

# Deep Traffic Sign Detection and Recognition Without Target Domain Real Images

Lucas Tabelini, Rodrigo Berriel, Thiago M. Paixão, Alberto F. De Souza,  
Claudine Badue, Nicu Sebe and Thiago Oliveira-Santos

**Abstract**—Deep learning has been successfully applied to several problems related to autonomous driving, often relying on large databases of real target-domain images for proper training. The acquisition of such real-world data is not always possible in the self-driving context, and sometimes their annotation is not feasible. Moreover, in many tasks, there is an intrinsic data imbalance that most learning-based methods struggle to cope with. Particularly, traffic sign detection is a challenging problem in which these three issues are seen altogether. To address these challenges, we propose a novel database generation method that requires only (i) arbitrary natural images, i.e., requires no real image from the target-domain, and (ii) templates of the traffic signs. The method does not aim at overcoming the training with real data, but to be a compatible alternative when the real data is not available. The effortlessly generated database is shown to be effective for the training of a deep detector on traffic signs from multiple countries. On large data sets, training with a fully synthetic data set almost matches the performance of training with a real one. When compared to training with a smaller data set of real images, training with synthetic images increased the accuracy by 12.25%. The proposed method also improves the performance of the detector when target-domain data are available.

**Index Terms**—Traffic Sign Detection, Deep Learning, Autonomous Driving, Object Detection, Faster R-CNN, Template

## I. INTRODUCTION

DEEP neural networks (DNNs) have been widely used to tackle a variety of computer vision tasks, particularly on several problems related to autonomous driving [1]. Many of these applications rely on large networks which usually require large amounts of data to be properly trained. This requirement, however, is not always easy to be fulfilled. Acquiring problem-specific real-world databases, especially in robotics, is often a hard task, particularly when considering the additional annotation process. In this context, it is desirable to produce high-performance models without the need of annotated real-world images.

The success of deep learning on autonomous driving and on advanced driver assistance systems (ADAS) has been demonstrated on several applications: scene semantic segmentation [2], traffic light detection [3], crosswalk classification [4], traffic sign detection [5], pedestrian analysis [6], car heading

direction estimation [7] and many other applications. This work focuses on the traffic sign detection problem, whose goal is to locate signs of interest along the road (with the help of a camera mounted on a vehicle) and classify their specific type (e.g., whether it is a 60 or a 80 km/h sign). This is an important task to be performed by autonomous driving systems and ADAS because traffic signs set rules which (i) the drivers are expected to abide by and (ii) road users should rely on while making decisions.

The traffic sign detection problem has been investigated by the research community for a while. Researchers have been proposing all types of solutions such as the ones using hand-crafted features in model-based solutions [8], leveraging simple features in learning-based approaches [9], and, the more recent and state-of-the-art, using deep learning based methods [5], [10] that is the focus of this work.

Apart from the major advances on the topic, there are still many issues requiring further investigation, specially when considering deep learning approaches for detection. In general, the training of deep detectors require (i) expensive annotation, (ii) real images from the target domain, and (iii) balanced data sets. The annotation process is expensive because each traffic sign has to be marked with a bounding box, which is more difficult than just assigning a class for an object in a classification problem. Moreover, deep detectors are still known for being data hungry, i.e., they require many real images to perform well. Therefore, the acquisition of such images with traffic signs can be troublesome, because it requires finding many traffic sign samples along the roads. Since traffic legislation changes from country to country, the traffic signs are not standardized across the world and a new data set has to be created for every country. Finally, the image acquisition process should yield a balanced number of exemplars of each class. This would require collecting many more images to have a minimum balance across the classes because some traffic signs are rarer than others under common driving circumstances, causing a long-tail effect [5].

In this context, we hypothesize that the training of a deep traffic sign detector without annotated real images from the domain of interest (i.e., target domain real images) can yield a performance similar to those models trained on manually annotated images from the domain of interest. The method does not aim at overcoming the training with real data, but to be a compatible alternative when the real data is not available. Therefore, this work proposes a novel effortless method for generation of synthetic databases that requires only (i) arbitrary natural images (i.e., images out of the domain

L. Tabelini, R. Berriel, C. Badue, A. F. De Souza, and T. Oliveira-Santos are with Universidade Federal do Espírito Santo (UFES), Brazil.

T. M. Paixão is with Instituto Federal do Espírito Santo (IFES), Serra, Brazil, and Universidade Federal do Espírito Santo (UFES), Brazil.

Nicu Sebe is with University of Trento, Trento, Italy.

or context of interest) as background and (ii) templates of the traffic signs (i.e., synthetic representative images of the different traffic signs). The synthetic database is used to train a country-optimized deep traffic sign detector. We argue that, in the context of traffic sign detection, the proposed database generation process handles the three previously mentioned issues altogether facilitating the training of country-specific detectors. In a preliminary work [11], the hypothesis was verified, but only for a simplified case, as detailed next.

The traffic sign detection task can be divided in two steps: localization and recognition. Although the first step is crucial, it becomes more useful in practice when is followed by the second step. It is not enough to know that there are traffic signs in the scene, since different traffic signs convey different information. It is in the recognition step that traffic signs are distinguished, for example, to verify the speed limit on the road. In the preliminary version of this paper [11], the recognition step was not addressed since the focus was precisely on locating traffic signs in the scene. In that occasion, the proposed method was shown to be effective on the localization task for German traffic signs. This work extends and consolidates the preliminary investigation by including: the recognition task, an extensive experimentation, and a more in-depth discussion of the results. Although the recognition step implies significant additional complexity, experimental results show that the proposed method is successful, which makes it useful in practice. In the experimentation, two large data sets were added, both more than ten times larger than the one used in the preliminary work. To assess the impact of each step in the proposed method, an ablation study was performed. In addition, the previous work raised a question: how many real images are equivalent to using synthetic ones? This work shows that the amount of real data required to match the performance of the system trained with synthetic data only is huge for two data sets (in the order of tens of thousands). Finally, two additional use cases were investigated for the synthetic data, data augmentation and pretrain. The data augmentation investigation shows that the synthetic data together with a few real data samples can remarkably boost the performance of a detector to match those trained with a large amount of real images. The pretrain investigation shows that the proposed synthetic data generation process is useful even when a large amount of real data from the target domain is available. Results show that pretraining the detector with the synthetic data generated with our method improves the learning of the detection model and substantially increases the final accuracy.

## II. RELATED WORKS

This section presents a brief review of general traffic sign detection methods and of methods for generating synthetic data for deep training.

