Traffic Signs Detection and Classification for European Urban Environments

Citlalli Gámez Serna, and Yassine Ruichek

***Abstract*—Traffic signs play an important role for Advanced Driver Assistance Systems (ADAS) as well as for autonomous driving vehicles. Most of the works done focus on recognizing symbol based signs leaving apart important information provided by other type of signs like complementary panels or text based signs. In this paper, we include detection and classification of both symbol and text based signs focusing on the most common ones found in European urban environments. The system consists of two stages, traffic sign detection and traffic sign classification. The detection is performed using Mask R-CNN while the classification is achieved with a proposed Convolutional Neural Network (CNN) architecture. We introduced the extended version of the German Traffic Sign Detection Benchmark (GTSDB), labeled in a pixel manner (masks) with 164 classes grouped into 9 categories. It is used for the detection and classification steps. Experimental results on German environments show that our proposed system is capable of detecting all categories of traffic signs while at the same time recognizing them with high accuracy achieving comparable performance with the state of the art.**

***Index Terms*—Convolutional neural networks, Mask R-CNN, Traffic sign classification, Traffic sign detection, Traffic signs in urban environments.**

I. Introduction

COMPUTER vision aided systems have been widely used during the past years to solve different kinds of problems. Among them reside the ones for detection and recognition of certain context-relevant objects in complex environments. Recently, deep learning methods, specially CNNs have shown significant progress to solve localization and classification [1].

One challenging real-world problem for autonomous driving systems is traffic sign recognition. In theory, traffic signs are designed to be easily detectable and recognizable because they follow standard shapes and colors [2]. Nevertheless, as they are placed in outside environments, weather conditions, illumination variations, perspectives, occlusions, aging among others, are some of the variations that make traffic sign recognition a complex task to solve. In addition, as traffic signs appear in a small portion on the image, their small sizes make them harder to localize. For example, if a traffic sign is 16×16 pixels in an image size of 1360×1024, it represents less than 0.1% and some methods might fail. This issue is very important to consider for traffic sign recognition in order to 1) find the right location and sizes of the signs (traffic sign detection) and 2) assign the correct labels to them (traffic sign classification).

Manuscript received January 31, 2019; revise on February XXX, 2019. This work was supported by the Mexican National Council of Science and Technology (CONACYT).

C. Gámez Serna and Y. Ruichek are at CIAD, University Bourgogne Franche-Comté, UTBM, F-90010 Belfort, France. E-mails: (citlalli.gamez-serna@utbm.fr, yassine.ruichek@utbm.fr).

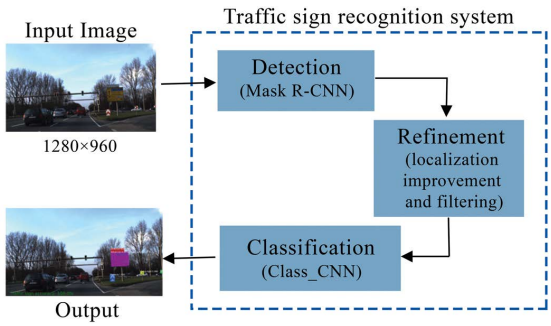


Fig. 1. Overview of our proposed traffic sign recognition system.

The detection normally shares features to locate the signs through bounding boxes while the classification focuses on differentiating certain features for each specific traffic class. These tasks have been widely studied around the world proposing specific methods and datasets [3]–[12]. Nonetheless, as there was not a common dataset to use, it was not possible to compare methods until the German Traffic Sign Recognition Benchmark (GTSRB) [8] and the German Traffic Sign Detection Benchmark (GTSDB) [13] were proposed in 2011 and 2013 respectively. These datasets are labeled with 43 symbol-based classes (use for classification tasks) grouped into 4 categories (use for detection tasks).

Current detection and classification approaches achieve near perfect results with 100% precision-recall Area Under the Curve (AUC) in 3 categories on the GTSDB [14], and 99.75% accuracy on GTSRB dataset. However, these benchmarks are not representative for real tasks since: 1) they include only symbol-based signs with standard shapes and colors discarding other important and challenging signs based on text with very different aspect ratios; 2) they do not take into account the intra variability of traffic sign classes to deal with the cases where symbols or inscriptions vary a little; and 3) methods trained with them will only succeed to recognize traffic signs of one country (Germany).

In order to alleviate the previously mentioned problems, we propose a traffic sign recognition system capable of detecting and identifying small and big signs, symbol and text based signs and multi-country traffic signs using a proposed extended version of the GTSDB and a urban/rural subset of traffic signs carefully chosen from the European Traffic Sign Dataset (ETSD) [15].

Our traffic sign recognition system consists of 3 modules: traffic sign detection, refinement and category/class classification. First, the detection is performed through Mask R-

CNN [16] to detect traffic signs as a unique class with its respective instance masks. Second, the traffic sign refinement is performed extracting Regions of Interest (RoIs) from the masks and filtering out the ones with probability less than 0.8. Finally, we identify up to 9 categories of traffic signs together with their respective specific classes using our proposed CNN. Our system overview is illustrated in Fig. 1.

The contributions of this paper are summarized as follows:

- We trained Mask R-CNN as a base detector on the Mapillary Vistas dataset [17] to identify 19 instance classes as possible objects of interest for autonomous driving systems. Later, we fine-tuned the weights on the GTSDB to focus only on symbol and text traffic signs for German roads.
- In order to identify text-based signs, we proposed an extended version of the GTSDB containing 9 categories following the ETSD [15]: danger, priority, prohibitory, mandatory, special regulation, information, direction, additional panels and, others. Additionally, we selected from the ETSD, the most common classes to be found in urban environments.
- A CNN architecture to distinguish the different categories and classes of traffic signs is proposed in this study together with a simpler CNN architecture to identify only the traffic sign categories.
- Evaluation on the original and extended GTSDB is performed to compare our proposed system with the state of the art. Furthermore, we analyzed the difficulties encountered when other important traffic signs (apart from the symbol based ones) are taken into account.

The rest of the paper is organized as follows: Section II presents related work. Section III describes the method used for traffic sign detection and provides details about the proposed CNNs for classification. In section IV the data labeling procedure is explained for the extended version of the GTSDB [13] and the selection procedure for urban signs from the ETSD is described too. Section V continues with details about the training procedures and the analysis results obtained evaluating German roads with symbol and text based signs. Conclusions and future work are presented in Section VI.

II. Related work

Since traffic sign recognition is important for ADAS systems as well as for autonomous vehicles, an enormous amount of research has been published.

In the following subsections a review of methods regarding detection and classification of traffic signs is provided.

A. Traffic sign detection

Detection consists of localizing spatially in an image the region where the object of interest appears. Normally this region is represented through pixel coordinates called Region of Interest (RoI) and it can be detected through different approaches depending on the final aim.

As traffic signs are grouped into categories with similar shapes and colors [2], a considerable amount of approaches have been proposed taking advantage of these key characteristics. Color-based detection methods [18]–[23] segment the most used traffic sign colors (red, blue and yellow) converting RGB images to other color spaces in order to reduce illumination sensitivity and enhance the target hues to extract RoIs. For example, Maldonado et al. [20], Kuo et al. [19] and De la Escalera et al. [24] use the hue-saturation-intensity (HSI) color space to enhance red, blue and yellow colors using the hue and saturation components. Then color thresholds together with size and aspect ratios are used to segment traffic sign regions for further processing. Also, Yang et al. [22] exploit color for detection, but instead of HSI, they converted RGB values to Ohta space proposing a color probability model to transform color images into probability maps.

On the other hand, as color is a key component but not suitable for outside environments due to illumination changes, shape-based approaches came into play discarding this information and focusing only on detecting certain contours like squares, circles or triangles to identify specific traffic sign categories. Edge detectors, Hough transforms [19], [21], [25], [26], radial symmetry voting [27], [28] as well as template matching [29] are the most common techniques. Edge based methods make use of Canny algorithms and Hough transforms to identify shapes. Radial symmetry voting is a variation of Hough transform that finds points of interest in an image and detect circular signs in an more efficient way. This kind of approach was extended in [30] to detect also triangular, square and octagonal signs. Template matching as its name suggests, utilizes templates to scan the whole image and find similarities through distance transforms.

Some approaches use a combination of both color and shape to detect traffic signs [19], [23], [24]. In this case, color locates the signs roughly and the shape filters out false positives that do not fall into the characteristics of certain categories. Additionally, other methods like support vector machines (SVM) [20], [22], clustering [31] or Artificial Neural Networks (ANNs) [32] are used in combination with color and shape to detect traffic signs. However, all the previously mentioned methods are limited to specific definitions and are sensitive to weather conditions, illumination changes, reflections, occlusions, rotations, etc. reasons that make them very ineffective for real world environments.

Besides shape and color methods, machine learning approaches have been used for traffic sign detection after the successful application in image classification and their adaptation to object detection. These methods treat detection as a classification task and their performance depends on the features selected. Hand-crafted features like saliency [33], local binary patterns (LBP) [34], [35], histogram of gradients (HoG) [22], [33], [36], integral channel features (ICF) [37] or aggregated channel features (ACF) [38] have been applied for this task. Nevertheless, it has been proven that automatically learned features through CNNs perform better than the hand-crafted ones. More comprehensive reviews about detection methods for traffic signs are presented in [11], [39].

Since 2013, CNNs have been used for object detection to first calculate some generic region proposals and then perform classification on the candidates. R-CNN [40] is one example of

the formerly mentioned. It extracts regions through Selective Search [41] and with a CNN, feature maps are extracted for each proposal to be passed to bounding box regressors and SVM classifiers. However, due to the inefficiency of R-CNN, numerous efforts were made to improve this approach. Fast R-CNN [42] jointly optimizes classification and bounding box regression, replacing the SVM classifier with a Softmax layer. Faster R-CNN [43] improves the object proposal step introducing Region Proposal Networks (RPNs) which share full-image convolutional features making the detection suitable for real-time systems. Thanks to the object detection success of these approaches, researches have been inspired and different methods have been proposed for traffic sign detection. Quian et al. [44] focused on detecting traffic signs painted on the road using Maximally Stable Extremal Regions (MSER) [45] detectors and EdgeBoxes [46] (bounding box proposals using edges) to identify region proposals and pass them to Fast R-CNN for further feature extraction, classification and bounding box regression. Huang et al. [47] made use of Generative Adversial Networks (GANs) together with Faster R-CNN to deal with the problem of detecting small objects like traffic signs. Their idea was to map features of the small traffic signs into large object features with the same feature distribution to be able to provide a more accurate prediction. Li and Wang [14] also used Faster R-CNN to detect candidate traffic signs and proposed a refinement of the candidate regions through color segmentation and Hough transforms to identify 4 shapes (circle, rectangle, triangle, octagon).

Until now and despite the good performance of detection approaches based on CNNs like Faster R-CNN, small size objects are not detected accurately and, in consequence, further processing is required. In order to deal with this issue, we use as a detection approach, Mask R-CNN [16]. Mask R-CNN is based on Faster R-CNN predicting the class, bounding box coordinates and a mask for each RoI. In our case, we aim to use the mask branch as a localization refinement for traffic signs. In this way, if the RoI predicted is not accurate enough as Li and Wang [14] encountered, the mask will discard the extra background pixels and a more accurate classification will be performed in further steps.

