Including traffic light recognition in general object detection with YOLOv2

Master thesis

Evert Bos

supervised by Julian Kooij
assisted by Ewoud Pool

TU Delft

Including traffic light recognition in general object detection with YOLOv2

Master thesis

by

Evert Bos

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Wednesday June 19, 2019 at 3.30 PM.

Student number:	4218566	
Project duration:	November 13, 2017 – June 19, 2019	
Thesis committee:	Assistant Professor F. P. Kooij,	TU Delft, supervisor
	Ir. E. A. I. Pool,	TU Delft, daily supervisor
	Professor M. Gavrila,	TU Delft, head examining committee
	Dr. Ing. J. Kober,	TU Delft

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft

Abstract

With an in vehicle camera many different things can be done that are essential for ADAS or autonomous driving mode in a vehicle. First, it can be used for detection of general objects, for example cars, cyclists or pedestrians. Secondly, the camera can be used for traffic light recognition, which is localization of traffic light position and traffic light state recognition. No method exists at the moment able to perform general object detection and traffic light recognition at the same time, therefore this work proposes methods to combine general object detection and traffic light recognition. The novel method presented is including traffic light recognition in a general object detection framework. The single shot object detector YOLOv2 is used as base detector. As general object class dataset COCO is used and the traffic light dataset is LISA. Two different methods for combined detection are proposed: adaptive combined training and YOLOv2++. For combined training YOLOv2 is trained on both datasets with the YOLOv2 network unchanged and the loss function adapted to optimize training on both datasets. For YOLOv2++ the feature extractor of YOLOv2 pre-trained on COCO is used as feature extractor. On the features LISA traffic light states are trained with a small sub-network. It is concluded the best performing method is adaptive combined training which reaches for IOU 0.5 a AUC of 24.02% for binary and 21.23% for multi-class classification. For IOU of 0.1 this increases to 56.74% for binary and 41.87% for multi-class classification. The performance of the adaptive combined detector is 20% lower than the baseline performance of an detector only detecting LISA traffic light states and 5% lower than the baseline of a detector only detecting COCO classes, however detection of classes from both dataset is almost twice as fast as separate detection with different networks for both datasets.

Contents

1 Introduction

Traffic lights plays an important role in current traffic control, as they are vital for regulating traffic at many intersections and other parts of infrastructure. In a large city like London over 6000 traffic lights are in use [89] and still a lot of accidents happen at traffic light controlled intersections. Over half a million crashes happened in a year at intersections controlled by a traffic light in the US [63], out of which half are related to a human recognition error [15]. A traffic light recognition system as part of a automated driving mode or driver assistance system can assist to prevent these errors.

Different methods are available for traffic light recognition. First, vehicle to infrastructure communication, for example with a method like ANET (Vehicular Ad Hoc NETworks)[23] is possible. While promising these kind of solutions are still being investigated and not yet implemented in the current infrastructure. Furthermore, vehicle to infrastructure communication demands communication devices in both traffic lights and vehicles to communicate with each other, making it costly and labor intensive to implement. On top of this, the standard to use for vehicle to infrastructure is still to be determined [3]. Therefore, the second method worth investigating is traffic light recognition with a specialized detection system on board of the vehicle, able to recognize the in place humanly detectable traffic lights. Over the last years these on board traffic light recognition systems have been an active research area [38, 55, 58]. Good performing detectors combine visual information with detailed maps [38]. However, implementing and maintaining a traffic light position map is labor intensive and costly. A traffic light recognition system based on sensor input in the ego vehicle is therefore preferred.

Up till now, all computer vision based methods for traffic light recognition are specifically designed for only this task, making them able to only detect traffic lights in a scene. However, a self driving vehicle or ADAS system can benefit from more information out of the scene, like other vehicles, pedestrians, cyclists or traffic signs.

This work investigates different methods to combine subclass detection with general object class detection for a traffic light subclass dataset with a general object class dataset using a modern single shot object detector based on a convolutional neural network. The general object classes consist of 80 different classes including classes relevant for traffic scenes, like cars or pedestrians, see figure 1.1a. The subclass dataset purely consists of 7 traffic light states, see figure 1.1b. At the moment no dataset exist containing general object classes and traffic light sub-classes in one set. The detector used is YOLOv2 [68], a single shot object detector with an excelent speed performance trade-off. The advantages of combined detection of classes from both dataset types is the single detector requires less computational power and time then multiple detectors. In a self driving vehicle this is important because computer power is limited and fast detection of all objects is essential for the self driving performance, and thus the safety, of the vehicle.

Subclass detection combined with general object class detection can be applied to more situations than traffic lights. Other examples could be pedestrian or cyclist poses, vehicle classes or traffic sign information. The methods described in this work can therefore be generalized to subclass detection within general object class detection in the general case.

(a) Samples images out of general object database. Most annotated object are relative large compared to the image size and the lighting conditions are good.

(b) Sample images out of traffic light database. The annotated objects are relatively small and the lighting conditions are challenging.

Figure 1.1: samples of images from general object class database and a traffic light database. The general object class database has no traffic light state annotations, the traffic light database has no general object class annotations.

Making a combined detector for general object classes and traffic lights is challenging for a couple of reasons. Figure 1.1a and 1.1b show the difficulty of detecting the traffic light objects due to the small size of the traffic light objects compared to other objects and difficult lighting conditions. Figure 1.4 shows the difficulty of combining the detectors. In the images in 1.2 a detector trained on general object classes is used, in the images in 1.3 a simple combined detector trained on both general objects and traffic light states is used. For the simple combined detector images from both the general object class and the subclass dataset are combined into one set and the detector is trained on both type of images. With the simple combined detector the traffic light states are recognized, as is seen in the images in figure 1.3. However, the amount of general objects detected has dropped significantly compared to the detections in figure 1.2. There are less general object detected with the simple combined detector because during training of the detector the images from the traffic light subclass dataset used for training have no annotations for general objects, which results in the detector learning not to detect general object classes in those images. Methods to solve these difficulties are proposed and tested in this work.

The main research question for this work is:

- What is a good way to make an object detector with good performance on both a general object dataset and a subclass dataset?

Sub-questions for this investigation are:

- How can subclass dataset traffic light states be combined with a general object dataset to train YOLOv2 on both sets?
- Is it possible to do traffic light state subclass detection based on the YOLOv2 feature map?

General object class
object detector

Figure 1.2

Simple combined
general class object and traffic light state
object detector

Figure 1.3

Figure 1.4: Samples of frames detected with a general object class detector (left) and a simple combined detector trained on general object classes and subclasses (right). The first value next to the detected class gives the confidence the detector has of the detection being an object, the second value the confidence of class of the object. In the images on the right general object detection is affected negatively by the inclusion of the sub-class dataset during training. Despite it now being able to detect traffic light classes a lot of the ability to detect general object like cars has been diminished.

2

Related work

Traffic light detection based on visual information from an in vehicle camera is first mentioned in [25] as part of a set of techniques for an intelligent vehicle. The traffic light recognition system is based on color segmentation, filtering and classification. The classifier is a small neural net fed with region proposals able to perform binary classification.

Since [25] much research has been published on traffic light recognition. An overview paper [38] describes earlier systems up to 2015, stating up to 2015 most traffic light recognition systems follow a computer vision pipeline that breaks down the traffic light recognition problem into three sub-problems: detection, classification and tracking. The flow for this computer vision pipeline is shown in figure 2.1. A detector is first used to find possible traffic light candidates in the input feed. After the detector the classifier extracts features out of the candidates to determine whether a candidate is a traffic light or not, and possibly also the state of the traffic light. Tracking is an optional method that can be added to feed back temporal information to improve detection. In [38] detection is separated into model based detection and learning based detection. Model based detection is based on heuristic models based on shape, color and intensity to find the traffic light candidates, the learning based detection methods utilize machine learning for detection. Up to 2015 mostly model based detection is used, but new research starts focusing on learning based detection. Disadvantages for the learning based detectors are the amount of training data and computational time needed, but the advantages are the higher robustness to variation and less tendency to overfitting. Some of the best working systems in [38] also incorporate detailed maps of the route and temporal information to improve performance.

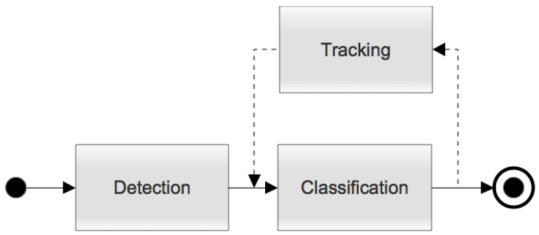


Figure 2.1: breakdown of a computer vision pipeline for traffic light recognition system [1]

The following sections first describes recent relevant research done on traffic light recognition, separated into the computer vision pipeline methods and the non computer vision pipeline methods. After this section the sections 2.3 and 2.4 describe relevant research for the model used in this work. Section 2.5 describes relevant datasets for this investigation.

[1] Jensen [38]

2.1. Computer vision pipeline methods

As described before computer vision pipeline methods are separated into two or three steps: detection, classification and possibly tracking, as seen in 2.1. The computer vision pipeline methods utilize methods like blob detection, color thresholds, and filtering shape and position for detection. For classification methods like support vector machines (SVM), neural networks or boosted classifiers are used. The following sections describe different methods used for detection, feature extraction and classification in computer vision pipeline methods.

2.1.1. Detection

During the first step in the traffic light recognition task, detection, it is most important not to miss any objects. The classifications module further processes the candidates of the detection module, meaning any false positive can be corrected for, but a missed object in the detection module can not be corrected for with the classification module. Following [38], a distinction is made in this section between model based detection and learning based detection. First the model based methods are described in section 2.1.1, then the learning based models are in section 2.1.1.

Model based detection

The model based detection methods utilize heuristically determined models to find traffic lights based on their characteristic properties like color or shape.

An often used property to detect a traffic light candidate is color. Traffic lights contain bright color bulbs with a known range of colors against the dark background of the traffic light box. The color bulbs can be found with semantic segmentation techniques. Color regions can be found by applying a threshold to a color channel. Color thresholding is widely used [13, 14, 32, 59, 61, 64, 65, 75–77, 82, 83, 85, 88]. Beside the color, also the intensity of the traffic light light bulb is used as a characteristic property to detect traffic light candidates [52]. An useful algorithm that can help with the segmentation task is blob detection, that groups pixels based on the position and value. It is used for noise reduction by [13, 14, 76]. The common next step after color thresholding and blob detection is to use some morphological operations on the resulting segmented areas of the images to close the gap [14, 64, 65, 74, 75, 77, 83].

The next characteristic used to find traffic lights is the shape of the traffic light and of its light bulb. Methods used for shape detection are Hough Circle detection [65, 88] or aspect ratio checks [13, 52, 64, 75–77, 83, 85]. [59] proposes a refinement to the shape detection by adding a stereo camera for measuring the size of the traffic light to reduce candidates. [14] and [61] do not use the shape directly, but the relative position and size of the different components of the traffic light as a template for traffic lights.

There are also more complex systems used to detect traffic lights. One of these is the Adaptive Background suppression Filter (AdaBSF) algorithm by [91]. It is designed to overcome computational and multi-scale problems of other methods and works by making a 4-channel feature map with edge features and RGB pixel values. The feature map is used to filter multi-scale input images that are selected as background or not background by thresholding and non-maximum suppression, combined with some other tricks to correct for different circumstances. [40] uses a Visual Selective Attention (VSA) method, proposed in [35] that uses Gaussian based filtering in the HSI and LAB color space from where a spectral residual approach gets an A channel Saliency Map (ASM) and a Saturation Saliency Map (SSM) that are combined to get candidate regions.

Learning based detection

Few methods are described using a learning based detector. To begin with, [26] uses a lookup table per pixel that has learned the characteristics colors from annotated traffic lights and segments the pixels into the *traffic light* or *background* category. A different approach is in [64], which makes use of thresholding on HSL images with ratio check for circularity and filling techniques to detect the traffic lights. The threshold values for this operations are learned from their own captured dataset. A form of feature learning is used in [52], where candidate regions are extracted with an ellipsoid geometry threshold model. An ellipsoid region in the statistical distribution of the pixels in the HSL color space is selected, resulting in improving results compared to traditional linear color thresholding. The threshold parameters are learned using hand labelled traffic light regions. Training of a neural network for detection is used in [10], where YOLOv1 [71] is trained on traffic light instances for the detection phase. The detector returns image crops that are used for the classification. The version of YOLOv1 used in [10] is a modified version of the convolutional neural network, which is YOLOv1 with the classification module removed.

2.1.2. Features and Feature extraction

Classification of the proposals found by the detector is realised using features the computer is able to interpret. The features must be representative of the characteristics of a traffic light in order to be used for the two possible tasks of the classification step: verification and state classification. For state classification the most simple features that can be used are the color values of the pixels of the image, used by [9, 10, 19, 26, 39, 43, 60, 64, 75, 82, 87, 88]. The disadvantage of this color feature is the sensibility of the system to lighting conditions. Alternatively [11] does use HSV color features made by flood filling area's within a certain color range and splitting them into X×X area's for each channel of the HSV color space and combined into 3*(X×X) features. [11]'s feature has the advantage that it detects area's of similar color, which results in more reliable classification of the traffic light. Another way of representing the characteristics of the color features is combining them into histograms that form a filtered output for the classifier [13, 36, 37, 54, 59, 65, 91].

For the verification of traffic light candidates shape is an important characteristic. A popular feature representing shape in images is the HOG (Histogram of Oriented Gradients) feature [18]. HOG features are able to differentiate images based on the direction of the edges in the image, utilised by [19, 36, 40, 52, 61, 76, 91]. An alternative for HOG are HAAR-like features, which are used among other features by [77]. In this technique rectangular kernels consisting of a black and a white area are subtracted from the area of where the kernel is applied and the final results are summed to give the final value of the kernel operation. Different arrangements of the black and white area highlight different properties in the image. For efficiency HOG and HAAR-like features are combined using AdaBoost to only keep the best performing features and feed into a cascade classifier to only keep relevant areas of the image.

Extra information can be calculated after a segmentation process, like size, or ratio of the found area's are used as shape features for the classifier [14, 26, 32, 43, 75, 82]. [83] uses the position and the size of the ROI's (Region Of Interest) found in the detection stage as input parameters for their detector, because traffic lights are often found in the same region of the frame with similar sizes. [52] uses LBP (Local Binary Patterns) in combination with HOG. LBP checks for each pixel if the value is higher or lower than each of the eight surrounding pixels. A 0 is assigned to pixels having a lower value over the central one, a 1 otherwise. These eight bits are now combined in a byte that represents the value of one pixel.

For classification a combination of different features can also be used. [86] use PCANet for combining features. PCANet is an algorithm based on Principal Component Analysis (PCA) to automatically learn and select the best features. PCA is a common method for data analysis that can be utilized to extract the main feature of component data. [37] and [47] do use an approach with self learning features using an Aggregated Channel Features (ACF) algorithm. This algorithm takes different channel (color or gradient histograms for example) and combines that into a boosted feature. [85] use OTSU segmentation algorithm for segmentation of the traffic light with the background after initial color and shape segmentation. This has as advantage that both circular and arrow shape traffic lights can be detected and separated in the classification step. Since the OTSU results are binary this is used as a pre-processing step before HOG features are extracted.

2.1.3. Classification

The classification phase analyses the candidates from the detection phase to verify them as a traffic light or not a traffic light. Additionally, the state of the traffic light can be determined in this phase. A separation is distinguished between supervised and unsupervised classification methods. Supervised methods make use of labelled data for learning to classify proposals, unsupervised methods do classification by data clustering or separation based on commonalities in the feature space.

Supervised methods

Supervised classifiers rely on labelled training data to optimize the parameters classification. With sufficient training data the classifier can learn how to deal with real life situations and is in theory able to adept. A common classifier is the SVM (Support Vector Machine) [11, 14, 57, 65, 76, 77, 85, 86]. An SVM learns makes a decision plane in the feature space by finding the place where the data is separated the most. [32] does not only use an SVM, but also two Deep Neural Networks (DNN), LeNet and AlexNet, are used to find the traffic lights and give the state. Similarly [76] uses an SVM only for classifying the traffic light structure, for the state recognition they use a Convolutional Neural Network (CNN), [40] classifies the traffic lights using a SVM and uses a threshold on the amount of pixels in a certain range of the A channel from the LAB color space to determine if the light is red or green.

Different methods exist to combine multiple SVM's into a boosted classifier. Boosting is a method that combines different weak learners into one boosted classifier. To do this [91] uses a cascade SVM classifier, in

which there are several SVM's chained together with each SVM fed with the output of the predecessor. [61] makes use of a hierarchical SVM, a decision tree with an SVM at each decision point. [36] uses 3D prior information for candidate ROI's that are first classified using a AdaBoost classifier based on Haar-like features, then a cascade classifier distinguishes the traffic lights from the background and gives the state. [13] uses PCANet, which is a PCA Network and a multiclass SVM. The PCA network emulates a CNN to filter the input images before they are fed to the SVM. With this method they are also able to classify arrow shaped traffic lights. AdaBoost is another technique that is able to combine different classifiers (weak learners) to an improved boosted classifier [37, 54]. Adaboost works by only using features that improve the prediction capabilities of the model. A method similar to AdaBoost is Jointboost, which is used by Vladimir Haltakov and Ilic [82] to combine different classifiers. [52] uses two of the Extreme Learning Machine (ELM) variant K-ELM classifiers to classify the traffic lights. ELM is a feed-forward Neural Network, designed for fast classification of complex multi-class objects. The first K-ELM is for green traffic lights, the second for red. [83] tested different methods and concludes that a Random Forest algorithm outperforms SVM, Logistic Regression and Neural Network. Their classifier is based on position information. A Random Forest consists of several decision trees that are combined by bagging (bootstrap aggregating), which is another method designed to improve stability and accuracy of machine learning algorithm such as decision trees. [10] uses a small convolutional neural network fed with small crops of detected traffic lights. Small margins around the TL boxes are used as regional context which is used to exclude false positives. [76] use a convolutional neural network to classify the state of the traffic light after an SVM has found the regions.

Some works focus on the classification stage. In [57] the classification is done by first using color segmentation in the HSV color space to extract the green, yellow and red regions. Pixel clustering and aspect ratio checks are used to further localize the lights in the traffic light. The pictogram (contour of the traffic light bulb) is then classified using an SVM inputted with HOG features. Wang et al. [86] uses GPS + GIS information for the candidate ROI which are then classified using an SVM using machine learned features by PCANet.

Unsupervised methods

Unsupervised classifiers use unlabelled training data, which can be an advantage as good annotated training data is difficult to produce or find. Few unsupervised classification methods for traffic light recognition are described in literature. An unsupervised classifier is made in [75] with a Decision Tree classifier based on shape, ratio's and color information to find the traffic lights. Good results are found in night and day situations using this method. A partly unsupervised technique is utilized in [26], that makes use of the depth information from stereo camera's to verify the bounding boxes found by their detector. A simple unsupervised method is used in [88]. They use Hough transform on the color segmented and smoothed images to found the circular traffic lights, using their detection module directly for classification.

2.1.4. Fusion and temporal integration, tracking

To improve the performance of a traffic light recognition system tracking can be used. With tracking a detection is followed over multiple frames and is used to make the system more stable and better at cutting out false positives. For traffic light recognition it can make sure that only static objects are found, which is used as a distinguishing between a traffic light and a tail light from a car. Besides this it can keep a traffic light in sight, even when not visible on a particular frame, what can be caused by synchronization issues between the blinking frequency of LED traffic lights and the camera shutter speed.

A simple tracking approach is used by [82]. In this work a traffic light is detected if it was also detected in the previous three frames. [13] implements tracking with mean shift on the detected frames to track traffic lights and a Simple Moving Average (SMA) to forecast traffic light positions. Another variant of the mean shift algorithm, Continually Adaptive Meanshift (CAMShift), is used by [74]. Alternativaly, [9] does use a recursive Labeled Multi-Bernoulli (LMB) filter to track traffic lights, which is a 'computationally tractable approximation of the recursive multi-object Bayes filter using random finite sets'. [10] use an odometry based motion model that tracks the bounding boxes using their position over time and their static position in space. To process all data a CNN is used to estimate misplacement of a traffic light from a prototype image.