### A. Traffic Sign Detection

In the past, most methods proposed in the literature to tackle the problem of traffic sign detection used classic computer

vision approaches. Since traffic signs may be easily distinguished by humans due to the contrasting colors, the first works took advantage of those features [12]. In this context, techniques such as color thresholding are used to segment the traffic signs, followed by a post-processing step [13]. Although those methods are generally faster than others, they are not robust to factors such as weather conditions, occlusion, or time of the day [14]. Other approaches take advantage of the well-defined shapes of traffic signs. To detect them by the shape, many methods may be used, but the most common ones employ the Hough transform [15]. Although these two approaches (based on color and shape) work well separately, the results can be improved if they are combined [16]. When combined, either method can be used first, while the second is usually applied to filter the results. With the traffic signs detected, the next step is the recognition. For this step, the most common methods are neural networks [17], [18], genetic algorithms [19], AdaBoost classifiers [20] and SVMs [21]. More recently, with data becoming easier to acquire, deep learning has also shown success on the traffic sign recognition task, particularly with the use of convolutional neural networks (CNNs) [22], [23]. In 2010, a benchmark for traffic sign recognition was proposed [24]. The best results were achieved using CNNs [25]. It is important to note that in many works that tackle the traffic sign detection problem with deep learning the classification is done only to distinguish between super-categories and not between specific classes (e.g., a 70 or 80 km/h traffic sign has the same class label) [10], [23]. The main issue in most works with deep detectors is the need for large amounts of data. In this work, we propose a method to train a deep detector with no real data and focus on distinguishing between every specific class (fine-grained class recognition). For a more in-depth review of traffic sign detection the reader can refer to [26].

### B. Synthetic Data Generation for Deep Training

Methods to generate synthetic data for deep training have been extensively studied in the past. Those methods can be divided in two groups: non-learning (i.e., there is no learning during the synthetic data generation) and learning based (i.e., there is learning during the synthetic data generation). For the detection task, most non-learning based methods cut and paste objects from an image to another. For instance, Dwibedi et al. [27] cut objects with a segmentation mask and then blend them on other images from the target domain. Wang et al. [28] use a similar approach, where different objects from the same category switch positions. 3D models have also been used to generate data for training, overlaying 2D renderings of those models on real images [29]. In the task of traffic sign classification, some works used image processing to generate training samples from traffic sign templates [30], [31]. In the detection task, Møgelmoose et al. [32] tried using synthetic data to train a Viola-Jones traffic sign detector, but the results were not satisfactory. Learning based approaches are motivated by the premise that training the model with more realistic synthetic data will lead to a better performance on real-world data. The learning process can be used to, for example,

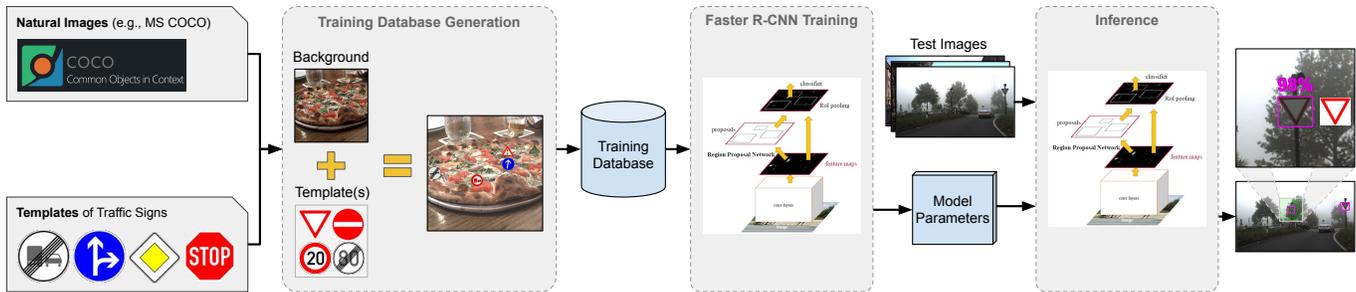


Fig. 1. Overview of the proposal method. From left to right, the method receives as input natural images (e.g., from publicly available large-scale databases) and templates of traffic signs and generates a synthetic training database. The synthetic database is used to train a deep detector (e.g., Faster R-CNN). Finally, the model is ready to detect and recognize traffic signs.

generate data with objects in a more natural position, as Dvornik et al. [33] have shown that cutting and pasting objects at random positions may not be ideal. In particular, Georgakis et al. [34] place cropped objects of interest from public data sets on locations that are most likely to be a surface, being such positions estimated via semantic segmentation. The scale of the objects to be placed is determined according to the depth value associated to the surface position. Gupta et al. [35] use a similar approach for text localization, predicting a depth map of each background image. With the depth map, regions are filtered to gather suitable regions for text placement. Then, the text is superimposed on those regions, also using the depth map to determine the text’s perspective. In the context of classification using one-shot learning, Grigorescu [36] proposes to generate data with predefined functions that make templates more realistic. To set those functions’ parameters, a network is trained using templates (called one shot objects, in his work) and real traffic sign samples. Kim et al. [37] propose an approach with a variational prototyping-encoder. In the training process, real training images are encoded to a latent space and then decoded to a prototype (template). In the testing phase, the encoder is used as a feature extractor and a nearest neighbor classifier is used on the features extracted from the test image and the templates.

In all the aforementioned works, the proposed method is either evaluated on the classification task only or requires annotated real-world data from the problem domain. In this work, we tackle the detection task with fine-grained class recognition using no problem domain real-world data.

### III. PROPOSED METHOD

The proposed method (illustrated in Figure 1) comprises mainly the generation of a synthetic training data set that requires no real image from the domain of interest. After that, this synthetic data set is used to train a deep traffic sign detector. Finally, the trained deep detector model can be used to infer the position and the class of traffic signs on real images.

#### A. Training Database Generation

The generation of the training database is three-fold. First, templates of the traffic signs of interest are acquired. Then,

background images that do not belong to the domain of interest are collected (e.g., arbitrary natural images). Lastly, the training samples, comprising images with annotated traffic signs, are generated.

**Template acquisition.** The first step towards the generation of the training samples is the acquisition of a template for each traffic sign of interest. The traffic signs of interest are those which the system is expected to operate with. Frequently, traffic signs are part of country-wise specific legislations defined by governmental agencies. This usual country-wise standardization helps the acquisition of templates (which is the goal of this step), because their very definitions (i.e., the templates) are part of pieces of legislation commonly available on-line on the websites of these agencies. In fact, templates are graphic representations of these definitions. In addition, some of the publicly available data sets for traffic sign detection (e.g., [38]) also distribute the templates of the classes annotated in their samples. All of this makes it easy and convenient to acquire templates virtually for any given set of standardized traffic signs worldwide. In case the templates are not available online, it is always possible to draw it manually. These templates are acquired and stored to be used later.

