accumulated negative samples is defined by *nAccNeg*. The negative samples for the first training stage are sampled randomly to provide diverse samples for training. The negative samples for the remaining training stages are sampled by applying the trained detector model after each stage, and using these detections for generating the additional negative samples for the next training stage. These detections are evaluated with the Intersection over Union (IoU) [EVW+10] overlap. Each detection box that has an IoU overlap of smaller than 0.1 is considered as negative sample. Since the negative samples are generated by the detector model that is improved at every training stage, this kind of generating negative samples is called hard negative mining. Data augmentation is applied by flipping every positive sample horizontally. Flipping as data augmentation is a popular technique to enlarge the training data [VHV+16] [WYL+15] [MG06] [LTCFS16] [CSVZ14] [BMTV13].

The parameters *pPyramid.nPerOct*, *pPyramid.nOctUp*, *pPyramid.nApprox*, *pPyramid.smooth*, and *pPyramid.pChns.shrink* affect the generation process of the channel features pyramid (channel features of different scales) w.r.t. the method proposed by Dollár et al. [DABP14]. The idea is to avoid computing the channel features on each scale of the input image pyramid separately. Instead, the channel features of a few scales are computed and the others are approximated. Therefore, the parameter *pPyramid.nPerOct* determines the number of scales per octave in the input image pyramid. Dollár et al. define an *octave* as the set of scales up to half of the initial scale. As a typical value, they suggest *nPerOct = 8* so that each scale in the pyramid is $2^{-1/8} \approx 0.917$ times the size of the previous. The largest scale in the pyramid is determined by *pPyramid.nOctUp*, which specifies the number of octaves computed above the original scale. *pPyramid.nApprox* determines how many intermediate scales are approximated using the approximation technique of Dollár et al. [DABP14]. Considering the previous example with *nPerOct = 8* and *nApprox = 7*, 7 intermediate scales are approximated and only power of two scales are actually computed starting from a scaled input image. *pPyramid.smooth* defines the radius for channel smoothing after the channel features are generated. *pPyramid.pChns.shrink* determines the amount to sub-sample the computed channels.

For a BDF trained with AdaBoost, there is the parameter *nWeak* that sets the number of weak classifiers used. To choose between the discrete (1) and real AdaBoost (0), the binary parameter *pBoost.discrete* is used. *pBoost.pTree.maxDepth* defines

Filtered Channel Features **Parameters**	**Original** **Config [HPK+15]**	**Config1** small / big model size	**Config2** small / big model size	**Config3** small / big model size
modelDs	20.5 × 50	20.5 × 50 / 36 × 96	20.5 × 50 / 36 × 96	20.5 × 50 / 36 × 96
modelDsPad	32 × 64	32 × 64 / 60 × 120	32 × 64 / 60 × 120	32 × 64 / 60 × 120
stride	4	4	6	4
cascThr	-1	-1	-1	-1
cascCal	0.005	0.005	0.005	0.005
nWeak	[32,128,512,2048]	[32,512,1024,2048,4096]	[32,512,1024,2048,4096]	[32,512,1024,2048,4096]
nPos	Inf	Inf	Inf	Inf
nNeg	5,000	10,000	10,000	10,000
nPerNeg	25	25	25	25
nAccNeg	10,000	50,000	50,000	50,000
pPyramid.nPerOct	8	8	10	10
pPyramid.nOctUp	0	0	1	1
pPyramid.nApprox	7	7	0	0
pPyramid.smooth	0.5	0.5	0	0
pPyramid.pChns.shrink	4	4	6	4
pBoost.discrete	1	0	0	0
pBoost.pTree.maxDepth	2	4	4	4
pBoost.pTree.fracFtrs	0.0625	0.0625	1	1
pLoad.squarify	aspect ratio = 0.41	aspect ratio = 0.41	aspect ratio = 0.41	aspect ratio = 0.41
pLoad.lbls	person	person	person	person
pLoad.ilbls	people, person?, cyclist	people, person?, cyclist	people, person?, cyclist	people, person?, cyclist
pLoad.hRng	[55; Inf)	[50; Inf)	[50; Inf)	[50; Inf)
pLoad.vType	none occluded	none occluded	none occluded	none occluded

Table 5.1: Different parameter configurations of filtered channel features based detectors. The parameters of the three configurations, which differ from the original configuration of Hwang et al. [HPK+15], are highlighted in red.

the maximal depth of each decision tree. If not all features are used to train each node split of a decision tree, the fraction of features to be sampled from the complete feature pool is specified by *pBoost.pTree.fracFtrs*.

The remaining parameters control the criteria for loading the training data. *pLoad.lbls* defines the bounding box labels used for training or evaluation, whereas *pLoad.ilbls* lists all labels of bounding boxes that are handled as ignore regions. *pLoad.hRng* specifies the height range of the considered bounding boxes. In Table 5.1 all annotations equal to or larger than 50 or 55 pixels are used. The height range is extended from 55 to 50 pixels to be consistent to the Caltech dataset (see Section 4.3). Bounding boxes with labels smaller than 50 pixels are marked as ignore regions. *pLoad.vType* specifies the allowed occlusion levels. For training only annotated bounding boxes that contain non-occluded persons are used. In turn, for testing person bounding boxes with no or partial occlusions are used. The reason for only using sane bounding boxes is to avoid confusing the detector training with corrupt data. As proposed by Dollár et al. [DWSP12], the bounding boxes (GT and detections) are set to a fixed aspect ratio. The height of annotated persons is an accurate reflection of their scale while the width also depends on the person's pose. To avoid undesired effects on the evaluation results, caused by the variability of bounding box width, the standardization of the aspect ratio adapts the bounding box width till the desired aspect ratio $ar = \frac{w_{bb}}{h_{bb}} = 0.41$ is reached. Without standardizing, the different aspect ratios have an positive or negative influence on the IoU overlap. With a higher aspect ratio, the box is larger and therefore can have a higher IoU overlap that improves the detection results. In turn, a smaller aspect ratio decreases the detection performance.

After classifying each RoI of the sliding window approach, NMS is applied to reduce nearby detections. The NMS uses the IoU overlap criteria [EVW+10]. In Piotr's Matlab Toolbox [DoI] a modified version of IoU is used. Instead of dividing the intersection area of both bounding boxes by the union of both boxes, the intersection area is divided by the area of the smaller bounding box. For each pair of bounding boxes, the IoU is calculated, and if their overlap is greater than a defined threshold such as 0.65, the bounding box with the lower score is suppressed. Once a bounding box is suppressed it can no longer suppress other bounding boxes. The bounding boxes are processed in order of decreasing score.

There are some evaluations about different parameters. Walk et al. [WMSS10] and Zhang et al. [ZBO+16] analyze the usage of ignore regions. According to them, areas covering crowds, human shapes that are not persons such as posters and statues, and areas for which it cannot be decided certainly whether they contain a person or not, need to be considered as ignore regions.

There are several methods for data augmentation that can be evaluated for the channel features based detectors such as random bounding box rotation, Gaussian blur, noise or gamma adjustment [LTCFS16], changing the brightness [SL11] [KSH12] or the contrast [SL11] [HPK+15], random bounding box shifting [MG06]

[LTCFS16], or random scaling of bounding boxes [KSH12] [RDGF15] [LTCFS16]. A good overview and evaluation of current data augmentation approaches is given by Ohn-Bar and Trivedi [OBT16].

5.3 Usage as Proposal Generator

The filtered channel features based detectors are usually designed for object/person detection. However, they can be adapted for proposal generation of a certain object class such as persons. The two parameters *cascThr* and *cascCal* are the key parameters for tuning the expected number of FN occurences. The goal is to reduce the number of FNs and thus lower the miss rate. At the same time, this generates more FPs, which are assumed to be rejected during the subsequent classification step. Compared to the sliding window paradigm, the number of evaluated windows can be reduced dramatically when using the filtered channel features base detectors. This is the main goal of using them as a proposal generator. *cascThr* is the constant rejection threshold and *cascCal* the calibration value used for the constant soft cascades approach [BB05] [DAK12] [ZV08] [AFD13]. A recommendation in the Piotr's Matlab Toolbox [Dol] and also experimentally determined by Dollár et al. [DAK12] is to set *cascThr = -1* as a typical constant rejection threshold value and adjust *cascCal* until the desired miss rate is reached. *cascCal* controls the trade off between recall and speed by manipulating the rejection threshold of each weak classifier, called re-calibration. For proposal generation, Li et al. [LLS+16] propose *cascThr = -70* with leaving *cascCal = 0.005* unchanged (ACF detector). Another proposed value for *cascThr = -50* is given by Liu et al. [LZWM16] to adjust the ACF-T-THOG for proposal generation. The miss rate can be decreased by increasing the *cascCal* calibration value. This increases the number of FPs, but reduces the number of FNs. Similar effect can be obtained by decreasing the constant rejection threshold *cascThr*. It is recommended to leave *cascThr* unchanged and do finetuning by using the *cascCal* only. It is analyzed that finetuning using *cascCal* can be performed more accurately.

6 Person Detection based on Deep Learning

After introducing the filtered channel features based detectors for person detection, this chapter presents the deep learning approaches of this work. First, the fundamentals are recapitulated. The proposed approaches are based on VGG-16 network and Faster R-CNN. The three works [ZLLH16] [LZWM16] [WFHB16] are reviewed that serve as inspiration for CNN architecture design and training. In Section 6.4 and Section 6.5 the proposed CNN architectures of this thesis are introduced.

6.1 VGG-16 Network

The VGG-16 network is a CNN originally designed by Simonyan and Zisserman [SZ15] for the task of image recognition. *VGG* originates from the abbreviation of the research group of Simonyan and Zisserman called *Visual Geometry Group* of the University of Oxford. The number *16* represents the number of trainable layers: 13 convolutional (conv) layers and 3 fully connected (fc) layers. Together with VGG-19 [SZ15], Clarifai [ZF14], AlexNet [KSH12], GoogLeNet [SLJ+15], or ResNet [HZRS15a], VGG-16 is a widely used network architecture.

Figure 6.1 shows the VGG-16 architecture on the left. Convolutional layers with the same layer parameters and therefore providing the same feature map resolution are combined visually in a common conv block. conv1 and conv2 consist of two convolutional layers depicted by the small line on both sides of the boxes. For conv3, conv4, and conv5, each block has one additional convolutional layer with its corresponding Rectified Linear Unit (ReLU) used as activation function. The corresponding layers of each block are outlined in Table 6.2. The trainable layers are highlighted in italic font. Rather than using relatively large receptive fields (large kernel size and stride) [KSH12] [ZF14], the convolutional layers of VGG-16 have

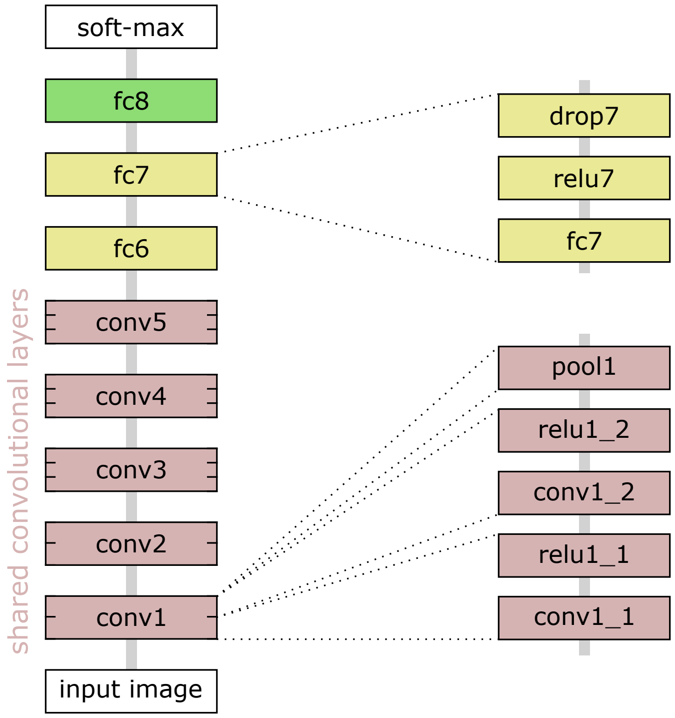


Figure 6.1: Structure of the VGG-16 network on the left side. Each red 🟥 block on the left contains multiple layers as depicted on the right (convolution, ReLU, and pooling layers), same for the yellow 🟨 blocks (fully connected, ReLU, and dropout layers). The green 🟩 layer is a single fully connected one.

a kernel size of 3×3 throughout the entire network. Those kernels are convolved with the input image at every pixel (with stride 1). Therefore, a stack of two 3×3 convolutional layers without spatial pooling in between, has an Effective Receptive Field (ERF) size of 5×5; three such layers have a 7×7 ERF size. The ERF size corresponds to the size of the region in the input image that encompasses all pixels, which contribute to the activation of one feature map pixel of a certain convolutional feature map. ERF stride and feature stride are synonymously used and projects the stride of one pixel of a specific feature map back to the input image. Simonyan and Zisserman pursue the goal of making the decision function more discriminative by using a stack of convolutional layers instead of one convolutional layer with a larger ERF size. A second advantage is the reduction of the number of trainable parameters [SZ15]. The stride and size of the ERF of each layer of the VGG-16 are listed in Table 6.1. Yu et al. [YYB+16] analyze the difference between AlexNet and VGG-16 network by considering the stride and size of the ERF.

Layer Name	conv1_1	conv1_2	pool1	conv2_1	conv2_2	pool2	conv3_1	conv3_2	conv3_3	pool3
Size of ERF	3 × 3	5 × 5	6 × 6	10 × 10	14 × 14	16 × 16	24 × 24	32 × 32	40 × 40	44 × 44
Stride of ERF	1	1	2	2	2	4	4	4	4	8

Layer Name	conv_4_1	conv4_2	conv4_3	pool4	conv5_1	conv5_2	conv5_3	conv-prop
Size of ERF	60 × 60	76 × 76	92 × 92	100 × 100	132 × 132	164 × 164	196 × 196	128x128
Stride of ERF	8	8	8	16	16	16	16	16

Table 6.1: Overview of the VGG-16 layers, and the size and stride of the Effective Receptive Field (ERF) their output feature maps provide [YYB+16]. The size and stride values are measured in pixels.

VGG-16 layers	**conv1**	**conv2**	**conv3**	**conv4**	**conv5**	**fc6, fc7**	**fc8**
layers *(trainable)*	*conv1_1* relu1_1 *conv1_2* relu1_2 pool1	*conv2_1* relu2_1 *conv2_2* relu2_2 pool2	*conv3_1* relu3_1 *conv3_2* relu3_2 *conv3_3* relu3_3 pool3	*conv4_1* relu4_1 *conv4_2* relu4_2 *conv4_3* relu4_3 pool4	*conv5_1* relu5_1 *conv5_2* relu5_2 *conv5_3* relu5_3 pool5	*fc6*, *fc7* relu7, relu8 drop7, drop8	*fc8*
type	convolution	convolution	convolution	convolution	convolution	fully connected	fully connected
kernel size	3 × 3	3 × 3	3 × 3	3 × 3	3 × 3	-	-
number of filters	64	128	256	512	512	-	-
stride	1	1	1	1	1	-	-
pad	1	1	1	1	1	-	-
number of output neurons	-	-	-	-	-	4,096	1,000

Table 6.2: Parameters of the trainable layers for the VGG-16 network.

In their work, Simonyan and Zisserman describe the training procedure using fixed size 224 × 224 input images, which are randomly cropped from re-scaled training images (data augmentation). Additionally, they do data augmentation by random horizontal flipping of the cropped regions. As common for CNNs, they subtract the mean image from the RGB input image. Since they use the Caffe toolbox [JSD+14], they provide a Caffe model pre-trained on the ImageNet dataset [RDS+15] for the task of image classification. This pre-trained model is used for initialization of the approaches in the remainder of this thesis, especially the *shared convolutional layers* as denoted in Figure 6.1.

VGG-16 blocks	conv1	conv2	conv3	conv4	conv5	fc6	fc7	fc8	total
number of weights	38,592	221,184	1,474,560	5,898,240	7,077,888	102,760,448	16,777,216	4,096,000	138,344,128
number of biases	128	256	768	1,536	1,536	4,096	4,096	1,000	13,416
number of trainable parameters	38,720	221,440	1,475,328	5,899,776	7,079,424	102,764,544	16,781,312	4,097,000	**138,357,544**

Table 6.3: Number of trainable parameters given for each block of the VGG-16 network in Figure 6.1. See Equation 6.1 and Equation 6.2 for the calculation approach.

To get an overview of how many weight and bias parameters are adapted during a training procedure for each layer, the numbers are listed for each convolutional and fully connected block of Figure 6.1 in Table 6.3. The *number of weights* denotes the trainable filter weights of the convolutional layers or the weights of the neurons of the fully connected layers. The *number of biases* means the trainable bias of each filter or output neuron, which serves for shifting the activation function (ReLU) left or right. Equation 6.1 shows how to calculate the number trainable parameters of one convolutional layer.

$$N_{conv} = f_k^2 \cdot i \cdot o + o \tag{6.1}$$

Equation 6.1 calculates the number of trainable weights N_{conv} for one convolutional layer such as conv1_1. The filter kernel size is denoted by f_k. Square filters for every convolutional layer are assumed. The variable i represents the number of channels that the input feature map provides. Each convolution convolves a three-dimensional input volume with a three-dimensional filter, to output one pixel of the resulting feature map. The convolutional result is a two-dimensional feature map. Repeating the convolution for all o filters of the convolutional layer, a three-dimensional output feature map with o channels is obtained. As for each filter a bias is adapted during training, o bias parameters are added to N_{conv}.

$$N_{fc} = i \cdot o + o \tag{6.2}$$

Calculating the number of trainable weights N_{fc} for one fully connected layer is presented in Equation 6.2. Since every input neuron is connected to every output neuron (fully connected) the number of weights is calculated by multiplying the number of input neurons i by the number of output neurons o. As before, the additional term contains the bias parameter for each output neuron.

6.2 Faster R-CNN

This section reviews the Faster R-CNN architecture of Ren et al. [RHGS16]. In comparison to the predecessors R-CNN (Regions with CNN features) [GDDM14] and Fast R-CNN [Gir16], Ren et al. introduce the Region Proposal Network (RPN) for proposal generation instead of using Selective Search [UVGS13]. By sharing the convolutional layers between RPN and the classification network they accelerate processing. The regions proposed by the RPN are processed by the Fast R-CNN detector [Gir16]. Faster R-CNN is an approach originally proposed for object detection in general, but can be adapted to many other domains.

As depicted in Figure 6.2, the input image is processed by the shared convolutional layers having the same parameterization as the VGG-16 network in Figure 6.1. The output of the shared convolutional layers are the *conv feature maps*, also called *output feature maps*. Based on the conv feature maps, the RPN is applied to generate candidate windows (RoIs). For each proposed region, RoI pooling [Gir16] [HZRS15b] is applied to the conv feature maps in order to get a fixed length feature vector, which serves as input for the classification network.

The classification network is the same as used in Fast R-CNN [Gir16]. The fixed length feature vector of each RoI, resulting from the RoI pooling layer, is fed into two fully connected layers (each with 4,096 output neurons), denoted with FCs in Figure 1 of [Gir16]. The resulting 4,096-dimensional feature vector is fed into two sibling fully connected output layers. One output layer produces softmax probability estimates over the object classes and background (classification). The other layer outputs four real-valued numbers for each of the object classes, encoding the refined bounding box positions for each class (regression). Therefore, simple bounding box regression is used to improve localization [GDDM14] [FGMR10]. Details about the training of the two sibling output layers with multi-task loss are provided by [Gir16].

After reviewing the coarse structure of Faster R-CNN as well as the classification part, now the RPN is presented, which is the main contribution compared to Fast R-CNN. The RPN generates region proposals, confidence scores, and feature

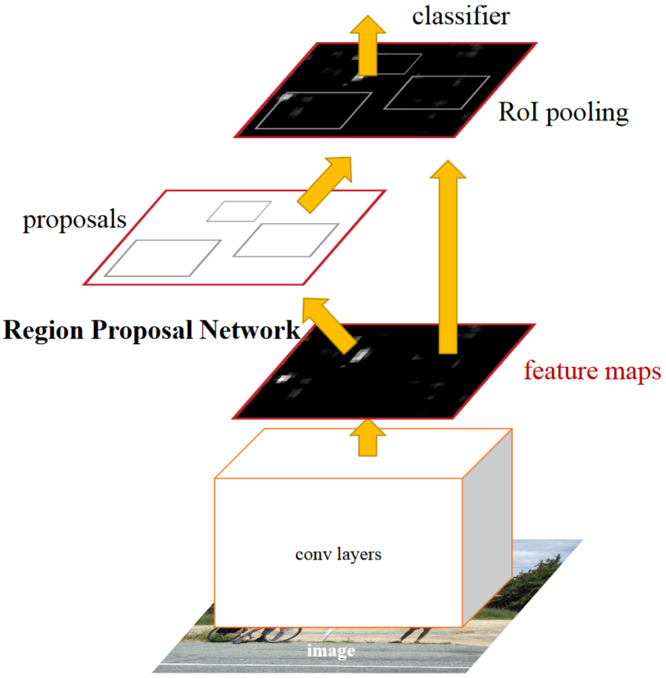


Figure 6.2: Structure of the Faster R-CNN framework [RHGS16] as a unified network. Starting from the input image, using the shared convolutional layers, the conv feature maps are computed, which represent the input for the RPN. After applying RoI pooling on the conv feature maps using the proposed regions of the RPN, the fixed length feature vectors are fed into the classification network.

maps. Considering Figure 6.2 and Figure 6.3, both start with the input image that is propagated through the shared convolutional layers, which are equal to those of the VGG-16 network except for the last pooling layer (pool5) being omitted. As the same parameters for the shared convolutional layers are used as for the VGG-16 network, these layers can be initialized by using a VGG-16 model pre-trained for the task of image classification on the ImageNet dataset. The advantage of using a model pre-trained on a large dataset (also called auxiliary dataset) is that useful weights are available for the low-level filters. After the shared convolutional layers, the output feature maps are fed into an intermediate 3×3 convolutional layer (*conv-prop*) followed by two sibling 1×1 convolutional layers. One of the sibling layers is used for classification (*cls-score*), providing the objectness scores for each RoI. Objectness measures the probability of the RoI to contain an object. The second convolutional layer (*bbox-pred*) performs bounding box regression. Since the RPN only consists of convolutional layers it is an FCN [LSD15] and can be trained

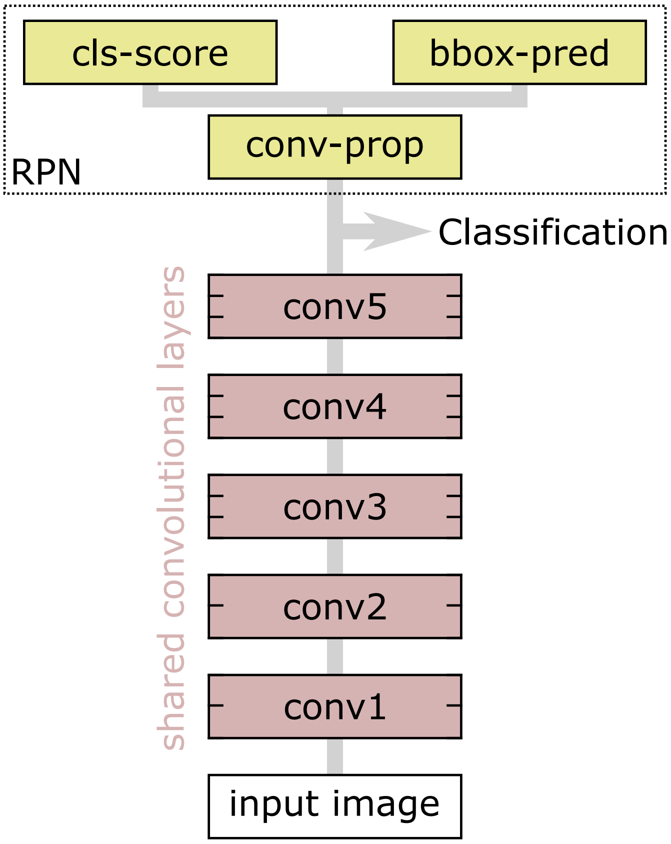


Figure 6.3: Structure of the RPN showing the shared convolutional layers, which are identical to those of VGG-16. The relation to the Faster R-CNN approach in Figure 6.2 is given by the arrow pointing to the right (Classification). Afterwards there are three convolutional layers forming the RPN network.

end-to-end. As the RPN is an FCN it can handle input images of any size. The cls-score 1 × 1 convolutional layer has $2k$ output filters to generate two predictions (person/no person) for each of the k anchor boxes (introduced later). Same with the bbox-pred 1 × 1 convolutional layer: it has $4k$ output filters to account for the four coordinates of each anchor box.