B. Traffic sign classification

Classification consists in predicting a label for the input based on a series of features learned by the classifier. Regarding traffic sign recognition, classification is performed after the detection is carried out identifying the RoIs and using them as input to infer whether or not a RoI is a traffic sign and its specific class.

According to [48], the first work on traffic sign recognition was performed in 1984 in Japan while trying several computer vision methods to detect objects in outdoor scenes. Since then, several methods for traffic sign classification have been proposed [39], [49] with the aim to 1) include them in driver assistance systems, 2) make inventories to identify the traffic signs and their looking conditions [4], [50] or 3) incorporate them in full control systems for autonomous vehicles.

At the beginning, classification was performed through simple but computationally expensive methods like template matching [51]–[53]. Then, features were used as representative information extracted from the image. Methods using hand-crafted [22], [54], [55] and automatic features were developed, outstanding the ones that learn them automatically. An overview is presented by Saadna and Behloul [39].

CNNs have the capacity to simultaneously learn features and classify data. They were first used in traffic sign classification by Sermanet and LeCun [56] and Ciresan et al. [57] during the German Traffic Sign Recognition Benchmark (GTSRB) competition [8]. Since then, several works have been proposed and used the same benchmark to evaluate their results. Previously, authors were acquiring and labeling their own data [3], [5], [9], [11], [58] to solve this problem for specific signs and countries. For example, Timofte et al. [9] recognized 6 types of Belgium traffic signs based on shapes and colors using AdaBoost classifiers and SVMs. Larson and Felsberg [5] focused on 7 Swedish signs segmenting locally contours through Fourier descriptors and matching them with prototypes by a fast cascaded matching scheme. Grigorescu and Petkov [3] proposed a descriptor called distance set to recognize 3 traffic signs from images collected in Netherlands. Belaroussi et al. [58] compared 3 methods to recognize French signs from their proposed Stereopolis dataset.

Yet with the GTSRB competition and the study made to compare human performance [8], works are able to measure quantitatively their approaches and some, normally based on CNNs, claim to outperform humans. Ciseran et al. work [57] is an example of the formerly mentioned. They won the GTSRB competition obtaining an accuracy of 99.46% on the test images surpassing the average human performance of 98.84%. Their multi-column deep neural network is composed of 25 CNNs ensemble by an averaging approach and each with 3 convolutional layers followed by maxpooling layers and 2 fully connected layers. The original dataset is preprocessed in 4 ways and trained 5 times each for 25 epochs using a scaled hyperbolic tangent activation while translating, scaling and rotating the images.

Sermanet and LeCun [56] obtained the second place on the final stage of the GTSRB competition with 98.31% accuracy. They proposed a CNN similar to ResNet [59] basis where they skip a connection to be able to learn features with different scales and called it Multi-scale CNN. They used a rectified sigmoid activation function, preprocessed the dataset with global and local contrast normalization in the Y channel (RGB converted to YUV space) and applied 5 transformations to each image (random translation, scale and rotation) increasing 5 times the training dataset.

No matter how the previous CNN architectures obtained good accuracy results on the test set, their performances were quite slow due to the high number of arithmetic operations. In order to deal with this issue, Jin et al. [60] proposed to recognize traffic signs using a hinge loss stochastic gradient descent cost function, Rectified Linear Unit (ReLU) activation and a local response normalization layer. Their architecture is based on 3 convolutional layers followed by max-pooling, non-linear activation and normalization layers to finish with two fully connected layers. Data augmentation is applied with 3 transformations on the training set together with 4 image

processing methods to have slightly different input datasets. Their CNN is trained 5 times on each of the 4 sets ending up with an ensemble of 20 CNNs and obtaining 99.65% accuracy on the GTSRB test set.

Yang et al. [22] went beyond all the previously mentioned works, classifying not only the respective traffic sign classes but also their superclass (categories). Their system is based on 4-class SVM classifiers with Radial Basis Function (RBF) kernel using Color Histogram of Oriented Gradients (HOG) features to detect the traffic sign categories. Then, three CNNs are used to perform real time traffic sign recognition. Each CNN contains two convolutional layers followed by sub-sampling layers, plus a fully-connected MultiLayer Perceptron (MLP) in the last two layers. Color information was discarded, Contrast Limited Adaptive Histogram Equalization (CLAHE) was applied to the training set and 3 transformations were applied as data augmentation. Their method was trained and evaluated with the GTSRB [8] resulting on 97.75% accuracy on the test set.

Inspired by Ciresan et al. [57], Aghdam et al. [49] designed a CNN architecture reducing by 65% the number of training parameters of Ciresan CNN replacing the hyperbolic tangent activation by a ReLU activation function. Their proposed CNN adds a transformation layer between the input and the first convolutional layer, divides the middle convolution-pooling layers into separable layers and adds a dropout layer before the final classification layer. Moreover, an ensemble of 2 CNNs was performed following its proposed method and compared it with an ensemble of 10 CNNs formed by averaging method. Accuracies of 99.23% vs 99.1% were obtained with each ensemble respectively. Their proposed ensemble reduces the number of multiplications by 88% compared with Ciresan ensemble. The results are based on the GTSRB test set [8] performing 12 transformations.

Recent work [61] performed RoI refinement and classification with a single CNN. The idea behind this is that usually detection methods are not perfect and return some background information as traffic signs (false positives) for which further processing is required. Luo et al. [61] proposed a Multi-task CNN composed of 3 modules, each of them with a defined number of convolutional and max-pooling layers. The input is a color image passed to module 1 which serves as the base for the localization refinement to classify the image as true or false (traffic sign or not a traffic sign). Module 2 and 3 are used for the classification task. Unfortunately, this CNN cannot be compared with other methods since it is trained with real world and synthetic images captured in China and Street views.

Additionally, Li and Wang [14] managed traffic sign detection and classification. Their classification task is performed with a proposed CNN that uses different asymmetric kernels in order to reduce the number of convolutional operations. At the same time, an inception module concatenates 2 blocks of asymmetric convolutional layers to fuse different spatial information. Their CNN model trained with the GTSRB achieved 99.66% accuracy using ReLU activations and applying data augmentation with random rotation, scaling, translation and shear transformations.

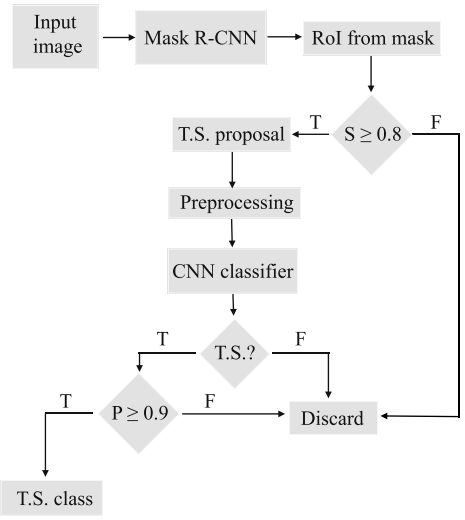


Fig. 2. Pipeline of our proposed traffic sign recognition system. T.S. refers to Traffic Sign. The Mask R-CNN block represents the detection module while the refinement is composed of the RoI extraction and class score (S) filtering. The CNN classifier block makes reference to the classification module and will output a traffic sign class only if the predicted probability (P) is at least 0.9.

Arcos-García et al. [62] defined a CNN for classifying traffic signs using Spatial Transformer Networks (STNs). The STNs are introduced to substitute data augmentation process as they perform geometric transformation on the input map with the aim to make it spatially invariant to the input data. Global normalization and local contrast normalization are applied to the images to enhance the recognition. Their CNN architecture is composed of three STNs, three convolutional blocks and two fully connected layers. Each convolutional block consists of a convolutional layer followed by ReLU activation, max-pooling and local contrast normalization layers. Their final architecture obtained 99.71% on the GTSRB.

In our study, we propose a CNN trained on the GTSRB and compare its performance with the state of the art methods.

III. Traffic sign recognition

Our traffic sign recognition system is illustrated in Fig. 2. The detection is capable of localizing the RoI where the sign is present. In our case this step is carried out by Mask R-CNN [16]. The advantage of using this approach resides in the localization refinement of the traffic sign performed at the same time that the mask is obtained. The classification, on the other hand, is accomplished with the proposed CNN to categorize the traffic signs and provide the specific classes for each category. This step takes as input the filtered output of the detection approach, pre-processes the image and labels each object with a corresponding class.

In the following subsections, we describe in details the detection, refinement and classification modules.

A. Traffic sign detection

Traffic signs in real scenes have large differences in color, shape and size, reasons that make them hard to detect. Fortunately, deep learning approaches came into play with CNNs

extracting automatically complex features without the need to design or chose manually the ones that will perform better to detect traffic signs.

We chose Mask R-CNN [16] as a general detector to identify static and moving objects seen in outside environments, but in this work, we only focus on traffic signs. This approach is based on Faster R-CNN [43] which detects objects in two stages: 1) propose candidate regions (RoI) through a Region Proposal Network (RPN) and 2) extract features for each RoI using the RoIPool layer to perform classification and bounding box regression. On the other hand, Mask R-CNN introduces a new branch to segment a binary mask for each RoI and replaces the RoIPool layer with a RoIAlign layer. The whole pipeline is illustrated in Fig. 3.

The input image is passed to a series of convolutional layers (based Conv layers) that output a feature map. In our case, ResNet-101 architecture was used together with Feature Pyramid Network (FPN) as feature extractor. Then region proposals are obtained with the RPN. This network uses a $n \times n$ spatial window to slide it through the whole feature map. Each window is mapped to a lower dimensional feature which serves as input to two fully connected layers: a box-regression layer and a box-classification layer. k number of region proposals, called anchors, are predicted at each sliding window location where the box-regression layer outputs $4k$ values (box coordinates) and the class-regression layer $2k$ scores (foreground or background). Normally the anchors predicted are centered to the window and set with different sizes and aspect ratios (normally 3 each). In order to reduce redundancy of the overlapping region proposals, non-maximum suppression (NMS) is used based on the RoI scores. Only the regions with at least 0.7 IoU are then forwarded to the detection and mask branches.

An intermediate layer to extract the features for each RoI from the image feature map is carried out by RoIAlign. This layer replaced RoIPool to overcome the feature misalignment introduced by quantization while mapping the extracted feature map into fixed size. Instead of quantizing the smaller feature map using max pool into a certain number of bins (e.g. 7×7), RoiAlign samples each bin into 4 locations and computes their value by bi-linear interpolation from the nearby grid points. In this way, each bin aggregates the result of the 4 points using maximum or average operation to extract the exact values of the input features. This substitution lead to a more precise per-pixel segmentation and box regression [16].

Following the data flow in Fig. 3, each RoI feature vector is fed to the detection and mask branches to obtain its corresponding bounding box coordinates (x,y,w,h), its score class and its respective binary mask, where (x,y) indicate the top-left corner and (w,h) the width and hight of the box.

The decision of choosing this detector resides on the following reasons:

- It is based on a fast and accurate detector: Faster R-CNN [43], which has been used in several works [14], [44], [47], [63], [64] to identify traffic signs due to its capacity to detect small objects. In [65], authors compared several architectures to detect traffic signs, specifically German signs from the GTSDB [13], and proved that Faster R-CNN outperforms the others no matter the CNN used as feature extractor.
- Detection evaluation in [16], showed that Mask R-CNN results surpassed the ones of Faster R-CNN thanks to the RoIAlign layer.
- No further processing of the RoIs is required to detect certain shapes [14] since the mask branch serves as localization refinement.
- The detector is not constrained to certain shapes like all the previous works mentioned in Section II-A. Any traffic sign shape (circle, rectangle, triangle, square, octagon, etc.) will be detected and further classified.
- Its implementation is publicly available for different deep learning frameworks making it easy to use, adaptable to specific tasks and reproducible.