**Background acquisition.** In addition to the templates, the system requires images to use as background of the training samples. Although the natural choice would be images that belong to the domain of interest, e.g., images of roads, highways, streets, etc., we argue that this is not required. Moreover, we believe that choosing background images from the domain of interest may introduce unwanted noise in the training data if not carefully annotated. Images from the domain of interest eventually will present the object of interest, which, in turn, will be treated as background as well. By not constraining the background acquisition to images of the domain of interest, many of the freely available large-scale data sets (e.g., ImageNet [39], Microsoft COCO [40], etc.) can be exploited. For this work, the Microsoft COCO data set was chosen to be used as background, except for the images containing classes that are closely related to the domain of interest in order to avoid the introduction of noise in the training data set. Details of the background acquisition are presented in Section IV. After choosing the background images, the training samples can be generated.

**Training samples generation.** The last step of the training database generation is blending the background images and the templates of the traffic signs. This blending process aims to reduce the appearance difference between the background and the templates. If successful, the samples generation process is advantageous because it may tackle all the three aforementioned issues: (i) the training samples are automatically annotated, since the position of the traffic signs on the image is defined by the method; (ii) a large-scale database can be generated without much cost, given that there are a lot of different possible combinations between the random natural images (i.e., the backgrounds) and the objects of interest (i.e., the transformed traffic sign templates); and (iii) the training data set will not suffer from imbalance, since the method can sample the classes uniformly. The details of the blending process are described next.

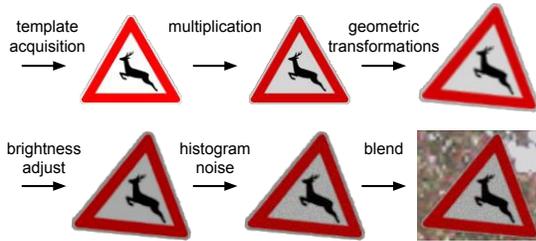


Fig. 2. Steps of the template blending process.

Let  $\mathcal{B} = \{B_i\}_{i=1}^S$  be the background set with  $S$  random natural images, in total;  $\mathcal{C} = \{C_i\}_{i=1}^M$  be the set of classes of interest;  $\mathcal{T} = \{T_i \mid i \in \mathcal{C}\}$  be the set of templates of traffic signs; and  $a, b, c, d, e,$  and  $f$  be input parameters. First, for a training data set with  $N$  samples, the training set is defined as  $\mathcal{X} = \{X_1, X_2, \dots, X_N\} \stackrel{iid}{\sim} \mathcal{B}$ . The first step is to randomly change the brightness and contrast of the image by randomly adding and multiplying each training sample, i.e.,  $\alpha_i X_i + \beta$ , where  $\alpha_i \sim U(a, b)$ ,  $\beta \sim U(c, d)$ , and  $X_i$  is the background image sampled for the  $i$ -th training sample. The next step is to add a random amount of  $|K^i|$  templates into the  $i$ -th sample  $X_i$ , where  $|K^i| \in \{1, 2, \dots, M\}$  and  $K_j^i \sim \mathcal{T}$ . The templates are placed in random configurations (e.g., a  $2 \times 4$  grid,  $1 \times 3$  row, etc) from a predefined set. This step attempts to mimic a common behavior in real world (considering the country of interest), where sometimes multiple signs are seen together. The process of adding a template into a background image is as follows: (i) multiply the template  $K_j^i$  by the same  $\alpha_i$  used on the background image  $X_i$ ; (ii) apply geometric transformations (3D rotations and scaling) (iii) adjust the brightness by adding to the template the average of the region on which the template is being added, minus a constant; (iv) add noise, i.e., jitter; (v) place the template into a random position (unless it is tied to another template) with no intersection with the others; and (vi) fade the borders of the template to create a smooth transition from the template to the background image. Lastly, a Gaussian blur  $\sigma \sim U(0, \max(e, f \times \text{scale}))$  is applied to the resulting image, generating the final training sample. Some training samples can be seen in Figure 3, while a step-by-step overview of the blending process can be seen in Figure 2.

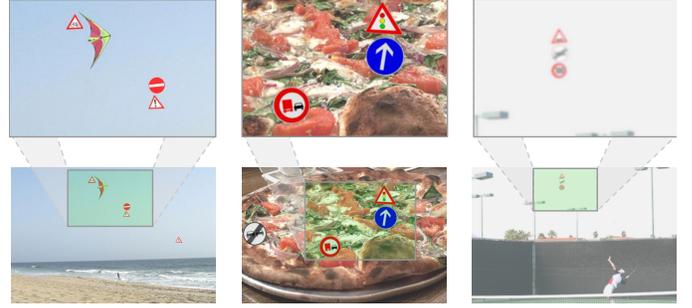


Fig. 3. Some training samples can be seen in the bottom row, and zoomed-in figures highlighting the regions with traffic signs can be seen in the top row. These samples were generated using the process described in Section III-A.

Finally, it is important to generate the templates according to the range of operation of the application. Therefore, it is important to set up the minimum and maximum size of the templates to be detected and sample the random scales accordingly. This procedure can be seen as a calibration step on which, in a real-world application, one could determine the minimum and maximum size of a traffic sign by looking at few images of a particular target camera. Models, code and the parameters used will be made available<sup>1</sup>.

## B. Model Training and Inference

After generating the training database, a deep detector can be trained. In this work, the state-of-the-art for traffic sign detection [10], Faster R-CNN [41] framework, was chosen. In every experiment conducted, the training model was initialized from a model pre-trained on ImageNet [39]. Roughly, the Faster R-CNN is a 2-step detection framework comprising (i) the Region Proposal Network (RPN), which predicts regions that are likely to contain an object, then (ii) two fully-connected networks, one to refine the predicted regions, and the other to predict the class of each object. After being trained, the Faster R-CNN can process RGB input images of traffic scenes predicting bounding boxes, classes, and confidence scores of the predicted traffic signs.

## IV. EXPERIMENTAL METHODOLOGY

This section introduces the data sets used to train and evaluate the detection models, the metrics for performance quantification, and the experiments conducted to validate our proposal. The experimental platform is described at the end of the section.

### A. Data sets

A public image data set was used as source for background images, while three data sets of traffic scenes were used to evaluate the proposed method. Each data set is described in the following paragraphs.

**Backgrounds source.** Microsoft COCO (MS-COCO) [40] is a large-scale data set (more than 200k labeled images

<sup>1</sup>The link will be available upon acceptance.

divided into training and test sets) designed for the tasks of object detection, segmentation, and visual captioning. For this work, the images of the 2017 version of MS-COCO are used as background for the traffic signs templates, as described in Section III-A. More specifically, the sign templates are overlaid onto the images of the MS-COCO training partition in order to train the traffic sign detector. For our purposes, traffic-related scenes should be disregarded, which is done by filtering out those images originally labeled as “traffic light”, “bicycle”, “car”, “motorcycle”, “bus”, “truck”, “fire hydrant”, “stop sign”, and “parking meter”. Images with height less than 600 pixels or width less than 400 pixels are also removed, totaling at the end 58078 images. The remaining images are further uniformly scaled so that the shortest dimension has 1500 pixels. Finally, the central  $1500 \times 1500$  pixels area is cropped from the scaled image.