The Lucas-Kanade tracker [53] is used by [39] to track traffic light with a ROI that is separated in different regions to recognize a change in the traffic light state. Lucas-Kanade tracking assumes constant optical flow in the neighborhood of a tracked pixel and solves optical flow equations for neighboring pixels. [52] also uses the Lucas-Kanade tracker for their final-state framework used to enhance the reliability of their traffic light recognition system.

Kalman Filters [66] can be used for multi-object tracking. In [36] a Kalman Filter is used for estimating four states including top-left position, with, height and traffic light box in image coordinates. The Kalman Filter in this work is assisted by a Nearest Neighborhood Filter (NNF) for data association. [59] uses a Kalman Filter to track each new candidate region their detection method finds. The state that is tracked is the x and y positions of the found traffic light as well as the velocity of this position. A correction term is used to consider the color transition for the position of the traffic light in horizontal mounted traffic light cases. A different correction term has to be used for vertically mounted traffic lights.

2.2. Non computer vision pipeline methods

The non computer vision pipeline methods combine detection and classification into one single step for traffic light recognition. Different machine learning methods are used in literature to make con computer vision pipeline methods.

One of the first papers to show an improvement of a learning based traffic light detection system over heuristic systems is in [54]. An Aggregated Channel Features (ACF) detector outperforms two heuristic methods, Back Projection and Spotlight Detection on their new and challenging LISA dataset. The ACF uses features from 10 channels (input representations) and a sliding window approach. The features are represented as small rectangular blocks in every channel and classification is done using a modified AdaBoost classifier that boosts depth-2 decision trees as weak learners into the final classifier. In [37] an ACF detector is used for detecting traffic lights at night as well as [47], who do take the ACF detector as their baseline method and improves it by using channel feature modification, multi-size detection, bulb detection, fuzzy logic and inter-frame correlation analysis. [83] investigates different machine learning techniques to improve their baseline detection method that uses color segmentation and some morphological operations. They conclude that a Random Forest algorithm outperforms a SVM, Logistic Regression and a Neural Network in their position based traffic light candidate elimination step of their traffic light detection system. [34] approaches the traffic light recognition problem as a binary labeling problem based on color, shape and height characteristics. Furthermore, spatio-temporal constraints are used for coherency in space and time. This method is implemented in a Bayesian statistical network that calculates per pixel the probability of it being a traffic light. Spacial coherency is achieved by assuming that neighbouring pixels with similar color belong to the same object. For the temporal coherency it is assumed that the traffic lights are static in the world plane. Moreover, information from GIS maps and the activation pattern of traffic light lenses is used to further detect the traffic lights. Some parameters in the network such as mean and variance are not known beforehand. These are learned using an Expectation Maximization Network.

Deep learning methods work with a neural network that is trained with a large amount of labelled training data. [87] uses a deep learning approach called deepTLR, a single 7 layer convolutional neural network made specifically for traffic light recognition feeded with RGB input images. First a probability map is made by classification of fined grained pixel regions. Above certain thresholds pixels are activated. Over each activated pixel region a bounding bounding box prediction per class is found by the convolutional neural network. A specialized traffic light recognition detection network is designed by [84], called Circular Traffic Light (CTL) Deep Neural Network. For training the amount of convolutional layers is set back to 0, reducing the network to a basic backpropagated neural network using standard gradient descent optimization and different activation functions with a cross-entropy and a quadratic cost function. For the classification several convolutional layers with max-pooling reduce the dimensions and dropout, preventing the network from overfitting. A pre-trained convolutional neural network is used in [42] to generate a saliency map in the offline phase (training of the system) that is later used to localize traffic lights in the online phase (usage of the system). The saliency map contains the vehicles' position and orientation combined with a given image of the scene. The traffic light is detected by first finding a region of interest from a prior database, over which a CNN is trained. The trained CNN works as a multi-class classifier to detect traffic light states. In [44] six different color spaces (RGB, normilized RGB, Ruta's RYG, YCbCr, HSV and CIE Lab) are tested on three different end-to-end deep convolutional neural networks: Faster R-CNN [72] with Inception-Resnet-v2 [81], Faster R-CNN with Resnet-101 [33] and R-FCN [17] with Restnet-101. They conclude that for traffic light state detection the RGB color space works best, regardless the model. Their Faster R-CNN with Inception-Resnet-v2 has the highest performance on traffic light recognition, with the observation that yellow traffic lights are more often mis-classified. This can by caused by the low amount of yellow traffic lights in their training set. A traffic light detector including state and type (arrow, circle, etc) based on Faster R-CNN [72] is made in [8]. They trained their network directly on the traffic light states on the novel DriveU Traffic Light Dataset [27]. They report good detection

on medium sized traffic lights, but it struggles with small size traffic lights. An analysis of the false positives reveals that the detector is confused about other kinds of traffic lights, like pedestrian or tram traffic lights.

Single shot detectors are methods that only need one pass through the network per image, making traffic light recognition faster than with the earlier mentioned CNN methods. The single shot detector YOLOv2 is used in [60] to make a traffic light detection. In this work different training data and scaling of the training data was tested. The highest result was found by training on all the daytime training data from the LISA traffic light dataset including an evaluation set. They also found that random scaling of the input images does have a positive effect on the UAC, but making the initial input scale of the training data bigger has a larger impact than scaling. [60] only uses daytime data and the results do not include state recognition. They get a AUC of 90.94 by training on all the dayTime training data and the second evaluation set of the LISA dataset on a input resolution of 672 by 672 pixels. No traffic light states are included in this work. Another single shot detector is used in [62]. Here SSD [51] is used and state detection is included. For the feature extractor Inception [80] network is used, because the better accuracy-speed trade off compared to VGG [78] and because the inception modules concatenate different receptive fields, which is beneficial for traffic light state detection because of the combination of contextual and local information. A state prediction module is used for traffic light state detection. The state prediction module adds an additional layer beside the binary classification layer (traffic light vs background), which is optimized via its own loss functions, hence not affecting the classification performance of the network.

2.3. Convolutional neural networks for object detecting and classification

The previous section discussed different methods using machine learning with a convolutional neural network. In this section more information is given about these and an overview is given of different CNNs used for object detection and classification in images. A CNN works as follows: an $W \times H \times C$ image object is taken as input, with W and H being the image width and height and C the color channels. A set of convolutional filters, scaling operators and activation functions is applied to the image pixels to reduce it to a $aW \times aH \times F$ feature map, with aW and aH being the adapted width and height of the image and F the amount of features. Scaling operations include down-scaling of the width and height with pooling operations or up-scaling using interpolation. Batch normalization [46] can be used to improve the speed, performance and stability by normalizing the input layer. Classification is typically done with some fully connected layers on top of the feature map. The parameters for the convolutional and fully connected layers are learned during training. Data augmentation is often used on the input images to make the network better at generalizing. The first convolutional neural network described for image classification outperforming traditional methods at the time is AlexNet [45]. AlexNet was the first to utilize GPU for faster calculations of the convolutions. Networks like AlexNet are trained for classification, which means the output layer of the network is a probability distribution over the different classes.

Object detection is finding the location of objects in an image, which is a difficult task because objects can appear at different places in the images at different scales. The first method to use a convolutional neural network for image classification and localization is RCNN (Regions with CNN features) [31]. RCNN uses a region proposal network to extract crops from the image that are classified with the CNN. The disadvantage of such a method is that for one image multiple image crops have to be processed by the CNN, RCNN for example uses around 2000 region proposals per image, which makes such a system to slow for real time object detection (especially when higher resolution input images will be used, which is needed to detect small objects like traffic lights). Despite further development of RCNN designed to make it faster with Fast-RCNN and Faster-RCNN [30, 72] it is by design not the optimal method for real time object detection and classification because the CNN has to process image crops multiple times per image. Faster-RCNN gets up to 5fps.

A different framework for real time object detection and classification is proposed by two networks with a similar approach: YOLO (You Only Look Once) [71] and SSD (Single Shot Detector) [51]. Per image a single run though the SSD or YOLO network is enough to output bounding box and class for every object at every scale in the image. SSD uses a base network outputting features layers at different scales upon which an extra convolutional layer is added for detection and classification on that scale. YOLO on the other hand uses a customized network resizing the image to a $S \times S \times F$ single scale feature map. Two fully connected layers use this feature map to detect and classify images. YOLO and SSD perform at 45 to 59 fps with a precision comparable to Faster-RCNN. SDD has been improved with DSSD (Deconvolutional SSD) [28] that added some deconvolutional layers on top of SSD to include additional context into SSD, and MDSSD (Multi-scale DSSD)

[16] that adds high-level features to the low-level features via deconvolutional Fusion Blocks. YOLO has been improved with YOLOv2 (or YOLO9000) [68] and YOLOv3 [70]. YOLOv2 removed the final fully connected layers and replaced it with a region layer for speed. Batch normalization and anchors boxes are also added for better and more stable results. YOLOv3 made the YOLO network significantly larger, from 19 to 53 convolutional layers. It also implemented bounding box predictions at 3 different scales [49]. YOLO has a lower accuracy than the SSD variants, but the trade-off is that YOLO is faster.

Overall one-stage detectors like SSD and YOLO are faster than two-stage detectors like RCNN, but RCNN has a higher accuracy. In [50] it is found that this difference is mainly due to class imbalance during training. During training a lot of candidate locations per image, out of which only a few contain objects. This makes training inefficient and too much focused on the negative samples. They introduced the loss function *focal loss* by adding a modulation factor to the *cross entropy loss*. The *focal loss* leads to better prediction for one-stage detectors.

2.4. Incremental learning

Incremental learning is a method for adding new sub-classes to a trained classifier, which may be beneficial for a traffic light recognition system bases on an existing object detector.

[12] uses a feature extractor module, that can be any (convolutional) neural network structure, followed by multiple classification layers in parallel. The classification layer produces logits that are used to calculate a distillation loss. These are combined into a cross distillation loss for the complete model. At test time the loss function is replaced with a softmax layer. Adding more training data to this model means adding an extra classification layer, while the structure of the feature extractor stays the same. [56] trained a convolutional network using different databases. The network used is a single feature extractor network which output is used for several different detectors. They tackle the problem of semantic level-of-detail, which can occur when some database has overlapping classes with classes another database. For example when one database has a class traffic light and another has separate classes for the traffic light states. [56] proposes a semantic label hierarchy, where the lower level classes consist of sub classes of the top level classes. Each level of the hierarchy tree has its own classifier in the network. They also solved class imbalances between sets by placing classes with similar order of examples into the same classifier, which makes that all classes have bigger probability to be represented in the same batch.

2.5. Datasets

Two different kind of datasets are relevant for this work: the general object class datasets and the subclass datasets. The general object class dataset consists of general images with a broad range of objects annotated. The subclass dataset contains annotations for one or more (sub)classes out of the general object class dataset on specific images for this (sub)class. In this work the subclass dataset is a traffic light dataset containing traffic light states as subclass of the general *traffic light* class in a general object class dataset.

For general object detection COCO [48], VOC [22] and ImageNet [41] are used extensively in literature. Table 2.1 gives an overview of the amount of classes and images in these dataset. ImageNet is the largest set with over 14 million image at the moment of writing and is based on the WordNet hierarchy. ImageNet is mainly used for training image classifiers. In their yearly challenge a training set of 1.2 million images annotated in 1000 classes is released. VOC is an image classification and localization dataset also having a yearly challenge with a dataset annotated for 20 classes. Two different sets came out for the VOC challange: the VOC_2007 and VOC_2012. VOC_2007 consists of 9963 images, VOC_2012 of 11530. COCO is a dataset containing 80 classes, including all the VOC classes. The COCO set is therefore more challenging than VOC. For COCO bounding box and segmentation annotations are included. Two configurations of COCO has been released over the past years, one in 2014 and one in 2017 [6]. YOLOv2 has been trained on the 2014 set, so the 2014 set will be used in this work. Table 2.4 gives an overview of the 2014 train and validation set, the 80 classes are shown in table 2.2b.

Multiple datasets exist containing annotated traffic lights. Table 2.2 gives an overview of the datasets containing annotated traffic light states including the amount of classes and annotation. Figures 2.3 to 2.8 show samples out of the different traffic light datasets.

LISA (Laboratory for Intelligent & Save Automobiles) [54] is the largest and the newest dataset used in literature, containing 43007 images. It contains over 100.000 traffic light annotations and contains day and night data over multiple routes. The data is captured with a camera fitted on a vehicle driving different routes containing traffic lights. Annotations are present for the traffic light boxes as well as only for the lights. RCMP

Table 2.1: General object detection datasets

Name	Classes	#Images
ImageNet	1000	1200000
VOC_2007	20	9963
VOC_2012	20	11530
COCO_2014	80	82081

Table 2.2: Overview of the different publicly available TL databases

Name	Classes	#Frames	#Annotations	Location	Year	Video included	Image spec.
LISA (VIVA)	7	43007	118331	San Diego, USA	2015	yes, 44 min, 41 s 16fps *	Stereo 1280 x 960 8-bit RGB
RCMP (LARA)	4	11179	9168	Paris, France	2010	yes, 8 min, 49 s 25fps	640 x 480 8-bit RGB
WPI - Training holder	12	22588	15489	Worcester, USA-summer	2016	no	1920 x 1080 RGB
WPI - Training lights	8	3376	3631	Worcester, USA-summer	2016	no	1920 x 1080 RGB
WPI - Testing	12	8112	4207	Worcester, USA-winter	2016	no	1920 x 1080 RGB
Bosch - Training	15	5093	10756	El Camino Real, San Francisco Bay Area, USA	2017	yes, 2 fps	1280 x 720 8-bit RGB
Bosch - Testing	4	8334	13493	University Avenue in Palo Alto, USA	2017	yes, 15.6 fps	1280 x 720 8-bit RGB
KITTI	-		440	Karlsruhe, Germany	2011	yes	1027 x 768 RGB

* Although the video is mentioned in the database description, no video file is present in the files available for download.

(Robotics Centre of Mines ParisTech) [2, 21] contains over 10.000 images with under 10.000 annotations. It contains only one image set, making it less fitting for training a neural network. WPI (Worcester Polytechnic Institute) [13] dataset does contain training data and has over 20.000 images with more than 15.000 annotated traffic lights. It has annotations for both the traffic light boxes as the lights. It consists of high resolution images taken in various seasons and lighting conditions. Data without traffic lights is also included into the WPI dataset. KITTI (Karlsruhe Institute of Technology an Toyota Technical Institute at Chicago) [29] traffic light dataset is part of the KITTI Vision Benchmark Suite containing a wide range of sensor data for investigation of intelligent vehicles. The traffic light objects are not labelled in the KITTI benchmark.

Table 2.3: Overview of LISA dataset

set type	set	length	frames	annotations	classes
Train	dayTrain	14m 32s	14025	40764	go, go left, warning, warning left, stop, stop left
	nightTrain	7m 8s	5910	14297	go, go left, warning, warning left, stop, stop left
Test	daySequence1	4m 14s	4060	10308	go, warning, warning left, stop, top left
	daySequence2	7m 11s	6894	11144	go, go forward, go left, warning, stop, stop left
	nightSequence1	5m 12s	4993	18984	go, go left, warning stop, stop left
	nightSequence2	6m 48s	6534	23734	go, go left, warning stop, stop left

The current top of the line database recommended by Jensen [38] is the LISA dataset, as part of the VIVA challenge [24]. The LISA dataset contains challenging day and night scenes with traffic light annotations categorized into seven classes: *go*, *go forward*, *go left*, *warning*, *warning left*, *stop* and *stop left*

LISA consist of a training and a testing part. The training part is 13 day clips totaling 14m32s and 5 night clips totaling 7m8s. For testing two day and two night sequences are available, as can be seen in 2.3. Table 2.2a shows the distribution of the classes over the dayTrain training set. Out of the traffic light subclass datasets LISA is best fitted for this work because of the amount of annotations and classes, and because of the usage of the set in other papers to compare to.

(a) LISA dayTrain classes

class	annotations
go	13830
go forward	0
go left	851
warning	755
warning left	290
stop	15113
stop left	6971

(b) COCO 80 classes

person	fire hydrant	elephant	skis	wine glass	broccoli	diningtable	toaster
bicycle	stop sign	bear	snowboard	cup	carrot	toilet	sink
car	parking meter	zebra	sports ball	fork	hot dog	tvmonitor	refrigerator
motorbike	bench	giraffe	kite	knife	pizza	laptop	book
aeroplane	bird	backpack	baseball bat	spoon	donut	mouse	clock
bus	cat	umbrella	baseball glove	bowl	cake	remote	vase
train	dog	handbag	skateboard	banana	chair	keyboard	scissors
truck	horse	tie	surfboard	apple	sofa	cell phone	teddy bear
boat	sheep	suitcase	tennis racket	sandwich	pottedplant	microwave	hair drier
traffic light	cow	frisbee	bottle	orange	bed	oven	toothbrush

Table 2.4: Overview of COCO dataset

set type	set	images	annotations	classes
Train	2014 Train	82081	608695	80 COCO classes
Validation	2014 val	5000	35757	80 COCO classes
Test	2014 test	40775	n.a.	80 COCO classes
Test	2017 test	40670	n.a.	80 COCO classes

Figure 2.3: Screenshots of the LARA video sequence

Figure 2.4: Sample images of the LISA dataset

Figure 2.5: Sample images of the Bosch dataset

Figure 2.6: Sample images from KITTI dataset

Figure 2.7: Sample images of WPI datasets scenes

Figure 2.8: Samples ROI's of the WPI dataset

2.6. Contributions of this work

In this work a traffic light recognizing system is integrated into a general object detector. All described systems in literature made a traffic light recognition only able to detector traffic light (states). On the other hand, self driving cars will make use of traffic light recognition as well as detection of other classes like *cars*, *pedestrians* or *cyclists*. It is possible to use two different detector for these tasks, however neural networks use significant computer power (even real time single shot detectors) and computer power is limited in vehicles. For these reasons this work proposes to combine general object class detection and traffic light subclass detection.

From the literature a transition toward non computer vision pipeline deep learning methods is visible, pushed by the better performance of these systems. The single shot convolutional neural networks are best suited to make a fast and accurate traffic light recognition system. YOLOv2 is therefore used in this work as it provides the best speed-accuracy trade-off. Furthermore, YOLOv2 uses a single feature map data representation, which is interesting to test as a feature extractor for traffic light recognition system and has proven good performance on general object class datasets VOC and COCO.

In this work the COCO dataset is chosen as the general object class dataset and LISA as the subclass traffic light dataset. The COCO set is chosen as the general object class dataset for the amount of training data and the presence of the *traffic light* class. For the traffic light subclass dataset LISA is chosen. The LISA set is the one of the largest and most used dataset in literature, and is therefore the best choice or this work.

COCO has around 6 times more training images than LISA. This might lead to problems during training due to imbalances data. In the COCO training set there are 599528 annotations, an average of 7.3 annotations per image. For equally large classes this means each class has 7589 annotations. In comparison LISA has 40764 annotations in dayTrain. Out of this the *go* and *stop* class have the most annotations. *go* and *stop* have with 13830 and 15113 annotations respectively around 2 times more annotations then the classes from COCO. In the end the extra amount of COCO training images does not lead to COCO classes outnumbering the LISA classes. The problem that can occur though is that the other LISA classes *go left*, *warning*, *warning left* and *stop left* are not correctly trained for.

Another possible downside of the LISA dataset is that the training and test sets do not have all the same classes. The dayTrain train set does have 6 classes, but the daySequence1 test set only has 5 classes, with *go left* missing. The *go forward* classes is only present in the daySequence2 test set and is missing in the training sets.

Making a combined general object class and subclass detector is challenging as different datasets have to used to train one system. Section 2.4 shows different solutions for this challenge. In this work adds to these two different methods for combining general object class detection and traffic light subclass detection using YOLOv2.