In this work we used the implementation made for Tensorflow [66] and trained the detector on Mapillary dataset [17]. Details about the training and evaluations are presented in Section V.

B. Traffic sign refinement and classification

Once the outputs of Mask R-CNN are obtained, the refinement module extracts the exact RoIs using mask outputs and only keeps the regions for which class scores of being a traffic sign are bigger or equal than 0.8. This filtering is done to reduce the number of mistaken objects as traffic signs and to pass to the classifier a cleaner set. It is worth mentioning that the class scores do not influenced the detection results but they are used for filtering because it provides a good hint to discriminate objects which do not belong to the traffic sign class.

The goal of the classifier is to identify properly the traffic signs classes and discard the false positives (background RoIs) for the final recognition system. Traditionally, this filtering is done in the detection part, but as detectors are never perfect, the classifier needs to account for this as well. Fortunately, with the capability of CNNs, this task can also be done adding to the classifier an extra class for learning background objects. In this section we will present two CNNs, one capable of learning traffic sign categories (Cat_CNN) and a deeper model able to identify the specific traffic sign classes (Class_CNN). The performance of the deeper CNNs is competitive with the state of the art CNNs for traffic sign classification. Any of the CNNs is referenced as the classification module (CNN classifier) in Fig. 2.

After referring to several architectures [14], [22], [49], [56], [57], [60], [61], we define a CNN based on 5 blocks to identify the specific class and a CNN with 3 blocks to identify the traffic sign category. The category classifier (Cat_CNN) shares the same definition of blocks 1 and 2 from the class classifier (Class_CNN) changing the third block with a convolutional layer of size 80. Fig. 4 illustrates their definitions and Table I and Table II show specific details.

Our proposed architectures are inspired by VGG-16 network [67] setting up blocks of convolutional layers with kernel sizes of 3×3. In an attempt to reduce the number of parameters and convolutional operations, we replace the $n \times n$ kernel by a

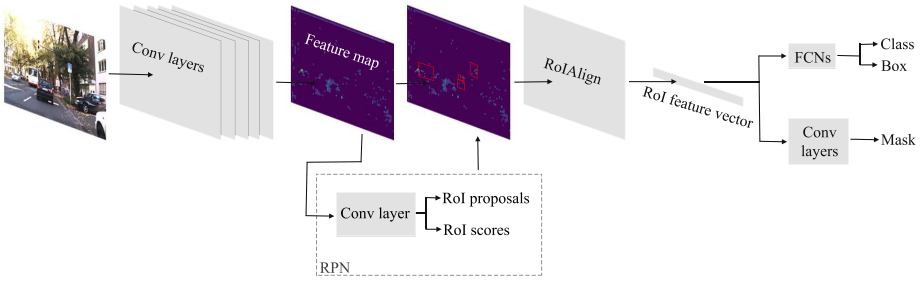


Fig. 3. Mask R-CNN data flow.

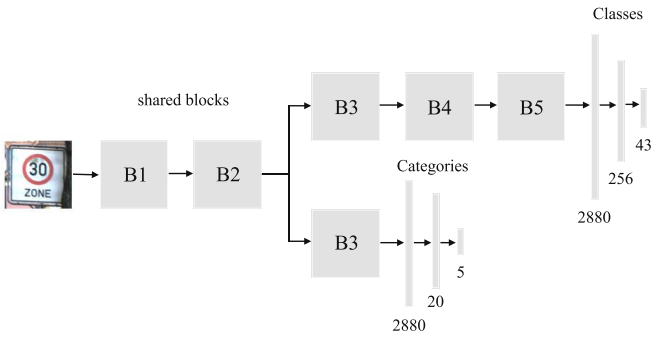


Fig. 4. CNN structure represented by blocks for the category and class classifiers. Block 1 and 2 (B1 & B2) share the same definition.

TABLE I
ARCHITECTURE OF OUR CNN FOR TRAFFIC SIGN CATEGORIES IDENTIFICATION (CAT_CNN). 'CHAN' MAKES REFERENCE TO THE NUMBER OF CHANNELS, 'KS' TO KERNEL SIZE, 'STD' TO STRIDE AND 'KPPROB' TO THE PROBABILITY OF KEEPING THE ORIGINAL FEATURE.

	type	outMap	chan	kS	std	keep prob
I	input image	48×48	3	-	-	-
B1	convolution	48×48	32	3×3	1	-
	convolution	48×48	32	3×3	1	-
	maxpooling	24×24	32	2×2	2	-
	dropblock	24×24	32	3×3	-	0.8
B2	convolution	24×24	64	3×3	1	-
	convolution	24×24	64	3×3	1	-
	maxpooling	12×12	64	2×2	2	-
	dropblock	12×12	64	3×3	-	0.8
B3	convolution	12×12	80	3×3	1	-
	maxpooling	6×6	80	2×2	2	-
	dropblock	6×6	80	3×3		0.75
C	fully connected	1×1	20	-	-	-
	dropout	1×1	20	-	-	0.5
	fully connected	1×1	5	-	-	-

$n \times 1$ and $1 \times n$ kernels [68]. Using asymmetric convolutions, the computational cost decreases by certain percent calculated with $(n \times n - 1 \times n - n \times 1)/(n \times n)$. For example, replacing a 5×5 kernel with asymmetric ones will save $(5 \times 5 - 1 \times 5 - 5 \times 1)/(5 \times 5) \approx 60\%$ of computational cost, while for a 3×3 will decrease by 33%. This strategy was also used by Li et al. [14] in the second and third layers of their CNN model including the inception module. Authors in [68] claimed that this replacement doesn't work good in early layers but gives good results when applied to feature maps of sizes between 12 and 20. We applied asymmetric convolutions in blocks 2 and 3 (B2 and B3) but this only made worse the results. In our case, applying them in blocks 4 and 5 (B4 and B5) gave us the best outcome.

TABLE II
ARCHITECTURE OF OUR CNN FOR TRAFFIC SIGN CLASS IDENTIFICATION (CLASS_CNN). 'CHAN' MAKES REFERENCE TO THE NUMBER OF CHANNELS, 'KS' TO KERNEL SIZE, 'STD' TO STRIDE AND 'KPPROB' TO THE PROBABILITY OF KEEPING THE ORIGINAL FEATURE.

	type	outMap	chan	kS	std	kpProb
I	input image	48×48	3	-	-	-
B1	convolution	48×48	32	3×3	1	-
	convolution	48×48	32	3×3	1	-
	maxpooling	24×24	32	2×2	2	-
	dropblock	24×24	32	3×3	-	0.8
B2	convolution	24×24	64	3×3	1	-
	convolution	24×24	64	3×3	1	-
	maxpooling	12×12	64	2×2	2	-
	dropblock	12×12	64	3×3	-	0.8
B3	convolution	12×12	128	3×3	1	-
	convolution	12×12	128	3×3	1	-
	maxpooling	6×6	128	2×2	2	-
	dropblock	6×6	128	3×3	-	0.75
B4	convolution	6×6	256	5×1	1	-
	convolution	6×6	256	1×5	1	-
	dropblock	6×6	256	3×3	-	0.75
B5	convolution	6×6	64	3×1	1	-
	convolution	6×6	64	1×3	1	-
	maxpooling	3×3	64	2×2	2	-
	dropblock	3×3	-	3×3	-	0.75
C	fully connected	1×1	256	-	-	-
	dropout	1×1	-	-	-	0.5
	fully connected	1×1	43	-	-	-

Differently from VGG-16, we applied 2 types of regularization techniques: dropout [69] and dropblock [70]. Dropout excludes randomly activations in the feature maps, keeping only certain percentage. It helps the model learn more robustly the parameters preventing overfitting. Dropblock, on the other hand, discards information of a continuous region (defined by a block of size $n \times n$) of the map. In this way, only the most representative feature activations of each region are kept, and in consequence, the model is more stable during learning. As dropblock applies two dimensional kernels, it can only be used for convolutional operations whose feature maps have at least the same size as the region block. We performed experiments using both regularization techniques (see Section V) and found that dropblock improves our model performance.

Additionally, Batch Normalization [71] (BN) is applied after

each convolution to help the model learn more easily its hyperparameters. ReLU [72] activations are used after the BN layer for each convolutional layer and fully connected layer, except for the last one, which uses a Softmax activation to output 43 probabilities for each input. The output number is set to 43 in Table II due to the number of classes that GTSRB [8] has. This output changes depending on the desired number of classes. In Table I, the output is set to 5 in order to recognize the 4 categories defined in the GTSDB [13] (Danger, Prohibitory, Mandatory and Others) plus a background class. The category classifier has fewer layers compared to the class classifier, because recognizing the category is simpler than the class, as mostly shapes and colors are necessary. It is well known that the first few layers of a CNN detect lines and corners, while deeper layers learn more complex features. For this reason, only three modules are used for the category classifier whose details are shown in Table I.

For comparison purposes and in order to evaluate our proposed CNN, we trained the class classifier (Table II) with 43 output classes according to the GTSRB. Furthermore, for the final traffic sign recognition system, we trained 6 more CNNs with the same architecture: one with GTSRB 43 classes plus the background class, four more, one for each category of the GTSDB and another one with the chosen urban/rural classes of the European Traffic Sign Dataset [15] (ETSD). In the same way, we trained one CNN with the category classifier architecture (Table I) and 5 outputs to recognize the GTSDB categories plus background. The experimental results are provided in Section V.

IV. Datasets

In order to evaluate the proposed traffic sign recognition system, we use the GTSRB [8] and the GTSDB [13] to compare our method with other approaches. Nevertheless, these benchmarks are not representative enough for real world applications since 1) they include only pictographic/symbol based traffic signs with regular shapes and colors, 2) important classes are discarded like temporal signs, text-based signs, supplemental panels, etc. 3) symbol variations are not contemplated either inside the same country or for multi-country purposes (intra-class variability), and 4) images used for detection present considerable big traffic signs.

We proposed a dataset for detection approaches that handles the shortcomings one, two, and four mentioned before. At the same time, we introduced a subset of traffic signs used in urban environments capable of handling intra-class variability (shortcoming three), symbol and text based signs, and multi-shape and multi-color signs.

A. Detection dataset - GTSDB extended

The GTSDB dataset [13] introduced in 2013 is composed of 900 images of resolution 1360×1024 pixels. The annotated traffic signs correspond to 43 classes of the GTSRB [8] grouped into 3 main categories: prohibitive, mandatory and danger signs. The annotations are provided in a txt file with the RoI for each sign together with its corresponding label class and the relations for each category. The dataset also includes labels of irrelevant classes considered in the category named other. This last category was not included in their competition, but later works [14], [22] considered it for evaluation as well.

The training set consists of 600 images and 300 images correspond to the evaluation set. The traffic sign sizes vary between 16 and 128 pixels. Even though sizes seem relatively small, there are cases where smaller signs are visible and recognizable in the picture, but they were not labeled. Furthermore, there are many more important signs in the same scenario that were not considered. For this reason, we introduced an extended version of the GTSDB which comprises signs belonging to the 9 sub-categories of the ETSD [15]. The ETSD comprises a broad selection of traffic signs including text-based signs as well as signs with very different aspect ratios. A more detailed description about this dataset is provided in the next subsection IV-B.