**Evaluation datasets.** The three datasets used to evaluate the proposed method were Belgian Traffic Signs Dataset (BTSD) [42], Tsinghua-Tencent 100K benchmark (TT100K) [5] and German Traffic Sign Detection Benchmark (GTSD) [38]. On BTSD, although the data set has annotations for more than a hundred classes, it also provides a reduced list of 62, which was used in this work. On CTSD, originally, there are 151 traffic sign classes, however, only a subset of 42 classes is used. To perform this selection, the ones with less than 100 instances were simply ignored, as done in the data set’s original paper [5]. In addition, three other classes were removed. Those three categories (namely “po”, “io”, “wo”) refer to three groups of signs that are not traffic signs, thus irrelevant. On GTSD, there are 43 unbalanced traffic sign classes. Commonly, the classification step used on this data set uses only three super-classes [10], [23]: prohibitory, indicative and warning. In this work, the classification is performed using all 43 classes. In all datasets, only images containing traffic signs were used.

## B. Experiments

Five experiments were carried out to evaluate the effectiveness of the proposed approach. The first was (i) an ablation study, followed by (ii) training with a fully synthetic data set and (iii) search for model performance correspondence between training with real and synthetic images. Finally, two other applications were evaluated, using: (iv) the proposed method as data augmentation and (v) the proposed method for finetuning.

1) *Ablation Study:* The proposed approach to generate the training samples comprises a sequence of processes. To assess the importance of each process, the performance of the proposed approach was measured when disabling a process at a time. This study was conducted on the well-known GTSD data set. The study was performed on the following components from the training samples generation step: blur, brightness adjust, geometric transformations, background augmentation, blend, histogram noise and traffic sign grouping. Although other works have already shown that the Poisson blending algorithm [43] does not improve results in some problem domains [27], it has not been shown yet on the traffic sign

detection problem. Thus, the blending algorithm was tested as an alternative to the naive procedure adopted in our method. Two additional experiments were performed to verify the impact of using COCO’s traffic scenes as background to train the deep detector: training with a data set generated using only images from the driving domain in COCO and with the full COCO data set as backgrounds. Additionally, training with a data set generated using images from the target data set that contains no traffic signs as background was also evaluated. This experiment was performed only on TT100K because GTSD does not provide images without traffic signs. The last experiment evaluates (for the three test collections) the replacing of COCO backgrounds by uniform random patterns. Some samples of training instances are shown in Figure 4.

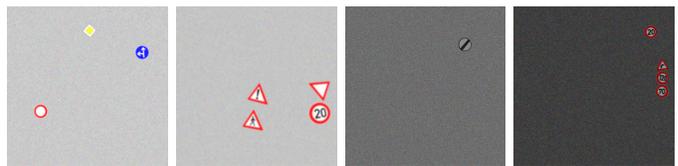


Fig. 4. Four training samples with uniform noise as background for the ablation study.

2) *Training with a Fully Synthetic Data set:* The main purpose of the proposed method is to be able to train a deep detector effortlessly, i.e., without manually annotating data, while maintaining its efficacy. For the evaluation, a deep detector was trained using only synthetic data generated by our method and compared against a model trained using only the full collection of real target-domain images (vanilla training). For fairer comparison, offline data augmentation was applied on the real image training set. This data augmentation comprises the same effects used on our method’s background augmentation: brightness, contrast and blur. The total number of images in the training set after augmentation is the same as in training set with synthetic images (70k). This experiment aims to measure the performance cost of training only with samples generated by the proposed method instead of training only with real data of the target data set. To compare with the literature, the performance of two state-of-the-art methods, with objectives similar to ours, were also evaluated. In the first method [27], objects of interest instances from the target data set’s training set are cut and then pasted on images also from the target data set that contains no objects of interest. In the second [33], a similar process is performed, but uses a neural network to predict regions that are likely to contain instances of each class. Then, it pastes instances from other images on those regions. Both methods rely on segmentation annotations, which are much more expensive to acquire than bounding box annotations. As a consequence, they are evaluated only on TT100K since this data set is the only that provides segmentation annotations. Also, in both methods, the total number of training samples is 70k (same number as in every other training in this work).

3) *Model Performance Correspondence Between Training with Real and Synthetic Images:* Another interesting question to answer is: How many real images, manually annotated,

are necessary to train a model that matches the performance of one trained with the proposed method? As an attempt to answer this question, an experiment was conducted using an approximate binary search in two data sets: BTSD and TT100K. GTSD was not evaluated because the performance with only synthetic images is already better than the vanilla training, as further discussed in Section V. Due to time constraints, the number of splits in the binary search was set to a maximum of four for each data set, each split requiring a new training section. The search is approximate since: (i) there is randomness in the training step, (ii) the number of training sessions was limited to four (due to time restrictions), and (iii) the number of images used at each step does not match exactly the number given by the binary search. The latter is because of the method used to define the set of real image sets at each search step. This set is defined as  $\mathcal{S} = \{S_i\}_{i=1}^N$ , which is a set of  $N$  sets of real images. Each set of images  $S_i$  contains the same set of images as  $S_{i-1}$  ( $S_0 = \emptyset$ ) plus  $C$  more images selected in a way to maintain the original distribution of each class, where  $C$  is the number of classes in the data set. This procedure was performed to reduce the randomness in the experiment. With the set of real images defined, data augmentation was applied to keep the number of training samples constant and equal to the number of samples used in the proposed method (i.e., 70k).

4) *Proposed Method as Data Augmentation:* One of the applications for the proposed method is data augmentation. To evaluate its effectiveness in this application, an experiment was performed on BTSD and TT100K. It was not performed on GTSD since it has not enough samples for each category. In this experiment, for each data set, Faster R-CNN was trained with a set of images  $\mathcal{I} = O \cup S \cup A$ , where  $O$  is a subset of the target country’s training set,  $S$  contains 35k synthetic images generated with the proposed method and  $A$  contains  $35k - |O|$  images generated using basic data augmentation (brightness, contrast and blur). In total,  $\mathcal{I}$  contains 70k images. The data augmented is a set of images that contains at least  $n$  original instances of each class. This approximate number is because it is very difficult to form a set with exactly  $n$  instances per class, since an image can have multiple instances of different classes. Although it is not exactly  $n$  instances, the set is chosen in a way that minimizes the difference. For each data set the experiment was performed with  $n = 1, 10, M$  (maximum number possible, i.e., using the whole training set). To reduce the randomness, each set is a super-set of the previous, using the same procedure described on Section IV-B3.