In the first system YOLOv2 is trained on classes from the COCO general object dataset and LISA traffic light dataset. In this method the overall network structure is kept intact, with a modified loss function optimizing combined training for general object classes and subclasses (see section 3.2.1 for the details). The second system uses the backbone network of YOLOv2 pre-trained on COCO as feature extractor. On this feature map a traffic light recognition system will be trained with LISA data. The advantage of the second system is that the out-of-the-box performance of YOLOv2 on COCO remains untouched. It will on the other hand be slower to run than the first method and the performance is dependent on how well the traffic light state is represented in the feature map.

This work makes the following contributions:

- It presents a novel method to train YOLOv2 on COCO general object classes and LISA traffic light subclasses.
- It investigates training of a subclass detector on top of the YOLOv2 feature extractor.

3

Method

The goal of this work is the investigation of methods to combine a general object class detector with a subclass detector. This chapter describes the proposed methods using the YOLOv2 [68] detector and the COCO [48] general object class dataset and the LISA subclass dataset [54]. Section 3.1 first explains YOLOv2, then section 3.2 proposes two methods for a combined object detector. The first method is separated into multiple tested detectors, all described in section 3.2.1 and together named the **combined training methods**. In these detectors YOLOv2 is trained with both the general object class and subclass datasets combined. In the second method, described in section 3.2.2, the traffic light recognition part is trained separately using the feature map extracted with the YOLOv2 backbone network trained on COCO with the **YOLOv2++ method**. Finally, section 3.3 and 3.3.1 give details about testing and evaluation of the different methods.

3.1. YOLOv2

YOLOv2 [68] is a further development of the YOLO (You Only Look Once) single shot object detector [71]. YOLOv2 is a convolutional neural network designed for object detection and classification with one single pass trough the network per image, making it able to perform in real time. Objects at different scales at different positions are detected in this single network pass, which is achieved by dividing the image into a S×S grid with each grid cell responsible for detecting an object, if the object has its center in the grid cell. The bounding box position is calculated as an offset from an anchor box, of which 5 are assigned per grid cell. The anchors are calculated per dataset by k-means over the ground truth bounding boxes in the dataset. YOLOv2 decouples the class prediction and spatial location models. For each bounding box it calculates the objectness, that is, the certainty that the network has for this bounding box to contain an object. Additionally to objectness it calculates the probability for each class, so if one bounding box has a high enough objectness the class probability predicts the class for this bounding box. Each bounding box contains the boxes' coordinates, objectness, and class scores. The final output of the YOLOv2 network is a S×S ×(4+1+C)×5 tensor, which contains per grid cell the bounding box position and objectness plus class propabilities per anchor box.

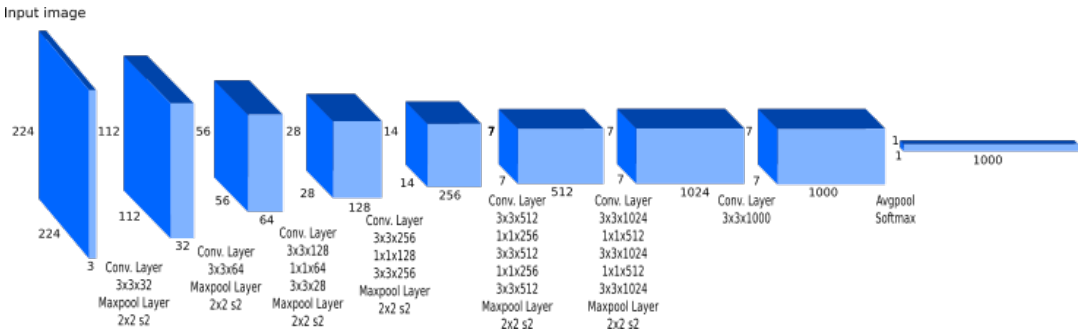
Network

YOLOv2 works with a convolutional neural network for extracting features out of the incoming images. The network structure is mapped out in figure 3.1b. The base of YOLOv2's network consists of 19 convolutional layers and is called Darknet-19. Darknet-19 is based on Google's GoogLeNet [79] with the inception modules changed to 1×1 reduction layer followed by a 3×3 convolutional layer. Furthermore, batch normalization [46] has been added to the convolutional layers. Darknet-19 gradually downsamples an input image to $1/32^{th}$ of the input resolution, resulting in a S×S×1024 feature map. A final convolutional layer, the region layer, reduces the S×S×1024 feature map to the S×S×(4+1+C)×5 mapping. Indicating 4 bounding box coordinates plus confidence for every anchor for every cell. For example a 224×224 pixels input images finally results in a 7×7×(4+1+C)×5 mapping, in which a maximum of 7×7×5 objects can be detected, if spaced perfectly. Each object can be one of C different classes. During detection non maximum suppression is used to reduce the bounding boxes per object and only the bounding boxes with a objectness above a certain detection threshold will be displayed.

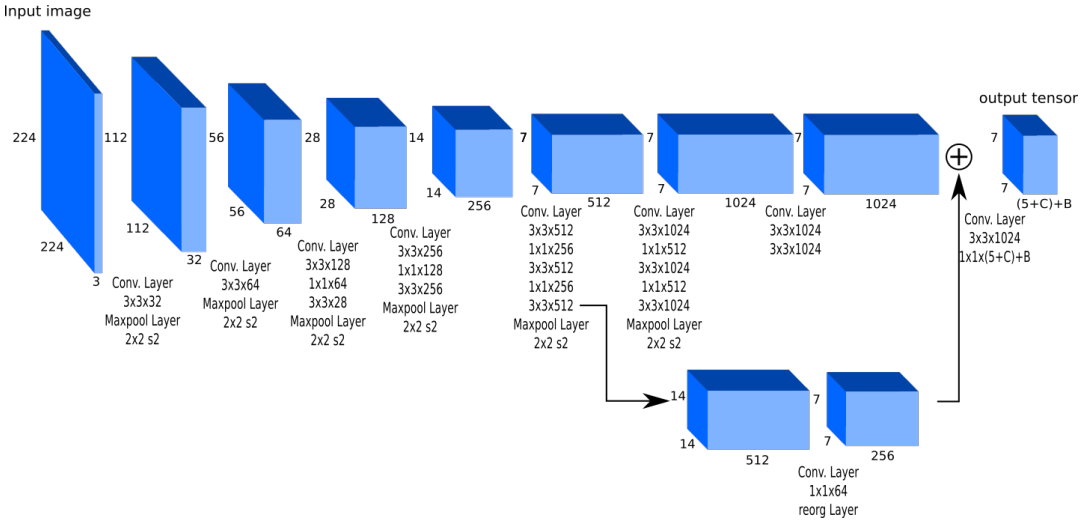
YOLOv2 is originally trained in two stages, first the network is trained for classification on the 1000 class classification network from ImageNet [73] on the network configuration of figure 3.1a. Then the network is altered to the network configuration of figure 3.1b for detection on the Pascal VOC [22] and COCO [48]

datasets. The modifications for the detection network are removing of the final convolutional layer and replacing it with three 3×3 convolutional layers with 1024 filters, followed by a final 1×1 convolutional layer and the S×S×(4+1+C)×5 region layer, and inserting of a passtrough layer from the 16th convolutional layer to the second to last convolutional layer to include fine grained details.

In this work the YOLOv2 detection network from figure 3.1b is used. For this network training is done from pre-trained weights for the base 19 layer CNN downloadable from YOLOv2's website that are already trained on ImageNet or COCO.



(a) Architecture of the YOLOv2 network for classification



(b) Architecture of the YOLOv2 network for detection

Figure 3.1: Architecture of the YOLOv2 detection and classification networks for a 224×224 pixels 3-channel input image. Every block represents an (intermediate) feature mapping, with the neural network layers written below the blocks.

Loss function

During training YOLOv2 optimizes a multi-part loss function, which is unfortunately not defined in YOLOv2's paper [68]. It is however defined for YOLOv1 [71]. With the network alterations mentioned in YOLOv2's paper and following this blogpost [5], the loss function for YOLOv2 used in this work is defined as described in this section.

YOLOv2 outputs S^2 cells with B anchor boxes per cell. For each grid cell i and and anchor j the total loss is

$$loss_{i,j} = loss_{i,j}^{xywh} + loss_{i,j}^{p} + loss_{i,j}^{c}, \tag{3.1}$$

where $loss_{i,j}^{xywh}$ is the bounding box position loss, $loss_{i,j}^{p}$ is the class loss and $loss_{i,j}^{c}$ is the confidence loss. The bounding box position loss calculates the error between the correct position of the bounding box (x,y,w,h) and the predicted position $(\hat{x}, \hat{y}, \hat{w}, \hat{h})$ with

$$loss_{i,j}^{xywh} = 0.5 * \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} L_{i,j}^{obj} \left[(x_{i,j} - \hat{x}_{i,j})^2 + (y_{i,j} - \hat{y}_{i,j})^2 + \left(h_{i,j} - \hat{h}_{i,j}\right)^2 + \left(w_{i,j} - \hat{w}_{i,j}\right)^2 \right], \tag{3.2}$$

which is the mean squared error over the bounding box coordinates. The term $L_{i,j}^{obj}$ is a 0/1 indicator that is 1 if the found bounding box has the best IOU with the ground truth bounding box, which ensures the bounding box position loss only contributes to the total los for found bounding boxes belonging to an object.

The class loss calculates the error between the correct class p and the predicted class $\hat{p}$ with

$$loss_{i,j}^{p} = -\lambda_{class} \sum_{i=0}^{S^2} \sum_{j=0}^{B} L_{i,j}^{obj} \sum_{c \in classes} p_{i,j}^{c} \log\left(\hat{p}_{i,j}^{c}\right), \tag{3.3}$$

which calculates the cross entropy loss for c classes for every bounding box with a found object, as indicated with the $L_{i,j}^{obj}$ 0/1 indicator.

Finally the confidence loss calculates the error between the predicted confidence $\hat{C}$ and the IOU of the ground truth box and the predicted box for boxes containing an object, and for boxes not containing an object with

$$loss_{i,j}^{c} = 0.5 * \lambda_{obj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} L_{i,j}^{obj} \left(IOU_{prediction_{i,j}}^{groundtruth_{i,j}} - \hat{C}_{i,j}\right)^2 + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} L_{i,j}^{noobj} \left(IOU_{prediction_{i,j}}^{groundtruth_{i,j}} - \hat{C}_{i,j}\right). \tag{3.4}$$

In these losses the λ's are scalar values determining how much every loss component contributes to the total loss. For YOLOv2 λ_{coord}, λ_{class}, λ_{noobj} are 1 and λ_{obj} is 5. The parameters $L_{i,j}^{obj}$ and $L_{i,j}^{noobj}$ are 0/1 indicator functions defined as

$$L_{i,j}^{obj} = \begin{cases} 1, & \text{if } IOU_{prediction_{i,j}}^{groundtruth_{i,j}} = max\left(IOU_{prediction_{i,j}}^{groundtruth_{i,j}}\right) \\ 0, & \text{otherwise} \end{cases}, \tag{3.5}$$

and

$$L_{i,j}^{noobj} = \begin{cases} 1, & \text{if } max\left(IOU_{prediction_{i,j}}^{groundtruth_{i,j}}\right) < 0.6 \text{ and } C_{i,j} = 0 \\ 0, & \text{otherwise} \end{cases}, \tag{3.6}$$

with $L_{i,j}^{obj}$ indicating a box contains an object and $L_{i,j}^{noobj}$ indicating a box does not contain an object. The box does not contain an object if the confidence of the current predicted bounding box is zero and there is no predicted bounding box that has a higher IOU with a ground truth bounding box that 0.6.

At the start of training the network struggles to predict bounding boxes in the correct place. To improve this YOLOv2 calculates the bounding box position loss of all found boxes for the first 12800 images seen by the network during training, which results in the found bounding boxes converting to the anchor boxes. The anchors boxes give a representation of the average correct size for the bounding boxes in the dataset, so these are better fitted for the bounding boxes the network should find. After 12800 seen images the bounding box position loss is calculated as described in equation 3.2.

Implementation

The original YOLOv2 network is build in C and CUDA [67], and is uncommented and low level, which makes it hard to make alteration into. Therefore in this work an adaptation of YOLOv2 into the PyTorch [7] framework is used [90]. PyTorch is a high level open source deep learning framework in python, which alows for high level and convenient programming of machine learning methods. Pytorch includes accelerated computation for tensors on a GPU using CUDA [1]. The YOLOv2 PyTorch implementation includes training and evaluation. It works with the same configuration and weight files as the original implementation. This network is not as fast as the original implementation, although it does make use of GPU acceleration. The Pytorch implementation uses more GPU memory than the original implementation, so a smaller batch size of 16 images instead of 64 (for 608×608 pixels rgb images) can be used during training on a Nvidia Titan V with 12GB of video memory.

3.2. Proposed methods

Two different methods for combined detection and classification of the COCO general object class dataset and the LISA subclass dataset are described in this section. The first method is combined detection with YOLOv2 trained on a combination of COCO and LISA classes. For the second method YOLOv2 is altered to YOLOv2++ to detect LISA classes on the YOLOv2 feature map.

3.2.1. Combined training methods

To train YOLOv2 on general object classes and subclasses two combined training methods are used. For the combined training method the whole network is trained for both COCO general object classes and LISA subclasses. The output is a prediction over 87 classes (80 from COCO and 7 from LISA). Two different methods are investigated for combined training: the simple combined training method and the adaptive combined training method. The simple combined training method works with a dataset alteration, the adaptive combined training method works with an alteration of the YOLOv2 loss function.

Simple combined training method

The simple combined training method uses the YOLOv2 network adapted to output 87 classes, with the training data the combined list of training images from COCO and LISA. To reduce conflicts during training between the COCO *traffic light* class and the LISA traffic light state classes during training the simple combined training method removes the *traffic light* annotations in the COCO dataset. At locations the COCO *traffic light* objects where located the LISA subclass annotations are used. Training is done from pre-trained weights on COCO.

adaptive combined training method

The adaptive combined training method is designed to overcome problems encountered with the simple combined training method. The first problem is subclass dataset images from LISA do not have general object classes from COCO annotated and vice versa. When during training a LISA image is processed and a COCO class object is detected this is flagged as a false positive, which could very much be wrong because the detector is also training on COCO images.

To take this into account during training the loss function of the adaptive combined training method detects whether an image is from one dataset or another by looking at the label (the ground truth annotation) of the image. When in the image a class $\hat{c}$ is predicted from one dataset d while that image is from the other dataset this will not contribute to the object confidence loss. By applying the condition to $L_{i,j}^{noobj}$ and $L_{i,j}^{obj}$ via

$$L_{i,j}^{noobj}, L_{i,j}^{obj} = \begin{cases} 0, & \text{if } \hat{c} \in c_d \oplus c \in c_d \\ 1, & \text{otherwise} \end{cases} \tag{3.7}$$

there is no confidence loss for a found box with class c_d in an image not from dataset d.

The second problem specific for the COCO and LISA datasets is the conflicting position of the LISA classes and the COCO *traffic light* class. At a traffic light object annotated in LISA two correct classes can be assigned to it, the *traffic light* from COCO and the traffic light state from LISA. In YOLOv2 only one class is assigned per location. However, this relation between the datasets can be used to improve the confidence of the detector on the LISA traffic light state classes. During training the class confidence of a COCO *traffic light* $\hat{p}_{i,j}^{c=COCO\_TL}$ is added to the class confidences of the LISA classes $\hat{p}_{i,j}^{c \in LISA}$ via

$$\hat{p}_{i,j}^{c \in LISA} = \hat{p}_{i,j}^{c \in LISA} + \hat{p}_{i,j}^{c=COCO\_TL}, \tag{3.8}$$

so the class confidence that COCO has for a traffic light is helping to detect the LISA traffic light state classes.

Both the simple combined training method and the adaptive combined training method use weights pre-trained on COCO. These weights are trained on 608 by 608 pixels making YOLOv2 detected objects in a 19×19 grid. Because of this resolution in the weights both combined training methods are trained on 608×608 pixel input images.

3.2.2. YOLOv2++

For the YOLOv2++ method the YOLOv2 network is modified to detect both general object classes from COCO and subclasses from LISA. YOLOv2++ trains the subclasses on the final feature map of Darknet-19, which is

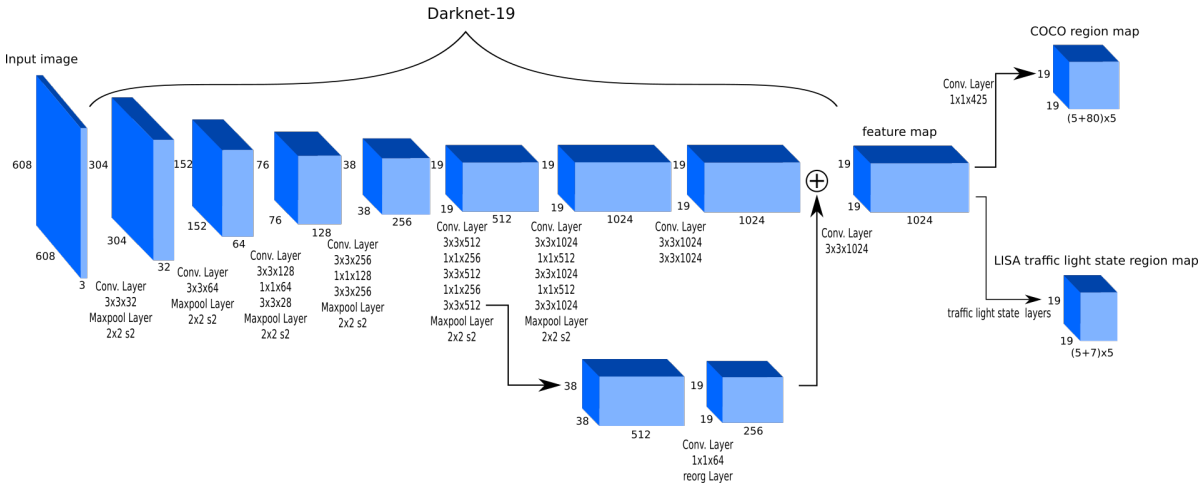


Figure 3.2: YOLOv2++ architecture. The feature map extracted by Darknet-19 is as input for the COCO region layer and the traffic light state layers.

a S×S×1024 tensor. To find the traffic light instances in this feature map a new smaller network, the traffic light state net, is trained on top of the this feature map. The whole system, feature extractor plus the region proposal layers, is the YOLOv2++ system. Figure 3.2 shows the YOLOv2++ system with Darknet-19 feature extractor on the left and two region proposal networks on the right. For COCO images the network configuration remains unchanged from YOLOv2. For the LISA part a few custom traffic light state region proposal layers are added, forming the traffic light state net. For this layers only convolutional layers with a 1×1 filter size are used to maintain the spatial mapping of the feature map. Using the weights pre-trained on COCO for YOLOv2 it is possible to make a traffic light recognition system by only training the traffic light state region proposal layers.

Training traffic light state region proposal layers
Different configuration are proposed for the traffic light state region proposal layers. Because Darknet-19 with the correct pre-trained weights is already able to detect traffic lights the simplest solution is just a 1×1×60 convolutional layer remapping the feature map to the region map for the 7 LISA classes (traffic light state region proposal 1-layer). To give YOLOv2++ some more flexibility to find the traffic light states in the feature map an extra 1×1×516 layer can be used in between (traffic light state region proposal 2-layer). For YOLOv2++ the anchors of the anchor boxes for the traffic light state net are recalculated for the LISA train set.

3.3. Testing

To evaluate the performance of all systems one testing method is used. Testing is done on both the general object class and the subclass dataset. Different tests are performed per dataset, see table 3.1 for an overview. For the subclass traffic light recognition part LISA daySequence1 is used as the test set. On daySequence1 two tests are performed. The first test is a binary classification test that tests whether a found bounding box belongs to a traffic light instance. The second test is a multi-class classification test that tests the found class of the bounding boxes. The first test tests the object detection performance of the network, the second test also tests the classification performance. The detector trained with the combined training method does output more classes than only the ones belonging to the LISA annotations, however only the boxes that are classified in a LISA class are evaluated in the traffic light recognition tests. For the multi-class classification the results are evaluated using precision-recall curve and a confidence matrix shows the performance between classes. To test the general object class detection multi-class testing is done over the COCO 2014 evaluation set as test set. Evaluation is done on all 80 COCO classes including *traffic light*.