In Table 6.4 the number of weights that are trainable for the RPN architecture are listed (compared to those of VGG-16 in Table 6.3). The shared convolutional part is equal to the one of the VGG-16 network and therefore has the same number of trainable weights. As the RPN is an FCN and only convolutional layers are utilized, the total number of weights is dramatically reduced from 138,357,544 (VGG-16) to

RPN blocks	conv1	conv2	conv3	conv4	conv5	conv-prop	cls-score	bbox-pred	total
number of weights	38,592	221,184	1,474,560	5,898,240	7,077,888	2,359,296	9,216	18,432	17,097,408
number of biases	128	256	768	1,536	1,536	512	18	36	4,790
number of trainable parameters	38,720	221,440	1,475,328	5,899,776	7,079,424	2,359,808	9,234	18,468	**17,102,198**

Table 6.4: Number of trainable parameters given for each block of the RPN in Figure 6.3. See Equation 6.1 and Equation 6.2 for the calculation approach.

17,102,198 (RPN). When considering the number of weights for each block, one recognizes a large number of trainable weights at the transition from the convolutional part to the fully connected of VGG-16 (conv5 to fc6), making up 74 % of the total number of trainable weights. This transition does not exist in the RPN architecture.

Using Figure 6.4, the usage of *anchors* for Faster R-CNN is explained, and therefore how to determine RoIs, based on the output feature maps. To generate region proposals, a small network is shifted over the conv feature maps in sliding window fashion. This network takes as input a 3 × 3 spatial window of the conv feature maps (red box). Each sliding window is mapped to a lower-dimensional feature. The *intermediate* layer corresponds to the convolutional layer called conv-prop (512 filters) in Figure 6.3 and is used to map each sliding window to a 512-dimensional (512-d) feature vector. This feature vector is fed into the two sibling layers, cls-score and bbox-pred. The anchors are used to simultaneously predict multiple region proposals at each sliding window location. The maximum number of possible proposals for each location is denoted as k and represents the number of anchor boxes for each location. Thus, for each location on the conv feature map and its generated 512-d feature vector, the bbox-pred layer (denoted as *reg layer* in Figure 6.4) encodes $4k$ real-valued outputs for the bounding box coordinates, and the cls-score layer (denoted as *cls layer* in Figure 6.4) outputs $2k$ scores that estimate the probability of object or background. The anchors are used during RPN training. The bounding box prediction can be thought of as bounding box regression from an anchor box to a nearby GT box. To account for varying sizes, a set of k bounding box regressors is learned. Each regressor is responsible for one scale and one aspect ratio, and the k regressors do not share weights. In their approach, Ren et al. propose three different scales and three aspect ratios yielding to $k = 9$ anchors at each sliding window position. When applying the method for each location of the conv feature map with size $W \times H$, in total there are $W \cdot H \cdot k$ anchors. The three anchor aspect ratios are 1:1, 1:2 and 2:1. Three scales with box areas 128^2, 256^2, and 512^2 pixels are used. This means that an anchor is the reference to a certain location of the conv feature map and for each reference/location there are k different anchor boxes specified.

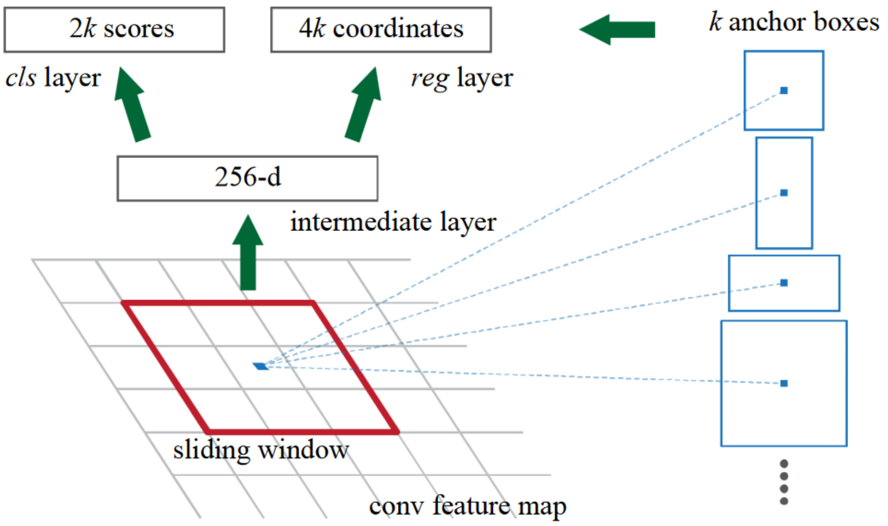


Figure 6.4: Visualization of the anchor scheme used for solving the regression problem of RPN [RHGS16]. The anchors enable multi-scale detection.

For RPN training, a binary class label (of being an object or not) is assigned to each anchor. A positive label is assigned to two kinds of anchors: (i) the anchor/anchors with the highest IoU overlap with a GT box, or (ii) an anchor that has an IoU overlap higher than 0.8 with any GT box [RHGS16]. Thus, a single GT box can assign positive labels to multiple anchors. For the regression part, the offset from each anchor box to the related GT box is prepared as label for learning the regression [Gir16]. The RPN is trained end-to-end by backpropagation and Stochastic Gradient Descent (SGD) [LBD+89]. Ren et al. use mini-batch training, i.e. a mini-batch arises from a single image containing 120 positive and negative example anchors. As already mentioned, the shared convolutional layers are initialized with the pre-trained VGG-16 ImageNet model. Details about the training parameters are provided in [RHGS16].

Since utilized for the Faster R-CNN approach and necessary for subsequent approaches, RoI pooling is reviewed. RoI pooling is introduced by Girshick [Gir16] and inspired by [HZRS15b]. The motivation for RoI pooling in Fast R-CNN is that there are RoIs of arbitrary size proposed by Selective Search. As the fully connected layers for classification and regression expect a feature vector of fixed length, a technique for transforming the arbitrarily sized RoIs to a fixed length is necessary. Thus, Girshick uses max pooling to convert the features inside any valid region of interest into a small feature map with a fixed spatial extent. RoI pooling works by dividing the arbitrarily sized $w_{RoI} \times h_{RoI}$ RoI window into a grid of sub-windows with a fixed number of grid cells $W_{RoI} \times H_{RoI}$, that is projected onto the RoI. For each sub-window, the contained RoI values are max pooled into the corresponding cell of the output grid (grid size is $W_{RoI} \times H_{RoI}$). Pooling is applied spatially and independently to each feature map channel just like in standard max pooling. The RoI pooling layer is a special case of the *spatial pyramid pooling layer* used in SPP nets [HZRS15b], in which there is only one pyramid level. Ren et al. propose $W_{RoI} = H_{RoI} = 7$ as default hyper-parameters for the RoI pooling layer, leading to a 7×7 output grid per feature map, and if vectorized to a feature vector of length 49. Since the RoI coordinates are given in the image domain, but RoI pooling is applied on the conv feature maps that have been sub-sampled several times (pooling or convolution), a mapping strategy for mapping the window in the image domain to the conv feature map domain is necessary. As described by He et al. [HZRS15b], the corner point of a window is projected onto a pixel in the conv feature maps, such that this corner point in the image domain is closest to the center of the receptive field of that conv feature map pixel. The mapping is complicated by the padding of all convolutional and pooling layers [HZRS15b]. In the Caffe framework the RoI pooling layer provides three parameters for adjustment. Two parameters define the pooled output grid size $W_{RoI} \times H_{RoI}$. The third parameter specifies the multiplicative spatial scale factor to translate RoI coordinates from their input scale to the scale used when pooling is performed. This factor is often referred to as the feature stride at a given layer in the network.

6.3 Inspirational Work

The following subsections review three approaches that play a key role for the detection approaches of this thesis. Zhang et al. [ZLLH16] apply the Faster R-CNN framework to person detection. They reveal that the RPN without Fast R-CNN as detection network (stand-alone) outperforms Faster R-CNN, and use a BDF for classification instead. For multi-spectral person detection, Liu et al. [LZWM16] present four different fusion approaches based on Faster R-CNN. Wagner et al. [WFHB16] propose two fusion architectures based on CaffeNet and an effective method for training the subnets to improve performance.

6.3.1 Region Proposal Network (RPN)

Zhang et al. [ZLLH16] perform person detection using the Faster R-CNN framework. They use the *Caltech10x-train* sub-dataset for training and the *Caltech-test-Reasonable* testing subset for evaluation. In Table 6.5, there is an overview of the results. Compared to the original Faster R-CNN, Zhang et al. adapt the anchors to use a single aspect ratio of 0.41 (width to height). This is the average aspect ratio of persons as inspired in [DWSP12]. While decreasing the number of aspect ratios, they increase the number of different anchor scales to 9. The 9 anchor boxes start from 40 pixels height with a scaling stride of 1.3, resulting in bounding box heights of 40, 52, 68, 88, 114, 149, 193, 251, and 326 pixels. As applied for the Faster R-CNN, the shared convolutional layers are initialized by using a model pre-trained on ImageNet. For training the RPN, an anchor is considered as a positive example, if it has an IoU ratio greater than 0.5 with a GT box, and as negative example otherwise. Other than Ren et al. (1:1), the ratio of positive and negative samples is 1:5 in a mini-batch. Each mini-batch consists of one image and 120 randomly sampled anchors for computing the loss. The NMS applied on the RPN results is also adapted with an IoU threshold of 0.5 to filter the proposal regions. The proposals are ranked by their scores. The Faster R-CNN implementation of Zhang et al. is used as code base for this thesis.

Zhang et al. consider the RPN as stand-alone detector and outperform Faster R-CNN (RPN + Fast R-CNN in Table 6.5). The good performance of the stand-alone RPN they explain by stating that RoI pooling suffers from small regions, but RPN is essentially based on fixed sliding windows (in a fully convolutional fashion) and thus avoids *collapsing bins*: if a RoI's input resolution is smaller than the output resolution after RoI pooling (i.e. 7×7), the pooling bins collapse and the features become 'flat' and not discriminative. The RPN predicts small objects by using small anchors. After establishing a baseline, Zhang et al. introduce a BDF for classification as depicted in Figure 6.5. The implementation is based on Piotr's Matlab Toolbox [DoI]. They bootstrap the training seven times, ending up with a BDF of 2,048 trees. Hard negative mining is applied by randomly sampling negative examples in the first training stage, mining additional 5,000 hard negative examples and adding them to the training set. To guarantee comparability the classifiers (R-CNN, Fast R-CNN and BDF) are all trained based on the same proposals given by the stand-alone RPN.

For analyzing the difference in feature map resolution used as input for the classifiers, Zhang et al. train and evaluate the classifiers using different conv feature

Methods	**RoI Features**	**MR (%)**
RPN stand-alone	-	14.9
RPN + R-CNN	raw pixels	13.1
RPN + Fast R-CNN	conv5_3	20.2
RPN + Fast R-CNN	conv5_3, à trous	16.2
RPN + BDF	conv3_3	12.4
RPN + BDF	conv4_3	12.6
RPN + BDF	conv5_3	18.2
RPN + BDF	conv3_3, conv4_3	11.5
RPN + BDF	conv3_3, conv4_3, conv5_3	11.9
RPN + BDF	conv3_3, (conv4_3, à trous)	**9.6**

Table 6.5: Excerpt of the results of Zhang et al. [ZLLH16], for using the RPN as stand-alone and with different classifiers (R-CNN, Fast R-CNN and BDF). The column *RoI Features* denotes the source of the features used for classification. Evaluation is performed on the *Caltech-test-Reasonable* sub-dataset.

maps. The results are listed in Table 6.5, and the different conv feature maps used for generating features are provided in the second column. The RPN + R-CNN detector uses the RPN instead of the Selective Search algorithm for proposal generation and passes the warped input RoI through the complete R-CNN pipeline: shared convolutional layers for feature extraction, SVM for classification and regression for improving the localization. This approach avoids RoI pooling and CNNs for classification and improves the performance of the stand-alone RPN. This result suggests that if reliable features can be extracted, the classifier is able to improve accuracy [ZLLH16]. Training the Fast R-CNN classifier on the same set of RPN proposals actually degrades the results. Zhang et al. assume that the reason for the degrading performance is partially because of the low-resolution feature maps in conv5_3. To prove their assumption, they train a Fast R-CNN on the same set of RPN proposals with the à trous trick applied to conv5 convolutional layers, reducing the feature stride from 16 pixels to 8. The à trous trick is explained together with Figure 6.6 later. When shifting the sliding window on a feature map exactly one pixel (stride

1) and project this stride onto the input image, this is called feature stride. This experiment proves that higher resolution feature maps are useful, but the detection rate compared to the stand-alone RPN is still lower.

For using the BDF with the convolutional features, RoI pooling is applied to generate a fixed length feature vector. The BDF is applied on different convolutional feature maps separately, yielding good results for conv3_3 (feature stride of 4 pixels) and conv4_3 (feature stride of 8 pixels). This shows that the featues of intermediate convolutional layers provide better information than those of the last convolutional layer. The concatenation of feature maps of multiple layers additionally boosts the performance of the RPN. Finally, Zhang et al. end up by combining the feature maps of conv3_3 with the à trous trick version of conv4_3. They state that the features from different layers can be simply concatenated without normalization due to the flexibility of the BDF classifier. In contrast, feature normalization needs to be carefully addressed [LRB15] for CNN classifiers when concatenating features.

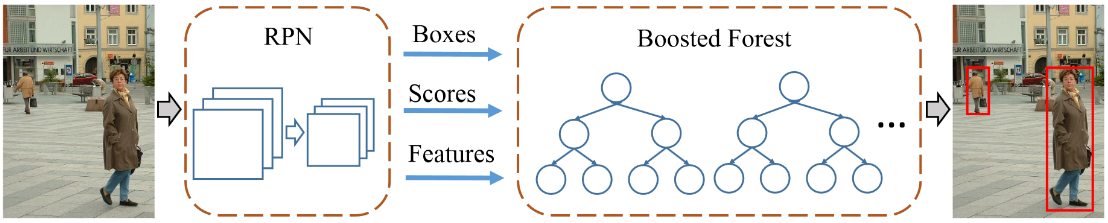


Figure 6.5: Pipeline for the approach of using an RPN combined with a BDF [ZLLH16]. The RPN provides bounding boxes, objectness scores, and the conv feature maps as input for the BDF.

The à trous trick, as applied by Zhang et al. for achieving feature maps of higher resolution, is a popular method used for FCNs applied for semantic segmentation [LSD15] [YK16] [CPK$^+$14]. Another name for the à trous trick is filter dilation. For motivating the à trous trick the initial situation is illustrated on the left of Figure 6.6. The feature maps after the pooling layer pool3 are considered as the à trous trick is applied for conv4 convolutional layers. The goal is to provide conv4 feature maps of higher resolution. The red box in Figure 6.6 outlines the filter kernel (3×3) of the subsequent convolutional layer conv4_1. The colored numbers represent the nine trained filter weights of conv4_1. To increase the feature map resolution, the stride of pool3 is reduced from 2 to 1, resulting in a four times bigger feature map and a smaller feature stride of 4 instead of 8. But considering the proportion of the filter kernel to the larger feature map, the receptive field is reduced in this step. For adjusting the receptive field of conv4_1, the *filter stride* is increased from 1 to 2, resulting in zero-padding of the filter kernel. The à trous trick enlarges the filter and thus preserves the initial ratio between filter size and feature map size. A larger feature map is provided with a smaller feature stride by simultaneously preserving the receptive field of the convolutional layers and without re-training any filter weights. Thus, applying the à trous trick to conv4 means changing the pooling stride of pool3 from 2 to 1 and adapting the filter strides of all convolutional layers (conv4_1, conv4_2, and conv4_3) from 1 to 2.

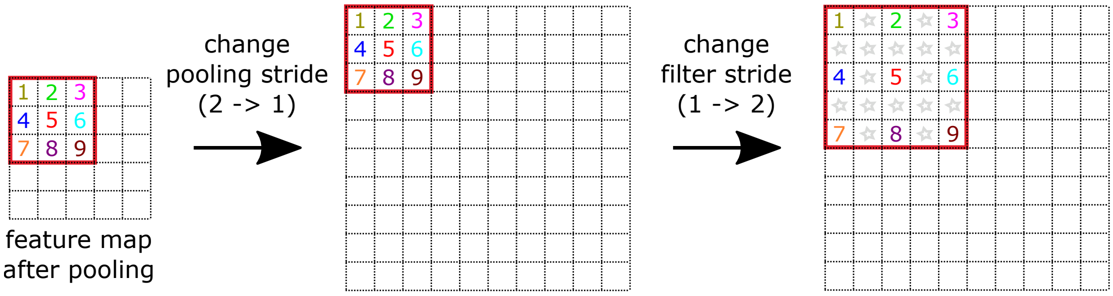


Figure 6.6: Illustration of the à trous trick that is used for generating denser features maps. The red box depicts the filter kernel of the subsequent convolutional layer.

6.3.2 Multi-spectral Fusion Architecture using Faster R-CNN

Based on the Faster R-CNN implementation of Zhang et al. [ZLLH16], which is implemented for working with the Caltech dataset, an approach for using Faster R-CNN for multi-spectral input images is required. Therefore the work of Liu et al. [LZWM16] is reviewed. They evaluate the four fusion approaches based on the Faster R-CNN framework in Figure 6.7.

As motivated in Chapter 1, Liu et al. separately train the Faster R-CNN-VIS and Faster R-CNN-IR on VIS and IR images, respectively, to show the complementary information of both image types for person detection. Different to the original Faster R-CNN approach, they remove the fourth max pooling layer (pool4). This is motivated by the observation that larger feature maps are beneficial for detecting persons of small image sizes. Faster R-CNN uses multi-scale (3 scales) and multi-ratio (3 ratios) reference anchors to predict locations of region proposals. Liu et al. discard the anchor aspect ratio of 2:1, due to the typical person aspect ratio of 0.41 [DWSP12]. The shared convolutional layers are initialized as for the original approach by using a VGG-16 model pre-trained on the ImageNet dataset. The

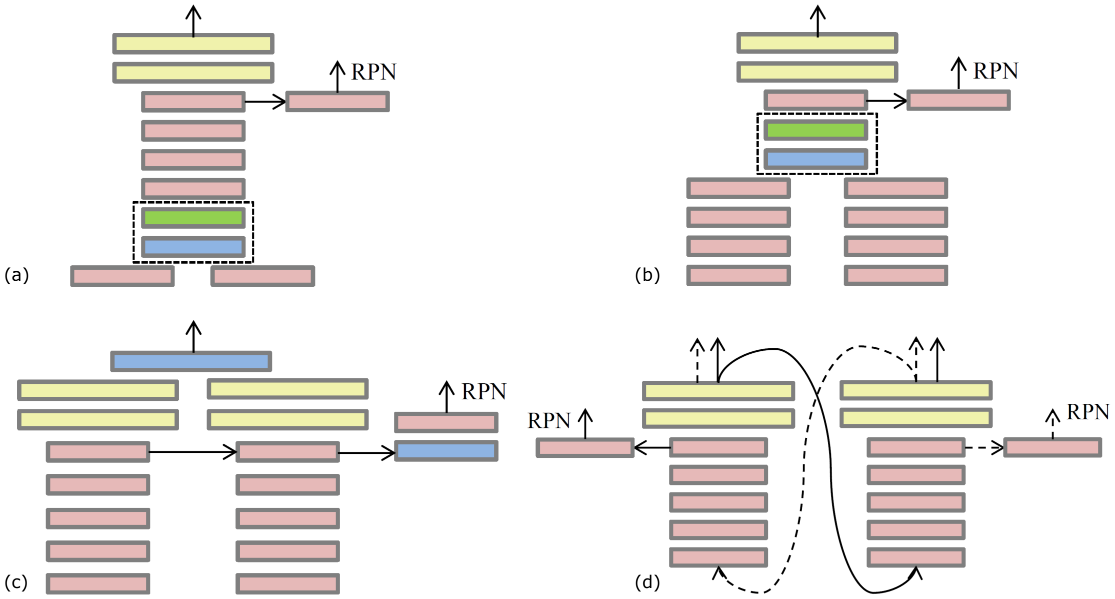


Figure 6.7: Different Faster R-CNN fusion approaches [LZWM16]. (a) Early Fusion, (b) Halfway Fusion, (c) Late Fusion, and (d) Score Fusion.

other layers are initialized randomly using a Gaussian distribution with variance of 0.01. For finetuning they use the *KAIST10x-train* sub-dataset.