5) *Proposed Method for Finetuning:* The proposed method can also be used as a way to pretrain a model. After the pretraining, the model can be finetuned on a set of real images. In this experiment, for each data set, the model trained with a fully synthetic data set (proposed method) was finetuned on real images with simple augmentation (vanilla training). The inverse was also evaluated, i.e., training with real images with simple augmentation and then finetuning on the proposed method. In the finetuning process, the final learning rate of the pretrain ( $10^{-4}$ ) was used during the whole training session, i.e., there was no learning rate decay.

TABLE I  
ABLATION STUDY RESULTS ON GTSD

Factor	mAP (%) (normalized)
no brightness adjust	22.86 (0.2492)
no geometric transformations	46.11 (0.5026)
no background augmentation	54.82 (0.5975)
no blur	55.74 (0.6075)
no histogram noise	70.52 (0.7686)
no blend	83.02 (0.9049)
Poisson blend	86.00 (0.9373)
no traffic sign grouping	86.77 (0.9457)
domain-only COCO backgrounds	86.88 (0.9469)
no COCO filtering	87.01 (0.9483)
none (full method)	91.75 (1.0000)

### C. Faster R-CNN parameters

Every training on Faster R-CNN used the same base settings, except for the input size for each data set. The feature-extractor used was the state-of-the-art ResNet-101 [44]. For tests on BTSD and GTSD, the proposed system was trained with an input size of  $1500 \times 1500$  pixels, and tested with an input size of  $3500 \times 2060$  pixels. The vanilla training for the GTSD data set was performed with an input size of  $1360 \times 800$ , while for BTSD the input size was  $1628 \times 1236$ . For TT100K, the input size was  $2048 \times 2048$  for training with both the proposed and vanilla methods. That is, all three vanilla trainings were executed in their respective original image resolutions. The input size for testing on TT100K was  $4096 \times 4096$ . For every target data set, the input size at test time for the proposed system and the vanilla trainings were the same. The increased test sizes are due to the poor performance of the tested detectors on small objects.

### D. Performance Metrics

In addition to the precision and recall metrics, the Mean Average Precision (mAP) was also used to quantify the detection performance. The Average Precision (AP) metric, from which mAP is derived, follows the same approach in the PASCAL VOC 2012 challenge [45]. Basically, AP is defined as the approximate area under the precision/recall curve obtained for a fixed IoU threshold (0.5, in this work). Then, the mAP value is the average of APs for all object classes.

## V. RESULTS AND DISCUSSION

In the following paragraphs, quantitative and qualitative results are presented and discussed following the same order of the respective experiments introduced in the last section. The section ends with a discussion of potential applications of the proposed method.

**Ablation Study.** The results for the ablation study are shown in Table I. Overall, it can be observed that every component of the proposed method has a significant impact on the model’s performance. Disabling the brightness adjustment was the step which yielded the worst performance, achieving only 24.92% of the full method’s performance, which may be a result of the high amount of brightness variations in the data set.

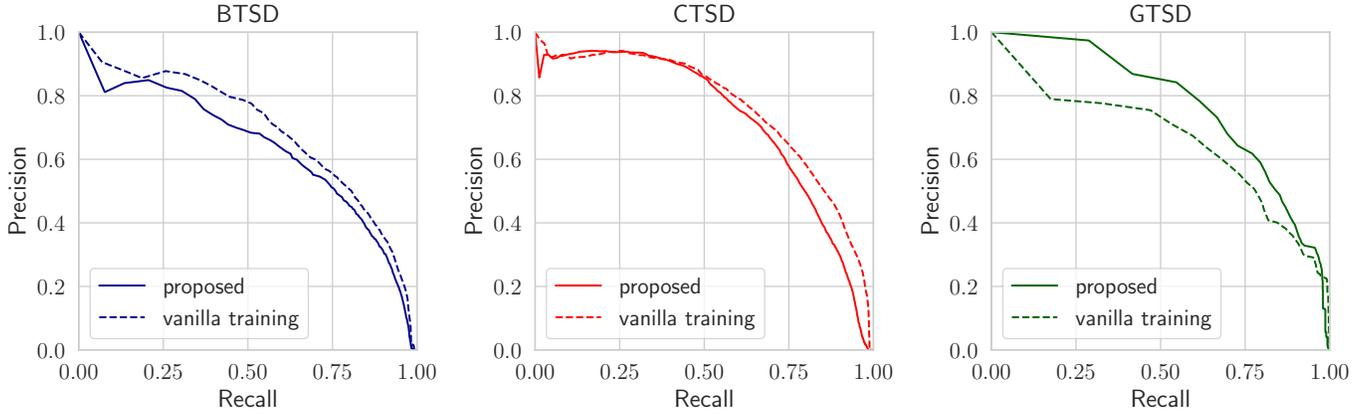


Fig. 5. Precision-Recall curves for each data set.

TABLE II

RESULTS OF TRAINING WITH UNIFORM NOISE AS BACKGROUND INSTEAD OF COCO IMAGES ON BTSD, TT100K AND GTSD.  $\Delta$  mAP CORRESPONDS TO THE DIFFERENCE WHEN COMPARED TO TRAINING WITH COCO IMAGES AS BACKGROUNDS.

Data set	mAP (%)	$\Delta$ mAP (%)
BTSD	71.61	-8.67
GTSD	76.66	-14.17
TT100K	63.99	-19.13

Concurrently, traffic sign grouping had the lowest impact. Moreover, despite the simplicity of the blending process, results show that it substantially improves (9.51 p.p. more mAP) the performance of the detector on the target data set. The results with the Poisson blending indicate that a more complex blending process may not be necessary. In fact, it can worsen the model’s performance, as the results suggest. Furthermore, the hypothesis that COCO driving domain images may hinder the learning process was also confirmed. In this scenario, the model’s performance decays by 5.17%.

The results for the study using uniform noise as background instead of real images are shown in Table II. In summary, they evidence that training with real images instead of noise indeed produces a better model, as expected. We observed, looking at the precision-recall curve, that the difference in mAP is mostly because of the increasing number of false positives. This behavior might be a result of the network being unable to learn features other than those in traffic signs, thus not learning well what a background is. On the other hand, if images that are known to not contain any traffic signs of interest are used, the performance increases by 2.12 p.p. on TT100K, but, in this case, more data were used.

**Training with a Fully Synthetic Data set.** Results of training with a fully synthetic data set generated by the proposed method are shown in Table III, while precision vs. recall curves (as computed in [45]) are shown in Figure 5. The results described as “full method” in Table I are the same ones described as “Proposed” in Table III. As evidenced by

the GTSD results in the table, training with a fully synthetic data set may be even better than training with a smaller set of real images. For GTSD, the mAP difference between the proposed method and the vanilla training is +12.25 p.p.. Additionally, for BTSD and TT100K, which are larger data sets, the proposed method still performs similarly, although it does not surpass their corresponding vanilla trainings. The difference in mAP between the proposed method and the BTSD and TT100K vanilla trainings are -5.22 p.p. and -6.16 p.p., respectively. In Table III the results of two state-of-the-art methods are also shown. Since both require real data to work, a fair comparison is against our result that uses real data, i.e., from the Proposed Method as Data Augmentation experiment. Our method outperforms both. This result shows that, for some problem domains, a more complex approach may not be necessary, in fact, it may produce a worse result.