In the original COCO challenge [6] testing is performed via the COCO evaluation server, which did not work for the test sets from 2014 or 2017. The test-dev set from 2017 does go through, but it does not give a correct results back. In the end in this work evaluation will be done offline on the 5k COCO 2014 evaluation set using AlexeyAB's fork for Darknet [4], which includes code to calculate the mAP for COCO. Both the combined training methods and YOLOv2++ are trained and tested on images with 608×608 pixel input resolution. The reason for this is the pre-trained weights on COCO used are trained on this resolution. To keep the com-

parison equal between all system in this work all systems are trained and evaluated on the same resolution.

Table 3.1: Overview of evaluation sets per database the and the testing method used per set

database	set	test type	metric
LISA	daySequence1	binary	precision-recall curve
LISA	daySequence1	multi-class	precision-recall curve + confidence matrix
COCO	2014 5k	multi-class	mAP

3.3.1. Evaluation metrics

Evaluation of the performance of the object detection system for traffic light recognition is done with the area under the precision-recall curve [20] as defined in the VIVA challenge [24]. Precision and recall are the preferred metrics for object detection as metrics based on true negatives like accuracy or false positive rate are inappropriate to use [22]. Multi-class object detection databases like COCO, VOC or LISA only have positive samples labelled. True negatives become now detector dependent as every bounding box the detector evaluates not corresponding to a object can be seen as a true negative. For the YOLOv2 detector, with $S^2 * 5$ bounding boxes per image, this leads to a skewed metric toward negative samples. Precision and recall do not rely on true negatives and are the preferred metrics for object detection, so precision and recall are the used metrics in this work.

To understand precision and recall first the matrix for binary detection is shown in table 3.2. The true positives are the detections that correctly detected an object in the image, false positives are the detections that have falsely detected something as an object in the image. False negatives are the objects in the image that are falsely not detected by the detector and the true negatives are the instances in the image that are correctly not detected.

Table 3.2: Binary confusion matrix

	Actual Positive	**Actual Negative**
Detected Positive	TP	FP
Detected Negative	FN	TN

Figure 3.3 gives a visual representation of values in the confusion matrix as well as the precision and recall. The precision is calculated with the true positives and false positives with

$$Precision = \frac{TP}{TP + FP} \tag{3.9}$$

and represents the fraction of the detections that is actually correct. The recall is calculated with the true positives and the false negatives with

$$Recall = \frac{TP}{TP + FN} \tag{3.10}$$

and represents the fraction of the detections that is correct compared to the ground truth annotations. Precision and recall are combined into the F measure (also F1-score, or F-score) with

$$F = 2 \times \frac{Precision \times Recall}{Precision + Recall}, \tag{3.11}$$

which represents the geometric mean between the precision and recall.

Precision and recall are evaluated together with a precision-recall plot, generated by plotting recall and precision points on the X and Y axis of a 2d plot for different object confidence thresholds. The perfect precision-recall curve starts in the top left corner, ends in the lower right corner and is pushed into the top right corner so the area under the curve is 1.

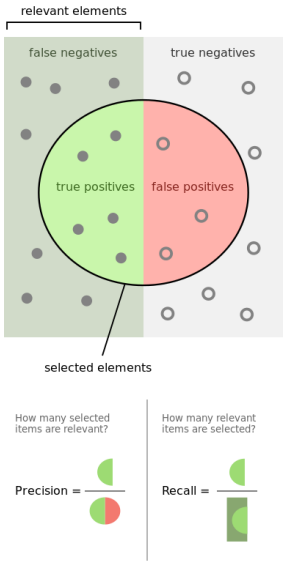


Figure 3.3: Visualization of the values in the binary confusion matrix, and of precision and recall[1]

[1] By Walber - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=36926283

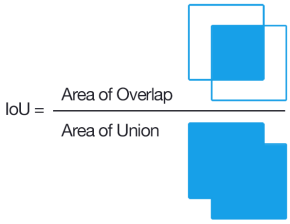
The area under the precision recall curve is called the AUC (Area Under Curve) and is equal to average precision (AP) [92] for binary classification which is defined as

$$AP = \int_0^1 Precision(Recall)\,\mathrm{d}Recall. \tag{3.12}$$

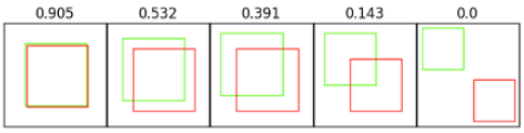
For general object dataset COCO the multi-class performance is measured as the mean Average Precision (mAP) with

$$mAP = \frac{\sum_{q=1}^{Q} AP(q)}{Q}, \tag{3.13}$$

which is the Average Precision (AP) averaged over all classes Q. For the evaluation of the multi-class LISA traffic light classes the AUC of the multi-class precision-recall curve is used, which is calculated almost the same as for the binary case, with the only difference a detection is labelled as true positive if also the class of the detection is correct instead of only the position. The reason to compare the COCO results with mAP and LISA with AUC is to compare results to other works via the common metric per dataset.



(a) Visualization of Intersection over Union (IOU)[2]



(b) Sample scores of IOU[3]

For image detection systems the position of a detection is correct when a detection has a certain overlap with the GT (Ground Truth) bounding box. The overlap is measured with the IOU (Intersection Over Union) calculated by

$$a_0 = \frac{area(B_d \cap B_{gt})}{area(B_d \cup B_{gt})} \tag{3.14}$$

and visualized in figures 3.4a and 3.4b. In the VIVA challenge [24] it is defined that for a IOU of 0.5 or higher a detection is a true positive, which corresponds to the PASCAL VOC criterion [22].

For multi-class detection an extra metric used is the confusion matrix, in which the confusion between classes is plotted. On the vertical axis are the actual classes and on the horizontal are the predicted classes. The values on the main diagonal of the matrix show the amount of correctly classified detection per class, the other cells show how classes get misclassified.

[2]Adrian Rosebrock - A visual equation for Intersection over Union http://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/

[3]http://ronny.rest/tutorials/module/localization_001/iou/

Detection on multiple IOU values

Detections with a lower IOU than 0.5 will be also be evaluated in this work. The first reason for this is that although it is important to know the location for detection of traffic lights, in most scenes the traffic lights have enough distance between each other that no confusion exists over to which traffic light a detection belongs. The performance of the detector can be improved by the lower IOU threshold with no consequence to the practical usage of the detector. The second reason is that in the LISA dataset the labeling of the boxes is not consistent as can be seen in consequent as is show for a sample frames in figure 3.5. The detector might give consequent bounding box positions at the correct location and the detection might still not be recognized as a true positive due to the poor labeling. With a lower IOU threshold this error can be reduced.

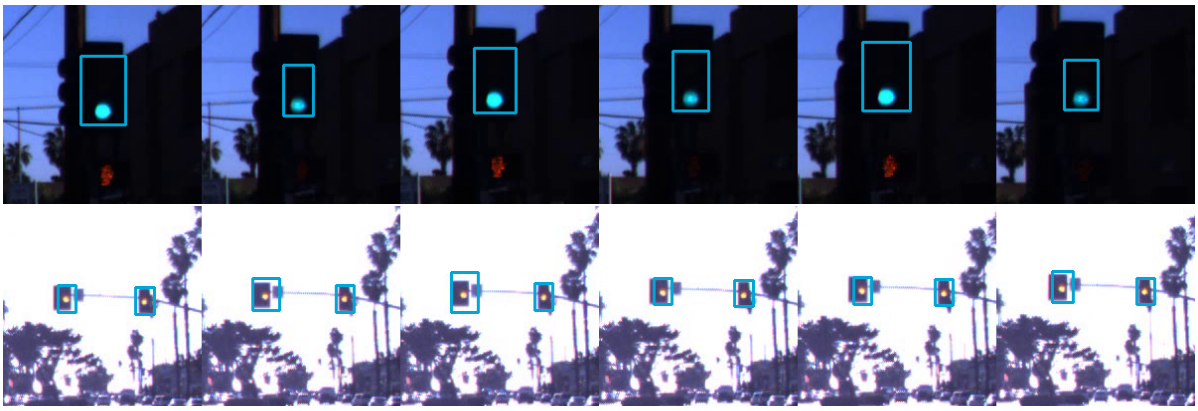


Figure 3.5: Samples non consistent labeling in LISA dataset. On top frames 357-362, on the bottom frames 1894-1899 are shown.

4

Experiments and results

The following chapter describes experiments measuring the performance of the proposed methods for building a combined general object class and subclass detector. In all experiments the COCO dataset is used as general object class dataset and the LISA traffic light dataset is the subclass set. For the traffic light subclass part of the experiments the traffic light detection and traffic light recognition performance is measured. First benchmarks will be given of existing methods, than the different methods for a combined detection method are tested in section 4.2 and 4.3.

4.0.1. Literature baselines

From literature several benchmarks for traffic light detection and traffic light recognition on the LISA set exist. No previous work exists measuring performance on both datasets with one system, therefore for every dataset and every situation a different benchmark is presented.

traffic light detection

For traffic light detection on LISA dataset using YOLOv2 the current benchmark is given by [60]. The best result on LISA daySequence1 is achieved by training on both LISA dayTrain and daySequence2 on input resolution 416×416 evaluated on a 672×672 input resolution. An AUC of 90.94% with a IOU of 0.5 is achieved. When only training on LISA dayTrain 416×416 pixels and using 416×416 pixels input for evaluation the highest result is 51.15%. With input resolution of 672×672 pixels for evaluation the AUC rises to 58.30%. Concluded is that increasing the evaluation resolution does have a positive effect on the results, but more training data has more effect.

traffic light recognition

The baseline for traffic light recognition comes from [47]. A model with combined prior feature analysis process with modified feature learning methods is used. On LISA evaluation set (it is not mentioned which sets are used exactly), an AUC of 72.50% for red, 27.67% for red left, 52.16% for green, and 40.47% for green left traffic lights is reported for IOU 0.5. The average AUC over all classes is 47.95%.

COCO object detection with YOLOv2

YOLOv2 reports an mAP of 44.0% [68] on COCO test-dev 2015 set on a resolution of 488×488 with IOU 0.5. In this benchmark YOLOv2 is only trained on COCO data.

4.0.2. Experiments description

To test the different systems the following experiments are conducted. All systems are trained on a 608×608 pixel input resolution because this is the highest resolution available for YOLOv2 weights trained on COCO [69].

1. The base performance of YOLOv2 trained on COCO is evaluated for traffic light detection and COCO in section 4.1. This gives the baseline performance of the out-of-the box system.
2. The simple combined system in section 4.2 is the first self trained model in this work. It is trained on both the COCO general object classes and the LISA traffic light state subclasses and serves as a comparison for the advantage of extra complexity introduced in the next systems in this work.
3. For the adaptive combined training method the detector is trained end-to-end on the same data as the simple combined training method. Results are presented in section 4.2. The hypothesis is that the adaptive combined training method will outperform the simple combined training method.

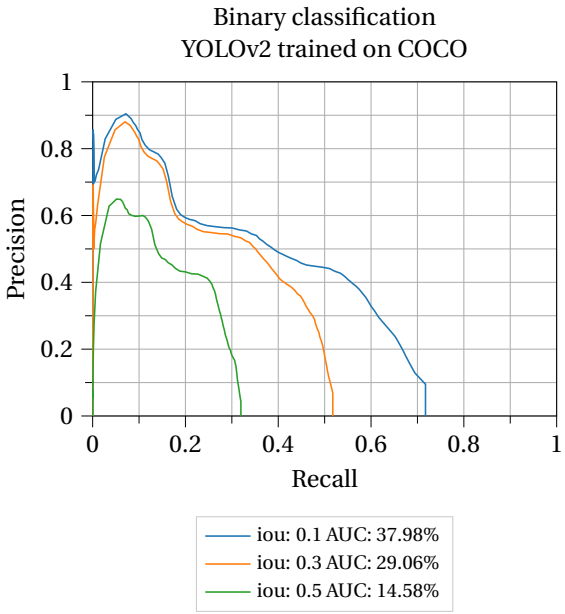
4. YOLO++ is tested by training LISA traffic light states on the feature map of YOLOv2. The LISA training data is the same as in the previous experiments. The base network for this is darknet-19 trained on COCO (so the feature map contains traffic light information). The performance on LISA and COCO is compared to simple combined training method and the adaptive combined training method. Different configuration for the traffic light state region proposal layers are tested. Results of YOLOv2++ are presented in section 4.3.

In the following section first the performance of YOLOv2 out-of-the box is presented, than the performance figures of combined training method and YOLOv2++ are presented separately. All methods are compared with precision recall curves under similar circumstances in section 4.4.1. Finally, a visual comparison of all systems is given in section 4.4.2.

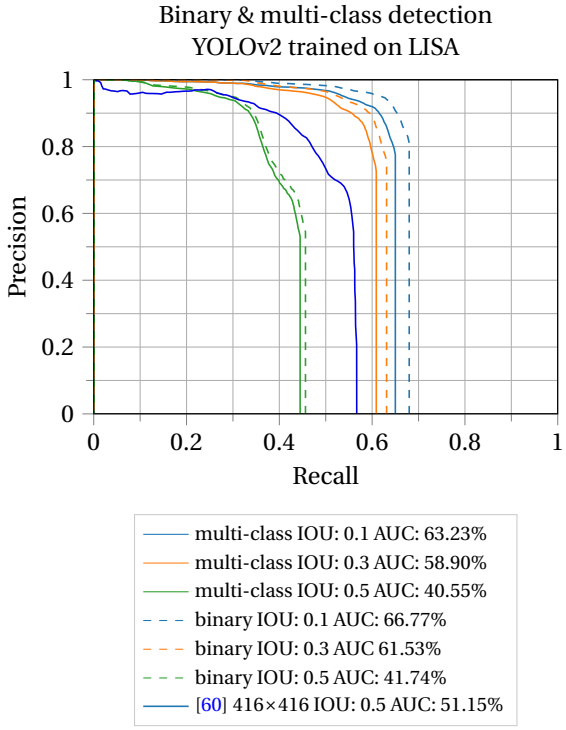
4.1. YOLOv2 out-of-the box baseline performance

Out of the box YOLOv2 is able to detect TL instances using the pre-trained weights trained on COCO classes. A few frames out of the LISA test set detected with YOLOv2 using the pre-trained weights from COCO are given in figure 4.2. The detector detects the TL instances at the right locations without state recognition. Included are detections not suitable for traffic light recognition, which are detections of traffic lights from the side and the back. When evaluating the detector on the LISA test set these traffic light instances are considered false positives.

Figure 4.1a shows the performance of YOLOv2 trained on COCO and evaluated on LISA daySequence1 at 608×608 pixels input resolution. Only detection of class *traffic light* are tested against the ground truths of the LISA test set (not considering the LISA classes) for 3 different IOU values. The dependency of the performance on the IOU is clearly visible: for IOU 0.1 the AUC is more than twice as for IOU 0.5. For lower values of the IOU the bounding boxes are allowed to fit less strict over the ground truths, so for a large increase in performance with lower IOU the localization of the detector can be more precise.



(a) Results YOLOv2 trained on out-of-the box COCO weights, detection on only the COCO *traffic light* class. Input resolution 608×608 pixels.



(b) Results YOLOv2 trained on LISA dayTrain, binary and multi-class detection on LISA daySequence1. Input resolution for model trained in this work 608×608 pixels. Comparison against [60] on 416×416 pixels.

Figure 4.1

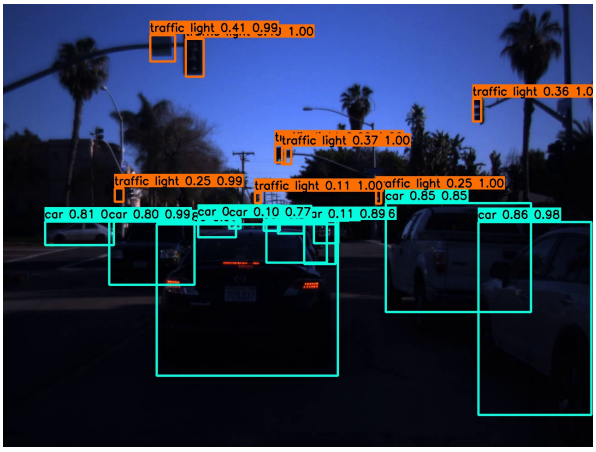
Figure 4.1b shows the results of YOLOv2 trained on LISA classes, forming an addition to [60] in which YOLOv2 is trained on LISA data. However, in [60], results are only calculated for binary traffic light detection, this work has the detector trained on LISA traffic light states. For binary detection only the bounding box position is compared to the ground truth, for multi-class detection also the class is compared. The binary classification results are up to 25 percent higher than the results of YOLOv2 only trained on COCO, with a little difference between binary and multi-class classification. The latter indicates YOLOv2 trained on LISA is loosing performance more on the localization than on the classification part of the detector. Comparing the resutls for IOU of 0.5 with [60], the recall is lower and the precision is on the same level. For a lower IOU the results of this work are better than [60]. The difference are assumed to be caused by [60] having only trained on traffic light instances instead of the traffic light states for LISA, as no multi-class classification results are given.

For the performance of YOLOv2 on general object detection table 4.1 shows the results of YOLOv2 trained on COCO on the COCO val_2014 set, which is the default test set for COCO in this work. Using the original weight downloaded from the YOLOv2 website [69] on COCO *test-dev* it reaches a mAP of 48.1%. Which is comparable the mAP of 48% that is reached with in the setup used in this work. Table 4.1 shows the same test for three confidence threshold values, 0.01, 0.25 and 0.4. The lower the confidence threshold, the lower the precision gets and the higher the recall gets. For lower confidence thresholds the detector outputs more bounding boxes, leading to some more false positives and less false negatives, leading to higher recall and lower precision. The mAP does not change for a different confidence threshold level, because it describes the area under the precision-recall curve constructed over various confidence threshold levels. The time needed to process all the images increases compared to the lower confidence threshold, which is expected because for a lower confidence threshold more objects will be detected in the image, giving a higher recall because less objects are missed and a lower precision because more false positives are found.

Table 4.1: performance of YOLOv2 on COCO val_2014 validation set for different confidence threshold (**thresh**) values of 0.01, 0.25 and 0.4

thresh	**precision**	**recall**	**F measure**	**mAP**	**Detection time (s)**
0.01	0.09	0.66	0.16	48	206
0.25	56	49	52	48	112
0.4	72	41	53	48	96

Visual results of both out-of-the box detection systems are presented in figure 4.2 for a low confidence threshold of 0.1. The system trained on COCO detects accurately vehicles and traffic lights. Due to the low confidence threshold a significant amount of false positives are displayed, however the detector detects these instances with low confidence. For traffic light detection the system trained on COCO detects the traffic light instances instead of the classes. The class detection is shown in the system trained on LISA. YOLOv2 trained on LISA has less false postives and less detections of traffic lights not front facing the camera. In one frame, frame 2072 an error is made with the classification, as the middle *stop* traffic light is detected as *warning*. Overall both systems show reliable detection results.



(a) daySequence1-1255 trained on COCO

(b) daySequence1-1255 trained on LISA

(c) daySequence1-2040

(d) daySequence1-2040 trained on LISA

(e) daySequence1-2072

(f) daySequence1-1255 trained on LISA

(g) daySequence1-2760

(h) daySequence1-1255 trained on LISA

Figure 4.2: Results of detecting on sample frames in LISA daySequence1 with YOLOv2 trained on COCO with confidence threshold 0.1 and IOU threshold 0.5. Traffic lights seen from the side and the back are also detected as traffic light.