The Early Fusion Faster R-CNN in Figure 6.7 (a) concatenates the feature maps from VIS and IR branches immediately after the first convolutional layers. That is this kind of fusion is later called Conv1 Fusion. The red rectangles represent the different convolutional blocks such as conv1, conv2, and the fully connected layers of the Faster R-CNN are depicted with yellow blocks. The intermediate layer of the RPN is the red box branched to the right. The blue rectangle is a simple Caffe concatenation layer, which concatenates the two feature maps of conv1-VIS and conv1-IR in channel dimension. As consequence, this concatenation doubles the number of feature map channels. As they use the VGG-16 pre-trained weights to initialize the convolutional layers after the fusion block, a dimension reduction is necessary. Therefore, Liu et al. introduce the Network-in-Network (NiN) layer after feature concatenation, which is actually an 1×1 convolutional layer reducing the number of channels from 128 to 64. On one side this additional layer is used to learn fusing the feature maps based on different input images, but also the VGG-16 model can be used for initializing the layers after the fusion module. The NiN layer is followed by a ReLU layer, enhancing the discriminability of local patches. The Early Fusion fuses low-level visual features, such as corners and line segments.

Fusion Architecture	**MR (%)**
Early Fusion	40.44
Halfway Fusion	**36.99**
Late Fusion	40.46
Score Fusion	40.06

Table 6.6: Excerpt of the results of Liu et al. [LZWM16] for different fusion approaches evaluated on the *KAIST-test-Reasonable* sub-dataset.

The Halfway Fusion Faster R-CNN in Figure 6.7 (b), puts the fusion module after the fourth convolutional layers (conv4). The fusion module consists of the concatenation and the NiN part as introduced for the Early Fusion. Features of conv4 are of higher semantic abstraction than those of conv1. Hence, Halfway Fusion fuses semantic features. The Late Fusion (c) concatenates the last fully connected layers for the classification part and the feature maps after conv5 layers for the RPN. Differently to the fusion module of (a) and (b), the Late Fusion only concatenates the feature maps without using NiN for learning the feature map fusion. Late Fusion performs high-level feature fusion. The fourth fusion approach is the Score Fusion Figure 6.7 (d). As explained by Liu et al., they first get detections from the Faster R-CNN-VIS, which are sent into the Faster R-CNN-IR to obtain detection scores based on the IR image, and vice versa. In practice, this can be accomplished by using RoI pooling. The conv feature map is computed once and used to generate proposals with the VIS RPN and the IR RPN. The RoIs of both RPNs are processed with both classification networks (Fast R-CNN). Detection scores from the two classification networks are combined using equal weights such as 0.5.

As listed in Table 6.6 the different fusion approaches of Liu et al. lead to the results that Halfway Fusion outperforms the other approaches. Whereby the other approaches, namely Early, Late and Score Fusion, have similar but worse results. Their fusion approach is intuitively and therefore the fusion module is adopted for the proposed fusion approaches.

6.3.3 Pre-Training, Pre-Finetuning, and Finetuning

The third reviewed work, is from Wagner et al. [WFHB16], who point out a training strategy consisting of pre-training, pre-finetuning, and finetuning. The CNN fusion architectures are not the same as from Liu et al. in the previous section. This work shows improvements by using their proposed training strategy. Wagner et al. build up on the R-CNN detection framework, i.e. a proposal generator combined with a CNN for classification. For generating the proposal regions they use the ACF+T+THOG detector proposed by Hwang et al. [HPK+15]. The two proposed architectures are based on the CaffeNet [JSD+14] architecture. The Early Fusion network concatenates the VIS image (3 channel) and IR image (1 channel) before feeding them into the network. The Late Fusion architecture trains two different sub-networks separately (CaffeNet-VIS and CaffeNet-IR). These sub-networks are combined by training two additional fully connected layers. Wagner et al. do some minor parameter adjustments for CaffeNet-VIS and CaffeNet-IR.

The training strategy of Wagner et al. comprises three stages. As the target dataset KAIST is a rather small dataset, the network cannot be trained from scratch [ZF14] [TLWT16] [HOBS15]. When training from scratch, the trainable parameters are initialized by random values [SZ15]. The variance of the initialized values should be high enough, otherwise no training will take place [HSK+12] [Ben12]. There are different available initialization methods. One popular method is the zero-mean Gaussian initialization and the other is called Xavier initialization [GB10] also used for the VGG-16 training [SZ15]. Another possibility for training is to use pre-trained models as applied in many works [TLWT16] [GDDM14] [AGM14] [HOBS15] [TMB+16]. As a consequence, the weights are usually not randomly initialized. Instead, the weights of a pre-trained model are utilized as initialization and the weights are adapted by training on the new dataset. This kind of training is often referred to as finetuning or transfer learning. The dataset used for generating the pre-trained model is called the auxiliary dataset. This auxiliary dataset commonly is a huge dataset able to train well-formed (useful) weights.

The third method for training network architectures is called pre-finetuning. Pre-finetuning uses a pre-trained model for initializing the weights. As the target dataset KAIST is rather small the goal is to adapt these general weights towards the task of person detection. The additional adaptation of the weights is performed without the target dataset to maintain the generalization capability of the trained model. Direct training with the target dataset could lead to overfitting [And13] [HSK+12]. The trick is to utilize a second auxiliary dataset, which can be used to train the net-

Fusion Architecture	**Pre-Finetuning**	**MR (%)**
Early Fusion	✗	57.96
	✓	53.94
Late Fusion	✗	51.30
	✓	**43.80**

Table 6.7: Excerpt of the results of Wagner et al. [WFHB16] for two different training strategies and two different architectures evaluated on the *KAIST-test-Reasonable* sub-dataset.

work for the desired task. After this pre-finetuning step, the network is finetuned on the target dataset. Wagner et al. use pre-finetuning to train their architecture. First, they pre-train their networks on the large auxiliary dataset ImageNet. Then, Wagner et al. apply pre-finetuning by using the Caltech dataset. Since the Caltech dataset has no IR images, Wagner et al. use the red channel as approximation of the IR image. After finetuning the three networks (Early Fusion network, CaffeNet-VIS and CaffeNet-IR) separately, they train the networks separately again on the KAIST dataset (*KAIST10x-train*). For training the Late Fusion, the two sub-networks (CaffeNet-VIS and CaffeNet-IR) are combined by two fully connected layers and trained a second time on the KAIST dataset while keeping the weights of the sub-networks fixed.

The results in Table 6.7 compare the two approaches of training from scratch and the use of pre-finetuning combined with pre-training and finetuning. Wagner et al. performed the evaluations with training from scratch (✗) with the KAIST dataset only. In a second experiment they apply pre-training with ImageNet dataset, pre-finetuning with Caltech dataset, and finetuning with the KAIST dataset (✓). The results evidently show for both network architectures that the pre-finetuning and fine-tuning boosts the detection performance. In this thesis the use of auxiliary datasets is analyzed in detail, as well as using an IR dataset for pre-finetuning the IR sub-network.

6.4 Proposed Multi-spectral Fusion Architectures

By reviewing the works of Ren et al. [RHGS16], Zhang et al. [ZLLH16], Liu et al. [LZWM16] and Wagner et al. [WFHB16] the inspiration of this work has been presented. The Faster R-CNN framework of Zhang et al. builds the fundament for analyzing different fusion and training strategies. As a sanity check the results of Zhang et al. are verified on the *Caltech-test-Reasonable* sub-dataset by training and evaluation. Nearly the same results are achieved, and the minor deviations result from the training procedure comprising random choices to generate each mini-batch.

In Figure 6.8 the different fusion approaches inspired by Liu et al. are illustrated. Instead of fusing the Faster R-CNN, RPN fusion is proposed. The *fusion module* comprises the concat layer for concatenation of the VIS and IR feature maps, and the NiN layer for dimension reduction. Figure 6.8 (a) is called Conv1 Fusion and fuses after the conv1 RPN layers, (b) is fused after conv2 RPN layers and analogously the other layers are fused. Regarding the Conv5 Fusion, once the conv feature maps of conv5 are fused by using the complete fusion module (e), but also by only concatenating the conv feature maps without the NiN layer (f).

For training, the RPNs are trained on VIS and IR data separately, resulting in two individual RPNs without fusion. The shared convolutional layers are always initialized by using the VGG-16 model pre-trained on the ImageNet dataset. Inspired by Wagner et al., the auxiliary datasets Caltech, CaltechR and CVC-09 are used for pre-finetuning and evaluating the trained models. CaltechR is used for imitating training with an IR dataset by using only the red channel of the RGB images. The red channel is utilized three times and stacked in channel dimension to obtain 3-channel images. This is necessary, because the VGG-16 pre-trained model is trained on 3-channel images and the conv1_1 layer expects three channels as input. Otherwise, the VGG-16 model has to be pre-trained for 1-channel input images from scratch. Wagner et al. apply the same procedure. In the similar way, an RPN is trained on the CVC-09 IR dataset. The RPNs initialized by the pre-trained VGG-16 model are trained (pre-finetuning) on the training subsets (Chapter 4) of each dataset (Caltech, CaltechR and CVC-09). Then, the performance on the testing subsets of each dataset is evaluated to analyze the influence of the dataset size (small or big training subset) as well as horizontal flipping of the training samples as data augmentation. Based on the pre-finetuned models, finetuning with the KAIST VIS and IR images is applied separately on the related models. The pre-finetuned CaltechR and CVC-09 models are trained with the IR images of KAIST.

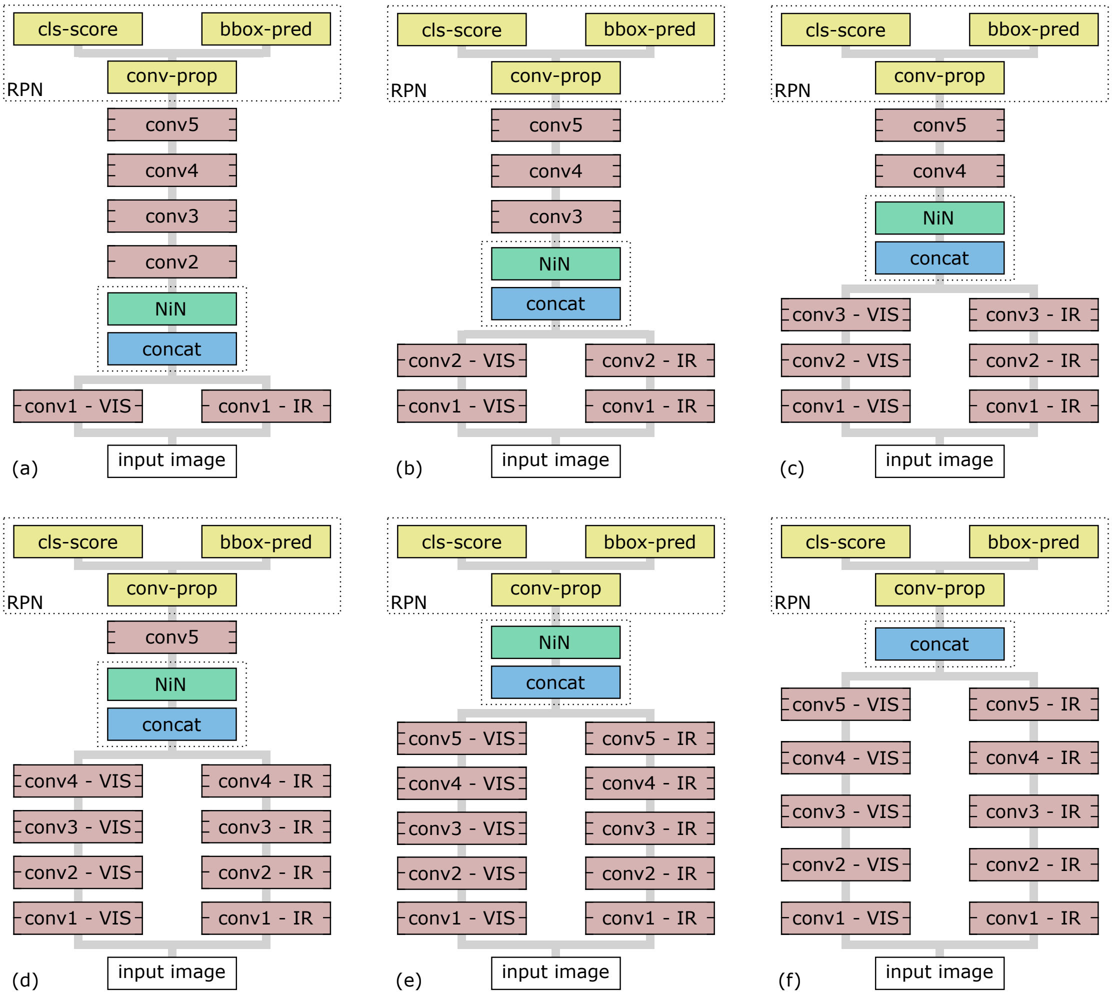


Figure 6.8: The proposed fusion approaches for RPN. (a) Conv1 Fusion RPN, (b) Conv2 Fusion RPN, (c) Conv3 Fusion RPN, (d) Conv4 Fusion RPN, (e) Conv5 Fusion RPN, and (f) Conv5 Fusion RPN without the NiN layer.

The pre-finetuned Caltech models are trained with VIS images of KAIST. Additionally, the models are trained directly based on the VGG-16 initialization without the pre-finetuning step. During training, the conv1 and conv2 layers are fixed due to overfitting concerns and therefore not affected by training. Thus, only the higher layers are affected by training. This is motivated by the observation that the earlier features of a CNN contain more generic features such as edge detectors or color blob detectors that should be useful for many tasks, but higher layers of the CNN become more specific to the details of the training dataset [TS09]. The results of this training stage are RPNs that can be applied on VIS or IR images of the KAIST dataset separately (VIS RPNs and IR RPNs).

The second training stage fuses the separately trained VIS and IR RPN models by using the architectures illustrated in Figure 6.8. The layers denoted by conv$X$-VIS and conv$X$-IR are initialized with the layers of the pre-finetuned VIS and IR RPN models, respectively. For the NiN layer the weights are randomly initialized according to a Gaussian distribution with standard deviation of 0.001 and the biases with zeros. The same initialization is applied to the RPN layers conv-prop, cls-score, and bbox-pred. The shared convolutional layers after the fusion module are initialized with the pre-trained VGG-16 model parameters.

For training, Zhang et al. use SGD for about 6 epochs. The Learning Rate (LR) is set to 0.001 and reduced to 0.0001 after 4 epochs. A value of 0.9 is used for the momentum term that adds a fraction of the previous weight update to the current one. For the weight decay factor, which causes the weights to exponentially decay to zero, a value of 0.0005 is used. The input images are re-sized such that its larger edge has 768 pixels for the KAIST dataset and 720 pixels for the Caltech and the CVC-09 dataset. The orginial image size of the KAIST dataset is 640×512, and the Caltech and CVC-09 dataset have an image size of 640×480 pixels. Thus, the images fed into the RPN are re-sized by factor 1.5 to 960×768 for KAIST and 960×720 for Caltech and CVC-09. The idea of re-sizing is to have feature maps of finer resolution (larger size). The size for re-sizing the input images is chosen w.r.t. the feature stride of 16. Instead of subtracting the mean image as for the VGG-16 training, the RGB mean values 123.68, 116.78, and 103.94 are used. For training, only bounding boxes containing non-occluded persons, and persons of height equal to or larger than 50 pixels are considered for generating positive samples. All other bounding boxes and the ones with labels different to 'person' such as 'people', 'person?', and 'bicyclist' are treated as ignore regions. The bounding box standardization proposed by Dollár et al. [DWSP12] is applied, i.e. for each bounding box the width is adapted until its aspect ratio (width to height) is equal to 0.41. Furthermore, this aspect ratio is utilized to determine the anchors. Equally to Zhang et al., one anchor aspect ratio of 0.41 is used and 9 different scales, starting from 40 bounding box height with scaling stride 1.3. Each mini-batch consists of one image with 120 randomly sampled anchors. The ratio of positive and negative samples is 1:6 in a mini-batch. Defined by the chosen RPN architecture with feature stride 16 and the scaling of the input image to a size of 960×768 pixels, the conv feature map of conv-prop layer has the size 60×48 pixels and the number of anchors is 9. Since every pixel of the conv feature maps represents an anchor with 9 anchor boxes, there are a total of 25,960 anchor boxes for one input image of the KAIST dataset. The anchor boxes have to be labeled whether they are foreground (person=1) or background (no person=0) boxes. An anchor box is considered as foreground if the IoU overlap is equal to or larger than 0.5, otherwise the anchor box is labeled as background. For evaluation, NMS is applied by using an IoU overlap of 0.5 for suppressing regions ranked w.r.t. their score [ZLLH16]. After the NMS, the 40 top-ranked detections make up the result of the fusion RPN. These training parameters are applied to the non-fusion RPNs, too.

6.5 Fusion RPN and Boosted Decision Forest (BDF)

In the previous section the RPN fusion approaches for generating region proposals for person detection based on multi-spectral images are proposed. In this section the fusion RPNs are adopted for feature extraction and the extracted feature maps are utilized for training a BDF using AdaBoost. The approach is similar to the filtered channel features based detectors in Chapter 5, but instead of using the filtered channel features, the deep features extracted from the RPN are used.

In Figure 6.9 the architecture for combining the Conv3 Fusion RPN with a BDF is introduced. Based on the input images, the trained Conv3 Fusion RPN generates region proposals (RoIs). In a slightly modified way, the Conv3 Fusion RPN architecture is used again to process the generated region proposals. As depicted in Figure 6.9 the fusion network is truncated after the conv4_3 layer and the à trous trick [LSD15] [ZLLH16] is applied to increase the resolution of the conv feature maps resulting from conv4. Thus, the feature maps of the conv3 and conv4 layers have the same size. Using the RPN region proposals, each RoI mapped to the output feature maps of conv3-VIS, conv3-IR and conv4 is individually RoI pooled by the three RoI pooling layers (orange). The resulting fixed length feature vectors are concatenated to obtain one feature vector as input for the BDF.

As the Conv1 and Conv2 Fusion RPNs are outperformed by the other fusion architectures they are not used together with a BDF (see Subsection 7.3.4). In Figure 6.10 the second approach of using the RPN feature maps with a BDF is shown. The presented architecture can be used for both, the Conv4 or Conv5 Fusion RPNs. Other than for the first architecture there is no fusion module since the Conv4 and Conv5 Fusion RPNs are fused after conv4 and conv5 layers, respectively. For the multi-spectral input images the RPNs propose regions (RoIs) that potentially contain persons. Then, the trained RPN models are used and modified by truncating the layers after conv4_3 and applying the à trous trick for larger conv feature maps. Similar to Figure 6.9, using the RoIs and the model weights of the Conv4 or Conv5

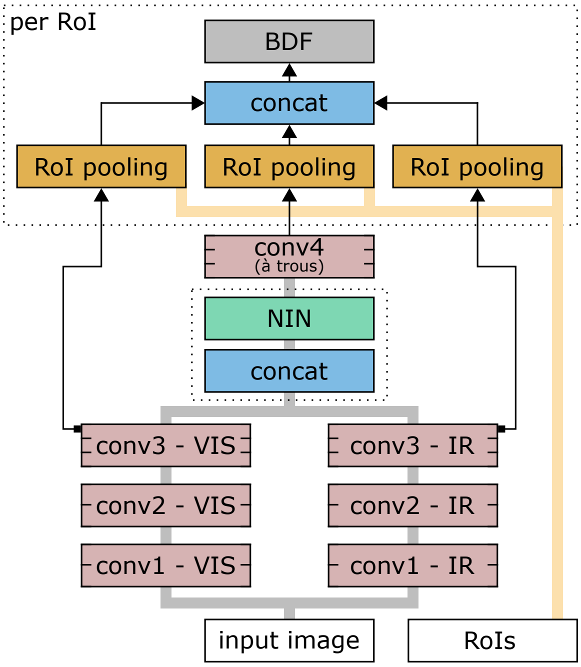


Figure 6.9: Conv3 Fusion RPN for feature extraction combined with a BDF. The RoIs are proposed by the Conv3 Fusion RPN. The extracted conv feature maps are RoI pooled to obtain a fixed length feature vector for the BDF.

Fusion RPN, each RoI is RoI pooled based on the conv feature maps of conv3-VIS, conv3-IR, conv4-VIS (à trous), and conv4-IR (à trous). Thus, there are four fixed length feature vectors that are concatenated and utilized as input for the BDF. To give an impression of the number of features fed into the BDF, the ACF+T+THOG detector with the small model size produces 5,504 features, the ACF+T+THOG with big model size 19,350 features, and the Checkerboards+T+THOG with small model size generates 214,656 features. In comparison, the BDF with Conv3 Fusion RPN produces $12,544 + 12,544 + 25,088 = 50,176$ and with Conv4 or Conv5 Fusion RPN $12,544 + 12,544 + 25,088 + 25,088 = 75,264$ deep features as input feature vector. The number of deep features depends on the RoI pooling output grid (7 × 7), and the number of channels of each utilized conv feature map. As the output feature maps after conv3 have 256 channels, one RoI is pooled to $7 \cdot 7 \cdot 256 = 12,544$ deep features. The output feature maps of conv4 have 512 channels and therefore produce $7 \cdot 7 \cdot 512 = 25,088$ deep features. The same RoI resolution (7 × 7) as Faster R-CNN is adopted, but these RoIs are used on higher resolution feature maps (e.g., conv3_3, conv4_3 or conv4_3 à trous) compared to Fast R-CNN or Faster R-CNN (conv5_3).