As expected, increasing the collection of annotated traffic scenes yields a better performance, however this implies more human effort. Furthermore, it should be considered that the mAP difference between the vanilla training baseline and the proposed method is affected by the fact that the baseline’s training and test sets share a particular geographic context. This implies more similarity between the two sets with respect to the overall appearance and the traffic scene structure, which is hard to be reached in real driving applications. By using natural images, the structural dependency is disregarded completely.

For a qualitative analysis, extensive qualitative results can be seen on the videos <sup>2</sup> Some of the false positives (i.e., false alarms) are indeed signs, but either they are not traffic signs, or they do not belong to set of the classes of interest. Furthermore, there is a considerable amount of false positives related to objects from the driving domain, such as headlights or traffic lights. This may be a result of the lack of those objects in the training set, although it can be mitigated by using real traffic scenes as backgrounds (as shown in the ablation study). Most false positives with higher confidence score arise from mistakes in the classification phase, but have sufficient

<sup>2</sup>youtube.com/playlist?list=PLm8amuguiXiKEEI1A3qrktup1SbmC1a5

TABLE III  
COMPARISON WITH STATE-OF-THE-ART METHODS ON BTSD, TT100K AND GTSD. THE “PROPOSED + REAL” RESULTS ARE FROM THE PROPOSED METHOD AS DATA AUGMENTATION EXPERIMENT, USING ALL REAL IMAGES AVAILABLE.

Method	Real target domain images	mAP (%)		
		BTSD	TT100K	GTSD
Vanilla	Required	85.50	89.28	79.50
CPL [27]	Required	—	89.66	—
CPL [27] + real	Required	—	92.03	—
Context-DA [33]	Required	—	74.96	—
Proposed	Not required	80.28	83.12	<b>91.75</b>
Proposed + real	Required	<b>89.64</b>	<b>92.25</b>	—

IoU. Some false negatives (i.e., undetected traffic signs) can also be seen in the videos. The mistakes are mainly caused by severe geometric distortion or occlusion.

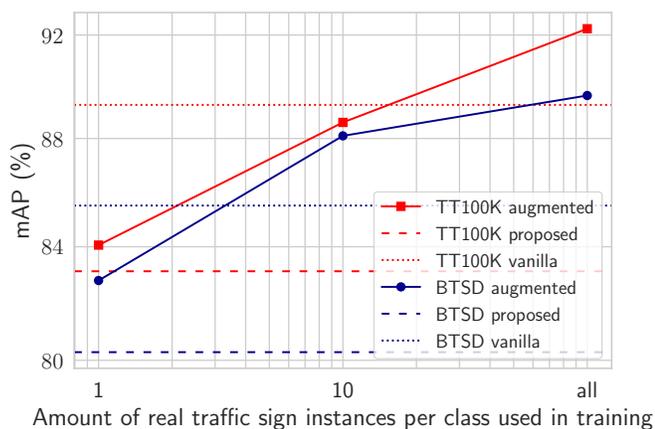


Fig. 6. Proposed method as data augmentation (logarithmic scale).

**Proposed Method as Data Augmentation.** The results for the experiments using the proposed method as a data augmentation method are shown in Figure 6. For TT100K, using  $n = 10$  instances (at least) per class (total of 231 images), resulted in a mAP of only 0.67 p.p. inferior to that obtained with the full training set that contains 5905 real images. For BTSD, training with also  $n = 10$  (total of 581 images), was enough to surpass the performance of training with the full training set that contains 4483 real images by 2.60 p.p.. Furthermore, when using the entire training set of real images ( $n = M$ ), the proposed method was able to increase the model’s performance significantly (2.97 p.p. and 4.41 p.p. of mAP, for TT100K and BTSD, respectively).

**Finetuning on Real and on Synthetic Images.** The results for finetuning on real and on synthetic images are shown in Table IV. As evidenced, using the proposed method to pretrain a model consistently increases its performance. When compared to the results with only an ImageNet pretrain, the mAP increases by 4.09 p.p., by 3.34 p.p. and by 18.98 p.p., for BTSD, TT100K and GTSD, respectively. Moreover,

TABLE IV  
FINETUNING RESULTS

Target	Training data set	Finetuning data set	mAP (%)
BTSD	proposed method	real + augmented	<b>89.89</b>
	real + augmented	proposed method	84.55
GTSD	proposed method	real + augmented	<b>98.48</b>
	real + augmented	proposed method	91.69
TT100K	proposed method	real + augmented	<b>92.62</b>
	real + augmented	proposed method	90.68

finetuning was slightly more effective than training with both data sets at the same time, as can be seen in Table III, in the “Proposed + real” row. Those results show another possible use for the proposed method, as a cheap (without acquiring new real data) way to increase a model’s performance. On the other hand, finetuning on synthetic data decreases the model’s performance. This result is on par with the general intuition, as finetuning on synthetic images may specialize the model on them.

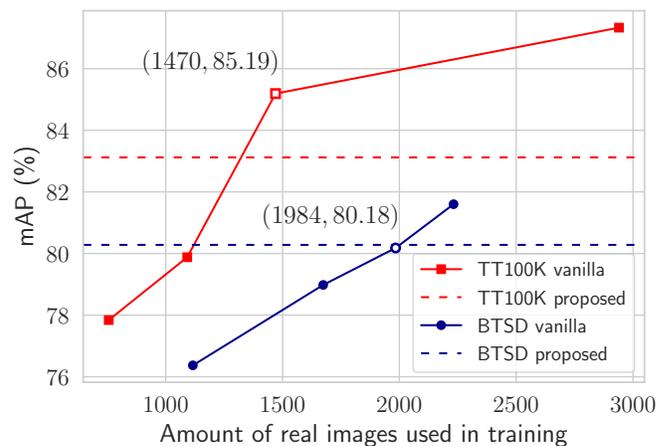


Fig. 7. Binary search for the equivalent amount of images on BTSD and TT100K. The highlighted points are the closest to the proposed method’s result.

**Model Performance Correspondence Between Training with Real and Synthetic Images.** Figure 7 shows the results of the necessary number of real images to yield a performance comparable to that achieved by synthetically generated data. It is important to recall that the number of training samples was the same in each step of the search (70k). The difference is in the number of original real images (i.e., the rest of the images are generated via simple offline data augmentation). The results reveal that, for BTSD, the equivalence number is around 2000 images. For TT100K, this number is around 1250 images. For a simple comparison, to collect and annotate TT100K, 100k images had to be manually annotated, in order to acquire 10k images with traffic signs [5]. If the ten-to-one relation is kept, our method would eliminate the need to annotate around 12.5k images on TT100K. On BTSD, it would be equivalent to annotating around 20k images manually. This suggests that the proposed method may be specially useful

when a large amount of annotated data is difficult to acquire.