4.2. Combined training methods

For the next experiments YOLOv2 is trained with data from COCO 2014 and LISA datasets in order to make one detector able to detect classes from both datasets. The class labels from COCO form the basis, the LISA labels are added at the end for a total of 87 classes (80 from COCO and 7 from LISA). A total of 94.856 training images is used, separated as follows:

total training images	94.856
LISA training images	12.775
COCO training images	82.081

Three different combined methods are tested. To begin, the simple combined training method adds the datasets together and uses the unaltered YOLOv2 network. The simple combined training method uses one dataset manipulation as the COCO traffic light is removed to prevent interference with the LISA traffic light classes. After that, the adaptive combined training methods uses the YOLOv2 network with a custom adaptive loss function to train on the combined dataset. The adaptive loss function is designed to overcome potential problems encountered with the simple combined training method. The simple combined training method is trained from weights pre-trained on COCO, the adaptive combined training method is trained once on pre-trained weights on ImageNet and once on weights pre-trained on COCO. All three methods are trained for 50 epochs.

4.2.1. Performance combined training methods

For all combined training methods the performance is shown for binary and multi-class classification. First, for binary classification the results are presented in figure 4.4 for the three combined training methods tested on the LISA test set. The simple combined training method is significantly outperformed by the adaptive combined training methods, proofing the simple combined training method is not sufficient for the task. Both the adaptive combined training methods perform similarly for IOU 0.1 and 0.3. For IOU 0.5 the system using COCO pre-trained weights has a significant drop of performance, All methods show a significant dependency on the IOU, more than for the the system only trained on LISA in figure 4.1b. The performance is 10 to 20 percent lower for the adaptive combined training methods.

The multi-class classification performance for the traffic light states in the LISA test set is in figure 4.6. The overall performance is around 10 percent lower than for binary classification, with a performance drop for both recall and precision. The difference between the adaptive combined training method pre-trained on ImageNet or COCO is still a few percent. However, the system with the COCO pre-trained weights outperforms the system with ImageNet pre-trained weights for the multi-class classification, opposed to to binary classification where the ImageNet pre-trained system reached the highest AUC values.

Compared to the system only trained on COCO or LISA the binary classification of the adaptive combined training methods is higher than the binary classification with YOLOv2 trained on COCO in figure 4.1a. The binary classification of the system trained only on LISA in figure 4.1b is more than 10 percent higher and the multi-class classification is even more than 20 percent higher.

Analyzing the performance of the adaptive combined training methods further the performance per class is presented in the figure 4.7. The *stop* and *go* classes have similar performance. The advantage of the system trained from the COCO pre-trained weights is the visible for the *warning* class, which has 30 percent better performance compared to the system trained from ImageNet. All other classes have negligible performance. In the confusion matrices is shown how the system pre-trained on ImageNet shows the largest error between the *warning* and *stop* class, corresponding to the performance on this class in the precision-recall curve. Furthermore the *go* class is too often misclassified as *stop*. The latter is still the problematic with the system pre-trained on COCO, the error between the *warning* and *go* is lower. Comparing these results to the baseline, the performance for the *stop* class is more than 25% lower, the performance for *go* is around 5% lower.

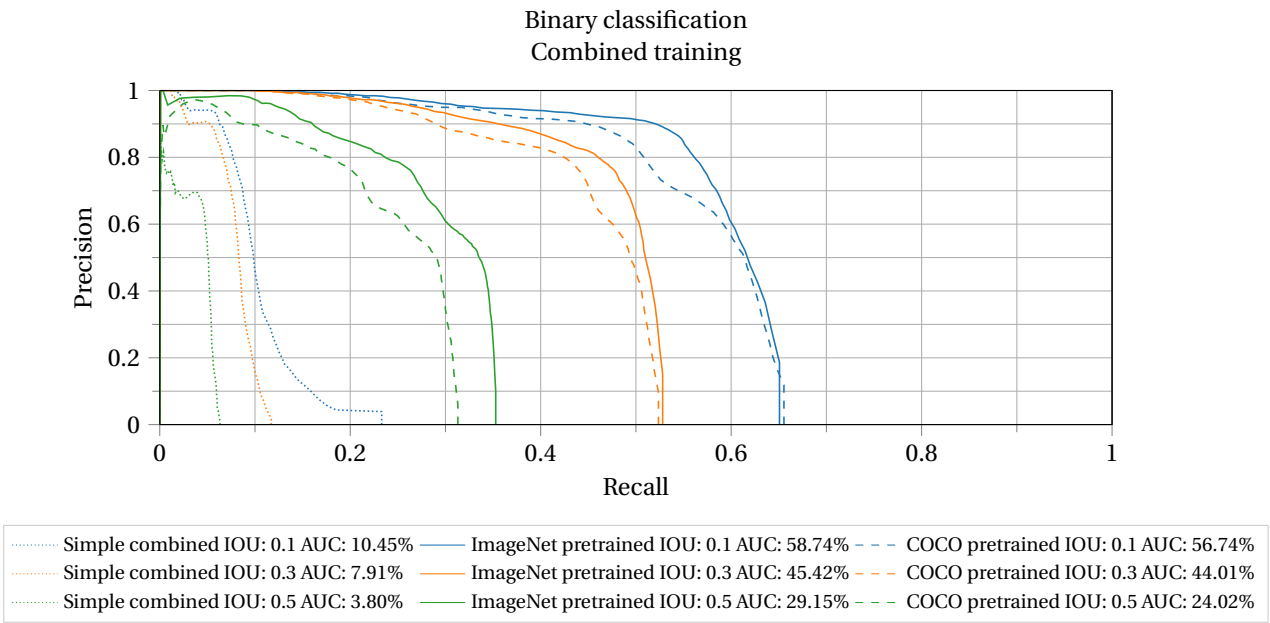


Figure 4.3

Figure 4.4: Precision-recall curves for model trained on COCO 2014 train set and LISA dayTrain traffic light states on input resolution 608×608 pixels. Binary classification for the simple combined method, the adaptive combined methods pre-trained on ImageNet and the adaptive combined methods pre-trained on COCO.

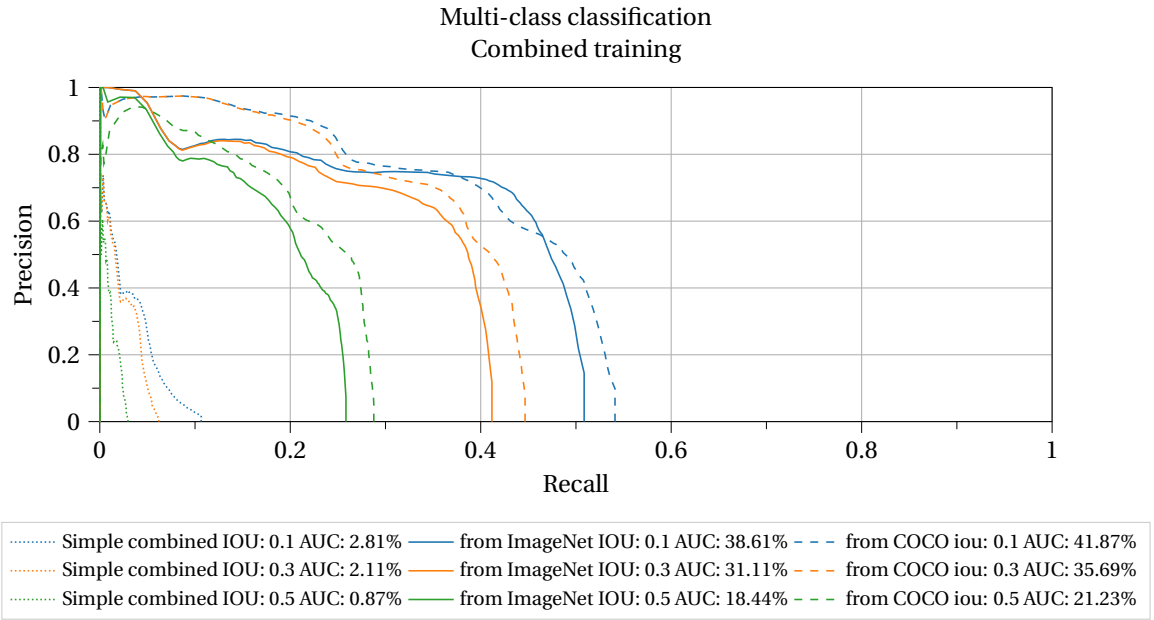
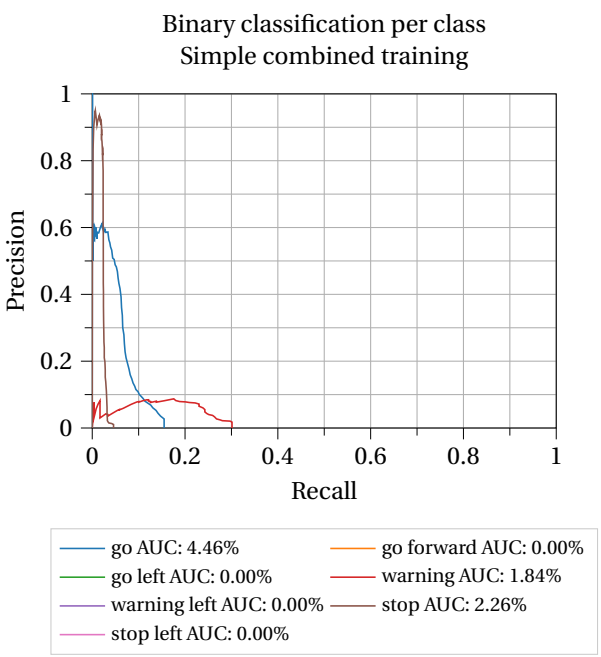
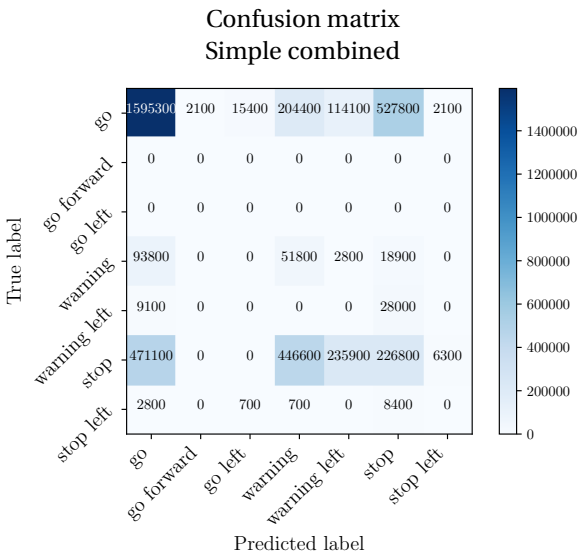


Figure 4.5

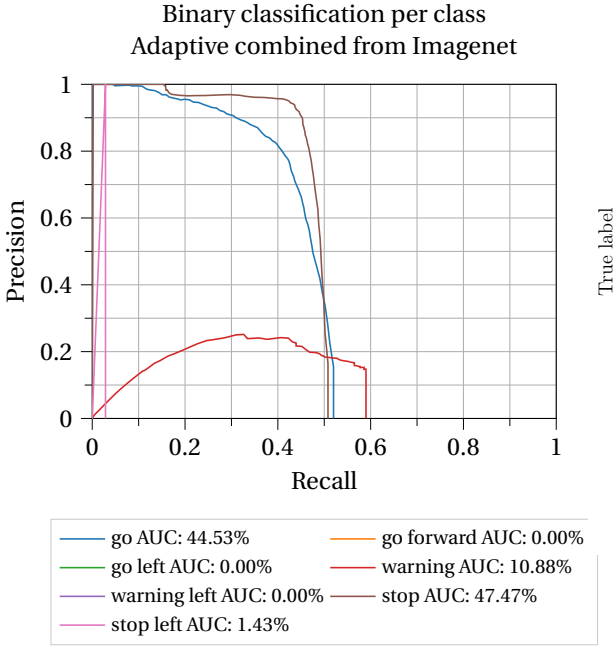
Figure 4.6: Precision-recall curves for model trained on COCO 2014 train set and LISA dayTrain traffic light states on input resolution 608×608 pixels. Multi-class classification for the simple combined method, the adaptive combined methods pre-trained on ImageNet and the adaptive combined methods pre-trained on COCO.


Binary classification per class
Simple combined training
Precision
Recall
go AUC: 4.46%
go forward AUC: 0.00%
go left AUC: 0.00%
warning AUC: 1.84%
warning left AUC: 0.00%
stop AUC: 2.26%
stop left AUC: 0.00%

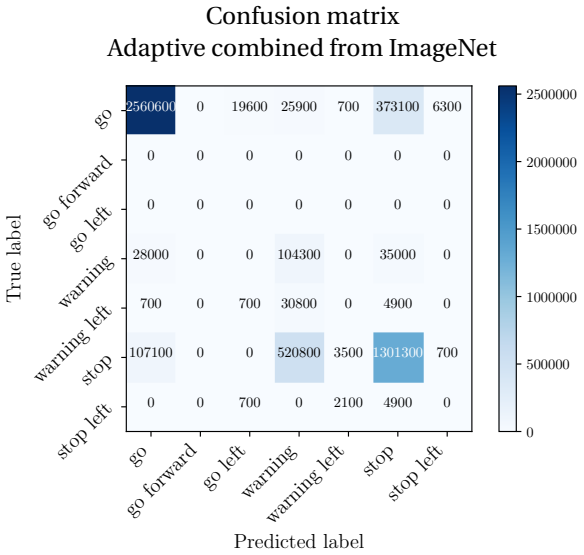
(a)


Confusion matrix
Simple combined
True label
Predicted label
go
go forward
go left
warning
warning left
stop
stop left

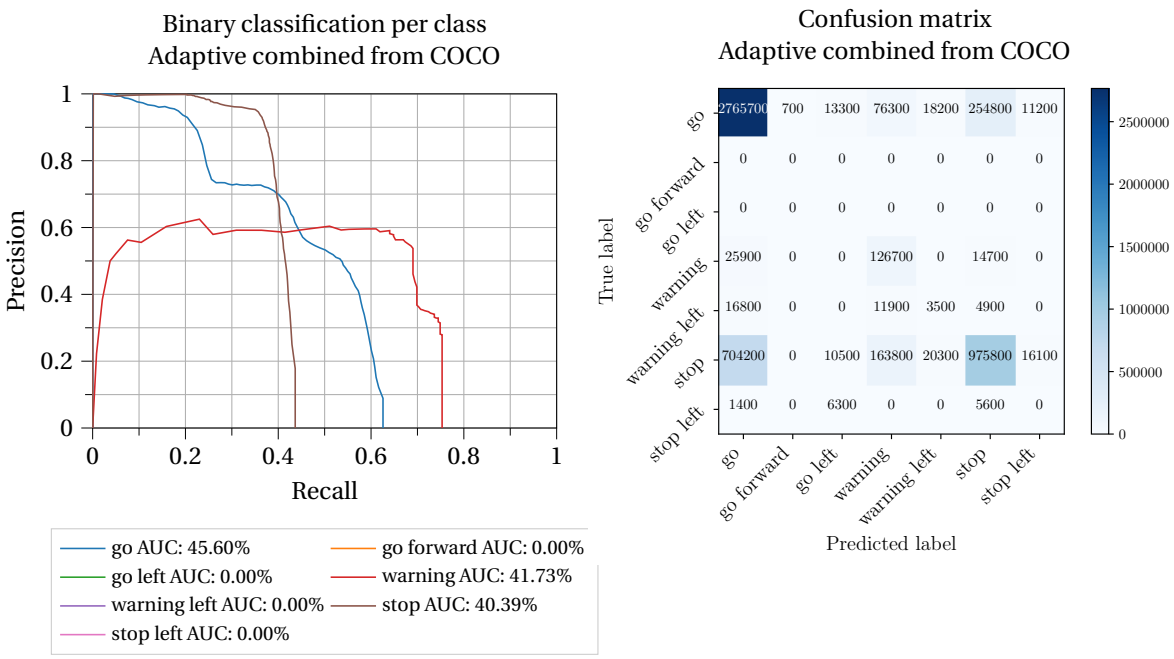
(b)


Binary classification per class
Adaptive combined from Imagenet
Precision
Recall
go AUC: 44.53%
go forward AUC: 0.00%
go left AUC: 0.00%
warning AUC: 10.88%
warning left AUC: 0.00%
stop AUC: 47.47%
stop left AUC: 1.43%

(c)


Confusion matrix
Adaptive combined from ImageNet
True label
Predicted label
go
go forward
go left
warning
warning left
stop
stop left

(d)



(e) (f)

Figure 4.7: Precision recall curves and confusion matrix for adaptive combined training methods pre-trained on ImagNet and COCO. Input resolution 608×608 pixels and IOU 0.1.

Table 4.2: Performance of the different detectors trained on COCO evaluated on COCO 2014 5k test set. Evaluated for confidence threshold 0.25. The out-of-box detector is only trained on COCO and uses the downloaded pre-trained weights [69]. The other three systems are detectors trained to detect general object classes from COCO and subclasses from LISA. Only the performance on COCO classes is evaluated.

system	**Pre-trained weights**	**precision**	**recall**	**F measure**	**mAP**	**Detection time (s)**
Out-of-the-box	*COCO*	*56*	*49*	*52*	*48.12*	*248*
Simple combined	COCO	26	30	28	24.75	307
Adaptive combined	ImageNet	32	41	36	33.3	279
Adaptive combined	COCO	34	51	41	43.78	237

The performance of the combined training methods for general object detection on the COCO test set are presented in table 4.2. The adaptive combined training methods with the COCO pre-trained weights gives the highest mAP of all the combined training systems, 10% above the ImageNet pre-trained weights. The performance of the adaptive combined method pre-trained on COCO is 5% lower than the system only trained on COCO, which indicates that YOLOv2 does loose performance with the addition of the extra subclasses. For the simple combined training method the performance on the COCO test set is not as low as on the LISA test set, which is originated by the weights pre-trained on COCO used for the simple combined method. However, during training of the simple combined training method the mAP of the simple combined training method on the COCO test set is almost half of the out-of-the-box performance on COCO, more than 20% more than for the adaptive combined training method. Proofing the adaptive combined training method better remains performance of the general object class dataset during combined training.

4.2.2. Visual results combined training methods

A visual comparison of the combined training methods is provided in figure 4.8. All detections above confidence threshold 0.3 are plotted for all the combined training systems for sample frames from the LISA test set. For the simple combined training methods the results reflect the performance visible in the precision-recall curves. There are little detections, which are mis-classified in frame 2040 and 2072. In frame 2760 class is correct, but the bounding box misplaced compared to the traffic light location. The difference between the pre-trained weights from ImageNet and COCO is visible in the traffic light objects detected from the back in frame 1255, 2040 and 2072. The detection of cars is similar for frame 1255 and 2040, in frame 2072 there are more cars detected with the system pre-trained on COCO. On the other hand does the systems pre-trained on COCO have a few more false positives on the last two frames. For traffic light recognition the system pre-trained on COCO has the advantage in frame 1255 with an extra detection and an extra correct classification in frame 2040 and 2072 for regarding the *stop* and *warning* classes compared to the system pre-trained on ImageNet, reflecting the performance per class in figure 4.7. In the sample frames the simple combined training method does have little general object detections, although the a mAP of almost 25% is measured on the COCO test set in table 4.2. The difference indicates the simple combined training method is still working relatively well on the COCO images, due to the pre-training on COCO, however on LISA images this performance is not represented. Making the simple combined method unfitted to be used as a combined general object and subclass detector.

(a) LISA daySequence1-1255 simple combined

(b) LISA daySequence1-1255 adaptive combined from ImageNet

(c) LISA daySequence1-1255 adaptive combined from COCO

(d) LISA daySequence1-2040 simple combined

(e) LISA daySequence1-2040 adaptive combined from ImageNet

(f) LISA daySequence1-2040 adaptive combined from COCO

(g) LISA daySequence1-2072 simple combined

(h) LISA daySequence1-2072 adaptive combined from ImageNet

(i) LISA daySequence1-2072 adaptive combined from COCO

(j) LISA daySequence1-2760 simple combined

(k) LISA daySequence1-2760 adaptive combined from ImageNet

(l) LISA daySequence1-2760 adaptive combined from COCO

Figure 4.8: Results of detection with combined training methods. Detection with confidence theshold 0.3.