The extended version of the GTSDB allows quantitative evaluation for recognition, detection and segmentation approaches. In other words, it does not only contain the RoI for each sign, as regular detection approaches need, but also a corresponding mask useful for segmentation and instance segmentation approaches. Meletis and Dubbelman [73] labeled the masks for the 43 classes originally considered in the GTSDB and provided them to us. We labeled the GTSDB dataset with all the signs not included in the original dataset but considered in the ETSD. The labels are provided in a pixel-precision manner to allow accurate learning of appearances and shapes. In other words, we labeled all the pixels in the image that correspond to a traffic sign, and the rest of pixels are labeled as background (2 classes - traffic sign and background). We used LIBLABEL [74], a tool written in MATLAB, to annotate object classes through polygons. Each polygon drawn is saved as an instance of the semantic class selected. In this way, it is possible to have an ID of the semantic class and an ID of the instance to simultaneously detect objects and segment them (instance segmentation).

We inspected each traffic sing labeled and discarded the signs which are not recognizable by human eye (normally very small and far signs). The annotation procedure took approximately 75 hours of work for 900 images.

After the labeling procedure, the masks of the original signs [73] were combined with the masks of the missing signs. This merging procedure was performed carefully to preserve unique instance IDs for each sign. The Mapillary [17] format was used as the standard definition for the image segmentation with 2 labels: traffic sign front and unlabeled classes. Fig. 5 shows an example of a labeled image of the GTSDB database together with its corresponding semantic and instance outputs.

Additionally, each labeled traffic sign is saved with its corresponding class based on the 164 classes of the ETSD. The total number of traffic signs is expanded from 1213 to 2655 for the full dataset (900 images). Table III shows a description of the traffic signs by category of the original GTSDB database and its proposed extended version.

In the same way as in the GTSDB [13], a txt file containing the traffic sign RoI and class information is provided for detection approaches. In case that semantic segmentation approaches require the specific traffic sign label as part of

a) Original image b) Semantic labels c) Instance labels

Fig. 5. Example of a GTSDB image and its corresponding semantic and instance labels. The semantic color class correspond to the Mapillary [17] definition. Instance labels are numbered from 1 to n, but for visualization purposes they are presented with random colors.

TABLE III
NUMBER OF SIGNS BY CATEGORY FOR THE TRAINING AND TESTING SETS OF EACH DATASET. THE OTHERS CATEGORY IN THE ORIGINAL GTSDB AND ITS EXTENDED VERSION DO NOT REPRESENT THE SAME CLASSES (REFER TO [13] FOR THE CLASSES IN THE ORIGINAL GTSDB AND SEE FIG. 7 FOR THE EXTENDED GTSDB) BUT WE PUT IT LIKE THAT FOR THE NAME CONVENTION. THE UNKNOWN CATEGORY CONTAINS TRAFFIC SIGNS LABELED WHICH CLASS DOES NOT CORRESPOND TO ANY OF THE CONSIDERED ONES.

Category	Original GTSDB		Extended GTSDB	
	Train	Test	Train	Test
Danger	156	63	156	66
Priority	-	-	128	74
Prohibitory	396	161	478	193
Mandatory	114	49	122	56
Special regulation	-	-	115	73
Information	-	-	0	1
Direction	-	-	355	171
Additional panels	-	-	172	109
Others	186	88	284	92
Unknown	-	-	4	6
Total	852	361	1814	841

the semantic classes, a multiplication of the RoI with its corresponding mask will allow to exchange the labels. The proposed extended GTSDB database can be found in https://github.com/citlag/Traffic-Sign-Recognition.

In this work, we used a general class for traffic signs including symbol and text signs to perform the detection with Mask R-CNN. The text file with the specific classes for each traffic sign is used for the classification step to compare the predicted label with its ground truth.

B. Classification dataset

As mentioned earlier, the GTSRB [8] has several drawbacks that need to be taken into account for a robust traffic sign recognition system. Particularly, it needs to consider: intraclass variability, missing signs (traffic signs not included in the GTSRB dataset) as well as text-based signs.

The GTSRB dataset is composed of 43 classes that belong to 4 categories: danger, prohibitory, mandatory and others. These categories have regular shapes and colors and include only symbol-based signs. In real environments, there are many more classes that need to be recognized with either symbol or text, squared or rectangular shapes with different colors and small variabilities inside the same class. In order to deal with all the previous issues, the ETSD [15] came along providing a more complete definition of traffic sign classes. The ETSD comprises 164 classes grouped into 9 sub-categories: danger, priority, prohibitory, mandatory, special regulation, information, direction, additional panels and others.

Differently from the GTSRB, the ETSD includes categories with significant classes to recognize while driving. For example, additional panels provide supplementary information to interpret appropriately the sign information, while temporary signs, like construction works, are placed together with deviation signs, barriers or obstacles to avoid driving into dangerous surfaces and provide an alternative and safe route. Fig. 6 shows some examples of relevant traffic signs not included in the GTSRB.

Even though, the ETSD comprises a broad selection of traffic signs, we carefully selected a group of 132 classes representative of urban/rural environments. The selection was made after a rigorous examination of frequent signs appearing in the GTSDB dataset. Fig. 7 shows the classes contemplated in this study which are analyzed further in the results section.

Text based signs (direction and additional panels categories) are more challenging since they have very different appearances even within the same class. At the same time, signs with very different aspect rations also encounter more difficulties for the classifier to make correct predictions due to the resizing procedure performed to the input images. Squared sizes are normally chosen for CNNs leading to information loss. We will see how the performance drops when the detection and classification on the GTSDB dataset includes other categories than symbol-based signs. We will compare the results obtained for the detection and classification on the GTSDB dataset and its extended version.

V. RESULTS

In this section we will evaluate the proposed traffic sign recognition system on the GTSDB and the GTSRB for comparison purposes with other works. Additionally, experimental results on the extended GTSDB and the urban subset of the ETSD will show the challenges encountered when including other than symbol-based signs. First we provide details about the training procedure for the detection and classification modules to follow up with comparisons between our results and the ones reported by the competitors.

A. Training

All models are trained in GPU mode using a NVIDIA GeForce GTX1080Ti with 11GB of memory, an IntelCore

a) Directional signs b) Additional panels c) Missing signs

Fig. 6. Examples of traffic signs not included in the GTSRB. (a) shows four directional signs. (b) shows two examples containing additional panels. In the first example, the truck below the 30 speed limit sign, indicates that the restriction only applies for that type of vehicles; while in the second example of (b), the first panel announces that the speed limit is only applicable for 600 m and the second panel indicates the timing applicable. The missing signs (c) make reference to curves, obstacles, barriers and limited access on the side.

i7K-8700K (6 cores 12 threads, 12 Mb cache memory) processor and RAM of 32GB.

1) Detection module.: We used the implementation of Mask R-CNN in Keras and Tensorflow[1]. This implementation uses ResNet101 with Feature Pyramid Networks (FPN) as backbone for feature extraction. The ResNet architecture divides its convolutional layers into 5 stages which are used later by the FPN to upsample them by a factor of 2 and fixed number of filters of 256 with kernel size 3×3. The feature map is composed of all ResNet stages upsampled. Once the feature map is extracted, the Region Proposal Network (RPN) takes it as input and feeds it into a based convolutional layer of 512 filters with kernel size of 3 to generate 9 region proposals (using 3 scales and 3 aspect ratios [16]) for every window size location in the feature map. As the feature map is composed of 5 stages (ResNet and FPN), every stage is passed to the RPN network to extract RoIs in different scales together with their probabilities (RoI scores) of being foreground or background. Considering that the positive RoIs may be overlapping, non-maximum suppression is used to obtain only one.

We set to 255 the number of maximum detected RoIs per image, where a positive output is selected if there is at least 70% overlap with the ground true and discarded it if the overlap is less than 30%. Subsequently, the target RoIs are passed to the RoIAling 3 layer to extract fixed size features per proposal and feed them into the class and mask branches. For the classification branch, the RoI feature is passed to 2 convolutional layers with 1024 filters of size 3×3 and 1×1 followed by BN [71] and ReLU activations [72]. This convolutional layers are shared for the class and box heads. The class head uses Softmax activation to output the probability of the RoI to belong to a certain class (traffic signs), while the bounding box head utilizes linear activation to obtain the spatial locations of the RoI. The mask branch, on the other hand, takes the RoI feature and performs 4 convolutional operations with 256 filters using kernel size 3×3. In the same way as in the classification branch, each convolutional operation is followed by BN and ReLU activations. The mask for every RoI is obtained in the last layer with a per-pixel Sigmoid activation in order to have a binary mask.

Taking into account that the GTSDB contains only 600 images for training, we first trained Mask R-CNN with the Mapillary Vistas dataset [17] to obtain some initial weights and fine-tune them later. This dataset contains 25,000 high resolution images representing street-level scenarios. It is labeled in a per-pixel manner with 64 semantic classes and 37 instance ones. We chose this dataset for training as 1) it contains only driving scenarios, 2) traffic signs class contains symbol and text-based signs, 3) a diversity of scenarios is provided (urban, rural, highways), 4) the high image resolution allows to choose the proper size according to the needs of the task and 5) the number of images is enough to train a robust detector/classifier.

The Mapillary dataset is divided into 18,000 / 2,000 images for training and validation respectively. For our approach, we set the training size to 1280×960 discarding all images in the dataset that do not fit our requirement. 17,979 images were used for training Mask R-CNN, while 1995 were used for validation. We chose carefully 19 instance classes comprising static and moving objects of interest to detect objects like cars, pedestrians, traffic lights, pedestrian crossing, traffic signs among others.

The training procedure was set to 32 epochs using Stochastic Gradient Descent (SGD) as the optimizer with learning rate equals to 0.01, momemtum of 0.9 and weight decay of 0.0001. Due to the image size and capabilities of our GPU, only 1 image per batch was used. The ImageNet weights, trained in ResNet50, were used as initialization for fine-tuning all layers of Mask R-CNN with the Mapillary dataset. This procedure took 18 days, 23 hours, 29 minutes and 49 seconds.

Once the Mapillary weights were obtained, we continue to the second training stage to fine-tune the previously obtained weights with the original GTSDB [13] and the extended GTSDB. The training parameters were almost the same changing the number of epochs to 50 and the training and validation steps to 600 and 300 respectively (number of images in the train an validation sets). These training procedures took approximately 7 hrs each. The minimum validation loss for each dataset was obtained at epoch 46 for the original GTSDB and at epoch 45 for its extended proposed version.

The learning time was quite long due to the number of parameters that Mask R-CNN has (63,830,282), the image size (1280×960) and the batch size (1) used for training.

2) Classification module.: We trained 7 classifiers with the 2 CNN architectures described in section III-B using Keras and Tensorflow back-end. The input images are preprocessed, converting them to HSV color space, equalizing the V channel, transforming the HSV image back to RGB and resizing them to 48×48 pixels.

[1]https://github.com/matterport/Mask_RCNN

(a) Danger warning signs

(b) Regulatory - Priority

(c) Regulatory - Prohibitory

(d) Regulatory - Mandatory

(e) Regularory - Special regulation

(f) Informative - Direction

(g) Informative - Additional panels

(h) Others

Fig. 7. Urban/rural subset selection of the ETSD. (f) and (g) correspond to text-based signs while the rest are symbol-based signs.