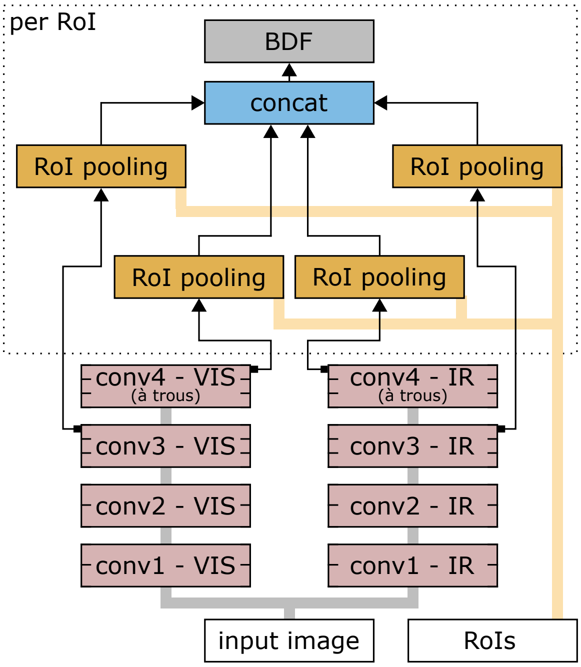


Figure 6.10: Conv4 or Conv5 Fusion RPN for feature extraction combined with a BDF. The RoIs are proposed by the Conv4 or Conv5 Fusion RPN, respectively. The extracted conv feature maps are RoI pooled to obtain a fixed length feature vector for the BDF.

The BDF used for training the classification stage of the proposed person detection framework is similar to the one from Chapter 5 and builds up on the Piotr's Matlab Toolbox [Dol]. First, the training data is generated by using the fusion RPNs as proposal generator on the *KAIST10x-train* training sub-dataset. The generated training data consists of the top-ranked 1,000 proposals of each image, after NMS with IoU overlap of 0.7. As proposed by Zhang et al. [ZLLH16], seven stages of bootstrapping are used (64, 128, 256, 512, 1,024, 1,536, and 2,048 per stage), resulting in 2,048 weak classifiers with a maximum tree depth of 5. The BDF training is performed using Real AdaBoost. The proposals of the Fusion RPN are used for generating positive and negative samples based on the extracted fixed length feature vectors for each RoI as explained in this chapter. These feature vectors have to be labeled w.r.t. their class (person/no person). A region proposal is regarded as person, if the bounding box has an IoU overlap with the GT equal to or higher than 0.8. These RoIs together with the GT boxes are used as positive samples. For the first stage 30,000 negative samples are generated. In each following stage 5,000 new negative samples are added to a maximum number of 50,000 samples. If the maximum number of samples is reached, the new training set comprises the

5,000 newly acquired samples and 45,000 samples that are sampled from the existing set of 50,000 negative samples. Negative samples are generated similar to the positive samples. A proposal of the Fusion RPN is marked as background (no person) if the IoU overlap with the GT is less than 0.5. Overlaps in between 0.5 and 0.8 are ignored. After RoI pooling, the generated feature vectors are used as the negative samples. As the number of available negative samples is larger than the number that can be added every stage, the 5,000 new negative samples are randomly sampled in the first. After the first stage of training, the BDF model is used for hard negative mining. Only negative samples with a score higher than -1 are added as hard negatives to the training set. For the NMS applied on the BDF detection results, an IoU overlap threshold of 0.5 is used.

7 Experiments and Results

This chapter presents the evaluations and results of this thesis. First, the evaluation metrics commonly used for person detection are reviewed. Then, the results for the filtered channel features based detectors are showed that serve as a baseline for the proposed CNN approaches. After presenting the results of the proposed fusion RPNs and the RPNs combined with BDFs, the results are compared to approaches taken from literature. Finally, some qualitative evaluations are presented with example detections and images taken from the KAIST dataset.

7.1 Evaluation Metrics

Before presenting the results, the commonly used evaluation metrics for person detection are recapitulated. The detection results are classified into four different categories as illustrated in Figure 7.1. For evaluation, there are the detections of the applied detector and the GT data provided together with the dataset. All detections provided by the detector are called predicted positives. Those detections contain the actual True Positives (TPs) and False Positives (FPs), which are not contained in the GT data. These FPs are either detections that do not contain a person or can be missing annotations in GT data. Originally, this categorization stems from evaluating classification results. Therefore, the predicted negatives are sliding windows or anchor boxes, which are labeled as non-persons. As for the predicted positives, the predicted negatives can be divided into the category of True Negatives (TNs), i.e. windows correctly classified as not containing any person, and the False Negatives (FNs), which are windows containing persons, but marked as background by the detector. The actual positives are equivalent to the set of GT data.

For determining a detection to belong to a certain category, the detections are matched to the GT data. To decide whether a detection belongs to a given GT bounding box, the score of the detection, and the so-called Intersection over Union

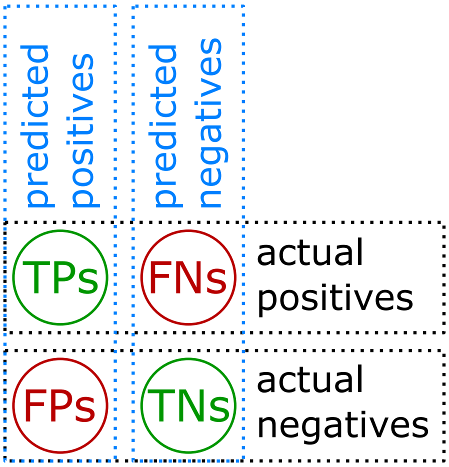


Figure 7.1: Illustration of the confusion matrix used to classify the detection results into True Positives (TPs), False Negatives (FNs), False Positives (FPs), and True Negatives (TNs).

(IoU) overlap criterion are considered. The IoU is a measure resulting from the PASCAL VOC Object Classification Challenge [EVW+10] (see Equation 7.1).

$$\text{IoU}(bb_{dt}, bb_{gt}) = \frac{\text{area}(bb_{dt} \cap bb_{gt})}{\text{area}(bb_{dt} \cup bb_{gt})} \tag{7.1}$$

As the name reveals, the IoU is the ratio between the intersection of detection bounding box bb_{dt} and GT bounding box bb_{gt} and the union of both bounding boxes. In this way, every detection bounding box is compared to every GT box and their IoU overlap is calculated. A commonly used IoU overlap threshold is 0.5, i.e. a detection bounding box can be matched to a GT bounding box, if the IoU overlap is greater than 0.5. For matching detection and GT boxes for an image, the detections are sorted by their scores in descending order to prefer detections with high scores. The GT boxes are re-arranged that the non-ignore boxes are matched first. Then, each detection is compared to each GT box. The detection is matched to the GT box with the highest IoU overlap, as far as the GT box is not already matched to a detection with a higher score. Thus, the best detection for a bounding box is considered as TP. Detections that do not match to any GT bounding box are marked as FPs. FNs are GT boxes without any matched detection bounding boxes. The TNs correspond to the number of sliding windows or anchors used for detection minus the number of FNs. This has to be applied for every image to get the total number of TNs. The ignore regions [DWSP12] are handled similar compared to GT boxes, but instead of matching maximally to one detection box, they can have multiple matched detection bounding boxes. Matches to non-ignore GT boxes are preferred. Ignore regions that are not assigned to any detection are not considered as FNs.

The evaluation metrics for person detection are based on the categorization of detection results. The two commonly used curves are the Precision-Recall (PR) and the Receiver Operating Characteristic (ROC) curves. Davis and Goadrich explain both curves and their relationship [DG06]. For person detection the ROC curve is often referred to as a curve showing the miss rate over the False Positives Per Image (FPPI). This kind of ROC curve is applied in Piotr's Matlab Toolbox [Dol], which is used for evaluation. The miss rate and FPPI is easier to evaluate for humans than the true positive rate and false positive rate of the orginial ROC curve. For example, in automotive applications it is more intuitive to define the FPs per image than a false positive rate. Equation 7.2, Equation 7.3, and Equation 7.5 describe the related metrics that are utilized for creating the curves. For all evaluations in this work, the evaluation scripts of Zhang et al. [ZLLH16] are used that are based on Piotr's Matlab Toolbox.

$$\begin{aligned} R &= \frac{TPs}{TPs + FNs} = \frac{TPs}{\#GTs} \\ P &= \frac{TPs}{TPs + FPs} = \frac{TPs}{\#DTs} \end{aligned} \tag{7.2}$$

$$\begin{aligned} TPR &= \frac{TPs}{TPs + FNs} = \frac{TPs}{\#GTs} \\ FPR &= \frac{FPs}{FPs + TNs} \end{aligned} \tag{7.3}$$

$$\begin{aligned} M = 1 - R &= \frac{FNs}{TPs + FNs} = \frac{FNs}{\#GTs} \\ FPPI &= \frac{FPs}{\#images} \end{aligned} \tag{7.4}$$

$\#GTs$ denotes the total number of GT bounding boxes and according to Figure 7.1 $TPs + FNs = \#GTs$. The total number of detections is denoted by $\#DTs = TPs + FNs$ and the total number of evaluated images by $\#images$. For the PR curve the recall (R) is plotted over precision (P) (Equation 7.2). The original ROC curves plot True Positive Rate (TPR) over False Positive Rate (FPR), where the TPR is equal to R (Equation 7.3). The ROC curve that is used for the evaluations of this thesis plots the Miss rate (M) over the FPPI (Equation 7.5). The miss rate can be calculated from the recall and vice versa. For plotting the curves, these metrics are computed by shifting the score threshold in small steps from the minimum available detection score to the highest. For every small step the score threshold is shifted, a pair can be calculated with the metrics (e.g., M and $FPPI$). Thus, a relatively continuous curve can be plotted.

For comparing the results, two commonly used metrics are reviewed that summarize the detector performance by a single reference value. The log-average Miss Rate (MR) [DWSP12] is based on the ROC curve and the Average Precision (AP) [EVW+10] is based on the PR curve. Both methods use nine reference points that are evenly spaced in log-space in the interval between 10^{-2} and 10^{0}. For curves that end before the end of the interval is reached, the missing reference points are approximated by the value of the nearest reference point. Conceptually, MR is similar to the AP. Equation 7.5 and Equation 7.6 describe the calculation for MR and AP. The parameter N corresponds to the number of reference points and is set to $N = 9$ [DWSP12].

$$MR = \exp(\frac{1}{N} \sum_{i=1}^{N} \ln(M_i)) \tag{7.5}$$

Equation 7.5 describes the calculation for the MR. The miss rates M_i at the $N = 9$ reference points are averaged, after scaling them logarithmically by using the natural logarithm. The reference points are FPPI rates in the interval between 10^{-2} and 10^{0}. The resulting average value is scaled by applying the natural exponential function. For calculating the AP (Equation 7.6), the $N = 9$ precision values P_i are averaged at the given reference points. These two reference values are commonly used for evaluating the detector performance.

$$AP = \frac{1}{N} \sum_{i=1}^{N} P_i \tag{7.6}$$

For evaluating proposal generators like the RPN, curves are used that plot recall against IoU or recall against the number of proposals [ZLLH16] [LZWM16]. For the curve, which plots recall against different IoU overlap thresholds, the recall is calculated for various IoU thresholds commonly reaching from 0.5 to 1.0. These kinds of curves are used to evaluate the accuracy of the proposals. The second type of curves that plots the recall against the number of proposals is used to compare the proposal generators for their classification performance. The faster the curves reach towards 1.0, the less FNs are contained in the detection sets.

7.2 Results of Person Detection based on Filtered Channel Features

This section describes the experiments with the filtered channel features based detectors. In Table 7.1, the results for three different parameter sets are presented that are described in Table 5.1. The original ACF detector model provided by Hwang et al. [HPK+15] achieves an MR of 54.40 %. Two model sizes of the ACF detector are regarded for each parameter set (Config1, Config2 and Config3). There are small (32 × 64 with padding) and big (60 × 120 with padding) model sizes. The smaller model size is proposed by Rajaram et al. [ROBT16] for VIS detectors and also used in the multi-spectral approach of Hwang et al. [HPK+15]. On the other side, Ohn-Bar et al. [OBT16] and Zhang et al. [ZBS15] use the big model size for parameterization of their detectors.

With using the first parameter set *Config1* the model capacity is enlarged by increasing the number of weak classifiers, training stages, negative training samples, and the maximal tree depth. Furthermore, Real AdaBoost is used for training instead of the Discrete AdaBoost used by Hwang et al.. The increasing MR values indicate that the increased model size is too large for only using *KAIST-train* for training. A reduced MR is achieved for training with the larger *KAIST10x-train* subdataset. Thus, for an enlarged model capacity an appropriate dataset is necessary, which provides more training data.

For the second parameter set *Config2* the stride and shrinkage parameter for the channel features pyramid are increased. Furthermore, all scales of the channel pyramid are computed instead of approximated. The results show an additional increase of the MR. It is assumed that the increased MR results from the bigger stride and shrinkage. Therefore, this two parameters are reset to the default values for *Config3*. With this third parameter set, the performance of the filtered channel features based detectors is improved compared to original ACF-T-THOG of Hwang et al..

Training Configurations		**MR (%)**		
Training Dataset	Parameter Set	**ACF** (big)	**ACF** (small)	**Checkerboards** (small)
KAIST-train	Config1	80.53	67.46	63.33
	Config2	71.63	71.50	69.21
	Config3	68.33	65.23	63.17
KAIST10x-train	Config1	74.98	47.21	46.57
	Config2	47.59	58.60	59.43
	Config3	44.31	42.57	**39.12**

Table 7.1: Evaluation of the filtered channel features based detectors for different parameter sets (Table 5.1). The evaluation of the provided model of Hwang et al. [HPK+15] yields to an MR of 54.40 % (model is trained on *KAIST-train*).

The Checkerboards-T-THOG detector (denoted with Checkerboards) outperforms both model sizes of the ACF-T-THOG detectors (denoted with ACF). But when considering the number of features used as input for the BDF, the ACF with the big model size is assumed to outperform the small model size. The small ACF detector computes 5,504 features, the big ACF detector 19,350 features and the Checkerboards detector 214,656 features. For the implementation in Piotr's Matlab Toolbox, the number of features directly influences the time for performing a detector training and the size of the Random Access Memory (RAM) required for storing the positive and negative samples. During training they hold all positive and negative samples constantly in RAM.

In Figure 7.2, the results of three detectors for Config3 are plotted and compared to the original ACF-T-THOG model. Surprisingly, the IoU curve of the big sized ACF detector always lies above the other curves. This means the big sized detector finds more TPs with correct localization compared to the two smaller sized detectors. It is assumed that the detectors with a larger model size have a larger sliding window on each scale of the channel features pyramid than the detectors with a small model size, and therefore the detections of the detectors with larger model size predict

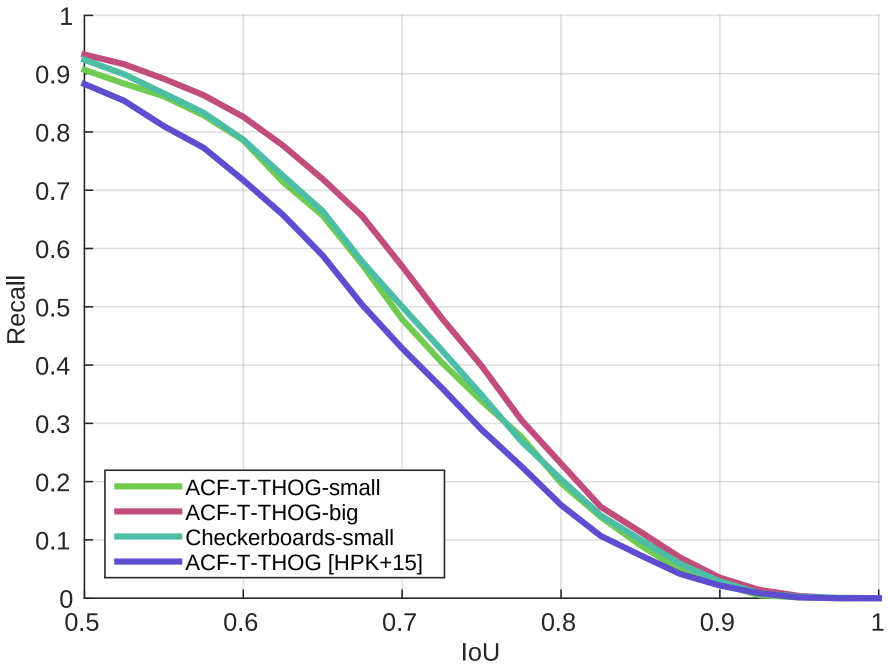


Figure 7.2: IoU vs. recall curve for the three detectors of parameter set *Config3* compared to the original ACF-T-THOG model [HPK+15].

larger detection boxes. These assumptions explain the higher recall values, since it is more likely that a larger detection box matches to a GT box. Rajaram et al. [ROBT16] state: Choosing a smaller model size allows the detection of smaller persons at the cost of lower detection accuracy. On the other side, a larger model size yields better detection accuracy for large persons at the cost of missing smaller ones. The IoU curve in Figure 7.2 is used to evaluate the localization accuracy of the three detectors compared to the original ACF.

Figure 7.3 presents the ROC curve for the three parameter tuned filtered channel features based detectors compared to the original ACF+T+THOG detector. The Checkerboards detector outperforms the three other detectors, while the original detector performs worst. It is assumed that the Checkerboard detector's performance results from the large and rich feature pool, and the small model size that provides a low number of FNs. The assumption of using the big model size for generating larger detection boxes is that if a window contains a person, for the small model size, the generated features mostly belong to the person. For the larger model size the considered window additionally contains background features and therefore it is harder to provide a correct classification result. Thus, the number of FPs and FNs increases and the ROC curve is shifted to the right and upwards, re-

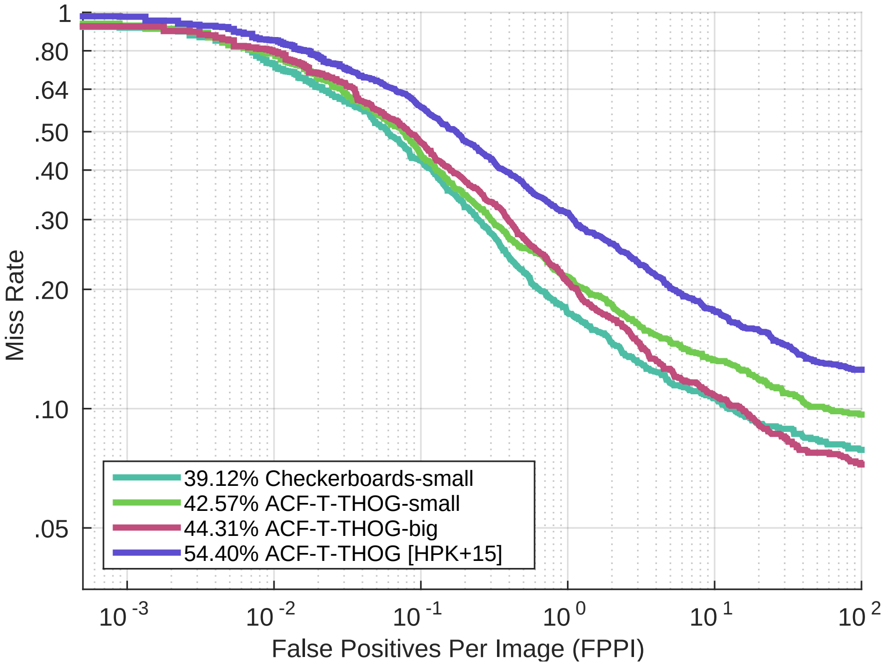


Figure 7.3: ROC curve for the three detectors of parameter set *Config3* compared to the original ACF-T-THOG model [HPK+15].

spectively. The number of FPs and FNs increases stronger than the number of TPs and therefore the ACF detector with small model size outperforms the one with the big model size when considering the MR. Training the Checkerboards detector for the big model size is not possible using this implementation due to lacking memory.

Compared to Hwang et al., the usage of the larger *KAIST10x-train* sub-dataset for training is introduced, both model sizes are evaluated, a model of bigger capacity (number of weak classifiers, tree depth and number of negative samples) is used and the Checkerboards filterbank is applied to the multi-spectral ACF.

7.3 Results of Person Detection based on Deep Learning

This section presents the results of pre-finetuning the RPN with the auxiliary datasets without fusion. Different parameter sets are evaluated including the size of the datasets used for training, and horizontal flipping as data augmentation. The RPN is initialized with the VGG-16 model weights, pre-trained on the ImageNet dataset.

7.3.1 RPN Results on Auxiliary Datasets

The results for training the RPN on the Caltech dataset are presented in Table 7.2. The parameter set *normal* represents the training of the RPN on the original Caltech dataset as introduced by Dollár et al. [DWSP12]. The improved annotations introduced by Zhang et al. [ZBO+16] are used for *new-annotations*. The parameter set *flipping* means applying random horizontal flipping of the training samples to perform data augmentation. Using the larger *Caltech10x-train* sub-dataset for training decreases the MR by about 4 percentage points. Surprisingly, the usage of the improved annotations has no influence on the detector performance for the large training subset. For the smaller training subset a performance gain of about 1 percentage point can be achieved, same for flipping used to augment the training data. For the large training set the performance of the RPN is even deteriorated. With the parameter set *normal*, an MR of 13.09 % is achieved for using the larger training subset. This is slightly better than the MR achieved by Zhang et al. [ZLLH16] (14.9 %).

Caltech RPN

Training Dataset	Parameter Set	**MR (%)**
Caltech-train	normal	18.65
	new-annotations	17.11
	flipping	17.10
Caltech10x-train	normal	**13.09**
	new-annotations	13.15
	flipping	14.55

Table 7.2: Results for training the RPN model on two different training subsets (*Caltech-train* and *Caltech10x-train*) and with three different parameter sets.

The CaltechR dataset images contain the cloned red channels of the RGB Caltech images. For the smaller subset a small gain can be reached when using horizontal flipping as shown in Table 7.3. Training on the large training sub-dataset (*CaltechR10x-train*) yields to a performance gain similar as for the Caltech dataset.

CaltechR RPN		
Training Dataset	Parameter Set	**MR (%)**
CaltechR-train	normal	23.75
	flipping	22.21
CaltechR10x-train	normal	**18.53**
	flipping	20.07

Table 7.3: Results for training the RPN model on two different training subsets (*CaltechR-train* and *CaltechR10x-train*) and with two different parameter sets.

The IR dataset CVC-09 consists of images with the single IR gray-scale image cloned three times to achieve a 3-channel image analogously to the CaltechR dataset. Other than for the CaltechR dataset, horizontal flipping results in a performance gain for both training subsets. The decrease of the MR for the *CVC10x-train* subset is significant. The results are listed in Table 7.4.