4.3. YOLOv2++

As explained in section 3.2.2 YOLOv2++ is used to train some custom layers on top of the Darknet-19 feature map trained on COCO for detecting LISA images. Two layer configurations for the traffic light state layers are tested, as shown in table 4.3. The first configuration has a single 1×1×60 layer, the second has a 1×1×512 plus a 1×1×60 layer. Table 4.3 gives an overview.

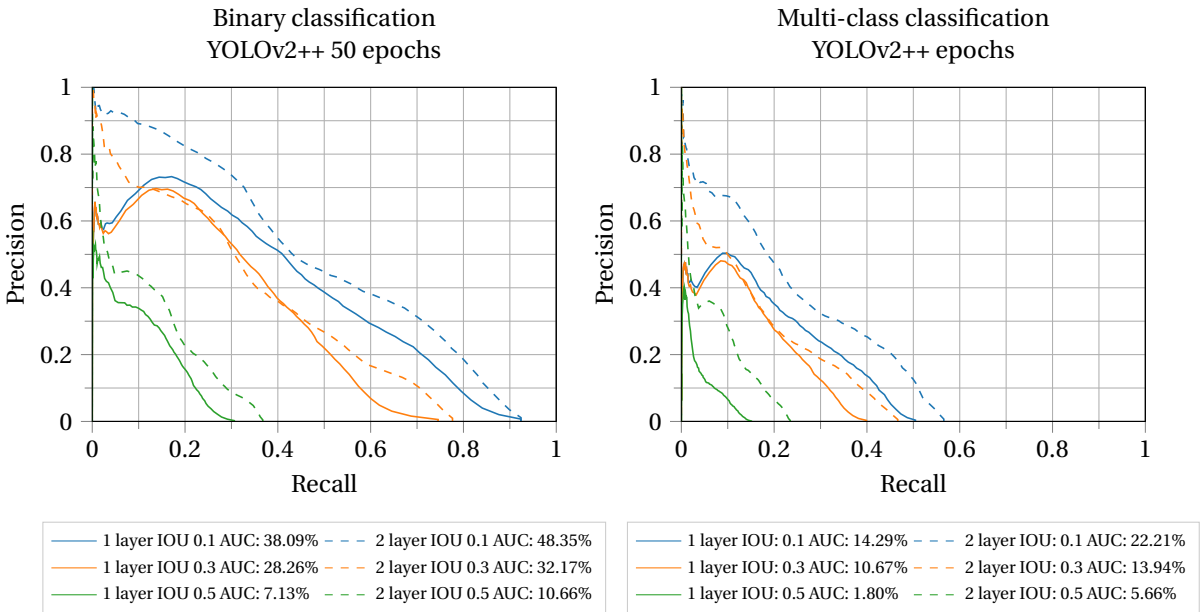
Table 4.3: Configurations of the different traffic light state detectors

	One layer configuration	**Two layer configuration**
Layer 1	1×1×60	1×1×512
Layer 2	n.a.	1×1×60

4.3.1. Performance of YOLOv2++

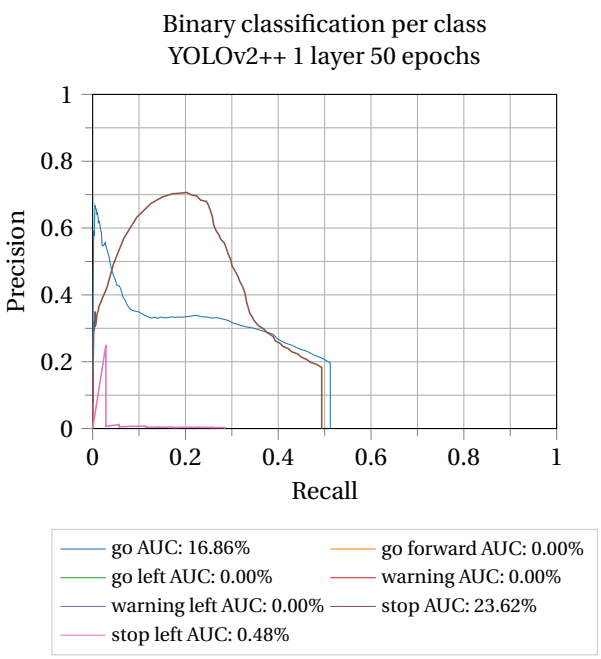
Figure 4.9a and 4.9b show the result of YOLOv2++ for binary classification and multi-class classification over different IOU's. The 2 layer configuration has consequently a higher AUC than the 1 layer configuration. The multi-class classification has half the AUC of the binary classification. The lower multi-class classification performance indicates YOLOv2++ can be improved for classification. For binary and multi-class classification the difference between different IOU's is significant. The localization of the bounding boxes is yet to be improved. On the positive note: for binary classification the recall of the system evaluated on IOU 0.1 of above 90%, which is important for a traffic light detection system in order to not miss any traffic light. The is low precision is the main disadvantage of YOLOv2++. For a decent traffic light detection system a high recall as well as a high precision is necessarily.

The performance per class of YOLOv2++ for the 1 and 2 layer configurations is shown in figure 4.10. The results for both configurations are similar with the biggest difference 10% for the *go* class in favor of the 2 layer configuration. Only the *stop* and *go* classes have noticeable detection performance. The *warning* class does not get detected by the YOLOv2++ system, which is a drawback compared to the adaptive combined for adaptive combined training. The confusion matrices reveals a lot of errors between the *stop* and *go* classes. The best performance of YOLOv2++ is around 25% lower than the traffic light recognition baseline for *go* and around 40% for *stop*.

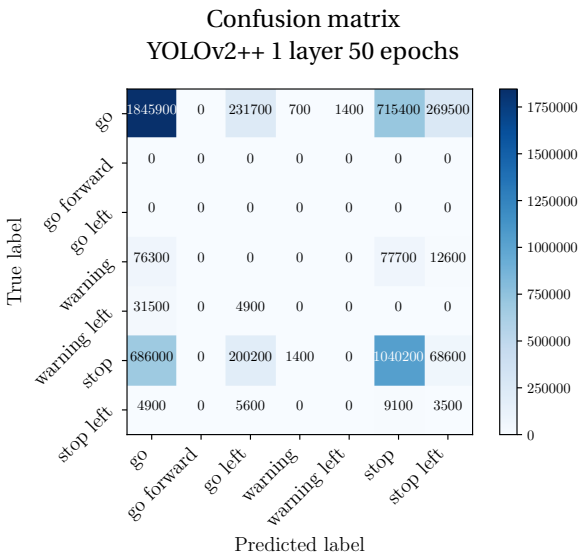


(a) (b)

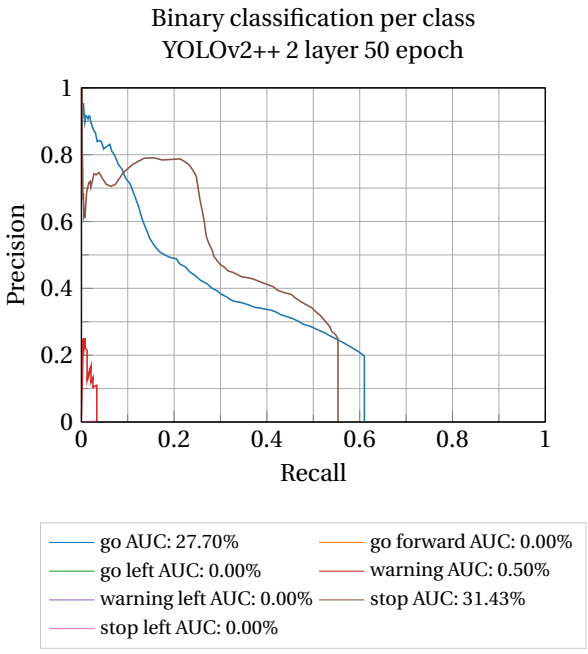
Figure 4.9: Precision recall curve for model trained for LISA traffic light states on feature map of LISA dayTrain for 50 epochs with 60×1×1 convolutional layer evaluated on LISA daySequence1 images. Input resolution 608×608 pixels. Evaluated on binary and multi-class case.



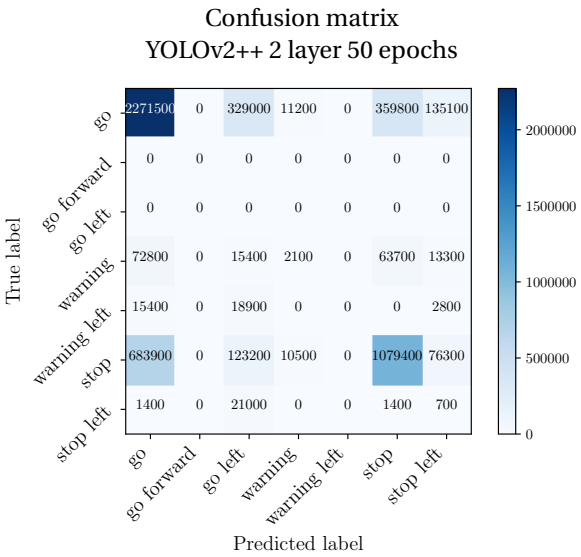
(a)



(b)



(c)



(d)

Figure 4.10: Precision recall curves and confusion matrix for YOLOv2++ 1 layer and 2 layer configurations trained for 50 epochs. Input resolution 608×608 pixels. IOU 0.01

4.3.2. Visual results YOLOv2++

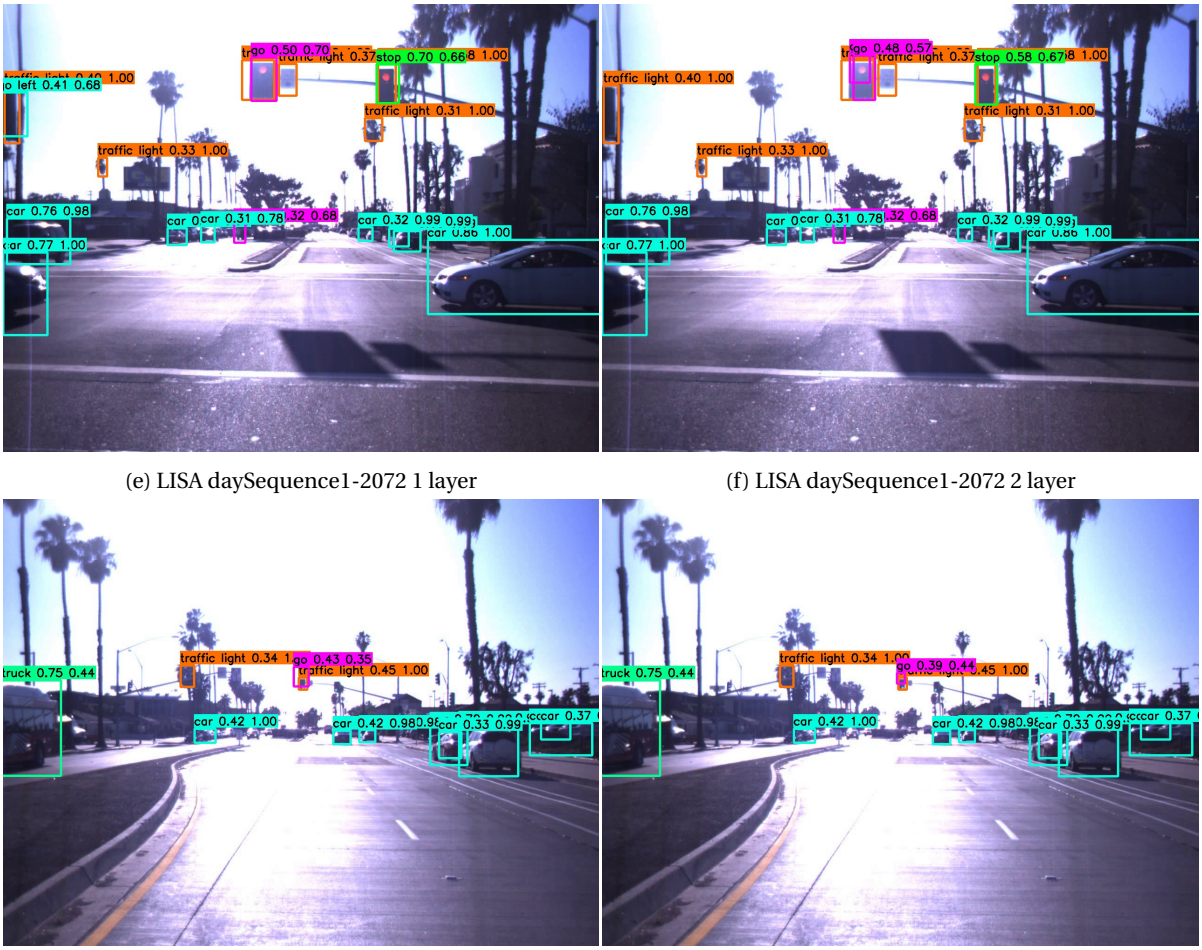
Visuals of the performance of YOLOv2++ on sample frames from the LISA test are in figure 4.11. In frame 1255, 2040 and 2072 the 2 layer configuration produces more detections than the 1 layer configuration, leading to more false positives in these frames for the 2 layer configuration. In frame 1255 is visible YOLOv2++ does detect the traffic light instances from the side and back, which is originated by the pre-trained weights used to train YOLOv2++. Comparing frame 1255 with 2040 and 2072 the localization of bounding boxes around the smaller traffic light instances is poor. On frame 2040 and 2072 the class prediction wrong for the middle traffic light, despite being easily human recognizable. Overall the YOLOv2++ detector has higher class confidence than object confidence, indicating during training a different balance between the class error and the localization error is needed for better training of YOLOv2++.

(a) LISA daySequence1-1255 1 layer

(b) LISA daySequence1-1255 2 layer

(c) LISA daySequence1-2040 1 layer

(d) LISA daySequence1-2040 2 layer



(e) LISA daySequence1-2072 1 layer

(f) LISA daySequence1-2072 2 layer

(g) LISA daySequence1-2760 1 layer

(h) LISA daySequence1-2760 2 layer

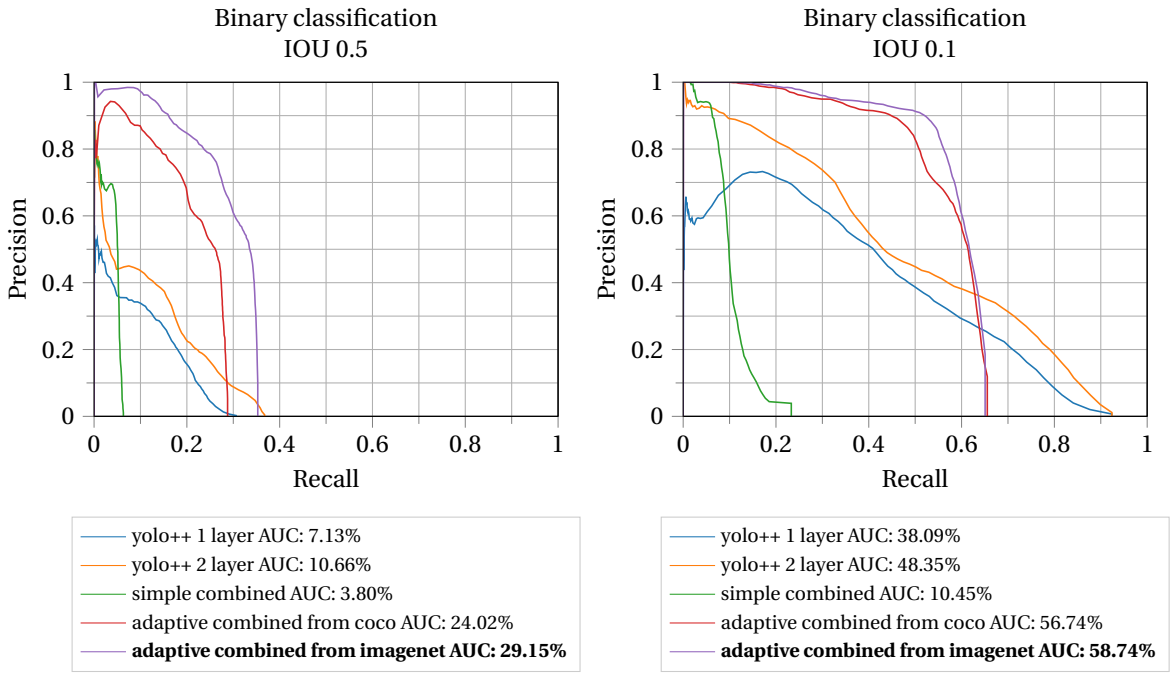
Figure 4.11: Sample frames of LISA daySequence1 trained with YOLOv2++ trained for 50 epochs. Detection of the traffic light state net are displayed combined with the detections of the default darknet configuration. The detection and class confidence scores are next to the predicted class.

4.4. Overview all systems

All systems are compared to evaluate the performance of all systems under equal conditions. The systems are compared for IOU 0.1 and 0.5. First the AUC's of the precision recall curves are provided in section 4.4.1, than a visual comparison is provided for the sample frames of the LISA test set in section 4.4.2.

4.4.1. Performance all systems

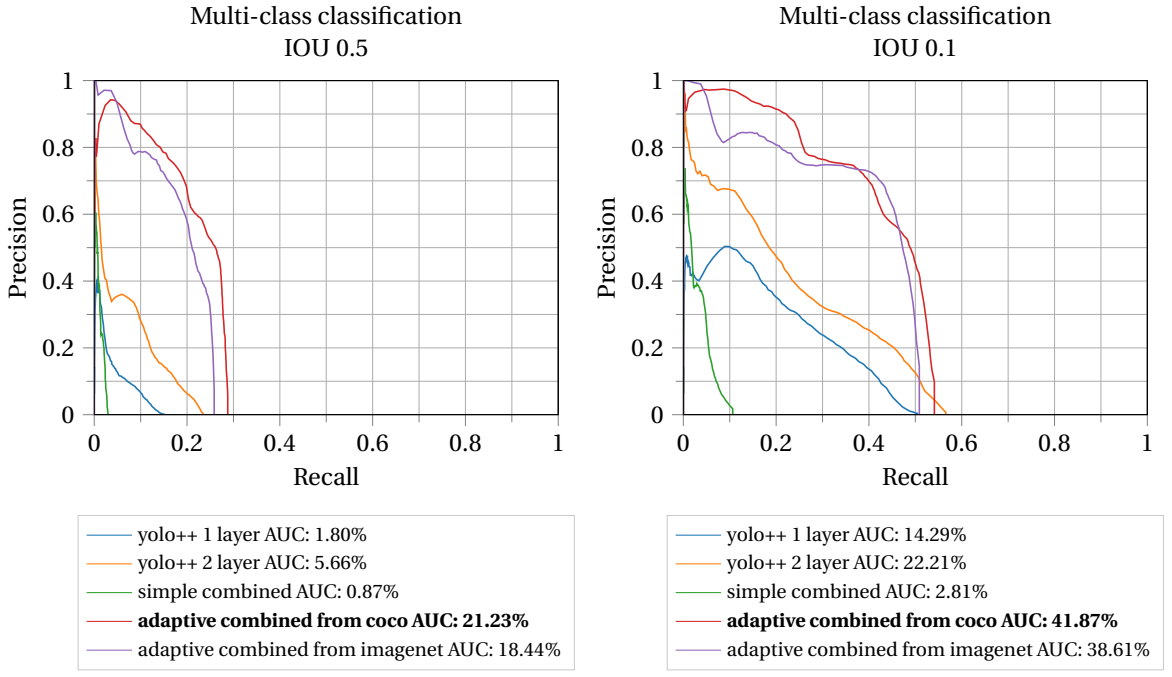
Figure 4.12 show the results for all tested systems for a IOU of 0.5 and 0.1. Overall a 30 to 50 percent increase of performance accomplished with the lower IOU, indicating overall localization inaccuracy for all systems. For binary classification adaptive combined training from ImageNet has slightly better performance than adaptive combined training from COCO, for multi-class classification the adaptive training system trained from COCO is slightly better. Both adaptive combined training systems outperform YOLOv2++ for all systems. YOLOv2++ perform especially worse compared to adaptive combined training for multi-class classification. The lowest performance is achieved by the simple combined training method. Compared to the binary classification baseline for IOU 0.1 the the adaptive combined methods outperform the baseline with around 5%, however for the same IOU of 0.5 as the baseline the performance is 20% lower. The average AUC of the traffic light recognition baseline is 47.95% on IOU 0.5, which is around 25% better than the best multiclass detection results of the adaptive combined systems.