Towards a fair comparison with the state of the art, we trained 5 times the CNN for traffic sign class identification (deeper architecture described in Table II) with the GTSRB. The train set was divided into 90%-10% for training and validation. The model was initially trained for 40 epochs using Adam optimizer with a learning rate of 1e-3 and a weight decay of 1e-6. Additionally, we monitored the validation loss with a patience of 20 and factor of 0.2 to reduce the learning rate when the validation loss stops decreasing; in this way, only the best weights are saved preventing over-fitting.

Regularly, the training capability can be increased if a larger dataset, containing a wider variability, is used. Data augmentation is a technique used to enlarge the dataset without the need of acquiring more data. It is common and generally applied when training a CNN [49], [57], [60], [61] to have better generalization. For our CNN architectures, each traffic sign in the training set undergo to the following 4 transformations: each signs is 1) randomly translated 10% of the image size in the vertical and horizontal axes, 2) randomly rotated within a range of [-10,+10] degrees, 3) randomly scaled with a factor from 0.8-1.2 and 4) randomly transformed with a shear angle of 0.1 degrees.

In Keras, data augmentation can be applied in real-time using the ImageDataGenerator class. We used this technique to train further the CNN architecture on the GTSRB but this time for 100 epochs. The weights obtained previously during the training without data augmentation are used as initialization. Thus, the architecture does not have to learn everything from scratch. The rest of the parameters are left unchanged as described previously. The results obtained are presented in the subsection V-B.

In the same manner, we trained the urban/rural subset subtracted from the ETSD [15] and used it to evaluate the detection and classification performances. As the detector is not perfect, some objects that are not traffic signs are passed to the classifier. For this reason, all classifiers used for the traffic sign recognition system include a background class. This class was generated with Mask R-CNN saving the negative RoIs obtained during the detection on the GTSDB.

In the literature, there are approaches which used only 1 classifier to perform the recognition, while others used several, one for each category to identify their classes [22]. Hence, we trained 5 classifiers to compare the performance of the recognition system. One classifier is trained using the Cat_CNN architecture described in Table I to identify the category of the detected sign, and four more classifiers using the Class_CNN architecture (Table II) to recognize the classes for each category. In such way, the output of the first category classifier (4 categories + background) filters out false positives and defines the input for the corresponding class classifier. The four category classifiers: Danger - 15 classes, Prohibitory - 12 classes, Mandatory - 8 classes and Others - 8 classes, (all of them adding the background class +1) are responsible for identifying the specific class in the respective category. The training procedure follows the same standard as described for the GTSRB (train for 40 epochs without data augmentation and use the weights as initialization to train it again for 100 epochs using data augmentation).

Because the detector module (Mask R-CNN) identifies a unique class for traffic signs, the category identification is performed further with the classification module. Hereupon, we will discuss first the classification results in order to

continue with the final detection performance.

B. Classification performance

In order to demonstrate the performance of the proposed classifier defined in Table II, we perform a self comparison of it with the following modifications:

1) A classifier that changes the asymmetric kernels for symmetric ones of size 3×3.
2) A classifier that uses only dropout.
3) A classifier which adds 2 residual blocks, one between B3 and B4 (see Table II) and one between B4 and B5. The first residual block uses a convolutional layer with 128 filters and kernel size 1×1 and the second one a convolutional layer with 64 filters and size 3×3. Both convolutions in the residual blocks are followed by BN and concatenated according to the ResNet [59] definition.

Table IV shows the results obtained for training the classifiers with the GTSRB [8] together with the parameters of each classifier. The accuracy result is obtained evaluating them on the test set.

From the Table IV we can see that, all classifiers perform better when Dropblock [70] is used instead of Dropout [69]. If we compared the accuracies obtained only using dropout as regularization, the asymmetric kernels not only reduce the number of parameters but also increased the performance by 22% compared to the symmetric ones, and 33% against the CNN with residual blocks (CNN RB). On the other hand, comparing the accuracies obtained using dropblock, the CNN with symmetric and asymmetric kernels is almost the same. The possible reason behind this behavior could be due to the way dropblock discards the activations. Dropout discards randomly the percentage defined, while dropblock, turns off the activations with continuous similar characteristics, adding more stabilization to the learning process. In both accuracy results, the CNN with residual blocks performs the worse, eliminating the possibility to use this strategy for not deep networks. The CNN with asymmetric kernels proposed in this study, provides the best trade off between accuracy and complexity.

Furthermore the proposed CNN with asymmetric kernels was trained 5 times with data augmentation to account for the classifier's stability. We perform the same procedure using dropout and dropblock to compare how the two regularizations affect the model. Table V shows the results obtained for each 100 epochs called trial.

Analyzing the results obtained in Table V, we confirm that the regularization has an important role in the model stability. The learning variability in our proposed CNN architecture decreased from 0.21 to 0.05 using Dropblock, reason why it was chosen. The best test accuracy obtained was 99.61% while the average was 99.53%. A comparison of our proposed model with the state of the art is shown in Table VI. There, we can see that, the models with accuracy above the one of our proposed CNN (top acc 99.61%), have high number of parameters compared to ours, which makes it an appropriate choice for real-time applications. It is worth mentioning that most of the models with high accuracy presented in Table VI, are composed of an ensemble of models of at least 5 CNNs. The only two models with accuracy higher than ours that used a single CNN, were obtained by USTC [14] with 99.66% and by Arcos-Garcia et al. [62] with 99.71% as reported in their studies. Nevertheless, the CNN architecture in [62] has seven times more number of parameters compared to our architecture; which makes it inefficient for recognition systems. Contrarily, the CNN in [14] has only 860,000 more parameters than ours, but once trained by ourselves with the same training procedure indicated in their study (code available in[2]), we obtained 99.59% accuracy (0.07% less than their reported results). However, authors in [14] do not mention if their results are based on the top accuracy obtained after several trials and how many times they performed the training. The difference between our obtained results and theirs could be due to the classifier stability during training, as the authors used dropout as regularization technique. Hence, we can conclude that our proposed CNN provides good stability and trade off between model complexity (parameters) and recognition accuracy compared to the state of the art.

Moreover, we inspected the running time and confidence of our proposed CNN. Our model is capable of classifying in our PC, in GPU mode, a traffic sign in 0.16 milliseconds which makes it optimal for the whole pipeline of traffic sign recognition. Evaluating its performance, our CNN is capable of classifying correctly 99.38% traffic signs with a confidence probability bigger or equal to 0.9. For this reason, this threshold was chosen for our pipeline (see Fig. 2) to predict or discard a detected traffic sign. In this way, we make sure the output of the proposed recognition system is accurate and reliable for driving vehicles.

Adding the background class, we trained boths CNNs described in Table I (Cat_CNN) and Table II (Class_CNN) with the same procedure explained in subsection V-A2. All classifiers are trained 5 times and only the top results are presented in Table VII together with their characteristics. The traffic sign recognition system is evaluated with the detection module (Mask R-CNN) and all the classifiers from Table VII.

C. Detection performance

Here, we evaluate the detection performance with Mask R-CNN and several classifiers. The first evaluation is made for symbol-based signs on the GTSDB to detect 4 Categories (43 classes) with 2 classifiers: one which is only capable of detecting the categories (Categories CNN from Table VII) and another one capable of performing the category and class identification (All CNNs from Table VII). The second evaluation is performed for symbol and text-based signs on the proposed extended version of the GTSDB to detect 8 categories (132 classes) of traffic signs (Urban classifier from Table VII): danger, priority, prohibitory, mandatory, special regulation, direction, additional panels, and others.

In both cases, a correct detected traffic sign is considered if its Jaccard similarity coefficient (Intersection over Union - IoU) is at least 0.5. The IoU is computed following (1).

[2]https://github.com/USTClj/Traffic_Sign_Recognition_Efficient_CNNs

TABLE IV
ACCURACY PERCENTAGE RESULTS OBTAINED ON THE GTSRB TEST SET. THE INPUT SIZE REFERS TO IMAGE "WIDTH×HEIGHT×CHANNELS", WHILE THE NUMBER OF PARAMETERS IS PRESENTED IN MILLIONS (M) AND TIME IN MILLISECONDS (MS).

CNNs	Input size	Data augmentation	Parameters	Accuracy (dropout)	Accuracy (dropblock)	Time
CNN Symmetric	48x48x3	Yes	3.59 M	99.19%	99.54%	0.2 ms
CNN Asymmetric	48x48x3	Yes	2.09 M	99.41%	99.51%	0.16 ms
CNN RB	48x48x3	Yes	2.40 M	99.08%	99.14%	0.17 ms

TABLE V
ACCURACY PERCENTAGE RESULTS OBTAINED ON THE GTSRB TEST SET.

Trials	Accuracy (dropout)	Accuracy (dropblock)
1	99.41%	99.51%
2	99.33%	99.48%
3	99.37%	99.61%
4	99.30%	99.54%
5	98.83%	99.50%
Average	99.25% ±0.21	99.53% ± 0.05

TABLE VI
RECOGNITION ACCURACY OF VARIOUS METHODS EVALUATED ON THE GTSRB TEST SET. THE NUMBER BETWEEN PARENTHESIS IN "ENSEMBLE" COLUMN REFERS TO THE NUMBER OF CNNS USED TO CONSTRUCT THE ENSEMBLE MODEL. INPUT SIZE REFERS TO THE $width \times height \times \#channels$. THE NUMBER OF PARAMETERS ARE PRESENTED IN MILLIONS (M).

Method	Data Augmentation	Ensemble	Input size	Parameters	Accuracy
CNN with 3 STNs [62]	No	No	$48 \times 48 \times 3$	14.63 M	99.71%
USTC [14]	Yes	No	$48 \times 48 \times 3$	2.95 M	99.66%
Hinge Loss CNN [60]	Yes	Yes (20)	$47 \times 47 \times 3$	23.2 M	99.65%
URV model [49]	Yes	Yes (5)	$48 \times 48 \times 3$	5.62 M	99.61%
IDSIA [57]	Yes	Yes (25)	$48 \times 48 \times 3$	38.59 M	99.46%
Improved CNN-8 model [15]	Yes	No	$48 \times 48 \times 3$	1.48 M	99.37%
Multi-scale CNN [56]	Yes	No	$32 \times 32 \times 1$	1.44 M	99.17%
Human performance [8]	-	-	-	-	98.84%
Ours	Yes	No	$48 \times 48 \times 3$	2.09 M	99.61%

TABLE VII
CLASSIFIERS' ACCURACY RESULTS AND CHARACTERISTICS TRAINED ON THE PROPOSED CNNS DESCRIBED IN SECTION III-B.

Model	Classifier name	No. Classes	Data augmentation	Parameters	Accuracy
Cat_CNN	Categories	5	Yes	170,621	99.92%
Class_CNN	Danger	16	Yes	2,081,712	99.25%
Class_CNN	Prohibitory	13	Yes	2,080,941	99.76%
Class_CNN	Mandatory	9	Yes	2,079,913	99.89%
Class_CNN	Unique	9	Yes	2,079,913	99.64%
Class_CNN	All	44	Yes	2,088,908	99.55%
Class_CNN	Urban	133	Yes	2,111,781	99.07%

$$IoU(A,B) = \frac{|\ A \cap B\ |}{|\ A \cup B\ |} = \frac{|\ A \cap B\ |}{|\ A\ | + |\ B\ | - |\ A \cap B\ |} \quad (1)$$

When the IoU is equal or bigger than 0.5, it counts as a True Positive (TP), otherwise as a False Positive (FP). The traffic signs which are not detected by Mask R-CNN are treated as False Negatives (FN).