CVC RPN		
Training Dataset	Parameter Set	**MR (%)**
CVC-train	normal	38.97
	flipping	37.10
CVC10x-train	normal	37.27
	flipping	**30.67**

Table 7.4: Results for training the RPN model on two different training subsets (*CVC-train* and *CVC10x-train*) and with two different parameter sets.

Considering these results, improved annotations and horizontal flipping for data augmentation yield to a decreased MR, especially for training datasets with an insufficient amount of training data. As the models trained on the original training subsets perform well and flipping causes an increased amount of training data, the RPNs trained on the larger training subsets with the *normal* parameter set are used for further experiments. Increased training data extends the time for training an RPN model.

7.3.2 RPN Results on the KAIST-VIS Dataset

Based on the auxiliary datasets used for pre-finetuning, finetuning of these RPN models is performed on the KAIST-VIS dataset that consists only of the VIS images of the KAIST dataset. The annotations remain unchanged. There are three parameter sets. The *normal* set represents training the RPN based on the pre-trained VGG-16 ImageNet model and *flipping* denotes additional data augmentation by horizontal flipping. The third option trains the RPN on KAIST-VIS but initializes the RPN model pre-finetuned on the *Caltech10x-train* training subset (see Table 7.2).

KAIST-VIS RPN		
Training Dataset	Parameter Set	**MR (%)**
KAIST-train (VIS images)	normal	50.93
	flipping	49.52
	on Caltech10x model	49.74
KAIST10x-train (VIS images)	normal	48.13
	flipping	**47.71**
	on Caltech10x model	49.07

Table 7.5: Results for training the VIS RPN model on two different training subsets (*KAIST-train* and *KAIST10x-train*) using only the VIS images and with three different parameter sets.

The results for the RPN trained on the KAIST-VIS data are presented in Table 7.5. The smaller training subset yields worse results than can be achieved for the larger training subset. The performance gain by applying flipping for data augmentation can be recognized for both datasets. As reported by Wagner et al. [WFHB16], a strongly decreased MR is expected by using the RPN model pre-finetuned on the *Caltech10x-train* training set for initialization. Instead, the MR is slightly increased when using the large training subset combined with pre-finetuning. For the smaller training set, data augmentation and initialization with the pre-finetuned model increases the performance slightly. For further experiments of this work the parameter sets *normal* and *on Caltech10x model* are used.

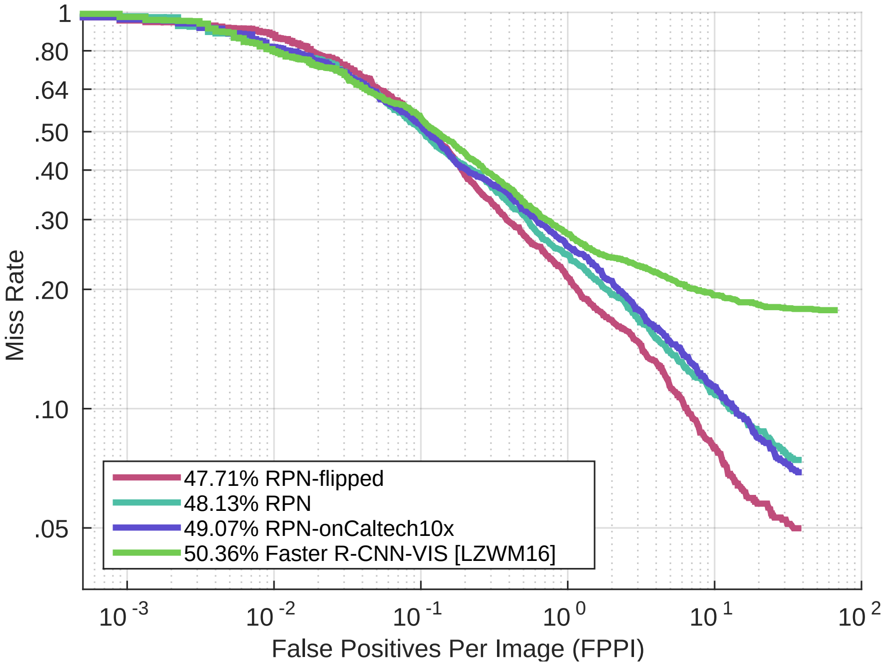


Figure 7.4: ROC curve of the VIS RPNs for the three parameter sets in Table 7.5 when using only VIS images of the *KAIST10x-train* subset for training. The results are compared to those of Faster R-CNN-VIS [LZWM16].

As for the auxiliary datasets, the results show that the larger training subset leads to a better performance. The ROC curves for the three parameter sets are plotted in Figure 7.4 when using the larger training set. The results are compared to those achieved by Liu et al. [LZWM16] for their Faster R-CNN-VIS. They also trained and evaluated their approach using only VIS images of the KAIST dataset. By using the same training subset, but the Faster R-CNN instead of only the RPN, they achieve slightly worse results. In this way, the conclusions of Zhang et al. [ZLLH16] that the RPN as stand-alone outperforms the Faster R-CNN with its additional classification network, can be confirmed. Considering the stagnating curve of the Faster R-CNN-VIS detector in Figure 7.4, the influence of the classification network, which rejects TPs and discards too less FPs, are visible. Thus, the miss rate of the detector increases. For the following experiments the influence of pre-finetuning is analyzed and therefore the flipping approach is discarded. Furthermore, only the models trained on the bigger training subsets are considered.

7.3.3 RPN Results on the KAIST-IR Dataset

Similar to the previous subsection, this one evaluates the finetuning of the RPN for the KAIST-IR subset. Only the IR images of *KAIST-train* and *KAIST10x-train* sub-dataset are used to construct the KAIST-IR subset. The first two parameter sets (*normal* and *flipping*) are analogous to the previous subsection and represent training based on the pre-trained VGG-16 model. The third option (*on Caltech10x model*) means using the RPN model pre-finetuned on the *Caltech10x-train* sub-dataset for initialization. Based on the initialization, the RPN model is finetuned on both KAIST-IR subsets. The fourth option (*on CaltechR10x model*) uses the RPN model pre-finetuned on the *CaltechR10x-train* training set for initialization. The last option (*on CVC10x model*) describes initialization with an RPN pre-trained on the *CVC10x-train* training sub-dataset.

KAIST-IR RPN		
Training Dataset	Parameter Set	**MR (%)**
KAIST-train (IR images)	normal	47.03
	flipping	46.65
	on Caltech10x model	48.58
	on CaltechR10x model	48.57
	on CVC10x model	47.61
KAIST10x-train (IR images)	normal	48.13
	flipping	45.80
	on Caltech10x model	46.91
	on CaltechR10x model	**44.17**
	on CVC10x model	44.26

Table 7.6: Results for training the IR RPN model on two different training subsets (*KAIST-train* and *KAIST10x-train*) using only the IR images and with five different parameter sets.

Table 7.6 lists the results achieved for different parameter sets. As for all previous experiments, the training on the smaller training subset is worse compared to the results of the larger one. Data augmentation by applying horizontal flipping improves the RPN performance. For the pre-finetuned RPN models used for training on the *KAIST10x-train* IR sub-dataset, improvements can be recognized compared to only using the pre-trained VGG-16 models for initialization. For further experiments the RPN models based on the pre-finetuned IR RPNs and the IR RPNs trained on the pre-trained VGG-16 model are used.

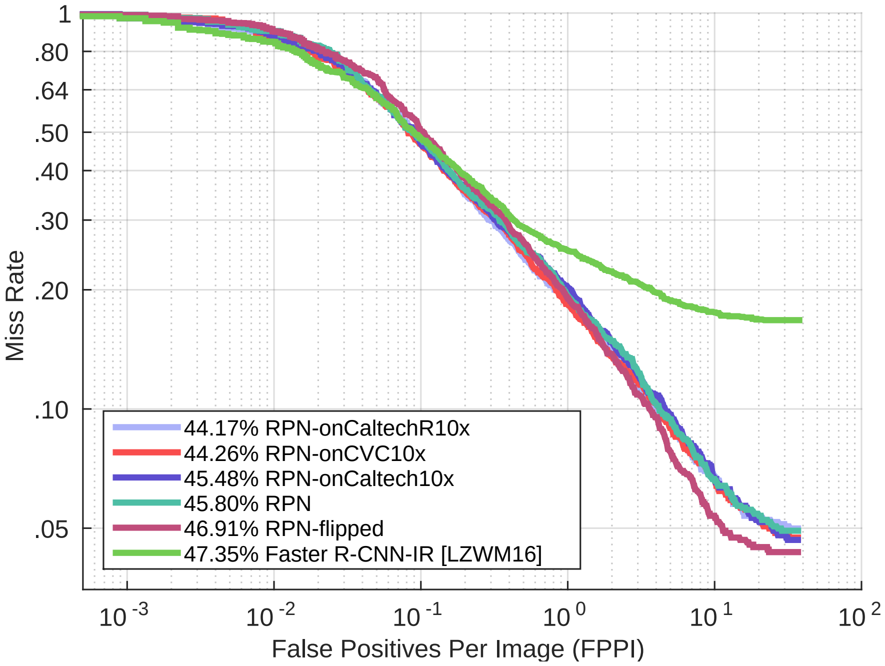


Figure 7.5: ROC curve of the IR RPNs for the five parameter sets in Table 7.6 when using only IR images of the *KAIST10x-train* subset for training. The results are compared to those of Faster R-CNN-IR [LZWM16].

Similar to Figure 7.4, the comparison of different parameter sets with the Faster R-CNN-IR of Liu et al. [LZWM16] is plotted in Figure 7.5. The negative influence of the classification network as described in the previous subsection can be recognized. With this results, the observation of Wagner et al. [WFHB16] can be confirmed that using pre-finetuning improves the RPN performance. Furthermore, the replication of IR gray-scale images for imitating an IR image, consisting of 3 channels leads to better results than using three different channels (compare RPN-onCaltechR10x and RPN-onCaltech10x). When comparing the results of VIS and IR RPNs (Table 7.5 and Table 7.6) there is a performance difference. The IR RPN outperforms the VIS RPN, although both RPNs are trained and evaluated with the same annotations. This supports the statement of Teutsch et al. [TMHB14] that person detection in IR images is commonly better than using VIS images.

7.3.4 RPN Results of Fusion Approaches on the KAIST Dataset

After training the RPNs without fusion in the previous subsections, results of different fusion approaches from Section 6.4 are presented in this subsection. Based on the different pre-finetuned RPN models for training the finetuned RPNs on VIS and IR data separately, the three different configurations are numerated from 1 to 3 in Table 7.7. Option 1 is called *KAIST* Fusion RPN and is based on the two RPNs initialized with the VGG-16 ImageNet model and trained on the *KAIST10x-train* training set, for VIS and IR images separately. The second option called *CaltechR* consists of the RPNs using the pre-finetuned VIS RPN *on Caltech10x model* and *on CaltechR10x model* as IR RPN. The third option is similar to the second one, but *on CVC10x model* is used as IR RPN and named *CVC*. These training options only determine the individual VIS and IR RPNs used for initializing the convolutional layers of the fusion models, located before the fusion module (Figure 6.8). All shared convolutional layers after the fusion module are initialized using the pre-trained VGG-16 model. Then, the initialized Fusion RPNs are trained (finetuned) on the multi-spectral KAIST images (VIS+IR). In the previous subsections the results when using the smaller *KAIST-train* training sub-dataset are always worse than compared to the RPN models trained with the larger *KAIST10x-train* training subset. Therefore, only the *KAIST10x-train* sub-dataset is used in the remainder of this work for training. For each training option there are five different fusion approaches that are evaluated.

The results listed in Table 7.7 show that the fusion applied after the conv1 and conv2 layers are clearly outperformed by the other fusion approaches. The reason for this observation is that for all the different finetunings and pre-finetunings, the layers of conv1 and conv2 are not adapted by training and therefore have the same weights as the pre-trained VGG-16 model. Thus, these low-level features respond to general features in the input image without higher semantic meanings, and are difficult to fuse. Furthermore, the ERF sizes of the convolutional layers (see Table 6.1) are small compared to the persons that are considered. After the conv1 layers there is an ERF size of 6×6 and after the conv2 layers the ERF size is 16×16 pixels. The small ERFs also provide low-level features considering only parts of the persons. The fusion approaches of conv3, conv4, and conv5 have similar results. The slight

KAIST Fusion RPNs							
Pre-finetuned RPN Model			**Fusion Approach**				
Training Option	VIS RPN	IR RPN	Conv1	Conv2	Conv3	Conv4	Conv5
(1) KAIST	KAIST-VIS	KAIST-IR	45.21	40.99	36.46	36.11	36.23
(2) CaltechR	Caltech + KAIST-VIS	CaltechR + KAIST-IR	47.94	39.87	36.86	35.81	36.18
(3) CVC	Caltech + KAIST-VIS	CVC + KAIST-IR	44.71	41.49	**35.50**	36.42	35.52
			MR (%)				

Table 7.7: Results for different RPN fusion approaches (Figure 6.8) with different training options. The results are evaluated by considering the MR on the *KAIST-test-Reasonable* sub-dataset.

differences may result from the training procedure that contains randomly generated mini-batches. The Conv5 Fusion approach in Figure 6.8 (e) leads to similar results compared to the approach of Liu et al. in Figure 6.8 (f) and is not considered in the remainder of this thesis. The approach in Figure 6.8 (f) is denoted with Conv5 Fusion RPN.

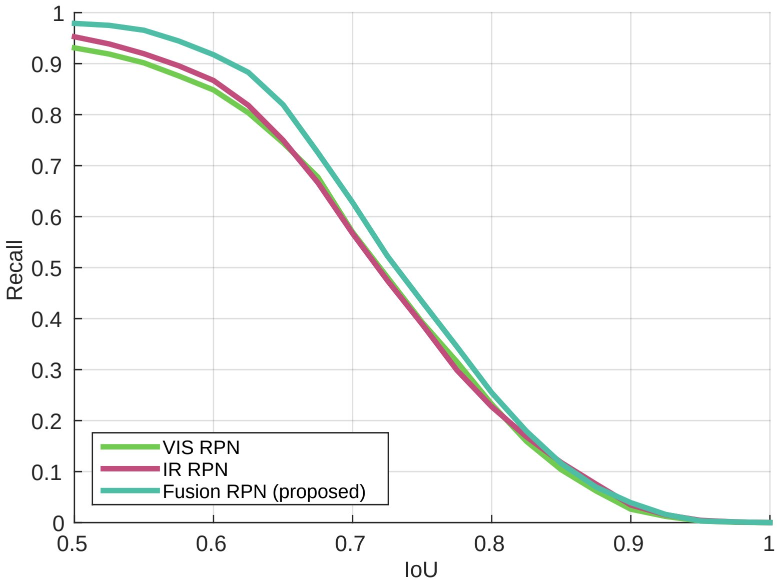


Figure 7.6: IoU vs. recall curve for the Conv3 CVC Fusion RPN (training option (3) and fusion after the conv3 layers) compared to the VIS and IR RPNs used to initialize the training of this Fusion RPN.

The IoU vs. recall curve in Figure 7.6 is used for evaluation. The plot shows the performance of the most promising fusion architecture called the Conv3 CVC Fusion RPN. Conv3 means that the fusion is applied after the conv3 layers (Figure 6.8). CVC refers to training option (3) and means that the VIS RPN is pre-finetuned on the *Caltech10x-train* sub-dataset. The IR RPN is pre-finetuned on the *CVC10x-train* training set. As initially motivated, the goal is to show that VIS and IR data contain complementary information. The VIS and IR RPNs have lower recall than the Fusion RPN. Thus, it can be stated that the fusion enables the RPN to find more TPs. Liu et al. [LZWM16] show that VIS and IR RPN do not provide only the same detections. Therefore, it can be stated that the VIS and IR RPNs provide complementary information that can be fused to create features of higher discriminativity.

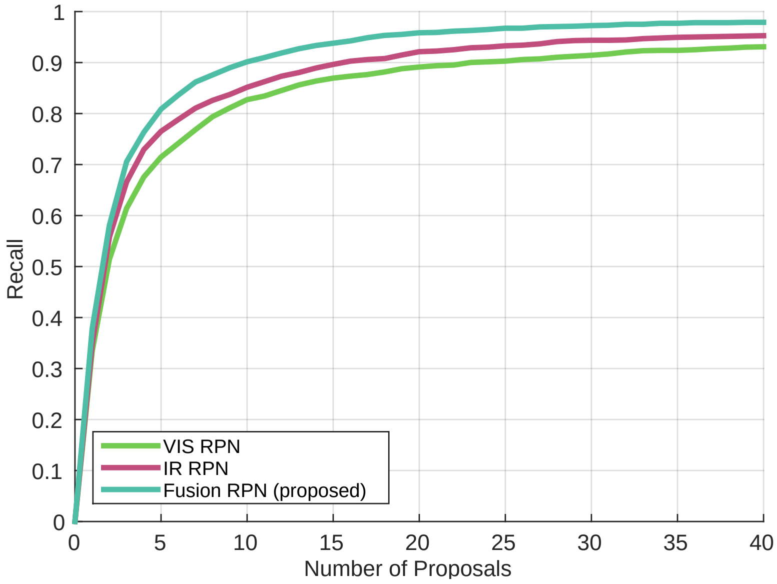


Figure 7.7: Number of proposals vs. recall curve for the Conv3 CVC Fusion RPN (training option (3) and fusion after the conv3 layers) compared to the VIS and IR RPNs used to initialize the training of this Fusion RPN.

Additionally, Figure 7.7 shows that the curve of the Conv3 CVC Fusion RPN is constantly above the other two curves. Independent of the number of proposals, the Fusion RPN obtains a higher number of TPs (recall). This means that the RPN layers (cls-prop and bbox-pred) have more discriminative features for performing localization and classification. The Fusion RPN reaches a recall of 0.9 for only 10 considered proposals. Thus, it can be concluded that the fusion improves the classification capability of the RPN, and increases the number of TPs compared to the VIS and IR RPNs.

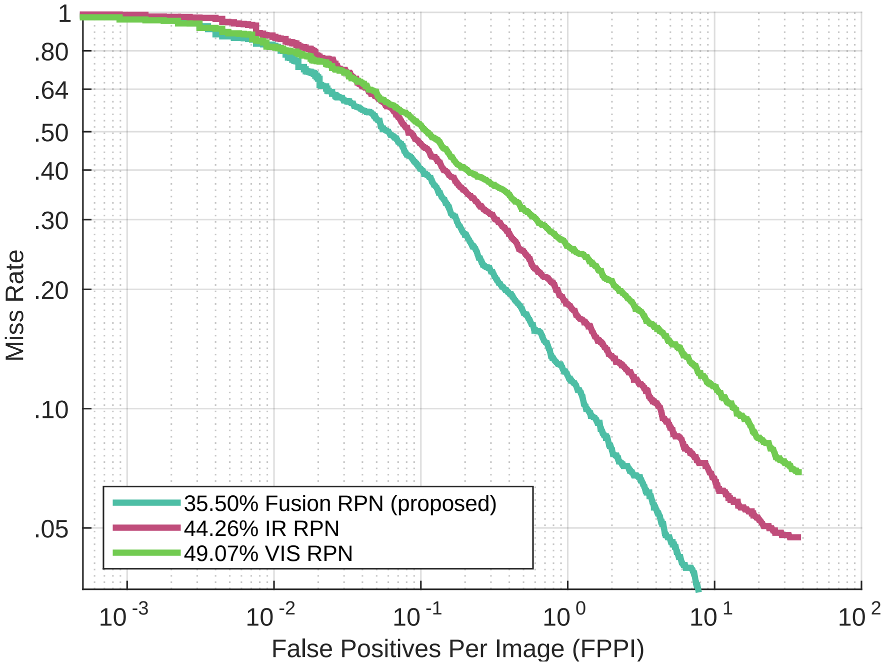


Figure 7.8: ROC curve for the Conv3 CVC Fusion RPN (training option (3) and fusion after the conv3 layers) compared to the VIS and IR RPNs used to initialize the training of this Fusion RPN.

In both figures (Figure 7.6 and Figure 7.7), the recall lies above the recall of the two separate RPNs (VIS and IR). This influences the ROC curve in Figure 7.8 and yields to a better MR for the Fusion RPN. The fusion architecture generates a significant performance boost, yielding to an MR of 35.50 % for the Fusion RPN compared to 44.26 % for the IR RPN and 49.07 % for the VIS RPN.

In Figure 7.9 the proposed fusion approach, the Conv3 CVC Fusion RPN, is compared to the different Faster R-CNN approaches proposed by Liu et al. [LZWM16]. The RPN architecture achieves the same results as the Faster R-CNN framework. With this subsection the complementary information of the VIS and IR images is shown, and it is definitely confirmed that the RPN achieves similar results compared to the Faster R-CNN approach on the KAIST dataset.

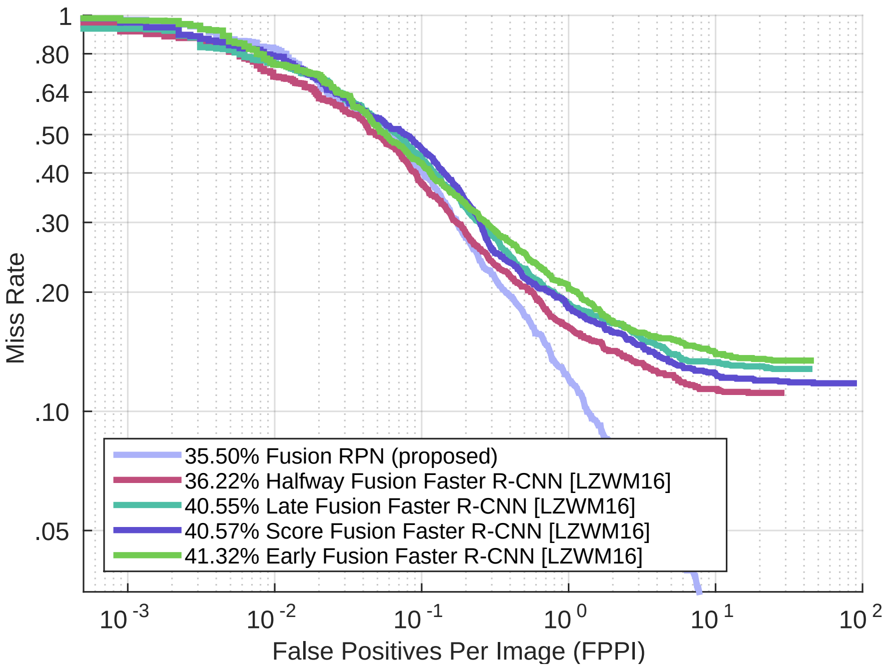


Figure 7.9: Comparison of the proposed Conv3 CVC Fusion RPN to the Faster R-CNN fusion approaches of Liu et al. [LZWM16] evaluated on the *KAIST-test-Reasonable* sub-dataset.