## VI. CONCLUSION

Solving challenging tasks with deep neural network usually requires large-scale annotated data with real image samples belonging to the domain of the problem. The human effort and other costs involved in gathering such data has motivated research on alternative ways to train those models. In particular, this work leverages templates to train a deep model to detect traffic signs in real traffic scenes. Besides eliminating the need for real traffic signs, we also propose a more flexible and effortless construction of the training set by superposing the templates on natural images, i.e., arbitrary background images available in computer vision benchmarks. The results showed that the proposed method can be used to train deep detectors without the need for manually annotated data sets, and these models can achieve competitive performance. Moreover, multiple applications for the proposed method were shown, such as improving a model's performance when few, or even a lot of, annotated data is available. Finally, the number of real images that need to be manually annotated to match the performance of training with the proposed method was shown to be in the order of tens of thousands for every data set used. All those results indicate that the proposed method may make it viable to train a deep detector when large amounts of annotated data is difficult or impossible to acquire.

## ACKNOWLEDGMENT

The authors thank the NVIDIA Corporation for their kind donation of the GPUs used in this research. This study was financed in part by CAPES, FAPES, and CNPq.

## REFERENCES

- [1] C. Badue, R. Guidolini, R. V. Carneiro, P. Azevedo, V. B. Cardoso, A. Forechi, L. F. R. Jesus, R. F. Berriel, T. M. Paixão, F. Mutz, T. Oliveira-Santos, and A. F. De Souza, "Self-Driving Cars: A Survey," *arXiv preprint arXiv:1901.04407*, 2019.
- [2] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation," in *European Conference on Computer Vision (ECCV)*, 2018.
- [3] L. C. Possatti, R. Guidolini, V. B. Cardoso, R. F. Berriel, T. M. Paixão, C. Badue, A. F. De Souza, and T. Oliveira-Santos, "Traffic light recognition using deep learning and prior maps for autonomous cars," in *International Joint Conference on Neural Networks (IJCNN)*, 2019.
- [4] R. F. Berriel, F. S. Rossi, A. F. de Souza, and T. Oliveira-Santos, "Automatic Large-Scale Data Acquisition via Crowdsourcing for Crosswalk Classification: A Deep Learning Approach," *Computers & Graphics*, vol. 68, pp. 32–42, 2017.
- [5] Z. Zhu, D. Liang, S. Zhang, X. Huang, B. Li, and S. Hu, "Traffic-Sign Detection and Classification in the Wild," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [6] R. Sarcinelli, R. Guidolini, V. B. Cardoso, T. M. Paixão, R. F. Berriel, P. Azevedo, A. F. D. Souza, C. Badue, and T. Oliveira-Santos, "Handling pedestrians in self-driving cars using image tracking and alternative path generation with Frenét frames," *Computers & Graphics*, vol. 84, pp. 173 – 184, 2019.
- [7] R. F. Berriel, L. T. Torres, V. B. Cardoso, R. Guidolini, C. Badue, A. F. D. Souza, and T. Oliveira-Santos, "Heading direction estimation using deep learning with automatic large-scale data acquisition," in *International Joint Conference on Neural Networks (IJCNN)*, 2018.
- [8] N. Barnes, A. Zelinsky, and L. S. Fletcher, "Real-Time Speed Sign Detection Using the Radial Symmetry Detector," *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 2, pp. 322–332, 2008.
- [9] S. Maldonado-Bascon, S. Lafuente-Arroyo, P. Gil-Jimenez, H. Gomez-Moreno, and F. Lopez-Ferrerias, "Road-Sign Detection and Recognition Based on Support Vector Machines," *IEEE Transactions on Intelligent Transportation Systems*, vol. 8, no. 2, pp. 264–278, 2007.
- [10] Álvaro 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.
- [11] L. T. Torres, T. M. Paixão, R. F. Berriel, A. F. D. Souza, C. Badue, N. Sebe, and T. Oliveira-Santos, "Effortless deep training for traffic sign detection using templates and arbitrary natural images," in *International Joint Conference on Neural Networks (IJCNN)*, 2019.
- [12] A. Ruta, Y. Li, and X. Liu, "Detection, tracking and recognition of traffic signs from video input," in *2008 11th International IEEE Conference on Intelligent Transportation Systems*, 2008, pp. 55–60.
- [13] S. Varun, S. Singh, R. S. Kunte, R. S. Samuel, and B. Philip, "A road traffic signal recognition system based on template matching employing tree classifier," in *International Conference on Computational Intelligence and Multimedia Applications (ICCI)*, 2007.
- [14] 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.
- [15] F. Zaklouta and B. Stanculescu, "Real-time traffic-sign recognition using tree classifiers," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 4, pp. 1507–1514, 2012.
- [16] G. Loy and N. Barnes, "Fast shape-based road sign detection for a driver assistance system," in *International Conference on Intelligent Robots and Systems (IROS)*, 2004.
- [17] J.-P. Carrasco, A. d. l. de la Escalera, J. M. Armingol *et al.*, "Recognition stage for a speed supervisor based on road sign detection," *Sensors*, vol. 12, no. 9, pp. 12 153–12 168, 2012.
- [18] H.-K. Kim, J. H. Park, and H.-Y. Jung, "An efficient color space for deep-learning based traffic light recognition," *Journal of Advanced Transportation*, vol. 2018, 2018.
- [19] A. De La Escalera, J. M. Armingol, and M. Salichs, "Traffic sign detection for driver support systems," in *International Conference on Field and Service Robotics*, 2001.
- [20] C.-C. Lin and M.-S. Wang, "Road sign recognition with fuzzy adaptive pre-processing models," *Sensors*, vol. 12, no. 5, pp. 6415–6433, 2012.
- [21] C.-Y. Fang, S.-W. Chen, and C.-S. Fuh, "Road-sign detection and tracking," *IEEE transactions on vehicular technology*, vol. 52, no. 5, pp. 1329–1341, 2003.
- [22] D. C. Cireşan, U. Meier, J. Masci, and J. Schmidhuber, "A committee of neural networks for traffic sign classification," in *IJCNN*, 2011, pp. 1918–1921.
- [23] Y. Wu, Y. Liu, J. Li, H. Liu, and X. Hu, "Traffic sign detection based on convolutional neural networks," in *International Joint Conference on Neural Networks (IJCNN)*, 2013.
- [24] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition," *Neural Networks*, 2012.
- [25] D. Cireşan, U. Meier, J. Masci, and J. Schmidhuber, "Multi-column deep neural network for traffic sign classification," *Neural Networks*, vol. 32, pp. 333–338, Aug. 2012.
- [26] S. B. Wali, M. A. Abdullah, M. A. Hannan, A. Hussain, S. A. Samad, P. J. Ker, and M. B. Mansor, "Vision-Based Traffic Sign Detection and Recognition Systems: Current Trends and Challenges," *Sensors*, vol. 19, no. 9, p. 2093, Jan. 2019.
- [27] D. Dwibedi, I. Misra, and M. Hebert, "Cut, paste and learn: Surprisingly easy synthesis for instance detection," in *International Conference on Computer Vision (ICCV)*, 2017.
- [28] H. Wang, Q. Wang, F. Yang, W. Zhang, and W. Zuo, "Data Augmentation for Object Detection via Progressive and Selective Instance-Switching," *arXiv:1906.00358 [cs]*, Jun. 2019, arXiv: 1906.00358.
- [29] X. Peng, B. Sun, K. Ali, and K. Saenko, "Learning Deep Object Detectors from 3d Models," in *International Conference on Computer Vision (ICCV)*, 2015.
- [30] B. Moiseev, A. Konev, A. Chigorin, and A. Konushin, "Evaluation of Traffic Sign Recognition Methods Trained on Synthetically Generated Data," in *Advanced Concepts for Intelligent Vision Systems*, ser. Lecture Notes in Computer Science. Springer International Publishing, 2013, pp. 576–583.
- [31] A. Stergiou, G. Kalliatakis, and C. Chrysoulas, "Traffic Sign Recognition based on Synthesised Training Data," *Big Data and Cognitive Computing*, 2018.
- [32] A. Møgelmoose, M. M. Trivedi, and T. B. Moeslund, "Learning to detect traffic signs: Comparative evaluation of synthetic and real-world