The evaluation of detection approaches is performed most commonly with Precision-Recall values and the area under this curve (AUC). Precision indicates how good the model is on the detected traffic signs and it is computed with equation 2. Recall, on the other hand, refers to how many good detections were actually identified according to the ground truth (equation 3). Most of the related works that evaluated their detectors on the GTSDB, only considered the AUC and Recall values. Hence, we will use the same metrics for these detectors.

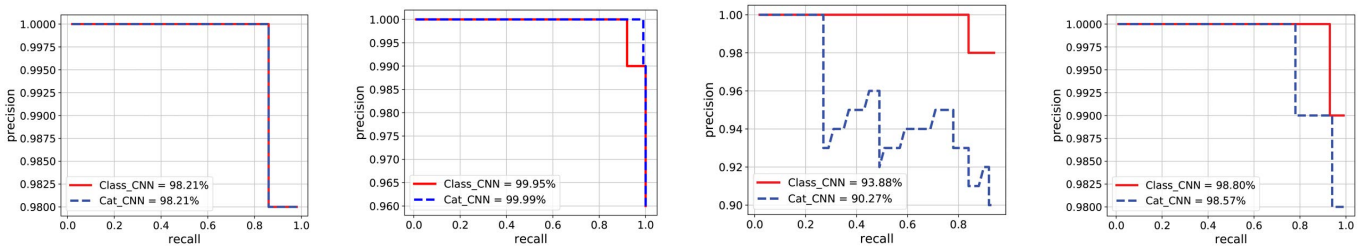
$$Precision = \frac{TP}{TP + FP} \quad (2)$$

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

Initially, and in order to compare the detection performance with other methods, we evaluate Mask R-CNN on the GTSDB using the fine-tuned weights obtained at epoch 46 to detect the original 4 Categories of the German symbol-based traffic signs. As Mask R-CNN only predicts if the detected RoI is a traffic sign, a lot of objects, including false matches, are passed to the Refinement module to perform some filtering and then pass its output to the classifier. This filtering process is carried out keeping only the detected RoIs whose confidence probability (class score) is equal or bigger than 0.8 (threshold set manually). As the class probability of a positive RoI does not influence in the detection, it gives a good notion of the actual traffic signs since negative or false matches tend to have low confidence probabilities. After this process is done, the inputs (filtered detected RoIs) are preprocessed as described in subsection V-A2. Next, the classifier receives the inputs which are not always traffic signs and has to be able to filter out the backgrounds while performing the category identification. This final filtering is made by the classifier, learning a background class together with the others.

The categories classifier, defined in Table I and trained with background class (first row in Table VII and the classes classifier, defined in Table II and trained as well with background class (All CNNs in Table VII), are evaluated together with Mask R-CNN to detect 4 categories on the GTSDB test set. Fig. 8 shows the precision-recall curves obtained with an IoU of 0.5. For the danger (Fig. 8a), prohibitory (Fig. 8b) and others (Fig. 8d) categories we can see that both CNNs performed almost the same, while for the mandatory category (Fig. 8c), the category classifier (Cat_CNN) has more troubles identifying the classes.

The detection performances are compared quantitatively with the state of the art in Table VIII. Threating all traffic signs as one category, we obtained directly from Mask R-CNN, 99% recall and 86% precision. The recall value reached, allow us to confirm that Mask R-CNN serves as good detector as it only missed to detect 5 signs of the 361 in the test set. However, for the precision value, we can see that, even if it is good detecting most of the correct traffic signs, it also detects several false matches (57). Nevertheless, after the



(a) Danger (b) Prohibitory (c) Mandatory (d) Others

Fig. 8. Precision-Recall curves of the detection module (Mask R-CNN) using 2 CNNs for category identification. Cat_CNN refers to the categories classifier defined in Table I, while Class_CNN refers to the classes classifier defined in Table II.

TABLE VIII
DETECTION RESULTS ON THE GTSDB TEST SET FOR 0.5 IOU (AUC VALUES)

Method	Danger	Prohibitory	Mandatory	Others
HOG+LDA+SVM [36]	99.91%	100%	100%	-
ICF [75]	100%	100%	96.98%	-
HOG+SVM [76]	98.85%	100%	92%	-
ROI+HOG+SVM [77]	98.72%	99.98%	95.76%	-
HOG+CNN [78]	99.73%	-	97.62%	-
ROI+HOG+SVM [22]	97.13%	99.29%	96.74%	-
ROI+Multi-task CNN [61]	98.34%	99.99%	98.72%	-
Faster-RCNN+CNN [14]	100%	96%	100%	99%
2Dpose-boundary CNN [79]	99.73%	99.89%	99.16%	-
modified YOLOv2 [80]	96.12%	96.81%	94.02%	-
MaskRCNN+Cat_CNN (Ours)	98.21%	99.99%	90.27%	98.57%
MaskRCNN+Class_CNN (Ours)	98.21%	99.95%	93.88%	98.80%

TABLE IX
DETECTION RESULTS ON THE EXTENDED GTSDB TEST SET WITH MASKRCNN+URBAN_CLASS_CNN FOR 0.5 IOU (AUC VALUES)

Category	AUC
Danger	98.41%
Priority	97.08%
Prohibitory	96.12%
Mandatory	93.50%
Special regulation	70.60%
Direction	66.02%
Additional panels	80.14%
Others	67.55%

classifier is applied, we are able to obtain a mean average precision mAP (considering the 4 Categories) of 96.76% with the categories classifier (Mask-RCNN+Cat_CNN) and 97.71% with the classes classifier (Mask-RCNN+Class_CNN). Such results let us affirm that, the classes classifier (deeper model) performs better than the Cat_CNN even if its classification accuracy was higher than learning all classes at once (Table V). Both CNNs have troubles learning the mandatory category as most of the state of the art methods shown in Table VIII. As the CNN with deeper layers (Class_CNN) performed better, we will consider it as the final classifier for our proposed system.

Methods based on HoG [36], [75] features obtained almost perfect results (see Table VIII), but these approaches are shape dependent, which makes them very specific for certain tasks. On the contrary, CNN based approaches are more suitable for general purposes, putting aside the tedious and hard work to chose the right features for specific shapes. Comparing our results with the state of the art methods that use CNNs, our proposed pipeline (Mask R-CNN + Class_CNN) is very competitive in terms of traffic sign detection including the Others (Unique) category that many studies did not take into account. Only authors in [14] (Faster-RCNN+CNN) considered this last category and reported very high AUC values in their results. However, they mentioned to have achieved a mAP of 84.5% and recall of 97.81%, which make us believe that their results reported are not actually AUC values, but instead recall ones. For this reason, we discarded the method Faster-RCNN+CNN [14] for comparison. Only methods [61], [79] perform slightly better than ours (around 1.3% and 1.8% mAP respectively) which makes our method competitive.

Furthermore, and in order to evaluate other important traffic signs, we performed detection on the proposed extended version of the GTSDB and analyzed the results. Mask R-CNN was run using the fine-tuned weights for this dataset, together with the classes classifier trained on the selected urban/rural subset of the ETSD (Urban CNN_classifier from Table V). The mAP obtained on this dataset is 83.68%, while the precision is 81% and recall 83%. Considering more challenging signs, the detector missed 140 signs (FN) and detected incorrectly 168 (FP). The AUC values for each category can be seen in Table IX.

According to Table VIII and Table IX, the detection rate for the danger category in the extended GTSDB, increased by 0.2% from the original GTSDB (see Fig. 9), even when the number of danger sign classes is higher in the urban set. The AUC value for the prohibitory and mandatory categories dropped from 3.79% and 0.38% respectively (Fig. 9). The reason behind this behavior is due to the variety of classes on the urban set and the sizes of the signs labeled. In the original GTSDB, only the most close and distinguishable signs are considered, while for the extended version, we labeled all signs visible and recognizable by human eye. The rest of the categories in the extended GTSDB performed poorly (below 90%) because the classes are harder to recognize due to the very different aspect ratios they possess and to the fact that they include text. By nature these signs are very difficult to recognize.

Since the Class_CNN outputs the label of the specific sign,

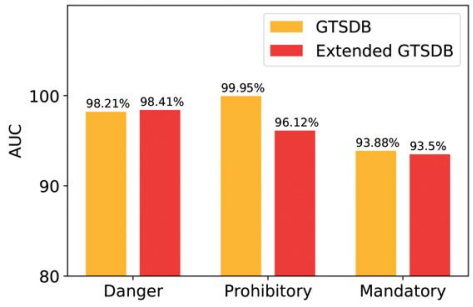


Fig. 9. Comparison between AUC values of the common categories between the GTSDB and its extended version.

TABLE X
CLASSIFICATION EVALUATION FOR GERMAN ROADS WITH THE GTSDB AND ITS PROPOSED EXTENDED VERSION TEST SETS. THE EVALUATION IS PERFORMED ON THE DETECTED SIGNS EXTRACTED BY MASK R-CNN + THE URBAN CLASS_CNN.

Category	Original GTSDB			Extended GTSDB		
	Detected	Correct class	Accuracy	Detected	Correct class	Accuracy
Danger	62	62	100.00%	59	59	100.00%
Priority	-	-	-	75	72	96.00%
Prohibitory	165	162	98.18%	202	187	92.57%
Mandatory	45	45	100.00%	63	53	84.13%
Special regulation	-	-	-	58	48	82.76%
Direction	-	-	-	124	101	81.45%
Additional panels	-	-	-	44	39	88.64%
Others	88	87	98.86%	76	63	82.89%
All	360	356	98.89%	701	622	88.73%

and from this label we infer it's category, Fig. 10 illustrates some examples of the results obtained by our recognition system. The traffic sign accuracy indicated in green text at the bottom left in every image, corresponds to the classification accuracy obtained for the detected signs only in that image. The column on the left represents results computed in the GTSDB recognizing 43 classes (4 Categories), while the column on the right, shows detection and classification results obtained on the extended version of the GTSDB recognizing 132 classes from the urban set extracted from the ETSD [15]. In the extended version, we are able to detect and classify directional signs, obstacle signs, additional panels as well as special regulation signs. Hence, a more reliable interpretation of the environment for driving scenarios can be made recognizing all traffic signs with our proposed system pipeline.

Additionally, we provide classification evaluation on the traffic sign proposals detected by Mask R-CNN with the Class_CNN trained with background class on the GTSRB and urban set of the ETSD [15]. We evaluate the recognition on the GTSDB and its extended version (see Table X). For the GTSDB we have contemplated 360 signs after filtering processes (detection class score $\geq$ 0.8 and classification probability ≥ 0.9) from which only 356 are classified correctly obtaining a classification accuracy of 98.89%. In the extended GTSDB, 701 signs were detected, where 622 are predicted correctly resulting in 88.73% accuracy. It should be noted that the Others category does not include the same classes in both datasets but are displayed like that in order to match the naming convention. Fig. 11 and Fig. 12 show some examples of the misclassified detected signs for the GTSDB and its extended version respectively. In Fig. 11, the detected signs are not labeled in the ground truth indicating that their label is background class (GT:43). For Fig. 12, we have the same issue, even if some signs seem to be traffic signs, because they were not recognizable by human eye, we did not label them.