7.3.5 Results of Fusion RPN and BDF on the KAIST Dataset

This subsection presents the results for using the Fusion RPNs together with a BDF as proposed by Zhang et al. [ZLLH16]. The results of the architectures explained in Section 6.5 are listed in Table 7.8. Similar to the previous section there are the three testing options that define the Fusion RPNs used to provide the RoIs and the conv feature maps for training and evaluation of the BDF. The testing options describe the different RPN models used in the pre-finetuning stage for initializing the VIS and IR RPNs for finetuning. All mentioned results use the RoI pooled conv3 and conv4 (with à trous trick) feature maps as input for the BDF classifier. The best resulting detector is the Conv3 CVC Fusion RPN with BDF.

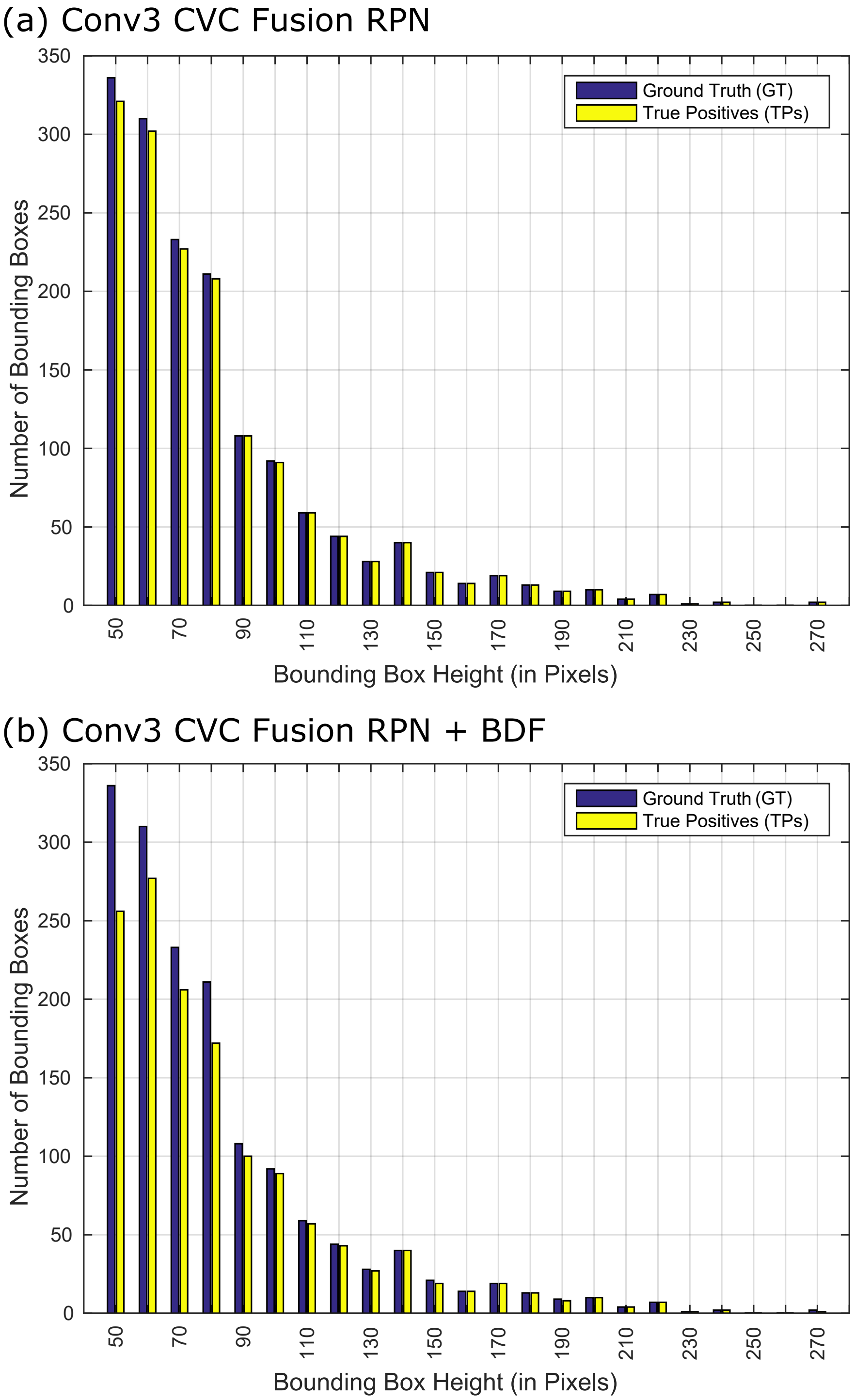
The following two histograms in Figure 7.10 show the TPs of (a) the Conv3 CVC Fusion RPN and (b) the Conv3 CVC Fusion RPN with BDF. By comparing both histograms a slight reduction of TPs resulting from the BDF can be recognized. According to the histograms, the BDF classifier misclassifies especially small-scale persons while classifying nearly all far and medium-scale persons correctly. The detection results of the Fusion RPN consists of the 40 top-ranked detection boxes

KAIST Fusion RPNs with BDF					
Pre-Finetuned RPN Model			**Fusion Approach**		
Testing Option	VIS RPN	IR RPN	Conv3	Conv4	Conv5
(1) KAIST	KAIST-VIS	KAIST-IR	29.94	30.19	30.68
(2) CaltechR	Caltech + KAIST-VIS	CaltechR + KAIST-IR	31.27	29.93	31.15
(3) CVC	Caltech + KAIST-VIS	CVC + KAIST-IR	**29.83**	30.58	30.92
			MR (%)		

Table 7.8: Results for different RPN fusion approaches (Figure 6.8) combined with a BDF classifier. The results are evaluated by considering the MR on the *KAIST-test-Reasonable* sub-dataset.

per image, whereas the number of detection boxes of the Fusion RPN with BDF per image depend on the BDF, but are definitely less than those of the Fusion RPN. This shows that the RPN works as proposal generator with high recall, and that the recall is reduced by the application of the BDF. To improve the overall detector performance, the BDF has to be improved for small-scale persons or another classifier has to be used for reducing the FPs. When checking the classification rate of the BDF by comparing the TPs of the Fusion RPN (1,530 TPs) to those of the Fusion RPN with BDF (1,365 TPs), the BDF classifier achieves a classification rate of around 89 %, which is an acceptable performance for the BDF.

The ROC curve in Figure 7.11 is plotted for comparing the Conv3 CVC Fusion RPN with BDF to the models it arises from: Conv3 CVC Fusion RPN, IR RPN based on the CVC training subset, and VIS RPN based on Caltech training set. There is an improvement of around 20 percentage points from the pure VIS RPN compared to the proposed Fusion architecture with BDF. The ROC curve of the Fusion RPN with BDF ends at a FPPI rate of around 10^0 as there are no more FPs left. This confirms the significant reduced number of FPs, recognized when considering the FPs of the Conv3 CVC Fusion RPN with and without BDF.


(a) Conv3 CVC Fusion RPN

(b) Conv3 CVC Fusion RPN + BDF

Figure 7.10: Histograms of TPs for (a) the TPs of the Conv3 CVC Fusion RPN, and (b) the TPs of the Conv3 CVC Fusion RPN with BDF, compared to the GT data (*KAIST-test-Reasonable*).

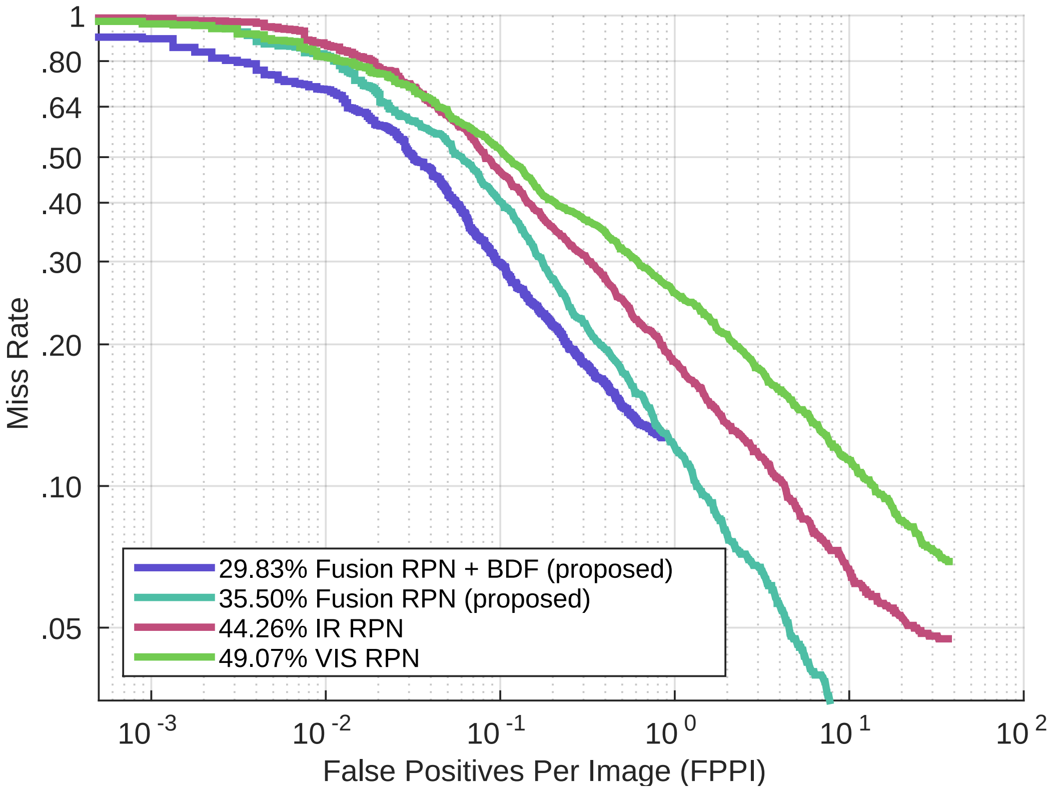


Figure 7.11: Comparison of the proposed Conv3 CVC Fusion RPN with BDF to the RPNs without BDF. The results are evaluated by considering the MR on the *KAIST-test-Reasonable* sub-dataset.

7.4 Summary

In this section, the proposed methods are compared to baseline approaches taken from the literature. The Conv3 CVC Fusion RPN with BDF, the Conv3 CVC Fusion RPN, and the filtered channel features based detectors are compared to the Halfway Fusion Faster R-CNN of Liu et al. [LZWM16] and the Late Fusion CNN of Wagner et al. [WFHB16], who use an ACF+T+THOG detector for proposal generation. Additionally, the results of the ACF+T+THOG detector of Hwang et al. [HPK+15] are shown. Compared to the Halfway Fusion Faster R-CNN an improvement of 5 MR percentage points is achieved with the best proposed detector. With adapting the Checkerboards detector for multi-spectral images and tuning the ACF-T-THOG detector, it can be stated that there is still potential to improve the performance of filtered channel features based detectors. The relative improvement of the best proposed detector compared to the ACF-T-THOG detector of Hwang is 45 %.

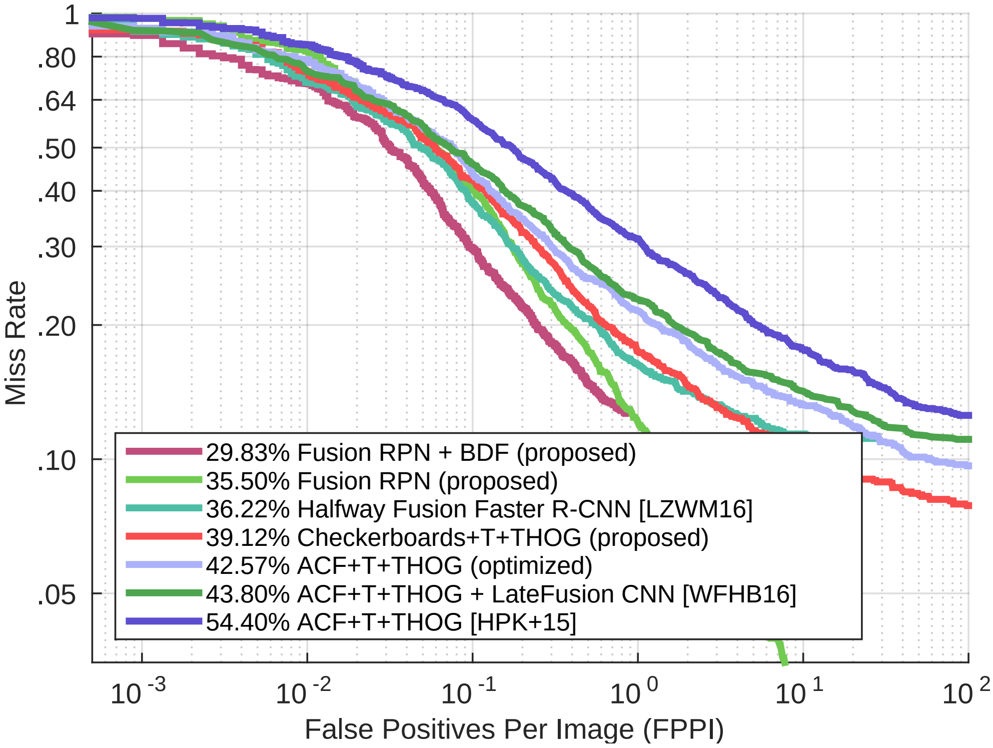


Figure 7.12: Comparison of the proposed Conv3 CVC Fusion RPN with BDF and the Conv3 CVC Fusion RPN to baseline approaches taken from the literature. The results are evaluated by considering the MR on the *KAIST-test-Reasonable* sub-dataset.

The two best proposed approaches are the Fusion RPN with 35.50 % MR and the Fusion RPN with BDF with 29.83 % MR, followed by the Halfway Fusion Faster R-CNN of Liu et al. with 36.22 % MR. The Checkerboards+T+THOG detector is next, which achieves a remarkably MR of 39.12 %, and the parameter tuned ACF+T+THOG detector with 42.57 % MR. Both filtered channel features based detectors (hand-crafted features) are ranked before the Late Fusion CNN of Wagner et al. with 43.80 % MR, which is based on deep learning. As baseline the original ACF-T-THOG model of Hwang et al. is utilized that is evaluated with the evaluation scripts taken from Zhang et al. [ZLLH16] and achieves an MR of 54.50 %.

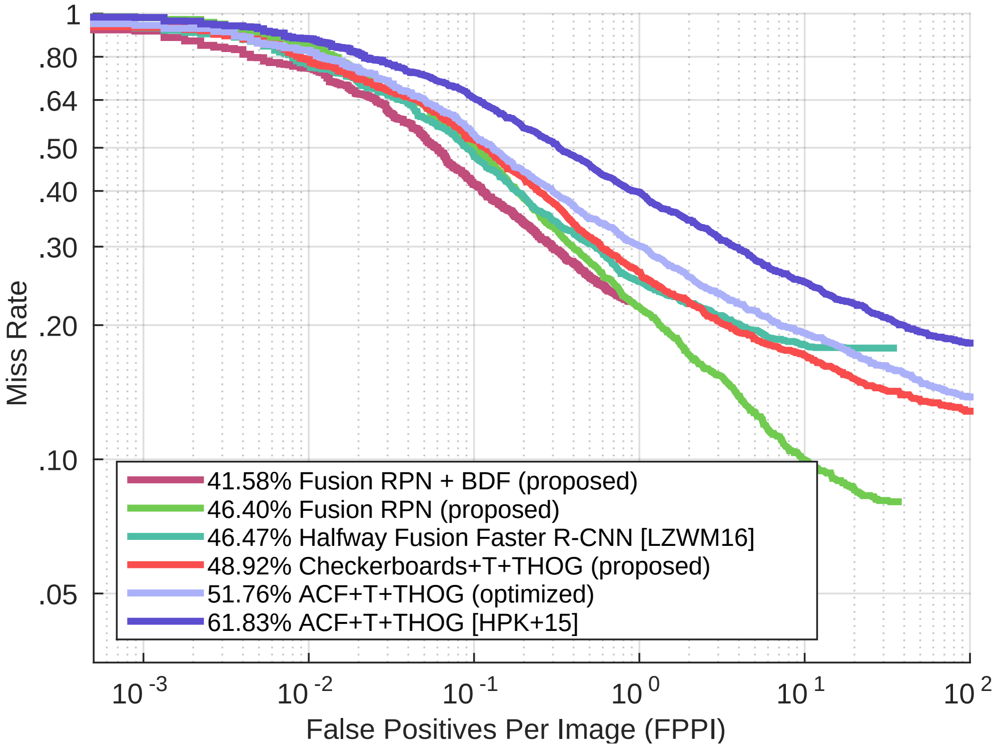


Figure 7.13: Comparison of the proposed Conv3 CVC Fusion RPN with BDF and the Conv3 CVC Fusion RPN to baseline approaches taken from the literature. The results are evaluated by considering the MR on the *KAIST-test-All* sub-dataset.

As stated in Chapter 4, the detector performance can be improved towards small-scale persons and therefore the proposed detectors are evaluated on the entire KAIST dataset instead of the *Reasonable* subset only. The proposed detectors, Conv3 CVC Fusion RPN with BDF and Conv3 CVC Fusion RPN, and the filtered channel features based detectors are evaluated on the *KAIST-test-All* testing subset, to provide results evaluating on the complete KAIST dataset. These results are compared to those of the baseline approaches. The ranking remains the same, while the MRs are raised by around 10 percentage points. Similar to occlusion handling, the detection of small-scale persons is still an active field of research. Thus, this results are proposed as a new baseline on the KAIST dataset.

7.5 Qualitative Evaluations

In order to provide a visual impression, some qualitative evaluations are shown in this section. Red bounding boxes represent the GT boxes, green boxes the detections of the *Conv3 CVC Fusion RPN with BDF* detector, and orange boxes refer to ignore regions. For each example, the VIS image (left) is plotted together with its corresponding IR image (right).

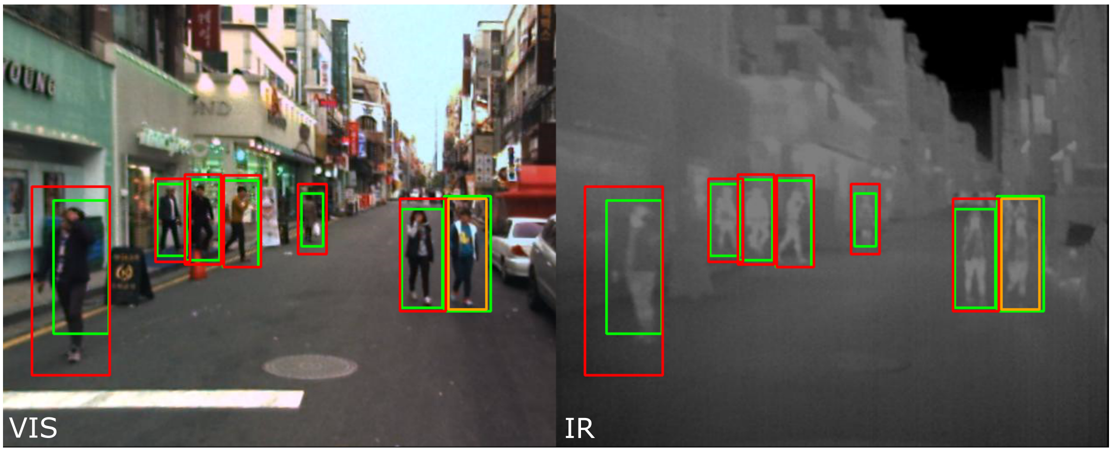


Figure 7.14: Visualization of the GT and detection boxes of *KAIST-test-Reasonable* testing subset for the Conv3 CVC Fusion RPN with BDF. GT boxes are colored in red, detection boxes in green, and ignore regions are marked by orange boxes.

The multi-spectral image pair in Figure 7.14 (frame 1,096) shows that all GT bounding boxes have an equivalent detection box. On the right side of the images there is an ignore region (orange). This box is marked as ignore region as in the background of the GT box there are two persons, which are very close to the actual labeled person. The two persons can be recognized best in the IR image. These additional persons can confuse the detector training. This figure is exemplary for the difficulty in providing consistent annotations. In the VIS image it looks like the most left person has a well aligned GT bounding box, whereas in the IR image the

GT box is definitely too large and the person is not centered. Such annotations result from registration errors between the VIS and IR images especially at the image border areas.

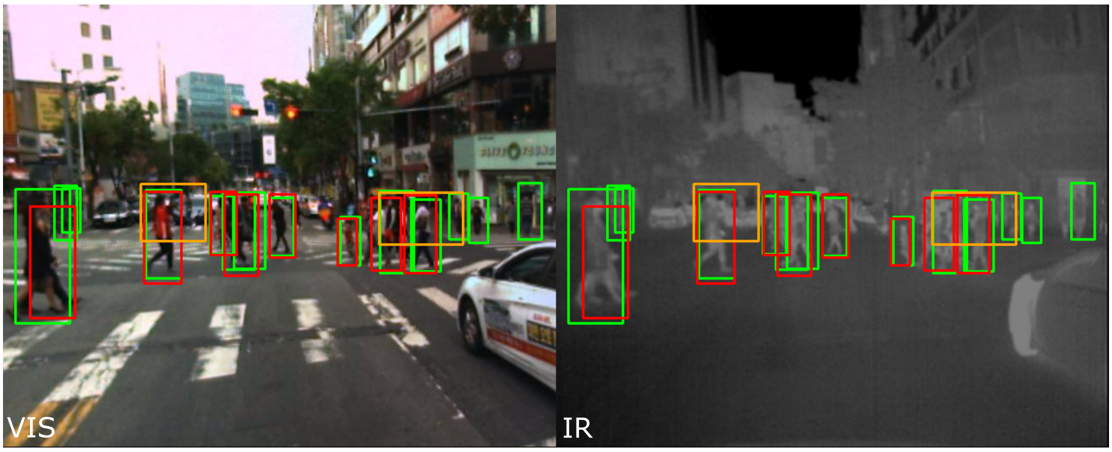


Figure 7.15: Visualization of the GT and detection boxes of *KAIST-test-Reasonable* testing subset for the Conv3 CVC Fusion RPN with BDF. GT boxes are colored in red, detection boxes in green, and ignore regions are marked by orange boxes.