(a) Overview comparison of all systems for IOU of 0.5

(b) Overview comparison of all systems for IOU of 0.1

Figure 4.12



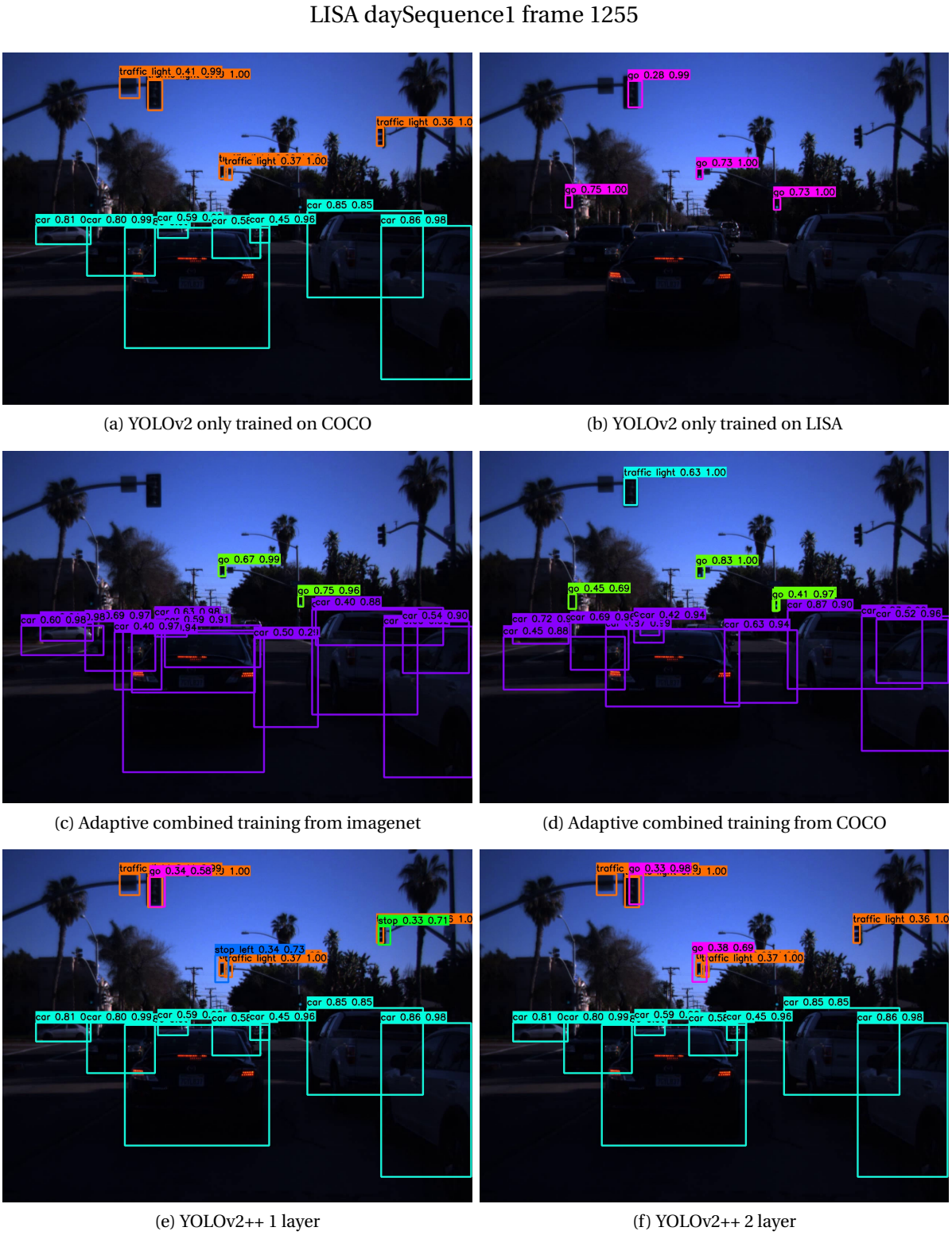
(a) Overview comparison of multi-class classification for all systems for IOU of 0.5

(b) Overview comparison of multi-class classification for all systems for IOU of 0.1

Figure 4.13

4.4.2. Overview visual results all systems

In figures 4.14 to 4.17 the detection results are displayed for the baseline methods of YOLOv2 only trained on COCO or LISA, as well as for the adaptive training methods and YOLOv2++ methods. The simple combined training method is not displayed because the results of this system are too low to be relevant for comparison. The detections of the adaptive combined training methods are more similar to the detections from the LISA traffic light of the system only trained on LISA compared to those of YOLOv2++. For traffic light recognition YOLOv2++ is especially outperformed by the adaptive combined methods for the traffic light far from the ego vehicle in frame 2760, where the adaptive trained methods detect the three *green* traffic lights (with a false positive for the farthest traffic light to the left) compared to YOLOv2++ methods only finding one traffic light instance. The YOLOv2++ methods make a misclassification for the *stop* traffic light in the middle of the scene, which is falsely classified as *go*. The adaptive combined training method does make a misclassification for this traffic light as well, detecting the traffic light as *warning*. The adaptive combined method from COCO does not make an error here. The general object detections of the adaptive combined training method pre-trained on COCO are similar to the detections from the YOLOv2 only trained on COCO, indicating the advantage of using the COCO pre-trained weights over the ImageNet pre-trained weights. However the adaptive combined method from COCO does not detect some *car* instances that are smaller and closer to other cars. YOLOv2++ has a better performance than the adaptive combined method from COCO by the use of the original network trained on COCO for general object detection used in YOLOv2++.



(a) YOLOv2 only trained on COCO

(b) YOLOv2 only trained on LISA

(c) Adaptive combined training from imagenet

(d) Adaptive combined training from COCO

(e) YOLOv2++ 1 layer

(f) YOLOv2++ 2 layer

Figure 4.14: Detections with all systems on test frame LISA daySequence 1255. Detected with confidence threshold 0.3.

(a) YOLOv2 only trained on COCO

(b) YOLOv2 only trained on LISA

(c) Adaptive combined training from imagenet

(d) Adaptive combined training from COCO

(e) YOLOv2++ 1 layer

(f) YOLOv2++ 2 layer

Figure 4.15: Detections with all systems on test frame LISA daySequence 2040. Detected with confidence threshold 0.3.

(a) YOLOv2 only trained on COCO

(b) YOLOv2 only trained on LISA

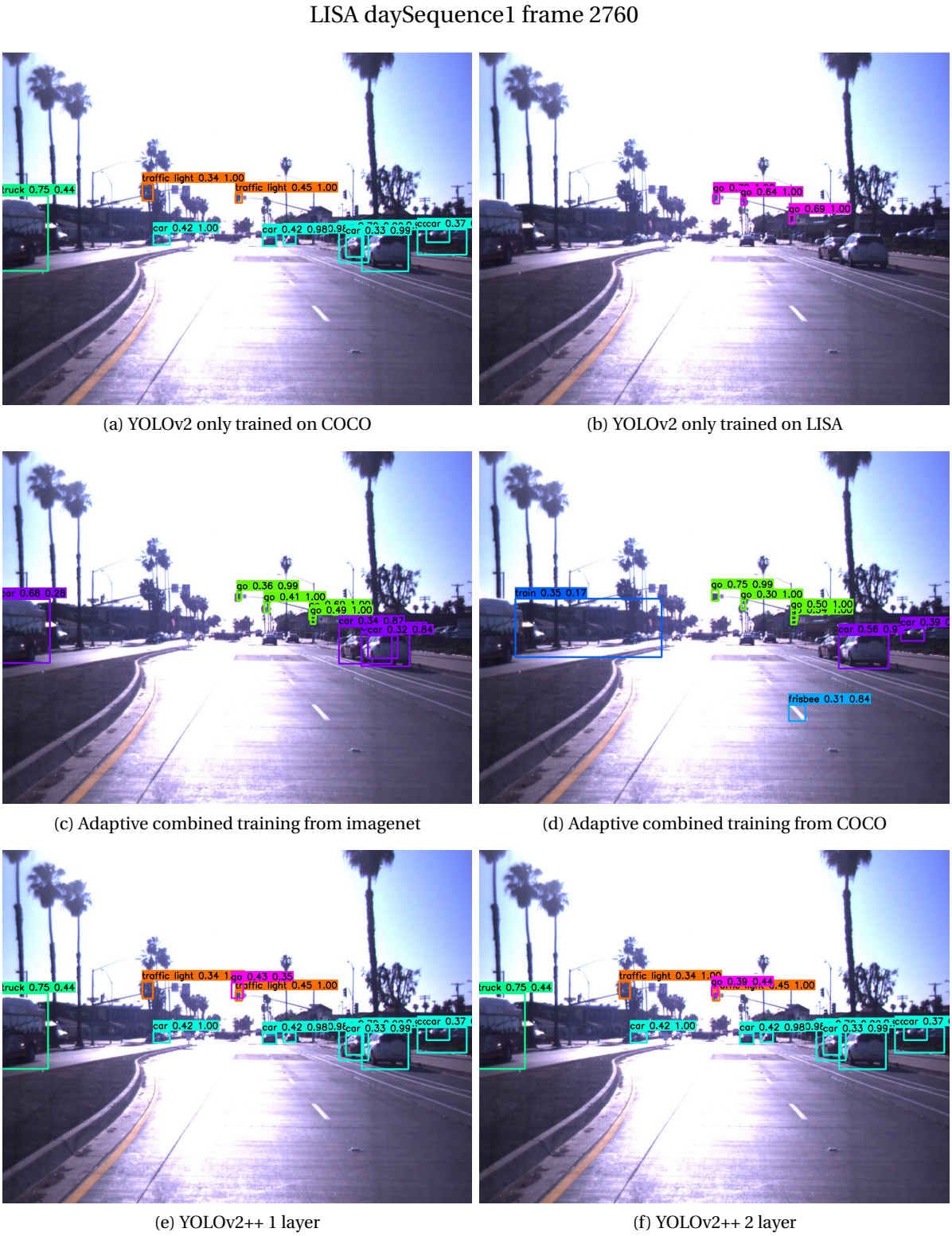
(c) Adaptive combined training from imagenet

(d) Adaptive combined training from COCO

(e) YOLOv2++ 1 layer

(f) YOLOv2++ 2 layer

Figure 4.16: Detections with all systems on test frame LISA daySequence 2072. Detected with confidence threshold 0.3.



(a) YOLOv2 only trained on COCO

(b) YOLOv2 only trained on LISA

(c) Adaptive combined training from imagenet

(d) Adaptive combined training from COCO

(e) YOLOv2++ 1 layer

(f) YOLOv2++ 2 layer

Figure 4.17: Detections with all systems on test frame LISA daySequence 2760. Detected with confidence threshold 0.3.

5

Conclusions

Two methods are tested in this work to find a good method to construct a detector able to simultaneously detect objects from a general object class and subclass dataset, using the COCO general object database with the LISA traffic light subclass database in the single shot object detector YOLOv2. The two systems are designed to answer the main research question and subquestions asked in section 1. The proposed methods are combined detection and YOLOv2++. The combined detection method trains YOLOv2 on both dataset at once, YOLOv2++ uses the base Darknet-19 network from YOLOv2 pre-trained on the general object class dataset as feature extractor for training of the subclass dataset.

5.1. How can subclass dataset traffic light states be combined with a general object dataset to train YOLOv2 on both sets?

In section 4.2 the results are shown of two different methods to combine the LISA traffic light states subclasses with the COCO general object classes: simple combined training and adaptive combined training. The results in 4.2.1 how that the simple combined training method results in system having some trouble with localization, as can be seen in the difference in performance for different IOU values. In the visual results on the test frames it is concluded the simple combined method suffers from many false negatives in combination with misclassfication and distinguishing errors between the general object traffic light class detection and the traffic light state classes. With these results it is concluded that the simple combined training method is unfitted to be used as a combined object detector for general object classes and subclasses. For the adaptive combined training methods the precision-recall plots in section 4.2.1 show the improvement of the adaptive combined training methods over the simple combined training method. The performance of the adaptive combined training methods is less dependent on the IOU, meaning the the detector is better at localization. Additionally, the multi-class performance is improved for the adaptive combined methods. Two variants of the adaptive combined training method are tested. The first using pre-trained weights on ImageNet and the second using pre-trained weights on COCO. On the LISA subclass test set both variations show similar performance. For binary classification the system pre-trained on ImageNet is a few percent better, especially for lower IOU, for multi-class classification the system trained on COCO has the advantage. However, table 4.2 shows system pre-trained on COCO has better performance for detecting COCO classes.

All results considered the preferred combined training method is the adaptive combined training method pre-trained on COCO. The results of this systems outperforms the other systems for multi-class classification on the LISA and COCO test set. The adaptive combined training method reaches with IOU 0.5 a AUC of 24.02% for binary and 21.23% for multi-class classification. For IOU of 0.1 this increases to 56.74% for binary and 41.87% for multi-class classification.

Comparing the combined training methods to the out-of-the-box system trained only on LISA the system only trained on LISA is better than the best combined training method. For the adaptive combined method pre-trained on COCO the binary classification for IOU 0.1 is 10% lower and the multi-class classification is 20% lower compared to the system only trained on LISA. For IOU 0.5 that is 25% and 20%. When comparing the combined training methods to the out-of-the box system only trained on COCO on the COCO test set the adaptive combined training system pre-trained on COCO loses around 5% on the mAP. Most performance is lost on the precision, with a difference of 12%. The The visual results on the test frames in section 4.4.2 show the combined training methods detect less general object class instances compared to out-of-the-box, with no big differences visible between the system pre-trained on ImagNet or COCO.

5.2. Is it possible to do traffic light state subclass detection based on the YOLOv2 feature map?

Section 4.3 investigates the second method of this work for including traffic light recognition in the general object detection with YOLOv2, which is YOLOv2++. In YOLOv2++ the convolutional neural network of YOLOv2, Darknet-19, produces a feature map on which a traffic light recognition system, called traffic light state net, is trained. In this system the detection of the general object classes is unaffected by the inclusion of traffic light state detection. YOLOv2++ is tested for two configurations, for a 1 and for a 2 layer traffic light state net. Section 4.3.1 shows the performance of the YOLOv2++ systems. The YOLOv2++ system performance is much dependent on IOU and multi-class classification is twice as low as binary classification. The 2 layer traffic light state net performs about 5% better for binary and multi-class classification than the 1 layer traffic light state net. For IOU of 0.5 the performance is low, especially compared to the out-of-the-box system only trained on LISA which is around 30% better for binary and multi-class detection.

The performance of YOLOv2++ is lower than the performance of the combined training methods, as is concluded in section 4.4. Only the simple combined training method, the poorest performing combined training method, is outperformed by the YOLOv2++ systems. Although YOLOv2++ has a higher recall up to 90%, for low IOU, compared to combined training reaching around 70% recall at best, the overall AUC of YOLOv2++ is lower. The man difference is in the lower precision of YOLOv2++. Combined training has a precision above 95% in the best case, compared to at best 80% for YOLOv2++, which drops for higher recall faster compared to the combined training methods. However, for low IOU of 0.1, the maximum recall of YOLOv2++ is higher than the system trained only on LISA.

The visual results concludes that YOLOv2++ with 2 layer traffic light state net produces more false positives than the 1 layer traffic light state net. Both systems have do misclassify the traffic light states often, in particular the *stop* and *go* classes are confused too often. On smaller traffic light further away from the ego vehicle the most detections are missed for YOLOv2++.

5.3. What is a good way to make an object detector with good performance on both COCO and LISA classes?

Comparing both methods tested in this work for making a combined detector on general object classes and subclasses out of the COCO and LISA datasets, combined training and YOLOv2++, it can be concluded that the best combined performance is achieved using the adaptive combined training method pre-trained on COCO. Although YOLOv2++ has a slightly higher recall at low precision and better performance on COCO, the overall performance of the combined training method on the LISA test set, especially for multi-class classification, makes the adaptive combined training method pre-trained on COCO the best way to make an object detector with good performance on both COCO and LISA classes. The performance of all the combined detection systems have a dependency on the IOU, explained by the inaccurate labeling within the LISA dataset and the detectors having trouble with localization. Noted is that the combined detector does not outperform a detector only trained on COCO or LISA. However, the adaptive combined training methods gets within 10 to 20 percent of the systems trained on a single dataset, with one run of the network. For lower IOU values combined detection on the general object classes and subclasses reaches competitive performance compared to the detection on general object classes and subclasses individually.

5.4. Future work

In this work methods are proposed and tested to combine general object detection and subclass detection for the specific case of COCO general objects and LISA traffic lights. Upon this research future work can be done in the following directions.

- None of the proposed systems in this work do include tracking. The influence of adding a tracking algorithm to achieve on the stability of detections and the performance is still to be investigated on these systems.
- For improved traffic light recognition with the proposed system the LISA dataset can be replace by the recent DriveU traffic light dataset [27]. The DriveU dataset is more consistently labeled, more even distributed, and larger than the LISA traffic light dataset. The DriveU dataset has not been available till after the start of this investigation and is therefore not used, although the dataset looks promising for achieving increased performance of traffic light recognition for machine learning based methods.

- The YOLOv2 base framework can be replaced by other networks like the more recent YOLOv3 [70], SSD [51], or MDSSD [16]. Some tweaking is necessarily to apply the YOLOv2++ method to these networks, because these networks do not have the single feature map representation from YOLOv2, but extract feature maps at different layers of the network. The combined training methods can be applied directly.
- Subclass detection combined with general object class detection with the proposed system can be extended to different detection problems, including traffic sign recognition, pedestrian pose recognition, or detection of any other subclass.

Bibliography

[1] CUDA Toolkit | NVIDIA Developer. URL https://developer.nvidia.com/cuda-toolkit.

[2] Traffic Lights Recognition (TLR) public benchmarks [La Route Automatisée]. URL http://www.lara.prd.fr/benchmarks/trafficlightsrecognition.

[3] Explainer: Betting on the past? Europe decides on connected car standards - Reuters. URL https://www.reuters.com/article/us-eu-autos-technology-explainer/explainer-betting-on-the-past-europe-decides-on-connected-car-standards-idUSKCN1RU214.

[4] GitHub - AlexeyAB/darknet: Windows and Linux version of Darknet Yolo v3 & v2 Neural Networks for object detection (Tensor Cores are used), . URL https://github.com/AlexeyAB/darknet.

[5] Part 4 Object Detection using YOLOv2 on Pascal VOC2012 - loss, . URL https://fairyonice.github.io/Part_4_Object_Detection_with_Yolo_using_VOC_2012_data_loss.html.

[6] COCO - Common Objects in Context. URL http://cocodataset.org/#download.

[7] PyTorch. URL https://pytorch.org/.

[8] M. Bach, D. Stumper, and K. Dietmayer. Deep convolutional traffic light recognition for automated driving. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 851–858, Nov 2018. doi: 10.1109/ITSC.2018.8569522.

[9] Martin Bach, Stephan Reuter, and Klaus Dietmayer. Multi-camera traffic light recognition using a classifying Labeled Multi-Bernoulli filter. In *IEEE Intelligent Vehicles Symposium, Proceedings*, pages 1045–1051, 2017. ISBN 9781509048045. doi: 10.1109/IVS.2017.7995852.

[10] Karsten Behrendt, Libor Novak, and Rami Botros. A deep learning approach to traffic lights: Detection, tracking, and classification. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1370–1377. IEEE, may 2017. ISBN 978-1-5090-4633-1. doi: 10.1109/ICRA.2017.7989163. URL http://ieeexplore.ieee.org/document/7989163/.

[11] Jeffrey L. Binangkit and Dwi H. Widyantoro. Increasing accuracy of traffic light color detection and recognition using machine learning. *2016 10th International Conference on Telecommunication Systems Services and Applications (TSSA)*, 2016. doi: 10.1109/tssa.2016.7871074.