Yang et al. [22] reported 98.24% accuracy recognition rate after all the signs were detected. Comparing their method to ours, we surpass their performance with 0.65% accuracy more. Other studies do not provide the recognition rate after the detection module. For this reason it is not possible to compare with them. Along with this accuracy, we evaluated the classification rate on the detected signs per image, as shown in the examples of our recognition system (Fig. 10) and we averaged the results, obtaining 99.39% with MaskRCNN+Class_CNN (43 classes + background) on the GTSDB test set (300 images) and 89.97% with MaskRCNN+Urban_Class_CNN (132 classes + background) on the extended GTSDB test set. Since the complexity of every image varies a lot, this type of evaluation is useful if individual outputs are required giving the same weight to each image.

In the same way, the running time is not considered in this study for comparison purposes, because the time reported by each approach depends on the hardware capabilities where they were tested. Nonetheless, we can give the reader an estimation of the running time of our traffic sign recognition system tested in our PC in GPU mode. For an image size of 1280×960 pixels, it takes approximately 0.32 seconds to detect and classify all traffic signs. In other words, our system is capable of running at $\sim 3.3 fps$ which makes it relatively a good choice for real-world applications.

Interestingly, no matter the difficulty of some categories, like additional panels, special regulation and, others; the recognition system needs to consider them to interpret correctly the driving rules in a similar way to how humans do. For this reason, our proposed system is a good choice for Driver Assistant Systems (ADAS) and autonomous vehicles.

VI. CONCLUSION

We proposed a symbol and text-based traffic sign recognition system for European urban environments. The detection module (Mask R-CNN) identifies possible traffic signs while at the same time segments a mask for each of them. These masks serve as a localization refinement avoiding other post processing tasks, like shape estimation, to provide an accurate RoI. The refinement module is introduced to discard objects that may not be traffic signs and pre-process the inputs before passing them to the next module. The classification module, performed with a proposed CNN, filters out the RoI inputs which are background and recognizes the specific classes and categories of the traffic signs.

The proposed Class_CNN architecture obtains a top accuracy of 99.61% on the GTSRB test set, while at the same time, its learning process is very stable using Dropblock as regularization technique. The performance of our CNN architecture provides a good trade off between the number and parameters and computational time compared to other methods

in the state of the art, which most of the time, use an ensemble model with severals CNNs.

A urban/rural subset subtracted from the ETSD [15], allowed us to consider a complete dataset with symbol and text signs dealing at the same time with intra-class variability from 6 countries (Belgium, Croatia, France, Germany, Netherlands and Sweden). Such characteristics are of vital importance for a system to recognize properly the same class, even if small variations are presented when driving from one place to another, or changing countries.

Simultaneously, we proposed an extended version of the GTSDB [13] labeling all the missing signs not included in the original dataset, but considering only the classes defined on the ETSD. This due to the reason that for other signs non included in the ETSD, the number of images will not be enough to learn that class, specially if some signs are presented a few times in the GTSDB. For the labeled signs, we provide the RoIs and masks with their respective classes, together with a dataset for instance or semantic segmentation approaches. Such a dataset, allows us to detect and classify traffic signs from several categories including: directional, priority, special regulation, additional panels and others, which provide important and complementary information to interpret traffic rules.

The recognition system works with image size of 1280×960 and provides an overall recall detection rate of 99% on the GTSDB test set with 98.89% accuracy classification on the detected signs and an average accuracy (accuracy per image) of 99.39%. For the proposed extended version of the GTSDB, it obtains 83% recall detecting symbol and text signs while achieving 88.73% accuracy when recognizing them and 89.97% accuracy as an average (evaluating images individually). Detection and classification dropped on the extended GTSDB version due to the nature of the classes it contains. The most challenging ones are based on text and with very different aspect ratios (belonging to Directional and Additional panels categories).

By nature, small signs like Additional panels are hard to detect, which makes them the most difficult ones for detection approaches as the portion they occupy in the image is relatively small compared to other objects in the scene. At the same time, text-based signs are confused with other objects containing text, like advertisements or store signs due to the variability of appearances they might have.

In future work we intent to: 1) take into account techniques like data augmentation in order to increase the detection performance of Mask R-CNN, since the training set of the GTSDB is quite small (600 images); 2) test the proposed traffic sign recognition system for French scenarios and evaluate the results, 3) train other detectors based on CNNs to evaluate the running times and compare their performances with the extended version of the GTSDB; and 4) apply GANs like authors in [47] did, to generate features for the very small traffic signs similar to the ones of clear and big signs, in order to improve our detection and recognition.

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[2] E. C. for Europe-Inland Transport Committee *et al.*, "Convention on road signs and signals," *United Nations Treaty Series*, vol. 1091, p. 3, 1968.

[3] C. Grigorescu and N. Petkov, "Distance sets for shape filters and shape recognition," *IEEE transactions on image processing*, vol. 12, no. 10, pp. 1274–1286, 2003.

[4] S. Šegvic, K. Brkić, Z. Kalafatić, V. Stanisavljević, M. Ševrović, D. Budimir, and I. Dadić, "A computer vision assisted geoinformation inventory for traffic infrastructure," in *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*. IEEE, 2010, pp. 66–73.

[5] F. Larsson and M. Felsberg, "Using Fourier descriptors and spatial models for traffic sign recognition," in *Scandinavian Conference on Image Analysis*. Springer, 2011, pp. 238–249.

[6] R. Timofte and L. Van Gool, "Sparse representation based projections," in *Proceedings of the 22nd British machine vision conference-BMVC 2011*. Citeseer, 2011, pp. 61–1.

[7] N. Paparoditis, J.-P. Papelard, B. Cannelle, A. Devaux, B. Soheilian, N. David, and E. Houzay, "Stereopolis ii: A multi-purpose and multi-sensor 3d mobile mapping system for street visualisation and 3d metrology," *Revue française de photogrammétrie et de télédétection*, vol. 200, no. 1, pp. 69–79, 2012.

[8] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition," *Neural networks*, vol. 32, pp. 323–332, 2012.

[9] R. Timofte, K. Zimmermann, and L. Van Gool, "Multi-view traffic sign detection, recognition, and 3d localisation," *Machine vision and applications*, vol. 25, no. 3, pp. 633–647, 2014.

[10] M. M. Lau, K. H. Lim *et al.*, "Malaysia traffic sign recognition with convolutional neural network," in *Digital Signal Processing (DSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 1006–1010.

[11] A. Møgelmose, M. M. Trivedi, and T. B. Moeslund, "Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey." *IEEE Trans. Intelligent Transportation Systems*, vol. 13, no. 4, pp. 1484–1497, 2012.

[12] S. Jung, U. Lee, J. Jung, and D. H. Shim, "Real-time traffic sign recognition system with deep convolutional neural network," in *The 13th International Conference on Ubiquitous Robots and Ambient Intelligence*. URAI, 2016.

[13] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, and C. Igel, "Detection of traffic signs in real-world images: The german traffic sign detection benchmark," in *Neural Networks (IJCNN), The 2013 International Joint Conference on*. IEEE, 2013, pp. 1–8.

[14] J. Li and Z. Wang, "Real-time traffic sign recognition based on efficient cnns in the wild," *IEEE Transactions on Intelligent Transportation Systems*, no. 99, pp. 1–10, 2018.

[15] C. G. Serna and Y. Ruichek, "Classification of traffic signs: The european dataset," *IEEE Access*, 2018.

[16] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Computer Vision (ICCV), 2017 IEEE International Conference on*. IEEE, 2017, pp. 2980–2988.

[17] G. Neuhold, T. Ollmann, S. R. Bulò, and P. Kontschieder, "The mapillary vistas dataset for semantic understanding of street scenes." in *ICCV*, 2017, pp. 5000–5009.

[18] C. Bahlmann, Y. Zhu, V. Ramesh, M. Pellkofer, and T. Koehler, "A system for traffic sign detection, tracking, and recognition using color, shape, and motion information," in *Intelligent Vehicles Symposium, 2005. Proceedings. IEEE*. IEEE, 2005, pp. 255–260.

[19] W.-J. Kuo and C.-C. Lin, "Two-stage road sign detection and recognition," in *Multimedia and Expo, 2007 IEEE International Conference on*. IEEE, 2007, pp. 1427–1430.

[20] S. Maldonado-Bascón, S. Lafuente-Arroyo, P. Gil-Jimenez, H. Gómez-Moreno, and F. López-Ferreras, "Road-sign detection and recognition based on support vector machines," *IEEE transactions on intelligent transportation systems*, vol. 8, no. 2, pp. 264–278, 2007.

[21] Á. González, M. Á. Garrido, D. F. Llorca, M. Gavilán, J. P. Fernández, P. F. Alcantarilla, I. Parra, F. Herranz, L. M. Bergasa, M. Á. G. Sotelo *et al.*, "Automatic traffic signs and panels inspection system using computer vision," *IEEE Transactions on intelligent transportation systems*, vol. 12, no. 2, pp. 485–499, 2011.

[22] Y. Yang, H. Luo, H. Xu, and F. Wu, "Towards real-time traffic sign detection and classification," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 7, pp. 2022–2031, 2016.

[23] B. Tian, R. Chen, Y. Yao, and N. Li, "Robust traffic sign detection in complex road environments," in *Vehicular Electronics and Safety (ICVES), 2016 IEEE International Conference on*. IEEE, 2016, pp. 1–5.

[24] A. De La Escalera, J. M. Armingol, J. M. Pastor, and F. J. Rodríguez, "Visual sign information extraction and identification by deformable models for intelligent vehicles," *IEEE transactions on intelligent transportation systems*, vol. 5, no. 2, pp. 57–68, 2004.

[25] G. Overett, L. Petersson, L. Andersson, and N. Pettersson, "Boosting a heterogeneous pool of fast hog features for pedestrian and sign detection," in *Intelligent Vehicles Symposium, 2009 IEEE*. IEEE, 2009, pp. 584–590.

[26] M. A. García-Garrido, M. Ocana, D. F. Llorca, E. Arroyo, J. Pozuelo, and M. Gavilán, "Complete vision-based traffic sign recognition supported by an i2v communication system," *Sensors*, vol. 12, no. 2, pp. 1148–1169, 2012.

[27] N. Barnes and A. Zelinsky, "Real-time radial symmetry for speed sign detection," in *Intelligent Vehicles Symposium, 2004 IEEE*. IEEE, 2004, pp. 566–571.

[28] N. Barnes and G. Loy, "Real-time regular polygonal sign detection," in *Field and Service Robotics*. Springer, 2006, pp. 55–66.

[29] D. M. Gavrila, "Traffic sign recognition revisited," in *Mustererkennung 1999*. Springer, 1999, pp. 86–93.

[30] G. Loy and N. Barnes, "Fast shape-based road sign detection for a driver assistance system," in *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, vol. 1. IEEE, 2004, pp. 70–75.

[31] K.-H. Jo *et al.*, "Information retrieval of led text on electronic road sign for driver-assistance system using spatial-based feature and nearest cluster neighbor classifier," in *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*. IEEE, 2014, pp. 531–536.

[32] J. Yamamoto, S. Karungaru, and K. Terada, "Japanese road signs recognition using neural networks," in *SICE Annual Conference (SICE), 2013 Proceedings of*. IEEE, 2013, pp. 1144–1150.