Figure 7.15 shows frame 1,306. The detection boxes are well aligned with the GT boxes, especially the small person (child) in the image center. However, there are detections on the left and right side of the image with missing annotations. Considering the IR image, obviously the green boxes contain persons. For the evaluation, these boxes are regarded as FPs although they are actually TPs. On the left side there are two persons where one of them is heavily occluded. This bounding box should by labeled as ignore region for avoiding to confuse the training procedure.

Figure 7.16 provides an example for imprecise localization of the GT box. The annotated person is located in the upper right corner and thus covers only half of the GT box. This can be a problem for the evaluation process as the detection box is matched with the GT box w.r.t. the IoU overlap criterion. An even bigger problem is the impairment of the training procedure since background covers much of the bounding box area. Thus, it can happen that the detector learns parts of the background instead of the person.

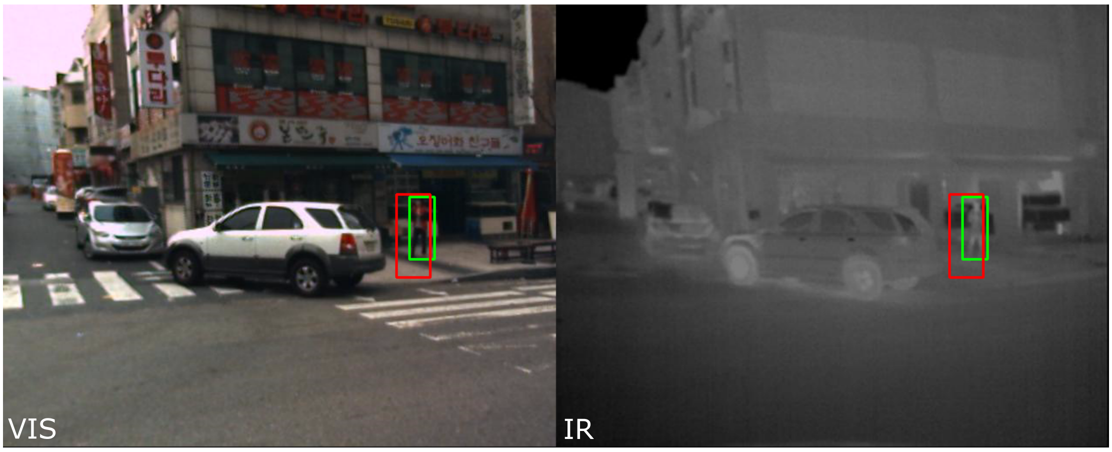


Figure 7.16: Visualization of the GT and detection boxes of *KAIST-test-Reasonable* testing subset for the Conv3 CVC Fusion RPN with BDF. GT boxes are colored in red, detection boxes in green, and ignore regions are marked by orange boxes.

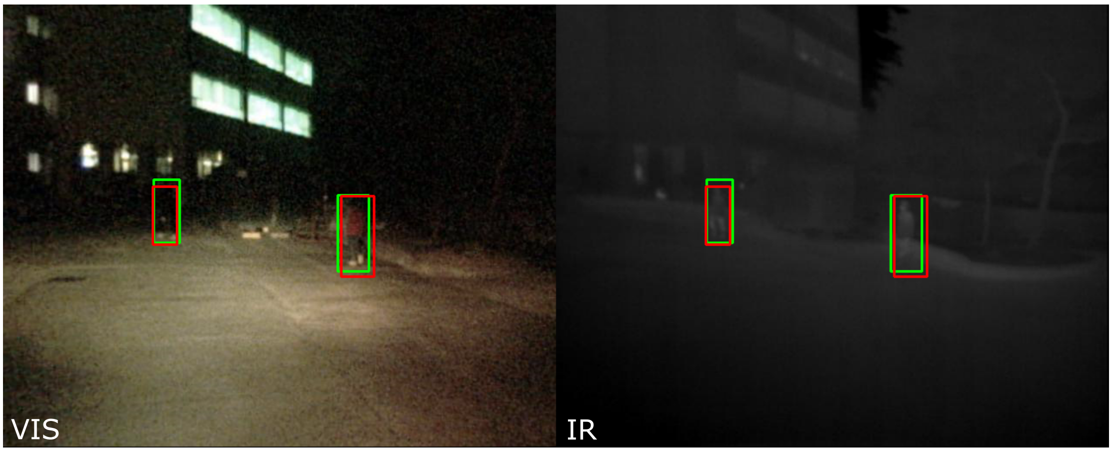


Figure 7.17: Visualization of the GT and detection boxes of *KAIST-test-Reasonable* testing subset for the Conv3 CVC Fusion RPN with BDF. GT boxes are colored in red, detection boxes in green, and ignore regions are marked by orange boxes.

Figure 7.17 provides an image pair acquired at nighttime. The person on the left of frame (1506) can be hardly recognized. Despite this rough conditions the night scene provides, the proposed detector is able to localize and classify the person correctly.

The goal of those visual example images is to visualize the challenges of the KAIST dataset. The dataset provides difficult conditions with various illumination changes, nighttime scenes, and annotated persons of different sizes including small-scale persons. Unfortunately, the annotations are not consistent throughout the dataset. There are missing annotations, missing ignore regions, imprecise localization of the annotations, and wrong annotations (e.g. bicyclists or statues should be labeled as ignore regions instead of as persons). This not only affects the RPN and classifier training, but also the evaluation leading to decreased TP rates or even increased FP and FN rates. In this way, the goal is not to diminish the author's great work behind the KAIST dataset. Instead, prospective authors should be encouraged to improve the annotations in the future.

8 Conclusion and Future Work

The goal of this thesis is to perform all-day person detection. Challenging detection scenarios result from heavy illumination changes in the acquired images, such as persons at night or in shady places. Previous works inspire the usage of multi-spectral images [WFHB16] [LZWM16]. Therefore, this work analyzes the complementary information of VIS and IR data and confirms the synergy effect achieved when using multi-spectral data. With parameter tuning of the ACF+T+THOG detector [HPK+15] and the introduced Checkerboards+T+THOG detector, the performance of current person detection approaches on the KAIST Multispectral Pedestrian Benchmark dataset is obtained. An RPN fusion approach is proposed to improve person detection on the KAIST dataset [HPK+15]. For training individual VIS and IR RPNs, auxiliary datasets are analyzed [WFHB16]. The Caltech Pedestrian Detection Benchmark dataset [DWSP12] and the CVC-09 FIR Sequence Pedestrian Dataset [SRV+11] are utilized for the so-called pre-finetuning of the VIS and IR RPNs. The Fusion RPN improves the MR by around 35 % compared to the ACF+T+THOG baseline of Hwang et al. and has similar results compared to the Faster R-CNN fusion approaches of Liu et al. [LZWM16]. The Fusion RPN outperforms the detectors based on filtered channel features, but using the classification CNN of the Faster R-CNN approach deteriorates the detection performance [ZLLH16]. Further enhancement is achieved by applying a BDF classifier to the proposals of the Fusion RPN and by using the feature maps of intermediate convolutional layers for classification. This method attains an additional improvement of the MR by 14 % compared to the Fusion RPN. Thus, a new baseline on the KAIST Multispectral Pedestrian Benchmark dataset is established with an MR of 29.83 %, using the Fusion RPN with a BDF.

As first further work, the usage of a better IR dataset than the CVC-09 dataset is proposed for pre-finetuning, as the CVC-09 dataset does not match with the characteristics of the Caltech and KAIST dataset. Additionally, other network architectures than the VGG-16 net should be analyzed, such as VGG-19 [SZ15] or ResNet [HZRS15a]. Furthermore, in-depth analysis of the RPN for person detection, especially for small-scale person detection is suggested. The anchor scales can be modified and their influence on the detection accuracy for different person heights should be evaluated. Adding additional anchor scales for detecting small-scale persons is assumed to improve the RPN's performance. Thus, evaluations of the RPN on the *KAIST-test-All* testing subset are aimed to reach similar results as for the *KAIST-test-Reasonable* subset. The approach of Cai et al. [CFFV16] proposes two separate RPN regression branches, one trained to find small-scale persons and the other is responsible for detecting large-scale persons. As well they propose to combine convolutional feature maps of intermediate layers with higher-level feature maps to improve small-scale person detection. Instead of using the BDF for classifying the region proposals, the usage of a small classification CNN is proposed similar to [HMWB16]. Additionally, visualization of the activations in different layers is suggested. This should provide a better understanding of the neurons of different layers and of the features in the input image these neurons respond to. The visualization of the fusion layer (NiN) would be useful to assess whether the VIS or IR features are higher weighted. Several further visualization methods are proposed by Zeiler and Fergus [ZF14].

Bibliography

[AAB⁺16] ABADI, Martín ; AGARWAL, Ashish ; BARHAM, Paul ; BREVDO, Eugene ; CHEN, Zhifeng ; CITRO, Craig ; CORRADO, Greg S. ; DAVIS, Andy ; DEAN, Jeffrey ; DEVIN, Matthieu ; OTHERS: TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. In: *arXiv:1603.04467v2* (2016), 19

[ADF12] ALEXE, Bogdan ; DESELAERS, Thomas ; FERRARI, Vittorio: Measuring the objectness of image windows. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 34 (2012), Nr. 11, S. 2189–2202

[AFD13] APPEL, Ron ; FUCHS, Thomas ; DOLLÁR, Piotr: Quickly Boosting Decision Trees - Pruning Underachieving Features Early -. In: *International Conference on Machine learning (ICML)* 28 (2013), S. 594–602

[AGM14] AGRAWAL, Pulkit ; GIRSHICK, Ross ; MALIK, Jitendra: Analyzing the performance of multilayer neural networks for object recognition. In: *European Conference on Computer Vision (ECCV)* 8695 LNCS (2014), Nr. PART 7, S. 329–344

[AM17] AFRAKHTEH, Masoud ; MIRYONG, Park: Pedestrian Detection with Minimal False Positives per Color-Thermal Image. In: *Arabian Journal for Science and Engineering* (2017), S. 1–13

[And13] ANDRADE, Anderson de: Best Practices for Convolutional Neural Networks Applied to Object Recognition in Images. In: *Neural Information Processing Systems (NIPS)* 1 (2013), S. 10

[BB05] BOURDEV, Lubomir ; BRANDT, Jonathan: Robust object detection via soft cascade. In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)* 2 (2005), S. 236–243

[BDTB08] BABENKO, Boris ; DOLLÁR, Piotr ; TU, Zhuowen ; BELONGIE, Serge: Simultaneous Learning and Alignment: Multi-Instance and Multi-Pose

Learning. In: *Workshop on Faces in 'Real-Life' Images: Detection, Alignment, and Recognition* (2008)

[BDX16] BUNEL, Rudy ; DAVOINE, Franck ; XU, Philippe: Detection of pedestrians at far distance. In: *International Conference on Robotics and Automation (ICRA)* 2016-June (2016), S. 2326–2331

[Ben12] BENGIO, Yoshua: Practical recommendations for gradient-based training of deep architectures. Version: 2012. In: *Neural Networks: Tricks of the Trade* Bd. 7700 LECTU. Springer, 2012. – ISBN 9783642352881, S. 437–478

[BLP+12] BASTIEN, Frédéric ; LAMBLIN, Pascal ; PASCANU, Razvan ; BERGSTRA, James ; GOODFELLOW, Ian ; BERGERON, Arnaud ; BOUCHARD, Nicolas ; WARDE-FARLEY, David ; BENGIO, Yoshua: Theano: new features and speed improvements. In: *arXiv:1211.5590* (2012), 1–10

[BMTV12] BENENSON, Rodrigo ; MATHIAS, Markus ; TIMOFTE, Radu ; VAN GOOL, Luc: Pedestrian detection at 100 frames per second. In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012, S. 2903–2910

[BMTV13] BENENSON, Rodrigo ; MATHIAS, Markus ; TUYTELAARS, Tinne ; VAN GOOL, Luc: Seeking the strongest rigid detector. In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013, S. 3666–3673

[BOHS14] BENENSON, Rodrigo ; OMRAN, Mohamed ; HOSANG, Jan ; SCHIELE, Bernt: Ten years of pedestrian detection, what have we learned? In: *European Conference on Computer Vision (ECCV)* Bd. 8926 Springer, 2014, 613–627

[BRC16] BAPPY, Jawadul H. ; ROY-CHOWDHURY, Amit K.: CNN based region proposals for efficient object detection. In: *International Conference on Image Processing (ICIP)* IEEE, 2016, S. 3658–3662

[Bre01] BREIMAN, Leo: Random forests. In: *Machine Learning* 45 (2001), Nr. 1, S. 5–32

[BVN14] BREHAR, Raluca ; VANCEA, Cristian ; NEDEVSCHI, Sergiu: Pedestrian detection in infrared images using Aggregated Channel Features. In: *Proceedings of the International Conference on Intelligent Computer Communication and Processing, (ICCP)* (2014), S. 127–132

[Byr08] BYRNES, James: *Unexploded ordnance detection and mitigation*. Springer Science & Business Media, 2008

[CFFV16] CAI, Zhaowei ; FAN, Quanfu ; FERIS, Rogerio S. ; VASCONCELOS, Nuno: A Unified Multi-scale Deep Convolutional Neural Network for Fast Object Detection. In: *European Conference on Computer Vision (ECCV)* Bd. 9905 Springer, 2016, 354–370

[CKF11] COLLOBERT, Ronan ; KAVUKCUOGLU, Koray ; FARABET, Clément: Torch7: A matlab-like environment for machine learning. In: *Neural Information Processing Systems (NIPS)* (2011), 1–6

[CKPS16] CHOI, Hangil ; KIM, Seungryong ; PARK, Kihong ; SOHN, Kwanghoon: Multi-spectral Pedestrian Detection Based on Accumulated Object Proposal with Fully Convolution Network. In: *International Conference on Pattern Recognition (ICPR)* (2016)

[CL11] CHANG, Chih-Chung ; LIN, Chih-Jen: LIBSVM: A library for support vector machines. In: *ACM Transactions on Intelligent Systems and Technology* 2 (2011), Nr. 3, S. 27:1—-27:27

[CLL+15] CHEN, Tianqi ; LI, Mu ; LI, Yutian ; LIN, Min ; WANG, Naiyan ; WANG, Minjie ; XIAO, Tianjun ; XU, Bing ; ZHANG, Chiyuan ; ZHANG, Zheng: MXNet: A Flexible and Efficient Machine Learning Library for Heterogeneous Distributed Systems. In: *Neural Information Processing Systems (NIPS), Workshop on Machine Learning Systems* (2015), 1–6

[CP10] CHOI, Eun J. ; PARK, Dong J.: Human detection using image fusion of thermal and visible image with new joint bilateral filter. In: *Proceedings of the International Conference on Computer Sciences and Convergence Information Technology, (ICCIT)* (2010), S. 882–885

[CPK+14] CHEN, Liang-Chieh ; PAPANDREOU, George ; KOKKINOS, Iasonas ; MURPHY, Kevin ; YUILLE, Alan L.: Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs. In: *International Conference on Learning Representations (ICLR)* (2014), 1–14

[CPL15] CAO, Jiale ; PANG, Yanwei ; LI, Xuelong: Pedestrian Detection Inspired by Appearance Constancy and Shape Symmetry. In: *arXiv:1511.08058* (2015)

[CSVZ14] CHATFIELD, Ken ; SIMONYAN, Karen ; VEDALDI, Andrea ; ZISSERMAN, Andrew: Return of the Devil in the Details: Delving Deep into Convolutional Nets. In: *British Machine Vision Conference (BMVC)*, 2014, 1–11

[CV95] CORTES, Corinna ; VAPNIK, Vladimir: Support-Vector Networks. In: *Machine Learning* 20 (1995), Nr. 3, S. 273–297

[CWK+14] CHEN, Xiaogang ; WEI, Pengxu ; KE, Wei ; YE, Qixiang ; JIAO, Jianbin: Pedestrian detection with deep convolutional neural network. In: *Asian Conference on Computer Vision (ACCV)* Springer, 2014, 354–365

[DABP14] DOLLÁR, Piotr ; APPEL, Ron ; BELONGIE, Serge J. ; PERONA, Pietro: Fast Feature Pyramids for Object Detection. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 36 (2014), Nr. 8, 1532–1545

[DAK12] DOLLÁR, Piotr ; APPEL, Ron ; KIENZLE, Wolf: Crosstalk cascades for frame-rate pedestrian detection. In: *European Conference on Computer Vision (ECCV)* 7573 LNCS (2012), Nr. PART 2, S. 645–659

[DBP10] DOLLÁR, Piotr ; BELONGIE, Serge J. ; PERONA, Pietro: The Fastest Pedestrian Detector in the West. In: *British Machine Vision Conference (BMVC)*, 2010, 68.1–68.11

[DDS+09] DENG, Jia Deng J. ; DONG, Wei Dong W. ; SOCHER, R. ; LI, Li-Jia Li Li-Jia ; LI, Kai Li K. ; FEI-FEI, Li Fei-Fei L.: ImageNet: A large-scale hierarchical image database. In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)* (2009), S. 2–9

[DG06] DAVIS, Jesse ; GOADRICH, Mark: The Relationship Between Precision-Recall and ROC Curves. In: *Proceedings of the International Conference on Machine learning (ICML)* (2006), 233–240

[DJV+14] DONAHUE, Jeff ; JIA, Yangqing ; VINYALS, Oriol ; HOFFMAN, Judy ; ZHANG, Ning ; TZENG, Eric ; DARRELL, Trevor: DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition. In: *International Conference on Machine learning (ICML)* 32 (2014), 647–655

[DK07] DAVIS, James W. ; KECK, Mark A.: A two-stage template approach to person detection in thermal imagery. In: *Workshop on Applications of Computer Vision (WACV)* 5 (2007), 364–369

[DN16] DANIEL COSTEA, Arthur ; NEDEVSCHI, Sergiu: Semantic Channels for Fast Pedestrian Detection. In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, S. 2360–2368

[Dol] DOLLÁR, Piotr: *Piotr's Computer Vision Matlab Toolbox (PMT)*. https://github.com/pdollar/toolbox,

[DT05] DALAL, Navneet ; TRIGGS, Bill: Histograms of oriented gradients for human detection. In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)* I (2005), 886–893

[DTPB09] DOLLÁR, Piotr ; TU, Zhuowen ; PERONA, Pietro ; BELONGIE, Serge: Integral Channel Features. In: *British Machine Vision Conference (BMVC)* (2009), 1–11

[DWSP09] DOLLÁR, Piotr ; WOJEK, Christian ; SCHIELE, Bernt ; PERONA, Pietro: Pedestrian detection: A benchmark. In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009, S. 304–311

[DWSP12] DOLLÁR, Piotr ; WOJEK, Christian ; SCHIELE, Bernt ; PERONA, Pietro: Pedestrian detection: An evaluation of the state of the art. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 34 (2012), Nr. 4, S. 743–761

[EG09] ENZWEILER, Markus ; GAVRILA, Dariu M.: Monocular pedestrian detection: Survey and experiments. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 31 (2009), Nr. 12, S. 2179–2195

[EH10] ENDRES, Ian ; HOIEM, Derek: Category Independent Object Proposals. In: *European Conference on Computer Vision (ECCV)* (2010), 575–588

[ELSa08] ESS, A ; LEIBE, B ; SCHINDLER, K ; AND L. VAN GOOL: A Mobile Vision System for Robust Multi-Person Tracking. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE Press, 2008, S. 1–8

[Enz11] ENZWEILER, Markus: Compound Models for Vision-Based Pedestrian Recognition. In: *Dissertation* phd thesis (2011), S. 1–192

[ESTA14] ERHAN, Dumitru ; SZEGEDY, Christian ; TOSHEV, Alexander ; ANGUELOV, Dragomir: Scalable Object Detection Using Deep Neural Networks. In: *Conference on Computer Vision and Pattern Recognition (CVPR)* (2014), 2155–2162

[EVW+10] EVERINGHAM, Mark ; VAN GOOL, Luc ; WILLIAMS, Christopher K I. ; WINN, John ; ZISSERMAN, Andrew: The pascal visual object classes

(VOC) challenge. In: *International Journal of Computer Vision (IJCV)* 88 (2010), Nr. 2, S. 303–338

[FGMR10] Felzenszwalb, Pedro F. ; Girshick, Ross B. ; McAllester, David ; Ramanan, Deva: Object detection with discriminatively trained part-based models. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 32 (2010), Nr. 9, S. 1627–1645

[FHTO00] Friedman, Jerome ; Hastie, Trevor ; Tibshirani, Robert ; Others: Additive logistic regression: a statistical view of boosting. In: *The annals of statistics* 28 (2000), Nr. 2, S. 337–407

[FS95] Freund, Yoav ; Schapire, Robert E.: A desicion-theoretic generalization of on-line learning and an application to boosting. In: *European Conference on Computational Learning Theory (EuroCOLT)* Bd. 55 Springer, 1995, 119–139

[FS96] Freund, Yoav ; Schapire, Re Robert E.: Experiments with a New Boosting Algorithm. In: *International Conference on Machine learning (ICML)* (1996), 148–156

[FS99] Freund, Yoav ; Schapire, Robert E.: A short introduction to boosting. In: *International Joint Conference on Artificial Intelligence (IJCAI)* 2 (1999), Nr. 5, S. 1401–1406

[FYN+03] Fang, Yajun ; Yamada, Keiichi ; Ninomiya, Yoshiki ; Horn, Berthold ; Masaki, Ichiro: Comparison between infrared-image-based and visible-image-based approaches for pedestrian detection. In: *Proceedings of the Intelligent Vehicles Symposium* IEEE, 2003, S. 505–510

[GB10] Glorot, Xavier ; Bengio, Yoshua: Understanding the difficulty of training deep feedforward neural networks. In: *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)* 9 (2010), 249–256

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

[GDP+15] Ghodrati, Amir ; Diba, Ali ; Pedersoli, Marco ; Tuytelaars, Tinne ; Van Gool, Luc: DeepProposal: Hunting Objects by Cascading Deep Convolutional Layers. In: *Proceedings of the International Conference on Computer Vision Systems (ICVS)*, 2015, 2578–2586

[GFS$^+$16] GONZÁLEZ, Alejandro ; FANG, Zhijie ; SOCARRAS, Yainuvis ; SERRAT, Joan ; VÁZQUEZ, David ; XU, Jiaolong ; LÓPEZ, Antonio M.: Pedestrian Detection at Day/Night Time with Visible and FIR Cameras: A Comparison. In: *Sensors* 16 (2016), Nr. 6, S. 1–11