[12] Francisco M. Castro, Manuel J. Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-End Incremental Learning. jul 2018. URL http://arxiv.org/abs/1807.09536.

[13] Zhilu Chen and Xinming Huang. Accurate and Reliable Detection of Traffic Lights Using Multiclass Learning and Multiobject Tracking. *IEEE Intelligent Transportation Systems Magazine*, 8(4):28–42, 2016. ISSN 1939-1390. doi: 10.1109/MITS.2016.2605381. URL http://ieeexplore.ieee.org/document/7637094/.

[14] Zhilu Chen, Quan Shi, and Xinming Huang. Automatic detection of traffic lights using support vector machine. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pages 37–40. IEEE, jun 2015. ISBN 978-1-4673-7266-4. doi: 10.1109/IVS.2015.7225659. URL http://ieeexplore.ieee.org/document/7225659/.

[15] E-H Choi. Crash Factors in Intersection-Related Crashes: An On-Scene Perspective. *NHTSA's National Center for Statistics and Analysis*, (September):37, 2010. doi: http://dx.doi.org/10.1037/e621942011-001. URL https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/811366.

[16] Lisha Cui. MDSSD: Multi-scale Deconvolutional Single Shot Detector for small objects. 2018. URL https://arxiv.org/pdf/1805.07009.pdf.

[17] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-FCN: Object Detection via Region-based Fully Convolutional Networks. may 2016. URL http://arxiv.org/abs/1605.06409.

[18] Navneet Dalal and Bill Triggs. HOG. In *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, 2005. ISBN 0769523722. doi: 10.1109/CVPR.2005.177.

[19] Will Maddern Dan Barnes and Ingmar Posner. Exploiting 3d semantic scene priors for online traffic light interpretation. *2015 IEEE Intelligent Vehicles Symposium (IV) June 28 - July 1, 2015. COEX, Seoul, Korea*, 2015.

[20] Jesse Davis and Mark Goadrich. The relationship between Precision-Recall and ROC curves. In *Proceedings of the 23rd international conference on Machine learning - ICML '06*, 2006. ISBN 1595933832. doi: 10.1145/1143844.1143874.

[21] Raoul De Charette and Fawzi Nashashibi. Real time visual traffic lights recognition based on spot light detection and adaptive traffic lights templates. In *IEEE Intelligent Vehicles Symposium, Proceedings*, 2009. ISBN 9781424435043. doi: 10.1109/IVS.2009.5164304.

[22] Mark Everingham, Luc Van Gool, Christopher K.I. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 2010. ISSN 09205691. doi: 10.1007/s11263-009-0275-4.

[23] Elias C. Eze, Sijing Zhang, and Enjie Liu. Vehicular ad hoc networks (VANETs): Current state, challenges, potentials and way forward. In *2014 20th International Conference on Automation and Computing*, pages 176–181. IEEE, sep 2014. ISBN 978-1-9095-2202-2. doi: 10.1109/IConAC.2014.6935482. URL http://ieeexplore.ieee.org/document/6935482/.

[24] Laboratory for Intelligent and UC San Diego Save Automobiles. Vision for intelligent vehicles adn applications (viva) challenge, 2015. URL http://cvrr.ucsd.edu/vivachallenge/.

[25] Uwe Franke, Dariu Gavrila, Steffen Görzig, Frank Lindner, Frank Paetzold, and Christian Wöhler. Autonomous driving goes downtown. *IEEE Intelligent Systems and Their Applications*, 1998. ISSN 10947167. doi: 10.1109/5254.736001.

[26] Andreas Fregin, Julian Muller, and Klaus Dietmayer. Three ways of using stereo vision for traffic light recognition. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 430–436. IEEE, jun 2017. ISBN 978-1-5090-4804-5. doi: 10.1109/IVS.2017.7995756. URL http://ieeexplore.ieee.org/document/7995756/.

[27] Andreas Fregin, Julian Muller, Ulrich Krebel, and Klaus Dietmayer. The DriveU Traffic Light Dataset: Introduction and Comparison with Existing Datasets. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3376–3383. IEEE, may 2018. ISBN 978-1-5386-3081-5. doi: 10.1109/ICRA.2018.8460737. URL https://ieeexplore.ieee.org/document/8460737/.

[28] Cheng-Yang Fu, Wei Liu, Ananth Ranga, Ambrish Tyagi, and Alexander C. Berg. DSSD : Deconvolutional Single Shot Detector. jan 2017. URL http://arxiv.org/abs/1701.06659.

[29] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. The KITTI Vision Benchmark Suite, 2013. URL http://www.cvlibs.net/datasets/kitti/eval{_}stereo{_}flow.php?benchmark=stereo.

[30] Ross Girshick. Fast R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision*, 2015. ISBN 9781467383912. doi: 10.1109/ICCV.2015.169.

[31] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. nov 2013. URL http://arxiv.org/abs/1311.2524.

[32] Byung Kwan PARK Gwang-Gook. LEE. Traffic light recognition using deep neural networks. *2017 IEEE International Conference on Consumer Electronics (ICCE)*, 2017.

[33] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. dec 2015. URL http://arxiv.org/abs/1512.03385.

[34] Siavash Hosseinyalamdary and Alper Yilmaz. A Bayesian approach to traffic light detection and mapping. *ISPRS Journal of Photogrammetry and Remote Sensing*, 125:184–192, mar 2017. ISSN 0924-2716. doi: 10.1016/J.ISPRSJPRS.2017.01.008. URL http://www.sciencedirect.com/science/article/pii/S092427161730028X.

[35] Xiaodi Hou and Liqing Zhang. Saliency detection: A spectral residual approach. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2007. ISBN 1424411807. doi: 10.1109/CVPR.2007.383267.

[36] Chulhoon Jang, Sungjin Cho, Sangwoo Jeong, Jae Kyu Suhr, Ho Gi Jung, and Myoungho Sunwoo. Traffic light recognition exploiting map and localization at every stage. *Expert Systems with Applications*, 88:290–304, dec 2017. ISSN 0957-4174. doi: 10.1016/J.ESWA.2017.07.003. URL https://www.sciencedirect.com/science/article/pii/S0957417417304724.

[37] Morten B. Jensen, Mark P. Philipsen, Chris Bahnsen, Andreas Møgelmose, Thomas B. Moeslund, and Mohan M. Trivedi. Traffic Light Detection at Night: Comparison of a Learning-Based Detector and Three Model-Based Detectors. pages 774–783, dec 2015. doi: 10.1007/978-3-319-27857-5_69.

[38] Philipsen M. P. Møgelmose A. Moeslund T. B. & Trivedi M. M. Jensen, M. B. Vision for looking at traffic lights: Issues, survey, and perspectives. *I E E E Transactions on Intelligent Transportation Systems, 17(7)*, 2016.

[39] Jeongmin Jeon, Sung-Ho Hwang, and Hyungpil Moon. Monocular vision-based object recognition for autonomous vehicle driving in a real driving environment. In *2016 13th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, pages 393–399. IEEE, aug 2016. ISBN 978-1-5090-0821-6. doi: 10.1109/URAI.2016.7734068. URL http://ieeexplore.ieee.org/document/7734068/.

[40] Yang Ji, Ming Yang, Zhengchen Lu, and Chunxiang Wang. Integrating visual selective attention model with HOG features for traffic light detection and recognition. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pages 280–285. IEEE, jun 2015. ISBN 978-1-4673-7266-4. doi: 10.1109/IVS.2015.7225699. URL http://ieeexplore.ieee.org/document/7225699/.

[41] Jia Deng, Wei Dong, R. Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009. ISBN 978-1-4244-3992-8. doi: 10.1109/CVPRW.2009.5206848.

[42] Vijay John, Keisuke Yoneda, Zheng Liu, and Seiichi Mita. Saliency Map Generation by the Convolutional Neural Network for Real-Time Traffic Light Detection Using Template Matching. *IEEE Transactions on Computational Imaging*, 1(3):159–173, 2015. ISSN 2333-9403. doi: 10.1109/TCI.2015.2480006. URL http://ieeexplore.ieee.org/document/7272062/.

[43] Vijay John, Lyu Zheming, and Seiichi Mita. Robust traffic light and arrow detection using optimal camera parameters and GPS-based priors. In *Proceedings of 2016 Asia-Pacific Conference on Intelligent Robot Systems, ACIRS 2016*, pages 204–208, 2016. ISBN 9781509013623. doi: 10.1109/ACIRS.2016.7556213.

[44] Hyun-Koo Kim, Ju H. Park, and Ho-Youl Jung. An Efficient Color Space for Deep-Learning Based Traffic Light Recognition. *Journal of Advanced Transportation*, 2018:1–12, dec 2018. ISSN 0197-6729. doi: 10.1155/2018/2365414. URL https://www.hindawi.com/journals/jat/2018/2365414/.

[45] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. *ImageNet Classification with Deep Convolutional Neural Networks*, 2012. ISSN 10495258. doi: 10.1145/3065386.

[46] Cesar Laurent, Gabriel Pereyra, Philemon Brakel, Ying Zhang, and Yoshua Bengio. Batch normalized recurrent neural networks. In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2016. ISBN 9781479999880. doi: 10.1109/ICASSP.2016.7472159.

[47] Xi Li, Huimin Ma, Xiang Wang, and Xiaoqin Zhang. Traffic Light Recognition for Complex Scene With Fusion Detections, 2017. ISSN 15249050.

[48] Tsung Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2014. ISBN 978-3-319-10601-4. doi: 10.1007/978-3-319-10602-1_48.

[49] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature Pyramid Networks for Object Detection. dec 2016. URL http://arxiv.org/abs/1612.03144.

[50] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal Loss for Dense Object Detection. aug 2017. URL http://arxiv.org/abs/1708.02002.

[51] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng Yang Fu, and Alexander C. Berg. SSD. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2016. ISSN 16113349. doi: 10.1007/978-3-319-46448-0_2.

[52] Wei Liu, Shuang Li, Jin Lv, Bing Yu, Ting Zhou, Huai Yuan, and Hong Zhao. Real-Time Traffic Light Recognition Based on Smartphone Platforms. *IEEE Transactions on Circuits and Systems for Video Technology*, 27(5):1118–1131, may 2017. ISSN 1051-8215. doi: 10.1109/TCSVT.2016.2515338. URL http://ieeexplore.ieee.org/document/7373593/.

[53] Bruce D. Lucas and Takeo Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. *Robotics*, 1981. ISSN 17486815. doi: 10.1109/HPDC.2004.1323531.

[54] Andreas Møgelmose Thomas B. Moeslund Mark P. Philipsen, Morten B. Jensen and Mohan M. Trivedi. Traffic light detection: A learning algorithm and evaluations on challenging dataset. *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, 2015.

[55] Mohan M. Trivedi Andreas Møgelmose Mark P. Philipsen, Morten B. Jensen. Ongoing work on traffic lights: Detection and evaluation. *Advanced Video and Signal Based Surveillance (AVSS), 2015 12th IEEE International Conference on*, 2015.

[56] Panagiotis Meletis and Gijs Dubbelman. Training of Convolutional Networks on Multiple Heterogeneous Datasets for Street Scene Semantic Segmentation. mar 2018. URL http://arxiv.org/abs/1803.05675.

[57] Matthias Michael and Marc Schlipsing. Extending traffic light recognition: Efficient classification of phase and pictogram. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, jul 2015. ISBN 978-1-4799-1960-4. doi: 10.1109/IJCNN.2015.7280499. URL http://ieeexplore.ieee.org/document/7280499/.

[58] Giuseppe Pirlo Miguel A. Ferrer Moises Diaz, Pietro Cerri and Donato Impedovo. A survey on traffic light detection. *V. Murino et al. (Eds.): ICIAP 2015 Workshops, LNCS 9281, pp. 201–208*, 2015.

[59] Hiroki Moizumi, Yoshihiro Sugaya, Masako Omachi, and Shinichiro Omachi. Traffic Light Detection Considering Color Saturation Using In-Vehicle Stereo Camera. *Journal of Information Processing*, 24(2): 349–357, 2016. ISSN 1882-6652. doi: 10.2197/ipsjjip.24.349. URL https://www.jstage.jst.go.jp/article/ipsjjip/24/2/24{_}349/{_}article.

[60] Kamal Nasrollahi Morten B. Jensen and Thomas B. Moeslund. Evaluating state-of-the-art object detector on challenging traffic light data. *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops: Traffic Surveillance Workshop and Challenge*, 2016.

[61] Guo Mu, Zhang Xinyu, Li Deyi, Zhang Tianlei, and An Lifeng. Traffic light detection and recognition for autonomous vehicles. *The Journal of China Universities of Posts and Telecommunications*, 22(1):50–56, 2015. ISSN 10058885. doi: 10.1016/S1005-8885(15)60624-0. URL http://linkinghub.elsevier.com/retrieve/pii/S1005888515606240.

[62] Julian Müller and Klaus Dietmayer. Detecting Traffic Lights by Single Shot Detection. may 2018. URL http://arxiv.org/abs/1805.02523.

[63] National Highway Traffic Safety Administration and US National Department of Transportation. Traffic Safety Facts - 2008 Data (DOT HS 811 170). 2008.

[64] Florin Oniga, Sergiu Prodan, and Sergiu Nedevschi. Traffic light detection on mobile devices. In *2015 IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*, pages 287–292, 2015. ISBN 978-1-4673-8200-7. doi: 10.1109/ICCP.2015.7312645. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7312645.

[65] Ziya Ozcelik, Canan Tastimur, Mehmet Karakose, and Erhan Akin. A vision based traffic light detection and recognition approach for intelligent vehicles. In *2017 International Conference on Computer Science and Engineering (UBMK)*, pages 424–429. IEEE, oct 2017. ISBN 978-1-5386-0930-9. doi: 10.1109/UBMK.2017.8093430. URL http://ieeexplore.ieee.org/document/8093430/.

[66] Md) R E Kalman (Reserach Institute for Advanced Study, Baltimore. A New Approach to Linear Filtering and Prediciton Problems. *Journal of Basic Engineering*, 1960.

[67] Joseph Redmon. Darknet: Open source neural networks in c. http://pjreddie.com/darknet/, 2013–2016.

[68] Joseph Redmon and Ali Farhadi. YOLO9000: better, faster, stronger. *CoRR*, abs/1612.08242, 2016. URL http://arxiv.org/abs/1612.08242.

[69] Joseph Redmon and Ali Farhadi. YOLOv2, 2016. URL www.pjreddie.com/darknet/yolov2.

[70] Joseph Redmon and Ali Farhadi. YOLOv3: An Incremental Improvement. 2018. ISSN 0146-4833. doi: 10.1109/CVPR.2017.690. URL http://arxiv.org/abs/1804.02767.

[71] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *Cvpr 2016*, 2016. ISSN 10636919. doi: 10.1016/j.nima.2015.05.028.

[72] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015. ISSN 01628828. doi: 10.1109/TPAMI.2016.2577031.

[73] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 2015. ISSN 15731405. doi: 10.1007/s11263-015-0816-y.

[74] G Russo, E Baccaglini, L Boulard, D Brevi, and R Scopigno. Video processing for V2V communications: A case study with traffic lights and plate recognition. *2015 IEEE 1st International Forum on Research and Technologies for Society and Industry Leveraging a better tomorrow (RTSI)*, pages 144–148, 2015. doi: 10.1109/RTSI.2015.7325064. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7325064.

[75] Asaad F. Said, Mehrnaz Kh. Hazrati, and Farshad Akhbari. Real-time detection and classification of traffic light signals. In *2016 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, pages 1–5. IEEE, oct 2016. ISBN 978-1-5090-3284-6. doi: 10.1109/AIPR.2016.8010547. URL http://ieeexplore.ieee.org/document/8010547/.

[76] Sanjay Saini, S Nikhil, Krishna Reddy Konda, Harish S Bharadwaj, and N Ganeshan. An efficient vision-based traffic light detection and state recognition for autonomous vehicles. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 606–611. IEEE, jun 2017. ISBN 978-1-5090-4804-5. doi: 10.1109/IVS.2017.7995785. URL http://ieeexplore.ieee.org/document/7995785/.

[77] Mehdi Salarian, Andrea Manavella, and Rashid Ansari. A vision based system for traffic lights recognition. In *2015 SAI Intelligent Systems Conference (IntelliSys)*, pages 747–753. IEEE, nov 2015. ISBN 978-1-4673-7606-8. doi: 10.1109/IntelliSys.2015.7361224. URL http://ieeexplore.ieee.org/document/7361224/.

[78] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. sep 2014. URL http://arxiv.org/abs/1409.1556.

[79] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015. ISBN 9781467369640. doi: 10.1109/CVPR.2015.7298594.

[80] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. dec 2015. URL http://arxiv.org/abs/1512.00567.

[81] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. feb 2016. URL http://arxiv.org/abs/1602.07261.

[82] Christian Unger Vladimir Haltakov, Jakob Mayr and Slobodan Ilic. Semantic segmentation based traffic light detection at day and at night. *J. Gall et al. (Eds.): GCPR 2015, LNCS 9358, pp. 446–457, 2015*, 2015.

[83] Winson Waisakurnia and Dwi H. Widyantoro. Traffic light candidate elimination based on position. In *2016 10th International Conference on Telecommunication Systems Services and Applications (TSSA)*, pages 1–5. IEEE, oct 2016. ISBN 978-1-5090-5170-0. doi: 10.1109/TSSA.2016.7871077. URL http://ieeexplore.ieee.org/document/7871077/.

[84] Di Wang, Hong Bao, and Feifei Zhang. CTL-DNNet: Effective Circular Traffic Light Recognition with a Deep Neural Network. *International Journal of Pattern Recognition and Artificial Intelligence*, 31(11):1750037, 2017. ISSN 0218-0014. doi: 10.1142/S0218001417500379. URL http://www.worldscientific.com/doi/pdf/10.1142/S0218001417500379http://www.worldscientific.com/doi/abs/10.1142/S0218001417500379.

[85] Wenhao Wang, Shanlin Sun, Mingxin Jiang, Yunyang Yan, and Xiaobing Chen. Traffic lights detection and recognition based on multi-feature fusion. *Multimedia Tools and Applications*, 76(13):14829–14846, 2017. ISSN 15737721. doi: 10.1007/s11042-016-4051-5.

[86] Zhaojing Wang, Zhuo Bi, Cheng Wang, Lan Lin, and Hui Wang. Traffic lights recognition based on PCANet. In *Proceedings - 2015 Chinese Automation Congress, CAC 2015*, pages 559–564, 2016. ISBN 9781467371896. doi: 10.1109/CAC.2015.7382563.

[87] Michael Weber, Peter Wolf, and J. Marius Zollner. DeepTLR: A single deep convolutional network for detection and classification of traffic lights. In *2016 IEEE Intelligent Vehicles Symposium (IV)*, pages 342–348. IEEE, jun 2016. ISBN 978-1-5090-1821-5. doi: 10.1109/IVS.2016.7535408. URL http://ieeexplore.ieee.org/document/7535408/.

[88] Dwi H. Widyantoro and Kevin I. Saputra. Traffic lights detection and recognition based on color segmentation and circle hough transform. In *2015 International Conference on Data and Software Engineering (ICoDSE)*, pages 237–240, 2015. ISBN 978-1-4673-8428-5. doi: 10.1109/ICODSE.2015.7437004. URL http://ieeexplore.ieee.org/document/7437004/.

[89] I Yass and R A C Foundation. Every second counts: choices in the operation of traffic lights. (February): 35p, 2011.

[90] YOLOv2 PyTorch. GitHub - marvis/pytorch-yolo2: Convert https://pjreddie.com/darknet/yolo/ into pytorch, 2017. URL https://github.com/marvis/pytorch-yolo2.

[91] IEEE Zhengxia Zou Zhenwei Shi, Member and IEEE Changshui Zhang, Member. Real-time traffic light detection with adaptive background suppression filter. *IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, VOL. 17, NO. 3, MARCH 2016*, 2016.

[92] Mu Zhu. Recall, precision and average precision. *Department of Statistics and Actuarial Science, ...*, 2004.