[33] D. Wang, S. Yue, J. Xu, X. Hou, and C.-L. Liu, "A saliency-based cascade method for fast traffic sign detection," in *Intelligent Vehicles Symposium (IV), 2015 IEEE*. IEEE, 2015, pp. 180–185.

[34] C. Liu, F. Chang, and Z. Chen, "Rapid multiclass traffic sign detection in high-resolution images," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 6, pp. 2394–2403, 2014.

[35] C. Liu, F. Chang, and C. Liu, "Occlusion-robust traffic sign detection via cascaded colour cubic feature," *IET Intelligent Transport Systems*, vol. 10, no. 5, pp. 354–360, 2016.

[36] G. Wang, G. Ren, Z. Wu, Y. Zhao, and L. Jiang, "A robust, coarse-to-fine traffic sign detection method," in *Neural Networks (IJCNN), The 2013 International Joint Conference on*. IEEE, 2013, pp. 1–5.

[37] Y. Yang and F. Wu, "Real-time traffic sign detection via color probability model and integral channel features," in *Chinese Conference on Pattern Recognition*. Springer, 2014, pp. 545–554.

[38] Y. Yuan, Z. Xiong, and Q. Wang, "An incremental framework for video-based traffic sign detection, tracking, and recognition," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 7, pp. 1918–1929, 2017.

[39] Y. Saadna and A. Behloul, "An overview of traffic sign detection and classification methods," *International Journal of Multimedia Information Retrieval*, vol. 6, no. 3, pp. 193–210, 2017.

[40] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[41] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International journal of computer vision*, vol. 104, no. 2, pp. 154–171, 2013.

[42] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[43] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[44] R. Qian, Q. Liu, Y. Yue, F. Coenen, and B. Zhang, "Road surface traffic sign detection with hybrid region proposal and fast r-cnn," in *Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD), 2016 12th International Conference on*. IEEE, 2016, pp. 555–559.

[45] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," *Image and vision computing*, vol. 22, no. 10, pp. 761–767, 2004.

[46] C. L. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *European conference on computer vision*. Springer, 2014, pp. 391–405.

[47] W. Huang, M. Huang, and Y. Zhang, "Detection of traffic signs based on combination of gan and faster-rcnn," in *Journal of Physics: Conference Series*, vol. 1069, no. 1. IOP Publishing, 2018, p. 012159.

[48] H. Fleyeh and M. Dougherty, "Road and traffic sign detection and recognition," in *Proceedings of the 16th Mini-EURO Conference and 10th Meeting of EWGT*, 2005, pp. 644–653.

[49] H. H. Aghdam, E. J. Heravi, and D. Puig, "A practical and highly optimized convolutional neural network for classifying traffic signs in real-time," *International Journal of Computer Vision*, vol. 122, no. 2, pp. 246–269, 2017.

[50] S. Maldonado-Bascon, S. Lafuente-Arroyo, P. Siegmann, H. Gomez-Moreno, and F. Acevedo-Rodriguez, "Traffic sign recognition system for inventory purposes," in *Intelligent Vehicles Symposium, 2008 IEEE*. IEEE, 2008, pp. 590–595.

[51] G. Piccioli, E. De Micheli, P. Parodi, and M. Campani, "Robust method for road sign detection and recognition," *Image and Vision Computing*, vol. 14, no. 3, pp. 209–223, 1996.

[52] X. W. Gao, L. Podladchikova, D. Shaposhnikov, K. Hong, and N. Shevtsova, "Recognition of traffic signs based on their colour and shape features extracted using human vision models," *Journal of Visual Communication and Image Representation*, vol. 17, no. 4, pp. 675–685, 2006.

[53] A. Ruta, Y. Li, and X. Liu, "Robust class similarity measure for traffic sign recognition," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 4, pp. 846–855, 2010.

[54] F. Zaklouta and B. Stanciulescu, "Real-time traffic sign recognition in three stages," *Robotics and autonomous systems*, vol. 62, no. 1, pp. 16–24, 2014.

[55] S. M. Bascón, J. A. Rodríguez, S. L. Arroyo, A. F. Caballero, and F. López-Ferreras, "An optimization on pictogram identification for the road-sign recognition task using svms," *Computer Vision and Image Understanding*, vol. 114, no. 3, pp. 373–383, 2010.

[56] P. Sermanet and Y. LeCun, "Traffic sign recognition with multi-scale convolutional networks," in *Neural Networks (IJCNN), The 2011 International Joint Conference on*. IEEE, 2011, pp. 2809–2813.

[57] D. Cireşan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," *arXiv preprint arXiv:1202.2745*, 2012.

[58] R. Belaroussi, P. Foucher, J.-P. Tarel, B. Soheilian, P. Charbonnier, and N. Paparoditis, "Road sign detection in images: A case study," in *Pattern Recognition (ICPR), 2010 20th International Conference on*. IEEE, 2010, pp. 484–488.

[59] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[60] J. Jin, K. Fu, and C. Zhang, "Traffic sign recognition with hinge loss trained convolutional neural networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 5, pp. 1991–2000, 2014.

[61] H. Luo, Y. Yang, B. Tong, F. Wu, and B. Fan, "Traffic sign recognition using a multi-task convolutional neural network," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 4, pp. 1100–1111, 2018.

[62] Á. Arcos-García, J. A. Álvarez-García, and L. M. Soria-Morillo, "Deep neural network for traffic sign recognition systems: An analysis of spatial transformers and stochastic optimisation methods," *Neural Networks*, vol. 99, pp. 158–165, 2018.

[63] X. Changzhen, W. Cong, M. Weixin, and S. Yanmei, "A traffic sign detection algorithm based on deep convolutional neural network," in *Signal and Image Processing (ICSIP), IEEE International Conference on*. IEEE, 2016, pp. 676–679.

[64] J. Li, X. Liang, Y. Wei, T. Xu, J. Feng, and S. Yan, "Perceptual generative adversarial networks for small object detection," in *IEEE CVPR*, 2017.

[65] Á. Arcos-García, J. A. Álvarez-García, and L. M. Soria-Morillo, "Evaluation of deep neural networks for traffic sign detection systems," *Neurocomputing*, vol. 316, pp. 332–344, 2018.

[66] W. Abdulla, "Mask r-cnn for object detection and instance segmentation on keras and tensorflow," https://github.com/matterport/Mask_RCNN, 2017.

[67] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[68] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[69] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from over-

fitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.
[70] G. Ghiasi, T.-Y. Lin, and Q. V. Le, "Dropblock: A regularization method for convolutional networks," in *Advances in Neural Information Processing Systems*, 2018, pp. 10 748–10 758.
[71] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.
[72] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.
[73] P. Meletis and G. Dubbelman, "Training of convolutional networks on multiple heterogeneous datasets for street scene semantic segmentation," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, 2018.
[74] A. Geiger, M. Lauer, C. Wojek, C. Stiller, and R. Urtasun, "3d traffic scene understanding from movable platforms," *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 5, pp. 1012–1025, 2014.
[75] M. Mathias, R. Timofte, R. Benenson, and L. Van Gool, "Traffic sign recognition—how far are we from the solution?" in *Neural Networks (IJCNN), The 2013 International Joint Conference on*. IEEE, 2013, pp. 1–8.
[76] M. Liang, M. Yuan, X. Hu, J. Li, and H. Liu, "Traffic sign detection by roi extraction and histogram features-based recognition," in *Neural Networks (IJCNN), The 2013 International Joint Conference on*. IEEE, 2013, pp. 1–8.
[77] S. Salti, A. Petrelli, F. Tombari, N. Fioraio, and L. Di Stefano, "A traffic sign detection pipeline based on interest region extraction," in *Neural Networks (IJCNN), The 2013 International Joint Conference on*. IEEE, 2013, pp. 1–7.
[78] Y. Wu, Y. Liu, J. Li, H. Liu, and X. Hu, "Traffic sign detection based on convolutional neural networks," in *Neural Networks (IJCNN), The 2013 International Joint Conference on*. Citeseer, 2013, pp. 1–7.
[79] H. S. Lee and K. Kim, "Simultaneous traffic sign detection and boundary estimation using convolutional neural network," *IEEE Transactions on Intelligent Transportation Systems*, 2018.
[80] J. Zhang, M. Huang, X. Jin, and X. Li, "A real-time chinese traffic sign detection algorithm based on modified yolov2," *Algorithms*, vol. 10, no. 4, p. 127, 2017.

Citlalli Gámez Serna received the B.S. degree in Information and Communication Technologies from ITESM, Mexico in 2011, and the M.S. degree in Color in Informatics and Media Technology (Erasmus program) in 2014 from Université Jean Monnet in France and Gjovik University College in Norway. She is currently pursuing her PhD degree in Informatics and Automation with the Systems and Transportations Laboratory at UTBM, France. Her research is focused on path tracking for intelligent vehicles covering deep learning techniques for the environment perception and localization together with the control part.

Yassine Ruichek received the Ph.D. degree in control and computer engineering and the Habilitation à Diriger des Recherches (HDR) degree in physic science from the University of Lille, France, in 1997 and 2005, respectively. Since 2007, he has been a Full Professor with the University of Technology of Belfort-Montbéliard. His research interests are concerned with multisensory data based perception and localization, including computer vision, pattern recognition and classification, machine learning, data fusion, with applications to intelligent transportation systems and video surveillance.

(a) GTSDB (b) Extended GTSDB

Fig. 10. Example of detection and classification results by our proposed traffic sign recognition system. Column (a) shows some images trained and evaluated on the original GTSDB using Mask R-CNN + Class_CNN trained on the GTSRB to detect 43 classes. Column (b) refers to the detection with Mask R-CNN on the extended version of the GTSDB + Urban_Class_CNN trained in the urban set of the ETSD to detect 132 classes.

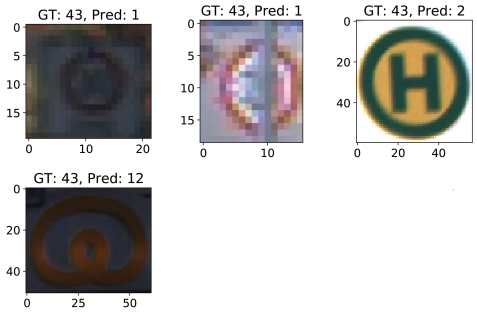


Fig. 11. Incorrect predictions after detection is performed on the GTSDB with the Class_CNN trained on the GTSRB.

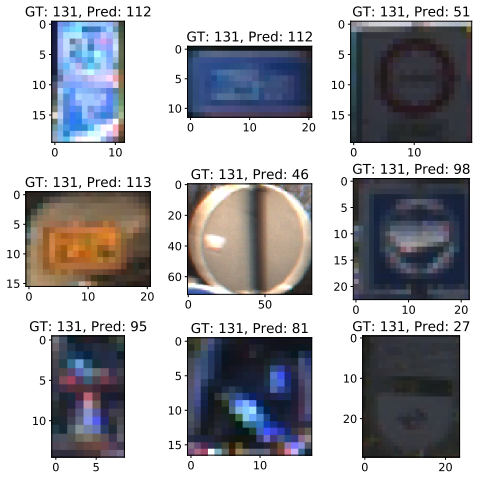


Fig. 12. Incorrect predictions after detection is performed on the GTSDB extended version with the Urban_Class_CNN trained in the selected urban set of the ETSD.