[Gir16] GIRSHICK, Ross: Fast R-CNN. In: *Proceedings of the International Conference on Computer Vision Systems (ICVS)* 11-18-Dece (2016), 1440–1448

[GLU12] GEIGER, Andreas ; LENZ, Philip ; URTASUN, Raquel: Are we ready for autonomous driving? the KITTI vision benchmark suite. In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012, S. 3354–3361

[GSLP07] GERÓNIMO, David ; SAPPA, A.D. ; LÓPEZ, Antonio ; PONSA, Daniel: Adaptive image sampling and windows classification for on-board pedestrian detection. In: *Proceedings of the International Conference on Computer Vision Systems (ICCVS)* 39 (2007), Nr. Icvs

[GSPL10] GERÓNIMO, David ; SAPPA, Angel D. ; PONSA, Daniel ; LÓPEZ, Antonio M.: 2D–3D-based on-board pedestrian detection system. In: *Computer Vision and Image Understanding* 114 (2010), Nr. 5, S. 583–595

[GWF13] GOODFELLOW, Ij ; WARDE-FARLEY, D: Pylearn2: a machine learning research library. In: *arXiv:1308.4214* 18 (2013), Nr. 3, 1–9

[HBDS16] HOSANG, Jan ; BENENSON, Rodrigo ; DOLLAR, Piotr ; SCHIELE, Bernt: What Makes for Effective Detection Proposals? In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 38 (2016), Nr. 4, 814–830

[HCD12] HOIEM, Derek ; CHODPATHUMWAN, Yodsawalai ; DAI, Qieyun: Diagnosing error in object detectors. In: *European Conference on Computer Vision (ECCV)* Bd. 7574 LNCS Springer, 2012, S. 340–353

[HMWB16] HERRMANN, Christian ; MÜLLER, Thomas ; WILLERSINN, Dieter ; BEYERER, Jürgen: Real-time person detection in low-resolution thermal infrared imagery with MSER and CNNs. In: *SPIE Security+ Defence* International Society for Optics and Photonics, 2016, S. 99870I—-99870I

[HOBS15] HOSANG, Jan ; OMRAN, Mohamed ; BENENSON, Rodrigo ; SCHIELE, Bernt: Taking a deeper look at pedestrians. In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)* Bd. 07-12-June, 2015, S. 4073–4082

[HPK+15] Hwang, Soonmin ; Park, Jaesik ; Kim, Namil ; Choi, Yukyung ; Kweon, In S.: Multispectral pedestrian detection: Benchmark dataset and baseline. In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)* Bd. 07-12-June, 2015, S. 1037–1045

[HSK+12] Hinton, Geoffrey E. ; Srivastava, Nitish ; Krizhevsky, Alex ; Sutskever, Ilya ; Salakhutdinov, Ruslan R.: Improving neural networks by preventing co-adaptation of feature detectors. In: *arXiv:1207.0580* (2012), 1–18

[HWS+16] Hu, Qichang ; Wang, Peng ; Shen, Chunhua ; Hengel, Anton van d. ; Porikli, Fatih: Pushing the Limits of Deep CNNs for Pedestrian Detection. In: *arXiv:1603.04525* (2016), 1–15

[HZRS15a] He, Kaiming ; Zhang, Xiangyu ; Ren, Shaoqing ; Sun, Jian: Deep Residual Learning for Image Recognition. In: *arXiv:1512.03385* 7 (2015), Nr. 3, 171–180

[HZRS15b] He, Kaiming ; Zhang, Xiangyu ; Ren, Shaoqing ; Sun, Jian: Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 37 (2015), Nr. 9, S. 1904–1916

[JDM00] Jain, Ak ; Duin, Rpw ; Mao, J: Statistical pattern recognition: A review. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 22 (2000), Nr. 1, 4–37

[JSD+14] Jia, Yangqing ; Shelhamer, Evan ; Donahue, Jeff ; Karayev, Sergey ; Long, Jonathan ; Girshick, Ross ; Guadarrama, Sergio ; Darrell, Trevor: Caffe: Convolutional architecture for fast feature embedding. In: *Proceedings of the International Conference on Multimedia (ACM)* ACM, 2014, 675–678

[Kri09] Krizhevsky, Alex: Learning Multiple Layers of Features from Tiny Images. In: *Technical Report, Science Department, University of Toronto* (2009), 1–60

[Kri14] Krizhevsky, Alex: *cuda-convnet: High-performance C++/CUDA implementation of convolutional neural networks*. Version: 2014

[KSH12] Krizhevsky, Alex ; Sutskever, Ilya ; Hinton, Geoffrey E.: ImageNet Classification with Deep Convolutional Neural Networks. In: *Advances in Neural Information Processing Systems (NIPS)* (2012), S. 1–9

[LBD+89] LeCun, Y. ; Boser, B. ; Denker, J. S. ; Henderson, D. ; Howard, R. E. ; Hubbard, W. ; Jackel, L. D.: *Backpropagation Applied to Handwritten Zip Code Recognition*

[LC15] Lin, Bo-Yao ; Chen, Chu-Song: Two Parallel Deep Convolutional Neural Networks for Pedestrian Detection. In: *Image and Vision Computing New Zealand (IVCNZ)* (2015)

[LCY13] Lin, Min ; Chen, Qiang ; Yan, Shuicheng: Network In Network. In: *arXiv:1312.4400* (2013), 10

[LLS+16] Li, Jianan ; Liang, Xiaodan ; Shen, ShengMei ; Xu, Tingfa ; Feng, Jiashi ; Yan, Shuicheng: Scale-aware Fast R-CNN for Pedestrian Detection. In: *arXiv:1510.08160* (2016), 1–10

[LMB+14] Lin, Tsung Y. ; Maire, Michael ; Belongie, Serge ; Hays, James ; Perona, Pietro ; Ramanan, Deva ; Dollár, Piotr ; Zitnick, C. L.: Microsoft COCO: Common objects in context. In: *European Conference on Computer Vision (ECCV)* 8693 LNCS (2014), Nr. PART 5, S. 740–755

[LRB15] Liu, Wei ; Rabinovich, Andrew ; Berg, Alexander C.: ParseNet: Looking Wider to See Better. In: *arXiv:1506.04579* (2015), 1–11

[LRH07] Leykin, Alex ; Ran, Yang ; Hammoud, Riad: Thermal-visible video fusion for moving target tracking and pedestrian classification. In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007, S. 1–8

[LSD15] Long, Jonathan ; Shelhamer, Evan ; Darrell, Trevor: Fully Convolutional Networks for Semantic Segmentation ppt. In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), S. 3431–3440

[LTCFS16] Leal-Taixé, Laura ; Canton-Ferrer, Cristian ; Schindler, Konrad: Learning by tracking: Siamese CNN for robust target association. In: *arXiv:1604.07866* (2016)

[LTWT14] Luo, Ping ; Tian, Yonglong ; Wang, Xiaogang ; Tang, Xiaoou: Switchable deep network for pedestrian detection. In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, S. 899–906

[LZL] LU, Zongqing ; ZHANG, Wenjian ; LIAO, Qingmin: Pedestrian Detection Aided by Scale-Discriminative Network.

[LZWM16] LIU, Jingjing ; ZHANG, Shaoting ; WANG, Shu ; METAXAS, Dimitris N.: Multispectral Deep Neural Networks for Pedestrian Detection. In: *British Machine Vision Conference (BMVC)*, 2016, S. 1–13

[MG06] MUNDER, S. ; GAVRILA, D. M.: An experimental study on pedestrian classification. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 28 (2006), Nr. 11, S. 1863–1868

[MP08] MIEZIANKO, Roland ; POKRAJAC, Dragoljub: People detection in low resolution infrared videos. In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)* (2008), 1–6

[Myr13] MYRTUE, Niels G.: Visual Detection of Humans in a Disaster Scenario. In: *Aalborg University* (2013)

[NDH14] NAM, Woonhyun ; DOLLÁR, Piotr ; HAN, Joon H.: Local Decorrelation For Improved Pedestrian Detection. In: *Neural Information Processing Systems (NIPS)*, 2014, 1–9

[NRD16] NAJIBI, Mahyar ; RASTEGARI, Mohammad ; DAVIS, Larry S.: G-CNN: an Iterative Grid Based Object Detector. In: *Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), 2369–2377

[OBT16] OHN-BAR, Eshed ; TRIVEDI, Mohan M.: To Boost or Not to Boost? On the Limits of Boosted Trees for Object Detection. In: *International Conference on Pattern Recognition (ICPR)* abs/1701.0 (2016)

[ÓCO+05] Ó CONAIRE, Ciarán ; COOKE, Eddie ; O'CONNOR, Noel E. ; MURPHY, Noel ; SMEATON, Alan F.: Fusion of infrared and visible spectrum video for indoor surveillance. (2005)

[OFG+97] OSUNA, E ; FREUND, R ; GIROSI, F ; E.˜OSUNA ; R.˜FREUND ; F.˜GIROSI: Training Support Vector Machines: An Application to Face Detection. In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)* (1997), S. 130–136

[OPN+13] OLMEDA, Daniel ; PREMEBIDA, Cristiano ; NUNES, Urbano ; ARMINGOL, José M ; ESCALERA, Arturo de l.: LSI far infrared pedestrian dataset. (2013)

[OW12] OUYANG, Wanli ; WANG, Xiaogang: A discriminative deep model for pedestrian detection with occlusion handling. In: *Proceedings of* *the Conference on Computer Vision and Pattern Recognition (CVPR)* (2012), S. 3258–3265

[OW13] OUYANG, Wanli ; WANG, Xiaogang: Joint deep learning for pedestrian detection. In: *Proceedings of the International Conference on Computer Vision (ICCV)* (2013), S. 2056–2063

[PBE+06] PONCE, Jean ; BERG, T. ; EVERINGHAM, Mark ; FORSYTH, D. ; HEBERT, Martial ; LAZEBNIK, S. ; MARSZALEK, Marcin ; SCHMID, Cordelia ; RUSSELL, B. ; TORRALBA, A. ; WILLIAMS, Christopher ; ZHANG, Jianguo ; ZISSERMAN, Andrew: *Dataset issues in object recognition*. 2006. – 29–48 S. – ISBN 978–3–540–68794–8

[PLCS14] PORTMANN, Jan ; LYNEN, Simon ; CHLI, Margarita ; SIEGWART, Roland: People detection and tracking from aerial thermal views. In: *International Conference on Robotics and Automation (ICRA)* IEEE, 2014, S. 1794–1800

[PP00] PAPAGEORGIOU, Constantine ; POGGIO, Tomaso: A Trainable system for object detection. In: *International Journal of Computer Vision (IJCV)* 38 (2000), Nr. 1, S. 15–33

[Qui86] QUINLAN, J. R.: Induction of Decision Trees. In: *Machine Learning* 1 (1986), Nr. 1, S. 81–106

[RDGF15] REDMON, Joseph ; DIVVALA, Santosh ; GIRSHICK, Ross ; FARHADI, Ali: You only look once: Unified, real-time object detection. In: *Conference on Computer Vision and Pattern Recognition (CVPR)* (2015)

[RDS+15] RUSSAKOVSKY, Olga ; DENG, Jia ; SU, Hao ; KRAUSE, Jonathan ; SATHEESH, Sanjeev ; MA, Sean ; HUANG, Zhiheng ; KARPATHY, Andrej ; KHOSLA, Aditya ; BERNSTEIN, Michael ; BERG, Alexander C. ; FEI-FEI, Li: ImageNet Large Scale Visual Recognition Challenge. In: *International Journal of Computer Vision (IJCV)* 115 (2015), Nr. 3, S. 211–252

[Red13] REDMON, Joseph: *Darknet: Open Source Neural Networks in C*. http://pjreddie.com/darknet/, 2013

[RHGS16] REN, S ; HE, K ; GIRSHICK, R ; SUN, J: Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* Bd. PP, 2016, S. 91–99

[ROBT16] Rajaram, R N. ; Ohn-Bar, E ; Trivedi, M M.: Looking at Pedestrians at Different Scales: A Multiresolution Approach and Evaluations. In: *IEEE Transactions on Intelligent Transportation Systems* 17 (2016), Nr. 12, 1–12

[SEZ+13] Sermanet, Pierre ; Eigen, David ; Zhang, Xiang ; Mathieu, Michael ; Fergus, Rob ; LeCun, Yann: OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks. In: *arXiv:1312.6229* (2013), 1312.6229

[Shr16] Shrivastava, Abhinav: Training Region-based Object Detectors with Online Hard Example Mining. In: *Conference on Computer Vision and Pattern Recognition (CVPR)* (2016)

[Shu14] Shu, Guang: *Human Detection, Tracking and Segmentation in Surveillance Video*, Citeseer, Diss., 2014. – 81–87 S

[SL11] Sermanet, Pierre ; LeCun, Yann: Traffic sign recognition with multiscale Convolutional Networks. In: *International Joint Conference on Neural Networks (IJCNN)* (2011), 2809–2813

[SLJ+15] Szegedy, Christian ; Liu, Wei ; Jia, Yangqing ; Sermanet, Pierre ; Reed, Scott ; Anguelov, Dragomir ; Erhan, Dumitru ; Vanhoucke, Vincent ; Rabinovich, Andrew: Going deeper with convolutions. In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, 1–9

[SIMP07] St-laurent, Louis ; Maldague, Xavier ; Prévost, Donald: Combination of colour and thermal sensors for enhanced object detection. In: *International Conference on Information Fusion (ISIF)* IEEE, 2007, S. 1–8

[SRBB06] Suard, Frédéric ; Rakotomamonjy, Alain ; Bensrhair, Aziz ; Broggi, Alberto: Pedestrian Detection using Infrared images and Histograms of Oriented Gradients. In: *IEEE Intelligent Vehicles Symposium* (2006), 206–212

[SRE+14] Szegedy, Christian ; Reed, Scott ; Erhan, Dumitru ; Anguelov, Dragomir ; Ioffe, Sergey: Scalable, high-quality object detection. In: *arXiv:1412.1441* (2014)

[SRV+11] Socarrás, Yainuvis ; Ramos, Sebastian ; Vázquez, David ; López, Antonio M. ; Gevers, Theo: Adapting Pedestrian Detection from Synthetic to Far Infrared Images. In: *Proceedings of the International*

Conference on Computer Vision (ICCV), Workshop on Visual Domain Adaptation and Dataset Bias Bd. 7, 2011, S. 2–4

[STE13] SZEGEDY, Christian ; TOSHEV, Alexander ; ERHAN, Dumitru: Deep neural networks for object detection. In: *Advances in Neural Information Processing Systems (NIPS)*, 2013, 2553–2561

[SZ15] SIMONYAN, Karen ; ZISSERMAN, Andrew: Very Deep Convolutional Networks for Large-Scale Image Recognition. In: *International Conference on Learning Representations (ICLR)* (2015), 1–14

[TE11] TORRALBA, Antonio ; EFROS, Alexei A.: Unbiased look at dataset bias. In: *Conference on Computer Vision and Pattern Recognition (CVPR)* IEEE, 2011, S. 1521–1528

[TLWT15] TIAN, Yonglong ; LUO, Ping ; WANG, Xiaogang ; TANG, Xiaoou: Pedestrian detection aided by deep learning semantic tasks. In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)* Bd. 07-12-June, 2015, 5079–5087

[TLWT16] TIAN, Yonglong ; LUO, Ping ; WANG, Xiaogang ; TANG, Xiaoou: Deep learning strong parts for pedestrian detection. In: *Proceedings of the International Conference on Computer Vision (ICCV)* 11-18-Dece (2016), S. 1904–1912

[TMB$^+$16] TOMÈ, Denis ; MONTI, Federico ; BAROFFIO, Luca ; BONDI, Luca ; TAGLIASACCHI, Marco ; TUBARO, Stefano: Deep convolutional neural networks for pedestrian detection. In: *Signal Processing: Image Communication* (2016)

[TMHB14] TEUTSCH, Michael ; MUELLER, Thomas ; HUBER, Marco ; BEYERER, Juergen: Low resolution person detection with a moving thermal infrared camera by hot spot classification. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, S. 209–216

[Tor04] TORRESAN, Helene: Advanced surveillance systems: combining video and thermal imagery for pedestrian detection. In: *Proceedings of SPIE* (2004), 506–515

[TS09] TORREY, Lisa ; SHAVLIK, Jude: Transfer Learning. In: *Machine Learning* (2009), S. 1–22

[UVGS13] UIJLINGS, J. R R. ; VAN DE SANDE, K. E A. ; GEVERS, T. ; SMEULDERS,

A. W M.: Selective Search for Object Recognition. In: *International Journal of Computer Vision (IJCV)* 104 (2013), Nr. 2, S. 154–171

[VHV+16] VERMA, Ankit ; HEBBALAGUPPE, Ramya ; VIG, Lovekesh ; KUMAR, Swagat ; HASSAN, Ehtesham: Pedestrian Detection via Mixture of CNN Experts and Thresholded Aggregated Channel Features. In: *Proceedings of the International Conference on Computer Vision (ICCV)* Bd. 2016-Febru, 2016, S. 555–563

[VJ01a] VIOLA, P ; JONES, M: Rapid object detection using a boosted cascade of simple features. In: *Conference on Computer Vision and Pattern Recognition (CVPR)* 1 (2001), S. 511–518

[VJ01b] VIOLA, Paul ; JONES, Michael J.: Robust Real-time Object Detection. In: *International Journal of Computer Vision (IJCV)* (2001), Nr. February, S. 1–30

[VL15] VEDALDI, A ; LENC, K: MatConvNet - Convolutional Neural Networks for MATLAB. In: *Proceedings of the International Conference on Multimedia (ACM)*, 2015, S. 689–692

[VV98] VAPNIK, Vladimir N. ; VAPNIK, Vlamimir: *Statistical learning theory*. Bd. 1. Wiley New York, 1998

[WFHB16] WAGNER, Jörg ; FISCHER, Volker ; HERMAN, Michael ; BEHNKE, Sven: Multispectral Pedestrian Detection using Deep Fusion Convolutional Neural Networks. In: *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)* (2016), Nr. April

[WFTB14] WU, Zheng ; FULLER, Nathan ; THERIAULT, Diane ; BETKE, Margrit: A thermal infrared video benchmark for visual analysis. In: *Conference on Computer Vision and Pattern Recognition (CVPR)* (2014), S. 201–208

[WMSS10] WALK, Stefan ; MAJER, Nikodem ; SCHINDLER, Konrad ; SCHIELE, Bernt: New features and insights for pedestrian detection. In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)* IEEE, 2010, S. 1030–1037

[WN05] WU, Bo ; NEVATIA, Ramakant: Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors. In: *International Conference on Computer Vision (ICCV)* Bd. 1 IEEE, 2005, S. 90–97

[WWS09] WOJEK, Christian ; WALK, Stefan ; SCHIELE, Bernt: Multi-cue onboard pedestrian detection. In: *Conference on Computer Vision and Pattern Recognition (CVPR)* IEEE, 2009, S. 794–801

[WYL+15] WANG, Shuo ; YANG, Bin ; LEI, Zhen ; WAN, Jun ; LI, Stan Z.: A convolutional neural network combined with aggregate channel feature for face detection. In: *International Conference on Wireless, Mobile and Multi-Media (ICWMMN)* IET, 2015, 304–308

[YK16] YU, Fisher ; KOLTUN, Vladlen: Multi-Scale Context Aggregation by Dilated Convolutions. In: *International Conference on Learning Representations (ICLR)* (2016), 1–9

[YYB+16] YU, Wei ; YANG, Kuiyuan ; BAI, Yalong ; XIAO, Tianjun ; YAO, Hongxun ; RUI, Yong: Visualizing and Comparing AlexNet and VGG using Deconvolutional Layers. In: *International Conference on Machine learning (ICML)* 48 (2016)

[YYLL16] YANG, Bin ; YAN, Junjie ; LEI, Zhen ; LI, Stan Z.: Convolutional channel features. In: *Proceedings of the International Conference on Computer Vision (ICCV)* 11-18-Dece (2016), S. 82–90

[ZBO+16] ZHANG, Shanshan ; BENENSON, Rodrigo ; OMRAN, Mohamed ; HOSANG, Jan ; SCHIELE, Bernt: How Far are We from Solving Pedestrian Detection? In: *Conference on Computer Vision and Pattern Recognition (CVPR)* (2016)

[ZBS15] ZHANG, Shanshan ; BENENSON, Rodrigo ; SCHIELE, Bernt: Filtered channel features for pedestrian detection. In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)* Bd. 07-12-June, 2015, S. 1751–1760

[ZBS17] ZHANG, Shanshan ; BENENSON, Rodrigo ; SCHIELE, Bernt: CityPersons: A Diverse Dataset for Pedestrian Detection. In: *arXiv:1702.05693* (2017)

[ZCST17] ZHANG, Xingguo ; CHEN, Guoyue ; SARUTA, Kazuki ; TERATA, Yuki: Deep Convolutional Neural Networks for All-Day Pedestrian Detection. In: *International Conference on Information Science and Applications* Springer, 2017, S. 171–178

[ZD14] ZITNICK, C. L. ; DOLLÁR, Piotr: Edge boxes: Locating object proposals from edges. In: *European Conference on Computer Vision (ECCV)* 8693 LNCS (2014), Nr. PART 5, S. 391–405

[ZF14] ZEILER, Matthew D. ; FERGUS, Rob: Visualizing and Understanding Convolutional Networks. In: *European Conference on Computer Vision (ECCV)* 8689 (2014), 818–833

[ZLLH16] ZHANG, Liliang ; LIN, Liang ; LIANG, Xiaodan ; HE, Kaiming: Is Faster R-CNN Doing Well for Pedestrian Detection? In: *European Conference on Computer Vision (ECCV)* (2016), S. 1–15

[ZV08] ZHANG, Cha ; VIOLA, Paul: Multiple-instance pruning for learning efficient cascade detectors. In: *Advances in Neural Information Processing Systems (NIPS)* (2008), 1–8

[ZWN07] ZHANG, Li ; WU, Bo ; NEVATIA, R.: Pedestrian Detection in Infrared Images based on Local Shape Features. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007, 0–7

Name: Daniel König Matriculation Number: 858777

Declaration of Authorship

I hereby declare that the thesis submitted is my own unaided work. All direct or indirect sources used are acknowledged as references.

Oberkochen, ...

Daniel König