datasets,” in *International Conference on Pattern Recognition (ICPR)*, 2012.

- [33] N. Dvornik, J. Mairal, and C. Schmid, “Modeling Visual Context Is Key to Augmenting Object Detection Datasets,” in *European Conference on Computer Vision (ECCV)*, 2018.
- [34] G. Georgakis, A. Mousavian, A. Berg, and J. Kosecka, “Synthesizing Training Data for Object Detection in Indoor Scenes,” in *Robotics: Science and Systems XIII*. Robotics: Science and Systems Foundation, Jul. 2017.
- [35] A. Gupta, A. Vedaldi, and A. Zisserman, “Synthetic Data for Text Localisation in Natural Images,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [36] S. M. Grigorescu, “Generative One-Shot Learning (GOL): A Semi-Parametric Approach to One-Shot Learning in Autonomous Vision,” in *International Conference on Robotics and Automation (ICRA)*, 2018.
- [37] J. Kim, T.-H. Oh, S. Lee, F. Pan, and I. S. Kweon, “Variational prototyping-encoder: One-shot learning with prototypical images,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [38] 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 *International Joint Conference on Neural Networks (IJCNN)*, 2013.
- [39] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2012.
- [40] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft COCO: Common Objects in Context,” in *European Conference on Computer Vision (ECCV)*, 2014.
- [41] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017.
- [42] 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, Apr. 2014.
- [43] P. Pérez, M. Gangnet, and A. Blake, “Poisson image editing,” *ACM Trans. Graph.*, vol. 22, no. 3, pp. 313–318, 2003.
- [44] 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.
- [45] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, “The PASCAL Visual Object Classes Challenge: A Retrospective,” *International Journal of Computer Vision (IJCV)*, vol. 111, no. 1, pp. 98–136, 2015.



**Lucas Tabelini** is currently pursuing a B.S. degree in Computer Engineering at Universidade Federal do Espírito Santo (UFES), Brazil. Current research areas of interest include deep learning and computer vision.



**Rodrigo Berriel** received the M.Sc. degree in computer science in 2016 from the Universidade Federal do Espírito Santo (UFES, Brazil) and is currently pursuing his Ph.D. degree in the same university. His current research interests include computer vision and deep learning, particularly few-shot and lifelong learning.



**Thiago M. Paixão** received the B.S. degree in computer science in 2007 from Universidade Federal de Minas Gerais (UFMG, Brazil), M.S. degree in computer science in 2010 from Universidade de São Paulo (USP, Brazil). He is currently pursuing the Ph.D. degree in computer science from Universidade Federal do Espírito Santo (UFES, Brazil). He is Professor at the Instituto Federal do Espírito Santo (IFES). His research interests include deep learning, computer vision, and structural pattern recognition.



**Alberto F. De Souza** is a Full Professor of Computer Science and Coordinator of the Laboratório de Computação de Alto Desempenho (LCAD High Performance Computing Laboratory) at the Universidade Federal do Espírito Santo (UFES), Brazil. He received B. Eng. (Cum Laude) in electronics engineering and M. Sc. in systems engineering and computer science from Universidade Federal do Rio de Janeiro (COPPE/UFRJ), Brazil, in 1988 and 1993, respectively; and Doctor of Philosophy (Ph.D.) in computer science from the University College London, United Kingdom in 1999. Alberto F. De Souza is Senior Member of the IEEE and Comendador of the Rubem Braga Order.



neural networks, and cognitive science.

**Claudine Badue** is an Associate Professor of Computer Science and Co-Coordinator of the Laboratório de Computação de Alto Desempenho (LCAD High Performance Computing Laboratory) at the Universidade Federal do Espírito Santo (UFES), Brazil. She received a Bachelor’s degree in Computer Science in 1998 from the Universidade Federal de Goiás (UFG), a Ms.C. in Computer Science in 2001 and a Ph.D. in Computer Science in 2007, both from the Universidade Federal de Minas Gerais (UFMG). Her current research interests include autonomous cars,



ACM Multimedia 2022. He is a fellow of the International Association for Pattern Recognition.

**Nicu Sebe** is Professor with the University of Trento, Italy, leading the research in the areas of multimedia information retrieval and human behavior understanding. He was the General Co-Chair of the IEEE FG Conference 2008 and ACM Multimedia 2013, and the Program Chair of the International Conference on Image and Video Retrieval in 2007 and 2010, ACM Multimedia 2007 and 2011. He was the Program Chair of ECCV 2016 and ICCV 2017, and a General Chair of ACM ICMR 2017. He is a program chair of ICPR 2020 and a general chair of



**Thiago Oliveira-Santos** received the B.S. degree in Computer Engineering in 2004 and the M.Sc. degree in Informatics in 2006 from Universidade Federal do Espírito Santo (UFES, Brazil). He received the Ph.D. degree in Biomedical Engineering in 2011 from Universität Bern (UNIBE, Switzerland). He is currently Professor at the UFES. Areas of interest include computer vision, image processing, computer graphics and image-guided surgery, cognition science, robotics.